



Developer Guide

# Amazon SageMaker AI



# Amazon SageMaker AI: Developer Guide

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

---

# Table of Contents

<b>What is Amazon SageMaker AI? .....</b>	<b>1</b>
Amazon SageMaker AI rename .....	1
Legacy namespaces remain the same .....	1
Amazon SageMaker and Amazon SageMaker AI .....	2
Pricing for Amazon SageMaker AI .....	2
Recommendations for a first-time user of Amazon SageMaker AI .....	3
Overview of machine learning with Amazon SageMaker AI .....	3
SageMaker AI Features .....	6
New features .....	6
Machine learning environments .....	7
Major features .....	9
<b>Setting up SageMaker AI .....</b>	<b>13</b>
Complete Amazon SageMaker AI prerequisites .....	14
Sign up for an AWS account .....	14
Create a user with administrative access .....	15
(Optional) Configure the AWS CLI .....	17
Use quick setup .....	18
Quick setup .....	18
After quick setup .....	20
Use custom setup .....	20
Authentication methods .....	20
Custom setup .....	22
Access the domain after onboarding .....	29
Domain overview .....	29
SageMaker AI domain entities .....	30
Choose an Amazon VPC .....	91
Supported Regions and Quotas .....	93
Quotas .....	93
<b>Automated ML, no-code, or low-code .....</b>	<b>94</b>
SageMaker Autopilot .....	95
Create Regression or Classification Jobs Using the AutoML API .....	99
Create an Image Classification Job using the AutoML API .....	184
Create a Text Classification job using the AutoML API .....	195
Create a time-series forecasting job using the AutoML API .....	206

Create an LLM fine-tuning job using the AutoML API .....	251
Create a Regression or Classification Job Using the Studio Classic UI .....	276
Example Notebooks .....	288
Videos .....	293
Tutorials .....	294
Quotas .....	294
API reference .....	296
SageMaker JumpStart .....	299
Open and use JumpStart in Studio .....	299
Open and use JumpStart in Studio Classic .....	302
Foundation models .....	305
Access control .....	360
Studio Classic .....	371
<b>Machine learning environments offered by Amazon SageMaker AI .....</b>	<b>416</b>
<b>Studio .....</b>	<b>418</b>
Migration from Amazon SageMaker Studio Classic .....	420
Launch Amazon SageMaker Studio .....	467
Amazon SageMaker Studio UI overview .....	469
Amazon EFS auto-mounting in Studio .....	473
Idle shutdown .....	478
Applications supported in Amazon SageMaker Studio .....	486
Lifecycle configurations .....	487
Amazon SageMaker Studio spaces .....	493
Perform common UI tasks .....	508
NVMe stores with Amazon SageMaker Studio .....	509
Local mode support in Amazon SageMaker Studio .....	511
View your instances, applications, and spaces .....	520
Stop and delete your Studio running applications and spaces .....	521
SageMaker Studio image support policy .....	529
Amazon SageMaker Studio pricing .....	537
Troubleshooting .....	537
<b>Studio Classic .....</b>	<b>541</b>
Studio Classic maintenance phase plan .....	542
Studio Classic Features .....	543
UI Overview .....	543
Launch Amazon SageMaker Studio Classic .....	551

JupyterLab Versioning .....	555
Use the Studio Classic Launcher .....	564
Use Studio Classic Notebooks .....	569
Customize Studio Classic .....	655
Perform Common Tasks .....	708
Studio Classic Pricing .....	722
Troubleshooting .....	723
SageMaker JupyterLab .....	729
JupyterLab user guide .....	730
JupyterLab administrator guide .....	741
SageMaker Notebook Instances .....	769
Maintenance .....	770
Machine Learning with the SageMaker Python SDK .....	770
Tutorial for building models with Notebook Instances .....	771
AL2 instances .....	797
JupyterLab versioning .....	801
Create an Amazon SageMaker notebook instance .....	806
Access Notebook Instances .....	811
Update a Notebook Instance .....	813
Customize a Notebook Instance using an LCC .....	813
Access example notebooks .....	826
Set the Notebook Kernel .....	829
Git Repos .....	829
Notebook Instance Metadata .....	841
Monitor Jupyter Logs in Amazon CloudWatch Logs .....	842
SageMaker Studio Lab .....	842
Studio Lab components overview .....	844
Onboard to Studio Lab .....	849
Manage your account .....	850
Launch Studio Lab .....	852
Use Studio Lab starter assets .....	854
Studio Lab pre-installed environments .....	857
Use the Studio Lab project runtime .....	858
Troubleshooting .....	882
SageMaker Canvas .....	885
Are you a first-time SageMaker Canvas user? .....	887

Getting started .....	887
Tutorial: Build a machine learning workflow in Canvas .....	895
Amazon SageMaker Canvas setup and permissions management (for IT administrators) ...	904
Generative AI assistance using Q Developer .....	964
Data import .....	976
Data preparation .....	1015
Generative AI foundation models .....	1117
Ready-to-use models .....	1144
Custom models .....	1154
Logging out .....	1283
Limitations and troubleshooting .....	1284
Billing and cost in SageMaker Canvas .....	1287
SageMaker geospatial capabilities .....	1289
How can I use SageMaker geospatial capabilities? .....	1290
First-time user? .....	1291
Getting started .....	1292
Geospatial processing job .....	1308
Earth Observation Jobs .....	1324
Vector Enrichment Jobs .....	1332
Visualization Using SageMaker geospatial capabilities .....	1333
Amazon SageMaker geospatial Map SDK .....	1337
SageMaker geospatial capabilities FAQ .....	1345
Security and Permissions .....	1346
Types of compute instances .....	1359
Data collections .....	1362
RStudio on Amazon SageMaker AI .....	1367
Region availability .....	1367
RStudio components .....	1368
Differences from Posit Workbench .....	1369
RStudio on SageMaker AI management .....	1369
RStudio on Amazon SageMaker AI user guide .....	1422
SageMaker Code Editor .....	1427
Using the Code Editor .....	1429
Code Editor administrator guide .....	1441
SageMaker HyperPod .....	1459
AWS Regions supported by SageMaker HyperPod .....	1460

Prerequisites .....	1461
IAM for HyperPod .....	1467
SageMaker HyperPod recipes .....	1477
Orchestrating HyperPod clusters with Slurm .....	1526
Orchestrating HyperPod clusters with Amazon EKS .....	1612
HyperPod in Studio .....	1710
References .....	1723
HyperPod release notes .....	1729
HyperPod AMI releases .....	1739
<b>Generative AI in SageMaker notebook environments .....</b>	<b>1754</b>
Installation .....	1755
Access features .....	1756
Model configuration .....	1758
Use Jupyter AI .....	1764
<b>Amazon Q Developer .....</b>	<b>1770</b>
Set up Amazon Q Developer for your users .....	1770
Use Amazon Q to Expedite Your Machine Learning Workflows .....	1775
<b>Amazon SageMaker Partner AI Apps overview .....</b>	<b>1775</b>
How it works .....	1775
Integration with AWS services .....	1776
Supported types .....	1776
Set up Partner AI Apps .....	1778
Partner AI App provisioning .....	1789
Set up the Amazon SageMaker Partner AI Apps SDKs .....	1790
Partner AI Apps in Studio .....	1795
<b>Data labeling with a human-in-the-loop .....</b>	<b>1797</b>
Ground Truth .....	1797
Are You a First-time User of Ground Truth? .....	1798
Getting started: Create a labeling job .....	1799
Label Images .....	1806
Label Text .....	1831
Videos and video frame labeling .....	1845
Label 3D Point Clouds .....	1883
Label verification and adjustment .....	1946
Custom workflows .....	1957
Create a Labeling Job .....	2009

Use input and output data .....	2059
Enhanced data labeling .....	2167
Security and Permissions .....	2183
Monitor Labeling Job Status .....	2222
Ground Truth Plus .....	2226
Getting Started with Amazon SageMaker Ground Truth Plus. ....	2228
Request a Project .....	2230
Create a Project Team .....	2232
Project Portal .....	2235
Create a Batch .....	2236
Batch Metrics .....	2238
Batch Details .....	2240
Accept or Reject Batches .....	2243
Workforces .....	2243
Using the Amazon Mechanical Turk Workforce .....	2244
Subscribe to vendor workforces .....	2249
Private workforce .....	2251
Crowd HTML Elements Reference .....	2283
SageMaker AI Crowd HTML Elements .....	2284
Augmented AI Crowd HTML Elements .....	2386
Augmented AI .....	2395
Get Started with Amazon Augmented AI .....	2397
Use Cases and Examples .....	2429
Create a Human Review Workflow .....	2440
Delete a Human Review Workflow .....	2467
Create and Start a Human Loop .....	2470
Delete a Human Loop .....	2477
Create and Manage Worker Task Templates .....	2481
Monitor and Manage Your Human Loop .....	2496
Output Data .....	2498
Permissions and Security .....	2512
CloudWatch Events .....	2520
API References .....	2523
<b>Prepare data .....</b>	<b>2525</b>
Choose a feature .....	2525
Use cases .....	2525

Recommended features .....	2525
Additional options .....	2529
Data preparation with SQL in Studio .....	2530
Quickstart: Query data in Amazon S3 .....	2534
Features overview and usage .....	2541
Configure network access (for administrators) .....	2550
Data source connections .....	2553
FAQs .....	2578
Connection parameters .....	2579
Data preparation at scale using Amazon EMR .....	2594
Configure network access .....	2596
Prepare data using EMR Serverless .....	2601
Data preparation using Amazon EMR .....	2626
Data preparation using AWS Glue interactive sessions .....	2681
Get started with AWS Glue interactive sessions .....	2682
AWS Glue interactive session pricing .....	2689
Prepare Data with Data Wrangler .....	2690
Get Started with Data Wrangler .....	2693
Import .....	2706
Create and Use a Data Wrangler Flow .....	2781
Get Insights On Data and Data Quality .....	2790
Automatically Train Models on Your Data Flow .....	2803
Transform Data .....	2804
Analyze and Visualize .....	2865
Reusing Data Flows for Different Datasets .....	2877
Export .....	2888
Use Data Preparation in a Studio Classic Notebook to Get Data Insights .....	2923
Security and Permissions .....	2929
Release Notes .....	2945
Troubleshoot .....	2951
Increase Amazon EC2 Instance Limit .....	2961
Update Data Wrangler .....	2962
Shut Down Data Wrangler .....	2964
<b>Processing jobs .....</b>	<b>2966</b>
Sample Notebooks .....	2967
CloudWatch Logs and Metrics .....	2968

Run a Processing Job with Apache Spark .....	2968
Run a Processing Job with scikit-learn .....	2969
Data Processing with Framework Processors .....	2970
Hugging Face Framework Processor .....	2971
MXNet Framework Processor .....	2973
PyTorch Framework Processor .....	2974
TensorFlow Framework Processor .....	2975
XGBoost Framework Processor .....	2977
Use Your Own Processing Code .....	2978
Run Scripts with a Processing Container .....	2979
How to Build Your Own Processing Container .....	2981
<b>Create, store, and share features .....</b>	<b>2988</b>
How Feature Store works .....	2989
Create feature groups .....	2990
Find, discover, and share features .....	2990
Real-time inference for features stored in the online store .....	2990
Offline store for model training and batch inference .....	2990
Feature data ingestion .....	2991
Resilience in Feature Store .....	2991
Get started with Amazon SageMaker Feature Store .....	2991
Feature Store concepts .....	2992
Adding policies to your IAM role .....	2999
Use Feature Store with SDK for Python (Boto3) .....	2999
Using Amazon SageMaker Feature Store in the console .....	3017
Delete a feature group .....	3017
Data sources and ingestion .....	3032
Stream ingestion .....	3032
Data Wrangler with Feature Store .....	3033
Feature Store Spark .....	3034
Feature Processing .....	3044
Feature Store Feature Processor SDK .....	3045
Running Feature Store Feature Processor remotely .....	3048
Creating and running Feature Store Feature Processor pipelines .....	3049
Scheduled and event based executions for Feature Processor pipelines .....	3051
Monitor Amazon SageMaker Feature Store Feature Processor pipelines .....	3053
IAM permissions and execution roles .....	3054

Feature Processor restrictions, limits, and quotas .....	3055
Data sources .....	3056
Example Feature Processing code for common use cases .....	3071
Time to live (TTL) duration for records .....	3075
Cross account feature group discoverability and access .....	3077
Enabling cross account discoverability .....	3079
Enabling cross account access .....	3084
Feature Store storage configurations .....	3095
Online store .....	3095
Offline store .....	3097
Throughput modes .....	3098
Collection types .....	3102
Add features and records to a feature group .....	3103
API .....	3103
Example code .....	3104
Find features in your feature groups .....	3106
How to search for your features .....	3107
Find feature groups in your Feature Store .....	3111
How to find feature groups .....	3113
Adding searchable metadata to your features .....	3120
How to add searchable metadata to your features .....	3120
Create a dataset from your feature groups .....	3128
Using the Amazon SageMaker Python SDK to get your data from your feature groups ....	3129
Sample Amazon Athena queries .....	3134
Delete records from your feature groups .....	3135
Delete records from the online store .....	3136
Delete records from the offline store .....	3138
Logging Feature Store operations by using AWS CloudTrail .....	3140
Management events .....	3141
Data events .....	3141
Security and access control .....	3143
Using AWS KMS permissions for Amazon SageMaker Feature Store .....	3143
Authorizing use of a customer managed Key for your online store .....	3144
Using grants to authorize Feature Store .....	3146
Monitoring Feature Store interaction with AWS KMS .....	3147
Accessing data in your online store .....	3147

Authorizing use of a customer managed key for your offline store .....	3147
Quotas, naming rules and data types .....	3148
Quota terminologies .....	3148
Limits and quotas .....	3148
Naming rules .....	3149
Data types .....	3149
Amazon SageMaker Feature Store offline store data format .....	3150
Amazon SageMaker Feature Store offline store URI structures .....	3150
Amazon SageMaker Feature Store resources .....	3152
Feature Store example notebooks and workshops .....	3152
Feature Store Python SDK and API .....	3153
<b>Reserve capacity with SageMaker training plans .....</b>	<b>3154</b>
What is SageMaker training plans .....	3154
Benefits .....	3155
Reservation .....	3155
User workflow .....	3156
Supported instance types, AWS Regions, and pricing .....	3158
Search behavior .....	3159
Considerations .....	3160
IAM for SageMaker training plans .....	3161
Managed policies .....	3162
Individual permissions .....	3162
Training plans creation .....	3166
Create a training plan using the console UI .....	3167
Create a training plan programmatically .....	3174
Training plans utilization for SageMaker training jobs .....	3183
Checkpoint your training job .....	3184
Create a training job using the console UI .....	3186
Create a training job programmatically .....	3188
Training plans utilization for SageMaker HyperPod clusters .....	3191
Create an HyperPod cluster on a training plan using the console UI .....	3192
Update an HyperPod cluster on a training plan using the console UI .....	3193
Create an HyperPod cluster on a training plan programmatically .....	3194
Update an HyperPod cluster on a training plan programmatically .....	3195
Quotas and pricing .....	3196
Release notes .....	3199

December 04, 2024 .....	3199
<b>Model training .....</b>	<b>3200</b>
The basic architecture of SageMaker Training .....	3200
Full view of the SageMaker Training workflow and features .....	3201
Before training .....	3203
During training .....	3205
After training .....	3207
Model Training .....	3209
Choosing a feature within Amazon SageMaker Training .....	3209
Additional options .....	3211
Types of Algorithms .....	3212
Choose an algorithm implementation .....	3213
Problem types for the basic machine learning paradigms .....	3216
Built-in algorithms and pretrained models .....	3219
Use Reinforcement Learning .....	3662
Run local code as a remote job .....	3671
Set up your environment .....	3672
Invoke a remote function .....	3680
Configuration file .....	3691
Customize your runtime environment .....	3693
Container image compatibility .....	3694
Logging parameters and metrics with Amazon SageMaker Experiments .....	3700
Using modular code with the @remote decorator .....	3704
Private repository for runtime dependencies .....	3707
Example notebooks .....	3709
Experiments with MLflow .....	3710
MLflow integrations .....	3710
Supported AWS Regions .....	3711
How it works .....	3711
Tracking servers .....	3716
Launch MLflow UI .....	3729
Integrate MLflow with your environment .....	3732
Tutorials .....	3743
Troubleshooting .....	3744
Cleanup .....	3745
Studio Classic .....	3748

Automatic Model Tuning .....	3752
Hyperparameter tuning strategies .....	3754
Define metrics and environment variables .....	3757
Define Hyperparameter Ranges .....	3760
Track and set completion criteria .....	3766
Tune Multiple Algorithms .....	3770
Example: Hyperparameter Tuning Job .....	3782
Stop Training Jobs Early .....	3799
Run a Warm Start Hyperparameter Tuning Job .....	3801
Resource Limits for Automatic Model Tuning .....	3807
Best Practices for Hyperparameter Tuning .....	3810
Data refining during training .....	3813
How SageMaker smart sifting works .....	3814
Supported frameworks and AWS Regions .....	3816
SageMaker smart sifting within your training script .....	3817
Troubleshooting .....	3828
Security in SageMaker smart sifting .....	3828
SageMaker smart sifting Python SDK reference .....	3829
Release notes .....	3832
Debugging and improving model performance .....	3833
TensorBoard in SageMaker AI .....	3834
SageMaker Debugger .....	3852
Access a training container through SSM for remote debugging .....	4024
Release notes .....	4033
Profile and optimize computational performance .....	4035
SageMaker Profiler .....	4037
Monitor AWS compute resource utilization in SageMaker Studio Classic .....	4061
Release notes .....	4143
Distributed training .....	4144
Distributed training concepts .....	4145
Get started with distributed training in Amazon SageMaker AI .....	4148
Strategies for distributed training .....	4154
Distributed training optimization .....	4156
Scaling training .....	4157
SageMaker AI distributed data parallelism library .....	4160
SageMaker model parallelism library v2 .....	4226

Distributed computing with SageMaker AI best practices .....	4431
Training Compiler .....	4436
What Is SageMaker Training Compiler? .....	4436
How It Works .....	4437
Supported Frameworks, AWS Regions, Instance Types, and Tested Models .....	4439
Bring Your Own Deep Learning Model .....	4473
Enable Training Compiler .....	4485
Example Notebooks and Blogs .....	4506
Best Practices and Considerations .....	4507
Training Compiler FAQ .....	4511
Troubleshooting .....	4513
Release Notes .....	4520
Setting up training jobs to access datasets .....	4526
SageMaker AI input modes and AWS cloud storage options .....	4527
Configure data input mode using the SageMaker Python SDK .....	4529
Configure data input channel to use Amazon FSx for Lustre .....	4531
Choosing an input mode and a storage unit .....	4535
Use attribute-based access control (ABAC) for multi-tenancy training .....	4538
Mapping of training storage paths .....	4542
Overview of how SageMaker AI maps storage paths .....	4543
Uncompressed model output .....	4544
Managing storage paths for different types of instance local storage .....	4545
SageMaker AI environment variables and the default paths for training storage locations .....	4546
Heterogeneous clusters .....	4549
Configure a training job with a heterogeneous cluster in Amazon SageMaker AI .....	4550
Run distributed training on a heterogeneous cluster in Amazon SageMaker AI .....	4554
Modify your training script to assign instance groups .....	4558
Use Incremental Training .....	4560
Perform Incremental Training (Console) .....	4561
Perform Incremental Training (API) .....	4564
Managed Spot Training .....	4567
Managed Spot Training Lifecycle .....	4569
Managed Warm Pools .....	4569
How it works .....	4570
Considerations .....	4575

Request a warm pool quota increase .....	4575
Use SageMaker AI managed warm pools .....	4576
CloudWatch Metrics for Training Jobs .....	4582
Define Training Metrics .....	4583
View training job metrics .....	4586
Example: Viewing a Training and Validation Curve .....	4589
Augmented Manifest Files .....	4590
Augmented Manifest File format .....	4591
Augmented Manifest File Format for Pipe Mode Training .....	4592
Use an Augmented Manifest File .....	4593
Checkpoints in SageMaker AI .....	4596
Frameworks and algorithms .....	4597
Considerations for checkpointing .....	4598
Enable checkpointing .....	4599
Browse checkpoint files .....	4601
Resume training from a checkpoint .....	4602
Cluster repairs for GPU errors .....	4602
<b>Deploy models for inference .....</b>	<b>4604</b>
Choosing a feature .....	4604
Use cases .....	4604
Recommended features .....	4605
Additional options .....	4606
Model Deployment .....	4607
Options for deploying models and getting inferences .....	4607
Before you begin .....	4608
Steps for model deployment .....	4608
Inference options .....	4609
Advanced endpoint options .....	4611
Next steps .....	4611
Model creation with ModelBuilder .....	4613
Build your model with ModelBuilder .....	4614
Define serialization and deserialization methods .....	4615
Customize model loading and handling of requests .....	4618
Build your model and deploy .....	4619
Bring your own container (BYOC) .....	4620
Using ModelBuilder in local mode .....	4621

ModelBuilder examples .....	4623
Inference optimization .....	4623
Optimization techniques .....	4624
Deploy a pre-optimized model .....	4626
Create an optimization job .....	4631
View the optimization job results .....	4645
Evaluate performance .....	4645
Supported models reference .....	4649
Options for evaluating your model .....	4657
Inference Recommender .....	4659
How it Works .....	4659
How to Get Started .....	4659
Example notebooks .....	4660
Prerequisites .....	4660
Recommendation jobs .....	4672
Real-time inference .....	4733
Deploy models .....	4734
Invoke models .....	4761
Endpoints .....	4768
Hosting options .....	4776
Automatic scaling .....	4856
Instance storage volumes .....	4885
Validation of models in production .....	4886
Online explainability .....	4899
Fine-tune with adapters .....	4925
Serverless Inference .....	4928
How it works .....	4929
Getting started .....	4932
Serverless endpoint operations .....	4933
Alarms and logs .....	4950
Automatically scale Provisioned Concurrency for a serverless endpoint .....	4952
Troubleshooting .....	4966
Asynchronous inference .....	4967
How It Works .....	4967
How Do I Get Started? .....	4968
Asynchronous endpoint operations .....	4968

Alarms and logs .....	4981
Check prediction results .....	4986
Autoscale an asynchronous endpoint .....	4989
Troubleshooting .....	4993
Batch transform .....	5001
Use batch transform to get inferences from large datasets .....	5002
Speed up a batch transform job .....	5004
Use batch transform to test production variants .....	5004
Sample Notebooks .....	5004
Associate Prediction Results with Input .....	5004
Storage in Batch Transform .....	5012
Troubleshooting .....	5013
Model parallelism and large model inference .....	5014
The LMI container documentation .....	5015
SageMaker AI endpoint parameters for LMI .....	5015
Deploying uncompressed models .....	5017
Deploy large models for inference with TorchServe .....	5018
Deployment guardrails .....	5028
How to get started .....	5029
Auto-Rollback Configuration and Monitoring .....	5030
Blue/Green Deployments .....	5034
Use rolling deployments .....	5049
Exclusions .....	5054
Shadow tests .....	5055
Create a shadow test .....	5056
How to view, monitor, and edit shadow tests .....	5061
Complete a shadow test .....	5068
Best practices .....	5071
Access containers through SSM .....	5071
Allowlist .....	5072
Enable SSM access .....	5072
IAM configuration .....	5073
SSM access with AWS PrivateLink .....	5074
Logging with Amazon CloudWatch Logs .....	5074
Accessing model containers .....	5075
Model servers .....	5076

Deploy models with TorchServe .....	5076
Deploy models with DJL Serving .....	5083
Model deployment with Triton Inference Server .....	5089
Model deployment at the edge .....	5098
Why Use Edge Manager? .....	5098
How Does it Work? .....	5098
How Do I Use SageMaker Edge Manager? .....	5099
First Steps .....	5099
Setup for Devices and Fleets .....	5122
How to Package Model .....	5130
The Edge Manager Agent .....	5137
Manage Model .....	5158
SageMaker Edge Manager end of life .....	5170
Model optimization with Neo .....	5172
What is SageMaker Neo? .....	5172
How it Works .....	5173
Compile Models .....	5173
Cloud Instances .....	5195
Edge Devices .....	5234
Troubleshoot Errors .....	5267
Stateful sessions .....	5277
How stateful sessions work .....	5278
Example implementation .....	5281
Best practices .....	5281
Best practices for deploying models on SageMaker AI Hosting Services .....	5281
Monitor Security Best Practices .....	5283
Low latency real-time inference with AWS PrivateLink .....	5283
Migrate inference workload from x86 to AWS Graviton .....	5285
Troubleshoot deployments .....	5289
Inference cost optimization best practices .....	5291
Best practices to minimize interruptions during GPU driver upgrades .....	5294
Best practices for endpoint security .....	5298
Updating containers for the NVIDIA Container Toolkit .....	5300
Supported features .....	5304
Resources .....	5310
Blogs, example notebooks, and additional resources .....	5311

Troubleshooting and reference .....	5314
Model Hosting FAQs .....	5315
<b>Implement MLOps .....</b>	<b>5325</b>
Why MLOps? .....	5325
Challenges with MLOps .....	5326
Benefits of MLOps .....	5327
Experiments .....	5328
Workflows .....	5328
ML Pipelines .....	5329
Kubernetes Orchestration .....	5489
Notebook Jobs .....	5585
Schedule your ML workflows .....	5650
ML Lineage Tracking .....	5653
Tracking Entities .....	5654
SageMaker AI-Created Entities .....	5657
Manually Create Entities .....	5659
Querying Lineage Entities .....	5664
Tracking Cross-Account Lineage .....	5673
Model Registry .....	5677
Models, Model Versions, and Model Groups .....	5678
Collections .....	5759
Model Deployment .....	5772
Model Monitor .....	5772
Projects .....	5773
SageMaker Projects .....	5773
Granting SageMaker Studio Permissions Required to Use Projects .....	5777
Create a MLOps Project .....	5779
Templates .....	5781
View Resources .....	5793
Update a MLOps Project .....	5794
Delete a MLOps Project .....	5797
Walk Through a Project Using Third-party Git Repos .....	5798
MLOps troubleshooting .....	5804
<b>Data and model quality monitoring .....</b>	<b>5806</b>
Model Monitoring .....	5807
How It Works .....	5807

Sample Notebooks .....	5810
Data capture .....	5811
Capture data from real-time endpoint .....	5811
Capture data from batch transform job .....	5819
Data quality .....	5823
Create a Baseline .....	5824
Schedule data quality monitoring jobs .....	5827
Statistics .....	5828
CloudWatch Metrics .....	5830
Violations .....	5831
Model quality .....	5833
Create a model quality baseline .....	5834
Schedule model quality monitoring jobs .....	5837
Ingest Ground Truth labels and merge them with predictions .....	5839
Model quality metrics and Amazon CloudWatch monitoring .....	5841
Bias drift .....	5846
Model Monitor Sample Notebook .....	5847
Create a Bias Drift Baseline .....	5848
Bias Drift Violations .....	5850
Parameters to Monitor Bias Drift .....	5851
Schedule Bias Drift Monitoring Jobs .....	5856
Inspect Reports for Data Bias Drift .....	5858
CloudWatch Metrics for Bias Drift Analysis .....	5859
Feature attribution drift .....	5860
Model Monitor Example Notebook .....	5862
Create a SHAP Baseline .....	5862
Feature Attribution Drift Violations .....	5864
Parameters to Monitor Attribution Drift .....	5866
Schedule Feature Attribute Drift Monitoring Jobs .....	5870
Inspect Reports for Feature Attribute Drift .....	5872
CloudWatch Metrics for Feature Drift Analysis .....	5873
Schedule monitoring jobs .....	5874
cron scheduling .....	5877
Configuring SCPs for monitoring schedules .....	5878
Prebuilt container .....	5880
Interpret results .....	5881

List Executions .....	5881
Inspect a Specific Execution .....	5882
List Generated Reports .....	5882
Violations Report .....	5883
Visualize results for real-time endpoints .....	5884
Advanced topics .....	5890
Custom monitoring schedules .....	5890
AWS CloudFormation Custom Resource for Real-time Endpoints .....	5910
Model Monitor FAQs .....	5914
<b>Evaluate, explain, and detect bias in models .....</b>	<b>5927</b>
Evaluate foundation models .....	5927
Model evaluations .....	5929
Get started .....	5933
Prompt datasets and evaluation dimensions .....	5934
Create a model evaluation job that uses human workers .....	5963
Automatic model evaluation .....	5980
Job results .....	6009
Using the fmeval library .....	6033
Model evaluation notebook tutorials .....	6039
Troubleshooting .....	6056
Fairness and explainability .....	6061
What is fairness and model explainability? .....	6061
SageMaker Clarify Processing Jobs .....	6064
Configure a SageMaker Clarify Processing Job .....	6066
Run SageMaker Clarify Processing Jobs .....	6154
Analysis Results .....	6174
Troubleshoot Jobs .....	6189
Sample notebooks .....	6193
Pre-training Data Bias .....	6194
Post-training Data and Model Bias .....	6215
Model Explainability .....	6249
Explainability with Autopilot .....	6255
<b>Model governance .....</b>	<b>6257</b>
Amazon SageMaker Role Manager .....	6257
Amazon SageMaker Model Cards .....	6257
Amazon SageMaker Model Dashboard .....	6257

Amazon SageMaker Assets .....	6258
Model Cards .....	6258
Prerequisites .....	6259
Intended uses of a model .....	6259
Risk ratings .....	6260
Model card JSON schema .....	6260
Create a model card .....	6275
Model cards actions .....	6283
Set up cross-account support .....	6285
Model card APIs .....	6290
Model card FAQs .....	6291
Controlled access to assets .....	6293
Set up SageMaker Assets (administrator guide) .....	6294
Work with assets (user guide) .....	6299
Model Dashboard .....	6310
Model Dashboard elements .....	6310
Model Monitor schedules and alerts .....	6312
View a model lineage graph .....	6316
View Endpoint Status .....	6318
Model Dashboard FAQ .....	6320
<b>Docker containers for training and deploying models .....</b>	<b>6324</b>
Scenarios and Guidance .....	6324
Use cases for using pre-built Docker containers with SageMaker AI .....	6325
Use cases for extending a pre-built Docker container .....	6326
Use case for building your own container .....	6326
Docker container basics .....	6328
Pre-built SageMaker AI Docker images .....	6328
Support Policy .....	6329
Prebuilt Deep Learning Images .....	6334
Prebuilt Scikit-learn and Spark ML Images .....	6335
Deep Graph Networks .....	6337
Extend a Pre-built Container .....	6340
Custom Docker containers with SageMaker AI .....	6353
Individual Framework Libraries .....	6354
SageMaker Training and Inference Toolkits .....	6354
Adapting your own training container .....	6356

Adapt your own inference container for Amazon SageMaker AI .....	6374
Container creation with your own algorithms and models .....	6389
Containers with custom training algorithms .....	6389
Containers with custom inference code .....	6407
Examples and more info .....	6423
Setup .....	6423
Host models trained in Scikit-learn .....	6424
Package TensorFlow and Scikit-learn models for use in SageMaker AI .....	6424
Train and deploy a neural network on SageMaker AI .....	6424
Training using pipe mode .....	6425
Bring your own R model .....	6425
Extend a pre-built PyTorch container Image .....	6425
Train and debug training jobs on a custom container .....	6425
Troubleshooting .....	6426
<b>Configure security in Amazon SageMaker AI .....</b>	<b>6427</b>
Data Privacy .....	6428
Types of information collected .....	6428
How to opt out of metadata collection .....	6428
Additional information .....	6430
Data Protection .....	6431
Protect Data at Rest Using Encryption .....	6432
Protecting Data in Transit with Encryption .....	6435
Key Management .....	6439
Internetwork Traffic Privacy .....	6440
Identity and Access Management .....	6440
Audience .....	6441
Authenticating with identities .....	6441
Managing access using policies .....	6445
How Amazon SageMaker AI works with IAM .....	6447
Identity-based policy examples .....	6453
Cross-service confused deputy prevention .....	6492
How to use SageMaker AI execution roles .....	6501
Role Manager .....	6540
Access Control .....	6560
Amazon SageMaker AI API Permissions Reference .....	6563
AWS managed policies for SageMaker AI .....	6601

Troubleshooting .....	6768
Logging and Monitoring .....	6770
Compliance validation .....	6771
Resilience .....	6772
Infrastructure Security .....	6773
SageMaker AI Scans AWS Marketplace Training and Inference Containers for Security	
Vulnerabilities .....	6774
Connect to Amazon SageMaker AI resources from within a VPC .....	6774
Run Training and Inference Containers in Internet-Free Mode .....	6784
Connect to SageMaker AI Within your VPC .....	6785
Give SageMaker AI Access to Resources in your Amazon VPC .....	6809
<b>Algorithms and packages in the AWS Marketplace .....</b>	<b>6841</b>
Topics .....	6841
SageMaker AI Algorithms .....	6841
SageMaker AI Model Packages .....	6842
Custom algorithms and models with the AWS Marketplace .....	6842
Creation of Algorithm and Model Package Resources .....	6842
Usage of Algorithm and Model Package Resources .....	6852
Listings for your own algorithms and models with the AWS Marketplace .....	6863
Topics .....	6863
Develop Algorithms and Models in Amazon SageMaker AI .....	6864
List Your Algorithm or Model Package on AWS Marketplace .....	6866
Find and Subscribe to Algorithms and Model Packages on AWS Marketplace .....	6866
Use Algorithms and Model Packages .....	6867
<b>Tools for monitoring the AWS resources provisioned while using Amazon SageMaker AI ....</b>	<b>6869</b>
Monitoring with CloudWatch .....	6870
Endpoint Invocation Metrics .....	6870
SageMaker AI inference component metrics .....	6874
Multi-Model Endpoint Metrics .....	6876
Jobs and Endpoint Metrics .....	6878
Inference Recommender Metrics .....	6884
Ground Truth Metrics .....	6885
Feature Store Metrics .....	6888
Pipelines Metrics .....	6891
Logging with CloudWatch .....	6894
CloudTrail logs .....	6896

Amazon SageMaker AI data events in CloudTrail .....	6898
Amazon SageMaker AI management events in CloudTrail .....	6900
Operations Performed by Automatic Model Tuning .....	6900
Amazon SageMaker AI event examples .....	6900
Monitor individual user resource access from SageMaker AI Studio Classic with sourcelidentity .....	6902
Considerations when using sourcelidentity .....	6903
Turn on sourcelidentity in CloudTrail logs for SageMaker AI Studio Classic .....	6904
SageMaker AI events with EventBridge .....	6907
Model state change .....	6908
Training job state change .....	6908
HyperParameter tuning job state change .....	6910
Transform job state change .....	6912
Endpoint state change .....	6914
Feature group state change .....	6914
Model package state change .....	6916
Pipeline execution state change .....	6917
Pipeline step state change .....	6918
Processing job state change .....	6920
SageMaker image state change .....	6921
SageMaker image version state change .....	6922
Endpoint deployment state change .....	6923
Model card state change .....	6926
<b>Reference .....</b>	<b>6928</b>
ML Frameworks and Languages .....	6928
Apache MXNet .....	6929
Apache Spark .....	6930
Chainer .....	6944
Hugging Face .....	6944
PyTorch .....	6948
R .....	6949
Scikit-learn .....	6953
SparkML Serving .....	6955
TensorFlow .....	6955
Triton Inference Server .....	6957
API Reference .....	6958

---

Programming Model for Amazon SageMaker AI .....	6958
APIs, CLI, and SDKs .....	6960
SageMaker AI Document History .....	6961
Python SDK Troubleshooting .....	6977
Create a Training Job .....	6978
Update a Training Job .....	6980
Create a Processing Job .....	6981
Create an Endpoint .....	6984
Update an Endpoint .....	6985
Guidance on exception handling .....	6986

# What is Amazon SageMaker AI?

Amazon SageMaker AI is a fully managed machine learning (ML) service. With SageMaker AI, data scientists and developers can quickly and confidently build, train, and deploy ML models into a production-ready hosted environment. It provides a UI experience for running ML workflows that makes SageMaker AI ML tools available across multiple integrated development environments (IDEs).

With SageMaker AI, you can store and share your data without having to build and manage your own servers. This gives you or your organizations more time to collaboratively build and develop your ML workflow, and do it sooner. SageMaker AI provides managed ML algorithms to run efficiently against extremely large data in a distributed environment. With built-in support for bring-your-own-algorithms and frameworks, SageMaker AI offers flexible distributed training options that adjust to your specific workflows. Within a few steps, you can deploy a model into a secure and scalable environment from the SageMaker AI console.

## Topics

- [Amazon SageMaker AI rename](#)
- [Amazon SageMaker and Amazon SageMaker AI](#)
- [Pricing for Amazon SageMaker AI](#)
- [Recommendations for a first-time user of Amazon SageMaker AI](#)
- [Overview of machine learning with Amazon SageMaker AI](#)
- [Amazon SageMaker AI Features](#)

## Amazon SageMaker AI rename

On December 03, 2024, Amazon SageMaker was renamed to Amazon SageMaker AI. This name change does not apply to any of the existing Amazon SageMaker features.

## Legacy namespaces remain the same

The `sagemaker` API namespaces, along with the following related namespaces, remain unchanged for backward compatibility purposes.

- AWS CLI commands
- [Managed policies](#) containing `AmazonSageMaker` prefixes

- [Service endpoints](#) containing sagemaker
- [AWS CloudFormation](#) resources containing AWS::SageMaker prefixes
- Service-linked role containing AWSServiceRoleForSageMaker
- Console URLs containing sagemaker
- Documentation URLs containing sagemaker

## Amazon SageMaker and Amazon SageMaker AI

On December 03, 2024, Amazon released the next generation of Amazon SageMaker.

Amazon SageMaker is a unified platform for data, analytics, and AI. Bringing together AWS machine learning and analytics capabilities, the next generation of SageMaker delivers an integrated experience for analytics and AI with unified access to all your data.

Amazon SageMaker includes the following capabilities:

- Amazon SageMaker AI (formerly Amazon SageMaker) - Build, train, and deploy ML and foundation models, with fully managed infrastructure, tools, and workflows
- Amazon SageMaker Lakehouse – Unify data access across Amazon S3 data lakes, Amazon Redshift, and other data sources
- Amazon SageMaker Data and AI Governance – Discover, govern, and collaborate on data and AI securely with Amazon SageMaker Catalog, built on Amazon DataZone
- SQL Analytics - Gain insights with the most price-performant SQL engine with Amazon Redshift
- Amazon SageMaker Data Processing - Analyze, prepare, and integrate data for analytics and AI using open-source frameworks on Amazon Athena, Amazon EMR, and AWS Glue
- Amazon SageMaker Unified Studio (preview) – Build with all your data and tools for analytics and AI in a single development environment
- Amazon Bedrock - Build and scale generative AI applications

For more information, refer to [Amazon SageMaker](#).

## Pricing for Amazon SageMaker AI

For information about [AWS Free Tier](#) limits and the cost of using SageMaker AI, see [Amazon SageMaker AI Pricing](#).

# Recommendations for a first-time user of Amazon SageMaker AI

If you're a first-time user of SageMaker AI, we recommend that you complete the following:

1. [\*\*Overview of machine learning with Amazon SageMaker AI\*\*](#) – Get an overview of the machine learning (ML) lifecycle and learn about solutions that are offered. This page explains key concepts and describes the core components involved in building AI solutions with SageMaker AI.
2. [\*\*Guide to getting set up with Amazon SageMaker AI\*\*](#) – Learn how to set up and use SageMaker AI based on your needs.
3. [\*\*Automated ML, no-code, or low-code\*\*](#) – Learn about low-code and no-code ML options that simplify a ML workflow by automating machine learning tasks. These options are helpful ML learning tools because they provide visibility into the code by generating notebooks for each of the automated ML tasks.
4. [\*\*Machine learning environments offered by Amazon SageMaker AI\*\*](#) – Familiarize yourself with the ML environments that you can use to develop your ML workflow, such as information and examples about ready-to-use and custom models.
5. [\*\*Explore other topics\*\*](#) – Use the SageMaker AI Developer Guide's table of contents to explore more topics. For example, you can find information about ML lifecycle stages, in [Overview of machine learning with Amazon SageMaker AI](#), and various solutions that SageMaker AI offers.
6. [\*\*Amazon SageMaker AI resources\*\*](#) – Refer to the various developer resources that SageMaker AI offers.

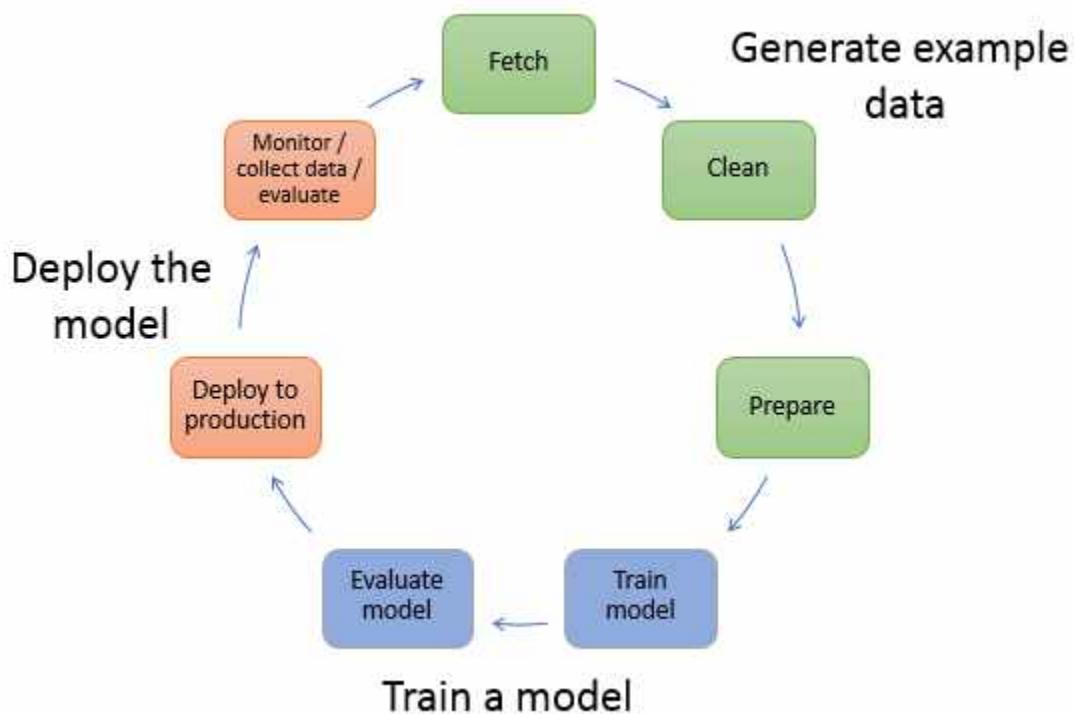
## Overview of machine learning with Amazon SageMaker AI

This section describes a typical machine learning (ML) workflow and describes how to accomplish those tasks with Amazon SageMaker AI.

In machine learning, you *teach* a computer to make predictions or inferences. First, you use an algorithm and example data to train a model. Then, you integrate your model into your application to generate inferences in real time and at scale.

The following diagram shows the typical workflow for creating an ML model. It includes three stages in a circular flow that we cover in more detail proceeding the diagram:

- Generate example data
- Train a model
- Deploy the model



The diagram shows how to perform the following tasks in most typical scenarios:

1. **Generate example data** – To train a model, you need example data. The type of data that you need depends on the business problem that you want the model to solve. This relates to the inferences that you want the model to generate. For example, if you want to create a model that predicts a number from an input image of a handwritten digit. To train this model, you need example images of handwritten numbers.

Data scientists often devote time exploring and preprocessing example data before using it for model training. To preprocess data, you typically do the following:

- a. **Fetch the data** – You might have in-house example data repositories, or you might use datasets that are publicly available. Typically, you pull the dataset or datasets into a single repository.

- b. **Clean the data** – To improve model training, inspect the data and clean it, as needed. For example, if your data has a country name attribute with values United States and US, you can edit the data to be consistent.
- c. **Prepare or transform the data** – To improve performance, you might perform additional data transformations. For example, you might choose to combine attributes for a model that predicts the conditions that require de-icing an aircraft. Instead of using temperature and humidity attributes separately, you can combine those attributes into a new attribute to get a better model.

In SageMaker AI, you can preprocess example data using [SageMaker APIs](#) with the [SageMaker Python SDK](#) in an integrated development environment (IDE). With SDK for Python (Boto3) you can fetch, explore, and prepare your data for model training. For information about data preparation, processing, and transforming your data, see [Recommendations for choosing the right data preparation tool in SageMaker AI](#), [Data transformation workloads with SageMaker Processing](#), and [Create, store, and share features with Feature Store](#).

## 2. Train a model – Model training includes both training and evaluating the model, as follows:

- **Training the model** – To train a model, you need an algorithm or a pre-trained base model. The algorithm you choose depends on a number of factors. For a built-in solution, you can use one of the algorithms that SageMaker provides. For a list of algorithms provided by SageMaker and related considerations, see [Built-in algorithms and pretrained models in Amazon SageMaker](#). For a UI-based training solution that provides algorithms and models, see [SageMaker JumpStart pretrained models](#).

You also need compute resources for training. Your resource use depends on the size of your training dataset and how quickly you need the results. You can use resources ranging from a single general-purpose instance to a distributed cluster of GPU instances. For more information, see [Train a Model with Amazon SageMaker](#).

- **Evaluating the model** – After you train your model, you evaluate it to determine whether the accuracy of the inferences is acceptable. To train and evaluate your model, use the [SageMaker Python SDK](#) to send requests to the model for inferences through one of the available IDEs. For more information about evaluating your model, see [Data and model quality monitoring with Amazon SageMaker Model Monitor](#).

## 3. Deploy the model – You traditionally re-engineer a model before you integrate it with your application and deploy it. With SageMaker AI hosting services, you can deploy your model

independently, which decouples it from your application code. For more information, see [Deploy models for inference](#).

Machine learning is a continuous cycle. After deploying a model, you monitor the inferences, collect more high-quality data, and evaluate the model to identify drift. You then increase the accuracy of your inferences by updating your training data to include the newly collected high-quality data. As more example data becomes available, you continue retraining your model to increase accuracy.

## Amazon SageMaker AI Features

Amazon SageMaker AI includes the following features.

### Topics

- [New features for re:Invent 2024](#)
- [Machine learning environments](#)
- [Major features](#)

## New features for re:Invent 2024

SageMaker AI includes the following new features for re:Invent 2024.

### [HyperPod recipes](#)

You can run recipes within Amazon SageMaker HyperPod or as SageMaker training jobs. You use the HyperPod training adapter as the framework to help you run end-to-end training workflows. The training adapter is built on the NVIDIA NeMo framework and Neuronx Distributed Training package.

### [HyperPod in Studio](#)

In Amazon SageMaker Studio, you can launch machine learning workloads on HyperPod clusters and view HyperPod cluster information. The increased visibility into cluster details and hardware metrics can help your team identify the right candidate for your pre-training or fine-tuning workloads.

## HyperPod task governance

Amazon SageMaker HyperPod task governance is a robust management system designed to streamline resource allocation and ensure efficient utilization of compute resources across teams and projects for your Amazon EKS clusters. HyperPod task governance also provides Amazon EKS cluster Observability, offering real-time visibility into cluster capacity, compute availability and usage, team allocation and utilization, and task run and wait time information.

## Amazon SageMaker Partner AI Apps

With Amazon SageMaker Partner AI Apps, users get access to generative artificial intelligence (AI) and machine learning (ML) development applications built, published, and distributed by industry-leading application providers. Partner AI Apps are certified to run on SageMaker AI. With Partner AI Apps, users can accelerate and improve how they build solutions based on foundation models (FM) and classic ML models without compromising the security of their sensitive data, which stays completely within their trusted security configuration and is never shared with a third party.

## Q Developer is available in Canvas

You can chat with Amazon Q Developer in Amazon SageMaker Canvas using natural language for generative AI assistance with solving your machine learning problems. You can converse with Q Developer to discuss the steps of a machine learning workflow and leverage Canvas functionality such as data transforms, model building, and deployment.

## SageMaker training plans

Amazon SageMaker training plans are a compute reservation capability designed for large-scale AI model training workloads running on SageMaker training jobs and HyperPod clusters. They provide predictable access to high-demand GPU-accelerated computing resources within specified timelines. You can specify a desired timeline, duration, and maximum compute resources, and SageMaker training plans automatically manages infrastructure setup, workload execution, and fault recovery. This allows for efficiently planning and executing mission-critical AI projects with a predictable cost model.

# Machine learning environments

SageMaker AI includes the following machine learning environments.

## SageMaker Canvas

An auto ML service that gives people with no coding experience the ability to build models and make predictions with them.

## Code Editor

Code Editor extends Studio so that you can write, test, debug and run your analytics and machine learning code in an environment based on Visual Studio Code - Open Source ("Code-OSS").

## SageMaker geospatial capabilities

Build, train, and deploy ML models using geospatial data.

## SageMaker HyperPod

Amazon SageMaker HyperPod is a capability of SageMaker AI that provides an always-on machine learning environment on resilient clusters that you can run any machine learning workloads for developing large machine learning models such as large language models (LLMs) and diffusion models.

## JupyterLab in Studio

JupyterLab in Studio improves latency and reliability for Studio Notebooks

## Studio

Studio is the latest web-based experience for running ML workflows. Studio offers a suite of IDEs, including Code Editor, a new Jupyterlab application, RStudio, and Studio Classic.

## Amazon SageMaker Studio Classic

An integrated machine learning environment where you can build, train, deploy, and analyze your models all in the same application.

## SageMaker Studio Lab

A free service that gives customers access to AWS compute resources in an environment based on open-source JupyterLab.

## RStudio on Amazon SageMaker AI

An integrated development environment for R, with a console, syntax-highlighting editor that supports direct code execution, and tools for plotting, history, debugging and workspace management.

## Major features

SageMaker AI includes the following major features in alphabetical order excluding any SageMaker AI prefix.

### Amazon Augmented AI

Build the workflows required for human review of ML predictions. Amazon A2I brings human review to all developers, removing the undifferentiated heavy lifting associated with building human review systems or managing large numbers of human reviewers.

### AutoML step

Create an AutoML job to automatically train a model in Pipelines.

### SageMaker Autopilot

Users without machine learning knowledge can quickly build classification and regression models.

### Batch Transform

Preprocess datasets, run inference when you don't need a persistent endpoint, and associate input records with inferences to assist the interpretation of results.

### SageMaker Clarify

Improve your machine learning models by detecting potential bias and help explain the predictions that models make.

### Collaboration with shared spaces

A shared space consists of a shared JupyterServer application and a shared directory. All user profiles in a Amazon SageMaker AI domain have access to all shared spaces in the domain.

### SageMaker Data Wrangler

Import, analyze, prepare, and featurize data in SageMaker Studio. You can integrate Data Wrangler into your machine learning workflows to simplify and streamline data pre-processing and feature engineering using little to no coding. You can also add your own Python scripts and transformations to customize your data prep workflow.

### Data Wrangler data preparation widget

Interact with your data, get visualizations, explore actionable insights, and fix data quality issues.

## SageMaker Debugger

Inspect training parameters and data throughout the training process. Automatically detect and alert users to commonly occurring errors such as parameter values getting too large or small.

## SageMaker Edge Manager

Optimize custom models for edge devices, create and manage fleets and run models with an efficient runtime.

## SageMaker Experiments

Experiment management and tracking. You can use the tracked data to reconstruct an experiment, incrementally build on experiments conducted by peers, and trace model lineage for compliance and audit verifications.

## SageMaker Feature Store

A centralized store for features and associated metadata so features can be easily discovered and reused. You can create two types of stores, an Online or Offline store. The Online Store can be used for low latency, real-time inference use cases and the Offline Store can be used for training and batch inference.

## SageMaker Ground Truth

High-quality training datasets by using workers along with machine learning to create labeled datasets.

## SageMaker Ground Truth Plus

A turnkey data labeling feature to create high-quality training datasets without having to build labeling applications and manage the labeling workforce on your own.

## SageMaker Inference Recommender

Get recommendations on inference instance types and configurations (e.g. instance count, container parameters and model optimizations) to use your ML models and workloads.

## Inference shadow tests

Evaluate any changes to your model-serving infrastructure by comparing its performance against the currently deployed infrastructure.

## SageMaker JumpStart

Learn about SageMaker AI features and capabilities through curated 1-click solutions, example notebooks, and pretrained models that you can deploy. You can also fine-tune the models and deploy them.

## SageMaker ML Lineage Tracking

Track the lineage of machine learning workflows.

## SageMaker Model Building Pipelines

Create and manage machine learning pipelines integrated directly with SageMaker AI jobs.

## SageMaker Model Cards

Document information about your ML models in a single place for streamlined governance and reporting throughout the ML lifecycle.

## SageMaker Model Dashboard

A pre-built, visual overview of all the models in your account. Model Dashboard integrates information from SageMaker Model Monitor, transform jobs, endpoints, lineage tracking, and CloudWatch so you can access high-level model information and track model performance in one unified view.

## SageMaker Model Monitor

Monitor and analyze models in production (endpoints) to detect data drift and deviations in model quality.

## SageMaker Model Registry

Versioning, artifact and lineage tracking, approval workflow, and cross account support for deployment of your machine learning models.

## SageMaker Neo

Train machine learning models once, then run anywhere in the cloud and at the edge.

## Notebook-based Workflows

Run your SageMaker Studio notebook as a non-interactive, scheduled job.

## Preprocessing

Analyze and preprocess data, tackle feature engineering, and evaluate models.

## SageMaker Projects

Create end-to-end ML solutions with CI/CD by using SageMaker Projects.

## Reinforcement Learning

Maximize the long-term reward that an agent receives as a result of its actions.

## SageMaker Role Manager

Administrators can define least-privilege permissions for common ML activities using custom and preconfigured persona-based IAM roles.

## SageMaker Serverless Endpoints

A serverless endpoint option for hosting your ML model. Automatically scales in capacity to serve your endpoint traffic. Removes the need to select instance types or manage scaling policies on an endpoint.

## Studio Classic Git extension

A Git extension to enter the URL of a Git repository, clone it into your environment, push changes, and view commit history.

## SageMaker Studio Notebooks

The next generation of SageMaker notebooks that include AWS IAM Identity Center (IAM Identity Center) integration, fast start-up times, and single-click sharing.

## SageMaker Studio Notebooks and Amazon EMR

Easily discover, connect to, create, terminate and manage Amazon EMR clusters in single account and cross account configurations directly from SageMaker Studio.

## SageMaker Training Compiler

Train deep learning models faster on scalable GPU instances managed by SageMaker AI.

# Guide to getting set up with Amazon SageMaker AI

To use the features in Amazon SageMaker AI, you must have access to Amazon SageMaker AI. To set up Amazon SageMaker AI and its features, use one of the following options.

- [\*\*Use quick setup\*\*](#): Fastest setup for individual users with default settings.
- [\*\*Use custom setup\*\*](#): Advanced setup for enterprise Machine Learning (ML) administrators. Ideal option for ML administrators setting up SageMaker AI for many users or an organization.

## Note

You do not need to set up SageMaker AI if:

- An email is sent to you inviting you to create a password to use the IAM Identity Center authentication. The email also contains the AWS access portal URL you use to sign in. For more information about signing in to the AWS access portal, see [Sign in to the AWS access portal](#).
- You intend to use the Amazon SageMaker Studio Lab ML environment. Studio Lab does not require you to have an AWS account. For information about Studio Lab, see [Amazon SageMaker Studio Lab](#).
- If you are using the AWS CLI, SageMaker APIs, or SageMaker SDKs

You do not need to set up SageMaker AI if any of the prior situations apply. You can skip the rest of this [Guide to getting set up with Amazon SageMaker AI](#) chapter and navigate to the following:

- [Automated ML, no-code, or low-code](#)
- [Machine learning environments offered by Amazon SageMaker AI](#)
- [APIs, CLI, and SDKs](#)

## Topics

- [Complete Amazon SageMaker AI prerequisites](#)
- [Use quick setup for Amazon SageMaker AI](#)
- [Use custom setup for Amazon SageMaker AI](#)

- [Amazon SageMaker AI domain overview](#)
- [Supported Regions and Quotas](#)

## Complete Amazon SageMaker AI prerequisites

Before you can set up Amazon SageMaker AI, you must complete the following prerequisites.

- **Required:** You will need to create an Amazon Web Services (AWS) account to get access to all of the AWS services and resources for the account.
- **Highly recommended:** We highly recommend that you create an administrative user to manage AWS resources for the account, to adhere to the [Security best practices in IAM](#). It is assumed that you have an administrative user for many of the administrative tasks throughout the SageMaker AI developer guide.
- **Optional:** Configure the AWS Command Line Interface (AWS CLI) if you intend to manage your AWS services and resources for the account using the AWS CLI.

### Topics

- [Sign up for an AWS account](#)
- [Create a user with administrative access](#)
- [\(Optional\) Configure the AWS CLI](#)

## Sign up for an AWS account

If you do not have an AWS account, complete the following steps to create one.

### To sign up for an AWS account

1. Open <https://portal.aws.amazon.com/billing/signup>.
2. Follow the online instructions.

Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

When you sign up for an AWS account, an *AWS account root user* is created. The root user has access to all AWS services and resources in the account. As a security best practice, assign

administrative access to a user, and use only the root user to perform [tasks that require root user access](#).

AWS sends you a confirmation email after the sign-up process is complete. At any time, you can view your current account activity and manage your account by going to <https://aws.amazon.com/> and choosing **My Account**.

## Create a user with administrative access

After you sign up for an AWS account, secure your AWS account root user, enable AWS IAM Identity Center, and create an administrative user so that you don't use the root user for everyday tasks.

### Secure your AWS account root user

1. Sign in to the [AWS Management Console](#) as the account owner by choosing **Root user** and entering your AWS account email address. On the next page, enter your password.

For help signing in by using root user, see [Signing in as the root user](#) in the *AWS Sign-In User Guide*.

2. Turn on multi-factor authentication (MFA) for your root user.

For instructions, see [Enable a virtual MFA device for your AWS account root user \(console\)](#) in the *IAM User Guide*.

### Create a user with administrative access

1. Enable IAM Identity Center.

For instructions, see [Enabling AWS IAM Identity Center](#) in the *AWS IAM Identity Center User Guide*.

2. In IAM Identity Center, grant administrative access to a user.

For a tutorial about using the IAM Identity Center directory as your identity source, see [Configure user access with the default IAM Identity Center directory](#) in the *AWS IAM Identity Center User Guide*.

## Sign in as the user with administrative access

- To sign in with your IAM Identity Center user, use the sign-in URL that was sent to your email address when you created the IAM Identity Center user.

For help signing in using an IAM Identity Center user, see [Signing in to the AWS access portal](#) in the *AWS Sign-In User Guide*.

## Assign access to additional users

1. In IAM Identity Center, create a permission set that follows the best practice of applying least-privilege permissions.

For instructions, see [Create a permission set](#) in the *AWS IAM Identity Center User Guide*.

2. Assign users to a group, and then assign single sign-on access to the group.

For instructions, see [Add groups](#) in the *AWS IAM Identity Center User Guide*.

When you create an administrative user to set up SageMaker AI, the administrative user should include specific permissions to create SageMaker AI resources. To view the permissions, expand the following administrator permissions section.

## **Administrator permissions**

When you create your administrative user using the preceding instructions, your administrative user should already include the permissions contained in the [AmazonSageMakerFullAccess](#) policy, as well as the following permissions. These policies are needed to create a SageMaker AI domain among other tasks.

If you intend to create your own custom policy, these permissions are required to create a domain and get set up with SageMaker AI. For information about adding policies, see [Adding and removing IAM identity permissions](#) in the *AWS Identity and Access Management User Guide*.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [
```

```
        "sagemaker:*"
    ],
    "Resource": [
        "arn:aws:sagemaker:*:*:domain/*",
        "arn:aws:sagemaker:*:*:user-profile/*",
        "arn:aws:sagemaker:*:*:app/*",
        "arn:aws:sagemaker:*:*:flow-definition/*"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "iam:GetRole",
        "servicecatalog:*"
    ],
    "Resource": [
        "*"
    ]
}
]
```

**Optional:** If you intend to manage your AWS services and resources for the account using the AWS CLI, proceed to the following instructions ([\(Optional\) Configure the AWS CLI](#)).

**After you have completed your prerequisites**, continue on to the setup instructions. You can continue on to your setup instructions by choosing one of the following options.

- [\*\*Use quick setup\*\*](#): Fastest setup for individual users with default settings.
- [\*\*Use custom setup\*\*](#): Advanced setup for enterprise Machine Learning (ML) administrators. Ideal option for ML administrators setting up SageMaker AI for many users or an organization.

## (Optional) Configure the AWS CLI

To manage your domain and other AWS services and resources using the AWS CLI, complete the setup in [Set up the AWS CLI](#) in the *AWS Command Line Interface User Guide for Version 2*.

**After you have completed your prerequisites**, continue on to the setup instructions. You can continue on to your setup instructions by choosing one of the following options.

- [\*\*Use quick setup\*\*](#): Fastest setup for individual users with default settings.

- **Use custom setup**: Advanced setup for enterprise Machine Learning (ML) administrators. Ideal option for ML administrators setting up SageMaker AI for many users or an organization.

## Use quick setup for Amazon SageMaker AI

The **Set up for single users** (quick setup) procedure gets you set up with default settings. Use this option if you want to get started with SageMaker AI quickly and you do not intend to customize your settings at this time. The default settings include granting access to the common SageMaker AI services for individual users to get started. For example, Amazon SageMaker Studio and Amazon SageMaker Canvas.

### Setup for single users (Quick setup)

After satisfying the prerequisites in [Complete Amazon SageMaker AI prerequisites](#), use the following instructions.

1. Open the [SageMaker AI console](#).
2. Open the left navigation pane.
3. Under **Admin configurations**, choose **Domains**.
4. Choose **Create domain**.
5. Choose **Set up for single user (Quick setup)**. Your domain and user profile are created automatically.

The **Set up for single user** process creates a domain and user profile for you automatically. If you want to learn about how the domain is set up for you when using the quick setup option, expand the following section.

#### Default settings

When you onboard to Amazon SageMaker AI domain using the **Set up for single user** procedure, your domain is automatically set up with the following default settings. For information about domains, see [Amazon SageMaker AI domain overview](#).

- **Domain name**: SageMaker AI automatically assigns the name of the domain with a timestamp in the following format.

QuickSetupDomain-YYYYMMDDTHHMMSS

- **User profile name:** SageMaker AI automatically assigns the name of the user profile with a timestamp in the following format.

default-YYYYMMDDTHHMMSS

- **Domain execution role:** SageMaker AI creates a new IAM role and attaches the [AmazonSageMakerFullAccess](#) policy. When using the quick setup and the updated Amazon SageMaker Studio is your default experience, your IAM role also includes the [AmazonSageMakerCanvasFullAccess](#), [AmazonSageMakerCanvasAI ServicesAccess](#), [AmazonS3FullAccess](#) policies.
- **User profile execution role:** SageMaker AI sets the user profile execution role to the same IAM role used for the domain execution role.
- **Shared space execution role:** SageMaker AI sets the shared space execution role to the same IAM role used for the domain execution role.
- **SageMaker Canvas time series forecasting role:** SageMaker AI creates a new IAM role with the permissions required to use the SageMaker Canvas time series forecasting feature.
- **Amazon S3 bucket:** SageMaker AI creates an Amazon S3 bucket named with the following format.

sagemaker-studio-XXXXXXXXXXXXXX

- **Amazon VPC:** SageMaker AI selects a public VPC with the following logic.
  1. If there is a default VPC with associated subnets in the Region, SageMaker AI uses it.
  2. If there is no default VPC or the default VPC has no associated subnets, then SageMaker AI uses any existing VPC with associated subnets. If there are multiple existing VPCs, SageMaker AI can select any of them.
- **Studio experience:** Amazon SageMaker Studio is set as the UI default experience and Studio Classic is made hidden. That is, in [UserSettings](#):
  - DefaultLandingUri is set to studio::.
  - [StudioWebPortalSettings](#) HiddenAppTypes is set to ["JupyterServer"]

For information about hidden applications, see [Hide machine learning tools and applications in the Amazon SageMaker Studio UI](#).

After the domain is set up, the administrative user can [Edit domain settings](#).

## After quick setup

Do you want to start SageMaker AI features right away, and do not intend to learn about domains or customize your domain? If so, skip the rest of this [Guide to getting set up with Amazon SageMaker AI](#) chapter and do the following:

- Open the [SageMaker AI console](#) and choose an environment from the left navigation pane. For example, choose **Studio** from the left navigation pane and choose **Open Studio**.
- Begin learning how to:
  - [Automated ML, no-code, or low-code](#)
  - [Machine learning environments offered by Amazon SageMaker AI](#)

RStudio support is not currently available when onboarding using the [Set up for single users \(Use quick setup for Amazon SageMaker AI\)](#) option. To use RStudio, you must onboard using the [Set up for organizations \(Use custom setup for Amazon SageMaker AI\)](#) option. For more information, see [Use custom setup for Amazon SageMaker AI](#).

## Use custom setup for Amazon SageMaker AI

The **Set up for organizations** (custom setup) guides you through an advanced setup for your Amazon SageMaker AI domain. This option provides information and recommendations to help you understand and control all aspects of the account configuration, including permissions, integrations, and encryption. Use this option if you want to set up a custom domain. For information about domains, see [Amazon SageMaker AI domain overview](#).

### Topics

- [Authentication methods](#)
- [Setup for organizations \(custom setup\)](#)
- [Access the domain after onboarding](#)

## Authentication methods

Before you set up the domain consider the authentication methods for your users to access the domain.

**AWS Identity Center:**

- **Helps simplify administration of access permissions to groups of users.** You can grant or deny permissions to groups of users, instead of applying those permissions to each individual user. If a user moves to a different organization, you can move that user to a different AWS Identity and Access Management Identity center (AWS IAM Identity Center) group. The user then automatically receives the permissions that are needed for the new organization.

Note that the IAM Identity Center needs to be in the same AWS Region as the domain.

To set up with IAM Identity Center, use the following instructions from the *AWS IAM Identity Center User Guide*:

- Begin with [Enabling AWS IAM Identity Center](#).
- [Create a permission set](#) that follows the best practice of applying least-privilege permissions.
- [Add groups](#) to your IAM Identity Center directory.
- [Assign single sign-on access](#) to users and groups.
- View the basic workflows to [get started with common tasks in IAM Identity Center](#).
- The users in IAM Identity Center can access the domain using an AWS access portal URL that is emailed to them. The email provides instructions to create an account to access the domain. For more information, see [Sign in to the AWS access portal](#).

As an administrator you can find the AWS access portal URL by navigating to the [IAM Identity Center](#) and finding the **AWS access portal URL** under **Settings summary**.

- Your domain must use AWS Identity and Access Management (IAM) authentication if you wish to restrict access to your domains exclusively to particular Amazon Virtual Private Clouds (VPCs), interface endpoints, or a predefined set of IP addresses. This feature is not supported for domains that use IAM Identity Center authentication. You can still use IAM Identity Center to enable centralized workforce identity control. For instructions on how to implement these restrictions while keeping IAM Identity Center to provide a consistent user sign-in experience, see [Secure access to Amazon SageMaker Studio Classic with IAM Identity Center and a SAML application](#) in the *AWS machine learning blog*. Note that AWS SSO is IAM Identity Center in this blog.

## Login through IAM:

- The user profiles can access the domain through the SageMaker AI console after logging into the account.

- You can restrict access to your domains exclusively to particular Amazon Virtual Private Clouds (VPCs), interface endpoints, or a predefined set of IP addresses when using AWS Identity and Access Management (IAM) authentication. For more information, see [Allow Access Only from Within Your VPC](#).

## Setup for organizations (custom setup)

### Custom setup using the console

After satisfying the prerequisites in [Complete Amazon SageMaker AI prerequisites](#), open the **Set up SageMaker AI Domain** (custom setup) page and expand the following sections for information on the setup.

#### Open the Set up SageMaker AI Domain from the SageMaker AI console

1. Open the [SageMaker AI console](#).
2. On the left navigation pane, choose **Admin configurations** to expand the options.
3. Under **Admin configurations**, choose **Domains**.
4. From the **Domains** page, choose **Create domain**.
5. On the **Set up SageMaker AI domain** page, choose **Set up for organizations**.
6. Choose **Set up**.

Once you opened the **Set up SageMaker AI Domain** page, use the following instructions:

#### Step 1: Domain details

1. For **Domain name**, enter a unique name for your domain. For example, this can be your project or team name.
2. Choose **Next**.

#### Step 2: Users and ML Activities

In this step you set up the authentication method, users, and permissions for your domain.

1. Under **How do you want to access Studio?**, you can choose one of two options. For information on the authentication methods, see [Authentication methods](#). Details on the options are provided in the following:

- **AWS Identity Center:**

Under **Who will use Studio?** choose an AWS IAM Identity Center group that will access the domain.

If you choose **No Identity Center user group** you create a domain with no users. You can add IAM Identity Center groups to the domain after the domain's creation. For more information, see [Edit domain settings](#).

- **Login through IAM:**

Under **Who will use Studio?** choose **+ Add user**, enter a new user profile name, and choose **Add** to create and add a user profile name.

You can repeat this process to create multiple user profiles.

2. Under **Who will use Studio?** select the IAM Identity Center users or groups, then choose **Select**. You need to set up Amazon SageMaker Studio within the same Region in which your IAM Identity Center is configured. You can change the Region of your domain by choosing the Region from the dropdown list on the top right of the console or you can change your IAM Identity Center Region by navigating to the [AWS access portal](#).
3. Under **What ML activities do they perform?** you can use an existing role by choosing **Use an existing role** or you can create a new role by choosing **Create a new role** and checking the ML activities you want the role to have access.
4. While selecting ML activities, you may need to satisfy requirements. To satisfy a requirement, choose **Add** and complete the requirement.
5. After all requirements are satisfied, choose **Next**.

### Step 3: Applications

In this step, you can configure the applications you have enabled in the previous step. For more information on the ML activities, see [ML activity reference](#).

If the application has not been enabled, you receive a warning for that application. To enable an application that has not been enabled, return to the previous step by choosing **Back** and follow the previous instructions.

- **Studio configuration:**

Under **Studio**, you have the option to choose between the newer and classic version of Studio as your default experience. This means choosing which ML environment you interact with when you open Studio.

- **Studio** includes multiple integrated development environments (IDEs) and applications, including Amazon SageMaker Studio Classic. If chosen, the Studio Classic IDE has default settings. For information on the default settings, see [Default settings](#).

For information on Studio, see [Amazon SageMaker Studio](#).

- **Studio Classic** includes the Jupyter IDE. If chosen, you may configure your Studio Classic configuration.

For information on Studio Classic, see [Amazon SageMaker Studio Classic](#).

- **SageMaker Canvas** configuration:

If you have Amazon SageMaker Canvas enabled, see [Getting started with using Amazon SageMaker Canvas](#) for the instructions and configuration details for onboarding.

- **Studio Classic** configuration:

If you chose **Studio** (recommended) as your default experience, the Studio Classic IDE has default settings. For information on the default settings, see [Default settings](#).

If you chose Studio Classic as your default experience, you can choose to enable or disable notebook resource sharing. Notebook resources include artifacts such as cell output and Git repositories. For more information on Notebook resources, see [Share and Use an Amazon SageMaker Studio Classic Notebook](#).

If you enabled notebook resource sharing:

1. Under **S3 location for shareable notebook resources**, input your Amazon S3 location.
2. Under **Encryption key - optional**, leave as **No Custom Encryption** or choose an existing AWS KMS key or choose **Enter a KMS key ARN** and enter your AWS KMS key's ARN.
3. Under **Notebook cell output sharing preference**, choose **Allow users to share cell output** or **Disable cell output sharing**.

- **RStudio** configuration:

To enable RStudio, you need an RStudio license. To set that up, see [Get an RStudio license](#).

1. Under **RStudio Workbench**, verify that your RStudio license is automatically detected. For more information about getting an RStudio license and activating it with SageMaker AI, see [Get an RStudio license](#).
2. Select an instance type to launch your RStudio Server on. For more information, see [RStudioServerPro instance type](#).
3. Under **Permission**, create your role or select an existing role. The role must have the following permissions policy. This policy allows the RStudioServerPro application to access necessary resources. It also allows Amazon SageMaker AI to automatically launch an RStudioServerPro application when the existing RStudioServerPro application is in a Deleted or Failed status. For information about adding permissions to a role, see [Modifying a role permissions policy \(console\)](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "VisualEditor0",  
            "Effect": "Allow",  
            "Action": [  
                "license-manager:ExtendLicenseConsumption",  
                "license-manager>ListReceivedLicenses",  
                "license-manager:GetLicense",  
                "license-manager:CheckoutLicense",  
                "license-manager:CheckInLicense",  
                "logs>CreateLogDelivery",  
                "logs>CreateLogGroup",  
                "logs>CreateLogStream",  
                "logs>DeleteLogDelivery",  
                "logs>Describe*",  
                "logs>GetLogDelivery",  
                "logs>GetLogEvents",  
                "logs>ListLogDeliveries",  
                "logs>PutLogEvents",  
                "logs>PutResourcePolicy",  
                "logs>UpdateLogDelivery",  
                "sagemaker>CreateApp"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

{

4. Under **RStudio Connect**, add the URL for your RStudio Connect server. RStudio Connect is a publishing platform for Shiny applications, R Markdown reports, dashboards, plots, and more. When you onboard to RStudio on SageMaker AI, an RStudio Connect server is not created. For more information, see [Add an RStudio Connect URL](#).
5. Under **RStudio Package Manager**, add the URL for your RStudio Package Manager. SageMaker AI creates a default package repository for the Package Manager when you onboard RStudio. For more information about RStudio Package Manager, see [Update the RStudio Package Manager URL](#).
6. Select **Next**.

- **Code Editor** configuration:

If you have Code Editor enabled, see [Code Editor in Amazon SageMaker Studio](#) for an overview and the configuration details.

## Step 4: Customize Studio UI

In this section you can customize the viewable applications and machine learning (ML) tools displayed in Studio. This customization only hides the applications and ML tools in the left navigation pane in Studio. For information on the Studio UI, see [Amazon SageMaker Studio UI overview](#).

For information about the applications, see [Applications supported in Amazon SageMaker Studio](#).

The customize Studio UI feature is not available in Studio Classic. If you wish to set Studio as your default experience, choose **Previous** and to return to the previous step.

1. On the **Customize Studio UI** page you can hide applications and ML tools displayed in Studio by toggling them off.
2. Once you have reviewed your changes, choose **Next**.

## Step 5: Set up network settings

Choose how you want Studio to connect to other AWS services.

You can choose to disable internet access to your Studio by specifying using **Virtual Private Cloud (VPC) Only** network access type. If you choose this option, you cannot run a Studio notebook unless your VPC has an interface endpoint to the SageMaker API and runtime, or a Network

Address Translation (NAT) gateway with internet access, and your security groups allow outbound connections. For more information on Amazon VPCs, see [Choose an Amazon VPC](#).

If you choose Virtual Private Cloud (VPC) Only the following steps are required. If you choose **Public internet access**, the first two of the following steps are required.

1. Under **VPC**, choose the Amazon VPC ID.
2. Under **Subnet**, choose one or more subnets. If you don't choose any subnets, SageMaker AI uses all the subnets in the Amazon VPC. We recommend that you use multiple subnets that are not created in constrained Availability Zones. Using subnets in these constrained Availability Zones can result in insufficient capacity errors and longer application creation times. For more information about constrained Availability Zones, see [Availability Zones](#).
3. Under **Security group(s)**, choose one or more subnets.

If **VPC only** is selected, SageMaker AI automatically applies the security group settings defined for the domain to all shared spaces created in the domain. If **Public internet only** is selected, SageMaker AI does not apply the security group settings to shared spaces created in the domain.

## Step 6: Configure storage

You have the option to encrypt your data. The [Amazon Elastic File System \(Amazon EFS\)](#) and [Amazon Elastic Block Store \(Amazon EBS\)](#) file systems that are created for you when you create a domain. Amazon EBS sizes are used by both Code Editor and JupyterLab spaces.

You cannot change the encryption key after you encrypt your Amazon EFS and Amazon EBS file systems. To encrypt your Amazon EFS and Amazon EBS file systems, you can use the following configurations.

- Under **Encryption key - optional**, leave as **No Custom Encryption** or choose an existing KMS key or choose **Enter a KMS key ARN** and enter the ARN of your KMS key.
- Under **Default space size - optional**, enter the default space size.
- Under **Maximum space size - optional**, enter the maximum space size.

## Step 7: Review and create

Review your domain settings. If you need to change the settings, choose **Edit** next to the relevant step. Once you confirm that your domain settings are accurate, choose **Submit** and the domain is created for you. This process may take a few minutes.

## Custom setup using the AWS CLI

The following sections provide AWS CLI instructions for the custom setup your domain using the IAM Identity Center or IAM authentication methods.

After satisfying the prerequisites, including setting up your AWS CLI credentials, in [Complete Amazon SageMaker AI prerequisites](#), use the following the steps.

1. Create an execution role that is used to create a domain and attach the [AmazonSageMakerFullAccess](#) policy. You can also use an existing role that has, at a minimum, an attached trust policy that grants SageMaker AI permission to assume the role. For more information, see [How to use SageMaker AI execution roles](#).

```
aws iam create-role --role-name execution-role-name --assume-role-policy-document file://execution-role-trust-policy.json
aws iam attach-role-policy --role-name execution-role-name --policy-arn
arn:aws:iam::aws:policy/AmazonSageMakerFullAccess
```

2. Get the default Amazon Virtual Private Cloud (Amazon VPC) of your account.

```
aws --region region ec2 describe-vpcs --filters Name=isDefault,Values=true --query
"Vpcs[0].VpcId" --output text
```

3. Get the list of subnets in the default Amazon VPC.

```
aws --region region ec2 describe-subnets --filters Name=vpc-id,Values=default-vpc-id --query "Subnets[*].SubnetId" --output json
```

4. Create a domain by passing the default Amazon VPC ID, subnets, and execution role ARN. You must also pass a SageMaker image ARN. For information on the available JupyterLab version ARNs, see [Setting a default JupyterLab version](#).

For *authentication-mode*, use SSO for IAM Identity Center authentication or IAM for IAM authentication.

```
aws --region region sagemaker create-domain --domain-name domain-name --vpc-id default-vpc-id --subnet-ids subnet-ids --auth-mode authentication-mode --default-user-settings
"ExecutionRole=arn:aws:iam::account-number:role/execution-role-name,JupyterServerAppSettings={DefaultResourceSpec={InstanceType=system,SageMakerImageArn=arn}}" \ --query DomainArn --output text
```

You can use the AWS CLI to customize the applications and ML tools displayed in Studio for the domain, using [StudioWebPortalSettings](#). Use `HiddenAppTypes` to hide applications and `HiddenMLTools` to hide ML tools. For more information on customizing the left navigation of the Studio UI, see [Hide machine learning tools and applications in the Amazon SageMaker Studio UI](#). This feature is not available for Studio Classic.

5. Verify that the domain has been created.

```
aws --region region sagemaker list-domains
```

## Custom setup using AWS CloudFormation

For information about creating a domain using AWS CloudFormation, see [AWS::SageMaker::Domain](#) in the *AWS CloudFormation User Guide*.

For an example of an AWS CloudFormation template that you can use to set up your domain, see [Creating Amazon SageMaker AI domains using AWS CloudFormation](#) in the `aws-samples` GitHub repository.

After the domain is set up, the administrative user can view and edit the domain. For information, see [View domains](#) and [Edit domain settings](#).

## Access the domain after onboarding

The users can access SageMaker AI using:

- The sign-in URL if the domain was set up using the IAM Identity Center authentication. For information, see [How to sign in to the user portal](#).
- The [SageMaker AI console](#).

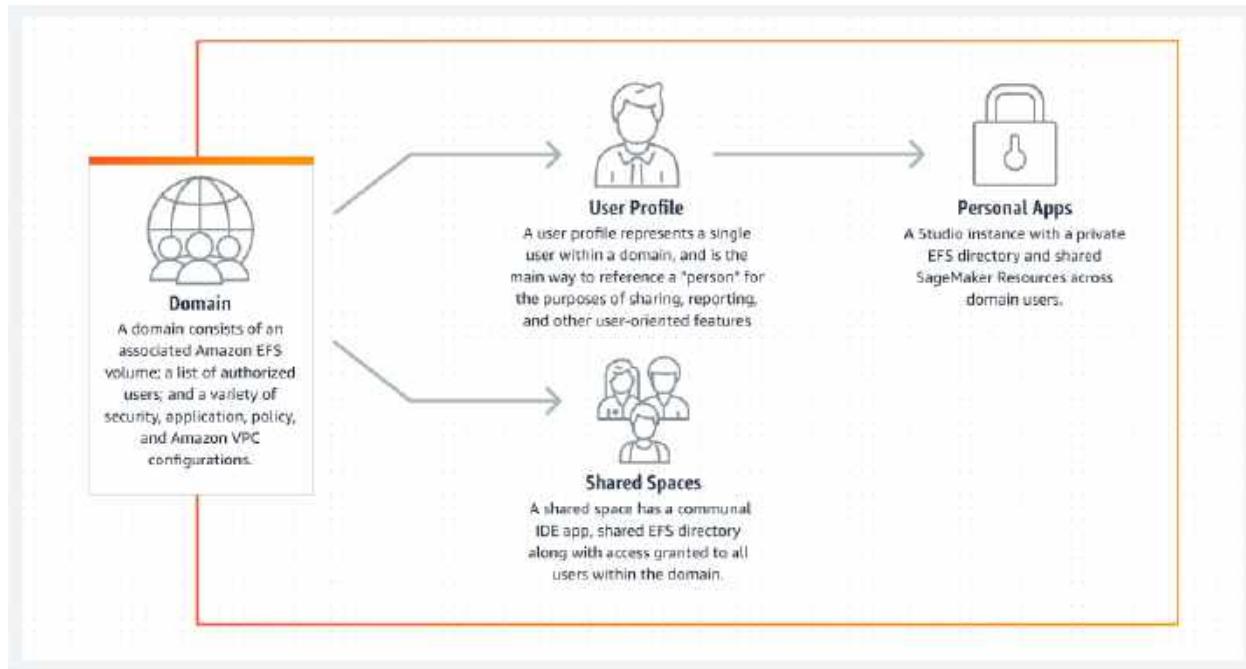
## Amazon SageMaker AI domain overview

Amazon SageMaker AI uses domains to organize user profiles, applications, and their associated resources. An Amazon SageMaker AI domain consists of the following:

- An associated Amazon Elastic File System (Amazon EFS) volume
- A list of authorized users

- A variety of security, application, policy, and Amazon Virtual Private Cloud (Amazon VPC) configurations

The following diagram provides an overview of private apps and shared spaces within each domain.



To have access to most Amazon SageMaker AI environments and resources, you must complete the Amazon SageMaker AI domain onboarding process using the SageMaker AI console or the AWS CLI. For a guide describing how to get started using SageMaker AI based on how you want to access SageMaker AI, and if necessary how to set up a domain, see [Guide to getting set up with Amazon SageMaker AI](#).

## Topics

- [Amazon SageMaker AI domain entities and statuses](#)
- [Choose an Amazon VPC](#)

## Amazon SageMaker AI domain entities and statuses

Amazon SageMaker AI domain supports SageMaker AI machine learning (ML) environments. A SageMaker AI domain is composed of the following entities and their associated status values. For onboarding steps to create a domain, see [Amazon SageMaker AI domain overview](#).

- **Domain:** A domain consists of the following.

- An associated Amazon Elastic File System (Amazon EFS) volume.
- A list of authorized users.
- A variety of security, application, policy, and Amazon Virtual Private Cloud (Amazon VPC) configurations.

Users within a domain can share notebook files and other artifacts with each other. An account can have multiple domains. For more information about multiple domains, see [Multiple domains overview](#).

- **User profile:** A user profile represents a single user within a domain. It is the main way to reference a user for the purposes of sharing, reporting, and other user-oriented features. This entity is created when a user onboards to the Amazon SageMaker AI domain. For more information about user profiles, see [Domain user profiles](#).
- **Shared space:** A shared space consists of a shared JupyterServer application and shared directory. All users within the domain have access to the shared space. All user profiles in a domain have access to all shared spaces in the domain. For more information about shared spaces, see [Collaboration with shared spaces](#).
- **App:** An app represents an application that supports the reading and execution experience of the user's notebooks, terminals, and consoles. The type of app can be JupyterServer, KernelGateway, RStudioServerPro, or RSession. A user may have multiple apps active simultaneously.

The following tables describe the status values for the domain, UserProfile, shared space, and App entities. Where applicable, they also give troubleshooting steps.

#### domain status values

Value	Description
Pending	Ongoing creation of domain.
InService	Successful creation of domain.
Updating	Ongoing update of domain.
Deleting	Ongoing deletion of domain.
Failed	Unsuccessful creation of domain. Call the <code>DescribeDomain</code> API to see the failure

Value	Description
	reason for domain creation. Delete the failed domain and recreate the domain after fixing the error mentioned in <code>FailureReason</code> .
Update_Failed	Unsuccessful update of domain. Call the <code>DescribeDomain</code> API to see the failure reason for domain update. Call the <code>UpdateDomain</code> API after fixing the error mentioned in <code>FailureReason</code> .
Delete_Failed	Unsuccessful deletion of domain. Call the <code>DescribeDomain</code> API to see the failure reason for domain deletion. Because deletion failed, you might have some resources that are still running, but you cannot use or update the domain. Call the <code>DeleteDomain</code> API again after fixing the error mentioned in <code>FailureReason</code> .

## UserProfile status values

Value	Description
Pending	Ongoing creation of <code>UserProfile</code> .
InService	Successful creation of <code>UserProfile</code> .
Updating	Ongoing update of <code>UserProfile</code> .
Deleting	Ongoing deletion of <code>UserProfile</code> .
Failed	Unsuccessful creation of <code>UserProfile</code> . Call the <code>DescribeUserProfile</code> API to see the failure reason for <code>UserProfile</code> creation. Delete the failed <code>UserProfile</code> and

Value	Description
	recreate it after fixing the error mentioned in FailureReason .
Update_Failed	Unsuccessful update of UserProfile . Call the DescribeUserProfile API to see the failure reason for UserProfile update. Call the UpdateUserProfile API again after fixing the error mentioned in FailureReason .
Delete_Failed	Unsuccessful deletion of UserProfile . Call the DescribeUserProfile API to see the failure reason for UserProfile deletion. Because deletion failed, you might have some resources that are still running, but you cannot use or update the UserProfile . Call the DeleteUserProfile API again after fixing the error mentioned in FailureReason .

## shared space status values

Value	Description
Pending	Ongoing creation of shared space.
InService	Successful creation of shared space.
Deleting	Ongoing deletion of shared space.
Failed	Unsuccessful creation of shared space. Call the DescribeSpace API to see the failure reason for shared space creation. Delete the failed shared space and recreate it after fixing the error mentioned in FailureReason .

Value	Description
Update_Failed	Unsuccessful update of shared space. Call the <code>DescribeSpace</code> API to see the failure reason for shared space update. Call the <code>UpdateSpace</code> API again after fixing the error mentioned in <code>FailureReason</code> .
Delete_Failed	Unsuccessful deletion of shared space. Call the <code>DescribeSpace</code> API to see the failure reason for shared space deletion. Because deletion failed, you might have some resources that are still running, but you cannot use or update the shared space. Call the <code>DeleteSpace</code> API again after fixing the error mentioned in <code>FailureReason</code> .
Deleted	Successful deletion of shared space.

## App status values

Value	Description
Pending	Ongoing creation of App.
InService	Successful creation of App.
Deleting	Ongoing deletion of App.
Failed	Unsuccessful creation of App. Call the <code>DescribeApp</code> API to see the failure reason for App creation. Call the <code>CreateApp</code> API again after fixing the error mentioned in <code>FailureReason</code> .
Deleted	Successful deletion of App.

## Maintenance of applications

At least once every 90 days, SageMaker AI performs security and performance updates to the underlying software for Amazon SageMaker Studio Classic JupyterServer and KernelGateway, SageMaker Canvas, and Amazon SageMaker Data Wrangler applications. Some maintenance items, such as operating system upgrades, require that SageMaker AI takes your application offline for a short time during the maintenance window. Because this maintenance takes the application offline, you cannot perform any operations while the underlying software is being updated. When the maintenance activity is in progress, the state of the application transitions from **InService** to **Pending**. When maintenance is complete, the status of the application transitions back to **InService**. If patching fails, then the status of the application becomes **Failed**. If an application is in the **Failed** state, we recommend creating a new application of the same type. For information about creating Studio Classic applications, see [Shut Down and Update SageMaker Studio Classic and Studio Classic Apps](#). For information about creating SageMaker Canvas applications, see [Applications management](#).

For more information, contact <https://aws.amazon.com/premiumsupport/>.

### Topics

- [Complete prerequisites](#)
- [Hide machine learning tools and applications in the Amazon SageMaker Studio UI](#)
- [Hide instance types and images in the Amazon SageMaker Studio UI](#)
- [Multiple domains overview](#)
- [Isolate domain resources](#)
- [domain default settings](#)
- [Custom tag propagation](#)
- [Adding a custom file system to a domain](#)
- [View domain environment details](#)
- [View domains](#)
- [Edit domain settings](#)
- [Delete an Amazon SageMaker AI domain](#)
- [Domain user profiles](#)
- [IAM Identity Center groups in a domain](#)
- [Understanding domain space permissions and execution roles](#)

- [View SageMaker AI resources in your domain](#)
- [Shut down SageMaker AI resources in your domain](#)
- [Where to shut down resources per SageMaker AI features](#)

## Complete prerequisites

To use the features available in an Amazon SageMaker AI domain, you must complete the following prerequisites.

- Onboard to a domain. For more information, see [Onboard to Amazon SageMaker AI domain](#).
- (Optional) If you are interacting with your domain using the AWS CLI, you must also complete the following prerequisites.
  - Update the AWS CLI by following the steps in [Installing the current AWS CLI Version](#).
  - From your local machine, run `aws configure` and provide your AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).

## Hide machine learning tools and applications in the Amazon SageMaker Studio UI

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

This topic shows how to hide applications and machine learning (ML) tools displayed in the Amazon SageMaker Studio user interface (UI). For information on the Studio UI, see [Amazon SageMaker Studio UI overview](#).

This customization does not block access to these resources. If, instead, you want to block access to an application, see [Amazon SageMaker Role Manager](#).

For information about the applications, see [Applications supported in Amazon SageMaker Studio](#).

The customize Studio UI feature is not available in Amazon SageMaker Studio Classic.

You can customize the Studio UI on a domain level and a user level:

- Customization on a domain level sets the default for all users in the domain.

These default settings apply for all users in the domain who have *not* had these changes made to their individual user settings.

- Customization on a user level will take priority over the domain level settings.

Use the following topics to learn more on the different customization levels and how to apply them.

## Topics

- [Hide machine learning tools and applications on a domain level](#)
- [Hide machine learning tools and applications on a user level](#)

### Hide machine learning tools and applications on a domain level

The following shows how to use the console to customize the applications and ML tools displayed in Studio on a domain level. For more information, see [Hide machine learning tools and applications in the Amazon SageMaker Studio UI](#).

This feature is not available if Amazon SageMaker Studio Classic is set as your default experience.

#### Hide machine learning tools and applications on a domain level instructions (console)

##### To hide machine learning tools and applications Studio UI on a domain level (console)

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, choose the link to the domain you wish to edit.
5. On the **Domain details** page, choose the **App Configurations** tab.
6. In the **SageMaker Studio** section, choose **Customize Studio UI**.
7. On the **Customize Studio UI** page you can hide applications and ML tools displayed in Studio by toggling them off.

Note that not all ML features are available in all regions.

8. Once you have reviewed your changes, choose **Save**.

Once completed, you will see a green banner containing a success message at the top of the page.

## Hide machine learning tools and applications on a domain level instructions (AWS CLI)

### Note

To use this feature you may need to update to the latest AWS CLI version. For more information, see [Installing or updating to the latest version of the AWS CLI](#).

You can use the AWS CLI to customize the applications and ML tools displayed in Studio on a domain level, using [StudioWebPortalSettings](#). Use `HiddenAppTypes` to hide applications and `HiddenMLTools` to hide ML tools.

In the following example, SageMaker Canvas and Code Editor are being hidden for users in the domain `domainId`.

```
aws sagemaker update-domain \
--domain-id domainId \
--default-user-settings '{"StudioWebPortalSettings": {"HiddenAppTypes": ["Canvas", "CodeEditor"]}}'
```

Note that not all ML features are available in all AWS Regions.

## Hide machine learning tools and applications on a user level

The following shows how to customize the applications and ML tools displayed in Studio on a user level. For more information, see [Hide machine learning tools and applications in the Amazon SageMaker Studio UI](#).

This feature is not available if Studio Classic is set as your default experience.

## Hide machine learning tools and applications on a user level instructions (console)

### To hide machine learning tools and applications Studio UI on a user level (console)

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.

4. From the list of domains, choose the link to the domain you wish to edit.
5. On the **Domain details** page, choose the **User profiles** tab.
6. In the **User profiles** section, choose the link to the user profile you wish to edit.
7. Choose the **App Configurations** tab.
8. In the **SageMaker Studio** section, choose **Customize Studio UI**.
9. On the **Customize Studio UI** page you can hide applications and ML tools displayed in Studio by toggling them off.

Note that not all ML features are available in all regions.

10. Once you have reviewed your changes, choose **Save**. This will take you back to the user profile edit flow.
11. Choose **Save changes**.

Once completed, you will see a green banner containing a success message at the top of the page.

### Hide machine learning tools and applications on a user level instructions (AWS CLI)

#### Note

To use this feature you may need to update to the latest AWS CLI version. For more information, see [Installing or updating to the latest version of the AWS CLI](#).

You can use the AWS CLI to customize the applications and ML tools displayed in Studio on a user level, using [StudioWebPortalSettings](#). Use `HiddenAppTypes` to hide applications and `HiddenMLTools` to hide ML tools.

In the following example, SageMaker Canvas and Code Editor are being hidden for user `userProfileName` in the domain `domainId`.

```
aws sagemaker update-user-profile \
--domain-id domainId \
--user-profile-name userProfileName \
--user-settings '{"StudioWebPortalSettings": {"HiddenAppTypes": ["Canvas", "CodeEditor"]}}'
```

Note that not all ML features are available in all AWS Regions.

## Hide instance types and images in the Amazon SageMaker Studio UI

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

This topic shows how to hide Amazon SageMaker AI instance types and images displayed in the Amazon SageMaker Studio user interface (UI). For information on the Studio UI, see [Amazon SageMaker Studio UI overview](#).

When you hide SageMaker AI instance types and images:

- The impacted users will not be able to view the hidden resources in the Studio UI.
- The impacted users will not be able to run or create a new space with the hidden configurations.
- Any currently running spaces for the impacted users will not be effected.
- When an impacted user attempts to run a space with the hidden resources, they will be notified that the relevant resources have been disabled by the administrator.

### Note

If, instead of *hiding*, you would like to *restrict* the instance types available to users through an AWS Identity and Access Management policy, see:

- [Can I limit the type of instances that data scientists can launch for training jobs in SageMaker AI?](#) in AWS re:Post.
- [Limiting instances types on Amazon SageMaker AI via IAM policy](#) in StackOverflow.

The customize Studio UI feature is not available in Amazon SageMaker Studio Classic.

You can customize the Studio UI on a domain level and a user level:

- Customization on a domain level sets the default for all users in the domain.
- Customization on a user level will take priority over the domain level settings.

Use the following topics to learn more on the different customization levels and how to apply them.

## Topics

- [Hide instance types and images on a domain level](#)
- [Hide instance types and images on a user level](#)

### Hide instance types and images on a domain level

The following shows how to use the console to set rules to hide Amazon SageMaker AI instance types and images from being displayed in the Amazon SageMaker Studio Classic UI on a *domain level*. For more information, see [Hide instance types and images in the Amazon SageMaker Studio UI](#).

Once these changes are made on a domain level:

- These changes will not effect any currently open spaces.
- These changes will impact the domain's users' *default* visibility from that point onward.

These default settings apply for all users in the domain who have *not* had these changes made to their individual user settings.

- User level settings take priority over the domain level settings.

The customize Studio UI feature is not available in Amazon SageMaker Studio Classic.

### Hide instance types and images on a domain level instructions (console)

#### To hide instance types and images Studio UI on a domain level (console)

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, choose the link to the domain you wish to edit.
5. On the **Domain details** page, choose **Domain settings**.
6. In the **Domain settings** tab, you can view the domain rules in the **Domain rules** section.
7. In the **Domain rules** section choose **Manage rules**.

8. On the **Manage domain rules** page choose a **Rule type**.

Note that not all instance types and images are available in all AWS Regions.

- a. If you choose **Instance type**, you can use the **Hide** action to hide SageMaker AI instance types you choose in the dropdown list under **Instance types**.
- b. If you choose **Image**, you can use the **Hide** action to hide SageMaker images you choose under the dropdown list under **Image**.

9. (Optional) Choose **+ Add new rule** to add more rules.

10. Once you have reviewed your changes, choose **Submit**.

Once completed, you will see a green banner containing a success message at the top of the page.

### Hide instance types and images on a domain level instructions (AWS CLI)

 **Note**

To use this feature you may need to update to the latest AWS CLI version. For more information, see [Installing or updating to the latest version of the AWS CLI](#).

You can use the AWS CLI to customize the SageMaker AI instances and images displayed in the Studio UI on a domain level, using [StudioWebPortalSettings](#). Use `HiddenInstanceTypes` to hide instance types and use `HiddenSageMakerImageVersionAliases` to hide SageMaker images.

Note that when you use `HiddenSageMakerImageVersionAliases`:

- The API only accepts minor `VersionAliases` (for example, `1.9`), rather than patch versions (For example, `1.9.1`).
- You may enter unpublished versions through the CLI or SDK. However, these versions will not be displayed in the console and will be overwritten after the rules are edited through the console.

In the following example, for Code Editor, based on Code-OSS, Visual Studio Code - Open Source and JupyterLab, the following are being hidden for users by default in the domain `domainId`:

- The instance types `ml.r6id.24xlarge` and `ml.r6id.32xlarge`.
- The image `sagemaker_distribution` versions `1.9` and `1.8`.

```
aws sagemaker update-domain \
--domain-id domainId \
--default-user-settings '{
    "StudioWebPortalSettings": {
        "HiddenInstanceTypes": [ "ml.r6id.24xlarge", "ml.r6id.32xlarge" ],
        "HiddenSageMakerImageVersionAliases": [
            {
                "SageMakerImageName": "sagemaker_distribution",
                "VersionAliases": [ "1.9", "1.8" ]
            }
        ]
    }
}'
```

Note that not all instance types and images are available in all AWS Regions.

## Hide instance types and images on a user level

### Warning

Customizing a user profile is a permanent action. If custom settings are saved, this user profile will overwrite the domain settings, and no longer dynamically update with the domain in the future.

The following shows how to use the console to set rules to hide Amazon SageMaker AI instance types and images from being displayed in the Amazon SageMaker Studio Classic UI on a *user level*. For more information, see [Hide instance types and images in the Amazon SageMaker Studio UI](#).

This setting will take priority over the domain level settings.

The customize Studio UI feature is not available in Studio Classic.

## Hide instance types and images on a user level instructions (console)

### To hide instance types and images Studio UI on a user level (console)

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.

4. From the list of domains, choose the link to the domain you wish to edit.
5. On the **Domain details** page, choose the **User profiles** tab.
6. In the **User profiles** section, choose the link to the user profile you wish to edit.
7. On the User details tab, you can view the rules applied to the user in the User profile rules section.
8. In the User profile rules section choose Manage rules.
9. On the Manage user profile rules page choose a Rule type.

Note that not all instance types and images are available in all AWS Regions.

- a. If you choose **Instance type**, you can use the **Hide** action to hide SageMaker AI instance types you choose in the dropdown list under **Instance types**.
  - b. If you choose **Image**, you can use the **Hide** action to hide SageMaker images you choose under the dropdown list under **Image**.
10. (Optional) Choose **+ Add new rule** to add more rules.
  11. Once you have reviewed your changes, choose **Submit**.

Once completed, you will see a green banner containing a success message at the top of the page.

### Hide instance types and images on a user level instructions (AWS CLI)

 **Note**

To use this feature you may need to update to the latest AWS CLI version. For more information, see [Installing or updating to the latest version of the AWS CLI](#).

You can use the AWS CLI to customize the applications and ML tools displayed in Studio on a user level, using [StudioWebPortalSettings](#). Use `HiddenInstanceTypes` to hide instance types and use `HiddenSageMakerImageVersionAliases` to hide SageMaker images.

Note that when you use `HiddenSageMakerImageVersionAliases`:

- The API only accepts minor VersionAliases (for example, 1.9), rather than patch versions (For example, 1.9.1).
- You may enter unpublished versions through the CLI or SDK. However, these versions will not be displayed in the console and will be overwritten after the rules are edited through the console.

In the following example, for Code Editor, based on Code-OSS, Visual Studio Code - Open Source and JupyterLab, the following are being hidden for user `userProfileName` in the domain `domainId`:

- The instance types `ml.r6id.24xlarge` and `ml.r6id.32xlarge`.
- The image `sagemaker_distribution` versions `1.9` and `1.8`.

```
aws sagemaker update-user-profile \
--domain-id domainId \
--user-profile-name userProfileName \
--user-settings '{
    "StudioWebPortalSettings": {
        "HiddenInstanceTypes": [ "ml.r6id.24xlarge", "ml.r6id.32xlarge" ],
        "HiddenSageMakerImageVersionAliases": [
            {
                "SageMakerImageName": "sagemaker_distribution",
                "VersionAliases": [ "1.9", "1.8" ]
            }
        ]
    }
}'
```

Note that not all instance types and images are available in all AWS Regions.

## Multiple domains overview

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

Having multiple Amazon SageMaker AI domain simplifies managing machine learning workflows for administrators of enterprises with diverse business units, teams, or projects. Each domain acts as a logically separate environment with its own configurations, settings, and user access controls. This compartmentalization enables organizations to enforce clear boundaries between different groups, teams, or use cases, enhancing the ability to securely allocate AWS resources and permissions on a broad and granular level.

The following provides information about creating multiple domains.

- Amazon SageMaker AI supports the creation of multiple Amazon SageMaker AI domains in a single AWS Region for each account.
- Additional domains in an AWS Region have the same features and capabilities as the first domain in a Region.
- Each domain can have distinct domain settings.
- The same user profile cannot be added to multiple domains in a single Region within the same account.

For information about domain limits, see [Amazon SageMaker AI endpoints and quotas](#).

The following topics provides information on how to use tags for your domain.

## Topics

- [Automatic tag propagation](#)
- [How domain resource display filtering works](#)
- [Backfill domain tags](#)

### Automatic tag propagation

Tags allow you to categorize and label your resources based on various criteria, such as project, team, environment (For example, dev, staging, prod), or any other custom metadata. You can tag resources by your domain automatically when they are created within your domain. This makes it easier to identify and manage your resources across your domains. You can also use these tags for cost allocation using AWS Billing and Cost Management. For more information, see [Using AWS cost allocation tags](#).

By default, any SageMaker AI resources that support tagging and are created from within the Amazon SageMaker Studio or Amazon SageMaker Studio Classic UI after 11/30/2022 are

automatically tagged with a domain ARN tag. The domain ARN tag is based on the domain ID of the domain that the resource is created in.

To backfill your SageMaker AI resources, you can add the `sagemaker:domain-arn` tag to untagged resources by following the steps in [Backfill domain tags](#).

The following list describes the only SageMaker AI resources that *do not* support automatic tag propagation, as well as the impacted API calls where the tag is not returned because it was not automatically set.

 **Note**

All SageMaker List APIs do not support tag-based resource isolation.

The default app, which manages the Studio UI, is not automatically tagged.

SageMaker AI resource	Affected API calls
ImageVersionArn	<ul style="list-style-type: none"><li>• <a href="#">describe-image-version</a></li><li>• <a href="#">update-image-version</a></li><li>• <a href="#">delete-image-version</a></li></ul>
ModelCardExportJobArn	<a href="#">describe-model-card-export-job</a>
ModelPackageArn	<a href="#">describe-model-package</a>

## How domain resource display filtering works

Amazon SageMaker AI automatically filters the resources displayed in Studio or Studio Classic based on the Amazon SageMaker AI domain. This filtering is done by using the `sagemaker:domain-arn` tag attached to SageMaker AI resources. Resources created in other domains are automatically hidden.

 **Note**

This only applies to the Studio or Studio Classic UI. SageMaker AI does not support resource filtering using the AWS CLI by default.

In Amazon SageMaker Studio or Amazon SageMaker Studio Classic, you'll only see resources that:

- Were created within the current domain.
- Don't have the `sagemaker:domain-arn` tag associated with them. These untagged resources are either created outside the context of a domain or were created before 11/30/2022.

To improve resource filtering, you can add the `sagemaker:domain-arn` tag to untagged resources by following the steps in [Backfill domain tags](#).

Additionally, all resources created in shared spaces are automatically filtered to that particular shared space.

### Backfill domain tags

You can improve resource filtering by adding domain tags to untagged resources. If you have resources that are not tagged, you can backfill them.

If you have created resources in a domain before 11/30/2022, those resources are not automatically tagged with the domain Amazon Resource Name (ARN) tag.

To accurately attribute resources to their respective domain, you must add the domain tag to existing resources using the AWS CLI, as follows.

1. Map all existing SageMaker AI resources and their respective ARNs to the domains that exist in your account.
2. Run the following command from your local machine to tag the resource with the ARN of the resource's respective domain. This must be repeated for every SageMaker AI resource in your account.

```
aws resourcegroupstaggingapi tag-resources \
  --resource-arn-list arn:aws:sagemaker:region:account-id:space/domain-id/space-name \
  --tags sagemaker:domain-arn=arn:aws:sagemaker:region:account-id:domain/domain-id
```

## Isolate domain resources

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

You can isolate resources between each of the domains in your account and AWS Region using an AWS Identity and Access Management (IAM) policy. The isolated resources will no longer be accessed from other domains. In this topic we will discuss the conditions required for the IAM policy and how to apply them.

The resources that can be isolated by this policy are the resource types that have condition keys containing `aws:ResourceTag/${TagKey}` or `sagemaker:ResourceTag/${TagKey}`. For a reference on the SageMaker AI resources and associated condition keys, see [Actions, resources, and condition keys for Amazon SageMaker AI](#).

### Warning

The resource types that *do not* contain the above condition keys (and therefore the [Actions](#) that use the resource types) are *not* impacted by this resource isolation policy. For example, the [pipeline-execution](#) resource type does *not* contain the above condition keys and is *not* impacted by this policy. Therefore, the following actions, with the pipeline-execution resource type, are *not* supported for resource isolation:

- `DescribePipelineExecution`
- `StopPipelineExecution`
- `UpdatePipelineExecution`
- `RetryPipelineExecution`

- [DescribePipelineDefinitionForExecution](#)
- [ListPipelineExecutionSteps](#)
- [SendPipelineExecutionStepSuccess](#)
- [SendPipelineExecutionStepFailure](#)

The following topic shows how to create a new IAM policy that limits access to resources in the domain to user profiles with the domain tag, as well as how to attach this policy to the IAM execution role of the domain. You must repeat this process for each domain in your account. For more information about domain tags and backfilling these tags, see [Multiple domains overview](#)

## Console

The following section shows how to create a new IAM policy that limits access to resources in the domain to user profiles with the domain tag, as well as how to attach this policy to the IAM execution role of the domain, from the Amazon SageMaker AI console.

### Note

This policy only works in domains that use Amazon SageMaker Studio Classic as the default experience.

1. Create an IAM policy named `StudioDomainResourceIsolationPolicy-domain-id` with the following JSON policy document by completing the steps in [Creating IAM policies \(console\)](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "CreateAPIs",  
            "Effect": "Allow",  
            "Action": "sagemaker:Create*",  
            "NotResource": [  
                "arn:aws:sagemaker:*:*:domain/*",  
                "arn:aws:sagemaker:*:*:user-profile/*",  
                "arn:aws:sagemaker:*:*:space/*"  
            ]  
        }  
    ]  
}
```

```
},
{
    "Sid": "ResourceAccessRequireDomainTag",
    "Effect": "Allow",
    "Action": [
        "sagemaker:Update*",
        "sagemaker:Delete*",
        "sagemaker:Describe*"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/sagemaker:domain-arn": "domain-arn"
        }
    }
},
{
    "Sid": "AllowActionsThatDontSupportTagging",
    "Effect": "Allow",
    "Action": [
        "sagemaker:DescribeImageVersion",
        "sagemaker:UpdateImageVersion",
        "sagemaker:DeleteImageVersion",
        "sagemaker:DescribeModelCardExportJob",
        "sagemaker:DescribeAction"
    ],
    "Resource": "*"
},
{
    "Sid": "DeleteDefaultApp",
    "Effect": "Allow",
    "Action": "sagemaker>DeleteApp",
    "Resource": "arn:aws:sagemaker:*:*:app/domain-id/*/jupyterserver/default"
}
]
```

2. Attach the **StudioDomainResourceIsolationPolicy-*domain-id*** policy to the domain's execution role by completing the steps in [Modifying a role \(console\)](#).

## AWS CLI

The following section shows how to create a new IAM policy that limits access to resources in the domain to user profiles with the domain tag, as well as how to attach this policy to the execution role of the domain, from the AWS CLI.

### Note

This policy only works in domains that use Amazon SageMaker Studio Classic as the default experience.

1. Create a file named `StudioDomainResourceIsolationPolicy-domain-id` with the following content from your local machine.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "CreateAPIs",  
            "Effect": "Allow",  
            "Action": "sagemaker:Create*",  
            "NotResource": [  
                "arn:aws:sagemaker:*::domain/*",  
                "arn:aws:sagemaker:*::user-profile/*",  
                "arn:aws:sagemaker:*::space/*"  
            ]  
        },  
        {  
            "Sid": "ResourceAccessRequireDomainTag",  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker:Update*",  
                "sagemaker:Delete*",  
                "sagemaker:Describe*"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {  
                    "aws:ResourceTag/sagemaker:domain-arn": "domain-arn"  
                }  
            }  
        }  
    ]  
}
```

```
        },
        {
            "Sid": "AllowActionsThatDontSupportTagging",
            "Effect": "Allow",
            "Action": [
                "sagemaker:DescribeImageVersion",
                "sagemaker:UpdateImageVersion",
                "sagemaker:DeleteImageVersion",
                "sagemaker:DescribeModelCardExportJob",
                "sagemaker:DescribeAction"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DeleteDefaultApp",
            "Effect": "Allow",
            "Action": "sagemaker:DeleteApp",
            "Resource": "arn:aws:sagemaker:*:*:app/domain-id/*/jupyterserver/default"
        }
    ]
}
```

2. Create a new IAM policy using the `StudioDomainResourceIsolationPolicy-domain-id` file.

```
aws iam create-policy --policy-name StudioDomainResourceIsolationPolicy-domain-id
--policy-document file://StudioDomainResourceIsolationPolicy-domain-id
```

3. Attach the newly created policy to a new or existing role that is used as the domain's execution role.

```
aws iam attach-role-policy --policy-arn arn:aws:iam:account-id:policy/StudioDomainResourceIsolationPolicy-domain-id --role-name domain-execution-role
```

## domain default settings

With SageMaker AI, you can set default settings for your resources at the Amazon SageMaker AI domain level. These default settings are used in the creation of resources within the domain. The

following sections list default settings for domain and give information on using context keys when setting defaults.

## Topics

- [Domain default settings](#)
- [Context keys](#)

### Domain default settings

You can set the following defaults when creating or updating a domain. Values passed at the user profile and shared space level override defaults set at the domain level.

- [DefaultUserSettings](#)
- DefaultSpaceSettings

#### Note

DefaultSpaceSettings only supports the use of JupyterLab 3 image ARNs for SageMakerImageArn. For more information, see [JupyterLab Versioning](#).

```
"DefaultSpaceSettings": {  
    "ExecutionRole": "string",  
    "JupyterServerAppSettings": {  
        "DefaultResourceSpec": {  
            "InstanceType": "string",  
            "LifecycleConfigArn": "string",  
            "SageMakerImageArn": "string",  
            "SageMakerImageVersionArn": "string"  
        },  
        "LifecycleConfigArns": [ "string" ]  
    },  
    "KernelGatewayAppSettings": {  
        "CustomImages": [  
            {  
                "AppImageConfigName": "string",  
                "ImageName": "string",  
                "ImageVersionNumber": number  
            }  
        ],  
    },  
}
```

```
        "DefaultResourceSpec": {  
            "InstanceType": "string",  
            "LifecycleConfigArn": "string",  
            "SageMakerImageArn": "string",  
            "SageMakerImageVersionArn": "string"  
        },  
        "LifecycleConfigArns": [ "string" ]  
    },  
    "SecurityGroups": [ "string" ]  
}
```

## Context keys

You can add context keys to the IAM policy that creates a domain. This restricts the values that users can pass for those fields. The following list shows the context keys that domain supports and where they're implemented.

- `sagemaker:ImageArns`
  - **Implemented as part of DefaultUserSettings:SagemakerImageArn** in `DefaultUserSettings.JupyterServerAppSettings` and `DefaultUserSettings.KernelGatewayAppSettings.CustomImages` in `DefaultUserSettings.KernelGatewayAppSettings`.
  - **Implemented as part of DefaultSpaceSettings:SagemakerImageArn** in `DefaultSpaceSettings.JupyterServerAppSettings` and `DefaultSpaceSettings.KernelGatewayAppSettings.CustomImages` in `DefaultSpaceSettings.KernelGatewayAppSettings`.
- `sagemaker:VpcSecurityGroupIds`
  - **Implemented as part of DefaultUserSettings:SecurityGroups** in `DefaultUserSettings`.
  - **Implemented as part of DefaultSpaceSettings:SecurityGroups** in `DefaultSpaceSettings`.
- `sagemaker:DomainSharingOutputKmsKey`  
**Implemented as part of DefaultUserSettings:S3KmsKeyId** in `DefaultSpaceSettings.SharingSettings`.

You cannot restrict users to passing incompatible values when using context keys for the defaults. For example, the values for `SageMakerImageArn` set as part of `DefaultUserSettings` and `DefaultSpaceSettings` must be compatible. You cannot set incompatible default values.

## Custom tag propagation

Amazon SageMaker AI supports the ability to propagate custom tags set at the domain, user profile, and space level to all of the SageMaker AI resources created in the context of Amazon SageMaker Studio, JupyterLab, Code Editor, based on Code-OSS, Visual Studio Code - Open Source, and Amazon SageMaker Canvas. With custom tag propagation, users can propagate their own custom tags to resources to improve cost tracking and tie resources to specific projects and teams.

To activate this feature, use the `TagPropagation` attribute in the [CreateDomain](#) and [UpdateDomain](#) APIs. Custom tag propagation can only be set at the domain level, which means that all users and spaces in a domain use the feature when it is activated. It is not possible to modify custom tag propagation settings at the user profile or space level. For more information about using custom tag propagation, see [Add custom tags to resources](#).

 **Note**

System tags added by AWS services on a domain, user profile, and space are not propagated.

## Example use cases

Custom tag propagation is particularly useful for the following use cases.

- Track cost across all of the SageMaker AI resources created in Amazon SageMaker Studio.
- Track cost for SageMaker AI resources that are created in Amazon SageMaker Canvas. This includes models deployed on a SageMaker AI endpoint.
- Track cost incurred for an Amazon DataZone project by propagating the Amazon DataZone project ID to all the resources created by Amazon SageMaker Studio.

## Tag merging

With custom tag propagation activated, resources created at the user profile and space level take on the tags specified at the domain level, as well as those specified during user profile or space creation.

SageMaker AI resources have a 50 tag limit. If the number of tags added to a resource exceeds 50, SageMaker AI returns an error during resource creation. We recommend limiting the number of tags to avoid this. For example, assume a user has 25 tags for their domain and 30 tags for their user profile. When the user creates a resource, a total of 55 tags propagate to the resource. Because the aggregate tag total exceeds 50, resource creation fails until the user removes at least 5 tags.

### Note

By default, SageMaker AI automatically adds the `sagemaker:user-profile-arn`, `sagemaker:domain-arn`, or `sagemaker:space-arn` tag to SageMaker AI resources.

SageMaker AI adds the ARN tag regardless of whether or not the domain is using custom tag propagation. These ARN tags also contribute toward the 50 tag limit.

## Add custom tags to resources

The following page demonstrates the steps needed to use custom tag propagation. Custom tag propagation requires the following steps:

- Opt-in to custom tag propagation
- Add custom tags to resources

When you activate custom tag propagation in an existing domain, tag propagation does not work for existing applications until the application is restarted. Similarly, tags are not updated on an existing resource when new custom tags are added. For example, assume a domain has two tags and a user creates a resource in that domain. The resource then has two tags. If a new tag is added to the domain, then that new tag is not added to the existing resource. However, any new resource created will have the new tag attached to the resource.

## Prerequisites

- Users must have the `sagemaker:AddTags` permission for any resource creation.
  - For new domains created with the `SageMakerFullAccess` managed policy or using the SageMaker Role Manager, the `sagemaker:AddTags` permission is pre-populated.
  - For existing domains using custom AWS Identity and Access Management policies, you must update the policies to include the `sagemaker:AddTags` permission to allow users to create resources.

## Opt-in to custom tag propagation

The process to opt-in to custom tag propagation differs based on if you are opting-in from the console or from the AWS CLI. From the console, you can only opt-in to custom tag propagation by updating an existing domain. From the AWS CLI, you can opt-in to custom tag propagation when creating a domain or updating an existing domain.

### Opt-in from the console

The following steps outline how to opt-in to custom tag propagation from the console. You can only opt-in to custom tag propagation from the console by updating an existing domain.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation, select **Admin configurations**. Under **Admin configurations**, select **Domains**.
3. On the **Domains** page, select the domain that you want to activate custom tag propagation for.
4. From the **Domain details** page, select the **Domain settings** tab.
5. On the **Domain settings** tab, navigate to **Custom Tag Propagation**.
6. Select **Edit**.
7. From the **Edit custom tag propagation** page, select **Automatically propagate custom tags**
8. Select **Submit**.

### Opt-in using the AWS CLI

To opt-in to custom tag propagation using the AWS CLI, use the `TagPropagation` attribute in the [CreateDomain](#) and [UpdateDomain](#) APIs. By default, the value of this field is `DISABLED`. An empty value also defaults to `DISABLED`. The following example shows how to activate custom tag propagation.

```
aws sagemaker update-domain \
--domain-id domain-id \
--region region \
--tag-propagation ENABLED
```

## Add custom tags

The process to add custom tags propagation differs based on if you are adding them from the console or from the AWS CLI.

### Add from the console

The following steps outline how to add custom tags to a domain from the console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation, select **Admin configurations**. Under **Admin configurations**, select **Domains**.
3. On the **Domains** page, select the domain that you want to add custom tags to.
4. From the **Domain details** page, select the **Domain settings** tab.
5. On the **Domain settings** tab, navigate to **Tags**.
6. Select **Edit**.
7. From the **Tags** page, select **Add tag**. Add a key and value pair for the custom tag.
8. Select **Save**. This custom tag is now propagated to the SageMaker AI resources created in the domain.

The following steps outline how to add custom tags to a user profile from the console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation, select **Admin configurations**. Under **Admin configurations**, select **Domains**.
3. On the **Domains** page, select the domain containing the user profile that you want to add custom tags to.
4. From the **Domain details** page, select the **User profiles** tab.
5. On the **User profiles** tab, select the user profile you want to add custom tags to.
6. On the **User Details** tab, navigate to the **Details** section.
7. Select **Edit**.
8. From the **Tags** section, select **Add tag**. Add a key and value pair for the custom tag.
9. Select **Submit**. This custom tag is now propagated to the SageMaker AI resources created in the domain.

## Add using the AWS CLI

After you have activated custom tag propagation, you can add custom tags using the AWS CLI at the domain, user profile, or space level during creation or update. The method to add custom tags differs depending on you are creating a new resource or adding tags to an existing resource.

The following example shows how to add custom tags at the domain level during creation.

```
aws sagemaker create-domain \
  --domain-name domain-id \
  --auth-mode IAM \
  --default-user-settings '{"ExecutionRole": "execution-role"}' \
  --subnet-ids subnet-id \
  --vpc-id vpc-id \
  --tags Key=key,Value=value \
  --tag-propagation ENABLED
```

You must use the [AddTags](#) API to add custom tags for existing domain, user profile, and spaces as follows.

```
aws sagemaker add-tags \
  --resource-arn resource-arn-to-attach-tags \
  --tags Key=key, Value=value
```

## Opt-out of custom tag propagation

The process to opt-out of custom tag propagation differs based on if you are opting-out from the console or from the AWS CLI.

### Opt-out from the console

The following steps outline how to opt-out of custom tag propagation from the console. You can only opt-out of custom tag propagation from the console by updating an existing domain.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation, select **Admin configurations**. Under **Admin configurations**, select **Domains**.
3. On the **Domains** page, select the domain that you want to opt-out of custom tag propagation for.
4. From the **Domain details** page, select the **Domain settings** tab.

5. On the **Domain settings** tab, navigate to **Custom Tag Propagation**.
6. Select **Edit**.
7. From the **Edit custom tag propagation** page, select **Automatically propagate custom tags**
8. Select **Submit**.

## Opt-out using the AWS CLI

To opt-out of custom tag propagation, set the `TagPropagation` attribute in the [CreateDomain](#) and [UpdateDomain](#) APIs to `DISABLED` as shown in the following example. By default, the value of this field is `DISABLED`. An empty value also defaults to `DISABLED`.

### Note

Tag propagation is not automatically turned off for existing applications when `TagPropagation` is set to `DISABLED`. Applications must be restarted for opt-out to take effect for existing apps.

```
aws sagemaker update-domain \
--domain-id domain-id \
--region region \
--tag-propagation DISABLED
```

## Adding a custom file system to a domain

When you create a domain, Amazon SageMaker AI adds a default Amazon Elastic File System (Amazon EFS) volume to the domain. SageMaker AI creates this volume for you. You also have the option to add a custom Amazon EFS or a custom Amazon FSx for Lustre file system that you've created. After you add it, your file system is available to users who belong to your domain. Your users can access the file system when they use Amazon SageMaker Studio. They can attach the file system to spaces that they create for the following supported applications:

- JupyterLab
- Code Editor

After running a space and starting the application, your users can access any data, code, or other artifacts that your file system contains.

You can enable your users to access your file system in the following ways:

- Through *shared spaces* – A shared space can be created by any user who belongs to your domain. Then, it can be used by any user who belongs to your domain.
- Through *private spaces* – A private space can be created by any user who belongs to your domain. Then, it can be used by only that user.
- Exclusively as an individual user – If you don't want to enable all of your users to access the file system, you can enable only a specific user to access it. If you do that, the file system is available only in private spaces that the specific user creates.

You can add a custom file system by using the Amazon SageMaker API, the AWS SDKs, or the AWS CLI. You can't add a custom file system by using the SageMaker AI console.

## Prerequisites

Before you can add a custom file system to a domain, you must meet the following requirements:

- You have a domain in SageMaker AI. Before you can add a file system, you need the domain ID. You can look up the ID by using the SageMaker AI console. You can also run the [list-domains](#) command with the AWS CLI.
- You have an Amazon EFS or FSx for Lustre file system in your AWS account.

### For Amazon EFS

- For the steps to create an Amazon EFS, see [Create your Amazon EFS file system](#) in the *Amazon Elastic File System User Guide*.
- Before Studio can access your file system, it must have a mount target in each of the subnets that you associate with the domain. For more information about assigning mount targets to subnets, see [Creating and managing mount targets and security groups](#) in the *Amazon Elastic File System User Guide*.
- For each mount target, you must add the security group that Amazon SageMaker AI created in your AWS account when you created the domain. The security group name has the format `security-group-for-inbound-nfs-domain-id`. For instructions on how to obtain your domain ID, see [View domains](#).
- Your IAM permissions must allow you to use the `elasticfilesystem:DescribeMountTargets` action. For more information about this action, see [Actions, resources, and condition keys for Amazon Elastic File System](#) in the *Service Authorization Reference*.

## For FSx for Lustre

- For the steps to create a FSx for Lustre file system, see [Getting started with Amazon FSx for Lustre](#) in the *Amazon FSx for Lustre User Guide*. Ensure that the FSx for Lustre file system exists in:
  - The same Amazon VPC as your domain.
  - One of the subnets present in your domain.
- Before Studio can access the FSx for Lustre file system, you must add your domain's security group to all of the elastic network interfaces (ENIs) in your FSx for Lustre file system. Without this step, the app creation fails with an error. Use the following instructions to add the domain security group to your FSx for Lustre file system ENIs.

### Add your domain security group to FSx for Lustre file system ENIs (console)

1. Navigate to the [Amazon FSx console](#).
2. Choose **File systems**.
3. Choose your FSx for Lustre file system by using the corresponding link under **File system ID**.
4. If not selected already, choose the **Network & security** tab.
5. Under **Subnet** choose **To see all the ENIs, see the Amazon EC2 console**. This will take you to the Amazon EC2 console and shows all of the ENIs linked to your FSx for Lustre file system.
6. For each ENI:
  - a. Choose the ENI by choosing the corresponding link under **Network interface ID**.
  - b. Choose **Actions** at the top right of the summary page to expand a drop-down menu.
  - c. In the drop-down menu, choose **Choose security group**.
  - d. Search for your domain security group.

The security group name has the format `security-group-for-inbound-nfs-domain-id`. For instructions on how to obtain your domain ID, see [View domains](#).

- e. Choose **Add security group**.

## Adding a custom file system to a domain with the AWS CLI

To add a custom file system to a domain or user profile with the AWS CLI, you pass a `CustomFileSystemConfigs` definition when you use any of the following commands:

- [create-domain](#)
- [update-domain](#)
- [create-user-profile](#)
- [update-user-profile](#)

The following examples show how to add a file system to an existing domain or user profile.

### To add a file system that is accessible in shared spaces

- Update the default space settings for your domain. The following example adds the file system settings to the default space settings:

```
aws sagemaker update-domain --domain-id domain-id \  
--default-space-settings file://file-system-settings.json
```

This example passes the file system configuration as a JSON file, which is shown in a later example.

### To add a file system that is accessible in private spaces

- Update the default user settings for your domain. The following example adds the file system settings to the default user settings:

```
aws sagemaker update-domain --domain-id domain-id \  
--default-user-settings file://file-system-settings.json
```

This example passes the file system configuration as a JSON file, which is shown in a later example.

## To add a file system that is accessible only to an individual user

- Update the user profile for the user. The following example adds the file system settings to a user profile:

```
aws sagemaker update-user-profile --domain-id domain-id \  
--user-profile-name user-profile-name \  
--user-settings file://file-system-settings.json
```

This example passes the file system configuration as a JSON file, which is shown in the following example.

### Example file system settings file

The file in the preceding examples, `file-system-settings.json`, has the following settings:

For your FSx for Lustre file systems

```
{  
    "CustomFileSystemConfigs":  
    [  
        {  
            "FSxLustreFileSystemConfig":  
            {  
                "FileSystemId": "file-system-id",  
                "FileSystemPath": "/"  
            }  
        }  
    ]  
}
```

This example configuration has the following keys:

`CustomFileSystemConfigs`

Settings for custom file systems (only Amazon EFS file systems are supported).

`FSxLustreFileSystemConfig`

Settings for custom FSx for Lustre file systems.

## FileSystemId

The ID of your Amazon EFS file system.

## FileSystemPath

The path to the file system directory that is accessible to the domain users in their spaces in Studio. Permitted users can access only this directory and below. The default path is the file system root: /.

## For your Amazon EFS file systems

```
{  
    "CustomFileSystemConfigs":  
    [  
        {  
            "EFSFileSystemConfig":  
            {  
                "FileSystemId": "file-system-id",  
                "FileSystemPath": "/"  
            }  
        }  
    ]  
}
```

This example configuration has the following keys:

### CustomFileSystemConfigs

Settings for custom file systems (only Amazon EFS file systems are supported).

#### EFSFileSystemConfig

Settings for custom Amazon EFS file systems.

## FileSystemId

The ID of your Amazon EFS file system.

## FileSystemPath

The path to the file system directory that is accessible to the domain users in their spaces in Studio. Permitted users can access only this directory and below. The default path is the file system root: /.

When you assign a file system to the default space settings for a domain, you must also include the execution role in the settings:

```
{  
    "ExecutionRole": "execution-role-arn"  
}
```

This example configuration has the following key:

#### ExecutionRole

The default execution role for the users of the domain.

If you want to apply POSIX permissions for your file system, you can also pass the following settings to the `create-domain` or `create-user-profile` commands:

```
{  
    "CustomPosixUserConfig":  
    {  
        "Uid": UID,  
        "Gid": GID  
    }  
}
```

This example configuration has the following keys:

#### CustomPosixUserConfig

The default POSIX identities that are used for file system operations. You can use these settings to apply your existing POSIX permission structure to the user profiles that access the custom file system. At a POSIX permissions level, you can control which users can access the file system and which files or data they can access.

You can also apply `CustomPosixUserConfig` settings when you create a user profile by using the `create-user-profile` command. The settings that you apply to a user profile override those that you apply to the associated domain.

**Note**

You can apply `CustomPosixUserConfig` settings when you use the `create-domain` and `create-user-profile` commands. However, you can't apply these settings when you do the following:

- Use the `update-domain` command for a domain that is already associated with any user profiles. You can apply these settings only to domains that have no user profiles.
- Use the `update-user-profile` command. To apply these settings to profile that you've already created, delete the profile, and create a new one that has the updated settings.

**Uid**

The POSIX user ID. The default is 200001.

**Gid**

The POSIX group ID. The default is 1001.

## Attaching a custom file system to a space with the AWS CLI

After you add a custom file system to a domain, the domain users can attach the file system to spaces that they create. For instance, they can attach the file system when they use Studio or the [create-space](#) command with the AWS CLI.

### To attach a custom file system to a space

- Add the file system configuration to the space settings. The following example command attaches a file system to a new space.

```
aws sagemaker create-space \
--space-name space-name \
--domain-id domain-id \
--ownership-settings "OwnerUserProfileName=user-profile-name" \
--space-sharing-settings "SharingType=Private" \
--space-settings file://space-settings.json
```

In this example, the file space-settings.json has the following settings, which include the CustomFileSystems configuration with the FileSystemId key.

### For your FSx for Lustre file systems

```
{  
    "AppType": "JupyterLab",  
    "JupyterLabAppSettings":  
    {  
        "DefaultResourceSpec":  
        {  
            "InstanceType": "instance-type"  
        }  
    },  
    "CustomFileSystems":  
    [  
        {  
            "FSxLustreFileSystem":  
            {  
                "FileSystemId": "file-system-id"  
            }  
        }  
    ]  
}
```

### For your Amazon EFS file systems

```
{  
    "AppType": "JupyterLab",  
    "JupyterLabAppSettings":  
    {  
        "DefaultResourceSpec":  
        {  
            "InstanceType": "instance-type"  
        }  
    },  
    "CustomFileSystems":  
    [  
        {  
            "EFSFileSystem":  
            {  
                "FileSystemId": "file-system-id"  
            }  
        }  
    ]  
}
```

```
        }  
    }  
}
```

SageMaker AI creates a symbolic link at the following path: /home/sagemaker-user/custom-file-systems/*file-system-type*/*file-system-id*. With this, the domain users can navigate to the custom file system from within their home directory, /home/sagemaker-user.

## View domain environment details

This page gives information about modifications to the Amazon SageMaker AI domain environment. Complete the following procedure to view the custom images, lifecycle configurations, and git repositories attached to a domain environment.

### Open the Environment page

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select a domain to open the **Environment** page.
5. On the **domain details** page, choose the **Environment** tab.

For more information about bringing a custom Amazon SageMaker Studio Classic image, see [Bring your own SageMaker image](#).

For more information about bringing a custom RStudio image, see [Bring your own image to RStudio on SageMaker](#).

For instructions on using a lifecycle configuration with Studio Classic, see [Use Lifecycle Configurations with Amazon SageMaker Studio](#).

For information about attaching a git repository to a domain, see [Attach Suggested Git Repos to SageMaker AI](#).

These can also be attached to a shared space using the AWS CLI by passing values to the [create-space](#) command using the space-settings parameter.

## View domains

The following section shows how to view a list of your domains, and details of an individual domain from the SageMaker AI console or the AWS CLI.

### Console

The console's domain overview page gives information about the structure of a domain, and it provides a list of your domains. The page's domain structure diagram describes domain components and how they interact with each other.

The following procedure shows how to view a list of your domains from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.

To view the details of the domain, complete the following procedure. This page gives information about the general settings for the domain, including the name, domain ID, execution role used to create the domain, and the authentication method of the domain.

1. From the list of domains, select the domain for which you want to open the **domain settings** page.
2. On the **domain details** page, choose the **domain settings** tab.

### AWS CLI

Run the following command from the terminal of your local machine to view a list of domains from the AWS CLI.

```
aws sagemaker list-domains --region region
```

## Edit domain settings

You can edit the settings of a domain from the SageMaker AI console or the AWS CLI. The following considerations apply when updating the settings of a domain.

- If `DefaultUserSettings` and `DefaultSpaceSettings` are set, they cannot be unset.

- `DefaultUserSettings.ExecutionRole` can only be updated if there are no applications running in any user profile within the domain. This value cannot be unset.
- `DefaultSpaceSettings.ExecutionRole` can only be updated if there are no applications running in any of shared spaces within the domain. This value cannot be unset.
- If the domain was created in **VPC only** mode, SageMaker AI automatically applies updates to the security group settings defined for the domain to all shared spaces created in the domain.
- `DomainId` and `DomainName` cannot be edited.

The following section shows how to edit domain settings from the SageMaker AI console or the AWS CLI.

## Console

You can edit the domain from the SageMaker AI console using the following procedure.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain for which you want to open the **domain settings** page.
5. On the **domain details** page, you can configure and manage your domain details by choosing the appropriate tab.
6. To configure the general settings, on the **domain details** page choose the **domain settings** tab then choose **Edit**.

## AWS CLI

Run the following command from the terminal of your local machine to update a domain from the AWS CLI. For more information about the structure of `default-user-settings`, see [CreateDomain](#).

```
aws sagemaker update-domain \
--domain-id domain-id \
--default-user-settings default-user-settings \
--default-space-settings default-space-settings \
--domain-settings-for-update settings-for-update \
--region region
```

## Delete an Amazon SageMaker AI domain

This page explains how to delete a domain and the requirements needed. A domain consists of a list of authorized users, configuration settings, and an Amazon Elastic File System (Amazon EFS) volume. The Amazon EFS volume contains data for the users, including notebooks, resources, and artifacts. A user can have multiple applications (apps) which support the reading and execution experience of the user's notebooks, terminals, and consoles. You can delete your domain using one of the following:

- AWS console
- AWS Command Line Interface (AWS CLI)
- SageMaker SDK

### Requirements

You must satisfy the following requirements to delete a domain.

- You must have admin permission to delete a domain.
- You can only delete an app with the status **InService** displayed as **Ready** in the domain. To delete the containing domain, you don't need to delete an app whose status is **Failed**. In the domain, an attempt to delete an app in the failed state results in an error.
- To delete a domain, the domain cannot contain any user profiles or shared spaces. To delete a user profile or shared space, the user profile or space cannot contain any non-failed apps.

When you delete these resources, the following occurs:

- App – The data (files and notebooks) in a user's home directory is saved. Unsaved notebook data is lost.
- User profile – The user can no longer sign in to the domain. The user loses access to their home directory, but the data is not deleted. An admin can retrieve the data from the Amazon EFS volume where it is stored under the user's AWS account.
- To switch authentication modes from IAM to IAM Identity Center, you must delete the domain.

### EFS files

Your files are kept in an Amazon EFS volume as a backup. This backup includes the files in the mounted directory, which is `/home/sagemaker-user` for Amazon SageMaker Studio Classic and `/root` for kernels.

When you delete files from these mounted directories, the kernel or app may move the deleted files into a hidden trash folder. If the trash folder is inside the mounted directory, those files are copied into the Amazon EFS volume and will incur charges. To avoid these Amazon EFS charges, you must identify and clean the trash folder location. The trash folder location for default apps and kernels is `~/.local/`. This may vary depending on the Linux distribution used for custom apps or kernels. For more information about the Amazon EFS volume, see [Manage Your Amazon EFS Storage Volume in SageMaker Studio Classic](#).

When you use the SageMaker AI console to delete the domain, the Amazon EFS volume is detached but not deleted. The same behavior occurs by default when you use the AWS CLI or the SageMaker Python SDK to delete the domain. However, when you use the AWS CLI or the SageMaker Python SDK, you can set the `RetentionPolicy` to `HomeEfsFileSystem=Delete`. This deletes the Amazon EFS volume along with the domain.

## Delete an Amazon SageMaker AI domain (console)

### Important

When a user, space, or domain is deleted, the Amazon EFS volume that contains the corresponding data will be lost. This includes notebooks and other artifacts.

### To delete a domain

1. Open the [SageMaker AI console](#).
2. On the left navigation pane, choose **Admin configurations** to expand the options, if not already expanded.
3. Under **Admin configurations**, choose **Domains**.
4. Select the domain name link that you want to delete.
5. Choose the **User profiles** tab.
6. Repeat the following steps for each user in the **User profiles** list.
  - a. Choose the user name link.
  - b. If not already selected, choose the **User Details** tab
  - c. Find any apps and spaces and choose **Delete** under the corresponding **Action** column.
  - d. Follow the delete instructions.

- e. Once all of the app and spaces have **Status** as **Deleted**, choose **Delete** at the top right of the page.
  - f. Follow the delete instructions.
7. When all users are deleted, choose the **Space management** tab.
  8. Repeat the following steps for each space in the **Spaces** list.
    - a. Select the bubble corresponding to the space.
    - b. Choose **Delete**.
    - c. Follow the delete instructions.
  9. When all users and spaces are deleted, choose the **Domain settings** tab.
  10. Find the **Delete domain** section.
  11. Choose **Delete domain**. If this button is not available, you must repeat the previous steps to delete all spaces and users.
  12. Follow the delete instructions.

## Delete an Amazon SageMaker AI domain (AWS CLI)

### To delete a domain

1. Retrieve the list of domains in your account.

```
aws --region Region sagemaker list-domains
```

2. Retrieve the list of applications for the domain to be deleted.

```
aws --region Region sagemaker list-apps \  
--domain-id-equals DomainID
```

3. Delete each application in the list.

```
aws --region Region sagemaker delete-app \  
--domain-id DomainId \  
--app-name AppName \  
--app-type AppType \  
--user-profile-name UserProfileName
```

4. Retrieve the list of user profiles in the domain.

```
aws --region Region sagemaker list-user-profiles \  
--domain-id-equals DomainID
```

5. Delete each user profile in the list.

```
aws --region Region sagemaker delete-user-profile \  
--domain-id DomainID \  
--user-profile-name UserProfileName
```

6. Retrieve the list of shared spaces in the domain.

```
aws --region Region sagemaker list-spaces \  
--domain-id DomainID
```

7. Delete each shared space in the list.

```
aws --region Region sagemaker delete-space \  
--domain-id DomainID \  
--space-name SpaceName
```

8. Delete the domain. To also delete the Amazon EFS volume, specify HomeEfsFileSystem=Delete.

```
aws --region Region sagemaker delete-domain \  
--domain-id DomainID \  
--retention-policy HomeEfsFileSystem=Retain
```

## Domain user profiles

A user profile represents a single user within an Amazon SageMaker AI domain. The user profile is the main way to reference a user for the purposes of sharing, reporting, and other user-oriented features. This entity is created when a user onboards to the Amazon SageMaker AI domain. A user profile can have (at most) a single JupyterServer application outside the context of a shared space. The user profile's Studio Classic application is directly associated with the user profile and has an isolated Amazon EFS directory, an execution role associated with the user profile, and Kernel Gateway applications. A user profile can also create other applications from the console or from Amazon SageMaker Studio.

## Topics

- [Add user profiles](#)
- [Remove user profiles](#)
- [View user profiles in a domain](#)
- [View user profile details](#)

## Add user profiles

The following section shows how to add user profiles to a domain using the SageMaker AI console or the AWS CLI.

After you add a user profile to the domain, users can login using a URL. If the domain uses AWS IAM Identity Center for authentication, users receive an email that contains the URL to sign in to the domain. If the domain uses AWS Identity and Access Management, you can create a URL for a user profile using [CreatePresignedDomainUrl](#)

### Add user profiles from the console

You can add user profiles to a domain from the SageMaker AI console by following this procedure.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to add a user profile to.
5. On the **domain details** page, choose the **User profiles** tab.
6. Choose **Add user**. This opens a new page.
7. Use the default name for your user profile or add a custom name.
8. For **Execution role**, choose an option from the role selector. If you choose **Enter a custom IAM role ARN**, the role must have, at a minimum, an attached trust policy that grants SageMaker AI permission to assume the role. For more information, see [SageMaker AI Roles](#).

If you choose **Create a new role**, the **Create an IAM role** dialog box opens:

- a. For **S3 buckets you specify**, specify additional Amazon S3 buckets that users of your notebooks can access. If you don't want to add access to more buckets, choose **None**.
- b. Choose **Create role**. SageMaker AI creates a new IAM role, `AmazonSageMaker-ExecutionPolicy`, with the [AmazonSageMakerFullAccess](#) policy attached.

9. (Optional) Add tags to the user profile. All resources that the user profile creates will have a domain ARN tag and a user profile ARN tag. The domain ARN tag is based on domain ID, while the user profile ARN tag is based on the user profile name.
10. Choose **Next**.
11. In the **SageMaker Studio** section, you have the option to choose between the newer and classic version of Studio as your default experience.
  - If you choose **SageMaker Studio** (recommended) as your default experience, the Studio Classic IDE has default settings. For information on the default settings, see [Default settings](#).  
For information on Studio, see [Amazon SageMaker Studio](#).
  - If you choose **Studio Classic** as your default experience, you can choose to enable or disable notebook resource sharing. Notebook resources include artifacts such as cell output and Git repositories. For more information on Notebook resources, see [Share and Use an Amazon SageMaker Studio Classic Notebook](#).
12. Under **SageMaker Canvas**, you can configure your SageMaker Canvas settings. For the instructions and configuration details for onboarding, see [Getting started with using Amazon SageMaker Canvas](#).
  - For the **Canvas base permissions configuration**, select whether to establish the minimum required permissions to use the SageMaker Canvas application.
13. Under **RStudio**, if RStudio license, select whether you want to create the user with one of the following authorizations:
  - Unauthorized
  - RStudio Admin
  - RStudio User
14. Choose **Next**.
15. In the **Customize Studio UI** page you can customize the viewable applications and machine learning (ML) tools displayed in Studio. This customization only hides the applications and ML tools in the left navigation pane in Studio. For information on the Studio UI, see [Amazon SageMaker Studio UI overview](#).

For information about the applications, see [Applications supported in Amazon SageMaker Studio](#).

The customize Studio UI feature is not available in Studio Classic. If you wish to set Studio as your default experience, choose **Previous** and to return to the previous step.

16. Choose **Next**.

17. After you have reviewed your changes, choose **Create user profile**.

## Create user profiles from the AWS CLI

To create a user profile in a domain from the AWS CLI, run the following command from the terminal of your local machine. For information about the available JupyterLab version ARNs, see [Setting a default JupyterLab version](#).

```
aws --region region \
sagemaker create-user-profile \
--domain-id domain-id \
--user-profile-name user-name \
--user-settings '{
    "JupyterServerAppSettings": {
        "DefaultResourceSpec": {
            "SageMakerImageArn": "sagemaker-image-arn",
            "InstanceType": "system"
        }
    }
}'
```

You can use the AWS CLI to customize the applications and ML tools displayed in Studio for the user, using [StudioWebPortalSettings](#). Use `HiddenAppTypes` to hide applications and `HiddenMLTools` to hide ML tools. For more information on customizing the left navigation of the Studio UI, see [Hide machine learning tools and applications in the Amazon SageMaker Studio UI](#). This feature is not available for Studio Classic.

## Remove user profiles

All apps launched by a user profile must be deleted to delete the user profile. The following section shows how to remove user profiles from a domain using the SageMaker AI console or AWS CLI.

### Remove user profiles from the console

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.

3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to remove a user profile from.
5. On the **domain details** page, choose the **User profiles** tab.
6. Select the user profile that you want to delete.
7. On the **User Details** page, for each non-failed app in the **Apps** list, choose **Action**.
8. From the dropdown list, choose **Delete**.
9. On the **Delete app** dialog box, choose **Yes, delete app**. Then enter *delete* in the confirmation field, and choose **Delete**.
10. When **Status** shows as **Deleted** for all apps, choose **Edit**.
11. On the **Edit User** page, choose **Delete user**.
12. On the **Delete user** pop-up, choose **Yes, delete user**.
13. Enter *delete* in the field to confirm deletion.
14. Choose **Delete**.

## Remove user profiles from the AWS CLI

To delete a user profile from the AWS CLI, run the following command from the terminal of your local machine.

```
aws sagemaker delete-user-profile \
--region region \
--domain-id domain-id \
--user-profile-name user-name
```

## View user profiles in a domain

The following section describes how to view a list of user profiles in a domain from the SageMaker AI console or the AWS CLI.

### View user profiles from the console

Complete the following procedure to view a list of user profiles in the domain from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.

3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to view a list of user profiles for.
5. On the **domain details** page, choose the **User profiles** tab.

## View user profiles from the AWS CLI

To view the user profiles in a domain from the AWS CLI, run the following command from the terminal of your local machine.

```
aws sagemaker list-user-profiles \
--region region \
--domain-id domain-id
```

## View user profile details

The following section describes how to view the details of a user profile from the SageMaker AI console or the AWS CLI.

### View user profile details from the console

Complete the following procedure to view the details of a user profile from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to view a list of user profiles for.
5. On the **domain details** page, choose the **User profiles** tab.
6. Select the user profile that you want to view details for.

### View user profile details from the AWS CLI

To describe a user profile from the AWS CLI, run the following command from the terminal of your local machine.

```
aws sagemaker describe-user-profile \
```

```
--region region \
--domain-id domain-id \
--user-profile-name user-name
```

## IAM Identity Center groups in a domain

AWS IAM Identity Center is the recommended AWS service for managing human user access to AWS resources. It is a single place where you can assign your users consistent access to multiple AWS accounts and applications. For more information about IAM Identity Center authentication, see [What is IAM Identity Center?](#).

If you use AWS IAM Identity Center authentication for your Amazon SageMaker AI domain, you can use the following topics to learn how to view, add, and remove IAM Identity Center groups and users to a domain.

### Topics

- [View groups and users](#)
- [Add groups and users](#)
- [Remove groups](#)

### View groups and users

Complete the following procedure to view a list of IAM Identity Center groups and users from the Amazon SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to open the **domain settings** page for.
5. On the **domain details** page, choose the **Groups** tab.

### Add groups and users

The following sections show how to add groups and users to a domain from the SageMaker AI console or AWS CLI.

**Note**

If the domain was created before October 1st, 2023, you can only add groups and users to the domain from the SageMaker AI console.

## SageMaker AI console

Complete the following procedure to add groups and users to your domain from the SageMaker AI console.

1. On the **Groups** tab, choose **Assign users and groups**.
2. On the **Assign users and groups** page, select the users and groups that you want to add.
3. Choose **Assign users and groups**.

## AWS CLI

Complete the following procedure to add groups and users to your domain from the AWS CLI.

1. Fetch the SingleSignOnApplicationArn of the domain with a call to [describe-domain](#). SingleSignOnApplicationArn is the ARN of the application managed in IAM Identity Center.

```
aws sagemaker describe-domain \
--region region \
--domain-id domain-id
```

2. Associate the user or group with the domain. To accomplish this, pass the SingleSignOnApplicationArn value returned from the [describe-domain](#) command as the application-arn parameter in a call to [create-application-assignment](#). You must also pass the type and ID of the entity to associate.

```
aws sso-admin create-application-assignment \
--application-arn application-arn \
--principal-id principal-id \
--principal-type principal-type
```

## Remove groups

Complete the following procedure to remove groups from your domain from the SageMaker AI console. For information about deleting a user, see [Remove user profiles](#).

1. On the **Groups** tab, choose the group that you want to remove.
2. Choose **Unassign groups**.
3. On the pop-up window, choose **Yes, unassign groups**.
4. Enter *unassign* in the field.
5. Choose **Unassign groups**.

## Understanding domain space permissions and execution roles

For many SageMaker AI applications, when you start up a SageMaker AI application within a domain, a space is created for the application. When a user profile creates a space, that space assumes an AWS Identity and Access Management (IAM) role that defines the permissions granted to that space. The following page gives information about space types and the execution roles that define permissions for the space.

An [IAM role](#) is an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user in that it is an AWS identity with permissions policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session.

### Note

When you start up Amazon SageMaker Canvas or RStudio, it does not create a space that assumes an IAM role. Instead, you change the role associated with the user profile to manage their permissions for the application. For information on obtaining a SageMaker AI user profile's role, see [Get user execution role](#).

For SageMaker Canvas, see [Amazon SageMaker Canvas setup and permissions management \(for IT administrators\)](#).

For RStudio, see [Create Amazon SageMaker AI domain with RStudio App](#).

Users can access their SageMaker AI applications within a shared or private space.

## Shared spaces

- There can only be one space associated with an application. A shared space can be accessed by all of the user profiles within the domain. This grants all user profiles in the domain access to the same underlying file storage system for the application.
- The shared space will be granted the permissions defined by the **space default execution role**. If you wish to modify the shared space's execution role, you must modify the space default execution role.

For information on obtaining the space default execution role, see [Get space execution role](#).

For information on modifying your execution role, see [Modify permissions to execution role](#).

- For information about shared spaces, see [Collaboration with shared spaces](#).
- To create a shared space, see [Create a shared space](#).

## Private spaces

- There can only be one space associated with an application. A private space can only be accessed by the user profile who created it. This space cannot be shared with other users.
- The private space will assume the **user profile execution role** of the user profile that created it. If you wish to modify the private space's execution role, you must modify the user profile's execution role.

For information on obtaining the user profile's execution role, see [Get user execution role](#).

For information on modifying your execution role, see [Modify permissions to execution role](#).

- All applications that support spaces also support private spaces.
- A private space for Studio Classic is already created for each user profile by default.

## Topics

- [SageMaker AI execution roles](#)
- [Example of flexible permissions with execution roles](#)

## SageMaker AI execution roles

A SageMaker AI execution role is an [AWS Identity and Access Management \(IAM\) role](#) that is assigned to an IAM identity that is performing executions in SageMaker AI. An [IAM identity](#) provides access to an AWS account and represents a human user or programmatic workload that can be authenticated and then authorized to perform actions in AWS, that grants permissions to SageMaker AI to access other AWS resources on your behalf. This role allows SageMaker AI to perform actions like launching compute instances, accessing data and model artifacts stored in Amazon S3, or writing logs to CloudWatch. SageMaker AI assumes the execution role at runtime and is temporarily granted the permissions defined in the role's policy. The role should contain the necessary permissions that define the actions the identity can perform and resources the identity has access to. You can assign roles to various identities to provide a flexible and granular approach to managing permissions and access within your domain. For more information on domains, see [Amazon SageMaker AI domain overview](#). For example, you can assign IAM roles to the:

- **Domain execution role** to grant broad permissions to all of the user profiles within the domain.
- **Space execution role** to grant broad permissions for a shared spaces within the domain. All user profiles in the domain can access shared spaces and will use the space's execution role while within the shared space.
- **User profile execution role** to grant fine-grained permissions for specific user profiles. A private space created by a user profile will assume that user profile's execution role.

This enables you to grant the necessary permissions to the domain while still maintaining the principle of least-privilege permissions for user profiles, to adhere to the [security best practices in IAM](#) in the [AWS IAM Identity Center User Guide](#).

Any changes or modifications to the execution roles may take a few minutes to propagate. For more information, see [Change your execution role](#) or [Modify permissions to execution role](#), respectively.

### Example of flexible permissions with execution roles

With [IAM roles](#) you can manage and grant permissions on broad and granular levels. The following example includes granting permissions on a space-level and a user-level.

Suppose you are an administrator setting up a domain for a team of data scientists. You can allow the user profiles within the domain to have full access to Amazon Simple Storage Service (Amazon S3) buckets, run SageMaker training jobs, and deploy models using an application in a *shared space*.

In this example, you can create an IAM role called "DataScienceTeamRole" with broad permissions. Then you can assign "DataScienceTeamRole" as the *space default execution role*, granting broad permissions for your team. When a user profile creates a *shared space*, that space will assume the *space default execution role*. For information on assigning an execution role to an existing domain, see [Get space execution role](#).

Instead of allowing any individual user profile working in their own *private space* to have full access to Amazon S3 buckets, you can restrict a user profile's permissions and not allow them to alter the Amazon S3 buckets. In this example, you can give them read access to Amazon S3 buckets to retrieve data, run SageMaker training jobs, and deploy models in their *private space*. You can create a user-level execution role called "DataScientistRole" with the relatively more limited permissions. Then you can assign "DataScientistRole" to the *user profile execution role*, granting the necessary permissions to perform their specific data science tasks within the defined scope. When a user profile creates a *private space*, that space will assume the *user execution role*. For information on assigning an execution role to an existing user profile, see [Get user execution role](#).

For information on SageMaker AI execution roles and adding additional permissions to them, see [How to use SageMaker AI execution roles](#).

## View SageMaker AI resources in your domain

You can view Amazon SageMaker AI resources in your Amazon SageMaker AI domain using the SageMaker AI console. Use the following instructions to learn how to view the resources tagged by the domain ARN.

The displayed SageMaker resources following this procedure are those that have the relevant `sagemaker:domain-arn` tag associated to them. Untagged resources may have been created outside the context of a domain or were created before 11/30/2022, when resources were not automatically tagged with the domain ARN. You can add a tag to untagged resources for better filtration by following the steps in [Backfill domain tags](#). Resources created in other domains are automatically filtered out.

### Note

This is not a complete list of active resources on your domain. For all active SageMaker resources, see [AWS Cost Explorer](#).

## To view SageMaker AI resources in your domain using the console

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Expand the left navigation pane, if not already expanded.
3. Under **Admin configurations**, choose **Domains**.
4. From the list of domains, select the domain that you want to open the **Domain settings** page for.
5. On the **Domain details** page, choose the **Resources** tab.
6. On the **Domain resources** page, you can view the details of the resources tagged with the relative domain ARN. The running resources are displayed by default.
7. (Optional) You can filter the displayed resources for each resource type by using the search icon or **Filter status** at the top of each resource type.

## Shut down SageMaker AI resources in your domain

You can shut down Amazon SageMaker AI resources in your Amazon SageMaker AI domain using the SageMaker AI console. Use the following instructions to learn how to shut down the resources tagged by the domain ARN.

The displayed SageMaker resources following this procedure are those that have the relevant `sagemaker:domain-arn` tag associated to them. Untagged resources may have been created outside the context of a domain or were created before 11/30/2022, when resources were not automatically tagged with the domain ARN. You can add a tag to untagged resources for better filtration by following the steps in [Backfill domain tags](#). Resources created in other domains are automatically filtered out.

### Note

This is not a complete list of active resources on your domain. For all active SageMaker resources, see [AWS Cost Explorer](#).

## To shut down SageMaker AI resources in your domain using the console

1. [View SageMaker AI resources in your domain](#)
2. Under a resource type section, check the boxes for the resources you wish to shut down.

- Once the resources are selected, a shutdown option will become available at the top of the resource type section. Choose the option and follow the instructions to shut down the selected resources.

For instructions on how to delete your resources per SageMaker AI feature, see [Where to shut down resources per SageMaker AI features](#).

## Where to shut down resources per SageMaker AI features

You can shut down your Amazon SageMaker AI resources to avoid incurring unwanted charges. In the following table we list the SageMaker AI features or resources and provide links to the documentation on how to shut down SageMaker AI resources.

You can also use the [APIs, CLI, and SDKs](#) provided by SageMaker AI. For example, you can search the [Amazon SageMaker API Reference](#) for Delete\* commands to delete some of the resources you have created. More specifically, you can search for the [DeleteDomain](#) API to learn how to delete a Amazon SageMaker AI domain.

 **Note**

This is not a complete list of active resources on your domain. For all active SageMaker AI resources, see [AWS Cost Explorer](#).

SageMaker AI feature, infrastructure, resources	Instructions to shutting down
<a href="#">Canvas</a>	<a href="#">Logging out of Amazon SageMaker Canvas</a>
<a href="#">Code Editor</a>	<a href="#">Shut down Code Editor resources</a>
<a href="#">Domain</a>	<ul style="list-style-type: none"><li>• <a href="#">Delete an Amazon SageMaker AI domain</a></li><li>• <a href="#">Remove user profiles</a></li></ul>
<a href="#">EMR in Studio Classic</a>	<a href="#">Terminate an Amazon EMR cluster from Studio or Studio Classic</a>
<a href="#">Experiments</a>	<a href="#">Clean up MLflow resources</a>

SageMaker AI feature, infrastructure, resources	Instructions to shutting down
<a href="#">HyperPod</a>	<ul style="list-style-type: none"><li>• <a href="#">Delete a SageMaker HyperPod cluster</a></li><li>• <a href="#">Delete a cluster</a></li></ul>
<a href="#">Inference endpoints</a>	<a href="#">Delete Endpoints and Resources</a>
<a href="#">JupyterLab</a>	<a href="#">Delete unused resources</a>
<a href="#">MLOps</a>	<a href="#">Delete a MLOps Project using Amazon SageMaker Studio or Studio Classic</a>
<a href="#">Notebook instances</a>	<a href="#">Clean up Amazon SageMaker notebook instance resources</a>
<a href="#">Pipelines</a>	<a href="#">Stop a pipeline</a>
<a href="#">Projects</a>	<a href="#">Delete a MLOps Project using Amazon SageMaker Studio or Studio Classic</a>
<a href="#">RStudio on Amazon SageMaker AI</a>	<ul style="list-style-type: none"><li>• <a href="#">Clean up image resources</a></li><li>• <a href="#">Shut down RStudio</a></li><li>• <a href="#">Launch RSessions from the RStudio Launcher</a></li></ul>
<a href="#">Studio</a>	<a href="#">View your Studio running instances, applications, and spaces</a>
<a href="#">Studio Classic</a>	<ul style="list-style-type: none"><li>• <a href="#">Stacks with AWS CloudFormation</a></li><li>• <a href="#">Clean up resources: images</a></li><li>• <a href="#">Stop a Training Job in SageMaker Studio Classic</a></li><li>• <a href="#">Delete a shared space</a></li></ul>
<a href="#">Stacks in AWS CloudFormation</a>	<a href="#">Deleting a stack on the AWS CloudFormation console</a>

SageMaker AI feature, infrastructure, resources	Instructions to shutting down
<a href="#">TensorBoard in SageMaker AI</a>	<a href="#">Delete unused TensorBoard applications</a>

## Choose an Amazon VPC

This topic provides detailed information about choosing an Amazon Virtual Private Cloud (Amazon VPC) when you onboard to Amazon SageMaker AI domain. For more information about onboarding to SageMaker AI domain, see [Amazon SageMaker AI domain overview](#).

By default, SageMaker AI domain uses two Amazon VPCs. One Amazon VPC is managed by Amazon SageMaker AI and provides direct internet access. You specify the other Amazon VPC, which provides encrypted traffic between the domain and your Amazon Elastic File System (Amazon EFS) volume.

You can change this behavior so that SageMaker AI sends all traffic over your specified Amazon VPC. When you choose this option, you must provide the subnets, security groups, and interface endpoints that are necessary to communicate with the SageMaker API and SageMaker AI runtime, and various AWS services, such as Amazon Simple Storage Service (Amazon S3) and Amazon CloudWatch, that are used by Studio.

When you onboard to SageMaker AI domain, you tell SageMaker AI to send all traffic over your Amazon VPC by setting the network access type to **VPC only**.

### To specify the Amazon VPC information

When you specify the Amazon VPC entities (that is, the Amazon VPC, subnet, or security group) in the following procedure, one of three options is presented based on the number of entities you have in the current AWS Region. The behavior is as follows:

- One entity – SageMaker AI uses that entity. This can't be changed.
- Multiple entities – You must choose the entities from the dropdown list.
- No entities – You must create one or more entities in order to use domain. Choose **Create <entity>** to open the VPC console in a new browser tab. After you create the entities, return to the domain **Get started** page to continue the onboarding process.

This procedure is part of the Amazon SageMaker AI domain onboarding process when you choose **Set up for organizations**. Your Amazon VPC information is specified under the **Network** section.

1. Select the network access type.

 **Note**

If **VPC only** is selected, SageMaker AI automatically applies the security group settings defined for the domain to all shared spaces created in the domain. If **Public internet only** is selected, SageMaker AI does not apply the security group settings to shared spaces created in the domain.

- **Public internet only** – Non-Amazon EFS traffic goes through a SageMaker AI managed Amazon VPC, which allows internet access. Traffic between the domain and your Amazon EFS volume is through the specified Amazon VPC.
  - **VPC only** – All SageMaker AI traffic is through the specified Amazon VPC and subnets. You must use a subnet that does not have direct internet access in **VPC only** mode. Internet access is disabled by default.
2. Choose the Amazon VPC.
  3. Choose one or more subnets. If you don't choose any subnets, SageMaker AI uses all the subnets in the Amazon VPC. We recommend that you use multiple subnets that are not created in constrained Availability Zones. Using subnets in these constrained Availability Zones can result in insufficient capacity errors and longer application creation times. For more information about constrained Availability Zones, see [Availability Zones](#).
  4. Choose the security groups. If you chose **Public internet only**, this step is optional. If you chose **VPC only**, this step is required.

 **Note**

For the maximum number of allowed security groups, see [UserSettings](#).

For Amazon VPC requirements in **VPC only** mode, see [Connect Studio notebooks in a VPC to external resources](#).

# Supported Regions and Quotas

This page gives information about the AWS Regions supported by Amazon SageMaker AI and the Amazon Elastic Compute Cloud (Amazon EC2) instance types, as well as quotas for Amazon SageMaker AI resources.

For information about the instance types that are available in each Region, see [Amazon SageMaker Pricing](#).

For a list of the SageMaker AI service endpoints for each Region, see [Amazon SageMaker AI endpoints and quotas](#) in the *AWS General Reference*.

## Quotas

For a list of SageMaker AI quotas, see [Amazon SageMaker AI endpoints and quotas](#) in the *AWS General Reference*.

The [Service Quotas console](#) provides information about your service quotas. You can use the Service Quotas console to view your default service quotas or to request quota increases. To request a quota increase for adjustable quotas, see [Requesting a quota increase](#).

You can set up a quota request template for your AWS Organization that automatically requests quota increases during account creation. For more information, see [Using Service Quotas request templates](#).

# Automated ML, no-code, or low-code

Amazon SageMaker AI offers the following features to automate key machine learning tasks and use no-code or low-code solutions.

- **Amazon SageMaker Canvas:** For a UI-based, no-code AutoML experience, new users should use the [Amazon SageMaker Canvas](#) application in [Amazon SageMaker Studio](#).

Amazon SageMaker Canvas provides analysts and citizen data scientists no-code capabilities for tasks such as data preparation, feature engineering, algorithm selection, training and tuning, inference, and more. Users can leverage built-in visualizations and what-if analysis to explore their data and different scenarios, with automated predictions enabling them to easily productionize their models. SageMaker Canvas supports a variety of use cases, including computer vision, demand forecasting, intelligent search, and generative AI.

- **Amazon SageMaker Autopilot:** [Amazon SageMaker Autopilot](#) is an automated machine learning (AutoML) feature-set that automates the end-to-end process of building, training, tuning, and deploying machine learning models. Amazon SageMaker Autopilot analyzes your data, selects algorithms suitable for your problem type, preprocesses the data to prepare it for training, handles automatic model training, and performs hyperparameter optimization to find the best performing model for your dataset.
  - As of November 30, 2023, the user interface (UI) for Autopilot is integrated into the [Amazon SageMaker Canvas](#) application in Studio.
  - Users of [Amazon SageMaker Studio Classic](#), the previous experience of Studio, can continue using the Autopilot UI in Studio Classic. Users with coding experience can continue using the [AutoML API references](#) in any supported SDK for technical implementation.

 **Note**

If you have been using Autopilot in Studio Classic until now and want to migrate to SageMaker Canvas, you might have to grant additional permissions to your user profile or IAM role so that you can create and use the SageMaker Canvas application. For more information, see [the section called “\(Optional\) Migrate from Autopilot in Studio Classic to SageMaker Canvas”](#).

- **Amazon SageMaker JumpStart:** SageMaker JumpStart provides pretrained, open-source models for a wide range of problem types to help you get started with machine learning. You can

incrementally train and tune these models before deployment. JumpStart also provides solution templates that set up infrastructure for common use cases, and executable example notebooks for machine learning with SageMaker AI.

## Topics

- [SageMaker Autopilot](#)
- [SageMaker JumpStart pretrained models](#)

# SageMaker Autopilot

### Important

As of November 30, 2023, Autopilot's UI is migrating to [Amazon SageMaker Canvas](#) as part of the updated [Amazon SageMaker Studio](#) experience. SageMaker Canvas provides analysts and citizen data scientists no-code capabilities for tasks such as data preparation, feature engineering, algorithm selection, training and tuning, inference, and more. Users can leverage built-in visualizations and what-if analysis to explore their data and different scenarios, with automated predictions enabling them to easily productionize their models. Canvas supports a variety of use cases, including computer vision, demand forecasting, intelligent search, and generative AI.

Users of [Amazon SageMaker Studio Classic](#), the previous experience of [Studio](#), can continue using the Autopilot UI in Studio Classic. Users with coding experience can continue using all [API references](#) in any supported SDK for technical implementation.

If you have been using Autopilot in Studio Classic until now and want to migrate to SageMaker Canvas, you might have to grant additional permissions to your user profile or IAM role so that you can create and use the SageMaker Canvas application. For more information, see [the section called “\(Optional\) Migrate from Autopilot in Studio Classic to SageMaker Canvas”](#).

All UI-related instructions in this guide pertain to Autopilot's standalone features before migrating to [Amazon SageMaker Canvas](#). Users following these instructions should use [Studio Classic](#).

Amazon SageMaker Autopilot is a feature set that simplifies and accelerates various stages of the machine learning workflow by automating the process of building and deploying machine

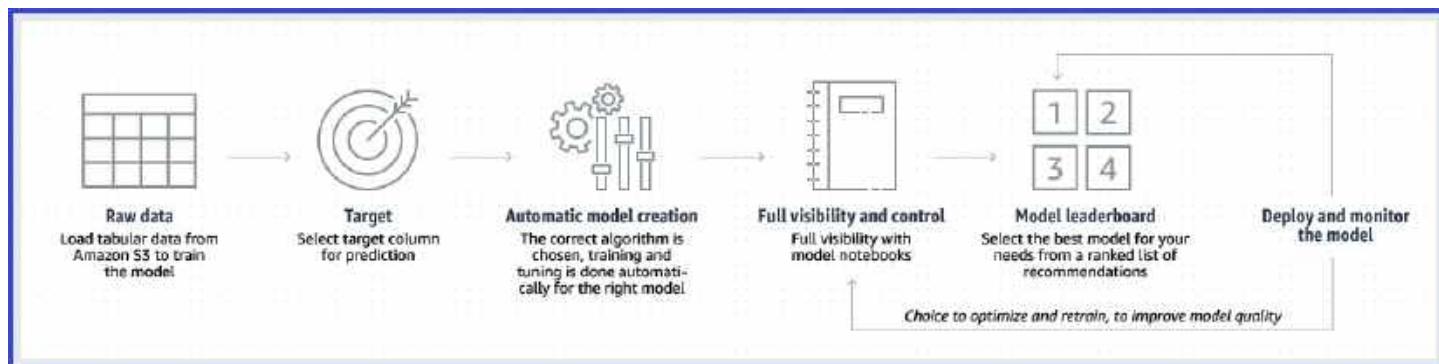
learning models (AutoML). The following page explains key information about Amazon SageMaker Autopilot.

Autopilot performs the following key tasks that you can use on autopilot or with various degrees of human guidance:

- **Data analysis and preprocessing:** Autopilot identifies your specific problem type, handles missing values, normalizes your data, selects features, and overall prepares the data for model training.
- **Model selection:** Autopilot explores a variety of algorithms and uses a cross-validation resampling technique to generate metrics that evaluate the predictive quality of the algorithms based on predefined objective metrics.
- **Hyperparameter optimization:** Autopilot automates the search for optimal hyperparameter configurations.
- **Model training and evaluation:** Autopilot automates the process of training and evaluating various model candidates. It splits the data into training and validation sets, trains the selected model candidates using the training data, and evaluates their performance on the unseen data of the validation set. Lastly, it ranks the optimized model candidates based on their performance and identifies the best performing model.
- **Model deployment:** Once Autopilot has identified the best performing model, it provides the option to deploy the model automatically by generating the model artifacts and the endpoint exposing an API. External applications can send data to the endpoint and receive the corresponding predictions or inferences.

Autopilot supports building machine learning models on large datasets up to hundreds of GBs.

The following diagram outlines the tasks of this AutoML process managed by Autopilot.



Depending on your comfort level with the machine learning process and coding experience, you can use Autopilot in different ways:

- **Using the Studio Classic UI**, users can choose between a no-code experience or have some level of human input.

 **Note**

Only experiments created from tabular data for problem types such as regression or classification are available via the Studio Classic UI.

- **Using the AutoML API**, users with coding experience can use available SDKs to create AutoML jobs. This approach provides greater flexibility and customization options and is available for all problem types.

Autopilot currently supports the following problem types:

 **Note**

For regression or classification problems involving tabular data, users can choose between two options: using the Studio Classic user interface or the [API Reference](#).

Tasks such as text and image classification, time-series forecasting, and fine-tuning of large language models are exclusively available through the version 2 of the [AutoML REST API](#).

If your language of choice is Python, you can refer to [AWS SDK for Python \(Boto3\)](#) or the [AutoMLV2 object](#) of the Amazon SageMaker Python SDK directly.

Users who prefer the convenience of a user interface can use [Amazon SageMaker Canvas](#) to access pre-trained models and generative AI foundation models, or create custom models tailored for specific text, image classification, forecasting needs, or generative AI.

- **Regression, binary, and multiclass classification** with tabular data formatted as CSV or Parquet files in which each column contains a feature with a specific data type and each row contains an observation. The column data types accepted include numerical, categorical, text, and time series that consists of strings of comma-separated numbers.
  - To create an Autopilot job as a pilot experiment using the SageMaker API reference, see [Create Regression or Classification Jobs for Tabular Data Using the AutoML API](#).

- To create an Autopilot job as a pilot experiment using the Studio Classic UI, see [Create a Regression or Classification Autopilot experiment for tabular data using the Studio Classic UI](#).
- If you are an administrator looking to pre-configure default infrastructure, networking, or security parameters of Autopilot experiments in Studio Classic UI, see [Configure the default parameters of an Autopilot experiment \(for administrators\)](#).
- **Text classification** with data formatted as CSV or Parquet files in which a column provides the sentences to be classified, while another column should provide the corresponding class label. See [Create an AutoML job for text classification using the API](#).
- **Image classification** with image formats such as PNG, JPEG, or a combination of both. See [Create an Image Classification Job using the AutoML API](#).
- **Time-series forecasting** with time-series data formatted as CSV or Parquet files. See [Create an AutoML job for time-series forecasting using the API](#).
- Fine-tuning of large language models (LLMs) for **text generation** with data formatted as CSV or Parquet files. See [Create an AutoML job to fine-tune text generation models using the API](#).

Additionally, Autopilot helps users understand how models make predictions by automatically generating reports that show the importance of each individual feature. This provides transparency and insights into the factors influencing the predictions, which can be used by risk and compliance teams and external regulators. Autopilot also provides a model performance report, which encompasses a summary of evaluation metrics, a confusion matrix, various visualizations such as receiver operating characteristic curves and precision-recall curves, and more. The specific content of each report vary depending on the problem type of the Autopilot experiment.

The explainability and performance reports for the best model candidate in an Autopilot experiment are available for text, image, and tabular data classification problem types.

For tabular data use cases such as regression or classification, Autopilot offers additional visibility into how the data was wrangled and how the model candidates were selected, trained, and tuned by generating notebooks that contain the code used to explore the data and find the best performing model. These notebooks provide an interactive and exploratory environment to help you learn about the impact of various inputs or the trade-offs made in the experiments. You can experiment further with the higher performing model candidate by making your own modifications to the data exploration and candidate definition notebooks provided by Autopilot.

With Amazon SageMaker AI, you pay only for what you use. You pay for the underlying compute and storage resources within SageMaker AI or other AWS services, based on your usage. For more information about the cost of using SageMaker AI, see [Amazon SageMaker Pricing](#).

## Topics

- [Create Regression or Classification Jobs for Tabular Data Using the AutoML API](#)
- [Create an Image Classification Job using the AutoML API](#)
- [Create an AutoML job for text classification using the API](#)
- [Create an AutoML job for time-series forecasting using the API](#)
- [Create an AutoML job to fine-tune text generation models using the API](#)
- [Create a Regression or Classification Autopilot experiment for tabular data using the Studio Classic UI](#)
- [Amazon SageMaker Autopilot example notebooks](#)
- [Videos: Use Autopilot to automate and explore the machine learning process](#)
- [Tutorials: Get started with Amazon SageMaker Autopilot](#)
- [Autopilot quotas](#)
- [API Reference guide for Autopilot](#)

## Create Regression or Classification Jobs for Tabular Data Using the AutoML API

You can create an Autopilot regression or classification job for tabular data programmatically by calling the [CreateAutoMLJobV2](#) API action in any language supported by Autopilot or the AWS CLI. The following is a collection of mandatory and optional input request parameters for the CreateAutoMLJobV2 API action. You can find the alternative information for the previous version of this action, CreateAutoMLJob. However, we recommend using CreateAutoMLJobV2.

For information on how this API action translates into a function in the language of your choice, see the [See Also](#) section of CreateAutoMLJobV2 and choose an SDK. As an example, for Python users, see the full request syntax of [create\\_auto\\_ml\\_job\\_v2](#) in AWS SDK for Python (Boto3).

### Note

[CreateAutoMLJobV2](#) and [DescribeAutoMLJobV2](#) are new versions of [CreateAutoMLJob](#) and [DescribeAutoMLJob](#) which offer backward compatibility.

We recommend using the `CreateAutoMLJobV2`. `CreateAutoMLJobV2` can manage tabular problem types identical to those of its previous version `CreateAutoMLJob`, as well as non-tabular problem types such as image or text classification, or time-series forecasting.

At a minimum, all experiments on tabular data require the specification of the experiment name, providing locations for the input and output data, and specifying which target data to predict. Optionally, you can also specify the type of problem that you want to solve (regression, classification, multiclass classification), choose your modeling strategy (*stacked ensembles* or *hyperparameters optimization*), select the list of algorithms used by the Autopilot job to train the data, and more.

After the experiment runs, you can compare trials and delve into the details of the pre-processing steps, algorithms, and hyperparameter ranges of each model. You also have the option to download their [explainability](#) and [performance](#) reports. Use the provided [notebooks](#) to see the results of the automated data exploration or the candidate model definitions.

Find guidelines on how to migrate a `CreateAutoMLJob` to `CreateAutoMLJobV2` in [Migrate a CreateAutoMLJob to CreateAutoMLJobV2](#).

## Required parameters

### `CreateAutoMLJobV2`

When calling [CreateAutoMLJobV2](#) to create an Autopilot experiment for tabular data, you must provide the following values:

- An [AutoMLJobName](#) to specify the name of your job.
- At least one [AutoMLJobChannel](#) in [AutoMLJobInputDataConfig](#) to specify your data source.
- Both an [AutoMLJobObjective](#) metric and your chosen type of supervised learning problem (binary classification, multiclass classification, regression) in `AutoMLProblemTypeConfig`, or none at all. For tabular data, you must choose [TabularJobConfig](#) as the type of [AutoMLProblemTypeConfig](#). You set the supervised learning problem in the `ProblemType` attribute of `TabularJobConfig`.
- An [OutputDataConfig](#) to specify the Amazon S3 output path to store the artifacts of your AutoML job.

- A [RoleArn](#) to specify the ARN of the role used to access your data.

## CreateAutoMLJob

When calling [CreateAutoMLJob](#) to create an AutoML experiment, you must provide the following four values:

- An [AutoMLJobName](#) to specify the name of your job.
- At least one [AutoMLChannel](#) in [InputDataConfig](#) to specify your data source.
- An [OutputDataConfig](#) to specify the Amazon S3 output path to store the artifacts of your AutoML job.
- A [RoleArn](#) to specify the ARN of the role used to access your data.

All other parameters are optional.

## Optional parameters

The following sections provide details of some optional parameters that you can pass to your CreateAutoMLJobV2 API action when using tabular data. You can find the alternative information for the previous version of this action, CreateAutoMLJob. However, we recommend using CreateAutoMLJobV2.

### How to set the training mode of an AutoML job

For tabular data, the set of algorithms run on your data to train your model candidates is dependent on your modeling strategy (ENSEMBLING or HYPERPARAMETER\_TUNING). The following details how to set this training mode.

If you keep blank (or null), the Mode is inferred based on the size of your dataset.

For information on Autopilot's *stacked ensembles* and *hyperparameters optimization* training methods, see [Training modes and algorithm support](#)

## CreateAutoMLJobV2

For tabular data, you must choose [TabularJobConfig](#) as the type of [AutoMLProblemTypeConfig](#).

You can set the [training method](#) of an AutoML job V2 with the [TabularJobConfig.Mode](#) parameter.

## CreateAutoMLJob

You can set the [training method](#) of an AutoML job with the [AutoMLJobConfig.Mode](#) parameter.

### How to select features and algorithms for training an AutoML job

#### Features selection

Autopilot provides automatic data-preprocessing steps including feature selection and feature extraction. However, you can manually provide the features to be used in training with the `FeatureSpecificationS3Uri` attribute.

Selected features should be contained within a JSON file in the following format:

```
{ "FeatureAttributeNames": ["col1", "col2", ...] }
```

The values listed in `["col1", "col2", ...]` are case sensitive. They should be a list of strings containing unique values that are subsets of the column names in the input data.

#### Note

The list of columns provided as features cannot include the target column.

## CreateAutoMLJobV2

For tabular data, you must choose [TabularJobConfig](#) as the type of [AutoMLProblemTypeConfig](#).

You can set the URL to your selected features with the [TabularJobConfig.FeatureSpecificationS3Uri](#) parameter.

## CreateAutoMLJob

You can set the `FeatureSpecificationS3Uri` attribute of [AutoMLCandidateGenerationConfig](#) within the [CreateAutoMLJob](#) API with the following format:

```
{
  "AutoMLJobConfig": {
    "CandidateGenerationConfig": {
      "FeatureSpecificationS3Uri": "string"
    }
  }
}
```

```
        },
    }
}
```

## Algorithms selection

By default, your Autopilot job runs a pre-defined list of algorithms on your dataset to train model candidates. The list of algorithms depends on the training mode (ENSEMBLING or HYPERPARAMETER\_TUNING) used by the job.

You can provide a subset of the default selection of algorithms.

### CreateAutoMLJobV2

For tabular data, you must choose [TabularJobConfig](#) as the type of [AutoMLProblemTypeConfig](#).

You can specify an array of selected AutoMLAlgorithms in the AlgorithmsConfig attribute of [CandidateGenerationConfig](#).

The following is an example of an AlgorithmsConfig attribute listing exactly three algorithms ("xgboost", "fastai", "catboost") in its AutoMLAlgorithms field for the ensembling training mode.

```
{
  "AutoMLProblemTypeConfig": {
    "TabularJobConfig": {
      "Mode": "ENSEMBLING",
      "CandidateGenerationConfig": {
        "AlgorithmsConfig": [
          {"AutoMLAlgorithms": ["xgboost", "fastai", "catboost"]}
        ]
      },
    },
  },
}
```

### CreateAutoMLJob

You can specify an array of selected AutoMLAlgorithms in the AlgorithmsConfig attribute of [AutoMLCandidateGenerationConfig](#).

The following is an example of an `AlgorithmsConfig` attribute listing exactly three algorithms ("xgboost", "fastai", "catboost") in its `AutoMLAlgorithms` field for the ensembling training mode.

```
{  
    "AutoMLJobConfig": {  
        "CandidateGenerationConfig": {  
            "AlgorithmsConfig": [  
                {"AutoMLAlgorithms": ["xgboost", "fastai", "catboost"]}  
            ]  
        },  
        "Mode": "ENSEMBLING"  
    }  
}
```

For the list of available algorithms per training Mode, see [AutoMLAlgorithms](#). For details on each algorithm, see [Training modes and algorithm support](#).

## How to specify the training and validation datasets of an AutoML job

You can provide your own validation dataset and custom data split ratio, or let Autopilot split the dataset automatically.

### CreateAutoMLJobV2

Each [AutoMLJobChannel](#) object (see the required parameter [AutoMLJobInputDataConfig](#)) has a `ChannelType`, which can be set to either `training` or `validation` values that specify how the data is to be used when building a machine learning model. At least one data source must be provided and a maximum of two data sources is allowed: one for training data and one for validation data.

How you split the data into training and validation datasets depends on whether you have one or two data sources.

- If you only have **one data source**, the `ChannelType` is set to `training` by default and must have this value.
  - If the `ValidationFraction` value in [AutoMLDataSplitConfig](#) is not set, 0.2 (20%) of the data from this source is used for validation by default.

- If the ValidationFraction is set to a value between 0 and 1, the dataset is split based on the value specified, where the value specifies the fraction of the dataset used for validation.
- If you have **two data sources**, the ChannelType of one of the AutoMLJobChannel objects must be set to training, the default value. The ChannelType of the other data source must be set to validation. The two data sources must have the same format, either CSV or Parquet, and the same schema. You must not set the value for the ValidationFraction in this case because all of the data from each source is used for either training or validation. Setting this value causes an error.

## CreateAutoMLJob

Each [AutoMLChannel1](#) object (see the required parameter [InputDataConfig](#)) has a ChannelType, which can be set to either training or validation values that specify how the data is to be used when building a machine learning model. At least one data source must be provided and a maximum of two data sources is allowed: one for training data and one for validation data.

How you split the data into training and validation datasets depends on whether you have one or two data sources.

- If you only have **one data source**, the ChannelType is set to training by default and must have this value.
  - If the ValidationFraction value in [AutoMLDataSplitConfig](#) is not set, 0.2 (20%) of the data from this source is used for validation by default.
  - If the ValidationFraction is set to a value between 0 and 1, the dataset is split based on the value specified, where the value specifies the fraction of the dataset used for validation.
- If you have **two data sources**, the ChannelType of one of the AutoMLChannel objects must be set to training, the default value. The ChannelType of the other data source must be set to validation. The two data sources must have the same format, either CSV or Parquet, and the same schema. You must not set the value for the ValidationFraction in this case because all of the data from each source is used for either training or validation. Setting this value causes an error.

For information on split and cross-validation in Autopilot see [Cross-validation in Autopilot](#).

## How to set the problem type of an AutoML job

### CreateAutoMLJobV2

For tabular data, you must choose [TabularJobConfig](#) as the type of [AutoMLProblemTypeConfig](#).

You can further specify the type of supervised learning problem (binary classification, multiclass classification, regression) available for the model candidates of your AutoML job V2 with the [TabularJobConfig.ProblemType](#) parameter.

### CreateAutoMLJob

You can set the [type of problem](#) on an AutoML job with the [CreateAutoPilot.ProblemType](#) parameter. This limits the kind of preprocessing and algorithms that Autopilot tries.

After the job is finished, if you had set the [CreateAutoPilot.ProblemType](#), then the [ResolvedAttribute.ProblemType](#) matches the ProblemType you set. If you keep it blank (or null), the ProblemType is inferred on your behalf.

#### Note

In some cases, Autopilot is unable to infer the ProblemType with high enough confidence, in which case you must provide the value for the job to succeed.

## How to add sample weights to an AutoML job

You can add a sample weights column to your tabular dataset and then pass it to your AutoML job to request dataset rows to be weighted during training and evaluation.

Support for sample weights is available in [ensembling mode](#) only. Your weights should be numeric and non-negative. Data points with invalid or no weight value are excluded. For more information on the available objective metrics, see [Autopilot weighted metrics](#).

### CreateAutoMLJobV2

For tabular data, you must choose [TabularJobConfig](#) as the type of [AutoMLProblemTypeConfig](#).

To set sample weights when creating an experiment (see [CreateAutoMLJobV2](#)), you can pass the name of your sample weights column in the SampleWeightAttributeName attribute of the

TabularJobConfig object. This ensures that your objective metric uses the weights for the training, evaluation, and selection of model candidates.

## CreateAutoMLJob

To set sample weights when creating an experiment (see [CreateAutoMLJob](#)), you can pass the name of your sample weights column in the SampleWeightAttributeName attribute of the [AutoMLChannel](#) object. This ensures that your objective metric uses the weights for the training, evaluation, and selection of model candidates.

## How to configure AutoML to initiate a remote job on EMR Serverless for large datasets

You can configure your AutoML job V2 to automatically initiate a remote job on Amazon EMR Serverless when additional compute resources are needed to process large datasets. By seamlessly transitioning to EMR Serverless when required, the AutoML job can handle datasets that would otherwise exceed the initially provisioned resources, without any manual intervention from you. EMR Serverless is available for the tabular and time series problem types. We recommend setting up this option for tabular datasets larger than 5 GB.

To allow your AutoML job V2 to automatically transition to EMR Serverless for large dataset, you need to provide an EmrServerlessComputeConfig object, which includes an ExecutionRoleARN field, to the AutoMLComputeConfig of the AutoML job V2 input request.

The ExecutionRoleARN is the ARN of the IAM role granting the AutoML job V2 the necessary permissions to run EMR Serverless jobs.

This role should have the following trust relationship:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "emr-serverless.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

And grant the permissions to:

- Create, list, and update EMR Serverless applications.
- Start, list, get, or cancel job runs on an EMR Serverless application.
- Tag EMR Serverless resources.
- Pass an IAM role to the EMR Serverless service for execution.

By granting the `iam:PassRole` permission, the AutoML job V2 can temporarily assume the `EMRServerlessRuntimeRole-*` role and pass it to the EMR Serverless service. These are the IAM roles used by the EMR Serverless job execution environments to access other AWS services and resources needed during runtime, such as Amazon S3 for data access, CloudWatch for logging, access to the AWS Glue Data Catalog or other services based on your workload requirements.

See [Job runtime roles for Amazon EMR Serverless](#) for details on this role permissions.

The IAM policy defined in the provided JSON document grants those permissions:

```
{  
    "Version": "2012-10-17",  
    "Statement": [{  
        "Sid": "EMRServerlessCreateApplicationOperation",  
        "Effect": "Allow",  
        "Action": "emr-serverless>CreateApplication",  
        "Resource": "arn:aws:emr-serverless:*:*:/*",  
        "Condition": {  
            "StringEquals": {  
                "aws:RequestTag/sagemaker:is-canvas-resource": "True",  
                "aws:ResourceAccount": "${aws:PrincipalAccount}"  
            }  
        }  
    },  
    {  
        "Sid": "EMRServerlessListApplicationOperation",  
        "Effect": "Allow",  
        "Action": "emr-serverless>ListApplications",  
        "Resource": "arn:aws:emr-serverless:*:*:/*",  
        "Condition": {  
            "StringEquals": {  
                "aws:ResourceAccount": "${aws:PrincipalAccount}"  
            }  
        }  
    }]
```

```
+         }
+     }
+ },
{
    "Sid": "EMRServerlessApplicationOperations",
    "Effect": "Allow",
    "Action": [
        "emr-serverless:UpdateApplication",
        "emr-serverless:GetApplication"
    ],
    "Resource": "arn:aws:emr-serverless:*:*:/applications/*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/sagemaker:is-canvas-resource": "True",
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
},
{
    "Sid": "EMRServerlessStartJobRunOperation",
    "Effect": "Allow",
    "Action": "emr-serverless:StartJobRun",
    "Resource": "arn:aws:emr-serverless:*:*:/applications/*",
    "Condition": {
        "StringEquals": {
            "aws:RequestTag/sagemaker:is-canvas-resource": "True",
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
},
{
    "Sid": "EMRServerlessListJobRunOperation",
    "Effect": "Allow",
    "Action": "emr-serverless>ListJobRuns",
    "Resource": "arn:aws:emr-serverless:*:*:/applications/*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/sagemaker:is-canvas-resource": "True",
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
},
{
    "Sid": "EMRServerlessJobRunOperations",
```

```
+         "Effect": "Allow",
+         "Action": [
+             "emr-serverless:GetJobRun",
+             "emr-serverless:CancelJobRun"
+         ],
+         "Resource": "arn:aws:emr-serverless:*:*:/applications/*/jobruns/*",
+         "Condition": {
+             "StringEquals": {
+                 "aws:ResourceTag/sagemaker:is-canvas-resource": "True",
+                 "aws:ResourceAccount": "${aws:PrincipalAccount}"
+             }
+         }
+     },
+     {
+         "Sid": "EMRServerlessTagResourceOperation",
+         "Effect": "Allow",
+         "Action": "emr-serverless:TagResource",
+         "Resource": "arn:aws:emr-serverless:*:*:/*",
+         "Condition": {
+             "StringEquals": {
+                 "aws:RequestTag/sagemaker:is-canvas-resource": "True",
+                 "aws:ResourceAccount": "${aws:PrincipalAccount}"
+             }
+         }
+     },
+     {
+         "Sid": "IAMPassOperationForEMRServerless",
+         "Effect": "Allow",
+         "Action": "iam:PassRole",
+         "Resource": "arn:aws:iam::*:role/EMRServerlessRuntimeRole-*",
+         "Condition": {
+             "StringEquals": {
+                 "iam:PassedToService": "emr-serverless.amazonaws.com",
+                 "aws:ResourceAccount": "${aws:PrincipalAccount}"
+             }
+         }
+     }
]
}
```

## Migrate a CreateAutoMLJob to CreateAutoMLJobV2

We recommend users of `CreateAutoMLJob` to migrate to `CreateAutoMLJobV2`.

This section explains the differences in the input parameters between [CreateAutoMLJob](#) and [CreateAutoMLJobV2](#) by highlighting the changes in the position, name, or structure of the objects and attributes of the input request between the two versions.

- **Request attributes that did not change between versions.**

```
{  
    "AutoMLJobName": "string",  
    "AutoMLJobObjective": {  
        "MetricName": "string"  
    },  
    "ModelDeployConfig": {  
        "AutoGenerateEndpointName": boolean,  
        "EndpointName": "string"  
    },  
    "OutputDataConfig": {  
        "KmsKeyId": "string",  
        "S3OutputPath": "string"  
    },  
    "RoleArn": "string",  
    "Tags": [  
        {  
            "Key": "string",  
            "Value": "string"  
        }  
    ]  
}
```

- **Request attributes that changed position and structure between versions.**

The following attributes changed position: DataSplitConfig, Security Config, CompletionCriteria, Mode, FeatureSpecificationS3Uri, SampleWeightAttributeName, TargetAttributeName.

### CreateAutoMLJob

```
{  
    "AutoMLJobConfig": {  
        "Mode": "string",  
        "CompletionCriteria": {  
            "MaxAutoMLJobRuntimeInSeconds": number,  
            "MaxCandidates": number,  
            "MaxRuntimePerTrainingJobInSeconds": number  
        }  
    }  
}
```

```
        },
        "DataSplitConfig": {
            "ValidationFraction": number
        },
        "SecurityConfig": {
            "EnableInterContainerTrafficEncryption": boolean,
            "VolumeKmsKeyId": "string",
            "VpcConfig": {
                "SecurityGroupIds": [ "string" ],
                "Subnets": [ "string" ]
            }
        },
        "CandidateGenerationConfig": {
            "FeatureSpecificationS3Uri": "string"
        }
    },
    "GenerateCandidateDefinitionsOnly": boolean,
    "ProblemType": "string"
}
```

## CreateAutoMLJobV2

```
{
    "AutoMLProblemTypeConfig": {
        "TabularJobConfig": {
            "Mode": "string",
            "ProblemType": "string",
            "GenerateCandidateDefinitionsOnly": boolean,
            "CompletionCriteria": {
                "MaxAutoMLJobRuntimeInSeconds": number,
                "MaxCandidates": number,
                "MaxRuntimePerTrainingJobInSeconds": number
            },
            "FeatureSpecificationS3Uri": "string",
            "SampleWeightAttributeName": "string",
            "TargetAttributeName": "string"
        }
    },
    "DataSplitConfig": {
        "ValidationFraction": number
    },
    "SecurityConfig": {
        "EnableInterContainerTrafficEncryption": boolean,
```

```
        "VolumeKmsKeyId": "string",
        "VpcConfig": {
            "SecurityGroupIds": [ "string" ],
            "Subnets": [ "string" ]
        }
    }
}
```

- The following attributes changed position and structure between versions.

The following JSON illustrates how [AutoMLJobConfig.CandidateGenerationConfig](#) of type [AutoMLCandidateGenerationConfig](#) moved to [AutoMLProblemTypeConfig.TabularJobConfig.CandidateGenerationConfig](#) of type [CandidateGenerationConfig](#) in V2.

## CreateAutoMLJob

```
{
    "AutoMLJobConfig": {
        "CandidateGenerationConfig": {
            "AlgorithmsConfig": [
                {
                    "AutoMLAlgorithms": [ "string" ]
                }
            ],
            "FeatureSpecificationS3Uri": "string"
        }
    }
}
```

## CreateAutoMLJobV2

```
{  
    "AutoMLProblemTypeConfig": {  
        "TabularJobConfig": {  
            "CandidateGenerationConfig": {  
                "AlgorithmsConfig": [  
                    {  
                        "AutoMLAlgorithms": [ "string" ]  
                    }  
                ],  
                },  
            },  
        },  
    },  
},
```

```
}
```

- **Request attributes that changed name and structure.**

The following JSON illustrates how [InputDataConfig](#) (An array of [AutoMLChannel](#)) changed to [AutoMLJobInputDataConfig](#) (An array of [AutoMLJobChannel](#)) in V2. Note that the attributes SampleWeightAttributeName and TargetAttributeName move out of InputDataConfig and into AutoMLProblemTypeConfig.

### CreateAutoMLJob

```
{
    "InputDataConfig": [
        {
            "ChannelType": "string",
            "CompressionType": "string",
            "ContentType": "string",
            "DataSource": {
                "S3DataSource": {
                    "S3DataType": "string",
                    "S3Uri": "string"
                }
            },
            "SampleWeightAttributeName": "string",
            "TargetAttributeName": "string"
        }
    ]
}
```

### CreateAutoMLJobV2

```
{
    "AutoMLJobInputDataConfig": [
        {
            "ChannelType": "string",
            "CompressionType": "string",
            "ContentType": "string",
            "DataSource": {
                "S3DataSource": {
                    "S3DataType": "string",
                    "S3Uri": "string"
                }
            }
        }
    ]
}
```

```
}
```

## Autopilot datasets and problem types

For tabular data (that is data in which each column contains a feature with a specific data type and each row contains an observation), Autopilot gives you the option of specifying the type of supervised learning problem available for the model candidates of the AutoML job, such as binary classification or regression, or of detecting it on your behalf based on the data you provide. Autopilot also supports multiple data formats and data types.

### Topics

- [Autopilot datasets, data types, and formats](#)
- [Autopilot problem types](#)

### Autopilot datasets, data types, and formats

Autopilot supports tabular data formatted as CSV files or as Parquet files: each column contains a feature with a specific data type and each row contains an observation. The properties of these two file formats differ considerably.

- **CSV** (comma-separated-values) is a row-based file format that stores data in human readable plaintext which a popular choice for data exchange as they are supported by a wide range of applications.
- **Parquet** is a column-based file format where the data is stored and processed more efficiently than row-based file formats. This makes them a better option for big data problems.

The **data types** accepted for columns include numerical, categorical, text, and time series that consists of strings of comma-separated numbers. If Autopilot detects it is dealing with **time series** sequences, it processes them through specialized feature transformers provided by the [tsfresh](#) library. This library takes the time series as an input and outputs a feature such as the highest absolute value of the time series or descriptive statistics on autocorrelation. These outputted features are then used as inputs to one of the three problem types.

Autopilot supports building machine learning models on large datasets up to hundreds of GBs. For details on the default resource limits for input datasets and how to increase them, see [Autopilot quotas](#).

## Autopilot problem types

For the tabular data, you further specify the type of supervised learning problems available for the model candidates as follows:

### Regression

Regression estimates the values of a dependent target variable based on one or more other variables or attributes that are correlated with it. An example is the prediction of house prices using features like the number of bathrooms and bedrooms, square footage of the house and garden. Regression analysis can create a model that takes one or more of these features as an input and predicts the price of a house.

### Binary classification

Binary classification is a type of supervised learning that assigns an individual to one of two predefined and mutually exclusive classes based on their attributes. It is supervised because the models are trained using examples where the attributes are provided with correctly labeled objects. A medical diagnosis for whether an individual has a disease or not based on the results of diagnostic tests is an example of binary classification.

### Multiclass classification

Multiclass classification is a type of supervised learning that assigns an individual to one of several classes based on their attributes. It is supervised because the models are trained using examples where the attributes are provided with correctly labelled objects. An example is the prediction of the topic most relevant to a text document. A document may be classified as being about, say, religion or politics or finance, or about one of several other predefined topic classes.

## Training modes and algorithm support

Autopilot supports different training modes and algorithms to address machine learning problems, report on quality and objective metrics, and to use cross-validation automatically, when needed.

### Training modes

SageMaker Autopilot can automatically select the training method based on the dataset size, or you can select it manually. The choices are as follows:

- **Ensembling** – Autopilot uses the [AutoGluon](#) library to train several base models. To find the best combination for your dataset, ensemble mode runs 10 trials with different model and meta parameter settings. Then Autopilot combines these models using a stacking ensemble method to create an optimal predictive model. For a list of algorithms that Autopilot supports in ensembling mode for tabular data, see the following **Algorithms support** section.
- **Hyperparameter optimization (HPO)** – Autopilot finds the best version of a model by tuning hyperparameters using Bayesian optimization or multi-fidelity optimization while running training jobs on your dataset. HPO mode selects the algorithms that are most relevant to your dataset and selects the best range of hyperparameters to tune your models. To tune your models, HPO mode runs up to 100 trials (default) to find the optimal hyperparameters settings within the selected range. If your dataset size is less than 100 MB, Autopilot uses Bayesian optimization. Autopilot chooses multi-fidelity optimization if your dataset is larger than 100 MB.

In multi-fidelity optimization, metrics are continuously emitted from the training containers. A trial that is performing poorly against a selected objective metric is stopped early. A trial that is performing well is allocated more resources.

For a list of algorithms that Autopilot supports in HPO mode, see the following **Algorithm support** section.

- **Auto** – Autopilot automatically chooses either ensembling mode or HPO mode based on your dataset size. If your dataset is larger than 100 MB, Autopilot chooses HPO. Otherwise, it chooses ensembling mode. Autopilot can fail to read the size of your dataset in the following cases.
  - If you enable Virtual Private Cloud (VPC) mode, for an AutoML job but the S3 bucket containing the dataset only allows access from the VPC.
  - The input [S3DataType](#) of your dataset is a [ManifestFile](#).
  - The input [S3Uri](#) contains more than 1000 items.

If Autopilot is unable to read your dataset size, it defaults to choosing HPO mode.

 **Note**

For optimal runtime and performance, use ensemble training mode for datasets that are smaller than 100 MB.

## Algorithms support

In **HPO mode**, Autopilot supports the following types of machine learning algorithms:

- [Linear learner](#) – A supervised learning algorithm that can solve either classification or regression problems.
- [XGBoost](#) – A supervised learning algorithm that attempts to accurately predict a target variable by combining an ensemble of estimates from a set of simpler and weaker models.
- Deep learning algorithm – A multilayer perceptron (MLP) and feedforward artificial neural network. This algorithm can handle data that is not linearly separable.

 **Note**

You don't need to specify an algorithm to use for your machine learning problem. Autopilot automatically selects the appropriate algorithm to train.

In **ensembling mode**, Autopilot supports the following types of machine learning algorithms:

- [LightGBM](#) – An optimized framework that uses tree-based algorithms with gradient boosting. This algorithm uses trees that grow in breadth, rather than depth, and is highly optimized for speed.
- [CatBoost](#) – A framework that uses tree-based algorithms with gradient boosting. Optimized for handling categorical variables.
- [XGBoost](#) – A framework that uses tree-based algorithms with gradient boosting that grows in depth, rather than breadth.
- [Random Forest](#) – A tree-based algorithm that uses several decision trees on random sub-samples of the data with replacement. The trees are split into optimal nodes at each level. The decisions of each tree are averaged together to prevent overfitting and improve predictions.
- [Extra Trees](#) – A tree-based algorithm that uses several decision trees on the entire dataset. The trees are split randomly at each level. The decisions of each tree are averaged to prevent overfitting and to improve predictions. Extra trees add a degree of randomization in comparison to the random forest algorithm.
- [Linear Models](#) – A framework that uses a linear equation to model the relationship between two variables in observed data.
- Neural network torch – A neural network model that's implemented using [Pytorch](#).

- Neural network fast.ai – A neural network model that's implemented using [fast.ai](#).

## Metrics and validation

This guide shows metrics and validation techniques that you can use to measure machine learning model performance. Amazon SageMaker Autopilot produces metrics that measure the predictive quality of machine learning model candidates. The metrics calculated for candidates are specified using an array of [MetricDatum](#) types.

### Autopilot metrics

The following list contains the names of the metrics that are currently available to measure model performance within Autopilot.

 **Note**

Autopilot supports sample weights. To learn more about sample weights and the available objective metrics, see [Autopilot weighted metrics](#).

The following are the available metrics.

### Accuracy

The ratio of the number of correctly classified items to the total number of (correctly and incorrectly) classified items. It is used for both binary and multiclass classification. Accuracy measures how close the predicted class values are to the actual values. Values for accuracy metrics vary between zero (0) and one (1). A value of 1 indicates perfect accuracy, and 0 indicates perfect inaccuracy.

### AUC

The area under the curve (AUC) metric is used to compare and evaluate binary classification by algorithms that return probabilities, such as logistic regression. To map the probabilities into classifications, these are compared against a threshold value.

The relevant curve is the receiver operating characteristic curve. The curve plots the true positive rate (TPR) of predictions (or recall) against the false positive rate (FPR) as a function of the threshold value, above which a prediction is considered positive. Increasing the threshold results in fewer false positives, but more false negatives.

AUC is the area under this receiver operating characteristic curve. Therefore, AUC provides an aggregated measure of the model performance across all possible classification thresholds. AUC scores vary between 0 and 1. A score of 1 indicates perfect accuracy, and a score of one half (0.5) indicates that the prediction is not better than a random classifier.

## BalancedAccuracy

BalancedAccuracy is a metric that measures the ratio of accurate predictions to all predictions. This ratio is calculated after normalizing true positives (TP) and true negatives (TN) by the total number of positive (P) and negative (N) values. It is used in both binary and multiclass classification and is defined as follows:  $0.5 * ((TP/P) + (TN/N))$ , with values ranging from 0 to 1. BalancedAccuracy gives a better measure of accuracy when the number of positives or negatives differ greatly from each other in an imbalanced dataset, such as when only 1% of email is spam.

## F1

The F1 score is the harmonic mean of the precision and recall, defined as follows:  $F1 = 2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$ . It is used for binary classification into classes traditionally referred to as positive and negative. Predictions are said to be true when they match their actual (correct) class, and false when they do not.

Precision is the ratio of the true positive predictions to all positive predictions, and it includes the false positives in a dataset. Precision measures the quality of the prediction when it predicts the positive class.

Recall (or sensitivity) is the ratio of the true positive predictions to all actual positive instances. Recall measures how completely a model predicts the actual class members in a dataset.

F1 scores vary between 0 and 1. A score of 1 indicates the best possible performance, and 0 indicates the worst.

## F1macro

The F1macro score applies F1 scoring to multiclass classification problems. It does this by calculating the precision and recall, and then taking their harmonic mean to calculate the F1 score for each class. Lastly, the F1macro averages the individual scores to obtain the F1macro score. F1macro scores vary between 0 and 1. A score of 1 indicates the best possible performance, and 0 indicates the worst.

## InferenceLatency

Inference latency is the approximate amount of time between making a request for a model prediction to receiving it from a real time endpoint to which the model is deployed. This metric is measured in seconds and only available in ensembling mode.

## LogLoss

Log loss, also known as cross-entropy loss, is a metric used to evaluate the quality of the probability outputs, rather than the outputs themselves. It is used in both binary and multiclass classification and in neural nets. It is also the cost function for logistic regression. Log loss is an important metric to indicate when a model makes incorrect predictions with high probabilities. Values range from 0 to infinity. A value of 0 represents a model that perfectly predicts the data.

## MAE

The mean absolute error (MAE) is a measure of how different the predicted and actual values are, when they're averaged over all values. MAE is commonly used in regression analysis to understand model prediction error. If there is linear regression, MAE represents the average distance from a predicted line to the actual value. MAE is defined as the sum of absolute errors divided by the number of observations. Values range from 0 to infinity, with smaller numbers indicating a better model fit to the data.

## MSE

The mean squared error (MSE) is the average of the squared differences between the predicted and actual values. It is used for regression. MSE values are always positive. The better a model is at predicting the actual values, the smaller the MSE value is.

## Precision

Precision measures how well an algorithm predicts the true positives (TP) out of all of the positives that it identifies. It is defined as follows:  $\text{Precision} = \text{TP}/(\text{TP}+\text{FP})$ , with values ranging from zero (0) to one (1), and is used in binary classification. Precision is an important metric when the cost of a false positive is high. For example, the cost of a false positive is very high if an airplane safety system is falsely deemed safe to fly. A false positive (FP) reflects a positive prediction that is actually negative in the data.

## PrecisionMacro

The precision macro computes precision for multiclass classification problems. It does this by calculating precision for each class and averaging scores to obtain precision for several classes. PrecisionMacro scores range from zero (0) to one (1). Higher scores reflect the model's

ability to predict true positives (TP) out of all of the positives that it identifies, averaged across multiple classes.

## R2

$R^2$ , also known as the coefficient of determination, is used in regression to quantify how much a model can explain the variance of a dependent variable. Values range from one (1) to negative one (-1). Higher numbers indicate a higher fraction of explained variability.  $R^2$  values close to zero (0) indicate that very little of the dependent variable can be explained by the model. Negative values indicate a poor fit and that the model is outperformed by a constant function. For linear regression, this is a horizontal line.

## Recall

Recall measures how well an algorithm correctly predicts all of the true positives (TP) in a dataset. A true positive is a positive prediction that is also an actual positive value in the data. Recall is defined as follows:  $\text{Recall} = \text{TP}/(\text{TP}+\text{FN})$ , with values ranging from 0 to 1. Higher scores reflect a better ability of the model to predict true positives (TP) in the data. It is used in binary classification.

Recall is important when testing for cancer because it's used to find all of the true positives. A false positive (FP) reflects a positive prediction that is actually negative in the data. It is often insufficient to measure only recall, because predicting every output as a true positive yields a perfect recall score.

## RecallMacro

The RecallMacro computes recall for multiclass classification problems by calculating recall for each class and averaging scores to obtain recall for several classes. RecallMacro scores range from 0 to 1. Higher scores reflect the model's ability to predict true positives (TP) in a dataset, whereas a true positive reflects a positive prediction that is also an actual positive value in the data. It is often insufficient to measure only recall, because predicting every output as a true positive will yield a perfect recall score.

## RMSE

Root mean squared error (RMSE) measures the square root of the squared difference between predicted and actual values, and is averaged over all values. It is used in regression analysis to understand model prediction error. It's an important metric to indicate the presence of large model errors and outliers. Values range from zero (0) to infinity, with smaller numbers indicating a better model fit to the data. RMSE is dependent on scale, and should not be used to compare datasets of different sizes.

Metrics that are automatically calculated for a model candidate are determined by the type of problem being addressed.

Refer to the [Amazon SageMaker API reference documentation](#) for the list of available metrics supported by Autopilot.

## Autopilot weighted metrics



### Note

Autopilot supports sample weights in ensembling mode only for all [available metrics](#) with the exception of Balanced Accuracy and InferenceLatency. BalanceAccuracy comes with its own weighting scheme for imbalanced datasets that does not require sample weights. InferenceLatency does not support sample weights. Both objective Balanced Accuracy and InferenceLatency metrics ignore any existing sample weights when training and evaluating a model.

Users can add a sample weights column to their data to ensure that each observation used to train a machine learning model is given a weight corresponding to its perceived importance to the model. This is especially useful in scenarios in which the observations in the dataset have varying degrees of importance, or in which a dataset contains a disproportionate number of samples from one class compared to others. Assigning a weight to each observation based on its importance or greater importance to a minority class can help a model's overall performance, or ensure that a model is not biased toward the majority class.

For information about how to pass sample weights when creating an experiment in the Studio Classic UI, see *Step 7* in [Create an Autopilot experiment using Studio Classic](#).

For information about how to pass sample weights programmatically when creating an Autopilot experiment using the API, see *How to add sample weights to an AutoML job* in [Create an Autopilot experiment programmatically](#).

## Cross-validation in Autopilot

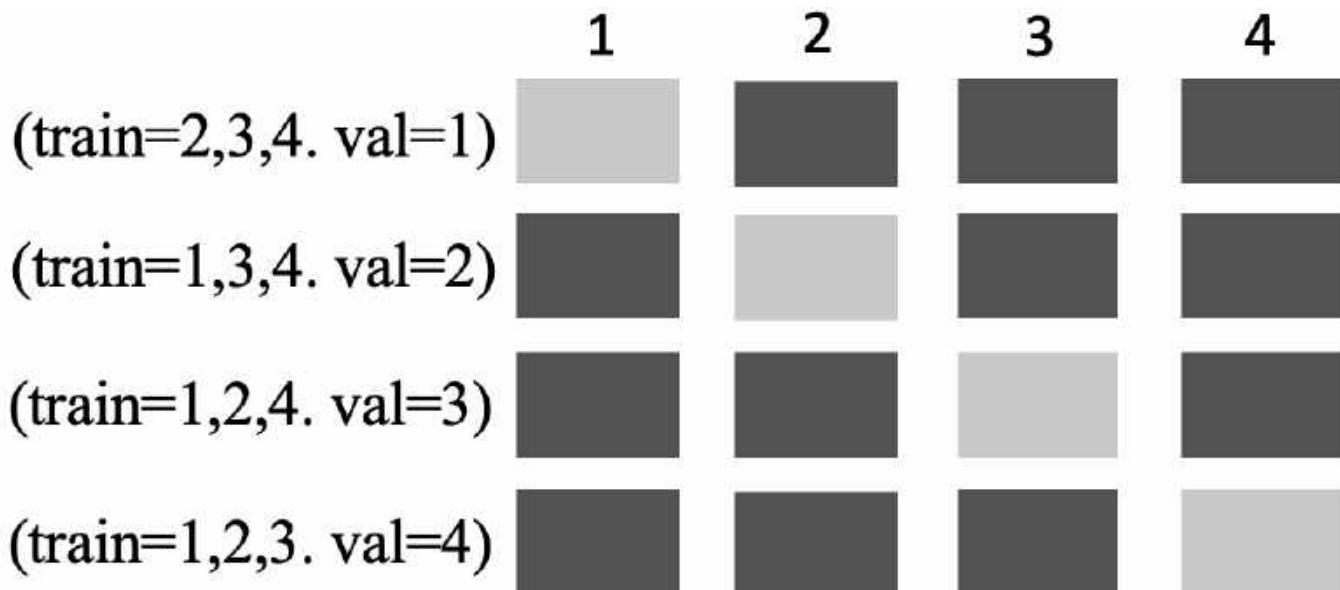
Cross-validation is used to reduce overfitting and bias in model selection. It is also used to assess how well a model can predict the values of an unseen validation dataset, if the validation dataset is drawn from the same population. This method is especially important when training on datasets that have a limited number of training instances.

Autopilot uses cross-validation to build models in hyperparameter optimization (HPO) and ensemble training mode. The first step in the Autopilot cross-validation process is to split the data into k-folds.

## K-fold splitting

K-fold splitting is a method that separates an input training dataset into multiple training and validation datasets. The dataset is split into k equally-sized sub-samples called folds. Models are then trained on  $k - 1$  folds and tested against the remaining  $k^{\text{th}}$  fold, which is the validation dataset. The process is repeated k times using a different data set for validation.

The following image depicts k-fold splitting with  $k = 4$  folds. Each fold is represented as a row. The dark-toned boxes represent the parts of the data used in training. The remaining light-toned boxes indicate the validation datasets.



## *4-fold splitting*

Autopilot uses k-fold cross-validation for both hyperparameter optimization (HPO) mode and ensembling mode.

You can deploy Autopilot models that are built using cross-validation like you would with any other Autopilot or SageMaker AI model.

## HPO mode

K-fold cross-validation uses the k-fold splitting method for cross-validation. In HPO mode, Autopilot automatically implements k-fold cross-validation for small datasets with 50,000 or fewer training instances. Performing cross-validation is especially important when training on small datasets because it protects against overfitting and selection bias.

HPO mode uses a  $k$  value of 5 on each of the candidate algorithms that are used to model the dataset. Multiple models are trained on different splits, and the models are stored separately. When training is complete, validation metrics for each of the models are averaged to produce a single estimation metric. Lastly, Autopilot combines the models from the trial with the best validation metric into an ensemble model. Autopilot uses this ensemble model to make predictions.

The validation metric for the models trained by Autopilot is presented as the objective metric in the model leaderboard. Autopilot uses the default validation metric for each problem type that it handles, unless you specify otherwise. For the list of all metrics that Autopilot uses, see [Autopilot metrics](#).

For example, the [Boston Housing dataset](#) contains only 861 samples. If you build a model to predict house sale prices using this dataset without cross-validation, you risk training on a dataset that is not representative of the Boston housing stock. If you split the data only once into training and validation subsets, the training fold may only contain data mainly from the suburbs. As a result, you would train on data that isn't representative of the rest of the city. In this example, your model would likely overfit on this biased selection. K-fold cross-validation can reduce the risk of this kind of error by making full and randomized use of the available data for both training and validation.

Cross-validation can increase training times by an average of 20%. Training times may also increase significantly for complex datasets.

### Note

In HPO mode, you can see the training and validation metrics from each fold in your `/aws/sagemaker/TrainingJobs` CloudWatch Logs. For more information about CloudWatch Logs, see [Log groups and streams that Amazon SageMaker AI sends to Amazon CloudWatch Logs](#).

## Ensembling mode

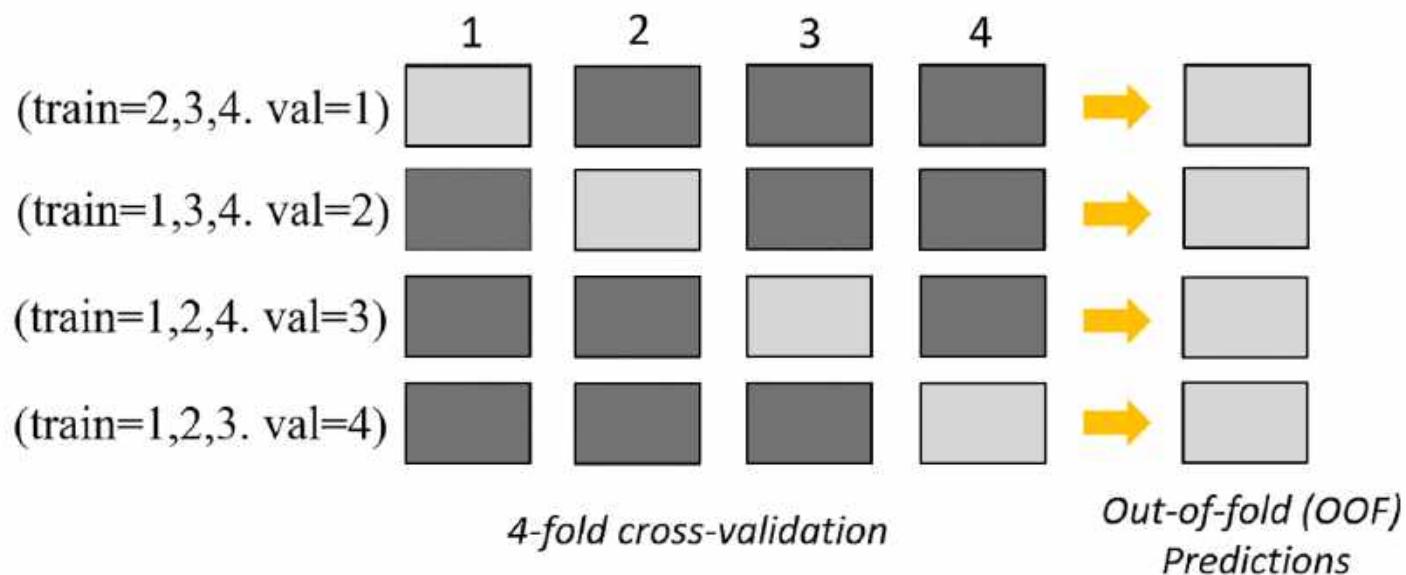
### Note

Autopilot supports sample weights in ensembling mode. For the list of available metrics supporting sample weights, see [Autopilot metrics](#).

In ensembling mode, cross-validation is performed regardless of dataset size. Customers can either provide their own validation dataset and custom data split ratio, or let Autopilot split the dataset automatically into an 80-20% split ratio. The training data is then split into k-folds for cross-validation, where the value of k is determined by the AutoGluon engine. An ensemble consists of multiple machine learning models, where each model is known as the base model. A single base model is trained on  $(k-1)$  folds and makes out-of-fold predictions on the remaining fold. This process is repeated for all k folds, and the out-of-fold (OOF) predictions are concatenated to form a single set of predictions. All base models in the ensemble follow this same process of generating OOF predictions.

The following image depicts k-fold validation with k = 4 folds. Each fold is represented as a row. The dark-toned boxes represent the parts of the data used in training. The remaining light-toned boxes indicate the validation datasets.

In the upper part of the image, in each fold, the first base model makes predictions on the validation dataset after training on the training datasets. At each subsequent fold, the datasets change roles. A dataset that was previously used for training is now used for validation, and this also applies in reverse. At the end of k folds, all of the predictions are concatenated to form a single set of predictions called an out-of-fold (OOF) prediction. This process is repeated for each n base models.



The OOF predictions for each base model are then used as features to train a stacking model. The stacking model learns the importance weights for each base model. These weights are used to combine the OOF predictions to form the final prediction. Performance on the validation dataset determines which base or stacking model is the best, and this model is returned as the final model.

In ensemble mode, you can either provide your own validation dataset or let Autopilot split the input dataset automatically into 80% train and 20% validation datasets. The training data is then split into k-folds for cross-validation and produces an OOF prediction and a base model for each fold.

These OOF predictions are used as features to train a stacking model, which simultaneously learns weights for each base model. These weights are used to combine the OOF predictions to form the final prediction. The validation datasets for each fold are used for hyperparameter tuning of all base models and the stacking model. Performance on the validation datasets determines which base or stacking model is the best model, and this model is returned as the final model.

## Autopilot model deployment and prediction

This Amazon SageMaker Autopilot guide includes steps for model deployment, setting up real-time inference, and running inference with batch jobs.

After you train your Autopilot models, you can deploy them to get predictions in one of two ways:

1. Use [Deploy models for real-time inference](#) to set up an endpoint and obtain predictions interactively. Real-time inference is ideal for inference workloads where you have real-time, interactive, low latency requirements.
2. Use [Run batch inference jobs](#) to make predictions in parallel on batches of observations on an entire dataset. Batch inference is a good option for large datasets or if you don't need an immediate response to a model prediction request.

 **Note**

To avoid incurring unnecessary charges: After the endpoints and resources that were created from model deployment are no longer needed, you can delete them. For information about pricing of instances by Region, see [Amazon SageMaker Pricing](#).

## Deploy models for real-time inference

Real-time inference is ideal for inference workloads where you have real-time, interactive, low latency requirements. This section shows how you can use real-time inferencing to obtain predictions interactively from your model.

To deploy the model that produced the best validation metric in an Autopilot experiment, you have several options. For example, when using Autopilot in SageMaker Studio Classic, you can deploy the model automatically or manually. You can also use SageMaker APIs to manually deploy an Autopilot model.

The following tabs show three options for deploying your model. These instructions assume that you have already created a model in Autopilot. If you don't have a model, see [Create Regression or Classification Jobs for Tabular Data Using the AutoML API](#). To see examples for each option, open each tab.

### Deploy using the Autopilot User Interface (UI)

The Autopilot UI contains helpful dropdown menus, toggles, tooltips, and more to help you navigate through model deployment. You can deploy using either one of the following procedures: Automatic or Manual.

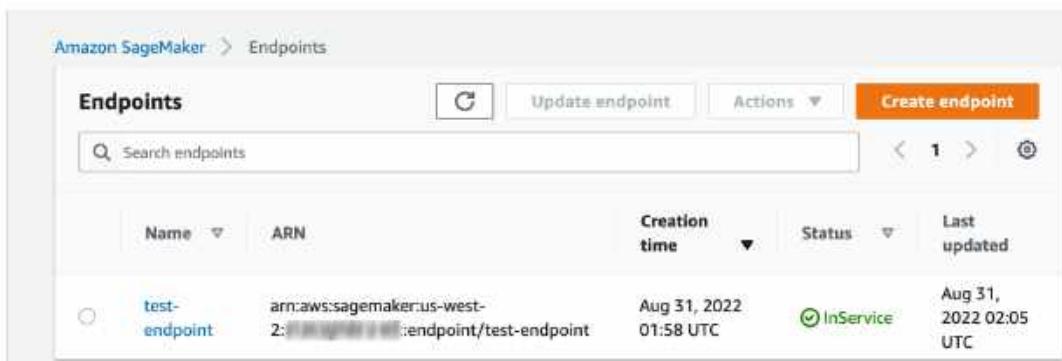
- **Automatic Deployment:** To automatically deploy the best model from an Autopilot experiment to an endpoint

1. [Create an experiment](#) in SageMaker Studio Classic.
2. Toggle the **Auto deploy** value to **Yes**.

### Note

**Automatic deployment will fail if either the default resource quota or your customer quota for endpoint instances in a Region is too limited.** In hyperparameter optimization (HPO) mode, you are required to have at least two ml.m5.2xlarge instances. In ensembling mode, you are required to have at least one ml.m5.12xlarge instance. If you encounter a failure related to quotas, you can [request a service limit increase](#) for SageMaker AI endpoint instances.

- **Manual Deployment:** To manually deploy the best model from an Autopilot experiment to an endpoint
  1. [Create an experiment](#) in SageMaker Studio Classic.
  2. Toggle the **Auto deploy** value to **No**.
  3. Select the model that you want to deploy under **Model name**.
  4. Select the orange **Deployment and advanced settings** button located on the right of the leaderboard. This opens a new tab.
  5. Configure the endpoint name, instance type, and other optional information.
  6. Select the orange **Deploy model** to deploy to an endpoint.
  7. Check the progress of the endpoint creation process in the <https://console.aws.amazon.com/sagemaker/> by navigating to the Endpoints section. That section is located in the **Inference** dropdown menu in the navigation panel.
  8. After the endpoint status changes from **Creating** to **InService**, as shown below, return to Studio Classic and invoke the endpoint.



Name	ARN	Creation time	Status	Last updated
test-endpoint	arn:aws:sagemaker:us-west-2:...:endpoint/test-endpoint	Aug 31, 2022 01:58 UTC	 InService	Aug 31, 2022 02:05 UTC

## Deploy using SageMaker APIs

You can also obtain real-time inference by deploying your model using **API calls**. This section shows the five steps of this process using AWS Command Line Interface (AWS CLI) code snippets.

For complete code examples for both AWS CLI commands and AWS SDK for Python (boto3), open the tabs directly following these steps.

### 1. Obtain candidate definitions

Obtain the candidate container definitions from [InferenceContainers](#). These candidate definitions are used to create a SageMaker AI model.

The following example uses the [DescribeAutoMLJob](#) API to obtain candidate definitions for the best model candidate. See the following AWS CLI command as an example.

```
aws sagemaker describe-auto-ml-job --auto-ml-job-name <job-name> --region <region>
```

### 2. List candidates

The following example uses the [ListCandidatesForAutoMLJob](#) API to list all candidates. See the following AWS CLI command as an example.

```
aws sagemaker list-candidates-for-auto-ml-job --auto-ml-job-name <job-name> --region <region>
```

### 3. Create a SageMaker AI model

Use the container definitions from the previous steps to create a SageMaker AI model by using the [CreateModel](#) API. See the following AWS CLI command as an example.

```
aws sagemaker create-model --model-name '<your-custom-model-name>' \
    --containers ['<container-definition1>, <container-definition2>, <container-definition3>'] \
    --execution-role-arn '<execution-role-arn>' --region '<region>'
```

### 4. Create an endpoint configuration

The following example uses the [CreateEndpointConfig](#) API to create an endpoint configuration. See the following AWS CLI command as an example.

```
aws sagemaker create-endpoint-config --endpoint-config-name '<your-custom-endpoint-config-name>' \
    --production-variants '<list-of-production-variants>' \
    --region '<region>'
```

## 5. Create the endpoint

The following AWS CLI example uses the [CreateEndpoint](#) API to create the endpoint.

```
aws sagemaker create-endpoint --endpoint-name '<your-custom-endpoint-name>' \
    --endpoint-config-name '<endpoint-config-name-you-just-created>' \
    --region '<region>'
```

Check the progress of your endpoint deployment by using the [DescribeEndpoint](#) API. See the following AWS CLI command as an example.

```
aws sagemaker describe-endpoint --endpoint-name '<endpoint-name>' --region <region>
```

After the EndpointStatus changes to InService, the endpoint is ready to use for real-time inference.

## 6. Invoke the endpoint

The following command structure invokes the endpoint for real-time inferencing.

```
aws sagemaker invoke-endpoint --endpoint-name '<endpoint-name>' \
    --region '<region>' --body '<your-data>' [--content-type] \
    '<content-type>' <outfile>'
```

The following tabs contain complete code examples for deploying a model with AWS SDK for Python (boto3) or the AWS CLI.

### AWS SDK for Python (boto3)

#### 1. Obtain the candidate definitions by using the following code example.

```
import sagemaker
import boto3
```

```
session = sagemaker.session.Session()

sagemaker_client = boto3.client('sagemaker', region_name='us-west-2')
job_name = 'test-auto-ml-job'

describe_response = sm_client.describe_auto_ml_job(AutoMLJobName=job_name)
# extract the best candidate definition from DescribeAutoMLJob response
best_candidate = describe_response['BestCandidate']
# extract the InferenceContainers definition from the caandidate definition
inference_containers = best_candidate['InferenceContainers']
```

## 2. Create the model by using the following the code example.

```
# Create Model
model_name = 'test-model'
sagemaker_role = 'arn:aws:iam:44445556666:role/sagemaker-execution-role'
create_model_response = sagemaker_client.create_model(
    ModelName = model_name,
    ExecutionRoleArn = sagemaker_role,
    Containers = inference_containers
)
```

## 3. Create the endpoint configuration by using the following the code example.

```
endpoint_config_name = 'test-endpoint-config'

instance_type = 'ml.m5.2xlarge'
# for all supported instance types, see
# https://docs.aws.amazon.com/sagemaker/latest/APIReference/
API_ProductionVariant.html#sagemaker-Type-ProductionVariant-InstanceType      #
Create endpoint config

endpoint_config_response = sagemaker_client.create_endpoint_config(
    EndpointConfigName=endpoint_config_name,
    ProductionVariants=[
        {
            "VariantName": "variant1",
            "ModelName": model_name,
            "InstanceType": instance_type,
            "InitialInstanceCount": 1
        }
    ]
)
```

```
)  
  
print(f"Created EndpointConfig: {endpoint_config_response['EndpointConfigArn']}")
```

#### 4. Create the endpoint and deploy the model with the following code example.

```
# create endpoint and deploy the model  
endpoint_name = 'test-endpoint'  
create_endpoint_response = sagemaker_client.create_endpoint(  
    EndpointName=endpoint_name,  
  
    EndpointConfigName=endpoint_config_name)  
print(create_endpoint_response)
```

#### Check the status of creating the endpoint by using the following code example.

```
# describe endpoint creation status  
status = sagemaker_client.describe_endpoint(EndpointName=endpoint_name)  
["EndpointStatus"]
```

#### 5. Invoke the endpoint for real-time inferencing by using the following command structure.

```
# once endpoint status is InService, you can invoke the endpoint for inferencing  
if status == "InService":  
    sm_runtime = boto3.Session().client('sagemaker-runtime')  
    inference_result = sm_runtime.invoke_endpoint(EndpointName='test-endpoint',  
    ContentType='text/csv', Body='1,2,3,4,class')
```

## AWS Command Line Interface (AWS CLI)

#### 1. Obtain the candidate definitions by using the following code example.

```
aws sagemaker describe-auto-ml-job --auto-ml-job-name 'test-automl-job' --  
region us-west-2
```

#### 2. Create the model by using the following code example.

```
aws sagemaker create-model --model-name 'test-sagemaker-model'  
--containers '[{
```

```
"Image": "348316444620.dkr.ecr.us-west-2.amazonaws.com/sagemaker-sklearn-  
automl:2.5-1-cpu-py3", amzn-s3-demo-bucket1  
"ModelDataUrl": "s3://amzn-s3-demo-bucket/output/model.tar.gz",  
"Environment": {  
    "AUTOML_SPARSE_ENCODE_RECORDIO_PROTOBUF": "1",  
    "AUTOML_TRANSFORM_MODE": "feature-transform",  
    "SAGEMAKER_DEFAULT_INVOCATIONS_ACCEPT": "application/x-recordio-protobuf",  
    "SAGEMAKER_PROGRAM": "sagemaker_serve",  
    "SAGEMAKER_SUBMIT_DIRECTORY": "/opt/ml/model/code"  
}  
, {  
    "Image": "348316444620.dkr.ecr.us-west-2.amazonaws.com/sagemaker-  
xgboost:1.3-1-cpu-py3",  
    "ModelDataUrl": "s3://amzn-s3-demo-bucket/output/model.tar.gz",  
    "Environment": {  
        "MAX_CONTENT_LENGTH": "20971520",  
        "SAGEMAKER_DEFAULT_INVOCATIONS_ACCEPT": "text/csv",  
        "SAGEMAKER_INFERENCE_OUTPUT": "predicted_label",  
        "SAGEMAKER_INFERENCE_SUPPORTED":  
            "predicted_label,probability,probabilities"  
    }  
, {  
    "Image": "348316444620.dkr.ecr.us-west-2.amazonaws.com/sagemaker-sklearn-  
automl:2.5-1-cpu-py3", aws-region  
    "ModelDataUrl": "s3://amzn-s3-demo-bucket/output/model.tar.gz",  
    "Environment": {  
        "AUTOML_TRANSFORM_MODE": "inverse-label-transform",  
        "SAGEMAKER_DEFAULT_INVOCATIONS_ACCEPT": "text/csv",  
        "SAGEMAKER_INFERENCE_INPUT": "predicted_label",  
        "SAGEMAKER_INFERENCE_OUTPUT": "predicted_label",  
        "SAGEMAKER_INFERENCE_SUPPORTED":  
            "predicted_label,probability,labels,probabilities",  
        "SAGEMAKER_PROGRAM": "sagemaker_serve",  
        "SAGEMAKER_SUBMIT_DIRECTORY": "/opt/ml/model/code"  
    }  
}]\` \\\n--execution-role-arn 'arn:aws:iam::1234567890:role/sagemaker-execution-role' \\\n--region 'us-west-2'
```

For additional details, see [creating a model](#).

The `create_model` command will return a response in the following format.

```
{  
    "ModelArn": "arn:aws:sagemaker:us-west-2:1234567890:model/test-sagemaker-  
model"  
}
```

### 3. Create an endpoint configuration by using the following code example.

```
aws sagemaker create-endpoint-config --endpoint-config-name 'test-endpoint-config'  
\\  
--production-variations '[{"VariantName": "variant1",  
    "ModelName": "test-sagemaker-model",  
    "InitialInstanceCount": 1,  
    "InstanceType": "ml.m5.2xlarge"  
}]\'' \\  
--region us-west-2
```

The `create endpoint configuration` command will return a response in the following format.

```
{  
    "EndpointConfigArn": "arn:aws:sagemaker:us-west-2:1234567890:endpoint-config/  
test-endpoint-config"  
}
```

### 4. Create an endpoint by using the following code example.

```
aws sagemaker create-endpoint --endpoint-name 'test-endpoint' \\  
--endpoint-config-name 'test-endpoint-config' \\  
--region us-west-2
```

The `create endpoint` command will return a response in the following format.

```
{  
    "EndpointArn": "arn:aws:sagemaker:us-west-2:1234567890:endpoint/test-endpoint"  
}
```

Check the progress of the endpoint deployment by using the following [describe-endpoint](#) CLI code example.

```
aws sagemaker describe-endpoint --endpoint-name 'test-endpoint' --region us-west-2
```

The previous progress check will return a response in the following format.

```
{  
    "EndpointName": "test-endpoint",  
    "EndpointArn": "arn:aws:sagemaker:us-west-2:1234567890:endpoint/test-  
    endpoint",  
    "EndpointConfigName": "test-endpoint-config",  
    "EndpointStatus": "Creating",  
    "CreationTime": 1660251167.595,  
    "LastModifiedTime": 1660251167.595  
}
```

After the EndpointStatus changes to InService, the endpoint is ready for use in real-time inference.

## 5. Invoke the endpoint for real-time inferencing by using the following command structure.

```
aws sagemaker-runtime invoke-endpoint --endpoint-name 'test-endpoint' \  
--region 'us-west-2' \  
--body '1,51,3.5,1.4,0.2' \  
--content-type 'text/csv' \  
'/tmp/inference_output'
```

For more options, see [invoking an endpoint](#).

## Deploy models from different accounts

You can deploy an Autopilot model from a different account than the original account that a model was generated in. To implement cross-account model deployment, this section shows how to do the following:

### 1. Grant permission to the deploying account

To assume the role in the generating account, you must grant permission to the deploying account. This allows the deploying account to describe Autopilot jobs in the generating account.

The following example uses a generating account with a trusted `sagemaker-role` entity. The example shows how to give a deploying account with the ID `111122223333` permission to assume the role of the generating account.

```
"Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Service": [
                "sagemaker.amazonaws.com"
            ],
            "AWS": [ "111122223333" ]
        },
        "Action": "sts:AssumeRole"
    }
]
```

The new account with the ID `111122223333` can now assume the role for the generating account.

Next, call the `DescribeAutoMLJob` API from the deploying account to obtain a description of the job created by the generating account.

The following code example describes the model from the deploying account.

```
import sagemaker
import boto3
session = sagemaker.session.Session()

sts_client = boto3.client('sts')
sts_client.assume_role

role = 'arn:aws:iam::111122223333:role/sagemaker-role'
role_session_name = "role-session-name"
_assumed_role = sts_client.assume_role(RoleArn=role,
    RoleSessionName=role_session_name)

credentials = _assumed_role["Credentials"]
access_key = credentials["AccessKeyId"]
secret_key = credentials["SecretAccessKey"]
session_token = credentials["SessionToken"]
```

```
session = boto3.session.Session()

sm_client = session.client('sagemaker', region_name='us-west-2',
                            aws_access_key_id=access_key,
                            aws_secret_access_key=secret_key,
                            aws_session_token=session_token)

# now you can call describe automl job created in account A

job_name = "test-job"
response= sm_client.describe_auto_ml_job(AutoMLJobName=job_name)
```

## 2. Grant access to the deploying account to the model artifacts in the generating account.

The deploying account only needs access to the model artifacts in the generating account to deploy it. These are located in the [S3OutputPath](#) that was specified in the original CreateAutoMLJob API call during model generation.

To give the deploying account access to the model artifacts, choose one of the following options:

- a. [Give access](#) to the ModelDataUrl from the generating account to the deploying account.

Next, you need to give the deploying account permission to assume the role. follow the [real-time inferencing steps](#) to deploy.

- b. [Copy model artifacts](#) from the generating account's original [S3OutputPath](#) to the generating account.

To grant access to the model artifacts, you must define a best\_candidate model and reassign model containers to the new account.

The following example shows how to define a best\_candidate model and reassign the ModelDataUrl.

```
best_candidate = automl.describe_auto_ml_job()['BestCandidate']

# reassigning ModelDataUrl for best_candidate containers below
new_model_locations = ['new-container-1-ModelDataUrl', 'new-container-2-
ModelDataUrl', 'new-container-3-ModelDataUrl']
new_model_locations_index = 0
for container in best_candidate['InferenceContainers']:
```

```
container['ModelDataUrl'] = new_model_locations[new_model_locations_index++]
```

After this assignment of containers, follow the steps in [Deploy using SageMaker APIs](#) to deploy.

To build a payload in real-time inferencing, see the notebook example to [define a test payload](#).

To create the payload from a CSV file and invoke an endpoint, see the **Predict with your model** section in [Create a machine learning model automatically](#).

## Run batch inference jobs

Batch inferencing, also known as offline inferencing, generates model predictions on a batch of observations. Batch inference is a good option for large datasets or if you don't need an immediate response to a model prediction request. By contrast, online inference ([real-time inferencing](#)) generates predictions in real time. You can make batch inferences from an Autopilot model using the [SageMaker Python SDK](#), the Autopilot user interface (UI), the [AWS SDK for Python \(boto3\)](#), or the AWS Command Line Interface ([AWS CLI](#)).

The following tabs show three options for deploying your model: Using APIs, Autopilot UI, or using APIs to deploy from different accounts. These instructions assume that you have already created a model in Autopilot. If you don't have a model, see [Create Regression or Classification Jobs for Tabular Data Using the AutoML API](#). To see examples for each option, open each tab.

## Deploy a model using Autopilot UI

The Autopilot UI contains helpful dropdown menus, toggles, tooltips, and more to help you navigate through model deployment.

The following steps show how to deploy a model from an Autopilot experiment for batch predictions.

1. Sign in at <https://console.aws.amazon.com/sagemaker/> and select **Studio** from the navigation pane.
2. On the left navigation pane, choose **Studio**.
3. Under **Get started**, select the Domain that you want to launch the Studio application in. If your user profile only belongs to one Domain, you do not see the option for selecting a Domain.

4. Select the user profile that you want to launch the Studio Classic application for. If there is no user profile in the domain, choose **Create user profile**. For more information, see [Add user profiles](#).
5. Choose **Launch Studio**. If the user profile belongs to a shared space, choose **Open Spaces**.
6. When the SageMaker Studio Classic console opens, choose the **Launch SageMaker Studio** button.
7. Select **AutoML** from the left navigation pane.
8. Under **Name**, select the Autopilot experiment corresponding to the model that you want to deploy. This opens a new **AUTOPILOT JOB** tab.
9. In the **Model name** section, select the model that you want to deploy.
10. Choose **Deploy model**. This opens a new tab.
11. Choose **Make batch predictions** at the top of the page.
12. For **Batch transform job configuration**, input the **Instance type**, **Instance count** and other optional information.
13. In the **Input data configuration** section, open the dropdown menu.
  - a. For **S3 data type**, choose **ManifestFile** or **S3Prefix**.
  - b. For **Split type**, choose **Line**, **RecordIO**, **TFRecord** or **None**.
  - c. For **Compression**, choose **Gzip** or **None**.
14. For **S3 location**, enter the Amazon S3 bucket location of the input data and other optional information.
15. Under **Output data configuration**, enter the S3 bucket for the output data, and choose how to [assemble the output](#) of your job.
  - a. For **Additional configuration (optional)**, you can enter a MIME type and an **S3 Encryption key**.
16. For **Input/output filtering and data joins (optional)**, you enter a JSONpath expression to filter your input data, join the input source data with your output data, and enter a JSONpath expression to filter your output data.
  - a. For examples for each type of filter, see the [DataProcessing API](#).
17. To perform batch predictions on your input dataset, select **Create batch transform job**. A new **Batch Transform Jobs** tab appears.
18. In the **Batch Transform Jobs** tab: Locate the name of your job in **Status** section. Then check the progress of the job.

## Deploy using SageMaker APIs

To use the SageMaker APIs for batch inferencing, there are three steps:

### 1. Obtain candidate definitions

Candidate definitions from [InferenceContainers](#) are used to create a SageMaker AI model.

The following example shows how to use the [DescribeAutoMLJob](#) API to obtain candidate definitions for the best model candidate. See the following AWS CLI command as an example.

```
aws sagemaker describe-auto-ml-job --auto-ml-job-name <job-name> --region <region>
```

Use the [ListCandidatesForAutoMLJob](#) API to list all candidates. See the following AWS CLI command as an example.

```
aws sagemaker list-candidates-for-auto-ml-job --auto-ml-job-name <job-name> --region <region>
```

### 2. Create a SageMaker AI model

To create a SageMaker AI model using the [CreateModel](#) API, use the container definitions from the previous steps. See the following AWS CLI command as an example.

```
aws sagemaker create-model --model-name '<your-custom-model-name>' \
    --containers ['<container-definition1>, <container-definition2>, <container-definition3>']' \
    --execution-role-arn '<execution-role-arn>' --region '<region>'
```

### 3. Create a SageMaker AI transform job

The following example creates a SageMaker AI transform job with the [CreateTransformJob](#) API. See the following AWS CLI command as an example.

```
aws sagemaker create-transform-job --transform-job-name '<your-custom-transform-job-name>' --model-name '<your-custom-model-name-from-last-step>'\ \
    --transform-input '{ \
        "DataSource": { \
            "S3DataSource": { \
                "S3DataType": "S3Prefix", \
                "S3Uri": "<your-input-data>" \
            } \
        } \
    }'
```

```
    },
    "ContentType": "text/csv",
    "SplitType": "Line"
}' \
--transform-output '{
    "S3OutputPath": "<your-output-path>",
    "AssembleWith": "Line"
}' \
--transform-resources '{
    "InstanceType": "<instance-type>",
    "InstanceCount": 1
}' --region '<region>'
```

Check the progress of your transform job using the [DescribeTransformJob](#) API. See the following AWS CLI command as an example.

```
aws sagemaker describe-transform-job --transform-job-name '<your-custom-transform-job-name>' --region <region>
```

After the job is finished, the predicted result will be available in <your-output-path>.

The output file name has the following format: <input\_data\_file\_name>.out. As an example, if your input file is text\_x.csv, the output name will be text\_x.csv.out.

The following tabs show code examples for SageMaker Python SDK, AWS SDK for Python (boto3), and the AWS CLI.

## SageMaker Python SDK

The following example uses the [SageMaker Python SDK](#) to make predictions in batches.

```
from sagemaker import AutoML

sagemaker_session= sagemaker.session.Session()

job_name = 'test-auto-ml-job' # your autopilot job name
automl = AutoML.attach(auto_ml_job_name=job_name)
output_path = 's3://test-auto-ml-job/output'
input_data = 's3://test-auto-ml-job/test_X.csv'

# call DescribeAutoMLJob API to get the best candidate definition
best_candidate = automl.describe_auto_ml_job()['BestCandidate']
```

```
best_candidate_name = best_candidate['CandidateName']

# create model
model = automl.create_model(name=best_candidate_name,
                             candidate=best_candidate)

# create transformer
transformer = model.transformer(instance_count=1,
                                instance_type='ml.m5.2xlarge',
                                assemble_with='Line',
                                output_path=output_path)

# do batch transform
transformer.transform(data=input_data,
                      split_type='Line',
                      content_type='text/csv',
                      wait=True)
```

## AWS SDK for Python (boto3)

The following example uses **AWS SDK for Python (boto3)** to make predictions in batches.

```
import sagemaker
import boto3

session = sagemaker.Session()

sm_client = boto3.client('sagemaker', region_name='us-west-2')
role = 'arn:aws:iam::1234567890:role/sagemaker-execution-role'
output_path = 's3://test-auto-ml-job/output'
input_data = 's3://test-auto-ml-job/test_X.csv'

best_candidate = sm_client.describe_auto_ml_job(AutoMLJobName=job_name)
['BestCandidate']
best_candidate_containers = best_candidate['InferenceContainers']
best_candidate_name = best_candidate['CandidateName']

# create model
reponse = sm_client.create_model(
    ModelName = best_candidate_name,
    ExecutionRoleArn = role,
    Containers = best_candidate_containers
)
```

```
# Launch Transform Job
response = sm_client.create_transform_job(
    TransformJobName=f'{best_candidate_name}-transform-job',
    ModelName=model_name,
    TransformInput={
        'DataSource': {
            'S3DataSource': {
                'S3DataType': 'S3Prefix',
                'S3Uri': input_data
            }
        },
        'ContentType': "text/csv",
        'SplitType': 'Line'
    },
    TransformOutput={
        'S3OutputPath': output_path,
        'AssembleWith': 'Line',
    },
    TransformResources={
        'InstanceType': 'ml.m5.2xlarge',
        'InstanceCount': 1,
    },
)
)
```

The batch inference job returns a response in the following format.

```
{'TransformJobArn': 'arn:aws:sagemaker:us-west-2:1234567890:transform-job/test-transform-job',
'ResponseMetadata': {'RequestId': '659f97fc-28c4-440b-b957-a49733f7c2f2',
'HTTPStatusCode': 200,
'HTTPHeaders': {'x-amzn-requestid': '659f97fc-28c4-440b-b957-a49733f7c2f2',
'content-type': 'application/x-amz-json-1.1',
'content-length': '96',
'date': 'Thu, 11 Aug 2022 22:23:49 GMT'},
'RetryAttempts': 0}}
```

## AWS Command Line Interface (AWS CLI)

### 1. Obtain the candidate definitions by using the following the code example.

```
aws sagemaker describe-auto-ml-job --auto-ml-job-name 'test-automl-job' --
region us-west-2
```

## 2. Create the model by using the following the code example.

```
aws sagemaker create-model --model-name 'test-sagemaker-model'  
--containers '[{  
    "Image": "348316444620.dkr.ecr.us-west-2.amazonaws.com/sagemaker-sklearn-  
automl:2.5-1-cpu-py3",  
    "ModelDataUrl": "s3://amzn-s3-demo-bucket/out/test-job1/data-processor-models/  
test-job1-dpp0-1-e569ff7ad77f4e55a7e549a/output/model.tar.gz",  
    "Environment": {  
        "AUTOML_SPARSE_ENCODE_RECORDIO_PROTOBUF": "1",  
        "AUTOML_TRANSFORM_MODE": "feature-transform",  
        "SAGEMAKER_DEFAULT_INVOCATIONS_ACCEPT": "application/x-recordio-protobuf",  
        "SAGEMAKER_PROGRAM": "sagemaker_serve",  
        "SAGEMAKER_SUBMIT_DIRECTORY": "/opt/ml/model/code"  
    }  
}, {  
    "Image": "348316444620.dkr.ecr.us-west-2.amazonaws.com/sagemaker-  
xgboost:1.3-1-cpu-py3",  
    "ModelDataUrl": "s3://amzn-s3-demo-bucket/out/test-job1/tuning/flicdf10v2-  
dpp0-xgb/test-job1E9-244-7490a1c0/output/model.tar.gz",  
    "Environment": {  
        "MAX_CONTENT_LENGTH": "20971520",  
        "SAGEMAKER_DEFAULT_INVOCATIONS_ACCEPT": "text/csv",  
        "SAGEMAKER_INFERENCE_OUTPUT": "predicted_label",  
        "SAGEMAKER_INFERENCE_SUPPORTED":  
            "predicted_label,probability,probabilities"  
    }  
}, {  
    "Image": "348316444620.dkr.ecr.us-west-2.amazonaws.com/sagemaker-sklearn-  
automl:2.5-1-cpu-py3",  
    "ModelDataUrl": "s3://amzn-s3-demo-bucket/out/test-job1/data-processor-models/  
test-job1-dpp0-1-e569ff7ad77f4e55a7e549a/output/model.tar.gz",  
    "Environment": {  
        "AUTOML_TRANSFORM_MODE": "inverse-label-transform",  
        "SAGEMAKER_DEFAULT_INVOCATIONS_ACCEPT": "text/csv",  
        "SAGEMAKER_INFERENCE_INPUT": "predicted_label",  
        "SAGEMAKER_INFERENCE_OUTPUT": "predicted_label",  
        "SAGEMAKER_INFERENCE_SUPPORTED":  
            "predicted_label,probability,labels,probabilities",  
            "SAGEMAKER_PROGRAM": "sagemaker_serve",  
            "SAGEMAKER_SUBMIT_DIRECTORY": "/opt/ml/model/code"  
    }  
}]' \
```

```
--execution-role-arn 'arn:aws:iam::1234567890:role/sagemaker-execution-role' \
--region 'us-west-2'
```

### 3. Create the transform job by using the following the code example.

```
aws sagemaker create-transform-job --transform-job-name 'test-transform-job' \
--model-name 'test-sagemaker-model' \
--transform-input '{
    "DataSource": {
        "S3DataSource": {
            "S3DataType": "S3Prefix",
            "S3Uri": "s3://amzn-s3-demo-bucket/data.csv"
        }
    },
    "ContentType": "text/csv",
    "SplitType": "Line"
}' \
--transform-output '{
    "S3OutputPath": "s3://amzn-s3-demo-bucket/output/",
    "AssembleWith": "Line"
}' \
--transform-resources '{
    "InstanceType": "ml.m5.2xlarge",
    "InstanceCount": 1
}' \
--region 'us-west-2'
```

### 4. Check the progress of the transform job by using the following the code example.

```
aws sagemaker describe-transform-job --transform-job-name 'test-transform-job' \
--region us-west-2
```

The following is the response from the transform job.

```
{
    "TransformJobName": "test-transform-job",
    "TransformJobArn": "arn:aws:sagemaker:us-west-2:1234567890:transform-job/test-
transform-job",
    "TransformJobStatus": "InProgress",
    "ModelName": "test-model",
    "TransformInput": {
        "DataSource": {
            "S3DataSource": {
```

```
        "S3DataType": "S3Prefix",
        "S3Uri": "s3://amzn-s3-demo-bucket/data.csv"
    },
},
"ContentType": "text/csv",
"CompressionType": "None",
"SplitType": "Line"
},
"TransformOutput": {
    "S3OutputPath": "s3://amzn-s3-demo-bucket/output/",
    "AssembleWith": "Line",
    "KmsKeyId": ""
},
"TransformResources": {
    "InstanceType": "ml.m5.2xlarge",
    "InstanceCount": 1
},
"CreationTime": 1662495635.679,
"TransformStartTime": 1662495847.496,
"DataProcessing": {
    "InputFilter": "$",
    "OutputFilter": "$",
    "JoinSource": "None"
}
}
```

After the `TransformJobStatus` changes to `Completed`, you can check the inference result in the `S3OutputPath`.

## Deploy models from different accounts

To create a batch inferencing job in a different account than the one that the model was generated in, follow the instructions in [Deploy models from different accounts](#). Then you can create models and transform jobs by following the [Deploy using SageMaker APIs](#).

## View model details

Autopilot generates details about the candidate models that you can obtain. These details include the following:

- A plot of the aggregated SHAP values that indicate the importance of each feature. This helps explain your models predictions.

- The summary statistics for various training and validation metrics, including the objective metric.
- A list of the hyperparameters used to train and tune the model.

To view model details after running an Autopilot job, follow these steps:

1. Choose the **Home** icon  from the left navigation pane to view the top-level **Amazon SageMaker Studio Classic** navigation menu.
2. Select the **AutoML** card from the main working area. This opens a new **Autopilot** tab.
3. In the **Name** section, select the Autopilot job that has the details that you want to examine. This opens a new **Autopilot job** tab.
4. The **Autopilot job** panel lists the metric values including the **Objective** metric for each model under **Model name**. The **Best model** is listed at the top of the list under **Model name** and is also highlighted in the **Models** tab.
  - To review model details, select the model that you are interested in and select **View model details**. This opens a new **Model Details** tab.
5. The **Model Details** tab is divided into four subsections.
  1. The top of the **Explainability** tab contains a plot of aggregated SHAP values that indicate the importance of each feature. Following that are the metrics and hyperparameter values for this model.
  2. The **Performance** tab contains metrics statistics a confusion matrix.
  3. The **Artifacts** tab contains information about model inputs, outputs, and intermediate results.
  4. The **Network** tab summarizes your network isolation and encryption choices.

 **Note**

Feature importance and information in the **Performance** tab is only generated for the **Best model**.

For more information about how the SHAP values help explain predictions based on feature importance, see the whitepaper [Understanding the model explainability](#). Additional information is also available in the [Model Explainability](#) topic in the SageMaker AI Developer Guide.

## View an Autopilot model performance report

An Amazon SageMaker AI model quality report (also referred to as performance report) provides insights and quality information for the best model candidate generated by an AutoML job. This includes information about the job details, model problem type, objective function, and other information related to the problem type. This guide shows how to view Amazon SageMaker Autopilot performance metrics graphically, or view metrics as raw data in a JSON file.

For example, in classification problems, the model quality report includes the following:

- Confusion matrix
- Area under the receiver operating characteristic curve (AUC)
- Information to understand false positives and false negatives
- Tradeoffs between true positives and false positives
- Tradeoffs between precision and recall

Autopilot also provides performance metrics for all of your candidate models. These metrics are calculated using all of the training data and are used to estimate model performance. The main working area includes these metrics by default. The type of metric is determined by the type of problem being addressed.

Refer to the [Amazon SageMaker API reference documentation](#) for the list of available metrics supported by Autopilot.

You can sort your model candidates with the relevant metric to help you select and deploy the model that addresses your business needs. For definitions of these metrics, see the [Autopilot candidate metrics](#) topic.

To view a performance report from an Autopilot job, follow these steps:

1. Choose the **Home** icon



)

from the left navigation pane to view the top-level **Amazon SageMaker Studio Classic** navigation menu.

2. Select the **AutoML** card from the main working area. This opens a new **Autopilot** tab.
3. In the **Name** section, select the Autopilot job that has the details that you want to examine. This opens a new **Autopilot job** tab.
4. The **Autopilot job** panel lists the metric values including the **Objective** metric for each model under **Model name**. The **Best model** is listed at the top of the list under **Model name** and it is highlighted in the **Models** tab.
  - To review model details, select the model that you are interested in and select **View in model details**. This opens a new **Model Details** tab.
5. Choose the **Performance** tab between the **Explainability** and **Artifacts** tab.
  - a. On the top right section of the tab, select the down arrow on the **Download Performance Reports** button.
  - b. The down arrow provides two options to view Autopilot performance metrics:
    - i. You can download a PDF of the performance report to view the metrics graphically.
    - ii. You can view metrics as raw data and download it as a JSON file.

For instructions on how to create and run an AutoML job in SageMaker Studio Classic, see [Create Regression or Classification Jobs for Tabular Data Using the AutoML API](#).

The performance report contains two sections. The first contains details about the Autopilot job that produced the model. The second section contains a model quality report.

### Autopilot Job details

This first section of the report gives some general information about the Autopilot job that produced the model. These job details include the following information:

- Autopilot candidate name
- Autopilot job name
- Problem type

- Objective metric
- Optimization direction

## Model quality report

Model quality information is generated by Autopilot model insights. The report's content that is generated depends on the problem type it addressed: regression, binary classification, or multiclass classification. The report specifies the number of rows that were included in the evaluation dataset and the time at which the evaluation occurred.

## Metrics tables

The first part of the model quality report contains metrics tables. These are appropriate for the type of problem that the model addressed.

The following image is an example of a metrics table that Autopilot generates for a regression problem. It shows the metric name, value, and standard deviation.

### Metrics table

Metric Name	Value	Standard Deviation
<code>mae</code>	5.347324	0.118636
<code>mse</code>	87.874017	4.346468
<code>rmse</code>	9.374114	0.232349
<code>r2</code>	0.924700	0.003710

The following image is an example of a metrics table generated by Autopilot for a multiclass classification problem. It shows the metric name, value, and standard deviation.

## Metrics table

Metric Name	Value	Standard Deviation
<code>weighted_recall</code>	0.597104	0.005410
<code>weighted_precision</code>	0.591693	0.005729
<code>accuracy</code>	0.597104	0.005410
<code>weighted_f0_5</code>	0.592155	0.005659
<code>weighted_f1</code>	0.593423	0.005554
<code>weighted_f2</code>	0.595392	0.005456
<code>accuracy_best_constant_classifier</code>	0.200699	0.004422
<code>weighted_recall_best_constant_classifier</code>	0.200699	0.004422
<code>weighted_precision_best_constant_classifier</code>	0.040280	0.001753
<code>weighted_f0_5_best_constant_classifier</code>	0.047944	0.002039
<code>weighted_f1_best_constant_classifier</code>	0.067094	0.002684
<code>weighted_f2_best_constant_classifier</code>	0.111716	0.003808

## Graphical model performance information

The second part of the model quality report contains graphical information to help you evaluate model performance. The contents of this section depend on the problem type used in modeling.

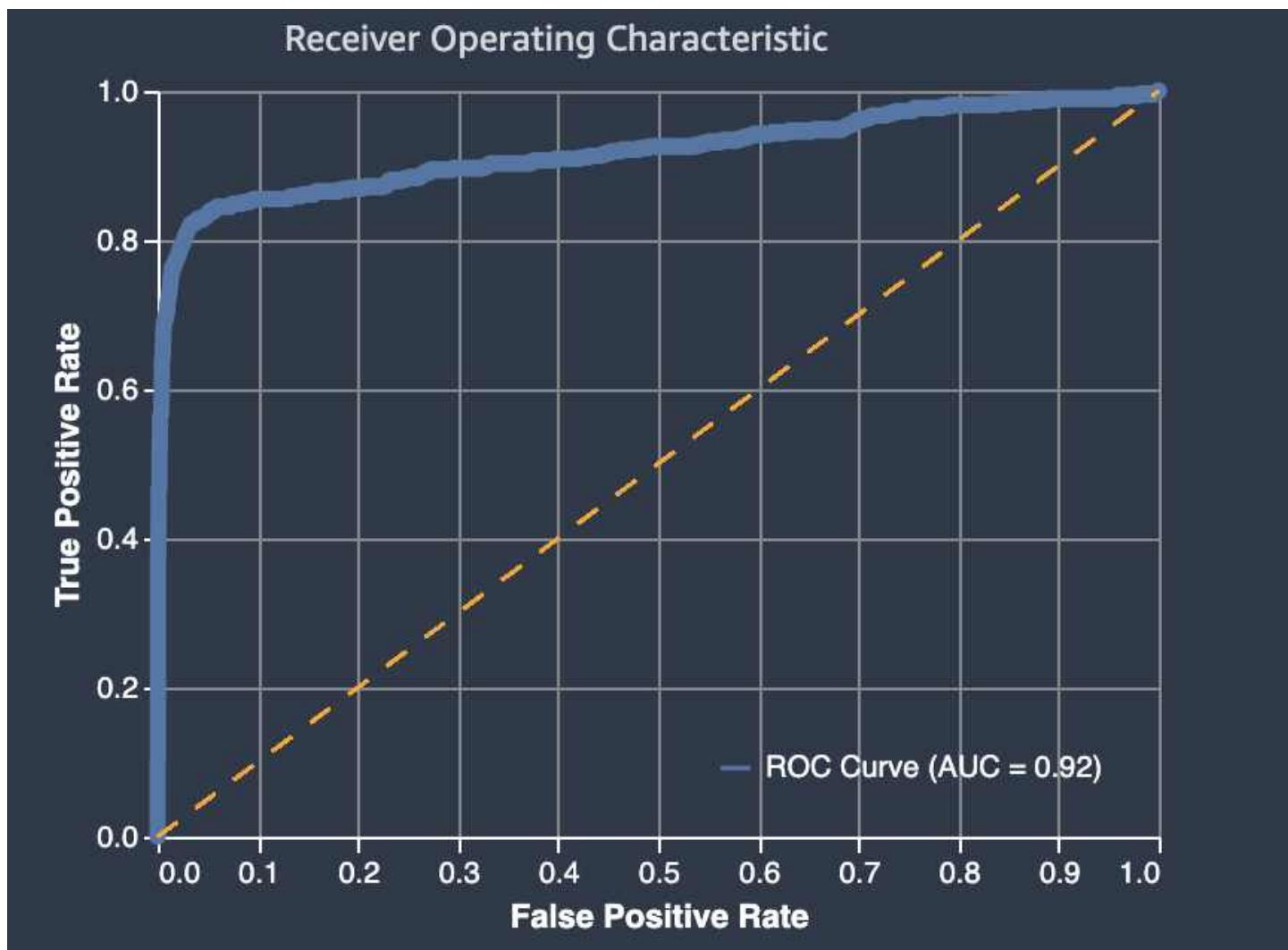
### The area under the receiver operating characteristic curve

The area under the receiver operating characteristic curve represents the trade-off between true positive and false positive rates. It is an industry-standard accuracy metric used for binary classification models. AUC (area under the curve) measures the ability the model to predict a higher score for positive examples, as compared to negative examples. The AUC metric provides an aggregated measure of the model performance across all possible classification thresholds.

The AUC metric returns a decimal value from 0 to 1. AUC values near 1 indicate that the machine learning model is highly accurate. Values near 0.5 indicate that the model is performing no better than guessing at random. AUC values close to 0 indicate that the model has learned the correct patterns, but is making predictions that are as inaccurate as possible. Values near zero can indicate a problem with the data. For more information about the AUC metric, see the [Receiver operating characteristic](#) article on Wikipedia.

The following is an example of an area under the receiver operating characteristic curve graph to evaluate predictions made by a binary classification model. The dashed thin line represents the area under the receiver operating characteristic curve that a model which classifies no-better-than-

random guessing would score, with an AUC score of 0.5. The curves of more accurate classification models lie above this random baseline, where the rate of true positives exceeds the rate of false positives. The area under the receiver operating characteristic curve representing the performance of the binary classification model is the thicker solid line.



A summary of the graph's components of **false positive rate (FPR)** and **true positive rate (TPR)** are defined as follows.

- Correct predictions
  - **True positive (TP)**: The predicted value is 1, and the true value is 1.
  - **True negative (TN)**: The predicted value is 0, and the true value is 0.
- Erroneous predictions
  - **False positive (FP)**: The predicted value is 1, but the true value is 0.
  - **False negative (FN)**: The predicted value is 0, but the true value is 1.

The **false positive rate (FPR)** measures the fraction of true negatives (TN) that were falsely predicted as positives (FP), over the sum of FP and TN. The range is 0 to 1. A smaller value indicates better predictive accuracy.

- $FPR = FP/(FP+TN)$

The **true positive rate (TPR)** measures the fraction true positives that were correctly predicted as positives (TP) over the sum of TP and false negatives (FN). The range is 0 to 1. A larger value indicates better predictive accuracy.

- $TPR = TP/(TP+FN)$

## Confusion matrix

A confusion matrix provides a way to visualize the accuracy of the predictions made by a model for binary and multiclass classification for different problems. The confusion matrix in the model quality report contains the following.

- The number and percentage of correct and incorrect predictions for the actual labels
- The number and percentage of accurate predictions on the diagonal from the upper-left to the lower-right corner
- The number and percentage of inaccurate predictions on the diagonal from the upper-right to the lower-left corner

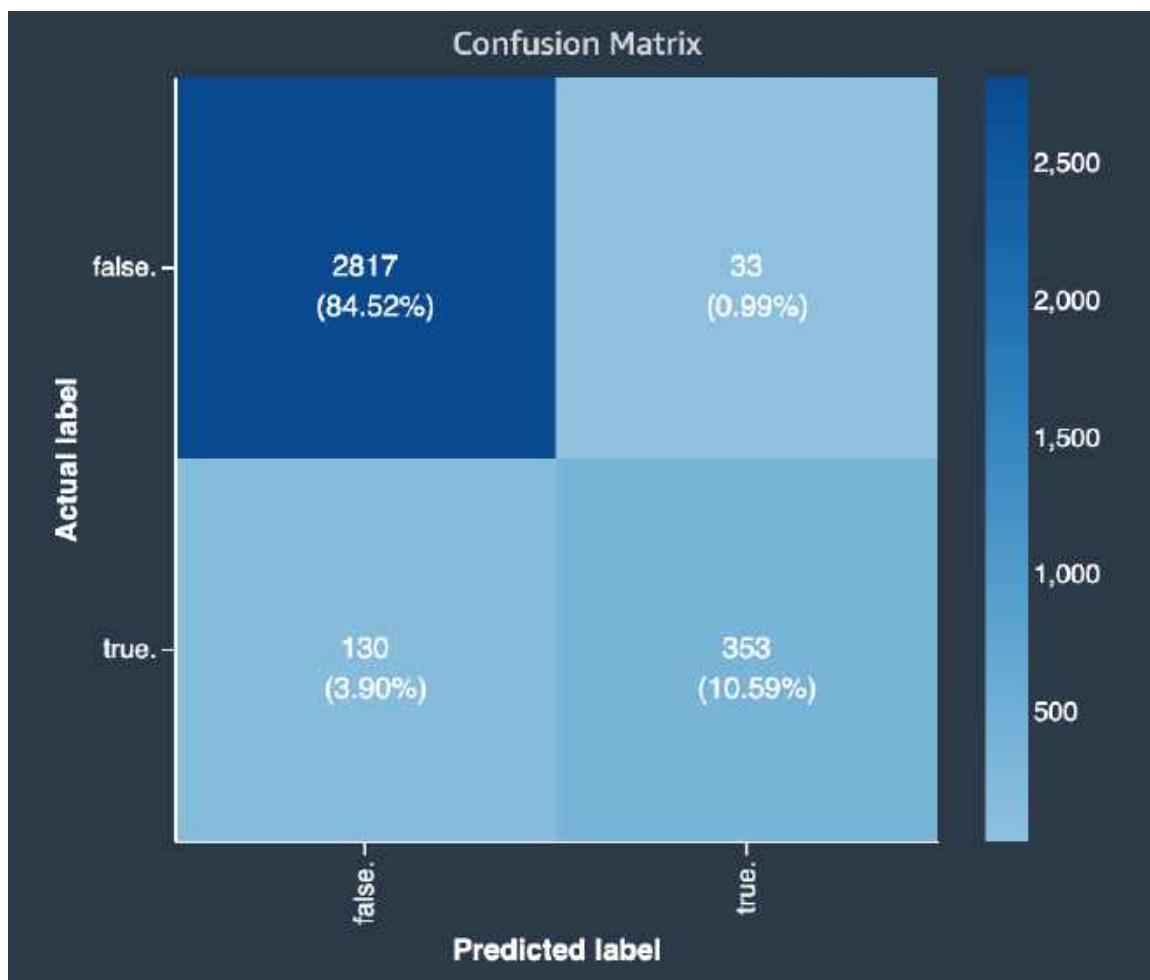
The incorrect predictions on a confusion matrix are the confusion values.

The following diagram is an example of a confusion matrix for a binary classification problem. It contains the following information:

- The vertical axis is divided into two rows containing true and false actual labels.
- The horizontal axis is divided into two columns containing true and false labels that were predicted by the model.
- The color bar assigns a darker tone to a larger number of samples to visually indicate the number of values that were classified in each category.

In this example, the model predicted actual 2817 false values correctly, and 353 actual true values correctly. The model incorrectly predicted 130 actual true values to be false and 33 actual false

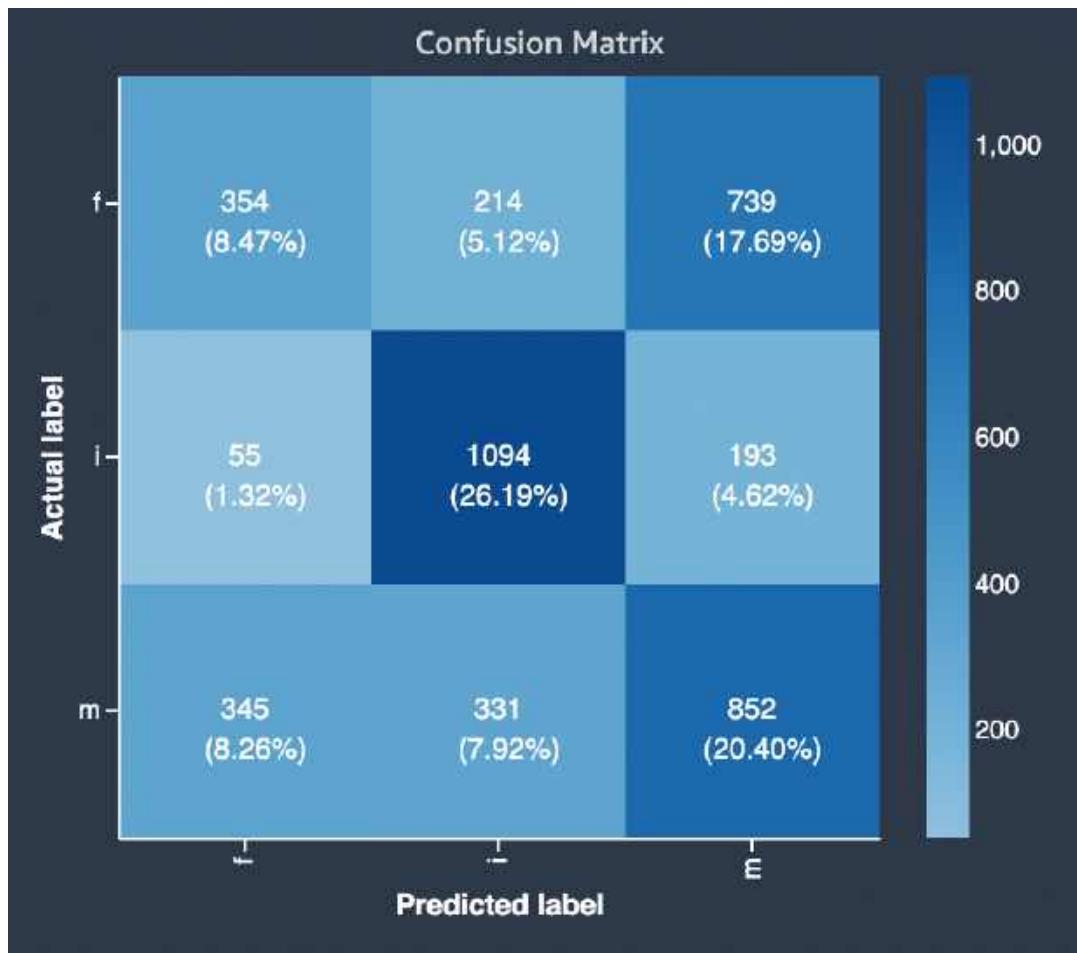
values to be true. The difference in tone indicates that the dataset is not balanced. The imbalance is because there are many more actual false labels than actual true labels.



The following diagram is an example of a confusion matrix for a multi-class classification problem. The confusion matrix in the model quality report contains the following.

- The vertical axis is divided into three rows containing three different actual labels.
- The horizontal axis is divided into three columns containing labels that were predicted by the model.
- The color bar assigns a darker tone to a larger number of samples to visually indicate the number of values that were classified in each category.

In the example below, the model correctly predicted actual 354 values for label **f**, 1094 values for label **i** and 852 values for label **m**. The difference in tone indicates that the dataset is not balanced because there are many more labels for the value **i** than for **f** or **m**.



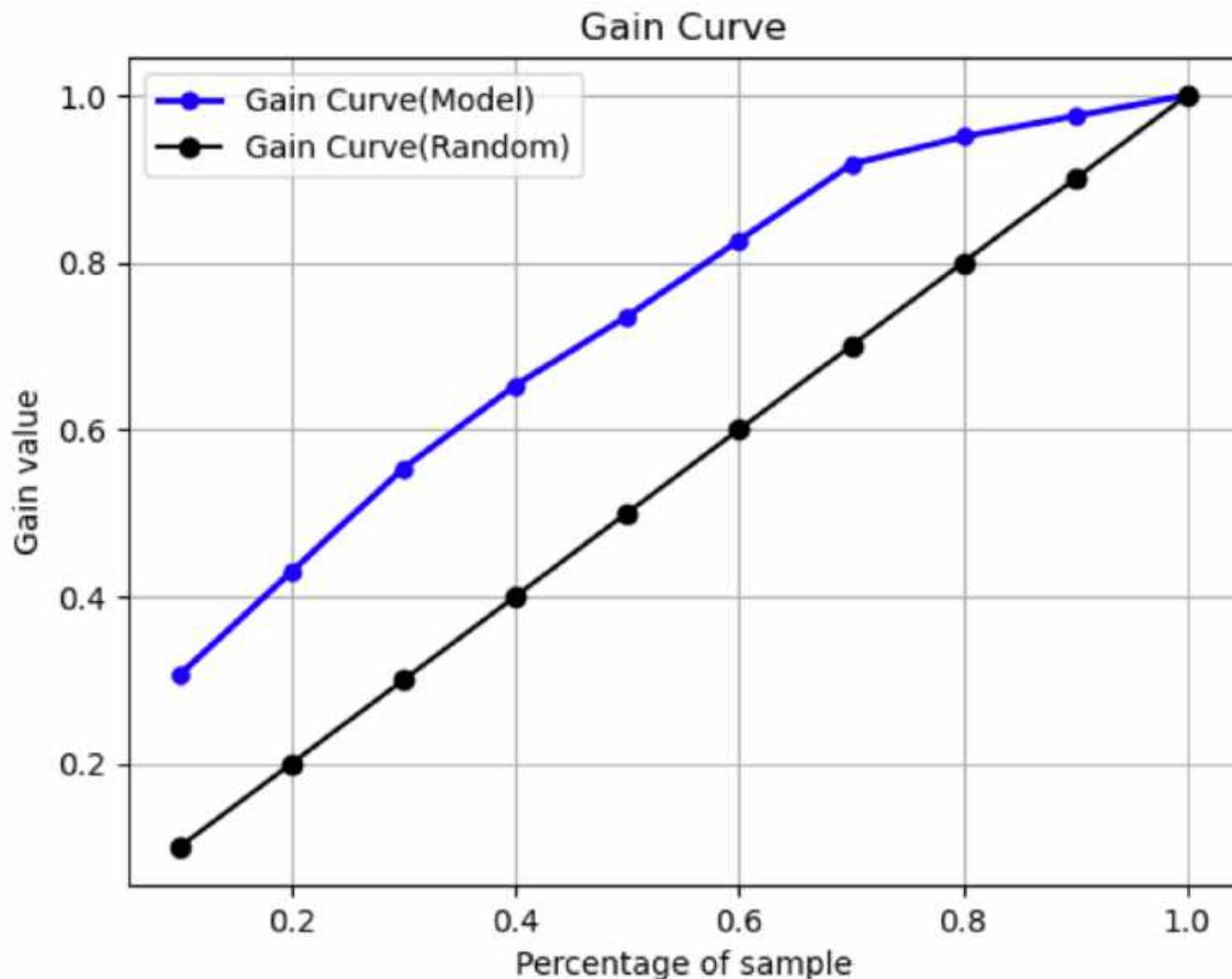
The confusion matrix in the model quality report provided can accommodate a maximum of 15 labels for multiclass classification problem types. If a row corresponding to a label shows a Nan value, it means that the validation dataset used to check model predictions does not contain data with that label.

## Gain curve

In binary classification, a gain curve predicts the cumulative benefit of using a percentage of the dataset to find a positive label. The gain value is calculated during training by dividing the cumulative number of positive observations by the total number of positive observations in the data, at each decile. If the classification model created during training is representative of the unseen data, you can use the gain curve to predict the percentage of data that you must target to obtain a percentage of positive labels. The greater the percentage of the dataset used, the higher the percentage of positive labels found.

In the following example graph, the gain curve is the line with changing slope. The straight line is the percentage of positive labels found by selecting a percentage of data from the dataset at

random. Upon targeting 20% of the dataset, you would expect to find larger than 40% of the positive labels. As an example, you might consider using a gain curve to determine your efforts in a marketing campaign. Using our gain curve example, for 83% of people in a neighborhood to purchase cookies, you'd send an advertisement to about 60% of the neighborhood.

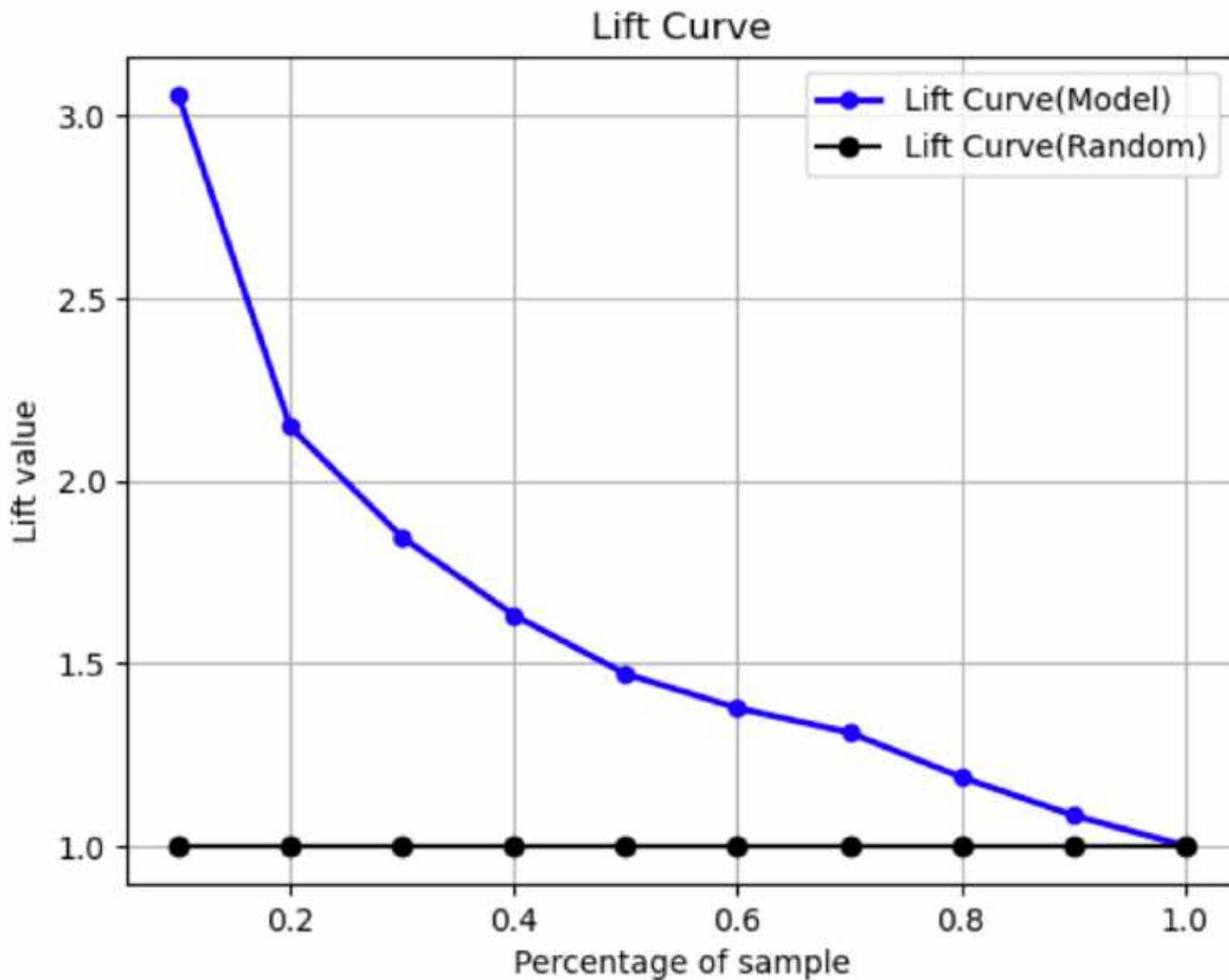


## Lift curve

In binary classification, the lift curve illustrates the uplift of using a trained model to predict the likelihood of finding a positive label compared to a random guess. The lift value is calculated during training using the ratio of percentage gain to the ratio of positive labels at each decile. If the model created during training is representative of the unseen data, use the lift curve to predict the benefit of using the model over randomly guessing.

In the following example graph, the lift curve is the line with changing slope. The straight line is the lift curve associated with selecting the corresponding percentage randomly from the dataset.

Upon targeting 40% of the dataset with your model's classification labels, you would expect to find about 1.7 times the number of the positive labels that you would have found by randomly selecting 40% of the unseen data.



### Precision-recall curve

The precision-recall curve represents the tradeoff between precision and recall for binary classification problems.

**Precision** measures the fraction of actual positives that are predicted as positive (TP) out of all positive predictions (TP and false positive). The range is 0 to 1. A larger value indicates better accuracy in the predicted values.

- Precision =  $TP/(TP+FP)$

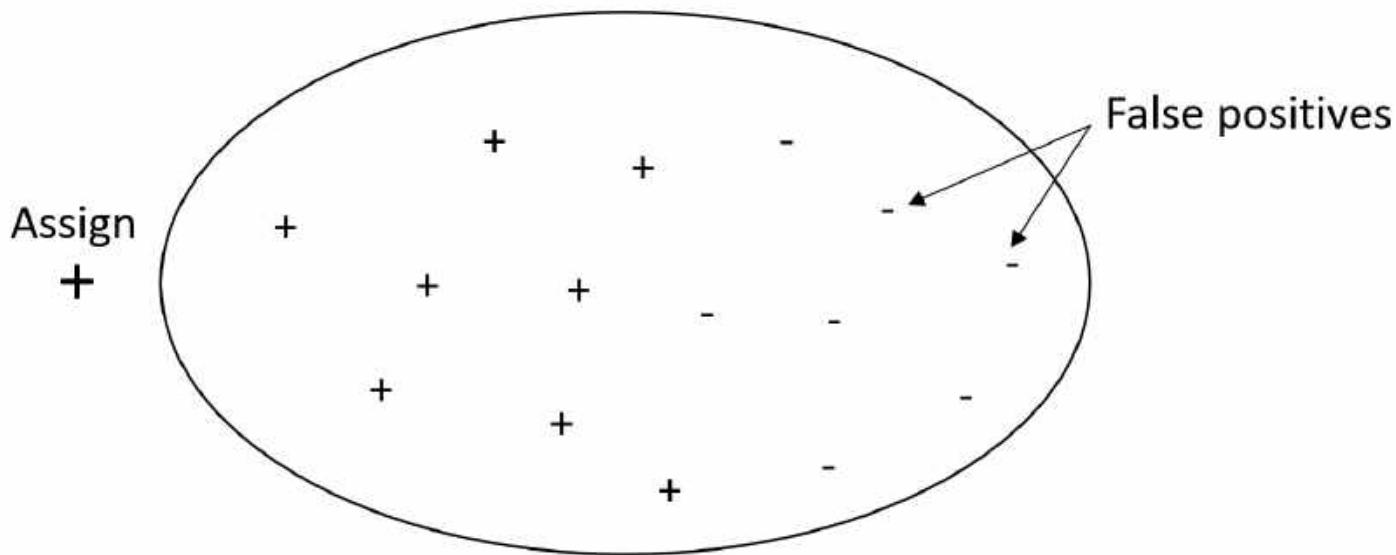
**Recall** measures the fraction of actual positives that are predicted as positive (TP) out of all actual positive predictions (TP and false negative). This is also known as the sensitivity or as the true positive rate. The range is 0 to 1. A larger value indicates better detection of positive values from the sample.

- $\text{Recall} = \text{TP}/(\text{TP}+\text{FN})$

The objective of a classification problem is to correctly label as many elements as possible. A system with high recall but low precision returns a high percentage of false positives.

The following graphic depicts a spam filter that marks every email as spam. It has high recall, but low precision, because recall doesn't measure false positives.

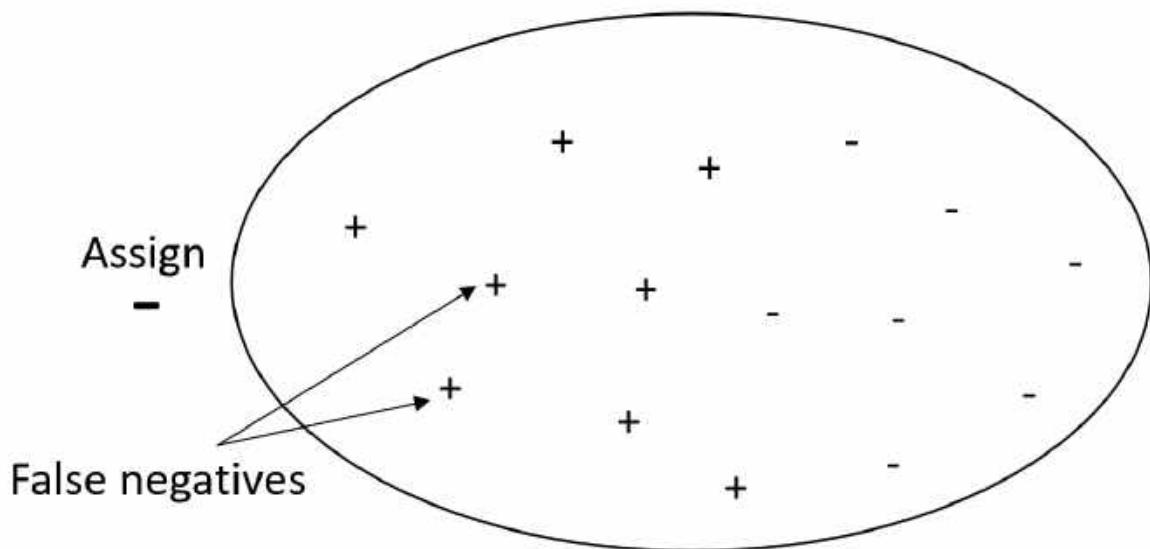
Give more weight to recall over precision if your problem has a low penalty for false positive values, but a high penalty for missing a true positive result. For example, detecting an impending collision in a self-driving vehicle.



By contrast, a system with high precision, but low recall, returns a high percentage of false negatives. A spam filter that marks every email as desirable (not spam) has high precision but low recall because precision doesn't measure false negatives.

If your problem has a low penalty for false negative values, but a high penalty for missing a true negative results, give more weight to precision over recall. For example, flagging a suspicious filter for a tax audit.

The following graphic depicts a spam filter that has high precision but low recall, because precision doesn't measure false negatives.

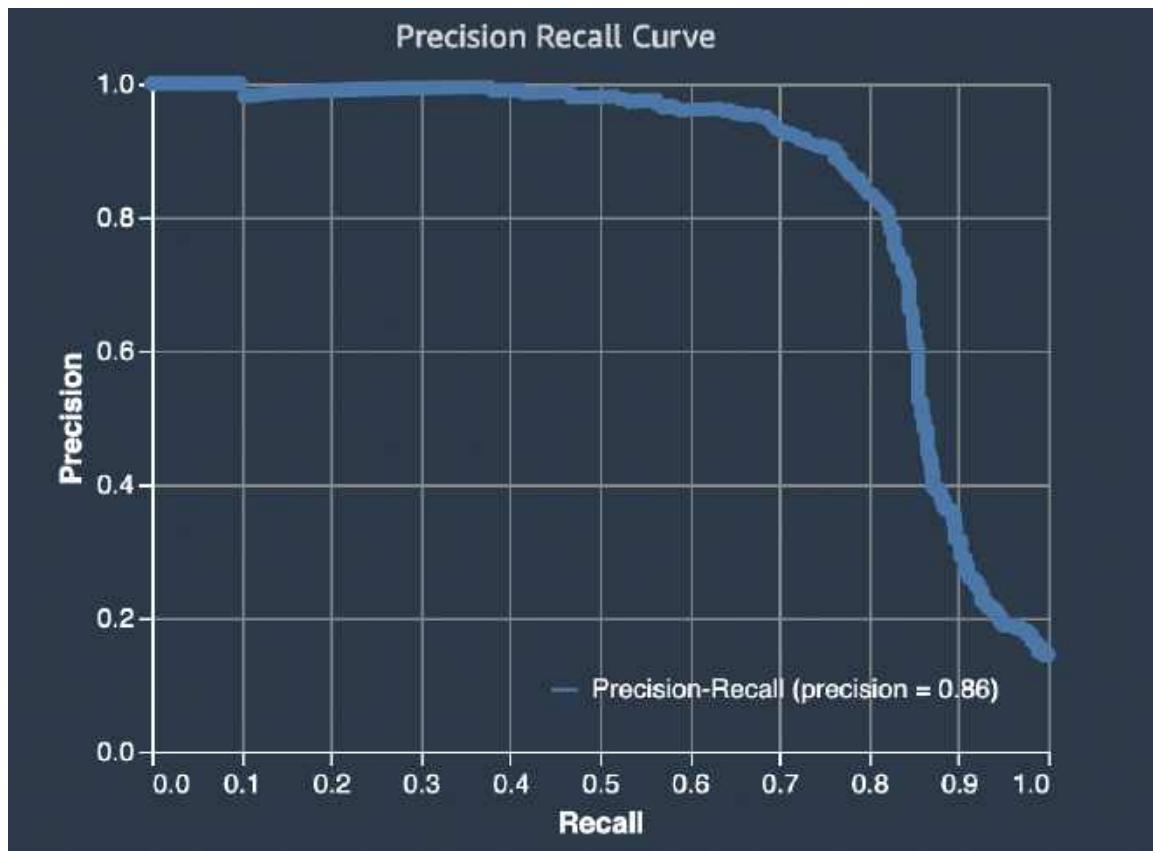


A model that makes predictions with both high precision and high recall produces a high number of correctly labeled results. For more information, see [Precision and recall](#) article in Wikipedia.

### Area under precision-recall curve (AUPRC)

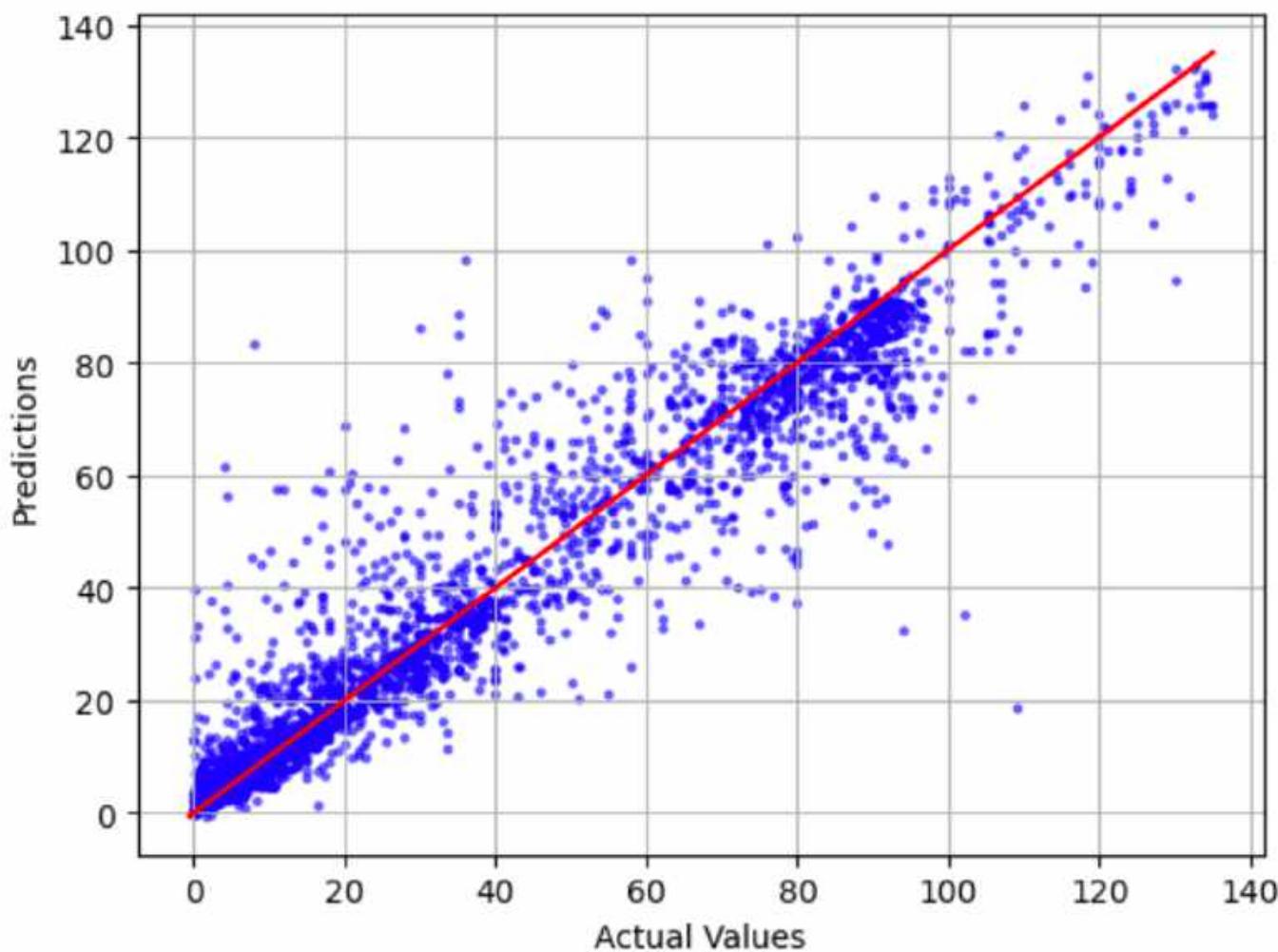
For binary classification problems, Amazon SageMaker Autopilot includes a graph of the area under the precision-recall curve (AUPRC). The AUPRC metric provides an aggregated measure of the model performance across all possible classification thresholds and uses both precision and recall. AUPRC does not take the number of true negatives into account. Therefore, it can be useful to evaluate model performance in cases where there's a large number of true negatives in the data. For example, to model a gene containing a rare mutation.

The following graphic is an example of an AUPRC graph. Precision at its highest value is 1, and recall is at 0. In the lower right corner of the graph, recall is its highest value (1) and precision is 0. In between these two points , the AUPRC curve illustrates the tradeoff between precision and recall at different thresholds.



### Actual against predicted plot

The actual against predicted plot shows the difference between actual and predicted model values. In the following example graph, the solid line is a linear line of best fit. If the model were 100% accurate, each predicted point would equal its corresponding actual point and lie on this line of best fit. The distance away from the line of best fit is a visual indication of model error. The larger the distance away from the line of best fit, the higher the model error.



## Standardized residual plot

A standardized residual plot incorporates the following statistical terms:

### **residual**

A (raw) residual shows the difference between actual and values predicted by your model. The larger the difference, the larger the residual value.

### **standard deviation**

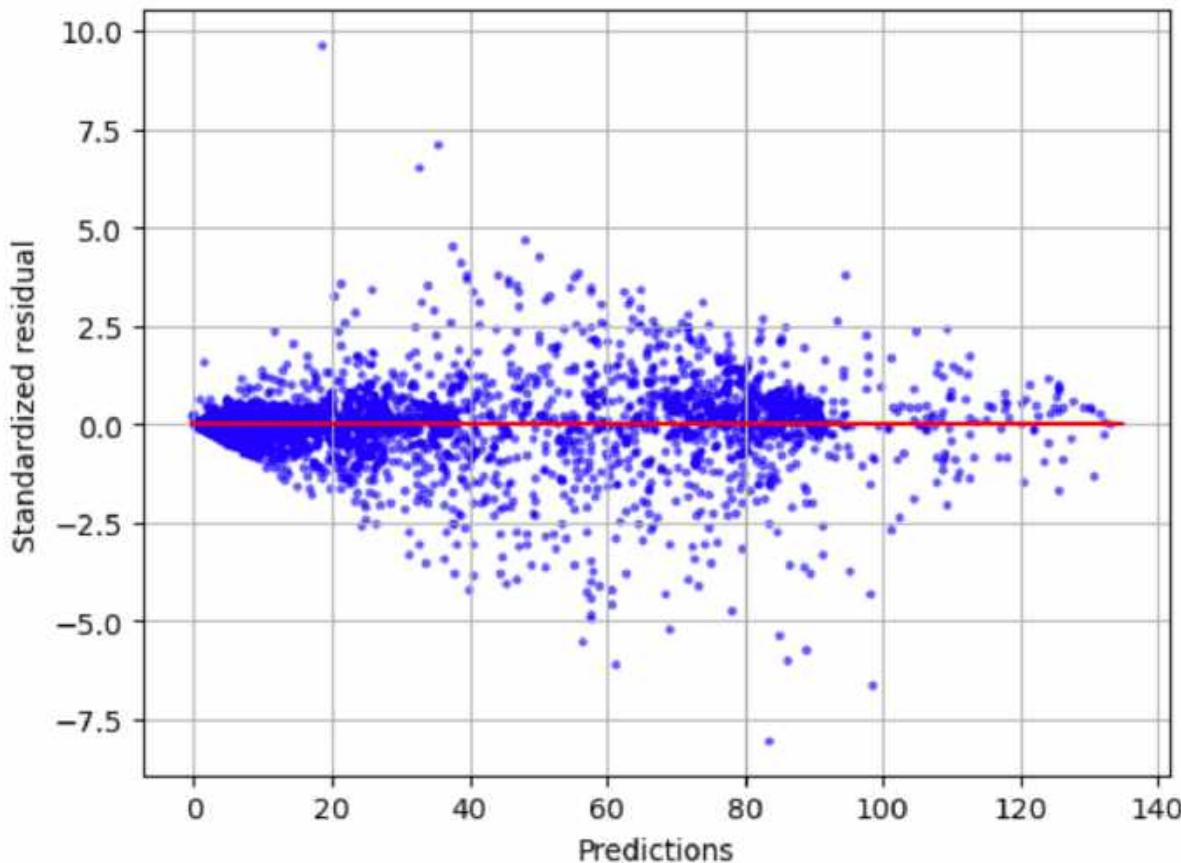
The standard deviation is a measure of how values vary from an average value. A high standard deviation indicates that many values are very different from their average value. A low standard deviation indicates that many values are close to their average value.

## standardized residual

A standardized residual divides the raw residuals by their standard deviation. Standardized residuals have units of standard deviation and are useful in identifying outliers in data regardless of the difference in scale of the raw residuals. If a standardized residual is much smaller or larger than the other standardized residuals, it indicates that the model is not fitting these observations well.

The standardized residual plot measures the strength of the difference between observed and expected values. The actual predicted value is displayed on the x axis. A point with a value larger than an absolute value of 3 is commonly regarded as an outlier.

The following example graph shows that a large number of standardized residuals are clustered around 0 on the horizontal axis. The values close to zero indicate that the model is fitting these points well. The points towards the top and bottom of the plot are not predicted well by the model.



## Residual histogram

A residual histogram incorporates the following statistical terms:

### **residual**

A (raw) residual shows the difference between actual and values predicted by your model. The larger the difference, the larger the residual value.

### **standard deviation**

The standard deviation is a measure of how much values vary from an average value. A high standard deviation indicates that many values are very different from their average value. A low standard deviation indicates that many values are close to their average value.

### **standardized residual**

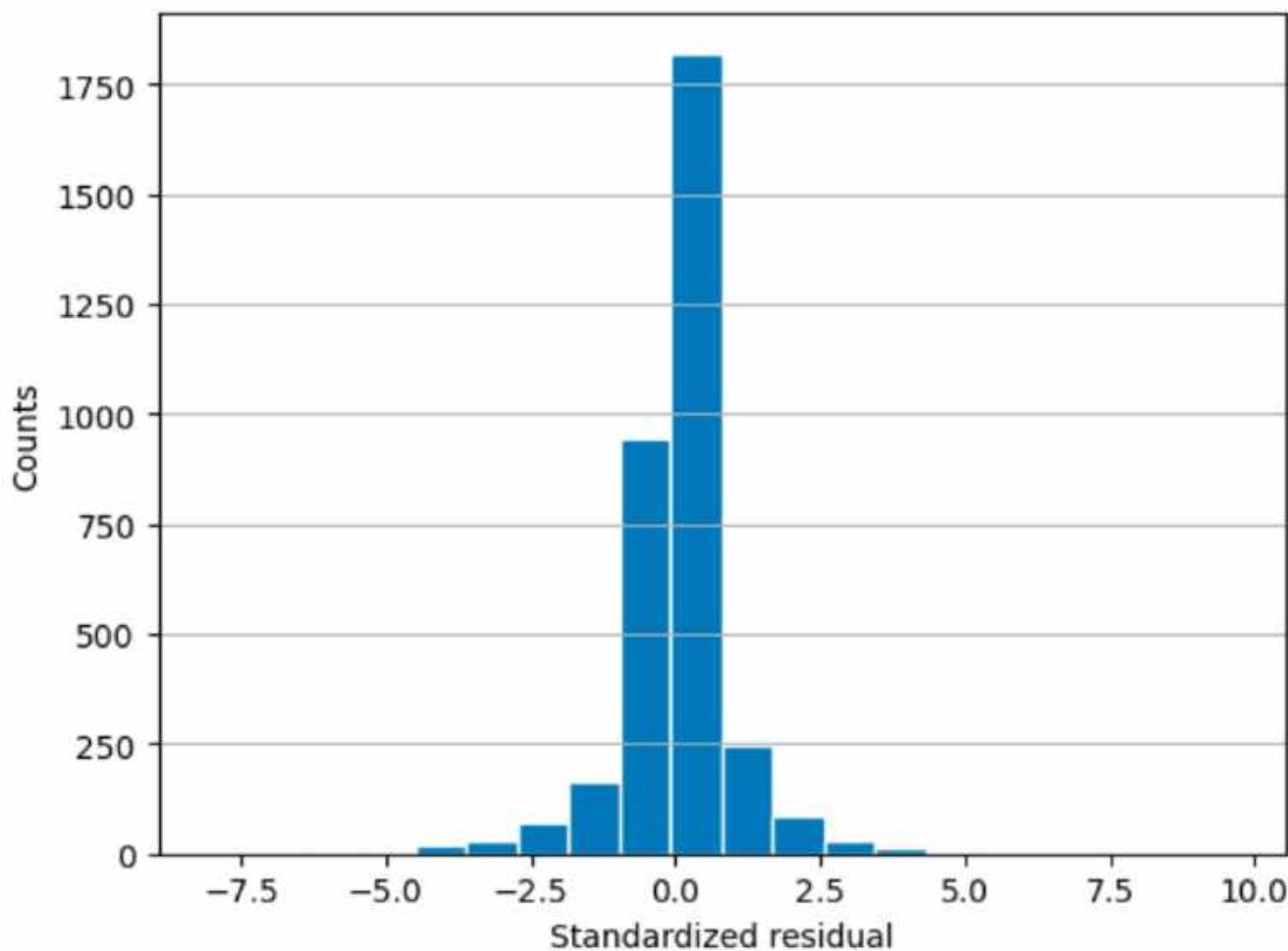
A standardized residual divides the raw residuals by their standard deviation. Standardized residuals have units of standard deviation. These are useful in identifying outliers in data regardless of the difference in scale of the raw residuals. If a standardized residual is much smaller or larger than the other standardized residuals, it would indicate that the model is not fitting these observations well.

### **histogram**

A histogram is a graph that shows how often a value occurred.

The residual histogram shows the distribution of standardized residual values. A histogram distributed in a bell shape and centered at zero indicates that the model does not systematically overpredict or underpredict any particular range of target values.

In the following graphic, the standardized residual values indicate that the model is fitting the data well. If the graph showed values far away from the center value, it would indicate that those values don't fit the model well.



## Autopilot notebooks generated to manage AutoML tasks

Amazon SageMaker Autopilot manages the key tasks in an automatic machine learning (AutoML) process using an AutoML job. The AutoML job creates three notebook-based reports that describe the plan that Autopilot follows to generate candidate models.

A candidate model consists of a (pipeline, algorithm) pair. First, there's a **data exploration** notebook that describes what Autopilot learned about the data that you provided. Second, there's a **candidate definition** notebook, which uses the information about the data to generate candidates. Third, a **model insights** report that can help detail the performance characteristics of the best model in the leaderboard of an Autopilot experiment.

### Topics

- [Autopilot data exploration report](#)
- [Find and run the candidate definition notebook](#)

You can run these notebooks in Amazon SageMaker AI, or locally, if you have installed the [Amazon SageMaker Python SDK](#). You can share the notebooks just like any other SageMaker Studio Classic notebook. The notebooks are created for you to conduct experiments. For example, you could edit the following items in the notebooks:

- Preprocessors used on the data
- Amount of hyperparameter optimization (HPO) runs and their parallelism
- Algorithms to try
- Instance types used for the HPO jobs
- Hyperparameter ranges

Modifications to the candidate definition notebook are encouraged as a learning tool. With this capability, you learn how decisions made during the machine learning process impact your results.

 **Note**

When you run the notebooks in your default instance, you incur baseline costs. However, when you run HPO jobs from the candidate notebook, these jobs use additional compute resources that incur additional costs.

## Autopilot data exploration report

Amazon SageMaker Autopilot cleans and pre-processes your dataset automatically. High-quality data improves machine learning efficiency and produces models that make more accurate predictions.

There are issues with customer-provided datasets that cannot be fixed automatically without the benefit of some domain knowledge. Large outlier values in the target column for regression problems, for example, may cause suboptimal predictions for the non-outlier values. Outliers may need to be removed depending on the modeling objective. If a target column is included by accident as one of the input features, the final model will validate well, but be of little value for future predictions.

To help customers discover these sorts of issues, Autopilot provides a data exploration report that contains insights into potential issues with their data. The report also suggests how to handle the issues.

A data exploration notebook containing the report is generated for every Autopilot job. The report is stored in an Amazon S3 bucket and can be accessed from your output path. The path of the data exploration report usually adheres to the following pattern.

```
[s3 output path]/[name of the automl job]/sagemaker-automl-candidates/  
[name of processing job used for data analysis]/notebooks/SageMaker  
AIAutopilotDataExplorationNotebook.ipynb
```

The location of the data exploration notebook can be obtained from the Autopilot API using the [DescribeAutoMLJob](#) operation response, which is stored in [DataExplorationNotebookLocation](#).

When running Autopilot from SageMaker Studio Classic, you can open the data exploration report using the following steps:

1. Choose the **Home** icon



from the *left navigation pane* to view the top-level **Amazon SageMaker Studio Classic** navigation menu.

2. Select the **AutoML** card from the main working area. This opens a new **Autopilot** tab.
3. In the **Name** section, select the Autopilot job that has the data exploration notebook that you want to examine. This opens a new **Autopilot job** tab.
4. Select **Open data exploration notebook** from the top right section of the **Autopilot job** tab.

The data exploration report is generated from your data before the training process begins. This allows you to stop Autopilot jobs that might lead to meaningless results. Likewise, you can address any issues or improvements with your dataset before rerunning Autopilot. This way, you can use your domain expertise to improve the data quality manually, before you train a model on a better-curated dataset.

The data report contains only static markdown and can be opened in any Jupyter environment. The notebook that contains the report can be converted to other formats, such as PDF or HTML. For more information about conversions, see [Using the nbconvert script to convert Jupyter notebooks to other formats..](#)

## Topics

- [Dataset Summary](#)
- [Target Analysis](#)

- [Data Sample](#)
- [Duplicate rows](#)
- [Cross column correlations](#)
- [Anomalous Rows](#)
- [Missing values, cardinality, and descriptive statistics](#)

## Dataset Summary

This **Dataset Summary** provides key statistics characterizing your dataset including the number of rows, columns, percent duplicate rows and missing target values. It is intended to provide you with a quick alert when there are issue with your dataset that Amazon SageMaker Autopilot has detected and that are likely to require your intervention. The insights are surfaced as warnings that are classified as being of either “high” or “low” severity. The classification depends on the level of confidence that the issue will adversely impact the performance of the model.

The high and low severity insights appear in the summary as pop-ups. For most of the insights, recommendations are offered for how to confirm that there is an issue with the dataset that requires your attention. Proposals are also provided for how to resolve the issues.

Autopilot provides additional statistics about missing or not valid target values in our dataset to help you detect other issues that may not be captured by high severity insights. An unexpected number of columns of a particular type might indicate that some columns that you want to use may be missing from the dataset. It could also indicate that there was an issue with how the data was prepared or stored. Fixing these data problems brought to your attention by Autopilot can improve the performance of the machine learning models trained on your data.

High severity insights are shown in the summary section and in other relevant sections in the report. Examples of high and low-severity insights are usually given depending on the section of the data report.

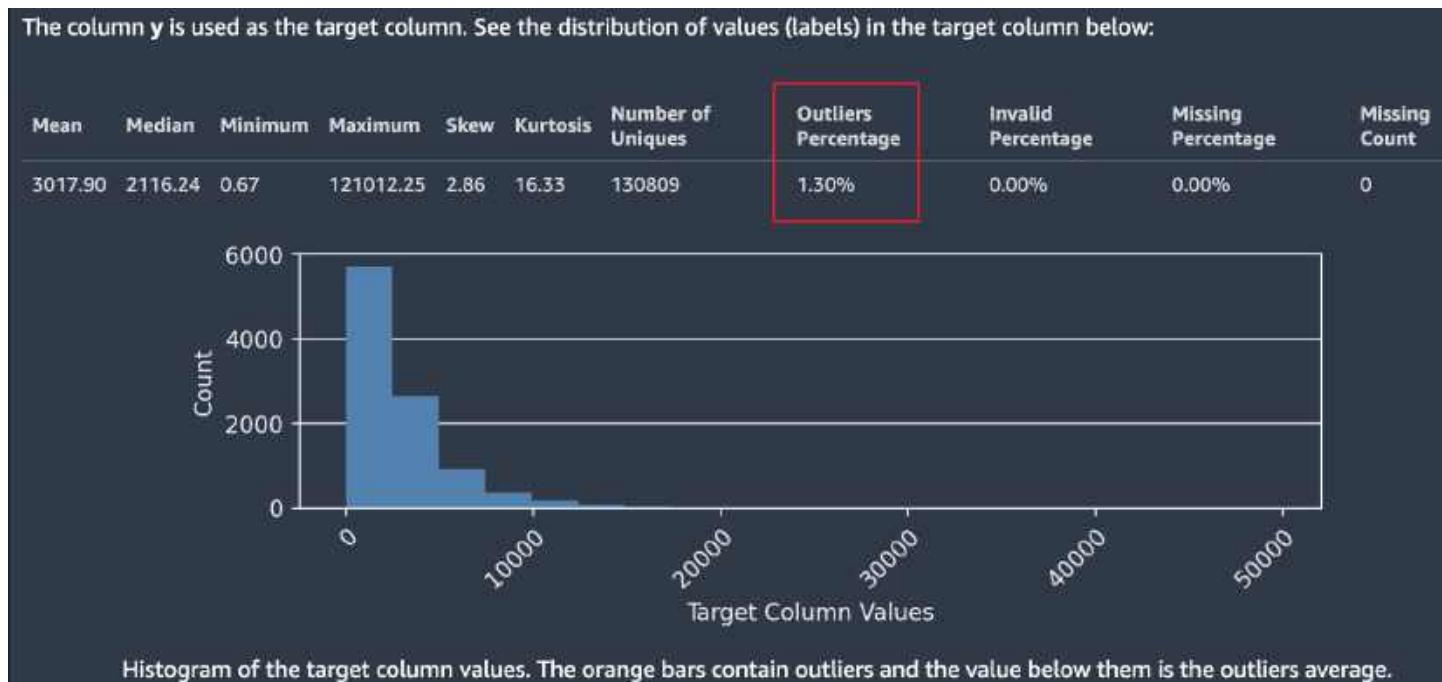
## Target Analysis

Various high and low-severity insights are shown in this section related to the distribution of values in the target column. Check that target column contains the correct values. Incorrect values in target column will likely result in a machine learning model that doesn't serve the intended business purpose. Several data insights of high and low severity are present in this section. Here are several examples.

- **Outlier target values** - Skewed or unusual target distribution for regression, such as heavy tailed targets.
- **High or low target cardinality** - Infrequent number of class labels or a large number of unique classes for classification.

For both regression and classification problem types, not valid values such as numeric infinity, NaN or empty space in target column are surfaced. Depending on the problem type, different dataset statistics are presented. A distribution of target column values for a regression problem allows you to verify if the distribution is what you expected.

The following screenshot shows an Autopilot data report, which includes statistics such as the mean, median, minimum, maximum, percentage of outliers in your dataset. The screenshot also includes a histogram showing the distribution of labels in the target column. The histogram shows **Target Column Values** on the horizontal axis and **Count** on the vertical axis. A box highlights the **Outliers Percentage** section of the screenshot to indicate where this statistic appears.



Multiple statistics are shown regarding target values and their distribution. If any of the outliers, not valid values, or missing percentages are greater than zero, these values are surfaced so you can investigate why your data contains unusable target values. Some unusable target values are highlighted as a low severity insight warning.

In the following screenshot, a ` symbol was added accidentally to the target column, which prevented the numeric value of the target from being parsed. A **Low severity insight: "Invalid**

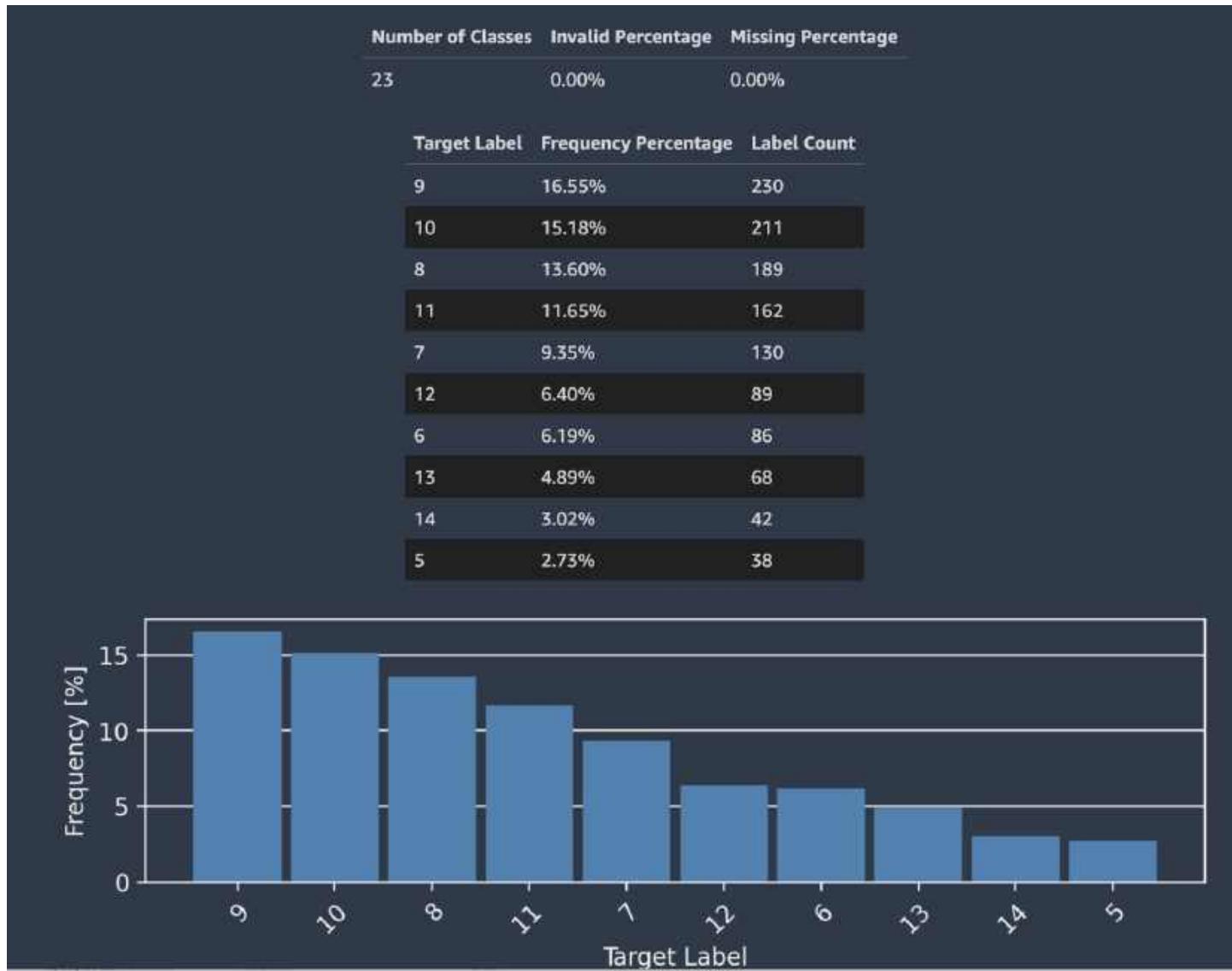
"**target values**" warning appears. The warning in this example states "0.14% of the labels in the target column could not be converted to numeric values. The most common non-numeric values are: `[-3.8e-05, -9e-05, -4.7e-05, -1.499999999999999e-05, -4.3e-05]`. That usually indicates that there are problems with data collection or processing. Amazon SageMaker Autopilot ignores all observations with invalid target label."

**⚠ Low severity insight: "Invalid target values"**

0.14% of the labels in the target column could not be converted to numeric values. The most common non-numeric values are: `[-3.8e-05, -9e-05, -4.7e-05, -1.499999999999999e-05, -4.3e-05]`. That usually indicates that there are problems with data collection or processing. Amazon SageMaker Autopilot ignores all observations with invalid target label.

Autopilot also provides a histogram showing the distribution of labels for classification.

The following screenshot shows an example of statistics given for your target column including the number of classes, missing or not valid values. A histogram with **Target Label** on the horizontal axis and **Frequency** on the vertical axis shows the distribution of each label category.



**Note**

You can find definitions of all the terms presented in this and other sections in **Definitions** section at the bottom of the report notebook.

## Data Sample

Autopilot presents an actual sample of your data to help you spot issues with your dataset. The sample table scrolls horizontally. Inspect the sample data to verify that all the necessary columns are present in the dataset.

Autopilot also calculates a measure of prediction power, that can be used to identify a linear or nonlinear relationship between a feature and the target variable. A value of 0 indicates that the feature has no predictive value in predicting the target variable. A value of 1 indicates the highest predictive power for the target variable. For more information on predictive power, see the [Definitions](#) section.

### Note

It is not recommended that you use prediction power as a substitute for feature importance. Only use it if you're certain that prediction power is an appropriate measure for your use case.

The following screenshot shows example data sample. The top row contains the prediction power of each column in your dataset. The second row contains the column data type. Subsequent rows contain the labels. The columns contain the target column followed by each feature column. Each feature column has an associated prediction power, highlighted in this screenshot, with a box. In this example, the column containing the feature x51 has a predictive power of 0.68 for the target variable y. The feature x55 is slightly less predictive with a prediction power of 0.59.

	y	x51	x55	x54	x52	x20	x56	x15
Prediction Power	-	0.680107	0.594356	0.580346	0.548662	0.543034	0.480431	0.448701
Column Types	-	numeric	numeric	numeric	numeric	numeric	numeric	numeric
0	0.0	0.0	2.0	1.4280000000000002	0.0	0.0	10.0	0.0
1	1.0	0.152	19.0	1.357	0.0	1.18	148.0	0.0
2	1.0	0.0	46.0	4.8180000000000005	0.0	2.63	106.0	1.31
3	0.0	0.134	121.0	3.08	0.0	1.56	693.0	0.0
4	0.0	0.377	1.0	1.0	0.0	0.0	33.0	0.0
5	0.0	0.0	1.0	1.0	0.0	0.0	10.0	0.0
6	0.0	0.327	2.0	1.068	0.0	0.61	47.0	0.0
7	0.0	0.039	6.0	1.2919999999999998	0.0	0.42	106.0	0.21

## Duplicate rows

If duplicate rows are present in the dataset, Amazon SageMaker Autopilot displays a sample of them.

### Note

It is not recommended to balance a dataset by up-sampling before providing it to Autopilot. This may result in inaccurate validation scores for the models trained by Autopilot, and the models that are produced may be unusable.

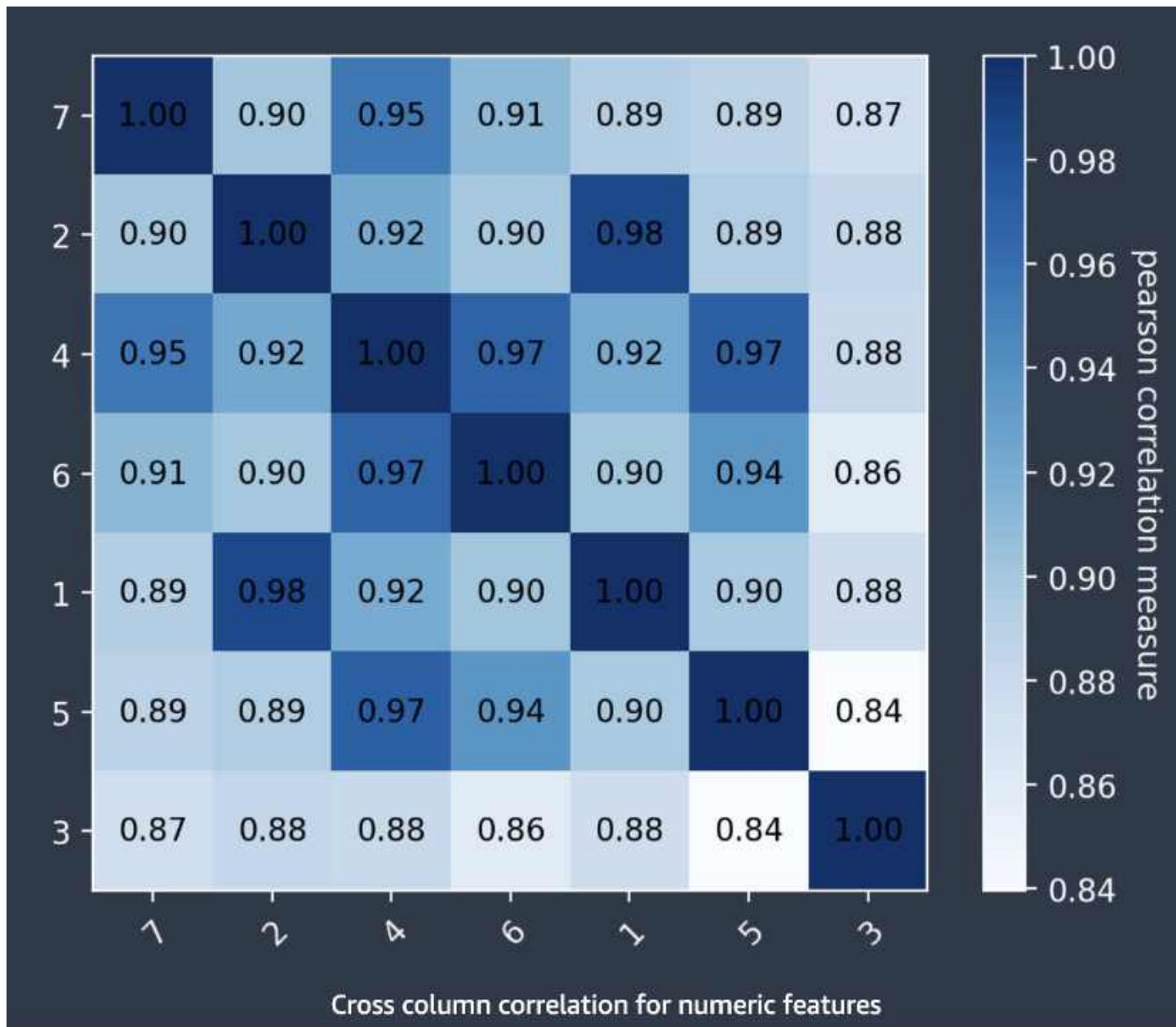
## Cross column correlations

Autopilot uses the Pearson's correlation coefficient, a measure of linear correlation between two features, to populate a correlation matrix. In the correlation matrix, numeric features are plotted on both the horizontal and vertical axes, with the Pearson's correlation coefficient plotted at their intersections. The higher the correlation between two features, the higher the coefficient, with a maximum value of  $|1|$ .

- A value of  $-1$  indicates that the features are perfectly negatively correlated.
- A value of  $1$ , which occurs when a feature is correlated with itself, indicates perfect positive correlation.

You can use the information in the correlation matrix to remove highly correlated features. A smaller number of features reduces chances of overfitting a model and can reduce the costs of production in two ways. It lessens the Autopilot runtime needed and, for some applications, can make data collection procedures cheaper.

The following screenshot shows an example of a correlation matrix between 7 features. Each feature is displayed in a matrix on both the horizontal and vertical axes. The Pearson's correlation coefficient is displayed at the intersection between two features. Each feature intersection has a color tone associated with it. The higher the correlation, the darker the tone. The darkest tones occupy the diagonal of the matrix, where each feature is correlated with itself, representing perfect correlation.



## Anomalous Rows

Amazon SageMaker Autopilot detects which rows in your dataset might be anomalous. It then assigns an anomaly score to each row. Rows with negative anomaly scores are considered anomalous.

The following screenshot shows the output from an Autopilot analysis for rows containing anomalies. A column containing an anomalous score appears next to the dataset columns for each row.

	Anomaly Scores	0	1	2	3	4	5	6	7
1237	-0.215202	F	0.8	0.63	0.195	2.526	0.933	0.59	0.62
405	-0.200257	F	0.815	0.65	0.25	2.255	0.8905	0.42	0.7975
861	-0.194832	F	0.75	0.61	0.235	2.5085	1.232	0.519	0.612
1319	-0.193176	M	0.73	0.595	0.23	2.8255	1.1465	0.419	0.897
403	-0.184558	M	0.77	0.62	0.195	2.5155	1.1155	0.6415	0.642
229	-0.182169	F	0.735	0.6	0.22	2.555	1.1335	0.44	0.6
989	-0.171010	I	0.11	0.09	0.03	0.008	0.0025	0.002	0.003
1066	-0.160921	M	0.665	0.535	0.225	2.1835	0.7535	0.391	0.885
1056	-0.155347	I	0.14	0.105	0.035	0.014	0.0055	0.0025	0.004
637	-0.154234	M	0.175	0.125	0.04	0.024	0.0095	0.006	0.005

## Missing values, cardinality, and descriptive statistics

Amazon SageMaker Autopilot examines and reports on properties of the individual columns of your dataset. In each section of the data report that presents this analysis, the content is arranged in order. This is so you can check the most “suspicious” values first. Using these statistics you can improve contents of individual columns, and improve the quality of the model produced by Autopilot.

Autopilot calculates several statistics on the categorical values in columns that contain them. These include the number of unique entries and, for text, the number of unique words.

Autopilot calculates several standard statistics on the numerical values in columns that contain them. The following image depicts these statistics, including the mean, median, minimum and maximum values, and the percentages of numerical types and of outlier values.

	% of Numerical Values	Mean	Median	Min	Max	% of Outlier Values
y	100.0%	9.93957	9.0	3.0	27.0	nan
1	100.0%	0.523612	0.545	0.11	0.815	0.0
2	100.0%	0.407799	0.425	0.09	0.65	0.0
3	100.0%	0.13995	0.145	0.015	0.515	0.1
4	100.0%	0.828266	0.81	0.008	2.8255	0.0
5	100.0%	0.358844	0.339	0.0025	1.2395	0.0
6	100.0%	0.180348	0.1725	0.002	0.6415	0.0
7	100.0%	0.238783	0.235	0.003	1.005	0.2

## Find and run the candidate definition notebook

The candidate definition notebook contains each suggested preprocessing step, algorithm, and hyperparameter ranges.

You can choose which candidate to train and tune in two ways. The first, by running sections of the notebook. The second, by running the entire notebook to optimize all candidates to identify a best candidate. If you run the entire notebook, only the best candidate is displayed after job completion.

To run Autopilot from SageMaker Studio Classic, open the candidate definition notebook by following these steps:

1. Choose the **Home** icon  
 from the left navigation pane to view the top-level **Amazon SageMaker Studio Classic** navigation menu.
2. Select the **AutoML** card from the main working area. This opens a new **Autopilot** tab.
3. In the **Name** section, select the Autopilot job that has the candidate definition notebook that you want to examine. This opens a new **Autopilot job** tab.

4. Choose **Open candidate generation notebook** from the top right section of the **Autopilot job** tab. This opens a new read-only preview of the **Amazon SageMaker Autopilot Candidate Definition Notebook**.

To run the candidate definition notebook, follow these steps:

1. Choose **Import notebook** at the top right of the **Amazon SageMaker Autopilot Candidate Definition Notebook** tab. This opens a tab to set up a new notebook environment to run the notebook.
2. Select an existing SageMaker **Image** or use a **Custom Image**.
3. Select a **Kernel**, an **Instance type**, and an optional **Start-up script**.

You can now run the notebook in this new environment.

## Configure inference output in generated containers

Autopilot generates an ordered [ContainerDefinition](#) list. This can be used to build a model to deploy in a machine learning pipeline. This model can be used for online hosting and inference.

Customers can list inference container definitions with the [ListCandidateForAutoMLJob](#) API. The list of inference container definitions that represent the best candidate is also available in the [DescribeAutoMLJob](#) response.

### Inference container definitions for regression and classification problem types

Autopilot generates inference containers specific to the [training mode](#) and the [problem type](#) of the job.

### Container definitions for hyperparameter optimization (HPO) mode

- **Regression:** HPO generates two containers:
  1. A feature engineering container that transforms the original features into features that the regression algorithms can train on.
  2. An algorithm container that transforms features and generates a regression score for the dataset.
- **Classification:** HPO generates three containers:
  1. A feature engineering container that transforms the original features into features that the classification algorithms can train on.

2. An algorithm container that generates the `predicted_label` with the highest probability.  
This container can also produce the various probabilities associated with the classification outcomes in the inference response.
3. A feature engineering container that performs post-processing of the algorithm prediction.  
For example, it can perform an inverse transform on the predicted label and change it to the original label.

## Container definitions for ensembling mode

In ensembling mode, both regression and classification problem types have only one inference container. This inference container transforms the features and generates the predictions based on problem type.

## Inference responses per problem type

### Inference responses for classification models

For classification inference containers, you can select the content of the inference response by using four predefined keys:

- `predicted_label`: The label with the highest probability of predicting the correct label, as determined by Autopilot.
- `probability`:
  - **HPO models**: The probability of the True class for binary classification. The probability of the `predicted_label` for multiclass classification.
  - **Ensemble models**: The probability of the `predicted_label` for binary and multiclass classification.
- `probabilities`: The list of probabilities for all corresponding classes.
- `labels`: The list of all labels.

For example, for a binary classification problem, if you pass the inference response keys `['predicted_label', 'probability', 'probabilities', 'labels']` and the output response appears as `[1, 0.1, "[0.9, 0.1]", "[1', '0']"]`, you should interpret it as follows:

1. `predicted_label` equals 1 because label "1" has a higher probability (0.9 in this case).

2. For HPO models, probability equals 0.1 which is the probability of the positive\_class (0 in this case) selected by Autopilot.

For Ensemble models, probability equals 0.9 which is the probability of the predicted\_label.

3. probabilities lists the probability of each label in labels.

4. labels are the unique labels in the dataset, where the second label ("0" in this case) is the positive\_class selected by Autopilot.

By default, inference containers are configured to generate only the predicted\_label. To select additional inference content, you can update the inference\_response\_keys parameter to include up to these three environment variables:

- SAGEMAKER\_INFERENCE\_SUPPORTED: This is set to provide hints to you about what content each container supports.
- SAGEMAKER\_INFERENCE\_INPUT: This should be set to the keys that the container expects in input payload.
- SAGEMAKER\_INFERENCE\_OUTPUT: This should be populated with the set of keys that the container outputs.

## Inference responses for classification models in HPO mode

This section shows how to configure the inference response from classification models using hyperparameter optimization (HPO) mode.

To choose the inference response content in HPO mode: Add the SAGEMAKER\_INFERENCE\_INPUT and SAGEMAKER\_INFERENCE\_OUTPUT variables to the second and third containers that are generated in HPO mode for classification problems.

The keys supported by the second container (algorithm) are predicted\_label, probability, and probabilities. Note that labels is deliberately not added to SAGEMAKER\_INFERENCE\_SUPPORTED.

The keys supported by the third classification model container are predicted\_label, labels, probability, and probabilities. Therefore, the SAGEMAKER\_INFERENCE\_SUPPORTED environment includes the names of these keys.

To update the definition of the inference containers to receive predicted\_label and probability, use the following code example.

```
containers[1]['Environment'].update({'SAGEMAKER_INFERENCE_OUTPUT': 'predicted_label, probability'})  
containers[2]['Environment'].update({'SAGEMAKER_INFERENCE_INPUT': 'predicted_label, probability'})  
containers[2]['Environment'].update({'SAGEMAKER_INFERENCE_OUTPUT': 'predicted_label, probability'})
```

The following code example updates the definition of the inference containers to receive `predicted_label`, `probabilities`, and `labels`. Do not pass the `labels` to the second container (the algorithm container), because it is generated by the third container independently.

```
containers[1]['Environment'].update({'SAGEMAKER_INFERENCE_OUTPUT': 'predicted_label, probabilities'})  
containers[2]['Environment'].update({'SAGEMAKER_INFERENCE_INPUT': 'predicted_label, probabilities'})  
containers[2]['Environment'].update({'SAGEMAKER_INFERENCE_OUTPUT': 'predicted_label, probabilities, labels'})
```

The following collapsible sections provide code examples for AWS SDK for Python (Boto3) and for SageMaker SDK for Python. Each section shows how to select the content of the inference responses in HPO mode for the respective code example.

## AWS SDK for Python (Boto3)

```
import boto3  
  
sm_client = boto3.client('sagemaker', region_name='<Region>')  
  
role = '<IAM role>'  
input_data = '<S3 input uri>'  
output_path = '<S3 output uri>'  
  
best_candidate = sm_client.describe_auto_ml_job(AutoMLJobName='<AutoML Job Name>')[  
    'BestCandidate']  
best_candidate_containers = best_candidate['InferenceContainers'][  
    best_candidate_name = best_candidate['CandidateName']]  
  
best_candidate_containers[1]['Environment'].update({'SAGEMAKER_INFERENCE_OUTPUT':  
    'predicted_label, probability'})  
best_candidate_containers[2]['Environment'].update({'SAGEMAKER_INFERENCE_INPUT':  
    'predicted_label, probability'})
```

```
best_candidate_containers[2]['Environment'].update({'SAGEMAKER_INFERENCE_OUTPUT':  
    'predicted_label, probability'})  
  
# create model  
reponse = sm_client.create_model(  
    ModelName = '<Model Name>',  
    ExecutionRoleArn = role,  
    Containers = best_candidate_containers  
)  
  
# Lauch Transform Job  
response = sm_client.create_transform_job(  
    TransformJobName='<Transform Job Name>',  
    ModelName='<Model Name>',  
    TransformInput={  
        'DataSource': {  
            'S3DataSource': {  
                'S3DataType': 'S3Prefix',  
                'S3Uri': input_data  
            }  
        },  
        'ContentType': "text/CSV",  
        'SplitType': 'Line'  
    },  
    TransformOutput={  
        'S3OutputPath': output_path,  
        'AssembleWith': 'Line',  
    },  
    TransformResources={  
        'InstanceType': 'ml.m4.xlarge',  
        'InstanceCount': 1,  
    },  
)
```

## SageMaker SDK for Python

```
from sagemaker import AutoML  
  
aml = AutoML.attach(auto_ml_job_name='<AutoML Job Name>')  
aml_best_model = aml.create_model(name='<Model Name>',  
                                    candidate=None,  
                                    inference_response_keys**=['probabilities',  
'labels'])
```

```
aml_transformer = aml_best_model.transformer(accept='text/csv',
                                         assemble_with='Line',
                                         instance_type='ml.m5.xlarge',
                                         instance_count=1,)

aml_transformer.transform('<S3 input uri>',
                        content_type='text/csv',
                        split_type='Line',
                        job_name='<Transform Job Name>',
                        wait=True)
```

## Inference responses for classification models in ensembling mode

This section shows how to configure the inference response from classification models using ensembling mode.

In **ensembling mode**, to choose the content of the inference response, update the SAGEMAKER\_INFERENCE\_OUTPUT environment variable.

The keys supported by the classification model container are predicted\_label, labels, probability, and probabilities. These keys are included in the SAGEMAKER\_INFERENCE\_SUPPORTED environment.

To update the inference container definition to receive predicted\_label and probability, refer to the following code example.

```
containers[0]['Environment'].update({'SAGEMAKER_INFERENCE_OUTPUT': 'predicted_label,
probability'})
```

The following collapsible section provides a code example for selecting the content of the inference responses in ensembling mode. The example uses AWS SDK for Python (Boto3).

### AWS SDK for Python (Boto3)

```
import boto3
sm_client = boto3.client('sagemaker', region_name='<Region>')

role = '<IAM role>'
input_data = '<S3 input uri>'
output_path = '<S3 output uri>'
```

```
best_candidate = sm_client.describe_auto_ml_job(AutoMLJobName='<AutoML Job Name>')
['BestCandidate']
best_candidate_containers = best_candidate['InferenceContainers']
best_candidate_name = best_candidate['CandidateName']

*best_candidate_containers[0]['Environment'].update({'SAGEMAKER_INFERENCE_OUTPUT':
    'predicted_label, probability'})
*
# create model
reponse = sm_client.create_model(
    ModelName = '<Model Name>',
    ExecutionRoleArn = role,
    Containers = best_candidate_containers
)

# Launch Transform Job
response = sm_client.create_transform_job(
    TransformJobName='<Transform Job Name>',
    ModelName='<Model Name>',
    TransformInput={
        'DataSource': {
            'S3DataSource': {
                'S3DataType': 'S3Prefix',
                'S3Uri': input_data
            }
        },
        'ContentType': "text/CSV",
        'SplitType': 'Line'
    },
    TransformOutput={
        'S3OutputPath': output_path,
        'AssembleWith': 'Line',
    },
    TransformResources={
        'InstanceType': 'ml.m4.xlarge',
        'InstanceCount': 1,
    },
)
```

The following collapsible section provides a code example that is identical to the SageMaker SDK for Python example for HPO. It is included for your convenience.

## SageMaker SDK for Python

The following HPO code example uses SageMaker SDK for Python.

```
from sagemaker import AutoML

aml = AutoML.attach(auto_ml_job_name='<AutoML Job Name>')
aml_best_model = aml.create_model(name='<Model Name>',
                                  candidate=None,
                                  *inference_response_keys**=['probabilities',
                                  'labels'])

aml_transformer = aml_best_model.transformer(accept='text/csv',
                                            assemble_with='Line',
                                            instance_type='ml.m5.xlarge',
                                            instance_count=1,)

aml_transformer.transform('<S3 input uri>',
                        content_type='text/csv',
                        split_type='Line',
                        job_name='<Transform Job Name>',
                        wait=True)
```

## Create an Image Classification Job using the AutoML API

The following instructions show how to create an Amazon SageMaker Autopilot job as a pilot experiment for image classification problem types using SageMaker [API Reference](#).

### Note

Tasks such as text and image classification, time-series forecasting, and fine-tuning of large language models are exclusively available through the version 2 of the [AutoML REST API](#).

If your language of choice is Python, you can refer to [AWS SDK for Python \(Boto3\)](#) or the [AutoMLV2 object](#) of the Amazon SageMaker Python SDK directly.

Users who prefer the convenience of a user interface can use [Amazon SageMaker Canvas](#) to access pre-trained models and generative AI foundation models, or create custom models tailored for specific text, image classification, forecasting needs, or generative AI.

You can create an Autopilot image classification experiment programmatically by calling the [CreateAutoMLJobV2](#) API action in any language supported by Amazon SageMaker Autopilot or the AWS CLI.

For information on how this API action translates into a function in the language of your choice, see the [See Also](#) section of CreateAutoMLJobV2 and choose an SDK. As an example, for Python users, see the full request syntax of [create\\_auto\\_ml\\_job\\_v2](#) in AWS SDK for Python (Boto3).

The following is a collection of mandatory and optional input request parameters for the CreateAutoMLJobV2 API action used in image classification.

## Required parameters

When calling [CreateAutoMLJobV2](#) to create an Autopilot experiment for image classification, you must provide the following values:

- An [AutoMLJobName](#) to specify the name of your job.
- At least one [AutoMLJobChannel](#) in [AutoMLJobInputDataConfig](#) to specify your data source.
- An [AutoMLProblemTypeConfig](#) of type [ImageClassificationJobConfig](#).
- An [OutputDataConfig](#) to specify the Amazon S3 output path to store the artifacts of your AutoML job.
- A [RoleArn](#) to specify the ARN of the role used to access your data.

All other parameters are optional.

## Optional parameters

The following sections provide details of some optional parameters that you can pass to your image classification AutoML job.

### How to specify the training and validation datasets of an AutoML job

You can provide your own validation dataset and custom data split ratio, or let Autopilot split the dataset automatically.

Each [AutoMLJobChannel](#) object (see the required parameter [AutoMLJobInputDataConfig](#)) has a ChannelType, which can be set to either training or validation values that specify how the data is to be used when building a machine learning model.

At least one data source must be provided and a maximum of two data sources is allowed: one for training data and one for validation data. How you split the data into training and validation datasets depends on whether you have one or two data sources.

How you split the data into training and validation datasets depends on whether you have one or two data sources.

- If you only have **one data source**, the ChannelType is set to training by default and must have this value.
  - If the ValidationFraction value in [AutoMLDataSplitConfig](#) is not set, 0.2 (20%) of the data from this source is used for validation by default.
  - If the ValidationFraction is set to a value between 0 and 1, the dataset is split based on the value specified, where the value specifies the fraction of the dataset used for validation.
- If you have **two data sources**, the ChannelType of one of the AutoMLJobChannel objects must be set to training, the default value. The ChannelType of the other data source must be set to validation. The two data sources must have the same format, either CSV or Parquet, and the same schema. You must not set the value for the ValidationFraction in this case because all of the data from each source is used for either training or validation. Setting this value causes an error.

## How to specify the automatic model deployment configuration for an AutoML job

To enable automatic deployment for the best model candidate of an AutoML job, include a [ModelDeployConfig](#) in the AutoML job request. This will allow the deployment of the best model to a SageMaker AI endpoint. Below are the available configurations for customization.

- To let Autopilot generate the endpoint name, set [AutoGenerateEndpointName](#) to True.
- To provide your own name for the endpoint, set [AutoGenerateEndpointName](#) to False and provide a name of your choice in [EndpointName](#).

## Datasets format and objective metric for image classification

In this section we learn about the available formats for datasets used in image classification as well as the objective metric used to evaluate the predictive quality of machine learning model candidates. The metrics calculated for candidates are specified using an array of [MetricDatum](#) types.

## Datasets formats

Autopilot supports .png, .jpg, and .jpeg image formats. If your dataset contains all .png images use `image/png`, if it contains all .jpg or .jpeg images use `image/jpeg`, and if your dataset contains a mix of image formats use `image/*`.

## Objective metric

The following list contains the names of the metrics that are currently available to measure the performance of models for image classification.

### Accuracy

The ratio of the number of correctly classified items to the total number of (correctly and incorrectly) classified items. Accuracy measures how close the predicted class values are to the actual values. Values for accuracy metrics vary between zero (0) and one (1). A value of 1 indicates perfect accuracy, and 0 indicates perfect inaccuracy.

## Deploy Autopilot models for real-time inference

After you train your Amazon SageMaker Autopilot models, you can set up an endpoint and obtain predictions interactively. The following section describes the steps for deploying your model to a SageMaker AI real-time inference endpoint to get predictions from your model.

### Real-time inferencing

Real-time inference is ideal for inference workloads where you have real-time, interactive, low latency requirements. This section shows how you can use real-time inferencing to obtain predictions interactively from your model.

You can use SageMaker APIs to manually deploy the model that produced the best validation metric in an Autopilot experiment as follows.

Alternatively, you can chose the automatic deployment option when creating your Autopilot experiment. For information on setting up the automatic deployment of models, see [ModelDeployConfig](#) in the request parameters of [CreateAutoMLJobV2](#). This creates an endpoint automatically.

**Note**

To avoid incurring unnecessary charges, you can delete unneeded endpoint and resources created from model deployment. For information about pricing of instances by Region, see [Amazon SageMaker Pricing](#).

## 1. Obtain the candidate container definitions

Obtain the candidate container definitions from [InferenceContainers](#). A container definition for inference refers to the containerized environment designed for deploying and running your trained SageMaker AI model to make predictions.

The following AWS CLI command example uses the [DescribeAutoMLJobV2](#) API to obtain candidate definitions for the best model candidate.

```
aws sagemaker describe-auto-ml-job-v2 --auto-ml-job-name job-name --region region
```

## 2. List candidates

The following AWS CLI command example uses the [ListCandidatesForAutoMLJob](#) API to list all model candidates.

```
aws sagemaker list-candidates-for-auto-ml-job --auto-ml-job-name <job-name> --region <region>
```

## 3. Create a SageMaker AI model

Use the container definitions from the previous steps and a candidate of your choice to create a SageMaker AI model by using the [CreateModel](#) API. See the following AWS CLI command as an example.

```
aws sagemaker create-model --model-name '<your-candidate-name>' \
    --containers ['<container-definition1>, <container-definition2>, <container-definition3>']' \
    --execution-role-arn '<execution-role-arn>' --region '<region>'
```

## 4. Create an endpoint configuration

The following AWS CLI command example uses the [CreateEndpointConfig](#) API to create an endpoint configuration.

```
aws sagemaker create-endpoint-config --endpoint-config-name '<your-endpoint-config-name>' \
    --production-variants '<list-of-production-variants>' \
    --region '<region>'
```

## 5. Create the endpoint

The following AWS CLI example uses the [CreateEndpoint](#) API to create the endpoint.

```
aws sagemaker create-endpoint --endpoint-name '<your-endpoint-name>' \
    --endpoint-config-name '<endpoint-config-name-you-just-created>' \
    --region '<region>'
```

Check the progress of your endpoint deployment by using the [DescribeEndpoint](#) API. See the following AWS CLI command as an example.

```
aws sagemaker describe-endpoint --endpoint-name '<endpoint-name>' --region <region>
```

After the EndpointStatus changes to InService, the endpoint is ready to use for real-time inference.

## 6. Invoke the endpoint

The following command structure invokes the endpoint for real-time inferencing.

```
aws sagemaker invoke-endpoint --endpoint-name '<endpoint-name>' \
    --region '<region>' --body '<your-data>' [--content-type] \
    '<content-type>' <outfile>
```

## Explainability report

Amazon SageMaker Autopilot provides an explainability report to help explain how a best model candidate makes predictions for image classification problems. This report can assist ML engineers, product managers, and other internal stakeholders in understanding the characteristics of the model. Both consumers and regulators rely on transparency in machine learning to trust and

interpret decisions made on model predictions. You can use these explanations for auditing and meeting regulatory requirements, establishing trust in the model, supporting human decision-making, and debugging and improving model performance.

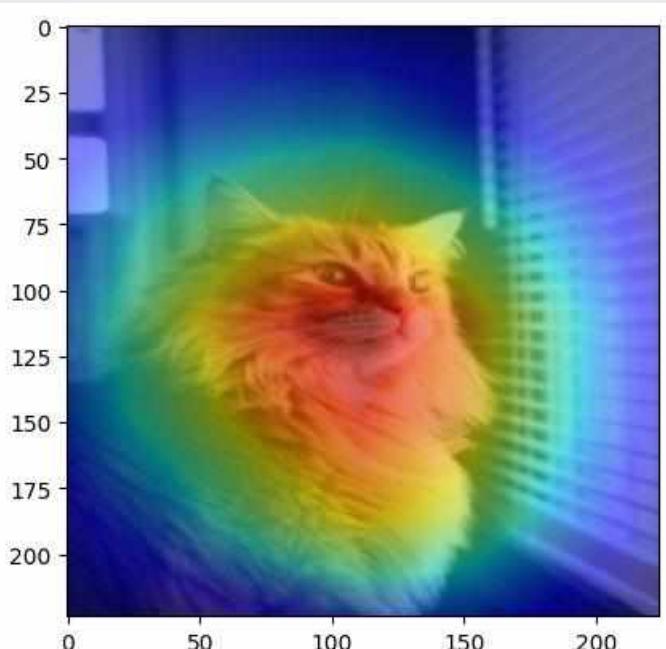
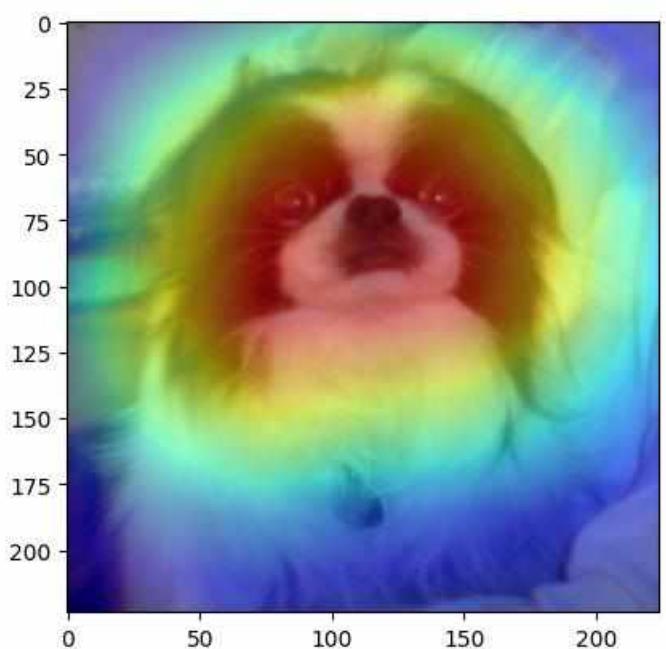
The Autopilot explanatory functionality for image classification uses a visual class activation map (CAM) approach that produces a heatmap where the distribution and intensity of each color highlights the areas of an image that contribute the most to a specific prediction. This approach relies on principal components derived from an implementation of [Eigen-CAM](#).

Autopilot generates the explainability report as a JSON file. The report includes analysis details that are based on the validation dataset. Each image used to generate the report contains the following information:

- `input_image_uri`: The Amazon S3 URI to the input image taken as input for the heatmap.
- `heatmap_image_uri`: The Amazon S3 URI to the heatmap image generated by Autopilot.
- `predicted_label`: The label class predicted by best model trained by Autopilot.
- `probability`: The confidence with which the `predicted_label` is predicted.

You can find the Amazon S3 prefix to the explainability artifacts generated for the best candidate in the response to [DescribeAutoMLJobV2](#) at [BestCandidate.CandidateProperties.CandidateArtifactLocations.Explainability](#).

The following examples illustrates what the heatmaps look like on few samples from [Oxford-IIIT Pet Dataset](#). The heatmap image displays color gradients that indicate the relative importance of different features within the image. The red color represents regions with greater importance in predicting the "predicted\_label" of the input image compared to the features represented by the blue color.

**Input Image****Heatmap Image**

## Model performance report

An Amazon SageMaker AI model quality report (also referred to as performance report) provides insights and quality information for the best model candidate generated by an AutoML job. This includes information about the job details, model problem type, objective function, and various metrics. This section details the content of a performance report for image classification problems and explains how to access the metrics as raw data in a JSON file.

You can find the Amazon S3 prefix to the model quality report artifacts generated for the best candidate in the response to [DescribeAutoMLJobV2](#) at `BestCandidate.CandidateProperties.CandidateArtifactLocations.ModelInsights`.

The performance report contains two sections:

- The first section contains details about the Autopilot job that produced the model.
- The second section contains a model quality report with various performance metrics.

### Autopilot job details

This first section of the report gives some general information about the Autopilot job that produced the model. These details include the following information:

- Autopilot candidate name: The name of the best model candidate.
- Autopilot job name: The name of the job.
- Problem type: The problem type. In our case, *image classification*.
- Objective metric: The objective metric used to optimize the performance of the model. In our case, *Accuracy*.
- Optimization direction: Indicates whether to minimize or maximize the objective metric.

### Model quality report

Model quality information is generated by Autopilot model insights. The report's content that is generated depends on the problem type it addressed. The report specifies the number of rows that were included in the evaluation dataset and the time at which the evaluation occurred.

### Metrics tables

The first part of the model quality report contains metrics tables. These are appropriate for the type of problem that the model addressed.

The following image is an example of a metrics table generates by Autopilot for an image or text classification problem. It shows the metric name, value, and standard deviation.

## Metrics table

Metric Name	Value	Standard Deviation
<code>weighted_recall</code>	0.597104	0.005410
<code>weighted_precision</code>	0.591693	0.005729
<code>accuracy</code>	0.597104	0.005410
<code>weighted_f0_5</code>	0.592155	0.005659
<code>weighted_f1</code>	0.593423	0.005554
<code>weighted_f2</code>	0.595392	0.005456
<code>accuracy_best_constant_classifier</code>	0.200699	0.004422
<code>weighted_recall_best_constant_classifier</code>	0.200699	0.004422
<code>weighted_precision_best_constant_classifier</code>	0.040280	0.001753
<code>weighted_f0_5_best_constant_classifier</code>	0.047944	0.002039
<code>weighted_f1_best_constant_classifier</code>	0.067094	0.002684
<code>weighted_f2_best_constant_classifier</code>	0.111716	0.003808

## Graphical model performance information

The second part of the model quality report contains graphical information to help you evaluate model performance. The contents of this section depend on the selected problem type.

### Confusion matrix

A confusion matrix provides a way to visualize the accuracy of the predictions made by a model for binary and multiclass classification for different problems.

A summary of the graph's components of **false positive rate (FPR)** and **true positive rate (TPR)** are defined as follows.

- Correct predictions
  - **True positive (TP)**: The predicted the value is 1, and the true value is 1.
  - **True negative (TN)**: The predicted the value is 0, and the true value is 0.
- Erroneous predictions
  - **False positive (FP)**: The predicted the value is 1, but the true value is 0.
  - **False negative (FN)**: The predicted the value is 0, but the true value is 1.

The confusion matrix in the model quality report contains the following.

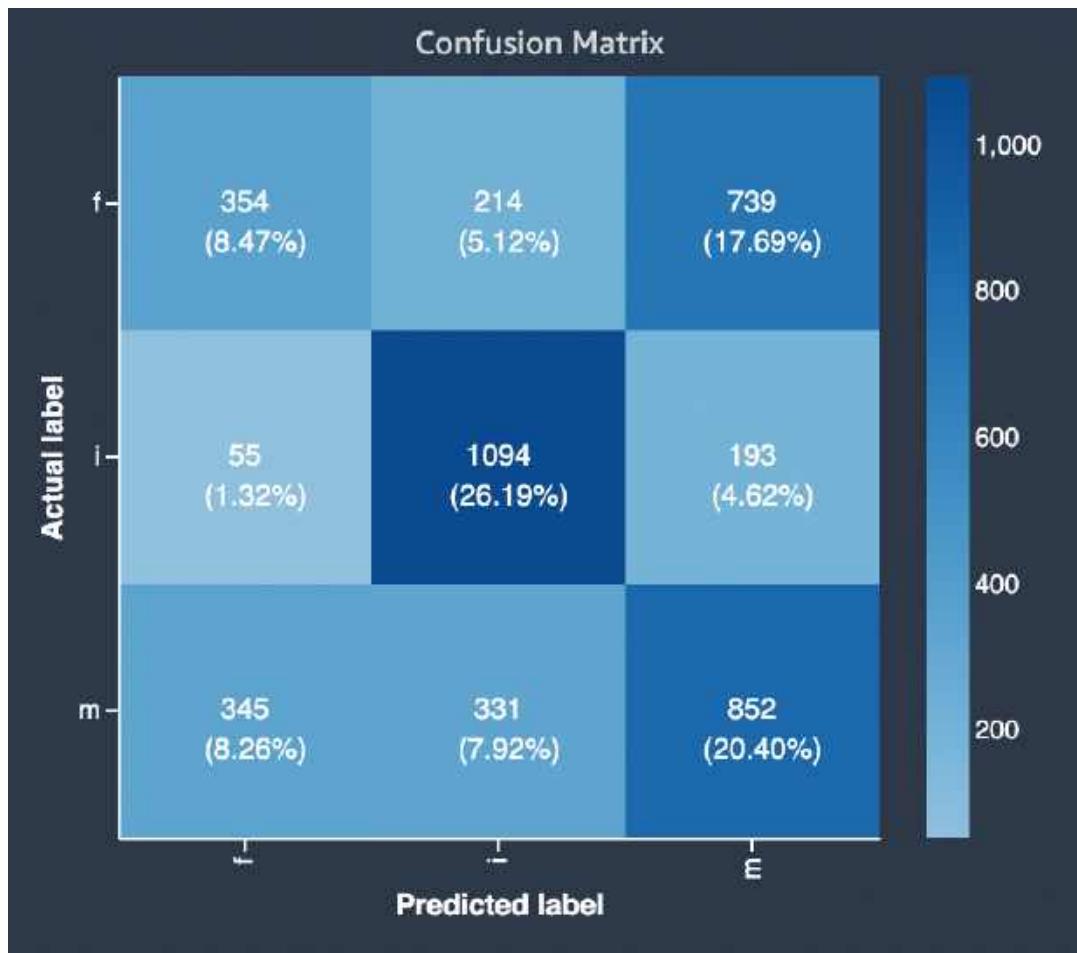
- The number and percentage of correct and incorrect predictions for the actual labels
- The number and percentage of accurate predictions on the diagonal from the upper-left to the lower-right corner
- The number and percentage of inaccurate predictions on the diagonal from the upper-right to the lower-left corner

The incorrect predictions on a confusion matrix are the confusion values.

The following diagram is an example of a confusion matrix for a multi-class classification problem. The confusion matrix in the model quality report contains the following.

- The vertical axis is divided into three rows containing three different actual labels.
- The horizontal axis is divided into three columns containing labels that were predicted by the model.
- The color bar assigns a darker tone to a larger number of samples to visually indicate the number of values that were classified in each category.

In the example below, the model correctly predicted actual 354 values for label **f**, 1094 values for label **i** and 852 values for label **m**. The difference in tone indicates that the dataset is not balanced because there are many more labels for the value **i** than for **f** or **m**.



The confusion matrix in the model quality report provided can accommodate a maximum of 15 labels for multiclass classification problem types. If a row corresponding to a label shows a Nan value, it means that the validation dataset used to check model predictions does not contain data with that label.

## Create an AutoML job for text classification using the API

The following instructions show how to create an Amazon SageMaker Autopilot job as a pilot experiment for text classification problem types using SageMaker [API Reference](#).

### Note

Tasks such as text and image classification, time-series forecasting, and fine-tuning of large language models are exclusively available through the version 2 of the [AutoML REST API](#). If your language of choice is Python, you can refer to [AWS SDK for Python \(Boto3\)](#) or the [AutoMLV2 object](#) of the Amazon SageMaker Python SDK directly.

Users who prefer the convenience of a user interface can use [Amazon SageMaker Canvas](#) to access pre-trained models and generative AI foundation models, or create custom models tailored for specific text, image classification, forecasting needs, or generative AI.

You can create an Autopilot text classification experiment programmatically by calling the [CreateAutoMLJobV2](#) API action in any language supported by Amazon SageMaker Autopilot or the AWS CLI.

For information on how this API action translates into a function in the language of your choice, see the [See Also](#) section of CreateAutoMLJobV2 and choose an SDK. As an example, for Python users, see the full request syntax of [create\\_auto\\_ml\\_job\\_v2](#) in AWS SDK for Python (Boto3).

The following is a collection of mandatory and optional input request parameters for the CreateAutoMLJobV2 API action used in text classification.

## Required parameters

When calling [CreateAutoMLJobV2](#) to create an Autopilot experiment for text classification, you must provide the following values:

- An [AutoMLJobName](#) to specify the name of your job.
- At least one [AutoMLJobChannel](#) in [AutoMLJobInputDataConfig](#) to specify your data source.
- An [AutoMLProblemTypeConfig](#) of type [TextClassificationJobConfig](#).
- An [OutputDataConfig](#) to specify the Amazon S3 output path to store the artifacts of your AutoML job.
- A [RoleArn](#) to specify the ARN of the role used to access your data.

All other parameters are optional.

## Optional parameters

The following sections provide details of some optional parameters that you can pass to your text classification AutoML job.

### How to specify the training and validation datasets of an AutoML job

You can provide your own validation dataset and custom data split ratio, or let Autopilot split the dataset automatically.

Each [AutoMLJobChannel](#) object (see the required parameter [AutoMLJobInputDataConfig](#)) has a ChannelType, which can be set to either training or validation values that specify how the data is to be used when building a machine learning model.

At least one data source must be provided and a maximum of two data sources is allowed: one for training data and one for validation data. How you split the data into training and validation datasets depends on whether you have one or two data sources.

How you split the data into training and validation datasets depends on whether you have one or two data sources.

- If you only have **one data source**, the ChannelType is set to training by default and must have this value.
  - If the ValidationFraction value in [AutoMLDataSplitConfig](#) is not set, 0.2 (20%) of the data from this source is used for validation by default.
  - If the ValidationFraction is set to a value between 0 and 1, the dataset is split based on the value specified, where the value specifies the fraction of the dataset used for validation.
- If you have **two data sources**, the ChannelType of one of the AutoMLJobChannel objects must be set to training, the default value. The ChannelType of the other data source must be set to validation. The two data sources must have the same format, either CSV or Parquet, and the same schema. You must not set the value for the ValidationFraction in this case because all of the data from each source is used for either training or validation. Setting this value causes an error.

## How to specify the automatic model deployment configuration for an AutoML job

To enable automatic deployment for the best model candidate of an AutoML job, include a [ModelDeployConfig](#) in the AutoML job request. This will allow the deployment of the best model to a SageMaker AI endpoint. Below are the available configurations for customization.

- To let Autopilot generate the endpoint name, set [AutoGenerateEndpointName](#) to True.
- To provide your own name for the endpoint, set [AutoGenerateEndpointName](#) to False and provide a name of your choice in [EndpointName](#).

## Datasets format and objective metric for text classification

In this section we learn about the available formats for datasets used in text classification as well as the metric used to evaluate the predictive quality of machine learning model candidates. The metrics calculated for candidates are specified using an array of [MetricDatum](#) types.

### Datasets formats

Autopilot supports tabular data formatted as CSV files or as Parquet files. For tabular data, each column contains a feature with a specific data type and each row contains an observation. The properties of these two file formats differ considerably.

- **CSV** (comma-separated-values) is a row-based file format that stores data in human readable plaintext which a popular choice for data exchange as they are supported by a wide range of applications.
- **Parquet** is a column-based file format where the data is stored and processed more efficiently than row-based file formats. This makes them a better option for big data problems.

The **data types** accepted for columns include numerical, categorical, text.

Autopilot supports building machine learning models on large datasets up to hundreds of GBs. For details on the default resource limits for input datasets and how to increase them, see [Amazon SageMaker Autopilot quotas](#).

### Objective metric

The following list contains the names of the metrics that are currently available to measure the performance of models for text classification.

#### Accuracy

The ratio of the number of correctly classified items to the total number of (correctly and incorrectly) classified items. Accuracy measures how close the predicted class values are to the actual values. Values for accuracy metrics vary between zero (0) and one (1). A value of 1 indicates perfect accuracy, and 0 indicates perfect inaccuracy.

## Deploy Autopilot models for real-time inference

After you train your Amazon SageMaker Autopilot models, you can set up an endpoint and obtain predictions interactively. The following section describes the steps for deploying your model to a SageMaker AI real-time inference endpoint to get predictions from your model.

### Real-time inferencing

Real-time inference is ideal for inference workloads where you have real-time, interactive, low latency requirements. This section shows how you can use real-time inferencing to obtain predictions interactively from your model.

You can use SageMaker APIs to manually deploy the model that produced the best validation metric in an Autopilot experiment as follows.

Alternatively, you can choose the automatic deployment option when creating your Autopilot experiment. For information on setting up the automatic deployment of models, see [ModelDeployConfig](#) in the request parameters of [CreateAutoMLJobV2](#). This creates an endpoint automatically.

#### Note

To avoid incurring unnecessary charges, you can delete unneeded endpoint and resources created from model deployment. For information about pricing of instances by Region, see [Amazon SageMaker Pricing](#).

### 1. Obtain the candidate container definitions

Obtain the candidate container definitions from [InferenceContainers](#). A container definition for inference refers to the containerized environment designed for deploying and running your trained SageMaker AI model to make predictions.

The following AWS CLI command example uses the [DescribeAutoMLJobV2](#) API to obtain candidate definitions for the best model candidate.

```
aws sagemaker describe-auto-ml-job-v2 --auto-ml-job-name job-name --region region
```

### 2. List candidates

The following AWS CLI command example uses the [ListCandidatesForAutoMLJob](#) API to list all model candidates.

```
aws sagemaker list-candidates-for-auto-ml-job --auto-ml-job-name <job-name> --region <region>
```

### 3. Create a SageMaker AI model

Use the container definitions from the previous steps and a candidate of your choice to create a SageMaker AI model by using the [CreateModel](#) API. See the following AWS CLI command as an example.

```
aws sagemaker create-model --model-name '<your-candidate-name>' \  
    --containers ['<container-definition1>, <container-definition2>, <container-definition3>'] \  
    --execution-role-arn '<execution-role-arn>' --region '<region>'
```

### 4. Create an endpoint configuration

The following AWS CLI command example uses the [CreateEndpointConfig](#) API to create an endpoint configuration.

```
aws sagemaker create-endpoint-config --endpoint-config-name '<your-endpoint-config-name>' \  
    --production-variants '<list-of-production-variants>' \  
    --region '<region>'
```

### 5. Create the endpoint

The following AWS CLI example uses the [CreateEndpoint](#) API to create the endpoint.

```
aws sagemaker create-endpoint --endpoint-name '<your-endpoint-name>' \  
    --endpoint-config-name '<endpoint-config-name-you-just-created>' \  
    --region '<region>'
```

Check the progress of your endpoint deployment by using the [DescribeEndpoint](#) API. See the following AWS CLI command as an example.

```
aws sagemaker describe-endpoint --endpoint-name '<endpoint-name>' --region <region>
```

After the EndpointStatus changes to InService, the endpoint is ready to use for real-time inference.

## 6. Invoke the endpoint

The following command structure invokes the endpoint for real-time inferencing.

```
aws sagemaker invoke-endpoint --endpoint-name '<endpoint-name>' \
    --region '<region>' --body '<your-data>' [--content-type]
    '<content-type>' <outfile>
```

## Explainability report

Amazon SageMaker Autopilot provides an explainability report to help explain how a best model candidate makes predictions for text classification problems. This report can assist ML engineers, product managers, and other internal stakeholders in understanding the characteristics of the model. Both consumers and regulators rely on transparency in machine learning to trust and interpret decisions made on model predictions. You can use these explanations for auditing and meeting regulatory requirements, establishing trust in the model, supporting human decision-making, and debugging and improving model performance.

The Autopilot explanatory functionality for text classification uses the axiomatic attribution method *Integrated Gradients*. This approach relies on an implementation of [Axiomatic Attribution for Deep Network](#).

Autopilot generates the explainability report as a JSON file. The report includes analysis details that are based on the validation dataset. Each sample used to generate the report contains the following information:

- **text**: The input text content explained.
- **token\_scores**: The list of scores for every token in the text.
- **attribution**: The score depicting the importance of the token.
  - **description.partial\_text**: The partial substring that represents the token.
- **predicted\_label**: The label class predicted by the best model candidate.
- **probability**: The confidence with which the **predicted\_label** was predicted.

You can find the Amazon S3 prefix to the explainability artifacts generated for the best candidate in the response to [DescribeAutoMLJobV2](#) at [BestCandidate.CandidateProperties.CandidateArtifactLocations.Explainability](#).

The following is an example of analysis content that you could find in the explainability artifacts.

```
{  
    "text": "It was a fantastic movie!",  
    "predicted_label": 2,  
    "probability": 0.9984835,  
    "token_scores": [  
        {  
            "attribution": 0,  
            "description": {  
                "partial_text": "It"  
            }  
        },  
        {  
            "attribution": -0.022447118861679088,  
            "description": {  
                "partial_text": "was"  
            }  
        },  
        {  
            "attribution": -0.2164326456817965,  
            "description": {  
                "partial_text": "a"  
            }  
        },  
        {  
            "attribution": 0.675,  
            "description": {  
                "partial_text": "fantastic"  
            }  
        },  
        {  
            "attribution": 0.416,  
            "description": {  
                "partial_text": "movie!"  
            }  
        }  
    ]  
}
```

In this sample of the JSON report, the explanatory functionality evaluates the text `It was a fantastic movie!` and scores the contribution of each of its token to the overall predicted label. The predicted label is 2, which is a strong positive sentiment, with a probability of 99.85%. The JSON sample then details the contribution of each individual token to this prediction. For example, the token `fantastic` has a stronger attribution than the token `was`. It is the token that contributed the most to the final prediction.

## Model performance report

An Amazon SageMaker AI model quality report (also referred to as performance report) provides insights and quality information for the best model candidate generated by an AutoML job. This includes information about the job details, model problem type, objective function, and various metrics. This section details the content of a performance report for text classification problems and explains how to access the metrics as raw data in a JSON file.

You can find the Amazon S3 prefix to the model quality report artifacts generated for the best candidate in the response to [DescribeAutoMLJobV2](#) at [`BestCandidate.CandidateProperties.CandidateArtifactLocations.ModelInsights`](#).

The performance report contains two sections:

- The first section contains details about the Autopilot job that produced the model.
- The second section contains a model quality report with various performance metrics.

### Autopilot job details

This first section of the report gives some general information about the Autopilot job that produced the model. These details include the following information:

- Autopilot candidate name: The name of the best model candidate.
- Autopilot job name: The name of the job.
- Problem type: The problem type. In our case, *text classification*.
- Objective metric: The objective metric used to optimize the performance of the model. In our case, *Accuracy*.
- Optimization direction: Indicates whether to minimize or maximize the objective metric.

## Model quality report

Model quality information is generated by Autopilot model insights. The report's content that is generated depends on the problem type it addressed. The report specifies the number of rows that were included in the evaluation dataset and the time at which the evaluation occurred.

### Metrics tables

The first part of the model quality report contains metrics tables. These are appropriate for the type of problem that the model addressed.

The following image is an example of a metrics table generated by Autopilot for an image or text classification problem. It shows the metric name, value, and standard deviation.

#### Metrics table

Metric Name	Value	Standard Deviation
<code>weighted_recall</code>	0.597104	0.005410
<code>weighted_precision</code>	0.591693	0.005729
<code>accuracy</code>	0.597104	0.005410
<code>weighted_f0_5</code>	0.592155	0.005659
<code>weighted_f1</code>	0.593423	0.005554
<code>weighted_f2</code>	0.595392	0.005456
<code>accuracy_best_constant_classifier</code>	0.200699	0.004422
<code>weighted_recall_best_constant_classifier</code>	0.200699	0.004422
<code>weighted_precision_best_constant_classifier</code>	0.040280	0.001753
<code>weighted_f0_5_best_constant_classifier</code>	0.047944	0.002039
<code>weighted_f1_best_constant_classifier</code>	0.067094	0.002684
<code>weighted_f2_best_constant_classifier</code>	0.111716	0.003808

### Graphical model performance information

The second part of the model quality report contains graphical information to help you evaluate model performance. The contents of this section depend on the selected problem type.

### Confusion matrix

A confusion matrix provides a way to visualize the accuracy of the predictions made by a model for binary and multiclass classification for different problems.

A summary of the graph's components of **false positive rate (FPR)** and **true positive rate (TPR)** are defined as follows.

- Correct predictions
  - **True positive (TP)**: The predicted the value is 1, and the true value is 1.
  - **True negative (TN)**: The predicted the value is 0, and the true value is 0.
- Erroneous predictions
  - **False positive (FP)**: The predicted the value is 1, but the true value is 0.
  - **False negative (FN)**: The predicted the value is 0, but the true value is 1.

The confusion matrix in the model quality report contains the following.

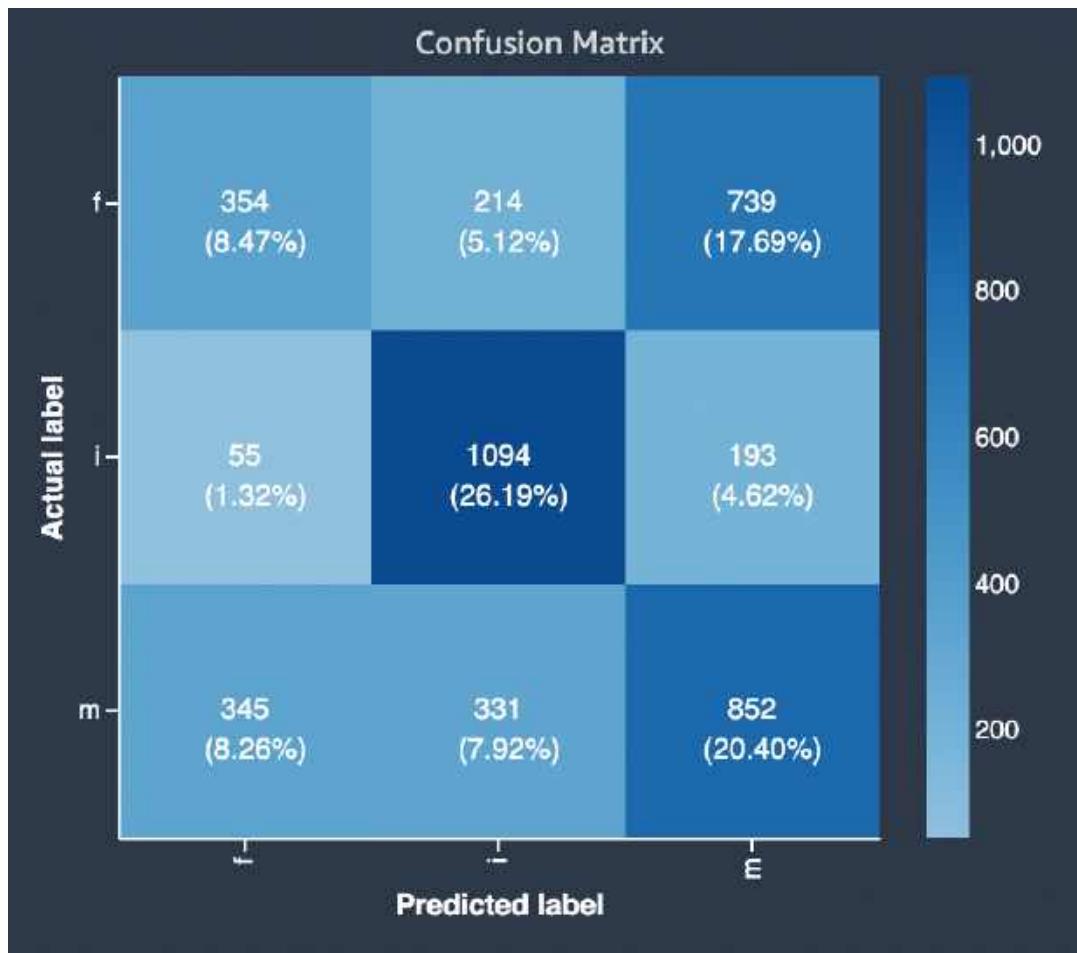
- The number and percentage of correct and incorrect predictions for the actual labels
- The number and percentage of accurate predictions on the diagonal from the upper-left to the lower-right corner
- The number and percentage of inaccurate predictions on the diagonal from the upper-right to the lower-left corner

The incorrect predictions on a confusion matrix are the confusion values.

The following diagram is an example of a confusion matrix for a multi-class classification problem. The confusion matrix in the model quality report contains the following.

- The vertical axis is divided into three rows containing three different actual labels.
- The horizontal axis is divided into three columns containing labels that were predicted by the model.
- The color bar assigns a darker tone to a larger number of samples to visually indicate the number of values that were classified in each category.

In the example below, the model correctly predicted actual 354 values for label **f**, 1094 values for label **i** and 852 values for label **m**. The difference in tone indicates that the dataset is not balanced because there are many more labels for the value **i** than for **f** or **m**.



The confusion matrix in the model quality report provided can accommodate a maximum of 15 labels for multiclass classification problem types. If a row corresponding to a label shows a Nan value, it means that the validation dataset used to check model predictions does not contain data with that label.

## Create an AutoML job for time-series forecasting using the API

Forecasting in machine learning refers to the process of predicting future outcomes or trends based on historical data and patterns. By analyzing past time-series data and identifying underlying patterns, machine learning algorithms can make predictions and provide valuable insights into future behavior. In forecasting, the goal is to develop models that can accurately capture the relationship between input variables and the target variable over time. This involves examining various factors such as trends, seasonality, and other relevant patterns within the data. The collected information is then used to train a machine learning model. The trained model is capable of generating predictions by taking new input data and applying the learned patterns and

relationships. It can provide forecasts for a wide range of use cases, such as sales projections, stock market trends, weather forecasts, demand forecasting, and many more.

The following instructions show how to create an Amazon SageMaker Autopilot job as a pilot experiment for time-series forecasting problem types using SageMaker [API Reference](#).

### Note

Tasks such as text and image classification, time-series forecasting, and fine-tuning of large language models are exclusively available through the version 2 of the [AutoML REST API](#).

If your language of choice is Python, you can refer to [AWS SDK for Python \(Boto3\)](#) or the [AutoMLV2 object](#) of the Amazon SageMaker Python SDK directly.

Users who prefer the convenience of a user interface can use [Amazon SageMaker Canvas](#) to access pre-trained models and generative AI foundation models, or create custom models tailored for specific text, image classification, forecasting needs, or generative AI.

You can create an Autopilot time-series forecasting experiment programmatically by calling the [CreateAutoMLJobV2](#) API in any language supported by Amazon SageMaker Autopilot or the AWS CLI.

For information on how this API action translates into a function in the language of your choice, see the [See Also](#) section of [CreateAutoMLJobV2](#) and choose an SDK. As an example, for Python users, see the full request syntax of [create\\_auto\\_ml\\_job\\_v2](#) in [AWS SDK for Python \(Boto3\)](#).

Autopilot trains several model candidates with your target time-series, then selects an optimal forecasting model for a given objective metric. When your model candidates have been trained, you can find the best candidate metrics in the response to [DescribeAutoMLJobV2](#) at [BestCandidate](#).

The following sections define the mandatory and optional input request parameters for the [CreateAutoMLJobV2](#) API used in time-series forecasting.

### Note

Refer to the notebook [Time-Series Forecasting with Amazon SageMaker Autopilot](#) for a practical, hands-on time-series forecasting example. In this notebook, you use Amazon SageMaker Autopilot to train a time-series model and produce predictions using the trained

model. The notebook provides instructions for retrieving a ready-made dataset of tabular historical data on Amazon S3.

## Prerequisites

Before using Autopilot to create a time-series forecasting experiment in SageMaker AI, make sure to:

- Prepare your time-series dataset. Dataset preparation involves collecting relevant data from various sources, cleaning and filtering it to remove noise and inconsistencies, and organizing it into a structured format. See [Time-series datasets format and missing values filling methods](#) to learn more about time-series formats requirements in Autopilot. Optionally, you can supplement your dataset with the public holiday calendar of the country of your choice to capture associated patterns. For more information on holiday calendars, see [National holiday calendars](#).

### Note

We recommend providing at least 3-5 historical data points for each 1 future data point you want to predict. For example, to forecast 7 days ahead (horizon of 1 week) based on daily data, train your model on a minimum of 21-35 days of historical data. Make sure to provide enough data to capture seasonal and recurrent patterns.

- Place your time-series data in an Amazon S3 bucket.
- Grant full access to the Amazon S3 bucket containing your input data for the SageMaker AI execution role used to run your experiment. Once this is done, you can use the ARN of this execution role in Autopilot API requests.
  - For information on retrieving your SageMaker AI execution role, see [Get your execution role](#).
  - For information on granting your SageMaker AI execution role permissions to access one or more specific buckets in Amazon S3, see *Add Additional Amazon S3 Permissions to a SageMaker AI Execution Role* in [Create execution role](#).

## Required parameters

When calling [CreateAutoMLJobV2](#) to create an Autopilot experiment for time-series forecasting, you must provide the following values:

- An [AutoMLJobName](#) to specify the name of your job. The name should be of type `string`, and should have a minimum length of 1 character and a maximum length of 32.
- At least one [AutoMLJobChannel](#) in [AutoMLJobInputDataConfig](#) in which you specify the name of the Amazon S3 bucket that contains your data. Optionally, you can specify the content (CSV or Parquet files) and compression (GZip) types.
- An [AutoMLProblemTypeConfig](#) of type [TimeSeriesForecastingJobConfig](#) to configure the settings of your time-series forecasting job. In particular, you must specify:
  - The **frequency** of predictions, which refers to the desired granularity (hourly, daily, monthly, etc) of your forecast.

Valid intervals are an integer followed by Y (Year), M (Month), W (Week), D (Day), H (Hour), and min (Minute). For example, 1D indicates every day and 15min indicates every 15 minutes. The value of a frequency must not overlap with the next larger frequency. For example, you must use a frequency of 1H instead of 60min.

The valid values for each frequency are the following:

- Minute - 1-59
- Hour - 1-23
- Day - 1-6
- Week - 1-4
- Month - 1-11
- Year - 1
- The **horizon** of predictions in your forecast, which refers to the number of time-steps that the model predicts. The forecast horizon is also called the prediction length. The maximum forecast horizon is the lesser of 500 time-steps or 1/4 of the time-steps in the dataset.
- A [TimeSeriesConfig](#) in which you define the schema of your dataset to map the column headers to your forecast by specifying:
  - A `TargetAttributeName`: The column that contains historical data of the target field to forecast.
  - A `TimestampAttributeName`: The column that contains a point in time at which the target value of a given item is recorded.
  - A `ItemIdentifierAttributeName`: The column that contains the item identifiers for which you want to predict the target value.

The following is an example of those request parameters. In this example, you are setting up a daily forecast for the expected quantity or level of demand of specific items over a period of 20 days.

```
"AutoMLProblemTypeConfig": {  
    "ForecastFrequency": "D",  
    "ForecastHorizon": 20,  
    "TimeSeriesConfig": {  
        "TargetAttributeName": "demand",  
        "TimestampAttributeName": "timestamp",  
        "ItemIdentifierAttributeName": "item_id"  
    },  
},
```

- An [OutputDataConfig](#) to specify the Amazon S3 output path to store the artifacts of your AutoML job.
- A [RoleArn](#) to specify the ARN of the role used to access your data. You can use the ARN of the execution role to which you have granted access to your data.

All other parameters are optional. For example, you can set specific forecast quantiles, choose a filling method for missing values in the dataset, or define how to aggregate data that does not align with forecast frequency. To learn how to set those additional parameters, see [Optional parameters](#).

## Optional parameters

The following sections provide details of some optional parameters that you can pass to your time-series forecasting AutoML job.

### How to specify algorithms

By default, your Autopilot job trains a pre-defined list of algorithms on your dataset. However, you can provide a subset of the default selection of algorithms.

For time-series forecasting, you must choose [TimeSeriesForecastingJobConfig](#) as the type of [AutoMLProblemTypeConfig](#).

Then, you can specify an array of selected AutoMLAlgorithms in the AlgorithmsConfig attribute of [CandidateGenerationConfig](#).

The following is an example of an `AlgorithmsConfig` attribute listing exactly three algorithms ("cnn-qr", "prophet", "arima") in its `AutoMLAlgorithms` field.

```
{  
    "AutoMLProblemTypeConfig": {  
        "TimeSeriesForecastingJobConfig": {  
            "CandidateGenerationConfig": {  
                "AlgorithmsConfig": [  
                    {"AutoMLAlgorithms": ["cnn-qr", "prophet", "arima"]}  
                ]  
            },  
        },  
    },  
}
```

For the list of available algorithms for time-series forecasting, see [AutoMLAlgorithms](#). For details on each algorithm, see [Algorithms support for time-series forecasting](#).

## How to specify custom quantiles

Autopilot trains 6 models candidates with your target time-series, then combines these models using a stacking ensemble method to create an optimal forecasting model for a given objective metric. Each Autopilot forecasting model generates a probabilistic forecast by producing forecasts at quantiles between P1 and P99. These quantiles are used to account for forecast uncertainty. By default, forecasts will be generated for the 0.1 (p10), 0.5 (p50), and 0.9 (p90). You can choose to specify your own quantiles.

In Autopilot, you can specify up to five forecast quantiles from 0.01 (p1) to 0.99 (p99), by increments of 0.01 or higher in the `ForecastQuantiles` attribute of [TimeSeriesForecastingJobConfig](#).

In the following example, you are setting up a daily 10th, 25th, 50th, 75th, and 90th percentile forecast for the expected quantity or level of demand of specific items over a period of 20 days.

```
"AutoMLProblemTypeConfig": {  
    "ForecastFrequency": "D",  
    "ForecastHorizon": 20,  
    "ForecastQuantiles": ["p10", "p25", "p50", "p75", "p90"],  
    "TimeSeriesConfig": {  
        "TargetAttributeName": "demand",  
        "TimestampAttributeName": "timestamp",  
    }  
}
```

```
    "ItemIdentifierAttributeName": "item_id"  
},
```

## How to aggregate data for different forecast frequencies

To create a forecast model (also referred to as the best model candidate from your experiment), you must specify a forecast frequency. The forecast frequency determines the frequency of predictions in your forecasts. For example, monthly sales forecasts. Autopilot best model can generate forecasts for data frequencies that are higher than the frequency at which your data is recorded.

During training, Autopilot aggregates any data that does not align with the forecast frequency you specify. For example, you might have some daily data but specify a weekly forecast frequency. Autopilot aligns the daily data based on the week that it belongs in. Autopilot then combines it into single record for each week.

During aggregation, the default transformation method is to sum the data. You can configure the aggregation when you create your AutoML job in the `Transformations` attribute of [`TimeSeriesForecastingJobConfig`](#). The supported aggregation methods are `sum` (default), `avg`, `first`, `min`, `max`. Aggregation is only supported for the target column.

In the following example, you configure the aggregation to calculate the average of the individual promo forecasts to provide the final aggregated forecast values.

```
"Transformations": {  
    "Aggregation": {  
        "promo": "avg"  
    }  
}
```

## How to handle missing values in your input datasets

Autopilot provides a number of filling methods to handle missing values in the target and other numeric columns of your time-series datasets. For information on the list of supported filling methods and their available filling logic, see [Handle missing values](#).

You configure your filling strategy in the `Transformations` attribute of [`TimeSeriesForecastingJobConfig`](#) when creating your AutoML job.

To set a filling method, you need to provide a key-value pair:

- The key is the name of the column for which you want to specify the filling method.
- The value associated with the key is an object that defines the filling strategy for that column.

You can specify multiple filling methods for a single column.

To set a specific value for the filling method, you should set the fill parameter to the desired filling method value (for example "backfill" : "value"), and define the actual filling value in an additional parameter suffixed with "\_value". For example, to set backfill to a value of 2, you must include two parameters: "backfill": "value" and "backfill\_value": "2".

In the following example, you specify the filling strategy for the incomplete data column, "price" as follows: All missing values between the first data point of an item and the last are set to 0 after which all missing values are filled with the value 2 until the end date of the dataset.

```
"Transformations": {  
    "Filling": {  
        "price": {  
            "middlefill" : "zero",  
            "backfill" : "value",  
            "backfill_value": "2"  
        }  
    }  
}
```

## How to specify an objective metric

Autopilot produces accuracy metrics to evaluate the model candidates and help you choose which to use to generate forecasts. When you run a time-series forecasting experiment, you can either choose AutoML to let Autopilot optimize the predictor for you, or you can manually choose an algorithm for your predictor.

By default, Autopilot uses the Average Weighted Quantile Loss. However, you can configure the objective metric when you create your AutoML job in the MetricName attribute of [AutoMLJobObjective](#).

For the list of available algorithms, see [Algorithms support for time-series forecasting](#).

## How to incorporate national holiday information to your dataset

In Autopilot, you can incorporate a feature-engineered dataset of national holiday information to your time-series. Autopilot provide native support for the holiday calendars of over 250 countries.

After you choose a country, Autopilot applies that country's holiday calendar to every item in your dataset during training. This allows the model to identify patterns associated with specific holidays.

You can enable the holiday featurization when you create your AutoML job by passing an [HolidayConfigAttributes](#) object to the HolidayConfig attribute of [TimeSeriesForecastingJobConfig](#). The HolidayConfigAttributes object contains the two letters CountryCode attribute that determines the country of the public national holiday calendar used to augment your time-series dataset.

Refer to [Country Codes](#) for the list of supported calendars and their corresponding country code.

## How to enable automatic deployment

Autopilot allows you to automatically deploy your forecast model to an endpoint. To enable automatic deployment for the best model candidate of an AutoML job, include a [ModelDeployConfig](#) in the AutoML job request. This allows the deployment of the best model to a SageMaker AI endpoint. Below are the available configurations for customization.

- To let Autopilot generate the endpoint name, set [AutoGenerateEndpointName](#) to True.
- To provide your own name for the endpoint, set [AutoGenerateEndpointName](#) to False and provide a name of your choice in [EndpointName](#).

## How to configure AutoML to initiate a remote job on EMR Serverless for large datasets

You can configure your AutoML job V2 to automatically initiate a remote job on Amazon EMR Serverless when additional compute resources are needed to process large datasets. By seamlessly transitioning to EMR Serverless when required, the AutoML job can handle datasets that would otherwise exceed the initially provisioned resources, without any manual intervention from you. EMR Serverless is available for the tabular and time series problem types. We recommend setting up this option for time-series datasets larger than 30 GB.

To allow your AutoML job V2 to automatically transition to EMR Serverless for large dataset, you need to provide an `EmrServerlessComputeConfig` object, which includes an `ExecutionRoleARN` field, to the `AutoMLComputeConfig` of the AutoML job V2 input request.

The `ExecutionRoleARN` is the ARN of the IAM role granting the AutoML job V2 the necessary permissions to run EMR Serverless jobs.

This role should have the following trust relationship:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "Service": "emr-serverless.amazonaws.com"  
            },  
            "Action": "sts:AssumeRole"  
        }  
    ]  
}
```

And grant the permissions to:

- Create, list, and update EMR Serverless applications.
- Start, list, get, or cancel job runs on an EMR Serverless application.
- Tag EMR Serverless resources.
- Pass an IAM role to the EMR Serverless service for execution.

By granting the `iam:PassRole` permission, the AutoML job V2 can temporarily assume the `EMRServerlessRuntimeRole-*` role and pass it to the EMR Serverless service. These are the IAM roles used by the EMR Serverless job execution environments to access other AWS services and resources needed during runtime, such as Amazon S3 for data access, CloudWatch for logging, access to the AWS Glue Data Catalog or other services based on your workload requirements.

See [Job runtime roles for Amazon EMR Serverless](#) for details on this role permissions.

The IAM policy defined in the provided JSON document grants those permissions:

```
{  
    "Version": "2012-10-17",  
    "Statement": [{  
        "Sid": "EMRServerlessCreateApplicationOperation",  
        "Effect": "Allow",  
        "Action": "emr-serverless>CreateApplication",  
        "Resource": "arn:aws:emr-serverless:*:*:/*",  
        "Condition": {  
            "StringEquals": {  
                "String": "true"  
            }  
        }  
    }]  
}
```

```
        "aws:RequestTag/sagemaker:is-canvas-resource": "True",
        "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
},
{
    "Sid": "EMRServerlessListApplicationOperation",
    "Effect": "Allow",
    "Action": "emr-serverless>ListApplications",
    "Resource": "arn:aws:emr-serverless:*:*:/",
    "Condition": {
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
},
{
    "Sid": "EMRServerlessApplicationOperations",
    "Effect": "Allow",
    "Action": [
        "emr-serverless>UpdateApplication",
        "emr-serverless>GetApplication"
    ],
    "Resource": "arn:aws:emr-serverless:*:*:/applications/*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/sagemaker:is-canvas-resource": "True",
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
},
{
    "Sid": "EMRServerlessStartJobRunOperation",
    "Effect": "Allow",
    "Action": "emr-serverless>StartJobRun",
    "Resource": "arn:aws:emr-serverless:*:*:/applications/*",
    "Condition": {
        "StringEquals": {
            "aws:RequestTag/sagemaker:is-canvas-resource": "True",
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
},
{

```

```
"Sid": "EMRServerlessListJobRunOperation",
"Effect": "Allow",
>Action": "emr-serverless>ListJobRuns",
"Resource": "arn:aws:emr-serverless:*:*:/applications/*",
"Condition": {
    "StringEquals": {
        "aws:ResourceTag/sagemaker:is-canvas-resource": "True",
        "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
},
{
    "Sid": "EMRServerlessJobRunOperations",
    "Effect": "Allow",
    "Action": [
        "emr-serverless:GetJobRun",
        "emr-serverless:CancelJobRun"
    ],
    "Resource": "arn:aws:emr-serverless:*:*:/applications/*/jobruns/*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/sagemaker:is-canvas-resource": "True",
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
},
{
    "Sid": "EMRServerlessTagResourceOperation",
    "Effect": "Allow",
    "Action": "emr-serverless:TagResource",
    "Resource": "arn:aws:emr-serverless:*:*:/*",
    "Condition": {
        "StringEquals": {
            "aws:RequestTag/sagemaker:is-canvas-resource": "True",
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
},
{
    "Sid": "IAMPassOperationForEMRServerless",
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "arn:aws:iam::*:role/EMRServerlessRuntimeRole-*",
    "Condition": {
```

```
        "StringEquals": {
            "iam:PassedToService": "emr-serverless.amazonaws.com",
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
]
}
```

## Time-series datasets format and missing values filling methods

Time-series data refers to a collection of observations or measurements recorded over regular intervals of time. In this type of data, each observation is associated with a specific timestamp or time period, creating a sequence of data points ordered chronologically.

The specific columns you include in your time-series dataset depend on the goals of your analysis and the data available to you. At a minimum, the time-series data is composed of a 3-column table where:

- One column contains unique identifiers assigned to individual items to refer to their value at a specific moment.
- Another column represents the point-in-time value or **target** to log the value of a given item at a specific moment. After the model is trained on those target values, this target column contains the values that the model predicts at a specified frequency within a defined horizon.
- And a timestamp column is included to record the date and time when the value was measured.
- Additional columns can contain other factors that may influence the forecast performance. For example, in a time-series dataset for retail where the target is the sales or revenue, you might include features that provide information about units sold, product ID, store location, customer count, inventory levels, as well as covariate indicators such as weather data or demographic information.

### Note

You can add a feature-engineered dataset of national holiday information to your time-series. By including holidays in your time series model, you can capture the periodic patterns that holidays create. This helps your forecasts better reflect the underlying

seasonality of your data. For information on the available calendars per country, see [National holiday calendars](#)

## Datasets format for time-series forecasting

Autopilot supports numeric, categorical, text, and datetime data types. The data type of the target column must be numeric.

Autopilot supports time-series data formatted as CSV (default) files or as Parquet files.

- **CSV** (comma-separated-values) is a row-based file format that stores data in human readable plaintext which is a popular choice for data exchange as they are supported by a wide range of applications.
- **Parquet** is a column-based file format where the data is stored and processed more efficiently than row-based file formats. This makes them a better option for big data problems.

For more information about the resource limits on time-series datasets for forecasting in Autopilot, see [Time-series forecasting resource limits for Autopilot](#).

## Handle missing values

A common issue in time-series forecasting data is the presence of missing values. Your data might contain missing values for a number of reasons, including measurement failures, formatting problems, human errors, or a lack of information to record. For instance, if you are forecasting product demand for a retail store and an item is sold out or unavailable, there would be no sales data to record while that item is out of stock. If prevalent enough, missing values can significantly impact a model's accuracy.

Autopilot provides a number of filling methods to handle missing values, with distinct approaches for the target column and other additional columns. Filling is the process of adding standardized values to missing entries in your dataset.

Refer to [How to handle missing values in your input datasets](#) to learn how to set the method for filling missing values in your time-series dataset.

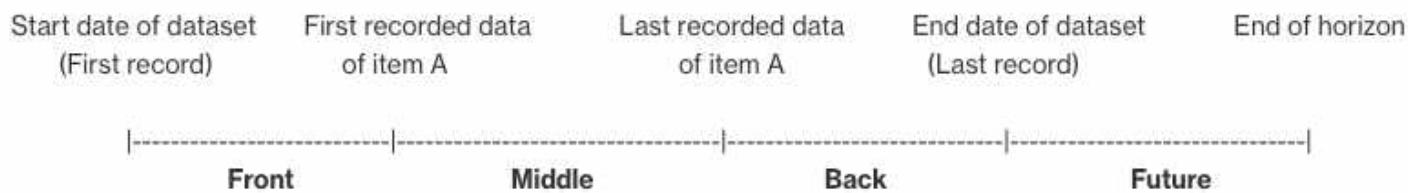
Autopilot supports the following filling methods:

- **Front filling:** Fills any missing values between the earliest recorded data point among all items and the starting point of each item (each item can start at a different time). This ensures that the

data for each item is complete and spans from the earliest recorded data point to its respective starting point.

- **Middle filling:** Fills any missing values between the start and end dates of the items in the dataset.
- **Back filling:** Fills any missing values between the last data point of each item (each item can stop at a different time) and the last recorded data point among all items.
- **Future filling:** Fills any missing values between the last recorded data point among all items and the end of the forecast horizon.

The following image provides a visual representation of the different filling methods.



## Choose a filling logic

When choosing a filling logic, you should consider how the logic will be interpreted by your model. For instance, in a retail scenario, recording 0 sales of an available item is different from recording 0 sales of an unavailable item, as the latter does not imply a lack of customer interest in the item. Because of this, 0 filling in the target column of the time-series might cause the predictor to be under-biased in its predictions, while NaN filling might ignore actual occurrences of 0 available items being sold and cause the predictor to be over-biased.

## Filling logic

You can perform filling on the target column and other numeric columns in your datasets. Target columns have different filling guidelines and restrictions than the rest of the numeric columns.

### Filling Guidelines

Column type	Filling by default?	Supported filling methods	Default filling logic	Accepted filling logic
Target column	Yes	Middle and back filling	0	<ul style="list-style-type: none"> <li>• zero - 0 filling.</li> </ul>

Column type	Filling by default?	Supported filling methods	Default filling logic	Accepted filling logic
				<ul style="list-style-type: none"><li>• value - an integer or float number.</li><li>• nan - not a number.</li><li>• mean - the mean value from the data series.</li><li>• median - the median value from the data series.</li><li>• min - the minimum value from the data series.</li><li>• max - the maximum value from the data series.</li></ul>

Column type	Filling by default?	Supported filling methods	Default filling logic	Accepted filling logic
Other numeric columns	No	Middle, back, and future filling	No default	<ul style="list-style-type: none"><li>• zero - 0 filling.</li><li>• value - an integer or float value.</li><li>• mean - the mean value from the data series.</li><li>• median - the median value from the data series.</li><li>• min - the minimum value from the data series.</li><li>• max - the maximum value from the data series.</li></ul>

 **Note**

For both the target and other numeric columns, mean, median, min, and max are calculated based on a rolling window of the 64 most recent data entries before the missing values.

## National holiday calendars

Autopilot supports a feature-engineered dataset of national holiday information that provides access to the holiday calendars of over 250 countries. Holiday calendar features are especially useful in the retail domain, where public holidays can significantly affect demand. The following section lists the country codes that you can use to access the holiday calendars of each supported country.

Refer to [How to incorporate national holiday information to your dataset](#) to learn how to add a calendar to your dataset.

### Country Codes

Autopilot provides native support for the public holiday calendars of the following countries. Use the **Country Code** when specifying a country with the API.

Country	Country Code
Afghanistan	AF
Åland Islands	AX
Albania	AL
Algeria	DZ
American Samoa	AS
Andorra	AD
Angola	AO
Anguilla	AI
Antartica	AQ
Antigua and Barbuda	AG
Argentina	AR
Armenia	AM

Country	Country Code
Aruba	AW
Australia	AU
Austria	AT
Azerbaijan	AZ
Bahamas	BS
Bahrain	BH
Bangladesh	BD
Barbados	BB
Belarus	BY
Belgium	BE
Belize	BZ
Benin	BJ
Bermuda	BM
Bhutan	BT
Bolivia	BO
Bosnia and Herzegovina	BA
Botswana	BW
Bouvet Island	BV
Brazil	BR
British Indian Ocean Territory	IO

Country	Country Code
British Virgin Islands	VG
Brunei Darussalam	BN
Bulgaria	BG
Burkina Faso	BF
Burundi	BI
Cambodia	KH
Cameroon	CM
Canada	CA
Cape Verde	CV
Caribbean Netherlands	BQ
Cayman Islands	KY
Central African Republic	CF
Chad	TD
Chile	CL
China	CN
Christmas Island	CX
Cocos (Keeling) Islands	CC
Colombia	CO
Comoros	KM
Cook Islands	CK

Country	Country Code
Costa Rica	CR
Croatia	HR
Cuba	CU
Curaçao	CW
Cyprus	CY
Czechia	CZ
Democratic Republic of the Congo	CD
Denmark	DK
Djibouti	DJ
Dominica	DM
Dominican Republic	DO
Ecuador	EC
Egypt	EG
El Salvador	SV
Equatorial Guinea	GQ
Eritrea	ER
Estonia	EE
Eswatini	SZ
Ethiopia	ET
Falkland Islands	FK

Country	Country Code
Faroe Islands	FO
Fiji	FJ
Finland	FI
France	FR
French Guiana	GF
French Polynesia	PF
French Southern Territories	TF
Gabon	GA
Gambia	GM
Georgia	GE
Germany	DE
Ghana	GH
Gibraltar	GI
Greece	GR
Greenland	GL
Grenada	GD
Guadeloupe	GP
Guam	GU
Guatemala	GT
Guernsey	GG

Country	Country Code
Guinea	GN
Guinea-Bissau	GW
Guyana	GY
Haiti	HT
Heard Island and McDonald Islands	HM
Honduras	HN
Hong Kong	HK
Hungary	HU
Iceland	IS
India	IN
Indonesia	ID
Iran	IR
Iraq	IQ
Ireland	IE
Isle of Man	IM
Israel	IL
Italy	IT
Ivory Coast	CI
Jamaica	JM
Japan	JP

Country	Country Code
Jersey	JE
Jordan	JO
Kazakhstan	KZ
Kenya	KE
Kiribati	KI
Kosovo	XK
Kuwait	KW
Kyrgyzstan	KG
Laos	LA
Latvia	LV
Lebanon	LB
Lesotho	LS
Liberia	LR
Libya	LY
Liechtenstein	LI
Lithuania	LT
Luxembourg	LU
Macao	MO
Madagascar	MG
Malawi	MW

Country	Country Code
Malaysia	MY
Maldives	MV
Mali	ML
Malta	MT
Marshall Islands	MH
Martinique	MQ
Mauritania	MR
Mauritius	MU
Mayotte	YT
Mexico	MX
Micronesia	FM
Moldova	MD
Monaco	MC
Mongolia	MN
Montenegro	ME
Montserrat	MS
Morocco	MA
Mozambique	MZ
Myanmar	MM
Namibia	NA

Country	Country Code
Nauru	NR
Nepal	NP
Netherlands	NL
New Caledonia	NC
New Zealand	NZ
Nicaragua	NI
Niger	NE
Nigeria	NG
Niue	NU
Norfolk Island	NF
North Korea	KP
North Macedonia	MK
Northern Mariana Islands	MP
Norway	NO
Oman	OM
Pakistan	PK
Palau	PW
Palestine	PS
Panama	PA
Papua New Guinea	PG

Country	Country Code
Paraguay	PY
Peru	PE
Philippines	PH
Pitcairn Islands	PN
Poland	PL
Portugal	PT
Puerto Rico	PR
Qatar	QA
Republic of the Congo	CG
Réunion	RE
Romania	RO
Russian Federation	RU
Rwanda	RW
Saint Barthélémy	BL
"Saint Helena, Ascension and Tristan da Cunha "	SH
Saint Kitts and Nevis	KN
Saint Lucia	LC
Saint Martin	MF
Saint Pierre and Miquelon	PM
Saint Vincent and the Grenadines	VC

Country	Country Code
Samoa	WS
San Marino	SM
Sao Tome and Principe	ST
Saudi Arabia	SA
Senegal	SN
Serbia	RS
Seychelles	SC
Sierra Leone	SL
Singapore	SG
Sint Maarten	SX
Slovakia	SK
Slovenia	SI
Solomon Islands	SB
Somalia	SO
South Africa	ZA
South Georgia and the South Sandwich Islands	GS
South Korea	KR
South Sudan	SS
Spain	ES
Sri Lanka	LK

Country	Country Code
Sudan	SD
Suriname	SR
Svalbard and Jan Mayen	SJ
Sweden	SE
Switzerland	CH
Syrian Arab Republic	SY
Taiwan	TW
Tajikistan	TJ
Tanzania	TZ
Thailand	TH
Timor-Leste	TL
Togo	TG
Tokelau	TK
Tonga	TO
Trinidad and Tobago	TT
Tunisia	TN
Turkey	TR
Turkmenistan	TM
Turks and Caicos Islands	TC
Tuvalu	TV

Country	Country Code
Uganda	UG
Ukraine	UA
United Arab Emirates	AE
United Kingdom	UK
United Nations	UN
United States	US
United States Minor Outlying Islands	UM
United States Virgin Islands	VI
Uruguay	UY
Uzbekistan	UZ
Vanuatu	VU
Vatican City	VA
Venezuela	VE
Vietnam	VN
Wallis and Futuna	WF
Western Sahara	EH
Yemen	YE
Zambia	ZM
Zimbabwe	ZW

## Objective metrics

Autopilot produces accuracy metrics to evaluate the model candidates and help you choose which to use to generate forecasts. You can either let Autopilot optimize the predictor for you, or you can manually choose an algorithm for your predictor. By default, Autopilot uses the Average Weighted Quantile Loss.

The following list contains the names of the metrics that are currently available to measure the performance of models for time-series forecasting.

### RMSE

Root mean squared error (RMSE) – Measures the square root of the squared difference between predicted and actual values, and is averaged over all values. It's an important metric to indicate the presence of large model errors and outliers. Values range from zero (0) to infinity, with smaller numbers indicating a better model fit to the data. RMSE is dependent on scale, and should not be used to compare datasets of different sizes.

### wQL

Weighted Quantile Loss (wQL) – Assess the accuracy of the forecast by measuring the weighted absolute differences between predicted and actual P10, P50, and P90 quantiles with lower values indicating better performance.

### Average wQL (default)

Average Weighted Quantile Loss (Average wQL) – Evaluates the forecast by averaging the accuracy at the P10, P50, and P90 quantiles. A lower value indicates a more accurate model.

### MASE

Mean Absolute Scaled Error (MASE) – The mean absolute error of the forecast normalized by the mean absolute error of a simple baseline forecasting method. A lower value indicates a more accurate model, where  $MASE < 1$  is estimated to be better than the baseline and  $MASE > 1$  is estimated to be worse than the baseline.

### MAPE

Mean Absolute Percent Error (MAPE) – The percentage error (percent difference of the mean forecasted value versus the actual value) averaged over all time points. A lower value indicates a more accurate model, where  $MAPE = 0$  is a model with no errors.

## WAPE

Weighted Absolute Percent Error (WAPE) – The sum of the absolute error normalized by the sum of the absolute target, which measure the overall deviation of forecasted values from observed values. A lower value indicates a more accurate model.

## Algorithms support for time-series forecasting

Autopilot trains the following six built-in algorithms with your target time-series. Then, using a stacking ensemble method, it combines these model candidates to create an optimal forecasting model for a given objective metric.

- **Convolutional Neural Network - Quantile Regression (CNN-QR)** – CNN-QR is a proprietary machine learning algorithm for forecasting time-series using causal convolutional neural networks (CNNs). CNN-QR works best with large datasets containing hundreds of time-series.
- **DeepAR+** – DeepAR+ is a proprietary machine learning algorithm for forecasting time-series using recurrent neural networks (RNNs). DeepAR+ works best with large datasets containing hundreds of feature time-series.
- **Prophet** – [Prophet](#) is a popular local Bayesian structural time series model based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality. The Autopilot Prophet algorithm uses the [Prophet class](#) of the Python implementation of Prophet. It works best with time-series with strong seasonal effects and several seasons of historical data.
- **Non-Parametric Time Series (NPTS)** – The NPTS proprietary algorithm is a scalable, probabilistic baseline forecaster. It predicts the future value distribution of a given time-series by sampling from past observations. NPTS is especially useful when working with sparse or intermittent time series.
- **Autoregressive Integrated Moving Average (ARIMA)** – ARIMA is a commonly used statistical algorithm for time-series forecasting. The algorithm captures standard temporal structures (patterned organizations of time) in the input dataset. It is especially useful for simple datasets with under 100 time series.
- **Exponential Smoothing (ETS)** – ETS is a commonly used statistical algorithm for time-series forecasting. The algorithm is especially useful for simple datasets with under 100 time series, and datasets with seasonality patterns. ETS computes a weighted average over all observations in the time series dataset as its prediction, with exponentially decreasing weights over time.

## Forecast a deployed Autopilot model

After training your models using the AutoML API, you can deploy them for real-time or batch-based forecasting.

The AutoML API trains several model candidates for your time-series data and selects an optimal forecasting model based on your target objective metric. Once your model candidates have been trained, you can find the best candidate in the response [DescribeAutoMLJobV2](#) at [BestCandidate](#).

To get predictions using this best performing model, you can either set up an endpoint to obtain forecasts interactively or use batch forecasting to make predictions on a batch of observations.

### Considerations

- When providing input data for forecasting, the schema of your data should remain the same as the one used to train your model, including the number of columns, column headers, and data types. You can forecast for existing or new item IDs within the same or different timestamp range to predict for a different time period.
- Forecasting models predict for the forecast horizon points in the future specified in the input request at training, which is from the *target end date* to the *target end date + forecast horizon*. To use the model for predicting specific dates, you should provide the data in the same format as the original input data, extending up to a specified *target end date*. In this scenario, the model will start predicting from the new target end date.

For example, if your dataset had monthly data from January to June with a Forecast horizon of 2, then the model would predict the target value for the next 2 months, which would be July and August. If in August, you want to predict for the next 2 months, this time your input data should be from January to August and the model will predict for the next 2 months (September, October).

- When forecasting future data points, there is no set minimum for the amount of historical data to provide. Include enough data to capture seasonal and recurrent patterns in your time-series.

### Topics

- [Real-time forecasting](#)
- [Batch forecasting](#)

## Real-time forecasting

Real-time forecasting is useful when you need to generate predictions on-the-fly, such as for applications that require immediate responses or when forecasting for individual data points.

By deploying your AutoML model as a real-time endpoint, you can generate forecasts on-demand and minimize the latency between receiving new data and obtaining predictions. This makes real-time forecasting well-suited for applications that require immediate, personalized, or event-driven forecasting capabilities.

For real time forecasting, the dataset should be a subset of the input dataset. The real time endpoint has an input data size of approximately 6MB and a response timeout limitation of 60 seconds. We recommend bringing in one or few items at a time.

You can use SageMaker APIs to retrieve the best candidate of an AutoML job and then create a SageMaker AI endpoint using that candidate.

Alternatively, you can chose the automatic deployment option when creating your Autopilot experiment. For information on setting up the automatic deployment of models, see [How to enable automatic deployment](#).

### To create a SageMaker AI endpoint using your best model candidate:

#### 1. Retrieve the details of the AutoML job.

The following AWS CLI command example uses the [DescribeAutoMLJobV2](#) API to obtain details of the AutoML job, including the information about the best model candidate.

```
aws sagemaker describe-auto-ml-job-v2 --auto-ml-job-name job-name --region region
```

#### 2. Extract the container definition from [InferenceContainers](#) for the best model candidate.

A container definition is the containerized environment used to host the trained SageMaker AI model for making predictions.

```
BEST_CANDIDATE=$(aws sagemaker describe-auto-ml-job-v2 \
--auto-ml-job-name job-name
--region region \
--query 'BestCandidate.InferenceContainers[0]' \
--output json
```

This command extracts the container definition for the best model candidate and stores it in the `BEST_CANDIDATE` variable.

### 3. Create a SageMaker AI model using the best candidate container definition.

Use the container definitions from the previous steps to create a SageMaker AI model by using the [CreateModel](#) API.

```
aws sagemaker create-model \
    --model-name 'your-candidate-name' \
    --primary-container "$BEST_CANDIDATE"
    --execution-role-arn 'execution-role-arn' \
    --region 'region'
```

The `--execution-role-arn` parameter specifies the IAM role that SageMaker AI assumes when using the model for inference. For details on the permissions required for this role, see [CreateModel API: Execution Role Permissions](#).

### 4. Create a SageMaker AI endpoint configuration using the model.

The following AWS CLI command uses the [CreateEndpointConfig](#) API to create an endpoint configuration.

```
aws sagemaker create-endpoint-config \
    --production-variants file://production-variants.json \
    --region 'region'
```

Where the `production-variants.json` file contains the model configuration, including the model name and instance type.

 **Note**

We recommend using [m5.12xlarge](#) instances for real-time forecasting.

```
[  
  {  
    "VariantName": "variant-name",  
    "ModelName": "model-name",  
    "InitialInstanceCount": 1,
```

```
        "InstanceType": "m5.12xlarge"
    }
]
```

## 5. Create the SageMaker AI endpoint using the endpoint configuration.

The following AWS CLI example uses the [CreateEndpoint](#) API to create the endpoint.

```
aws sagemaker create-endpoint \
    --endpoint-name 'endpoint-name' \
    --endpoint-config-name 'endpoint-config-name' \
    --region 'region'
```

Check the progress of your real-time inference endpoint deployment by using the [DescribeEndpoint](#) API. See the following AWS CLI command as an example.

```
aws sagemaker describe-endpoint \
    --endpoint-name 'endpoint-name' \
    --region 'region'
```

After the EndpointStatus changes to InService, the endpoint is ready to use for real-time inference.

## 6. Invoke the SageMaker AI endpoint to make predictions.

```
aws sagemaker invoke-endpoint \
    --endpoint-name 'endpoint-name' \
    --region 'region' \
    --body file://input-data-in-bytes.json \
    --content-type 'application/json' outfile
```

Where the `input-data-in-bytes.json` file contains the input data for the prediction.

### Batch forecasting

Batch forecasting, also known as offline inferencing, generates model predictions on a batch of observations. Batch inference is a good option for large datasets or if you don't need an immediate response to a model prediction request.

By contrast, online inference (real-time inferencing) generates predictions in real time.

You can use SageMaker APIs to retrieve the best candidate of an AutoML job and then submit a batch of input data for inference using that candidate.

## 1. Retrieve the details of the AutoML job.

The following AWS CLI command example uses the [DescribeAutoMLJobV2](#) API to obtain details of the AutoML job, including the information about the best model candidate.

```
aws sagemaker describe-auto-ml-job-v2 --auto-ml-job-name job-name --region region
```

## 2. Extract the container definition from [InferenceContainers](#) for the best model candidate.

A container definition is the containerized environment used to host the trained SageMaker AI model for making predictions.

```
BEST_CANDIDATE=$(aws sagemaker describe-auto-ml-job-v2 \  
    --auto-ml-job-name job-name \  
    --region region \  
    --query 'BestCandidate.InferenceContainers[0]' \  
    --output json
```

This command extracts the container definition for the best model candidate and stores it in the BEST\_CANDIDATE variable.

## 3. Create a SageMaker AI model using the best candidate container definition.

Use the container definitions from the previous steps to create a SageMaker AI model by using the [CreateModel](#) API.

```
aws sagemaker create-model \  
    --model-name 'model-name' \  
    --primary-container "$BEST_CANDIDATE" \  
    --execution-role-arn 'execution-role-arn' \  
    --region 'region'
```

The --execution-role-arn parameter specifies the IAM role that SageMaker AI assumes when using the model for inference. For details on the permissions required for this role, see [CreateModel API: Execution Role Permissions](#).

## 4. Create a batch transform job.

The following example creates a transform job using the [CreateTransformJob](#) API.

```
aws sagemaker create-transform-job \
    --transform-job-name 'transform-job-name' \
    --model-name 'model-name' \
    --transform-input file://transform-input.json \
    --transform-output file://transform-output.json \
    --transform-resources file://transform-resources.json \
    --region 'region'
```

The input, output, and resource details are defined in separate JSON files:

- `transform-input.json`:

```
{  
    "DataSource": {  
        "S3DataSource": {  
            "S3DataType": "S3Prefix",  
            "S3Uri": "s3://my-input-data-bucket/path/to/input/data"  
        }  
    },  
    "ContentType": "text/csv",  
    "SplitType": "None"  
}
```

- `transform-output.json`:

```
{  
    "S3OutputPath": "s3://my-output-bucket/path/to/output",  
    "AssembleWith": "Line"  
}
```

- `transform-resources.json`:

 **Note**

We recommend using [m5.12xlarge](#) instances for general-purpose workloads and m5.24xlarge instances for big data forecasting tasks.

```
{  
    "InstanceType": "instance-type",
```

```
    "InstanceCount": 1  
}
```

## 5. Monitor the progress of your transform job using the [DescribeTransformJob API](#).

See the following AWS CLI command as an example.

```
aws sagemaker describe-transform-job \  
  --transform-job-name 'transform-job-name' \  
  --region region
```

## 6. Retrieve the batch transform output.

After the job is finished, the predicted result is available in the S3OutputPath.

The output file name has the following format: `input_data_file_name.out`. As an example, if your input file is `text_x.csv`, the output name will be `text_x.csv.out`.

```
aws s3 ls s3://my-output-bucket/path/to/output/
```

The following code examples illustrate the use of the AWS SDK for Python (`boto3`) and the AWS CLI for batch forecasting.

### AWS SDK for Python (`boto3`)

The following example uses **AWS SDK for Python (`boto3`)** to make predictions in batches.

```
import sagemaker  
import boto3  
  
session = sagemaker.Session()  
  
sm_client = boto3.client('sagemaker', region_name='us-west-2')  
role = 'arn:aws:iam::1234567890:role/sagemaker-execution-role'  
output_path = 's3://test-auto-ml-job/output'  
input_data = 's3://test-auto-ml-job/test_X.csv'  
  
best_candidate = sm_client.describe_auto_ml_job_v2(AutoMLJobName=job_name)  
['BestCandidate']  
best_candidate_containers = best_candidate['InferenceContainers']  
best_candidate_name = best_candidate['CandidateName']
```

```
# create model
reponse = sm_client.create_model(
    ModelName = best_candidate_name,
    ExecutionRoleArn = role,
    Containers = best_candidate_containers
)

# Lauch Transform Job
response = sm_client.create_transform_job(
    TransformJobName=f'{best_candidate_name}-transform-job',
    ModelName=model_name,
    TransformInput={
        'DataSource': {
            'S3DataSource': {
                'S3DataType': 'S3Prefix',
                'S3Uri': input_data
            }
        },
        'ContentType': "text/csv",
        'SplitType': 'None'
    },
    TransformOutput={
        'S3OutputPath': output_path,
        'AssembleWith': 'Line',
    },
    TransformResources={
        'InstanceType': 'ml.m5.2xlarge',
        'InstanceCount': 1,
    },
)
```

The batch inference job returns a response in the following format.

```
{'TransformJobArn': 'arn:aws:sagemaker:us-west-2:1234567890:transform-job/test-transform-job',
'ResponseMetadata': {'RequestId': '659f97fc-28c4-440b-b957-a49733f7c2f2',
'HTTPStatusCode': 200,
'HTTPHeaders': {'x-amzn-requestid': '659f97fc-28c4-440b-b957-a49733f7c2f2',
'content-type': 'application/x-amz-json-1.1',
'content-length': '96',
'date': 'Thu, 11 Aug 2022 22:23:49 GMT'},
'RetryAttempts': 0}}
```

## AWS Command Line Interface (AWS CLI)

### 1. Obtain the best candidate container definitions.

```
aws sagemaker describe-auto-ml-job-v2 --auto-ml-job-name 'test-automl-job' --region us-west-2
```

### 2. Create the model.

```
aws sagemaker create-model --model-name 'test-sagemaker-model'  
--containers '[{  
    "Image": "348316444620.dkr.ecr.us-west-2.amazonaws.com/sagemaker-sklearn-  
automl:2.5-1-cpu-py3",  
    "ModelDataUrl": "s3://amzn-s3-demo-bucket/out/test-job1/data-processor-models/  
test-job1-dpp0-1-e569ff7ad77f4e55a7e549a/output/model.tar.gz",  
    "Environment": {  
        "AUTOML_SPARSE_ENCODE_RECORDIO_PROTOBUF": "1",  
        "AUTOML_TRANSFORM_MODE": "feature-transform",  
        "SAGEMAKER_DEFAULT_INVOCATIONS_ACCEPT": "application/x-recordio-protobuf",  
        "SAGEMAKER_PROGRAM": "sagemaker_serve",  
        "SAGEMAKER_SUBMIT_DIRECTORY": "/opt/ml/model/code"  
    }  
}, {  
    "Image": "348316444620.dkr.ecr.us-west-2.amazonaws.com/sagemaker-  
xgboost:1.3-1-cpu-py3",  
    "ModelDataUrl": "s3://amzn-s3-demo-bucket/out/test-job1/tuning/flicdf10v2-  
dpp0-xgb/test-job1E9-244-7490a1c0/output/model.tar.gz",  
    "Environment": {  
        "MAX_CONTENT_LENGTH": "20971520",  
        "SAGEMAKER_DEFAULT_INVOCATIONS_ACCEPT": "text/csv",  
        "SAGEMAKER_INFERENCE_OUTPUT": "predicted_label",  
        "SAGEMAKER_INFERENCE_SUPPORTED":  
            "predicted_label,probability,probabilities"  
    }  
}, {  
    "Image": "348316444620.dkr.ecr.us-west-2.amazonaws.com/sagemaker-sklearn-  
automl:2.5-1-cpu-py3",  
    "ModelDataUrl": "s3://amzn-s3-demo-bucket/out/test-job1/data-processor-models/  
test-job1-dpp0-1-e569ff7ad77f4e55a7e549a/output/model.tar.gz",  
    "Environment": {  
        "AUTOML_TRANSFORM_MODE": "inverse-label-transform",  
        "SAGEMAKER_DEFAULT_INVOCATIONS_ACCEPT": "text/csv",  
        "SAGEMAKER_INFERENCE_INPUT": "predicted_label",
```

```
"SAGEMAKER_INFERENCE_OUTPUT": "predicted_label",
"SAGEMAKER_INFERENCE_SUPPORTED":
"predicted_label,probability,labels,probabilities",
"SAGEMAKER_PROGRAM": "sagemaker_serve",
"SAGEMAKER_SUBMIT_DIRECTORY": "/opt/ml/model/code"
}
}]\` \
--execution-role-arn 'arn:aws:iam::1234567890:role/sagemaker-execution-role' \
--region 'us-west-2'
```

### 3. Create a transform job.

```
aws sagemaker create-transform-job --transform-job-name 'test-transform-job' \
--model-name 'test-sagemaker-model' \
--transform-input '{
  "DataSource": {
    "S3DataSource": {
      "S3DataType": "S3Prefix",
      "S3Uri": "s3://amzn-s3-demo-bucket/data.csv"
    }
  },
  "ContentType": "text/csv",
  "SplitType": "None"
}' \
--transform-output '{
  "S3OutputPath": "s3://amzn-s3-demo-bucket/output/",
  "AssembleWith": "Line"
}' \
--transform-resources '{
  "InstanceType": "ml.m5.2xlarge",
  "InstanceCount": 1
}' \
--region 'us-west-2'
```

### 4. Check the progress of the transform job.

```
aws sagemaker describe-transform-job --transform-job-name 'test-transform-job' \
--region 'us-west-2'
```

The following is the response from the transform job.

```
{ \
  "TransformJobName": "test-transform-job",
```

```
"TransformJobArn": "arn:aws:sagemaker:us-west-2:1234567890:transform-job/test-transform-job",
"TransformJobStatus": "InProgress",
"ModelName": "test-model",
"TransformInput": {
    "DataSource": {
        "S3DataSource": {
            "S3DataType": "S3Prefix",
            "S3Uri": "s3://amzn-s3-demo-bucket/data.csv"
        }
    },
    "ContentType": "text/csv",
    "CompressionType": "None",
    "SplitType": "None"
},
"TransformOutput": {
    "S3OutputPath": "s3://amzn-s3-demo-bucket/output/",
    "AssembleWith": "Line",
    "KmsKeyId": ""
},
"TransformResources": {
    "InstanceType": "ml.m5.2xlarge",
    "InstanceCount": 1
},
"CreationTime": 1662495635.679,
"TransformStartTime": 1662495847.496,
"DataProcessing": {
    "InputFilter": "$",
    "OutputFilter": "$",
    "JoinSource": "None"
}
}
```

After the TransformJobStatus changes to Completed, you can check the inference result in the S3outputPath.

## Amazon SageMaker Autopilot data exploration notebook

Amazon SageMaker Autopilot cleans and pre-processes your dataset automatically. To help users understand their data, uncover patterns, relationships, and anomalies about the time-series, Amazon SageMaker Autopilot generates a **data exploration** static report in the form of a notebook for users to reference.

The data exploration notebook is generated for every Autopilot job. The report is stored in an Amazon S3 bucket and can be accessed from the job output path.

You can find the Amazon S3 prefix to the data exploration notebook in the response to [DescribeAutoMLJobV2](#) at [AutoMLJobArtifacts.DataExplorationNotebookLocation](#).

## Reports generated by Amazon SageMaker Autopilot

In addition to the data exploration notebook, Autopilot generates various reports for the best model candidate of each experiment.

- An explainability report provides insights into how the model makes forecasts.
- A performance report provides a quantitative assessment of the model's forecasting capabilities.
- A backtest results report is generated after testing the model's performance on historical data.

### Explainability report

Autopilot explainability report helps you better understand how the attributes in your datasets impact forecasts for specific time-series (item and dimension combinations) and time points.

Autopilot uses a metric called *Impact scores* to quantify the relative impact of each attribute and determine whether they increase or decrease forecast values.

For example, consider a forecasting scenario where the target is sales and there are two related attributes: price and color. Autopilot may find that the item's color has a high impact on sales for certain items, but a negligible effect for other items. It may also find that a promotion in the summer has a high impact on sales, but a promotion in the winter has little effect.

The explainability report is generated only when:

- The time series dataset includes additional feature columns or is associated with a holiday calendar.
- The base models CNN-QR and DeepAR+ are included in the final ensemble.

### Interpret Impact scores

Impact scores measure the relative impact attributes have on forecast values. For example, if the price attribute has an impact score that is twice as large as the store location attribute, you can conclude that the price of an item has twice the impact on forecast values than the store location.

Impact scores also provide information on whether attributes increase or decrease forecast values.

The Impact scores range from -1 to 1, where the sign denotes the direction of the impact. A score of 0 indicates no impact, while scores close to 1 or -1 indicate a significant impact.

It is important to note that Impact scores measure the relative impact of attributes, not the absolute impact. Therefore, Impact scores cannot be used to determine whether particular attributes improve model accuracy. If an attribute has a low Impact score, that does not necessarily mean that it has a low impact on forecast values; it means that it has a lower impact on forecast values than other attributes used by the predictor.

## Find the explainability report

You can find the Amazon S3 prefix to the explainability artifacts generated for the best candidate in the response to [DescribeAutoMLJobV2](#) at [BestCandidate.CandidateProperties.CandidateArtifactLocations.Explainability](#).

## Model performance report

Autopilot model quality report (also referred to as performance report) provides insights and quality information for the best model candidate (best predictor) generated by an AutoML job. This includes information about the job details, objective function, and accuracy metrics (wQL, MAPE, WAPE, RMSE, MASE).

You can find the Amazon S3 prefix to the model quality report artifacts generated for the best candidate in the response to [DescribeAutoMLJobV2](#) at [BestCandidate.CandidateProperties.CandidateArtifactLocations.ModelInsights](#).

## Backtests results report

Backtests results provide insights into the performance of a time-series forecasting model by evaluating its predictive accuracy and reliability. It helps analysts and data scientists assess its performance on historical data and assists in understanding its potential performance on future, unseen data.

Autopilot uses backtesting to tune parameters and produce accuracy metrics. During backtesting, Autopilot automatically splits your time-series data into two sets, a training set and a testing set. The training set is used to train a model which is then used to generate forecasts for data points in the testing set. Autopilot uses this testing dataset to evaluate the model's accuracy by comparing forecasted values with observed values in the testing set.

You can find the Amazon S3 prefix to the model quality report artifacts generated for the best candidate in the response to [DescribeAutoMLJobV2](#) at [BestCandidate.CandidateProperties.CandidateArtifactLocations.BacktestResults](#).

## Time-series forecasting resource limits for Autopilot

The following table lists the resource limits for time-series forecasting jobs in Amazon SageMaker Autopilot and whether or not you can adjust each limit.

Resource limits	Default limit	Adjustable
Size of input dataset	30 GB	Yes
Size of a single Parquet file	2 GB	No
Maximum number of rows in a dataset	3 billion	Yes
Maximum number of grouping columns	5	No
Maximum number of numerical features	13	No
Maximum number of categorical features	10	No
Maximum number of time-series (unique combinations of item and grouping columns) per dataset	5,000,000	Yes
Maximum Forecast horizon	500	Yes

## Create an AutoML job to fine-tune text generation models using the API

Large language models (LLMs) excel in multiple generative tasks, including text generation, summarization, completion, question answering, and more. Their performance can be attributed to their significant size and extensive training on diverse datasets and various tasks. However, specific

domains, such as healthcare and financial services, may require customized fine-tuning to adapt to unique data and use cases. By tailoring their training to their particular domain, LLMs can improve their performance and provide more accurate outputs for targeted applications.

Autopilot offers the capability to fine-tune a selection of pre-trained generative text models. In particular, Autopilot supports the **instruction-based fine tuning** of a selection of general-purpose large language models (LLMs) powered by JumpStart.

### Note

The text generation models that support fine-tuning in Autopilot are currently accessible exclusively in Regions supported by SageMaker Canvas. See the documentation of SageMaker Canvas for the [full list of its supported Regions](#).

Fine-tuning a pre-trained model requires a specific dataset of clear instructions that guide the model on how to generate output or behave for that task. The model learns from the dataset, adjusting its parameters to conform to the provided instructions. Instruction-based fine-tuning involves using labeled examples formatted as prompt-response pairs and phrased as instructions. For more information about fine-tuning, see [Fine-tune a foundation model](#).

The following guidelines outline the process of creating an Amazon SageMaker Autopilot job as a pilot experiment to fine-tune text generation LLMs using the SageMaker [API Reference](#).

### Note

Tasks such as text and image classification, time-series forecasting, and fine-tuning of large language models are exclusively available through the version 2 of the [AutoML REST API](#).

If your language of choice is Python, you can refer to [AWS SDK for Python \(Boto3\)](#) or the [AutoMLV2 object](#) of the Amazon SageMaker Python SDK directly.

Users who prefer the convenience of a user interface can use [Amazon SageMaker Canvas](#) to access pre-trained models and generative AI foundation models, or create custom models tailored for specific text, image classification, forecasting needs, or generative AI.

To create an Autopilot experiment programmatically for fine-tuning an LLM, you can call the [CreateAutoMLJobV2](#) API in any language supported by Amazon SageMaker Autopilot or the AWS CLI.

For information about how this API action translates into a function in the language of your choice, see the [See Also](#) section of `CreateAutoMLJobV2` and choose an SDK. As an example, for Python users, see the full request syntax of [`create\_auto\_ml\_job\_v2`](#) in AWS SDK for Python (Boto3).

### Note

Autopilot fine-tunes large language models without requiring multiple candidates to be trained and evaluated. Instead, using your dataset, Autopilot directly fine-tunes your target model to enhance a default objective metric, the cross-entropy loss. Fine-tuning language models in Autopilot does not require setting the `AutoMLJobObjective` field.

Once your LLM is fine-tuned, you can evaluate its performance by accessing various ROUGE scores through the [BestCandidate](#) when making a [DescribeAutoMLJobV2](#) API call. The model also provides information about its training and validation loss as well as perplexity. For a comprehensive list of metrics for evaluating the quality of the text generated by the fine-tuned models, see [Metrics for fine-tuning large language models in Autopilot](#).

## Prerequisites

Before using Autopilot to create a fine-tuning experiment in SageMaker AI, make sure to take the following steps:

- (Optional) Choose the pre-trained model you want to fine-tune.

For the list of pre-trained models available for fine-tuning in Amazon SageMaker Autopilot, see [Supported large language models for fine-tuning](#). The selection of a model is not mandatory; if no model is specified, Autopilot automatically defaults to the model *Falcon7BInstruct*.

- Create a dataset of instructions. See [Dataset file types and input data format](#) to learn about the format requirements for your instruction-based dataset.
- Place your dataset in an Amazon S3 bucket.
- Grant full access to the Amazon S3 bucket containing your input data for the SageMaker AI execution role used to run your experiment.
  - For information on retrieving your SageMaker AI execution role, see [Get your execution role](#).
  - For information on granting your SageMaker AI execution role permissions to access one or more specific buckets in Amazon S3, see [Add Additional Amazon S3 Permissions to a SageMaker AI Execution Role](#) in [Create execution role](#).

- Additionally, you should provide your execution role with the necessary permissions to access the default storage Amazon S3 bucket used by JumpStart. This access is required for storing and retrieving pre-trained model artifacts in JumpStart. To grant access to this Amazon S3 bucket, you must create a new inline custom policy on your execution role.

Here's an example policy that you can use in your JSON editor when configuring AutoML fine-tuning jobs in `us-west-2`:

*JumpStart's bucket names follow a predetermined pattern that depends on the AWS Regions. You must adjust the name of the bucket accordingly.*

```
{  
    "Sid": "Statement1",  
    "Effect": "Allow",  
    "Action": [  
        "s3:GetObject",  
        "s3:PutObject",  
        "s3>ListBucket"  
    ],  
    "Resource": [  
        "arn:aws:s3:::jumpstart-cache-prod-us-west-2",  
        "arn:aws:s3:::jumpstart-cache-prod-us-west-2/*"  
    ]  
}
```

Once this is done, you can use the ARN of this execution role in Autopilot API requests.

## Required parameters

When calling [CreateAutoMLJobV2](#) to create an Autopilot experiment for LLM fine-tuning, you must provide the following values:

- An [AutoMLJobName](#) to specify the name of your job. The name should be of type `string`, and should have a minimum length of 1 character and a maximum length of 32.
- At least one [AutoMLJobChannel](#) of the [training](#) type within the [AutoMLJobInputDataConfig](#). This channel specifies the name of the Amazon S3 bucket where your fine-tuning dataset is located. You have the option to define a validation channel. If no validation channel is provided, and a `ValidationFraction` is configured in the [AutoMLDataSplitConfig](#), this fraction is utilized to randomly divide the training dataset into

training and validation sets. Additionally, you can specify the type of content (CSV or Parquet files) for the dataset.

- An [AutoMLProblemTypeConfig](#) of type [TextGenerationJobConfig](#) to configure the settings of your training job.

In particular, you can specify the name of the base model to fine-tune in the `BaseModelName` field. For the list of pre-trained models available for fine-tuning in Amazon SageMaker Autopilot, see [Supported large language models for fine-tuning](#).

- An [OutputDataConfig](#) to specify the Amazon S3 output path to store the artifacts of your AutoML job.
- A [RoleArn](#) to specify the ARN of the role used to access your data.

The following is an example of the full request format used when making an API call to `CreateAutoMLJobV2` for fine-tuning a (`Falcon7BInstruct`) model.

```
{  
    "AutoMLJobName": "<job_name>",  
    "AutoMLJobInputDataConfig": [  
        {  
            "ChannelType": "training",  
            "CompressionType": "None",  
            "ContentType": "text/csv",  
            "DataSource": {  
                "S3DataSource": {  
                    "S3DataType": "S3Prefix",  
                    "S3Uri": "s3://<bucket_name>/<input_data>.csv"  
                }  
            }  
        }  
    ],  
    "OutputDataConfig": {  
        "S3OutputPath": "s3://<bucket_name>/output",  
        "KmsKeyId": "arn:aws:kms:<region>:<account_id>:key/<key_value>"  
    },  
    "RoleArn": "arn:aws:iam::<account_id>:role/<sagemaker_execution_role_name>",  
    "AutoMLProblemTypeConfig": {  
        "TextGenerationJobConfig": {  
            "BaseModelName": "Falcon7BInstruct"  
        }  
    }  
}
```

}

All other parameters are optional.

## Optional parameters

The following sections provide details of some optional parameters that you can pass to your fine-tuning AutoML job.

### How to specify the training and validation datasets of an AutoML job

You can provide your own validation dataset and custom data split ratio, or let Autopilot split the dataset automatically.

Each [AutoMLJobChannel](#) object (see the required parameter [AutoMLJobInputDataConfig](#)) has a ChannelType, which can be set to either training or validation values that specify how the data is to be used when building a machine learning model.

At least one data source must be provided and a maximum of two data sources is allowed: one for training data and one for validation data. How you split the data into training and validation datasets depends on whether you have one or two data sources.

- If you only have **one data source**, the ChannelType is set to training by default and must have this value.
  - If the ValidationFraction value in [AutoMLDataSplitConfig](#) is not set, 0.2 (20%) of the data from this source is used for validation by default.
  - If the ValidationFraction is set to a value between 0 and 1, the dataset is split based on the value specified, where the value specifies the fraction of the dataset used for validation.
- If you have **two data sources**, the ChannelType of one of the AutoMLJobChannel objects must be set to training, the default value. The ChannelType of the other data source must be set to validation. The two data sources must have the same format, either CSV or Parquet, and the same schema. You must not set the value for the ValidationFraction in this case because all of the data from each source is used for either training or validation. Setting this value causes an error.

### How to enable automatic deployment

With Autopilot, you can automatically deploy your fine-tuned model to an endpoint. To enable automatic deployment for your fine-tuned model, include a [ModelDeployConfig](#) in the AutoML

job request. This allows the deployment of your fine-tuned model to a SageMaker AI endpoint. Below are the available configurations for customization.

- To let Autopilot generate the endpoint name, set [AutoGenerateEndpointName](#) to True.
- To provide your own name for the endpoint, set [AutoGenerateEndpointName](#) to False and provide a name of your choice in [EndpointName](#).

## How to set the EULA acceptance when fine-tuning a model using the AutoML API

For models requiring the acceptance of an end-user license agreement before fine-tuning, you can accept the EULA by setting the `AcceptEula` attribute of the [ModelAccessConfig](#) to True in [TextGenerationJobConfig](#) when configuring your [AutoMLProblemTypeConfig](#).

## How to set hyperparameters to optimize the learning process of a model

You can optimize the learning process of your text generation model by setting hyperparameter values in the `TextGenerationHyperParameters` attribute of [TextGenerationJobConfig](#) when configuring your [AutoMLProblemTypeConfig](#).

Autopilot allows for the setting of four common hyperparameters across all models.

- `epochCount`: Its value should be a string containing an integer value within the range of 1 to 10.
- `batchSize`: Its value should be a string containing an integer value within the range of 1 to 64.
- `learningRate`: Its value should be a string containing a floating-point value within the range of 0 to 1.
- `learningRateWarmupSteps`: Its value should be a string containing an integer value within the range of 0 to 250.

For more details on each hyperparameter, see [Hyperparameters for optimizing the learning process of your text generation models](#).

The following JSON example shows a `TextGenerationHyperParameters` field passed to the `TextGenerationJobConfig` where all four hyperparameters are configured.

```
"AutoMLProblemTypeConfig": {  
    "TextGenerationJobConfig": {  
        "BaseModelName": "Falcon7B",  
        "TextGenerationHyperParameters": {"epochCount": "5", "learningRate": "0.000001",  
        "batchSize": "32", "learningRateWarmupSteps": "10"}  
    }  
}
```

```
}
```

## Supported large language models for fine-tuning

Using Autopilot API, users can fine-tune large language models (LLMs) that are powered by Amazon SageMaker JumpStart.

### Note

For fine-tuning models that require the acceptance of an end-user license agreement, you must explicitly declare EULA acceptance when creating your AutoML job. Note that after fine-tuning a pretrained model, the weights of the original model are changed, so you do not need to later accept a EULA when deploying the fine-tuned model.

For information on how to accept the EULA when creating a fine-tuning job using the AutoML API, see [the section called "Set EULA".](#)

You can find the full details of each model by searching for your **JumpStart Model ID** in the following [model table](#), and then following the link in the **Source** column. These details might include the languages supported by the model, biases it may exhibit, the datasets employed for fine-tuning, and more.

The following table lists the supported JumpStart models that you can fine-tune with an AutoML job.

JumpStart Model ID	BaseModelName in API request	Description
huggingface-textgeneration-dolly-v2-3b-bf16	Dolly3B	Dolly 3B is a 2.8 billion parameter instruction-following large language model based on <a href="#">pythia-2.8b</a> . It is trained on the instruction/response fine tuning dataset <a href="#">databricks-dolly-15k</a> and can perform tasks including brainstorming, classification,

JumpStart Model ID	BaseModelName in API request	Description
		questions and answers, text generation, information extraction, and summarization.
huggingface-textgeneration-dolly-v2-7b-bf16	Dolly7B	Dolly 7B is a 6.9 billion parameter instruction-following large language model based on <a href="#">pythia-6.9b</a> . It is trained on the instruction/response fine tuning dataset <a href="#">databricks-dolly-15k</a> and can perform tasks including brainstorming, classification, questions and answers, text generation, information extraction, and summarization.
huggingface-textgeneration-dolly-v2-12b-bf16	Dolly12B	Dolly 12B is a 12 billion parameter instruction-following large language model based on <a href="#">pythia-12b</a> . It is trained on the instruction/response fine tuning dataset <a href="#">databricks-dolly-15k</a> and can perform tasks including brainstorming, classification, questions and answers, text generation, information extraction, and summarization.

JumpStart Model ID	BaseModelName in API request	Description
huggingface-llm-falcon-7b-bf16	Falcon7B	Falcon 7B is a 7 billion parameter causal large language model trained on 1,500 billion tokens enhanced with curated corpora. Falcon-7B is trained on English and French data only, and does not generalize appropriately to other languages. Because the model was trained on large amounts of web data, it carries the stereotypes and biases commonly found online.
huggingface-llm-falcon-7b-instruct-bf16	Falcon7BInstruct	Falcon 7B Instruct is a 7 billion parameter causal large language model built on Falcon 7B and fine-tuned on a 250 million tokens mixture of chat/instruct datasets. Falcon 7B Instruct is mostly trained on English data, and does not generalize appropriately to other languages. Furthermore, as it is trained on a large-scale corpora representative of the web, it carries the stereotypes and biases commonly encountered online.

JumpStart Model ID	BaseModelName in API request	Description
huggingface-llm-falcon-40b-bf16	Falcon40B	<p>Falcon 40B is a 40 billion parameter causal large language model trained on 1,000 billion tokens enhanced with curated corpora. It is trained mostly on English, German, Spanish, and French, with limited capabilities in Italian, Portuguese, Polish, Dutch, Romanian, Czech, and Swedish. It does not generalize appropriately to other languages. Furthermore, as it is trained on a large-scale corpora representative of the web, it carries the stereotypes and biases commonly encountered online.</p>
huggingface-llm-falcon-40b-instruct-bf16	Falcon40BInstruct	<p>Falcon 40B Instruct is a 40 billion parameter causal large language model built on Falcon40B and fine-tuned on a mixture of Baize. It is mostly trained on English and French data, and does not generalize appropriately to other languages. Furthermore, as it is trained on a large-scale corpora representative of the web, it carries the stereotypes and biases commonly encountered online.</p>

JumpStart Model ID	BaseModelName in API request	Description
huggingface-text2text-flan-t5-large	FlanT5L	<p>The <a href="#">Flan-T5</a> model family is a set of large language models that are fine-tuned on multiple tasks and can be further trained. These models are well-suited for tasks such as language translation, text generation, sentence completion, word sense disambiguation, summarization, or question answering. Flan T5 L is a 780 million parameter large language model trained on numerous languages. You can find the list of the languages supported by Flan T5 L in the details of the model retrieved from your search by model ID in JumpStart's <a href="#">model table</a>.</p>

JumpStart Model ID	BaseModelName in API request	Description
huggingface-text2text-flan-t5-xl	FlanT5XL	<p>The <a href="#">Flan-T5</a> model family is a set of large language models that are fine-tuned on multiple tasks and can be further trained. These models are well-suited for tasks such as language translation, text generation, sentence completion, word sense disambiguation, summarization, or question answering. Flan T5 XL is a 3 billion parameter large language model trained on numerous languages. You can find the list of the languages supported by Flan T5 XL in the details of the model retrieved from your search by model ID in JumpStart's <a href="#">model table</a>.</p>

JumpStart Model ID	BaseModelName in API request	Description
huggingface-text2text-flan-t5-xxll	FlanT5XXL	<p>The <a href="#">Flan-T5</a> model family is a set of large language models that are fine-tuned on multiple tasks and can be further trained. These models are well-suited for tasks such as language translation, text generation, sentence completion, word sense disambiguation, summarization, or question answering. Flan T5 XXL is a 11 billion parameter model. You can find the list of the languages supported by Flan T5 XXL in the details of the model retrieved from your search by model ID in JumpStart's <a href="#">model table</a>.</p>
meta-textgeneration-llama-2-7b	Llama2-7B	<p>Llama 2 is a collection of pretrained and fine-tuned generative text models, ranging in scale from 7 billion to 70 billion parameters. Llama2-7B is the 7 billion parameter model that is intended for English use and can be adapted for a variety of natural language generation tasks.</p>

JumpStart Model ID	BaseModelName in API request	Description
meta-textgeneration-llama-2-7b-f	Llama2-7BChat	<p>Llama 2 is a collection of pretrained and fine-tuned generative text models, ranging in scale from 7 billion to 70 billion parameters.</p> <p>Llama2-7B is the 7 billion parameter chat model that is optimized for dialogue use cases.</p>
meta-textgeneration-llama-2-13b	Llama2-13B	<p>Llama 2 is a collection of pretrained and fine-tuned generative text models, ranging in scale from 7 billion to 70 billion parameters.</p> <p>Llama2-13B is the 13 billion parameter model that is intended for English use and can be adapted for a variety of natural language generation tasks.</p>
meta-textgeneration-llama-2-13b-f	Llama2-13BChat	<p>Llama 2 is a collection of pretrained and fine-tuned generative text models, ranging in scale from 7 billion to 70 billion parameters.</p> <p>Llama2-13B is the 13 billion parameter chat model that is optimized for dialogue use cases.</p>

JumpStart Model ID	BaseModelName in API request	Description
huggingface-llm-mistral-7b	Mistral7B	Mistral 7B is a seven billion parameters code and general purpose English text generation model. It can be used in a variety of use cases including text summarization, classification, text completion, or code completion.
huggingface-llm-mistral-7b-instruct	Mistral7BInstruct	Mistral 7B Instruct is the fine-tuned version of Mistral 7B for conversational use cases. It was specialized using a variety of publicly available conversation datasets in English.
huggingface-textgeneration1-mpt-7b-bf16	MPT7B	MPT 7B is a decoder-style transformer large language model with 6.7 billion parameters, pre-trained from scratch on 1 trillion tokens of English text and code. It is prepared to handle long context lengths.
huggingface-textgeneration1-mpt-7b-instruct-bf16	MPT7BInstruct	MPT 7B Instruct is a model for short-form instruction following tasks. It is built by fine-tuning MPT 7B on a dataset derived from <a href="#">databricks-dolly-15k</a> and the <a href="#">Anthropic Helpful and Harmless (HH-RLHF)</a> datasets.

## Dataset file types and input data format

Instruction-based fine-tuning uses labeled datasets to improve the performance of pre-trained LLMs on specific natural language processing (NLP) tasks. The labeled examples are formatted as prompt-response pairs and phrased as instructions.

To learn about the supported dataset file types, see [Supported dataset file types](#).

To learn about input data format, see [Input data format for instruction-based fine-tuning](#).

### Supported dataset file types

Autopilot supports instruction-based fine-tuning datasets formatted as CSV files (default) or as Parquet files.

- **CSV** (comma separated values) is a row-based file format that stores data in human readable plaintext, which is a popular choice for data exchange as it is supported by a wide range of applications.
- **Parquet** is a binary, column-based file format where the data is stored and processed more efficiently than in human readable file formats such as CSV. This makes it a better option for big data problems.

#### Note

The dataset may consist of multiple files, each of which must adhere to a specific template. For information on how to format your input data, see [Input data format for instruction-based fine-tuning](#).

## Input data format for instruction-based fine-tuning

Each file in the dataset must adhere to the following format:

- The dataset must contain exactly two comma-separated and named columns, `input` and `output`. Autopilot does not allow any additional columns.
- The `input` columns contain the prompts, and their corresponding `output` contains the expected answer. Both the `input` and `output` are in string format.

The following example illustrates the input data format for instruction-based fine-tuning in Autopilot.

```
input, output  
"<prompt text>","<expected generated text>"
```

 **Note**

We recommend using datasets with a minimum of 1000 rows to ensure optimal learning and performance of the model.

Additionally, Autopilot sets a maximum limit on the number of rows in the dataset and the context length based on the type of model being used.

- The limits on the number of rows in a dataset apply to the cumulative count of rows across all files within the dataset, including multiple files. If there are two [channel types](#) defined (one for training and one for validation), the limit applies to the total number of rows across all datasets within both channels. When the number of rows exceeds the threshold, the job fails with a validation error.
- When the length of the input or output of a row in the dataset exceeds the limit set on the context of the language model, it is automatically truncated. If more than 60% of the rows in the dataset are truncated, whether in their input or output, Autopilot fails the job with a validation error.

The following table presents those limits for each model.

JumpStart Model ID	BaseModelName in API request	Row Limit	Context Length Limit
huggingface-textgeneration-dolly-v2-3b-bf16	Dolly3B	10,000 rows	1024 tokens
huggingface-textgeneration-dolly-v2-7b-bf16	Dolly7B	10,000 rows	1024 tokens

<b>JumpStart Model ID</b>	<b>BaseModelName in API request</b>	<b>Row Limit</b>	<b>Context Length Limit</b>
huggingface-textgeneration-dolly-v2-12b-bf16	Dolly12B	10,000 rows	1024 tokens
huggingface-llm-falcon-7b-bf16	Falcon7B	1,000 rows	1024 tokens
huggingface-llm-falcon-7b-instruct-bf16	Falcon7BInstruct	1,000 rows	1024 tokens
huggingface-llm-falcon-40b-bf16	Falcon40B	10,000 rows	1024 tokens
huggingface-llm-falcon-40b-instruct-bf16	Falcon40B Instruct	10,000 rows	1024 tokens
huggingface-text2text-flan-t5-large	FlanT5L	10,000 rows	1024 tokens
huggingface-text2text-flan-t5-xl	FlanT5XL	10,000 rows	1024 tokens
huggingface-text2text-flan-t5-xxl	FlanT5XXL	10,000 rows	1024 tokens
meta-textgeneration-llama-2-7b	Llama2-7B	10,000 rows	2048 tokens
meta-textgeneration-llama-2-7b-f	Llama2-7BChat	10,000 rows	2048 tokens
meta-textgeneration-llama-2-13b	Llama2-13B	7,000 rows	2048 tokens

JumpStart Model ID	BaseModelName in API request	Row Limit	Context Length Limit
meta-textgeneration-llama-2-13b-f	Llama2-13BChat	7,000 rows	2048 tokens
huggingface-llm-mistral-7b	Mistral7B	10,000 rows	2048 tokens
huggingface-llm-mistral-7b-instruct	Mistral7B Instruct	10,000 rows	2048 tokens
huggingface-textgeneration1-mpt-7b-bf16	MPT7B	10,000 rows	1024 tokens
huggingface-textgeneration1-mpt-7b-instruct-bf16	MPT7BInstruct	10,000 rows	1024 tokens

## Hyperparameters for optimizing the learning process of your text generation models

You can optimize the learning process of your base model by adjusting any combination of the following hyperparameters. These parameters are available for all models.

- Epoch Count:** The epochCount hyperparameter determines how many times the model goes through the entire training dataset. It influences the training duration and can prevent overfitting when set appropriately. Large number of epochs may increase the overall runtime of fine-tuning jobs. We recommend setting a large MaxAutoMLJobRuntimeInSeconds within the CompletionCriteria of the [TextGenerationJobConfig](#) to avoid fine-tuning jobs from stopping prematurely.
- Batch Size:** The batchSize hyperparameter defines the number of data samples used in each iteration of training. It can affect the convergence speed and memory usage. With large batch size, the risk of out of memory (OOM) errors increases, which may surface as an internal server error in Autopilot. To check for such error, check the /aws/sagemaker/TrainingJobs log group for the training jobs launched by your Autopilot job. You can access those logs in

CloudWatch from in the AWS management console. Choose **Logs**, and then choose the `/aws/sagemaker/TrainingJobs` **log group**. To remedy OOM errors, reduce the batch size.

We recommend starting with a batch size of 1, then incrementally increase it until an out of memory error occurs. As a reference, 10 epochs typically takes up to 72h to complete.

- **Learning Rate:** The `learningRate` hyperparameter controls the step size at which a model's parameters are updated during training. It determines how quickly or slowly the model's parameters are updated during training. A high learning rate means that the parameters are updated by a large step size, which can lead to faster convergence but may also cause the optimization process to overshoot the optimal solution and become unstable. A low learning rate means that the parameters are updated by a small step size, which can lead to more stable convergence but at the cost of slower learning.
- **Learning Rate Warmup Steps:** The `learningRateWarmupSteps` hyperparameter specifies the number of training steps during which the learning rate gradually increases before reaching its target or maximum value. This helps the model converge more effectively and avoid issues like divergence or slow convergence that can occur with an initially high learning rate.

To learn about how to adjust hyperparameters for your fine-tuning experiment in Autopilot and discover their possible values, see [How to set hyperparameters to optimize the learning process of a model](#).

## Metrics for fine-tuning large language models in Autopilot

The following section describes the metrics that you can use to understand your fine-tuned large language models (LLMs). Using your dataset, Autopilot directly fine-tunes a target LLM to enhance a default objective metric, the cross-entropy loss.

Cross-entropy loss is a widely used metric to assess the dissimilarity between the predicted probability distribution and the actual distribution of words in the training data. By minimizing cross-entropy loss, the model learns to make more accurate and contextually relevant predictions, particularly in tasks related to text generation.

After fine-tuning an LLM you can evaluate the quality of its generated text using a range of ROUGE scores. Additionally, you can analyze the perplexity and cross-entropy training and validation losses as part of the evaluation process.

- Perplexity loss measures how well the model can predict the next word in a sequence of text, with lower values indicating a better understanding of the language and context.

- Recall-Oriented Understudy for Gisting Evaluation (ROUGE) is a set of metrics used in the field of natural language processing (NLP) and machine learning to evaluate the quality of machine-generated text, such as text summarization or text generation. It primarily assesses the similarities between the generated text and the ground truth reference (human-written) text of a validation dataset. ROUGE measures are designed to assess various aspects of text similarity, including the precision and recall of n-grams (contiguous sequences of words) in the system-generated and reference texts. The goal is to assess how well a model captures the information present in the reference text.

There are several variants of ROUGE metrics, depending on the type of n-grams used and the specific aspects of text quality being evaluated.

The following list contains the name and description of the ROUGE metrics available after the fine-tuning of large language models in Autopilot.

### **ROUGE-1, ROUGE-2**

ROUGE-N, the primary ROUGE metric, measures the overlap of n-grams between the system-generated and reference texts. ROUGE-N can be adjusted to different values of n (here 1 or 2) to evaluate how well the system-generated text captures the n-grams from the reference text.

### **ROUGE-L**

ROUGE-L (ROUGE-Longest Common Subsequence) calculates the longest common subsequence between the system-generated text and the reference text. This variant considers word order in addition to content overlap.

### **ROUGE-L-Sum**

ROUGE-L-SUM (Longest Common Subsequence for Summarization) is designed for the evaluation of text summarization systems. It focuses on measuring the longest common subsequence between the machine-generated summary and the reference summary. ROUGE-L-SUM takes into account the order of words in the text, which is important in text summarization tasks.

## **Autopilot model deployment and predictions**

After fine-tuning a large language model (LLM), you can deploy the model for real-time text generation by setting up an endpoint to obtain interactive predictions.

**Note**

We recommend running real-time inference jobs on `m1.g5.12xlarge` for better performances. Alternatively, `m1.g5.8xlarge` instances are suitable for Falcon-7B-Instruct and MPT-7B-Instruct text generation tasks.

You can find the specifics of these instances within the [Accelerated Computing](#) category in the selection of instance types provided by Amazon EC2.

## Real-time text generation

You can use SageMaker APIs to manually deploy your fine-tuned model to a SageMaker AI Hosting [real-time inference endpoint](#), then begin making predictions by invoking the endpoint as follows.

**Note**

Alternatively, you can chose the automatic deployment option when creating your fine-tuning experiment in Autopilot. For information on setting up the automatic deployment of models, see [How to enable automatic deployment](#).

You can also use the SageMaker Python SDK and the `JumpStartModel` class to perform inferences with models fine-tuned by Autopilot. This can be done by specifying a custom location for the model's artifact in Amazon S3. For information on defining your model as a JumpStart model and deploying your model for inference, see [Low-code deployment with the `JumpStartModel` class](#).

## 1. Obtain the candidate inference container definitions

You can find the `InferenceContainerDefinitions` within the `BestCandidate` object retrieved from the response to the [DescribeAutoMLJobV2](#) API call. A container definition for inference refers to the containerized environment designed for deploying and running your trained model to make predictions.

The following AWS CLI command example uses the [DescribeAutoMLJobV2](#) API to obtain recommended container definitions for your job name.

```
aws sagemaker describe-auto-ml-job-v2 --auto-ml-job-name job-name --region region
```

## 2. Create a SageMaker AI model

Use the container definitions from the previous step to create a SageMaker AI model by using the [CreateModel](#) API. See the following AWS CLI command as an example. Use the CandidateName for your model name.

```
aws sagemaker create-model --model-name '<your-candidate-name>' \
    --primary-container '<container-definition>' \
    --execution-role-arn '<execution-role-arn>' --region '<region>'
```

### 3. Create an endpoint configuration

The following AWS CLI command example uses the [CreateEndpointConfig](#) API to create an endpoint configuration.

#### Note

To prevent the endpoint creation from timing out due to a lengthy model download, we recommend setting ModelDataDownloadTimeoutInSeconds = 3600 and ContainerStartupHealthCheckTimeoutInSeconds = 3600.

```
aws sagemaker create-endpoint-config --endpoint-config-name '<your-endpoint-config-name>' \
    --production-variants '<list-of-production-variants>' ModelDataDownloadTimeoutInSeconds=3600 \
    ContainerStartupHealthCheckTimeoutInSeconds=3600 \
    --region '<region>'
```

### 4. Create the endpoint

The following AWS CLI example uses the [CreateEndpoint](#) API to create the endpoint.

```
aws sagemaker create-endpoint --endpoint-name '<your-endpoint-name>' \
    --endpoint-config-name '<endpoint-config-name-you-just-created>' \
    --region '<region>'
```

Check the progress of your endpoint deployment by using the [DescribeEndpoint](#) API. See the following AWS CLI command as an example.

```
aws sagemaker describe-endpoint --endpoint-name '<endpoint-name>' --region <region>
```

After the EndpointStatus changes to InService, the endpoint is ready to use for real-time inference.

## 5. Invoke the endpoint

The following command invokes the endpoint for real-time inferencing. Your prompt needs to be encoded in bytes.

### Note

The format of your input prompt depends on the language model. For more information on the format of text generation prompts, see [Request format for text generation models real-time inference](#).

```
aws sagemaker invoke-endpoint --endpoint-name '<endpoint-name>' \  
    --region '<region>' --body '<your-promt-in-bytes>' [--content-type] \  
    'application/json' <outfile>
```

## Request format for text generation models real-time inference

Different large language models (LLMs) may have specific software dependencies, runtime environments, and hardware requirements influencing Autopilot's recommended container to host the model for inference. Additionally, each model dictates the required input data format and the expected format for predictions and outputs.

Here are example inputs for some models and recommended containers.

- For Falcon models with the recommended container `huggingface-pytorch-tgi-inference:2.0.1-tgi1.0.3-gpu-py39-cu118-ubuntu20.04`:

```
payload = {  
    "inputs": "Large language model fine-tuning is defined as",  
    "parameters": {  
        "do_sample": false,  
        "top_p": 0.9,
```

```
        "temperature": 0.1,  
        "max_new_tokens": 128,  
        "stop": ["<|endoftext|>", "</s>"]  
    }  
}
```

- For all other models with the recommended container `djl-inference:0.22.1-fastertransformer5.3.0-cu118`:

```
payload= {  
    "text_inputs": "Large language model fine-tuning is defined as"  
}
```

## Create a Regression or Classification Autopilot experiment for tabular data using the Studio Classic UI

### Important

As of November 30, 2023, Autopilot's UI is migrating to [Amazon SageMaker Canvas](#) as part of the updated [Amazon SageMaker Studio](#) experience. SageMaker Canvas provides analysts and citizen data scientists no-code capabilities for tasks such as data preparation, feature engineering, algorithm selection, training and tuning, inference, and more. Users can leverage built-in visualizations and what-if analysis to explore their data and different scenarios, with automated predictions enabling them to easily productionize their models. Canvas supports a variety of use cases, including computer vision, demand forecasting, intelligent search, and generative AI.

Users of [Amazon SageMaker Studio Classic](#), the previous experience of [Studio](#), can continue using the Autopilot UI in Studio Classic. Users with coding experience can continue using all [API references](#) in any supported SDK for technical implementation.

If you have been using Autopilot in Studio Classic until now and want to migrate to SageMaker Canvas, you might have to grant additional permissions to your user profile or IAM role so that you can create and use the SageMaker Canvas application. For more information, see [the section called “\(Optional\) Migrate from Autopilot in Studio Classic to SageMaker Canvas”](#).

All UI-related instructions in this guide pertain to Autopilot's standalone features before migrating to [Amazon SageMaker Canvas](#). Users following these instructions should use [Studio Classic](#).

You can use the Amazon SageMaker Studio Classic UI to create Autopilot experiments for classification or regression problems on tabular data. The UI helps you specify the name of your experiment, provide locations for the input and output data, and specify which target data to predict. Optionally, you can also specify the type of problem that you want to solve (regression, classification, multiclass classification), choose your modeling strategy (*stacked ensembles* or *hyperparameters optimization*), select the list of algorithms used by the Autopilot job to train the data, and more.

The UI has descriptions, toggle switches, dropdown menus, radio buttons, and more to help you navigate creating your model candidates. After the experiment runs, you can compare trials and delve into the details of the pre-processing steps, algorithms, and hyperparameter ranges of each model. Optionally, you can download their [explainability](#) and [performance](#) reports. Use the provided [notebooks](#) to see the results of the automated data exploration or the candidate model definitions.

Alternatively, you can use Autopilot AutoML API in [Create Regression or Classification Jobs for Tabular Data Using the AutoML API](#).

## Configure the default parameters of an Autopilot experiment (for administrators)

Autopilot supports setting default values to simplify the configuration of Amazon SageMaker Autopilot when you create an Autopilot experiment using the Studio Classic UI. Administrators can use Studio Classic [lifecycle configurations](#) (LCC) to set infrastructure, networking, and security values in configuration files and pre-populate the [advanced settings](#) of AutoML jobs.

By doing so, they can fully control network connectivity and access permissions for the resources associated with Amazon SageMaker Studio Classic, including SageMaker AI instances, data sources, output data, and other related services. Specifically, administrators can configure a desired network architecture, such as Amazon VPC, subnets, and security groups, for a Studio Classic domain or individual user profiles. Data scientists can focus on data science specific parameters when creating their Autopilot experiments using the Studio Classic UI. Furthermore, administrators can manage the encryption of data on the instance in which Autopilot experiments run by setting default encryption keys.

**Note**

This feature is currently not available in the Asia Pacific (Hong Kong) and Middle East (Bahrain) opt-in Regions.

In the following sections, you can find the full list of parameters supporting the setting of defaults when creating an Autopilot experiment using the Studio Classic UI, and learn how to set those default values.

**Topics**

- [List of default parameters supported](#)
- [Set default Autopilot experiment parameters](#)

**List of default parameters supported**

The following parameters support setting default values with a configuration file for creating an Autopilot experiment using the Studio Classic UI. Once set, the values automatically fill in their corresponding field in the Autopilot' **Create Experiment** tab in the Studio Classic UI. See [Advanced settings \(optional\)](#) for a full description of each field.

- **Security:** Amazon VPC, subnets, and security groups.
- **Access:** AWS IAM role ARNs.
- **Encryption:** AWS KMS key IDs.
- **Tags:** Key-value pairs used to label and organize SageMaker AI resources.

**Set default Autopilot experiment parameters**

Administrators can set default values in a configuration file, then manually place the file in a recommended location within the Studio Classic environment of specific users, or they can pass the file to a lifecycle configuration script (LCC) to automate the customization of the Studio Classic environment for a given domain or user profile.

- To set up the configuration file, start by filling in its default parameters.

To configure any or all default values listed in [List of default parameters supported](#), administrators can create a configuration file named `config.yaml`, the structure of which

should adhere to this [sample configuration file](#). The following snippet shows a sample configuration file with all the supported AutoML parameters. For more information on the format of this file, refer to the [full schema](#).

```
SchemaVersion: '1.0'
SageMaker:
  AutoMLJob:
    # https://docs.aws.amazon.com/sagemaker/latest/APIReference/API_CreateAutoMLJob.html
    AutoMLJobConfig:
      SecurityConfig:
        EnableInterContainerTrafficEncryption: true
        VolumeKmsKeyId: 'kms-key-id'
      VpcConfig:
        SecurityGroupIds:
          - 'security-group-id-1'
          - 'security-group-id-2'
        Subnets:
          - 'subnet-1'
          - 'subnet-2'
      OutputDataConfig:
        KmsKeyId: 'kms-key-id'
      RoleArn: 'arn:aws:iam::1112223344:role/Admin'
      Tags:
        - Key: 'tag_key'
          Value: 'tag_value'
```

- Then, place the configuration file in the recommended location by either [manually copying the file](#) to its recommended paths or using a [lifecycle configuration](#) (LCC).

The configuration file needs to be present in at least one of the following locations in the user's Studio Classic environment. By default, SageMaker AI searches for a configuration file in two locations:

- First, in `/etc/xdg/sagemaker/config.yaml`. We refer to this file as the *administrator configuration file*.
- Then, in `/root/.config/sagemaker/config.yaml`. We refer to this file as the *user configuration file*.

Using the *administrator* configuration file, administrators can define a set of default values. Optionally, they can use the *user* configuration file to override values set in the *administrator* configuration file, or set additional default parameter values.

The following snippet shows a sample script which writes the default parameters configuration file to the *administrator* location in the user's Studio Classic environment. You can replace `/etc/xdg/sagemaker` with `/root/.config/sagemaker` to write the file to the *user* location.

```
## Sample script with AutoML intelligent defaults
#!/bin/bash

sudo mkdir -p /etc/xdg/sagemaker

echo "SchemaVersion: '1.0'
CustomParameters:
  AnyStringKey: 'AnyStringValue'
SageMaker:
  AutoMLJob:
    # https://docs.aws.amazon.com/sagemaker/latest/APIReference/API_CreateAutoMLJob.html
    AutoMLJobConfig:
      SecurityConfig:
        EnableInterContainerTrafficEncryption: true
        VolumeKmsKeyId: 'kms-key-id'
      VpcConfig:
        SecurityGroupIds:
          - 'security-group-id-1'
          - 'security-group-id-2'
        Subnets:
          - 'subnet-1'
          - 'subnet-2'
      OutputDataConfig:
        KmsKeyId: 'kms-key-id'
      RoleArn: 'arn:aws:iam::1112223344:role/Admin'
      Tags:
        - Key: 'tag_key'
          Value: 'tag_value'"
" | sudo tee /etc/xdg/sagemaker/config.yaml
```

- **Copy the files manually** – To copy the configuration files manually, run the [script](#) created in the previous step from a Studio Classic terminal. In this case, the user profile that executed the script can create Autopilot experiments with the default values applicable only to them.
- **Create a SageMaker AI lifecycle configuration** – Alternatively, you can use a [lifecycle configuration](#) (LCC) to automate the customization of your Studio Classic environment. LCC are shell scripts triggered by Amazon SageMaker Studio Classic lifecycle events such as starting a

Studio Classic application. This customization includes installing custom packages, configuring notebook extensions, pre-loading datasets, setting up source code repositories, or, in our case, pre-populating default parameters. Administrators can attach the LCC to a Studio Classic domain to automate the configuration of default values for each user profile within that domain.

The following sections detail how to create a lifecycle configuration so users can load Autopilot default parameters automatically when launching Studio Classic. You can choose to create an LCC using the SageMaker AI Console or the AWS CLI.

### Create a LCC from the SageMaker AI Console

Use the following steps to create an LCC containing your default parameters, attach the LCC to a domain or a user profile, then launch a Studio Classic application pre-populated with the default parameters set by the LCC using the SageMaker AI Console.

- **To create a lifecycle configuration that runs the [script](#) containing your default values using the SageMaker AI Console**
  - Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
  - On the left side, navigate to **Admin configurations**, then **Lifecycle configurations**.
  - From the **Lifecycle configurations** page, navigate to the Studio Classic tab, then choose **Create configuration**.
  - For **Name**, type a name using alphanumeric characters and "-", but no spaces. The name can have a maximum of 63 characters.
  - Paste your [script](#) in the **Scripts** section.
  - Choose **Create configuration** to create the lifecycle configuration. This creates an LCC of type Kernel gateway app.
- **To attach the lifecycle configuration to a Studio Classic domain, a space, or a user profile**

Follow the steps in [Attach the lifecycle configuration to Studio Classic domain or user profile](#) to attach your LCC to a Studio Classic domain or a specific user profile.

- **To launch your Studio Classic application with the lifecycle configuration**

Once the LCC is attached to a domain or a user profile, impacted users can start a Studio Classic application from the landing page of Studio Classic in Studio to pick up the defaults set by the LCC automatically. This auto-populates the Studio Classic UI when [creating an Autopilot experiment](#).

## Create a LCC from the AWS CLI

Use the following snippets to launch a Studio Classic application that runs your [script](#) using the AWS CLI. Note that `lifecycle_config.sh` is the name given to your script in this example.

Before getting started:

- Ensure that you have updated and configured AWS CLI by completing the prerequisites described in [Create a lifecycle configuration from the AWS CLI](#).
- Install [OpenSSL](#) documentation. The AWS CLI command uses the open-source library `OpenSSL` to encode your script in Base64 format. This requirement prevents errors that occur from spacing and line break encoding.

You can now follow these three steps:

- **Create a new lifecycle configuration referencing the configuration script `lifecycle_config.sh`**

```
LCC_CONTENT=`openssl base64 -A -in lifecycle_config.sh`  
  
## Create a new lifecycle config  
aws sagemaker create-studio-lifecycle-config --region region \  
--studio-lifecycle-config-name lcc-name \  
--studio-lifecycle-config-content $LCC_CONTENT \  
--studio-lifecycle-config-app-type default
```

Note the ARN of the newly created lifecycle configuration that is returned. This ARN is required to attach the lifecycle configuration to your application.

- **Attach the lifecycle configuration to your JupyterServerApp**

The following example shows how to create a new user profile with a lifecycle configuration attached. To update an existing user profile, use the AWS CLI [update-user-profile](#) command. To create or update a domain, see [create-domain](#) and [update-domain](#). Add the lifecycle configuration ARN from the previous step to the settings of the JupyterServerAppSettings application type. You can add multiple lifecycle configurations at the same time by using a list of lifecycle configurations.

```
# Create a new UserProfile  
aws sagemaker create-user-profile --domain-id domain-id \  
--user-profile-name user-profile-name \  
--user-profile-settings { "JupyterServerAppSettings": { "LifecycleConfigArn": "arn:aws:sagemaker:us-west-2:123456789012:studio-lifecycle-config/MyLCC" } }
```

```
--user-profile-name user-profile-name \
--region region \
--user-settings '{
  "JupyterServerAppSettings": {
    "LifecycleConfigArns": [
      "lifecycle-configuration-arn"
    ]
  }
}'
```

Once the LCC is attached to a domain or a user profile, impacted users can shut down and update their existing Studio Classic application by following the steps in [Shut down and Update Amazon SageMaker Studio Classic](#), or start a new Studio Classic application from the AWS Console to pick up the defaults set by the LCC automatically. This auto-populates the Studio Classic UI when creating an Autopilot experiment. Alternatively, they can launch a new Studio Classic application using the AWS CLI as follows.

- **Launch your Studio Classic application with the lifecycle configuration using the AWS CLI**

```
# Create a Jupyter Server application
aws sagemaker create-app --domain-id domain-id \
--user-profile-name user-profile-name \
--region region \
--app-type JupyterServer \
--resource-spec LifecycleConfigArn=lifecycle-configuration-arn \
--app-name default
```

For more information on creating a lifecycle configuration using the AWS CLI, see [Create a Lifecycle Configuration from the AWS CLI](#).

## To create an Autopilot experiment using Studio Classic UI

1. Sign in at <https://console.aws.amazon.com/sagemaker/>, choose **Studio** from the left navigation pane, select your Domain and user profile, then **Open Studio**.
2. In Studio, choose the Studio Classic icon in the top left navigation pane. This opens a Studio Classic app.
3. Run or open a Studio Classic application from the space of your choice, or **Create Studio Classic space**. On the **Home** tab, choose the **AutoML** card. This opens a new **AutoML** tab.
4. Choose **Create an AutoML experiment**. This opens a new **Create experiment** tab.

5. In the **Experiment and data details** section, enter the following information:
  - a. **Experiment name** – Must be unique to your account in the current AWS Region and contain a maximum of 63 alphanumeric characters. Can include hyphens (-) but not spaces.
  - b. **Input data** – Provide the Amazon Simple Storage Service (Amazon S3) bucket location of your input data. This S3 bucket must be in your current AWS Region. The URL must be in an s3:// format where Amazon SageMaker AI has write permissions. The file must be in CSV or Parquet format and contain at least 500 rows. Select **Browse** to scroll through available paths and **Preview** to see a sample of your input data.
  - c. **Is your S3 input a manifest file?** – A manifest file includes metadata with your input data. The metadata specifies the location of your data in Amazon S3. It also specifies how the data is formatted and which attributes from the dataset to use when training your model. You can use a manifest file as an alternative to preprocessing when your labeled data is being streamed in Pipe mode.
  - d. **Auto split data?** – Autopilot can split your data into an 80-20% split for training and validation data. If you prefer a custom split, you can choose the **Specify split ratio**. To use a custom dataset for validation, choose **Provide a validation set**.
  - e. **Output data location (S3 bucket)** – The name of the S3 bucket location where you want to store the output data. The URL for this bucket must be in an Amazon S3 format where Amazon SageMaker AI has write permissions. The S3 bucket must be in the current AWS Region. Autopilot can also create this for you in the same location as your input data.
6. Choose **Next: Target and features**. The **Target and features** tab opens.
7. In the **Target and features** section:
  - Select a column to set as a target for model predictions.
  - Optionally, you can pass the name of a sample weights column in the **Sample weight** section to request your dataset rows to be weighted during training and evaluation. For more information on the available objective metrics, see [Autopilot weighted metrics](#).

 **Note**

Support for sample weights is available in [ensembling mode](#) only.

- You can also select features for training and change their data type. The following data types are available: Text, Numerical, Categorical, Datetime, Sequence, and Auto. All features are selected by default.
8. Choose **Next: Training method**. The **Training method** tab opens.
9. In the **Training method** section, select your training option: **Ensembling, Hyperparameter optimization (HPO)**, or **Auto** to let Autopilot choose the training method automatically based on the dataset size. Each training mode runs a pre-defined set of algorithms on your dataset to train model candidates. By default, Autopilot pre-selects all the available algorithms for the given training mode. You can run an Autopilot training experiment with all the algorithms or choose your own subset.

For more information on the training modes and the available algorithms, see the **Autopilot training modes** section in the [Training modes and algorithms](#) page.

10. Choose **Next: Deployment and advanced settings** to open the **Deployment and advanced settings** tab. Settings include the auto-display endpoint name, machine learning problem type, and additional choices for running your experiment.
- a. **Deployment settings** – Autopilot can automatically create an endpoint and deploy your model for you.

To auto-deploy to an automatically generated endpoint, or to provide an endpoint name for custom deployment, set the toggle to **Yes** under **Auto deploy?** If you are importing data from Amazon SageMaker Data Wrangler, you have additional options to auto-deploy the best model with or without the transforms from Data Wrangler.

 **Note**

If your Data Wrangler flow contains multi-row operations such as groupby, join, or concatenate, you can't auto-deploy with these transforms. For more information, see [Automatically Train Models on Your Data Flow](#).

- b. **Advanced settings (optional)** – Autopilot provides additional controls to manually set experimental parameters such as defining your problem type, time constraints on your Autopilot job and trials, security, and encryption settings.

**Note**

Autopilot supports the setting of default values to simplify the configuration of Autopilot experiments using Studio Classic UI. Administrators can use Studio Classic [lifecycle configurations](#) (LCC) to set infrastructure, networking, and security values in configuration files and pre-populate the *advanced settings* of AutoML jobs.

To learn about how administrators can automate the customization of an Autopilot experiment, see [Configure the default parameters of an Autopilot experiment \(for administrators\)](#).

- i. **Machine learning problem type** – Autopilot can automatically infer the type of supervised learning problem from your dataset. If you prefer to choose it manually, you can use the **Select the machine learning problem type** dropdown menu. Note that it defaults to **Auto**. In some cases, SageMaker AI is unable to infer accurately. When that happens, you must provide the value for the job to succeed. In particular, you can choose from the following types:
  - **Binary classification** – Binary classification assigns input data to one of two predefined and mutually exclusive classes, based on their attributes, such as medical diagnosis based on results of diagnostic tests that determine if someone has a disease.
  - **Regression** – Regression establishes a relationship between the input variables (also known as independent variables or features) and the target variable (also known as the dependent variable). This relationship is captured through a mathematical function or model that maps the input variables to a continuous output. It is commonly used for tasks such as predicting house prices based on features like square footage and the number of bathrooms, stock market trends, or estimating sales figures.
  - **Multiclass classification** – Multiclass classification assigns input data to one of several classes based on their attributes, like the prediction of the topic most relevant to a text document, such as politics, finance, or philosophy.
- ii. **Runtime** – You can define a maximum time limit. Upon reaching the time limit, trials and jobs that exceed the time constraint automatically stop.

- iii. **Access** – You can choose the role that Amazon SageMaker Studio Classic assumes to gain temporary access to AWS services (in particular, SageMaker AI and Amazon S3) on your behalf. If no role is explicitly defined, Studio Classic automatically uses the default SageMaker AI execution role attached to your user profile.
  - iv. **Encryption** – To enhance the security of your data at rest and protect it against unauthorized access, you can specify encryption keys to encrypt data in your Amazon S3 buckets and in the Amazon Elastic Block Store (Amazon EBS) volume attached to your Studio Classic domain.
  - v. **Security** – You can choose the virtual private cloud (Amazon VPC) in which your SageMaker AI job runs. Ensure that the Amazon VPC has access to your input and output Amazon S3 buckets.
  - vi. **Project** – Specify the name of the SageMaker AI project to associate with this Autopilot experiment and model outputs. When you specify a project, Autopilot tags the project to an experiment. This lets you know which model outputs are associated with this project.
  - vii. **Tags** – Tags are an array of key-value pairs. Use tags to categorize your resources from AWS services, such as their purpose, owner, or environment.
- c. Choose **Next: Review and create** to get a summary of your Autopilot experiment before you create it.
11. Select **Create experiment**. The creation of the experiment starts an Autopilot job in SageMaker AI. Autopilot provides the status of the experiment, information on the data exploration process and model candidates in notebooks, a list of generated models and their reports, and the job profile used to create them.

For information on the notebooks generated by an Autopilot job, see [Autopilot notebooks generated to manage AutoML tasks](#). For information on the details of each model candidate and their reports, see [View model details](#) and [View an Autopilot model performance report](#).

 **Note**

To avoid incurring unnecessary charges: If you deploy a model that is no longer needed, delete the endpoints and resources that were created during that deployment. Information about pricing instances by Region is available at [Amazon SageMaker Pricing](#).

## Amazon SageMaker Autopilot example notebooks

The following notebooks serve as practical, hands-on examples that address various use cases of Autopilot.

You can find all of Autopilot's notebooks in the [autopilot](#) directory of SageMaker AI GitHub examples repository.

We recommend cloning the full Git repository within Studio Classic to access and run the notebooks directly. For information on how to clone a Git repository in Studio Classic, see [Clone a Git Repository in SageMaker Studio Classic](#).

Use case	Description
<a href="#">Serverless inference</a>	By default, Autopilot allows deploying generated models to real-time inference endpoints. In this repository, the notebook illustrates how to deploy Autopilot models trained with ENSEMBLING and HYPERPARAMETER OPTIMIZATION (HPO) modes to serverless endpoints. Serverless endpoints automatically launch compute resources and scale them in and out depending on traffic, eliminating the need to choose instance types or manage scaling policies.
<a href="#">Custom feature selection</a>	Autopilot inspects your data set, and runs a number of candidates to figure out the optimal combination of data preprocessing steps, machine learning algorithms, and hyperparameters. You can easily deploy either on a real-time endpoint or for batch processing.  In some cases, you might want to have the flexibility to bring custom data processing code to Autopilot. For example, your datasets might contain a large number of

Use case	Description
	independent variables, and you may wish to incorporate a custom feature selection step to remove irrelevant variables first. The resulting smaller dataset can then be used to launch an Autopilot job. Ultimately, you would also want to include both the custom processing code and models from Autopilot for real-time or batch processing.

Use case	Description
<a href="#">Pipeline example</a>	<p>While Autopilot streamlines the process of building ML models, MLOps engineers are still responsible for creating, automating, and managing end-to-end ML workflows in production. SageMaker Pipelines can assist in automating various steps of the ML lifecycle, such as data preprocessing, model training, hyperparameter tuning, model evaluation, and deployment. This notebook serves as a demonstration of how to incorporate Autopilot into a SageMaker Pipelines end-to-end AutoML training workflow. To launch an Autopilot experiment within Pipelines, you must create a model-building workflow by writing custom integration code using Pipelines <a href="#">Lambda</a> or <a href="#">Processing</a> steps. For more information, refer to <a href="#">Move Amazon SageMaker Autopilot ML models from experimentation to production using Amazon SageMaker Pipelines</a>.</p> <p>Alternatively, when using Autopilot in <a href="#">Ensembling mode</a>, you can refer to the notebook example that demonstrates how to use native AutoML step in <a href="#">SageMaker Pipeline's native AutoML step</a>. With Autopilot supported as a native step within Pipelines, you can now add an automated training step (<a href="#">AutoMLStep</a>) to your Pipelines and invoke an Autopilot experiment in Ensembling mode.</p>

Use case	Description
<a href="#"><u>Direct marketing with Amazon SageMaker Autopilot</u></a>	This notebook demonstrates how uses the <a href="#"><u>Bank Marketing Data Set</u></a> to predict whether a customer will enroll for a term deposit at a bank. You can use Autopilot on this dataset to get the most accurate ML pipeline by exploring options contained in various candidate pipelines. Autopilot generates each candidate in a two-step procedure. The first step performs automated feature engineering on the dataset. The second step trains and tunes an algorithm to produce a model. The notebook contains instructions on how to train the model and how to deploy the model to perform batch inference using the best candidate.
<a href="#"><u>Customer Churn Prediction with Amazon SageMaker Autopilot</u></a>	This notebook describes using machine learning for the automated identification of unhappy customers, also known as customer churn prediction. The example shows how to analyze a publicly available dataset and perform feature engineering on it. Next it shows how to tune a model by selecting the best performing pipeline along with the optimal hyperparameters for the training algorithm. Finally, it shows how to deploy the model to a hosted endpoint and how to evaluate its predictions against ground truth. However, ML models rarely give perfect predictions. That's why this notebook also shows how to incorporate the relative costs of prediction mistakes when determining the financial outcome of using ML.

Use case	Description
<a href="#"><u>Top Candidates Customer Churn Prediction with Amazon SageMaker Autopilot and Batch Transform (Python SDK)</u></a>	<p>This notebook also describes using machine learning for the automated identification of unhappy customers, also known as customer churn prediction. This notebook demonstrates how to configure the model to obtain the inference probability, select the top N models, and make Batch Transform on a hold-out test set for evaluation.</p> <div data-bbox="833 635 1530 910" style="border: 1px solid #ccc; padding: 10px;"><p> <b>Note</b> This notebook works with SageMaker Python SDK &gt;= 1.65.1 released on 6/19/2020.</p></div>
<a href="#"><u>Bringing your own data processing code to Amazon SageMaker Autopilot</u></a>	<p>This notebook demonstrates how to incorporate and deploy custom data processing code when using Amazon SageMaker Autopilot . It adds a custom feature selection step to remove irrelevant variables to an Autopilot job. It then shows how to deploy both the custom processing code and models generated by Autopilot on a real-time endpoint and, alternatively, for batch processing.</p>
More notebooks	You can find more notebooks illustrating other use cases such as <a href="#"><u>batch transform</u></a> , <a href="#"><u>time-series forecasting</u></a> and more in the root directory.

## Videos: Use Autopilot to automate and explore the machine learning process

Here is a video series that provides a tour of Amazon SageMaker Autopilot capabilities using Studio Classic. They show how to start an AutoML job, analyze and preprocess data, how to do feature engineering and hyperparameter optimization on candidate models, and how to visualize and compare the resulting model metrics.

### Topics

- [Start an AutoML job with Amazon SageMaker Autopilot](#)
- [Review data exploration and feature engineering automated in Autopilot.](#)
- [Tune models to optimize performance](#)
- [Choose and deploy the best model](#)
- [Amazon SageMaker Autopilot tutorial](#)

### **Start an AutoML job with Amazon SageMaker Autopilot**

This video shows you how to start an AutoML job with Autopilot. (Length: 8:41)

[Amazon SageMaker Studio - AutoML with Amazon SageMaker Autopilot \(part 1\)](#)

### **Review data exploration and feature engineering automated in Autopilot.**

This video shows you how to review the data exploration and candidate definition notebooks generated by Amazon SageMaker Autopilot. (Length: 10:04)

[Amazon SageMaker Studio - AutoML with Amazon SageMaker Autopilot \(part 2\)](#)

### **Tune models to optimize performance**

This video shows you how to optimize model performance during training using hyperparameter tuning. (Length: 4:59)

[SageMaker Studio - AutoML with Amazon SageMaker Autopilot \(part 3\)](#)

### **Choose and deploy the best model**

This video shows you how to use job metrics to choose the best model and then how to deploy it. (Length: 5:20)

## [SageMaker Studio - AutoML with Amazon SageMaker Autopilot \(part 4\)](#)

### Amazon SageMaker Autopilot tutorial

This video walks you through an end to end demo where we first build a binary classification model automatically with Amazon SageMaker Autopilot. We see how candidate models have been built and optimized using auto-generated notebooks. We also look at the top candidates with Amazon SageMaker Experiments. Finally, we deploy the top candidate (based on XGBoost), and configure data capture with SageMaker Model Monitor.

#### [End to end demo with AutoML on SageMaker AI](#)

### Tutorials: Get started with Amazon SageMaker Autopilot

Get started tutorials for Autopilot demonstrate how to create a machine learning model automatically without writing code. They show you how Autopilot simplifies the machine learning experience by helping you explore your data and try different algorithms. Autopilot builds the best machine learning model for the problem type using AutoML capabilities while allowing full control and visibility.

- [Create a machine learning model automatically with Autopilot](#): You assume the role of a developer working at a bank in this tutorial. You have been asked to develop a machine learning model to predict if a customer will enroll for a certificate of deposit (CD). This is a binary classification problem. The model is trained on the marketing dataset that contains information on customer demographics, responses to marketing events, and external factors.

### Autopilot quotas

There are quotas that limit the resources available to you when using Amazon SageMaker Autopilot. Some of these limits are increaseable and some are not.

#### Note

The resource quotas documented in the following sections are valid for versions of Amazon SageMaker Studio Classic 3.22.2 and higher. For information on updating your version of SageMaker Studio Classic, see [Shut Down and Update SageMaker Studio Classic and Studio Classic Apps](#).

## Topics

- [Quotas that you can increase](#)
- [Resource quotas](#)

## Quotas that you can increase

The following table contains the resource limits for quotas you can increase:

Resource	Regions	Default limits	Can be increased up to
Size of input dataset	All	100 GB	Hundreds of GBs
Size of a single Parquet file*	All	2 GB	N/A
Target dataset size for subsampling**	All	5 GB	Hundreds of GBs
Number of concurrent Autopilot jobs	us-east-1, us-east-2, us-west-2, ap-northeast-1, eu-west-1, eu-central-1	4	Hundreds
Number of concurrent Autopilot jobs	ap-northeast-2, ap-southeast-2, eu-west-2, ap-southest-1	2	Hundreds
Number of concurrent Autopilot jobs	All other Regions	1	Tens

### Note

\*This 2 GB size limit is for a single compressed Parquet file. You can provide a Parquet dataset that includes multiple compressed Parquet files up to the input dataset maximum size. After the files are decompressed, they may each expand to a larger size.

\*\*Autopilot automatically subsamples input datasets that are larger than the target dataset size while accounting for class imbalance and preserving rare class labels.

### To request a quota increase:

1. Open the [Service Quotas console](#).
2. Select your quota increase, then choose **Request increase at account level**.
3. In the **Increase quota value**, enter the new limit value that you are requesting.
4. Choose **Request**.

## Resource quotas

The following table contains the runtime resource limits for an Amazon SageMaker Autopilot job in an AWS Region.

Resource	Limit per Autopilot job
Maximum runtime for an Autopilot job	30 days

## API Reference guide for Autopilot

This section provides a subset of the HTTP service REST APIs for creating and managing Amazon SageMaker Autopilot resources (AutoML jobs) programmatically.

If your language of choice is Python, you can refer to [AWS SDK for Python \(Boto3\)](#) or the [AutoMLV2 object](#) of the Amazon SageMaker Python SDK directly.

### AutoML API Actions

This list details the operations available in the Reference API to manage AutoML jobs programmatically.

- [CreateAutoMLJob](#)
- [CreateAutoMLJobV2](#)
- [DescribeAutoMLJob](#)

- [DescribeAutoMLJobV2](#)
- [ListAutoMLJobs](#)
- [ListCandidatesForAutoMLJob](#)
- [StopAutoMLJob](#)

 **Note**

[CreateAutoMLJobV2](#) and [DescribeAutoMLJobV2](#) are new versions of [CreateAutoMLJob](#) and [DescribeAutoMLJob](#) which offer backward compatibility.

We recommend using the CreateAutoMLJobV2. CreateAutoMLJobV2 can manage tabular problem types identical to those of its previous version CreateAutoMLJob, as well as non-tabular problem types such as image or text classification, or time-series forecasting.

Find guidelines about how to migrate a CreateAutoMLJob to CreateAutoMLJobV2 in [Migrate a CreateAutoMLJob to CreateAutoMLJobV2](#).

## AutoML API Data Types

This list details the API AutoML objects used by the actions above as inbound requests or outbound responses.

- [AutoMLAlgorithmConfig](#)
- [AutoMLCandidate](#)
- [AutoMLCandidateGenerationConfig](#)
- [AutoMLCandidateStep](#)
- [AutoMLChannel](#)
- [AutoMLContainerDefinition](#)
- [AutoMLDataSource](#)
- [AutoMLDataSplitConfig](#)
- [AutoMLInferenceContainerDefinitions](#)
- [AutoMLJobArtifacts](#)
- [AutoMLJobChannel](#)
- [AutoMLJobCompletionCriteria](#)

- [AutoMLJobInputDataConfig](#)
- [AutoMLJobConfig](#)
- [AutoMLJobObjective](#)
- [AutoMLJobStepMetadata](#)
- [AutoMLJobSummary](#)
- [AutoMLOutputDataConfig](#)
- [AutoMLProblemTypeConfig](#)
- [AutoMLJobCompletionCriteria](#)
- [AutoMLJobSummary](#)
- [AutoMLOutputDataConfig](#)
- [AutoMLPartialFailureReason](#)
- [AutoMLProblemTypeConfig](#)
- [AutoMLProblemTypeResolvedAttributes](#)
- [AutoMLResolvedAttributes](#)
- [AutoMLSecurityConfig](#)
- [AutoMLS3DataSource](#)
- [CandidateArtifactLocations](#)
- [CandidateGenerationConfig](#)
- [CandidateProperties](#)
- [FinalAutoMLJobObjectiveMetric](#)
- [HolidayConfigAttributes](#)
- [ImageClassificationJobConfig](#)
- [MetricDatum](#)
- [ModelDeployConfig](#)
- [ModelDeployResult](#)
- [ResolvedAttributes](#)
- [TabularJobConfig](#)
- [TabularResolvedAttributes](#)
- [TextGenerationJobConfig](#)
- [TextGenerationResolvedAttribute](#)

- [TimeSeriesConfig](#)
- [TimeSeriesForecastingJobConfig](#)
- [TimeSeriesTransformations](#)
- [TuningJobCompletionCriteria](#)

## SageMaker JumpStart pretrained models

Amazon SageMaker JumpStart provides pretrained, open-source models for a wide range of problem types to help you get started with machine learning. You can incrementally train and tune these models before deployment. JumpStart also provides solution templates that set up infrastructure for common use cases, and executable example notebooks for machine learning with SageMaker AI.

You can deploy, fine-tune, and evaluate pretrained models from popular models hubs through the JumpStart landing page in the updated Studio experience.

You can also access pretrained models, solution templates, and examples through the JumpStart landing page in Amazon SageMaker Studio Classic.

The following steps show how to access JumpStart models using Amazon SageMaker Studio and Amazon SageMaker Studio Classic.

You can also access JumpStart models using the SageMaker Python SDK. For information about how to use JumpStart models programmatically, see [Use SageMaker JumpStart Algorithms with Pretrained Models](#).

## Open and use JumpStart in Studio

The following sections give information on how to open, use, and manage JumpStart from the Studio UI.

### Important

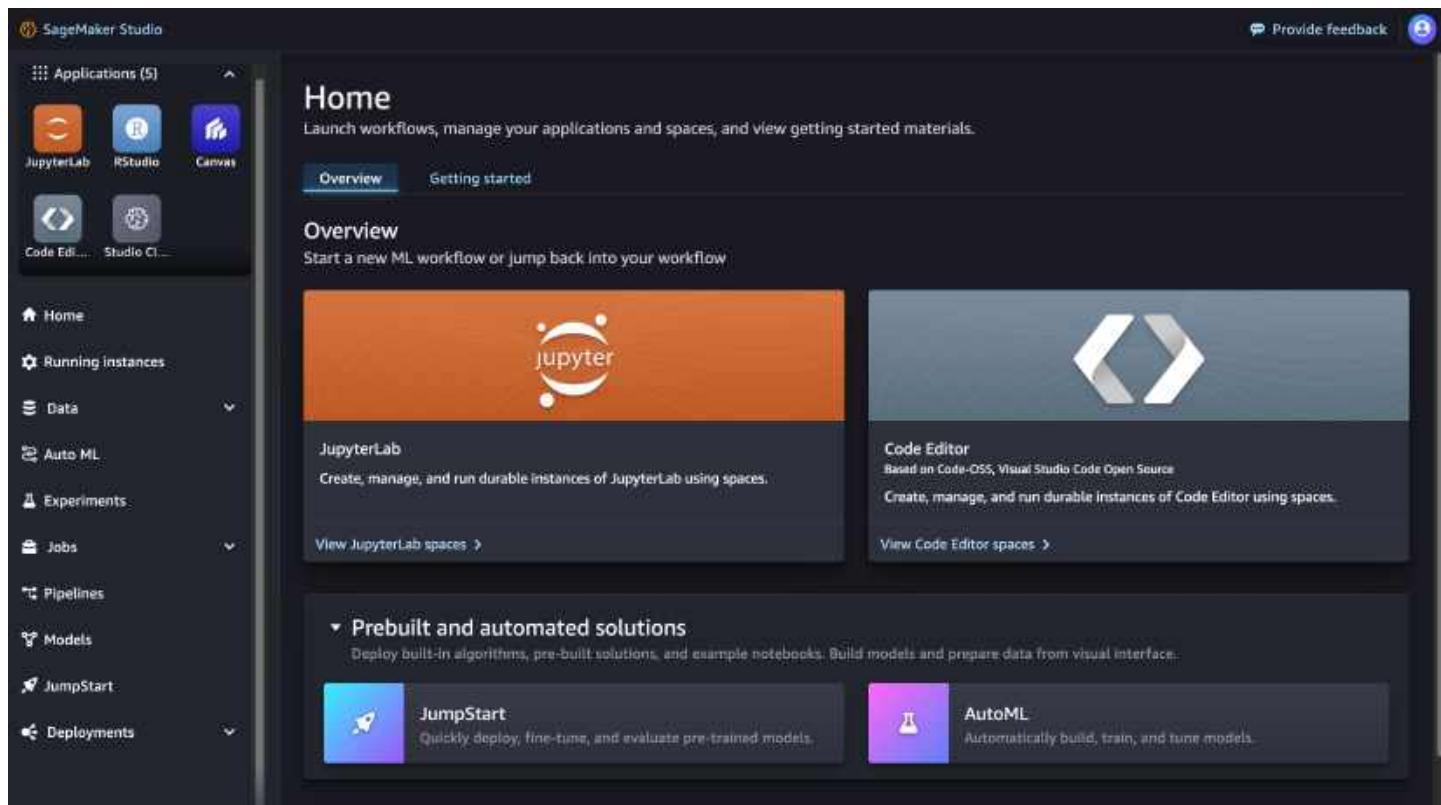
As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

## Open JumpStart in Studio

In Amazon SageMaker Studio, open the JumpStart landing page either through the **Home** page or the **Home** menu on the left-side panel. This opens the **SageMaker JumpStart** landing page where you can explore model hubs and search for models.

- From the **Home** page, choose **JumpStart** in the **Prebuilt and automated solutions** pane.
- From the **Home** menu in the left panel, navigate to the **SageMaker JumpStart** node.

For more information on getting started with Amazon SageMaker Studio, see [Amazon SageMaker Studio](#).



### ⚠️ Important

Before downloading or using third-party content: You are responsible for reviewing and complying with any applicable license terms and making sure that they are acceptable for your use case.

## Use JumpStart in Studio

From the **SageMaker JumpStart** landing page in Studio, you can explore model hubs from providers of both proprietary and publicly available models.

The screenshot shows the SageMaker JumpStart landing page with a dark background. At the top, there's a header with the title "JumpStart" and a sub-header: "Deploy, fine-tune, and evaluate pre-trained models from the most popular model hubs." Below this, a section titled "Hubs 10" includes a search bar labeled "Search hubs or models...". Six model hubs are displayed in a grid:

- HuggingFace**: Features a yellow emoji-like icon. Description: "Explore hundreds of popular and trending models from HuggingFace." Button: "View 4416 models >"
- Meta**: Features a blue icon with a white "n" inside. Description: "Explore popular and trending models from Meta including Llama, Code Llama, and more." Button: "View 240 models >"
- AI21 labs**: Features an orange icon with a white clipboard. Description: "Explore popular and trending models from AI21 Labs including Jurassic and more." Button: "View 96 models >"
- stability.ai**: Features a purple icon with two white "n" icons. Description: "Explore popular and trending models from Stability.ai including Stable Diffusion and more." Button: "View 160 models >"
- cohere**: Features a green icon with a speech bubble. Description: "Explore popular and trending models from Cohere including Command, Rerank, and more." Button: "View 64 models >"
- TensorFlow**: Features an orange icon with a white upward arrow. Description: "Explore popular and trending models from TensorFlow for computer vision and NLP tasks." Button: "View 5104 models >"

You can find specific hubs or models using the search bar. Within each model hub, you can search directly for models, sort by provided attributes, or filter based on a list of provided model tasks.

## Manage JumpStart in Studio

Choose a model to see its model detail card. In the upper right-hand corner of the model detail card, choose **Fine-tune**, **Deploy**, or **Evaluate** to start working through the fine-tuning, deployment, or evaluation workflows, respectively. Note that not all models are available for fine-tuning or evaluation. For more information on each of these options, see [Use foundation models in Studio](#).

## Open and use JumpStart in Studio Classic

The following sections give information on how to open, use, and manage JumpStart from the Amazon SageMaker Studio Classic UI.

### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

## Open JumpStart in Studio Classic

In Amazon SageMaker Studio Classic, open the JumpStart landing page either through the **Home** page or the **Home** menu on the left-side panel.

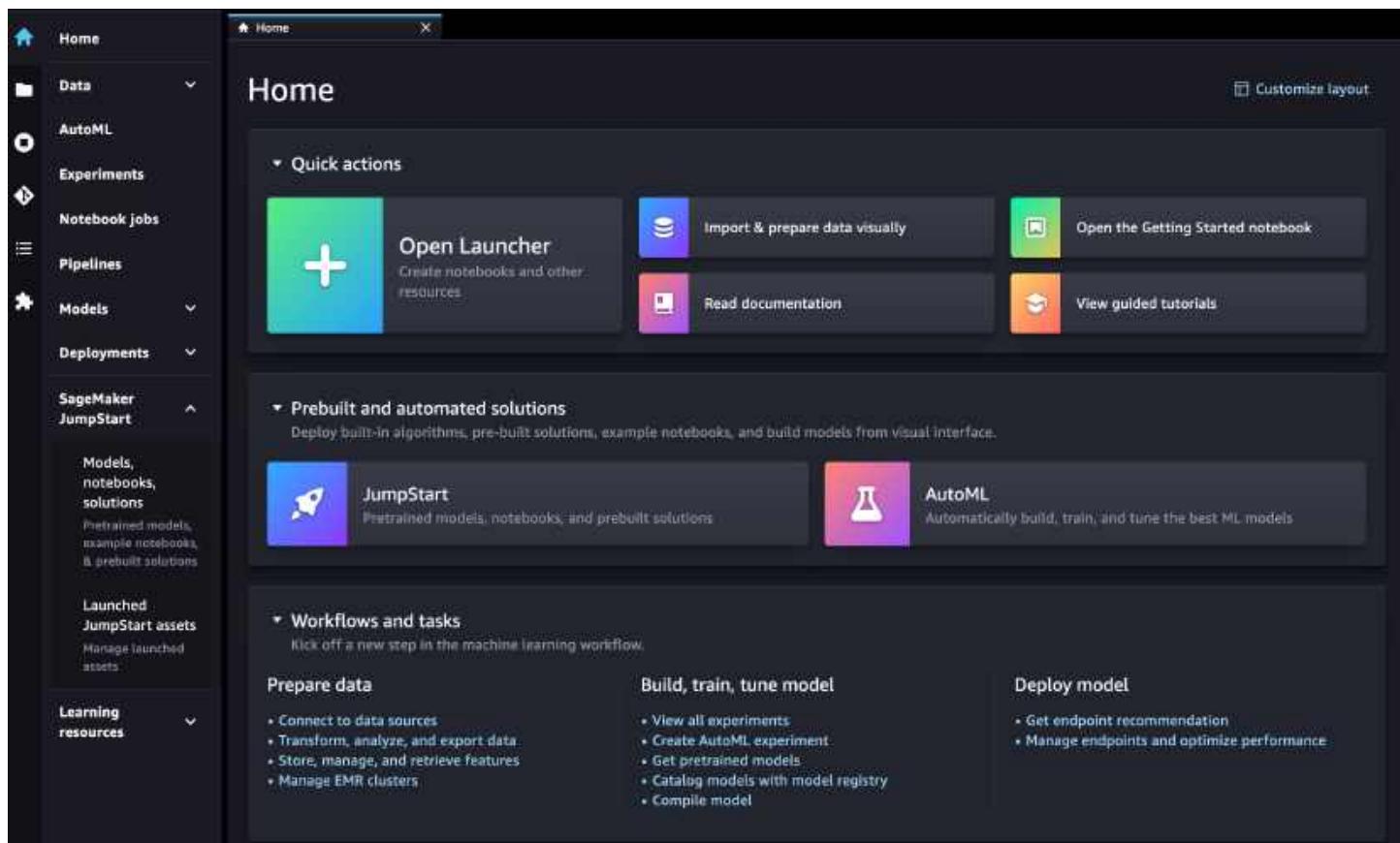
- From the **Home** page you can either:
  - Choose **JumpStart** in the **Prebuilt and automated solutions** pane. This opens the **SageMaker JumpStart** landing page.
  - Choose a model directly in the **SageMaker JumpStart** landing page, or choose the **Explore All** option to see available solutions or models of a specific type.
- From the **Home** menu in the left panel you can either:
  - Navigate to the **SageMaker JumpStart** node, then choose **Models, notebooks, solutions**. This opens the **SageMaker JumpStart** landing page.
  - Navigate to the **JumpStart** node, then choose **Launched JumpStart assets**.

The **Launched JumpStart assets** page lists your currently launched solutions, deployed model endpoints, and training jobs created with JumpStart. You can access the JumpStart landing page from this tab by clicking on the **Browse JumpStart** button at the top right of the tab.

The JumpStart landing page lists available end-to-end machine learning solutions, pretrained models, and example notebooks. From any individual solution or model page, you can choose the **Browse JumpStart** button



at the top right of the tab to return to the **SageMaker JumpStart** page.

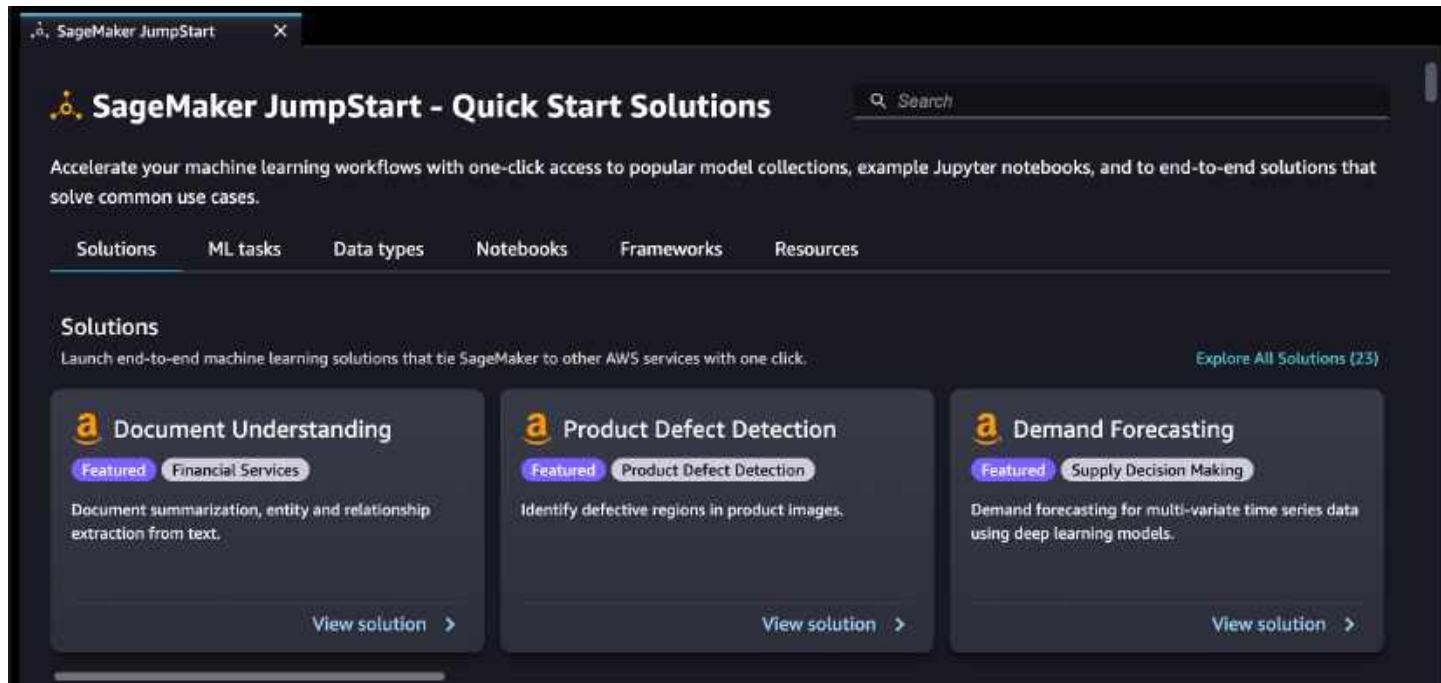


## **⚠ Important**

Before downloading or using third-party content: You are responsible for reviewing and complying with any applicable license terms and making sure that they are acceptable for your use case.

## Use JumpStart in Studio Classic

From the **SageMaker JumpStart** landing page, you can browse for solutions, models, notebooks, and other resources.



You can find JumpStart resources by using the search bar, or by browsing each category. Use the tabs to filter the available solutions by categories:

- **Solutions** – In one step, launch comprehensive machine learning solutions that tie SageMaker AI to other AWS services. Select **Explore All Solutions** to view all available solutions.
- **Resources** – Use example notebooks, blogs, and video tutorials to learn and head start your problem types.
  - **Blogs** – Read details and solutions from machine learning experts.
  - **Video tutorials** – Watch video tutorials for SageMaker AI features and machine learning use cases from machine learning experts.
  - **Example notebooks** – Run example notebooks that use SageMaker AI features like Spot Instance training and experiments over a large variety of model types and use cases.
- **Data types** – Find a model by data type (e.g., Vision, Text, Tabular, Audio, Text Generation). Select **Explore All Models** to view all available models.
- **ML tasks** – Find a model by problem type (e.g., Image Classification, Image Embedding, Object Detection, Text Generation). Select **Explore All Models** to view all available models.
- **Notebooks** – Find example notebooks that use SageMaker AI features across multiple model types and use cases. Select **Explore All Notebooks** to view all available example notebooks.
- **Frameworks** – Find a model by framework (e.g., PyTorch, TensorFlow, Hugging Face).

## Manage JumpStart in Studio Classic

From the **Home** menu in the left panel, navigate to **SageMaker JumpStart**, then choose **Launched JumpStart assets** to list your currently launched solutions, deployed model endpoints, and training jobs created with JumpStart.

### Topics

- [Amazon SageMaker JumpStart Foundation Models](#)
- [Private curated hubs for foundation model access control in JumpStart](#)
- [Amazon SageMaker JumpStart in Studio Classic](#)

## Amazon SageMaker JumpStart Foundation Models

Amazon SageMaker JumpStart offers state-of-the-art foundation models for use cases such as content writing, code generation, question answering, copywriting, summarization, classification, information retrieval, and more. Use JumpStart foundation models to build your own generative AI solutions and integrate custom solutions with additional SageMaker AI features. For more information, see [Getting started with Amazon SageMaker JumpStart](#).

A foundation model is a large pre-trained model that is adaptable to many downstream tasks and often serves as the starting point for developing more specialized models. Examples of foundation models include LLaMa-3-70b, BLOOM 176B, FLAN-T5 XL, or GPT-J 6B, which are pre-trained on massive amounts of text data and can be fine-tuned for specific language tasks.

Amazon SageMaker JumpStart onboards and maintains publicly available foundation models for you to access, customize, and integrate into your machine learning lifecycles. For more information, see [Publicly available foundation models](#). Amazon SageMaker JumpStart also includes proprietary foundation models from third-party providers. For more information, see [Proprietary foundation models](#).

To get started exploring and experimenting with available models, see [JumpStart foundation model usage](#). All foundation models are available to use programmatically with the SageMaker Python SDK. For more information, see [Use foundation models with the SageMaker Python SDK](#).

For more information on considerations to make when choosing a model, see [Model sources and license agreements](#).

For specifics about customization and fine-tuning foundation models, see [Foundation model customization](#).

For more general information on foundation models, see the paper [On the Opportunities and Risks of Foundation Models](#).

## Topics

- [Available foundation models](#)
- [JumpStart foundation model usage](#)
- [Model sources and license agreements](#)
- [Foundation model customization](#)
- [Evaluate a text generation foundation model in Studio](#)
- [Example notebooks](#)

## Available foundation models

Amazon SageMaker JumpStart offers state-of-the-art, built-in publicly available and proprietary foundation models to customize and integrate into your generative AI workflows.

### Publicly available foundation models

Amazon SageMaker JumpStart onboards and maintains open source foundation models from third-party sources. To get started with one of these publicly available models, see [JumpStart foundation model usage](#) or explore one of the available [Example notebooks](#). In a given example notebook for a publicly available model, try switching out the model ID to experiment with different models within the same model family.

For more information on model IDs and resources on deploying publicly available JumpStart foundation models with the SageMaker Python SDK, see [Use foundation models with the SageMaker Python SDK](#).

By definition, foundation models are adaptable to many downstream tasks. Foundation models are trained on massive amounts of general domain data and the same model can be implemented or customized for multiple use cases. When choosing your foundation model, start with defining a specific task, such as text generation or image generation.

### Publicly available time series forecasting models

Time series forecasting models are designed to analyze and make predictions on sequential data over time. These models can be applied to various domains such as finance, weather forecasting,

or energy demand forecasting. The Chronos models are tailored for time series forecasting tasks, enabling accurate predictions based on historical data patterns.

Model Name	Model ID	Model Source	Fine-tunable
Chronos T5 Small	autogluon-forecasting-chronos-t5-small	Amazon	No
Chronos T5 Base	autogluon-forecasting-chronos-t5-base	Amazon	No
Chronos T5 Large	autogluon-forecasting-chronos-t5-large	Amazon	No

### Publicly available text generation models

Text generation foundation models can be used for a variety of downstream tasks, including text summarization, text classification, question answering, long-form content generation, short-form copywriting, information extraction, and more.

### Publicly available text generation model table

Model Name	Model ID	Model Source	Fine-tunable
Alexa TM 20B	pytorch-textgeneration1-alexa20b	Amazon	No
Bloom 1b1	huggingface-textgeneration-bloom-1b1	Hugging Face	No
Bloom 1b7	huggingface-textgeneration-bloom-1b7	Hugging Face	No
Bloom 3B	huggingface-textgeneration1-bloom-3b	Hugging Face	Yes

Model Name	Model ID	Model Source	Fine-tunable
Bloom 560m	huggingface-textgeneration-bloom-560m	Hugging Face	No
Bloom 7B1	huggingface-textgeneration1-bloom-7b1	Hugging Face	Yes
Bloomz 1b1	huggingface-textgeneration-bloomz-1b1	Hugging Face	No
Bloomz 1b7	huggingface-textgeneration-bloomz-1b7	Hugging Face	No
BloomZ 3B FP16	huggingface-textgeneration1-bloom-3b-fp16	Hugging Face	Yes
Bloomz 560m	huggingface-textgeneration-bloomz-560m	Hugging Face	No
BloomZ 7B1 FP16	huggingface-textgeneration1-bloomz-7b1-fp16	Hugging Face	Yes
Code Llama 13B	meta-textgeneration-llama-codellama-13b	Meta	Yes
Code Llama 13B Instruct	meta-textgeneration-llama-codellama-13b-instruct	Meta	No
Code Llama 13B Python	meta-textgeneration-llama-codellama-13b-python	Meta	Yes
Code Llama 34B	meta-textgeneration-llama-codellama-34b	Meta	Yes
Code Llama 34B Instruct	meta-textgeneration-llama-codellama-34b-instruct	Meta	No

Model Name	Model ID	Model Source	Fine-tunable
Code Llama 34B Python	meta-textgeneration-llama-codellama-34b-python	Meta	Yes
Code Llama 70B	meta-textgeneration-llama-codellama-70b	Meta	Yes
Code Llama 70B Instruct	meta-textgeneration-llama-codellama-70b-instruct	Meta	No
Code Llama 70B Python	meta-textgeneration-llama-codellama-70b-python	Meta	Yes
Code Llama 7B	meta-textgeneration-llama-codellama-7b	Meta	Yes
Code Llama 7B Instruct	meta-textgeneration-llama-codellama-7b-instruct	Meta	No
Code Llama 7B Python	meta-textgeneration-llama-codellama-7b-python	Meta	Yes
CyberAgent tLM2-7B-Chat (CALM2-7B-Chat)	huggingface-llm-calm2-7b-chat-bf16	Hugging Face	Yes
DistilGPT2	huggingface-textgeneration-distilgpt2	Hugging Face	No
Dolly V2 12b BF16	huggingface-textgeneration-dolly-v2-12b-bf16	Hugging Face	No
Dolly V2 3b BF16	huggingface-textgeneration-dolly-v2-3b-bf16	Hugging Face	No
Dolly V2 7b BF16	huggingface-textgeneration-dolly-v2-7b-bf16	Hugging Face	No

Model Name	Model ID	Model Source	Fine-tunable
Dolphin 2.2.1 Mistral 7B	huggingface-llm-dolphin-2-2-1-mistral-7b	Hugging Face	No
Dolphin 2.5 Mixtral 8 7B	huggingface-llm-dolphin-2-5-mixtral-8x7b	Hugging Face	No
Dolphin 2.7 Mixtral 8 7B	huggingface-llm-dolphin-2-7-mixtral-8x7b	Hugging Face	No
EleutherAI GPT Neo 2.7B	huggingface-llm-eleutherai-gpt-neo-1-3b	Hugging Face	No
EleutherAI GPT Neo 2.7B	huggingface-llm-eleutherai-gpt-neo-2-7b	Hugging Face	No
Falcon 180B BF16	huggingface-llm-falcon-180b-bf16	Hugging Face	No
Falcon 180B Chat BF16	huggingface-llm-falcon-180b-chat-bf16	Hugging Face	No
Falcon 40B BF16	huggingface-llm-falcon-40b-bf16	Hugging Face	Yes
Falcon 40B Instruct BF16	huggingface-llm-falcon-40b-instruct-bf16	Hugging Face	Yes
Falcon 7B BF16	huggingface-llm-falcon-7b-bf16	Hugging Face	Yes
Falcon 7B Instruct BF16	huggingface-llm-falcon-7b-instruct-bf16	Hugging Face	Yes
Falcon Lite	huggingface-llm-amazon-falconlite	Hugging Face	No

Model Name	Model ID	Model Source	Fine-tunable
Falcon Lite 2	huggingface-llm-amazon-falconlite2	Hugging Face	No
Falcon RW 1B	huggingface-llm-tiiuae-falcon-rw-1b	Hugging Face	No
Flan-T5 Base	huggingface-text2text-flan-t5-base	Hugging Face	Yes
Flan-T5 Base Model Fine-tuned on the Samsum Dataset	huggingface-text2text-flan-t5-base-samsum	Hugging Face	No
Flan-T5 Large	huggingface-text2text-flan-t5-large	Hugging Face	Yes
Flan-T5 Small	huggingface-text2text-flan-t5-small	Hugging Face	Yes
Flan-T5 XL	huggingface-text2text-flan-t5-xl	Hugging Face	Yes
Flan-T5 XXL	huggingface-text2text-flan-t5-xxl	Hugging Face	Yes
Flan-UL2 BF16	huggingface-text2text-flan-ul2-bf16	Hugging Face	No
Gemma 2B	huggingface-llm-gemma-2b	Hugging Face	Yes
Gemma 2B Instruct	huggingface-llm-gemma-2b-instruct	Hugging Face	Yes
Gemma 7B	huggingface-llm-gemma-7b	Hugging Face	Yes

Model Name	Model ID	Model Source	Fine-tunable
Gemma 7B Instruct	huggingface-llm-gemma-7b-instruct	Hugging Face	Yes
GPT 2	huggingface-textgeneration-gpt2	Hugging Face	No
GPT NeoX 20B FP16	huggingface-textgeneration2-gpt-neox-20b-fp16	Hugging Face	No
GPT NeoXT Chat Base 20B FP16	huggingface-textgeneration2-gpt-neoxt-chat-base-20b-fp16	Hugging Face	No
GPT-2 XL	huggingface-textgeneration1-gpt-2-xl	Hugging Face	Yes
GPT-J 6B	huggingface-textgeneration1-gpt-j-6b	Hugging Face	Yes
GPT-Neo 1.3B	huggingface-textgeneration1-gpt-neo-1-3b	Hugging Face	Yes
GPT-Neo 125M	huggingface-textgeneration1-gpt-neo-125m	Hugging Face	Yes
GPT-NEO 2.7B	huggingface-textgeneration1-gpt-neo-2-7b	Hugging Face	Yes
Japanese StableLM Instruct Alpha 7B v2	model-textgenerationjp-japanese-stablelm-instruct-alpha-7b-v2	Hugging Face	No
LightGPT Instruct 6B	huggingface-textgeneration1-lightgpt	Hugging Face	Yes
Lite Llama 460M 1T	huggingface-llm-ahxt-litella-ama-460m-1t	Hugging Face	No

Model Name	Model ID	Model Source	Fine-tunable
Llama 2 13B	meta-textgeneration-llama-2-13b	Meta	Yes
Llama 2 13B Chat	meta-textgeneration-llama-2-13b-f	Meta	Yes
Llama 2 13B Chat Neuron	meta-textgenerationneuron-1lama-2-13b-f	Meta	No
Llama 2 13B Neuron	meta-textgenerationneuron-1lama-2-13b	Meta	Yes
Llama 2 70B	meta-textgeneration-llama-2-70b	Meta	Yes
Llama 2 70B Chat	meta-textgeneration-llama-2-70b-f	Meta	Yes
Llama 2 70B Chat Neuron	meta-textgenerationneuron-1lama-2-70b-f	Meta	No
Llama 2 70B Neuron	meta-textgenerationneuron-1lama-2-70b	Meta	No
Llama 2 7B	meta-textgeneration-llama-2-7b	Meta	Yes
Llama 2 7B Chat	meta-textgeneration-llama-2-7b-f	Meta	Yes
Llama 2 7B Chat Neuron	meta-textgenerationneuron-1lama-2-7b-f	Meta	No
Llama 2 7B Neuron	meta-textgenerationneuron-1lama-2-7b	Meta	Yes

Model Name	Model ID	Model Source	Fine-tunable
Llama 3 8B	meta-textgeneration-llama-3-8b	Meta	Yes
Llama 3 8B Instruct	meta-textgeneration-llama-3-8b-instruct	Meta	Yes
Llama 3 70B	meta-textgeneration-llama-3-70b	Meta	Yes
Llama 3 70B Instruct	meta-textgeneration-llama-3-70b-instruct	Meta	Yes
Llama Guard 7B	meta-textgeneration-llama-guard-7b	Meta	No
Mistral 7B	huggingface-llm-mistral-7b	Hugging Face	Yes
Mistral 7B Instruct	huggingface-llm-mistral-7b-instruct	Hugging Face	No
Mistral 7B OpenOrca AWQ	huggingface-llm-thebloke-mistral-7b-openorca-awq	Hugging Face	No
Mistral 7B SFT Alpha	huggingface-llm-huggingface-h4-mistral-7b-sft-alpha	Hugging Face	No
Mistral 7B SFT Beta	huggingface-llm-huggingface-h4-mistral-7b-sft-beta	Hugging Face	No
Mistral Lite	huggingface-llm-amazon-mistral-lite	Hugging Face	No
Mistral Trix V1	huggingface-llm-cultrix-mistraltrix-v1	Hugging Face	No

Model Name	Model ID	Model Source	Fine-tunable
Mixtral 8x7B	huggingface-llm-mixtral-8x7b	Hugging Face	Yes
Mixtral 8x7B Instruct	huggingface-llm-mixtral-8x7b-instruct	Hugging Face	Yes
MPT 7B BF16	huggingface-textgeneration1-mpt-7b-bf16	Hugging Face	No
MPT 7B Instruct BF16	huggingface-textgeneration1-mpt-7b-instruct-bf16	Hugging Face	No
MPT 7B StoryWriter-65k+ BF16	huggingface-textgeneration1-mpt-7b-storywriter-bf16	Hugging Face	No
Multilingual GPT	huggingface-llm-ai-forever- mgpt	Hugging Face	No
Nous Hermes 2 SOLAR 10.7B	huggingface-llm-nousresearch-nous-hermes-2-solar-10-7b	Hugging Face	No
Nous Hermes Llama 2 13B	huggingface-llm-nousresearch-nous-hermes-llama2-13b	Hugging Face	No
Nous Hermes Llama 2 7B	huggingface-llm-nousresearch-nous-hermes-llama2-7b	Hugging Face	No
Open Hermes 2 Mistral 7B	huggingface-llm-teknium-openhermes-2-mistral-7b	Hugging Face	No
Open LLaMA	huggingface-textgeneration-open-llama	Hugging Face	No
Open Llama 7B V2	huggingface-llm-openlm-research-open-llama-7b-v2	Hugging Face	No

Model Name	Model ID	Model Source	Fine-tunable
Platypus 2 7B	huggingface-llm-garage-baind-platypus2-7b	Hugging Face	No
Pythia 160m Deduped	huggingface-llm-eleutherai-pythia-160m-deduped	Hugging Face	No
Pythia 7m Deduped	huggingface-llm-eleutherai-pythia-70m-deduped	Hugging Face	No
Quality Controlled Paraphrase Generation	huggingface-text2text-qcpq-sentences	Hugging Face	No
RedPajama INCITE Base 3B V1	huggingface-textgeneration1-redpajama-incite-base-3B-v1-fp16	Hugging Face	Yes
RedPajama INCITE Base 7B V1	huggingface-textgeneration1-redpajama-incite-base-7B-v1-fp16	Hugging Face	Yes
RedPajama INCITE Chat 3B V1	huggingface-textgeneration1-redpajama-incite-chat-3B-v1-fp16	Hugging Face	Yes
RedPajama INCITE Chat 7B V1	huggingface-textgeneration1-redpajama-incite-chat-7B-v1-fp16	Hugging Face	Yes
RedPajama INCITE Instruct 3B V1	huggingface-textgeneration1-redpajama-incite-instruct-3B-v1-fp16	Hugging Face	Yes
RedPajama INCITE Instruct 7B V1	huggingface-textgeneration1-redpajama-incite-instruct-7B-v1-fp16	Hugging Face	Yes

Model Name	Model ID	Model Source	Fine-tunable
Rinna Bilingual GPT NeoX 4B Instruction PPO	huggingface-llm-bilingual-rinna-4b-instruction-ppo-bf16	Hugging Face	No
Rinna Japanese GPT NeoX 3.6B Instruction PPO	huggingface-llm-rinna-3-6b-instruction-ppo-bf16	Hugging Face	No
Star Chat Alpha	huggingface-llm-huggingface-h4-starchat-alpha	Hugging Face	No
Star Chat Beta	huggingface-llm-huggingface-h4-starchat-beta	Hugging Face	No
StarCoder	huggingface-llm-starcoder	Hugging Face	No
StarCoderBase	huggingface-llm-starcoderbase	Hugging Face	No
T0pp	huggingface-text2text-bigscale-t0pp	Hugging Face	No
T5 One Line Summary	huggingface-text2text-t5-one-line-summary	Hugging Face	No
Tiny Llama 1.1B	huggingface-llm-tinyllama-1-1b-intermediate-step-1431k-3	Hugging Face	No
Tiny Llama 1.1B Chat V0.6	huggingface-llm-tinyllama-tinyllama-1-1b-chat-v0-6	Hugging Face	No
Tiny Llama 1.1B Chat V1	huggingface-llm-tinyllama-tinyllama-1-1b-chat-v1-0	Hugging Face	No

Model Name	Model ID	Model Source	Fine-tunable
Writer Palmyra Small	huggingface-llm-writer-palmyra-small	Hugging Face	No
YARN Mistral 7B 128k	huggingface-llm-nousresearch-yarn-mistral-7b-128k	Hugging Face	No
Zephyr 7B Alpha	huggingface-llm-huggingface-h4-zephyr-7b-alpha	Hugging Face	No
Zephyr 7B Beta	huggingface-llm-huggingface-h4-zephyr-7b-beta	Hugging Face	No

To explore the latest text generation JumpStart foundation models, use the **Text Generation** filter on the [Getting started with Amazon SageMaker JumpStart](#) product description page. You can also explore foundation models based on tasks directly in the Amazon SageMaker Studio UI or SageMaker Studio Classic UI. Only a subset of publicly available text generation models are available for fine-tuning in JumpStart. For more information, see [Use foundation models in Amazon SageMaker Studio Classic](#).

## Publicly available image generation models

JumpStart provides a wide variety of Stable Diffusion image generation foundation models including base models from Stability AI as well as pre-trained models for specific text-to-image tasks from Hugging Face. If you need to fine-tune your text-to-image foundation model, you can use Stable Diffusion 2.1 base from Stability AI. If you want to explore models that are already trained on specific art styles, you can explore one of the many third-party models from Hugging Face directly in the Amazon SageMaker Studio UI or SageMaker Studio Classic UI.

To explore the latest image generation JumpStart foundation models, use the **Text to Image** filter on the [Getting started with Amazon SageMaker JumpStart](#) product description page. To get started with your chosen text-to-image foundation model, see [JumpStart foundation model usage](#).

## Proprietary foundation models

Amazon SageMaker JumpStart provides access to proprietary foundation models from third-party providers such as [AI21 Labs](#), [Cohere](#), and [LightOn](#).

To get started with one of these proprietary models, see [JumpStart foundation model usage](#). To use a proprietary foundation model, you must first subscribe to the model in AWS Marketplace. After subscribing to the model, locate the foundation model in Studio or SageMaker Studio Classic. For more information, see [SageMaker JumpStart pretrained models](#).

To explore the latest proprietary foundation models for a variety of use cases, see [Getting started with Amazon SageMaker JumpStart](#).

## JumpStart foundation model usage

Choose, train, or deploy foundation models through Amazon SageMaker Studio or Amazon SageMaker Studio Classic, use JumpStart foundation models programmatically with the SageMaker Python SDK, or discover JumpStart foundation models directly through the SageMaker AI console.

### Topics

- [Use foundation models in Studio](#)
- [Use foundation models in Amazon SageMaker Studio Classic](#)
- [Use foundation models with the SageMaker Python SDK](#)
- [Discover foundation models in the SageMaker AI Console](#)

### Use foundation models in Studio

Amazon SageMaker Studio allows you to fine-tune, deploy, and evaluate both publicly available and proprietary JumpStart foundation models directly through the Studio UI.

#### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

To get started, navigate to the JumpStart landing page in Amazon SageMaker Studio. You can access it from the **Home** page or the left-side panel menu. On the **JumpStart** landing page, you can explore model hubs from providers of both publicly available and proprietary models, and search for models.

Within each model hub, you can sort models by **Most likes**, **Most downloads**, **Recently updated**, or filter them by task. Choose a model to view its detail card. On the model detail card, you can choose to **Fine-tune**, **Deploy**, or **Evaluate** the model, depending on the available option. Note that not all models are available for fine-tuning or evaluation.

For more information on getting started with Amazon SageMaker Studio, see [Amazon SageMaker Studio](#).

## Topics

- [Fine-tune a model in Studio](#)
- [Deploy a model in Studio](#)
- [Evaluate a model in Studio](#)
- [Use your SageMaker JumpStart Models in Amazon Bedrock](#)

## Fine-tune a model in Studio

Fine-tuning trains a pre-trained model on a new dataset without training from scratch. This process, also known as transfer learning, can produce accurate models with smaller datasets and less training time. To fine-tune JumpStart foundation models, navigate to a model detail card in the Studio UI. For more information on how to open JumpStart in Studio, see [Open and use JumpStart in Studio](#). After navigating to the model detail card of your choice, choose **Train** in the upper right corner. Note that not all models have fine-tuning available.

### Important

Some foundation models require explicit acceptance of an end-user license agreement (EULA) before fine-tuning. For more information, see [EULA acceptance in Amazon SageMaker Studio](#).

## Model settings

When using a pre-trained JumpStart foundation model in Amazon SageMaker Studio, the **Model artifact location (Amazon S3 URI)** is populated by default. To edit the default Amazon S3 URI, choose **Enter model artifact location**. Not all models support changing the model artifact location.

## Data settings

In the **Data** field, provide an Amazon S3 URI point to your training dataset location. The default Amazon S3 URI points to an example training dataset. To edit the default Amazon S3 URI, choose **Enter training dataset** and change the URI. Be sure to review the model detail card in Amazon SageMaker Studio for information on formatting training data.

## Hyperparameters

You can customize the hyperparameters of the training job that are used to fine-tune the model. The hyperparameters available for each fine-tunable model differ depending on the model.

The following hyperparameters are common among models:

- **Epochs** – One epoch is one cycle through the entire dataset. Multiple intervals complete a batch, and multiple batches eventually complete an epoch. Multiple epochs are run until the accuracy of the model reaches an acceptable level, or when the error rate drops below an acceptable level.
- **Learning rate** – The amount that values should be changed between epochs. As the model is refined, its internal weights are being nudged and error rates are checked to see if the model improves. A typical learning rate is 0.1 or 0.01, where 0.01 is a much smaller adjustment and could cause the training to take a long time to converge, whereas 0.1 is much larger and can cause the training to overshoot. It is one of the primary hyperparameters that you might adjust for training your model. Note that for text models, a much smaller learning rate (5e-5 for BERT) can result in a more accurate model.
- **Batch size** – The number of records from the dataset that is to be selected for each interval to send to the GPUs for training.

Review the tool tip prompts and additional information in the model detail card in the Studio UI to learn more about hyperparameters specific to the model of your choice.

For more information on available hyperparameters, see [Commonly supported fine-tuning hyperparameters](#).

## Deployment

Specify the training instance type and output artifact location for your training job. You can only choose from instances that are compatible with the model of your choice within the fine-tuning the Studio UI. The default output artifact location is the SageMaker AI default bucket. To change the output artifact location, choose **Enter output artifact location** and change the Amazon S3 URI.

## Security

Specify the security settings to use for your training job, including the IAM role that SageMaker AI uses to train your model, whether your training job should connect to a virtual private cloud (VPC), and any encryption keys to secure your data.

## Additional information

In the **Additional Information** field you can edit the training job name. You can also add and remove tags in the form of key-value pairs to help organize and categorize your fine-tuning training jobs.

After providing information for your fine-tuning configuration, choose **Submit**. If the pre-trained foundation model that you chose to fine-tune requires explicit agreement of an end-user license agreement (EULA) before training, the EULA is provided in a pop-up window. To accept the terms of the EULA, choose **Accept**. You are responsible for reviewing and complying with any applicable license terms and making sure they are acceptable for your use case before downloading or using a model.

## Deploy a model in Studio

To deploy JumpStart foundation models, navigate to a model detail card in the Studio UI. For more information on how to open JumpStart in Studio, see [Open and use JumpStart in Studio](#). After navigating to the model detail page of your choice, choose **Deploy** in the upper right corner of the Studio UI. Then, follow the steps in [Deploy models with SageMaker Studio](#).

### Important

Some foundation models require explicit acceptance of an end-user license agreement (EULA) before deployment. For more information, see [EULA acceptance in Amazon SageMaker Studio](#).

## Evaluate a model in Studio

Amazon SageMaker JumpStart has integrations with SageMaker Clarify foundation model evaluations (FME) in Studio. If a JumpStart model has built-in evaluation capabilities available, you can choose **Evaluate** in the upper right corner of the model detail page in the JumpStart Studio UI. For more information, see [Evaluate a foundation model](#).

## Use your SageMaker JumpStart Models in Amazon Bedrock

You can register the models that you've deployed from Amazon SageMaker JumpStart to Amazon Bedrock. With Amazon Bedrock, you can host your model behind multiple endpoints. You can also use Amazon Bedrock features, such as Agents and Knowledge Bases. For more information about using Amazon Bedrock's models, see <https://docs.aws.amazon.com/bedrock/latest/userguide/amazon-bedrock-marketplace.html>.

### Important

To migrate your models to Amazon Bedrock, we recommend attaching [AmazonBedrockFullAccess](#) policy to your IAM role. If you can't attach the managed policy, make sure your IAM role has the following permissions:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "BedrockAll",  
            "Effect": "Allow",  
            "Action": [  
                "bedrock:*"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "DescribeKey",  
            "Effect": "Allow",  
            "Action": [  
                "kms:DescribeKey"  
            ],  
            "Resource": "arn:aws:kms:*::::  
        },  
        {  
            "Sid": "APIsWithAllResourceAccess",  
            "Effect": "Allow",  
            "Action": [  
                "iam>ListRoles",  
                "ec2:DescribeVpcs",  
                "ec2:DescribeSubnets",  
                "ec2:DescribeSecurityGroups"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

```
    "Resource": "*"
},
{
  "Sid": "MarketplaceModelEndpointMutatingAPIs",
  "Effect": "Allow",
  "Action": [
    "sagemaker:CreateEndpoint",
    "sagemaker:CreateEndpointConfig",
    "sagemaker:CreateModel",
    "sagemaker:CreateInferenceComponent",
    "sagemaker:DeleteInferenceComponent",
    "sagemaker:DeleteEndpoint",
    "sagemaker:UpdateEndpoint"
  ],
  "Resource": [
    "arn:aws:sagemaker:*:*:endpoint/*",
    "arn:aws:sagemaker:*:*:endpoint-config/*",
    "arn:aws:sagemaker:*:*:model/*"
  ],
  "Condition": {
    "StringEquals": {
      "aws:CalledViaLast": "bedrock.amazonaws.com"
    }
  }
},
{
  "Sid": "BedrockEndpointTaggingOperations",
  "Effect": "Allow",
  "Action": [
    "sagemaker:AddTags",
    "sagemaker:DeleteTags"
  ],
  "Resource": [
    "arn:aws:sagemaker:*:*:endpoint/*",
    "arn:aws:sagemaker:*:*:endpoint-config/*",
    "arn:aws:sagemaker:*:*:model/*"
  ]
},
{
  "Sid": "MarketplaceModelEndpointNonMutatingAPIs",
  "Effect": "Allow",
  "Action": [
    "sagemaker:DescribeEndpoint",
    "sagemaker:DescribeEndpointConfig",
```

```
"sagemaker:DescribeModel",
"sagemaker:DescribeInferenceComponent",
"sagemaker>ListEndpoints",
"sagemaker>ListTags"
],
"Resource": [
"arn:aws:sagemaker:*::endpoint/*",
"arn:aws:sagemaker:*::endpoint-config/*",
"arn:aws:sagemaker:*::model/*"
],
"Condition": {
"StringEquals": {
"aws:CalledViaLast": "bedrock.amazonaws.com"
}
}
},
{
"Sid": "BedrockEndpointInvokingOperations",
"Effect": "Allow",
"Action": [
"sagemaker:InvokeEndpoint",
"sagemaker:InvokeEndpointWithResponseStream"
],
"Resource": [
"arn:aws:sagemaker:*::endpoint/*"
],
"Condition": {
"StringEquals": {
"aws:CalledViaLast": "bedrock.amazonaws.com"
}
}
},
{
"Sid": "DiscoveringMarketplaceModel",
"Effect": "Allow",
"Action": [
"sagemaker:DescribeHubContent"
],
"Resource": [
"arn:aws:sagemaker*:aws:hub-content/SageMakerPublicHub/Model/*",
"arn:aws:sagemaker*:aws:hub/SageMakerPublicHub"
]
},
{
```

```
"Sid": "AllowMarketplaceModelsListing",
"Effect": "Allow",
>Action": [
    "sagemaker>ListHubContents"
],
"Resource": "arn:aws:sagemaker:*:aws:hub/SageMakerPublicHub"
},
{
"Sid": "RetrieveSubscribedMarketplaceLicenses",
"Effect": "Allow",
>Action": [
    "license-manager>ListReceivedLicenses"
],
"Resource": [
    "*"
],
{
"Sid": "PassRoleToSageMaker",
"Effect": "Allow",
>Action": [
    "iam:PassRole"
],
"Resource": [
    "arn:aws:iam::*:role/*Sagemaker*ForBedrock*"
],
"Condition": {
    "StringEquals": {
        "iam:PassedToService": [
            "sagemaker.amazonaws.com",
            "bedrock.amazonaws.com"
        ]
    }
}
},
{
"Sid": "PassRoleToBedrock",
"Effect": "Allow",
>Action": [
    "iam:PassRole"
],
"Resource": "arn:aws:iam::*:role/*AmazonBedrock*",
"Condition": {
    "StringEquals": {
```

```
        "iam:PassedToService": [
            "bedrock.amazonaws.com"
        ]
    }
}
]
}
```

## ⚠ Important

The Amazon Bedrock Full Access policy only provides permissions to the Amazon Bedrock API. To use Amazon Bedrock in the AWS Management Console, your IAM role must also have the following permissions:

```
{
    "Sid": "AllowConsoleS3AccessForBedrockMarketplace",
    "Effect": "Allow",
    "Action": [
        "s3:GetObject",
        "s3:GetBucketCORS",
        "s3>ListBucket",
        "s3>ListBucketVersions",
        "s3:GetBucketLocation"
    ],
    "Resource": "*"
}
```

If you're writing your own policy, you must include the policy statement that allows the Amazon Bedrock Marketplace action for the resource. For example, the following policy allows Amazon Bedrock to use the InvokeModel operation for a model that you've deployed to an endpoint.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "lambda:InvokeFunction",
            "FunctionName": "arn:aws:lambda:us-east-1:123456789012:function:my-model"
        }
    ]
}
```

```
        "Sid": "BedrockAll",
        "Effect": "Allow",
        "Action": [
            "bedrock:InvokeModel"
        ],
        "Resource": [
            "arn:aws:bedrock:AWS Region:111122223333:marketplace/example-model-endpoint/all-access"
        ]
    },
    {
        "Sid": "VisualEditor1",
        "Effect": "Allow",
        "Action": ["sagemaker:InvokeEndpoint"],
        "Resource": "arn:aws:sagemaker:AWS Region:111122223333:endpoint/*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/project": "example-project-id",
                "aws:CalledViaLast": "bedrock.amazonaws.com"
            }
        }
    }
]
```

}

After you've deployed a model, you might be able to use it in Amazon Bedrock. To see if you can use it in Amazon Bedrock, navigate to the model detail card in the Studio UI. If the model card says that it's **Bedrock Ready**, you can register the model with Amazon Bedrock.

### **Important**

By default Amazon SageMaker JumpStart disables network access for the models that you deploy. If you've enabled network access, you won't be able to use the model with Amazon Bedrock. If you want to use the model with Amazon Bedrock, you must redeploy it with network access disabled.

To use it with Amazon Bedrock, navigate to the **Endpoint details** page and choose **Use with Bedrock** in the upper right corner of the Studio UI. After you see the pop-up, choose **Register to Bedrock**.

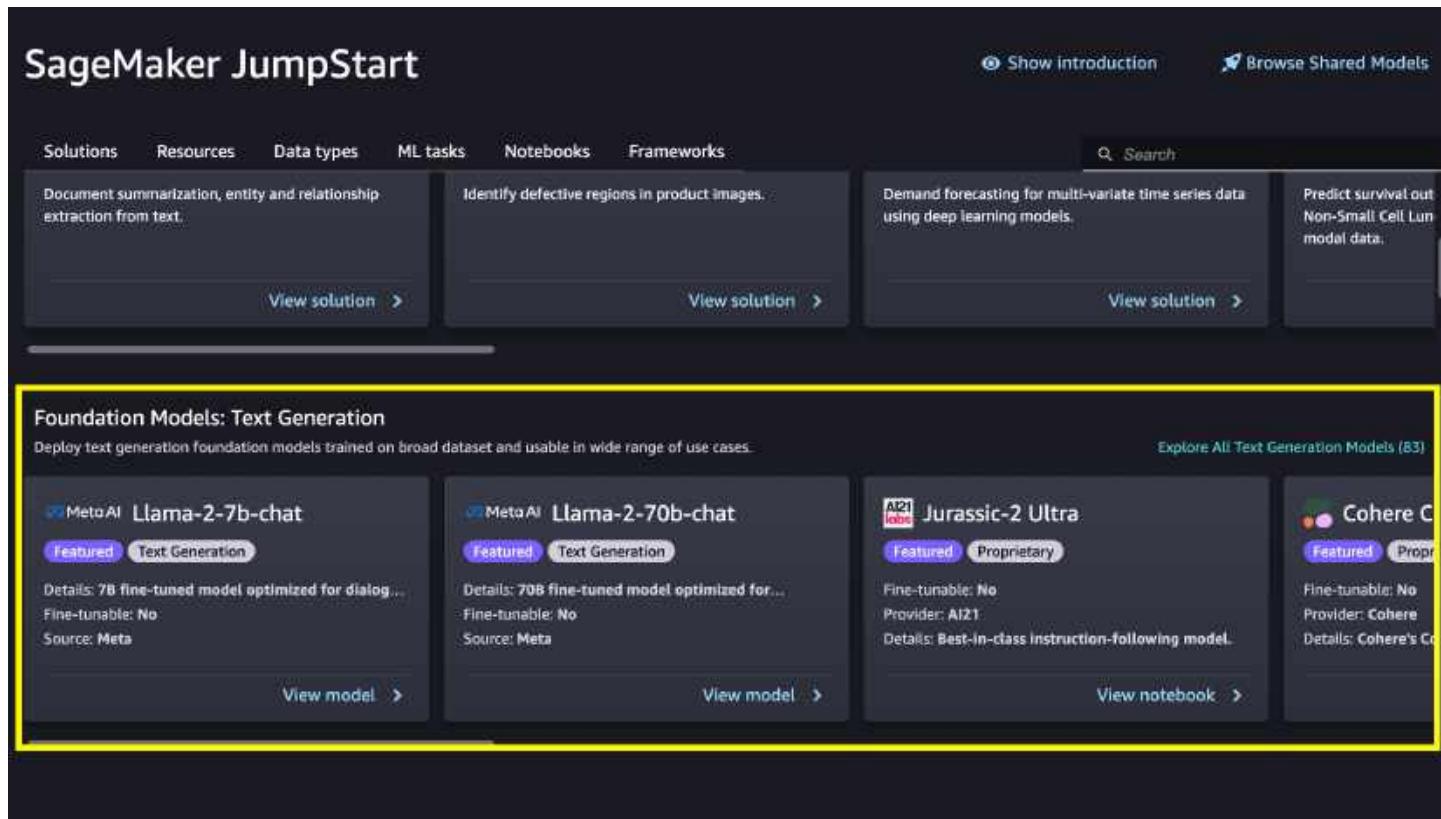
## Use foundation models in Amazon SageMaker Studio Classic

You can fine-tune and deploy both publicly available and proprietary JumpStart foundation models through the Studio Classic UI.

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

To get started with Studio Classic, see [Launch Amazon SageMaker Studio Classic](#).



The screenshot shows the SageMaker JumpStart interface. At the top, there are navigation links for Solutions, Resources, Data types, ML tasks, Notebooks, and Frameworks. A search bar is also present. Below this, there are four cards representing different solutions:

- Document summarization, entity and relationship extraction from text. Status: Available. View solution >
- Identify defective regions in product images. Status: Available. View solution >
- Demand forecasting for multi-variate time series data using deep learning models. Status: Available. View solution >
- Predict survival outcome for Non-Small Cell Lung cancer model data. Status: Available. View solution >

Below these cards, a section titled "Foundation Models: Text Generation" is highlighted with a yellow border. It contains four cards for text generation models:

- MetaAI Llama-2-7b-chat**: Featured, Text Generation. Details: 7B fine-tuned model optimized for dialog... Fine-tunable: No. Source: Meta. View model >
- MetaAI Llama-2-70b-chat**: Featured, Text Generation. Details: 70B fine-tuned model optimized for... Fine-tunable: No. Source: Meta. View model >
- AI21 Jurassic-2 Ultra**: Proprietary. Details: Best-in-class instruction-following model. Fine-tunable: No. Provider: AI21. View notebook >
- Cohere C**: Proprietary. Details: Cohere's Co... Fine-tunable: No. Provider: Cohere. View notebook >

At the bottom left of the interface, there is a link to "Foundation models".

After opening Amazon SageMaker Studio Classic, choose **Models, notebooks, solutions** in the SageMaker JumpStart section of the navigation pane. Then, scroll down to find either the **Foundation Models: Text Generation** or **Foundation Models: Image Generation** section depending on your use case.

You can choose **View model** on a suggested foundation model card, or choose **Explore All Models** to see all available foundation models for either text generation or image generation. If you choose to see all available models, you can further filter available models by task, data type, content type, or framework. You can also search for a model name directly in the **Search** bar. If you need guidance on selecting a model, see [Available foundation models](#).

### Important

Some foundation models require explicit acceptance of an end-user license agreement (EULA). For more information, see [EULA acceptance in Amazon SageMaker Studio](#).

After you choose **View model** for the foundation model of your choice in Studio Classic, you can deploy the model. For more information, see [Deploy a Model](#).

You can also choose **Open notebook** in the **Run in notebook** section to run an example notebook for the foundation model directly in Studio Classic.

### Note

To deploy a proprietary foundation model in Studio Classic, you must first subscribe to the model in AWS Marketplace. The AWS Marketplace link is provided in the associated example notebook within Studio Classic.

If the model is fine-tunable, you can also fine-tune the model. For more information, see [Fine-Tune a Model](#). For a list of which JumpStart foundation models are fine-tunable, see [Foundation models and hyperparameters for fine-tuning](#).

## Use foundation models with the SageMaker Python SDK

All JumpStart foundation models are available for programmatic deployment using the SageMaker Python SDK.

To deploy publicly available foundation models, you can use their model ID. You can find the model IDs for all publicly available foundation models in the [Built-in Algorithms with pre-trained Model Table](#). Search for the name of a foundation model in the **Search** bar. Use the **Show entries** dropdown or the pagination controls to navigate the available models.

Proprietary models must be deployed using the model package information after subscribing to the model in AWS Marketplace.

You can find the list of JumpStart available models in [the section called “Available models”](#).

### Important

Some foundation models require explicit acceptance of an end-user license agreement (EULA). For more information, see [EULA acceptance with the SageMaker Python SDK](#).

The following sections show how to fine-tune publicly available foundation models using the `JumpStartEstimator` class, deploy publicly available foundation models using the `JumpStartModel` class, and deploy proprietary foundation models using the `ModelPackage` class.

## Topics

- [Fine-tune publicly available foundation models with the `JumpStartEstimator` class](#)
- [Deploy publicly available foundation models with the `JumpStartModel` class](#)
- [Deploy proprietary foundation models with the `ModelPackage` class](#)

### Fine-tune publicly available foundation models with the `JumpStartEstimator` class

You can fine-tune a built-in algorithm or pre-trained model in just a few lines of code using the SageMaker Python SDK.

1. First, find the model ID for the model of your choice in the [Built-in Algorithms with pre-trained Model Table](#).
2. Using the model ID, define your training job as a `JumpStart` estimator.

```
from sagemaker.jumpstart.estimator import JumpStartEstimator

model_id = "huggingface-textgeneration1-gpt-j-6b"
estimator = JumpStartEstimator(model_id=model_id)
```

3. Run `estimator.fit()` on your model, pointing to the training data to use for fine-tuning.

```
estimator.fit(  
    {"train": "training_dataset_s3_path", "validation": "validation_dataset_s3_path"}  
)
```

4. Then, use the `deploy` method to automatically deploy your model for inference. In this example, we use the GPT-J 6B model from Hugging Face.

```
predictor = estimator.deploy()
```

5. You can then run inference with the deployed model using the `predict` method.

```
question = "What is Southern California often abbreviated as?"  
response = predictor.predict(question)  
print(response)
```

### Note

This example uses the foundation model GPT-J 6B, which is suitable for a wide range of text generation use cases including question answering, named entity recognition, summarization, and more. For more information about model use cases, see [Available foundation models](#).

You can optionally specify model versions or instance types when creating your `JumpStartEstimator`. For more information about the `JumpStartEstimator` class and its parameters, see [JumpStartEstimator](#).

### Check default instance types

You can optionally include specific model versions or instance types when fine-tuning a pre-trained model using the `JumpStartEstimator` class. All `JumpStart` models have a default instance type. Retrieve the default training instance type using the following code:

```
from sagemaker import instance_types  
  
instance_type = instance_types.retrieve_default(  
    model_id=model_id,
```

```
model_version=model_version,  
scope="training")  
print(instance_type)
```

You can see all supported instance types for a given JumpStart model with the `instance_types.retrieve()` method.

## Check default hyperparameters

To check the default hyperparameters used for training, you can use the `retrieve_default()` method from the `hyperparameters` class.

```
from sagemaker import hyperparameters  
  
my_hyperparameters = hyperparameters.retrieve_default(model_id=model_id,  
model_version=model_version)  
print(my_hyperparameters)  
  
# Optionally override default hyperparameters for fine-tuning  
my_hyperparameters["epoch"] = "3"  
my_hyperparameters["per_device_train_batch_size"] = "4"  
  
# Optionally validate hyperparameters for the model  
hyperparameters.validate(model_id=model_id, model_version=model_version,  
hyperparameters=my_hyperparameters)
```

For more information on available hyperparameters, see [Commonly supported fine-tuning hyperparameters](#).

## Check default metric definitions

You can also check the default metric definitions:

```
print(metric_definitions.retrieve_default(model_id=model_id,  
model_version=model_version))
```

## Deploy publicly available foundation models with the `JumpStartModel` class

You can deploy a built-in algorithm or pre-trained model to a SageMaker AI endpoint in just a few lines of code using the SageMaker Python SDK.

1. First, find the model ID for the model of your choice in the [Built-in Algorithms with pre-trained Model Table](#).
2. Using the model ID, define your model as a JumpStart model.

```
from sagemaker.jumpstart.model import JumpStartModel  
  
model_id = "huggingface-text2text-flan-t5-xl"  
my_model = JumpStartModel(model_id=model_id)
```

3. Use the `deploy` method to automatically deploy your model for inference. In this example, we use the FLAN-T5 XL model from Hugging Face.

```
predictor = my_model.deploy()
```

4. You can then run inference with the deployed model using the `predict` method.

```
question = "What is Southern California often abbreviated as?"  
response = predictor.predict(question)  
print(response)
```

### Note

This example uses the foundation model FLAN-T5 XL, which is suitable for a wide range of text generation use cases including question answering, summarization, chatbot creation, and more. For more information about model use cases, see [Available foundation models](#).

For more information about the `JumpStartModel` class and its parameters, see [JumpStartModel](#).

### Check default instance types

You can optionally include specific model versions or instance types when deploying a pre-trained model using the `JumpStartModel` class. All `JumpStart` models have a default instance type. Retrieve the default deployment instance type using the following code:

```
from sagemaker import instance_types  
  
instance_type = instance_types.retrieve_default(  
    model_id=model_id,
```

```
model_version=model_version,  
scope="inference")  
print(instance_type)
```

See all supported instance types for a given JumpStart model with the `instance_types.retrieve()` method.

## Use inference components to deploy multiple models to a shared endpoint

An inference component is a SageMaker AI hosting object that you can use to deploy one or more models to an endpoint for increased flexibility and scalability. You must change the `endpoint_type` for your JumpStart model to be inference-component-based rather than the default model-based endpoint.

```
predictor = my_model.deploy(  
    endpoint_name = 'jumpstart-model-id-123456789012',  
    endpoint_type = EndpointType.INFERENCE_COMPONENT_BASED  
)
```

For more information on creating endpoints with inference components and deploying SageMaker AI models, see [Shared resource utilization with multiple models](#).

## Check valid input and output inference formats

To check valid data input and output formats for inference, you can use the `retrieve_options()` method from the `Serializers` and `Deserializers` classes.

```
print(sagemaker.serializers.retrieve_options(model_id=model_id,  
    model_version=model_version))  
print(sagemaker.deserializers.retrieve_options(model_id=model_id,  
    model_version=model_version))
```

## Check supported content and accept types

Similarly, you can use the `retrieve_options()` method to check the supported content and accept types for a model.

```
print(sagemaker.content_types.retrieve_options(model_id=model_id,  
    model_version=model_version))
```

```
print(sagemaker.accept_types.retrieve_options(model_id=model_id,
model_version=model_version))
```

For more information about utilities, see [Utility APIs](#).

## Deploy proprietary foundation models with the ModelPackage class

Proprietary models must be deployed using the model package information after subscribing to the model in AWS Marketplace. For more information about SageMaker AI and AWS Marketplace, see [Buy and Sell Amazon SageMaker AI Algorithms and Models in AWS Marketplace](#). To find AWS Marketplace links for the latest proprietary models, see [Getting started with Amazon SageMaker JumpStart](#).

After subscribing to the model of your choice in AWS Marketplace, you can deploy the foundation model using the SageMaker Python SDK and the SDK associated with the model provider. For example, AI21 Labs, Cohere, and LightOn use the "ai21[SM]", cohene-sagemaker, and lightonsage packages, respectively.

For example, to define a JumpStart model using Jurassic-2 Jumbo Instruct from AI21 Labs, use the following code:

```
import sagemaker
import ai21

role = get_execution_role()
sagemaker_session = sagemaker.Session()
model_package_arn = "arn:aws:sagemaker:us-east-1:865070037744:model-package/j2-jumbo-
instruct-v1-1-43-4e47c49e61743066b9d95efed6882f35"

my_model = ModelPackage(
    role=role, model_package_arn=model_package_arn, sagemaker_session=sagemaker_session
)
```

For step-by-step examples, find and run the notebook associated with the proprietary foundation model of your choice in SageMaker Studio Classic. See [Use foundation models in Amazon SageMaker Studio Classic](#) for more information. For more information on the SageMaker Python SDK, see [ModelPackage](#).

## Discover foundation models in the SageMaker AI Console

You can explore JumpStart foundation models directly through the Amazon SageMaker AI Console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Find **JumpStart** on the left navigation panel and choose **Foundation models**.
3. Browse models or search for a specific model. If you need guidance for model selection, see [Available foundation models](#). Choose **View model** to view the model detail page for the foundation model of your choice.
4. If the model is a proprietary model, choose **Subscribe** in the upper right corner of the model detail page to subscribe to the model in AWS Marketplace. You should receive an email confirming your subscription to the model of your choice. For more information about SageMaker AI and AWS Marketplace, see [Buy and Sell Amazon SageMaker AI Algorithms and Models in AWS Marketplace](#). Publicly available foundation models do not require a subscription.
5. To view an example notebook in GitHub, choose **View code** in the upper right corner of the model detail page.
6. To view and run an example notebook directly in Amazon SageMaker Studio Classic, choose **Open notebook in Studio** in the upper right corner of the model detail page.

## Model sources and license agreements

Amazon SageMaker JumpStart provides access to hundreds of publicly available and proprietary foundation models from third-party sources and partners. You can explore the JumpStart foundation model selection directly in the SageMaker AI console, Studio, or Studio Classic.

### Licenses and model sources

Amazon SageMaker JumpStart provides access to both publicly available and proprietary foundation models. Foundation models are onboarded and maintained from third-party open source and proprietary providers. As such, they are released under different licenses as designated by the model source. Be sure to review the license for any foundation model that you use. You are responsible for reviewing and complying with any applicable license terms and making sure they are acceptable for your use case before downloading or using the content. Some examples of common foundation model licenses include:

- Alexa Teacher Model
- Apache 2.0
- BigScience Responsible AI License v1.0
- CreativeML Open RAIL++-M license

Similarly, for any proprietary foundation models, be sure to review and comply with any terms of use and usage guidelines from the model provider. If you have questions about license information for a specific proprietary model, reach out to model provider directly. You can find model provider contact information in the **Support** tab of each model page in AWS Marketplace.

## End-user license agreements

Some JumpStart foundation models require explicit acceptance of an end-user license agreement (EULA) before use.

### EULA acceptance in Amazon SageMaker Studio

You may be prompted to accept an end-user license agreement before fine-tuning, deploying, or evaluating a JumpStart foundation model in Studio. To get started with JumpStart foundation models in Studio, see [Use foundation models in Studio](#).

#### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

Some JumpStart foundation models require acceptance of an end-user license agreement before deployment. If this applies to the foundation model that you choose to use, Studio prompts you with a window containing the EULA content. You are responsible for reviewing and complying with any applicable license terms and making sure they are acceptable for your use case before downloading or using a model.

### EULA acceptance in Amazon SageMaker Studio Classic

You may be prompted to accept an end-user license agreement before deploying a JumpStart foundation model or opening a JumpStart foundation model notebook in Studio Classic. To get started with JumpStart foundation models in Studio Classic, see [Use foundation models in Amazon SageMaker Studio Classic](#).

## Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Some JumpStart foundation models require acceptance of an end-user license agreement before deployment. If this applies to the foundation model that you choose to use, Studio Classic prompts you with a window titled **Review the End User License Agreement (EULA) and Acceptable Use Policy (AUP) below** after you choose either **Deploy** or **Open notebook**. You are responsible for reviewing and complying with any applicable license terms and making sure they are acceptable for your use case before downloading or using a model.

### EULA acceptance with the SageMaker Python SDK

The following sections show you how to explicitly declare EULA acceptance when deploying or fine-tuning a JumpStart model with the SageMaker Python SDK. For more information on getting started with JumpStart foundation models using the SageMaker Python SDK, see [Use foundation models with the SageMaker Python SDK](#).

Before you begin, make sure that you do the following:

- Upgrade to the latest version of the model that you use.
- Install the latest version of the SageMaker Python SDK.

## Important

To use the following workflow you must have [v2.198.0](#) or later of the SageMaker Python SDK installed.

### EULA acceptance when deploying a JumpStart model

For models that require the acceptance of an end-user license agreement, you must explicitly declare EULA acceptance when deploying your JumpStart model.

```
from sagemaker.jumpstart.model import JumpStartModel
```

```
model_id = "meta-textgeneration-llama-2-13b"
my_model = JumpStartModel(model_id=model_id)

# Declare EULA acceptance when deploying your JumpStart model
predictor = my_model.deploy(accept_eula=True)
```

The `accept_eula` value is `None` by default and must be explicitly redefined as `True` in order to accept the end-user license agreement. For more information, see [JumpStartModel](#).

### EULA acceptance when fine-tuning a JumpStart model

For fine-tuning models that require the acceptance of an end-user license agreement, you must explicitly declare EULA acceptance when defining your JumpStart estimator. After fine-tuning a pre-trained model, the weights of the original model are changed. Therefore, when you deploy the fine-tuned model later, you do not need to accept a EULA.

```
from sagemaker.jumpstart.estimator import JumpStartEstimator
model_id = "meta-textgeneration-llama-2-13b"

# Declare EULA acceptance when defining your JumpStart estimator
estimator = JumpStartEstimator(model_id=model_id, environment={"accept_eula": "true"})
estimator.fit(
    {"train": training_dataset_s3_path, "validation": validation_dataset_s3_path}
)
```

The `accept_eula` value is `None` by default and must be explicitly redefined as `"true"` within the estimator environment in order to accept the end-user license agreement. For more information, see [JumpStartEstimator](#).

### EULA acceptance SageMaker Python SDK versions earlier than 2.198.0

#### Important

When using versions earlier than [2.198.0](#) of the SageMaker Python SDK, you must use the SageMaker Predictor class to accept a model EULA.

After deploying a JumpStart foundation model programmatically using the SageMaker Python SDK, you can run inference against your deployed endpoint with the SageMaker [Predictor](#) class. For models that require the acceptance of an end-user license agreement, you must explicitly declare EULA acceptance in your call to the Predictor class:

```
predictor.predict(payload, custom_attributes="accept_eula=true")
```

The `accept_eula` value is `false` by default and must be explicitly redefined as `true` in order to accept the end-user license agreement. The predictor returns an error if you try to run inference while `accept_eula` is set to `false`. For more information on getting started with JumpStart foundation models using the SageMaker Python SDK, see [Use foundation models with the SageMaker Python SDK](#).

### **Important**

The `custom_attributes` parameter accepts key-value pairs in the format `"key1=value1;key2=value2"`. If you use the same key multiple times, the inference server uses the last value associated with the key. For example, if you pass `"accept_eula=false;accept_eula=true"` to the `custom_attributes` parameter, then the inference server associates the value `true` with the `accept_eula` key.

## Foundation model customization

Foundation models are extremely powerful models able to solve a wide array of tasks. To solve most tasks effectively, these models require some form of customization.

The recommended way to first customize a foundation model to a specific use case is through prompt engineering. Providing your foundation model with well-engineered, context-rich prompts can help achieve desired results without any fine-tuning or changing of model weights. For more information, see [Prompt engineering for foundation models](#).

If prompt engineering alone is not enough to customize your foundation model to a specific task, you can fine-tune a foundation model on additional domain-specific data. For more information, see [Foundation models and hyperparameters for fine-tuning](#). The fine-tuning process involves changing model weights.

If you want to customize your model with information from a knowledge library without any retraining, see [Retrieval Augmented Generation](#).

### Prompt engineering for foundation models

Prompt engineering is the process of designing and refining the prompts or input stimuli for a language model to generate specific types of output. Prompt engineering involves selecting

appropriate keywords, providing context, and shaping the input in a way that encourages the model to produce the desired response and is a vital technique to actively shape the behavior and output of foundation models.

Effective prompt engineering is crucial for directing model behavior and achieving desired responses. Through prompt engineering, you can control a model's tone, style, and domain expertise without more involved customization measures like fine-tuning. We recommend dedicating time to prompt engineering before you consider fine-tuning a model on additional data. The goal is to provide sufficient context and guidance to the model so that it can generalize and perform well on unseen or limited data scenarios.

## Zero-shot learning

Zero-shot learning involves training a model to generalize and make predictions on unseen classes or tasks. To perform prompt engineering in zero-shot learning environments, we recommend constructing prompts that explicitly provide information about the target task and the desired output format. For example, if you want to use a foundation model for zero-shot text classification on a set of classes that the model did not see during training, a well-engineered prompt could be: "Classify the following text as either sports, politics, or entertainment: **[input text]**." By explicitly specifying the target classes and the expected output format, you can guide the model to make accurate predictions even on unseen classes.

## Few-shot learning

Few-shot learning involves training a model with a limited amount of data for new classes or tasks. Prompt engineering in few-shot learning environments focuses on designing prompts that effectively use the limited available training data. For example, if you use a foundation model for an image classification task and only have a few examples of a new image class, you can engineer a prompt that includes the available labeled examples with a placeholder for the target class. For example, the prompt could be: "[image 1], [image 2], and [image 3] are examples of **[target class]**. Classify the following image as **[target class]**". By incorporating the limited labeled examples and explicitly specifying the target class, you can guide the model to generalize and make accurate predictions even with minimal training data.

## Supported inference parameters

Changing inference parameters might also affect the responses to your prompts. While you can try to add as much specificity and context as possible to your prompts, you can also experiment with supported inference parameters. The following are examples of some commonly supported inference parameters:

Inference Parameter	Description
max_new_tokens	The maximum output length of a foundation model response. Valid values: integer, range: Positive integer.
temperature	Controls the randomness in the output. Higher temperature results in an output sequence with low-probability words and lower temperature results in output sequence with high-probability words. If <code>temperature=0</code> , the response is made up of only the highest probability words (greedy decoding). Valid values: float, range: Positive float.
top_p	In each step of text generation, the model samples from the smallest possible set of words with a cumulative probability of <code>top_p</code> . Valid values: float, range: 0.0, 1.0.
return_full_text	If True, then the input text is part of the generated output text. Valid values: boolean, default: False.

For more information on foundation model inference, see [Deploy publicly available foundation models with the JumpStartModel class](#).

If prompt engineering is not sufficient to adapt your foundation model to specific business needs, domain-specific language, target tasks, or other requirements, you can consider fine-tuning your model on additional data or using Retrieval Augmented Generation (RAG) to augment your model architecture with enhanced context from archived knowledge sources. For more information, see [Foundation models and hyperparameters for fine-tuning](#) or [Retrieval Augmented Generation](#).

## Foundation models and hyperparameters for fine-tuning

Foundation models are computationally expensive and trained on a large, unlabeled corpus. Fine-tuning a pre-trained foundation model is an affordable way to take advantage of their broad capabilities while customizing a model on your own small, corpus. Fine-tuning is a customization method that involved further training and does change the weights of your model.

Fine-tuning might be useful to you if you need:

- to customize your model to specific business needs

- your model to successfully work with domain-specific language, such as industry jargon, technical terms, or other specialized vocabulary
- enhanced performance for specific tasks
- accurate, relative, and context-aware responses in applications
- responses that are more factual, less toxic, and better-aligned to specific requirements

There are two main approaches that you can take for fine-tuning depending on your use case and chosen foundation model.

1. If you're interested in fine-tuning your model on domain-specific data, see [Fine-tune a large language model \(LLM\) using domain adaptation](#).
2. If you're interested in instruction-based fine-tuning using prompt and response examples, see [Fine-tune a large language model \(LLM\) using prompt instructions](#).

## Foundation models available for fine-tuning

You can fine-tune any of the following JumpStart foundation models:

- Bloom 3B
- Bloom 7B1
- BloomZ 3B FP16
- BloomZ 7B1 FP16
- Code Llama 13B
- Code Llama 13B Python
- Code Llama 34B
- Code Llama 34B Python
- Code Llama 70B
- Code Llama 70B Python
- Code Llama 7B
- Code Llama 7B Python
- CyberAgentLM2-7B-Chat (CALM2-7B-Chat)
- Falcon 40B BF16
- Falcon 40B Instruct BF16

- Falcon 7B BF16
- Falcon 7B Instruct BF16
- Flan-T5 Base
- Flan-T5 Large
- Flan-T5 Small
- Flan-T5 XL
- Flan-T5 XXL
- Gemma 2B
- Gemma 2B Instruct
- Gemma 7B
- Gemma 7B Instruct
- GPT-2 XL
- GPT-J 6B
- GPT-Neo 1.3B
- GPT-Neo 125M
- GPT-NEO 2.7B
- LightGPT Instruct 6B
- Llama 2 13B
- Llama 2 13B Chat
- Llama 2 13B Neuron
- Llama 2 70B
- Llama 2 70B Chat
- Llama 2 7B
- Llama 2 7B Chat
- Llama 2 7B Neuron
- Mistral 7B
- Mixtral 8x7B
- Mixtral 8x7B Instruct
- RedPajama INCITE Base 3B V1
- RedPajama INCITE Base 7B V1

- RedPajama INCITE Chat 3B V1
- RedPajama INCITE Chat 7B V1
- RedPajama INCITE Instruct 3B V1
- RedPajama INCITE Instruct 7B V1
- Stable Diffusion 2.1

## Commonly supported fine-tuning hyperparameters

Different foundation models support different hyperparameters when fine-tuning. The following are commonly-supported hyperparameters that can further customize your model during training:

Inference Parameter	Description
epoch	The number of passes that the model takes through the fine-tuning dataset during training. Must be an integer greater than 1.
learning_rate	The rate at which the model weights are updated after working through each batch of fine-tuning training examples. Must be a positive float greater than 0.
instruction_tuned	Whether to instruction-train the model or not. Must be 'True' or 'False'.
per_device_train_batch_size	The batch size per GPU core or CPU for training. Must be a positive integer.
per_device_eval_batch_size	The batch size per GPU core or CPU for evaluation. Must be a positive integer.
max_train_samples	For debugging purposes or quicker training, truncate the number of training examples to this value. Value -1 means that the model uses all of the training samples. Must be a positive integer or -1.
max_val_samples	For debugging purposes or quicker training, truncate the number of validation examples to this value. Value -1 means

Inference Parameter	Description
	that the model uses all of the validation samples. Must be a positive integer or -1.
max_input_length	Maximum total input sequence length after tokenization. Sequences longer than this will be truncated. If -1, <code>max_input_length</code> is set to the minimum of 1024 and the <code>model_max_length</code> defined by the tokenizer. If set to a positive value, <code>max_input_length</code> is set to the minimum of the provided value and the <code>model_max_length</code> defined by the tokenizer. Must be a positive integer or -1.
validation_split_ratio	If there is no validation channel, ratio of train-validation split from the training data. Must be between 0 and 1.
train_data_split_seed	If validation data is not present, this fixes the random splitting of the input training data to training and validation data used by the model. Must be an integer.
preprocessing_num_workers	The number of processes to use for the pre-processing. If None, main process is used for pre-processing.
lora_r	Low-rank adaptation (LoRA) r value, which acts as the scaling factor for weight updates. Must be a positive integer.
lora_alpha	Low-rank adaptation (LoRA) alpha value, which acts as the scaling factor for weight updates. Generally 2 to 4 times the size of <code>lora_r</code> . Must be a positive integer.
lora_dropout	Dropout value for low-rank adaptation (LoRA) layers. Must be a positive float between 0 and 1.
int8_quantization	If True, model is loaded with 8 bit precision for training.
enable_fsdp	If True, training uses Fully Sharded Data Parallelism.

You can specify hyperparameter values when you fine-tune your model in Studio. For more information, see [Fine-tune a model in Studio](#).

You can also override default hyperparameter values when fine-tuning your model using the SageMaker Python SDK. For more information, see [Fine-tune publicly available foundation models with the JumpStartEstimator class](#).

## Fine-tune a large language model (LLM) using domain adaptation

Domain adaptation fine-tuning allows you to leverage pre-trained foundation models and adapt them to specific tasks using limited domain-specific data. If prompt engineering efforts do not provide enough customization, you can use domain adaption fine-tuning to get your model working with domain-specific language, such as industry jargon, technical terms, or other specialized data. This fine-tuning process modifies the weights of the model.

To fine-tune your model on a domain-specific dataset:

1. Prepare your training data. For instructions, see [the section called “Prepare and upload training data for domain adaptation fine-tuning”](#).
2. Create your fine-tuning training job. For instructions, see [the section called “Create a training job for instruction-based fine-tuning”](#).

You can find end-to-end examples in [the section called “Example notebooks”](#).

Domain adaptation fine-tuning is available with the following foundation models:

### Note

Some JumpStart foundation models, such as Llama 2 7B, require acceptance of an end-user license agreement before fine-tuning and performing inference. For more information, see [End-user license agreements](#).

- Bloom 3B
- Bloom 7B1
- BloomZ 3B FP16
- BloomZ 7B1 FP16
- GPT-2 XL
- GPT-J 6B

- GPT-Neo 1.3B
- GPT-Neo 125M
- GPT-NEO 2.7B
- Llama 2 13B
- Llama 2 13B Chat
- Llama 2 13B Neuron
- Llama 2 70B
- Llama 2 70B Chat
- Llama 2 7B
- Llama 2 7B Chat
- Llama 2 7B Neuron

## Prepare and upload training data for domain adaptation fine-tuning

Training data for domain adaptation fine-tuning can be provided in CSV, JSON, or TXT file format. All training data must be in a single file within a single folder.

The training data is taken from the **Text** column for CSV or JSON training data files. If no column is labeled **Text**, then the training data is taken from the first column for CSV or JSON training data files.

The following is an example body of a TXT file to be used for fine-tuning:

This report includes estimates, projections, statements relating to our business plans, objectives, and expected operating results that are “forward-looking statements” within the meaning of the Private Securities Litigation Reform Act of 1995, Section 27A of the Securities Act of 1933, and Section 21E of ....

## Split data for training and testing

You can optionally provide another folder containing validation data. This folder should also include one CSV, JSON, or TXT file. If no validation dataset is provided, then a set amount of the training data is set aside for validation purposes. You can adjust the percentage of training data used for validation when you choose the hyperparameters for fine-tuning your model.

## Upload fine-tuning data to Amazon S3

Upload your prepared data to Amazon Simple Storage Service (Amazon S3) to use when fine-tuning a JumpStart foundation model. You can use the following commands to upload your data:

```
from sagemaker.s3 import S3Uploader
import sagemaker
import random

output_bucket = sagemaker.Session().default_bucket()
local_data_file = "train.txt"
train_data_location = f"s3://{output_bucket}/training_folder"
S3Uploader.upload(local_data_file, train_data_location)
S3Uploader.upload("template.json", train_data_location)
print(f"Training data: {train_data_location}")
```

## Create a training job for instruction-based fine-tuning

After your data is uploaded to Amazon S3, you can fine-tune and deploy your JumpStart foundation model. To fine-tune your model in Studio, see [Fine-tune a model in Studio](#). To fine-tune your model using the SageMaker Python SDK, see [Fine-tune publicly available foundation models with the JumpStartEstimator class](#).

## Example notebooks

For more information on domain adaptation fine-tuning, see the following example notebooks:

- [SageMaker JumpStart Foundation Models - Fine-tuning text generation GPT-J 6B model on domain specific dataset](#)
- [Fine-tune LLaMA 2 models on JumpStart](#)

## Fine-tune a large language model (LLM) using prompt instructions

Instruction-based fine-tuning uses labeled examples to improve the performance of a pre-trained foundation model on a specific task. The labeled examples are formatted as prompt, response pairs and phrased as instructions. This fine-tuning process modifies the weights of the model. For more information on instruction-based fine-tuning, see the papers [Introducing FLAN: More generalizable Language Models with Instruction Fine-Tuning](#) and [Scaling Instruction-Finetuned Language Models](#).

Fine-tuned LAnguage Net (FLAN) models use instruction tuning to make models more amenable to solving general downstream NLP tasks. Amazon SageMaker JumpStart provides a number of foundation models in the FLAN model family. For example, FLAN-T5 models are instruction fine-tuned on a wide range of tasks to increase zero-shot performance for a variety of common use cases. With additional data and fine-tuning, instruction-based models can be further adapted to more specific tasks that weren't considered during pre-training.

To fine-tune a LLM on a specific task using prompt-response pairs task instructions:

1. Prepare your instructions in JSON files. For more information about the required format for the prompt-response pair files and the structure of the data folder, see [the section called “Prepare and upload training data for instruction-based fine-tuning”](#).
2. Create your fine-tuning training job. For instructions, see [the section called “Create a training job for instruction-based fine-tuning”](#).

You can find end-to-end examples in [the section called “Example notebooks”](#).

Only a subset of JumpStart foundation models are compatible with instruction-based fine-tuning. Instruction-based fine-tuning is available with the following foundation models:

 **Note**

Some JumpStart foundation models, such as Llama 2 7B, require acceptance of an end-user license agreement before fine-tuning and performing inference. For more information, see [End-user license agreements](#).

- Flan-T5 Base
- Flan-T5 Large
- Flan-T5 Small
- Flan-T5 XL
- Flan-T5 XXL
- Llama 2 13B
- Llama 2 13B Chat
- Llama 2 13B Neuron
- Llama 2 70B

- Llama 2 70B Chat
- Llama 2 7B
- Llama 2 7B Chat
- Llama 2 7B Neuron
- Mistral 7B
- RedPajama INCITE Base 3B V1
- RedPajama INCITE Base 7B V1
- RedPajama INCITE Chat 3B V1
- RedPajama INCITE Chat 7B V1
- RedPajama INCITE Instruct 3B V1
- RedPajama INCITE Instruct 7B V1

## Prepare and upload training data for instruction-based fine-tuning

Training data for instruction-based fine-tuning must be provided in JSON Lines text file format, where each line is a dictionary. All training data must be in a single folder. The folder can include multiple .jsonl files.

The training folder can also include a template JSON file (`template.json`) that describes the input and output formats of your data. If no template file is provided, the following template file is used:

```
{  
    "prompt": "Below is an instruction that describes a task, paired with an input that  
    provides further context. Write a response that appropriately completes the request.\n\n#### Instruction:\n{n[instruction]}\n\n#### Input:\n{n[context]}  
    "completion": "{response}"  
}
```

According to the `template.json` file, each .jsonl entry of the training data must include `{instruction}`, `{context}`, and `{response}` fields.

If you provide a custom template JSON file, use the "prompt" and "completion" keys to define your own required fields. According to the following custom template JSON file, each .jsonl entry of the training data must include `{question}`, `{context}`, and `{answer}` fields:

```
{  
  "prompt": "question: {question} context: {context}",  
  "completion": "{answer}"  
}
```

## Split data for training and testing

You can optionally provide another folder containing validation data. This folder should also include one or more .jsonl files. If no validation dataset is provided, then a set amount of the training data is set aside for validation purposes. You can adjust the percentage of training data used for validation when you choose the hyperparameters for fine-tuning your model.

## Upload fine-tuning data to Amazon S3

Upload your prepared data to Amazon Simple Storage Service (Amazon S3) to use when fine-tuning a JumpStart foundation model. You can use the following commands to upload your data:

```
from sagemaker.s3 import S3Uploader  
import sagemaker  
import random  
  
output_bucket = sagemaker.Session().default_bucket()  
local_data_file = "train.jsonl"  
train_data_location = f"s3://{output_bucket}/dolly_dataset"  
S3Uploader.upload(local_data_file, train_data_location)  
S3Uploader.upload("template.json", train_data_location)  
print(f"Training data: {train_data_location}")
```

## Create a training job for instruction-based fine-tuning

After your data is uploaded to Amazon S3, you can fine-tune and deploy your JumpStart foundation model. To fine-tune your model in Studio, see [Fine-tune a model in Studio](#). To fine-tune your model using the SageMaker Python SDK, see [Fine-tune publicly available foundation models with the JumpStartEstimator class](#).

## Example notebooks

For more information on instruction-based fine-tuning, see the following example notebooks:

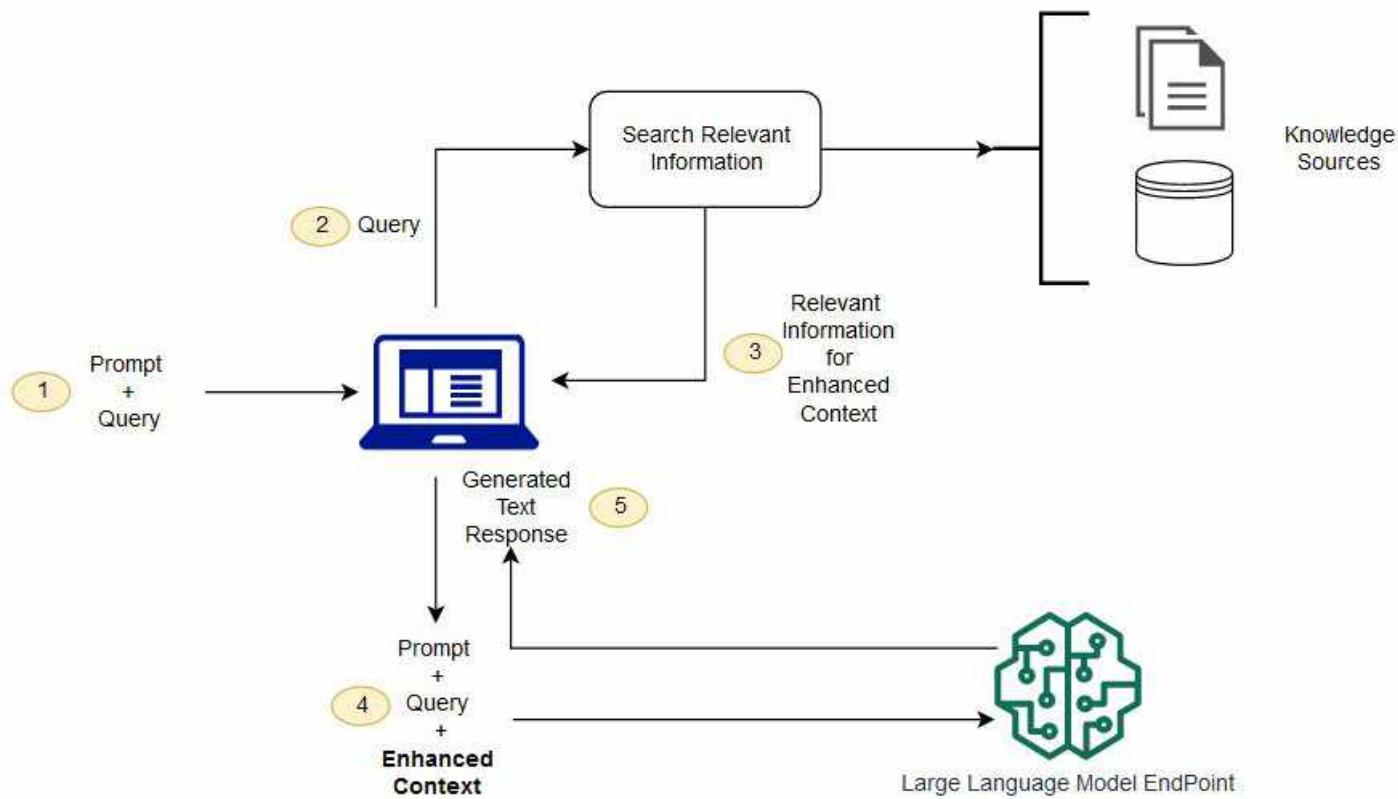
- [Fine-tune LLaMA 2 models on JumpStart](#)

- [Introduction to SageMaker JumpStart - Text Generation with Mistral models](#)
- [Introduction to SageMaker JumpStart - Text Generation with Falcon models](#)
- [SageMaker JumpStart Foundation Models - HuggingFace Text2Text Instruction Fine-Tuning](#)

## Retrieval Augmented Generation

Foundation models are usually trained offline, making the model agnostic to any data that is created after the model was trained. Additionally, foundation models are trained on very general domain corpora, making them less effective for domain-specific tasks. You can use Retrieval Augmented Generation (RAG) to retrieve data from outside a foundation model and augment your prompts by adding the relevant retrieved data in context. For more information about RAG model architectures, see [Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks](#).

With RAG, the external data used to augment your prompts can come from multiple data sources, such as a document repositories, databases, or APIs. The first step is to convert your documents and any user queries into a compatible format to perform relevancy search. To make the formats compatible, a document collection, or knowledge library, and user-submitted queries are converted to numerical representations using embedding language models. *Embedding* is the process by which text is given numerical representation in a vector space. RAG model architectures compare the embeddings of user queries within the vector of the knowledge library. The original user prompt is then appended with relevant context from similar documents within the knowledge library. This augmented prompt is then sent to the foundation model. You can update knowledge libraries and their relevant embeddings asynchronously.



The retrieved document should be large enough to contain useful context to help augment the prompt, but small enough to fit into the maximum sequence length of the prompt. You can use task-specific JumpStart models, such as the General Text Embeddings (GTE) model from Hugging Face, to provide the embeddings for your prompts and knowledge library documents. After comparing the prompt and document embeddings to find the most relevant documents, construct a new prompt with the supplemental context. Then, pass the augmented prompt to a text generation model of your choosing.

## Example notebooks

For more information on RAG foundation model solutions, see the following example notebooks:

- [Retrieval-Augmented Generation: Question Answering using LangChain and Cohere's Generate and Embedding Models from SageMaker JumpStart](#)
- [Retrieval-Augmented Generation: Question Answering using Llama-2, Pinecone and Custom Dataset](#)
- [Retrieval-Augmented Generation: Question Answering based on Custom Dataset with Open-sourced LangChain Library](#)

- [Retrieval-Augmented Generation: Question Answering based on Custom Dataset](#)
- [Retrieval-Augmented Generation: Question Answering using Llama-2 and Text Embedding Models](#)
- [Amazon SageMaker JumpStart - Text Embedding and Sentence Similarity](#)

You can clone the [Amazon SageMaker AI examples repository](#) to run the available JumpStart foundation model examples in the Jupyter environment of your choice within Studio. For more information on applications that you can use to create and access Jupyter in SageMaker AI, see [Applications supported in Amazon SageMaker Studio](#).

## Evaluate a text generation foundation model in Studio

### Note

Foundation Model Evaluations (FMEval) is in preview release for Amazon SageMaker Clarify and is subject to change.

### Important

In order to use SageMaker Clarify Foundation Model Evaluations, you must upgrade to the new Studio experience. As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The foundation evaluation feature can only be used in the updated experience. For information about how to update Studio, see [Migration from Amazon SageMaker Studio Classic](#). For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

Amazon SageMaker JumpStart has integrations with SageMaker Clarify Foundation Model Evaluations (FMEval) in Studio. If a JumpStart model has built-in evaluation capabilities available, you can choose **Evaluate** in the upper right corner of the model detail page in the JumpStart Studio UI. For more information on navigating the JumpStart Studio UI, see [Open and use JumpStart in Studio](#),

Use Amazon SageMaker JumpStart to evaluate text-based foundation models with FMEval. You can use these model evaluations to compare model quality and responsibility metrics for

one model, between two models, or between different versions of the same model, to help you quantify model risks. FMEval can evaluate text-based models that perform the following tasks:

- **Open-ended generation** – The production of natural human responses to text that does not have a pre-defined structure.
- **Text summarization** – The generation of a concise and condensed summary while retaining the meaning and key information contained in larger text.
- **Question Answering** – The generation of an answer in natural language to a question.
- **Classification** – The assignment of a class, such as positive versus negative to a text passage based on its content.

You can use FMEval to automatically evaluate model responses based on specific benchmarks. You can also evaluate model responses against your own criteria by bringing your own prompt datasets. FMEval provides a user interface (UI) that guides you through the setup and configuration of an evaluation job. You can also use the FMEval library inside your own code.

Every evaluation requires quota for two instances:

- **Hosting instance** – An instance that hosts and deploys an LLM.
- **Evaluation instance** – An instance that is used to prompt and perform an evaluation of an LLM on the hosting instance.

If your LLM is already deployed, provide the endpoint, and SageMaker AI will use your **hosting instance** to host and deploy the LLM.

If you are evaluating a JumpStart model that is not yet deployed to your account, FMEval creates a temporary **hosting instance** for you in your account, and keeps it deployed only for the length of your evaluation. FMEval uses the default instance that JumpStart recommends for the chosen LLM as your hosting instance. You must have sufficient quota for this recommended instance.

Every evaluation also uses an evaluation instance to provide prompts to and score the responses from the LLM. You must also have sufficient quota and memory to run the evaluation algorithms. The quota and memory requirements of the evaluation instance are generally smaller than those required for a hosting instance. We recommend selecting the `m1.m5.2xlarge` instance. For more information about quota and memory, see [Resolve errors when creating a model evaluation job in Amazon SageMaker AI](#).

Automatic evaluations can be used to score LLMs across the following dimensions:

- Accuracy – For text summarization, question answering, and text classification
- Semantic robustness – For open-ended generation, text summarization and text classification tasks
- Factual knowledge – For open-ended generation
- Prompt stereotyping – For open-ended generation
- Toxicity – For open-ended generation, text summarization, and question answering

You can also use human evaluations to manually evaluate model responses. The FMEval UI guides you through a workflow of selecting one or more models, provisioning resources, and writing instructions for and contacting your human workforce. After the human evaluation is complete, the results are displayed in FMEval.

You can access model evaluation through the JumpStart landing page in Studio by selecting a model to evaluate and then choosing **Evaluate**. Note that not all JumpStart models have evaluation capabilities available. For more information about how to configure, provision and run FMEval, see [What are Foundation Model Evaluations?](#)

## Example notebooks

For step-by-step examples on how to use publicly available JumpStart foundation models with the SageMaker Python SDK, refer to the following notebooks on text generation, image generation, and model customization.

### Note

Proprietary and publicly available JumpStart foundation models have different SageMaker AI Python SDK deployment workflows. Discover proprietary foundation model example notebooks through Amazon SageMaker Studio Classic or the SageMaker AI console. For more information, see [JumpStart foundation model usage](#).

You can clone the [Amazon SageMaker AI examples repository](#) to run the available JumpStart foundation model examples in the Jupyter environment of your choice within Studio. For more information on applications that you can use to create and access Jupyter in SageMaker AI, see [Applications supported in Amazon SageMaker Studio](#).

## Time series forecasting

You can use the Chronos models to forecast time series data. They're based on the language model architecture. Use the [Introduction to SageMaker JumpStart - Time Series Forecasting with Chronos](#) notebook to get started.

For information about the available Chronos models, see [Available foundation models](#).

## Text generation

Explore text generation example notebooks, including guidance on general text generation workflows, multilingual text classification, real-time batch inference, few-shot learning, chatbot interactions, and more.

- [SageMaker JumpStart Foundation Models - HuggingFace Text2Text Generation with FLAN-T5 XL as an example](#)
- [SageMaker JumpStart Foundation Models - BloomZ: Multilingual Text Classification, Question and Answering, Code Generation, Paragraph rephrase, and More](#)
- [SageMaker JumpStart Foundation Models - HuggingFace Text2Text Generation Batch Transform and Real-Time Batch Inference](#)
- [SageMaker JumpStart Foundation Models - GPT-J, GPT-Neo Few-shot learning](#)
- [SageMaker JumpStart Foundation Models - Chatbots](#)
- [Introduction to SageMaker JumpStart - Text Generation with Mistral models](#)
- [Introduction to SageMaker JumpStart - Text Generation with Falcon models](#)

## Image generation

Get started with text-to-image Stable Diffusion models, learn how to deploy an inpainting model, and experiment with a simple workflow to generate images of your dog.

- [Introduction to JumpStart - Text to Image](#)
- [Introduction to JumpStart Image editing - Stable Diffusion Inpainting](#)
- [Generate fun images of your dog](#)

## Model customization

Sometimes your use case requires greater foundation model customization for specific tasks. For more information on model customization approaches, see [Foundation model customization](#) or explore one of the following example notebooks.

- [SageMaker JumpStart Foundation Models - Fine-tuning text generation GPT-J 6B model on domain specific dataset](#)
- [SageMaker JumpStart Foundation Models - HuggingFace Text2Text Instruction Fine-Tuning](#)
- [Retrieval-Augmented Generation: Question Answering using LangChain and Cohere's Generate and Embedding Models from SageMaker JumpStart](#)
- [Retrieval-Augmented Generation: Question Answering using Llama-2, Pinecone and Custom Dataset](#)
- [Retrieval-Augmented Generation: Question Answering based on Custom Dataset with Open-sourced LangChain Library](#)
- [Retrieval-Augmented Generation: Question Answering based on Custom Dataset](#)
- [Retrieval-Augmented Generation: Question Answering using Llama-2 and Text Embedding Models](#)
- [Amazon SageMaker JumpStart - Text Embedding and Sentence Similarity](#)

## Private curated hubs for foundation model access control in JumpStart

Curate pretrained JumpStart foundation models for your organization with private hubs. Use the latest publicly available and proprietary foundation models while enforcing governance guardrails and ensuring that your organization can only access approved models.

Use private model hubs to share models and notebooks, centralize model artifacts, improve model discoverability, and streamline model use within your organization. Administrators can create private hubs that include subsets of models tailored to different teams, use cases, or security requirements. Administrators can create a JumpStart private model hub using the SageMaker Python SDK. Users can then browse, train, and deploy the curated set of models using Amazon SageMaker Studio or the SageMaker Python SDK.

For more information on creating a private model hub, see [Admin guide for private model hubs in Amazon SageMaker JumpStart](#).

For more information on sharing private model hubs across accounts, see [Cross-account sharing for private model hubs with AWS Resource Access Manager](#).

For more information on accessing a private model hub, see [Access curated model hubs in Amazon SageMaker JumpStart](#).

## Admin guide for private model hubs in Amazon SageMaker JumpStart

There are actions that administrators can take related to curated model hubs that users within your organization can access. This includes creating, adding, deleting, and managing access of private hubs. This page also includes information about the supported AWS Regions for curated private hubs, as well as the prerequisites needed to use curated private model hubs.

### Supported AWS Regions

Curated private hubs are currently generally available in the following AWS commercial Regions:

- us-east-1
- us-east-2
- us-west-2
- eu-west-1
- eu-central-1
- ap-northeast-1
- ap-northeast-2
- ap-south-1
- ap-southeast-1
- ap-southeast-2
- il-central-1 (SDK only)

The default maximum number of hubs allowed in a single Region is 50.

### Prerequisites

To use a curated private hub in Studio, you must have the following prerequisites:

- An AWS account with administrator access
- An AWS Identity and Access Management (IAM) role with access to Amazon SageMaker Studio
- An Amazon SageMaker AI domain with JumpStart enabled

- If your users try to use proprietary models, they must have subscriptions to those models in AWS Marketplace.
- AWS accounts that are deploying proprietary models must have subscriptions to those models in AWS Marketplace.

For more information on getting started with Studio, see [Amazon SageMaker Studio](#).

## Create a private model hub

Use the following steps to create a private hub to manage access control for pretrained JumpStart foundation models for your organization. You must install the SageMaker Python SDK and configure the necessary IAM permissions before creating a model hub.

### Create a private hub

1. Install the SageMaker Python SDK and import the necessary Python packages.

```
# Install the SageMaker Python SDK
!pip3 install sagemaker --force-reinstall --quiet

# Import the necessary Python packages
import boto3
from sagemaker import Session
from sagemaker.jumpstart.hub.hub import Hub
```

2. Initialize a SageMaker AI Session.

```
sm_client = boto3.client('sagemaker')
session = Session(sagemaker_client=sm_client)
session.get_caller_identity_arn()
```

3. Configure the details of your private hub such as the internal hub name, UI display name, and UI hub description.

#### Note

If you do not specify an Amazon S3 bucket name when creating your hub, the SageMaker hub service creates a new bucket on your behalf. The new bucket has the following naming structure: `sagemaker-hubs-REGION-ACCOUNT_ID`.

```
HUB_NAME="Example-Hub"  
HUB_DISPLAY_NAME="Example Hub UI Name"  
HUB_DESCRIPTION="A description of the example private curated hub."  
REGION="us-west-2"
```

4. Check that your **Admin** IAM role has the necessary Amazon S3 permissions to create a private hub. If your role does not have the necessary permissions, navigate to the **Roles** page in the IAM console. Choose the **Admin** role and then choose **Add permissions** in the **Permissions policies** pane to create an inline policy with the following permissions using the JSON editor:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Action": [  
                "s3>ListBucket",  
                "s3GetObject",  
                "s3GetObjectTagging"  
            ],  
            "Resource": [  
                "arn:aws:s3:::jumpstart-cache-prod-REGION",  
                "arn:aws:s3:::jumpstart-cache-prod-REGION/*"  
            ],  
            "Effect": "Allow"  
        }  
    ]  
}
```

5. Create a private model hub using your configurations from **Step 3** using `hub.create()`.

```
hub = Hub(hub_name=HUB_NAME, sagemaker_session=session)  
  
try:  
    # Create the private hub  
    hub.create(  
        description=HUB_DESCRIPTION,  
        display_name=HUB_DISPLAY_NAME  
    )  
    print(f"Successfully created Hub with name {HUB_NAME} in {REGION}")  
    # Check that no other hubs with this internal name exist  
except Exception as e:
```

```
if "ResourceInUse" in str(e):
    print(f"A hub with the name {HUB_NAME} already exists in your account.")
else:
    raise e
```

6. Verify the configuration of your new private hub with the following `describe` command:

```
hub.describe()
```

## Add models to a private hub

After creating a private hub, you can then add allow-listed models. For the full list of available JumpStart models, see the [Built-in Algorithms with pre-trained Model Table](#) in the SageMaker Python SDK reference.

1. You can filter through the available models programmatically using the `hub.list_sagemaker_public_hub_models()` method. You can optionally filter by categories such as framework ("framework == pytorch"), tasks such as image classification ("task == ic"), and more. For more information about filters, see [notebook\\_utils.py](#). The filter parameter in the `hub.list_sagemaker_public_hub_models()` method is optional.

```
filter_value = "framework == meta"
response = hub.list_sagemaker_public_hub_models(filter=filter_value)
models = response["hub_content_summaries"]
while response["next_token"]:
    response = hub.list_sagemaker_public_hub_models(filter=filter_value,
    next_token=response["next_token"])
    models.extend(response["hub_content_summaries"])

print(models)
```

2. You can then add the filtered models by specifying the model ARN in the `hub.create_model_reference()` method.

```
for model in models:
    print(f"Adding {model.get('hub_content_name')} to Hub")
    hub.create_model_reference(model_arn=model.get("hub_content_arn"),
    model_name=model.get("hub_content_name"))
```

## Cross-account sharing for private model hubs with AWS Resource Access Manager

After creating a private model hub, you can share the hub to the necessary accounts using AWS Resource Access Manager (AWS RAM). For more information on creating a private hub, see [Create a private model hub](#). The following page gives in-depth information about managed permissions related to private hubs within AWS RAM. For information about how to create a resource share within AWS RAM, see [Set up cross-account hub sharing](#).

### Managed permissions for curated private hubs

The available access permissions are read, read and use, and full access permissions. The permission name, description, and list of specific APIs available for each permission are listed in the following:

- Read permission (`AWSRAMPermissionSageMaker AIHubRead`): The read privilege allows resource consumer accounts to read contents in the shared hubs and view details and metadata.
  - `DescribeHub`: Retrieves details about a hub and its configuration
  - `DescribeHubContent`: Retrieves details about a model available in a specific hub
  - `ListHubContent`: Lists all models available in a hub
  - `ListHubContentVersions`: Lists the version of all models available in a hub
- Read and use permission (`AWSRAMPermissionSageMaker AIHubReadAndUse`): The read and use privilege allows resource consumer accounts to read contents in the shared hubs and deploy available models for inference.
  - `DescribeHub`: Retrieves details about a hub and its configuration
  - `DescribeHubContent`: Retrieves details about a model available in a specific hub
  - `ListHubContent`: Lists all models available in a hub
  - `ListHubContentVersions`: Lists the version of all models available in a hub
  - `DeployHubModel`: Allows access to deploy available open-weight hub models for inference
- Full access permission (`AWSRAMPermissionSageMaker AIHubFullAccessPolicy`): The full access privilege allows resource consumer accounts to read contents in the shared hubs, add and remove hub content, and deploy available models for inference.
  - `DescribeHub`: Retrieves details about a hub and its configuration
  - `DescribeHubContent`: Retrieves details about a model available in a specific hub
  - `ListHubContent`: Lists all models available in a hub
  - `ListHubContentVersions`: Lists the version of all models available in a hub
  - `ImportHubContent`: Imports hub content

- `DeleteHubContent`: Deletes hub content
- `CreateHubContentReference`: Creates a hub content reference that shares a model from the SageMaker AI **Public models** hub to a private hub
- `DeleteHubContentReference`: Delete a hub content reference that shares a model from the SageMaker AI **Public models** hub to a private hub
- `DeployHubModel`: Allows access to deploy available open-weight hub models for inference

`DeployHubModel` permissions are not required for proprietary models.

## Set up cross-account hub sharing

SageMaker uses [AWS Resource Access Manager \(AWS RAM\)](#) to help you securely share your private hubs across accounts. Set up cross-account hub sharing using the following instructions along with the [Sharing your AWS resources](#) instructions in the *AWS RAM User Guide*.

### Create a resource share

1. Select **Create resource share** through the [AWS RAM console](#).
2. When specifying resource share details, choose the **SageMaker Hubs** resource type and select one or more private hubs that you want to share. When you share a hub with any other account, all of its contents are also shared implicitly.
3. Associate permissions with your resource share. For more information about managed permissions, see [Managed permissions for curated private hubs](#)
4. Use AWS account IDs to specify the accounts to which you want to grant access to your shared resources.
5. Review your resource share configuration and select **Create resource share**. It may take a few minutes for the resource share and principal associations to complete.

For more information, see [Sharing your AWS resources](#) in the *AWS Resource Access Manager User Guide*.

After the resource share and principal associations are set, the specified AWS accounts receive an invitation to join the resource share. The AWS accounts must accept the invite to gain access to any shared resources.

For more information on accepting a resource share invite through AWS RAM, see [Using shared AWS resources](#) in the *AWS Resource Access Manager User Guide*.

## Delete models from a private hub

You can delete models from a private hub used by your organization by specifying the model ARN in the `hub.delete_model_reference()` method. This removes access to the model from the private hub.

```
hub.delete_model_reference(model-name)
```

## Remove access to the SageMaker Public models hub

In addition to adding a private curated hub to JumpStart in Studio, you can also remove access to the SageMaker **Public models** hub for your users. The SageMaker **Public models** hub has access to all available JumpStart foundation models.

If you remove access to the SageMaker **Public models** hub and a user has access to only one private hub, then the user is taken directly into that private hub when they choose **JumpStart** in the left navigation pane in Studio. If a user has access to multiple private hubs, then the user is taken to a **Hubs** menu page when they choose **JumpStart** in the left navigation pane in Studio.

Remove access to the SageMaker **Public models** hub for your users with the following inline policy:

### Note

You can specify any additional Amazon S3 buckets that you want your hub to access in the policy below. Be sure to replace **REGION** with the Region of your hub.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "s3:*",
            "Effect": "Deny",
            "NotResource": [
                "arn:aws:s3:::jumpstart-cache-prod-REGION/*.ipynb",
                "arn:aws:s3:::jumpstart-cache-prod-REGION/*eula*",
                "Additional-S3-bucket-ARNs-as-needed"
            ],
        },
        {
    }
```

```
        "Action": "sagemaker:*",
        "Effect": "Deny",
        "Resource": [
            "arn:aws:sagemaker:REGION:aws:hub/SageMakerPublicHub",
            "arn:aws:sagemaker:REGION:aws:hub-content/SageMakerPublicHub/*/*"
        ]
    }
]
```

## Delete a private hub

You can delete a private hub from your admin account. Before deleting a private hub, you must first remove any content in that hub. Delete hub contents and hubs with the following commands:

```
# List the model references in the private hub
response = hub.list_models()
models = response["hub_content_summaries"]
while response["next_token"]:
    response = hub.list_models(next_token=response["next_token"])
    models.extend(response["hub_content_summaries"])

# Delete all model references in the hub
for model in models:
    hub.delete_model_reference(model_name=model.get('HubContentName'))

# Delete the private hub
hub.delete()
```

## Troubleshooting

The following sections give information about IAM permissions issues that might arise when creating a private model hub, as well as information about how to resolve those issues.

### **ValidationException when calling the CreateModel operation: Could not access model data**

This exception arises when you do not have the appropriate Amazon S3 permissions configured for your **Admin** role. For more information on the Amazon S3 permissions needed to create a private hub, see [Step 3 in Create a private model hub](#).

### **Access Denied or Forbidden when calling create()**

You are denied access when creating a private hub if you do not have the appropriate permissions to access the Amazon S3 bucket associated with the SageMaker **Public models** hub. For more information on the Amazon S3 permissions needed to create a private hub, see **Step 3** in [Create a private model hub](#).

## Access curated model hubs in Amazon SageMaker JumpStart

You can access a private model hub either through Studio or through the SageMaker Python SDK.

### Access your private model hub in Studio

#### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

In Amazon SageMaker Studio, open the JumpStart landing page either through the **Home** page or the **Home** menu on the left-side panel. This opens the **SageMaker JumpStart** landing page where you can explore model hubs and search for models.

- From the **Home** page, choose **JumpStart** in the **Prebuilt and automated solutions** pane.
- From the **Home** menu in the left panel, navigate to the **JumpStart** node.

For more information on getting started with Amazon SageMaker Studio, see [Amazon SageMaker Studio](#).

From the **SageMaker JumpStart** landing page in Studio, you can explore any private model hubs that include allow-listed models for your organization. If you only have access to one model hub, then the **SageMaker JumpStart** landing page takes you directly into that hub. If you have access to multiple hubs, you are taken to the **Hubs** page.

For more information on fine-tuning, deploying, and evaluating models that you have access to in Studio, see [Use foundation models in Studio](#).

## Access your private model hub using the SageMaker Python SDK

You can access your private model hub using the SageMaker Python SDK. Your access to read, use, or edit your curated hub is provided by your administrator.

### Note

If a hub is shared across accounts, then the HUB\_NAME must be the hub ARN. If a hub is not shared across accounts, then the HUB\_NAME can be the hub name.

1. Install the SageMaker Python SDK and import the necessary Python packages.

```
# Install the SageMaker Python SDK
!pip3 install sagemaker --force-reinstall --quiet

# Import the necessary Python packages
import boto3
from sagemaker import Session
from sagemaker.jumpstart.hub.hub import Hub
from sagemaker.jumpstart.model import JumpStartModel
from sagemaker.jumpstart.estimator import JumpStartEstimator
```

2. Initialize a SageMaker AI session and connect to your private hub using the hub name and Region.

```
# If a hub is shared across accounts, then the HUB_NAME must be the hub ARN
HUB_NAME="Example-Hub-ARN"
REGION="us-west-2"

# Initialize a SageMaker session
sm_client = boto3.client('sagemaker')
sm_runtime_client = boto3.client('sagemaker-runtime')
session = Session(sagemaker_client=sm_client,
                  sagemaker_runtime_client=sm_runtime_client)

# Initialize the private hub
hub = Hub(hub_name=HUB_NAME, sagemaker_session=session)
```

3. After connecting to a private hub, you can list all available models in that hub using the following commands:

```
response = hub.list_models()
models = response["hub_content_summaries"]
while response["next_token"]:
    response = hub.list_models(next_token=response["next_token"])
    models.extend(response["hub_content_summaries"])

print(models)
```

4. You can get more information about a specific model using the model name with the following command:

```
response = hub.describe_model(model_name="example-model")
print(response)
```

For more information on fine-tuning and deploying models that you have access to using the SageMaker Python SDK, see [Use foundation models with the SageMaker Python SDK](#).

## Amazon SageMaker JumpStart in Studio Classic

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following JumpStart features are only available in Amazon SageMaker Studio Classic.

- [Task-Specific Models](#)
- [Shared Models and Notebooks](#)
- [End-to-end JumpStart solution templates](#)
- [Amazon SageMaker JumpStart Industry: Financial](#)

## Task-Specific Models

JumpStart supports task-specific models across fifteen of the most popular problem types. Of the supported problem types, Vision and NLP-related types total thirteen. There are eight problem types that support incremental training and fine-tuning. For more information about incremental training and hyper-parameter tuning, see [SageMaker AI Automatic Model Tuning](#). JumpStart also supports four popular algorithms for tabular data modeling.

You can search and browse models from the JumpStart landing page in Studio or Studio Classic. When you select a model, the model detail page provides information about the model, and you can train and deploy your model in a few steps. The description section describes what you can do with the model, the expected types of inputs and outputs, and the data type needed for fine-tuning your model.

You can also programmatically utilize models with the [SageMaker Python SDK](#). For a list of all available models, see the [JumpStart Available Model Table](#).

The list of problem types and links to their example Jupyter notebooks are summarized in the following table.

Problem types	Supports inference with pre-trained models	Trainable on a custom dataset	Supported frameworks	Example Notebooks
Image classification	Yes	Yes	PyTorch, TensorFlow	<a href="#">Introduction to JumpStart - Image Classification</a>
Object detection	Yes	Yes	PyTorch, TensorFlow, MXNet	<a href="#">Introduction to JumpStart - Object Detection</a>
Semantic segmentation	Yes	Yes	MXNet	<a href="#">Introduction to JumpStart - Semantic Segmentation</a>

Problem types	Supports inference with pre-trained models	Trainable on a custom dataset	Supported frameworks	Example Notebooks
Instance segmentation	Yes	Yes	MXNet	<a href="#">Introduction to JumpStart - Instance Segmentation</a>
Image embedding	Yes	No	TensorFlow, MXNet	<a href="#">Introduction to JumpStart - Image Embedding</a>
Text classification	Yes	Yes	TensorFlow	<a href="#">Introduction to JumpStart - Text Classification</a>
Sentence pair classification	Yes	Yes	TensorFlow, Hugging Face	<a href="#">Introduction to JumpStart - Sentence Pair Classification</a>
Question answering	Yes	Yes	PyTorch, Hugging Face	<a href="#">Introduction to JumpStart - Question Answering</a>
Named entity recognition	Yes	No	Hugging Face	<a href="#">Introduction to JumpStart - Named Entity Recognition</a>
Text summarization	Yes	No	Hugging Face	<a href="#">Introduction to JumpStart - Text Summarization</a>

Problem types	Supports inference with pre-trained models	Trainable on a custom dataset	Supported frameworks	Example Notebooks
Text generation	Yes	No	Hugging Face	<a href="#">Introduction to JumpStart - Text Generation</a>
Machine translation	Yes	No	Hugging Face	<a href="#">Introduction to JumpStart - Machine Translation</a>
Text embedding	Yes	No	TensorFlow, MXNet	<a href="#">Introduction to JumpStart - Text Embedding</a>

Problem types	Supports inference with pre-trained models	Trainable on a custom dataset	Supported frameworks	Example Notebooks
Tabular classification	Yes	Yes	LightGBM, CatBoost, XGBoost, AutoGluon -Tabular, TabTransformer, Linear Learner	<a href="#">Introduction to JumpStart - Tabular Classification - LightGBM, CatBoost</a> <a href="#">Introduction to JumpStart - Tabular Classification - XGBoost, Linear Learner</a> <a href="#">Introduction to JumpStart - Tabular Classification - AutoGluon Learner</a> <a href="#">Introduction to JumpStart - Tabular Classification - TabTransformer Learner</a>

Problem types	Supports inference with pre-trained models	Trainable on a custom dataset	Supported frameworks	Example Notebooks
Tabular regression	Yes	Yes	LightGBM, CatBoost, XGBoost, AutoGluon -Tabular, TabTransformer, Linear Learner	<a href="#">Introduction to JumpStart - Tabular Regression - LightGBM, CatBoost</a> <a href="#">Introduction to JumpStart – Tabular Regression - XGBoost, Linear Learner</a> <a href="#">Introduction to JumpStart – Tabular Regression - AutoGluon Learner</a> <a href="#">Introduction to JumpStart – Tabular Regression - TabTransformer Learner</a>

## Deploy a Model

When you deploy a model from JumpStart, SageMaker AI hosts the model and deploys an endpoint that you can use for inference. JumpStart also provides an example notebook that you can use to access the model after it's deployed.

### ⚠ Important

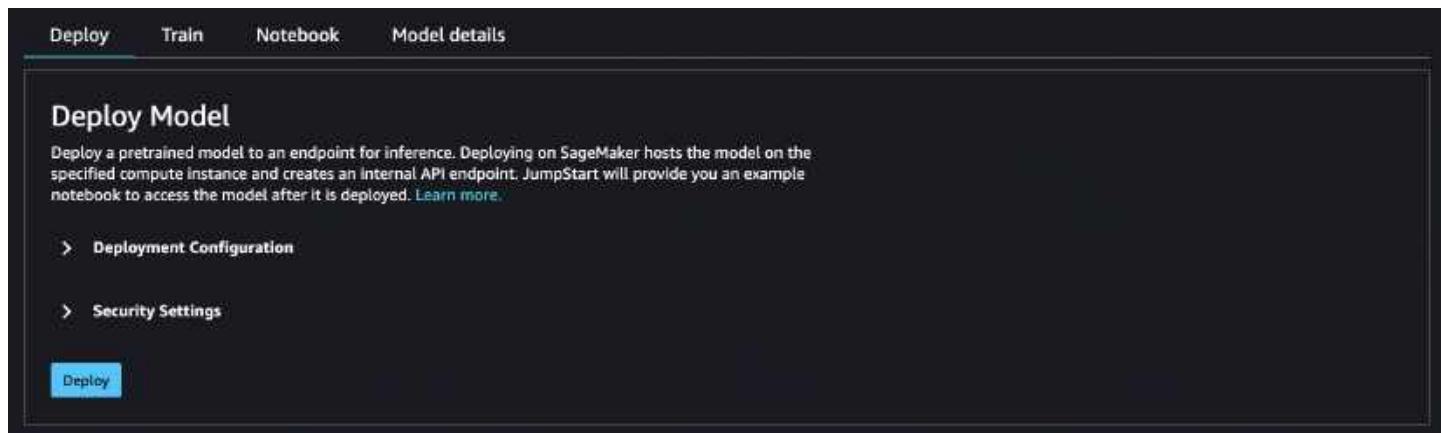
As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

### ℹ Note

For more information on JumpStart model deployment in Studio, see [Deploy a model in Studio](#)

## Model deployment configuration

After you choose a model, the model's tab opens. In the **Deploy Model** pane, choose **Deployment Configuration** to configure your model deployment.



The default instance type for deploying a model depends on the model. The instance type is the hardware that the training job runs on. In the following example, the `m1.p2.xlarge` instance is the default for this particular BERT model.

You can also change the endpoint name, add key:value resource tags, activate or deactivate the jumpstart- prefix for any JumpStart resources related to the model, and specify an Amazon S3 bucket for storing model artifacts used by your SageMaker AI endpoint.

✓ **Deployment Configuration**

Customize the machine type and endpoint name. [Learn more.](#)

SageMaker hosting instance [i](#)

ml.p2.xlarge ▾

Endpoint name

tf-tc-bert-en-uncased-l-12-h-768-a-12-2

Custom resource tags [i](#)

key:value [Add](#)

Use JumpStart prefix [i](#)

Custom model artifact S3 bucket [i](#)

Default model artifact S3 bucket  Find S3 bucket  Enter S3 bucket location

The model artifact used by your SageMaker endpoint will be stored in your SageMaker default bucket.

s3://sagemaker-us-west-2-671655899342

[Reset to default](#)

Choose **Security Settings** to specify the AWS Identity and Access Management (IAM) role, Amazon Virtual Private Cloud (Amazon VPC), and encryption keys for the model.

The screenshot shows the 'Security Settings' section of the Amazon SageMaker AI JumpStart configuration interface. It includes fields for specifying IAM roles, VPC connectivity, and encryption keys.

**Specify the IAM role that Amazon SageMaker should use to deploy your model.** [Learn more.](#)

Default IAM role    Find IAM role    Input IAM role

Amazon SageMaker will deploy your model using your Studio execution role.

**Specify whether your model should connect to a virtual private cloud (VPC).** [Learn more.](#)

No VPC    Find VPC    Input VPC

No VPC will be used to access your model container.

**Specify the encryption keys to secure your data.** [Learn more.](#)

Default encryption keys    Find encryption keys    Input encryption keys

Encrypt your model artifact at rest using your account's default KMS key for S3. [Learn more.](#)

## Model deployment security

When you deploy a model with JumpStart, you can specify an IAM role, Amazon VPC, and encryption keys for the model. If you don't specify any values for these entries: The default IAM role is your Studio Classic runtime role; default encryption is used; no Amazon VPC is used.

### IAM role

You can select an IAM role that is passed as part of training jobs and hosting jobs. SageMaker AI uses this role to access training data and model artifacts. If you don't select an IAM role, SageMaker AI deploys the model using your Studio Classic runtime role. For more information about IAM roles, see [AWS Identity and Access Management for Amazon SageMaker AI](#).

The role that you pass must have access to the resources that the model needs, and must include all of the following.

- For training jobs: [CreateTrainingJob API: Execution Role Permissions](#).
- For hosting jobs: [CreateModel API: Execution Role Permissions](#).

**Note**

You can scope down the Amazon S3 permissions granted in each of the following roles. Do this by using the ARN of your Amazon Simple Storage Service (Amazon S3) bucket and the JumpStart Amazon S3 bucket.

```
[  
 {  
     "Effect": "Allow",  
     "Action": [  
         "s3:GetObject",  
         "s3>ListBucket"  
     ],  
     "Resource": [  
         "arn:aws:s3:::jumpstart-cache-prod-<region>/*",  
         "arn:aws:s3:::jumpstart-cache-prod-<region>",  
         "arn:aws:s3:::<bucket>/*"  
     ]  
 },  
 {  
     "Effect": "Allow",  
     "Action": [  
         "cloudwatch:PutMetricData",  
         "logs>CreateLogStream",  
         "logs:PutLogEvents",  
         "logs>CreateLogGroup",  
         "logs:DescribeLogStreams",  
         "ecr:GetAuthorizationToken"  
     ],  
     "Resource": [  
         "*"  
     ]  
 },  
 {  
     "Effect": "Allow",  
     "Action": [  
         "ecr:BatchGetImage",  
         "ecr:BatchCheckLayerAvailability",  
         "ecr:GetDownloadUrlForLayer"  
     ],  
     "Resource": [  
         "*"  
     ]  
 }
```

```
    },  
]  
}
```

## Find IAM role

If you select this option, you must select an existing IAM role from the dropdown list.

**Specify the IAM role that Amazon SageMaker should use to deploy your model. [Learn more.](#)**

Default IAM role     Find IAM role     Input IAM role

Amazon SageMaker will deploy your model using the IAM role you select below.

**Execution role ⓘ**

**Select...** ▼

## Input IAM role

If you select this option, you must manually enter the ARN for an existing IAM role. If your Studio Classic runtime role or Amazon VPC block the `iam:list*` call, you must use this option to use an existing IAM role.

**Specify the IAM role that Amazon SageMaker should use to deploy your model. [Learn more.](#)**

Default IAM role     Find IAM role     Input IAM role

Amazon SageMaker will deploy your model using the IAM role you type below.

**Execution role arn ⓘ**

`arn:aws:iam::account-id:role/role-name`

## Amazon VPC

All JumpStart models run in network isolation mode. After the model container is created, no more calls can be made. You can select an Amazon VPC that is passed as part of training jobs and

hosting jobs. SageMaker AI uses this Amazon VPC to push and pull resources from your Amazon S3 bucket. This Amazon VPC is different from the Amazon VPC that limits access to the public internet from your Studio Classic instance. For more information about the Studio Classic Amazon VPC, see [Connect Studio notebooks in a VPC to external resources](#).

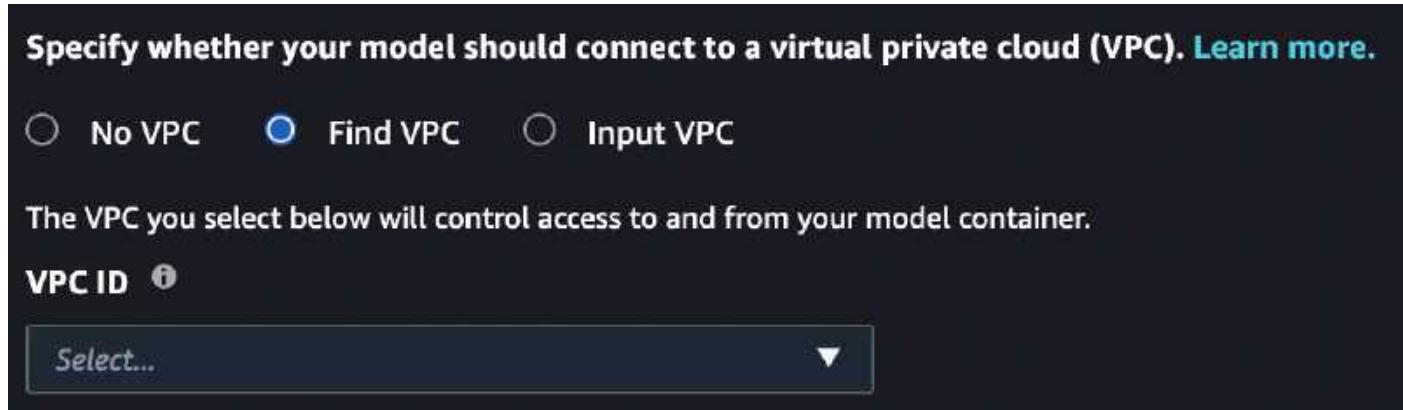
The Amazon VPC that you pass does not need access to the public internet, but it does need access to Amazon S3. The Amazon VPC endpoint for Amazon S3 must allow access to at least the following resources that the model needs.

```
{  
    "Effect": "Allow",  
    "Action": [  
        "s3:GetObject",  
        "s3:PutObject",  
        "s3>ListMultipartUploadParts",  
        "s3>ListBucket"  
    ],  
    "Resources": [  
        "arn:aws:s3:::jumpstart-cache-prod-<region>/*",  
        "arn:aws:s3:::jumpstart-cache-prod-<region>",  
        "arn:aws:s3:::<bucket>/*"  
    ]  
}
```

If you do not select an Amazon VPC, no Amazon VPC is used.

## Find VPC

If you select this option, you must select an existing Amazon VPC from the dropdown list. After you select an Amazon VPC, you must select a subnet and security group for your Amazon VPC. For more information about subnets and security groups, see [Overview of VPCs and subnets](#).



## Input VPC

If you select this option, you must manually select the subnet and security group that compose your Amazon VPC. If your Studio Classic runtime role or Amazon VPC blocks the `ec2:list*` call, you must use this option to select the subnet and security group.

**Specify whether your model should connect to a virtual private cloud (VPC). [Learn more.](#)**

No VPC     Find VPC     Input VPC

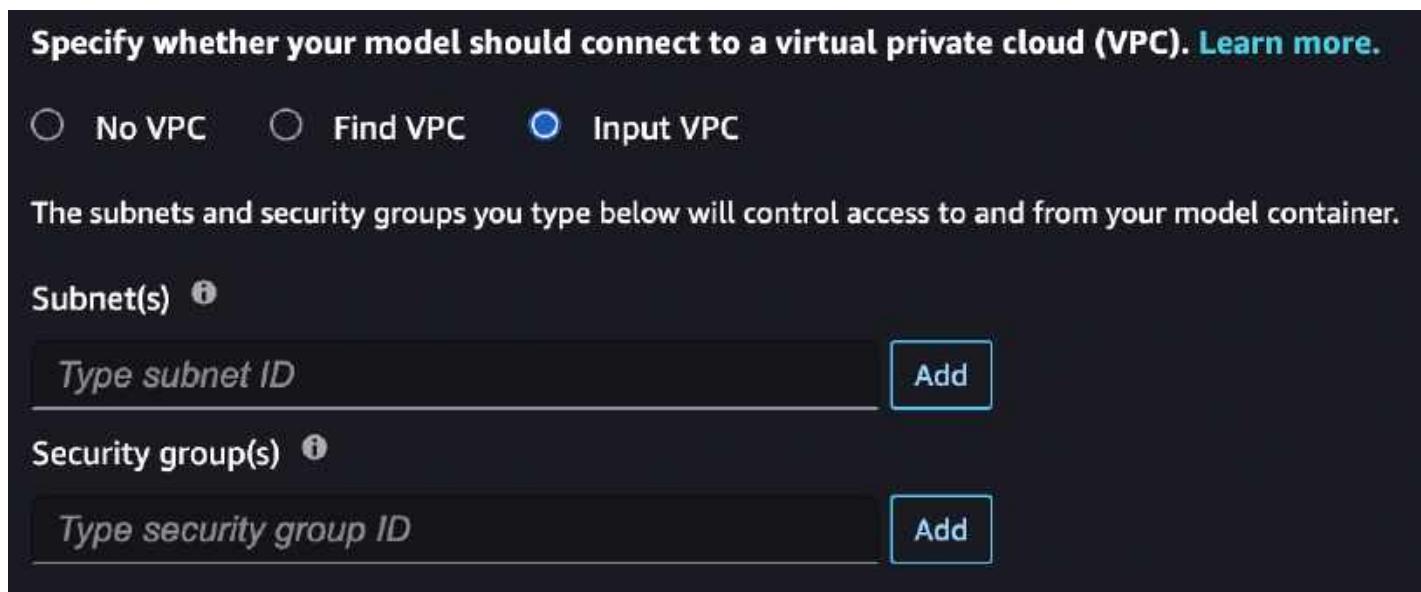
The subnets and security groups you type below will control access to and from your model container.

**Subnet(s) i**

Type *subnet ID* **Add**

**Security group(s) i**

Type *security group ID* **Add**



## Encryption keys

You can select an AWS KMS key that is passed as part of training jobs and hosting jobs. SageMaker AI uses this key to encrypt the Amazon EBS volume for the container, and the repackaged model in Amazon S3 for hosting jobs and the output for training jobs. For more information about AWS KMS keys, see [AWS KMS keys](#).

The key that you pass must trust the IAM role that you pass. If you do not specify an IAM role, the AWS KMS key must trust your Studio Classic runtime role.

If you do not select an AWS KMS key, SageMaker AI provides default encryption for the data in the Amazon EBS volume and the Amazon S3 artifacts.

## Find encryption keys

If you select this option, you must select existing AWS KMS keys from the dropdown list.

**Specify the encryption keys to secure your data. [Learn more.](#)**

Default encryption keys     Find encryption keys     Input encryption keys

Encrypt your data in the storage volume attached to your ML compute instance and at rest in S3.

**Volume encryption key**  ⓘ

Select... ▾

**Model encryption key**  ⓘ

Select... ▾

## Input encryption keys

If you select this option, you must manually enter the AWS KMS keys. If your Studio Classic execution role or Amazon VPC block the `kms:list*` call, you must use this option to select existing AWS KMS keys.

**Specify the encryption keys to secure your data. [Learn more.](#)**

Default encryption keys     Find encryption keys     Input encryption keys

Encrypt your data in the storage volume attached to your ML compute instance and at rest in S3.

**Volume encryption key**  ⓘ

Enter encryption key

**Model encryption key**  ⓘ

Enter encryption key

## Configure default values for JumpStart models

You can configure default values for parameters such as IAM roles, VPCs, and KMS keys to pre-populate for JumpStart model deployment and training. After configuring default values, the Studio Classic UI automatically provides your specified security settings and tags to JumpStart

models to simplify deployment and training workflows. Administrators and end-users can initialize default values specified in a configuration file in YAML format.

By default, the SageMaker Python SDK uses two configuration files: one for the administrator and one for the user. Using the administrator configuration file, administrators can define a set of default values. End-users can override values set in the administrator configuration file and set additional default values using the end-user configuration file. For more information, see [Default configuration file location](#).

The following code sample lists the default locations of the configuration files when using the SageMaker Python SDK in Amazon SageMaker Studio Classic.

```
# Location of the admin config file  
/etc/xdg/sagemaker/config.yaml  
  
# Location of the user config file  
/root/.config/sagemaker/config.yaml
```

Values specified in the user configuration file override values set in the administrator configuration file. The configuration file is unique to each user profile within an Amazon SageMaker AI domain. The user profile's Studio Classic application is directly associated with the user profile. For more information, see [Domain user profiles](#).

Administrators can optionally set configuration defaults for JumpStart model training and deployment through JupyterServer lifecycle configurations. For more information, see [Create and associate a lifecycle configuration](#).

## Default value configuration YAML file

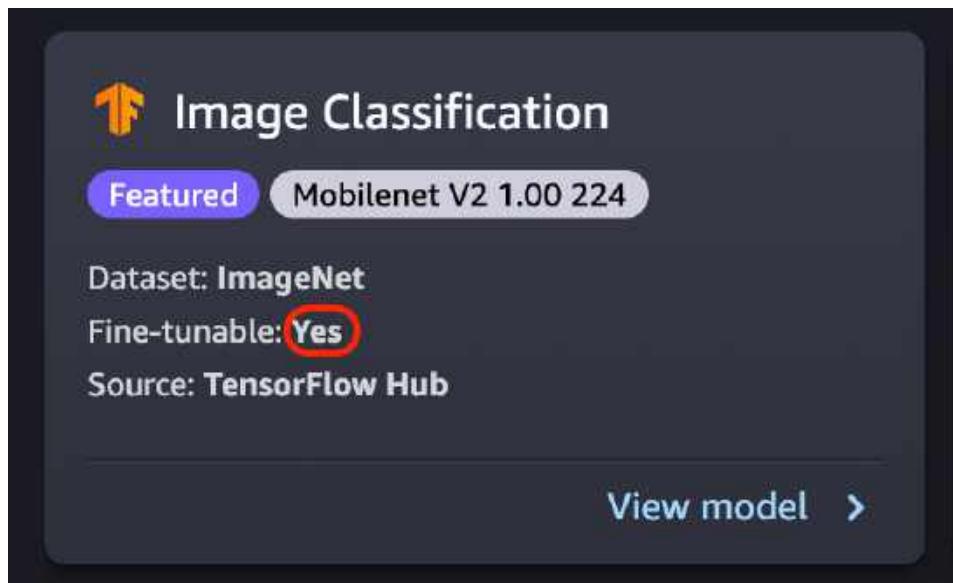
Your configuration file should adhere to the SageMaker Python SDK [configuration file structure](#). Note that specific fields in the TrainingJob, Model, and EndpointConfig configurations apply to JumpStart model training and deployment default values.

```
SchemaVersion: '1.0'  
SageMaker:  
  TrainingJob:  
    OutputDataConfig:  
      KmsKeyId: example-key-id  
    ResourceConfig:  
      # Training configuration - Volume encryption key  
      VolumeKmsKeyId: example-key-id
```

```
# Training configuration form - IAM role
RoleArn: arn:aws:iam::123456789012:role/SageMakerExecutionRole
VpcConfig:
  # Training configuration - Security groups
  SecurityGroupIds:
    - sg-1
    - sg-2
  # Training configuration - Subnets
  Subnets:
    - subnet-1
    - subnet-2
# Training configuration - Custom resource tags
Tags:
- Key: Example-key
  Value: Example-value
Model:
  EnableNetworkIsolation: true
  # Deployment configuration - IAM role
  ExecutionRoleArn: arn:aws:iam::123456789012:role/SageMakerExecutionRole
  VpcConfig:
    # Deployment configuration - Security groups
    SecurityGroupIds:
      - sg-1
      - sg-2
    # Deployment configuration - Subnets
    Subnets:
      - subnet-1
      - subnet-2
EndpointConfig:
  AsyncInferenceConfig:
    OutputConfig:
      KmsKeyId: example-key-id
  DataCaptureConfig:
    # Deployment configuration - Volume encryption key
    KmsKeyId: example-key-id
    KmsKeyId: example-key-id
  # Deployment configuration - Custom resource tags
  Tags:
    - Key: Example-key
      Value: Example-value
```

## Fine-Tune a Model

Fine-tuning trains a pretrained model on a new dataset without training from scratch. This process, also known as transfer learning, can produce accurate models with smaller datasets and less training time. You can fine-tune a model if its card shows a **fine-tunable** attribute set to **Yes**.



### ⚠ Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

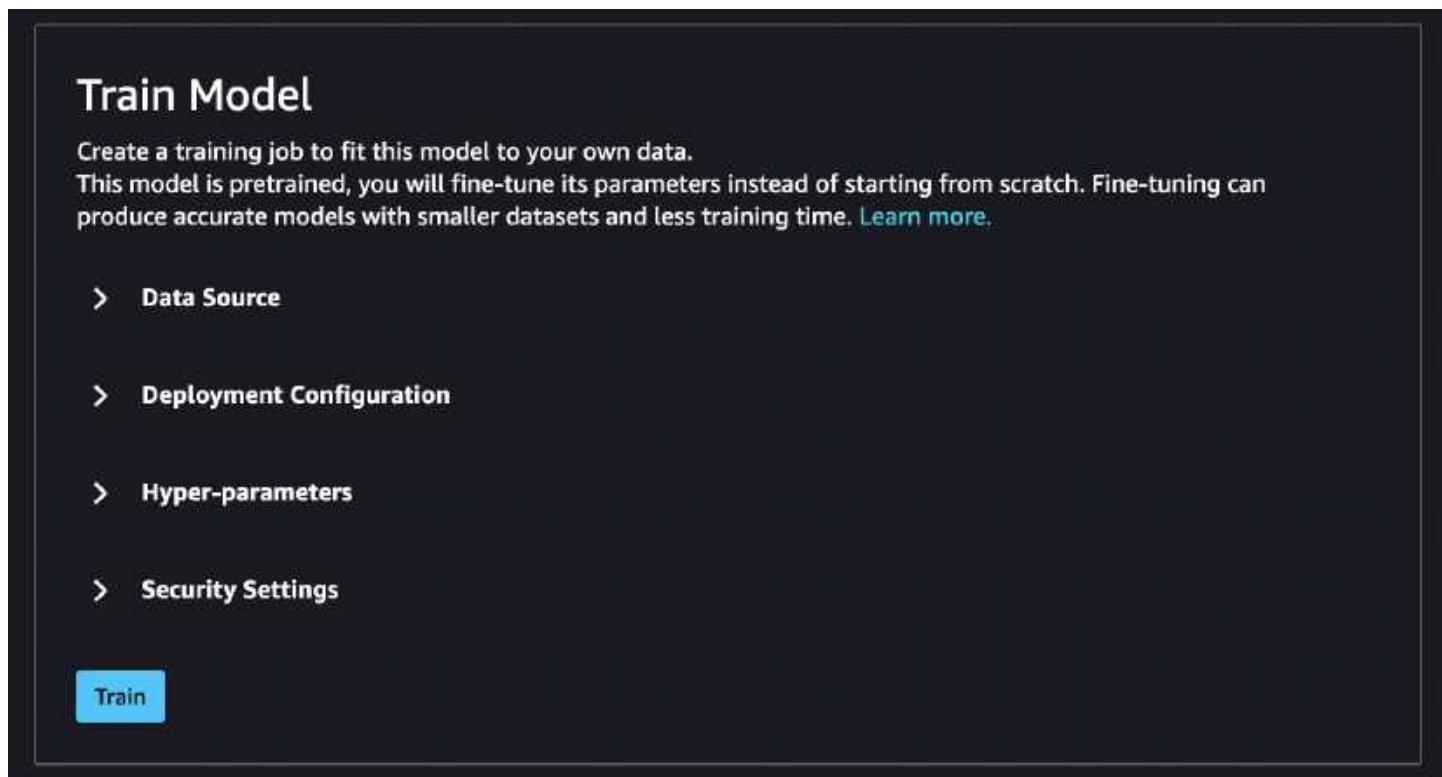
### ℹ Note

For more information on JumpStart model fine-tuning in Studio, see [Fine-tune a model in Studio](#)

## Fine-Tuning data source

When you fine-tune a model, you can use the default dataset or choose your own data, which is located in an Amazon S3 bucket.

To browse the buckets available to you, choose **Find S3 bucket**. These buckets are limited by the permissions used to set up your Studio Classic account. You can also specify an Amazon S3 URI by choosing **Enter Amazon S3 bucket location**.



### Tip

To find out how to format the data in your bucket, choose **Learn more**. The description section for the model has detailed information about inputs and outputs.

For text models:

- The bucket must have a data.csv file.
- The first column must be a unique integer for the class label. For example: 1, 2, 3, 4, n
- The second column must be a string.
- The second column should have the corresponding text that matches the type and language for the model.

For vision models:

- The bucket must have as many subdirectories as the number of classes.
- Each subdirectory should contain images that belong to that class in .jpg format.

### Note

The Amazon S3 bucket must be in the same AWS Region where you're running SageMaker Studio Classic because SageMaker AI doesn't allow cross-Region requests.

## Fine-Tuning deployment configuration

The p3 family is recommended as the fastest for deep learning training, and this is recommended for fine-tuning a model. The following chart shows the number of GPUs in each instance type. There are other available options that you can choose from, including p2 and g4 instance types.

Instance type	GPUs
p3.2xlarge	1
p3.8xlarge	4
p3.16xlarge	8
p3dn.24xlarge	8

## Hyperparameters

You can customize the hyperparameters of the training job that are used to fine-tune the model. The hyperparameters available for each fine-tunable model differ depending on the model. For information on each available hyperparameter, reference the hyperparameters documentation for the model of your choosing in [Built-in algorithms and pretrained models in Amazon SageMaker](#). For example, see [Image Classification - TensorFlow Hyperparameters](#) for details on the fine-tunable Image Classification - TensorFlow hyperparameters.

If you use the default dataset for text models without changing the hyperparameters, you get a nearly identical model as a result. For vision models, the default dataset is different from the dataset used to train the pretrained models, so your model is different as a result.

The following hyperparameters are common among models:

- **Epochs** – One epoch is one cycle through the entire dataset. Multiple intervals complete a batch, and multiple batches eventually complete an epoch. Multiple epochs are run until the accuracy of the model reaches an acceptable level, or when the error rate drops below an acceptable level.
- **Learning rate** – The amount that values should be changed between epochs. As the model is refined, its internal weights are being nudged and error rates are checked to see if the model improves. A typical learning rate is 0.1 or 0.01, where 0.01 is a much smaller adjustment and could cause the training to take a long time to converge, whereas 0.1 is much larger and can cause the training to overshoot. It is one of the primary hyperparameters that you might adjust for training your model. Note that for text models, a much smaller learning rate (5e-5 for BERT) can result in a more accurate model.
- **Batch size** – The number of records from the dataset that is to be selected for each interval to send to the GPUs for training.

In an image example, you might send out 32 images per GPU, so 32 would be your batch size. If you choose an instance type with more than one GPU, the batch is divided by the number of GPUs. Suggested batch size varies depending on the data and the model that you are using. For example, how you optimize for image data differs from how you handle language data.

In the instance type chart in the deployment configuration section, you can see the number of GPUs per instance type. Start with a standard recommended batch size (for example, 32 for a vision model). Then, multiply this by the number of GPUs in the instance type that you selected. For example, if you're using a p3.8xlarge, this would be 32(batch size) multiplied by 4 (GPUs), for a total of 128, as your batch size adjusts for the number of GPUs. For a text model like BERT, try starting with a batch size of 64, and then reduce as needed.

## Training output

When the fine-tuning process is complete, JumpStart provides information about the model: parent model, training job name, training job ARN, training time, and output path. The output path is where you can find your new model in an Amazon S3 bucket. The folder structure uses the model name that you provided and the model file is in an /output subfolder and it's always named `model.tar.gz`.

Example: `s3://bucket/model-name/output/model.tar.gz`

## Configure default values for model training

You can configure default values for parameters such as IAM roles, VPCs, and KMS keys to pre-populate for JumpStart model deployment and training. For more information, see, [Configure default values for JumpStart models](#).

## Share Models

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You can share JumpStart models through the Studio Classic UI directly from the **Launched JumpStart assets** page using the following procedure:

1. Open Amazon SageMaker Studio Classic and choose **Launched JumpStart assets** in the **JumpStart** section of the lefthand navigation pane.
2. Select the **Training jobs** tab to view the list of your model training jobs.
3. Under the **Training jobs** list, select the training job that you want to share. This opens the training job details page. You cannot share more than one training job at a time.
4. In the header for the training job, choose **Share**, and select **Share with my organization**.

For more information about sharing models with your organization, see [Shared Models and Notebooks](#).

## Shared Models and Notebooks

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Share your models and notebooks to centralize model artifacts, facilitate discoverability, and increase the reuse of models within your organization. When sharing your models, you can provide training and inference environment information and allow collaborators to use these environments for their own training and inference jobs.

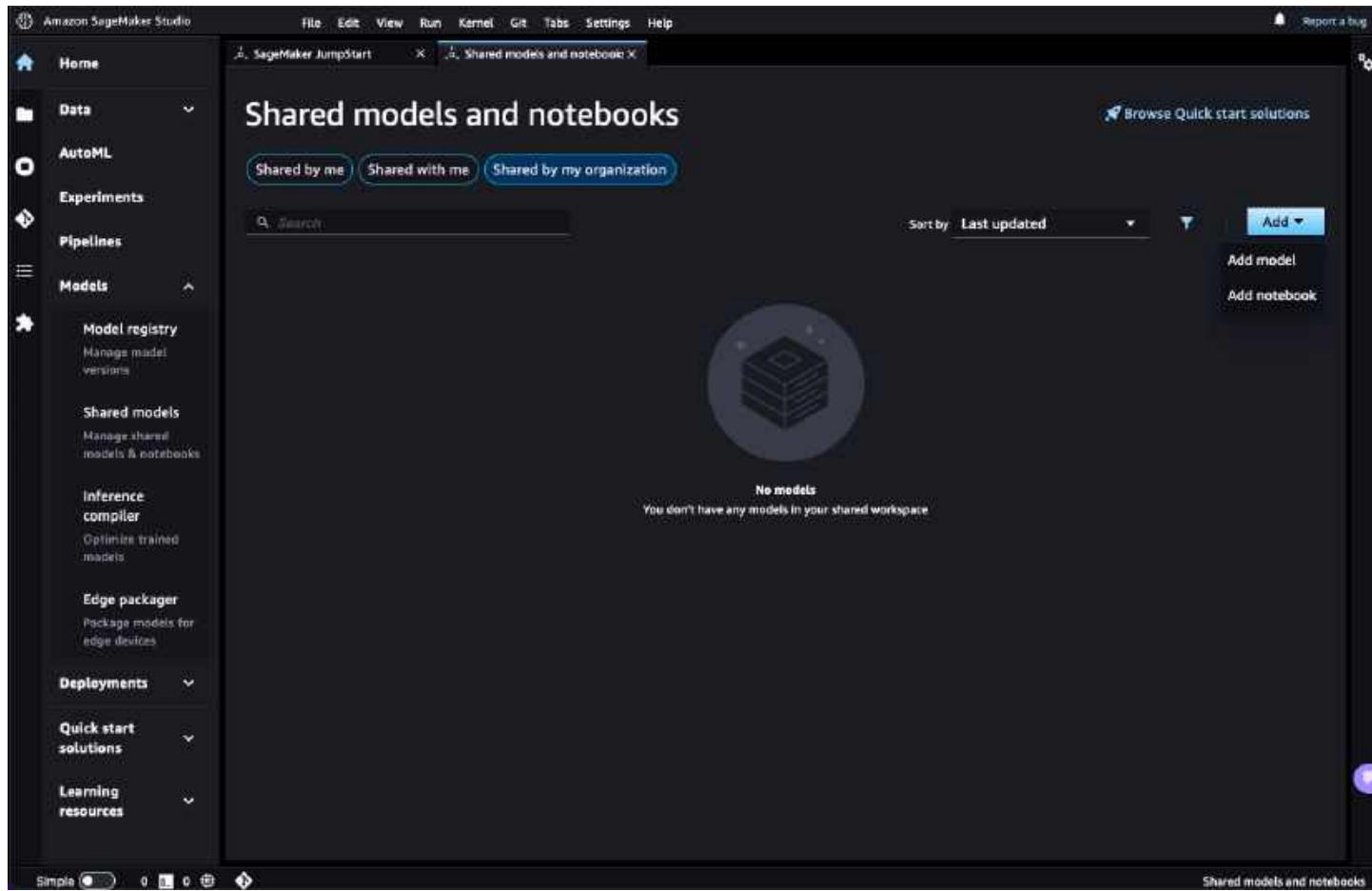
All models that you share and models that are shared with you are searchable in a centralized location directly in Amazon SageMaker Studio Classic. For information on the onboarding steps to sign into Amazon SageMaker Studio Classic, see [Onboard to Amazon SageMaker AI Domain](#).

## Topics

- [Model and notebook sharing](#)
- [Access shared content](#)
- [Add a model](#)

### Model and notebook sharing

To share models and notebooks, navigate to the **Shared models** section in Amazon SageMaker Studio Classic, choose **Shared by my organization**, and then select the **Add** dropdown list. Choose to either add a model or add a notebook.



## Access shared content

From the Amazon SageMaker Studio Classic UI, you can access shared content and filter what you see.

There are three main options for filtering shared models and notebooks:

1. **Shared by me** – Models and notebooks that you shared to JumpStart.
2. **Shared with me** – Models and notebooks shared with you
3. **Shared by my organization** – All models and notebooks that are shared to anyone in your organization

You can also sort your models and notebooks based on the time they were last updated or by ascending or descending alphabetical order. Choose the filter icon



)

to further sort your selections.

## Add a model

To add a model, choose **Shared by my organization**, and then select **Add model** from the **Add** dropdown list. Enter the basic information for your model, and add any training or inference information you want to share with collaborators to train or deploy your model. After you enter all the necessary information, choose **Add model** in the lower right corner of the screen.

### Topics

- [Add basic information](#)
- [Enable training](#)
- [Enable deployment](#)
- [Add a notebook](#)

### Add basic information

Adding a model in JumpStart involves providing some basic information about the model you want to train. This information helps define the characteristics and capabilities of your model, as well as improving its discoverability and searchability. To create a new model, follow these steps:

1. Add a title for this model. Adding a title automatically populates a unique identifier in the ID field based on the model title.
2. Add a description of the model.
3. Select a data type from the options: *text*, *vision*, *tabular*, or *audio*.
4. Select a machine learning task from the list of available tasks, such as *image classification* or *text generation*.
5. Select a machine learning framework.
6. Add metadata information with keywords or phrases to use when searching for a model. Use commas to separate keywords. Any spaces are automatically replaced with commas.

### Enable training

When adding a model to share, you can optionally provide a training environment and allow collaborators in your organization to train the shared model.

**Note**

If you are adding a tabular model, you also need to specify a column format and target column to enable training.

After providing the basic details about your model, you'll need to configure the settings for the training job that will be used to train your model. This involves specifying the container environment, code scripts, datasets, output locations, and various other parameters to control how the training job is executed. To configure the training job settings, follow these steps:

1. Add a container to use for model training. You can select a container used for an existing training job, bring your own container in Amazon ECR, or use an Amazon SageMaker Deep Learning Container.
2. Add environment variables.
3. Provide a training script location.
4. Provide a script mode entry point.
5. Provide an Amazon S3 URI for model artifacts generated during training.
6. Provide the Amazon S3 URI to the default training dataset.
7. Provide a model output path. The model output path should be the Amazon S3 URI path for any model artifacts generated from training. SageMaker AI saves the model artifacts as a single compressed TAR file in Amazon S3.
8. Provide a validation dataset to use for evaluating your model during training. Validation datasets must contain the same number of columns and the same feature headers as the training dataset.
9. Turn on network isolation. Network isolation isolates the model container so that no inbound or outbound network calls can be made to or from the model container.
10. Provide training channels through which SageMaker AI can access your data. For example, you might specify input channels named `train` or `test`. For each channel, specify a channel name and a URI to the location of your data. Choose **Browse** to search for Amazon S3 locations.
11. Provide hyperparameters. Add any hyperparameters with which collaborators should experiment during training. Provide a range of valid values for these hyperparameters. This range is used for training job hyperparameter validation. You can define ranges based on the datatype of the hyperparameter.

12. Select an instance type. We recommend a GPU instance with more memory for training with large batch sizes. For a comprehensive list of SageMaker training instances across AWS Regions, see the **On-Demand Pricing** table in [Amazon SageMaker Pricing](#).
13. Provide metrics. Define metrics for a training job by specifying a name and a regular expression for each metric that your training monitors. Design the regular expressions to capture the values of metrics that your algorithm emits. For example, the metric `loss` might have the regular expression `"Loss =(. *?);"`.

## Enable deployment

When adding a model to share, you can optionally provide an inference environment in which collaborators in your organization can deploy the shared model for inference.

After training your machine learning model, you'll need to deploy it to an Amazon SageMaker AI endpoint for inference. This involves providing a container environment, an inference script, the model artifacts generated during training, and selecting an appropriate compute instance type. Configuring these settings properly is crucial for ensuring your deployed model can make accurate predictions and handle inference requests efficiently. To set up your model for inference, follow these steps:

1. Add a container to use for inference. You can bring your own container in Amazon ECR or use an Amazon SageMaker Deep Learning Container.
2. Provide the Amazon S3 URI to an inference script. Custom inference scripts run inside your chosen container. Your inference script should include a function for model loading, and optionally functions generating predictions, and input and output processing. For more information on creating inference scripts for the framework of your choice, see [Frameworks](#) in the SageMaker Python SDK documentation. For example, for TensorFlow, see [How to implement the pre- and/or post-processing handler\(s\)](#).
3. Provide an Amazon S3 URI for model artifacts. Model artifacts are the output that results from training a model, and typically consist of trained parameters, a model definition that describes how to compute inferences, and other metadata. If you trained your model in SageMaker AI, the model artifacts are saved as a single compressed TAR file in Amazon S3. If you trained your model outside SageMaker AI, you need to create this single compressed TAR file and save it in an Amazon S3 location.
4. Select an instance type. We recommend a GPU instance with more memory for training with large batch sizes. For a comprehensive list of SageMaker training instances across AWS Regions, see the **On-Demand Pricing** table in [Amazon SageMaker Pricing](#).

## Add a notebook

To add a notebook, choose **Shared by my organization**, and then select **Add notebook** from the **Add** dropdown list. Enter the basic information for your notebook and provide an Amazon S3 URI for the location of that notebook.

First, add the basic descriptive information about your notebook. This information is used to improve the searchability of your notebook.

1. Add a title for this notebook. Adding a title automatically populates a unique identifier in the ID field based on the notebook title.
2. Add a description of the notebook.
3. Select a data type from the options: *text*, *vision*, *tabular*, or *audio*.
4. Select an ML task from the list of available tasks, such as *image classification* or *text generation*.
5. Select an ML framework.
6. Add metadata information with keywords or phrases to use when searching for a notebook. Use commas to separate keywords. Any spaces are automatically replaced with commas.

After you've specified the basic information, you can provide an Amazon S3 URI for the location of that notebook. You can choose **Browse** to search through your Amazon S3 buckets for your notebook file location. After you find your notebook, copy the Amazon S3 URI, choose **Cancel**, and then add the Amazon S3 URI to the **Notebook Location** field.

After you enter all the necessary information, choose **Add notebook** in the lower right corner.

## End-to-end JumpStart solution templates

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

 **Note**

JumpStart Solutions are only available in Studio Classic.

SageMaker JumpStart provides one-click, end-to-end solutions that are designed to address common machine learning use cases. They use proven algorithms for their domains and provide a complete workflow which typically includes data processing, model training, deployment, inference, and monitoring. Explore the following use cases for more information on available solution templates.

- [Demand forecasting](#)
- [Credit rating prediction](#)
- [Fraud detection](#)
- [Computer vision](#)
- [Extract and analyze data from documents](#)
- [Predictive maintenance](#)
- [Churn prediction](#)
- [Personalized recommendations](#)
- [Reinforcement learning](#)
- [Healthcare and life sciences](#)
- [Financial pricing](#)
- [Causal inference](#)

Choose the solution template that best fits your use case from the JumpStart landing page. When you choose a solution template, JumpStart opens a new tab showing a description of the solution and a **Launch** button. When you select **Launch**, JumpStart creates all of the resources that you need to run the solution, including training and model hosting instances. For more information on launching a JumpStart solution, see [the section called “Launch a Solution”](#).

After launching the solution, you can explore solution features and any generated artifacts in JumpStart. Use the **Launched JumpStart assets** menu to find your solution. In your solution's tab, select **Open Notebook** to use provided notebooks and explore the solution's features. When artifacts are generated during launch or after running the provided notebooks, they're

listed in the **Generated Artifacts** table. You can delete individual artifacts with the trash icon



).

You can delete all of the solution's resources by choosing **Delete solution resources**.

## Demand forecasting

Demand forecasting uses historical time series data in order to make future estimations in relation to customer demand over a specific period and streamline the supply-demand decision-making process across businesses.

Demand forecasting use cases include predicting ticket sales in the transportation industry, stock prices, number of hospital visits, number of customer representatives to hire for multiple locations in the next month, product sales across multiple regions in the next quarter, cloud server usage for the next day for a video streaming service, electricity consumption for multiple regions over the next week, number of IoT devices and sensors such as energy consumption, and more.

Time series data is categorized as *univariate* and *multi-variate*. For example, the total electricity consumption for a single household is a univariate time series over a period of time. When multiple univariate time series are stacked on each other, it's called a multi-variate time series. For example, the total electricity consumption of 10 different (but correlated) households in a single neighborhood make up a multi-variate time series dataset.

Solution name	Description	Get started
Demand forecasting	Demand forecasting for multivariate time series data using three state-of-the-art time series forecasting algorithms: <a href="#">LSTNet</a> , <a href="#">Prophet</a> , and <a href="#">SageMaker AI DeepAR</a> .	<a href="#">GitHub »</a>

## Credit rating prediction

Use JumpStart's credit rating prediction solutions to predict corporate credit ratings or to explain credit prediction decisions made by machine learning models. Compared to traditional credit rating modeling methods, machine learning models can automate and improve the accuracy of credit prediction.

Solution name	Description	Get started
Corporate credit rating prediction	Multimodal (long text and tabular) machine learning for quality credit predictions using AWS <a href="#">AutoGluon Tabular</a> .	<a href="#">GitHub »</a>
Graph-based credit scoring	Predict corporate credit ratings using tabular data and a corporate network by training a <a href="#">Graph Neural Network GraphSAGE</a> and AWS <a href="#">AutoGluon Tabular</a> model.	Find in Amazon SageMaker Studio Classic.
Explain credit decisions	Predict credit default in credit applications and provide explanations using <a href="#">LightGBM</a> and <a href="#">SHAP (SHapley Additive exPlanations)</a> .	<a href="#">GitHub »</a>

## Fraud detection

Many businesses lose billions annually to fraud. Machine learning based fraud detection models can help systematically identify likely fraudulent activities from a tremendous amount of data. The following solutions use transaction and user identity datasets to identify fraudulent transactions.

Solution name	Description	Get started
Detect malicious users and transactions	Automatically detect potentially fraudulent activity in transactions using <a href="#">SageMaker AI XGBoost</a> with the over-sampling technique <a href="#">Synthetic Minority Over-sampling (SMOTE)</a> .	<a href="#">GitHub »</a>

Solution name	Description	Get started
Fraud detection in financial transactions using deep graph library	Detect fraud in financial transactions by training a <a href="#">graph convolutional network</a> with the <a href="#">deep graph library</a> and a <a href="#">SageMaker AI XGBoost</a> model.	<a href="#">GitHub »</a>
Financial payment classification	Classify financial payments based on transaction information using <a href="#">SageMaker AI XGBoost</a> . Use this solution template as an intermediate step in fraud detection, personalization, or anomaly detection.	Find in Amazon SageMaker Studio Classic.

## Computer vision

With the rise of business use cases such as autonomous vehicles, smart video surveillance, healthcare monitoring and various object counting tasks, fast and accurate object detection systems are rising in demand. These systems involve not only recognizing and classifying every object in an image, but localizing each one by drawing the appropriate bounding box around it. In the last decade, the rapid advances of deep learning techniques greatly accelerated the momentum of object detection.

Solution name	Description	Get started
Visual product defect detection	Identify defective regions in product images either by training an <a href="#">object detection model from scratch</a> or fine-tuning pretrained SageMaker AI models.	<a href="#">GitHub »</a>

Solution name	Description	Get started
Handwriting recognition	Recognize handwritten text in images by training an <a href="#">object detection model</a> and <a href="#">handwriting recognition model</a> . Label your own data using <a href="#">SageMaker Ground Truth</a> .	<a href="#">GitHub »</a>
Object detection for bird species	Identify birds species in a scene using a <a href="#">SageMaker AI object detection model</a> .	Find in Amazon SageMaker Studio Classic.

## Extract and analyze data from documents

JumpStart provides solutions for you to uncover valuable insights and connections in business-critical documents. Use cases include text classification, document summarization, handwriting recognition, relationship extraction, question and answering, and filling in missing values in tabular records.

Solution name	Description	Get started
Privacy for sentiment classification	<a href="#">Anonymize text</a> to better preserve user privacy in sentiment classification.	<a href="#">GitHub »</a>
Document understanding	Document summarization, entity, and relations hip extraction using the <a href="#">transformers</a> library in PyTorch.	<a href="#">GitHub »</a>
Handwriting recognition	Recognize handwritten text in images by training an <a href="#">object detection model</a> and <a href="#">handwriting recognition</a>	<a href="#">GitHub »</a>

Solution name	Description	Get started
	<p>model. Label your own data using <a href="#">SageMaker Ground Truth</a>.</p>	
Filling in missing values in tabular records	Fill missing values in tabular records by training a <a href="#">SageMaker Autopilot</a> model.	<a href="#">GitHub »</a>

## Predictive maintenance

Predictive maintenance aims to optimize the balance between corrective and preventative maintenance by facilitating the timely replacement of components. The following solutions use sensor data from industrial assets to predict machine failures, unplanned downtime, and repair costs.

Solution name	Description	Get started
Predictive maintenance for vehicle fleets	Predict vehicle fleet failures using vehicle sensor and maintenance information with a convolutional neural network model.	<a href="#">GitHub »</a>
Predictive maintenance for manufacturing	Predict the remaining useful life for each sensor by training a <a href="#">stacked Bidirectional LSTM neural network</a> model using historical sensor readings.	<a href="#">GitHub »</a>

## Churn prediction

Customer churn, or rate of attrition, is a costly problem faced by a wide range of companies. In an effort to reduce churn, companies can identify customers that are likely to leave their service in order to focus their efforts on customer retention. Use a JumpStart churn prediction solution to

analyze data sources such as user behavior and customer support chat logs to identify customers that are at a high risk of cancelling a subscription or service.

Solution name	Description	Get started
Churn prediction with text	Predict churn using numerical, categorical, and textual features with <a href="#">BERT encoder</a> and <a href="#">RandomForestClassifier</a> .	<a href="#">GitHub »</a>
Churn prediction for mobile phone customers	Identify unhappy mobile phone customers using <a href="#">SageMaker AI XGBoost</a> .	Find in Amazon SageMaker Studio Classic.

## Personalized recommendations

You can use JumpStart solutions to analyze customer identity graphs or user sessions to better understand and predict customer behavior. Use the following solutions for personalized recommendations to model customer identity across multiple devices, to determine the likelihood of a customer making a purchase, or to create a custom movie recommender based on past customer behavior.

Solution name	Description	Get started
Entity resolution in identity graphs with deep graph library	Perform cross-device entity linking for online advertising by training a <a href="#">graph convolutional network</a> with <a href="#">deep graph library</a> .	<a href="#">GitHub »</a>
Purchase modeling	Predict whether a customer will make a purchase by training a <a href="#">SageMaker AI XGBoost</a> model.	<a href="#">GitHub »</a>
Customized recommender system	Train and deploy a custom recommender system that	Find in Amazon SageMaker Studio Classic.

Solution name	Description	Get started
	generates movie suggestions for a customer based on past behavior using Neural Collaborative Filtering in SageMaker AI.	

## Reinforcement learning

Reinforcement learning (RL) is a type of learning that is based on interaction with the environment. This type of learning is used by an agent that must learn behavior through trial-and-error interactions with a dynamic environment in which the goal is to maximize the long-term rewards that the agent receives as a result of its actions. Rewards are maximized by trading off exploring actions that have uncertain rewards with exploiting actions that have known rewards.

RL is well-suited for solving large, complex problems, such as supply chain management, HVAC systems, industrial robotics, game artificial intelligence, dialog systems, and autonomous vehicles.

Solution name	Description	Get started
Reinforcement learning for Battlesnake AI competitions	Provide a reinforcement learning workflow for training and inference with the <a href="#">BattleSnake</a> AI competitions.	<a href="#">GitHub »</a>
Distributed reinforcement learning for Progen challenge	Distributed reinforcement learning starter kit for <a href="#">NeurIPS 2020 Progen</a> Reinforcement learning challenge.	<a href="#">GitHub »</a>

## Healthcare and life sciences

Clinicians and researchers can use JumpStart solutions to analyze medical imagery, genomic information, and clinical health records.

Solution name	Description	Get started
Lung cancer survival prediction	Predict non-small cell lung cancer patient survival status with 3-dimensional lung computerized tomography (CT) scans, genomic data, and clinical health records using <a href="#">SageMaker AI XGBoost</a> .	<a href="#">GitHub »</a>

## Financial pricing

Many businesses dynamically adjust pricing on a regular basis in order to maximize their returns. Use the following JumpStart solutions for price optimization, dynamic pricing, option pricing, or portfolio optimization use cases.

Solution name	Description	Get started
Price optimization	Estimate price elasticity using Double Machine Learning (ML) for causal inference and the <a href="#">Prophet</a> forecasting procedure. Use these estimates to optimize daily prices.	Find in Amazon SageMaker Studio Classic.

## Causal inference

Researchers can use machine learning models such as Bayesian networks to represent causal dependencies and draw causal conclusions based on data. Use the following JumpStart solution to understand the causal relationship between Nitrogen-based fertilizer application and corn crop yields.

Solution name	Description	Get started
Crop yield counterfactuals	Generate a counterfactual analysis of corn response to nitrogen. This solution learns the crop phenology cycle in its entirety using multi-spectral satellite imagery and <a href="#">ground-level observations</a> .	Find in Amazon SageMaker Studio Classic.

## Launch a Solution

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

### Note

JumpStart Solutions are only available in Studio Classic.

First, choose a solution through the SageMaker JumpStart landing page in the Amazon SageMaker Studio Classic UI. For information on the onboarding steps to sign in to Amazon SageMaker Studio Classic, see [Onboard to Amazon SageMaker AI domain](#). For details on getting to the SageMaker JumpStart landing page, see [Open and use JumpStart in Studio Classic](#).

After you choose a solution, a solution's tab opens showing a description of the solution and a Launch button. To launch a solution, select Launch in the **Launch Solution** section. JumpStart then creates all of the resources needed to run the solution. This includes training and model hosting instances.

## Advanced parameters

The solution that you choose may have advanced parameters that you can select. Choose **Advanced Parameters** to specify the AWS Identity and Access Management role for the solution.

Solutions are able to launch resources across 9 AWS services that interact with each other. For the solution to work as expected, newly created components from one service must be able to act on newly created components from another service. We recommend that you use the default IAM role to ensure that all needed permissions are added. For more information about IAM roles, see [AWS Identity and Access Management for Amazon SageMaker AI](#).

### Default IAM role

If you select this option, the default IAM roles that are required by this solution are used. Each solution requires different resources. The following list describes the default roles that are used for the solutions based on the service needed. For a description of the permissions required for each service, see [AWS Managed Policies for SageMaker Projects and JumpStart](#).

- **API Gateway** – AmazonSageMakerServiceCatalogProductsApiGatewayRole
- **CloudFormation** – AmazonSageMakerServiceCatalogProductsCloudformationRole
- **CodeBuild** – AmazonSageMakerServiceCatalogProductsCodeBuildRole
- **CodePipeline** – AmazonSageMakerServiceCatalogProductsCodePipelineRole
- **Events** – AmazonSageMakerServiceCatalogProductsEventsRole
- **Firehose** – AmazonSageMakerServiceCatalogProductsFirehoseRole
- **Glue** – AmazonSageMakerServiceCatalogProductsGlueRole
- **Lambda** – AmazonSageMakerServiceCatalogProductsLambdaRole
- **SageMaker AI** – AmazonSageMakerServiceCatalogProductsExecutionRole

If you are using a new SageMaker AI domain with JumpStart project templates enabled, these roles are automatically created in your account.

If you are using an existing SageMaker AI domain, these roles may not exist in your account. If this is the case, you will receive the following error when launching the solution.

Unable to locate the updated roles required to launch this solution, a general role '/service-role/AmazonSageMakerServiceCatalogProductsUserRole' will be used. Please update your studio domain to generate these roles.

You can still launch a solution without the needed role, but the legacy default role `AmazonSageMakerServiceCatalogProductsUserRole` is used in place of the needed role. The legacy default role has trust relationships with all of the services that JumpStart solutions need to interact with. For the best security, we recommend that you update your domain to have the newly created default roles for each AWS service.

If you have already onboarded to a SageMaker AI domain, you can update your domain to generate the default roles using the following procedure.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Choose **Control Panel** at the top left of the page.
3. From the **domain** page, choose the **Settings** icon  ) to edit the domain settings.
4. On **General Settings** choose **Next**.
5. Under **SageMaker Projects and JumpStart**, select **Enable Amazon SageMaker project templates and Amazon SageMaker JumpStart for this account** and **Enable Amazon SageMaker project templates and Amazon SageMaker JumpStart for Studio Classic users**, choose **Next**.
6. Select **Submit**.

You should be able to see the default roles listed in **Projects - Amazon SageMaker project templates enabled for this account** under the **Apps - Studio** tab.

## Find IAM role

If you select this option, you must select an existing IAM role from the dropdown list for each of the required services. The selected role must have at least the minimum permissions required for the corresponding service. For a description of the permissions required for each service, see [AWS Managed Policies for SageMaker Projects and JumpStart](#).

## Input IAM role

If you select this option, you must manually enter the ARN for an existing IAM role. The selected role must have at least the minimum permissions required for the corresponding service. For a description of the permissions required for each service, see [AWS Managed Policies for SageMaker Projects and JumpStart](#).

## Amazon SageMaker JumpStart Industry: Financial

Use SageMaker JumpStart Industry: Financial solutions, models, and example notebooks to learn about SageMaker AI features and capabilities through curated one-step solutions and example notebooks of industry-focused machine learning (ML) problems. The notebooks also walk through how to use the SageMaker JumpStart Industry Python SDK to enhance industry text data and fine-tune pretrained models.

### Topics

- [Amazon SageMaker JumpStart Industry Python SDK](#)
- [Amazon SageMaker JumpStart Industry: Financial Solution](#)
- [Amazon SageMaker JumpStart Industry: Financial Models](#)
- [Amazon SageMaker JumpStart Industry: Financial Example Notebooks](#)
- [Amazon SageMaker JumpStart Industry: Financial Blog Posts](#)
- [Amazon SageMaker JumpStart Industry: Financial Related Research](#)
- [Amazon SageMaker JumpStart Industry: Financial Additional Resources](#)

### Amazon SageMaker JumpStart Industry Python SDK

SageMaker Runtime JumpStart provides processing tools for curating industry datasets and fine-tuning pretrained models through its client library called SageMaker JumpStart Industry Python SDK. For detailed API documentation of the SDK, and to learn more about processing and enhancing industry text datasets for improving the performance of state-of-the-art models on SageMaker JumpStart, see the [SageMaker JumpStart Industry Python SDK open source documentation](#).

### Amazon SageMaker JumpStart Industry: Financial Solution

SageMaker JumpStart Industry: Financial provides the following solution notebooks:

- **Corporate Credit Rating Prediction**

This SageMaker JumpStart Industry: Financial solution provides a template for a text-enhanced corporate credit rating model. It shows how to take a model based on numeric features (in this case, Altman's famous 5 financial ratios) combined with texts from SEC filings to achieve an improvement in the prediction of credit ratings. In addition to the 5 Altman ratios, you can add more variables as needed or set custom variables. This solution notebook shows how SageMaker

JumpStart Industry Python SDK helps process Natural Language Processing (NLP) scoring of texts from SEC filings. Furthermore, the solution demonstrates how to train a model using the enhanced dataset to achieve a best-in-class model, deploy the model to a SageMaker AI endpoint for production, and receive improved predictions in real time.

- **Graph-Based Credit Scoring**

Credit ratings are traditionally generated using models that use financial statement data and market data, which is tabular only (numeric and categorical). This solution constructs a network of firms using [SEC filings](#) and shows how to use the network of firm relationships with tabular data to generate accurate rating predictions. This solution demonstrates a methodology to use data on firm linkages to extend the traditionally tabular-based credit scoring models, which have been used by the ratings industry for decades, to the class of machine learning models on networks.

 **Note**

The solution notebooks are for demonstration purposes only. They should not be relied on as financial or investment advice.

You can find these financial services solutions through the SageMaker JumpStart page in Studio Classic.

 **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

 **Note**

The SageMaker JumpStart Industry: Financial solutions, model cards, and example notebooks are hosted and runnable only through SageMaker Studio Classic. Log in to the [SageMaker AI console](#), and launch SageMaker Studio Classic. For more information about how to find the solution card, see the previous topic at [SageMaker JumpStart](#).

## Amazon SageMaker JumpStart Industry: Financial Models

SageMaker JumpStart Industry: Financial provides the following pretrained [Robustly Optimized BERT approach \(RoBERTa\)](#) models:

- **Financial Text Embedding (RoBERTa-SEC-Base)**
- **RoBERTa-SEC-WIKI-Base**
- **RoBERTa-SEC-Large**
- **RoBERTa-SEC-WIKI-Large**

The RoBERTa-SEC-Base and RoBERTa-SEC-Large models are the text embedding models based on [GluonNLP's RoBERTa model](#) and pretrained on S&P 500 SEC 10-K/10-Q reports of the decade of the 2010's (from 2010 to 2019). In addition to these, SageMaker AI JumpStart Industry: Financial provides two more RoBERTa variations, RoBERTa-SEC-WIKI-Base and RoBERTa-SEC-WIKI-Large, which are pretrained on the SEC filings and common texts of Wikipedia.

You can find these models in SageMaker JumpStart by navigating to the **Text Models** node, choosing **Explore All Text Models**, and then filtering for the ML Task **Text Embedding**. You can access any corresponding notebooks after selecting the model of your choice. The paired notebooks will walk you through how the pretrained models can be fine-tuned for specific classification tasks on multimodal datasets, which are enhanced by the SageMaker JumpStart Industry Python SDK.

 **Note**

The model notebooks are for demonstration purposes only. They should not be relied on as financial or investment advice.

The following screenshot shows the pretrained model cards provided through the SageMaker AI JumpStart page on Studio Classic.

**m Financial Text Embedding**

Featured Roberta-Sec-Base

Pre-training Dataset: S&P 500 10-K/10-Q (2010-2019)

Fine-tunable: No

Source: Gluon NLP

[View model >](#)

**m RoBERTa-SEC-WIKI-Base**

Text Embedding

Pre-training Dataset: S&P 500 10-K/10-Q (2010-2019)

Fine-tunable: No

Source: Gluon NLP

[View model >](#)

**m RoBERTa-SEC-Large**

Text Embedding

Pre-training Dataset: S&P 500 10-K/10-Q (2010-2019)

Fine-tunable: No

Source: Gluon NLP

[View model >](#)

**m RoBERTa-SEC-WIKI-Large**

Text Embedding

Pre-training Dataset: S&P 500 10-K/10-Q (2010-2019)

Fine-tunable: No

Source: Gluon NLP

[View model >](#)

**i Note**

The SageMaker JumpStart Industry: Financial solutions, model cards, and example notebooks are hosted and runnable only through SageMaker Studio Classic. Log in to the [SageMaker AI console](#), and launch SageMaker Studio Classic. For more information about how to find the model cards, see the previous topic at [SageMaker JumpStart](#).

## Amazon SageMaker JumpStart Industry: Financial Example Notebooks

SageMaker JumpStart Industry: Financial provides the following example notebooks to demonstrate solutions to industry-focused ML problems:

- **Financial TabText Data Construction** – This example introduces how to use the SageMaker JumpStart Industry Python SDK for processing the SEC filings, such as text summarization and scoring texts based on NLP score types and their corresponding word lists. To preview the content of this notebook, see [Simple Construction of a Multimodal Dataset from SEC Filings and NLP Scores](#).

- **Multimodal ML on TabText Data** – This example shows how to merge different types of datasets into a single dataframe called TabText and perform multimodal ML. To preview the content of this notebook, see [Machine Learning on a TabText Dataframe – An Example Based on the Paycheck Protection Program](#).
- **Multi-category ML on SEC filings data** – This example shows how to train an AutoGluon NLP model over the multimodal (TabText) datasets curated from SEC filings for a multiclass classification task. [Classify SEC 10K/Q Filings to Industry Codes Based on the MDNA Text Column](#).

 **Note**

The example notebooks are for demonstrative purposes only. They should not be relied on as financial or investment advice.

 **Note**

The SageMaker JumpStart Industry: Financial solutions, model cards, and example notebooks are hosted and runnable only through SageMaker Studio Classic. Log in to the [SageMaker AI console](#), and launch SageMaker Studio Classic. For more information about how to find the example notebooks, see the previous topic at [SageMaker JumpStart](#).

To preview the content of the example notebooks, see [Tutorials – Finance](#) in the *SageMaker JumpStart Industry Python SDK documentation*.

## Amazon SageMaker JumpStart Industry: Financial Blog Posts

For thorough applications of using SageMaker JumpStart Industry: Financial solutions, models, examples, and the SDK, see the following blog posts:

- [Use pre-trained financial language models for transfer learning in Amazon SageMaker JumpStart](#)
- [Use SEC text for ratings classification using multimodal ML in Amazon SageMaker JumpStart](#)
- [Create a dashboard with SEC text for financial NLP in Amazon SageMaker JumpStart](#)
- [Build a corporate credit ratings classifier using graph machine learning in Amazon SageMaker JumpStart](#)

- [Domain-adaptation Fine-tuning of Foundation Models in Amazon SageMaker JumpStart on Financial data](#)

## Amazon SageMaker JumpStart Industry: Financial Related Research

For research related to SageMaker JumpStart Industry: Financial solutions, see the following papers:

- [Context, Language Modeling, and Multimodal Data in Finance](#)
- [Multimodal Machine Learning for Credit Modeling](#)
- [On the Lack of Robust Interpretability of Neural Text Classifiers](#)
- [FinLex: An Effective Use of Word Embeddings for Financial Lexicon Generation](#)

## Amazon SageMaker JumpStart Industry: Financial Additional Resources

For additional documentation and tutorials, see the following resources:

- [The SageMaker JumpStart Industry: Financial Python SDK](#)
- [SageMaker JumpStart Industry: Financial Python SDK Tutorials](#)
- [The SageMaker JumpStart Industry: Financial GitHub repository](#)
- [Getting started with Amazon SageMaker AI - Machine Learning Tutorials](#)

# Machine learning environments offered by Amazon SageMaker AI

## Important

Amazon SageMaker Studio and Amazon SageMaker Studio Classic are two of the machine learning environments that you can use to interact with SageMaker AI.

If your domain was created after November 30, 2023, Studio is your default experience.

If your domain was created before November 30, 2023, Amazon SageMaker Studio Classic is your default experience. To use Studio if Amazon SageMaker Studio Classic is your default experience, see [Migration from Amazon SageMaker Studio Classic](#).

When you migrate from Amazon SageMaker Studio Classic to Amazon SageMaker Studio, there is no loss in feature availability. Studio Classic also exists as an IDE within Amazon SageMaker Studio to help you run your legacy machine learning workflows.

SageMaker AI supports the following machine learning environments:

- *Amazon SageMaker Studio* (Recommended): The latest web-based experience for running ML workflows with a suite of IDEs. Studio supports the following applications:
  - Amazon SageMaker Studio Classic
  - Code Editor, based on Code-OSS, Visual Studio Code - Open Source
  - JupyterLab
  - Amazon SageMaker Canvas
  - RStudio
- *Amazon SageMaker Studio Classic*: Lets you build, train, debug, deploy, and monitor your machine learning models.
- *Amazon SageMaker Notebook Instances*: Lets you prepare and process data, and train and deploy machine learning models from a compute instance running the Jupyter Notebook application.
- *Amazon SageMaker Studio Lab*: Studio Lab is a free service that gives you access to AWS compute resources, in an environment based on open-source JupyterLab, without requiring an AWS account.
- *Amazon SageMaker Canvas*: Gives you the ability to use machine learning to generate predictions without needing to code.

- *Amazon SageMaker geospatial*: Gives you the ability to build, train, and deploy geospatial models.
- *RStudio on Amazon SageMaker AI*: RStudio is an IDE for [R](#), with a console, syntax-highlighting editor that supports direct code execution, and tools for plotting, history, debugging and workspace management.
- *SageMaker HyperPod*: SageMaker HyperPod lets you provision resilient clusters for running machine learning (ML) workloads and developing state-of-the-art models such as large language models (LLMs), diffusion models, and foundation models (FMs).

To use these machine learning environments, you or your organization's administrator must create an Amazon SageMaker AI domain. The exceptions are Studio Lab, SageMaker Notebook Instances, and SageMaker HyperPod.

Instead of manually provisioning resources and managing permissions for yourself and your users, you can create a Amazon DataZone domain. The process of creating a Amazon DataZone domain creates a corresponding Amazon SageMaker AI domain with AWS Glue or Amazon Redshift databases for your ETL workflows. Setting up a domain through Amazon DataZone reduces the amount of time it takes to set up SageMaker AI environments for your users. For more information about setting up a Amazon SageMaker AI domain within Amazon DataZone, see [Set up SageMaker Assets \(administrator guide\)](#).

Users within the Amazon DataZone domain have permissions to all Amazon SageMaker AI actions, but their permissions are scoped down to resources within the Amazon DataZone domain.

Creating a Amazon DataZone domain streamlines creating a domain that allows your users to share data and models with each other. For information about how they can share data and models, see [Controlled access to assets with Amazon SageMaker Assets](#).

## Topics

- [Amazon SageMaker Studio](#)
- [Amazon SageMaker Studio Classic](#)
- [SageMaker JupyterLab](#)
- [Amazon SageMaker Notebook Instances](#)
- [Amazon SageMaker Studio Lab](#)
- [Amazon SageMaker Canvas](#)
- [Amazon SageMaker geospatial capabilities](#)

- [RStudio on Amazon SageMaker AI](#)
- [Code Editor in Amazon SageMaker Studio](#)
- [Amazon SageMaker HyperPod](#)
- [Generative AI in SageMaker notebook environments](#)
- [Amazon Q Developer](#)
- [Amazon SageMaker Partner AI Apps overview](#)

## Amazon SageMaker Studio

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

Amazon SageMaker Studio is the latest web-based experience for running ML workflows. Studio offers a suite of integrated development environments (IDEs). These include Code Editor, based on Code-OSS, Visual Studio Code - Open Source, a new JupyterLab application, RStudio, and Amazon SageMaker Studio Classic. For more information, see [Applications supported in Amazon SageMaker Studio](#).

The new web-based UI in Studio is faster and provides access to all SageMaker AI resources, including jobs and endpoints, in one interface. ML practitioners can also choose their preferred IDE to accelerate ML development. A data scientist can use JupyterLab to explore data and tune models. In addition, a machine learning operations (MLOps) engineer can use Code Editor with the pipelines tool in Studio to deploy and monitor models in production.

The previous Studio experience is still being supported as Amazon SageMaker Studio Classic. Studio Classic is the default experience for existing customers, and is available as an application in Studio. For more information about Studio Classic, see [Amazon SageMaker Studio Classic](#). For information about how to migrate from Studio Classic to Studio, see [Migration from Amazon SageMaker Studio Classic](#).

Studio offers the following benefits:

- A new JupyterLab application that has a faster start-up time and is more reliable than the existing Studio Classic application. For more information, see [SageMaker JupyterLab](#).
- A suite of IDEs that open in a separate tab, including the new Code Editor, based on Code-OSS, Visual Studio Code - Open Source application. Users can interact with supported IDEs in a full screen experience. For more information, see [Applications supported in Amazon SageMaker Studio](#).
- Access to all of your SageMaker AI resources in one place. Studio displays running instances across all of your applications.
- Access to all training jobs in a single view, regardless of whether they were scheduled from notebooks or initiated from Amazon SageMaker JumpStart.
- Simplified model deployment workflows and endpoint management and monitoring directly from Studio. You don't need to access the SageMaker AI console.
- Automatic creation of all configured applications when you onboard to a domain. For information about onboarding to a domain, see [Amazon SageMaker AI domain overview](#).
- An improved JumpStart experience where you can discover, import, register, fine tune, and deploy a foundation model. For more information, see [SageMaker JumpStart pretrained models](#).

## Topics

- [Migration from Amazon SageMaker Studio Classic](#)
- [Launch Amazon SageMaker Studio](#)
- [Amazon SageMaker Studio UI overview](#)
- [Amazon EFS auto-mounting in Studio](#)
- [Idle shutdown](#)
- [Applications supported in Amazon SageMaker Studio](#)
- [Lifecycle configurations within Amazon SageMaker Studio](#)
- [Amazon SageMaker Studio spaces](#)
- [Perform common UI tasks](#)
- [NVMe stores with Amazon SageMaker Studio](#)
- [Local mode support in Amazon SageMaker Studio](#)
- [View your Studio running instances, applications, and spaces](#)
- [Stop and delete your Studio running applications and spaces](#)
- [SageMaker Studio image support policy](#)

- [Amazon SageMaker Studio pricing](#)
- [Troubleshooting](#)

## Migration from Amazon SageMaker Studio Classic

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

When you open Amazon SageMaker Studio, the web-based UI is based on the chosen default experience. Amazon SageMaker AI currently supports two different default experiences: the Amazon SageMaker Studio experience and the Amazon SageMaker Studio Classic experience. To access the latest Amazon SageMaker Studio features, you must migrate existing domains from the Amazon SageMaker Studio Classic experience. When you migrate your default experience from Studio Classic to Studio, you don't lose any features, and can still access the Studio Classic IDE within Studio. For information about the added benefits of the Studio experience, see [Amazon SageMaker Studio](#).

### Note

- For existing customers that created their accounts before November 30, 2023, Studio Classic may be the default experience. You can enable Studio as your default experience using the AWS Command Line Interface (AWS CLI) or the Amazon SageMaker AI console. For more information about Studio Classic, see [Amazon SageMaker Studio Classic](#).
- For customers that created their accounts after November 30, 2023, we recommend using Studio as the default experience because it contains various integrated

development environments (IDEs), including the Studio Classic IDE, and other new features.

JupyterLab 3 reached its end of maintenance date on May 15, 2024. After December 31, 2024, you can only create new Studio Classic notebooks on JupyterLab 3 for a limited period. However after December 31, 2024, SageMaker AI will no longer provide fixes for critical issues on Studio Classic notebooks on JupyterLab 3. We recommend that you migrate your workloads to the new Studio experience, which supports JupyterLab 4.

- If Studio is your default experience, the UI is similar to the images found in [Amazon SageMaker Studio UI overview](#).
- If Studio Classic is your default experience, the UI is similar to the images found in [Amazon SageMaker Studio Classic UI Overview](#).

To migrate, you must update an existing domain. Migrating an existing domain from Studio Classic to Studio requires three distinct phases:

1. **Migrate the UI from Studio Classic to Studio:** One time, low lift task that requires creating a test domain to ensure Studio is compliant with your organization's network configurations before migrating the existing domain's UI from Studio Classic to Studio.
2. **(Optional) Migrate custom images and lifecycle configuration scripts:** Medium lift task for migrating your custom images and LCC scripts from Studio Classic to Studio.
3. **(Optional) Migrate data from Studio Classic to Studio:** Heavy lift task that requires using AWS DataSync to migrate data from the Studio Classic Amazon Elastic File System volume to either a target Amazon EFS or Amazon Elastic Block Store volume.
  - **(Optional) Migrate data flows from Data Wrangler in Studio Classic:** One time, low lift task for migrating your data flows from Data Wrangler in Studio Classic to Studio, which you can then access in the latest version of Studio through SageMaker Canvas. For more information, see [Migrate data flows from Data Wrangler](#).

The following topics show how to complete these phases to migrate an existing domain from Studio Classic to Studio.

## Automatic migration

Between July 2024 and August 2024, we are automatically upgrading the default landing experience for users to the new Studio experience. This only changes the default landing UI to the updated Studio UI. The Studio Classic application is still accessible from the new Studio UI.

To ensure that migration works successfully for your users, see [Migrate the UI from Studio Classic to Studio](#). In particular, ensure the following:

- the domain's execution role has the right permissions
- the default landing experience is set to Studio
- the domain's Amazon VPC, if applicable, is configured to Studio using the Studio VPC endpoint

However, if you need to continue having Studio Classic as your default UI for a limited time, set the landing experience to Studio Classic explicitly. For more information, see [Set Studio Classic as the default experience](#).

### Topics

- [Complete prerequisites to migrate the Studio experience](#)
- [Migrate the UI from Studio Classic to Studio](#)
- [\(Optional\) Migrate custom images and lifecycle configurations](#)
- [\(Optional\) Migrate data from Studio Classic to Studio](#)

## Complete prerequisites to migrate the Studio experience

Migration of the default experience from Studio Classic to Studio is managed by the administrator of the existing domain. If you do not have permissions to set Studio as the default experience for the existing domain, contact your administrator. To migrate your default experience, you must have administrator permissions or at least have permissions to update the existing domain, AWS Identity and Access Management (IAM), and Amazon Simple Storage Service (Amazon S3). Complete the following prerequisites before migrating an existing domain from Studio Classic to Studio.

- The AWS Identity and Access Management role used to complete migration must have a policy attached with at least the following permissions. For information about creating an IAM policy, see [Creating IAM policies](#).

**Note**

The release of Studio includes updates to the AWS managed policies. For more information, see [SageMaker AI Updates to AWS Managed Policies](#).

- Phase 1 required permissions:

- `iam:CreateServiceLinkedRole`
- `iam:PassRole`
- `sagemaker:DescribeDomain`
- `sagemaker:UpdateDomain`
- `sagemaker>CreateDomain`
- `sagemaker:CreateUserProfile`
- `sagemaker>ListApps`
- `sagemaker:AddTags`
- `sagemaker>DeleteApp`
- `sagemaker>DeleteSpace`
- `sagemaker:UpdateSpace`
- `sagemaker:DeleteUserProfile`
- `sagemaker>DeleteDomain`
- `s3:PutBucketCORS`

- Phase 2 required permissions (Optional, only if using lifecycle configuration scripts):

No additional permissions needed. If the existing domain has lifecycle configurations and custom images, the admin will already have the required permissions.

- Phase 3 using custom Amazon Elastic File System required permissions (Optional, only if transferring data):

- `efs>CreateFileSystem`
- `efs>CreateMountTarget`
- `efs:DescribeFileSystems`
- `efs:DescribeMountTargets`

- `efs:ModifyMountTargetSecurityGroups`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeNetworkInterfaceAttribute`
- `ec2:DescribeNetworkInterfaces`
- `ec2:AuthorizeSecurityGroupEgress`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2>CreateNetworkInterface`
- `ec2:CreateNetworkInterfacePermission`
- `ec2:RevokeSecurityGroupIngress`
- `ec2:RevokeSecurityGroupEgress`
- `ec2>DeleteSecurityGroup`
- `datasync>CreateLocationEfs`
- `datasync>CreateTask`
- `datasync:StartTaskExecution`
- `datasync>DeleteTask`
- `datasync>DeleteLocation`
- `sagemaker>ListUserProfiles`
- `sagemaker:DescribeUserProfile`
- `sagemaker:UpdateDomain`
- `sagemaker:UpdateUserProfile`
- Phase 3 using Amazon Simple Storage Service required permissions (Optional, only if transferring data):
  - `iam>CreateRole`
  - `iam:GetRole`
  - `iam:AttachRolePolicy`
  - `iam:DetachRolePolicy`
  - `iam>DeleteRole`
  - `efs:DescribeFileSystems`
  - `efs:DescribeMountTargets`

- `efs:DescribeMountTargetSecurityGroups`
  - `ec2:DescribeSubnets`
  - `ec2>CreateSecurityGroup`
  - `ec2:DescribeSecurityGroups`
  - `ec2:DescribeNetworkInterfaces`
  - `ec2:CreateNetworkInterface`
  - `ec2:CreateNetworkInterfacePermission`
  - `ec2:DetachNetworkInterfaces`
  - `ec2:DeleteNetworkInterface`
  - `ec2:DeleteNetworkInterfacePermission`
  - `ec2:CreateTags`
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:AuthorizeSecurityGroupIngress`
  - `ec2:RevokeSecurityGroupIngress`
  - `ec2:RevokeSecurityGroupEgress`
  - `ec2:DeleteSecurityGroup`
  - `datasync>CreateLocationEfs`
  - `datasync>CreateLocationS3`
  - `datasync>CreateTask`
  - `datasync:StartTaskExecution`
  - `datasync:DescribeTaskExecution`
  - `datasync>DeleteTask`
  - `datasync>DeleteLocation`
  - `sagemaker>CreateStudioLifecycleConfig`
  - `sagemaker:UpdateDomain`
  - `s3>ListBucket`
  - `s3:GetObject`
- Access to AWS services from a terminal environment on either:
    - Your local machine using the AWS CLI version 2.13+. Use the following command to verify the AWS CLI version.

```
aws --version
```

- AWS CloudShell. For more information, see [What is AWS CloudShell?](#)
- From your local machine or AWS CloudShell, run the following command and provide your AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).

```
aws configure
```

- Verify that the lightweight JSON processor, jq, is installed in the terminal environment. jq is required to parse AWS CLI responses.

```
jq --version
```

If jq is not installed, install it using one of the following commands:

- ```
sudo apt-get install -y jq
```
- ```
sudo yum install -y jq
```

## Migrate the UI from Studio Classic to Studio

The first phase for migrating an existing domain involves migrating the UI from Amazon SageMaker Studio Classic to Amazon SageMaker Studio. This phase does not include the migration of data. Users can continue working with their data the same way as they were before migration. For information about migrating data, see [\(Optional\) Migrate data from Studio Classic to Studio](#).

Phase 1 consists of the following steps:

1. Update application creation permissions for new applications available in Studio.
2. Update the VPC configuration for the domain.
3. Upgrade the domain to use the Studio UI.

### Prerequisites

Before running these steps, complete the prerequisites in [Complete prerequisites to migrate the Studio experience](#).

## Step 1: Update application creation permissions

Before migrating the domain, update the domain's execution role to grant users permissions to create applications.

1. Create an AWS Identity and Access Management policy with one of the following contents by following the steps in [Creating IAM policies](#):
  - Use the following policy to grant permissions for all application types and spaces.

 **Note**

If the domain uses the `SageMakerFullAccess` policy, you do not need to perform this action. `SageMakerFullAccess` grants permissions to create all applications.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "SMStudioUserProfileAppPermissionsCreateAndDelete",  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker:CreateApp",  
                "sagemaker:DeleteApp"  
            ],  
            "Resource": "arn:aws:sagemaker:region:account-id:app/*",  
            "Condition": {  
                "Null": {  
                    "sagemaker:OwnerUserProfileArn": "true"  
                }  
            }  
        },  
        {  
            "Sid": "SMStudioCreatePresignedDomainUrlForUserProfile",  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker>CreatePresignedDomainUrl"  
            ],  
            "Resource": "arn:aws:sagemaker:region:account-id:user-profile/  
${sagemaker:DomainId}/${sagemaker:UserProfileName}"  
        },  
    ]  
}
```

```
{  
    "Sid": "SMStudioAppPermissionsListAndDescribe",  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker>ListApps",  
        "sagemaker>ListDomains",  
        "sagemaker>ListUserProfiles",  
        "sagemaker>ListSpaces",  
        "sagemaker>DescribeApp",  
        "sagemaker>DescribeDomain",  
        "sagemaker>DescribeUserProfile",  
        "sagemaker>DescribeSpace"  
    ],  
    "Resource": "*"  
,  
{  
    "Sid": "SMStudioAppPermissionsTagOnCreate",  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker>AddTags"  
    ],  
    "Resource": "arn:aws:sagemaker:region:account-id:*/*",  
    "Condition": {  
        "Null": {  
            "sagemaker>TaggingAction": "false"  
        }  
    }  
,  
{  
    "Sid": "SMStudioRestrictSharedSpacesWithoutOwners",  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker>CreateSpace",  
        "sagemaker>UpdateSpace",  
        "sagemaker>DeleteSpace"  
    ],  
    "Resource": "arn:aws:sagemaker:region:account-id:space/  
${sagemaker:DomainId}/*",  
    "Condition": {  
        "Null": {  
            "sagemaker>OwnerUserProfileArn": "true"  
        }  
    }  
,  
},
```

```
{  
    "Sid": "SMStudioRestrictSpacesToOwnerUserProfile",  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker:CreateSpace",  
        "sagemaker:UpdateSpace",  
        "sagemaker:DeleteSpace"  
    ],  
    "Resource": "arn:aws:sagemaker:region:account-id:space/  
${sagemaker:DomainId}/*",  
    "Condition": {  
        "ArnLike": {  
            "sagemaker:OwnerUserProfileArn": "arn:aws:sagemaker:us-  
east-1:account-id:user-profile/${sagemaker:DomainId}/  
${sagemaker:UserName}"  
        },  
        "StringEquals": {  
            "sagemaker:SpaceSharingType": [  
                "Private",  
                "Shared"  
            ]  
        }  
    }  
},  
{  
    "Sid": "SMStudioRestrictCreatePrivateSpaceAppsToOwnerUserProfile",  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker>CreateApp",  
        "sagemaker>DeleteApp"  
    ],  
    "Resource": "arn:aws:sagemaker:region:account-id:app/  
${sagemaker:DomainId}/*",  
    "Condition": {  
        "ArnLike": {  
            "sagemaker:OwnerUserProfileArn": "arn:aws:sagemaker:us-  
east-1:account-id:user-profile/${sagemaker:DomainId}/  
${sagemaker:UserName}"  
        },  
        "StringEquals": {  
            "sagemaker:SpaceSharingType": [  
                "Private"  
            ]  
        }  
    }  
}
```

```
        }
    },
    {
        "Sid": "AllowAppActionsForSharedSpaces",
        "Effect": "Allow",
        "Action": [
            "sagemaker>CreateApp",
            "sagemaker>DeleteApp"
        ],
        "Resource": "arn:aws:sagemaker:*:*:app/${sagemaker:DomainId}/*/*/*",
        "Condition": {
            "StringEquals": {
                "sagemaker:SpaceSharingType": [
                    "Shared"
                ]
            }
        }
    }
]
```

- Because Studio shows an expanded set of applications, users may have access to applications that weren't displayed before. Administrators can limit access to these default applications by creating an AWS Identity and Access Management (IAM) policy that grants denies permissions for some applications to specific users.

 **Note**

Application type can be either `jupyterlab` or `codeeditor`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenySageMakerCreateAppForSpecificAppTypes",
            "Effect": "Deny",
            "Action": "sagemaker>CreateApp",
            "Resource": "arn:aws:sagemaker:region:account-id:app/domain-id/*/app-type/*"
        }
    ]
}
```

{}

2. Attach the policy to the execution role of the domain. For instructions, follow the steps in [Adding IAM identity permissions \(console\)](#).

## Step 2: Update VPC configuration

If you use your domain in VPC-Only mode, ensure your VPC configuration meets the requirements for using Studio in VPC-Only mode. For more information, see [Connect Amazon SageMaker Studio in a VPC to External Resources](#).

## Step 3: Upgrade to the Studio UI

Before you migrate your existing domain from Studio Classic to Studio, we recommend creating a test domain using Studio with the same configurations as your existing domain.

### (Optional) Create a test domain

Use this test domain to interact with Studio, test out networking configurations, and launch applications, before migrating the existing domain.

1. Get the domain ID of your existing domain.
  - a. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
  - b. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
  - c. Choose the existing domain.
  - d. On the **Domain details** page, choose the **Domain settings** tab.
  - e. Copy the **Domain ID**.
2. Add the domain ID of your existing domain.

```
export REF_DOMAIN_ID="domain-id"  
export SM_REGION="region"
```

3. Use `describe-domain` to get important information about the existing domain.

```
export REF_EXECROLE=$(aws sagemaker describe-domain --region=$SM_REGION --domain-id=$REF_DOMAIN_ID | jq -r '.DefaultUserSettings.ExecutionRole')  
export REF_VPC=$(aws sagemaker describe-domain --region=$SM_REGION --domain-id=$REF_DOMAIN_ID | jq -r '.VpcId')
```

```
export REF_SIDS=$(aws sagemaker describe-domain --region=$SM_REGION --domain-id=$REF_DOMAIN_ID | jq -r '.SubnetIds | join(",")')
export REF_SGS=$(aws sagemaker describe-domain --region=$SM_REGION --domain-id=$REF_DOMAIN_ID | jq -r '.DefaultUserSettings.SecurityGroups | join(",")')
export AUTHMODE=$(aws sagemaker describe-domain --region=$SM_REGION --domain-id=$REF_DOMAIN_ID | jq -r '.AuthMode')
```

#### 4. Validate the parameters.

```
echo "Execution Role: $REF_EXECROLE || VPCID: $REF_VPC || SubnetIDs: $REF_SIDS || Security GroupIDs: $REF_SGS || AuthMode: $AUTHMODE"
```

#### 5. Create a test domain using the configurations from the existing domain.

```
IFS=',' read -r -a subnet_ids <<< "$REF_SIDS"
IFS=',' read -r -a security_groups <<< "$REF_SGS"
security_groups_json=$(printf '%s\n' "${security_groups[@]}") | jq -R . | jq -s .

aws sagemaker create-domain \
--domain-name "TestV2Config" \
--vpc-id $REF_VPC \
--auth-mode $AUTHMODE \
--subnet-ids "${subnet_ids[@]}" \
--app-network-access-type VpcOnly \
--default-user-settings "
{
    \"ExecutionRole\": \"$REF_EXECROLE\",
    \"StudioWebPortal\": \"ENABLED\",
    \"DefaultLandingUri\": \"studio::\",
    \"SecurityGroups\": $security_groups_json
}
"
```

#### 6. After the test domain is In Service, use the test domain's ID to create a user profile. This user profile is used to launch and test applications.

```
aws sagemaker create-user-profile \
--region="$SM_REGION" --domain-id=test-domain-id \
--user-profile-name test-network-user
```

## Test Studio functionality

Launch the test domain using the `test-network-user` user profile. We suggest that you thoroughly test the Studio UI and create applications to test Studio functionality in VPCOnly mode. Test the following workflows:

- Create a new JupyterLab Space, test environment and connectivity.
- Create a new Code Editor, based on Code-OSS, Visual Studio Code - Open Source Space, test environment and connectivity.
- Launch a new Studio Classic App, test environment and connectivity.
- Test Amazon Simple Storage Service connectivity with test read and write actions.

If these tests are successful, then upgrade the existing domain. If you encounter any failures, we recommended fixing your environment and connectivity issues before updating the existing domain.

## Clean up test domain resources

After you have migrated the existing domain, clean up test domain resources.

1. Add the test domain's ID.

```
export TEST_DOMAIN="test-domain-id"  
export SM_REGION="region"
```

2. List all applications in the domain that are in a running state.

```
active_apps_json=$(aws sagemaker list-apps --region=$SM_REGION --domain-id=$TEST_DOMAIN)  
echo $active_apps_json
```

3. Parse the JSON list of running applications and delete them. If users attempted to create an application that they do not have permissions for, there may be spaces that are not captured in the following script. You must manually delete these spaces.

```
echo "$active_apps_json" | jq -c '.Apps[]' | while read -r app;  
do  
    if echo "$app" | jq -e '. | has("SpaceName")' > /dev/null;  
    then
```

```
app_type=$(echo "$app" | jq -r '.AppType')
app_name=$(echo "$app" | jq -r '.AppName')
domain_id=$(echo "$app" | jq -r '.DomainId')
space_name=$(echo "$app" | jq -r '.SpaceName')

echo "Deleting App - AppType: $app_type || AppName: $app_name || DomainId: $domain_id || SpaceName: $space_name"
aws sagemaker delete-app --region=$SM_REGION --domain-id=$domain_id \
--app-type $app_type --app-name $app_name --space-name $space_name

echo "Deleting Space - AppType: $app_type || AppName: $app_name || DomainId: $domain_id || SpaceName: $space_name"
aws sagemaker delete-space --region=$SM_REGION --domain-id=$domain_id \
--space-name $space_name

else

    app_type=$(echo "$app" | jq -r '.AppType')
    app_name=$(echo "$app" | jq -r '.AppName')
    domain_id=$(echo "$app" | jq -r '.DomainId')
    user_profile_name=$(echo "$app" | jq -r '.UserProfileName')

    echo "Deleting Studio Classic - AppType: $app_type || AppName: $app_name || DomainId: $domain_id || UserProfileName: $user_profile_name"
    aws sagemaker delete-app --region=$SM_REGION --domain-id=$domain_id \
    --app-type $app_type --app-name $app_name --user-profile-name $user_profile_name

fi

done
```

#### 4. Delete the test user profile.

```
aws sagemaker delete-user-profile \
--region=$SM_REGION --domain-id=$TEST_DOMAIN \
--user-profile-name "test-network-user"
```

#### 5. Delete the test domain.

```
aws sagemaker delete-domain \
--region=$SM_REGION --domain-id=$TEST_DOMAIN
```

After you have tested Studio functionality with the configurations in your test domain, migrate the existing domain. When Studio is the default experience for a domain, Studio is the default experience for all users in the domain. However, the user settings takes precedence over the domain settings. Therefore, if a user has their default experience set to Studio Classic in their user settings, then that user will have Studio Classic as their default experience.

You can migrate the existing domain by updating it from the SageMaker AI console, the AWS CLI, or AWS CloudFormation. Choose one of the following tabs to view the relevant instructions.

## Set Studio as the default experience for the existing domain using the SageMaker AI console

You can set Studio as the default experience for the existing domain by using the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation pane expand **Admin configurations** and choose **Domains**.
3. Choose the existing domain that you want to enable Studio as the default experience for.
4. On the **Domain details** page expand **Enable the new Studio**.
5. (Optional) To view the details about the steps involved in enabling Studio as your default experience, choose **View details**. The page shows the following.
  - In the **SageMaker Studio Overview** section you can view the applications that are included or available in the Studio web-based interface.
  - In the **Enablement process** section you can view descriptions of the workflow tasks to enable Studio.

### Note

You will need to migrate your data manually. For instructions about migrating your data, see [\(Optional\) Migrate data from Studio Classic to Studio](#).

- In the **Revert to Studio Classic experience** section you can view how to revert back to Studio Classic after enabling Studio as your default experience.
6. To begin the process to enable Studio as your default experience, choose **Enable the new Studio**.
  7. In the **Specify and configure role** section, you can view the default applications that are automatically included in Studio.

To prevent users from running these applications, choose the AWS Identity and Access Management (IAM) Role that has an IAM policy that denies access. For information about how to create a policy to limit access, see [Step 1: Update application creation permissions](#).

8. In the **Choose default S3 bucket to attach CORS policy** section, you can give Studio access to Amazon S3 buckets. The default Amazon S3 bucket, in this case, is the default Amazon S3 bucket for your Studio Classic. In this step you can do the following:

- Verify the domain's default Amazon S3 bucket to attach the CORS policy to. If your domain does not have a default Amazon S3 bucket, SageMaker AI creates an Amazon S3 bucket with the correct CORS policy attached.
- You can include 10 additional Amazon S3 buckets to attach the CORS policy to.

If you wish to include more than 10 buckets, you can add them manually. For more information about manually attaching the CORS policy to your Amazon S3 buckets, see [\(Optional\) Update your CORS policy to access Amazon S3 buckets](#).

To proceed, select the check box next to **Do you agree to overriding any existing CORS policy on the chosen Amazon S3 buckets?**.

9. The **Migrate data** section contains information about the different data storage volumes for Studio Classic and Studio. Your data will not be migrated automatically through this process. For instructions about migrating your data, lifecycle configurations, and JupyterLab extensions, see [\(Optional\) Migrate data from Studio Classic to Studio](#).
10. Once you have completed the tasks on the page and verified your configuration, choose **Enable the new Studio**.

## Set Studio as the default experience for the existing domain using the AWS CLI

To set Studio as the default experience for the existing domain using the AWS CLI, use the [update-domain](#) call. You must set ENABLED as the value for StudioWebPortal, and set studio:: as the value for DefaultLandingUri as part of the default-user-settings parameter.

StudioWebPortal indicates if the Studio experience is the default experience and DefaultLandingUri indicates the default experience that the user is directed to when accessing the domain. In this example, setting these values on a domain level (in default-user-settings) makes Studio the default experience for users within the domain.

If a user within the domain has their `StudioWebPortal` set to `DISABLED` and `DefaultLandingUri` set to `app:JupyterServer`: on a user level (in `UserSettings`), this takes precedence over the domain settings. In other words, that user will have Studio Classic as their default experience, regardless of the domain settings.

The following code example shows how to set Studio as the default experience for users within the domain:

```
aws sagemaker update-domain \
--domain-id existing-domain-id \
--region AWS Region \
--default-user-settings ' \
{
    "StudioWebPortal": "ENABLED",
    "DefaultLandingUri": "studio::"
}'
```

- To obtain your *existing-domain-id*, use the following instructions:

### To get *existing-domain-id*

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
  2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
  3. Choose the existing domain.
  4. On the **Domain details** page, choose the **Domain settings** tab.
  5. Copy the **Domain ID**.
- To ensure you are using the correct AWS Region for your domain, use the following instructions:

### To get *AWS Region*

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
3. Choose the existing domain.
4. On the **Domain details** page, verify that this is the existing domain.
5. Expand the AWS Region dropdown list from the top right of the SageMaker AI console, and use the corresponding AWS Region ID to the right of your AWS Region name. For example, `us-west-1`.

After you migrate your default experience to Studio, you can give Studio access to Amazon S3 buckets. For example, you can include access to your Studio Classic default Amazon S3 bucket and additional Amazon S3 buckets. To do so, you must manually attach a [Cross-Origin Resource Sharing \(CORS\)](#) configuration to the Amazon S3 buckets. For more information about how to manually attach the CORS policy to your Amazon S3 buckets, see [\(Optional\) Update your CORS policy to access Amazon S3 buckets](#).

Similarly, you can set Studio as the default experience when you create a domain from the AWS CLI using the [create-domain](#) call.

## Set Studio as the default experience for the existing domain using the AWS CloudFormation

You can set the default experience when creating a domain using the AWS CloudFormation. For an AWS CloudFormation migration template, see [SageMaker Studio Administrator IaC Templates](#). For more information about creating a domain using AWS CloudFormation, see [Creating Amazon SageMaker AI domain using AWS CloudFormation](#).

For information about the domain resource supported by AWS CloudFormation, see [AWS::SageMaker AI::Domain](#).

After you migrate your default experience to Studio, you can give Studio access to Amazon S3 buckets. For example, you can include access to your Studio Classic default Amazon S3 bucket and additional Amazon S3 buckets. To do so, you must manually attach a [Cross-Origin Resource Sharing \(CORS\)](#) configuration to the Amazon S3 buckets. For information about how to manually attach the CORS policy to your Amazon S3 buckets, see [\(Optional\) Update your CORS policy to access Amazon S3 buckets](#).

## (Optional) Update your CORS policy to access Amazon S3 buckets

In Studio Classic, users can create, list, and upload files to Amazon Simple Storage Service (Amazon S3) buckets. To support the same experience in Studio, administrators must attach a [Cross-Origin Resource Sharing \(CORS\)](#) configuration to the Amazon S3 buckets. This is required because Studio makes Amazon S3 calls from the internet browser. The browser invokes CORS on behalf of users. As a result, all of the requests to Amazon S3 buckets fail unless the CORS policy is attached to the Amazon S3 buckets.

You may need to manually attach the CORS policy to Amazon S3 buckets for the following reasons.

- If there is already an existing Amazon S3 default bucket that doesn't have the correct CORS policy attached when you migrate the existing domain's default experience to Studio.

- If you are using the AWS CLI to migrate the existing domain's default experience to Studio. For information about using the AWS CLI to migrate, see [Set Studio as the default experience for the existing domain using the AWS CLI](#).
- If you want to attach the CORS policy to additional Amazon S3 buckets.

### Note

If you plan to use the SageMaker AI console to enable Studio as your default experience, the Amazon S3 buckets that you attach the CORS policy to will have their existing CORS policies overridden during the migration. For this reason, you can ignore the following manual instructions.

However, if you have already used the SageMaker AI console to migrate and want to include more Amazon S3 buckets to attach the CORS policy to, then continue with the following manual instructions.

The following procedure shows how to manually add a CORS configuration to an Amazon S3 bucket.

### To add a CORS configuration to an Amazon S3 bucket

1. Verify that there is an Amazon S3 bucket in the same AWS Region as the existing domain with the following name. For instructions, see [Viewing the properties for an Amazon S3 bucket](#).

`sagemaker-region-account-id`

2. Add a CORS configuration with the following content to the default Amazon S3 bucket. For instructions, see [Configuring cross-origin resource sharing \(CORS\)](#).

```
[  
 {  
     "AllowedHeaders": [  
         "*"  
     ],  
     "AllowedMethods": [  
         "POST",  
         "PUT",  
         "GET",  
         "HEAD",  
     ]  
 }]
```

```
        "DELETE"
    ],
    "AllowedOrigins": [
        "https://*.sagemaker.aws"
    ],
    "ExposeHeaders": [
        "ETag",
        "x-amz-delete-marker",
        "x-amz-id-2",
        "x-amz-request-id",
        "x-amz-server-side-encryption",
        "x-amz-version-id"
    ]
}
```

## (Optional) Migrate from Data Wrangler in Studio Classic to SageMaker Canvas

Amazon SageMaker Data Wrangler exists as its own feature in the Studio Classic experience. When you enable Studio as your default experience, use the [Amazon SageMaker Canvas](#) application to access Data Wrangler functionality. SageMaker Canvas is an application in which you can train and deploy machine learning models without writing any code, and Canvas provides data preparation features powered by Data Wrangler.

The new Studio experience doesn't support the classic Data Wrangler UI, and you must create a Canvas application if you want to continue using Data Wrangler. However, you must have the necessary permissions to create and use Canvas applications.

Complete the following steps to attach the necessary permissions policies to your SageMaker AI domain's or user's AWS IAM role.

### To grant permissions for Data Wrangler functionality inside Canvas

1. Attach the AWS managed policy [AmazonSageMakerFullAccess](#) to your user's IAM role. For a procedure that shows you how to attach IAM policies to a role, see [Adding IAM identity permissions \(console\)](#) in the *AWS IAM User Guide*.

If this permissions policy is too permissive for your use case, you can create scoped-down policies that include at least the following permissions:

{

```
"Sid": "AllowStudioActions",
"Effect": "Allow",
>Action": [
    "sagemaker>CreatePresignedDomainUrl",
    "sagemaker>DescribeDomain",
    "sagemaker>ListDomains",
    "sagemaker>DescribeUserProfile",
    "sagemaker>ListUserProfiles",
    "sagemaker>DescribeSpace",
    "sagemaker>ListSpaces",
    "sagemaker>DescribeApp",
    "sagemaker>ListApps"
],
"Resource": "*"
},
{
"Sid": "AllowAppActionsForUserProfile",
"Effect": "Allow",
>Action": [
    "sagemaker>CreateApp",
    "sagemaker>DeleteApp"
],
"Resource": "arn:aws:sagemaker:region:account-id:app/domain-id/user-profile-name/canvas/*",
"Condition": {
    "Null": {
        "sagemaker:OwnerUserProfileArn": "true"
    }
}
}
```

2. Attach the AWS managed policy [AmazonSageMakerCanvasDataPrepFullAccess](#) to your user's IAM role.

After attaching the necessary permissions, you can create a Canvas application and log in. For more information, see [Getting started with using Amazon SageMaker Canvas](#).

When you've logged into Canvas, you can directly access Data Wrangler and begin creating data flows. For more information, see [Data preparation](#) in the Canvas documentation.

## (Optional) Migrate from Autopilot in Studio Classic to SageMaker Canvas

[Amazon SageMaker Autopilot](#) exists as its own feature in the Studio Classic experience. When you migrate to using the updated Studio experience, use the [Amazon SageMaker Canvas](#) application to continue using the same automated machine learning (AutoML) capabilities via a user interface (UI). SageMaker Canvas is an application in which you can train and deploy machine learning models without writing any code, and Canvas provides a UI to run your AutoML tasks.

The new Studio experience doesn't support the classic Autopilot UI. You must create a Canvas application if you want to continue using Autopilot's AutoML features via a UI.

However, you must have the necessary permissions to create and use Canvas applications.

- If you are accessing SageMaker Canvas from Studio, add those permissions to the execution role of your SageMaker AI domain or user profile.
- If you are accessing SageMaker Canvas from the Console, add those permissions to your user's AWS IAM role.
- If you are accessing SageMaker Canvas via a [presigned URL](#), add those permissions to the IAM role that you're using for Okta SSO access.

To enable AutoML capabilities in Canvas, add the following policies to your execution role or IAM user role.

- AWS managed policy: [CanvasFullAccess](#).
- Inline policy:

```
{  
    "Sid": "AllowAppActionsForUserProfile",  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker>CreateApp",  
        "sagemaker>DeleteApp"  
    ],  
    "Resource": "arn:aws:sagemaker:region:account-id:app/domain-id/user-profile-name/canvas/*",  
    "Condition": {  
        "Null": {  
            "sagemaker:OwnerUserProfileArn": "true"  
        }  
    }  
}
```

{}

## To attach IAM policies to an execution role

1. **Find the execution role attached to your SageMaker AI user profile**
  - a. In the SageMaker AI console <https://console.aws.amazon.com/sagemaker/>, navigate to **Domains**, then choose your SageMaker AI domain.
  - b. The execution role ARN is listed under *Execution role* on the **User Details** page of your user profile. Make note of the execution role name in the ARN.
  - c. In the IAM console <https://console.aws.amazon.com/iam/>, choose **Roles**.
  - d. Search for your role by name in the search field.
  - e. Select the role.
2. Add policies to the role
  - a. In the IAM console <https://console.aws.amazon.com/iam/>, choose **Roles**.
  - b. Search for your role by name in the search field.
  - c. Select the role.
  - d. In the **Permissions** tab, navigate to the dropdown menu **Add permissions**.
    - For managed policies: Select **Attach policies**, search for the name of the manage policy you want to attach.  
Select the policy then choose **Add permissions**.
      - For inline policies: Select **Create inline policy**, paste your policy in the JSON tab, choose next, name your policy, and choose **Create**.

For a procedure that shows you how to attach IAM policies to a role, see [Adding IAM identity permissions \(console\)](#) in the *AWS IAM User Guide*.

After attaching the necessary permissions, you can create a Canvas application and log in. For more information, see [Getting started with using Amazon SageMaker Canvas](#).

## Set Studio Classic as the default experience

Administrators can revert to Studio Classic as the default experience for an existing domain. This can be done through the AWS CLI.

### Note

When Studio Classic is set as the default experience on a domain level, Studio Classic is the default experience for all users in the domain. However, settings on a user level takes precedence over the domain level settings. So if a user has their default experience set to Studio, then that user will have Studio as their default experience.

To revert to Studio Classic as the default experience for the existing domain using the AWS CLI, use the [update-domain](#) call. As part of the `default-user-settings` field, you must set:

- `StudioWebPortal` value to `DISABLED`.
- `DefaultLandingUri` value to `app:JupyterServer`:

`StudioWebPortal` indicates if the Studio experience is the default experience and `DefaultLandingUri` indicates the default experience that the user is directed to when accessing the domain. In this example, setting these values on a domain level (in `default-user-settings`) makes Studio Classic the default experience for users within the domain.

If a user within the domain has their `StudioWebPortal` set to `ENABLED` and `DefaultLandingUri` set to `studio::` on a user level (in `UserSettings`), this takes precedence over the domain level settings. In other words, that user will have Studio as their default experience, regardless of the domain level settings.

The following code example shows how to set Studio Classic as the default experience for users within the domain:

```
aws sagemaker update-domain \
--domain-id existing-domain-id \
--region AWS Region \
--default-user-settings ' \
{ \
    "StudioWebPortal": "DISABLED", \
    "DefaultLandingUri": "app:JupyterServer:" \
} \
'
```

Use the following instructions to obtain your *existing-domain-id*.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
3. Choose the existing domain.
4. On the **Domain details** page, choose the **Domain settings** tab.
5. Copy the **Domain ID**.

To obtain your *AWS Region*, use the following instructions to ensure you are using the correct AWS Region for your domain.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
3. Choose the existing domain.
4. On the **Domain details** page, verify that this is the existing domain.
5. Expand the AWS Region dropdown list from the top right of the SageMaker AI console, and use the corresponding AWS Region ID to the right of your AWS Region name. For example, us-west-1.

## (Optional) Migrate custom images and lifecycle configurations

You must update your custom images and lifecycle configuration (LCC) scripts to work with the simplified local run model in Amazon SageMaker Studio. If you have not created custom images or lifecycle configurations in your domain, skip this phase.

Amazon SageMaker Studio Classic operates in a split environment with:

- A `JupyterServer` application running the Jupyter Server.
- Studio Classic notebooks running on one or more `KernelGateway` applications.

Studio has shifted away from a split environment. Studio runs the `JupyterLab` and `Code Editor`, based on `Code-OSS`, `Visual Studio Code - Open Source` applications in a local runtime model. For more information about the change in architecture, see [Boost productivity on Amazon SageMaker Studio](#).

## Migrate custom images

Your existing Studio Classic custom images may not work in Studio. We recommend creating a new custom image that satisfies the requirements for use in Studio. The release of Studio simplifies the process to build custom images by providing [SageMaker Studio image support policy](#). SageMaker AI Distribution images include popular libraries and packages for machine learning, data science, and data analytics visualization. For a list of base SageMaker Distribution images and Amazon Elastic Container Registry account information, see [Amazon SageMaker images available for use with Studio Classic](#).

To build a custom image, complete one of the following.

- Extend a SageMaker Distribution image with custom packages and modules. These images are pre-configured with JupyterLab and Code Editor, based on Code-OSS, Visual Studio Code - Open Source.
- Build a custom Dockerfile file by following the instructions in [Dockerfile specifications](#). You must install JupyterLab and the open source CodeServer on the image to make it compatible with Studio.

## Migrate lifecycle configurations

Because of the simplified local runtime model in Studio, we recommend migrating the structure of your existing Studio Classic LCCs. In Studio Classic, you often have to create separate lifecycle configurations for both KernelGateway and JupyterServer applications. Because the JupyterServer and KernelGateway applications run on separate compute resources within Studio Classic, Studio Classic LCCs can be one of either type:

- JupyterServer LCC: These LCCs mostly govern a user's home actions, including setting proxy, creating environment variables, and auto-shutdown of resources.
- KernelGateway LCC: These LCCs govern Studio Classic notebook environment optimizations. This includes updating numpy package versions in the Data Science 3.0 kernel and installing the snowflake package in Pytorch 2.0 GPU kernel.

In the simplified Studio architecture, you only need one LCC script that runs at application start up. While migration of your LCC scripts varies based on development environment, we recommend combining JupyterServer and KernelGateway LCCs to build a combined LCC.

LCCs in Studio can be associated with one of the following applications:

- JupyterLab
- Code Editor

Users can select the LCC for the respective application type when creating a space or use the default LCC set by the admin.

 **Note**

Existing Studio Classic auto-shutdown scripts do not work with Studio. For an example Studio auto-shutdown script, see [SageMaker Studio Lifecycle Configuration examples](#).

## Considerations when refactoring LCCs

Consider the following differences between Studio Classic and Studio when refactoring your LCCs.

- JupyterLab and Code Editor applications, when created, are run as `sagemaker-user` with `UID:1001` and `GID:101`. By default, `sagemaker-user` has permissions to assume sudo/root permissions. KernelGateway applications are run as `root` by default.
- SageMaker Distribution images that run inside JupyterLab and Code Editor apps use the Debian-based package manager, `apt-get`.
- Studio JupyterLab and Code Editor applications use the Conda package manager. SageMaker AI creates a single base Python3 Conda environment when a Studio application is launched. For information about updating packages in the base Conda environment and creating new Conda environments, see [JupyterLab user guide](#). In contrast, not all KernelGateway applications use Conda as a package manager.
- The Studio JupyterLab application uses JupyterLab 4.0, while Studio Classic uses JupyterLab 3.0. Validate that all JupyterLab extensions you use are compatible with JupyterLab 4.0. For more information about extensions, see [Extension Compatibility with JupyterLab 4.0](#).

## (Optional) Migrate data from Studio Classic to Studio

Studio Classic and Studio use two different types of storage volumes. Studio Classic uses a single Amazon Elastic File System (Amazon EFS) volume to store data across all users and shared spaces in the domain. In Studio, each space gets its own Amazon Elastic Block Store (Amazon EBS) volume.

When you update the default experience of an existing domain, SageMaker AI automatically mounts a folder in an Amazon EFS volume for each user in a domain. As a result, users are able to access files from Studio Classic in their Studio applications. For more information, see [Amazon EFS auto-mounting in Studio](#).

You can also opt out of Amazon EFS auto-mounting and manually migrate the data to give users access to files from Studio Classic in Studio applications. To accomplish this, you must transfer the files from the user home directories to the Amazon EBS volumes associated with those spaces. The following section gives information about this workflow. For more information about opting out of Amazon EFS auto-mounting, see [Opt out of Amazon EFS auto-mounting](#).

## Manually migrate all of your data from Studio Classic

The following section describes how to migrate all of the data from your Studio Classic storage volume to the new Studio experience.

When manually migrating a user's data, code, and artifacts from Studio Classic to Studio, we recommend one of the following approaches:

1. Using a custom Amazon EFS volume
2. Using Amazon Simple Storage Service (Amazon S3)

If you used Amazon SageMaker Data Wrangler in Studio Classic and want to migrate your data flow files, then choose one of the following options for migration:

- If you want to migrate all of the data from your Studio Classic storage volume, including your data flow files, go to [Manually migrate all of your data from Studio Classic](#) and complete the section **Use Amazon S3 to migrate data**. Then, skip to the [Import the flow files into Canvas](#) section.
- If you only want to migrate your data flow files and no other data from your Studio Classic storage volume, skip to the [Migrate data flows from Data Wrangler](#) section.

## Prerequisites

Before running these steps, complete the prerequisites in [Complete prerequisites to migrate the Studio experience](#). You must also complete the steps in [Migrate the UI from Studio Classic to Studio](#).

## Choosing an approach

Consider the following when choosing an approach to migrate your Studio Classic data.

### Pros and cons of using a custom Amazon EFS volume

In this approach, you use an Amazon EFS-to-Amazon EFS AWS DataSync task (one time or cadence) to copy data, then mount the target Amazon EFS volume to a user's spaces. This gives users access to data from Studio Classic in their Studio compute environments.

Pros:

- Only the user's home directory data is visible in the user's spaces. There is no data cross-pollination.
- Syncing from the source Amazon EFS volume to a target Amazon EFS volume is safer than directly mounting the source Amazon EFS volume managed by SageMaker AI into spaces. This avoids the potential to impact home directory user files.
- Users have the flexibility to continue working in Studio Classic and Studio applications, while having their data available in both applications if AWS DataSync is set up on a regular cadence.
- No need for repeated push and pull with Amazon S3.

Cons:

- No write access to the target Amazon EFS volume mounted to user's spaces. To get write access to the target Amazon EFS volume, customers would need to mount the target Amazon EFS volume to an Amazon Elastic Compute Cloud instance and provide appropriate permissions for users to write to the Amazon EFS prefix.
- Requires modification to the security groups managed by SageMaker AI to allow network file system (NFS) inbound and outbound flow.
- Costs more than using Amazon S3.
- If [migrating data flows from Data Wrangler in Studio Classic](#), you must follow the steps for manually exporting flow files.

### Pros and cons of using Amazon S3

In this approach, you use an Amazon EFS-to-Amazon S3 AWS DataSync task (one time or cadence) to copy data, then create a lifecycle configuration to copy the user's data from Amazon S3 to their private space's Amazon EBS volume.

## Pros:

- If the LCC is attached to the domain, users can choose to use the LCC to copy data to their space or to run the space with no LCC script. This gives users the choice to copy their files only to the spaces they need.
- If an AWS DataSync task is set up on a cadence, users can restart their Studio application to get the latest files.
- Because the data is copied over to Amazon EBS, users have write permissions on the files.
- Amazon S3 storage is cheaper than Amazon EFS.
- If [migrating data flows from Data Wrangler in Studio Classic](#), you can skip the manual export steps and directly import the data flows into SageMaker Canvas from Amazon S3.

## Cons:

- If administrators need to prevent cross-pollination, they must create AWS Identity and Access Management policies at the user level to ensure users can only access the Amazon S3 prefix that contains their files.

## Use a custom Amazon EFS volume to migrate data

In this approach, you use an Amazon EFS-to-Amazon EFS AWS DataSync to copy the contents of a Studio Classic Amazon EFS volume to a target Amazon EFS volume once or in a regular cadence, then mount the target Amazon EFS volume to a user's spaces. This gives users access to data from Studio Classic in their Studio compute environments.

1. Create a target Amazon EFS volume. You will transfer data into this Amazon EFS volume and mount it to a corresponding user's space using prefix-level mounting.

```
export SOURCE_DOMAIN_ID="domain-id"  
export REGION="region"  
  
export TARGET_EFS=$(aws efs create-file-system --performance-mode generalPurpose --throughput-mode bursting --encrypted --region $REGION | jq -r '.FileSystemId')  
  
echo "Target EFS volume Created: $TARGET_EFS"
```

2. Add variables for the source Amazon EFS volume currently attached to the domain and used by all users. The domain's Amazon Virtual Private Cloud information is required to ensure the

target Amazon EFS is created in the same Amazon VPC and subnet, with the same security group configuration.

```
export SOURCE_EFS=$(aws sagemaker describe-domain --domain-id $SOURCE_DOMAIN_ID | jq -r '.HomeEfsFileSystemId')
export VPC_ID=$(aws sagemaker describe-domain --domain-id $SOURCE_DOMAIN_ID | jq -r '.VpcId')

echo "EFS managed by SageMaker: $SOURCE_EFS | VPC: $VPC_ID"
```

3. Create an Amazon EFS mount target in the same Amazon VPC and subnet as the source Amazon EFS volume, with the same security group configuration. The mount target takes a few minutes to be available.

```
export EFS_VPC_ID=$(aws efs describe-mount-targets --file-system-id $SOURCE_EFS | jq -r ".MountTargets[0].VpcId")
export EFS_AZ_NAME=$(aws efs describe-mount-targets --file-system-id $SOURCE_EFS | jq -r ".MountTargets[0].AvailabilityZoneName")
export EFS_AZ_ID=$(aws efs describe-mount-targets --file-system-id $SOURCE_EFS | jq -r ".MountTargets[0].AvailabilityZoneId")
export EFS_SUBNET_ID=$(aws efs describe-mount-targets --file-system-id $SOURCE_EFS | jq -r ".MountTargets[0].SubnetId")
export EFS_MOUNT_TARG_ID=$(aws efs describe-mount-targets --file-system-id $SOURCE_EFS | jq -r ".MountTargets[0].MountTargetId")
export EFS_SG_IDS=$(aws efs describe-mount-target-security-groups --mount-target-id $EFS_MOUNT_TARG_ID | jq -r '.SecurityGroups[]')

aws efs create-mount-target \
--file-system-id $TARGET_EFS \
--subnet-id $EFS_SUBNET_ID \
--security-groups $EFS_SG_IDS
```

4. Create Amazon EFS source and destination locations for the AWS DataSync task.

```
export SOURCE_EFS_ARN=$(aws efs describe-file-systems --file-system-id $SOURCE_EFS | jq -r ".FileSystems[0].FileSystemArn")
export TARGET_EFS_ARN=$(aws efs describe-file-systems --file-system-id $TARGET_EFS | jq -r ".FileSystems[0].FileSystemArn")
export EFS_SUBNET_ID_ARN=$(aws ec2 describe-subnets --subnet-ids $EFS_SUBNET_ID | jq -r ".Subnets[0].SubnetArn")
export ACCOUNT_ID=$(aws ec2 describe-security-groups --group-id $EFS_SG_IDS | jq -r ".SecurityGroups[0].OwnerId")
```

```
export EFS_SG_ID_ARN=arn:aws:ec2:$REGION:$ACCOUNT_ID:security-group/$EFS_SG_IDS

export SOURCE_LOCATION_ARN=$(aws datasync create-location-efs --subdirectory
  "/" --efs-filesystem-arn $SOURCE_EFS_ARN --ec2-config SubnetArn=
$EFS_SUBNET_ID_ARN,SecurityGroupArns=$EFS_SG_ID_ARN --region $REGION | jq -r
".LocationArn")
export DESTINATION_LOCATION_ARN=$(aws datasync create-location-efs --
subdirectory "/" --efs-filesystem-arn $TARGET_EFS_ARN --ec2-config SubnetArn=
$EFS_SUBNET_ID_ARN,SecurityGroupArns=$EFS_SG_ID_ARN --region $REGION | jq -r
".LocationArn")
```

5. Allow traffic between the source and target network file system (NFS) mounts. When a new domain is created, SageMaker AI creates 2 security groups.

- NFS inbound security group with only inbound traffic.
- NFS outbound security group with only outbound traffic.

The source and target NFS are placed inside the same security groups. You can allow traffic between these mounts from the AWS Management Console or AWS CLI.

- Allow traffic from the AWS Management Console
  1. Sign in to the AWS Management Console and open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
  2. Choose **Security Groups**.
  3. Search for the existing domain's ID on the **Security Groups** page.

d-xxxxxxx

The results should return two security groups that include the domain ID in the name.

- **security-group-for-inbound-nfs-*domain-id***
  - **security-group-for-outbound-nfs-*domain-id***
4. Select the inbound security group ID. This opens a new page with details about the security group.
  5. Select the **Outbound Rules** tab.
  6. Select **Edit outbound rules**.

7. Update the existing outbound rules or add a new outbound rule with the following values:
    - **Type:** NFS
    - **Protocol:** TCP
    - **Port range:** 2049
    - **Destination:** security-group-for-outbound-nfs-*domain-id* | *security-group-id*
  8. Choose **Save rules**.
  9. Select the **Inbound Rules** tab.
  10. Select **Edit inbound rules**.
  11. Update the existing inbound rules or add a new outbound rule with the following values:
    - **Type:** NFS
    - **Protocol:** TCP
    - **Port range:** 2049
    - **Destination:** security-group-for-outbound-nfs-*domain-id* | *security-group-id*
  12. Choose **Save rules**.
- Allow traffic from the AWS CLI
    1. Update the security group inbound and outbound rules with the following values:
      - **Protocol:** TCP
      - **Port range:** 2049
      - **Group ID:** Inbound security group ID or outbound security group ID

```
export INBOUND_SG_ID=$(aws ec2 describe-security-groups --filters "Name=group-name,Values=security-group-for-inbound-nfs-$SOURCE_DOMAIN_ID" | jq -r ".SecurityGroups[0].GroupId")
export OUTBOUND_SG_ID=$(aws ec2 describe-security-groups --filters "Name=group-name,Values=security-group-for-outbound-nfs-$SOURCE_DOMAIN_ID" | jq -r ".SecurityGroups[0].GroupId")

echo "Outbound SG ID: $OUTBOUND_SG_ID | Inbound SG ID: $INBOUND_SG_ID"
aws ec2 authorize-security-group-egress \
--group-id $INBOUND_SG_ID \
--protocol tcp --port 2049 \
--source-group $OUTBOUND_SG_ID
```

```
aws ec2 authorize-security-group-ingress \
--group-id $OUTBOUND_SG_ID \
--protocol tcp --port 2049 \
--source-group $INBOUND_SG_ID
```

2. Add both the inbound and outbound security groups to the source and target Amazon EFS mount targets. This allows traffic between the 2 Amazon EFS mounts.

```
export SOURCE_EFS_MOUNT_TARGET=$(aws efs describe-mount-targets --file-
system-id $SOURCE_EFS | jq -r ".MountTargets[0].MountTargetId")
export TARGET_EFS_MOUNT_TARGET=$(aws efs describe-mount-targets --file-
system-id $TARGET_EFS | jq -r ".MountTargets[0].MountTargetId")

aws efs modify-mount-target-security-groups \
--mount-target-id $SOURCE_EFS_MOUNT_TARGET \
--security-groups $INBOUND_SG_ID $OUTBOUND_SG_ID

aws efs modify-mount-target-security-groups \
--mount-target-id $TARGET_EFS_MOUNT_TARGET \
--security-groups $INBOUND_SG_ID $OUTBOUND_SG_ID
```

6. Create a AWS DataSync task. This returns a task ARN that can be used to run the task on-demand or as part of a regular cadence.

```
export
EXTRA_XFER_OPTIONS='VerifyMode=ONLY_FILES_TRANSFERRED,OverwriteMode=ALWAYS,Atime=NONE,Mtime=NONE'
export DATASYNC_TASK_ARN=$(aws datasync create-task --source-location-arn
$SOURCE_LOCATION_ARN --destination-location-arn $DESTINATION_LOCATION_ARN --name
"SMEFS_to_CustomEFS_Sync" --region $REGION --options $EXTRA_XFER_OPTIONS | jq -r
".TaskArn")
```

7. Start a AWS DataSync task to automatically copy data from the source Amazon EFS to the target Amazon EFS mount. This does not retain the file's POSIX permissions, which allows users to read from the target Amazon EFS mount, but not write to it.

```
aws datasync start-task-execution --task-arn $DATASYNC_TASK_ARN
```

8. Mount the target Amazon EFS volume on the domain at the root level.

```
aws sagemaker update-domain --domain-id $SOURCE_DOMAIN_ID \
```

```
--default-user-settings '{"CustomFileSystemConfigs": [{"EFSFileSystemConfig": {"FileSystemId": """$TARGET_EFS""", "FileSystemPath": "/"}}]}'
```

9. Overwrite every user profile with a `FileSystemPath` prefix. The prefix includes the user's UID, which is created by SageMaker AI. This ensures users only have access to their data and prevents cross-pollination. When a space is created in the domain and the target Amazon EFS volume is mounted to the application, the user's prefix overwrites the domain prefix. As a result, SageMaker AI only mounts the `/user-id` directory on the user's application.

```
aws sagemaker list-user-profiles --domain-id $SOURCE_DOMAIN_ID | jq -r  
'.UserProfiles[] | "\(.UserName)"' | while read user; do  
export uid=$(aws sagemaker describe-user-profile --domain-id $SOURCE_DOMAIN_ID --  
user-profile-name $user | jq -r ".HomeEfsFileSystemUid")  
echo "$user $uid"  
aws sagemaker update-user-profile --domain-id $SOURCE_DOMAIN_ID --user-profile-  
name $user --user-settings '{"CustomFileSystemConfigs": [{"EFSFileSystemConfig": {"FileSystemId": """$TARGET_EFS""", "FileSystemPath": """/$uid/"""}}]}'  
done
```

10. Users can then select the custom Amazon EFS filesystem when launching an application. For more information, see [JupyterLab user guide](#) or [Launch a Code Editor application in Studio](#).

## Use Amazon S3 to migrate data

In this approach, you use an Amazon EFS-to-Amazon S3 AWS DataSync task to copy the contents of a Studio Classic Amazon EFS volume to an Amazon S3 bucket once or in a regular cadence, then create a lifecycle configuration to copy the user's data from Amazon S3 to their private space's Amazon EBS volume.

### Note

This approach only works for domains that have internet access.

1. Set the source Amazon EFS volume ID from the domain containing the data that you are migrating.

```
timestamp=$(date +%Y%m%d%H%M%S)  
export SOURCE_DOMAIN_ID="domain-id"  
export REGION="region"
```

```
export ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
export EFS_ID=$(aws sagemaker describe-domain --domain-id $SOURCE_DOMAIN_ID | jq -r
'.HomeEfsFileSystemId')
```

- Set the target Amazon S3 bucket name. For information about creating an Amazon S3 bucket, see [Creating a bucket](#). The bucket used must have a CORS policy as described in [\(Optional\) Update your CORS policy to access Amazon S3 buckets](#). Users in the domain must also have permissions to access the Amazon S3 bucket.

In this example, we are copying files to a prefix named studio-new. If you are using a single Amazon S3 bucket to migrate multiple domains, use the studio-new/<domain-id> prefix to restrict permissions to the files using IAM.

```
export BUCKET_NAME=s3-bucket-name
export S3_DESTINATION_PATH=studio-new
```

- Create a trust policy that gives AWS DataSync permissions to assume the execution role of your account.

```
export TRUST_POLICY=$(cat <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "datasync.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "$ACCOUNT_ID"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:datasync:$REGION:$ACCOUNT_ID:*
                }
            }
        }
    ]
}
EOF
)
```

#### 4. Create an IAM role and attach the trust policy.

```
export timestamp=$(date +%Y%m%d%H%M%S)
export ROLE_NAME="DataSyncS3Role-$timestamp"

aws iam create-role --role-name $ROLE_NAME --assume-role-policy-document
"$TRUST_POLICY"
aws iam attach-role-policy --role-name $ROLE_NAME --policy-arn
arn:aws:iam::aws:policy/AmazonS3FullAccess
echo "Attached IAM Policy AmazonS3FullAccess"
aws iam attach-role-policy --role-name $ROLE_NAME --policy-arn
arn:aws:iam::aws:policy/AmazonSageMakerFullAccess
echo "Attached IAM Policy AmazonSageMakerFullAccess"
export ROLE_ARN=$(aws iam get-role --role-name $ROLE_NAME --query 'Role.Arn' --
output text)
echo "Created IAM Role $ROLE_ARN"
```

#### 5. Create a security group to give access to the Amazon EFS location.

```
export EFS_ID=$(aws efs describe-file-systems --file-system-id $EFS_ID | jq -r
'.FileSystems[0].FileSystemArn' )
export EFS_SUBNET_ID=$(aws efs describe-mount-targets --file-system-id $EFS_ID | jq
-r '.MountTargets[0].SubnetId')
export EFS_VPC_ID=$(aws efs describe-mount-targets --file-system-id $EFS_ID | jq -r
'.MountTargets[0].VpcId')
export MOUNT_TARGET_ID=$(aws efs describe-mount-targets --file-system-id $EFS_ID | jq -r
'.MountTargets[0].MountTargetId')
export EFS_SECURITY_GROUP_ID=$(aws efs describe-mount-target-security-groups --
mount-target-id $MOUNT_TARGET_ID | jq -r '.SecurityGroups[0]')
export EFS_SUBNET_ARN=$(aws ec2 describe-subnets --subnet-ids $EFS_SUBNET_ID | jq -
r '.Subnets[0].SubnetArn')
echo "Subnet ID: $EFS_SUBNET_ID"
echo "Security Group ID: $EFS_SECURITY_GROUP_ID"
echo "Subnet ARN: $EFS_SUBNET_ARN"

timestamp=$(date +%Y%m%d%H%M%S)
sg_name="datasync-sg-$timestamp"
export DATASYNC_SG_ID=$(aws ec2 create-security-group --vpc-id $EFS_VPC_ID --group-
name $sg_name --description "DataSync SG" --output text --query 'GroupId')
aws ec2 authorize-security-group-egress --group-id $DATASYNC_SG_ID --protocol tcp
--port 2049 --source-group $EFS_SECURITY_GROUP_ID
aws ec2 authorize-security-group-ingress --group-id $EFS_SECURITY_GROUP_ID --
protocol tcp --port 2049 --source-group $DATASYNC_SG_ID
```

```
export DATASYNC_SG_ARN="arn:aws:ec2:$REGION:$ACCOUNT_ID:security-group/$DATASYNC_SG_ID"
echo "Security Group ARN: $DATASYNC_SG_ARN"
```

## 6. Create a source Amazon EFS location for the AWS DataSync task.

```
export SOURCE_ARN=$(aws datasync create-location-efs --efs-filesystem-arn $EFS_ARN
--ec2-config "{\"SubnetArn\": \"$EFS_SUBNET_ARN\", \"SecurityGroupArns\":[$DATASYNC_SG_ARN]}\" | jq -r '.LocationArn')
echo "Source Location ARN: $SOURCE_ARN"
```

## 7. Create a target Amazon S3 location for the AWS DataSync task.

```
export BUCKET_ARN="arn:aws:s3:::$BUCKET_NAME"
export DESTINATION_ARN=$(aws datasync create-location-s3 --s3-bucket-arn
$BUCKET_ARN --s3-config "{\"BucketAccessRoleArn\": \"$ROLE_ARN\"} --subdirectory
$S3_DESTINATION_PATH | jq -r '.LocationArn')
echo "Destination Location ARN: $DESTINATION_ARN"
```

## 8. Create a AWS DataSync task.

```
export TASK_ARN=$(aws datasync create-task --source-location-arn $SOURCE_ARN --
destination-location-arn $DESTINATION_ARN | jq -r '.TaskArn')
echo "DataSync Task: $TASK_ARN"
```

## 9. Start the AWS DataSync task. This task automatically copies data from the source Amazon EFS volume to the target Amazon S3 bucket. Wait for the task to be complete.

```
aws datasync start-task-execution --task-arn $TASK_ARN
```

## 10. Check the status of the AWS DataSync task to verify that it is complete. Pass the ARN returned in the previous step.

```
export TASK_EXEC_ARN=datasync-task-arn
echo "Task execution ARN: $TASK_EXEC_ARN"
export STATUS=$(aws datasync describe-task-execution --task-execution-arn
$TASK_EXEC_ARN | jq -r '.Status')
echo "Execution status: $STATUS"
while [ "$STATUS" = "QUEUED" ] || [ "$STATUS" = "LAUNCHING" ] || [ "$STATUS" =
"PREPARING" ] || [ "$STATUS" = "TRANSFERRING" ] || [ "$STATUS" = "VERIFYING" ]; do
    STATUS=$(aws datasync describe-task-execution --task-execution-arn
$TASK_EXEC_ARN | jq -r '.Status')
```

```
if [ $? -ne 0 ]; then
    echo "Error Running DataSync Task"
    exit 1
fi
echo "Execution status: $STATUS"
sleep 30
done
```

11. After the AWS DataSync task is complete, clean up the previously created resources.

```
aws datasync delete-task --task-arn $TASK_ARN
echo "Deleted task $TASK_ARN"
aws datasync delete-location --location-arn $SOURCE_ARN
echo "Deleted location source $SOURCE_ARN"
aws datasync delete-location --location-arn $DESTINATION_ARN
echo "Deleted location source $DESTINATION_ARN"
aws iam detach-role-policy --role-name $ROLE_NAME --policy-arn
arn:aws:iam::aws:policy/AmazonS3FullAccess
aws iam detach-role-policy --role-name $ROLE_NAME --policy-arn
arn:aws:iam::aws:policy/AmazonSageMakerFullAccess
aws iam delete-role --role-name $ROLE_NAME
echo "Deleted IAM Role $ROLE_NAME"
echo "Wait 5 minutes for the elastic network interface to detach..."
start_time=$(date +%s)
while [[ $(( $(date +%s) - start_time )) -lt 300 ]]; do
    sleep 1
done
aws ec2 revoke-security-group-ingress --group-id $EFS_SECURITY_GROUP_ID --protocol
tcp --port 2049 --source-group $DATASYNC_SG_ID
echo "Revoked Ingress from $EFS_SECURITY_GROUP_ID"
aws ec2 revoke-security-group-egress --group-id $DATASYNC_SG_ID --protocol tcp --
port 2049 --source-group $EFS_SECURITY_GROUP_ID
echo "Revoked Egress from $DATASYNC_SG_ID"
aws ec2 delete-security-group --group-id $DATASYNC_SG_ID
echo "Deleted DataSync SG $DATASYNC_SG_ID"
```

12. From your local machine, create a file named `on-start.sh` with the following content. This script copies the user's Amazon EFS home directory in Amazon S3 to the user's Amazon EBS volume in Studio and creates a prefix for each user profile.

```
#!/bin/bash
set -eo pipefail
```

```
sudo apt-get install -y jq

# Studio Variables
DOMAIN_ID=$(cat /opt/ml/metadata/resource-metadata.json | jq -r '.DomainId')
SPACE_NAME=$(cat /opt/ml/metadata/resource-metadata.json | jq -r '.SpaceName')
USER_PROFILE_NAME=$(aws sagemaker describe-space --domain-id=$DOMAIN_ID --space-name=$SPACE_NAME | jq -r '.OwnershipSettings.OwnerUserName')

# S3 bucket to copy from
BUCKET=s3-bucket-name
# Subfolder in bucket to copy
PREFIX=studio-new

# Getting HomeEfsFileSystemUid for the current user-profile
EFS_FOLDER_ID=$(aws sagemaker describe-user-profile --domain-id $DOMAIN_ID --user-profile-name $USER_PROFILE_NAME | jq -r '.HomeEfsFileSystemUid')

# Local destination directory
DEST=./studio-classic-efs-backup
mkdir -p $DEST

echo "Bucket: s3://$BUCKET/$PREFIX/$EFS_FOLDER_ID/"
echo "Destination $DEST/"
echo "Excluding .*"
echo "Excluding .*/"

aws s3 cp s3://$BUCKET/$PREFIX/$EFS_FOLDER_ID/ $DEST/ \
    --exclude ".*" \
    --exclude "**/.*" \
    --recursive
```

13. Convert your script into base64 format. This requirement prevents errors that occur from spacing and line break encoding. The script type can be either JupyterLab or CodeEditor.

```
export LCC_SCRIPT_NAME='studio-classic-sync'
export SCRIPT_FILE_NAME='on-start.sh'
export SCRIPT_TYPE='JupyterLab-or-CodeEditor'
LCC_CONTENT=`openssl base64 -A -in ${SCRIPT_FILE_NAME}`
```

14. Verify the following before you use the script:

- The Amazon EBS volume is large enough to store the objects that you're exporting.

- You aren't migrating hidden files and folders, such as `.bashrc` and `.condarc` if you aren't intending to do so.
- The AWS Identity and Access Management (IAM) execution role that's associated with Studio user profiles has the policies configured to access only the respective home directory in Amazon S3.

15. Create a lifecycle configuration using your script.

```
aws sagemaker create-studio-lifecycle-config \
--studio-lifecycle-config-name $LCC_SCRIPT_NAME \
--studio-lifecycle-config-content $LCC_CONTENT \
--studio-lifecycle-config-app-type $SCRIPT_TYPE
```

16. Attach the LCC to your domain.

```
aws sagemaker update-domain \
--domain-id $SOURCE_DOMAIN_ID \
--default-user-settings \
  {"JupyterLabAppSettings": \
    {"LifecycleConfigArns": \
      [ \
        "lifecycle-config-arn" \
      ] \
    } \
  }'
```

17. Users can then select the LCC script when launching an application. For more information, see [JupyterLab user guide](#) or [Launch a Code Editor application in Studio](#). This automatically syncs the files from Amazon S3 to the Amazon EBS storage for the user's space.

## Migrate data flows from Data Wrangler

If you have previously used Amazon SageMaker Data Wrangler in Amazon SageMaker Studio Classic for data preparation tasks, you can migrate to the new Amazon SageMaker Studio and access the latest version of Data Wrangler in Amazon SageMaker Canvas. Data Wrangler in SageMaker Canvas provides you with an enhanced user experience and access to the latest features, such as a natural language interface and faster performance.

You can onboard to SageMaker Canvas at any time to begin using the new Data Wrangler experience. For more information, see [Getting started with using Amazon SageMaker Canvas](#).

If you have data flow files saved in Studio Classic that you were previously working on, you can onboard to Studio and then import the flow files into Canvas. You have the following options for migration:

- One-click migration: When you sign in to Canvas, you can use a one-time import option that migrates all of your flow files on your behalf.
- Manual migration: You can manually import your flow files into Canvas. From Studio Classic, either export the files to Amazon S3 or download them to your local machine. Then, you sign in to the SageMaker Canvas application, import the flow files, and continue your data preparation tasks.

The following guide describes the prerequisites to migration and how to migrate your data flow files using either the one-click or manual option.

## Prerequisites

Review the following prerequisites before you begin migrating your flow files.

### Step 1. Migrate the domain and grant permissions

Before migrating data flow files, you need to follow specific steps of the [Migration from Amazon SageMaker Studio Classic](#) guide to ensure that your user profile's AWS IAM execution role has the required permissions. Follow the [Prerequisites](#) and [Migrate the UI from Studio Classic to Studio](#) before proceeding, which describe how to grant the required permissions, configure Studio as the new experience, and migrate your existing domain.

Specifically, you must have permissions to create a SageMaker Canvas application and use the SageMaker Canvas data preparation features. To obtain these permissions, you can either:

- Add the [AmazonSageMakerCanvasDataPrepFullAccess](#) policy to your IAM role, or
- Attach a least-permissions policy, as shown in the **(Optional) Migrate from Data Wrangler in Studio Classic to SageMaker Canvas** section of the page [Migrate the UI from Studio Classic to Studio](#).

Make sure to use the same user profile for both Studio and SageMaker Canvas.

After completing the prerequisites outlined in the migration guide, you should have a new domain with the required permissions to access SageMaker Canvas through Studio.

## Step 2. (Optional) Prepare an Amazon S3 location

If you are doing a manual migration and plan to use Amazon S3 to transfer your flow files instead of using the local download option, you should have an Amazon S3 bucket in your account that you'd like to use for storing the flow files.

### One-click migration method

SageMaker Canvas offers a one-time import option for migrating your data flows from Data Wrangler in Studio Classic to Data Wrangler in SageMaker Canvas. As long as your Studio Classic and Canvas applications share the same Amazon EFS storage volume, you can migrate in one click from Canvas. This streamlined process eliminates the need for manual export and import steps, and you can import all of your flows at once.

Use the following procedure to migrate all of your flow files:

1. Open your latest version of Studio.
2. In Studio, in the left navigation pane, choose the **Data** dropdown menu.
3. From the navigation options, choose **Data Wrangler**.
4. On the **Data Wrangler** page, choose **Run in Canvas**. If you have successfully set up the permissions, this creates a Canvas application for you. The Canvas application may take a few minutes before it's ready.
5. When Canvas is ready, choose **Open in Canvas**.
6. Canvas opens to the **Data Wrangler** page, and a banner at the top of the page appears that says Import your data flows from Data Wrangler in Studio Classic to Canvas. This is a one time import. Learn more. In the banner, choose **Import All**.

 **Warning**

If you close the banner notification, you won't be able to re-open it or use the one-click migration method anymore.

A pop-up notification appears, indicating that Canvas is importing your flow files from Studio Classic. If the import is fully successful, you receive another notification that X number of flow files were imported, and you can see your flow files on the **Data Wrangler** page of the Canvas application. Any imported flow files that have the same name as existing data flows in your Canvas application are renamed with a postfix. You can open a data flow to verify that it looks as expected.

In case any of your flow files don't import successfully, you receive a notification that the import was either partially successful or failed. Choose **View errors** on the notification message to check the individual error messages for guidance on how to reformat any incorrectly formatted flow files.

After importing your flow files, you should now be able to continue using Data Wrangler to prepare data in SageMaker Canvas.

## Manual migration method

The following sections describe how to manually import your flow files into Canvas in case the one-click migration method didn't work.

### Export the flow files from Studio Classic

#### Note

If you've already migrated your Studio Classic data to Amazon S3 by following the instructions in [\(Optional\) Migrate data from Studio Classic to Studio](#), you can skip this step and go straight to the [Import the flow files into Canvas](#) section in which you import your flow files from the Amazon S3 location where your Studio Classic data is stored.

You can export your flow files by either saving them to Amazon S3 or downloading them to your local machine. When you import your flow files into SageMaker Canvas in the next step, if you choose the local upload option, then you can only upload 20 flow files at a time. If you have a large number of flow files to import, we recommend that you use Amazon S3 instead.

Follow the instructions in either [Method 1: Use Amazon S3 to transfer flow files](#) or [Method 2: Use your local machine to transfer flow files](#) to proceed.

#### Method 1: Use Amazon S3 to transfer flow files

With this method, you use Amazon S3 as the intermediary between Data Wrangler in Studio Classic and Data Wrangler in SageMaker Canvas (accessed through the latest version of Studio). You export the flow files from Studio Classic to Amazon S3, and then in the next step, you access Canvas through Studio and import the flow files from Amazon S3.

Make sure that you have an Amazon S3 bucket prepared as the storage location for the flow files.

Use the following procedure to export your flow files from Studio Classic to Amazon S3:

1. Open Studio Classic.
2. Open a new terminal by doing the following:
  - a. On the top navigation bar, choose **File**.
  - b. In the context menu, hover over **New**, and then select **Terminal**.
3. By default, the terminal should open in your home directory. Navigate to the folder that contains all of the flow files that you want to migrate.
4. Use the following command to synchronize all of the flow files to the specified Amazon S3 location. Replace **{bucket-name}** and **{folder}** with the path to your desired Amazon S3 location. For more information about the command and parameters, see the [sync](#) command in the AWS AWS CLI Command Reference.

```
aws s3 sync . s3://{bucket-name}/{folder}/ --exclude "*.*" --include "*.flow"
```

If you are using your own AWS KMS key, then use the following command instead to synchronize the files, and specify your KMS key ID. Make sure that the user's IAM execution role (which should be the same role used in **Step 1. Migrate the domain and grant permissions** of the preceding [Prerequisites](#)) has been granted access to use the KMS key.

```
aws s3 sync . s3://{bucket-name}/{folder}/ --exclude "*.*" --include "*.flow" --sse-kms-key-id {your-key-id}
```

Your flow files should now be exported. You can check your Amazon S3 bucket to make sure that the flow files synchronized successfully.

To import these files in the latest version of Data Wrangler, follow the steps in [Import the flow files into Canvas](#).

### Method 2: Use your local machine to transfer flow files

With this method, you download the flow files from Studio Classic to your local machine. You can download the files directly, or you can compress them as a zip archive. Then, you unpack the zip file locally (if applicable), sign in to Canvas, and import the flow files by uploading them from your local machine.

Use the following procedure to download your flow files from Studio Classic:

1. Open Studio Classic.

2. (Optional) If you want to compress multiple flow files into a zip archive and download them all at once, then do the following:
  - a. On the top navigation bar of Studio Classic, choose **File**.
  - b. In the context menu, hover over **New**, and then select **Terminal**.
  - c. By default, the terminal opens in your home directory. Navigate to the folder that contains all of the flow files that you want to migrate.
  - d. Use the following command to pack the flow files in the current directory as a zip. The command excludes any hidden files:

```
find . -not -path "*/.*" -name "*.flow" -print0 | xargs -0 zip my_archive.zip
```

3. Download the zip archive or individual flow files to your local machine by doing the following:
  - a. In the left navigation pane of Studio Classic, choose **File Browser**.
  - b. Find the file you want to download in the file browser.
  - c. Right click the file, and in the context menu, select **Download**.

The file should download to your local machine. If you packed them as a zip archive, extract the files locally. After the files are extracted, to import these files in the latest version of Data Wrangler, follow the steps in [Import the flow files into Canvas](#).

## Import the flow files into Canvas

After exporting your flow files, access Canvas through Studio and import the files.

Use the following procedure to import flow files into Canvas:

1. Open your latest version of Studio.
2. In Studio, in the left navigation pane, choose the **Data** dropdown menu.
3. From the navigation options, choose **Data Wrangler**.
4. On the **Data Wrangler** page, choose **Run in Canvas**. If you have successfully set up the permissions, this creates a Canvas application for you. The Canvas application may take a few minutes before it's ready.
5. When Canvas is ready, choose **Open in Canvas**.
6. Canvas opens to the **Data Wrangler** page. In the top pane, choose **Import data flows**.

7. For **Data source**, choose either **Amazon S3** or **Local upload**.
8. Select your flow files from your Amazon S3 bucket, or upload the files from your local machine.

 **Note**

For local upload, you can upload a maximum of 20 flow files at a time. For larger imports, use Amazon S3. If you select a folder to import, any flow files in sub-folders are also imported.

9. Choose **Import data**.

If the import was successful, you receive a notification that X number of flow files were successfully imported.

In case your flow files don't import successfully, you receive a notification in the SageMaker Canvas application. Choose **View errors** on the notification message to check the individual error messages for guidance on how to reformat any incorrectly formatted flow files.

After your flow files are done importing, go to the **Data Wrangler** page of the SageMaker Canvas application to view your data flows. You can try opening a data flow to verify that it looks as expected.

## Launch Amazon SageMaker Studio

 **Important**

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

## **⚠️ Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

This page's topics demonstrate how to launch Amazon SageMaker Studio from the Amazon SageMaker AI console and the AWS Command Line Interface (AWS CLI).

## Topics

- [Prerequisites](#)
- [Launch from the Amazon SageMaker AI console](#)
- [Launch using the AWS CLI](#)

## Prerequisites

Before you begin, complete the following prerequisites:

- Onboard to a SageMaker AI domain with Studio access. If you don't have permissions to set Studio as the default experience for your domain, contact your administrator. For more information, see [Amazon SageMaker AI domain overview](#).
- Update the AWS CLI by following the steps in [Installing the current AWS CLI Version](#).
- From your local machine, run `aws configure` and provide your AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).

## Launch from the Amazon SageMaker AI console

Complete the following procedure to launch Studio from the Amazon SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation pane, choose Studio.
3. From the Studio landing page, select the domain and user profile for launching Studio.
4. Choose **Open Studio**.
5. To launch Studio, choose **Launch personal Studio**.

## Launch using the AWS CLI

This section demonstrates how to launch Studio using the AWS CLI. The procedure to access Studio using the AWS CLI depends if the domain uses AWS Identity and Access Management (IAM) authentication or AWS IAM Identity Center authentication. You can use the AWS CLI to launch Studio by creating a presigned domain URL when your domain uses IAM authentication. For information about launching Studio with IAM Identity Center authentication, see [Use custom setup for Amazon SageMaker AI](#).

### Launch if Studio is the default experience

The following code snippet demonstrates how to launch Studio from the AWS CLI using a presigned domain URL if Studio is the default experience. For more information, see [create-presigned-domain-url](#).

```
aws sagemaker create-presigned-domain-url \
--region region \
--domain-id domain-id \
--user-profile-name user-profile-name \
--session-expiration-duration-in-seconds 43200
```

### Launch if Amazon SageMaker Studio Classic is your default experience

The following code snippet demonstrates how to launch Studio from the AWS CLI using a presigned domain URL if Studio Classic is the default experience. For more information, see [create-presigned-domain-url](#).

```
aws sagemaker create-presigned-domain-url \
--region region \
--domain-id domain-id \
--user-profile-name user-profile-name \
--session-expiration-duration-in-seconds 43200 \
--landing-uri studio::
```

## Amazon SageMaker Studio UI overview

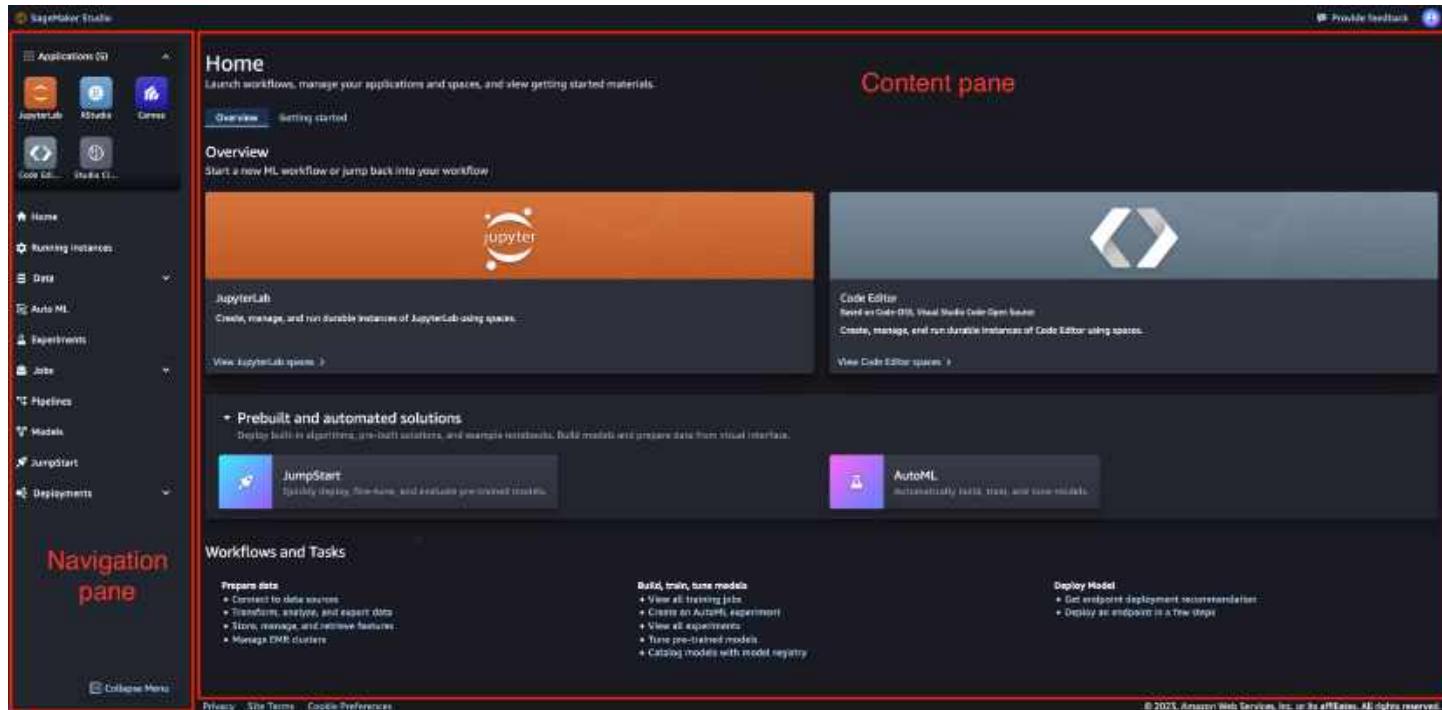
### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the

updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

The Amazon SageMaker Studio user interface is split into three distinct parts. This page gives information about the distinct parts and their components.

- **Navigation bar**— This section of the UI includes the URL, breadcrumbs, notifications, and user options.
- **Navigation pane**— This section of the UI includes a list of the applications that are supported in Studio and options for the main workflows in Studio.
- **Content pane**— The main working area that displays the current page of the Studio UI that you have open.



## Topics

- [Amazon SageMaker Studio navigation bar](#)
- [Amazon SageMaker Studio navigation pane](#)
- [Studio content pane](#)

## Amazon SageMaker Studio navigation bar

The navigation bar of the Studio UI includes the URL, breadcrumbs, notifications, and user options.

### URL Structure

The URL of Studio changes as you navigate the UI. When you navigate to a different page in the UI, the URL changes to reflect that page. With the updated URL, you open any page in the Studio UI directly without navigating to the landing page first.

### Breadcrumbs

As you navigate through the Studio UI, the breadcrumbs keep track of the parent pages of the current page. By choosing one of these breadcrumbs, you can navigate to parent pages in the UI.

### Notifications

The notifications section of the UI gives information about important changes to Studio, updates to applications, and issues to resolve.

### User options

Choose the user options icon



)

to get information about the user profile that is currently using Studio, and gives the option to sign out of Studio.

## Amazon SageMaker Studio navigation pane

### Navigation pane

The navigation pane of the UI includes a list of the applications that are supported in Studio. It also provides options for the main workflows in Studio.

This section of the UI can be used in an expanded or collapsed state. To change whether the section is expanded or collapsed, select the **Collapse** icon



).

### Applications

The applications section lists the applications that are available in Studio. If you choose one of the application types, you are directed to the landing page for that application.

## Workflows

The list of workflows includes all of the available actions that you can take in Studio. Choose one of the options to navigate to the landing page for that workflow. If there are multiple workflows available for that option, choosing the option opens a dropdown menu where you can select the desired landing page.

The following list describes the options and provides a link for more information.

- **Home**— The main landing page with an overview, getting started, and what's new.
- **Running instances**— All of the instances that are currently running in Studio. For more information, see [View your Studio running instances, applications, and spaces](#).
- **Data**— Data preparation options where you can collaborate to store, explore, prepare, transform, and share your data.
  - For more information about Amazon SageMaker Data Wrangler, see [Data preparation](#).
  - For more information about Amazon SageMaker Feature Store, see [Create, store, and share features with Feature Store](#).
  - For more information about Amazon EMR clusters, see [Data preparation using Amazon EMR](#).
- **Auto ML**— Automatically build, train, tune, and deploy machine learning (ML) models. For more information, see [Amazon SageMaker Canvas](#).
- **Experiments**— Create, manage, analyze, and compare your machine learning experiments using Amazon SageMaker Experiments. For more information, see [Amazon SageMaker Experiments in Studio Classic](#).
- **Jobs**— View jobs created in Studio.
  - For more information about training, see [Model training](#).
  - For more information about model evaluation, see [Understand options for evaluating large language models with SageMaker Clarify](#).
- **Pipelines**— Automate your ML workflow with Amazon SageMaker Pipelines, which provides resources to help you build, track, and manage your pipeline resources. For more information, see [Pipelines](#).
- **Models**— Organize your models into groups and collections in the model registry, where you can manage model versions, view metadata, and deploy models to production. For more information, see [Model Registration Deployment with Model Registry](#).

- **JumpStart**– Amazon SageMaker JumpStart provides pretrained, open-source models for a wide range of problem types to help you get started with machine learning. For more information, see [SageMaker JumpStart pretrained models](#).
- **Deployments**– Deploy your machine learning (ML) models for inference.
  - For more information about Amazon SageMaker Inference Recommender, see [Amazon SageMaker Inference Recommender](#).
  - For more information about endpoints, see [Deploy models for inference](#).

## Studio content pane

The main working area is also called the content pane. It displays the current page of the Studio UI that you have open.

### Studio home page

The Studio home page is the primary landing page in the main working area. The home page includes two distinct tabs. There is an **Overview** tab and a **Getting started** tab.

#### Overview

The **Overview** tab includes options to start spaces for popular application types, get started with pre-built and automated solutions for ML workflows, and links to common tasks in the Studio UI.

#### Getting started

The **Getting started** tab includes information, guidance, and resources on how to begin with Studio. This includes a guided tour of the Studio UI, a link to documentation about Studio, and a selection of quick tips.

## Amazon EFS auto-mounting in Studio

Amazon SageMaker AI supports automatically mounting a folder in an Amazon EFS volume for each user in a domain. Using this folder, users can share data between their own private spaces. However, users cannot share data with other users in the domain. Users only have access to their own folder.

The user's folder can be accessed through a folder named `user-default-efs`. This folder is present in the `$HOME` directory of the Studio application.

For information about opting out of Amazon EFS auto-mounting, see [Opt out of Amazon EFS auto-mounting](#).

Amazon EFS auto-mounting also facilitates the migration of data from Studio Classic to Studio. For more information, see [\(Optional\) Migrate data from Studio Classic to Studio](#).

## Access point information

When auto-mounting is activated, SageMaker AI uses an Amazon EFS access point to facilitate access to the data in the Amazon EFS volume. For more information about access points, see [Working with Amazon EFS access points](#). SageMaker AI creates a unique access point for each user profile in the domain during user profile creation or during application creation for an existing user profile. The POSIX user value of the access point matches the HomeEfsFileSystemUid value of the user profile that SageMaker AI creates the access point for. To get the value of the user, see [DescribeUserProfile](#). The root directory path is also set to the same value as the POSIX user value.

SageMaker AI sets the permissions of the new directory to the following values:

- Owner user ID: *POSIX user value*
- Owner group ID: 0
- Permissions 700

The access point is required to access the Amazon EFS volume. As a result, you cannot delete or update the access point without losing access to the Amazon EFS volume.

## Error resolution

If SageMaker AI encounters an issue when auto-mounting the Amazon EFS user folder during application creation, the application is still created. However, in this case, SageMaker AI creates a file named `error.txt` instead of mounting the Amazon EFS folder. This file describes the error encountered, as well as steps to resolve it. SageMaker AI creates the `error.txt` file in the `user-default-efs` folder located in the `$HOME` directory of the application.

## Opt out of Amazon EFS auto-mounting

You can opt-out of Amazon SageMaker AI auto-mounting Amazon EFS user folders during domain and user profile creation or for an existing domain or user profile.

## Opt out during domain creation

You can opt out of Amazon EFS auto-mounting when creating a domain using either the console or the AWS Command Line Interface.

### Console

Complete the following steps to opt out of Amazon EFS auto-mounting when creating a domain from the console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Complete the steps in [Use custom setup for Amazon SageMaker AI](#) with the following modification to set up a domain.
  - On the **Configure storage** step, turn off **Automatically mount EFS storage and data**.

### AWS CLI

Use the following command to opt out of Amazon EFS auto-mounting during domain creation using the AWS CLI. For more information about creating a domain using the AWS CLI, see [Use custom setup for Amazon SageMaker AI](#).

```
aws --region region sagemaker create-domain \
--domain-name "my-domain-$(date +%s)" \
--vpc-id default-vpc-id \
--subnet-ids subnet-ids \
--auth-mode IAM \
--default-user-settings "ExecutionRole=execution-role-arn,AutoMountHomeEFS=Disabled" \
--default-space-settings "ExecutionRole=execution-role-arn"
```

## Opt out for an existing domain

You can opt out of Amazon EFS auto-mounting for an existing domain using either the console or the AWS CLI.

### Console

Complete the following steps to opt out of Amazon EFS auto-mounting when updating a domain from the console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.

2. On the left navigation under **Admin configurations**, choose **Domains**.
3. On the **Domains** page, select the domain that you want to opt out of Amazon EFS auto-mounting for.
4. On the **Domain details** page, select the **Domain settings** tab.
5. Navigate to the **Storage configurations** section.
6. Select **Edit**.
7. From the **Edit storage settings** page, turn off **Automatically mount EFS storage and data**.
8. Select **Submit**.

## AWS CLI

Use the following command to opt out of Amazon EFS auto-mounting while updating an existing domain using the AWS CLI.

```
aws --region region sagemaker update-domain \  
--domain-id domain-id \  
--default-user-settings "AutoMountHomeEFS=Disabled"
```

## Opt out during user profile creation

You can opt out of Amazon EFS auto-mounting when creating a user profile using either the console or the AWS CLI.

### Console

Complete the following steps to opt out of Amazon EFS auto-mounting when creating a user profile from the console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Complete the steps in [Add user profiles](#) with the following modification to create a user profile.
  - On the **Data and Storage** step, turn off **Inherit settings from domain**. This allows the user to have a different value than the defaults that are set for the domain.
  - Turn off **Automatically mount EFS storage and data**.

## AWS CLI

Use the following command to opt out of Amazon EFS auto-mounting during user profile creation using the AWS CLI. For more information about creating a user profile using the AWS CLI, see [Add user profiles](#).

```
aws --region region sagemaker create-user-profile \  
--domain-id domain-id \  
--user-profile-name "user-profile-$(date +%s)" \  
--user-settings "ExecutionRole=arn:aws:iam::account-id:role/execution-role-name,AutoMountHomeEFS=Enabled/Disabled/DefaultAsDomain"
```

## Opt out for an existing user profile

You can opt out of Amazon EFS auto-mounting for an existing user profile using either the console or the AWS CLI.

### Console

Complete the following steps to opt out of Amazon EFS auto-mounting when updating a user profile from the console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation under **Admin configurations**, choose **Domains**.
3. On the **Domains** page, select the domain containing the user profile that you want to opt out of Amazon EFS auto-mounting for.
4. On the **Domains details** page, select the **User profiles** tab.
5. Select the user profile to update.
6. From the **User Details** tab, navigate to the **AutoMountHomeEFS** section.
7. Select **Edit**.
8. From the **Edit storage settings** page, turn off **Inherit settings from domain**. This allows the user to have a different value than the defaults that are set for the domain.
9. Turn off **Automatically mount EFS storage and data**.
10. Select **Submit**.

## AWS CLI

Use the following command to opt out of Amazon EFS auto-mounting while updating an existing user profile using the AWS CLI.

```
aws --region region sagemaker update-user-profile \  
--domain-id domain-id \  
--user-profile-name user-profile-name \  
--user-settings "AutoMountHomeEFS=DefaultAsDomain"
```

## Idle shutdown

Amazon SageMaker AI supports shutting down idle resources to manage costs and prevent cost overruns due to cost accrued by idle, billable resources. It accomplishes this by detecting an app's idle state and performing an app shutdown when idle criteria are met.

SageMaker AI supports idle shutdown for the following applications. Idle shutdown must be set for each application type independently.

- JupyterLab
- Code Editor, based on Code-OSS, Visual Studio Code - Open Source

Idle shutdown can be set at either the domain or user profile level. When idle shutdown is set at the domain level, the idle shutdown settings apply to all applications created in the domain. When set at the user profile level, the idle shutdown settings apply only to the specific users that they are set for. User profile settings override domain settings.

### Note

Idle shutdown requires the usage of the SageMaker-distribution (SMD) image with v2.0 or newer. Domains using an older SMD version can't use the feature. These users must use an LCC to manage auto-shutdown instead.

## Definition of idle

Idle shutdown settings only apply when the application becomes idle with no jobs running. SageMaker AI doesn't start the idle shutdown timing until the instance becomes idle. The definition on idle differs based on whether the application type is JupyterLab or Code Editor.

For JupyterLab applications, the instance is considered idle when the following conditions are met:

- No active Jupyter kernel sessions
- No active Jupyter terminal sessions

For Code Editor applications, the instance is considered idle when the following conditions are met:

- No text file or notebook changes
- No files being viewed
- No interaction with the terminal
- No background processes running
- No notebook kernels processing
- No unsaved work

## Set up idle shutdown

The following sections show how to set up idle shutdown from either the console or using the AWS CLI. Idle shutdown can be set at either the domain or user profile level.

### Prerequisites

To use idle shutdown with your application, you must complete the following prerequisites.

- Ensure that your application is using the SageMaker Distribution (SMD) version 2.0. You can select this version during application creation or update the image version of the application after creation. For more information, see [Update the SageMaker Distribution Image](#).
- For applications built with custom images, idle shutdown is supported if your custom image is created with SageMaker Distribution (SMD) version 2.0 or later as the base image. If the custom image is created with a different base image, then you must install the [jupyter-activity-monitor-extension >= 0.3.1](#) extension on the image and attach the image to your Amazon SageMaker AI domain for JupyterLab applications. For more information about custom images for JupyterLab applications, see [Provide users with access to custom images](#). For more information about custom images for Code Editor applications, see [Environment customization using custom images](#).

## From the Console

The following sections show how to enable idle shutdown from the console.

### Add when creating a new domain

1. Create a domain by following the steps in [Use custom setup for Amazon SageMaker AI](#)
2. When configuring the application settings in the domain, navigate to either the Code Editor or JupyterLab section.
3. Select **Enable idle shutdown**.
4. Enter a default idle shutdown time in minutes. This values defaults to 10,080 if no value is entered.
5. (Optional) Select **Allow users to set custom idle shutdown time** to allow users to modify the idle shutdown time.
  - Enter a maximum value that users can set the default idle shutdown time to. You must enter a maximum value. The minimum value is set by Amazon SageMaker AI and must be 60.

### Add to an existing domain

 **Note**

If idle shutdown is set when applications are running, they must be restarted for idle shutdown settings to take effect.

1. Navigate to the domain.
2. Choose the **App Configurations** tab.
3. From the **App Configurations** tab, navigate to either the Code Editor or JupyterLab section.
4. Select **Edit**.
5. Select **Enable idle shutdown**.
6. Enter a default idle shutdown time in minutes. This values defaults to 10,080 if no value is entered.
7. (Optional) Select **Allow users to set custom idle shutdown time** to allow users to modify the idle shutdown time.

- Enter a maximum value that users can set the default idle shutdown time to. You must enter a maximum value. The minimum value is set by Amazon SageMaker AI and must be 60.
8. Select **Submit**.

## Add when creating a new user profile

1. Add a user profile by following the steps at [Add user profiles](#)
2. When configuring the application settings for the user profile, navigate to either the Code Editor or JupyterLab section.
3. Select **Enable idle shutdown**.
4. Enter a default idle shutdown time in minutes. This values defaults to 10,080 if no value is entered.
5. (Optional) Select **Allow users to set custom idle shutdown time** to allow users to modify the idle shutdown time.
  - Enter a maximum value that users can set the default idle shutdown time to. You must enter a maximum value. The minimum value is set by Amazon SageMaker AI and must be 60.
6. Select “Save Changes”.

## Add to an existing user profile

Note: If idle shutdown is set when applications are running, they must be restarted for idle shutdown settings to take effect.

1. Navigate to the user profile.
2. Choose the **App Configurations** tab.
3. From the **App Configurations** tab, navigate to either the Code Editor or JupyterLab section.
4. Select **Edit**.
5. Idle shutdown settings will show domain settings by default if configured for the domain.
6. Select **Enable idle shutdown**.
7. Enter a default idle shutdown time in minutes. This values defaults to 10,080 if no value is entered.
8. (Optional) Select **Allow users to set custom idle shutdown time** to allow users to modify the idle shutdown time.

- Enter a maximum value that users can set the default idle shutdown time to. You must enter a maximum value. The minimum value is set by Amazon SageMaker AI and must be 60.

## 9. Select **Save Changes**.

### From the AWS CLI

The following sections show how to enable idle shutdown using the AWS CLI.

#### Note

To enforce a specific timeout value from the AWS CLI, you must set `IdleTimeoutInMinutes`, `MaxIdleTimeoutInMinutes`, and `MinIdleTimeoutInMinutes` to the same value.

### Domain

The following command shows how to enable idle shutdown when updating an existing domain. To add idle shutdown for a new domain, use the `create-domain` command instead.

#### Note

If idle shutdown is set when applications are running, they must be restarted for idle shutdown settings to take effect.

```
aws sagemaker update-domain --region region --domain-id domain-id \  
--default-user-settings file://default-user-settings.json  
  
## default-user-settings.json example for enforcing the default timeout  
{  
    "JupyterLabAppSettings": {  
        "AppLifecycleManagement": {  
            "IdleSettings": {  
                "LifecycleManagement": "ENABLED",  
                "IdleTimeoutInMinutes": 120,  
                "MaxIdleTimeoutInMinutes": 120,  
                "MinIdleTimeoutInMinutes": 120
```

```
        }
    }
}

## default-user-settings.json example for letting users customize the default timeout,
between 2-5 hours
{
    "JupyterLabAppSettings": {
        "AppLifecycleManagement": {
            "IdleSettings": {
                "LifecycleManagement": "ENABLED",
                "IdleTimeoutInMinutes": 120,
                "MinIdleTimeoutInMinutes": 120,
                "MaxIdleTimeoutInMinutes": 300
            }
        }
    }
}
```

## User profile

The following command shows how to enable idle shutdown when updating an existing user profile. To add idle shutdown for a new user profile, use the `create-user-profile` command instead.

### Note

If idle shutdown is set when applications are running, they must be restarted for idle shutdown settings to take effect.

```
aws sagemaker update-user-profile --region region --domain-id domain-id \
--user-profile-name user-profile-name --user-settings file://user-settings.json

## user-settings.json example for enforcing the default timeout
{
    "JupyterLabAppSettings": {
        "AppLifecycleManagement": {
            "IdleSettings": {
                "LifecycleManagement": "ENABLED",
                "IdleTimeoutInMinutes": 120,
                "MaxIdleTimeoutInMinutes": 120,
                "MinIdleTimeoutInMinutes": 120
            }
        }
    }
}
```

```
        }
    }
}

## user-settings.json example for letting users customize the default timeout, between
## 2-5 hours
{
    "JupyterLabAppSettings": {
        "AppLifecycleManagement": {
            "IdleSettings": {
                "LifecycleManagement": "ENABLED",
                "IdleTimeoutInMinutes": 120,
                "MinIdleTimeoutInMinutes": 120,
                "MaxIdleTimeoutInMinutes": 300
            }
        }
    }
}
```

## Update default idle shutdown settings

You can update the default idle shutdown settings at either the domain or user profile level.

 **Note**

If idle shutdown is set when applications are running, they must be restarted for idle shutdown settings to take effect.

### Update domain settings

1. Navigate to the domain.
2. Choose the **App Configurations** tab.
3. From the **App Configurations** tab, navigate to either the Code Editor or JupyterLab section.
4. In the section for the application that you want to modify the idle shutdown time limit for, select **Edit**.
5. Update the idle shutdown settings for the domain.
6. Select **Save Changes**.

## Update user profile settings

1. Navigate to the domain.
2. Choose the **User profiles** tab.
3. From the **User profiles** tab, select the user profile to edit.
4. From the **User profile** page, choose the **Applications** tab.
5. On the **Applications** tab, navigate to either the Code Editor or JupyterLab section.
6. In the section for the application that you want to modify the idle shutdown time limit for, select **Edit**.
7. Update the idle shutdown settings for the user profile.
8. Select **Save Changes**.

## Modify your idle shutdown time limit

Users may be able to modify the idle shutdown time limit if the admin gives access when adding support for idle shutdown. If support for idle shutdown is added, there may be a limit applied to the maximum time for idle shutdown. A user can set the value anywhere between the lower limit and upper limit.

1. Launch Amazon SageMaker Studio by following the steps in [Launch Amazon SageMaker Studio](#).
2. From the **Applications** section, select the application type to update the idle shutdown time for.
3. Select the space to update.
4. Update **Idle shutdown (mins)** with your desired value.

 **Note**

If idle shutdown is set when applications are running, they must be restarted for idle shutdown settings to take effect.

# Applications supported in Amazon SageMaker Studio

## Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

Amazon SageMaker Studio supports the following applications:

- **Code Editor, based on Code-OSS, Visual Studio Code - Open Source**– Code Editor offers a lightweight and powerful integrated development environment (IDE) with familiar shortcuts, terminal, and advanced debugging capabilities and refactoring tools. It is a fully managed, browser-based application in Studio. For more information, see [Code Editor in Amazon SageMaker Studio](#).
- **Amazon SageMaker Studio Classic**– Amazon SageMaker Studio Classic is a web-based IDE for machine learning. With Studio Classic, you can build, train, debug, deploy, and monitor your machine learning models. For more information, see [Amazon SageMaker Studio Classic](#).
- **JupyterLab**– JupyterLab offers a set of capabilities that augment the fully managed notebook offering. It includes kernels that start in seconds, a pre-configured runtime with popular data science, machine learning frameworks, and high performance block storage. For more information, see [SageMaker JupyterLab](#).
- **Amazon SageMaker Canvas**– With SageMaker Canvas, you can use machine learning to generate predictions without writing code. With Canvas, you can chat with popular large language models (LLMs), access ready-to-use models, or build a custom model that's trained on your data. For more information, see [Amazon SageMaker Canvas](#).
- **RStudio**– RStudio is an integrated development environment for R. It includes a console and syntax-highlighting editor that supports running code directly. It also includes tools for plotting, history, debugging, and workspace management. For more information, see [RStudio on Amazon SageMaker AI](#).

## Lifecycle configurations within Amazon SageMaker Studio

Administrators and users can create and attach lifecycle configurations (LCCs) to automate the customization of the following applications within your Amazon SageMaker Studio environment:

- Amazon SageMaker AI JupyterLab
- Code Editor, based on Code-OSS, Visual Studio Code - Open Source
- Studio Classic
- Notebook instance

Customizing your application includes:

- Installing custom packages
- Configuring extensions
- Preloading datasets
- Setting up source code repositories

Users create and attach built-in lifecycle configurations to their own user profiles. Administrators create and attach default or built-in lifecycle configurations at the domain, space, or user profile level.

### Important

Amazon SageMaker Studio first runs the built-in lifecycle configuration and then runs the default LCC. Amazon SageMaker AI won't resolve package conflicts between the user and administrator LCCs. For example, if the built-in LCC installs `python3.11` and the default LCC installs `python3.12`, Studio installs `python3.12`.

## Create and attach lifecycle configurations

You can create and attach lifecycle configurations using either the AWS Management Console or the AWS Command Line Interface.

### Topics

- [Create and attach lifecycle configurations \(AWS CLI\)](#)

- [Create and attach lifecycle configurations \(console\)](#)

## Create and attach lifecycle configurations (AWS CLI)

### Important

Before you begin, complete the following prerequisites:

- Update the AWS CLI by following the steps in [Installing the current AWS CLI Version](#).
- From your local machine, run `aws configure` and provide your AWS credentials.  
For information about AWS credentials, see [Understanding and getting your AWS credentials](#).
- Onboard to Amazon SageMaker AI domain. For conceptual information, see [Amazon SageMaker AI domain overview](#). For a quickstart guide, see [Use quick setup for Amazon SageMaker AI](#).

The following procedure shows how to create a lifecycle configuration script that prints Hello World within Code Editor or JupyterLab.

### Note

Each script can have up to **16,384 characters**.

1. From your local machine, create a file named `my-script.sh` with the following content:

```
#!/bin/bash
set -eux
echo 'Hello World!'
```

2. Use the following to convert your `my-script.sh` file into base64 format. This requirement prevents errors that occur from spacing and line break encoding.

```
LCC_CONTENT=`openssl base64 -A -in my-script.sh`
```

3. Create a lifecycle configuration for use with Studio. The following command creates a lifecycle configuration that runs when you launch an associated JupyterLab application:

```
aws sagemaker create-studio-lifecycle-config \
--region region \
--studio-lifecycle-config-name my-lcc \
--studio-lifecycle-config-content $LCC_CONTENT \
--studio-lifecycle-config-app-type application-type
```

For `studio-lifecycle-config-app-type`, specify either `CodeEditor` or `JupyterLab`.

 **Note**

The ARN of the newly created lifecycle configuration that is returned. This ARN is required to attach the lifecycle configuration to your application.

To ensure that the environments are customized properly, users and administrators use different commands to attach lifecycle configurations.

### Attach default lifecycle configurations (administrator)

To attach the lifecycle configuration, you must update the `UserSettings` for your domain or user profile. Lifecycle configuration scripts that are associated at the domain level are inherited by all users. However, scripts that are associated at the user profile level are scoped to a specific user.

You can create a new user profile, domain, or space with a lifecycle configuration attached by using the following commands:

- [create-user-profile](#)
- [create-domain](#)
- [create-space](#)

The following command creates a user profile with a lifecycle configuration for a JupyterLab application. Add the lifecycle configuration ARN from the preceding step to the `JupyterLabAppSettings` of the user. You can add multiple lifecycle configurations at the same time by passing a list of them. When a user launches a JupyterLab application with the AWS CLI, they can specify a lifecycle configuration instead of using the default one. The lifecycle configuration that the user passes must belong to the list of lifecycle configurations in `JupyterLabAppSettings`.

```
# Create a new UserProfile
aws sagemaker create-user-profile --domain-id domain-id \
--user-profile-name user-profile-name \
--region region \
--user-settings '{
"JupyterLabAppSettings": {
    "LifecycleConfigArns": [
        lifecycle-configuration-arn-list
    ]
}
}'
```

The following command creates a user profile with a lifecycle configuration for a Code Editor application. Add the lifecycle configuration ARN from the preceding step to the CodeEditorAppSettings of the user. You can add multiple lifecycle configurations at the same time by passing a list of them. When a user launches a Code Editor application with the AWS CLI, they can specify a lifecycle configuration instead of using the default one. The lifecycle configuration that the user passes must belong to the list of lifecycle configurations in CodeEditorAppSettings.

```
# Create a new UserProfile
aws sagemaker create-user-profile --domain-id domain-id \
--user-profile-name user-profile-name \
--region region \
--user-settings '{
"CodeEditorAppSettings": {
    "LifecycleConfigArns": [
        lifecycle-configuration-arn-list
    ]
}
}'
```

## Attach built-in lifecycle configurations (user)

To attach the lifecycle configuration, you must update the UserSettings for your user profile.

The following command creates a user profile with a lifecycle configuration for a JupyterLab application. Add the lifecycle configuration ARN from the preceding step to the JupyterLabAppSettings of your user profile.

```
# Update a UserProfile
aws sagemaker update-user-profile --domain-id domain-id \
```

```
--user-profile-name user-profile-name \
--region region \
--user-settings '{
  "JupyterLabAppSettings": {
    "BuiltInLifecycleConfigArn": "lifecycle-configuration-arn"
  }
}'
```

The following command creates a user profile with a lifecycle configuration for a Code Editor application. Add the lifecycle configuration ARN from the preceding step to the CodeEditorAppSettings of your user profile. The lifecycle configuration that the user passes must belong to the list of lifecycle configurations in CodeEditorAppSettings.

```
# Update a UserProfile
aws sagemaker update-user-profile --domain-id domain-id \
--user-profile-name user-profile-name \
--region region \
--user-settings '{
  "CodeEditorAppSettings": {
    "BuiltInLifecycleConfigArn": "lifecycle-configuration-arn"
  }
}'
```

## Create and attach lifecycle configurations (console)

To create and attach lifecycle configurations in the AWS Management Console, navigate to the [Amazon SageMaker AI console](#) and choose **Lifecycle configurations** in the left-hand navigation. The console will guide you through the process of creating the lifecycle configuration.

## Debug lifecycle configurations

The following topics show how to get information about and debug your lifecycle configurations.

### Topics

- [Verify lifecycle configuration process from CloudWatch Logs](#)
- [Lifecycle configuration timeout](#)

### Verify lifecycle configuration process from CloudWatch Logs

Lifecycle configurations only log STDOUT and STDERR.

STDOUT is the default output for bash scripts. You can write to STDERR by appending >&2 to the end of a bash command. For example, echo 'hello'>&2.

Logs for your lifecycle configurations are published to your AWS account using Amazon CloudWatch. These logs can be found in the /aws/sagemaker/studio log stream in the CloudWatch console.

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. Choose **Logs** from the left navigation pane. From the dropdown menu, select **Log groups**.
3. On the **Log groups** page, search for aws/sagemaker/studio.
4. Select the log group.
5. On the **Log group details** page, choose the **Log streams** tab.
6. To find the logs for a specific app, search the log streams using the following format:

*domain-id/user-profile-name/app-type/app-name*

The following search string finds the lifecycle configuration logs for the domain d-m851cu8vbqmz, user profile i-sonic-js, application type JupyterLab, and application name test-lcc-echo:

d-m851cu8vbqmz/i-sonic-js/JupyterLab/test-lcc-echo

7. To view the script execution logs, select the log stream appended with LifecycleConfigOnStart.

## Lifecycle configuration timeout

There is a lifecycle configuration timeout limitation of 5 minutes. If a lifecycle configuration script takes longer than 5 minutes to run, you get an error.

To resolve this error, make sure that your lifecycle configuration script completes in less than 5 minutes.

To help decrease the runtime of scripts, try the following:

- Reduce unnecessary steps. For example, limit which conda environments to install large packages in.

- Run tasks in parallel processes.
- Use the nohup command in your script to make sure that hangup signals are ignored so that the script runs without stopping.

## Amazon SageMaker Studio spaces

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

Spaces are used to manage the storage and resource needs of some Amazon SageMaker Studio applications. Each space is composed of multiple resources and can be either private or shared. Each space has a 1:1 relationship with an instance of an application. Every supported application that is created gets its own space. The following applications in Studio run on spaces:

- [Code Editor in Amazon SageMaker Studio](#)
- [SageMaker JupyterLab](#)
- [Amazon SageMaker Studio Classic](#)

A space is composed of the following resources:

- A storage volume.
  - For Studio Classic, the space is connected to the shared Amazon Elastic File System (Amazon EFS) volume for the domain.
  - For other applications, a distinct Amazon Elastic Block Store (Amazon EBS) volume is attached to the space. All applications are given their own Amazon EBS volume. Applications do not have access to the Amazon EBS volume of other applications. For more information about Amazon EBS volumes, see [Amazon Elastic Block Store \(Amazon EBS\)](#).
- The application type of the space.
- The image that the application is based on.

Spaces can be either private or shared:

- **Private:** Private spaces are scoped to a single user in a domain. Private spaces cannot be shared with other users. All applications that support spaces also support private spaces.
- **Shared:** Shared spaces are accessible by all users in the domain. For more information about shared spaces, see [Collaboration with shared spaces](#).

Spaces can be created in domains that use either AWS IAM Identity Center or AWS Identity and Access Management (IAM) authentication. The following sections give general information about how to access spaces. For specific information about creating and accessing a space, see the documentation for the respective application type of the space that you're creating.

For information about viewing, stopping, or deleting your applications, instances, or spaces, see [Stop and delete your Studio running applications and spaces](#).

## Topics

- [Launch spaces](#)
- [Collaboration with shared spaces](#)

## Launch spaces

The following sections give information about accessing spaces in a domain. Spaces can be accessed in one of the following ways:

- from the Amazon SageMaker AI console
- from Studio
- using the AWS CLI

## Accessing spaces from the Amazon SageMaker AI console

### To access spaces from the Amazon SageMaker AI console

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Under **Admin configurations**, choose **Domains**.
3. From the list of domains, select the domain that contains the spaces.
4. On the **Domain details** page, select the **Space management** tab. For more information about managing spaces, see [Collaboration with shared spaces](#).
5. From the list of spaces for that domain, select the space to launch.
6. Choose **Launch Studio** for the space that you want to launch.

## Accessing spaces from Studio

Follow these steps to access spaces from Studio for a specific application type.

### To access spaces from Studio

1. Open Studio by following the steps in [Launch Amazon SageMaker Studio](#).
2. Select the application type with spaces that you want to access.

## Accessing spaces using the AWS CLI

The following sections show how to access a space from the AWS Command Line Interface (AWS CLI). The procedures are for domains that use AWS Identity and Access Management (IAM) or AWS IAM Identity Center authentication.

### IAM authentication

The following procedure outlines generally how to access a space using IAM authentication from the AWS CLI.

1. Create a presigned domain URL specifying the name of the space that you want to access.

```
aws \
  --region region \
  sagemaker \
  create-presigned-domain-url \
  --domain-id domain-id \
  --user-profile-name user-profile-name \
  --space-name space-name
```

2. Navigate to the URL.

## Accessing a space in IAM Identity Center authentication

The following procedure outlines how to access a space using IAM Identity Center authentication from the AWS CLI.

1. Use the following command to return the URL associated with the space.

```
aws \
  --region region \
  sagemaker \
  describe-space \
  --domain-id domain-id \
  --space-name space-name
```

2. Append the respective redirect parameter for the application type to the URL to be federated through IAM Identity Center. For more information about the redirect parameters, see [describe-space](#).
3. Navigate to the URL to be federated through IAM Identity Center.

## Collaboration with shared spaces

An Amazon SageMaker Studio Classic shared space consists of a shared JupyterServer application and a shared directory. A JupyterLab shared space consists of a shared JupyterLab application and a shared directory within Amazon SageMaker Studio. All user profiles in a domain have access to all shared spaces in the domain. Amazon SageMaker AI automatically scopes resources in a shared space within the context of the Amazon SageMaker Studio Classic application that you launch in that shared space. Resources in a shared space include notebooks, files, experiments, and models. Use shared spaces to collaborate with other users in real-time using features like automatic tagging, real time co-editing of notebooks, and customization.

Shared spaces are available in:

- Amazon SageMaker Studio Classic
- JupyterLab

A Studio Classic shared space only supports Studio Classic and KernelGateway applications. A shared space only supports the use of a JupyterLab 3 image Amazon Resource Name (ARN). For more information, see [JupyterLab Versioning](#).

Amazon SageMaker AI automatically tags all SageMaker AI resources that you create within the scope of a shared space. You can use these tags to monitor costs and plan budgets using tools, such as AWS Budgets.

A shared space uses the same VPC settings as the domain that it's created in.

 **Note**

Shared spaces do not support the use of Amazon SageMaker Data Wrangler or Amazon EMR cross-account clusters.

## Automatic tagging

All resources created in a shared space are automatically tagged with a domain ARN tag and shared space ARN tag. The domain ARN tag is based on the domain ID, while the shared space ARN tag is based on the shared space name.

You can use these tags to monitor AWS CloudTrail usage. For more information, see [Log Amazon SageMaker API Calls with AWS CloudTrail](#).

You can also use these tags to monitor costs with AWS Billing and Cost Management. For more information, see [Using AWS cost allocation tags](#).

## Real time co-editing of notebooks

A key benefit of a shared space is that it facilitates collaboration between members of the shared space in real time. Users collaborating in a workspace get access to a shared Studio Classic application where they can access, read, and edit their notebooks in real time. Real time collaboration is only supported for JupyterServer applications within a shared space.

Users with access to a shared space can simultaneously open, view, edit, and execute Jupyter notebooks in the shared Studio Classic or JupyterLab application in that space.

The notebook indicates each co-editing user with a different cursor that shows the user profile name. While multiple users can view the same notebook, co-editing is best suited for small groups of two to five users.

To track changes being made by multiple users, we strongly recommended using Studio Classic's built-in Git-based version control.

## JupyterServer 2

To use shared spaces in Studio Classic, Jupyter Server version 2 is required. Certain JupyterLab extensions and packages can forcefully downgrade Jupyter Server to version 1. This prevents the use of shared space. Run the following from the command prompt to change the version number and continue using shared spaces.

```
conda activate studio
pip install jupyter-server==2.0.0rc3
```

## Customize a shared space

To attach a lifecycle configuration or custom image to a shared space, you must use the AWS CLI. For more information about creating and attaching lifecycle configurations, see [Create and associate a lifecycle configuration](#). For more information about creating and attaching custom images, see [Bring your own SageMaker image](#).

## Create a shared space

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

The following topic demonstrates how to create a shared space in an existing Amazon SageMaker AI domain. If you created your domain without support for shared spaces, you must add support for shared spaces to your existing domain before you can create a shared space.

## Topics

- [Add shared space support to an existing domain](#)
- [Create a shared space](#)

### Add shared space support to an existing domain

You can use the SageMaker AI console or the AWS CLI to add support for shared spaces to an existing domain. If the domain is using VPC only network access, then you can only add shared space support using the AWS CLI.

#### Console

Complete the following procedure to add support for Studio Classic shared spaces to an existing domain from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to open the **domain settings** page for.
5. On the **domain details** page, choose the **domain settings** tab.
6. Choose **Edit**.
7. For **Space default execution role**, set an IAM role that is used by default for all shared spaces created in the domain.
8. Choose **Next**.
9. Choose **Next**.

10. Choose **Next**.

11. Choose **Submit**.

## AWS CLI

### Studio Classic

Run the following command from the terminal of your local machine to add default shared space settings to a domain from the AWS CLI. If you are adding default shared space settings to a domain within an Amazon VPC, you must also include a list of security groups. Studio Classic shared spaces only support the use of JupyterLab 3 image ARNs. For more information, see [JupyterLab Versioning](#).

```
# Public Internet domain
aws --region region \
sagemaker update-domain \
--domain-id domain-id \
--default-space-settings "ExecutionRole=execution-role-arn,JupyterServerAppSettings={DefaultResourceSpec={InstanceType=example-instance-type,SageMakerImageArn=sagemaker-image-arnregion \
sagemaker update-domain \
--domain-id domain-id \
--default-space-settings "ExecutionRole=execution-role-arn,JupyterServerAppSettings={DefaultResourceSpec={InstanceType=system,SageMakerImageArn=sag-image-arnsecurity-groups]"
```

Use the following command to verify that the default shared space settings have been updated.

```
aws --region region \
sagemaker describe-domain \
--domain-id domain-id
```

### JupyterLab

Run the following command from the terminal of your local machine to add default shared space settings to a domain from the AWS CLI. If you are adding default shared space settings to

a domain within an Amazon VPC, you must also include a list of security groups. Studio Classic shared spaces only support the use of JupyterLab 4 image ARNs. For more information, see [JupyterLab Versioning](#).

```
# Public Internet domain
aws --region region \
sagemaker update-domain \
--domain-id domain-id \
--default-space-settings "ExecutionRole=execution-role-arn,
JupyterLabAppSettings={DefaultResourceSpec={InstanceType=example-instance-type,SageMakerImageArn=sagemaker-image-arn}}"

# VPCOnly domain
aws --region region \
sagemaker update-domain \
--domain-id domain-id \
--default-space-settings "ExecutionRole=execution-role-arn,
SecurityGroups=[security-groups]"
```

Use the following command to verify that the default shared space settings have been updated.

```
aws --region region \
sagemaker describe-domain \
--domain-id domain-id
```

## Create a shared space

The following sections demonstrate how to create a shared space from the Amazon SageMaker AI console, Amazon SageMaker Studio, or the AWS CLI.

### Create from Studio

Use the following procedures to create a shared space in a domain from Studio.

#### Studio Classic

1. Navigate to Studio following the steps in [Launch Amazon SageMaker Studio](#).
2. From the Studio UI, find the applications pane on the left side.
3. From the applications pane, select **Studio Classic**.

4. Choose **Create Studio Classic space**
5. In the pop up window, enter a name for the space.
6. Choose **Create space**.

## JupyterLab

1. Navigate to Studio following the steps in [Launch Amazon SageMaker Studio](#).
2. From the Studio UI, find the applications pane on the left side.
3. From the applications pane, select **JupyterLab**.
4. Choose **Create JupyterLab space**
5. In the pop up window, enter a name for the space.
6. Choose **Create space**.

## Create from the console

Complete the following procedure to create a shared space in a domain from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to create a shared space for.
5. On the **domain details** page, choose the **Space management** tab.
6. Choose **Create**.
7. Enter a name for your shared space. shared space names within a domain must be unique. The execution role for the shared space is set to the domain IAM execution role.

## Create from AWS CLI

This section shows how to create a shared space from the AWS CLI.

You cannot set the execution role of a shared space when creating or updating it.

The `DefaultDomainExecRole` can only be set when creating or updating the domain. shared spaces only support the use of JupyterLab 3 image ARNs. For more information, see [JupyterLab Versioning](#).

To create a shared space from the AWS CLI, run one of the following commands from the terminal of your local machine.

## Studio Classic

```
aws --region region \
sagemaker create-space \
--domain-id domain-id \
--space-name space-name \
--space-settings '{
    "JupyterServerAppSettings": {
        "DefaultResourceSpec": {
            "SageMakerImageArn": "sagemaker-image-arn",
            "InstanceType": "system"
        }
    }
}'
```

## JupyterLab

```
aws --region region \
sagemaker create-space \
--domain-id domain-id \
--space-name space-name \
--ownership-settings "[{"OwnerUserName": "user-profile-name"}]" \
--space-sharing-settings "[{"SharingType": "Shared"}]" \
--space-settings "[{"AppType": "JupyterLab"}]"
```

## Get information about shared spaces

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

This guide shows how to access a list of shared spaces in an Amazon SageMaker AI domain with the Amazon SageMaker AI console, Amazon SageMaker Studio, or the AWS CLI. It also shows how to view details of a shared space from the AWS CLI.

## Topics

- [List shared spaces](#)
- [View shared space details](#)

### List shared spaces

The following topic describes how to view a list of shared spaces within a domain from the SageMaker AI console or the AWS CLI.

#### List shared spaces from Studio

Complete the following procedure to view a list of the shared spaces in a domain from Studio.

1. Navigate to Studio following the steps in [Launch Amazon SageMaker Studio](#).
2. From the Studio UI, find the applications pane on the left side.
3. From the applications pane, select **Studio Classic** or **JupyterLab**. You can view the spaces that are being used to run the application type.

#### List shared spaces from the console

Complete the following procedure to view a list of the shared spaces in a domain from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to view the list of shared spaces for.
5. On the **domain details** page, choose the **Space management** tab.

#### List shared spaces from the AWS CLI

To list the shared spaces in a domain from the AWS CLI, run the following command from the terminal of your local machine.

```
aws --region region \  
sagemaker list-spaces \  
--domain-id domain-id
```

## View shared space details

The following section describes how to view shared space details from the SageMaker AI console, Studio, or the AWS CLI.

### View shared spaces details from Studio

Complete the following procedure to view the details of a shared spaces in a domain from Studio.

1. Navigate to Studio following the steps in [Launch Amazon SageMaker Studio](#).
2. From the Studio UI, find the applications pane on the left side.
3. From the applications pane, select **Studio Classic** or **JupyterLab**. You can view the spaces that are running the application.
4. Select the name of the space that you want to view more details for.

### View shared space details from the console

You can view the details of a shared space from the SageMaker AI console using the following procedure.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to view the list of shared spaces for.
5. On the **domain details** page, choose the **Space management** tab.
6. Select the name of the space to open a new page that lists details about the shared space.

### View shared space details from the AWS CLI

To view the details of a shared space from the AWS CLI, run the following command from the terminal of your local machine.

```
aws --region region \  
sagemaker describe-space \  
--domain-id domain-id \  
--space-name space-name
```

## Edit a shared space

You can only edit the details for an Amazon SageMaker Studio Classic or JupyterLab shared space using the AWS CLI. You can't edit the details of a shared space from the Amazon SageMaker AI console. You can only update workspace attributes when there are no running applications in the shared space.

### Studio Classic

To edit the details of a Studio Classic shared space from the AWS CLI, run the following one of the following commands from the terminal of your local machine. shared spaces only support the use of JupyterLab 3 image ARNs. For more information, see [JupyterLab Versioning](#).

```
aws --region region \  
sagemaker update-space \  
--domain-id domain-id \  
--space-name space-name \  
--query SpaceArn --output text \  
--space-settings '{  
    "JupyterServerAppSettings": {  
        "DefaultResourceSpec": {  
            "SageMakerImageArn": "sagemaker-image-arn",  
            "InstanceType": "system"  
        }  
    }  
}'
```

### JupyterLab

To edit the details of a JupyterLab shared space from the AWS CLI, run the following one of the following commands from the terminal of your local machine. shared spaces only support the use of JupyterLab 4 image ARNs. For more information, see [SageMaker JupyterLab](#).

```
aws --region region \  
sagemaker update-space \  
'
```

```
--domain-id domain-id \
--space-name space-name \
--space-settings "{"
    "SpaceStorageSettings": {
        "EbsStorageSettings": {
            "EbsVolumeSizeInGb":100
        }
    }
}"
}
```

## Delete a shared space

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following topic shows how to delete an Amazon SageMaker Studio Classic shared space from the Amazon SageMaker AI console or AWS CLI. A shared space can only be deleted if it has no running applications.

### Topics

- [Console](#)
- [AWS CLI](#)

### Console

Complete the following procedure to delete a shared space in the Amazon SageMaker AI domain from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to create a shared space for.

5. On the **domain details** page, choose the **Space management** tab.
6. Select the shared space that you want to delete. The shared space must not contain any non-failed apps.
7. Choose **Delete**. This opens a new window.
8. Choose **Yes, delete space**.
9. Enter *delete* in the field.
10. Choose **Delete space**.

## AWS CLI

To delete a shared space from the AWS CLI, run the following command from the terminal of your local machine.

```
aws --region region \  
sagemaker delete-space \  
--domain-id domain-id \  
--space-name space-name
```

## Perform common UI tasks

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

The following sections describe how to perform common tasks in the Amazon SageMaker Studio UI. For an overview of the Studio user interface, see [Amazon SageMaker Studio UI overview](#).

### Set cookie preferences

1. Launch Studio following the steps in [Launch Amazon SageMaker Studio](#).
2. At the bottom of the Studio user interface, choose **Cookie Preferences**.
3. Select the check box for each type of cookie that you want Amazon SageMaker AI to use.

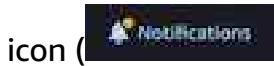
#### 4. Choose **Save preferences**.

### Manage notifications

Notifications give information about important changes to Studio, updates to applications, and issues to resolve.

1. Launch Studio following the steps in [Launch Amazon SageMaker Studio](#).

2. On the top navigation bar, choose the **Notifications**



icon ( ).

3. From the list of notifications, select the notification to get information about it.

### Leave feedback

We take your feedback seriously. We encourage you to provide feedback.

At the top navigation of Studio, choose **Provide feedback**.

### Sign out

Signing out of the Studio UI is different than closing the browser window. Signing out clears session data from the browser and deletes unsaved changes.

This same behavior also happens when the Studio session times out. This happens after 5 minutes.

1. Launch Studio following the steps in [Launch Amazon SageMaker Studio](#).

2. Choose the **User options** icon



( ).

3. Choose **Sign out**.

4. In the pop-up window, choose **Sign out**.

## NVMe stores with Amazon SageMaker Studio

Amazon SageMaker Studio applications and their associated notebooks run on Amazon Elastic Compute Cloud (Amazon EC2) instances. Some of the Amazon EC2 instance types, such as the m1 .m5d instance family, offer non-volatile memory express (NVMe) solid state drives (SSD) instance

stores. NVMe instance stores are local ephemeral disk stores that are physically connected to an instance for fast temporary storage. Studio applications support NVMe instance stores for supported instance types. For more information about instance types and their associated NVMe store volumes, see the [Amazon Elastic Compute Cloud Instance Type Details](#). This topic provides information about accessing and using NVMe instance stores, as well as considerations when using NVMe instance stores with Studio.

## Considerations

The following considerations apply when using NVMe instance stores with Studio.

- An NVMe instance store is temporary storage. The data stored on the NVMe store is deleted when the instance is terminated, stopped, or hibernated. When using NVMe stores with Studio applications, the data on the NVMe instance store is lost whenever the application is deleted, restarted, or patched. We recommend that you back up valuable data to persistent storage solutions, such as Amazon Elastic Block Store, Amazon Elastic File System, or Amazon Simple Storage Service.
- Studio patches instances periodically to install new security updates. When an instance is patched, the instance is restarted. This restart results in the deletion of data stored in the NVMe instance store. We recommend that you frequently back up necessary data from the NVMe instance store to persistent storage solutions, such as Amazon Elastic Block Store, Amazon Elastic File System, or Amazon Simple Storage Service.
- The following Studio applications support using NVMe storage:
  - JupyterLab
  - Code Editor, based on Code-OSS, Visual Studio Code - Open Source
  - KernelGateway

## Access NVMe instance stores

When you select an instance type with attached NVMe instance stores to host a Studio application, the NVMe instance store directory is mounted to the application container at the following location:

```
/mnt/sagemaker-nvme
```

If an instance has more than 1 NVMe instance store attached to it, Studio creates a striped logical volume that spans all of the local disks attached. Studio then mounts this striped logical volume

to the `/mnt/sagemaker-nvme` directory. As a result, the directory storage size is the sum of all NVMe instance store volume sizes attached to the instance.

If the `/mnt/sagemaker-nvme` directory does not exist, verify that the instance type hosting your application has an attached NVMe instance store volume.

## Local mode support in Amazon SageMaker Studio

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

Amazon SageMaker Studio applications support the use of local mode to create estimators, processors, and pipelines, then deploy them to a local environment. With local mode, you can test machine learning scripts before running them in Amazon SageMaker AI managed training or hosting environments. Studio supports local mode in the following applications:

- Amazon SageMaker Studio Classic
- JupyterLab
- Code Editor, based on Code-OSS, Visual Studio Code - Open Source

Local mode in Studio applications is invoked using the SageMaker Python SDK. In Studio applications, local mode functions similarly to how it functions in Amazon SageMaker notebook instances, with some differences. For more information about using local mode with the SageMaker Python SDK, see [Local Mode](#).

**Note**

Studio applications do not support multi-container jobs in local mode. Local mode jobs are limited to a single instance for training, inference, and processing jobs. When creating a local mode job, the instance count configuration must be 1.

## Docker support

As part of local mode support, Studio applications support limited Docker access capabilities. With this support, users can interact with the Docker API from Jupyter notebooks or the image terminal of the application. Customers can interact with Docker using one of the following:

- [Docker CLI](#)
- [Docker Compose CLI](#)
- Language specific Docker SDK clients

Studio also supports limited Docker access capabilities with the following restrictions:

- Usage of Docker networks is not supported.
- Docker [volume](#) usage is not supported during container run. Only volume bind mount inputs are allowed during container orchestration. The volume bind mount inputs must be located on the Amazon Elastic File System (Amazon EFS) volume for Studio Classic. For JupyterLab and Code Editor applications, it must be located on the Amazon Elastic Block Store (Amazon EBS) volume.
- Container inspect operations are allowed.
- Container port to host mapping is not allowed. However, you can specify a port for hosting. The endpoint is then accessible from Studio using the following URL:

`http://localhost:port`

## Docker operations supported

The following table lists all of the Docker API endpoints that are supported in Studio, including any support limitations. If an API endpoint is missing from the table, Studio doesn't support it.

API Documentation	Limitations
<a href="#">SystemAuth</a>	
<a href="#">SystemEvents</a>	
<a href="#">SystemVersion</a>	
<a href="#">SystemPing</a>	
<a href="#">SystemPingHead</a>	
<a href="#">ContainerCreate</a>	<ul style="list-style-type: none"><li>Containers cannot be run in Docker default bridge or custom Docker networks. Containers are run in the same network as the Studio application container.</li><li>Users can only use the following value for the network name: <code>sagemaker</code>. For example: <pre>docker run --net sagemaker <i>parameter</i>       -values</pre></li><li>Only bind mounts are allowed for volume usage. The host directory should exist on Amazon EFS for KernelGateway applications or Amazon EBS for other applications.</li><li>Containers cannot run in privileged mode or with elevated secure computing permissions.</li></ul>
<a href="#">ContainerStart</a>	
<a href="#">ContainerStop</a>	
<a href="#">ContainerKill</a>	
<a href="#">ContainerDelete</a>	

API Documentation	Limitations
<a href="#">ContainerList</a>	
<a href="#">ContainerLogs</a>	
<a href="#">ContainerInspect</a>	
<a href="#">ContainerWait</a>	
<a href="#">ContainerAttach</a>	
<a href="#">ContainerPrune</a>	
<a href="#">ContainerResize</a>	
<a href="#">ImageCreate</a>	VPC-only mode support is limited to Amazon ECR images in allowlisted accounts.
<a href="#">ImagePrune</a>	
<a href="#">ImagePush</a>	VPC-only mode support is limited to Amazon ECR images in allowlisted accounts.
<a href="#">ImageList</a>	
<a href="#">ImageInspect</a>	
<a href="#">ImageGet</a>	
<a href="#">ImageDelete</a>	
<a href="#">ImageBuild</a>	<ul style="list-style-type: none"><li>VPC-only mode support is limited to Amazon ECR images in allowlisted accounts.</li><li>Users can only use the following value for the network name: <code>sagemaker</code> . For example:<pre>docker build --network sagemaker <i>parameter-values</i></pre></li></ul>

## Topics

- [Getting started with local mode](#)

## Getting started with local mode

The following sections outline the steps needed to get started with local mode in Amazon SageMaker Studio, including:

- Completing prerequisites
- Setting `EnableDockerAccess`
- Docker installation

### Prerequisites

Complete the following prerequisites to use local mode in Studio applications:

- To pull images from an Amazon Elastic Container Registry repository, the account hosting the Amazon ECR image must provide access permission for the user's execution role. The domain's execution role must also allow Amazon ECR access.
- Verify that you are using the latest version of the Studio Python SDK by using the following command:

```
pip install -U sagemaker
```

- To use local mode and Docker capabilities, set the following parameter of the domain's `DockerSettings` using the AWS Command Line Interface (AWS CLI):

```
EnableDockerAccess : ENABLED
```

- Using `EnableDockerAccess`, you can also control whether users in the domain can use local mode. By default, local mode and Docker capabilities aren't allowed in Studio applications. For more information, see [Setting `EnableDockerAccess`](#).
- Install the Docker CLI in the Studio application by following the steps in [Docker installation](#).

## Setting EnableDockerAccess

The following sections show how to set `EnableDockerAccess` when the domain has public internet access or is in VPC-only mode.

### Note

Changes to `EnableDockerAccess` only apply to applications created after the domain is updated. You must create a new application after updating the domain.

### Public internet access

The following example commands show how to set `EnableDockerAccess` when creating a new domain or updating an existing domain with public internet access:

```
# create new domain
aws --region region \
    sagemaker create-domain --domain-name domain-name \
    --vpc-id vpc-id \
    --subnet-ids subnet-ids \
    --auth-mode IAM \
    --default-user-settings "ExecutionRole=execution-role" \
    --domain-settings '{"DockerSettings": {"EnableDockerAccess": "ENABLED"}}' \
    --query DomainArn \
    --output text

# update domain
aws --region region \
    sagemaker update-domain --domain-id domain-id \
    --domain-settings-for-update '{"DockerSettings": {"EnableDockerAccess": "ENABLED"}}'
```

### VPC-only mode

When using a domain in VPC-only mode, Docker image push and pull requests are routed through the service VPC instead of the VPC configured by the customer. Because of this functionality, administrators can configure a list of trusted AWS accounts that users can make Amazon ECR Docker pull and push operations requests to.

If a Docker image push or pull request is made to an AWS account that is not in the list of trusted AWS accounts, the request fails. Docker pull and push operations outside of Amazon Elastic Container Registry (Amazon ECR) aren't supported in VPC-only mode.

The following AWS accounts are trusted by default:

- The account hosting the SageMaker AI domain.
- SageMaker AI accounts that host the following SageMaker images:
  - DLC framework images
  - Sklearn, Spark, XGBoost processing images

To configure a list of additional trusted AWS accounts, specify the `VpcOnlyTrustedAccounts` value as follows:

```
aws --region region \
    sagemaker update-domain --domain-id domain-id \
    --domain-settings-for-update '{"DockerSettings": {"EnableDockerAccess": "ENABLED",
    "VpcOnlyTrustedAccounts": ["account-list"]}}'
```

## Docker installation

To use Docker, you must manually install Docker from the terminal of your Studio application. The steps to install Docker are different if the domain has access to the internet or not.

### Internet access

If the domain is created with public internet access or in VPC-only mode with limited internet access, use the following steps to install Docker.

1. (Optional) If your domain is created in VPC-only mode with limited internet access, create a public NAT gateway with access to the Docker website. For instructions, see [NAT gateways](#).
2. Navigate to the terminal of the Studio application that you want to install Docker in.
3. To return the operating system of the application, run the following command from the terminal:

```
cat /etc/os-release
```

4. Install Docker following the instructions for the operating system of the application in the [Amazon SageMaker AI Local Mode Examples repository](#).

For example, install Docker on Ubuntu following the script at [https://github.com/aws-samples/amazon-sagemaker-local-mode/blob/main/sagemaker\\_studio\\_docker\\_cli\\_install/sagemaker-ubuntu-focal-docker-cli-install.sh](https://github.com/aws-samples/amazon-sagemaker-local-mode/blob/main/sagemaker_studio_docker_cli_install/sagemaker-ubuntu-focal-docker-cli-install.sh) with the following considerations:

- If chained commands fail, run commands one at a time.
- Studio only supports Docker version 20.10.X. and Docker Engine API version 1.41.
- The following packages aren't required to use the Docker CLI in Studio and their installation can be skipped:
  - containerd.io
  - docker-ce
  - docker-buildx-plugin

 **Note**

You do not need to start the Docker service in your applications. The instance that hosts the Studio application runs Docker service by default. All Docker API calls are routed through the Docker service automatically.

5. Use the exposed Docker socket for Docker interactions within Studio applications. By default, the following socket is exposed:

```
unix:///docker/proxy.sock
```

The following Studio application environmental variable for the default USER uses this exposed socket:

```
DOCKER_HOST
```

## No internet access

If the domain is created in VPC-only mode with no internet access, use the following steps to install Docker.

1. Navigate to the terminal of the Studio application that you want to install Docker in.

- Run the following command from the terminal to return the operating system of the application:

```
cat /etc/os-release
```

- Download the required Docker .deb files to your local machine. For instructions about downloading the required files for the operating system of the Studio application, see [Install Docker Engine](#).

For example, install Docker from a package on Ubuntu following the steps 1–4 in [Install from a package](#) with the following considerations:

- Install Docker from a package. Using other methods to install Docker will fail.
- Install the latest packages corresponding to Docker version 20.10.X.
- The following packages aren't required to use the Docker CLI in Studio. You don't need to install the following:
  - containerd.io
  - docker-ce
  - docker-buildx-plugin

 **Note**

You do not need to start the Docker service in your applications. The instance that hosts the Studio application runs Docker service by default. All Docker API calls are routed through the Docker service automatically.

- Upload the .deb files to the Amazon EFS file system or to the Amazon EBS file system of the application.
- Manually install the docker-ce-cli and docker-compose-plugin .deb packages from the Studio application terminal. For more information and instructions, see step 5 in [Install from a package](#) on the Docker docs website.
- Use the exposed Docker socket for Docker interactions within Studio applications. By default, the following socket is exposed:

```
unix:///docker/proxy.sock
```

The following Studio application environmental variable for the default USER uses this exposed socket:

DOCKER\_HOST

## View your Studio running instances, applications, and spaces

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

The following topics include information and instructions about how to view your Studio running instances, applications, and spaces. For more information about Studio spaces, see [Amazon SageMaker Studio spaces](#).

## View your Studio running instances and applications

The **Running instances** page gives information about all running application instances that were created in Amazon SageMaker Studio by the user, or were shared with the user.

You can view and stop running instances for all of your applications and spaces. If an instance is stopped, it does not appear on this page. Stopped instances can be viewed from the landing page for their respective application types.

You can view a list of running applications and their details in Studio.

### To view running instances

1. Launch Studio following the steps in [Launch Amazon SageMaker Studio](#).
2. On the left navigation pane, choose **Running instances**.
3. From the **Running instances** page, you can view a list of running applications and details about those applications.

To view non-running instances, from the left navigation pane choose, the relevant application under **Applications**. The non-running applications will have the **Stopped** status under the **Status** column.

## View your Studio spaces

The **Spaces** section within your **Domain details** page gives information about Studio spaces within your domain. You can view, create, and delete spaces on this page.

The spaces that you can view in the **Spaces** section are running spaces for the following:

- JupyterLab private space. For information about JupyterLab, see [SageMaker JupyterLab](#).
- Code Editor private space. For information about Code Editor, based on Code-OSS, Visual Studio Code - Open Source, see [Code Editor in Amazon SageMaker Studio](#).
- Studio Classic shared space. For information about Studio Classic shared space, see [Collaboration with shared spaces](#).

There are no spaces for SageMaker Canvas, Studio Classic (private), or RStudio.

### To view Studio spaces in a domain

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
3. Choose the domain where you want to view the spaces.
4. On the **Domain details** page, choose the **Space management** tab to open the **Spaces** section.

## Stop and delete your Studio running applications and spaces

The following page includes information and instructions on how to stop and delete unused Amazon SageMaker Studio resources to avoid unwanted additional costs. For the Studio resources you no longer you wish to use, you will need to both:

- Stop the application: This stops both the application and deletes the instance that the application is running on. Once you stop an application you can start it back up again.
- Delete the space: This deletes the Amazon EBS volume that was created for the application and instance.

**⚠️ Important**

If you delete the space, you will lose access to the data within that space. Do not delete the space unless you're sure that you want to.

For more information about the differences between Studio spaces and applications, see [View your Studio running instances, applications, and spaces](#).

**Topics**

- [Stop your Amazon SageMaker Studio application](#)
- [Delete a Studio space](#)

## Stop your Amazon SageMaker Studio application

To avoid additional charges from unused running applications, you must stop them. The following includes information on what stopping an application does and how to do it.

- The following instructions uses the [DeleteApp](#) API to stop the application. This also stops the instance that the application is running on.
- After you stop an application, you can start up the application again later.
  - When you stop an application, the files in the space will persist. You can run the application again and expect to have access to the same files that are stored in the space, as you did before deleting the application.
  - When you stop an application, the *metadata* for the application will be deleted within 24 hours. For more information, see the note in the `CreationTime` response element for the [DescribeApp](#) API.

**ℹ️ Note**

If the service detects that an application is unhealthy, it assumes the [AmazonSageMakerNotebooksServiceRolePolicy](#) service linked role and deletes the application using the [DeleteApp](#) API.

The following tabs provide instructions to stop an application from your domain using the Studio UI, the SageMaker AI console, or the AWS CLI.

 **Note**

To view and stop all of your Studio running instances in one location, we recommend the [Stop applications using the Studio UI](#) workflow from the following options.

## Stop applications using the Studio UI

To stop your Studio applications using the Studio UI, use the following instructions.

### To delete your applications (Studio UI)

1. Launch Studio. This process may differ depending on your setup. For information about launching Studio, see [Launch Amazon SageMaker Studio](#).
2. From the left navigation pane, choose **Running instances**.

If the table on the page is empty, you don't have any running instances or applications in your spaces.

3. In the table under the **Name** and **Application** columns, find the space name and the application that you want to stop.
4. Choose the corresponding **Stop** button to stop the application.

## Stop applications using the SageMaker AI console

To view or stop Studio running instances from a centralized location, see [Stop applications using the Studio UI](#). Otherwise, use the following instructions.

In the SageMaker AI console, you can only stop the running Studio applications for the spaces that you are able to view in the **Spaces** section of the console. For a list of the viewable spaces, see [View your Studio spaces](#).

These steps show how to stop your Studio applications by using the SageMaker AI console.

### To stop your applications (SageMaker AI console)

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.

2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
3. Choose the domain that you want to revert.
4. On the **Domain details** page, choose the **Space management** tab.
- 5.

**⚠️ Important**

In the **Space management** tab, you have the option to delete the space. There is a difference between deleting the space and deleting an application. If you delete the space, you will lose access to the data within that space. Do not delete the space unless you're sure that you want to.

To stop the application, in the **Space management** tab and under the **Name** column, choose the space for the application.

6. In the **Apps** section and under the **App type** column, search for the app to stop.
7. Under the **Action** column, choose the corresponding **Delete app** button.
8. In the pop-up box, choose **Yes, delete app**. After you do so the delete input field becomes available.
9. Enter **delete** in the delete input field to confirm deletion.
10. Choose **Delete**.

### Stop your domain applications using the AWS CLI

To view or stop any of your Studio running instances from a centralized location, see [Stop applications using the Studio UI](#). Otherwise, use the following instructions.

The following code examples use the [DeleteApp](#) API to stop an application in an example domain.

To stop your running **JupyterLab** or **Code Editor** instances, use the following code example:

```
aws sagemaker delete-app \
--domain-id example-domain-id \
--region AWS Region \
--app-name default \
--app-type example-app-type \
--space-name example-space-name
```

- To obtain your *example-domain-id*, use the following instructions:

## To get *example-domain-id*

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
  2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
  3. Choose the relevant domain.
  4. On the **Domain details** page, choose the **Domain settings** tab.
  5. Copy the **Domain ID**.
- To obtain your *AWS Region*, use the following instructions to ensure you are using the correct AWS Region for your domain:

## To get *AWS Region*

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
  2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
  3. Choose the relevant domain.
  4. On the **Domain details** page, verify that this is the relevant domain.
  5. Expand the region dropdown list from the top right of the SageMaker AI console, and use the corresponding AWS Region ID to the right of your AWS Region name. For example, us-west-1.
- For *example-app-type*, use the application type that's relevant to the application that you want to stop. For example, replace *example-app-type* with one of the following application types:
    - JupyterLab application type: JupyterLab. For information about JupyterLab, see [SageMaker JupyterLab](#).
    - Code Editor application type: CodeEditor. For information about Code Editor, based on Code-OSS, Visual Studio Code - Open Source, see [Code Editor in Amazon SageMaker Studio](#).
  - To obtain your *example-space-name*, use the following steps:

## To get *example-space-name*

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
3. Choose the relevant domain.
4. On the **Domain details** page, choose the **Space management** tab.

## 5. Copy the relevant space name.

To stop running instances for **SageMaker Canvas**, **Studio Classic**, or **RStudio**, use the following code example:

```
aws sagemaker delete-app \
--domain-id example-domain-id \
--region AWS Region \
--app-name default \
--app-type example-app-type \
--user-profile example-user-name
```

- For *example-app-type*, use the application type relevant to the application that you want to stop. For example, replace *example-app-type* with one of the following application types:
  - SageMaker Canvas application type: Canvas. For information about SageMaker Canvas, see [Amazon SageMaker Canvas](#).
  - Studio Classic application type: JupyterServer. For information about Studio Classic, see [Amazon SageMaker Studio Classic](#).
  - RStudio application type: RStudioServerPro. For information about RStudio, see [RStudio on Amazon SageMaker AI](#).
- To obtain your *example-user-name*, navigate to the **Domain details** page.
  - Next, choose the **User profiles** tab, and copy the relevant space name.

For alternative instructions to stop your running Studio applications, see:

- JupyterLab: [Delete unused resources](#).
- Code Editor: [Shut down Code Editor resources](#).
- SageMaker Canvas: [Logging out of Amazon SageMaker Canvas](#).
- Studio Classic: [Shut Down and Update SageMaker Studio Classic and Studio Classic Apps](#).
- RStudio: [Shut down RStudio](#).

## Delete a Studio space

### Important

After you delete your space, you will lose all of the data stored in the space. We recommend that you back up your data before deleting your space.

You will need to have administrator permissions, or at least have permissions to update domain, IAM, and Amazon S3, to delete a Studio space.

- Spaces are used to manage the storage and resource needs of the relevant application. When you delete a space, the storage volume also deletes. Therefore, you lose access to the files stored on that space. For more information about Studio spaces, see [Amazon SageMaker Studio spaces](#).

We recommend that you back up your data if you choose to delete a space.

- After you delete a space, you can't access that space again.

You can delete the Studio spaces that are viewable in the **Spaces** section of the console. For a list of the viewable spaces, see [View your Studio spaces](#).

There are no spaces for SageMaker Canvas, Studio Classic (private), and RStudio. To stop and delete your SageMaker Canvas, Studio Classic (private), or RStudio applications, see [Stop your Amazon SageMaker Studio application](#).

### Delete a space using the SageMaker AI console

The **Spaces** section within your **Domain details** page gives information about Studio spaces within your domain. You can view, create, and delete spaces on this page.

#### To view Studio spaces in a domain

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
3. Choose the domain where you want to view the spaces.
4. On the **Domain details**, choose **Space management** to open the **Spaces** section.
5. Select the space to delete.

6. Choose **Delete**.
7. In the pop-up box titled **Delete space**, you have two options:
  - If you already shut down all applications in the space, choose **Yes, delete space**.
  - If you still have applications running in the space, choose **Yes, shut down all apps and delete space**.
8. Enter **delete** in the delete input field to confirm deletion.
9. To delete the space, you have two options:
  - If you already shut down all applications in the space, choose **Delete space**.
  - If you still have applications running in the space, choose **Shut down all apps and delete space**.

## Delete a space using the AWS CLI

Before you can delete a space using the AWS CLI, you must delete the application associated with it. For information about stopping your Studio applications, see [Stop your Amazon SageMaker Studio application](#).

Use the following AWS CLI command to delete a space within a domain:

```
aws sagemaker delete-space \
--domain-id example-domain-id \
--region AWS Region \
--space-name example-space-name
```

- To obtain your *example-domain-id*, use the following instructions:

### To get *example-domain-id*

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
  2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
  3. Choose the relevant domain.
  4. On the **Domain details** page, choose the **Domain settings** tab.
  5. Copy the **Domain ID**.
- To obtain your *AWS Region*, use the following instructions to ensure you are using the correct AWS Region for your domain:

## To get **AWS Region**

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
  2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
  3. Choose the relevant domain.
  4. On the **Domain details** page, verify that this is the relevant domain.
  5. Expand the region dropdown list from the top right of the SageMaker AI console, and use the corresponding AWS Region ID to the right of your AWS Region name. For example, us-west-1.
- To obtain your *example-space-name*, use the following steps:

## To get *example-space-name*

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation pane, expand **Admin configurations** and choose **Domains**.
3. Choose the relevant domain.
4. On the **Domain details** page, choose the **Space management** tab.
5. Copy the relevant space name.

## SageMaker Studio image support policy

### **⚠ Important**

Currently, all packages in SageMaker Distribution images are licensed for use with Amazon SageMaker AI and do not require additional commercial licenses. However, this might be subject to change in the future, and we recommend reviewing the licensing terms regularly for any updates.

Amazon SageMaker Distribution is a set of Docker images available on SageMaker Studio that include popular frameworks for machine learning, data science, and visualization.

The images include deep learning frameworks like PyTorch, TensorFlow and Keras; popular Python packages like numpy, scikit-learn and pandas; and IDEs like JupyterLab and Code Editor, based on

Code-OSS, Visual Studio Code - Open Source. The distribution contains the latest versions of all these packages such that they are mutually compatible.

This page details the support policy and availability for SageMaker Distribution Images on SageMaker Studio.

## Versioning, release cadence, and support policy

The table below outlines the release schedule for SageMaker Distribution Image versions and their planned support. AWS provides ongoing functionality and security updates for supported image versions. New minor versions are released for supported major versions, and supported minor versions receive ongoing functionality and security patches. In some cases, an image version may need to be designated end of support earlier than originally planned if (a) security issues cannot be addressed while maintaining semantic versioning guidelines or (b) any of our major dependencies, like Python, reach end-of-life. AWS releases ad-hoc major or minor versions on an as-needed basis.

Version	Description	Release cadence	Planned support
Major	Amazon SageMaker Distribution's major version releases involve upgrading all of its core dependencies to the latest compatible versions. These major releases may also add or remove packages as part of the update. Major versions are denoted by the first number in the version string, such as 1.0, 2.0, or 3.0.	6 months	12 months
Minor	Amazon SageMaker Distribution's minor version releases include upgrading all of its core dependencies to the latest compatible minor versions within the same major version. SageMaker Distribution can add new packages during a minor version release. Minor versions are denoted by the second number in the version string, for example, 1.1, 1.2, or 2.1.	1 month	6 months

Version	Description	Release cadence	Planned support
Patch	<p>Amazon SageMaker Distribution's patch version releases include updating all of its core dependencies to the latest compatible patch versions within the same minor version. SageMaker Distribution does not add or remove any packages during a patch version release. Patch versions are denoted by the third number in the version string, for example, 1.1.1, 1.2.1, or 2.1.3. Since patch versions are generally released for fixing security vulnerabilities, we recommend always upgrading to the newest patch version when they become available.</p>	As necessary for fixing security vulnerabilities	Until new patch version is released

Each major version of the Amazon SageMaker Distribution is available for 18 months. During the first 12 months, new minor versions are released monthly. For the remaining 6 months, the existing minor versions continue to be supported.

## Supported image versions

The tables below list the supported SageMaker Distribution image versions, their planned end of support dates, and their availability on SageMaker Studio. For image versions where support ends sooner than the planned end of support date, the versions continue to be available on Studio until the designated availability date. You can continue using the image to launch applications for up to 90 days or until the availability date on Studio, whichever comes first. For more information about such cases, reach out to Support.

You can migrate to a newer supported version as soon as possible to ensure that you receive ongoing functionality and security updates. When choosing an image version in SageMaker Studio, we recommend that you choose a supported image version from the tables below.

## Supported major versions

The following table lists the supported SageMaker Distribution major image versions.

Image version	Supported until	Description
1.x.x	Apr 30th, 2025	SageMaker Distribution major version 1 is built with Python 3.10.
2.x.x	Aug 25th, 2025	SageMaker Distribution major version 2 is built with Python 3.11.

## CPU image minor versions

The following table lists the supported SageMaker Distribution minor image versions for CPUs.

Image version	Amazon ECR image URI	Planned end of support date	Availability on Studio until	Release notes
2.3.x	public.ecr.aws/sagemaker/sagemaker-distribution:2.3-cpu	July 27th, 2025	July 27th, 2025	<a href="#">Release notes</a>
2.2.x	public.ecr.aws/sagemaker/sagemaker-distribution:2.2-cpu	May 15th, 2025	May 15th, 2025	<a href="#">Release notes</a>
2.1.x	public.ecr.aws/sagemaker/sagemaker-distribution:2.1-cpu	Apr 25th, 2025	Apr 25th, 2025	<a href="#">Release notes</a>
2.0.x	public.ecr.aws/sagemaker/sagemaker-distribution:2.0-cpu	Feb 25th, 2025	Apr 21st, 2025	<a href="#">Release notes</a>
1.12.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.12-cpu	July 23rd, 2025	July 23rd, 2025	<a href="#">Release notes</a>
1.11.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.11-cpu	Apr 1st, 2025	Apr 1st, 2025	<a href="#">Release notes</a>
1.10.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.10-cpu	Feb 5th, 2025	Apr 10th, 2025	<a href="#">Release notes</a>

Image version	Amazon ECR image URI	Planned end of support date	Availability on Studio until	Release notes
1.9.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.9-cpu	Jan 15th, 2025	Apr 10th, 2025	<a href="#">Release notes</a>
1.8.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.8-cpu	Dec 31st, 2024	Apr 10th, 2025	<a href="#">Release notes</a>
1.7.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.7-cpu	Dec 15th, 2024	Apr 10th, 2025	<a href="#">Release notes</a>
1.6.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.6-cpu	Dec 15th, 2024	Apr 10th, 2025	<a href="#">Release notes</a>

## GPU image minor versions

The following table lists the supported SageMaker Distribution minor image versions for GPUs.

Image version	Amazon ECR image URI	Planned end of support date	Availability on Studio until	Release notes for newest patch
2.3.x	public.ecr.aws/sagemaker/sagemaker-distribution:2.3-gpu	July 27th, 2025	July 27th, 2025	<a href="#">Release notes</a>
2.2.x	public.ecr.aws/sagemaker/sagemaker-distribution:2.2-gpu	May 15th, 2025	May 15th, 2025	<a href="#">Release notes</a>
2.1.x	public.ecr.aws/sagemaker/sagemaker-distribution:2.1-gpu	Apr 25th, 2025	Apr 25th, 2025	<a href="#">Release notes</a>
2.0.x	public.ecr.aws/sagemaker/sagemaker-distribution:2.0-gpu	Feb 25th, 2025	Apr 21st, 2025	<a href="#">Release notes</a>

Image version	Amazon ECR image URI	Planned end of support date	Availability on Studio until	Release notes for newest patch
1.12.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.12-gpu	July 23rd, 2025	July 23rd, 2025	<a href="#">Release notes</a>
1.11.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.11-gpu	Apr 1st, 2025	Apr 1st, 2025	<a href="#">Release notes</a>
1.10.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.10-gpu	Feb 5th, 2025	Apr 10th, 2025	<a href="#">Release notes</a>
1.9.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.9-gpu	Jan 15th, 2025	Apr 10th, 2025	<a href="#">Release notes</a>
1.8.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.8-gpu	Dec 31st, 2024	Apr 10th, 2025	<a href="#">Release notes</a>
1.7.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.7-gpu	Dec 15th, 2024	Apr 10th, 2025	<a href="#">Release notes</a>
1.6.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.6-gpu	Dec 15th, 2024	Apr 10th, 2025	<a href="#">Release notes</a>

## Unsupported images

The following table lists unsupported SageMaker Distribution image versions.

Image version	Amazon ECR image URI	End of support date	Availability on Studio until
1.5.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.5-cpu	Oct 31st 2024	Oct 31st 2024

Image version	Amazon ECR image URI	End of support date	Availability on Studio until
	public.ecr.aws/sagemaker/sagemaker-distribution:1.5-gpu		
1.4.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.4-cpu	Oct 31st 2024	Oct 31st 2024
	public.ecr.aws/sagemaker/sagemaker-distribution:1.4-gpu		
1.3.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.3-cpu	Jun 28th, 2024	Oct 1st, 2024
1.2.x	public.ecr.aws/sagemaker/sagemaker-distribution:1.2-cpu	Jun 28th, 2024	Oct 1st, 2024

## Frequently asked questions

### What constitutes a major image version release?

Major image versions are released every 6 months. A major image version release for Amazon SageMaker Distribution involves upgrading all core dependencies to the latest compatible versions and may include adding or removing packages. Python framework is only upgraded with new major version releases. For example, with major version 2 release, Python framework was upgraded from 3.10 to 3.11, PyTorch was upgraded from 2.0 to 2.3, TensorFlow was upgraded from 2.14 to 2.17, Autogluon was upgraded from 0.8 to 1.1, and 4 packages were added to the image.

### What constitutes a minor image version release?

Minor image versions are released for all supported major versions monthly. A minor image version release for Amazon SageMaker Distribution involves upgrading all core dependencies except Python and CUDA to the latest compatible minor versions within the same major version and may include adding new packages. For example, with a minor version release, langchain might be upgraded from 0.1 to 0.2 and jupyter-ai from 2.18 to 2.20.

### What constitutes a patch image version release?

Patch image versions are released as necessary to fix security vulnerabilities. A patch image version release for Amazon SageMaker Distribution involves updating all of its core dependencies to the latest compatible patch versions within the same minor version. SageMaker Distribution does not add or remove any packages during a patch version release. For example, with a patch version release, matplotlib might be upgraded from 3.9.1 to 3.9.2 and boto3 from 1.34.131 to 1.34.162.

## Where can I find the packages available in a specific image version?

Each image version has a `release.md` file in the [GitHub repository's build\\_artifacts folder](#), showing all packages and package versions for CPU and GPU images. Separate changelog files for CPU and GPU versions detail package upgrades. Changelogs compare the new image version to the previous. For example, version 1.9.0 compares to the latest patch version of 1.8, version 1.9.1 compares to 1.9.0, and version 2.0.0 compares to the latest patch version of the latest minor version available at the time.

## How are images scanned for Common Vulnerabilities and Exposures (CVEs)?

Amazon SageMaker AI leverages [Amazon Elastic Container Registry \(Amazon ECR\) enhanced scanning](#) to automatically detect vulnerabilities and fixes for SageMaker Distribution Images. AWS continuously runs ECR enhanced scanning for the latest patch version of all supported image versions. When vulnerabilities are detected and a fix is available, AWS releases an updated image version to remediate the issue.

## Can I still use older images after an image is no longer supported?

Images are available on SageMaker Studio until the designated availability date. Older images remain available in ECR after they reach end of support and are removed from Studio. You can download older image versions from ECR and [create a custom SageMaker image](#). However, we highly recommend upgrading to a supported image version that continuously receives security updates and bug fixes. Customers who build their own custom images are responsible for scanning and patching their images. For more information, see the [AWS Shared Responsibility model](#).

### **Important**

SageMaker Distribution v0.x.y is only used in Studio Classic. SageMaker Distribution v1.x.y is only used in JupyterLab.

# Amazon SageMaker Studio pricing

## Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

There is no additional charge for using the Amazon SageMaker Studio UI.

The following do incur costs:

- Amazon Elastic Block Store or Amazon Elastic File System volumes that are mounted with your applications.
- Any jobs and resources that users launch from Studio applications.
- Launching a JupyterLab application, even if no resources or jobs launched in the application.

For information about how Amazon SageMaker Studio Classic is billed, see [Amazon SageMaker Studio Classic Pricing](#).

For more information about billing along with pricing examples, see [Amazon SageMaker Pricing](#).

## Troubleshooting

## Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

## Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to

those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

This section shows how to troubleshoot common problems in Amazon SageMaker Studio.

## Cannot delete Code Editor, based on Code-OSS, Visual Studio Code - Open Source or JupyterLab application

This issue occurs when a user creates an application from Amazon SageMaker Studio that is only available in Studio, then reverts to the Studio Classic experience as their default. As a result, the user cannot delete an application for Code Editor, based on Code-OSS, Visual Studio Code - Open Source or JupyterLab because they can't access the Studio UI.

To resolve this issue, notify your administrator so that they can delete the application manually using the AWS Command Line Interface (AWS CLI).

## EC2InsufficientCapacityError

This issue occurs when you try to run a space and AWS does not currently have enough available on-demand capacity to fulfill your request.

To resolve this issue, complete the following.

- Wait a few minutes, then resubmit your request. Capacity can shift frequently.
- Run the space with an alternate instance size or type.

### Note

Capacity is available in different Availability Zones. To maximize capacity availability for users, we recommend setting up subnets in all Availability Zones. Studio retries all available Availability Zones for the domain.

Instance type availability differs between regions. For a list of supported instances types per Region, see [Amazon SageMaker AI pricing](#)

The following table lists instance families and their recommended alternatives.

Instance family	CPU Type	vCPUs	Memory (GiB)	GPU type	GPUs	GPU Memory (GiB)	Recommended alternative
G4dn	2nd Generation Intel Xeon Scalable Processors	4 to 96	16 to 384	NVIDIA T4 Tensor Core	1 to 8	16 per GPU	G6
G5	2nd generation AMD EPYC processors	4 to 192	16 to 768	NVIDIA A10G Tensor core	1 to 8	24 per GPU	G6e
G6	3rd generation AMD EPYC processors	4 to 192	16 to 768	NVIDIA L4 Tensor Core	1 to 8	24 per GPU	G4dn
G6e	3rd generation AMD EPYC	4 to 192	32 to 1536	NVIDIA L40S Tensor Core	1 to 8	48 per GPU	G5, P4

Instance family	CPU Type	vCPUs	Memory (GiB)	GPU type	GPUs	GPU Memory (GiB)	Recommended alternative
	processors						
P3	Intel Xeon Scalable Processor s	8 to 96	61 to 768	NVIDIA Tesla V100	1 to 8	16 per GPU (32 per GPU for P3dn)	G6e, P4
P4	2nd Generation Intel Xeon Scalable processor s	96	1152	NVIDIA A100 Tensor Core	8	320 (640 for P4de)	G6e
P5	3rd Gen AMD EPYC processor s	192	2000	NVIDIA H100 Tensor Core	8	640	P4de

## Insufficient limit (quota increase required)

This issue occurs when you get the following error when running a space. This error means that you have reached the limit on the number of instances of that type that you can launch in a Region. When you create your AWS account, we set default limits on the number of instances you can run in each Region.

Error when creating application for space: ... : The account-level service limit is X Apps, with current utilization Y Apps and a request delta of 1 Apps. Please use Service Quotas to request an increase for this quota.

To resolve this issue, request an instance limit increase for the Region that you are launching the space is. For more information, see [Requesting a quota increase](#).

## Amazon SageMaker Studio Classic

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker Studio Classic is a web-based integrated development environment (IDE) for machine learning (ML). Studio Classic lets you build, train, debug, deploy, and monitor your ML models. Studio Classic includes all of the tools you need to take your models from data preparation to experimentation to production with increased productivity. In a single visual interface, you can do the following tasks:

- Write and run code in Jupyter notebooks
- Prepare data for machine learning
- Build and train ML models
- Deploy the models and monitor the performance of their predictions
- Track and debug ML experiments
- Collaborate with other users in real time

For information on the onboarding steps for Studio Classic, see [Amazon SageMaker AI domain overview](#).

For information about collaborating with other users in real time, see [Collaboration with shared spaces](#).

For the AWS Regions supported by Studio Classic, see [Supported Regions and Quotas](#).

## Studio Classic maintenance phase plan

The following table gives information about the timeline for when Amazon SageMaker Studio Classic entered its extended maintenance phase.

Date	Description
12/31/2024	Starting December 31st, Studio Classic reaches end of maintenance. At this point, Studio Classic will no longer receive updates and security fixes. All new domains will be created with Amazon SageMaker Studio as the default.
1/31/2025	Starting January 31st, users will no longer be able to create new JupyterLab 3 notebooks in Studio Classic. Users will also not be able to restart or update existing notebooks. Users will be able to access existing Studio Classic applications from Studio only to delete or stop existing notebooks.

 **Note**

Your existing Studio Classic domain is not automatically migrated to Studio. For information about migrating, see [Migration from Amazon SageMaker Studio Classic](#).

### Topics

- [Studio Classic Features](#)
- [Amazon SageMaker Studio Classic UI Overview](#)
- [Launch Amazon SageMaker Studio Classic](#)
- [JupyterLab Versioning](#)
- [Use the Amazon SageMaker Studio Classic Launcher](#)
- [Use Amazon SageMaker Studio Classic Notebooks](#)
- [Customize Amazon SageMaker Studio Classic](#)
- [Perform Common Tasks in Amazon SageMaker Studio Classic](#)
- [Amazon SageMaker Studio Classic Pricing](#)

- [Troubleshooting Amazon SageMaker Studio Classic](#)

## Studio Classic Features

Studio Classic includes the following features:

- [SageMaker Autopilot](#)
- [SageMaker Clarify](#)
- [SageMaker Data Wrangler](#)
- [SageMaker Debugger](#)
- [SageMaker Experiments](#)
- [SageMaker Feature Store](#)
- [SageMaker JumpStart](#)
- [Amazon SageMaker Pipelines](#)
- [SageMaker Model Registry](#)
- [SageMaker Projects](#)
- [SageMaker Studio Classic Notebooks](#)
- [SageMaker Studio Universal Notebook](#)

## Amazon SageMaker Studio Classic UI Overview

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker Studio Classic extends the capabilities of JupyterLab with custom resources that can speed up your Machine Learning (ML) process by harnessing the power of AWS compute. Previous users of JupyterLab will notice the similarity of the user interface. The most prominent additions are detailed in the following sections. For an overview of the original JupyterLab interface, see [The JupyterLab Interface](#).

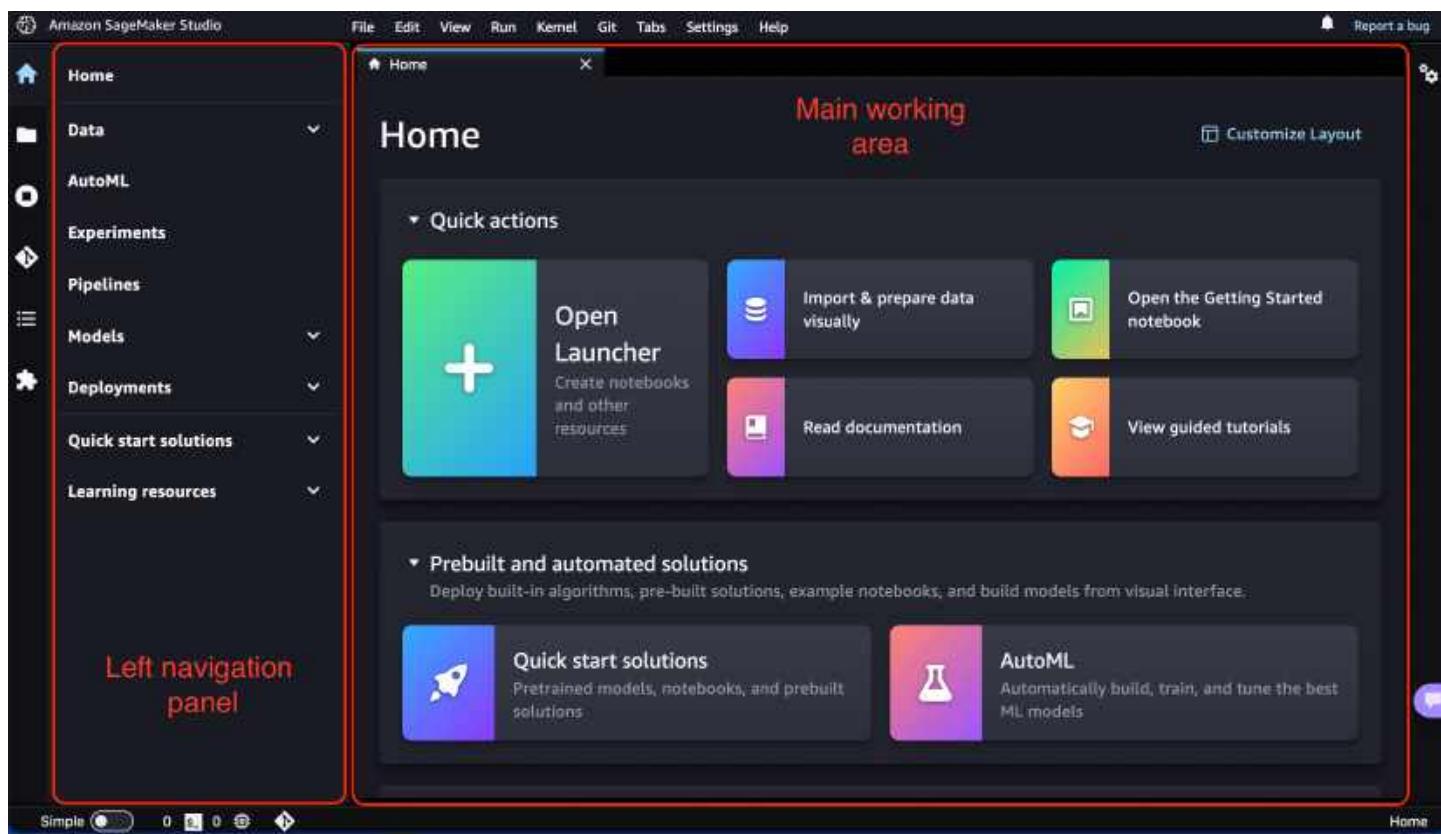
The following image shows the default view upon launching Amazon SageMaker Studio Classic. The *left navigation panel* displays all top-level categories of features, and a [Studio Classic Home page](#) is open in the *main working area*. Come back to this central point of orientation by choosing the **Home** icon



)

at any time, then selecting the **Home** node in the navigation menu.

Try the **Getting started notebook** for an in-product hands-on guide on how to set up and get familiar with Amazon SageMaker Studio Classic features. On the **Quick actions** section of the Studio Classic Home page, choose **Open the Getting started notebook**.



### Note

This chapter is based on Studio Classic's updated user interface (UI) available on version v5.38.x and above on JupyterLab3.

- To retrieve your version of Studio Classic UI, from the [Studio Classic Launcher](#), open a System Terminal, then
  1. Run `conda activate studio`

2. Run `jupyter labextension list`
  3. Search for the version displayed after `@amzn/sagemaker-ui` version in the output.
- For information about updating Amazon SageMaker Studio Classic, see [Shut down and Update SageMaker Studio Classic](#).

## Topics

- [Studio Classic Home page](#)
- [Studio Classic layout](#)

## Studio Classic Home page

The Home page provides access to common tasks and workflows. In particular, it includes a list of **Quick actions** for common tasks such as **Open Launcher** to create notebooks and other resources and **Import & prepare data visually** to create a new flow in Data Wrangler. The **Home** page also offers tooltips on key controls in the UI.

The **Prebuilt and automated solutions** help you get started quickly with SageMaker AI's low-code solutions such as Amazon SageMaker JumpStart and Autopilot.

In **Workflows and tasks**, you can find a list of relevant tasks for each step of your ML workflow that takes you to the right tool for the job. For example, **Transform, analyse, and export data** takes you to Amazon SageMaker Data Wrangler and opens the workflow to create a new data flow, or **View all experiments** takes you to SageMaker Experiments and opens the experiments list view.

Upon Studio Classic launch, the **Home** page is open in the main working area. You can customize your SageMaker AI **Home** page by choosing the **Customize Layout** icon



at the top right of the **Home** tab.

## Studio Classic layout

The Amazon SageMaker Studio Classic interface consists of a *menu bar* at the top, a collapsible *left sidebar* displaying a variety of icons such as the **Home** icon and the **File Browser**, a *status bar* at the bottom of the screen, and a *central area* divided horizontally into two panes. The left pane is

a collapsible *navigation panel*. The right pane, or main working area, contains one or more tabs for resources such as launchers, notebooks, terminals, metrics, and graphs, and can be further divided.

**Report a bug** in Studio Classic or choose the notification icon



)

to view notifications from Studio Classic, such as new Studio Classic versions and new SageMaker AI features, on the right corner of the menu bar. To update to a new version of Studio Classic, see [Shut Down and Update SageMaker Studio Classic and Studio Classic Apps](#).

The following sections describe the Studio Classic main user interface areas.

## Left sidebar

The *left sidebar* includes the following icons. When hovering over an icon, a tooltip displays the icon name. A single click on an icon opens up the left navigation panel with the described functionality. A double click minimizes the left navigation panel.

Icon	Description
	<p><b>Home</b></p> <p>Choose the <b>Home</b> icon to open a top-level navigation menu in the <i>left navigation</i> panel.</p> <p>Using the <b>Home</b> navigation menu, you can discover and navigate to the right tools for each step of your ML workflow. The menu also provides shortcuts to quick-start solutions and learning resources such as documentation and guided tutorials.</p> <p>The menu categories group relevant features together. Choosing <b>Data</b>, for example, expands the relevant SageMaker AI capabilities for your data preparations tasks. From here, you can prepare your data with Data Wrangler, create and store ML features with Amazon SageMaker Feature Store, and manage Amazon EMR clusters for large-scale data processing. The categories are ordered following a typical ML workflow from preparing data, to building, training, and deploying ML models (data, pipelines, models, and deployments).</p> <p>When you choose a specific node (such as Data Wrangler), a corresponding page opens in the main working area.</p>

Icon	Description
	Choose <b>Home</b> in the navigation menu to open the <a href="#">Studio Classic Home page</a>

Icon	Description
	<p><b>File Browser</b></p> <p>The <b>File Browser</b> displays lists of your notebooks, experiments, trials, trial components, endpoints, and low-code solutions.</p> <p>Whether you are in a personal or shared space determines who has access to your files. You can identify which type of space you are in by looking at the top right corner. If you are in a personal app, you see a user icon followed by <i>[user_name]</i> / <b>Personal Studio</b> and if you are in a collaborative space, you see a globe icon followed by "<i>[user_name]</i> / <i>[space_name]</i>".</p> <ul style="list-style-type: none"><li>• <b>Personal Studio Classic app:</b> A private Amazon EFS directory that only you can access.</li><li>• <b>Collaborative space:</b> A shared Amazon EFS directory with other members of your team for group access to notebooks and resources. Working in a shared space allows for real-time team collaboration on notebooks.</li><li>• <b>Studio Classic launcher:</b> Choose the plus (+) sign on the menu at the top of the file browser to open the <a href="#">Amazon SageMaker Studio Classic Launcher</a>.</li><li>• <b>Upload files:</b> Choose the <b>Upload Files</b> icon () to add files to Studio Classic or drag and drop them from your desktop.</li><li>• <b>Open files:</b> Double-click a file to open the file in a new tab or right-click and select <b>Open</b>.</li></ul>

Icon	Description
	<ul style="list-style-type: none"><li><b>Panel management:</b> To work in adjacent files, choose a tab that contains a notebook, Python, or text file, then choose <b>New View for File</b>.</li></ul> <p>For hierarchical entries, a selectable breadcrumb at the top of the browser shows your location in the hierarchy.</p>
	<h3>Property Inspector</h3> <p>The Property Inspector is a notebook cell tools inspector which displays contextual property settings when open.</p>
	<h3>Running Terminals and Kernels</h3> <p>You can check the list of all the <i>kernels</i> and <i>terminals</i> currently running across all notebooks, code consoles, and directories. You can shut down individual resources, including notebooks, terminals, kernels, apps, and instances. You can also shut down all resources in one of these categories at the same time.</p> <p>For more information, see <a href="#">Shut down resources from Amazon SageMaker Studio Classic</a>.</p>
	<h3>Git</h3> <p>You can connect to a Git repository and then access a full range of Git tools and operations.</p> <p>For more information, see <a href="#">Clone a Git Repository in SageMaker Studio Classic</a>.</p>

Icon	Description
	<p><b>Table of Contents</b></p> <p>You can navigate the structure of a document when a notebook or Python files are open.</p> <p>A table of contents is auto-generated in the left navigation panel when you have a notebook, Markdown files, or Python files opened. The entries are clickable and scroll the document to the heading in question.</p>
	<p><b>Extensions</b></p> <p>You can turn on and manage third-party JupyterLab extensions. You can check the already installed extensions and search for extensions by typing the name in the search bar. When you have found the extension you want to install, choose <b>Install</b>. After installing your new extensions, be sure to restart JupyterLab by refreshing your browser.</p> <p>For more information, see <a href="#">JupyterLab Extensions documentation</a>.</p>

## Left navigation panel

The left navigation panel content varies with the Icon selected in the left sidebar.

For example, choosing the **Home** icon displays the navigation menu. Choosing **File browser** lists all the files and directories available in your workspace (notebooks, experiments, data flows, trials, trial components, endpoints, or low-code solutions).

In the navigation menu, choosing a node brings up the corresponding feature page in the main working area. For example, choosing **Data Wrangler** in the **Data** menu opens up the **Data Wrangler** tab listing all existing flows.

## Main working area

The main working area consists of multiple tabs that contain your open notebooks, terminals, and detailed information about your experiments and endpoints. In the main working area, you can arrange documents (such as notebooks and text files) and other activities (such as terminals and code consoles) into panels of tabs that you can resize or subdivide. Drag a tab to the center of a tab

panel to move the tab to the panel. Subdivide a tab panel by dragging a tab to the left, right, top, or bottom of the panel. The tab for the current activity is marked with a colored top border (blue by default).

### Note

All feature pages provide in-product contextual help. To access help, choose **Show information**. The help interface provides a brief introduction to the tool and links to additional resources, such as videos, tutorials, or blogs.

## Launch Amazon SageMaker Studio Classic

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

After you have onboarded to an Amazon SageMaker AI domain, you can launch an Amazon SageMaker Studio Classic application from either the SageMaker AI console or the AWS CLI. For more information about onboarding to a domain, see [Amazon SageMaker AI domain overview](#).

## Topics

- [Launch Studio Classic Using the Amazon SageMaker AI Console](#)
- [Launch Studio Classic Using the AWS CLI](#)

## Launch Studio Classic Using the Amazon SageMaker AI Console

The process to navigate to Studio Classic from the Amazon SageMaker AI Console differs depending on if Studio Classic or Amazon SageMaker Studio are set as the default experience for your domain. For more information about setting the default experience for your domain, see [Migration from Amazon SageMaker Studio Classic](#).

## Topics

- [Prerequisite](#)

## Prerequisite

To complete this procedure, you must onboard to a domain by following the steps in [Onboard to Amazon SageMaker AI domain](#).

### Launch Studio Classic if Studio is your default experience

1. Navigate to Studio following the steps in [Launch Amazon SageMaker Studio](#).
2. From the Studio UI, find the applications pane on the left side.
3. From the applications pane, select **Studio Classic**.
4. From the Studio Classic landing page, select the Studio Classic instance to open.
5. Choose “Open”.

### Launch Studio Classic if Studio Classic is your default experience

When Studio Classic is your default experience, you can launch a Amazon SageMaker Studio Classic application from the SageMaker AI console using the Studio Classic landing page or the Amazon SageMaker AI domain details page. The following sections demonstrate how to launch the Studio Classic application from the SageMaker AI console.

## Launch Studio Classic from the domain details page

The following sections describe how to launch a Studio Classic application from the domain details page. The steps to launch the Studio Classic application after you have navigated to the domain details page differ depending on if you're launching a personal application or a shared space.

### Navigate to the domain details page

The following procedure shows how to navigate to the domain details page.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domain, select the domain that you want to launch the Studio Classic application in.

### Launch a user profile app

The following procedure shows how to launch a Studio Classic application that is scoped to a user profile.

1. On the domain details page, choose the **User profiles** tab.
2. Identify the user profile that you want to launch the Studio Classic application for.
3. Choose **Launch** for your selected user profile, then choose **Studio Classic**.

### Launch a shared space app

The following procedure shows how to launch a Studio Classic application that is scoped to a shared space.

1. On the domain details page, choose the **Space management** tab.
2. Identify the shared space that you want to launch the Studio Classic application for.
3. Choose **Launch Studio Classic** for your selected shared space.

### Launch Studio Classic from the Studio Classic landing page

The following procedure describes how to launch a Studio Classic application from the Studio Classic landing page.

## Launch Studio Classic

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose Studio Classic.
3. Under **Get started**, select the domain that you want to launch the Studio Classic application in. If your user profile only belongs to one domain, you do not see the option for selecting a domain.
4. Select the user profile that you want to launch the Studio Classic application for. If there is no user profile in the domain, choose **Create user profile**. For more information, see [Add user profiles](#).
5. Choose **Launch Studio Classic**. If the user profile belongs to a shared space, choose **Open Spaces**.
6. To launch a Studio Classic application scoped to a user profile, choose **Launch personal Studio Classic**.
7. To launch a shared Studio Classic application, choose the **Launch shared Studio Classic** button next to the shared space that you want to launch into.

## Launch Studio Classic Using the AWS CLI

You can use the AWS Command Line Interface (AWS CLI) to launch Amazon SageMaker Studio Classic by creating a presigned domain URL.

### Prerequisites

Before you begin, complete the following prerequisites:

- Onboard to Amazon SageMaker AI domain. For more information, see [Onboard to Amazon SageMaker AI domain](#).
- Update the AWS CLI by following the steps in [Installing the current AWS CLI Version](#).
- From your local machine, run `aws configure` and provide your AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).

The following code snippet demonstrates how to launch Amazon SageMaker Studio Classic from the AWS CLI using a presigned domain URL. For more information, see [create-presigned-domain-url](#).

```
aws sagemaker create-presigned-domain-url \
--region region \
--domain-id domain-id \
--space-name space-name \
--user-profile-name user-profile-name \
--session-expiration-duration-in-seconds 43200
```

## JupyterLab Versioning

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The Amazon SageMaker Studio Classic interface is based on JupyterLab, which is a web-based interactive development environment for notebooks, code, and data. Studio Classic only supports using JupyterLab 3.

If you created your domain and user profile using the AWS Management Console before 08/31/2022 or using the AWS Command Line Interface before 02/22/23, then your Studio Classic

instance defaulted to JupyterLab 1. After 07/01/2024, you cannot create any Studio Classic applications that run JupyterLab 1.

## JupyterLab 3

JupyterLab 3 includes the following features that are not available in previous versions. For more information about these features, see [JupyterLab 3.0 is released!](#).

- Visual debugger when using the Base Python 2.0 and Data Science 2.0 kernels.
- File browser filter
- Table of Contents (TOC)
- Multi-language support
- Simple mode
- Single interface mode

### Important changes to JupyterLab 3

Consider the following when using JupyterLab 3:

- When setting the JupyterLab version using the AWS CLI, select the corresponding image for your Region and JupyterLab version from the image list in [From the AWS CLI](#).
- In JupyterLab 3, you must activate the studio conda environment before installing extensions. For more information, see [Installing JupyterLab and Jupyter Server extensions](#).
- Debugger is only supported when using the following images:
  - Base Python 2.0
  - Data Science 2.0
  - Base Python 3.0
  - Data Science 3.0

### Restricting default JupyterLab version using an IAM policy condition key

You can use IAM policy condition keys to restrict the version of JupyterLab that your users can launch.

The following policy shows how to limit the JupyterLab version at the domain level.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "Block users from creating JupyterLab 3 apps at the domain level",  
            "Effect": "Deny",  
            "Action": [  
                "sagemaker>CreateDomain",  
                "sagemaker:UpdateDomain"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "ForAnyValue:StringLike": {  
                    "sagemaker:ImageArns": "*image/jupyter-server-3"  
                }  
            }  
        }  
    ]  
}
```

The following policy shows how to limit the JupyterLab version at the user profile level.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "Block users from creating JupyterLab 3 apps at the user profile  
level",  
            "Effect": "Deny",  
            "Action": [  
                "sagemaker>CreateUserProfile",  
                "sagemaker:UpdateUserProfile"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "ForAnyValue:StringLike": {  
                    "sagemaker:ImageArns": "*image/jupyter-server-3"  
                }  
            }  
        }  
    ]  
}
```

The following policy shows how to limit the JupyterLab version at the application level. The CreateApp request must include the image ARN for this policy to apply.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "Block users from creating JupyterLab 3 apps at the application  
level",  
            "Effect": "Deny",  
            "Action": "sagemaker>CreateApp",  
            "Resource": "*",  
            "Condition": {  
                "ForAnyValue:StringLike": {  
                    "sagemaker:ImageArns": "*image/jupyter-server-3"  
                }  
            }  
        }  
    ]  
}
```

## Setting a default JupyterLab version

The following sections show how to set a default JupyterLab version for Studio Classic using either the console or the AWS CLI.

### From the console

You can select the default JupyterLab version to use on either the domain or user profile level during resource creation. To set the default JupyterLab version using the console, see [Amazon SageMaker AI domain overview](#).

### From the AWS CLI

You can select the default JupyterLab version to use on either the domain or user profile level using the AWS CLI.

To set the default JupyterLab version using the AWS CLI, you must include the ARN of the desired default JupyterLab version as part of an AWS CLI command. This ARN differs based on the version and the Region of the SageMaker AI domain.

The following table lists the ARNs of the available JupyterLab versions for each Region:

Region	JL3
us-east-1	arn:aws:sagemaker:us-east-1:08132539 0199:image/jupyter-server-3
us-east-2	arn:aws:sagemaker:us-east-2:42970468 7514:image/jupyter-server-3
us-west-1	arn:aws:sagemaker:us-west-1:74209132 7244:image/jupyter-server-3
us-west-2	arn:aws:sagemaker:us-west-2:23651454 2706:image/jupyter-server-3
af-south-1	arn:aws:sagemaker:af-south-1:5593120 83959:image/jupyter-server-3
ap-east-1	arn:aws:sagemaker:ap-east-1:49364249 6378:image/jupyter-server-3
ap-south-1	arn:aws:sagemaker:ap-south-1:3941030 62818:image/jupyter-server-3
ap-northeast-2	arn:aws:sagemaker:ap-northeast-2:806 072073708:image/jupyter-server-3
ap-southeast-1	arn:aws:sagemaker:ap-southeast-1:492 261229750:image/jupyter-server-3
ap-southeast-2	arn:aws:sagemaker:ap-southeast-2:452 832661640:image/jupyter-server-3
ap-northeast-1	arn:aws:sagemaker:ap-northeast-1:102 112518831:image/jupyter-server-3
ca-central-1	arn:aws:sagemaker:ca-central-1:31090 6938811:image/jupyter-server-3

Region	JL3
eu-central-1	arn:aws:sagemaker:eu-central-1:93669 7816551:image/jupyter-server-3
eu-west-1	arn:aws:sagemaker:eu-west-1:47031725 9841:image/jupyter-server-3
eu-west-2	arn:aws:sagemaker:eu-west-2:71277966 5605:image/jupyter-server-3
eu-west-3	arn:aws:sagemaker:eu-west-3:61554785 6133:image/jupyter-server-3
eu-north-1	arn:aws:sagemaker:eu-north-1:2436375 12696:image/jupyter-server-3
eu-south-1	arn:aws:sagemaker:eu-south-1:5927512 61982:image/jupyter-server-3
eu-south-2	arn:aws:sagemaker:eu-south-2:1273631 02723:image/jupyter-server-3
sa-east-1	arn:aws:sagemaker:sa-east-1:78248440 2741:image/jupyter-server-3
cn-north-1	arn:aws-cn:sagemaker:cn-north-1:3900 48526115:image/jupyter-server-3
cn-northwest-1	arn:aws-cn:sagemaker:cn-northwest-1: 390780980154:image/jupyter-server-3

## Create or update domain

You can set a default JupyterServer version at the domain level by invoking [CreateDomain](#) or [UpdateDomain](#) and passing the `UserSettings.JupyterServerAppSettings.DefaultResourceSpec.SageMakerImageArn` field.

The following shows how to create a domain with JupyterLab 3 as the default, using the AWS CLI:

```
aws --region <REGION> \
sagemaker create-domain \
--domain-name <NEW_DOMAIN_NAME> \
--auth-mode <AUTHENTICATION_MODE> \
--subnet-ids <SUBNET-IDS> \
--vpc-id <VPC-ID> \
--default-user-settings '{
    "JupyterServerAppSettings": {
        "DefaultResourceSpec": {
            "SageMakerImageArn": "arn:aws:sagemaker:<REGION>:<ACCOUNT_ID>:image/jupyter-
server-3",
            "InstanceType": "system"
        }
    }
}'
```

The following shows how to update a domain to use JupyterLab 3 as the default, using the AWS CLI:

```
aws --region <REGION> \
sagemaker update-domain \
--domain-id <YOUR_DOMAIN_ID> \
--default-user-settings '{
    "JupyterServerAppSettings": {
        "DefaultResourceSpec": {
            "SageMakerImageArn": "arn:aws:sagemaker:<REGION>:<ACCOUNT_ID>:image/jupyter-
server-3",
            "InstanceType": "system"
        }
    }
}'
```

## Create or update user profile

You can set a default JupyterServer version at the user profile level by invoking [CreateUserProfile](#) or [UpdateUserProfile](#) and passing the `UserSettings.JupyterServerAppSettings.DefaultResourceSpec.SageMakerImageArn` field.

The following shows how to create a user profile with JupyterLab 3 as the default on an existing domain, using the AWS CLI:

```
aws --region <REGION> \
sagemaker create-user-profile \
--domain-id <YOUR_DOMAIN_ID> \
--user-profile-name <NEW_USERPROFILE_NAME> \
--query UserProfileArn --output text \
--user-settings '{
    "JupyterServerAppSettings": {
        "DefaultResourceSpec": {
            "SageMakerImageArn": "arn:aws:sagemaker:<REGION>:<ACCOUNT_ID>:image/jupyter-
server-3",
            "InstanceType": "system"
        }
    }
}'
```

The following shows how to update a user profile to use JupyterLab 3 as the default, using the AWS CLI:

```
aws --region <REGION> \
sagemaker update-user-profile \
--domain-id <YOUR_DOMAIN_ID> \
--user-profile-name <EXISTING_USERPROFILE_NAME> \
--user-settings '{
    "JupyterServerAppSettings": {
        "DefaultResourceSpec": {
            "SageMakerImageArn": "arn:aws:sagemaker:<REGION>:<ACCOUNT_ID>:image/jupyter-
server-3",
            "InstanceType": "system"
        }
    }
}'
```

## View and update the JupyterLab version of an application from the console

The following shows how to view and update the JupyterLab version of an application.

1. Navigate to the SageMaker AI **domains** page.
2. Select a domain to view its user profiles.
3. Select a user to view their applications.
4. To view the JupyterLab version of an application, select the application's name.

5. To update the JupyterLab version, select **Action**.
6. From the dropdown menu, select **Change JupyterLab version**.
7. From the **Studio Classic settings** page, select the JupyterLab version from the dropdown menu.
8. After the JupyterLab version for the user profile has been successfully updated, restart the JupyterServer application to make the version changes effective. For more information about restarting a JupyterServer application, see [Shut down and Update SageMaker Studio Classic](#).

## Installing JupyterLab and Jupyter Server extensions

In JupyterLab 3, you must activate the studio conda environment before installing extensions. The method for this differs if you're installing the extensions from within Studio Classic or using a lifecycle configuration script.

### Installing Extension from within Studio Classic

To install extensions from within Studio Classic, you must activate the studio environment before you install extensions.

```
# Before installing extensions
conda activate studio

# Install your extensions
pip install <JUPYTER_EXTENSION>

# After installing extensions
conda deactivate
```

### Installing Extensions using a lifecycle configuration script

If you're installing JupyterLab and Jupyter Server extensions in your lifecycle configuration script, you must modify your script so that it works with JupyterLab 3. The following sections show the code needed for existing and new lifecycle configuration scripts.

#### Existing lifecycle configuration script

If you're reusing an existing lifecycle configuration script that must work with both versions of JupyterLab, use the following code in your script:

```
# Before installing extension
```

```
export
AWS_SAGEMAKER_JUPYTERSERVER_IMAGE="${AWS_SAGEMAKER_JUPYTERSERVER_IMAGE:-'jupyter-
server'}"
if [ "$AWS_SAGEMAKER_JUPYTERSERVER_IMAGE" = "jupyter-server-3" ] ; then
    eval "$(conda shell.bash hook)"
    conda activate studio
fi;

# Install your extensions
pip install <JUPYTER_EXTENSION>

# After installing extension
if [ "$AWS_SAGEMAKER_JUPYTERSERVER_IMAGE" = "jupyter-server-3" ]; then
    conda deactivate
fi;
```

## New lifecycle configuration script

If you're writing a new lifecycle configuration script that only uses JupyterLab 3, you can use the following code in your script:

```
# Before installing extension
eval "$(conda shell.bash hook)"
conda activate studio

# Install your extensions
pip install <JUPYTER_EXTENSION>

conda deactivate
```

## Use the Amazon SageMaker Studio Classic Launcher

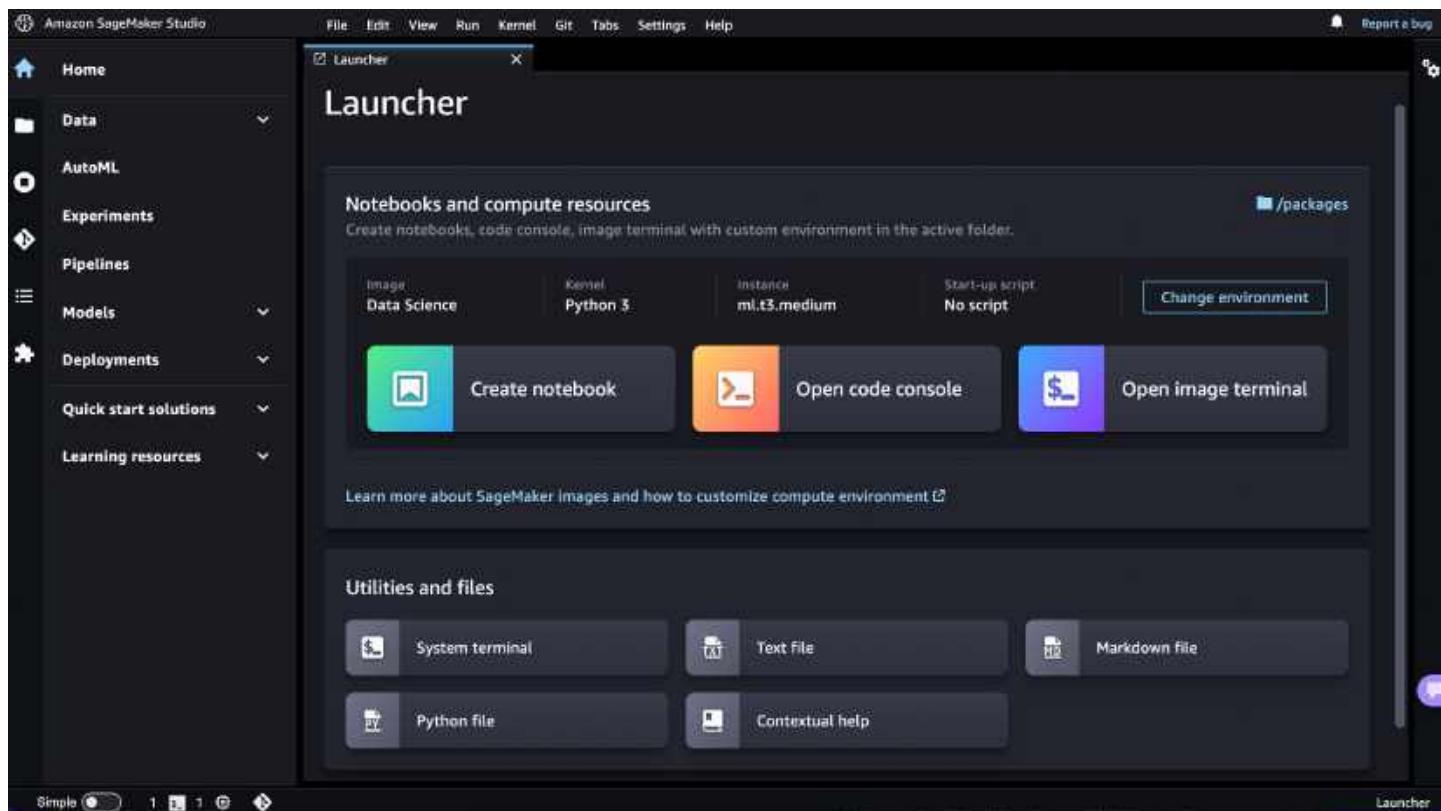
### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You can use the Amazon SageMaker Studio Classic Launcher to create notebooks and text files, and to launch terminals and interactive Python shells.

You can open Studio Classic Launcher in any of the following ways:

- Choose **Amazon SageMaker Studio Classic** at the top left of the Studio Classic interface.
- Use the keyboard shortcut **Ctrl + Shift + L**.
- From the Studio Classic menu, choose **File** and then choose **New Launcher**.
- If the SageMaker AI file browser is open, choose the plus (+) sign in the Studio Classic file browser menu.
- In the **Quick actions** section of the **Home** tab, choose **Open Launcher**. The Launcher opens in a new tab. The **Quick actions** section is visible by default but can be toggled off. Choose **Customize Layout** to turn this section back on.



The Launcher consists of the following two sections:

## Topics

- [Notebooks and compute resources](#)

- [Utilities and files](#)

## Notebooks and compute resources

In this section, you can create a notebook, open an image terminal, or open a Python console.

To create or launch one of those items:

1. Choose **Change environment** to select a SageMaker image, a kernel, an instance type, and, optionally, add a lifecycle configuration script that runs on image start-up. For more information on lifecycle configuration scripts, see [Use lifecycle configurations to customize Studio Classic](#). For more information about kernel updates, see [Change an Image or a Kernel](#).
2. Select an item.

 **Note**

When you choose an item from this section, you might incur additional usage charges. For more information, see [Usage Metering](#).

The following items are available:

- **Notebook**

Launches the notebook in a kernel session on the chosen SageMaker image.

Creates the notebook in the folder that you have currently selected in the file browser. To view the file browser, in the left sidebar of Studio Classic, choose the **File Browser** icon.

- **Console**

Launches the shell in a kernel session on the chosen SageMaker image.

Opens the shell in the folder that you have currently selected in the file browser.

- **Image terminal**

Launches the terminal in a terminal session on the chosen SageMaker image.

Opens the terminal in the root folder for the user (as shown by the **Home** folder in the file browser).

**Note**

By default, CPU instances launch on a `m1.t3.medium` instance, while GPU instances launch on a `m1.g4dn.xlarge` instance.

## Utilities and files

In this section, you can add contextual help in a notebook; create Python, Markdown and text files; and open a system terminal.

**Note**

Items in this section run in the context of Amazon SageMaker Studio Classic and don't incur usage charges.

The following items are available:

- **Show Contextual Help**

Opens a new tab that displays contextual help for functions in a Studio Classic notebook. To display the help, choose a function in an active notebook. To make it easier to see the help in context, drag the help tab so that it's adjacent to the notebook tab. To open the help tab from within a notebook, press `Ctrl + I`.

The following screenshot shows the contextual help for the `Experiment.create` method.

The screenshot shows a Jupyter notebook interface within the Amazon SageMaker Studio environment. The top bar indicates the session is named 'mnist-handwritten-digits-clas X', running on a 'Python 3 (Data Science)' kernel with '2 vCPU + 4 GiB' resources. The main code cell contains the following Python code:

```
[ ]: mnist_experiment = Experiment.create(
    experiment_name=f"mnist-hand-written-digits-classification-{int(time.time())}",
    description="Classification of mnist hand-written digits",
    sagemaker_boto_client=sm)
print(mnist_experiment)
```

A contextual help panel is open below the code cell, titled 'Create an Experiment'. It displays the following information:

- Signature:**

```
Experiment.create(  
    experiment_name=None,  
    description=None,  
    sagemaker_boto_client=None,  
)
```
- Docstring:**

Create a new experiment in SageMaker and return an ``Experiment`` object.
- Args:**
  - experiment\_name: (str): Name of the experiment. Must be unique. Required.
  - experiment\_description: (str, optional): Description of the experiment
  - sagemaker\_boto\_client (SageMaker.Client, optional): Boto3 client for SageMaker. If not supplied, a default boto3 client will be created and used.
- Returns:**

```
sagemaker.experiments.experiment.Experiment: A SageMaker ``Experiment`` object
```
- File:** /opt/conda/lib/python3.7/site-packages/smexperiments/experiment.py
- Type:** method

- **System terminal**

Opens a bash shell in the root folder for the user (as shown by the **Home** folder in the file browser).

- **Text File and Markdown File**

Creates a file of the associated type in the folder that you have currently selected in the file browser. To view the file browser, in the left sidebar, choose the **File Browser** icon



).

# Use Amazon SageMaker Studio Classic Notebooks

## Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker Studio Classic notebooks are collaborative notebooks that you can launch quickly because you don't need to set up compute instances and file storage beforehand. Studio Classic notebooks provide persistent storage, which enables you to view and share notebooks even if the instances that the notebooks run on are shut down.

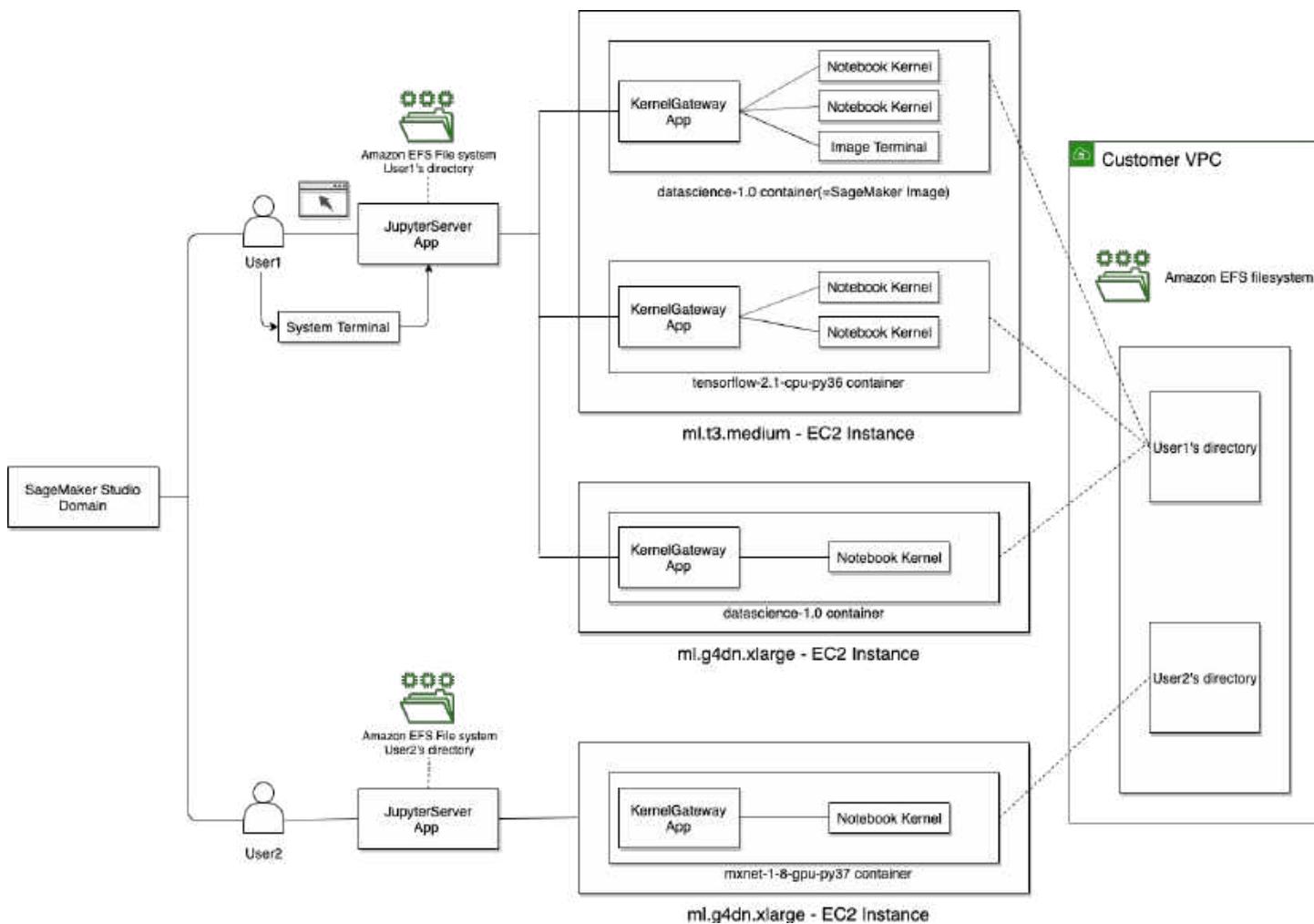
You can share your notebooks with others, so that they can easily reproduce your results and collaborate while building models and exploring your data. You provide access to a read-only copy of the notebook through a secure URL. Dependencies for your notebook are included in the notebook's metadata. When your colleagues copy the notebook, it opens in the same environment as the original notebook.

A Studio Classic notebook runs in an environment defined by the following:

- Amazon EC2 instance type – The hardware configuration the notebook runs on. The configuration includes the number and type of processors (vCPU and GPU), and the amount and type of memory. The instance type determines the pricing rate.
- SageMaker image – A container image that is compatible with SageMaker Studio Classic. The image consists of the kernels, language packages, and other files required to run a notebook in Studio Classic. There can be multiple images in an instance. For more information, see [Bring your own SageMaker image](#).
- KernelGateway app – A SageMaker image runs as a KernelGateway app. The app provides access to the kernels in the image. There is a one-to-one correspondence between a SageMaker AI image and a KernelGateway app.
- Kernel – The process that inspects and runs the code contained in the notebook. A kernel is defined by a *kernel spec* in the image. There can be multiple kernels in an image.

You can change any of these resources from within the notebook.

The following diagram outlines how a notebook kernel runs in relation to the KernelGateway App, User, and domain.



Sample SageMaker Studio Classic notebooks are available in the [aws\\_sagemaker\\_studio](#) folder of the [Amazon SageMaker example GitHub repository](#). Each notebook comes with the necessary SageMaker image that opens the notebook with the appropriate kernel.

We recommend that you familiarize yourself with the SageMaker Studio Classic interface and the Studio Classic notebook toolbar before creating or using a Studio Classic notebook. For more information, see [Amazon SageMaker Studio Classic UI Overview](#) and [Use the Studio Classic Notebook Toolbar](#).

## Topics

- [How Are Amazon SageMaker Studio Classic Notebooks Different from Notebook Instances?](#)
- [Get Started](#)
- [Amazon SageMaker Studio Classic Tour](#)

- [Create or Open an Amazon SageMaker Studio Classic Notebook](#)
- [Use the Studio Classic Notebook Toolbar](#)
- [Install External Libraries and Kernels in Amazon SageMaker Studio Classic](#)
- [Share and Use an Amazon SageMaker Studio Classic Notebook](#)
- [Get Studio Classic Notebook and App Metadata](#)
- [Get Notebook Differences](#)
- [Manage Resources](#)
- [Usage Metering](#)
- [Available Resources](#)

## How Are Amazon SageMaker Studio Classic Notebooks Different from Notebook Instances?

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

When you're starting a new notebook, we recommend that you create the notebook in Amazon SageMaker Studio Classic instead of launching a notebook instance from the Amazon SageMaker AI console. There are many benefits to using a Studio Classic notebook, including the following:

- **Faster:** Starting a Studio Classic notebook is faster than launching an instance-based notebook. Typically, it is 5-10 times faster than instance-based notebooks.
- **Easy notebook sharing:** Notebook sharing is an integrated feature in Studio Classic. Users can generate a shareable link that reproduces the notebook code and also the SageMaker image required to execute it, in just a few clicks.
- **Latest Python SDK:** Studio Classic notebooks come pre-installed with the latest [Amazon SageMaker Python SDK](#).
- **Access all Studio Classic features:** Studio Classic notebooks are accessed from within Studio Classic. This enables you to build, train, debug, track, and monitor your models without leaving Studio Classic.

- **Persistent user directories:** Each member of a Studio team gets their own home directory to store their notebooks and other files. The directory is automatically mounted onto all instances and kernels as they're started, so their notebooks and other files are always available. The home directories are stored in Amazon Elastic File System (Amazon EFS) so that you can access them from other services.
- **Direct access:** When using IAM Identity Center, you use your IAM Identity Center credentials through a unique URL to directly access Studio Classic. You don't have to interact with the AWS Management Console to run your notebooks.
- **Optimized images:** Studio Classic notebooks are equipped with a set of predefined SageMaker image settings to get you started faster.

### Note

Studio Classic notebooks don't support *local mode*. However, you can use a notebook instance to train a sample of your dataset locally, and then use the same code in a Studio Classic notebook to train on the full dataset.

When you open a notebook in SageMaker Studio Classic, the view is an extension of the JupyterLab interface. The primary features are the same, so you'll find the typical features of a Jupyter notebook and JupyterLab. For more information about the Studio Classic interface, see [Amazon SageMaker Studio Classic UI Overview](#).

## Get Started

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

To get started, you or your organization's administrator need to complete the SageMaker AI domain onboarding process. For more information, see [Amazon SageMaker AI domain overview](#).

You can access a Studio Classic notebook in any of the following ways:

- You receive an email invitation to access Studio Classic through your organization's IAM Identity Center, which includes a direct link to login to Studio Classic without having to use the Amazon SageMaker AI console. You can proceed to the [the section called "Next Steps"](#).
- You receive a link to a shared Studio Classic notebook, which includes a direct link to log in to Studio Classic without having to use the SageMaker AI console. You can proceed to the [the section called "Next Steps"](#).
- You onboard to a domain and then log in to the SageMaker AI console. For more information, see [Amazon SageMaker AI domain overview](#).

## Launch Amazon SageMaker AI

Complete the steps in [Launch Amazon SageMaker Studio Classic](#) to launch Studio Classic.

### Next Steps

Now that you're in Studio Classic, you can try any of the following options:

- To create a Studio Classic notebook or explore Studio Classic end-to-end tutorial notebooks – See [Amazon SageMaker Studio Classic Tour](#) in the next section.
- To familiarize yourself with the Studio Classic interface – See [Amazon SageMaker Studio Classic UI Overview](#) or try the **Getting started notebook** by selecting **Open the Getting started notebook** in the **Quick actions** section of the Studio Classic Home page.

## Amazon SageMaker Studio Classic Tour

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

For a walkthrough that takes you on a tour of the main features of Amazon SageMaker Studio Classic, see the [xgboost\\_customer\\_churn\\_studio.ipynb](#) sample notebook from the [aws/amazon-sagemaker-examples](#) GitHub repository. The code in the notebook trains multiple models and sets up the SageMaker Debugger and SageMaker Model Monitor. The walkthrough shows you

how to view the trials, compare the resulting models, show the debugger results, and deploy the best model using the Studio Classic UI. You don't need to understand the code to follow this walkthrough.

## Prerequisites

To run the notebook for this tour, you need:

- An IAM account to sign in to Studio. For information, see [Amazon SageMaker AI domain overview](#).
- Basic familiarity with the Studio user interface and Jupyter notebooks. For information, see [Amazon SageMaker Studio Classic UI Overview](#).
- A copy of the [aws/amazon-sagemaker-examples](#) repository in your Studio environment.

## To clone the repository

1. Launch Studio Classic following the steps in [Launch Amazon SageMaker Studio Classic](#) For users in IAM Identity Center, sign in using the URL from your invitation email.
2. On the top menu, choose **File**, then **New**, then **Terminal**.
3. At the command prompt, run the following command to clone the [aws/amazon-sagemaker-examples](#) GitHub repository.

```
$ git clone https://github.com/aws/amazon-sagemaker-examples.git
```

## To navigate to the sample notebook

1. From the **File Browser** on the left menu, select **amazon-sagemaker-examples**.
2. Navigate to the example notebook with the following path.

`~/amazon-sagemaker-examples/aws_sagemaker_studio/getting_started/xgboost_customer_churn_studio.ipynb`

3. Follow the notebook to learn about Studio Classic's main features.

**Note**

If you encounter an error when you run the sample notebook, and some time has passed from when you cloned the repository, review the notebook on the remote repository for updates.

## Create or Open an Amazon SageMaker Studio Classic Notebook

**Important**

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

**Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

When you [Create a Notebook from the File Menu](#) in Amazon SageMaker Studio Classic or [Open a notebook in Studio Classic](#) for the first time, you are prompted to set up your environment by choosing a SageMaker image, a kernel, an instance type, and, optionally, a lifecycle configuration script that runs on image start-up. SageMaker AI launches the notebook on an instance of the chosen type. By default, the instance type is set to `m1.t3.medium` (available as part of

the [AWS Free Tier](#)) for CPU-based images. For GPU-based images, the default instance type is `ml.g4dn.xlarge`.

If you create or open additional notebooks that use the same instance type, whether or not the notebooks use the same kernel, the notebooks run on the same instance of that instance type.

After you launch a notebook, you can change its instance type, SageMaker image, and kernel from within the notebook. For more information, see [Change an Instance Type](#) and [Change an Image or a Kernel](#).

### Note

You can have only one instance of each instance type. Each instance can have multiple SageMaker images running on it. Each SageMaker image can run multiple kernels or terminal instances.

Billing occurs per instance and starts when the first instance of a given instance type is launched. If you want to create or open a notebook without the risk of incurring charges, open the notebook from the **File** menu and choose **No Kernel** from the **Select Kernel** dialog box. You can read and edit a notebook without a running kernel but you can't run cells.

Billing ends when the SageMaker image for the instance is shut down. For more information, see [Usage Metering](#).

For information about shutting down the notebook, see [Shut down resources](#).

## Topics

- [Open a notebook in Studio Classic](#)
- [Create a Notebook from the File Menu](#)
- [Create a Notebook from the Launcher](#)
- [List of the available instance types, images, and kernels](#)

## Open a notebook in Studio Classic

Amazon SageMaker Studio Classic can only open notebooks listed in the Studio Classic file browser. For instructions on uploading a notebook to the file browser, see [Upload Files to SageMaker Studio Classic](#) or [Clone a Git Repository in SageMaker Studio Classic](#).

## To open a notebook

1. In the left sidebar, choose the **File Browser** icon (



)

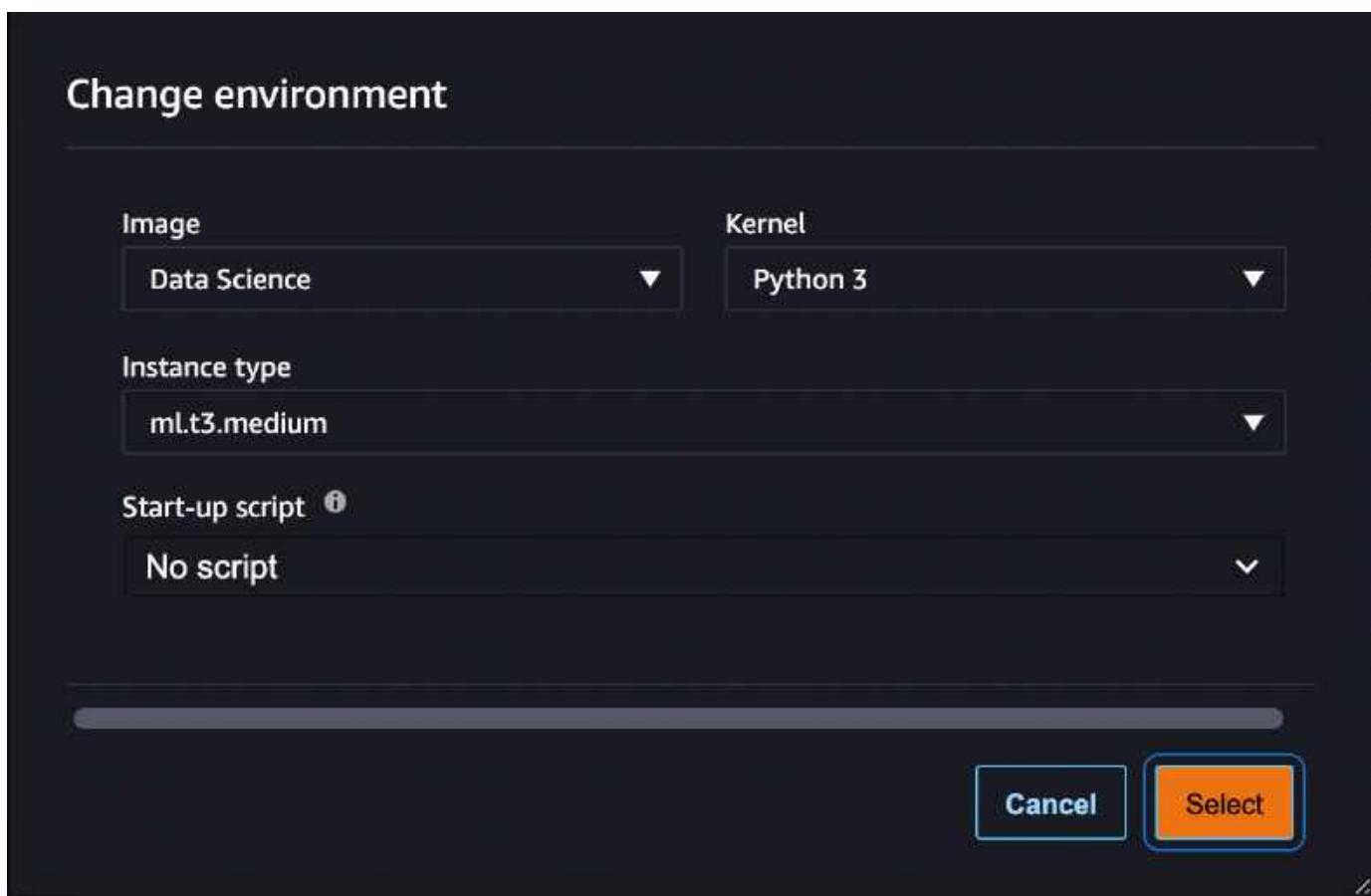
to display the file browser.

2. Browse to a notebook file and double-click it to open the notebook in a new tab.

## Create a Notebook from the File Menu

### To create a notebook from the File menu

1. From the Studio Classic menu, choose **File**, choose **New**, and then choose **Notebook**.
2. In the **Change environment** dialog box, use the dropdown menus to select your **Image**, **Kernel**, **Instance type**, and **Start-up script**, then choose **Select**. Your notebook launches and opens in a new Studio Classic tab.



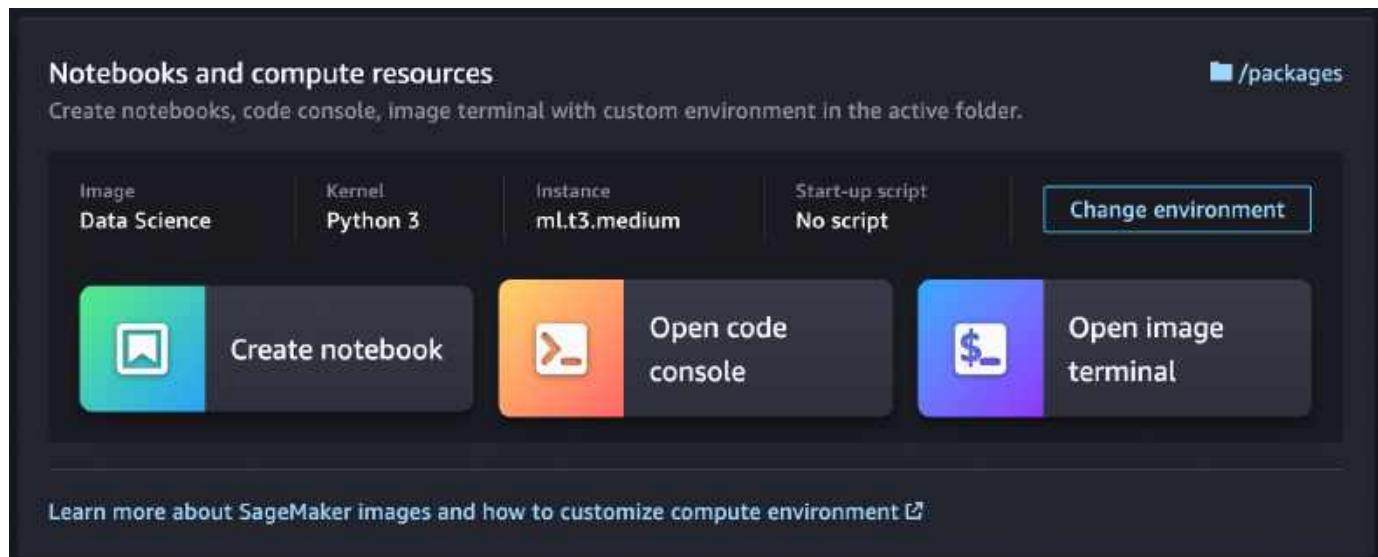
## Create a Notebook from the Launcher

### To create a notebook from the Launcher

1. To open the Launcher, choose **Amazon SageMaker Studio Classic** at the top left of the Studio Classic interface or use the keyboard shortcut **Ctrl + Shift + L**.

To learn about all the available ways to open the Launcher, see [Use the Amazon SageMaker Studio Classic Launcher](#)

2. In the Launcher, in the **Notebooks and compute resources** section, choose **Change environment**.



3. In the **Change environment** dialog box, use the dropdown menus to select your **Image**, **Kernel**, **Instance type**, and **Start-up script**, then choose **Select**.
4. In the Launcher, choose **Create notebook**. Your notebook launches and opens in a new Studio Classic tab.

To view the notebook's kernel session, in the left sidebar, choose the **Running Terminals and Kernels** icon

().

You can stop the notebook's kernel session from this view.

### List of the available instance types, images, and kernels

For a list of all available resources, see:

- [Instance types available for use with Studio Classic](#)
- [Amazon SageMaker images available for use with Studio Classic](#)

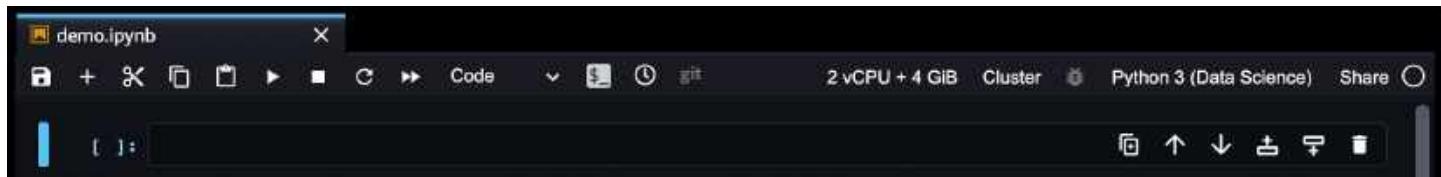
## Use the Studio Classic Notebook Toolbar

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker Studio Classic notebooks extend the JupyterLab interface. For an overview of the original JupyterLab interface, see [The JupyterLab Interface](#).

The following image shows the toolbar and an empty cell from a Studio Classic notebook.

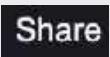


When you pause on a toolbar icon, a tooltip displays the icon function. Additional notebook commands are found in the Studio Classic main menu. The toolbar includes the following icons:

Icon	Description
	<b>Save and checkpoint</b> Saves the notebook and updates the checkpoint file. For more information, see <a href="#">Get the Difference Between the Last Checkpoint</a> .
	<b>Insert cell</b> Inserts a code cell below the current cell. The current cell is noted by the blue vertical marker in the left margin.
	<b>Cut, copy, and paste cells</b>

Icon	Description
	Cuts, copies, and pastes the selected cells.
	<b>Run cells</b> Runs the selected cells and then makes the cell that follows the last selected cell the new selected cell.
	<b>Interrupt kernel</b> Interrupts the kernel, which cancels the currently running operation. The kernel remains active.
	<b>Restart kernel</b> Restarts the kernel. Variables are reset. Unsaved information is not affected.
	<b>Restart kernel and run all cells</b> Restarts the kernel, then run all the cells of the notebook.
	<b>Cell type</b> Displays or changes the current cell type. The cell types are: <ul style="list-style-type: none"><li>• Code – Code that the kernel runs.</li><li>• Markdown – Text rendered as markdown.</li><li>• Raw – Content, including Markdown markup, that's displayed as text.</li></ul>
	<b>Launch terminal</b> Launches a terminal in the SageMaker image hosting the notebook. For an example, see <a href="#">Get App Metadata</a> .
	<b>Checkpoint diff</b> Opens a new tab that displays the difference between the notebook and the checkpoint file. For more information, see <a href="#">Get the Difference Between the Last Checkpoint</a> .

Icon	Description
	<p><b>Git diff</b></p> <p>Only enabled if the notebook is opened from a Git repository. Opens a new tab that displays the difference between the notebook and the last Git commit. For more information, see <a href="#">Get the Difference Between the Last Commit</a>.</p>
<b>2 vCPU + 4 GiB</b>	<p><b>Instance type</b></p> <p>Displays or changes the instance type the notebook runs in. The format is as follows:</p> <p>number of vCPUs + amount of memory + number of GPUs</p> <p>Unknown indicates the notebook was opened without specifying a kernel. The notebook runs on the SageMaker Studio instance and doesn't accrue runtime charges. You can't assign the notebook to an instance type. You must specify a kernel and then Studio assigns the notebook to a default type.</p> <p>For more information, see <a href="#">Create or Open an Amazon SageMaker Studio Classic Notebook</a> and <a href="#">Change an Instance Type</a>.</p>
<b>Cluster</b>	<p><b>Cluster</b></p> <p>Connect your notebook to an Amazon EMR cluster and scale your ETL jobs or run large-scale model training using Apache Spark, Hive, or Presto.</p> <p>For more information, see <a href="#">Data preparation using Amazon EMR</a>.</p>

Icon	Description
<b>Python 3 (Data Science)</b>	<p><b>Kernel and SageMaker Image</b></p> <p>Displays or changes the kernel that processes the cells in the notebook. The format is as follows:</p> <p>Kernel (SageMaker Image)</p> <p>No Kernel indicates the notebook was opened without specifying a kernel. You can edit the notebook but you can't run any cells.</p> <p>For more information, see <a href="#">Change an Image or a Kernel</a>.</p>
	<p><b>Kernel busy status</b></p> <p>Displays the busy status of the kernel. When the edge of the circle and its interior are the same color, the kernel is busy. The kernel is busy when it is starting and when it is processing cells. Additional kernel states are displayed in the status bar at the bottom-left corner of SageMaker Studio.</p>
	<p><b>Share notebook</b></p> <p>Shares the notebook. For more information, see <a href="#">Share and Use an Amazon SageMaker Studio Classic Notebook</a>.</p>

To select multiple cells, click in the left margin outside of a cell. Hold down the Shift key and use K or the Up key to select previous cells, or use J or the Down key to select following cells.

## Install External Libraries and Kernels in Amazon SageMaker Studio Classic

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker Studio Classic notebooks come with multiple images already installed. These images contain kernels and Python packages including scikit-learn, Pandas, NumPy, TensorFlow, PyTorch, and MXNet. You can also install your own images that contain your choice of packages and kernels. For more information on installing your own image, see [Bring your own SageMaker image](#).

The different Jupyter kernels in Amazon SageMaker Studio Classic notebooks are separate conda environments. For information about conda environments, see [Managing environments](#).

## Package installation tools

### Important

Currently, all packages in Amazon SageMaker notebooks are licensed for use with Amazon SageMaker AI and do not require additional commercial licenses. However, this might be subject to change in the future, and we recommend reviewing the licensing terms regularly for any updates.

The method that you use to install Python packages from the terminal differs depending on the image. Studio Classic supports the following package installation tools:

- **Notebooks** – The following commands are supported. If one of the following does not work on your image, try the other one.
  - `%conda install`
  - `%pip install`
- **The Jupyter terminal** – You can install packages using pip and conda directly. You can also use `apt-get install` to install system packages from the terminal.

### Note

We do not recommend using `pip install -u` or `pip install --user`, because those commands install packages on the user's Amazon EFS volume and can potentially block JupyterServer app restarts. Instead, use a lifecycle configuration to reinstall the required packages on app restarts as shown in [Install packages using lifecycle configurations](#).

We recommend using `%pip` and `%conda` to install packages from within a notebook because they correctly take into account the active environment or interpreter being used. For more information, see [Add %pip and %conda magic functions](#). You can also use the system command syntax (lines starting with !) to install packages. For example, `!pip install` and `!conda install`.

## Conda

Conda is an open source package management system and environment management system that can install packages and their dependencies. SageMaker AI supports using conda with the conda-forge channel. For more information, see [Conda channels](#). The conda-forge channel is a community channel where contributors can upload packages.

### Note

Installing packages from conda-forge can take up to 10 minutes. Timing relates to how conda resolves the dependency graph.

All of the SageMaker AI provided environments are functional. User installed packages may not function correctly.

Conda has two methods for activating environments: `conda activate`, and `source activate`. For more information, see [Managing environment](#).

## Supported conda operations

- `conda install` of a package in a single environment
- `conda install` of a package in all environments
- Installing a package from the main conda repository
- Installing a package from conda-forge
- Changing the conda install location to use Amazon EBS
- Supporting both `conda activate` and `source activate`

## Pip

Pip is the tool for installing and managing Python packages. Pip searches for packages on the Python Package Index (PyPI) by default. Unlike conda, pip doesn't have built in environment

support. Therfore, pip isn't as thorough as conda when it comes to packages with native or system library dependencies. Pip can be used to install packages in conda environments. You can use alternative package repositories with pip instead of the PyPI.

## Supported pip operations

- Using pip to install a package without an active conda environment
- Using pip to install a package in a conda environment
- Using pip to install a package in all conda environments
- Changing the pip install location to use Amazon EBS
- Using an alternative repository to install packages with pip

## Unsupported

SageMaker AI aims to support as many package installation operations as possible. However, if the packages were installed by SageMaker AI and you use the following operations on these packages, it might make your environment unstable:

- Uninstalling
- Downgrading
- Upgrading

Due to potential issues with network conditions or configurations, or the availability of conda or PyPi, packages may not install in a fixed or deterministic amount of time.

### Note

Attempting to install a package in an environment with incompatible dependencies can result in a failure. If issues occur, you can contact the library maintainer about updating the package dependencies. When you modify the environment, such as removing or updating existing packages, this may result in instability of that environment.

## Install packages using lifecycle configurations

Install custom images and kernels on the Studio Classic instance's Amazon EBS volume so that they persist when you stop and restart the notebook, and that any external libraries you install

are not updated by SageMaker AI. To do that, use a lifecycle configuration that includes both a script that runs when you create the notebook (`on-create`) and a script that runs each time you restart the notebook (`on-start`). For more information about using lifecycle configurations with Studio Classic, see [Use lifecycle configurations to customize Studio Classic](#). For sample lifecycle configuration scripts, see [SageMaker AI Studio Classic Lifecycle Configuration Samples](#).

## Share and Use an Amazon SageMaker Studio Classic Notebook

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You can share your Amazon SageMaker Studio Classic notebooks with your colleagues. The shared notebook is a copy. After you share your notebook, any changes you make to your original notebook aren't reflected in the shared notebook and any changes your colleague's make in their shared copies of the notebook aren't reflected in your original notebook. If you want to share your latest version, you must create a new snapshot and then share it.

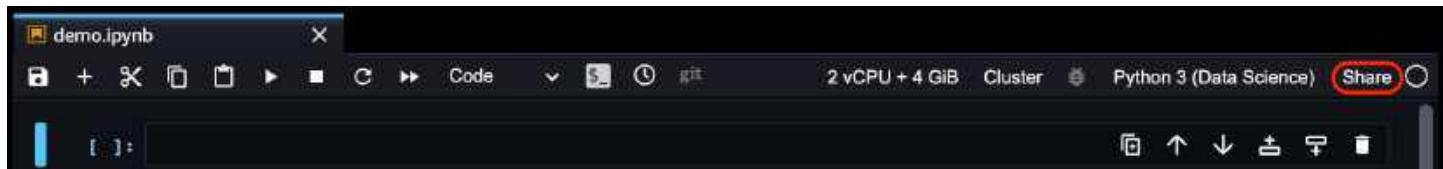
### Topics

- [Share a Notebook](#)

- [Use a Shared Notebook](#)
- [Shared spaces and realtime collaboration](#)

## Share a Notebook

The following screenshot shows the menu from a Studio Classic notebook.



### To share a notebook

1. In the upper-right corner of the notebook, choose **Share**.
2. (Optional) In **Create shareable snapshot**, choose any of the following items:
  - **Include Git repo information** – Includes a link to the Git repository that contains the notebook. This enables you and your colleague to collaborate and contribute to the same Git repository.
  - **Include output** – Includes all notebook output that has been saved.

**Note**

If you're an user in IAM Identity Center and you don't see these options, your IAM Identity Center administrator probably disabled the feature. Contact your administrator.

3. Choose **Create**.
4. After the snapshot is created, choose **Copy link** and then choose **Close**.
5. Share the link with your colleague.

After selecting your sharing options, you are provided with a URL. You can share this link with users that have access to Amazon SageMaker Studio Classic. When the user opens the URL, they're prompted to log in using IAM Identity Center or IAM authentication. This shared notebook becomes a copy, so changes made by the recipient will not be reproduced in your original notebook.

## Use a Shared Notebook

You use a shared notebook in the same way you would with a notebook that you created yourself. You must first login to your account, then open the shared link. If you don't have an active session, you receive an error.

When you choose a link to a shared notebook for the first time, a read-only version of the notebook opens. To edit the shared notebook, choose **Create a Copy**. This copies the shared notebook to your personal storage.

The copied notebook launches on an instance of the instance type and SageMaker image that the notebook was using when the sender shared it. If you aren't currently running an instance of the instance type, a new instance is started. Customization to the SageMaker image isn't shared. You can also inspect the notebook snapshot by choosing **Snapshot Details**.

The following are some important considerations about sharing and authentication:

- If you have an active session, you see a read-only view of the notebook until you choose **Create a Copy**.
- If you don't have an active session, you need to log in.
- If you use IAM to login, after you login, select your user profile then choose **Open Studio Classic**. Then you need to choose the link you were sent.
- If you use IAM Identity Center to login, after you login the shared notebook is opened automatically in Studio.

## Shared spaces and realtime collaboration

A shared space consists of a shared JupyterServer application and a shared directory. A key benefit of a shared space is that it facilitates collaboration between members of the shared space in real time. Users collaborating in a workspace get access to a shared Studio Classic application where they can access, read, and edit their notebooks in real time. Real time collaboration is only supported for JupyterServer applications within a shared space. Users with access to a shared space can simultaneously open, view, edit, and execute Jupyter notebooks in the shared Studio Classic application in that space. For more information about shared spaced and real time collaboration, see [Collaboration with shared spaces](#).

## Get Studio Classic Notebook and App Metadata

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You can access notebook metadata and App metadata using the Amazon SageMaker Studio Classic UI.

### Topics

- [Get Studio Classic Notebook Metadata](#)
- [Get App Metadata](#)

### Get Studio Classic Notebook Metadata

Jupyter notebooks contain optional metadata that you can access through the Amazon SageMaker Studio Classic UI.

#### To view the notebook metadata:

1. In the right sidebar, choose the **Property Inspector** icon  ).
2. Open the **Advanced Tools** section.

The metadata should look similar to the following.

```
{  
    "instance_type": "ml.t3.medium",  
    "kernelspec": {  
        "display_name": "Python 3 (Data Science)",  
        "language": "python",  
        "name": "python3__SAGEMAKER_INTERNAL__arn:aws:sagemaker:us-west-2:<acct-id>:image/datascience-1.0"  
    },
```

```
"language_info": {  
    "codemirror_mode": {  
        "name": "ipython",  
        "version": 3  
    },  
    "file_extension": ".py",  
    "mimetype": "text/x-python",  
    "name": "python",  
    "nbconvert_exporter": "python",  
    "pygments_lexer": "ipython3",  
    "version": "3.7.10"  
}  
}
```

## Get App Metadata

When you create a notebook in Amazon SageMaker Studio Classic, the App metadata is written to a file named `resource-metadata.json` in the folder `/opt/ml/metadata/`. You can get the App metadata by opening an Image terminal from within the notebook. The metadata gives you the following information, which includes the SageMaker image and instance type the notebook runs in:

- **AppType** – KernelGateway
- **DomainId** – Same as the Studio ClassicID
- **UserProfileName** – The profile name of the current user
- **ResourceArn** – The Amazon Resource Name (ARN) of the App, which includes the instance type
- **ResourceName** – The name of the SageMaker image

Additional metadata might be included for internal use by Studio Classic and is subject to change.

### To get the App metadata

1. In the center of the notebook menu, choose the **Launch Terminal** icon



).

This opens a terminal in the SageMaker image that the notebook runs in.

2. Run the following commands to display the contents of the `resource-metadata.json` file.

```
$ cd /opt/ml/metadata/
```

```
cat resource-metadata.json
```

The file should look similar to the following.

```
{  
    "AppType": "KernelGateway",  
    "DomainId": "d-xxxxxxxxxxxxxx",  
    "UserProfileName": "profile-name",  
    "ResourceArn": "arn:aws:sagemaker:us-east-2:account-id:app/d-xxxxxxxxxxxxx/  
profile-name/KernelGateway/datascience--1-0-ml-t3-medium",  
    "ResourceName": "datascience--1-0-ml",  
    "AppImageVersion": ""  
}
```

## Get Notebook Differences

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

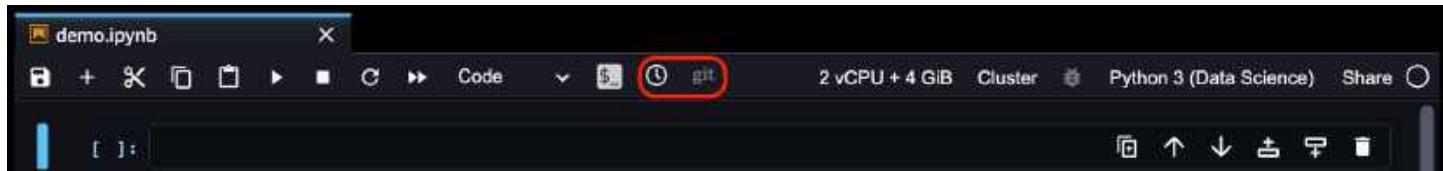
[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You can display the difference between the current notebook and the last checkpoint or the last Git commit using the Amazon SageMaker AI UI.

The following screenshot shows the menu from a Studio Classic notebook.



## Topics

- [Get the Difference Between the Last Checkpoint](#)
- [Get the Difference Between the Last Commit](#)

### Get the Difference Between the Last Checkpoint

When you create a notebook, a hidden checkpoint file that matches the notebook is created. You can view changes between the notebook and the checkpoint file or revert the notebook to match the checkpoint file.

By default, a notebook is auto-saved every 120 seconds and also when you close the notebook. However, the checkpoint file isn't updated to match the notebook. To save the notebook and update the checkpoint file to match, you must choose the **Save notebook and create checkpoint** icon (



)

on the left of the notebook menu or use the **Ctrl + S** keyboard shortcut.

To view the changes between the notebook and the checkpoint file, choose the **Checkpoint diff** icon



)

in the center of the notebook menu.

To revert the notebook to the checkpoint file, from the main Studio Classic menu, choose **File** then **Revert Notebook to Checkpoint**.

### Get the Difference Between the Last Commit

If a notebook is opened from a Git repository, you can view the difference between the notebook and the last Git commit.

To view the changes in the notebook from the last Git commit, choose the **Git diff** icon



)

in the center of the notebook menu.

## Manage Resources

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You can change the instance type, and SageMaker image and kernel from within an Amazon SageMaker Studio Classic notebook. To create a custom kernel to use with your notebooks, see [Bring your own SageMaker image](#).

### Topics

- [Change an Instance Type](#)
- [Change an Image or a Kernel](#)
- [Shut down resources from Amazon SageMaker Studio Classic](#)

### Change an Instance Type

When you open a new Studio Classic notebook for the first time, you are assigned a default Amazon Elastic Compute Cloud (Amazon EC2) instance type to run the notebook. When you open additional notebooks on the same instance type, the notebooks run on the same instance as the first notebook, even if the notebooks use different kernels.

You can change the instance type that your Studio Classic notebook runs on from within the notebook.

The following information only applies to Studio Classic notebooks. For information about how to change the instance type of a Amazon SageMaker notebook instance, see [Update a Notebook Instance](#).

## **⚠️ Important**

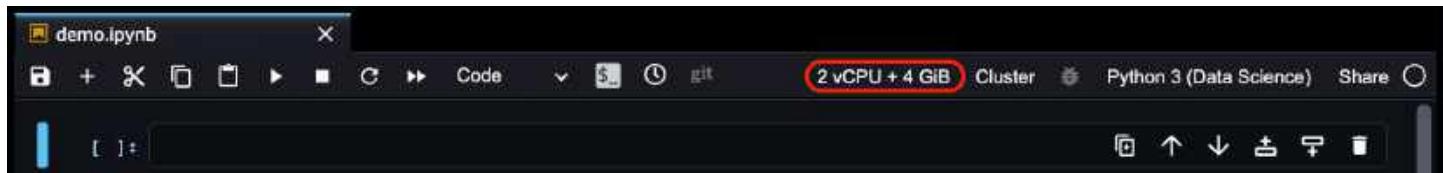
If you change the instance type, unsaved information and existing settings for the notebook are lost, and installed packages must be re-installed.

The previous instance type continues to run even if no kernel sessions or apps are active.

You must explicitly stop the instance to stop accruing charges. To stop the instance, see

[Shut down resources](#).

The following screenshot shows the menu from a Studio Classic notebook. The processor and memory of the instance type powering the notebook are displayed as **2 vCPU + 4 GiB**.



## To change the instance type

1. Choose the processor and memory of the instance type powering the notebook. This opens a pop up window.
2. From the **Set up notebook environment** pop up window, select the **Instance type** dropdown menu.
3. From the **Instance type** dropdown, choose one of the instance types that are listed.
4. After choosing a type, choose **Select**.
5. Wait for the new instance to become enabled, and then the new instance type information is displayed.

For a list of the available instance types, see [Instance types available for use with Studio Classic](#).

## Change an Image or a Kernel

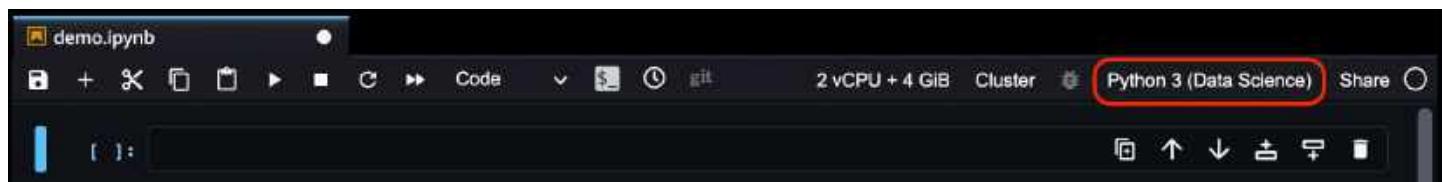
## **⚠️ Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the

Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

With Amazon SageMaker Studio Classic notebooks, you can change the notebook's image or kernel from within the notebook.

The following screenshot shows the menu from a Studio Classic notebook. The current SageMaker AI kernel and image are displayed as **Python 3 (Data Science)**, where Python 3 denotes the kernel and Data Science denotes the SageMaker AI image that contains the kernel. The color of the circle to the right indicates the kernel is idle or busy. The kernel is busy when the center and the edge of the circle are the same color.



### To change a notebook's image or kernel

1. Choose the image/kernel name in the notebook menu.
2. From the **Set up notebook environment** pop up window, select the **Image or Kernel** dropdown menu.
3. From the dropdown menu, choose one of the images or kernels that are listed.
4. After choosing an image or kernel, choose **Select**.
5. Wait for the kernel's status to show as idle, which indicates the kernel has started.

For a list of available SageMaker images and kernels, see [Amazon SageMaker images available for use with Studio Classic](#).

### Shut down resources from Amazon SageMaker Studio Classic

#### ⚠ Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You can shut down individual Amazon SageMaker AI resources, including notebooks, terminals, kernels, apps, and instances from Studio Classic. You can also shut down all of the resources in one of these categories at the same time. Amazon SageMaker Studio Classic does not support shutting down resources from within a notebook.

### Note

When you shut down a Studio Classic notebook instance, additional resources that you created in Studio Classic are not deleted. For example, additional resources can include SageMaker AI endpoints, Amazon EMR clusters, and Amazon S3 buckets. To stop the accrual of charges, you must manually delete these resources. For information about finding resources that are accruing charges, see [Analyzing your costs with AWS Cost Explorer](#).

The following topics demonstrate how to delete these SageMaker AI resources.

### Topics

- [Shut down an open notebook](#)
- [Shut down resources](#)

### Shut down an open notebook

When you shut down a Studio Classic notebook, the notebook is not deleted. The kernel that the notebook is running on is shut down and any unsaved information in the notebook is lost. You can shut down an open notebook from the Studio Classic **File** menu or from the Running Terminal and Kernels pane. The following procedure shows how to shut down an open notebook from the Studio Classic **File** menu.

#### To shut down an open notebook from the File menu

1. Launch Studio Classic by following the steps in [Launch Amazon SageMaker Studio Classic](#).
2. (Optional) Save the notebook contents by choosing **File**, then **Save Notebook**.
3. Choose **File**.
4. Choose **Close and Shutdown Notebook**. This opens a pop-up window.
5. From the pop-up window, choose **OK**.

## Shut down resources

You can reach the **Running Terminals and Kernels** pane of Amazon SageMaker Studio Classic by selecting the **Running Terminals and Kernels** icon



).

The **Running Terminals and Kernels** pane consists of four sections. Each section lists all the resources of that type. You can shut down each resource individually or shut down all the resources in a section at the same time.

When you choose to shut down all resources in a section, the following occurs:

- **RUNNING INSTANCES/RUNNING APPS** – All instances, apps, notebooks, kernel sessions, consoles/shells, and image terminals are shut down. System terminals aren't shut down.
- **KERNEL SESSIONS** – All kernels, notebooks and consoles/shells are shut down.
- **TERMINAL SESSIONS** – All image terminals and system terminals are shut down.

### To shut down resources

1. Launch Studio Classic by following the steps in [Launch Amazon SageMaker Studio Classic](#).
2. Choose the **Running Terminals and Kernels** icon.
3. Do either of the following:
  - To shut down a specific resource, choose the **Shut Down** icon on the same row as the resource.

For running instances, a confirmation dialog box lists all of the resources that SageMaker AI will shut down. A confirmation dialog box displays all running apps. To proceed, choose **Shut down all**.

#### Note

A confirmation dialog box isn't displayed for kernel sessions or terminal sessions.

- To shut down all resources in a section, choose the X to the right of the section label. A confirmation dialog box is displayed. Choose **Shut down all** to proceed.

**Note**

When you shut down these Studio Classic resources, any additional resources created from Studio Classic, such as SageMaker AI endpoints, Amazon EMR clusters, and Amazon S3 buckets are not deleted. You must manually delete these resources to stop the accrual of charges. For information about finding resources that are accruing charges, see [Analyzing your costs with AWS Cost Explorer](#).

## Usage Metering

**Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

There is no additional charge for using Amazon SageMaker Studio Classic. The costs incurred for running Amazon SageMaker Studio Classic notebooks, interactive shells, consoles, and terminals are based on Amazon Elastic Compute Cloud (Amazon EC2) instance usage.

When you run the following resources, you must choose a SageMaker image and kernel:

### From the Studio Classic Launcher

- Notebook
- Interactive Shell
- Image Terminal

### From the File menu

- Notebook
- Console

When launched, the resource is run on an Amazon EC2 instance of the chosen instance type. If an instance of that type was previously launched and is available, the resource is run on that instance.

For CPU based images, the default suggested instance type is `m1.t3.medium`. For GPU based images, the default suggested instance type is `m1.g4dn.xlarge`.

The costs incurred are based on the instance type. You are billed separately for each instance.

Metering starts when an instance is created. Metering ends when all the apps on the instance are shut down, or the instance is shut down. For information about how to shut down an instance, see [Shut down resources from Amazon SageMaker Studio Classic](#).

### **Important**

You must shut down the instance to stop incurring charges. If you shut down the notebook running on the instance but don't shut down the instance, you will still incur charges. When you shut down the Studio Classic notebook instances, any additional resources, such as SageMaker AI endpoints, Amazon EMR clusters, and Amazon S3 buckets created from Studio Classic are not deleted. Delete those resources to stop accrual of charges.

When you open multiple notebooks on the same instance type, the notebooks run on the same instance even if they are using different kernels. You are billed only for the time that one instance is running.

You can change the instance type from within the notebook after you open it. For more information, see [Change an Instance Type](#).

For information about billing along with pricing examples, see [Amazon SageMaker Pricing](#).

## Available Resources

### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following sections list the available resources for Amazon SageMaker Studio Classic notebooks.

## Topics

- [Instance types available for use with Studio Classic](#)
- [Amazon SageMaker images available for use with Studio Classic](#)

### Instance types available for use with Studio Classic

#### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker Studio Classic notebooks run on Amazon Elastic Compute Cloud (Amazon EC2) instances. The following Amazon EC2 instance types are available for use with Studio Classic notebooks. For detailed information on which instance types fit your use case, and their performance capabilities, see [Amazon Elastic Compute Cloud Instance types](#). For information about pricing for these instance types, see [Amazon EC2 Pricing](#).

For information about available Amazon SageMaker Notebook Instance types, see [CreateNotebookInstance](#).

#### **Note**

For most use cases, you should use a `m1.t3.medium`. This is the default instance type for CPU-based SageMaker images, and is available as part of the [AWS Free Tier](#).

## Topics

- [CPU instances](#)
- [Instances with 1 or more GPUs](#)

## CPU instances

The following table lists the Amazon EC2 CPU instance types with no GPU attached that are available for use with Studio Classic notebooks. It also lists information about the specifications of each instance type. The default instance type for CPU-based images is `ml.t3.medium`.

For detailed information on which instance types fit your use case, and their performance capabilities, see [Amazon Elastic Compute Cloud Instance types](#). For information about pricing for these instance types, see [Amazon EC2 Pricing](#).

### CPU instances

Instance	Use case	Fast launch	vCPU	Memory (GiB)	Instance Storage (GB)
<code>ml.t3.medium</code>	General purpose	Yes	2	4	Amazon EBS Only
<code>ml.t3.large</code>	General purpose	No	2	8	Amazon EBS Only
<code>ml.t3.xlarge</code>	General purpose	No	4	16	Amazon EBS Only
<code>ml.t3.2xlarge</code>	General purpose	No	8	32	Amazon EBS Only
<code>ml.m5.large</code>	General purpose	Yes	2	8	Amazon EBS Only
<code>ml.m5.xlarge</code>	General purpose	No	4	16	Amazon EBS Only

Instance	Use case	Fast launch	vCPU	Memory (GiB)	Instance Storage (GB)
ml.m5.2xlarge	General purpose	No	8	32	Amazon EBS Only
ml.m5.4xlarge	General purpose	No	16	64	Amazon EBS Only
ml.m5.8xlarge	General purpose	No	32	128	Amazon EBS Only
ml.m5.12xlarge	General purpose	No	48	192	Amazon EBS Only
ml.m5.16xlarge	General purpose	No	64	256	Amazon EBS Only
ml.m5.24xlarge	General purpose	No	96	384	Amazon EBS Only
ml.m5d.large	General purpose	No	2	8	1 x 75 NVMe SSD
ml.m5d.xlarge	General purpose	No	4	16	1 x 150 NVMe SSD
ml.m5d.2xlarge	General purpose	No	8	32	1 x 300 NVMe SSD

Instance	Use case	Fast launch	vCPU	Memory (GiB)	Instance Storage (GB)
ml.m5d.4xlarge	General purpose	No	16	64	2 x 300 NVMe SSD
ml.m5d.8xlarge	General purpose	No	32	128	2 x 600 NVMe SSD
ml.m5d.12xlarge	General purpose	No	48	192	2 x 900 NVMe SSD
ml.m5d.16xlarge	General purpose	No	64	256	4 x 600 NVMe SSD
ml.m5d.24xlarge	General purpose	No	96	384	4 x 900 NVMe SSD
ml.c5.large	Compute optimized	Yes	2	4	Amazon EBS Only
ml.c5.xlarge	Compute optimized	No	4	8	Amazon EBS Only
ml.c5.2xlarge	Compute optimized	No	8	16	Amazon EBS Only
ml.c5.4xlarge	Compute optimized	No	16	32	Amazon EBS Only

Instance	Use case	Fast launch	vCPU	Memory (GiB)	Instance Storage (GB)
ml.c5.9xlarge	Compute optimized	No	36	72	Amazon EBS Only
ml.c5.12xlarge	Compute optimized	No	48	96	Amazon EBS Only
ml.c5.18xlarge	Compute optimized	No	72	144	Amazon EBS Only
ml.c5.24xlarge	Compute optimized	No	96	192	Amazon EBS Only
ml.r5.large	Memory optimized	No	2	16	Amazon EBS Only
ml.r5.xlarge	Memory optimized	No	4	32	Amazon EBS Only
ml.r5.2xlarge	Memory optimized	No	8	64	Amazon EBS Only
ml.r5.4xlarge	Memory optimized	No	16	128	Amazon EBS Only
ml.r5.8xlarge	Memory optimized	No	32	256	Amazon EBS Only

Instance	Use case	Fast launch	vCPU	Memory (GiB)	Instance Storage (GB)
ml.r5.12xlarge	Memory optimized	No	48	384	Amazon EBS Only
ml.r5.16xlarge	Memory optimized	No	64	512	Amazon EBS Only
ml.r5.24xlarge	Memory optimized	No	96	768	Amazon EBS Only

## Instances with 1 or more GPUs

The following table lists the Amazon EC2 instance types with 1 or more GPUs attached that are available for use with Studio Classic notebooks. It also lists information about the specifications of each instance type. The default instance type for GPU-based images is `ml.g4dn.xlarge`.

For detailed information on which instance types fit your use case, and their performance capabilities, see [Amazon Elastic Compute Cloud Instance types](#). For information about pricing for these instance types, see [Amazon EC2 Pricing](#).

### Instances with 1 or more GPUs

Instance	Use case	Fast launch	GPUs	vCPU	Memory (GiB)	GPU Memor (GiB)	Instance Storage (GB)
ml.p3.2xlarge	Accelerated computing	No	1	8	61	16	Amazon EBS Only

Instance	Use case	Fast launch	GPUs	vCPU	Memor (GiB)	GPU Memor (GiB)	Instance Storage (GB)
ml.p3.8xlarge	Accelerated computing	No	4	32	244	64	Amazon EBS Only
ml.p3.16xlarge	Accelerated computing	No	8	64	488	128	Amazon EBS Only
ml.p3dn.24xlarge	Accelerated computing	No	8	96	768	256	2 x 900 NVMe SSD
ml.p4d.24xlarge	Accelerated computing	No	8	96	1152	320 GB HBM2	8 x 1000 NVMe SSD
ml.p4de.24xlarge	Accelerated computing	No	8	96	1152	640 GB HBM2e	8 x 1000 NVMe SSD
ml.g4dn.xlarge	Accelerated computing	Yes	1	4	16	16	1 x 125 NVMe SSD
ml.g4dn.2xlarge	Accelerated computing	No	1	8	32	16	1 x 225 NVMe SSD

Instance	Use case	Fast launch	GPUs	vCPU	Memor (GiB)	GPU Memor (GiB)	Instance Storage (GB)
ml.g4dn.4xlarge	Accelerated computing	No	1	16	64	16	1 x 225 NVMe SSD
ml.g4dn.8xlarge	Accelerated computing	No	1	32	128	16	1 x 900 NVMe SSD
ml.g4dn.12xlarge	Accelerated computing	No	4	48	192	64	1 x 900 NVMe SSD
ml.g4dn.16xlarge	Accelerated computing	No	1	64	256	16	1 x 900 NVMe SSD
ml.g5.xlarge	Accelerated computing	No	1	4	16	24	1 x 250 NVMe SSD
ml.g5.2xlarge	Accelerated computing	No	1	8	32	24	1 x 450 NVMe SSD
ml.g5.4xlarge	Accelerated computing	No	1	16	64	24	1 x 600 NVMe SSD

Instance	Use case	Fast launch	GPUs	vCPU	Memor (GiB)	GPU Memor (GiB)	Instance Storage (GB)
ml.g5.8xlarge	Accelerated computing	No	1	32	128	24	1 x 900 NVMe SSD
ml.g5.12xlarge	Accelerated computing	No	4	48	192	96	1 x 3800 NVMe SSD
ml.g5.16xlarge	Accelerated computing	No	1	64	256	24	1 x 1900 NVMe SSD
ml.g5.24xlarge	Accelerated computing	No	4	96	384	96	1 x 3800 NVMe SSD
ml.g5.48xlarge	Accelerated computing	No	8	192	768	192	2 x 3800 NVMe SSD

## Amazon SageMaker images available for use with Studio Classic

**⚠️ Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

This page lists the SageMaker images and associated kernels that are available in Amazon SageMaker Studio Classic. This page also gives information about the format needed to create the ARN for each image. SageMaker images contain the latest [Amazon SageMaker Python SDK](#) and the latest version of the kernel. For more information, see [Deep Learning Containers Images](#).

## Topics

- [Image ARN format](#)
- [Supported URI tags](#)
- [Supported images](#)
- [Images slated for deprecation](#)
- [Deprecated images](#)

### Image ARN format

The following table lists the image ARN and URI format for each Region. To create the full ARN for an image, replace the *resource-identifier* placeholder with the corresponding resource identifier for the image. The resource identifier is found in the SageMaker images and kernels table. To create the full URI for an image, replace the *tag* placeholder with the corresponding cpu or gpu tag. For the list of tags you can use, see [Supported URI tags](#).

 **Note**

SageMaker Distribution images use a distinct set of image ARNs, which are listed in the following table.

Region	Image ARN Format	SageMaker Distribution Image ARN Format	SageMaker Distribution Image URI Format
us-east-1	arn:aws:sagemaker:us-east-1:081325390199:image/ <i>resource-identifier</i>	arn:aws:sagemaker:us-east-1:885854791233:image/ <i>resource-identifier</i>	885854791233.dkr.ecr.us-east-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>

Region	Image ARN Format	SageMaker Distribution Image ARN Format	SageMaker Distribution Image URI Format
us-east-2	arn:aws:sagemaker:us-east-2:429704687514:image/ <i>resource-identifier</i>	arn:aws:sagemaker:us-east-2:137914896644:image/ <i>resource-identifier</i>	137914896644.dkr.ecr.us-east-2.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
us-west-1	arn:aws:sagemaker:us-west-1:742091327244:image/ <i>resource-identifier</i>	arn:aws:sagemaker:us-west-1:053634841547:image/ <i>resource-identifier</i>	053634841547.dkr.ecr.us-west-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
us-west-2	arn:aws:sagemaker:us-west-2:236514542706:image/ <i>resource-identifier</i>	arn:aws:sagemaker:us-west-2:542918446943:image/ <i>resource-identifier</i>	542918446943.dkr.ecr.us-west-2.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
af-south-1	arn:aws:sagemaker:af-south-1:559312083959:image/ <i>resource-identifier</i>	arn:aws:sagemaker:af-south-1:238384257742:image/ <i>resource-identifier</i>	238384257742.dkr.ecr.af-south-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
ap-east-1	arn:aws:sagemaker:ap-east-1:493642496378:image/ <i>resource-identifier</i>	arn:aws:sagemaker:ap-east-1:523751269255:image/ <i>resource-identifier</i>	523751269255.dkr.ecr.ap-east-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>

Region	Image ARN Format	SageMaker Distribution Image ARN Format	SageMaker Distribution Image URI Format
ap-south-1	arn:aws:sagemaker:ap-south-1:394103062818:image/ <i>resource-identifier</i>	arn:aws:sagemaker:ap-south-1:245090515133:image/ <i>resource-identifier</i>	245090515133.dkr.ecr.ap-south-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
ap-northeast-2	arn:aws:sagemaker:ap-northeast-2:806072073708:image/ <i>resource-identifier</i>	arn:aws:sagemaker:ap-northeast-2:064688005998:image/ <i>resource-identifier</i>	064688005998.dkr.ecr.ap-northeast-2.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
ap-southeast-1	arn:aws:sagemaker:ap-southeast-1:492261229750:image/ <i>resource-identifier</i>	arn:aws:sagemaker:ap-southeast-1:022667117163:image/ <i>resource-identifier</i>	022667117163.dkr.ecr.ap-southeast-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
ap-southeast-2	arn:aws:sagemaker:ap-southeast-2:452832661640:image/ <i>resource-identifier</i>	arn:aws:sagemaker:ap-southeast-2:648430277019:image/ <i>resource-identifier</i>	648430277019.dkr.ecr.ap-southeast-2.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
ap-northeast-1	arn:aws:sagemaker:ap-northeast-1:102112518831:image/ <i>resource-identifier</i>	arn:aws:sagemaker:ap-northeast-1:010972774902:image/ <i>resource-identifier</i>	010972774902.dkr.ecr.ap-northeast-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>

Region	Image ARN Format	SageMaker Distribution Image ARN Format	SageMaker Distribution Image URI Format
ca-central-1	arn:aws:sagemaker:ca-central-1:310906938811:image/ <i>resource-identifier</i>	arn:aws:sagemaker:ca-central-1:481561238223:image/ <i>resource-identifier</i>	481561238223.dkr.ecr.ca-central-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
eu-central-1	arn:aws:sagemaker:eu-central-1:936697816551:image/ <i>resource-identifier</i>	arn:aws:sagemaker:eu-central-1:545423591354:image/ <i>resource-identifier</i>	545423591354.dkr.ecr.eu-central-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
eu-west-1	arn:aws:sagemaker:eu-west-1:470317259841:image/ <i>resource-identifier</i>	arn:aws:sagemaker:eu-west-1:819792524951:image/ <i>resource-identifier</i>	819792524951.dkr.ecr.eu-west-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
eu-west-2	arn:aws:sagemaker:eu-west-2:712779665605:image/ <i>resource-identifier</i>	arn:aws:sagemaker:eu-west-2:021081402939:image/ <i>resource-identifier</i>	021081402939.dkr.ecr.eu-west-2.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
eu-west-3	arn:aws:sagemaker:eu-west-3:615547856133:image/ <i>resource-identifier</i>	arn:aws:sagemaker:eu-west-3:856416204555:image/ <i>resource-identifier</i>	856416204555.dkr.ecr.eu-west-3.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>

Region	Image ARN Format	SageMaker Distribution Image ARN Format	SageMaker Distribution Image URI Format
eu-north-1	arn:aws:sagemaker:eu-north-1:243637512696:image/ <i>resource-identifier</i>	arn:aws:sagemaker:eu-north-1:175620155138:image/ <i>resource-identifier</i>	175620155138.dkr.ecr.eu-north-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
eu-south-1	arn:aws:sagemaker:eu-south-1:592751261982:image/ <i>resource-identifier</i>	arn:aws:sagemaker:eu-south-1:810671768855:image/ <i>resource-identifier</i>	810671768855.dkr.ecr.eu-south-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
sa-east-1	arn:aws:sagemaker:sa-east-1:782484402741:image/ <i>resource-identifier</i>	arn:aws:sagemaker:sa-east-1:567556641782:image/ <i>resource-identifier</i>	567556641782.dkr.ecr.sa-east-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
ap-northeast-3	arn:aws:sagemaker:ap-northeast-3:792733760839:image/ <i>resource-identifier</i>	arn:aws:sagemaker:ap-northeast-3:564864627153:image/ <i>resource-identifier</i>	564864627153.dkr.ecr.ap-northeast-3.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
ap-southeast-3	arn:aws:sagemaker:ap-southeast-3:276181064229:image/ <i>resource-identifier</i>	arn:aws:sagemaker:ap-southeast-3:370607712162:image/ <i>resource-identifier</i>	370607712162.dkr.ecr.ap-southeast-3.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>

Region	Image ARN Format	SageMaker Distribution Image ARN Format	SageMaker Distribution Image URI Format
me-south-1	arn:aws:sagemaker:me-south-1:117516905037:image/ <i>resource-identifier</i>	arn:aws:sagemaker:me-south-1:523774347010:image/ <i>resource-identifier</i>	523774347010.dkr.ecr.me-south-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>
me-central-1	arn:aws:sagemaker:me-central-1:103105715889:image/ <i>resource-identifier</i>	arn:aws:sagemaker:me-central-1:358593528301:image/ <i>resource-identifier</i>	358593528301.dkr.ecr.me-central-1.amazonaws.com/sagemaker-distribution-prod: <i>tag</i>

## Supported URI tags

The following list shows the tags you can include in your image URI.

- 1-cpu
- 1-gpu
- 0-cpu
- 0-gpu

## The following examples show URIs with various tag formats:

- 542918446943.dkr.ecr.us-west-2.amazonaws.com/sagemaker-distribution-prod:1-cpu
- 542918446943.dkr.ecr.us-west-2.amazonaws.com/sagemaker-distribution-prod:0-gpu

## Supported images

The following table gives information about the SageMaker images and associated kernels that are available in Amazon SageMaker Studio Classic. It also gives information about the resource identifier and Python version included in the image.

### SageMaker images and kernels

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
SageMaker Distribution v1 CPU	<p>SageMaker Distribution v1 CPU is a Python 3.10 image that includes popular frameworks for machine learning, data science and data analytics on CPU. This includes deep learning frameworks like PyTorch, TensorFlow and Keras; popular Python packages like numpy, scikit-learn and pandas; and IDEs like Jupyter Lab. For more information, see the <a href="#">Amazon SageMaker Distribution</a> repo.</p>	sagemaker-distribution-cpu-v1	Python 3 (python3)	Python 3.10
SageMaker Distribution v1 GPU	<p>SageMaker Distribution v1 GPU is a Python 3.10 image that includes popular frameworks for</p>	sagemaker-distribution-gpu-v1	Python 3 (python3)	Python 3.10

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
	machine learning, data science and data analytics on GPU. This includes deep learning frameworks like PyTorch, TensorFlow and Keras; popular Python packages like numpy, scikit-learn and pandas; and IDEs like Jupyter Lab. For more information, see the <a href="#">Amazon SageMaker Distribution</a> repo.			
Base Python 3.0	Official Python 3.10 image from DockerHub with boto3 and AWS CLI included.	sagemaker-base-python-310-v1	Python 3 (python3)	Python 3.10

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
Data Science 4.0	Data Science 4.0 is a Python 3.11 <a href="#">conda</a> image based on Ubuntu version 22.04. It includes the most commonly used Python packages and libraries, such as NumPy and SciKit Learn.	sagemaker-data-science-311-v1	Python 3 (python3)	Python 3.11
Data Science 3.0	Data Science 3.0 is a Python 3.10 <a href="#">conda</a> image based on Ubuntu version 22.04. It includes the most commonly used Python packages and libraries, such as NumPy and SciKit Learn.	sagemaker-data-science-310-v1	Python 3 (python3)	Python 3.10

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
Geospatial 1.0	Amazon SageMaker geospatial is a Python image consisting of commonly used geospatial libraries such as GDAL, Fiona, GeoPandas, Shapley, and Rasterio. It allows you to visualize geospatial data within SageMaker AI. For more information, see <a href="#">Amazon SageMaker geospatial Notebook SDK</a>	sagemaker-geospatial-1.0	Python 3 (python3)	Python 3.10

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
SparkAnalytics 3.0	The SparkAnalytics 3.0 image provides Spark and PySpark kernel options on Amazon SageMaker Studio Classic, including SparkMagic Spark, SparkMagic PySpark, Glue Spark, and Glue PySpark, enabling flexible distributed data processing.	sagemaker-sparkanalytics-311-v1	<ul style="list-style-type: none"><li>• SparkMagic Spark (sparkkernel)</li><li>• SparkMagic PySpark (pysparkkernel)</li><li>• Glue Spark (glue_spark)</li><li>• Glue PySpark (glue_pyspark)</li></ul>	Python 3.11

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
SparkAnalytics 2.0	Anaconda Individual Edition with PySpark and Spark kernels. For more information, see <a href="#">sparkmagic</a> .	sagemaker-sparkanalytics-310-v1	<ul style="list-style-type: none"> <li>• SparkMagic Spark (conda-env-sm_sparkmagic-sparkkernel)</li> <li>• SparkMagic PySpark (conda-env-sm_sparkmagic-pysparkkernel)</li> <li>• Glue Spark (conda-env-sm_glue_is-glue_spark)</li> <li>• Glue Python [PySpark and Ray] (conda-env-sm_glue_is-glue_pyspark)</li> </ul>	Python 3.10
PyTorch 2.4.0 Python 3.11 CPU Optimized	The AWS Deep Learning Containers for PyTorch 2.4.0 with CUDA 12.4 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.4.0-cpu-py311	Python 3 (python3)	Python 3.11

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
PyTorch 2.4.0 Python 3.11 GPU Optimized	The AWS Deep Learning Containers for PyTorch 2.4.0 with CUDA 12.4 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.4.0-gpu-py311	Python 3 (python3)	Python 3.11
PyTorch 2.3.0 Python 3.11 CPU Optimized	The AWS Deep Learning Containers for PyTorch 2.3.0 with CUDA 12.1 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.3.0-cpu-py311	Python 3 (python3)	Python 3.11

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
PyTorch 2.3.0 Python 3.11 GPU Optimized	The AWS Deep Learning Containers for PyTorch 2.3.0 with CUDA 12.1 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.3.0-gpu-py311	Python 3 (python3)	Python 3.11
PyTorch 2.2.0 Python 3.10 CPU Optimized	The AWS Deep Learning Containers for PyTorch 2.2 with CUDA 12.1 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.2.0-cpu-py310	Python 3 (python3)	Python 3.10

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
PyTorch 2.2.0 Python 3.10 GPU Optimized	The AWS Deep Learning Containers for PyTorch 2.2 with CUDA 12.1 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.2.0-gpu-py310	Python 3 (python3)	Python 3.10
PyTorch 2.1.0 Python 3.10 CPU Optimized	The AWS Deep Learning Containers for PyTorch 2.1 with CUDA 12.1 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.1.0-cpu-py310	Python 3 (python3)	Python 3.10

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
PyTorch 2.1.0 Python 3.10 GPU Optimized	The AWS Deep Learning Containers for PyTorch 2.1 with CUDA 12.1 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.1.0-gpu-py310	Python 3 (python3)	Python 3.10
PyTorch 1.13 HuggingFace Python 3.10 Neuron Optimized	PyTorch 1.13 image with HuggingFace and Neuron packages installed for training on Trainium instances optimized for performance and scale on AWS.	pytorch-1.13-hf-neuron-py310	Python 3 (python3)	Python 3.10
PyTorch 1.13 Python 3.10 Neuron Optimized	PyTorch 1.13 image with Neuron packages installed for training on Trainium instances optimized for performance and scale on AWS.	pytorch-1.13-neuron-py310	Python 3 (python3)	Python 3.10

SageMaker Image	Description	Resource Identifier	Kernels (and Identifier)	Python Version
TensorFlow 2.14.0 Python 3.10 CPU Optimized	The AWS Deep Learning Containers for TensorFlow 2.14 with CUDA 11.8 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	tensorflow-2.14.1-cpu-py310-ubuntu20.04-sagemaker-v1.0	Python 3 (python3)	Python 3.10
TensorFlow 2.14.0 Python 3.10 GPU Optimized	The AWS Deep Learning Containers for TensorFlow 2.14 with CUDA 11.8 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	tensorflow-2.14.1-gpu-py310-cu118-ubuntu20.04-sagemaker-v1.0	Python 3 (python3)	Python 3.10

## Images slated for deprecation

SageMaker AI ends support for images the day after any of the packages in the image reach end-of-life by their publisher. The following SageMaker images are slated for deprecation.

Images based on Python 3.8 reached [end-of-life](#) on October 31st, 2024. Starting on November 1, 2024, SageMaker AI will discontinue support for these images and they will not be selectable from the Studio Classic UI. To avoid non-compliance issues, if you're using any of these images, we recommend that you move to an image with a later version.

## SageMaker images slated for deprecation

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
SageMaker Distribution v0.12 CPU	November 1, 2024	SageMaker Distribution v0 CPU is a Python 3.8 image that includes popular frameworks for machine learning, data science and visualization on CPU. This includes deep learning frameworks like PyTorch, TensorFlow and Keras; popular Python packages like numpy, scikit-learn and pandas; and IDEs like Jupyter Lab. For more information, see the <a href="#">Amazon</a>	sagemaker-distribution-cpu-v0	Python 3 (python3)	Python 3.8

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
		<p><a href="#">SageMaker AI Distribution</a> repo.</p>			
SageMaker Distribution v0.12 GPU	November 1, 2024	<p>SageMaker Distribution v0 GPU is a Python 3.8 image that includes popular frameworks for machine learning, data science and visualization on GPU. This includes deep learning frameworks like PyTorch, TensorFlow and Keras; popular Python packages like numpy, scikit-learn and pandas; and IDEs like Jupyter Lab. For more information, see the <a href="#">Amazon SageMaker AI Distribution</a> repo.</p>	sagemaker-distribution-gpu-v0	Python 3 (python3)	Python 3.8

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
Base Python 2.0	November 1, 2024	Official Python 3.8 image from DockerHub with boto3 and AWS CLI included.	sagemaker-base-python-38	Python 3 (python3)	Python 3.8
Data Science 2.0	November 1, 2024	Data Science 2.0 is a Python 3.8 <a href="#">conda</a> image based on Ubuntu version 22.04. It includes the most commonly used Python packages and libraries, such as NumPy and SciKit Learn.	sagemaker-data-science-38	Python 3 (python3)	Python 3.8

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
PyTorch 1.13 Python 3.9 CPU Optimized	November 1, 2024	The AWS Deep Learning Containers for PyTorch 1.13 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-1.13-cpu-py39	Python 3 (python3)	Python 3.9

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
PyTorch 1.13 Python 3.9 GPU Optimized	November 1, 2024	The AWS Deep Learning Containers for PyTorch 1.13 with CUDA 11.7 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-1.13-gpu-py39	Python 3 (python3)	Python 3.9

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
PyTorch 1.12 Python 3.8 CPU Optimized	November 1, 2024	The AWS Deep Learning Containers for PyTorch 1.12 with CUDA 11.3 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">AWS Deep Learning Containers for PyTorch 1.12.0</a> .	pytorch-1.12-cpu-py38	Python 3 (python3)	Python 3.8

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
PyTorch 1.12 Python 3.8 GPU Optimized	November 1, 2024	The AWS Deep Learning Containers for PyTorch 1.12 with CUDA 11.3 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">AWS Deep Learning Containers for PyTorch 1.12.0</a> .	pytorch-1.12-gpu-py38	Python 3 (python3)	Python 3.8

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
PyTorch 1.10 Python 3.8 CPU Optimized	November 1, 2024	The AWS Deep Learning Containers for PyTorch 1.10 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">AWS Deep Learning Containers for PyTorch 1.10.2 on SageMaker AI</a> .	pytorch-1.10-cpu-py38	Python 3 (python3)	Python 3.8

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
PyTorch 1.10 Python 3.8 GPU Optimized	November 1, 2024	The AWS Deep Learning Containers for PyTorch 1.10 with CUDA 11.3 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">AWS Deep Learning Containers for PyTorch 1.10.2 on SageMaker AI</a> .	pytorch-1.10-gpu-py38	Python 3 (python3)	Python 3.8

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
SparkAnalytics 1.0	November 1, 2024	Anaconda Individual Edition with PySpark and Spark kernels. For more information, see <a href="#">sparkmagic</a> .	sagemaker-sparkanalytics-v1	<ul style="list-style-type: none"> <li>SparkNC</li> <li>PySpark (conda-env-sm_sparkmagic_sparkkernel)</li> <li>SparkNC</li> <li>PySpark (conda-env-sm_sparkmagic_py_sparkkernel_el)</li> <li>Glue Spark (conda-env-sm_glue_is-glue_spark)</li> <li>Glue Python [PySpark and Ray]</li> </ul>	Python 3.8

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
				(conda-env-sm_glu_is-glue_pysparl	
TensorFlow 2.13.0 Python 3.10 CPU Optimized	November 1, 2024	The AWS Deep Learning Containers for TensorFlow 2.13 with CUDA 11.8 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers..</a>	tensorflow-2.13.0-cpu-py310-ubuntu20.04-sagemaker-v1.0	Python 3 (python3)	Python 3.10

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.13.0 Python 3.10 GPU Optimized	November 1, 2024	The AWS Deep Learning Containers for TensorFlow 2.13 with CUDA 11.8 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	tensorflow-w-2.13.0-gpu-py310-cu118-ubuntu20.04-sagemaker-v1.0	Python 3	Python 3.10 (python3)

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.6 Python 3.8 CPU Optimized	November 1, 2024	The AWS Deep Learning Containers for TensorFlow 2.6 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">AWS Deep Learning Containers for TensorFlow 2.6</a> .	tensorflow-2.6-cpu-py38-ubuntu20.04-v1	Python 3 (python3)	Python 3.8

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.6 Python 3.8 GPU Optimized	November 1, 2024	The AWS Deep Learning Containers for TensorFlow 2.6 with CUDA 11.2 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">AWS Deep Learning Containers for TensorFlow 2.6</a> .	tensorflow-2.6-gpu-py38-cu112-ubuntu20.04-v1	Python 3 (python3)	Python 3.8

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
PyTorch 2.0.1 Python 3.10 CPU Optimized	November 1, 2024	The AWS Deep Learning Containers for PyTorch 2.0.1 with CUDA 12.1 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.0.1-cpu-py310	Python 3 (python3)	Python 3.10

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
PyTorch 2.0.1 Python 3.10 GPU Optimized	November 1, 2024	The AWS Deep Learning Containers for PyTorch 2.0.1 with CUDA 12.1 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.0.1-gpu-py310	Python 3 (python3)	Python 3.10

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
PyTorch 2.0.0 Python 3.10 CPU Optimized	November 1, 2024	The AWS Deep Learning Containers for PyTorch 2.0.0 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.0.0-cpu-py310	Python 3 (python3)	Python 3.10

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
PyTorch 2.0.0 Python 3.10 GPU Optimized	November 1, 2024	The AWS Deep Learning Containers for PyTorch 2.0.0 with CUDA 11.8 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	pytorch-2.0.0-gpu-py310	Python 3 (python3)	Python 3.10

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.12.0 Python 3.10 CPU Optimized	November 1, 2024	The AWS Deep Learning Containers for TensorFlow 2.12.0 with CUDA 11.2 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	tensorflow-2.12.0-cpu-py310-ubuntu20.04-sagemaker-v1.0	Python 3 (python3)	Python 3.10

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.12.0 Python 3.10 GPU Optimized	November 1, 2024	The AWS Deep Learning Containers for TensorFlow 2.12.0 with CUDA 11.8 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	tensorflow-w-2.12.0-gpu-py310-cu118-ubuntu20.04-sagemaker-v1	Python 3	Python 3.10 (python3)

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.11.0 Python 3.9 CPU Optimized	November 1, 2024	The AWS Deep Learning Containers for TensorFlow 2.11.0 with CUDA 11.2 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	tensorflow-w-2.11.0-cpu-py39-ubuntu20.04-sagemaker-v1.1	Python 3 (python3)	Python 3.9

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.11.0 Python 3.9 GPU Optimized	November 1, 2024	The AWS Deep Learning Containers for TensorFlow 2.11.0 with CUDA 11.2 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	tensorflow-w-2.11.0-gpu-py39-cu112-ubuntu20.04-sagemaker-v1.1	Python 3	Python 3.9 (python3)

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.10 Python 3.9 CPU Optimized	November 1, 2024	The AWS Deep Learning Containers for TensorFlow 2.10 with CUDA 11.2 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	tensorflow-2.10.1-cpu-py39-ubuntu20.04-sagemaker-v1.2	Python 3 (python3)	Python 3.9

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.10 Python 3.9 GPU Optimized	November 1, 2024	The AWS Deep Learning Containers for TensorFlow 2.10 with CUDA 11.2 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">Release Notes for Deep Learning Containers</a> .	tensorflow-w-2.10.1-gpu-py39-ubuntu20.04-sagemaker-v1.2	Python 3	Python 3.9

## Deprecated images

SageMaker AI has ended support for the following images. Deprecation occurs the day after any of the packages in the image reach end-of life by their publisher.

### SageMaker images slated for deprecation

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
Data Science	October 30, 2023	Data Science is a Python 3.7 <a href="#">conda</a> image with the most commonly used Python	datascience-1.0	Python 3	Python 3.7

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
		packages and libraries, such as NumPy and SciKit Learn.			
SageMaker JumpStart Data Science 1.0	October 30, 2023	SageMaker JumpStart Data Science 1.0 is a JumpStart image that includes commonly used packages and libraries.	sagemaker-jumpstart-data-science-1.0	Python 3	Python 3.7
SageMaker JumpStart MXNet 1.0	October 30, 2023	SageMaker JumpStart MXNet 1.0 is a JumpStart image that includes MXNet.	sagemaker-jumpstart-mxnet-1.0	Python 3	Python 3.7
SageMaker JumpStart PyTorch 1.0	October 30, 2023	SageMaker JumpStart PyTorch 1.0 is a JumpStart image that includes PyTorch.	sagemaker-jumpstart-pytorch-1.0	Python 3	Python 3.7

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
SageMaker JumpStart TensorFlow 1.0	October 30, 2023	SageMaker JumpStart TensorFlow 1.0 is a JumpStart image that includes TensorFlow.	sagemaker-jumpstart-tensorflow-1.0	Python 3	Python 3.7
SparkMagic	October 30, 2023	Anaconda Individual Edition with PySpark and Spark kernels. For more information, see <a href="#">sparkmagic</a> .	sagemaker-sparkmagic	<ul style="list-style-type: none"> <li>PySpark</li> <li>Spark</li> </ul>	Python 3.7

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.3 Python 3.7 CPU Optimized	October 30, 2023	The AWS Deep Learning Containers for TensorFlow 2.3 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">AWS Deep Learning Containers with TensorFlow 2.3.0</a> .	tensorflow-2.3-cpu-py37-ubuntu18.04-v1	Python 3	Python 3.7

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 2.3 Python 3.7 GPU Optimized	October 30, 2023	The AWS Deep Learning Containers for TensorFlow 2.3 with CUDA 11.0 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">AWS Deep Learning Containers for TensorFlow 2.3.1 with CUDA 11.0.</a>	tensorflow-2.3-gpu-py37-cu10-ubuntu18.04-v3	Python 3	Python 3.7

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 1.15 Python 3.7 CPU Optimized	October 30, 2023	The AWS Deep Learning Containers for TensorFlow 1.15 include containers for training on CPU, optimized for performance and scale on AWS. For more information, see <a href="#">AWS Deep Learning Containers v7.0 for TensorFlow</a> .	tensorflow-w-1.15-cp37-ubuntu18.04-v7	Python 3	Python 3.7

SageMaker Image	Deprecation date	Description	Resource Identifier	Kernels	Python Version
TensorFlow 1.15 Python 3.7 GPU Optimized	October 30, 2023	The AWS Deep Learning Containers for TensorFlow 1.15 with CUDA 11.0 include containers for training on GPU, optimized for performance and scale on AWS. For more information, see <a href="#">AWS Deep Learning Containers v7.0 for TensorFlow</a> .	tensorflow-1.15-gpu-py37-cu110-ubuntu18.04-v8	Python 3	Python 3.7

## Customize Amazon SageMaker Studio Classic

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

There are four options for customizing your Amazon SageMaker Studio Classic environment. You bring your own SageMaker image, use a lifecycle configuration script, attach suggested Git repos to Studio Classic, or create kernels using persistent Conda environments in Amazon EFS. Use each option individually, or together.

- **Bring your own SageMaker image:** A SageMaker image is a file that identifies the kernels, language packages, and other dependencies required to run a Jupyter notebook in Amazon SageMaker Studio Classic. Amazon SageMaker AI provides many built-in images for you to use. If you need different functionality, you can bring your own custom images to Studio Classic.
- **Use lifecycle configurations with Amazon SageMaker Studio Classic:** Lifecycle configurations are shell scripts triggered by Amazon SageMaker Studio Classic lifecycle events, such as starting a new Studio Classic notebook. You can use lifecycle configurations to automate customization for your Studio Classic environment. For example, you can install custom packages, configure notebook extensions, preload datasets, and set up source code repositories.
- **Attach suggested Git repos to Studio Classic:** You can attach suggested Git repository URLs at the Amazon SageMaker AI domain or user profile level. Then, you can select the repo URL from the list of suggestions and clone that into your environment using the Git extension in Studio Classic.
- **Persist Conda environments to the Studio Classic Amazon EFS volume:** Studio Classic uses an Amazon EFS volume as a persistent storage layer. You can save your Conda environment on this Amazon EFS volume, then use the saved environment to create kernels. Studio Classic automatically picks up all valid environments saved in Amazon EFS as KernelGateway kernels. These kernels persist through restart of the kernel, app, and Studio Classic. For more information, see the **Persist Conda environments to the Studio Classic EFS volume** section in [Four approaches to manage Python packages in Amazon SageMaker Studio Classic notebooks](#).

The following topics show how to use these three options to customize your Amazon SageMaker Studio Classic environment.

## Topics

- [Bring your own SageMaker image](#)
- [Use lifecycle configurations to customize Studio Classic](#)
- [Attach Suggested Git Repos to Studio Classic](#)

## Bring your own SageMaker image

### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the

Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

A SageMaker image is a file that identifies the kernels, language packages, and other dependencies required to run a Jupyter notebook in Amazon SageMaker Studio Classic. These images are used to create an environment that you then run Jupyter notebooks from. Amazon SageMaker AI provides many built-in images for you to use. For the list of built-in images, see [Amazon SageMaker images available for use with Studio Classic](#).

If you need different functionality, you can bring your own custom images to Studio Classic. You can create images and image versions, and attach image versions to your domain or shared space, using the SageMaker AI control panel, the [AWS SDK for Python \(Boto3\)](#), and the [AWS Command Line Interface \(AWS CLI\)](#). You can also create images and image versions using the SageMaker AI console, even if you haven't onboarded to a SageMaker AI domain. SageMaker AI provides sample Dockerfiles to use as a starting point for your custom SageMaker images in the [SageMaker Studio Classic Custom Image Samples](#) repository.

The following topics explain how to bring your own image using the SageMaker AI console or AWS CLI, then launch the image in Studio Classic. For a similar blog article, see [Bringing your own R environment to Amazon SageMaker Studio Classic](#). For notebooks that show how to bring your own image for use in training and inference, see [Amazon SageMaker Studio Classic Container Build CLI](#).

## Key terminology

The following section defines key terms for bringing your own image to use with Studio Classic.

- **Dockerfile:** A Dockerfile is a file that identifies the language packages and other dependencies for your Docker image.
- **Docker image:** The Docker image is a built Dockerfile. This image is checked into Amazon ECR and serves as the basis of the SageMaker AI image.
- **SageMaker image:** A SageMaker image is a holder for a set of SageMaker AI image versions based on Docker images. Each image version is immutable.
- **Image version:** An image version of a SageMaker image represents a Docker image and is stored in an Amazon ECR repository. Each image version is immutable. These image versions can be attached to a domain or shared space and used with Studio Classic.

## Topics

- [Custom SageMaker image specifications](#)
- [Prerequisites](#)
- [Add a Docker image compatible with Studio Classic to Amazon ECR](#)
- [Create a custom SageMaker image](#)
- [Attach a custom SageMaker image](#)
- [Launch a custom SageMaker image in Amazon SageMaker Studio Classic](#)
- [Clean up resources](#)

### Custom SageMaker image specifications

#### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following specifications apply to the container image that is represented by a SageMaker AI image version.

#### Running the image

ENTRYPOINT and CMD instructions are overridden to enable the image to run as a KernelGateway app.

Port 8888 in the image is reserved for running the KernelGateway web server.

#### Stopping the image

The DeleteApp API issues the equivalent of a docker stop command. Other processes in the container won't get the SIGKILL/SIGTERM signals.

#### Kernel discovery

SageMaker AI recognizes kernels as defined by Jupyter [kernel specs](#).

You can specify a list of kernels to display before running the image. If not specified, python3 is displayed. Use the [DescribeAppImageConfig](#) API to view the list of kernels.

Conda environments are recognized as kernel specs by default.

## File system

The /opt/.sagemakerinternal and /opt/ml directories are reserved. Any data in these directories might not be visible at runtime.

## User data

Each user in a domain gets a user directory on a shared Amazon Elastic File System volume in the image. The location of the current user's directory on the Amazon EFS volume is configurable. By default, the location of the directory is /home/sagemaker-user.

SageMaker AI configures POSIX UID/GID mappings between the image and the host. This defaults to mapping the root user's UID/GID (0/0) to the UID/GID on the host.

You can specify these values using the [CreateAppImageConfig API](#).

## GID/UID limits

Amazon SageMaker Studio Classic only supports the following DefaultUID and DefaultGID combinations:

- DefaultUID: 1000 and DefaultGID: 100, which corresponds to a non-privileged user.
- DefaultUID: 0 and DefaultGID: 0, which corresponds to root access.

## Metadata

A metadata file is located at /opt/ml/metadata/resource-metadata.json. No additional environment variables are added to the variables defined in the image. For more information, see [Get App Metadata](#).

## GPU

On a GPU instance, the image is run with the --gpus option. Only the CUDA toolkit should be included in the image not the NVIDIA drivers. For more information, see [NVIDIA User Guide](#).

## Metrics and logging

Logs from the KernelGateway process are sent to Amazon CloudWatch in the customer's account. The name of the log group is /aws/sagemaker/studio. The name of the log stream is \$domainID/\$userProfileName/KernelGateway/\$appName.

## Image size

Limited to 35 GB. To view the size of your image, run docker image ls.

## Sample Dockerfile

The following sample Dockerfile creates an image based Amazon Linux 2, installs third party packages and the python3 kernel, and sets the scope to the non-privileged user.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:2

ARG NB_USER="sagemaker-user"
ARG NB_UID="1000"
ARG NB_GID="100"

RUN \
    yum install --assumeyes python3 shadow-utils && \
    useradd --create-home --shell /bin/bash --gid "${NB_GID}" --uid ${NB_UID} \
${NB_USER} && \
    yum clean all && \
    python3 -m pip install ipykernel && \
    python3 -m ipykernel install

USER ${NB_UID}
```

## Prerequisites

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You must satisfy the following prerequisites to bring your own container for use with Amazon SageMaker Studio Classic.

- The Docker application. For information about setting up Docker, see [Orientation and setup](#).
- Install the AWS CLI by following the steps in [Getting started with the AWS CLI](#).
- A local copy of any Dockerfile for creating a Studio Classic compatible image. For sample custom images, see the [SageMaker AI Studio Classic custom image samples](#) repository.
- Permissions to access the Amazon Elastic Container Registry (Amazon ECR) service. For more information, see [Amazon ECR Managed Policies](#).

- An AWS Identity and Access Management execution role that has the [AmazonSageMakerFullAccess](#) policy attached. If you have onboarded to Amazon SageMaker AI domain, you can get the role from the **Domain Summary** section of the SageMaker AI control panel.
- Install the Studio Classic image build CLI by following the steps in [SageMaker Docker Build](#). This CLI enables you to build a Dockerfile using AWS CodeBuild.

## Add a Docker image compatible with Studio Classic to Amazon ECR

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You perform the following steps to add a container image to Amazon ECR:

- Create an Amazon ECR repository.
- Authenticate to Amazon ECR.
- Build a Docker image compatible with Studio Classic.
- Push the image to the Amazon ECR repository.

### Note

The Amazon ECR repository must be in the same AWS Region as Studio Classic.

## To build and add a container image to Amazon ECR

1. Create an Amazon ECR repository using the AWS CLI. To create the repository using the Amazon ECR console, see [Creating a repository](#).

```
aws ecr create-repository \
--repository-name smstudio-custom \
```

```
--image-scanning-configuration scanOnPush=true
```

The response should look similar to the following.

```
{  
    "repository": {  
        "repositoryArn": "arn:aws:ecr:us-east-2:acct-id:repository/smstudio-  
custom",  
        "registryId": "acct-id",  
        "repositoryName": "smstudio-custom",  
        "repositoryUri": "acct-id.dkr.ecr.us-east-2.amazonaws.com/smstudio-custom",  
        ...  
    }  
}
```

2. Build the Dockerfile using the Studio Classic image build CLI. The period (.) specifies that the Dockerfile should be in the context of the build command. This command builds the image and uploads the built image to the ECR repo. It then outputs the image URI.

```
sm-docker build . --repository smstudio-custom:custom
```

The response should look similar to the following.

```
Image URI: <acct-id>.dkr.ecr.<region>.amazonaws.com/<image_name>
```

## Create a custom SageMaker image

### **⚠ Important**

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

## **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

This topic describes how you can create a custom SageMaker image using the SageMaker AI console or AWS CLI.

When you create an image from the console, SageMaker AI also creates an initial image version. The image version represents a container image in [Amazon Elastic Container Registry \(ECR\)](#). The container image must satisfy the requirements to be used in Amazon SageMaker Studio Classic. For more information, see [Custom SageMaker image specifications](#). For information on testing your image locally and resolving common issues, see the [SageMaker Studio Classic Custom Image Samples repo](#).

After you have created your custom SageMaker image, you must attach it to your domain or shared space to use it with Studio Classic. For more information, see [Attach a custom SageMaker image](#).

### Create a SageMaker image from the console

The following section demonstrates how to create a custom SageMaker image from the SageMaker AI console.

#### To create an image

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **Images**.
4. On the **Custom images** page, choose **Create image**.

5. For **Image source**, enter the registry path to the container image in Amazon ECR. The path is in the following format:

*acct-id.dkr.ecr.region.amazonaws.com/repo-name[:tag] or [@digest]*

6. Choose **Next**.
7. Under **Image properties**, enter the following:
  - Image name – The name must be unique to your account in the current AWS Region.
  - (Optional) Display name – The name displayed in the Studio Classic user interface. When not provided, Image name is displayed.
  - (Optional) Description – A description of the image.
  - IAM role – The role must have the [AmazonSageMakerFullAccess](#) policy attached. Use the dropdown menu to choose one of the following options:
    - Create a new role – Specify any additional Amazon Simple Storage Service (Amazon S3) buckets that you want users of your notebooks to have access to. If you don't want to allow access to additional buckets, choose **None**.

SageMaker AI attaches the [AmazonSageMakerFullAccess](#) policy to the role. The role allows users of your notebooks access to the S3 buckets listed next to the checkmarks.

- Enter a custom IAM role ARN – Enter the Amazon Resource Name (ARN) of your IAM role.
- Use existing role – Choose one of your existing roles from the list.
- (Optional) Image tags – Choose **Add new tag**. You can add up to 50 tags. Tags are searchable using the Studio Classic user interface, the SageMaker AI console, or the SageMaker AI Search API.

8. Choose **Submit**.

The new image is displayed in the **Custom images** list and briefly highlighted. After the image has been successfully created, you can choose the image name to view its properties or choose **Create version** to create another version.

### To create another image version

1. Choose **Create version** on the same row as the image.
2. For **Image source**, enter the registry path to the Amazon ECR container image. The container image shouldn't be the same image as used in a previous version of the SageMaker image.

## Create a SageMaker image from the AWS CLI

You perform the following steps to create a SageMaker image from the container image using the AWS CLI.

- Create an Image.
- Create an ImageVersion.
- Create a configuration file.
- Create an AppImageConfig.

### To create the SageMaker image entities

1. Create a SageMaker image.

```
aws sagemaker create-image \
--image-name custom-image \
--role-arn arn:aws:iam::<acct-id>:role/service-role/<execution-role>
```

The response should look similar to the following.

```
{  
    "ImageArn": "arn:aws:sagemaker:us-east-2:acct-id:image/custom-image"  
}
```

2. Create a SageMaker image version from the container image.

```
aws sagemaker create-image-version \
--image-name custom-image \
--base-image <acct-id>.dkr.ecr.<region>.amazonaws.com/smstudio-custom:custom-
image
```

The response should look similar to the following.

```
{  
    "ImageVersionArn": "arn:aws:sagemaker:us-east-2:acct-id:image-version/custom-
image/1"  
}
```

3. Check that the image version was successfully created.

```
aws sagemaker describe-image-version \
--image-name custom-image \
--version-number 1
```

The response should look similar to the following.

```
{  
    "ImageVersionArn": "arn:aws:sagemaker:us-east-2:acct-id:image-version/custom-  
image/1",  
    "ImageVersionStatus": "CREATED"  
}
```

### Note

If the response is "ImageVersionStatus": "CREATED\_FAILED", the response also includes the failure reason. A permissions issue is a common cause of failure. You also can check your Amazon CloudWatch logs if you experience a failure when starting or running the KernelGateway app for a custom image. The name of the log group is /aws/sagemaker/studio. The name of the log stream is \$domainID/\$userProfileName/KernelGateway/\$appName.

4. Create a configuration file, named `app-image-config-input.json`. The `Name` value of `KernelSpecs` must match the name of the `kernelSpec` available in the `Image` associated with this `AppImageConfig`. This value is case sensitive. You can find the available `kernelSpecs` in an image by running `jupyter-kernelspec list` from a shell inside the container. `MountPath` is the path within the image to mount your Amazon Elastic File System (Amazon EFS) home directory. It needs to be different from the path you use inside the container because that path will be overridden when your Amazon EFS home directory is mounted.

### Note

The following `DefaultUID` and `DefaultGID` combinations are the only accepted values:

- `DefaultUID: 1000 and DefaultGID: 100`
- `DefaultUID: 0 and DefaultGID: 0`

```
{  
    "AppImageConfigName": "custom-image-config",  
    "KernelGatewayImageConfig": {  
        "KernelSpecs": [  
            {  
                "Name": "python3",  
                "DisplayName": "Python 3 (ipykernel)"  
            }  
        ],  
        "FileSystemConfig": {  
            "MountPath": "/home/sagemaker-user",  
            "DefaultUid": 1000,  
            "DefaultGid": 100  
        }  
    }  
}
```

5. Create the AppImageConfig using the file created in the previous step.

```
aws sagemaker create-app-image-config \  
--cli-input-json file://app-image-config-input.json
```

The response should look similar to the following.

```
{  
    "AppImageConfigArn": "arn:aws:sagemaker:us-east-2:acct-id:app-image-config/  
    custom-image-config"  
}
```

## Attach a custom SageMaker image

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can

occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

To use a custom SageMaker image, you must attach a version of the image to your domain or shared space. When you attach an image version, it appears in the SageMaker Studio Classic Launcher and is available in the **Select image** dropdown list, which users use to launch an activity or change the image used by a notebook.

To make a custom SageMaker image available to all users within a domain, you attach the image to the domain. To make an image available to all users within a shared space, you can attach the image to the shared space. To make an image available to a single user, you attach the image to the user's profile. When you attach an image, SageMaker AI uses the latest image version by default. You can also attach a specific image version. After you attach the version, you can choose the version from the SageMaker AI Launcher or the image selector when you launch a notebook.

There is a limit to the number of image versions that can be attached at any given time. After you reach the limit, you must detach a version in order to attach another version of the image.

The following sections demonstrate how to attach a custom SageMaker image to your domain using either the SageMaker AI console or the AWS CLI. You can only attach a custom image to a share space using the AWS CLI.

## Attach the SageMaker image to a domain

### Attach the SageMaker image using the Console

This topic describes how you can attach an existing custom SageMaker image version to your domain using the SageMaker AI control panel. You can also create a custom SageMaker image and image version, and then attach that version to your domain. For the procedure to create an image and image version, see [Create a custom SageMaker image](#).

#### To attach an existing image

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the **Domains** page, select the domain to attach the image to.
5. From the **Domain details** page, select the **Environment** tab.
6. On the **Environment** tab, under **Custom SageMaker Studio Classic images attached to domain**, choose **Attach image**.
7. For **Image source**, choose **Existing image**.
8. Choose an existing image from the list.
9. Choose a version of the image from the list.
10. Choose **Next**.
11. Verify the values for **Image name**, **Image display name**, and **Description**.
12. Choose the IAM role. For more information, see [Create a custom SageMaker image](#).
13. (Optional) Add tags for the image.
14. Specify the EFS mount path. This is the path within the image to mount the user's Amazon Elastic File System (EFS) home directory.
15. For **Image type**, select **SageMaker Studio image**.
16. For **Kernel name**, enter the name of an existing kernel in the image. For information on how to get the kernel information from the image, see [DEVELOPMENT](#) in the SageMaker Studio Classic Custom Image Samples repository. For more information, see the **Kernel discovery** and **User data** sections of [Custom SageMaker image specifications](#).
17. (Optional) For **Kernel display name**, enter the display name for the kernel.
18. Choose **Add kernel**.

## 19. Choose **Submit**.

- Wait for the image version to be attached to the domain. When attached, the version is displayed in the **Custom images** list and briefly highlighted.

## Attach the SageMaker image using the AWS CLI

The following sections demonstrate how to attach a custom SageMaker image when creating a new domain or updating your existing domain using the AWS CLI.

### Attach the SageMaker image to a new domain

The following section demonstrates how to create a new domain with the version attached. These steps require that you specify the Amazon Virtual Private Cloud (VPC) information and execution role required to create the domain. You perform the following steps to create the domain and attach the custom SageMaker image:

- Get your default VPC ID and subnet IDs.
- Create the configuration file for the domain, which specifies the image.
- Create the domain with the configuration file.

### To add the custom SageMaker image to your domain

#### 1. Get your default VPC ID.

```
aws ec2 describe-vpcs \
  --filters Name=isDefault,Values=true \
  --query "Vpcs[0].VpcId" --output text
```

The response should look similar to the following.

```
vpc-xxxxxxxx
```

#### 2. Get your default subnet IDs using the VPC ID from the previous step.

```
aws ec2 describe-subnets \
  --filters Name=vpc-id,Values=<vpc-id> \
  --query "Subnets[*].SubnetId" --output json
```

The response should look similar to the following.

```
[  
    "subnet-b55171dd",  
    "subnet-8a5f99c6",  
    "subnet-e88d1392"  
]
```

3. Create a configuration file named `create-domain-input.json`. Insert the VPC ID, subnet IDs, ImageName, and AppImageConfigName from the previous steps. Because `ImageVersionNumber` isn't specified, the latest version of the image is used, which is the only version in this case.

```
{  
    "DomainName": "domain-with-custom-image",  
    "VpcId": "<vpc-id>",  
    "SubnetIds": [  
        "<subnet-ids>"  
    ],  
    "DefaultUserSettings": {  
        "ExecutionRole": "<execution-role>",  
        "KernelGatewayAppSettings": {  
            "CustomImages": [  
                {  
                    "ImageName": "custom-image",  
                    "AppImageConfigName": "custom-image-config"  
                }  
            ]  
        }  
    },  
    "AuthMode": "IAM"  
}
```

4. Create the domain with the attached custom SageMaker image.

```
aws sagemaker create-domain \  
    --cli-input-json file://create-domain-input.json
```

The response should look similar to the following.

```
{
```

```
"DomainArn": "arn:aws:sagemaker:us-east-2:acct-id:domain/d-xxxxxxxxxxxxx",  
"Url": "https://d-xxxxxxxxxxxxx.studio.us-east-2.sagemaker.aws/..."  
}
```

## Attach the SageMaker image to your current domain

If you have onboarded to a SageMaker AI domain, you can attach the custom image to your current domain. For more information about onboarding to a SageMaker AI domain, see [Amazon SageMaker AI domain overview](#). You don't need to specify the VPC information and execution role when attaching a custom image to your current domain. After you attach the version, you must delete all the apps in your domain and reopen Studio Classic. For information about deleting the apps, see [Delete an Amazon SageMaker AI domain](#).

You perform the following steps to add the SageMaker image to your current domain.

- Get your DomainID from SageMaker AI control panel.
- Use the DomainID to get the DefaultUserSettings for the domain.
- Add the ImageName and AppImageConfig as a CustomImage to the DefaultUserSettings.
- Update your domain to include the custom image.

## To add the custom SageMaker image to your domain

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the **Domains** page, select the domain to attach the image to.
5. From the **Domain details** page, select the **Domain settings** tab.
6. From the **Domain settings** tab, under **General settings**, find the DomainId. The ID is in the following format: d-xxxxxxxxxxxxx.
7. Use the domain ID to get the description of the domain.

```
aws sagemaker describe-domain \  
--domain-id <d-xxxxxxxxxxxxx>
```

The response should look similar to the following.

```
{  
    "DomainId": "d-xxxxxxxxxxxxxx",  
    "DefaultUserSettings": {  
        "KernelGatewayAppSettings": {  
            "CustomImages": [  
                ],  
                ...  
            }  
        }  
    }  
}
```

8. Save the default user settings section of the response to a file named `default-user-settings.json`.
9. Insert the `ImageName` and `AppImageConfigName` from the previous steps as a custom image. Because `ImageVersionNumber` isn't specified, the latest version of the image is used, which is the only version in this case.

```
{  
    "DefaultUserSettings": {  
        "KernelGatewayAppSettings": {  
            "CustomImages": [  
                {  
                    "ImageName": "string",  
                    "AppImageConfigName": "string"  
                }  
            ],  
            ...  
        }  
    }  
}
```

10. Use the domain ID and default user settings file to update your domain.

```
aws sagemaker update-domain \  
--domain-id <d-xxxxxxxxxxxxxx> \  
--cli-input-json file://default-user-settings.json
```

The response should look similar to the following.

```
{
```

```
        "DomainArn": "arn:aws:sagemaker:us-east-2:acct-id:domain/d-xxxxxxxxxxxxx"  
    }
```

## Attach the SageMaker image to a shared space

You can only attach the SageMaker image to a shared space using the AWS CLI. After you attach the version, you must delete all of the applications in your shared space and reopen Studio Classic. For information about deleting the apps, see [Delete an Amazon SageMaker AI domain](#).

You perform the following steps to add the SageMaker image to a shared space.

- Get your DomainID from SageMaker AI control panel.
- Use the DomainID to get the DefaultSpaceSettings for the domain.
- Add the ImageName and AppImageConfig as a CustomImage to the DefaultSpaceSettings.
- Update your domain to include the custom image for the shared space.

## To add the custom SageMaker image to your shared space

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the **Domains** page, select the domain to attach the image to.
5. From the **Domain details** page, select the **Domain settings** tab.
6. From the **Domain settings** tab, under **General settings**, find the DomainId. The ID is in the following format: d-xxxxxxxxxxxxx.
7. Use the domain ID to get the description of the domain.

```
aws sagemaker describe-domain \  
  --domain-id <d-xxxxxxxxxxxxx>
```

The response should look similar to the following.

```
{  
    "DomainId": "d-xxxxxxxxxxxxx",  
    ...
```

```
"DefaultSpaceSettings": {  
    "KernelGatewayAppSettings": {  
        "CustomImages": [  
            ],  
            ...  
        }  
    }  
}
```

8. Save the default space settings section of the response to a file named `default-space-settings.json`.
9. Insert the `ImageName` and `AppImageConfigName` from the previous steps as a custom image. Because `ImageVersionNumber` isn't specified, the latest version of the image is used, which is the only version in this case.

```
{  
    "DefaultSpaceSettings": {  
        "KernelGatewayAppSettings": {  
            "CustomImages": [  
                {  
                    "ImageName": "string",  
                    "AppImageConfigName": "string"  
                }  
            ],  
            ...  
        }  
    }  
}
```

10. Use the domain ID and default space settings file to update your domain.

```
aws sagemaker update-domain \  
--domain-id <d-xxxxxxxxxxxx> \  
--cli-input-json file://default-space-settings.json
```

The response should look similar to the following.

```
{  
    "DomainArn": "arn:aws:sagemaker:us-east-2:acct-id:domain/d-xxxxxxxxxxxx"  
}
```

## View the attached image in SageMaker AI

After you create the custom SageMaker image and attach it to your domain, the image appears in the **Environment** tab of the domain. You can only view the attached images for shared spaces using the AWS CLI by using the following command.

```
aws sagemaker describe-domain \
--domain-id <d-xxxxxxxxxxxx>
```

## Launch a custom SageMaker image in Amazon SageMaker Studio Classic

### **Important**

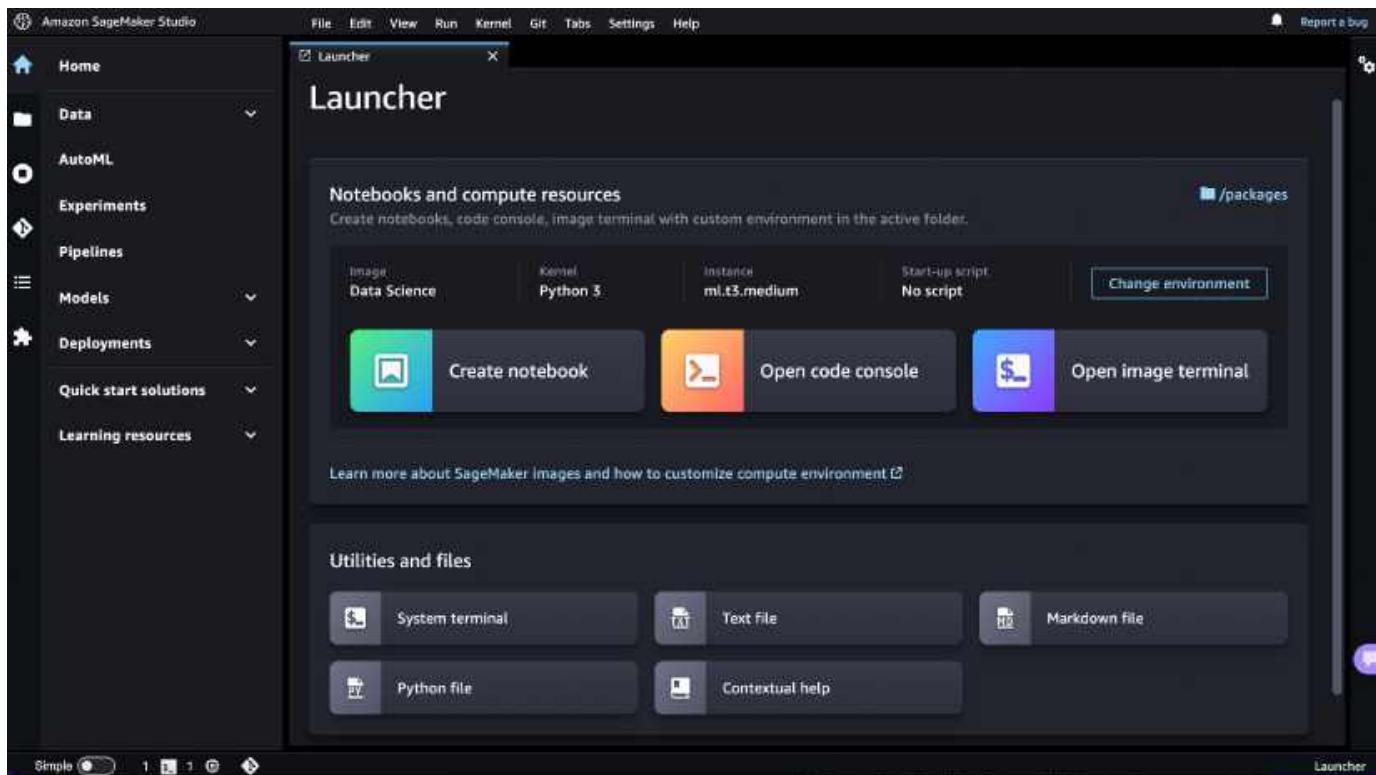
As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

After you create your custom SageMaker image and attach it to your domain or shared space, the custom image and kernel appear in selectors in the **Change environment** dialog box of the Studio Classic Launcher.

### To launch and select your custom image and kernel

1. In Amazon SageMaker Studio Classic, open the Launcher. To open the Launcher, choose **Amazon SageMaker Studio Classic** at the top left of the Studio Classic interface or use the keyboard shortcut **Ctrl + Shift + L**.

To learn about all the available ways to open the Launcher, see [Use the Amazon SageMaker Studio Classic Launcher](#)



2. In the Launcher, in the **Notebooks and compute resources** section, choose **Change environment**.
3. In the **Change environment** dialog, use the dropdown menus to select your **Image** from the **Custom Image** section, and your **Kernel**, then choose **Select**.
4. In the Launcher, choose **Create notebook** or **Open image terminal**. Your notebook or terminal launches in the selected custom image and kernel.

To change your image or kernel in an open notebook, see [Change an Image or a Kernel](#).

### Note

If you encounter an error when launching the image, check your Amazon CloudWatch logs. The name of the log group is `/aws/sagemaker/studio`. The name of the log stream is `$domainID/$userProfileName/KernelGateway/$appName`.

## Clean up resources

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following sections show how to clean up the resources you created in the previous sections from the SageMaker AI console or AWS CLI. You perform the following steps to clean up the resources:

- Detach the image and image versions from your domain.
- Delete the image, image version, and app image config.
- Delete the container image and repository from Amazon ECR. For more information, see [Deleting a repository](#).

### Clean up resources from the SageMaker AI console

The following section shows how to clean up resources from the SageMaker AI console.

When you detach an image from a domain, all versions of the image are detached. When an image is detached, all users of the domain lose access to the image versions. A running notebook that has a kernel session on an image version when the version is detached, continues to run. When the notebook is stopped or the kernel is shut down, the image version becomes unavailable.

#### To detach an image

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **Images**.
4. Under **Custom SageMaker Studio Classic images attached to domain**, choose the image and then choose **Detach**.
5. (Optional) To delete the image and all versions from SageMaker AI, select **Also delete the selected images ....** This does not delete the associated container images from Amazon ECR.

## 6. Choose **Detach**.

### Clean up resources from the AWS CLI

The following section shows how to clean up resources from the AWS CLI.

#### To clean up resources

1. Detach the image and image versions from your domain by passing an empty custom image list to the domain. Open the default-user-settings.json file you created in [Attach the SageMaker image to your current domain](#). To detach the image and image version from a shared space, open the default-space-settings.json file.
2. Delete the custom images and then save the file.

```
"DefaultUserSettings": {  
    "KernelGatewayAppSettings": {  
        "CustomImages": [  
            ],  
            ...  
        },  
        ...  
    }  
}
```

3. Use the domain ID and default user settings file to update your domain. To update your shared space, use the default space settings file.

```
aws sagemaker update-domain \  
--domain-id <d-xxxxxxxxxxxx> \  
--cli-input-json file://default-user-settings.json
```

The response should look similar to the following.

```
{  
    "DomainArn": "arn:aws:sagemaker:us-east-2:acct-id:domain/d-xxxxxxxxxxxx"  
}
```

4. Delete the app image config.

```
aws sagemaker delete-app-image-config \  
--app-image-config-name custom-image-config
```

5. Delete the SageMaker image, which also deletes all image versions. The container images in ECR that are represented by the image versions are not deleted.

```
aws sagemaker delete-image \
--image-name custom-image
```

## Use lifecycle configurations to customize Studio Classic

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker Studio Classic triggers lifecycle configurations shell scripts during important lifecycle events, such as starting a new Studio Classic notebook. You can use lifecycle configurations to automate customization for your Studio Classic environment. This customization includes installing custom packages, configuring notebook extensions, preloading datasets, and setting up source code repositories.

Using lifecycle configurations gives you flexibility and control to configure Studio Classic to meet your specific needs. For example, you can use customized container images with lifecycle configuration scripts to modify your environment. First, create a minimal set of base container images, then install the most commonly used packages and libraries in those images. After you have completed your images, use lifecycle configurations to install additional packages for specific use cases. This gives you the flexibility to modify your environment across your data science and machine learning teams based on need.

Users can only select lifecycle configuration scripts that they are given access to. While you can give access to multiple lifecycle configuration scripts, you can also set default lifecycle configuration scripts for resources. Based on the resource that the default lifecycle configuration is set for, the default either runs automatically or is the first option shown.

For example lifecycle configuration scripts, see the [Studio Classic Lifecycle Configuration examples GitHub repository](#). For a blog on implementing lifecycle configuration, see [Customize Amazon SageMaker Studio Classic using Lifecycle Configurations](#).

**Note**

Each script has a limit of **16384 characters**.

**Topics**

- [Create and associate a lifecycle configuration](#)
- [Set default lifecycle configurations](#)
- [Debug lifecycle configurations](#)
- [Update and detach lifecycle configurations](#)

**Create and associate a lifecycle configuration****⚠ Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker AI provides interactive applications that enable Studio Classic's visual interface, code authoring, and run experience. This series shows how to create a lifecycle configuration and associate it with a SageMaker AI domain.

Application types can be either **JupyterServer** or **KernelGateway**.

- **JupyterServer applications:** This application type enables access to the visual interface for Studio Classic. Every user and shared space in Studio Classic gets its own JupyterServer application.
- **KernelGateway applications:** This application type enables access to the code run environment and kernels for your Studio Classic notebooks and terminals. For more information, see [Jupyter Kernel Gateway](#).

For more information about Studio Classic's architecture and Studio Classic applications, see [Use Amazon SageMaker Studio Classic Notebooks](#).

## Topics

- [Create a lifecycle configuration from the AWS CLI](#)
- [Create a lifecycle configuration from the SageMaker AI console](#)

### Create a lifecycle configuration from the AWS CLI

#### **Important**

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

#### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following topic shows how to create a lifecycle configuration using the AWS CLI to automate customization for your Studio Classic environment.

## Prerequisites

Before you begin, complete the following prerequisites:

- Update the AWS CLI by following the steps in [Installing the current AWS CLI Version](#).
- From your local machine, run `aws configure` and provide your AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).

- Onboard to SageMaker AI domain by following the steps in [Amazon SageMaker AI domain overview](#).

## Step 1: Create a lifecycle configuration

The following procedure shows how to create a lifecycle configuration script that prints Hello World.

 **Note**

Each script can have up to **16,384 characters**.

1. From your local machine, create a file named `my-script.sh` with the following content.

```
#!/bin/bash
set -eux
echo 'Hello World!'
```

2. Convert your `my-script.sh` file into base64 format. This requirement prevents errors that occur from spacing and line break encoding.

```
LCC_CONTENT=`openssl base64 -A -in my-script.sh`
```

3. Create a lifecycle configuration for use with Studio Classic. The following command creates a lifecycle configuration that runs when you launch an associated KernelGateway application.

```
aws sagemaker create-studio-lifecycle-config \
--region region \
--studio-lifecycle-config-name my-studio-lcc \
--studio-lifecycle-config-content $LCC_CONTENT \
--studio-lifecycle-config-app-type KernelGateway
```

Note the ARN of the newly created lifecycle configuration that is returned. This ARN is required to attach the lifecycle configuration to your application.

## Step 2: Attach the lifecycle configuration to your domain, user profile, or shared space

To attach the lifecycle configuration, you must update the `UserSettings` for your domain or user profile, or the `SpaceSettings` for a shared space. Lifecycle configuration scripts that are associated at the domain level are inherited by all users. However, scripts that are associated at the user profile level are scoped to a specific user, while scripts that are associated at the shared space level are scoped to the shared space.

The following example shows how to create a new user profile with the lifecycle configuration attached. You can also create a new domain or space with a lifecycle configuration attached by using the [create-domain](#) and [create-space](#) commands, respectively.

Add the lifecycle configuration ARN from the previous step to the settings for the appropriate app type. For example, place it in the `JupyterServerAppSettings` of the user. You can add multiple lifecycle configurations at the same time by passing a list of lifecycle configurations. When a user launches a JupyterServer application with the AWS CLI, they can pass a lifecycle configuration to use instead of the default. The lifecycle configuration that the user passes must belong to the list of lifecycle configurations in `JupyterServerAppSettings`.

```
# Create a new UserProfile
aws sagemaker create-user-profile --domain-id domain-id \
--user-profile-name user-profile-name \
--region region \
--user-settings '{
  "JupyterServerAppSettings": {
    "LifecycleConfigArns": [
      lifecycle-configuration-arn-list
    ]
  }
}'
```

The following example shows how to update an existing shared space to attach the lifecycle configuration. You can also update an existing domain or user profile with a lifecycle configuration attached by using the [update-domain](#) or [update-user-profile](#) command. When you update the list of lifecycle configurations attached, you must pass all lifecycle configurations as part of the list. If a lifecycle configuration is not part of this list, it will not be attached to the application.

```
aws sagemaker update-space --domain-id domain-id \
--space-name space-name \
--region region \
--space-settings '{
  "JupyterServerAppSettings": {
```

```
"LifecycleConfigArns":  
    [lifecycle-configuration-arn-list]  
}  
}'
```

For information about setting a default lifecycle configuration for a resource, see [Set default lifecycle configurations](#).

### Step 3: Launch application with lifecycle configuration

After you attach a lifecycle configuration to a domain, user profile, or space, the user can select it when launching an application with the AWS CLI. This section describes how to launch an application with an attached lifecycle configuration. For information about changing the default lifecycle configuration after launching a JupyterServer application, see [Set default lifecycle configurations](#).

Launch the desired application type using the `create-app` command and specify the lifecycle configuration ARN in the `resource-spec` argument.

- The following example shows how to create a JupyterServer application with an associated lifecycle configuration. When creating the JupyterServer, the `app-name` must be `default`. The lifecycle configuration ARN passed as part of the `resource-spec` parameter must be part of the list of lifecycle configuration ARNs specified in `UserSettings` for your domain or user profile, or `SpaceSettings` for a shared space.

```
aws sagemaker create-app --domain-id domain-id \  
--region region \  
--user-profile-name user-profile-name \  
--app-type JupyterServer \  
--resource-spec LifecycleConfigArn=lifecycle-configuration-arn \  
--app-name default
```

- The following example shows how to create a KernelGateway application with an associated lifecycle configuration.

```
aws sagemaker create-app --domain-id domain-id \  
--region region \  
--user-profile-name user-profile-name \  
--app-type KernelGateway \  
--resource-spec LifecycleConfigArn=lifecycle-configuration-arn,SageMakerImageArn=sagemaker-image-arn,InstanceType=instance-type \  
--output output
```

```
--app-name app-name
```

## Create a lifecycle configuration from the SageMaker AI console

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following topic shows how to create a lifecycle configuration from the Amazon SageMaker AI console to automate customization for your Studio Classic environment.

### Prerequisites

Before you can begin this tutorial, complete the following prerequisite:

- Onboard to Amazon SageMaker Studio Classic. For more information, see [Onboard to Amazon SageMaker Studio Classic](#).

## Step 1: Create a new lifecycle configuration

You can create a lifecycle configuration by entering a script from the Amazon SageMaker AI console.

 **Note**

Each script can have up to **16,384 characters**.

The following procedure shows how to create a lifecycle configuration script that prints Hello World.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **Lifecycle configurations**.
4. Choose the **Studio** tab.
5. Choose **Create configuration**.
6. Under **Select configuration type**, select the type of application that the lifecycle configuration should be attached to. For more information about selecting which application to attach the lifecycle configuration to, see [Set default lifecycle configurations](#).
7. Choose **Next**.
8. In the section called **Configuration settings**, enter a name for your lifecycle configuration.
9. In the **Scripts** section, enter the following content.

```
#!/bin/bash
set -eux
echo 'Hello World!'
```

10. (Optional) Create a tag for your lifecycle configuration.
11. Choose **Submit**.

## Step 2: Attach the lifecycle configuration to a domain or user profile

Lifecycle configuration scripts associated at the domain level are inherited by all users. However, scripts that are associated at the user profile level are scoped to a specific user.

You can attach multiple lifecycle configurations to a domain or user profile for both JupyterServer and KernelGateway applications.

### Note

To attach a lifecycle configuration to a shared space, you must use the AWS CLI. For more information, see [Create a lifecycle configuration from the AWS CLI](#).

The following sections show how to attach a lifecycle configuration to your domain or user profile.

## Attach to a domain

The following shows how to attach a lifecycle configuration to your existing domain from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain to attach the lifecycle configuration to.
5. From the **Domain details**, choose the **Environment** tab.
6. Under **Lifecycle configurations for personal Studio apps**, choose **Attach**.
7. Under **Source**, choose **Existing configuration**.
8. Under **Studio lifecycle configurations**, select the lifecycle configuration that you created in the previous step.
9. Select **Attach to domain**.

## Attach to your user profile

The following shows how to attach a lifecycle configuration to your existing user profile.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that contains the user profile to attach the lifecycle configuration to.

5. Under **User profiles**, select the user profile.
6. From the **User Details** page, choose **Edit**.
7. On the left navigation, choose **Studio settings**.
8. Under **Lifecycle configurations attached to user**, choose **Attach**.
9. Under **Source**, choose **Existing configuration**.
10. Under **Studio lifecycle configurations**, select the lifecycle configuration that you created in the previous step.
11. Choose **Attach to user profile**.

### Step 3: Launch an application with the lifecycle configuration

After you attach a lifecycle configuration to a domain or user profile, you can launch an application with that attached lifecycle configuration. Choosing which lifecycle configuration to launch with depends on the application type.

- **JupyterServer:** When launching a JupyterServer application from the console, SageMaker AI always uses the default lifecycle configuration. You can't use a different lifecycle configuration when launching from the console. For information about changing the default lifecycle configuration after launching a JupyterServer application, see [Set default lifecycle configurations](#).

To select a different attached lifecycle configuration, you must launch with the AWS CLI. For more information about launching a JupyterServer application with an attached lifecycle configuration from the AWS CLI, see [Create a lifecycle configuration from the AWS CLI](#).

- **KernelGateway:** You can select any of the attached lifecycle configurations when launching a KernelGateway application using the Studio Classic Launcher.

The following procedure describes how to launch a KernelGateway application with an attached lifecycle configuration from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Launch Studio Classic. For more information, see [Launch Amazon SageMaker Studio Classic](#).
3. In the Studio Classic UI, open the Studio Classic Launcher. For more information, see [Use the Amazon SageMaker Studio Classic Launcher](#).
4. In the Studio Classic Launcher, navigate to the **Notebooks and compute resources** section.

5. Click the **Change environment** button.
6. On the **Change environment** dialog, use the dropdown menus to select your **Image**, **Kernel**, **Instance type**, and a **Start-up script**. If there is no default lifecycle configuration, the **Start-up script** value defaults to No script. Otherwise, the **Start-up script** value is your default lifecycle configuration. After you select a lifecycle configuration, you can view the entire script.
7. Click **Select**.
8. Back to the Launcher, click the **Create notebook** to launch a new notebook kernel with your selected image and lifecycle configuration.

#### Step 4: View logs for a lifecycle configuration

You can view the logs for your lifecycle configuration after it has been attached to a domain or user profile.

1. First, provide access to CloudWatch for your AWS Identity and Access Management (IAM) role. Add read permissions for the following log group and log stream.
  - **Log group:**/aws/sagemaker/studio
  - **Log stream:***domain/user-profile/app-type/app-name/LifecycleConfigOnStart*

For information about adding permissions, see [Enabling logging from certain AWS services](#).

2. From within Studio Classic, navigate to the **Running Terminals and Kernels** icon  ) to monitor your lifecycle configuration.
3. Select an application from the list of running applications. Applications with attached lifecycle configurations have an attached indicator icon 
4. Select the indicator icon for your application. This opens a new panel that lists the lifecycle configuration.
5. From the new panel, select **View logs**. This opens a new tab that displays the logs.

## Set default lifecycle configurations

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Although you can attach multiple lifecycle configuration scripts to a single resource, you can only set one default lifecycle configuration for each JupyterServer or KernelGateway application. The behavior of the default lifecycle configuration depends on whether it is set for JupyterServer or KernelGateway apps.

- **JupyterServer apps:** When set as the default lifecycle configuration script for JupyterServer apps, the lifecycle configuration script runs automatically when the user signs in to Studio Classic for the first time or restarts Studio Classic. Use this default lifecycle configuration to automate one-time setup actions for the Studio Classic developer environment, such as installing notebook extensions or setting up a GitHub repo. For an example of this, see [Customize Amazon SageMaker Studio using Lifecycle Configurations](#).
- **KernelGateway apps:** When set as the default lifecycle configuration script for KernelGateway apps, the lifecycle configuration is selected by default in the Studio Classic launcher. Users can launch a notebook or terminal with the default script selected, or they can select a different one from the list of lifecycle configurations.

SageMaker AI supports setting a default lifecycle configuration for the following resources:

- Domains
- User profiles
- Shared spaces

While domains and user profiles support setting a default lifecycle configuration from both the Amazon SageMaker AI console and AWS Command Line Interface, shared spaces only support setting a default lifecycle configuration from the AWS CLI.

You can set a lifecycle configuration as the default when creating a new resource or updating an existing resource. The following topics demonstrate how to set a default lifecycle configuration using the SageMaker AI console and AWS CLI.

## Default lifecycle configuration inheritance

Default lifecycle configurations set at the *domain* level are inherited by all users and shared spaces. Default lifecycle configurations set at the *user* and *shared space* level are scoped to only that user or shared space. User and space defaults override defaults set at the domain level.

A default KernelGateway lifecycle configuration set for a domain applies to all KernelGateway applications launched in the domain. Unless the user selects a different lifecycle configuration from the list presented in the Studio Classic launcher, the default lifecycle configuration is used. The default script also runs if No Script is selected by the user. For more information about selecting a script, see [Step 3: Launch an application with the lifecycle configuration](#).

## Topics

- [Set defaults from the AWS CLI](#)
- [Set defaults from the SageMaker AI console](#)

## Set defaults from the AWS CLI

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

## Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You can set default lifecycle configuration scripts from the AWS CLI for the following resources:

- Domains
- User profiles
- Shared spaces

The following sections outline how to set default lifecycle configuration scripts from the AWS CLI.

## Topics

- [Prerequisites](#)
- [Set a default lifecycle configuration when creating a new resource](#)
- [Set a default lifecycle configuration for an existing resource](#)

## Prerequisites

Before you begin, complete the following prerequisites:

- Update the AWS CLI by following the steps in [Installing the current AWS CLI version](#).
- From your local machine, run `aws configure` and provide your AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).
- Onboard to SageMaker AI domain by following the steps in [Amazon SageMaker AI domain overview](#).
- Create a lifecycle configuration following the steps in [Create and associate a lifecycle configuration](#).

## Set a default lifecycle configuration when creating a new resource

To set a default lifecycle configuration when creating a new domain, user profile, or space, pass the ARN of your previously created lifecycle configuration as part of one of the following AWS CLI commands:

- [create-user-profile](#)
- [create-domain](#)
- [create-space](#)

You must pass the lifecycle configuration ARN for the following values in the KernelGateway or JupyterServer default settings:

- `DefaultResourceSpec:LifecycleConfigArn` - This specifies the default lifecycle configuration for the application type.
- `LifecycleConfigArns` - This is the list of all lifecycle configurations attached to the application type. The default lifecycle configuration must also be part of this list.

For example, the following API call creates a new user profile with a default lifecycle configuration.

```
aws sagemaker create-user-profile --domain-id domain-id \  
--user-profile-name user-profile-name \  
--region region \  
--user-settings '{  
    "KernelGatewayAppSettings": {  
        "DefaultResourceSpec": {  
            "InstanceType": "ml.t3.medium",  
            "LifecycleConfigArn": "lifecycle-configuration-arn"  
        },  
        "LifecycleConfigArns": [lifecycle-configuration-arn-list]  
    }  
}'
```

## Set a default lifecycle configuration for an existing resource

To set or update the default lifecycle configuration for an existing resource, pass the ARN of your previously created lifecycle configuration as part of one of the following AWS CLI commands:

- [update-user-profile](#)
- [update-domain](#)
- [update-space](#)

You must pass the lifecycle configuration ARN for the following values in the KernelGateway or JupyterServer default settings:

- `DefaultResourceSpec:LifecycleConfigArn` - This specifies the default lifecycle configuration for the application type.
- `LifecycleConfigArns` - This is the list of all lifecycle configurations attached to the application type. The default lifecycle configuration must also be part of this list.

For example, the following API call updates a user profile with a default lifecycle configuration.

```
aws sagemaker update-user-profile --domain-id domain-id \  
--user-profile-name user-profile-name \  
--region region \  
--user-settings '{  
    "KernelGatewayAppSettings": {  
        "DefaultResourceSpec": {  
            "InstanceType": "ml.t3.medium",  
            "LifecycleConfigArn": "lifecycle-configuration-arn"  
        },  
        "LifecycleConfigArns": [lifecycle-configuration-arn-list]  
    }  
}'
```

The following API call updates a domain to set a new default lifecycle configuration.

```
aws sagemaker update-domain --domain-id domain-id \  
--region region \  
--default-user-settings '{  
    "JupyterServerAppSettings": {  
        "DefaultResourceSpec": {  
            "InstanceType": "system",  
            "LifecycleConfigArn": "lifecycle-configuration-arn"  
        },  
        "LifecycleConfigArns": [lifecycle-configuration-arn-list]  
    }  
}'
```

```
 }  
}'
```

## Set defaults from the SageMaker AI console

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You can set default lifecycle configuration scripts from the SageMaker AI console for the following resources.

- Domains
- User profiles

You cannot set default lifecycle configuration scripts for shared spaces from the SageMaker AI console. For information about setting defaults for shared spaces, see [Set defaults from the AWS CLI](#).

The following sections outline how to set default lifecycle configuration scripts from the SageMaker AI console.

## Topics

- [Prerequisites](#)
- [Set a default lifecycle configuration for a domain](#)
- [Set a default lifecycle configuration for a user profile](#)

## Prerequisites

Before you begin, complete the following prerequisites:

- Onboard to SageMaker AI domain by following the steps in [Amazon SageMaker AI domain overview](#).
- Create a lifecycle configuration following the steps in [Create and associate a lifecycle configuration](#).

## Set a default lifecycle configuration for a domain

The following procedure shows how to set a default lifecycle configuration for a domain from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the list of domains, select the name of the domain to set the default lifecycle configuration for.
3. From the **Domain details** page, choose the **Environment** tab.
4. Under **Lifecycle configurations for personal Studio apps**, select the lifecycle configuration that you want to set as the default for the domain. You can set distinct defaults for JupyterServer and KernelGateway applications.
5. Choose **Set as default**. This opens a pop up window that lists the current defaults for JupyterServer and KernelGateway applications.
6. Choose **Set as default** to set the lifecycle configuration as the default for its respective application type.

## Set a default lifecycle configuration for a user profile

The following procedure shows how to set a default lifecycle configuration for a user profile from the SageMaker AI console.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the list of domains, select the name of the domain that contains the user profile that you want to set the default lifecycle configuration for.
3. From the **Domain details** page, choose the **User profiles** tab.
4. Select the name of the user profile to set the default lifecycle configuration for. This opens a **User Details** page.
5. From the **User Details** page, choose **Edit**. This opens an **Edit user profile** page.
6. From the **Edit user profile** page, choose **Step 2 Studio settings**.
7. Under **Lifecycle configurations attached to user**, select the lifecycle configuration that you want to set as the default for the user profile. You can set distinct defaults for JupyterServer and KernelGateway applications.
8. Choose **Set as default**. This opens a pop up window that lists the current defaults for JupyterServer and KernelGateway applications.
9. Choose **Set as default** to set the lifecycle configuration as the default for its respective application type.

## Debug lifecycle configurations

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following topics show how to get information about and debug your lifecycle configurations.

### Topics

- [Verify lifecycle configuration process from CloudWatch Logs](#)

- [JupyterServer app failure](#)
- [KernelGateway app failure](#)
- [Lifecycle configuration timeout](#)

## Verify lifecycle configuration process from CloudWatch Logs

Lifecycle configurations only log STDOUT and STDERR.

STDOUT is the default output for bash scripts. You can write to STDERR by appending >&2 to the end of a bash command. For example, echo 'hello'>&2.

Logs for your lifecycle configurations are published to your AWS account using Amazon CloudWatch. These logs can be found in the /aws/sagemaker/studio log stream in the CloudWatch console.

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. Choose **Logs** from the left side. From the dropdown menu, select **Log groups**.
3. On the **Log groups** page, search for aws/sagemaker/studio.
4. Select the log group.
5. On the **Log group details** page, choose the **Log streams** tab.
6. To find the logs for a specific app, search the log streams using the following format:

*domain-id/user-profile-name/app-type/app-name*

For example, to find the lifecycle configuration logs for domain d-m851cu8vbqmz, user profile i-sonic-js, application type JupyterServer and application name test-lcc-echo, use the following search string:

d-m851cu8vbqmz/i-sonic-js/JupyterServer/test-lcc-echo

7. Select the log stream appended with `LifecycleConfigOnStart` to view the script execution logs.

## JupyterServer app failure

If your JupyterServer app crashes because of an issue with the attached lifecycle configuration, Studio Classic displays the following error message on the Studio Classic startup screen.

Failed to create SageMaker Studio due to start-up script failure

Select the `View script logs` link to view the CloudWatch logs for your JupyterServer app.

In the case where the faulty lifecycle configuration is specified in the `DefaultResourceSpec` of your domain, user profile, or shared space, Studio Classic continues to use the lifecycle configuration even after restarting Studio Classic.

To resolve this error, follow the steps in [Set default lifecycle configurations](#) to remove the lifecycle configuration script from the `DefaultResourceSpec` or select another script as the default. Then launch a new JupyterServer app.

### KernelGateway app failure

If your KernelGateway app crashes because of an issue with the attached lifecycle configuration, Studio Classic displays the error message in your Studio Classic Notebook.

Choose `View script logs` to view the CloudWatch logs for your KernelGateway app.

In this case, your lifecycle configuration is specified in the Studio Classic Launcher when launching a new Studio Classic Notebook.

To resolve this error, use the Studio Classic launcher to select a different lifecycle configuration or select `No script`.

#### Note

A default KernelGateway lifecycle configuration specified in `DefaultResourceSpec` applies to all KernelGateway images in the domain, user profile, or shared space unless the user selects a different script from the list presented in the Studio Classic launcher.

The default script also runs if `No Script` is selected by the user. For more information on selecting a script, see [Step 3: Launch an application with the lifecycle configuration](#).

### Lifecycle configuration timeout

There is a lifecycle configuration timeout limitation of 5 minutes. If a lifecycle configuration script takes longer than 5 minutes to run, Studio Classic throws an error.

To resolve this error, ensure that your lifecycle configuration script completes in less than 5 minutes.

To help decrease the run time of scripts, try the following:

- Cut down on necessary steps. For example, limit which conda environments to install large packages in.
- Run tasks in parallel processes.
- Use the nohup command in your script to ensure that hangup signals are ignored and do not stop the execution of the script.

## Update and detach lifecycle configurations

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

A lifecycle configuration script can't be changed after it's created. To update your script, you must create a new lifecycle configuration script and attach it to the respective domain, user profile, or shared space. For more information about creating and attaching the lifecycle configuration, see [Create and associate a lifecycle configuration](#).

The following topic shows how to detach a lifecycle configuration using the AWS CLI and SageMaker AI console.

### Topics

- [Prerequisites](#)
- [Detach using the AWS CLI](#)

### Prerequisites

Before detaching a lifecycle configuration, you must complete the following prerequisite.

- To successfully detach a lifecycle configuration, no running application can be using the lifecycle configuration. You must first shut down the running applications as shown in [Shut Down and Update SageMaker Studio Classic and Studio Classic Apps](#).

## Detach using the AWS CLI

To detach a lifecycle configuration using the AWS CLI, remove the desired lifecycle configuration from the list of lifecycle configurations attached to the resource and pass the list as part of the respective command:

- [update-user-profile](#)
- [update-domain](#)
- [update-space](#)

For example, the following command removes all lifecycle configurations for KernelGateways attached to the domain.

```
aws sagemaker update-domain --domain-id domain-id \  
--region region \  
--default-user-settings '{  
  "KernelGatewayAppSettings": {  
    "LifecycleConfigArns": [  
      []  
    ]  
  }  
}'
```

## Attach Suggested Git Repos to Studio Classic

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker Studio Classic offers a Git extension for you to enter the URL of a Git repository (repo), clone it into your environment, push changes, and view commit history. In addition to this Git extension, you can also attach suggested Git repository URLs at the Amazon SageMaker AI domain or user profile level. Then, you can select the repo URL from the list of suggestions and clone that into your environment using the Git extension in Studio Classic.

The following topics show how to attach Git repo URLs to a domain or user profile from the AWS CLI and SageMaker AI console. You'll also learn how to detach these repository URLs.

## Topics

- [Attach a Git Repository from the AWS CLI](#)
- [Attach a Git Repository from the SageMaker AI Console](#)
- [Detach Git Repos](#)

### Attach a Git Repository from the AWS CLI

#### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following topic shows how to attach a Git repository URL using the AWS CLI, so that Amazon SageMaker Studio Classic automatically suggests it for cloning. After you attach the Git repository URL, you can clone it by following the steps in [Clone a Git Repository in SageMaker Studio Classic](#).

## Prerequisites

Before you begin, complete the following prerequisites:

- Update the AWS CLI by following the steps in [Installing the current AWS CLI Version](#).
- From your local machine, run `aws configure` and provide your AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).
- Onboard to Amazon SageMaker AI domain. For more information, see [Amazon SageMaker AI domain overview](#).

### Attach the Git repo to a domain or user profile

Git repo URLs associated at the domain level are inherited by all users. However, Git repo URLs that are associated at the user profile level are scoped to a specific user. You can attach multiple Git repo URLs to a domain or user profile by passing a list of repository URLs.

The following sections show how to attach a Git repo URL to your domain and user profile.

## Attach to a domain

The following command attaches a Git repo URL to an existing domain.

```
aws sagemaker update-domain --region region --domain-id domain-id \  
  --default-user-settings  
  JupyterServerAppSettings=[{CodeRepositories=[{RepositoryUrl="repository"}]}]
```

## Attach to a user profile

The following shows how to attach a Git repo URL to an existing user profile.

```
aws sagemaker update-user-profile --domain-id domain-id --user-profile-name user-name \  
  --user-settings  
  JupyterServerAppSettings=[{CodeRepositories=[{RepositoryUrl="repository"}]}]
```

## Attach a Git Repository from the SageMaker AI Console

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following topic shows how to associate a Git repository URL from the Amazon SageMaker AI console to clone it in your Studio Classic environment. After you associate the Git repository URL, you can clone it by following the steps in [Clone a Git Repository in SageMaker Studio Classic](#).

### Prerequisites

Before you can begin this tutorial, you must onboard to Amazon SageMaker AI domain. For more information, see [Amazon SageMaker AI domain overview](#).

### Attach the Git repo to a domain or user profile

Git repo URLs associated at the domain level are inherited by all users. However, Git repo URL that are associated at the user profile level are scoped to a specific user.

The following sections show how to attach a Git repo URL to a domain and user profile.

## Attach to a domain

### To attach a Git repo URL to an existing domain

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. Select the domain to attach the Git repo to.
5. On the **domain details** page, choose the **Environment** tab.
6. On the **Suggested code repositories for the domain** tab, choose **Attach**.
7. Under **Source**, enter the Git repository URL.
8. Select **Attach to domain**.

## Attach to a user profile

The following shows how to attach a Git repository URL to an existing user profile.

### To attach a Git repository URL to a user profile

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. Select the domain that includes the user profile to attach the Git repo to.
5. On the **domain details** page, choose the **User profiles** tab.
6. Select the user profile to attach the Git repo URL to.
7. On the **User details** page, choose **Edit**.
8. On the **Studio settings** page, choose **Attach** from the **Suggested code repositories for the user** section.
9. Under **Source**, enter the Git repository URL.
10. Choose **Attach to user**.

## Detach Git Repos

### **⚠ Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

This guide shows how to detach Git repository URLs from an Amazon SageMaker AI domain or user profile using the AWS CLI or Amazon SageMaker AI console.

### Topics

- [Detach a Git repo using the AWS CLI](#)
- [Detach the Git repo using the SageMaker AI console](#)

### Detach a Git repo using the AWS CLI

To detach all Git repo URLs from a domain or user profile, you must pass an empty list of code repositories. This list is passed as part of the `JupyterServerAppSettings` parameter in an `update-domain` or `update-user-profile` command. To detach only one Git repo URL, pass the code repositories list without the desired Git repo URL. This section shows how to detach all Git repo URLs from your domain or user profile using the AWS Command Line Interface (AWS CLI).

#### Detach from a domain

The following command detaches all Git repo URLs from a domain.

```
aws sagemaker update-domain --region region --domain-name domain-name \  
--domain-settings JupyterServerAppSettings={CodeRepositories=[]}
```

#### Detach from a user profile

The following command detaches all Git repo URLs from a user profile.

```
aws sagemaker update-user-profile --domain-name domain-name --user-profile-name user-name\
```

```
--user-settings JupyterServerAppSettings={CodeRepositories=[]}
```

## Detach the Git repo using the SageMaker AI console

The following sections show how to detach a Git repo URL from a domain or user profile using the SageMaker AI console.

### Detach from a domain

Use the following steps to detach a Git repo URL from an existing domain.

#### To detach a Git repo URL from an existing domain

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. Select the domain with the Git repo URL that you want to detach.
5. On the **domain details** page, choose the **Environment** tab.
6. On the **Suggested code repositories for the domain** tab, select the Git repository URL to detach.
7. Choose **Detach**.
8. From the new window, choose **Detach**.

### Detach from a user profile

Use the following steps to detach a Git repo URL from a user profile.

#### To detach a Git repo URL from a user profile

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. Select the domain that includes the user profile with the Git repo URL that you want to detach.
5. On the **domain details** page, choose the **User profiles** tab.
6. Select the user profile with the Git repo URL that you want to detach.
7. On the **User details** page, choose **Edit**.

8. On the **Studio settings** page, select the Git repo URL to detach from the **Suggested code repositories for the user** tab.
9. Choose **Detach**.
10. From the new window, choose **Detach**.

## Perform Common Tasks in Amazon SageMaker Studio Classic

### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following sections describe how to perform common tasks in Amazon SageMaker Studio Classic. For an overview of the Studio Classic interface, see [Amazon SageMaker Studio Classic UI Overview](#).

### Topics

- [Upload Files to SageMaker Studio Classic](#)
- [Clone a Git Repository in SageMaker Studio Classic](#)
- [Stop a Training Job in SageMaker Studio Classic](#)
- [Use TensorBoard in Amazon SageMaker Studio Classic](#)
- [Amazon Q Developer with Amazon SageMaker Studio Classic](#)
- [Manage Your Amazon EFS Storage Volume in SageMaker Studio Classic](#)
- [Provide Feedback on SageMaker Studio Classic](#)
- [Shut Down and Update SageMaker Studio Classic and Studio Classic Apps](#)

## Upload Files to SageMaker Studio Classic

### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the

Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

When you onboard to Amazon SageMaker Studio Classic, a home directory is created for you in the Amazon Elastic File System (Amazon EFS) volume that was created for your team. Studio Classic can only open files that have been uploaded to your directory. The Studio Classic file browser maps to your home directory.

 **Note**

Studio Classic does not support uploading folders. While you can only upload individual files, you can upload multiple files at the same time.

## To upload files to your home directory

1. In the left sidebar, choose the **File Browser** icon (



).

2. In the file browser, choose the **Upload Files** icon



).

3. Select the files you want to upload and then choose **Open**.

4. Double-click a file to open the file in a new tab in Studio Classic.

## Clone a Git Repository in SageMaker Studio Classic

 **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker Studio Classic can only connect only to a local Git repository (repo). This means that you must clone the Git repo from within Studio Classic to access the files in the repo. Studio

Classic offers a Git extension for you to enter the URL of a Git repo, clone it into your environment, push changes, and view commit history. If the repo is private and requires credentials to access, then you are prompted to enter your user credentials. This includes your username and personal access token. For more information about personal access tokens, see [Managing your personal access tokens](#).

Admins can also attach suggested Git repository URLs at the Amazon SageMaker AI domain or user profile level. Users can then select the repo URL from the list of suggestions and clone that into Studio Classic. For more information about attaching suggested repos, see [Attach Suggested Git Repos to Studio Classic](#).

The following procedure shows how to clone a GitHub repo from Studio Classic.

### To clone the repo

1. In the left sidebar, choose the **Git** icon (



).

2. Choose **Clone a Repository**. This opens a new window.
3. In the **Clone Git Repository** window, enter the URL in the following format for the Git repo that you want to clone or select a repository from the list of **Suggested repositories**.

`https://github.com/path-to-git-repo/repo.git`

4. If you entered the URL of the Git repo manually, select **Clone "git-url"** from the dropdown menu.
5. Under **Project directory to clone into**, enter the path to the local directory that you want to clone the Git repo into. If this value is left empty, Studio Classic clones the repo into JupyterLab's root directory.
6. Choose **Clone**. This opens a new terminal window.
7. If the repo requires credentials, you are prompted to enter your username and personal access token. This prompt does not accept passwords, you must use a personal access token. For more information about personal access tokens, see [Managing your personal access tokens](#).
8. Wait for the download to finish. After the repo has been cloned, the **File Browser** opens to display the cloned repo.
9. Double click the repo to open it.
10. Choose the **Git** icon to view the Git user interface which now tracks the repo.

11. To track a different repo, open the repo in the file browser and then choose the **Git** icon.

## Stop a Training Job in SageMaker Studio Classic

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

You can stop a training job with the Amazon SageMaker Studio Classic UI. When you stop a training job, its status changes to **Stopping** at which time billing ceases. An algorithm can delay termination in order to save model artifacts after which the job status changes to **Stopped**. For more information, see the [`stop\_training\_job`](#) method in the AWS SDK for Python (Boto3).

### To stop a training job

1. Follow the [View experiments and runs](#) procedure on this page until you open the **Describe Trial Component** tab.
2. At the upper-right side of the tab, choose **Stop training job**. The **Status** at the top left of the tab changes to **Stopped**.
3. To view the training time and billing time, choose **AWS Settings**.

## Use TensorBoard in Amazon SageMaker Studio Classic

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following doc outlines how to install and run TensorBoard in Amazon SageMaker Studio Classic.

### Note

This guide shows how to open the TensorBoard application through a SageMaker Studio Classic notebook server of an individual SageMaker AI domain user profile. For a more comprehensive TensorBoard experience integrated with SageMaker Training and the access control functionalities of SageMaker AI domain, see [TensorBoard in Amazon SageMaker AI](#).

## Prerequisites

This tutorial requires a SageMaker AI domain. For more information, see [Amazon SageMaker AI domain overview](#)

### Set Up TensorBoardCallback

1. Launch Studio Classic, and open the Launcher. For more information, see [Use the Amazon SageMaker Studio Classic Launcher](#)
2. In the Amazon SageMaker Studio Classic Launcher, under Notebooks and compute resources, choose the **Change environment** button.
3. On the **Change environment** dialog, use the dropdown menus to select the TensorFlow 2.6 Python 3.8 CPU Optimized Studio Classic **Image**.
4. Back to the Launcher, click the **Create notebook** tile. Your notebook launches and opens in a new Studio Classic tab.
5. Run this code from within your notebook cells.
6. Import the required packages.

```
import os
import datetime
import tensorflow as tf
```

7. Create a Keras model.

```
mnist = tf.keras.datasets.mnist

(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

def create_model():
```

```
return tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(10, activation='softmax')
])
```

## 8. Create a directory for your TensorBoard logs

```
LOG_DIR = os.path.join(os.getcwd(), "logs/fit/" +
    datetime.datetime.now().strftime("%Y%m%d-%H%M%S"))
```

## 9. Run training with TensorBoard.

```
model = create_model()
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=LOG_DIR,
                                                       histogram_freq=1)

model.fit(x=x_train,
          y=y_train,
          epochs=5,
          validation_data=(x_test, y_test),
          callbacks=[tensorboard_callback])
```

## 10. Generate the EFS path for the TensorBoard logs. You use this path to set up your logs from the terminal.

```
EFS_PATH_LOG_DIR = "/".join(LOG_DIR.strip("/").split('/')[1:-1])
print (EFS_PATH_LOG_DIR)
```

Retrieve the EFS\_PATH\_LOG\_DIR. You will need it in the TensorBoard installation section.

## Install TensorBoard

1. Click on the Amazon SageMaker Studio Classic button on the top left corner of Studio Classic to open the Amazon SageMaker Studio Classic Launcher. This launcher must be opened

from your root directory. For more information, see [Use the Amazon SageMaker Studio Classic Launcher](#)

2. In the Launcher, under Utilities and files, click System terminal.
3. From the terminal, run the following commands. Copy EFS\_PATH\_LOG\_DIR from the Jupyter notebook. You must run this from the /home/sagemaker-user root directory.

```
pip install tensorboard  
tensorboard --logdir <EFS_PATH_LOG_DIR>
```

## Launch TensorBoard

1. To launch TensorBoard, copy your Studio Classic URL and replace lab? with proxy/6006/ as follows. You must include the trailing / character.

```
https://<YOUR_URL>.studio.region.sagemaker.aws/jupyter/default/proxy/6006/
```

2. Navigate to the URL to examine your results.

## Amazon Q Developer with Amazon SageMaker Studio Classic

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker Studio Classic is an integrated machine learning environment where you can build, train, deploy, and analyze your models all in the same application. You can generate code recommendations and suggest improvements related to code issues by using Amazon Q Developer with Amazon SageMaker AI.

Amazon Q Developer is a generative AI-powered conversational assistant that can help you to understand, build, extend, and operate AWS applications. For more information, see [What is Amazon Q Developer?](#) in the *Amazon Q Developer User Guide*.

Amazon Q Developer is a generative artificial intelligence (AI) powered conversational assistant that can help you understand, build, extend, and operate AWS applications. In the context of an integrated AWS coding environment, Amazon Q can generate code recommendations based on developers' code, as well as their comments in natural language.

Amazon Q has the most support for Java, Python, JavaScript, TypeScript, C#, Go, PHP, Rust, Kotlin, and SQL, as well as the Infrastructure as Code (IaC) languages JSON (AWS CloudFormation), YAML (AWS CloudFormation), HCL (Terraform), and CDK (TypeScript, Python). It also supports code generation for Ruby, C++, C, Shell, and Scala. For examples of how Amazon Q integrates with Amazon SageMaker AI and displays code suggestions in the Amazon SageMaker Studio Classic IDE, see [Code Examples](#) in the *Amazon Q Developer User Guide*.

For more information on using Amazon Q with Amazon SageMaker Studio Classic, see the [Amazon Q Developer User Guide](#).

## Manage Your Amazon EFS Storage Volume in SageMaker Studio Classic

### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The first time a user on your team onboards to Amazon SageMaker Studio Classic, Amazon SageMaker AI creates an Amazon Elastic File System (Amazon EFS) volume for the team. A home directory is created in the volume for each user who onboards to Studio Classic as part of your team. Notebook files and data files are stored in these directories. Users don't have access to other team member's home directories. Amazon SageMaker AI domain does not support mounting custom or additional Amazon EFS volumes.

### **Important**

Don't delete the Amazon EFS volume. If you delete it, the domain will no longer function and all of your users will lose their work.

## To find your Amazon EFS volume

1. Open the [SageMaker AI console](#).
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the **Domains** page, select the domain to find the ID for.
5. From the **Domain details** page, select the **Domain settings** tab.
6. Under **General settings**, find the **Domain ID**. The ID will be in the following format: d-xxxxxxxxxxxx.
7. Pass the Domain ID, as `DomainId`, to the [describe\\_domain](#) method.
8. In the response from `describe_domain`, note the value for the `HomeEfsFileSystemId` key. This is the Amazon EFS file system ID.
9. Open the [Amazon EFS console](#). Make sure the AWS Region is the same Region that's used by Studio Classic.
10. Under **File systems**, choose the file system ID from the previous step.
11. To verify that you've chosen the correct file system, select the **Tags** heading. The value corresponding to the `ManagedByAmazonSageMakerResource` key should match the Studio Classic ID.

For information on how to access the Amazon EFS volume, see [Using file systems in Amazon EFS](#).

To delete the Amazon EFS volume, see [Deleting an Amazon EFS file system](#).

## Provide Feedback on SageMaker Studio Classic

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

Amazon SageMaker AI takes your feedback seriously. We encourage you to provide feedback.

## To provide feedback

1. At the right of SageMaker Studio Classic, find the **Feedback** icon



).

2. Choose a smiley emoji to let us know how satisfied you are with SageMaker Studio Classic and add any feedback you'd care to share with us.
3. Decide whether to share your identity with us, then choose **Submit**.

## Shut Down and Update SageMaker Studio Classic and Studio Classic Apps

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

The following topics show how to shut down and update SageMaker Studio Classic and Studio Classic Apps.

Studio Classic provides a notification icon



)

in the upper-right corner of the Studio Classic UI. This notification icon displays the number of unread notices. To read the notices, select the icon.

Studio Classic provides two types of notifications:

- Upgrade – Displayed when Studio Classic or one of the Studio Classic apps have released a new version. To update Studio Classic, see [Shut down and Update SageMaker Studio Classic](#). To update Studio Classic apps, see [Shut down and Update Studio Classic Apps](#).
- Information – Displayed for new features and other information.

To reset the notification icon, you must select the link in each notice. Read notifications may still display in the icon. This does not indicate that updates are still needed after you have updated Studio Classic and Studio Classic Apps.

To learn how to update [Amazon SageMaker Data Wrangler](#), see [Shut down and Update Studio Classic Apps](#).

To ensure that you have the most recent software updates, update Amazon SageMaker Studio Classic and your Studio Classic apps using the methods outlined in the following topics.

## Topics

- [Shut down and Update SageMaker Studio Classic](#)
- [Shut down and Update Studio Classic Apps](#)

### Shut down and Update SageMaker Studio Classic

#### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

#### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

To update Amazon SageMaker Studio Classic to the latest release, you must shut down the JupyterServer app. You can shut down the JupyterServer app from the SageMaker AI console, from Amazon SageMaker Studio or from within Studio Classic. After the JupyterServer app is shut down,

you must reopen Studio Classic through the SageMaker AI console or from Studio which creates a new version of the JupyterServer app.

You cannot delete the JupyterServer application while the Studio Classic UI is still open in the browser. If you delete the JupyterServer application while the Studio Classic UI is still open in the browser, SageMaker AI automatically re-creates the JupyterServer application.

Any unsaved notebook information is lost in the process. The user data in the Amazon EFS volume isn't impacted.

Some of the services within Studio Classic, like Data Wrangler, run on their own app. To update these services you must delete the app for that service. To learn more, see [Shut down and Update Studio Classic Apps](#).

#### Note

A JupyterServer app is associated with a single Studio Classic user. When you update the app for one user it doesn't affect other users.

The following page shows how to update the JupyterServer App from the SageMaker AI console, from Studio, or from inside Studio Classic.

#### **Shut down and update from the SageMaker AI console**

1. Navigate to <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. Select the domain that includes the Studio Classic application that you want to update.
5. Under **User profiles**, select your user name.
6. Under **Apps**, in the row displaying **JupyterServer**, choose **Action**, then choose **Delete**.
7. Choose **Yes, delete app**.
8. Type **delete** in the confirmation box.
9. Choose **Delete**.
10. After the app has been deleted, launch a new Studio Classic app to get the latest version.

## Shut down and update from Studio

1. Navigate to Studio following the steps in [Launch Amazon SageMaker Studio](#).
2. From the Studio UI, find the applications pane on the left side.
3. From the applications pane, select **Studio Classic**.
4. From the Studio Classic landing page, select the Studio Classic instance to stop.
5. Choose **Stop**.
6. After the app has been stopped, select **Run** to use the latest version.

## Shut down and update from inside Studio Classic

1. Launch Studio Classic.
2. On the top menu, choose **File** then **Shut Down**.
3. Choose one of the following options:
  - **Shutdown Server** – Shuts down the JupyterServer app. Terminal sessions, kernel sessions, SageMaker images, and instances aren't shut down. These resources continue to accrue charges.
  - **Shutdown All** – Shuts down all apps, terminal sessions, kernel sessions, SageMaker images, and instances. These resources no longer accrue charges.
4. Close the window.
5. After the app has been deleted, launch a new Studio Classic app to use the latest version.

## Shut down and Update Studio Classic Apps

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

To update an Amazon SageMaker Studio Classic app to the latest release, you must first shut down the corresponding KernelGateway app from the SageMaker AI console. After the KernelGateway app is shut down, you must reopen it through SageMaker Studio Classic by running a new kernel. The kernel automatically updates. Any unsaved notebook information is lost in the process. The user data in the Amazon EFS volume isn't impacted.

After an application has been shut down for 24 hours, SageMaker AI deletes all metadata for the application. To be considered an update and retain application metadata, applications must be restarted within 24 hours after the previous application has been shut down. After this time window, creation of an application is considered a new application rather than an update of the previous application.

### **Note**

A KernelGateway app is associated with a single Studio Classic user. When you update the app for one user it doesn't effect other users.

## To update the KernelGateway app

1. Navigate to <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. Select the domain that includes the application that you want to update.

5. Under **User profiles**, select your user name.
6. Under **Apps**, in the row displaying the **App name**, choose **Action**, then choose **Delete**  
To update Data Wrangler, delete the app that starts with **sagemaker-data-wrang**.
7. Choose **Yes, delete app**.
8. Type **delete** in the confirmation box.
9. Choose **Delete**.
10. After the app has been deleted, launch a new kernel from within Studio Classic to use the latest version.

## Amazon SageMaker Studio Classic Pricing

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

When the first member of your team onboards to Amazon SageMaker Studio Classic, Amazon SageMaker AI creates an Amazon Elastic File System (Amazon EFS) volume for the team. When this member, or any member of the team, opens Studio Classic, a home directory is created in the volume for the member. A storage charge is incurred for this directory. Subsequently, additional storage charges are incurred for the notebooks and data files stored in the member's home directory. For pricing information on Amazon EFS, see [Amazon EFS Pricing](#).

Additional costs are incurred when other operations are run inside Studio Classic, for example, running a notebook, running training jobs, and hosting a model.

For information on the costs associated with using Studio Classic notebooks, see [Usage Metering](#).

For information about billing along with pricing examples, see [Amazon SageMaker Pricing](#).

If Amazon SageMaker Studio is your default experience, see [Amazon SageMaker Studio pricing](#) for more pricing information.

# Troubleshooting Amazon SageMaker Studio Classic

## Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

## Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

This topic describes how to troubleshoot common Amazon SageMaker Studio Classic issues during setup and use. The following are common errors that might occur while using Amazon SageMaker Studio Classic. Each error is followed by its solution.

## Studio Classic application issues

The following issues occur when launching and using the Studio Classic application.

- **Screen not loading: Clearing workspace and waiting doesn't help**

When launching the Studio Classic application, a pop-up displays the following message. No matter which option is selected, Studio Classic does not load.

Loading...

The loading screen is taking a long time. Would you like to clear the workspace or keep waiting?

The Studio Classic application can have a launch delay if multiple tabs are open in the Studio Classic workspace or several files are on Amazon EFS. This pop-up should disappear in a few seconds after the Studio Classic workspace is ready.

If you continue to see a loading screen with a spinner after selecting either of the options, there could be connectivity issues with the Amazon Virtual Private Cloud used by Studio Classic.

To resolve connectivity issues with the Amazon Virtual Private Cloud (Amazon VPC) used by Studio Classic, verify the following networking configurations:

- If your domain is set up in `VpcOnly` mode: Verify that there is an Amazon VPC endpoint for AWS STS, or a NAT Gateway for outbound traffic, including traffic over the internet. To do this, follow the steps in [Connect Studio notebooks in a VPC to external resources](#).
- If your Amazon VPC is set up with a custom DNS instead of the DNS provided by Amazon: Verify that the routes are configured using Dynamic Host Configuration Protocol (DHCP) for each Amazon VPC endpoint added to the Amazon VPC used by Studio Classic. For more information about setting default and custom DHCP option sets, see [DHCP option sets in Amazon VPC](#).
- **Internal Failure when launching Studio Classic**

When launching Studio Classic, you are unable to view the Studio Classic UI. You also see an error similar to the following, with **Internal Failure** as the error detail.

Amazon SageMaker Studio  
The JupyterServer app default encountered a problem and was stopped.

This error can be caused by multiple factors. If completion of these steps does not resolve your issue, create an issue with <https://aws.amazon.com/premiumsupport/>.

- **Missing Amazon EFS mount target:** Studio Classic uses Amazon EFS for storage. The Amazon EFS volume needs a mount target for each subnet that the Amazon SageMaker AI domain is created in. If this Amazon EFS mount target is deleted accidentally, the Studio Classic application cannot load because it cannot mount the user's file directory. To resolve this issue, complete the following steps.

## To verify or create mount targets.

1. Find the Amazon EFS volume that is associated with the domain by using the [DescribeDomain](#) API call.
  2. Sign in to the AWS Management Console and open the Amazon EFS console at <https://console.aws.amazon.com/efs/>.
  3. From the list of Amazon EFS volumes, select the Amazon EFS volume that is associated with the domain.
  4. On the Amazon EFS details page, select the **Network** tab. Verify that there are mount targets for all of the subnets that the domain is set up in.
  5. If mount targets are missing, add the missing Amazon EFS mount targets. For instructions, see [Creating and managing mount targets and security groups](#).
  6. After the missing mount targets are created, launch the Studio Classic application.
- **Conflicting files in the user's .local folder:** If you're using JupyterLab version 1 on Studio Classic, conflicting libraries in your .local folder can cause issues when launching the Studio Classic application. To resolve this, update your user profile's default JupyterLab version to JupyterLab 3.0. For more information about viewing and updating the JupyterLab version, see [JupyterLab Versioning](#).
  - **ConfigurationError: LifecycleConfig when launching Studio Classic**

You can't view the Studio Classic UI when launching Studio Classic. This is caused by issues with the default lifecycle configuration script attached to the domain.

## To resolve lifecycle configuration issues

1. View the Amazon CloudWatch Logs for the lifecycle configuration to trace the command that caused the failure. To view the log, follow the steps in [Verify lifecycle configuration process from CloudWatch Logs](#).
  2. Detach the default script from the user profile or domain. For more information, see [Update and detach lifecycle configurations](#).
  3. Launch the Studio Classic application.
  4. Debug your lifecycle configuration script. You can run the lifecycle configuration script from the system terminal to troubleshoot. When the script runs successfully from the terminal, you can attach the script to the user profile or the domain.
- **SageMaker Studio Classic core functionalities are not available.**

If you get this error message when opening Studio Classic, it may be due to Python package version conflicts. This occurs if you used the following commands in a notebook or terminal to install Python packages that have version conflicts with SageMaker AI package dependencies.

```
!pip install
```

```
pip install --user
```

To resolve this issue, complete the following steps:

1. Uninstall recently installed Python packages. If you're not sure which package to uninstall, create an issue with <https://aws.amazon.com/premiumsupport/>.
2. Restart Studio Classic:
  - a. Shut down Studio Classic from the **File** menu.
  - b. Wait for one minute.
  - c. Reopen Studio Classic by refreshing the page or opening it from the AWS Management Console.

The problem should be resolved if you have uninstalled the package which caused the conflict. To install packages without causing this issue again, use `%pip install` without the `--user` flag.

If the issue persists, create a new user profile and set up your environment with that user profile.

If these solutions don't fix the issue, create an issue with <https://aws.amazon.com/premiumsupport/>.

- **Unable to open Studio Classic from the AWS Management Console.**

If you are unable to open Studio Classic and cannot make a new running instance with all default settings, create an issue with <https://aws.amazon.com/premiumsupport/>.

## KernelGateway application issues

The following issues are specific to KernelGateway applications that are launched in Studio Classic.

- **Cannot access the Kernel session**

When the user launches a new notebook, they are unable to connect to the notebook session. If the KernelGateway application's status is `In Service`, you can verify the following to resolve the issue.

- **Check Security Group configurations**

If the domain is set up in VPCOnly mode, the security group associated with the domain must allow traffic between the ports in the range 8192–65535 for connectivity between the JupyterServer and KernelGateway apps.

### To verify the security group rules

1. Get the security groups associated with the domain using the [DescribeDomain API call](#).
2. Sign in to the AWS Management Console and open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
3. From the left navigation, under **Security**, choose **Security Groups**.
4. Filter by the IDs of the security groups that are associated with the domain.
5. For each security group:
  - a. Select the security group.
  - b. From the security group details page, view the **Inbound rules**. Verify that traffic is allowed between ports in the range 8192–65535.

For more information about security group rules, see [Control traffic to resources using security groups](#). For more information about requirements to use Studio Classic in VPCOnly mode, see [Connect Studio notebooks in a VPC to external resources](#).

- **Verify firewall and WebSocket connections**

If the KernelGateway apps have an `InService` status and the user is unable to connect to the Studio Classic notebook session, verify the firewall and WebSocket settings.

1. Launch the Studio Classic application. For more information, see [Launch Amazon SageMaker Studio Classic](#).
2. Open your web browser's developer tools.
3. Choose the **Network** tab.
4. Search for an entry that matches the following format.

```
wss://<domain-id>.studio.<region>.sagemaker.aws/jupyter/default/api/kernels/<unique-code>/channels?session_id=<unique-code>
```

If the status or response code for the entry is anything other than 101, then your network settings are preventing the connection between the Studio Classic application and the KernelGateway apps.

To resolve this issue, contact the team that manages your networking settings to allow list the Studio Classic URL and enable WebSocket connections.

- **Unable to launch an app caused by exceeded resource quotas**

When a user tries to launch a new notebook, the notebook creation fails with either of the following errors. This is caused by exceeding resource quotas.

- Unable to start more Apps of AppType [KernelGateway] and ResourceSpec(instanceType=[]) for UserProfile []. Please delete an App with a matching AppType and ResourceSpec, then try again

Studio Classic supports up to four running KernelGateway apps on the same instance. To resolve this issue, you can do either of the following:

- Delete an existing KernelGateway application running on the instance, then restart the new notebook.
- Start the new notebook on a different instance type

For more information, see [Change an Instance Type](#).

- An error occurred (ResourceLimitExceeded) when calling the CreateApp operation

In this case, the account does not have sufficient limits to create a Studio Classic application on the specified instance type. To resolve this, navigate to the Service Quotas console at <https://console.aws.amazon.com/servicequotas/>. In that console, request to increase the Studio KernelGateway Apps running on *instance-type* instance limit. For more information, see [AWS service quotas](#).

# SageMaker JupyterLab

Create a JupyterLab space within Amazon SageMaker Studio to launch the JupyterLab application. A JupyterLab space is a private or shared space within Studio that manages the storage and compute resources needed to run the JupyterLab application. The JupyterLab application is a web-based interactive development environment (IDE) for notebooks, code, and data. Use the JupyterLab application's flexible and extensive interface to configure and arrange machine learning (ML) workflows.

By default, the JupyterLab application comes with the SageMaker Distribution image. The distribution image has popular packages, such as the following:

- PyTorch
- TensorFlow
- Keras
- NumPy
- Pandas
- Scikit-learn

You can use shared spaces to collaborate on your Jupyter notebooks with other users in real time. For more information about shared spaces, see [Collaboration with shared spaces](#).

Within the JupyterLab application, you can use Amazon Q Developer, a generative AI powered code companion to generate, debug, and explain your code. For information about using Amazon Q Developer, see [JupyterLab user guide](#). For information about setting up Amazon Q Developer, see [JupyterLab administrator guide](#).

Build unified analytics and ML workflows in same Jupyter notebook. Run interactive Spark jobs on Amazon EMR and AWS Glue serverless infrastructure, right from your notebook. Monitor and debug jobs faster using the inline Spark UI. In a few steps, you can automate your data prep by scheduling the notebook as a job.

The JupyterLab application helps you work collaboratively with your peers. Use the built-in Git integration within the JupyterLab IDE to share and version code. Bring your own file storage system if you have an Amazon EFS volume.

The JupyterLab application runs on a single Amazon Elastic Compute Cloud (Amazon EC2) instance and uses a single Amazon Elastic Block Store (Amazon EBS) volume for storage. You can switch faster instances or increase the Amazon EBS volume size for your needs.

The JupyterLab 4 application runs in a JupyterLab space within Studio. Studio Classic uses the JupyterLab 3 application. JupyterLab 4 provides the following benefits:

- A faster IDE than Amazon SageMaker Studio Classic, especially with large notebooks
- Improved document search
- A more performant and accessible text editor

For more information about JupyterLab, see [JupyterLab Documentation](#).

## Topics

- [JupyterLab user guide](#)
- [JupyterLab administrator guide](#)

## JupyterLab user guide

This guide shows JupyterLab users how to run analytics and machine learning workflows within SageMaker Studio. You can get fast storage and scale your compute up or down, depending on your needs.

JupyterLab supports both private and shared spaces. Private spaces are scoped to a single user in a domain. Shared spaces let other users in your domain collaborate with you in real time. For information about Studio spaces, see [Amazon SageMaker Studio spaces](#).

To get started using JupyterLab, create a space and launch your JupyterLab application. The space running your JupyterLab application is a JupyterLab space. The JupyterLab space uses a single Amazon EC2 instance for your compute and a single Amazon EBS volume for your storage. Everything in your space such as your code, git profile, and environment variables are stored on the same Amazon EBS volume. The volume has 3000 IOPS and a throughput of 125 megabytes per second (MBps). You can use the fast storage to open and run multiple Jupyter notebooks on the same instance. You can also switch kernels in a notebook very quickly.

Your administrator has configured the default Amazon EBS storage settings for your space. The default storage size is 5 GB, but you can increase the amount of space that you get. You can talk to your administrator to provide you with guidelines.

You can switch the Amazon EC2 instance type that you're using to run JupyterLab, scaling your compute up or down depending on your needs. The **Fast launch** instances start up much faster than the other instances.

Your administrator might provide you with a lifecycle configuration that customizes your environment. You can specify the lifecycle configuration when you create the space.

If your administrator gives you access to an Amazon EFS, you can configure your JupyterLab space to access it.

By default, the JupyterLab application uses the SageMaker distribution image. This includes support for many machine learning, analytics, and deep learning packages. However, if you need a custom image, your administrator can help provide access to the custom images.

The Amazon EBS volume persists independently from the life of an instance. You won't lose your data when you change instances. Use the conda and pip package management libraries to create reproducible custom environments that persist even when you switch instance types.

After you open JupyterLab, you can configure your environment using the terminal. To open the terminal, navigate to the **Launcher** and choose **Terminal**.

The following are examples of different ways that you can configure an environment in JupyterLab.

### Note

Within Studio, you can use lifecycle configurations to customize your environment, but we recommend using a package manager instead. Using lifecycle configurations is a more error-prone method. It's easier to add or remove dependencies than it is to debug a lifecycle configuration script. It can also increase the JupyterLab startup time.

For information about lifecycle configurations, see [Lifecycle configurations with JupyterLab](#).

## Topics

- [Create a space](#)
- [Configure a space](#)
- [Customize your environment using a package manager](#)
- [Clean up a conda environment](#)
- [Share conda environments between instance types](#)

- [Use Amazon Q to Expedite Your Machine Learning Workflows](#)

## Create a space

To get started using JupyterLab, create a space or choose the space that your administrator created for you and open JupyterLab.

Use the following procedure to create a space and open JupyterLab.

### To create a space and open JupyterLab

1. Open Studio. For information about opening Studio, see [Launch Amazon SageMaker Studio](#).
2. Choose **JupyterLab**.
3. Choose **Create JupyterLab space**.
4. For **Name**, specify the name of the space.
5. (Optional) Select **Share with my domain** to create a shared space.
6. Choose **Create space**.
7. (Optional) For **Instance**, specify the Amazon EC2 instance that runs the space.
8. (Optional) For **Image**, specify an image that your administrator provided to customize your environment.

#### Important

Custom IAM policies that allow Studio users to create spaces must also grant permissions to list images (`sagemaker: ListImage`) to view custom images. To add the permission, see [Add or remove identity permissions](#) in the *AWS Identity and Access Management User Guide*.

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker AI resources already include permissions to list images while creating those resources.

9. (Optional) For **Space Settings**, specify the following:

- **Storage (GB)** – Up to 100 GB or the amount that your administrator specifies.
- **Lifecycle Configuration** – A lifecycle configuration that your administrator specifies.
- **Attach custom EFS filesystem** – An Amazon EFS to which your administrator provides access.

10. Choose **Run space**.
11. Choose **Open JupyterLab**.

## Configure a space

After you create a JupyterLab space, you can configure it to do the following:

- Change the instance type.
- Change the storage volume.
- (Admin set up required) Use a custom image.
- (Admin set up required) Use a lifecycle configuration.
- (Admin set up required) Attach a custom Amazon EFS.

### **⚠️ Important**

You must stop the JupyterLab space every time you configure it. Use the following procedure to configure the space.

## To configure a space

1. Within Studio, navigate to the JupyterLab application page.
2. Choose the name of the space.
3. (Optional) For **Image**, specify an image that your administrator provided to customize your environment.

### **⚠️ Important**

Custom IAM policies that allow Studio users to create spaces must also grant permissions to list images (`sagemaker: ListImage`) to view custom images. To add the permission, see [Add or remove identity permissions](#) in the *AWS Identity and Access Management User Guide*.

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker AI resources already include permissions to list images while creating those resources.

4. (Optional) For **Space Settings**, specify the following:

- **Storage (GB)** – Up to 100 GB or the amount that your administrator configured for the space.
- **Lifecycle Configuration** – A lifecycle configuration that your administrator provides.
- **Attach custom EFS filesystem** – An Amazon EFS to which your administrator provides access.

5. Choose **Run space**.

When you open the JupyterLab application, your space has the updated configuration.

## Customize your environment using a package manager

Use pip or conda to customize your environment. We recommend using package managers instead of lifecycle configuration scripts.

### Create and activate your custom environment

This section provides examples of different ways that you can configure an environment in JupyterLab.

A basic conda environment has the minimum number of packages that are required for your workflows in SageMaker AI. Use the following template to create a basic conda environment:

```
# initialize conda for shell interaction
conda init

# create a new fresh environment
conda create --name test-env

# check if your new environment is created successfully
conda info --envs

# activate the new environment
conda activate test-env

# install packages in your new conda environment
conda install pip boto3 pandas ipykernel
```

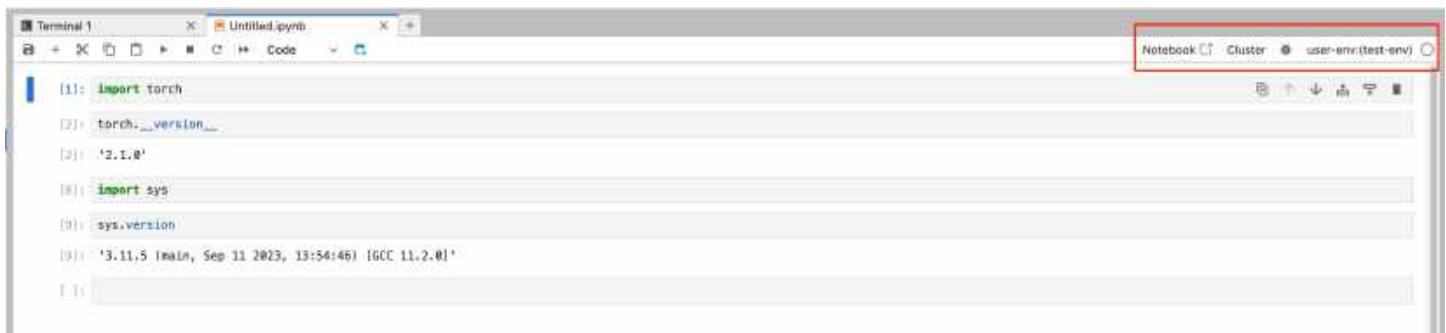
```
# list all packages install in your new environment
conda list

# parse env name information from your new environment
export CURRENT_ENV_NAME=$(conda info | grep "active environment" | cut -d : -f 2 | tr -
d ' ')

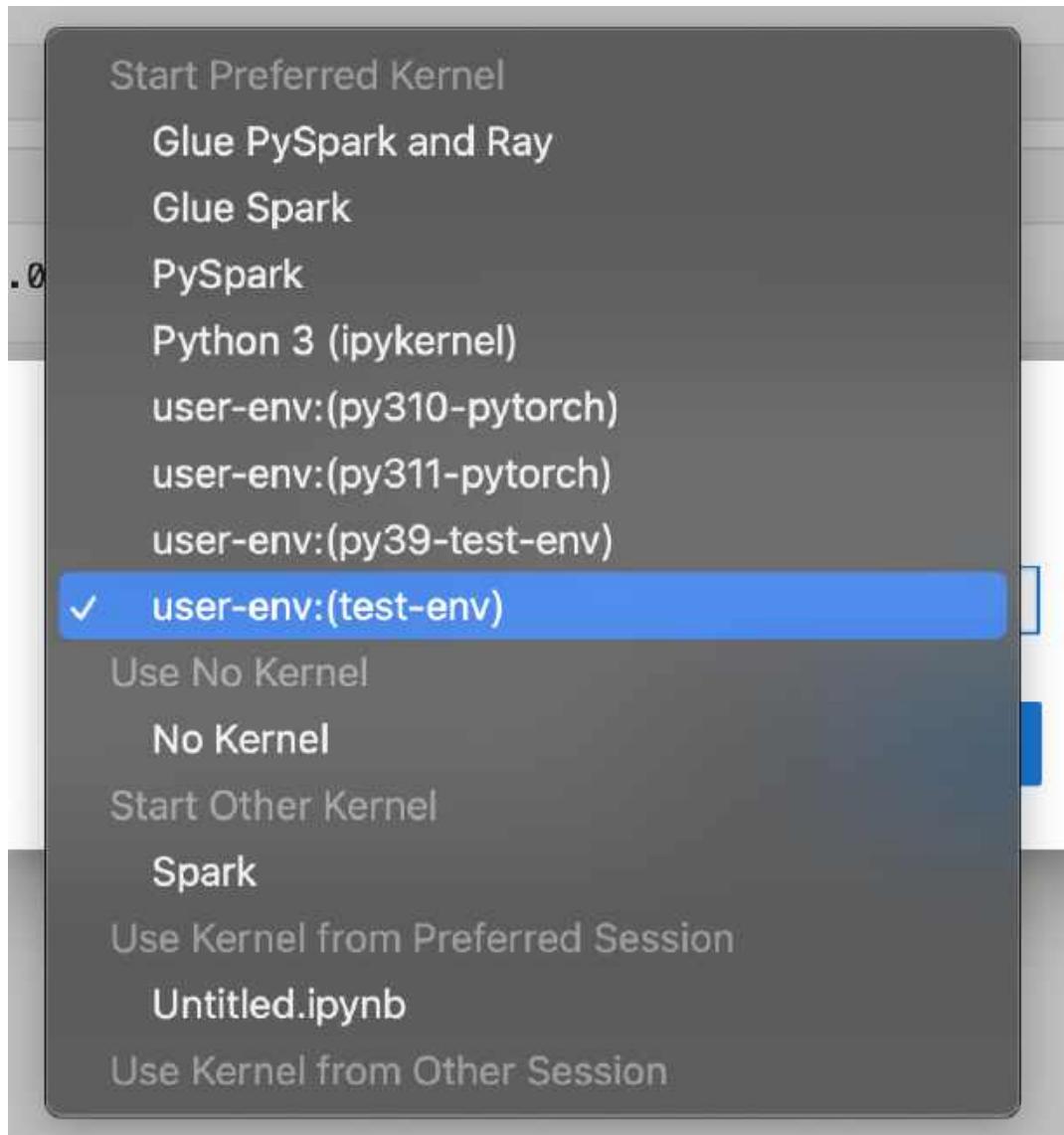
# register your new environment as Jupyter Kernel for execution
python3 -m ipykernel install --user --name $CURRENT_ENV_NAME --display-name "user-env:
($CURRENT_ENV_NAME)"

# to exit your new environment
conda deactivate
```

The following image shows the location of the environment that you've created.



To change your environment, choose it and select an option from the dropdown menu.



Choose **Select** to select a kernel for the environment.

### Create a conda environment with a specific Python version

Cleaning up conda environments that you're not using can help free up disk space and improve performance. Use the following template to clean up a conda environment:

```
# create a conda environment with a specific python version
conda create --name py38-test-env python=3.8.10

# activate and test your new python version
conda activate py38-test-env & python3 --version
```

```
# Install ipykernel to facilitate env registration
conda install ipykernel

# parse env name information from your new environment
export CURRENT_ENV_NAME=$(conda info | grep "active environment" | cut -d : -f 2 | tr -
d ' ')

# register your new environment as Jupyter Kernel for execution
python3 -m ipykernel install --user --name $CURRENT_ENV_NAME --display-name "user-env:
($CURRENT_ENV_NAME)"

# deactivate your py38 test environment
conda deactivate
```

## Create a conda environment with a specific set of packages

Use the following template to create a conda environment with a specific version of Python and set of packages:

```
# prefill your conda environment with a set of packages,
conda create --name py38-test-env python=3.8.10 pandas matplotlib=3.7 scipy ipykernel

# activate your conda environment and ensure these packages exist
conda activate py38-test-env

# check if these packages exist
conda list | grep -E 'pandas|matplotlib|scipy'

# parse env name information from your new environment
export CURRENT_ENV_NAME=$(conda info | grep "active environment" | cut -d : -f 2 | tr -
d ' ')

# register your new environment as Jupyter Kernel for execution
python3 -m ipykernel install --user --name $CURRENT_ENV_NAME --display-name "user-env:
($CURRENT_ENV_NAME)"

# deactivate your conda environment
conda deactivate
```

## Clone conda from an existing environment

Clone your conda environment to preserve its working state. You experiment in the cloned environment without having to worry about introducing breaking changes in your test environment.

Use the following command to clone an environment.

```
# create a fresh env from a base environment
conda create --name py310-base-ext --clone base # replace 'base' with another env

# activate your conda environment and ensure these packages exist
conda activate py310-base-ext

# install ipykernel to register your env
conda install ipykernel

# parse env name information from your new environment
export CURRENT_ENV_NAME=$(conda info | grep "active environment" | cut -d : -f 2 | tr -d ' ')

# register your new environment as Jupyter Kernel for execution
python3 -m ipykernel install --user --name $CURRENT_ENV_NAME --display-name "user-env:($CURRENT_ENV_NAME)"

# deactivate your conda environment
conda deactivate
```

## Clone conda from a reference YAML file

Create a conda environment from a reference YAML file. The following is an example of a YAML file that you can use.

```
# anatomy of a reference environment.yml
name: py311-new-env
channels:
  - conda-forge
dependencies:
  - python=3.11
  - numpy
```

```
- pandas
- scipy
- matplotlib
- pip
- ipykernel
- pip:
  - git+https://github.com/huggingface/transformers
```

Under pip, we recommend specifying only the dependencies that aren't available with conda.

Use the following commands to create a conda environment from a YAML file.

```
# create your conda environment
conda env create -f environment.yml

# activate your env
conda activate py311-new-env
```

## Clean up a conda environment

Cleaning up conda environments that you're not using can help free up disk space and improve performance. Use the following template to clean up a conda environment:

```
# list your environments to select an environment to clean
conda info --envs # or conda info -e

# once you've selected your environment to purge
conda remove --name test-env --all

# run conda environment list to ensure the target environment is purged
conda info --envs # or conda info -e
```

## Share conda environments between instance types

You can share conda environments by saving them to an Amazon EFS directory outside of your Amazon EBS volume. Another user can access the environment in the directory where you saved it.

## Important

There are limitations with sharing your environments. For example, we don't recommend an environment meant to run on a GPU Amazon EC2 instance over an environment running on a CPU instance.

Use the following commands as a template to specify the target directory where you're creating a custom environment. You're creating a conda within a particular path. You create it within the Amazon EFS directory. You can spin up a new instance and do conda activate path and do it within the Amazon EFS.

```
# if you know your environment path for your conda environment
conda create --prefix /home/sagemaker-user/my-project/py39-test python=3.9

# activate the env with full path from prefix
conda activate home/sagemaker-user/my-project/py39-test

# parse env name information from your new environment
export CURRENT_ENV_NAME=$(conda info | grep "active environment" | awk -F' : ' '{print $2}' | awk -F'/' '{print $NF}')

# register your new environment as Jupyter Kernel for execution
python3 -m ipykernel install --user --name $CURRENT_ENV_NAME --display-name "user-env-prefix:($CURRENT_ENV_NAME)"

# deactivate your conda environment
conda deactivate
```

## Use Amazon Q to Expedite Your Machine Learning Workflows

Amazon Q Developer is your AI-powered companion for machine learning development. With Amazon Q Developer, you can:

- Receive step-by-step guidance on using SageMaker AI features independently or in combination with other AWS services.
- Get sample code to get started on your ML tasks such as data preparation, training, inference, and MLOps.

- Receive troubleshooting assistance to debug and resolve errors encountered while running code.

Amazon Q Developer seamlessly integrates into your JupyterLab environment. To use Amazon Q Developer, choose the **Q** from the left-hand navigation of your JupyterLab environment or Code Editor environment.

If you don't see the **Q** icon, your administrator needs to set it up for you. For more information about setting up Amazon Q Developer, see [Set up Amazon Q Developer for your users](#).

Amazon Q automatically provides suggestions to help you write your code. You can also ask for suggestions through the chat interface.

## JupyterLab administrator guide

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

This guide for administrators describes SageMaker AI JupyterLab resources, such as those from Amazon Elastic Block Store (Amazon EBS) and Amazon Elastic Compute Cloud (Amazon EC2). The topics also show how to provide user access and change storage size.

A SageMaker AI JupyterLab space is composed of the following resources:

- A distinct Amazon EBS volume that stores all of the data, such as the code and the environment variables.
- The Amazon EC2 instance used to run the space.
- The image used to run JupyterLab.

**Note**

Applications do not have access to the EBS volume of other applications. For example, Code Editor, based on Code-OSS, Visual Studio Code - Open Source doesn't have access to the EBS volume for JupyterLab. For more information about EBS volumes, see [Amazon Elastic Block Store \(Amazon EBS\)](#).

You can use the Amazon SageMaker API to do the following:

- Change the default storage size of the EBS volume for your users.
- Change the maximum size of the EBS storage
- Specify the user settings for the application. For example, you can specify whether the user is using a custom image or a code repository.
- Specify the support application type.

The default size of the Amazon EBS volume is 5 GB. You can increase the volume size to a maximum of 16,384 GB. If you don't do anything, your users can increase their volume size to 100 GB. The volume size can be changed only once within a six hour period.

The kernels associated with the JupyterLab application run on the same Amazon EC2 instance that runs JupyterLab. When you create a space, the latest version of the SageMaker Distribution Image is used by default. For more information about SageMaker Distribution Images, see [SageMaker Studio image support policy](#).

**Important**

For information about updating the space to use the latest version of the SageMaker AI Distribution Image, see [Update the SageMaker Distribution Image](#).

The working directory of your users within the storage volume is `/home/sagemaker-user`. If you specify your own AWS KMS key to encrypt the volume, everything in the working directory is encrypted using your customer managed key. If you don't specify an AWS KMS key, the data inside `/home/sagemaker-user` is encrypted with an AWS managed key. Regardless of whether you specify an AWS KMS key, all of the data outside of the working directory is encrypted with an AWS Managed Key.

The following sections walk you through the configurations that you need to perform as an administrator.

## Topics

- [Give your users access to spaces](#)
- [Change the default storage size for your JupyterLab users](#)
- [Lifecycle configurations with JupyterLab](#)
- [Git repos in JupyterLab](#)
- [Customize environments using custom images](#)
- [Update the SageMaker Distribution Image](#)
- [Delete unused resources](#)
- [Quotas](#)

## Give your users access to spaces

To give users access to private or shared spaces, you must attach a permissions policy to their IAM roles. You can also use the permissions policy to restrict private spaces and their associated applications to a specific user profile.

The following permissions policy grants access to private and shared spaces. This allows users to create their own space and list other spaces within their domain. A user with this policy can't access the private space of a different user. For information about Studio spaces, see [Amazon SageMaker Studio spaces](#).

The policy provides users with permissions to the following:

- Private spaces or shared spaces.
- A user profile for accessing those spaces.

To provide permissions, you can scope down the permissions of the following policy and add it to the IAM roles of your users. You can also use this policy to restrict your spaces, and their associated applications, to a specific user profile.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {
```

```
{  
  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker>CreateApp",  
        "sagemaker>DeleteApp"  
    ],  
    "Resource": "arn:aws:sagemaker:{Region}:{AccountId}:app/*",  
    "Condition": {  
        "Null": {  
            "sagemaker:OwnerUserProfileArn": "true"  
        }  
    }  
},  
{  
    "Sid": "SMStudioCreatePresignedDomainUrlForUserProfile",  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker>CreatePresignedDomainUrl"  
    ],  
    "Resource": "arn:aws:sagemaker:{Region}:{AccountId}:user-profile/  
${sagemaker:DomainId}/${sagemaker:UserProfileName}"  
},  
,  
{  
    "Sid": "SMStudioAppPermissionsListAndDescribe",  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker>ListApps",  
        "sagemaker>ListDomains",  
        "sagemaker>ListUserProfiles",  
        "sagemaker>ListSpaces",  
        "sagemaker:DescribeApp",  
        "sagemaker:DescribeDomain",  
        "sagemaker:DescribeUserProfile",  
        "sagemaker:DescribeSpace"  
    ],  
    "Resource": "*"  
},  
{  
    "Sid": "SMStudioAppPermissionsTagOnCreate",  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker:AddTags"  
    ],  
}
```

```
"Resource": "arn:aws:sagemaker:{Region}:{AccountId}:*//*",
"Condition": {
    "Null": {
        "sagemaker:TaggingAction": "false"
    }
},
{
    "Sid": "SMStudioRestrictSharedSpacesWithoutOwners",
    "Effect": "Allow",
    "Action": [
        "sagemaker>CreateSpace",
        "sagemaker:UpdateSpace",
        "sagemaker>DeleteSpace"
    ],
    "Resource": "arn:aws:sagemaker:{Region}:{AccountId}:space/
${sagemaker:DomainId}/*",
    "Condition": {
        "Null": {
            "sagemaker:OwnerUserProfileArn": "true"
        }
    }
},
{
    "Sid": "SMStudioRestrictSpacesToOwnerUserProfile",
    "Effect": "Allow",
    "Action": [
        "sagemaker>CreateSpace",
        "sagemaker:UpdateSpace",
        "sagemaker>DeleteSpace"
    ],
    "Resource": "arn:aws:sagemaker:{Region}:{AccountId}:space/
${sagemaker:DomainId}/*",
    "Condition": {
        "ArnLike": {
            "sagemaker:OwnerUserProfileArn": "arn:aws:sagemaker:$AWS Region:
$111122223333:user-profile/${sagemaker:DomainId}/${sagemaker:UserProfileName}"
        },
        "StringEquals": {
            "sagemaker:SpaceSharingType": [
                "Private",
                "Shared"
            ]
        }
    }
}
```

```
    },
},
{
  "Sid": "SMStudioRestrictCreatePrivateSpaceAppsToOwnerUserProfile",
  "Effect": "Allow",
  "Action": [
    "sagemaker>CreateApp",
    "sagemaker>DeleteApp"
  ],
  "Resource": "arn:aws:sagemaker:{Region}:{AccountId}:app/${sagemaker:DomainId}/*",
  "Condition": {
    "ArnLike": {
      "sagemaker:OwnerUserProfileArn": "arn:aws:sagemaker:${aws:Region}:${aws:PrincipalAccount}:user-profile/${sagemaker:DomainId}/${sagemaker:UserProfileName}"
    },
    "StringEquals": {
      "sagemaker:SpaceSharingType": [
        "Private"
      ]
    }
  }
},
]
}
```

## Change the default storage size for your JupyterLab users

You can change the default storage settings for your users. You can also change the default storage settings based on your organizational requirements and the needs of your users.

To change the storage size, this section provides commands to do the following:

1. Update the Amazon EBS storage settings in the Amazon SageMaker AI domain (domain).
2. Create a user profile and specify the storage settings within it.

Use the following AWS Command Line Interface (AWS CLI) commands to change the default storage size.

Use the following AWS CLI command to update the domain:

```
aws --region AWS Region sagemaker update-domain \  
--domain-id domain-id \  
--default-user-settings '{  
    "SpaceStorageSettings": {  
        "DefaultEbsStorageSettings":{  
            "DefaultEbsVolumeSizeInGb":5,  
            "MaximumEbsVolumeSizeInGb":100  
        }  
    }  
}'
```

Use the following AWS CLI command to create the user profile and specify the default storage settings:

```
aws --region AWS Region sagemaker create-user-profile \  
--domain-id domain-id \  
--user-profile-name user-profile-name \  
--user-settings '{  
    "SpaceStorageSettings": {  
        "DefaultEbsStorageSettings":{  
            "DefaultEbsVolumeSizeInGb":5,  
            "MaximumEbsVolumeSizeInGb":100  
        }  
    }  
}'
```

Use the following AWS CLI commands to update the default storage settings in the user profile:

```
aws --region AWS Region sagemaker update-user-profile \  
--domain-id domain-id \  
--user-profile-name user-profile-name \  
--user-settings '{  
    "SpaceStorageSettings": {  
        "DefaultEbsStorageSettings":{  
            "DefaultEbsVolumeSizeInGb":25,  
            "MaximumEbsVolumeSizeInGb":200  
        }  
    }  
}'
```

```
    }  
}'
```

## Lifecycle configurations with JupyterLab

Lifecycle configurations are shell scripts that are triggered by JupyterLab lifecycle events, such as starting a new JupyterLab notebook. You can use lifecycle configurations to automate customization for your JupyterLab environment. This customization includes installing custom packages, configuring notebook extensions, preloading datasets, and setting up source code repositories.

Using lifecycle configurations gives you flexibility and control to configure JupyterLab to meet your specific needs. For example, you can create a minimal set of base container images with the most commonly used packages and libraries. Then you can use lifecycle configurations to install additional packages for specific use cases across your data science and machine learning teams.

 **Note**

Each script has a limit of **16,384 characters**.

### Topics

- [Lifecycle configuration creation](#)
- [Debug lifecycle configurations](#)
- [Detach lifecycle configurations](#)

### Lifecycle configuration creation

This topic includes instructions for creating and associating a lifecycle configuration with JupyterLab. You use the AWS Command Line Interface (AWS CLI) or the AWS Management Console to automate customization for your JupyterLab environment.

Lifecycle configurations are shell scripts triggered by JupyterLab lifecycle events, such as starting a new JupyterLab notebook. For more information about lifecycle configurations, see [Lifecycle configurations with JupyterLab](#).

## Create a lifecycle configuration (AWS CLI)

Learn how to create a lifecycle configuration using the AWS Command Line Interface (AWS CLI) to automate customization for your Studio environment.

### Prerequisites

Before you begin, complete the following prerequisites:

- Update the AWS CLI by following the steps in [Installing the current AWS CLI Version](#).
- From your local machine, run `aws configure` and provide your AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).
- Onboard to Amazon SageMaker AI domain. For conceptual information, see [Amazon SageMaker AI domain overview](#). For a quickstart guide, see [Use quick setup for Amazon SageMaker AI](#).

### Step 1: Create a lifecycle configuration

The following procedure shows how to create a lifecycle configuration script that prints Hello World.

#### Note

Each script can have up to **16,384 characters**.

1. From your local machine, create a file named `my-script.sh` with the following content:

```
#!/bin/bash
set -eux
echo 'Hello World!'
```

2. Use the following to convert your `my-script.sh` file into base64 format. This requirement prevents errors that occur from spacing and line break encoding.

```
LCC_CONTENT=`openssl base64 -A -in my-script.sh`
```

3. Create a lifecycle configuration for use with Studio. The following command creates a lifecycle configuration that runs when you launch an associated JupyterLab application:

```
aws sagemaker create-studio-lifecycle-config \
```

```
--region region \
--studio-lifecycle-config-name my-jl-lcc \
--studio-lifecycle-config-content $LCC_CONTENT \
--studio-lifecycle-config-app-type JupyterLab
```

Note the ARN of the newly created lifecycle configuration that is returned. This ARN is required to attach the lifecycle configuration to your application.

## Step 2: Attach the lifecycle configuration to your Amazon SageMaker AI domain (domain) and user profile

To attach the lifecycle configuration, you must update the UserSettings for your domain or user profile. Lifecycle configuration scripts that are associated at the domain level are inherited by all users. However, scripts that are associated at the user profile level are scoped to a specific user.

You can create a new user profile, domain, or space with a lifecycle configuration attached by using the following commands:

- [create-user-profile](#)
- [create-domain](#)
- [create-space](#)

The following command creates a user profile with a lifecycle configuration. Add the lifecycle configuration ARN from the preceding step to the JupyterLabAppSettings of the user. You can add multiple lifecycle configurations at the same time by passing a list of them. When a user launches a JupyterLab application with the AWS CLI, they can specify a lifecycle configuration instead of using the default one. The lifecycle configuration that the user passes must belong to the list of lifecycle configurations in JupyterLabAppSettings.

```
# Create a new UserProfile
aws sagemaker create-user-profile --domain-id domain-id \
--user-profile-name user-profile-name \
--region region \
--user-settings '{
  "JupyterLabAppSettings": {
    "LifecycleConfigArns": [
      [lifecycle-configuration-arn-list]
    ]
}
```

}'

## Create a lifecycle configuration (Console)

Learn how to create a lifecycle configuration using the AWS Management Console to automate customization for your Studio environment.

### Step 1: Create a lifecycle configuration

Use the following procedure to create a lifecycle configuration script that prints Hello World.

#### To create a lifecycle configuration

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **Lifecycle configurations**.
4. Choose the **JupyterLab** tab.
5. Choose **Create configuration**.
6. For **Name**, specify the name of the lifecycle configuration.
7. For the text box under **Scripts**, specify the following lifecycle configuration:

```
#!/bin/bash
set -eux
echo 'Hello World!'
```

8. Choose **Create configuration**.

### Step 2: Attach the lifecycle configuration to your Amazon SageMaker AI domain (domain) and user profile

Lifecycle configuration scripts associated at the domain level are inherited by all users. However, scripts that are associated at the user profile level are scoped to a specific user.

You can attach multiple lifecycle configurations to a domain or user profile for JupyterLab.

Use the following procedure to attach a lifecycle configuration to a domain.

## To attach a lifecycle configuration to a domain

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain to attach the lifecycle configuration to.
5. From the **Domain details**, choose the **Environment** tab.
6. Under **Lifecycle configurations for personal Studio apps**, choose **Attach**.
7. Under **Source**, choose **Existing configuration**.
8. Under **Studio lifecycle configurations**, select the lifecycle configuration that you created in the previous step.
9. Select **Attach to domain**.

Use the following procedure to attach a lifecycle configuration to a user profile.

## To attach a lifecycle configuration to a user profile

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that contains the user profile to attach the lifecycle configuration to.
5. Under **User profiles**, select the user profile.
6. From the **User Details** page, choose **Edit**.
7. On the left navigation, choose **Studio settings**.
8. Under **Lifecycle configurations attached to user**, choose **Attach**.
9. Under **Source**, choose **Existing configuration**.
10. Under **Studio lifecycle configurations**, select the lifecycle configuration that you created in the previous step.
11. Choose **Attach to user profile**.

## Debug lifecycle configurations

The following topics show how to get information about and debug your lifecycle configurations.

## Topics

- [Verify lifecycle configuration process from CloudWatch Logs](#)
- [Lifecycle configuration timeout](#)

### Verify lifecycle configuration process from CloudWatch Logs

Lifecycle configurations only log STDOUT and STDERR.

STDOUT is the default output for bash scripts. You can write to STDERR by appending >&2 to the end of a bash command. For example, echo 'hello'>&2.

Logs for your lifecycle configurations are published to your AWS account using Amazon CloudWatch. These logs can be found in the /aws/sagemaker/studio log stream in the CloudWatch console.

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. Choose **Logs** from the left navigation pane. From the dropdown menu, select **Log groups**.
3. On the **Log groups** page, search for aws/sagemaker/studio.
4. Select the log group.
5. On the **Log group details** page, choose the **Log streams** tab.
6. To find the logs for a specific app, search the log streams using the following format:

*domain-id/user-profile-name/app-type/app-name*

The following search string finds the lifecycle configuration logs for the domain d-m851cu8vbqmz, user profile i-sonic-js, application type JupyterLab, and application name test-lcc-echo:

d-m851cu8vbqmz/i-sonic-js/JupyterLab/test-lcc-echo

7. To view the script execution logs, select the log stream appended with LifecycleConfigOnStart.

### Lifecycle configuration timeout

There is a lifecycle configuration timeout limitation of 5 minutes. If a lifecycle configuration script takes longer than 5 minutes to run, you get an error.

To resolve this error, make sure that your lifecycle configuration script completes in less than 5 minutes.

To help decrease the runtime of scripts, try the following:

- Reduce unnecessary steps. For example, limit which conda environments to install large packages in.
- Run tasks in parallel processes.
- Use the nohup command in your script to make sure that hangup signals are ignored so that the script runs without stopping.

## Detach lifecycle configurations

To update your script, you must create a new lifecycle configuration script and attach it to the respective Amazon SageMaker AI domain (domain), user profile, or shared space. A lifecycle configuration script can't be changed after it's created. For more information about creating and attaching the lifecycle configuration, see [Lifecycle configuration creation](#).

The following section shows how to detach a lifecycle configuration using the AWS Command Line Interface (AWS CLI).

### Detach using the AWS CLI

To detach a lifecycle configuration using the (AWS CLI), remove the desired lifecycle configuration from the list of lifecycle configurations attached to the resource. You then pass the list as part of the respective command:

- [update-user-profile](#)
- [update-domain](#)
- [update-space](#)

For example, the following command removes all lifecycle configurations for the JupyterLab application that's attached to the domain.

```
aws sagemaker update-domain --domain-id domain-id \  
--region region \  
--default-user-settings '{  
"JupyterLabAppSettings": {  
"LifecycleConfigArns":
```

```
[  
]  
}  
}'
```

## Git repos in JupyterLab

JupyterLab offers a Git extension to enter the URL of a Git repository (repo), clone it into an environment, push changes, and view the commit history. You can also attach suggested Git repo URLs to a Amazon SageMaker AI domain (domain) or user profile.

The following sections show how to attach or detach Git repo URLs.

### Topics

- [Attach a Git repository \(AWS CLI\)](#)
- [Detach Git repo URLs](#)

### Attach a Git repository (AWS CLI)

This section shows how to attach a Git repository (repo) URL using the AWS CLI. After you attach the Git repo URL, you can clone it by following the steps in [Clone a Git repo in Amazon SageMaker Studio](#).

### Prerequisites

Before you begin, complete the following prerequisites:

- Update the AWS CLI by following the steps in [Installing the current AWS Command Line Interface Version](#).
- From your local machine, run `aws configure` and provide your AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).
- Onboard to Amazon SageMaker AI domain. For more information, see [Amazon SageMaker AI domain overview](#).

### Attach the Git repo to a Amazon SageMaker AI domain (domain) or user profile

Git repo URLs that are associated at the domain level are inherited by all users. However, Git repo URLs that are associated at the user profile level are scoped to a specific user. You can attach multiple Git repo URLs to a Amazon SageMaker AI domain or to a user profile by passing a list of repository URLs.

The following sections show how to attach a Git repo URL to your domain and your user profile.

## Attach to a Amazon SageMaker AI domain

The following command attaches a Git repo URL to an existing domain:

```
aws sagemaker update-domain --region region --domain-id domain-id \  
  --default-user-settings  
  JupyterLabAppSettings={CodeRepositories=[{RepositoryUrl="repository"}]}
```

## Attach to a user profile

The following command attaches a Git repo URL to an existing user profile:

```
aws sagemaker update-user-profile --domain-id domain-id --user-profile-name user-name \  
  --user-settings  
  JupyterLabAppSettings={CodeRepositories=[{RepositoryUrl="repository"}]}
```

## Clone a Git repo in Amazon SageMaker Studio

Amazon SageMaker Studio connects to a local Git repo only. To access the files in the repo, clone the Git repo from within Studio. To do so, Studio offers a Git extension for you to enter the URL of a Git repo, clone it into your environment, push changes, and view commit history.

If the repo is private and requires credentials to access, you receive a prompt to enter your user credentials. Your credentials include your username and personal access token. For more information about personal access tokens, see [Managing your personal access tokens](#).

Admins can also attach suggested Git repository URLs at the Amazon SageMaker AI domain or user profile level. Users can then select the repo URL from the list of suggestions and clone that into Studio. For more information about attaching suggested repos, see [Attach Suggested Git Repos to Studio Classic](#).

## Detach Git repo URLs

This section shows how to detach Git repository URLs from an Amazon SageMaker AI domain (domain) or a user profile. You can detach repo URLs by using the AWS Command Line Interface (AWS CLI) or the Amazon SageMaker AI console.

## Detach a Git repo using the AWS CLI

To detach all Git repo URLs from a domain or user profile, you must pass an empty list of code repositories. This list is passed as part of the `JupyterLabAppSettings` parameter in an `update-domain` or `update-user-profile` command. To detach only one Git repo URL, pass the code repositories list without the desired Git repo URL.

### Detach from an Amazon SageMaker AI domain

The following command detaches all Git repo URLs from a domain:

```
aws sagemaker update-domain --region region --domain-name domain-name \  
  --domain-settings JupyterLabAppSettings={CodeRepositories=[]}
```

### Detach from a user profile

The following command detaches all Git repo URLs from a user profile:

```
aws sagemaker update-user-profile --domain-name domain-name --user-profile-name user-  
name\ \  
  --user-settings JupyterLabAppSettings={CodeRepositories=[]}
```

## Customize environments using custom images

If you need functionality that is different than what's provided by SageMaker distribution, you can bring your own image with your custom extensions and packages. You can also use it to personalize the JupyterLab UI for your own branding or compliance needs.

For a tutorial that helps you create an image that your users can run in their JupyterLab environment, see [Provide users with access to custom images](#).

For requirements for your image, see [Dockerfile specifications](#).

### Topics

- [Provide users with access to custom images](#)
- [Dockerfile specifications](#)

## Provide users with access to custom images

This documentation provides step-by-step instructions to provide your users with access to custom images within their JupyterLab environments. You can use the information on this page to create custom environments for your user's workflows. The process involves utilizing:

- Docker
- AWS Command Line Interface
- Amazon Elastic Container Registry
- Amazon SageMaker AI AWS Management Console

After following the guidance on this page, JupyterLab users on the Amazon SageMaker AI domain will have access to the custom image and environment from their Jupyter spaces to empower their machine learning workflows.

### **Important**

This page assumes that you have the AWS Command Line Interface and Docker installed on your local machine.

To have your users successfully run their image within JupyterLab, you must do the following:

### **To have your users successfully run the image**

1. Create the Dockerfile
2. Build the image from the Dockerfile
3. Upload the image to Amazon Elastic Container Registry
4. Attach the image to your Amazon SageMaker AI domain
5. Have your users access the image from your JupyterLab space

#### **Step 1: Create the Dockerfile**

Create a Dockerfile to define the steps needed to create the environment needed to run the application in your users' containers.

**⚠️ Important**

Your Dockerfile must meet the specifications provided in [Dockerfile specifications](#).

For Dockerfile templates, see [Health check and URL for applications](#).

## Step 2: Build the image

In the same directory as your Dockerfile, build your image using the following command:

```
docker build -t username/imagename:tag your-account-id.dkr.ecr.AWS  
Region.amazonaws.com/your-repository-name:tag
```

**⚠️ Important**

Your image must be tagged in the following format: *123456789012*.dkr.ecr.your-region.amazonaws.com/*your-repository-name*:*tag*

You won't be able to push it to an Amazon Elastic Container Registry repository otherwise.

## Step 3: Push the image to the Amazon Elastic Container Registry repository

After you've built your image, log in to your Amazon ECR repository using the following command:

```
aws ecr get-login-password --region AWS Region | docker login --username AWS --  
password-stdin 123456789012.dkr.ecr.AWS Region.amazonaws.com
```

After you've logged in, push your Dockerfile using the following command:

```
docker push 123456789012.dkr.ecr.AWS Region.amazonaws.com/your-repository-name:tag
```

## Step 4: Attach image to the Amazon SageMaker AI domain of your users

### **⚠️ Important**

Custom IAM policies that allow Studio users to create spaces must also grant permissions to list images (`sagemaker: ListImage`) to view custom images. To add the permission, see [Add or remove identity permissions](#) in the *AWS Identity and Access Management User Guide*.

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker AI resources already include permissions to list images while creating those resources.

After you've pushed the image, you must access it from your Amazon SageMaker AI domain. Use the following procedure to attach the image to a SageMaker AI domain:

### Attach the image using the SageMaker AI console

1. Open the [SageMaker AI console](#).
2. Under **Admin configurations**, choose **domains**.
3. From the list of **domains**, select a domain.
4. Open the **Environment** tab.
5. For **Custom images for personal Studio apps**, choose **Attach image**.
6. Specify the image source.
7. Choose **Next**.
8. Choose **Submit**.

### Attach the image using the AWS CLI

Use the following procedure to attach the image to a SageMaker domain through the AWS CLI :

1. Create a SageMaker image. The `AmazonSageMakerFullAccess` policy must be attached to your role as you use the following AWS CLI commands.

```
aws sagemaker create-image \
--image-name custom-image \
```

```
--role-arn arn:aws:iam::account-id:role/service-role/execution-role
```

2. Create a SageMaker image version from the image. Pass the unique tag value that you chose when you pushed the image to Amazon ECR.

```
aws sagemaker create-image-version \  
  --image-name custom-image \  
  --base-image repository-uri:tag
```

3. Create a configuration file called `app-image-config-input.json`. The application image configuration is used as configuration for running a SageMaker image as a Code Editor application. You may also specify your [ContainerConfig](#) arguments here.

```
{  
    "AppImageConfigName": "app-image-config",  
    "CodeEditorAppImageConfig":  
    {  
        "ContainerConfig":  
        {}  
    }  
}
```

4. Create the AppImageConfig using the application image configuration file that you created.

```
aws sagemaker create-app-image-config \  
  --cli-input-json file://app-image-config-input.json
```

5. Create a configuration file, named `updateDomain.json`. Be sure to specify your domain ID.

```
{  
    "DomainId": "domain-id",  
    "DefaultUserSettings": {  
        "JupyterLabAppSettings": {  
            "CustomImages": [  
                {  
                    "ImageName": "custom-image",  
                    "AppImageConfigName": "app-image-config"  
                }  
            ]  
        }  
    }  
}
```

## 6. Call the UpdateDomain command with the configuration file as input.

### Note

You must delete all of the applications in your domain before updating the domain with the new image. Note that you only need to delete applications; you **do not** need to delete user profiles or shared spaces. For instructions on deleting applications, choose one of the following options.

- If you use the SageMaker AI console, run through Step 1 to 5d and Step 6 to 7d of the [Delete a domain \(Console\)](#) section.
- If you use the AWS CLI, run through Step 1 to 3 of the [Delete a domain \(AWS CLI\)](#) section.

```
aws sagemaker update-domain --cli-input-json file://updateDomain.json
```

Your users can now select the image that you've attached to their Domain from their JupyterLab space.

## Dockerfile specifications

The image that you specify in your Dockerfile must match the specifications in the following sections to create the image successfully.

## Running the image

- **Entrypoint** – We recommend embedding the entry point into the image using the Docker CMD or Entrypoint instructions. You can also configure ContainerEntrypoint and ContainerArguments that are passed to the container at runtime.
- **EnvVariables** – With Studio, you can configure ContainerEnvironment variables that are made available to a container. The environment variable is overwritten with the environment variables from SageMaker AI. To provide you with a better experience, the environment variables are usually AWS\_ and SageMaker AI\_namespaced to give priority to platform environments.

The following are the environment variables:

- AWS\_REGION

- AWS\_DEFAULT\_REGION
- AWS\_CONTAINER\_CREDENTIALS\_RELATIVE\_URI
- SageMaker\_AI\_SPACE\_NAME

## Specifications for the user and file system

- WorkingDirectory – The Amazon EBS volume for your space is mounted on the path /home/sagemaker-user. You can't change the mount path. Use the WORKDIR instruction to set the working directory of your image to a folder within /home/sagemaker-user.
- UID – The user ID of the Docker container. UID=1000 is a supported value. You can add sudo access to your users. The IDs are remapped to prevent a process running in the container from having more privileges than necessary.
- GID – The group ID of the Docker container. GID=100 is a supported value. You can add sudo access to your users. The IDs are remapped to prevent a process running in the container from having more privileges than necessary.
- Metadata directories – The /opt/.sagemakerintenal and /opt/ml directories that are used by AWS. The metadata file in /opt/ml contains metadata about resources such as DomainId.

Use the following command to show the file system contents:

```
cat /opt/ml/metadata/resource-metadata.json
>{"AppType":"JupyterLab","DomainId":"example-domain-id","UserProfileName":"example-
user-profile-name","ResourceArn":"arn:aws:sagemaker:AWS
Region:111122223333;:app/domain-ID/user-ID/Jupyter
Lab/default","ResourceName":"default","AppImageVersion":"current"}
```

- Logging directories – /var/log/studio are reserved for the logging directories of JupyterLab and the extensions associated with it. We recommend that you don't use the folders in creating your image.

## Health check and URL for applications

- Base URL – The base URL for the BYOI application must be jupyterlab/default. You can only have one application and it must always be named default.

- **HealthCheck API** – The HostAgent uses the HealthCheckAPI at port 8888 to check the health of the JupyterLab application. `jupyterlab/default/api/status` is the endpoint for the health check.
- **Home/Default URL** – The `/opt/.sagemakerinternal` and `/opt/ml` directories that are used by AWS. The metadata file in `/opt/ml` contains metadata about resources such as DomainId.
- **Authentication** – To enable authentication for your users, turn off the Jupyter notebooks token or password based authentication and allow all origins.

The following is a sample Amazon Linux 2 Dockerfile that meets the preceding specifications:

```
FROM public.ecr.aws/amazonlinux/amazonlinux:2023

ARG NB_USER="sagemaker-user"
ARG NB_UID="1000"
ARG NB_GID="100"

# Install Python3, pip, and other dependencies
RUN yum install -y \
    python3 \
    python3-pip \
    python3-devel \
    gcc \
    shadow-utils && \
    useradd --create-home --shell /bin/bash --gid "${NB_GID}" --uid ${NB_UID} \
${NB_USER} && \
    yum clean all

RUN python3 -m pip install --no-cache-dir \
    'jupyterlab>=4.0.0,<5.0.0' \
    urllib3 \
    jupyter-activity-monitor-extension \
    --ignore-installed

# Verify versions
RUN python3 --version && \
    jupyter lab --version

USER ${NB_UID}
CMD jupyter lab --ip 0.0.0.0 --port 8888 \
    --ServerApp.base_url="/jupyterlab/default" \
```

```
--ServerApp.token=''\ \
--ServerApp.allow_origin='*'
```

The following is a sample Amazon SageMaker Distribution Dockerfile that meets the preceding specifications:

```
FROM public.ecr.aws/sagemaker/sagemaker-distribution:latest-cpu
ARG NB_USER="sagemaker-user"
ARG NB_UID=1000
ARG NB_GID=100

ENV MAMBA_USER=$NB_USER

USER root

RUN apt-get update
RUN micromamba install sagemaker-inference --freeze-installed --yes --channel conda-forge --name base

USER $MAMBA_USER

ENTRYPOINT ["jupyter-lab"]
CMD ["--ServerApp.ip=0.0.0.0", "--ServerApp.port=8888", "--ServerApp.allow_origin=*",
"--ServerApp.token=''", "--ServerApp.base_url=/jupyterlab/default"]
```

## Update the SageMaker Distribution Image

### **Important**

This topic assumes that you've created a space and given the user access to it. For more information, see [Give your users access to spaces](#).

Update the JupyterLab spaces that you've already created to use the latest version of the SageMaker Distribution Image to access the latest features. You can use either the Studio UI or the AWS Command Line Interface (AWS CLI) to update the image.

The following sections provide information about updating an image.

## Update the image (UI)

Updating the image involves restarting the JupyterLab space of your user. Use the following procedure to update your user's JupyterLab space with the latest image.

### To update the image (UI)

1. Open Studio. For information about opening Studio, see [Launch Amazon SageMaker Studio](#).
2. Choose **JupyterLab**.
3. Select the JupyterLab space of your user.
4. Choose **Stop space**.
5. For **Image**, select an updated version of the SageMaker AI Distribution Image. For the latest image, choose **Latest**.
6. Choose **Run space**.

## Update the image (AWS CLI)

This section assumes that you have the AWS Command Line Interface (AWS CLI) installed. For information about installing the AWS CLI, see [Install or update to the latest version of the AWS CLI](#).

To update the image, you must do the following for your user's space:

1. Delete the JupyterLab application
2. Update the space
3. Create the application

### **⚠️ Important**

You must have the following information ready before you start updating the image:

- domain ID – The ID of your user's Amazon SageMaker AI domain.
- Application type – JupyterLab.
- Application name – default.
- Space name – The name specified for the space.
- Instance type – The Amazon EC2 instance type that you're using to run the application. For example, `m1.t3.medium`.

- SageMaker Image ARN – The Amazon Resource Name (ARN) of the SageMaker AI Distribution Image. You can provide the latest version of the SageMaker AI Distribution Image by specifying either `sagemaker-distribution-cpu` or `sagemaker-distribution-gpu` as the resource identifier.

To delete the JupyterLab application, run the following command:

```
aws sagemaker delete-app \
--domain-id your-user's-domain-id
--app-type JupyterLab \
--app-name default \
--space-name name-of-your-user's-space
```

To update your user's space, run the following command:

```
aws sagemaker update-space \
--space-name name-of-your-user's-space \
--domain-id your-user's-domain-id
```

If you've updated the space successfully, you'll see the space ARN in the response:

```
{
"SpaceArn": "arn:aws:sagemaker:AWS Region:111122223333:space/your-user's-domain-id/
name-of-your-user's-space"}
```

To create the application, run the following command:

```
aws sagemaker create-app \
--domain-id your-user's-domain-id \
--app-type JupyterLab \
```

```
--app-name default \
--space-name name-of-your-user's-space \
--resource-spec "InstanceType=instance-type,SageMakerImageArn=arn:aws:sagemaker:AWS Region:555555555555:image/sagemaker-distribution-resource-identifier"
```

## Delete unused resources

To avoid incurring additional costs running JupyterLab, we recommend deleting unused resources in the following order:

1. JupyterLab applications
2. Spaces
3. User profiles
4. domains

Use the following AWS Command Line Interface (AWS CLI) commands to delete resources within a domain:

### Delete a JupyterLab application

```
aws --region AWS Region sagemaker delete-app --domain-id example-domain-id --app-name default --app-type JupyterLab --space-name example-space-name
```

### Delete a space

#### **⚠ Important**

If you delete a space, you delete the Amazon EBS volume associated with it. We recommend backing up any valuable data before you delete your space.

```
aws --region AWS Region sagemaker delete-space --domain-id example-domain-id --space-name example-space-name
```

## Delete a user profile

```
aws --region AWS Region sagemaker delete-user-profile --domain-id example-domain-id  
--user-profile example-user-profile
```

## Quotas

JupyterLab, has quotas for the following:

- The sum of all Amazon EBS volumes within an AWS account.
- The instance types that are available for your users.
- The number of instances for a particular that your users can launch.

To get more storage and compute for your users, request an increase to your AWS quotas. For more information about requesting a quota increase, see [Amazon SageMaker AI endpoints and quotas](#).

## Amazon SageMaker Notebook Instances

An Amazon SageMaker notebook instance is a machine learning (ML) compute instance running the Jupyter Notebook application. One of the best ways for machine learning (ML) practitioners to use Amazon SageMaker AI is to train and deploy ML models using SageMaker notebook instances. The SageMaker notebook instances help create the environment by initiating Jupyter servers on Amazon Elastic Compute Cloud (Amazon EC2) and providing preconfigured kernels with the following packages: the Amazon SageMaker Python SDK, AWS SDK for Python (Boto3), AWS Command Line Interface (AWS CLI), Conda, Pandas, deep learning framework libraries, and other libraries for data science and machine learning.

Use Jupyter notebooks in your notebook instance to:

- prepare and process data
- write code to train models
- deploy models to SageMaker hosting
- test or validate your models

SageMaker AI also provides sample notebooks that contain complete code examples. These examples show how to use SageMaker AI to do common ML tasks. For more information, see [Access example notebooks](#).

For information about pricing with Amazon SageMaker notebook instance, see [Amazon SageMaker Pricing](#).

## Maintenance

SageMaker AI updates the underlying software for Amazon SageMaker Notebook Instances at least once every 90 days. Some maintenance updates, such as operating system upgrades, may require your application to be taken offline for a short period of time. It is not possible to perform any operations during this period while the underlying software is being updated. We recommend that you restart your notebooks at least once every 30 days to automatically consume patches.

For more information, contact [AWS Support](#).

## Machine Learning with the SageMaker Python SDK

To train, validate, deploy, and evaluate an ML model in a SageMaker notebook instance, use the SageMaker Python SDK. The SageMaker Python SDK abstracts AWS SDK for Python (Boto3) and SageMaker API operations. It enables you to integrate with and orchestrate other AWS services, such as Amazon Simple Storage Service (Amazon S3) for saving data and model artifacts, Amazon Elastic Container Registry (ECR) for importing and servicing the ML models, Amazon Elastic Compute Cloud (Amazon EC2) for training and inference.

You can also take advantage of SageMaker AI features that help you deal with every stage of a complete ML cycle: data labeling, data preprocessing, model training, model deployment, evaluation on prediction performance, and monitoring the quality of model in production.

If you're a first-time SageMaker AI user, we recommend you to use the SageMaker Python SDK, following the end-to-end ML tutorial. To find the open source documentation, see the [Amazon SageMaker Python SDK](#).

## Topics

- [Tutorial for building models with Notebook Instances](#)
- [Amazon Linux 2 notebook instances](#)
- [JupyterLab versioning](#)

- [Create an Amazon SageMaker notebook instance](#)
- [Access Notebook Instances](#)
- [Update a Notebook Instance](#)
- [Customization of a SageMaker notebook instance using an LCC script](#)
- [Access example notebooks](#)
- [Set the Notebook Kernel](#)
- [Git repositories with SageMaker AI Notebook Instances](#)
- [Notebook Instance Metadata](#)
- [Monitor Jupyter Logs in Amazon CloudWatch Logs](#)

## Tutorial for building models with Notebook Instances

This Get Started tutorial walks you through how to create a SageMaker notebook instance, open a Jupyter notebook with a preconfigured kernel with the Conda environment for machine learning, and start a SageMaker AI session to run an end-to-end ML cycle. You'll learn how to save a dataset to a default Amazon S3 bucket automatically paired with the SageMaker AI session, submit a training job of an ML model to Amazon EC2, and deploy the trained model for prediction by hosting or batch inferencing through Amazon EC2.

This tutorial explicitly shows a complete ML flow of training the XGBoost model from the SageMaker AI built-in model pool. You use the [US Adult Census dataset](#), and you evaluate the performance of the trained SageMaker AI XGBoost model on predicting individuals' income.

- [SageMaker AI XGBoost](#) – The [XGBoost](#) model is adapted to the SageMaker AI environment and preconfigured as Docker containers. SageMaker AI provides a suite of [built-in algorithms](#) that are prepared for using SageMaker AI features. To learn more about what ML algorithms are adapted to SageMaker AI, see [Choose an Algorithm](#) and [Use Amazon SageMaker Built-in Algorithms](#). For the SageMaker AI built-in algorithm API operations, see [First-Party Algorithms](#) in the [Amazon SageMaker Python SDK](#).
- [Adult Census dataset](#) – The dataset from the [1994 Census bureau database](#) by Ronny Kohavi and Barry Becker (Data Mining and Visualization, Silicon Graphics). The SageMaker AI XGBoost model is trained using this dataset to predict if an individual makes over \$50,000 a year or less.

### Topics

- [Create an Amazon SageMaker Notebook Instance for the tutorial](#)

- [Create a Jupyter notebook in the SageMaker notebook instance](#)
- [Prepare a dataset](#)
- [Train a Model](#)
- [Deploy the model to Amazon EC2](#)
- [Evaluate the model](#)
- [Clean up Amazon SageMaker notebook instance resources](#)

## Create an Amazon SageMaker Notebook Instance for the tutorial

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

An Amazon SageMaker notebook instance is a fully-managed machine learning (ML) Amazon Elastic Compute Cloud (Amazon EC2) compute instance. An Amazon SageMaker notebook instance runs the Jupyter Notebook application. Use the notebook instance to create and manage Jupyter notebooks for preprocessing data, train ML models, and deploy ML models.

### To create a SageMaker notebook instance

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Choose **Notebook instances**, and then choose **Create notebook instance**.
3. On the **Create notebook instance** page, provide the following information (if a field is not mentioned, leave the default values):
  - a. For **Notebook instance name**, type a name for your notebook instance.

- b. For **Notebook Instance type**, choose `m1.t2.medium`. This is the least expensive instance type that notebook instances support, and is enough for this exercise. If a `m1.t2.medium` instance type isn't available in your current AWS Region, choose `m1.t3.medium`.
- c. For **Platform Identifier**, choose a platform type to create the notebook instance on. This platform type defines the Operating System and the JupyterLab version that your notebook instance is created with. For information about platform identifier type, see [Amazon Linux 2 notebook instances](#). For information about JupyterLab versions, see [JupyterLab versioning](#).
- d. For **IAM role**, choose **Create a new role**, and then choose **Create role**. This IAM role automatically gets permissions to access any S3 bucket that has `sagemaker` in the name. It gets these permissions through the `AmazonSageMakerFullAccess` policy, which SageMaker AI attaches to the role.

 **Note**

If you want to grant the IAM role permission to access S3 buckets without `sagemaker` in the name, you need to attach the `S3FullAccess` policy. You can also limit the permissions to specific S3 buckets to the IAM role. For more information and examples of adding bucket policies to the IAM role, see [Bucket Policy Examples](#).

- e. Choose **Create notebook instance**.

In a few minutes, SageMaker AI launches a notebook instance and attaches a 5 GB of Amazon EBS storage volume to it. The notebook instance has a preconfigured Jupyter notebook server, SageMaker AI and AWS SDK libraries, and a set of Anaconda libraries.

For more information about creating a SageMaker notebook instance, see [Create a Notebook Instance](#).

## (Optional) Change SageMaker Notebook Instance Settings

To change the ML compute instance type or the size of the Amazon EBS storage of a SageMaker AI notebook instance, edit the notebook instance settings.

### To change and update the SageMaker Notebook instance type and the EBS volume

1. On the **Notebook instances** page in the SageMaker AI console, choose your notebook instance.

2. Choose **Actions**, choose **Stop**, and then wait until the notebook instance fully stops.
3. After the notebook instance status changes to **Stopped**, choose **Actions**, and then choose **Update settings**.
  - a. For **Notebook instance type**, choose a different ML instance type.
  - b. For **Volume size in GB**, type a different integer to specify a new EBS volume size.

 **Note**

EBS storage volumes are encrypted, so SageMaker AI can't determine the amount of available free space on the volume. Because of this, you can increase the volume size when you update a notebook instance, but you can't decrease the volume size. If you want to decrease the size of the ML storage volume in use, create a new notebook instance with the desired size.

4. At the bottom of the page, choose **Update notebook instance**.
5. When the update is complete, **Start** the notebook instance with the new settings.

For more information about updating SageMaker notebook instance settings, see [Update a Notebook Instance](#).

### (Optional) Advanced Settings for SageMaker Notebook Instances

The following tutorial video shows how to set up and use SageMaker notebook instances through the SageMaker AI console. It includes advanced options, such as SageMaker AI lifecycle configuration and importing GitHub repositories. (Length: 26:04)

For complete documentation about SageMaker notebook instance, see [Use Amazon SageMaker notebook Instances](#).

### Create a Jupyter notebook in the SageMaker notebook instance

 **Important**

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can

occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

To start scripting for training and deploying your model, create a Jupyter notebook in the SageMaker notebook instance. Using the Jupyter notebook, you can run machine learning (ML) experiments for training and inference while using SageMaker AI features and the AWS infrastructure.

## To create a Jupyter notebook

1. Open the notebook instance as follows:

- a. Sign in to the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
- b. On the **Notebook instances** page, open your notebook instance by choosing either:
  - **Open JupyterLab** for the JupyterLab interface
  - **Open Jupyter** for the classic Jupyter view

 **Note**

If the notebook instance status shows **Pending** in the **Status** column, your notebook instance is still being created. The status will change to **InService** when the notebook instance is ready to use.

2. Create a notebook as follows:

- If you opened the notebook in the JupyterLab view, on the **File** menu, choose **New**, and then choose **Notebook**. For **Select Kernel**, choose **conda\_python3**. This preinstalled environment includes the default Anaconda installation and Python 3.
- If you opened the notebook in the classic Jupyter view, on the **Files** tab, choose **New**, and then choose **conda\_python3**. This preinstalled environment includes the default Anaconda installation and Python 3.

3. Save the notebooks as follows:

- In the JupyterLab view, choose **File**, choose **Save Notebook As...**, and then rename the notebook.
- In the Jupyter classic view, choose **File**, choose **Save as...**, and then rename the notebook.

## Prepare a dataset

In this step, you load the [Adult Census dataset](#) to your notebook instance using the SHAP (SHapley Additive exPlanations) Library, review the dataset, transform it, and upload it to Amazon S3. SHAP is a game theoretic approach to explain the output of any machine learning model. For more information about SHAP, see [Welcome to the SHAP documentation](#).

To run the following example, paste the sample code into a cell in your notebook instance.

### Load Adult Census Dataset Using SHAP

Using the SHAP library, import the Adult Census dataset as shown following:

```
import shap
X, y = shap.datasets.adult()
X_display, y_display = shap.datasets.adult(display=True)
feature_names = list(X.columns)
feature_names
```

#### Note

If the current Jupyter kernel does not have the SHAP library, install it by running the following conda command:

```
%conda install -c conda-forge shap
```

If you're using JupyterLab, you must manually refresh the kernel after the installation and updates have completed. Run the following IPython script to shut down the kernel (the kernel will restart automatically):

```
import IPython
IPython.Application.instance().kernel.do_shutdown(True)
```

The `feature_names` list object should return the following list of features:

```
['Age',
 'Workclass',
 'Education-Num',
 'Marital Status',
 'Occupation',
 'Relationship',
 'Race',
 'Sex',
 'Capital Gain',
 'Capital Loss',
 'Hours per week',
 'Country']
```

 **Tip**

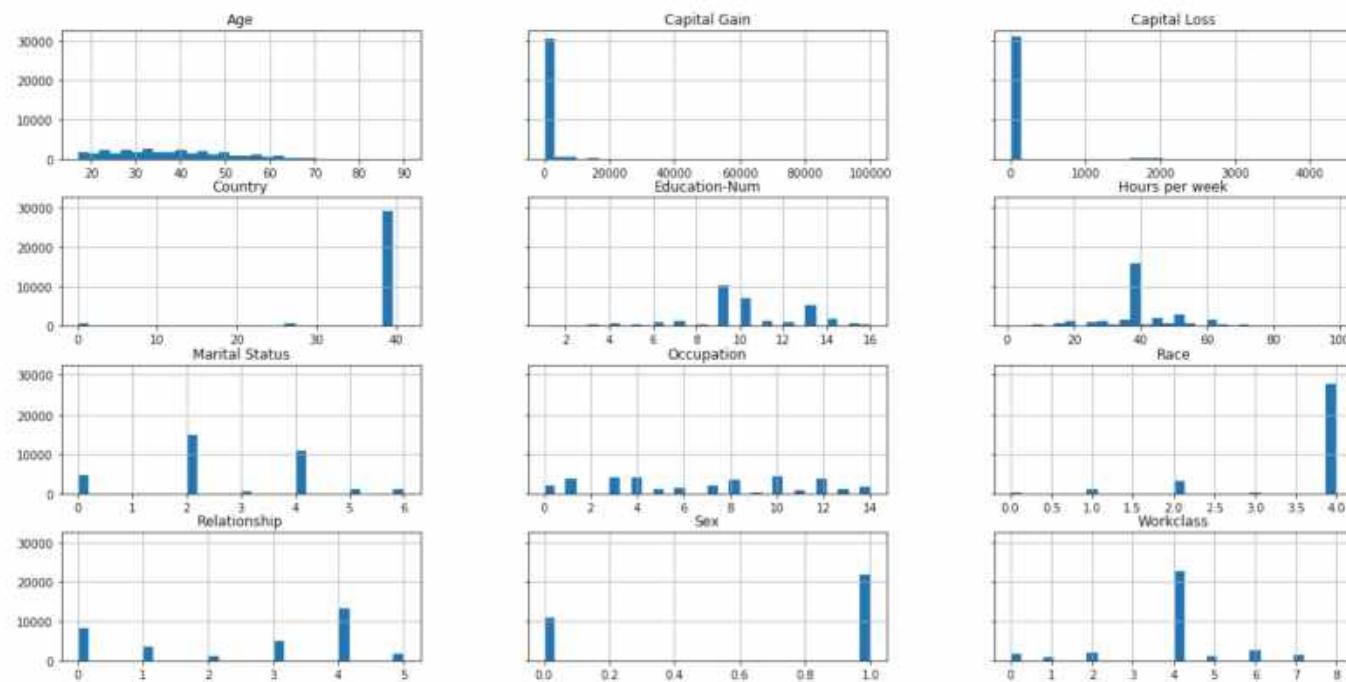
If you're starting with unlabeled data, you can use Amazon SageMaker Ground Truth to create a data labeling workflow in minutes. To learn more, see [Label Data](#).

## Overview the Dataset

Run the following script to display the statistical overview of the dataset and histograms of the numeric features.

```
display(X.describe())
hist = X.hist(bins=30, sharey=True, figsize=(20, 10))
```

	Age	Workclass	Education-Num	Marital Status	Occupation	Relationship	Race	Sex	Capital Gain	Capital Loss	Hours per week	Country
count	32561.000000	32561.000000	32561.000000	32561.000000	32561.000000	32561.000000	32561.000000	32561.000000	32561.000000	32561.000000	32561.000000	32561.000000
mean	38.581646	3.868892	10.980679	2.61836	6.572740	2.494518	3.665858	0.669205	1077.649170	87.303833	40.437454	36.718866
std	13.640442	1.455960	2.572562	1.506222	4.228857	1.758232	0.848806	0.470506	7885.911621	403.014771	12.347933	7.823782
min	17.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000
25%	28.000000	4.000000	9.000000	2.000000	3.000000	0.000000	4.000000	0.000000	0.000000	0.000000	40.000000	39.000000
50%	37.000000	4.000000	10.000000	2.000000	7.000000	3.000000	4.000000	1.000000	0.000000	0.000000	40.000000	39.000000
75%	46.000000	4.000000	12.000000	4.000000	10.000000	4.000000	4.000000	1.000000	0.000000	0.000000	45.000000	39.000000
max	90.000000	8.000000	16.000000	6.000000	14.000000	5.000000	4.000000	1.000000	99999.000000	4356.000000	99.000000	41.000000



### Tip

If you want to use a dataset that needs to be cleaned and transformed, you can simplify and streamline data preprocessing and feature engineering using Amazon SageMaker Data Wrangler. To learn more, see [Prepare ML Data with Amazon SageMaker Data Wrangler](#).

## Split the Dataset into Train, Validation, and Test Datasets

Using Sklearn, split the dataset into a training set and a test set. The training set is used to train the model, while the test set is used to evaluate the performance of the final trained model. The dataset is randomly sorted with the fixed random seed: 80 percent of the dataset for training set and 20 percent of it for a test set.

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,  
random_state=1)  
X_train_display = X_display.loc[X_train.index]
```

Split the training set to separate out a validation set. The validation set is used to evaluate the performance of the trained model while tuning the model's hyperparameters. 75 percent of the training set becomes the final training set, and the rest is the validation set.

```
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.25,  
random_state=1)  
X_train_display = X_display.loc[X_train.index]  
X_val_display = X_display.loc[X_val.index]
```

Using the pandas package, explicitly align each dataset by concatenating the numeric features with the true labels.

```
import pandas as pd  
train = pd.concat([pd.Series(y_train, index=X_train.index,  
                           name='Income>50K', dtype=int), X_train], axis=1)  
validation = pd.concat([pd.Series(y_val, index=X_val.index,  
                           name='Income>50K', dtype=int), X_val], axis=1)  
test = pd.concat([pd.Series(y_test, index=X_test.index,  
                           name='Income>50K', dtype=int), X_test], axis=1)
```

Check if the dataset is split and structured as expected:

```
train
```

	Income>50K	Age	Workclass	Education-Num	Marital Status	Occupation	Relationship	Race	Sex	Capital Gain	Capital Loss	Hours per week	Country
10911	1	47.0	4	9.0	2	3	4	4	1	0.0	0.0	40.0	39
17852	0	31.0	4	13.0	2	7	4	3	1	0.0	0.0	36.0	26
29165	1	32.0	4	10.0	2	13	5	4	0	0.0	0.0	32.0	39
30287	0	58.0	4	9.0	2	3	4	2	1	0.0	0.0	40.0	39
24019	0	17.0	4	6.0	4	6	3	4	1	0.0	0.0	20.0	39
...	...	...	...	...	...	...	...	...	...	...	...	...	...
21168	0	43.0	4	8.0	2	14	4	4	1	0.0	0.0	40.0	39
6452	0	26.0	4	9.0	4	7	0	4	1	0.0	0.0	52.0	39
31352	0	32.0	7	14.0	2	10	4	4	1	0.0	0.0	50.0	39
6575	0	45.0	4	9.0	4	6	0	4	1	0.0	0.0	40.0	39
23608	0	23.0	4	9.0	4	1	1	4	0	0.0	0.0	40.0	39

19536 rows × 13 columns

### validation

	Income>50K	Age	Workclass	Education-Num	Marital Status	Occupation	Relationship	Race	Sex	Capital Gain	Capital Loss	Hours per week	Country
16530	0	25.0	4	4.0	2	6	4	4	1	0.0	0.0	40.0	26
26723	0	41.0	6	9.0	2	5	5	4	0	0.0	0.0	40.0	39
3338	0	79.0	0	9.0	6	0	0	2	0	0.0	0.0	30.0	39
19367	1	43.0	2	15.0	2	10	4	4	1	15024.0	0.0	45.0	39
30274	0	51.0	5	9.0	4	12	2	4	1	0.0	0.0	40.0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1604	0	46.0	7	9.0	2	13	4	4	1	0.0	0.0	40.0	39
5937	1	71.0	4	10.0	6	12	0	4	1	0.0	0.0	35.0	39
11034	0	36.0	4	9.0	5	14	2	4	1	0.0	0.0	60.0	26
2819	0	31.0	4	9.0	4	8	0	4	0	0.0	0.0	40.0	39
14152	1	37.0	4	10.0	2	12	4	4	1	0.0	0.0	50.0	11

6512 rows × 13 columns

### test

	Income>50K	Age	Workclass	Education-Num	Marital Status	Occupation	Relationship	Race	Sex	Capital Gain	Capital Loss	Hours per week	Country
9646	0	62.0	6	4.0	6	8	0	4	0	0.0	0.0	66.0	39
709	0	18.0	4	7.0	4	8	2	4	1	0.0	0.0	26.0	39
7385	1	25.0	4	13.0	4	5	3	4	1	27828.0	0.0	50.0	39
16671	0	33.0	4	9.0	2	10	4	4	1	0.0	0.0	40.0	39
21932	0	36.0	4	7.0	4	7	1	4	0	0.0	0.0	40.0	39
...	...	...	...	...	...	...	...	...	...	...	...	...	...
5889	1	39.0	4	13.0	2	10	5	4	0	0.0	0.0	20.0	39
25723	0	17.0	4	6.0	4	12	3	4	0	0.0	0.0	20.0	39
29514	0	35.0	4	9.0	4	14	3	4	1	0.0	0.0	40.0	39
1600	0	30.0	4	7.0	2	3	4	4	1	0.0	0.0	45.0	39
639	1	52.0	6	16.0	2	10	4	4	1	0.0	0.0	60.0	39

6513 rows × 13 columns

## Convert the Train and Validation Datasets to CSV Files

Convert the train and validation dataframe objects to CSV files to match the input file format for the XGBoost algorithm.

```
# Use 'csv' format to store the data
# The first column is expected to be the output column
train.to_csv('train.csv', index=False, header=False)
validation.to_csv('validation.csv', index=False, header=False)
```

## Upload the Datasets to Amazon S3

Using the SageMaker AI and Boto3, upload the training and validation datasets to the default Amazon S3 bucket. The datasets in the S3 bucket will be used by a compute-optimized SageMaker instance on Amazon EC2 for training.

The following code sets up the default S3 bucket URI for your current SageMaker AI session, creates a new demo-sagemaker-xgboost-adult-income-prediction folder, and uploads the training and validation datasets to the data subfolder.

```
import sagemaker, boto3, os
bucket = sagemaker.Session().default_bucket()
prefix = "demo-sagemaker-xgboost-adult-income-prediction"

boto3.Session().resource('s3').Bucket(bucket).Object(
    os.path.join(prefix, 'data/train.csv')).upload_file('train.csv')
boto3.Session().resource('s3').Bucket(bucket).Object(
```

```
os.path.join(prefix, 'data/validation.csv')).upload_file('validation.csv')
```

Run the following AWS CLI to check if the CSV files are successfully uploaded to the S3 bucket.

```
! aws s3 ls {bucket}/{prefix}/data --recursive
```

This should return the following output:

```
2021-01-14 17:52:09    786285 demo-sagemaker-xgboost-adult-income-prediction/data/train.csv  
2021-01-14 17:52:10    262122 demo-sagemaker-xgboost-adult-income-prediction/data/validation.csv
```

## Train a Model

In this step, you choose a training algorithm and run a training job for the model. The [Amazon SageMaker Python SDK](#) provides framework estimators and generic estimators to train your model while orchestrating the machine learning (ML) lifecycle accessing the SageMaker AI features for training and the AWS infrastructures, such as Amazon Elastic Container Registry (Amazon ECR), Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3). For more information about SageMaker AI built-in framework estimators, see [Frameworks](#) in the [Amazon SageMaker Python SDK](#) documentation. For more information about built-in algorithms, see [Built-in algorithms and pretrained models in Amazon SageMaker](#).

### Topics

- [Choose the Training Algorithm](#)
- [Create and Run a Training Job](#)

### Choose the Training Algorithm

To choose the right algorithm for your dataset, you typically need to evaluate different models to find the most suitable models to your data. For simplicity, the SageMaker AI [XGBoost algorithm with Amazon SageMaker AI](#) built-in algorithm is used throughout this tutorial without the pre-evaluation of models.

#### Tip

If you want SageMaker AI to find an appropriate model for your tabular dataset, use Amazon SageMaker Autopilot that automates a machine learning solution. For more information, see [SageMaker Autopilot](#).

## Create and Run a Training Job

After you figured out which model to use, start constructing a SageMaker AI estimator for training. This tutorial uses the XGBoost built-in algorithm for the SageMaker AI generic estimator.

### To run a model training job

1. Import the [Amazon SageMaker Python SDK](#) and start by retrieving the basic information from your current SageMaker AI session.

```
import sagemaker

region = sagemaker.Session().boto_region_name
print("AWS Region: {}".format(region))

role = sagemaker.get_execution_role()
print("RoleArn: {}".format(role))
```

This returns the following information:

- `region` – The current AWS Region where the SageMaker AI notebook instance is running.
- `role` – The IAM role used by the notebook instance.

#### Note

Check the SageMaker Python SDK version by running `sagemaker.__version__`.

This tutorial is based on `sagemaker>=2.20`. If the SDK is outdated, install the latest version by running the following command:

```
! pip install -qU sagemaker
```

If you run this installation in your exiting SageMaker Studio or notebook instances, you need to manually refresh the kernel to finish applying the version update.

2. Create an XGBoost estimator using the `sagemaker.estimator.Estimator` class. In the following example code, the XGBoost estimator is named `xgb_model`.

```
from sagemaker.debugger import Rule, ProfilerRule, rule_configs
from sagemaker.session import TrainingInput
```

```
s3_output_location='s3://{} / {} / {}'.format(bucket, prefix, 'xgboost_model')

container=sagemaker.image_uris.retrieve("xgboost", region, "1.2-1")
print(container)

xgb_model=sagemaker.estimator.Estimator(
    image_uri=container,
    role=role,
    instance_count=1,
    instance_type='ml.m4.xlarge',
    volume_size=5,
    output_path=s3_output_location,
    sagemaker_session=sagemaker.Session(),
    rules=[
        Rule.sagemaker(rule_configs.create_xgboost_report()),
        ProfilerRule.sagemaker(rule_configs.ProfilerReport())
    ]
)
```

To construct the SageMaker AI estimator, specify the following parameters:

- **image\_uri** – Specify the training container image URI. In this example, the SageMaker AI XGBoost training container URI is specified using `sagemaker.image_uris.retrieve`.
- **role** – The AWS Identity and Access Management (IAM) role that SageMaker AI uses to perform tasks on your behalf (for example, reading training results, call model artifacts from Amazon S3, and writing training results to Amazon S3).
- **instance\_count** and **instance\_type** – The type and number of Amazon EC2 ML compute instances to use for model training. For this training exercise, you use a single `ml.m4.xlarge` instance, which has 4 CPUs, 16 GB of memory, an Amazon Elastic Block Store (Amazon EBS) storage, and a high network performance. For more information about EC2 compute instance types, see [Amazon EC2 Instance Types](#). For more information about billing, see [Amazon SageMaker pricing](#).
- **volume\_size** – The size, in GB, of the EBS storage volume to attach to the training instance. This must be large enough to store training data if you use File mode (File mode is on by default). If you don't specify this parameter, its value defaults to 30.
- **output\_path** – The path to the S3 bucket where SageMaker AI stores the model artifact and training results.

- `sagemaker_session` – The session object that manages interactions with SageMaker API operations and other AWS service that the training job uses.
- `rules` – Specify a list of SageMaker Debugger built-in rules. In this example, the `create_xgboost_report()` rule creates an XGBoost report that provides insights into the training progress and results, and the `ProfilerReport()` rule creates a report regarding the EC2 compute resource utilization. For more information, see [SageMaker Debugger interactive report for XGBoost](#).

 **Tip**

If you want to run distributed training of large sized deep learning models, such as convolutional neural networks (CNN) and natural language processing (NLP) models, use SageMaker AI Distributed for data parallelism or model parallelism. For more information, see [Distributed training in Amazon SageMaker AI](#).

3. Set the hyperparameters for the XGBoost algorithm by calling the `set_hyperparameters` method of the estimator. For a complete list of XGBoost hyperparameters, see [XGBoost hyperparameters](#).

```
xgb_model.set_hyperparameters(  
    max_depth = 5,  
    eta = 0.2,  
    gamma = 4,  
    min_child_weight = 6,  
    subsample = 0.7,  
    objective = "binary:logistic",  
    num_round = 1000  
)
```

 **Tip**

You can also tune the hyperparameters using the SageMaker AI hyperparameter optimization feature. For more information, see [Automatic model tuning with SageMaker AI](#).

4. Use the `TrainingInput` class to configure a data input flow for training. The following example code shows how to configure `TrainingInput` objects to use the training and

validation datasets you uploaded to Amazon S3 in the [Split the Dataset into Train, Validation, and Test Datasets](#) section.

```
from sagemaker.session import TrainingInput

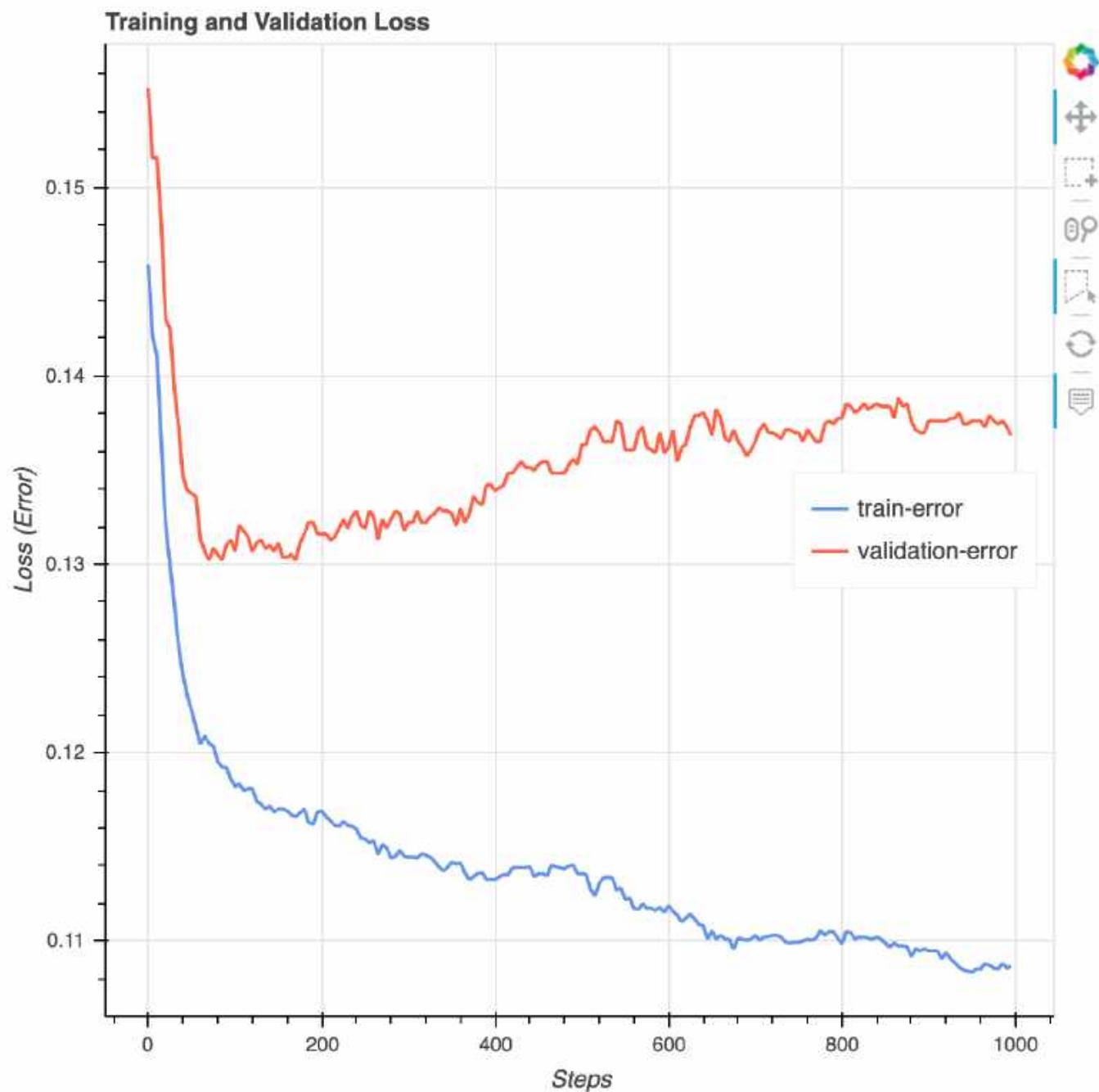
train_input = TrainingInput(
    "s3://{}:{}{}".format(bucket, prefix, "data/train.csv"), content_type="csv"
)
validation_input = TrainingInput(
    "s3://{}:{}{}".format(bucket, prefix, "data/validation.csv"),
    content_type="csv"
)
```

5. To start model training, call the estimator's `fit` method with the training and validation datasets. By setting `wait=True`, the `fit` method displays progress logs and waits until training is complete.

```
xgb_model.fit({"train": train_input, "validation": validation_input}, wait=True)
```

For more information about model training, see [Train a Model with Amazon SageMaker](#). This tutorial training job might take up to 10 minutes.

After the training job has done, you can download an XGBoost training report and a profiling report generated by SageMaker Debugger. The XGBoost training report offers you insights into the training progress and results, such as the loss function with respect to iteration, feature importance, confusion matrix, accuracy curves, and other statistical results of training. For example, you can find the following loss curve from the XGBoost training report which clearly indicates that there is an overfitting problem.



Run the following code to specify the S3 bucket URI where the Debugger training reports are generated and check if the reports exist.

```
rule_output_path = xgb_model.output_path + "/" +
xgb_model.latest_training_job.job_name + "/rule-output"
! aws s3 ls {rule_output_path} --recursive
```

Download the Debugger XGBoost training and profiling reports to the current workspace:

```
! aws s3 cp {rule_output_path} ./ --recursive
```

Run the following IPython script to get the file link of the XGBoost training report:

```
from IPython.display import FileLink, FileLinks  
display("Click link below to view the XGBoost Training report",  
       FileLink("CreateXgboostReport/xgboost_report.html"))
```

The following IPython script returns the file link of the Debugger profiling report that shows summaries and details of the EC2 instance resource utilization, system bottleneck detection results, and python operation profiling results:

```
profiler_report_name = [rule["RuleConfigurationName"]  
                       for rule in  
                       xgb_model.latest_training_job.rule_job_summary()  
                       if "Profiler" in rule["RuleConfigurationName"]][0]  
profiler_report_name  
display("Click link below to view the profiler report",  
       FileLink(profiler_report_name+"/profiler-output/profiler-report.html"))
```

### Tip

If the HTML reports do not render plots in the JupyterLab view, you must choose **Trust HTML** at the top of the reports.

To identify training issues, such as overfitting, vanishing gradients, and other problems that prevents your model from converging, use SageMaker Debugger and take automated actions while prototyping and training your ML models. For more information, see [Amazon SageMaker Debugger](#). To find a complete analysis of model parameters, see the [Explainability with Amazon SageMaker Debugger](#) example notebook.

You now have a trained XGBoost model. SageMaker AI stores the model artifact in your S3 bucket. To find the location of the model artifact, run the following code to print the `model_data` attribute of the `xgb_model` estimator:

```
xgb_model.model_data
```

**Tip**

To measure biases that can occur during each stage of the ML lifecycle (data collection, model training and tuning, and monitoring of ML models deployed for prediction), use SageMaker Clarify. For more information, see [Model Explainability](#). For an end-to-end example, see the [Fairness and Explainability with SageMaker Clarify](#) example notebook.

## Deploy the model to Amazon EC2

To get predictions, deploy your model to Amazon EC2 using Amazon SageMaker AI.

### Topics

- [Deploy the Model to SageMaker AI Hosting Services](#)
- [\(Optional\) Use SageMaker AI Predictor to Reuse the Hosted Endpoint](#)
- [\(Optional\) Make Prediction with Batch Transform](#)

### Deploy the Model to SageMaker AI Hosting Services

To host a model through Amazon EC2 using Amazon SageMaker AI, deploy the model that you trained in [Create and Run a Training Job](#) by calling the `deploy` method of the `xgb_model` estimator. When you call the `deploy` method, you must specify the number and type of EC2 ML instances that you want to use for hosting an endpoint.

```
import sagemaker
from sagemaker.serializers import CSVSerializer
xgb_predictor=xgb_model.deploy(
    initial_instance_count=1,
    instance_type='ml.t2.medium',
    serializer=CSVSerializer()
)
```

- `initial_instance_count` (int) – The number of instances to deploy the model.
- `instance_type` (str) – The type of instances that you want to operate your deployed model.

- `serializer` (int) – Serialize input data of various formats (a NumPy array, list, file, or buffer) to a CSV-formatted string. We use this because the XGBoost algorithm accepts input files in CSV format.

The `deploy` method creates a deployable model, configures the SageMaker AI hosting services endpoint, and launches the endpoint to host the model. For more information, see the [SageMaker AI generic Estimator's deploy class method](#) in the [Amazon SageMaker Python SDK](#). To retrieve the name of endpoint that's generated by the `deploy` method, run the following code:

```
xgb_predictor.endpoint_name
```

This should return the endpoint name of the `xgb_predictor`. The format of the endpoint name is "sagemaker-xgboost-YYYY-MM-DD-HH-MM-SS-SSS". This endpoint stays active in the ML instance, and you can make instantaneous predictions at any time unless you shut it down later. Copy this endpoint name and save it to reuse and make real-time predictions elsewhere in SageMaker Studio or SageMaker AI notebook instances.

 **Tip**

To learn more about compiling and optimizing your model for deployment to Amazon EC2 instances or edge devices, see [Compile and Deploy Models with Neo](#).

## (Optional) Use SageMaker AI Predictor to Reuse the Hosted Endpoint

After you deploy the model to an endpoint, you can set up a new SageMaker AI predictor by pairing the endpoint and continuously make real-time predictions in any other notebooks. The following example code shows how to use the SageMaker AI Predictor class to set up a new predictor object using the same endpoint. Re-use the endpoint name that you used for the `xgb_predictor`.

```
import sagemaker
xgb_predictor_reuse=sagemaker.predictor.Predictor(
    endpoint_name="sagemaker-xgboost-YYYY-MM-DD-HH-MM-SS-SSS",
    sagemaker_session=sagemaker.Session(),
    serializer=sagemaker.serializers.CSVSerializer()
)
```

The `xgb_predictor_reuse` Predictor behaves exactly the same as the original `xgb_predictor`. For more information, see the [SageMaker AI Predictor](#) class in the [Amazon SageMaker Python SDK](#).

## (Optional) Make Prediction with Batch Transform

Instead of hosting an endpoint in production, you can run a one-time batch inference job to make predictions on a test dataset using the SageMaker AI batch transform. After your model training has completed, you can extend the estimator to a `transformer` object, which is based on the [SageMaker AI Transformer](#) class. The batch transformer reads in input data from a specified S3 bucket and makes predictions.

### To run a batch transform job

1. Run the following code to convert the feature columns of the test dataset to a CSV file and uploads to the S3 bucket:

```
X_test.to_csv('test.csv', index=False, header=False)

boto3.Session().resource('s3').Bucket(bucket).Object(
    os.path.join(prefix, 'test/test.csv')).upload_file('test.csv')
```

2. Specify S3 bucket URIs of input and output for the batch transform job as shown following:

```
# The location of the test dataset
batch_input = 's3://{}{}'.format(bucket, prefix)

# The location to store the results of the batch transform job
batch_output = 's3://{}{}'.format(bucket, prefix)
```

3. Create a transformer object specifying the minimal number of parameters: the `instance_count` and `instance_type` parameters to run the batch transform job, and the `output_path` to save prediction data as shown following:

```
transformer = xgb_model.transformer(
    instance_count=1,
    instance_type='ml.m4.xlarge',
    output_path=batch_output
)
```

4. Initiate the batch transform job by executing the `transform( )` method of the `transformer` object as shown following:

```
transformer.transform(  
    data=batch_input,  
    data_type='S3Prefix',  
    content_type='text/csv',  
    split_type='Line'  
)  
transformer.wait()
```

- When the batch transform job is complete, SageMaker AI creates the test.csv.out prediction data saved in the batch\_output path, which should be in the following format: s3://sagemaker-<region>-111122223333/demo-sagemaker-xgboost-adult-income-prediction/batch-prediction. Run the following AWS CLI to download the output data of the batch transform job:

```
! aws s3 cp {batch_output} ./ --recursive
```

This should create the test.csv.out file under the current working directory. You'll be able to see the float values that are predicted based on the logistic regression of the XGBoost training job.

## Evaluate the model

Now that you have trained and deployed a model using Amazon SageMaker AI, evaluate the model to ensure that it generates accurate predictions on new data. For model evaluation, use the test dataset that you created in [Prepare a dataset](#).

### Evaluate the Model Deployed to SageMaker AI Hosting Services

To evaluate the model and use it in production, invoke the endpoint with the test dataset and check whether the inferences you get returns a target accuracy you want to achieve.

#### To evaluate the model

- Set up the following function to predict each line of the test set. In the following example code, the rows argument is to specify the number of lines to predict at a time. You can change the value of it to perform a batch inference that fully utilizes the instance's hardware resource.

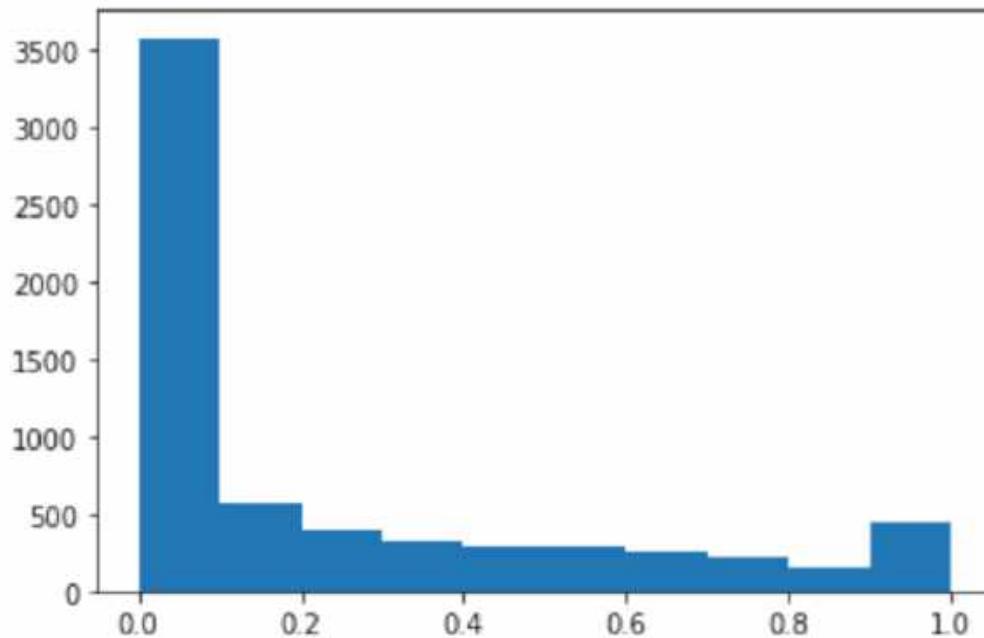
```
import numpy as np  
def predict(data, rows=1000):
```

```
split_array = np.array_split(data, int(data.shape[0] / float(rows) + 1))
predictions = ''
for array in split_array:
    predictions = ','.join([predictions,
    xgb_predictor.predict(array).decode('utf-8')])
return np.fromstring(predictions[1:], sep=',')
```

- Run the following code to make predictions of the test dataset and plot a histogram. You need to take only the feature columns of the test dataset, excluding the 0th column for the actual values.

```
import matplotlib.pyplot as plt

predictions=predict(test.to_numpy()[:,1:])
plt.hist(predictions)
plt.show()
```



- The predicted values are float type. To determine True or False based on the float values, you need to set a cutoff value. As shown in the following example code, use the Scikit-learn library to return the output confusion metrics and classification report with a cutoff of 0.5.

```
import sklearn

cutoff=0.5
```

```
print(sklearn.metrics.confusion_matrix(test.iloc[:, 0], np.where(predictions > cutoff, 1, 0)))
print(sklearn.metrics.classification_report(test.iloc[:, 0], np.where(predictions > cutoff, 1, 0)))
```

This should return the following confusion matrix:

```
[[4670  356]
 [ 480 1007]]
      precision    recall   f1-score   support
          0       0.91     0.93     0.92     5026
          1       0.74     0.68     0.71     1487

   accuracy                           0.87     6513
  macro avg       0.82     0.80     0.81     6513
weighted avg       0.87     0.87     0.87     6513
```

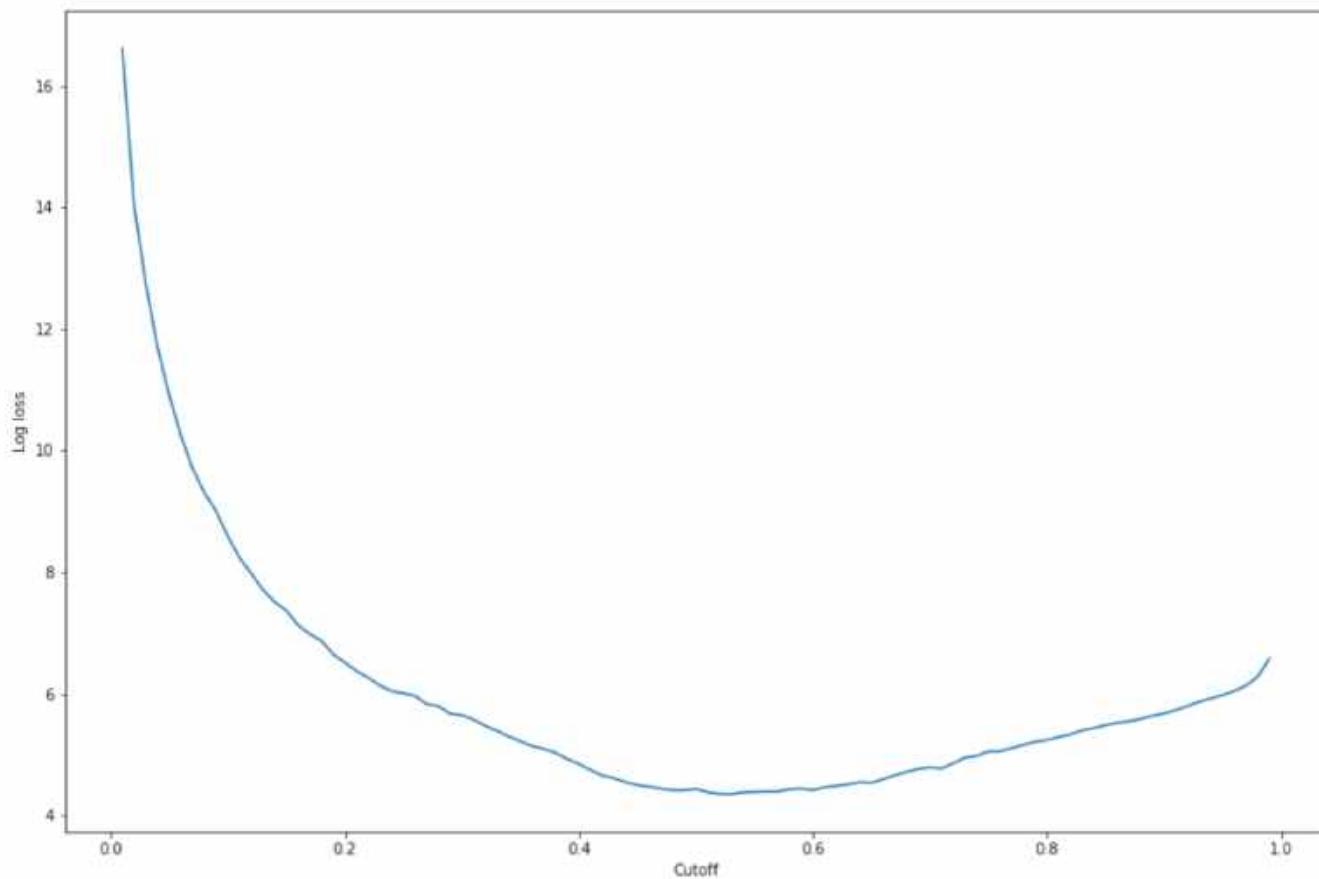
4. To find the best cutoff with the given test set, compute the log loss function of the logistic regression. The log loss function is defined as the negative log-likelihood of a logistic model that returns prediction probabilities for its ground truth labels. The following example code numerically and iteratively calculates the log loss values  $-(y \log(p) + (1-y) \log(1-p))$ , where  $y$  is the true label and  $p$  is a probability estimate of the corresponding test sample. It returns a log loss versus cutoff graph.

```
import matplotlib.pyplot as plt

cutoffs = np.arange(0.01, 1, 0.01)
log_loss = []
for c in cutoffs:
    log_loss.append(
        sklearn.metrics.log_loss(test.iloc[:, 0], np.where(predictions > c, 1, 0))
    )

plt.figure(figsize=(15,10))
plt.plot(cutoffs, log_loss)
plt.xlabel("Cutoff")
plt.ylabel("Log loss")
plt.show()
```

This should return the following log loss curve.



5. Find the minimum points of the error curve using the NumPy `argmin` and `min` functions:

```
print(  
    'Log loss is minimized at a cutoff of ', cthresholds[np.argmin(log_loss)],  
    ', and the log loss value at the minimum is ', np.min(log_loss)  
)
```

This should return: Log loss is minimized at a cutoff of 0.53, and the log loss value at the minimum is 4.348539186773897.

Instead of computing and minimizing the log loss function, you can estimate a cost function as an alternative. For example, if you want to train a model to perform a binary classification for a business problem such as a customer churn prediction problem, you can set weights to the elements of confusion matrix and calculate the cost function accordingly.

You have now trained, deployed, and evaluated your first model in SageMaker AI.

**Tip**

To monitor model quality, data quality, and bias drift, use Amazon SageMaker Model Monitor and SageMaker AI Clarify. To learn more, see [Amazon SageMaker Model Monitor](#), [Monitor Data Quality](#), [Monitor Model Quality](#), [Monitor Bias Drift](#), and [Monitor Feature Attribution Drift](#).

**Tip**

To get human review of low confidence ML predictions or a random sample of predictions, use Amazon Augmented AI human review workflows. For more information, see [Using Amazon Augmented AI for Human Review](#).

## Clean up Amazon SageMaker notebook instance resources

To avoid incurring unnecessary charges, use the AWS Management Console to delete the endpoints and resources that you created while running the exercises.

**Note**

Training jobs and logs cannot be deleted and are retained indefinitely.

**Note**

If you plan to explore other exercises in this guide, you might want to keep some of these resources, such as your notebook instance, S3 bucket, and IAM role.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/> and delete the following resources:
  - The endpoint. Deleting the endpoint also deletes the ML compute instance or instances that support it.
    1. Under **Inference**, choose **Endpoints**.

2. Choose the endpoint that you created in the example, choose **Actions**, and then choose **Delete**.
- The endpoint configuration.
    1. Under **Inference**, choose **Endpoint configurations**.
    2. Choose the endpoint configuration that you created in the example, choose **Actions**, and then choose **Delete**.
  - The model.
    1. Under **Inference**, choose **Models**.
    2. Choose the model that you created in the example, choose **Actions**, and then choose **Delete**.
  - The notebook instance. Before deleting the notebook instance, stop it.
    1. Under **Notebook**, choose **Notebook instances**.
    2. Choose the notebook instance that you created in the example, choose **Actions**, and then choose **Stop**. The notebook instance takes several minutes to stop. When the **Status** changes to **Stopped**, move on to the next step.
    3. Choose **Actions**, and then choose **Delete**.
  - 2. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>, and then delete the bucket that you created for storing model artifacts and the training dataset.
  - 3. Open the Amazon CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>, and then delete all of the log groups that have names starting with /aws/sagemaker/.

## Amazon Linux 2 notebook instances

### **Important**

Notebook instances running on the JupyterLab 1 and JupyterLab 3 platforms will reach end of support on June 30, 2025. We strongly recommend migrating your work to the new JupyterLab 4 notebook instances before this date to ensure you have a secure and supported environment. For more information, see [JupyterLab version maintenance](#).

Amazon SageMaker notebook instances currently support Amazon Linux 2 (AL2) operating systems. You can select the operating system that your notebook instance is based on when you create the notebook instance.

SageMaker AI supports notebook instances based on the following Amazon Linux 2 operating systems.

- **notebook-al2-v1:** These notebook instances support JupyterLab version 1. For information about JupyterLab versions, see [JupyterLab versioning](#).
- **notebook-al2-v2:** These notebook instances support JupyterLab version 3. For information about JupyterLab versions, see [JupyterLab versioning](#).
- **notebook-al2-v3:** These notebook instances support JupyterLab version 4. For information about JupyterLab versions, see [JupyterLab versioning](#).

Notebook instances created before 08/18/2021 automatically run on Amazon Linux (AL1).

Notebook instances based on AL1 entered a maintenance phase as of 12/01/2022 and are no longer available for new notebook instance creation as of 02/01/2023. To replace AL1, you now have the option to create Amazon SageMaker notebook instances with AL2. For more information, see [AL1 Maintenance Phase Plan](#).

## Topics

- [Supported instance types](#)
- [Available Kernels](#)
- [AL1 Maintenance Phase Plan](#)

## Supported instance types

Amazon Linux 2 supports instance types listed under **Notebook Instances** in [Amazon SageMaker Pricing](#) with the exception that Amazon Linux 2 does not support m1.p2 instances.

## Available Kernels

The following table gives information about the available kernels for SageMaker notebook instances. All of these images are supported on notebook instances based on the notebook-al2-v1, notebook-al2-v2, and notebook-al2-v3 operating systems.

### SageMaker notebook instance kernels

Kernel name	Description
R	A kernel used to perform data analysis and visualization using R code from a Jupyter notebook.
Sparkmagic (PySpark)	A kernel used to do data science with remote Spark clusters from Jupyter notebooks using the Python programming language. This kernel comes with Python 3.10.
Sparkmagic (Spark)	A kernel used to do data science with remote Spark clusters from Jupyter notebooks using the Scala programming language. This kernel comes with Python 3.10.
Sparkmagic (SparkR)	A kernel used to do data science with remote Spark clusters from Jupyter notebooks using the R programming language. This kernel comes with Python 3.10.
conda_python3	A conda environment that comes pre-installed with popular packages for data science and machine learning. This kernel comes with Python 3.10.
conda_pytorch_p310	A conda environment that comes pre-installed with PyTorch version 2.2.0, as well as popular data science and machine learning packages. This kernel comes with Python 3.10.
conda_tensorflow2_p310	A conda environment that comes pre-installed with TensorFlow version 2.16.0, as well as popular data science and machine learning packages. This kernel comes with Python 3.10.

## AL1 Maintenance Phase Plan

The following table is a timeline for when AL1 entered its extended maintenance phase. The AL1 maintenance phase also coincides with the deprecation of Python 2 and Chainer. Notebooks based on AL2 do not have managed Python 2 and Chainer kernels.

Date	Description
08/18/2021	Notebook instances based on AL2 are launched. Newly launched notebook instances still default to AL1. AL1 is supported with security patches and updates, but no new features. You can choose between the two operating systems when launching a new notebook instance.
10/31/2022	The default platform identifier for SageMaker notebook instances changes from Amazon Linux (al1-v1) to Amazon Linux 2 (al2-v2). You can choose between the two operating systems when launching a new notebook instance.
12/01/2022	AL1 is no longer supported with non-critical security patches and updates. AL1 still receives fixes for <a href="#">critical</a> security-related issues. You can still launch instances on AL1, but assume the risks associated with using an unsupported operating system.
02/01/2023	AL1 is no longer an available option for new notebook instance creation. After this date, customers can create notebook instances with the AL2 platform identifiers. Existing notebooks with an <code>INSERVICE</code> status should be migrated to the latest platform since

Date	Description
03/31/2024	<p>continuous availability of AL1 notebook instances cannot be guaranteed.</p> <p>AL1 reaches its end of life on notebook instances on March 31, 2024. After this date, AL1 will no longer receive any security updates, bug fixes, or be available for new notebook instance creation.</p> <ul style="list-style-type: none"><li>• Existing AL1 notebook instances with a STOPPED status cannot be restarted.</li><li>• Existing notebooks with an INSERVICE status should be migrated to the latest platform since continuous availability of AL1 notebook instances cannot be guaranteed.</li></ul>

## Migrating to Amazon Linux 2

Your existing AL1 notebook instance is not automatically migrated to Amazon Linux 2. To upgrade your AL1 notebook instance to Amazon Linux 2, you must create a new notebook instance, replicate your code and environment, and delete your old notebook instance. For more information, see the [Amazon Linux 2 migration blog](#).

## JupyterLab versioning

### Important

Notebook instances running on the JupyterLab 1 and JupyterLab 3 platforms will reach end of support on June 30, 2025. We strongly recommend migrating your work to the new JupyterLab 4 notebook instances before this date to ensure you have a secure and supported environment. For more information, see [JupyterLab version maintenance](#).

The Amazon SageMaker notebook instance interface is based on JupyterLab, which is a web-based interactive development environment for notebooks, code, and data. Notebooks now support

using either JupyterLab 1, JupyterLab 3, or JupyterLab 4. A single notebook instance can run a single instance of JupyterLab (at most). You can have multiple notebook instances with different JupyterLab versions.

You can configure your notebook to run your preferred JupyterLab version by selecting the appropriate platform identifier. Use either the AWS CLI or the SageMaker AI console when creating your notebook instance. For more information about platform identifiers, see [Amazon Linux 2 vs Amazon Linux notebook instances](#). If you don't explicitly configure a platform identifier, your notebook instance defaults to running JupyterLab 1.

## Topics

- [JupyterLab version maintenance](#)
- [JupyterLab 4](#)
- [JupyterLab 3](#)
- [Create a notebook with your JupyterLab version](#)
- [View the JupyterLab version of a notebook from the console](#)

## JupyterLab version maintenance

Notebook instances running on the JupyterLab 1 and JupyterLab 3 platforms will reach its end of standard support on June 30, 2025. Effective on this date:

- You will no longer be able to create new, or restart stopped, JupyterLab 1 and JupyterLab 3 notebook instances.
- Existing in-service JupyterLab 1 and JupyterLab 3 notebook instances may continue to function, but will no longer receive SageMaker AI security updates or critical bug fixes.
- You will be responsible for managing the security of these instances on JupyterLab 1 and JupyterLab 3.
- If any issues arise with the existing JupyterLab 1 or JupyterLab 3 notebook instances, SageMaker AI cannot guarantee their continued availability without migrating the workload to a new JupyterLab 4 notebook instance.

We strongly recommend migrating your work to the new JupyterLab 4 notebook instances (platform identifier [notebook-al2-v3](#)) before June 30, 2025 to ensure you have a secure and supported environment. This will allow you to leverage the latest versions of Jupyter notebooks,

JupyterLab, and other ML libraries. For instructions, see [migrate your work to an SageMaker AI notebook instance with Amazon Linux 2](#).

## JupyterLab 4

JupyterLab 4 support is available only on the Amazon Linux 2 operating system platform.

JupyterLab 4 includes the following features that are not available in JupyterLab 3:

- Optimized rendering for a faster experience
- Opt-in settings for faster tab switching and better performance with long notebooks. For more information, see the blog post [JupyterLab 4.0 is Here](#).
- Upgraded text editor
- New extension manager installing from pypi
- Added improvements to the UI, including document search and accessibility improvements

You can run JupyterLab 4 by specifying notebook-a12-v3 as the platform identifier when creating your notebook instance.

 **Note**

If you attempt to migrate to a JupyterLab 4 Notebook Instance from another JupyterLab version, the package version changes between JupyterLab 3 and JupyterLab 4 might break any existing lifecycle configurations or Jupyter/JupyterLab extensions.

## Package version changes

JupyterLab 4 has the following package version changes from JupyterLab 3:

- JupyterLab has been upgraded from 3.x to 4.x.
- Jupyter notebook has been upgraded from 6.x to 7.x.
- jupyterlab-git has been updated to version 0.50.0.

## JupyterLab 3

### Important

Notebook instances running on the JupyterLab 1 and JupyterLab 3 platforms will reach end of support on June 30, 2025. We strongly recommend migrating your work to the new JupyterLab 4 notebook instances before this date to ensure you have a secure and supported environment. For more information, see [JupyterLab version maintenance](#).

JupyterLab 3 support is available only on the Amazon Linux 2 operating system platform. JupyterLab 3 includes the following features that are not available in JupyterLab 1. For more information about these features, see [JupyterLab 3.0 is released!](#).

- Visual debugger when using the following kernels:
  - conda\_pytorch\_p38
  - conda\_tensorflow2\_p38
  - conda\_amazonei\_pytorch\_latest\_p37
- File browser filter
- Table of Contents (TOC)
- Multi-language support
- Simple mode
- Single interface mode
- Live editing SVG files with updated rendering
- User interface for notebook cell tags

### Important changes to JupyterLab 3

For information about important changes when using JupyterLab 3, see the following JupyterLab change logs:

- [v2.0.0](#)
- [v3.0.0](#)

### Package version changes

JupyterLab 3 has the following package version changes from JupyterLab 1:

- JupyterLab has been upgraded from 1.x to 3.x.
- Jupyter notebook has been upgraded from 5.x to 6.x.
- jupyterlab-git has been updated to version 0.37.1.
- nbserverproxy 0.x (0.3.2) has been replaced with jupyter-server-proxy 3.x (3.2.1).

## Create a notebook with your JupyterLab version

### Important

Notebook instances running on the JupyterLab 1 and JupyterLab 3 platforms will reach end of support on June 30, 2025. We strongly recommend migrating your work to the new JupyterLab 4 notebook instances before this date to ensure you have a secure and supported environment. For more information, see [JupyterLab version maintenance](#).

You can select the JupyterLab version when creating your notebook instance from the console following the steps in [Create an Amazon SageMaker notebook instance](#).

You can also select the JupyterLab version by passing the `platform-identifier` parameter when creating your notebook instance using the AWS CLI as follows:

```
create-notebook-instance --notebook-instance-name <NEW_NOTEBOOK_NAME> \
--instance-type <INSTANCE_TYPE> \
--role-arn <YOUR_ROLE_ARN> \
--platform-identifier <PLATFORM_TO_USE>
```

## View the JupyterLab version of a notebook from the console

### Important

Notebook instances running on the JupyterLab 1 and JupyterLab 3 platforms will reach end of support on June 30, 2025. We strongly recommend migrating your work to the new JupyterLab 4 notebook instances before this date to ensure you have a secure and supported environment. For more information, see [JupyterLab version maintenance](#).

You can view the JupyterLab version of a notebook using the following procedure:

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation, select **Notebook**.
3. From the dropdown menu, select **Notebook instances** to navigate to the **Notebook instances** page.
4. From the list of notebook instances, select your notebook instance name.
5. On the **Notebook instance settings** page, view the **Platform Identifier** to see the JupyterLab version of the notebook.

## Create an Amazon SageMaker notebook instance

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

An Amazon SageMaker notebook instance is a ML compute instance running the Jupyter Notebook application. SageMaker AI manages creating the instance and related resources. Use Jupyter notebooks in your notebook instance to:

- prepare and process data
- write code to train models
- deploy models to SageMaker AI hosting
- test or validate your models

To create a notebook instance, use either the SageMaker AI console or the [CreateNotebookInstance](#) API.

The notebook instance type you choose depends on how you use your notebook instance. Ensure that your notebook instance is not bound by memory, CPU, or IO. To load a dataset into memory on the notebook instance for exploration or preprocessing, choose an instance type with enough RAM memory for your dataset. This requires an instance with at least 16 GB of memory (.xlarge or larger). If you plan to use the notebook for compute intensive preprocessing, we recommend you choose a compute-optimized instance such as a c4 or c5.

A best practice when using a SageMaker notebook is to use the notebook instance to orchestrate other AWS services. For example, you can use the notebook instance to manage large dataset processing. To do this, make calls to AWS Glue for ETL (extract, transform, and load) services or Amazon EMR for mapping and data reduction using Hadoop. You can use AWS services as temporary forms of computation or storage for your data.

You can store and retrieve your training and test data using an Amazon Simple Storage Service bucket. You can then use SageMaker AI to train and build your model. As a result, the instance type of your notebook would have no bearing on the speed of your model training and testing.

After receiving the request, SageMaker AI does the following:

- **Creates a network interface**—If you choose the optional VPC configuration, SageMaker AI creates the network interface in your VPC. It uses the subnet ID that you provide in the request to determine which Availability Zone to create the subnet in. SageMaker AI associates the security group that you provide in the request with the subnet. For more information, see [Connect a Notebook Instance in a VPC to External Resources](#).
- **Launches an ML compute instance**—SageMaker AI launches an ML compute instance in a SageMaker AI VPC. SageMaker AI performs the configuration tasks that allow it to manage your notebook instance. If you specified your VPC, SageMaker AI enables traffic between your VPC and the notebook instance.
- **Installs Anaconda packages and libraries for common deep learning platforms**—SageMaker AI installs all of the Anaconda packages that are included in the installer. For more information, see [Anaconda package list](#). SageMaker AI also installs the TensorFlow and Apache MXNet deep learning libraries.
- **Attaches an ML storage volume**—SageMaker AI attaches an ML storage volume to the ML compute instance. You can use the volume as a working area to clean up the training dataset or to temporarily store validation, test, or other data. Choose any size between 5 GB and 16384 GB,

in 1 GB increments, for the volume. The default is 5 GB. ML storage volumes are encrypted, so SageMaker AI can't determine the amount of available free space on the volume. Because of this, you can increase the volume size when you update a notebook instance, but you can't decrease the volume size. If you want to decrease the size of the ML storage volume in use, create a new notebook instance with the desired size.

Only files and data saved within the `/home/ec2-user/SageMaker` folder persist between notebook instance sessions. Files and data that are saved outside this directory are overwritten when the notebook instance stops and restarts. Each notebook instance's `/tmp` directory provides a minimum of 10 GB of storage in an instance store. An instance store is temporary, block-level storage that isn't persistent. When the instance is stopped or restarted, SageMaker AI deletes the directory's contents. This temporary storage is part of the root volume of the notebook instance.

If the instance type used by the notebook instance has NVMe support, customers can use the NVMe instance store volumes available for that instance type. For instances with NVMe store volumes, all instance store volumes are automatically attached to the instance at launch. For more information about instance types and their associated NVMe store volumes, see the [Amazon Elastic Compute Cloud Instance Type Details](#).

To make the attached NVMe store volume available for your notebook instance, complete the steps in [Make instance store volumes available on your instance](#). Complete the steps with root access or by using a lifecycle configuration script.

 **Note**

NVMe instance store volumes are not persistent storage. This storage is short-lived with the instance and must be reconfigured every time an instance with this storage is launched.

- **Copies example Jupyter notebooks**— These Python code examples show model training and hosting exercises using different algorithms and training datasets.

### To create a SageMaker AI notebook instance:

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Choose **Notebook instances**, then choose **Create notebook instance**.
3. On the **Create notebook instance** page, provide the following information:

- a. For **Notebook instance name**, type a name for your notebook instance.
- b. For **Notebook instance type**, choose an instance size suitable for your use case. For a list of supported instance types and quotas, see [Amazon SageMaker AI Service Quotas](#).
- c. For **Platform Identifier**, choose a platform type to create the notebook instance on. This platform type dictates the Operating System and the JupyterLab version that your notebook instance is created with. For information about platform identifier type, see [Amazon Linux 2 notebook instances](#). For information about JupyterLab versions, see [JupyterLab versioning](#).

 **Important**

Notebook instances running on the JupyterLab 1 and JupyterLab 3 platforms will reach end of support on June 30, 2025. We strongly recommend migrating your work to the new JupyterLab 4 notebook instances before this date to ensure you have a secure and supported environment. For more information, see [JupyterLab version maintenance](#).

- d. (Optional) **Additional configuration** lets advanced users create a shell script that can run when you create or start the instance. This script, called a lifecycle configuration script, can be used to set the environment for the notebook or to perform other functions. For information, see [Customization of a SageMaker notebook instance using an LCC script](#).
- e. (Optional) **Additional configuration** also lets you specify the size, in GB, of the ML storage volume that is attached to the notebook instance. You can choose a size between 5 GB and 16,384 GB, in 1 GB increments. You can use the volume to clean up the training dataset or to temporarily store validation or other data.
- f. (Optional) For **Minimum IMDS Version**, select a version from the dropdown list. If this value is set to v1, both versions can be used with the notebook instance. If v2 is selected, then only IMDSv2 can be used with the notebook instance. For information about IMDSv2, see [Use IMDSv2](#).

 **Note**

Starting October 31, 2022, the default minimum IMDS Version for SageMaker notebook instances changes from IMDSv1 to IMDSv2.

Starting February 1, 2023, IMDSv1 is no longer be available for new notebook instance creation. After this date, you can create notebook instances with a minimum IMDS version of 2.

- g. For **IAM role**, choose either an existing IAM role in your account with the necessary permissions to access SageMaker AI resources or **Create a new role**. If you choose **Create a new role**, SageMaker AI creates an IAM role named `AmazonSageMaker-ExecutionRole-YYYYMMDDTHHmSS`. The AWS managed policy `AmazonSageMakerFullAccess` is attached to the role. The role provides permissions that allow the notebook instance to call SageMaker AI and Amazon S3.
- h. For **Root access**, to give root access for all notebook instance users, choose **Enable**. To remove root access for users, choose **Disable**. If you give root access, all notebook instance users have administrator privileges and can access and edit all files on it.
- i. (Optional) **Encryption key** lets you encrypt data on the ML storage volume attached to the notebook instance using an AWS Key Management Service (AWS KMS) key. If you plan to store sensitive information on the ML storage volume, consider encrypting the information.
- j. (Optional) **Network** lets you put your notebook instance inside a Virtual Private Cloud (VPC). A VPC provides additional security and limits access to resources in the VPC from sources outside the VPC. For more information on VPCs, see [Amazon VPC User Guide](#).

#### To add your notebook instance to a VPC:

- i. Choose the **VPC** and a **SubnetId**.
- ii. For **Security Group**, choose your VPC's default security group.
- iii. If you need your notebook instance to have internet access, enable direct internet access. For **Direct internet access**, choose **Enable**. Internet access can make your notebook instance less secure. For more information, see [Connect a Notebook Instance in a VPC to External Resources](#).
- k. (Optional) To associate Git repositories with the notebook instance, choose a default repository and up to three additional repositories. For more information, see [Git repositories with SageMaker AI Notebook Instances](#).
- l. Choose **Create notebook instance**.

In a few minutes, Amazon SageMaker AI launches an ML compute instance—in this case, a notebook instance—and attaches an ML storage volume to it. The notebook instance

has a preconfigured Jupyter notebook server and a set of Anaconda libraries. For more information, see the [CreateNotebookInstance API](#).

- When the status of the notebook instance is **InService**, in the console, the notebook instance is ready to use. Choose **Open Jupyter** next to the notebook name to open the classic Jupyter dashboard.

#### Note

To augment the security of your Amazon SageMaker notebook instance, all regional `notebook.region.sagemaker.aws` domains are registered in the internet [Public Suffix List \(PSL\)](#). For further security, we recommend that you use cookies with a `_Host-` prefix to set sensitive cookies for the domains of your SageMaker notebook instances. This helps to defend your domain against cross-site request forgery attempts (CSRF). For more information, see the [Set-Cookie](#) page in the [mozilla.org](#) developer documentation website.

You can choose **Open JupyterLab** to open the JupyterLab dashboard. The dashboard provides access to your notebook instance and sample SageMaker AI notebooks that contain complete code walkthroughs. These walkthroughs show how to use SageMaker AI to perform common machine learning tasks. For more information, see [Access example notebooks](#). For more information, see [Control root access to a SageMaker notebook instance](#).

For more information about Jupyter notebooks, see [The Jupyter notebook](#).

## Access Notebook Instances

#### Important

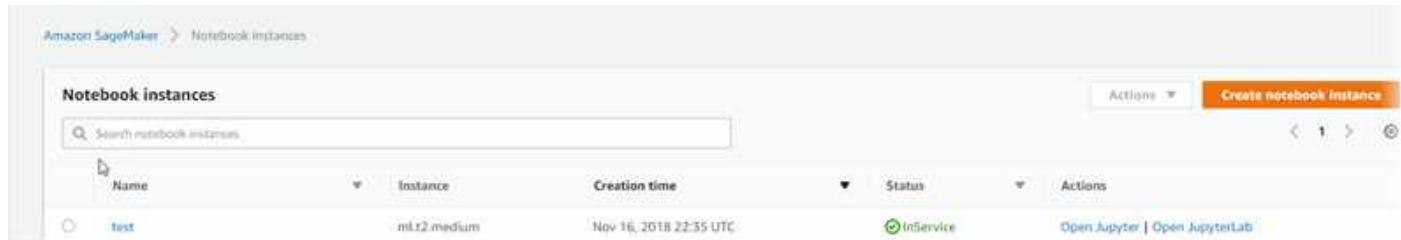
Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

To access your Amazon SageMaker notebook instances, choose one of the following options:

- Use the console.

Choose **Notebook instances**. The console displays a list of notebook instances in your account. To open a notebook instance with a standard Jupyter interface, choose **Open Jupyter** for that instance. To open a notebook instance with a JupyterLab interface, choose **Open JupyterLab** for that instance.



The console uses your sign-in credentials to send a [CreatePresignedNotebookInstanceUrl](#) API request to SageMaker AI. SageMaker AI returns the URL for your notebook instance, and the console opens the URL in another browser tab and displays the Jupyter notebook dashboard.

### Note

The URL that you get from a call to

[CreatePresignedNotebookInstanceUrl](#) is valid only for 5 minutes. If you try to use the URL after the 5-minute limit expires, you are directed to the AWS Management Console sign-in page.

- Use the API.

To get the URL for the notebook instance, call the

[CreatePresignedNotebookInstanceUrl](#) API and use the URL that the API returns to open the notebook instance.

Use the Jupyter notebook dashboard to create and manage notebooks and to write code. For more information about Jupyter notebooks, see <http://jupyter.org/documentation.html>.

## Update a Notebook Instance

After you create a notebook instance, you can update it using the SageMaker AI console and [UpdateNotebookInstance](#) API operation.

You can update the tags of a notebook instance that is `InService`. To update any other attribute of a notebook instance, its status must be `Stopped`.

### To update a notebook instance in the SageMaker AI console:

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Choose **Notebook instances**.
3. Choose the notebook instance that you want to update by selecting the notebook instance **Name** from the list.
4. If your notebook **Status** is not `Stopped`, select the **Stop** button to stop the notebook instance.

When you do this, the notebook instance status changes to `Stopping`. Wait until the status changes to `Stopped` to complete the following steps.

5. Select the **Edit** button to open the **Edit notebook instance** page. For information about the notebook properties you can update, see [Create an Amazon SageMaker notebook instance](#).
6. Update your notebook instance and select the **Update notebook instance** button at the bottom of the page when you are done to return to the notebook instances page. Your notebook instance status changes to `Updating`.

When the notebook instance update is complete, the status changes to `Stopped`.

## Customization of a SageMaker notebook instance using an LCC script

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio

and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

A *lifecycle configuration* (LCC) provides shell scripts that run only when you create the notebook instance or whenever you start one. When you create a notebook instance, you can create a new LCC or attach an LCC that you already have. Lifecycle configuration scripts are useful for the following use cases:

- Installing packages or sample notebooks on a notebook instance
- Configuring networking and security for a notebook instance
- Using a shell script to customize a notebook instance

You can also use a lifecycle configuration script to access AWS services from your notebook. For example, you can create a script that lets you use your notebook to control other AWS resources, such as an Amazon EMR instance.

We maintain a public repository of notebook lifecycle configuration scripts that address common use cases for customizing notebook instances at <https://github.com/aws-samples/amazon-sagemaker-notebook-instance-lifecycle-config-samples>.

### Note

Each script has a limit of 16384 characters.

The value of the \$PATH environment variable that is available to both scripts is /usr/local/sbin:/usr/local/bin:/usr/bin:/usr/sbin:/sbin:/bin. The working directory, which is the value of the \$PWD environment variable, is /.

View CloudWatch Logs for notebook instance lifecycle configurations in log group /aws/sagemaker/NotebookInstances in log stream [notebook-instance-name]/[LifecycleConfigHook].

Scripts cannot run for longer than 5 minutes. If a script runs for longer than 5 minutes, it fails and the notebook instance is not created or started. To help decrease the run time of scripts, try the following:

- Cut down on necessary steps. For example, limit which conda environments in which to install large packages.
- Run tasks in parallel processes.
- Use the nohup command in your script.

You can see a list of notebook instance lifecycle configurations you previously created by choosing **Lifecycle configuration** in the SageMaker AI console. You can attach a notebook instance LCC when you create a new notebook instance. For more information about creating a notebook instance, see [Create an Amazon SageMaker notebook instance](#).

## Create a lifecycle configuration script

The following procedure shows how to create a lifecycle configuration script for use with an Amazon SageMaker notebook instance. For more information about creating a notebook instance, see [Create an Amazon SageMaker notebook instance](#).

### To create a lifecycle configuration

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **Lifecycle configurations**.
4. From the **Lifecycle configurations** page, choose the **Notebook Instance** tab.
5. Choose **Create configuration**.
6. For **Name**, type a name using alphanumeric characters and "-", but no spaces. The name can have a maximum of 63 characters.
7. (Optional) To create a script that runs when you create the notebook and every time you start it, choose **Start notebook**.
8. In the **Start notebook** editor, type the script.
9. (Optional) To create a script that runs only once, when you create the notebook, choose **Create notebook**.
10. In the **Create notebook** editor, type the script configure networking.
11. Choose **Create configuration**.

## Lifecycle Configuration Best Practices

The following are best practices for using lifecycle configurations:

### Important

We do not recommend storing sensitive information in your lifecycle configuration script.

- Lifecycle configurations run as the `root` user. If your script makes any changes within the `/home/ec2-user/SageMaker` directory, (for example, installing a package with `pip`), use the command `sudo -u ec2-user` to run as the `ec2-user` user. This is the same user that Amazon SageMaker AI runs as.
- SageMaker AI notebook instances use conda environments to implement different kernels for Jupyter notebooks. If you want to install packages that are available to one or more notebook kernels, enclose the commands to install the packages with conda environment commands that activate the conda environment that contains the kernel where you want to install the packages.

For example, if you want to install a package only for the `python3` environment, use the following code:

```
#!/bin/bash
sudo -u ec2-user -i <<EOF

# This will affect only the Jupyter kernel called "conda_python3".
source activate python3

# Replace myPackage with the name of the package you want to install.
pip install myPackage
# You can also perform "conda install" here as well.

source deactivate

EOF
```

If you want to install a package in all conda environments in the notebook instance, use the following code:

```
#!/bin/bash
sudo -u ec2-user -i <<EOF
```

```
# Note that "base" is special environment name, include it there as well.  
for env in base /home/ec2-user/anaconda3/envs/*; do  
    source /home/ec2-user/anaconda3/bin/activate $(basename "$env")  
  
    # Installing packages in the Jupyter system environment can affect stability of  
    your SageMaker  
    # Notebook Instance. You can remove this check if you'd like to install Jupyter  
    extensions, etc.  
    if [ $env = 'JupyterSystemEnv' ]; then  
        continue  
    fi  
  
    # Replace myPackage with the name of the package you want to install.  
    pip install --upgrade --quiet myPackage  
    # You can also perform "conda install" here as well.  
  
    source /home/ec2-user/anaconda3/bin/deactivate  
done  
  
EOF
```

- You must store all conda environments in the default environments folder (/home/user/anaconda3/envs).

### **Important**

When you create or change a script, we recommend that you use a text editor that provides Unix-style line breaks, such as the text editor available in the console when you create a notebook. Copying text from a non-Linux operating system might introduce incompatible line breaks and result in an unexpected error.

## External library and kernel installation

### **Important**

Currently, all packages in notebook instance environments are licensed for use with Amazon SageMaker AI and do not require additional commercial licenses. However, this

might be subject to change in the future, and we recommend reviewing the licensing terms regularly for any updates.

Amazon SageMaker notebook instances come with multiple environments already installed. These environments contain Jupyter kernels and Python packages including: scikit, Pandas, NumPy, TensorFlow, and MXNet. These environments, along with all files in the sample-notebooks folder, are refreshed when you stop and start a notebook instance. You can also install your own environments that contain your choice of packages and kernels.

The different Jupyter kernels in Amazon SageMaker notebook instances are separate conda environments. For information about conda environments, see [Managing environments](#) in the *Conda* documentation.

Install custom environments and kernels on the notebook instance's Amazon EBS volume. This ensures that they persist when you stop and restart the notebook instance, and that any external libraries you install are not updated by SageMaker AI. To do that, use a lifecycle configuration that includes both a script that runs when you create the notebook instance (on-create) and a script that runs each time you restart the notebook instance (on-start). For more information about using notebook instance lifecycle configurations, see [Customization of a SageMaker notebook instance using an LCC script](#). There is a GitHub repository that contains sample lifecycle configuration scripts at [SageMaker AI Notebook Instance Lifecycle Config Samples](#).

The examples at <https://github.com/aws-samples/amazon-sagemaker-notebook-instance-lifecycle-config-samples/blob/master/scripts/persistent-conda-ebs/on-create.sh> and <https://github.com/aws-samples/amazon-sagemaker-notebook-instance-lifecycle-config-samples/blob/master/scripts/persistent-conda-ebs/on-start.sh> show the best practice for installing environments and kernels on a notebook instance. The on-create script installs the ipykernel library to create custom environments as Jupyter kernels, then uses pip install and conda install to install libraries. You can adapt the script to create custom environments and install libraries that you want. SageMaker AI does not update these libraries when you stop and restart the notebook instance, so you can ensure that your custom environment has specific versions of libraries that you want. The on-start script installs any custom environments that you create as Jupyter kernels, so that they appear in the dropdown list in the Jupyter New menu.

## Package installation tools

SageMaker notebooks support the following package installation tools:

- conda install
- pip install

You can install packages using the following methods:

- Lifecycle configuration scripts.

For example scripts, see [SageMaker AI Notebook Instance Lifecycle Config Samples](#). For more information on lifecycle configuration, see [Customize a Notebook Instance Using a Lifecycle Configuration Script](#).

- Notebooks – The following commands are supported.
  - `%conda install`
  - `%pip install`
- The Jupyter terminal – You can install packages using pip and conda directly.

From within a notebook you can use the system command syntax (lines starting with !) to install packages, for example, `!pip install` and `!conda install`. More recently, new commands have been added to IPython: `%pip` and `%conda`. These commands are the recommended way to install packages from a notebook as they correctly take into account the active environment or interpreter being used. For more information, see [Add %pip and %conda magic functions](#).

## Conda

Conda is an open source package management system and environment management system, which can install packages and their dependencies. SageMaker AI supports using Conda with either of the two main channels, the default channel, and the conda-forge channel. For more information, see [Conda channels](#). The conda-forge channel is a community channel where contributors can upload packages.

### Note

Due to how Conda resolves the dependency graph, installing packages from conda-forge can take significantly longer (in the worst cases, upwards of 10 minutes).

The Deep Learning AMI comes with many conda environments and many packages preinstalled. Due to the number of packages preinstalled, finding a set of packages that are guaranteed to

be compatible is difficult. You may see a warning "The environment is inconsistent, please check the package plan carefully". Despite this warning, SageMaker AI ensures that all the SageMaker AI provided environments are correct. SageMaker AI cannot guarantee that any user installed packages will function correctly.

### Note

Users of SageMaker AI, AWS Deep Learning AMIs and Amazon EMR can access the commercial Anaconda repository without taking a commercial license through February 1, 2024 when using Anaconda in those services. For any usage of the commercial Anaconda repository after February 1, 2024, customers are responsible for determining their own Anaconda license requirements.

Conda has two methods for activating environments: `conda activate/deactivate`, and `source activate/deactivate`. For more information, see [Should I use 'conda activate' or 'source activate' in Linux](#).

SageMaker AI supports moving Conda environments onto the Amazon EBS volume, which is persisted when the instance is stopped. The environments aren't persisted when the environments are installed to the root volume, which is the default behavior. For an example lifecycle script, see [persistent-conda-ebs](#).

### Supported conda operations (see note at the bottom of this topic)

- `conda install` of a package in a single environment
- `conda install` of a package in all environments
- `conda install` of a R package in the R environment
- Installing a package from the main conda repository
- Installing a package from conda-forge
- Changing the Conda install location to use EBS
- Supporting both `conda activate` and `source activate`

### Pip

Pip is the de facto tool for installing and managing Python packages. Pip searches for packages on the Python Package Index (PyPI) by default. Unlike Conda, pip doesn't have built in environment

support, and is not as thorough as Conda when it comes to packages with native/system library dependencies. Pip can be used to install packages in Conda environments.

You can use alternative package repositories with pip instead of the PyPi. For an example lifecycle script, see [on-start.sh](#).

### Supported pip operations (see note at the bottom of this topic)

- Using pip to install a package without an active conda environment (install packages system wide)
- Using pip to install a package in a conda environment
- Using pip to install a package in all conda environments
- Changing the pip install location to use EBS
- Using an alternative repository to install packages with pip

### Unsupported

SageMaker AI aims to support as many package installation operations as possible. However, if the packages were installed by SageMaker AI or DLAMI, and you use the following operations on these packages, it might make your notebook instance unstable:

- Uninstalling
- Downgrading
- Upgrading

We do not provide support for installing packages via yum install or installing R packages from CRAN.

Due to potential issues with network conditions or configurations, or the availability of Conda or PyPi, we cannot guarantee that packages will install in a fixed or deterministic amount of time.

#### Note

We cannot guarantee that a package installation will be successful. Attempting to install a package in an environment with incompatible dependencies can result in a failure. In such a case you should contact the library maintainer to see if it is possible to update the package dependencies. Alternatively you can attempt to modify the environment in such a way as

to allow the installation. This modification however will likely mean removing or updating existing packages, which means we can no longer guarantee stability of this environment.

## Notebook Instance Software Updates

Amazon SageMaker AI periodically tests and releases software that is installed on notebook instances. This includes:

- Kernel updates
- Security patches
- AWS SDK updates
- [Amazon SageMaker Python SDK](#) updates
- Open source software updates

To ensure that you have the most recent software updates, stop and restart your notebook instance, either in the SageMaker AI console or by calling

[StopNotebookInstance](#).

You can also manually update software installed on your notebook instance while it is running by using update commands in a terminal or in a notebook.

### Note

Updating kernels and some packages might depend on whether root access is enabled for the notebook instance. For more information, see [Control root access to a SageMaker notebook instance](#).

You can check the [Personal Health Dashboard](#) or the security bulletin at [Security Bulletins](#) for updates.

## Control an Amazon EMR Spark Instance Using a Notebook

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to

those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

You can use a notebook instance created with a custom lifecycle configuration script to access AWS services from your notebook. For example, you can create a script that lets you use your notebook with Sparkmagic to control other AWS resources, such as an Amazon EMR instance. You can then use the Amazon EMR instance to process your data instead of running the data analysis on your notebook. This allows you to create a smaller notebook instance because you won't use the instance to process data. This is helpful when you have large datasets that would require a large notebook instance to process the data.

The process requires three procedures using the Amazon SageMaker AI console:

- Create the Amazon EMR Spark instance
- Create the Jupyter Notebook
- Test the notebook-to-Amazon EMR connection

### To create an Amazon EMR Spark instance that can be controlled from a notebook using Sparkmagic

1. Open the Amazon EMR console at <https://console.aws.amazon.com/elasticmapreduce/>.
2. In the navigation pane, choose **Create cluster**.
3. On the **Create Cluster - Quick Options** page, under **Software configuration**, choose **Spark: Spark 2.4.4 on Hadoop 2.8.5 YARN with Ganglia 3.7.2 and Zeppelin 0.8.2**.
4. Set additional parameters on the page and then choose **Create cluster**.
5. On the **Cluster** page, choose the cluster name that you created. Note the **Master Public DNS**, the **EMR master's security group**, and the VPC name and subnet ID where the EMR cluster was created. You will use these values when you create a notebook.

## To create a notebook that uses Sparkmagic to control an Amazon EMR Spark instance

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. In the navigation pane, under **Notebook instances**, choose **Create notebook**.
3. Enter the notebook instance name and choose the instance type.
4. Choose **Additional configuration**, then, under **Lifecycle configuration**, choose **Create a new lifecycle configuration**.
5. Add the following code to the lifecycle configuration script:

```
# OVERVIEW
# This script connects an Amazon EMR cluster to an Amazon SageMaker notebook
# instance that uses Sparkmagic.
#
# Note that this script will fail if the Amazon EMR cluster's master node IP
# address is not reachable.
#   1. Ensure that the EMR master node IP is resolvable from the notebook instance.
#       One way to accomplish this is to have the notebook instance and the Amazon
#       EMR cluster in the same subnet.
#   2. Ensure the EMR master node security group provides inbound access from the
#       notebook instance security group.
#           Type      - Protocol - Port - Source
#           Custom TCP - TCP      - 8998 - $NOTEBOOK_SECURITY_GROUP
#   3. Ensure the notebook instance has internet connectivity to fetch the
#       SparkMagic example config.
#
# https://aws.amazon.com/blogs/machine-learning/build-amazon-sagemaker-notebooks-
# backed-by-spark-in-amazon-emr/
#
# PARAMETERS
EMR_MASTER_IP=your.emr.master.ip

cd /home/ec2-user/.sparkmagic

echo "Fetching Sparkmagic example config from GitHub..."
wget https://raw.githubusercontent.com/jupyter-incubator/sparkmagic/master/
sparkmagic/example_config.json

echo "Replacing EMR master node IP in Sparkmagic config..."
sed -i -- "s/localhost/$EMR_MASTER_IP/g" example_config.json
```

```
mv example_config.json config.json  
  
echo "Sending a sample request to Livy.."   
curl "$EMR_MASTER_IP:8998/sessions"
```

6. In the PARAMETERS section of the script, replace `your.emr.master.ip` with the Master Public DNS name for the Amazon EMR instance.
7. Choose **Create configuration**.
8. On the **Create notebook** page, choose **Network - optional**.
9. Choose the VPC and subnet where the Amazon EMR instance is located.
10. Choose the security group used by the Amazon EMR master node.
11. Choose **Create notebook instance**.

While the notebook instance is being created, the status is **Pending**. After the instance has been created and the lifecycle configuration script has successfully run, the status is **InService**.

 **Note**

If the notebook instance can't connect to the Amazon EMR instance, SageMaker AI can't create the notebook instance. The connection can fail if the Amazon EMR instance and notebook are not in the same VPC and subnet, if the Amazon EMR master security group is not used by the notebook, or if the Master Public DNS name in the script is incorrect.

### To test the connection between the Amazon EMR instance and the notebook

1. When the status of the notebook is **InService**, choose **Open Jupyter** to open the notebook.
2. Choose **New**, then choose **Sparkmagic (PySpark)**.
3. In the code cell, enter `%%info` and then run the cell.

The output should be similar to the following

```
Current session configs: {'driverMemory': '1000M', 'executorCores': 2, 'kind':  
'pyspark'}  
No active sessions.
```

## Access example notebooks

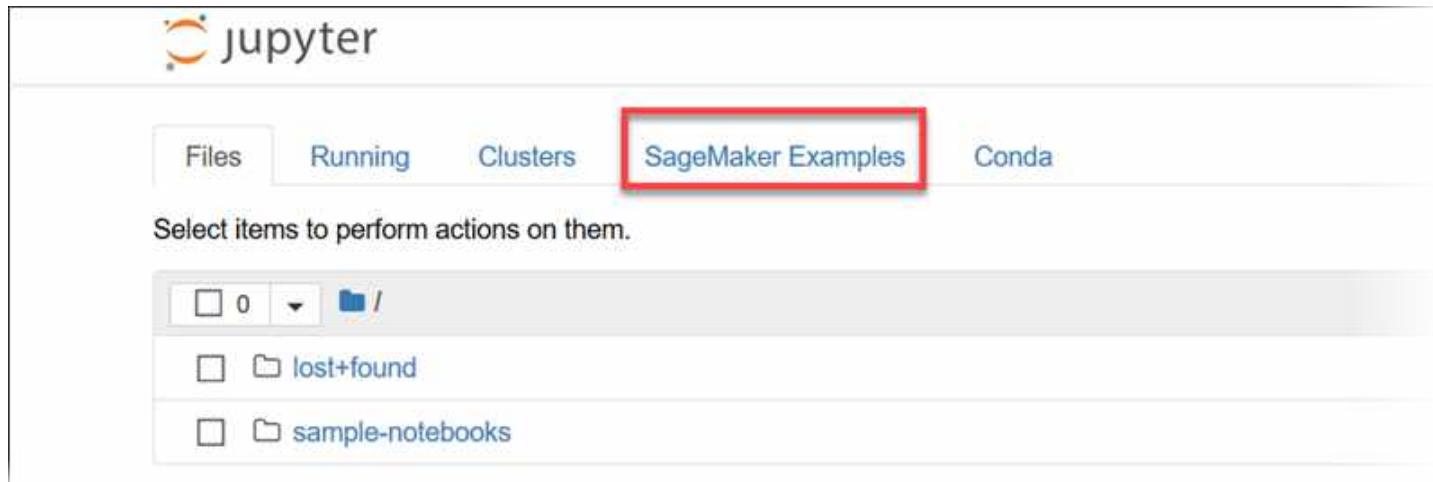
Your notebook instance contains example notebooks provided by Amazon SageMaker AI. The example notebooks contain code that shows how to apply machine learning solutions by using SageMaker AI. Notebook instances use the nbexamples Jupyter extension, which enables you to view a read-only version of an example notebook or create a copy of it that you can modify and run. For more information about the nbexamples extension, see <https://github.com/danielballan/nbexamples>. For information about example notebooks for SageMaker Studio, see [Use Amazon SageMaker Studio Classic Notebooks](#).

 **Note**

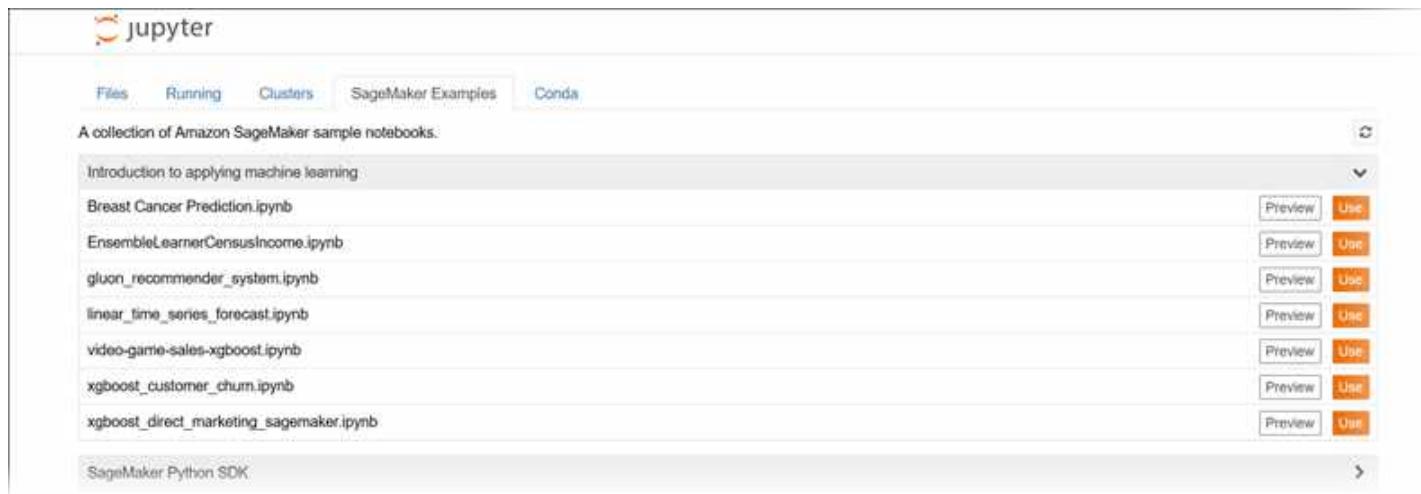
Example notebooks typically download datasets from the internet. If you disable SageMaker AI-provided internet access when you create your notebook instance, example notebooks might not work. For more information, see [Connect a Notebook Instance in a VPC to External Resources](#).

## Use or View Example Notebooks in Jupyter Classic

To view or use the example notebooks in the classic Jupyter view, choose the **SageMaker AI Examples** tab.



To view a read-only version of an example notebook in the Jupyter classic view, on the **SageMaker AI Examples** tab, choose **Preview** for that notebook. To create a copy of an example notebook in the home directory of your notebook instance, choose **Use**. In the dialog box, you can change the notebook's name before saving it.



## Use or View Example Notebooks in Jupyterlab

To view or use the example notebooks in the Jupyterlab view, choose the examples icon in the left navigation panel.

A COLLECTION OF AMAZON SAGEMAKER SAMPLE NOTEBOOKS

### INTRODUCTION TO APPLYING MACHINE LEARNING

- ▶ Breast Cancer Prediction.ipynb
- ▶ EnsembleLearnerCensusIncome.ipynb
- ▶ fair\_linear\_learner.ipynb
- ▶ gluon\_recommender\_system.ipynb
- ▶ linear\_time\_series\_forecast.ipynb
- ▶ ntm\_20newsgroups\_topic\_model.ipynb
- ▶ sagemaker-countycensusclustering.ipynb
- ▶ video-game-sales-xgboost.ipynb
- ▶ xgboost\_customer\_churn.ipynb
- ▶ xgboost\_direct\_marketing\_sagemaker.ipynb

### INTRODUCTION TO AMAZON ALGORITHMS

- ▶ blazingtext\_hosting\_pretrained\_fasttext.ipynb
- ▶ blazingtext\_text\_classification\_dbpedia.ipynb
- ▶ blazingtext\_word2vec\_subwords\_text8.ipynb
- ▶ blazingtext\_word2vec\_text8.ipynb
- ▶ deepar\_synthetic.ipynb
- ▶ DeepAR-Electricity.ipynb

To view a read-only version of an example notebook, choose the name of the notebook. This opens the notebook as a tab in the main area. To create a copy of an example notebook in the home directory of your notebook instance, choose **Create a Copy** in the top banner. In the dialog box, type a name for the notebook and then choose **CREATE COPY**.

For more information about the example notebooks, see the [SageMaker AI examples GitHub repository](#).

## Set the Notebook Kernel

Amazon SageMaker AI provides several kernels for Jupyter that provide support for Python 2 and 3, Apache MXNet, TensorFlow, and PySpark. To set a kernel for a new notebook in the Jupyter notebook dashboard, choose **New**, and then choose the kernel from the list. For more information about the available kernels, see [Available Kernels](#).



You can also create a custom kernel that you can use in your notebook instance. For information, see [External library and kernel installation](#).

## Git repositories with SageMaker AI Notebook Instances

Associate Git repositories with your notebook instance to save your notebooks in a source control environment that persists even if you stop or delete your notebook instance. You can associate one default repository and up to three additional repositories with a notebook instance. The repositories can be hosted in AWS CodeCommit, GitHub, or on any other Git server. Associating Git repositories with your notebook instance can be useful for:

- Persistence - Notebooks in a notebook instance are stored on durable Amazon EBS volumes, but they do not persist beyond the life of your notebook instance. Storing notebooks in a Git repository enables you to store and use notebooks even if you stop or delete your notebook instance.
- Collaboration - Peers on a team often work on machine learning projects together. Storing your notebooks in Git repositories allows peers working in different notebook instances to share notebooks and collaborate on them in a source-control environment.
- Learning - Many Jupyter notebooks that demonstrate machine learning techniques are available in publicly hosted Git repositories, such as on GitHub. You can associate your notebook instance with a repository to easily load Jupyter notebooks contained in that repository.

There are two ways to associate a Git repository with a notebook instance:

- Add a Git repository as a resource in your Amazon SageMaker AI account. Then, to access the repository, you can specify an AWS Secrets Manager secret that contains credentials. That way, you can access repositories that require authentication.
- Associate a public Git repository that is not a resource in your account. If you do this, you cannot specify credentials to access the repository.

## Topics

- [Add a Git repository to your Amazon SageMaker AI account](#)
- [Create a Notebook Instance with an Associated Git Repository](#)
- [Associate a CodeCommit Repository in a Different AWS Account with a Notebook Instance](#)
- [Use Git Repositories in a Notebook Instance](#)

## Add a Git repository to your Amazon SageMaker AI account

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

To manage your GitHub repositories, easily associate them with your notebook instances, and associate credentials for repositories that require authentication, add the repositories as resources in your Amazon SageMaker AI account. You can view a list of repositories that are stored in your account and details about each repository in the SageMaker AI console and by using the API.

You can add Git repositories to your SageMaker AI account in the SageMaker AI console or by using the AWS CLI.

**Note**

You can use the SageMaker AI API [CreateCodeRepository](#) to add Git repositories to your SageMaker AI account, but step-by-step instructions are not provided here.

## Add a Git repository to your SageMaker AI account (Console)

### To add a Git repository as a resource in your SageMaker AI account

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Under **Notebook**, choose **Git repositories**, then choose **Add repository**.
3. To add an CodeCommit repository, choose **AWS CodeCommit**. To add a GitHub or other Git-based repository, choose **GitHub/Other Git-based repo**.

### To add an existing CodeCommit repository

1. Choose **Use existing repository**.
2. For **Repository**, choose a repository from the list.
3. Enter a name to use for the repository in SageMaker AI. The name must be 1 to 63 characters. Valid characters are a-z, A-Z, 0-9, and - (hyphen).
4. Choose **Add repository**.

### To create a new CodeCommit repository

1. Choose **Create new repository**.
2. Enter a name for the repository that you can use in both CodeCommit and SageMaker AI. The name must be 1 to 63 characters. Valid characters are a-z, A-Z, 0-9, and - (hyphen).
3. Choose **Create repository**.

### To add a Git repository hosted somewhere other than CodeCommit

1. Choose **GitHub/Other Git-based repo**.
2. Enter a name of up to 63 characters. Valid characters include alpha-numeric characters, a hyphen (-), and 0-9.

3. Enter the URL for the repository. Do not provide a username in the URL. Add the sign-in credentials in AWS Secrets Manager as described in the next step.
4. For **Git credentials**, choose the credentials to use to authenticate to the repository. This is necessary only if the Git repository is private.

 **Note**

If you have two-factor authentication enabled for your Git repository, enter a personal access token generated by your Git service provider in the password field.

- a. To use an existing AWS Secrets Manager secret, choose **Use existing secret**, and then choose a secret from the list. For information about creating and storing a secret, see [Creating a Basic Secret](#) in the *AWS Secrets Manager User Guide*. The name of the secret you use must contain the string sagemaker.

 **Note**

The secret must have a staging label of AWSCURRENT and must be in the following format:

`{"username": UserName, "password": Password}`

For GitHub repositories, we recommend using a personal access token in the password field. For information, see <https://help.github.com/articles/creating-a-personal-access-token-for-the-command-line/>.

- b. To create a new AWS Secrets Manager secret, choose **Create secret**, enter a name for the secret, and then enter the sign-in credentials to use to authenticate to the repository. The name for the secret must contain the string sagemaker.

 **Note**

The IAM role you use to create the secret must have the `secretsmanager:GetSecretValue` permission in its IAM policy.

The secret must have a staging label of AWSCURRENT and must be in the following format:

`{"username": UserName, "password": Password}`

For GitHub repositories, we recommend using a personal access token.

- c. To not use any credentials, choose **No secret**.
5. Choose **Create secret**.

## Add a Git repository to your Amazon SageMaker AI account (CLI)

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

Use the `create-code-repository` AWS CLI command to add a Git repository to Amazon SageMaker AI to give users access to external resources. Specify a name for the repository as the value of the `code-repository-name` argument. The name must be 1 to 63 characters. Valid characters are a-z, A-Z, 0-9, and - (hyphen). Also specify the following:

- The default branch
- The URL of the Git repository

### Note

Do not provide a username in the URL. Add the sign-in credentials in AWS Secrets Manager as described in the next step.

- The Amazon Resource Name (ARN) of an AWS Secrets Manager secret that contains the credentials to use to authenticate the repository as the value of the `git-config` argument

For information about creating and storing a secret, see [Creating a Basic Secret](#) in the *AWS Secrets Manager User Guide*. The following command creates a new repository named `MyRepository`

in your Amazon SageMaker AI account that points to a Git repository hosted at <https://github.com/myprofile/my-repo>".

For Linux, OS X, or Unix:

```
aws sagemaker create-code-repository \
    --code-repository-name "MyRepository" \
    --git-config Branch=branch,RepositoryUrl=https://github.com/
myprofile/my-repo,SecretArn=arn:aws:secretsmanager:us-east-2:012345678901:secret:my-
secret-ABc0DE
```

For Windows:

```
aws sagemaker create-code-repository ^
    --code-repository-name "MyRepository" ^
    --git-config "{\"Branch\":\"master\", \"RepositoryUrl\" :
    \"https://github.com/myprofile/my-repo\", \"SecretArn\" :
    \"arn:aws:secretsmanager:us-east-2:012345678901:secret:my-secret-ABc0DE\"}"
```

### Note

The secret must have a staging label of AWSCURRENT and must be in the following format:

```
{"username": UserName, "password": Password}
```

For GitHub repositories, we recommend using a personal access token.

## Create a Notebook Instance with an Associated Git Repository

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

You can associate Git repositories with a notebook instance when you create the notebook instance by using the AWS Management Console, or the AWS CLI. If you want to use a CodeCommit repository that is in a different AWS account than the notebook instance, set up cross-account access for the repository. For information, see [Associate a CodeCommit Repository in a Different AWS Account with a Notebook Instance](#).

## Topics

- [Create a Notebook Instance with an Associated Git Repository \(Console\)](#)
- [Create a Notebook Instance with an Associated Git Repository \(CLI\)](#)

### Create a Notebook Instance with an Associated Git Repository (Console)

#### To create a notebook instance and associate Git repositories in the Amazon SageMaker AI console

1. Follow the instructions at [Create an Amazon SageMaker Notebook Instance for the tutorial](#).
2. For **Git repositories**, choose Git repositories to associate with the notebook instance.
  - a. For **Default repository**, choose a repository that you want to use as your default repository. SageMaker AI clones this repository as a subdirectory in the Jupyter startup directory at /home/ec2-user/SageMaker. When you open your notebook instance, it opens in this repository. To choose a repository that is stored as a resource in your account, choose its name from the list. To add a new repository as a resource in your account, choose **Add a repository to SageMaker AI (opens the Add repository flow in a new window)** and then follow the instructions at [Create a Notebook Instance with an Associated Git Repository \(Console\)](#). To clone a public repository that is not stored in your account, choose **Clone a public Git repository to this notebook instance only**, and then specify the URL for that repository.
  - b. For **Additional repository 1**, choose a repository that you want to add as an additional directory. SageMaker AI clones this repository as a subdirectory in the Jupyter startup directory at /home/ec2-user/SageMaker. To choose a repository that is stored as a resource in your account, choose its name from the list. To add a new repository as

a resource in your account, choose **Add a repository to SageMaker AI (opens the Add repository flow in a new window)** and then follow the instructions at [Create a Notebook Instance with an Associated Git Repository \(Console\)](#). To clone a repository that is not stored in your account, choose **Clone a public Git repository to this notebook instance only**, and then specify the URL for that repository.

Repeat this step up to three times to add up to three additional repositories to your notebook instance.

## Create a Notebook Instance with an Associated Git Repository (CLI)

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

To create a notebook instance and associate Git repositories by using the AWS CLI, use the `create-notebook-instance` command as follows:

- Specify the repository that you want to use as your default repository as the value of the `default-code-repository` argument. Amazon SageMaker AI clones this repository as a subdirectory in the Jupyter startup directory at `/home/ec2-user/SageMaker`. When you open your notebook instance, it opens in this repository. To use a repository that is stored as a resource in your SageMaker AI account, specify the name of the repository as the value of the `default-code-repository` argument. To use a repository that is not stored in your account, specify the URL of the repository as the value of the `default-code-repository` argument.
- Specify up to three additional repositories as the value of the `additional-code-repositories` argument. SageMaker AI clones this repository as a subdirectory in the Jupyter

startup directory at `/home/ec2-user/SageMaker`, and the repository is excluded from the default repository by adding it to the `.git/info/exclude` directory of the default repository. To use repositories that are stored as resources in your SageMaker AI account, specify the names of the repositories as the value of the `additional-code-repositories` argument. To use repositories that are not stored in your account, specify the URLs of the repositories as the value of the `additional-code-repositories` argument.

For example, the following command creates a notebook instance that has a repository named `MyGitRepo`, that is stored as a resource in your SageMaker AI account, as a default repository, and an additional repository that is hosted on GitHub:

```
aws sagemaker create-notebook-instance \
    --notebook-instance-name "MyNotebookInstance" \
    --instance-type "ml.t2.medium" \
    --role-arn "arn:aws:iam::012345678901:role/service-role/
AmazonSageMaker-ExecutionRole-20181129T121390" \
    --default-code-repository "MyGitRepo" \
    --additional-code-repositories "https://github.com/myprofile/my-
other-repo"
```

### Note

If you use an AWS CodeCommit repository that does not contain "SageMaker" in its name, add the `codecommit:GitPull` and `codecommit:GitPush` permissions to the role that you pass as the `role-arn` argument to the `create-notebook-instance` command. For information about how to add permissions to a role, see [Adding and Removing IAM Policies](#) in the *AWS Identity and Access Management User Guide*.

## Associate a CodeCommit Repository in a Different AWS Account with a Notebook Instance

To associate a CodeCommit repository in a different AWS account with your notebook instance, set up cross-account access for the CodeCommit repository.

## To set up cross-account access for a CodeCommit repository and associate it with a notebook instance:

1. In the AWS account that contains the CodeCommit repository, create an IAM policy that allows access to the repository from users in the account that contains your notebook instance. For information, see [Step 1: Create a Policy for Repository Access in AccountA](#) in the *CodeCommit User Guide*.
2. In the AWS account that contains the CodeCommit repository, create an IAM role, and attach the policy that you created in the previous step to that role. For information, see [Step 2: Create a Role for Repository Access in AccountA](#) in the *CodeCommit User Guide*.
3. Create a profile in the notebook instance that uses the role that you created in the previous step:
  - a. Open the notebook instance.
  - b. Open a terminal in the notebook instance.
  - c. Edit a new profile by typing the following in the terminal:

```
vi /home/ec2-user/.aws/config
```

- d. Edit the file with the following profile information:

```
[profile CrossAccountAccessProfile]
region = us-west-2
role_arn =
    arn:aws:iam::CodeCommitAccount:role/CrossAccountRepositoryContributorRole
credential_source=Ec2InstanceMetadata
output = json
```

Where *CodeCommitAccount* is the account that contains the CodeCommit repository, *CrossAccountAccessProfile* is the name of the new profile, and *CrossAccountRepositoryContributorRole* is the name of the role you created in the previous step.

4. On the notebook instance, configure git to use the profile you created in the previous step:
  - a. Open the notebook instance.
  - b. Open a terminal in the notebook instance.
  - c. Edit the Git configuration file typing the following in the terminal:

```
vi /home/ec2-user/.gitconfig
```

- d. Edit the file with the following profile information:

```
[credential]
    helper = !aws codecommit credential-helper --
profile CrossAccountAccessProfile $@
    UseHttpPath = true
```

Where *CrossAccountAccessProfile* is the name of the profile that you created in the previous step.

## Use Git Repositories in a Notebook Instance

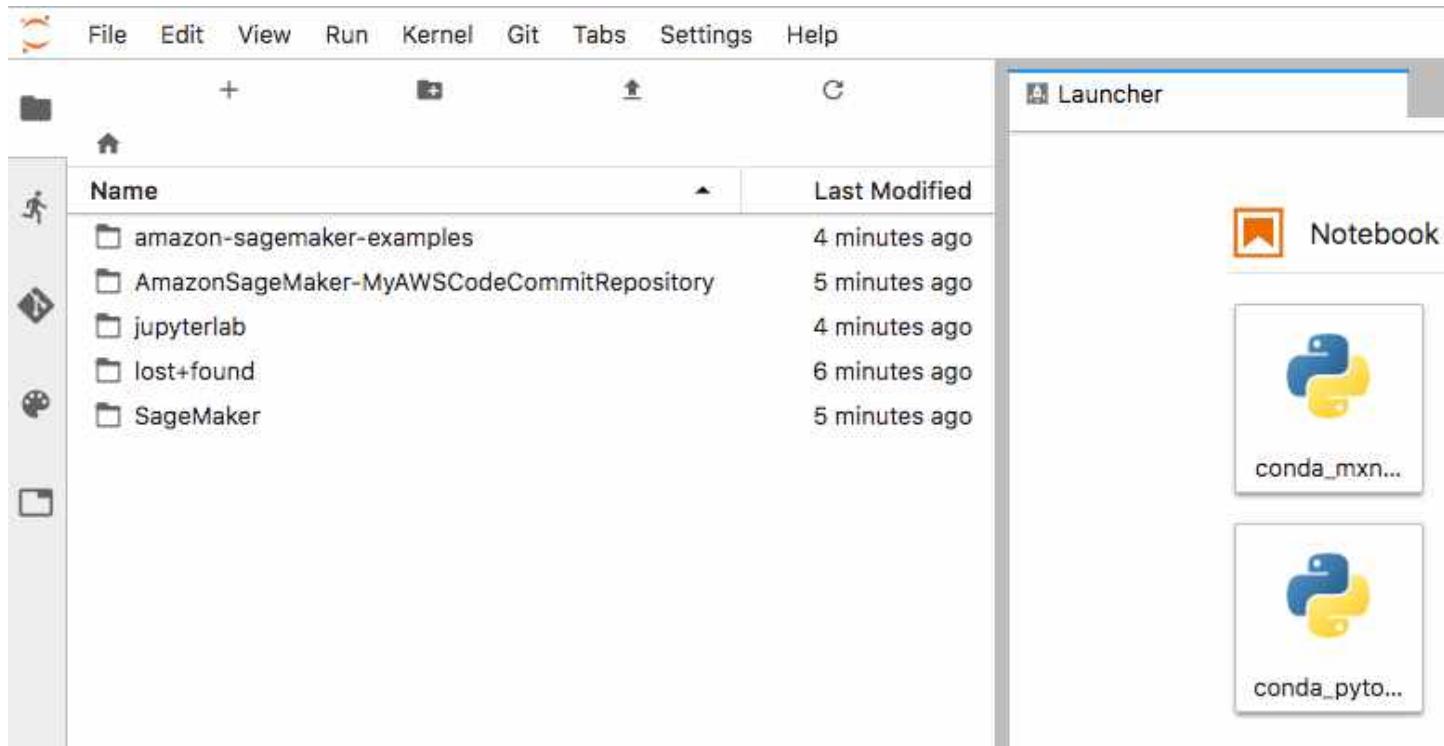
When you open a notebook instance that has Git repositories associated with it, it opens in the default repository, which is installed in your notebook instance directly under `/home/ec2-user/SageMaker`. You can open and create notebooks, and you can manually run Git commands in a notebook cell. For example:

```
!git pull origin master
```

To open any of the additional repositories, navigate up one folder. The additional repositories are also installed as directories under `/home/ec2-user/SageMaker`.

If you open the notebook instance with a JupyterLab interface, the `jupyter-git` extension is installed and available to use. For information about the `jupyter-git` extension for JupyterLab, see <https://github.com/jupyterlab/jupyterlab-git>.

When you open a notebook instance in JupyterLab, you see the git repositories associated with it on the left menu:



You can use the jupyter-git extension to manage git visually, instead of using the command line:

The screenshot shows the Jupyter Notebook interface with the jupyter-git extension active. The interface is divided into three main sections:

- Left Panel (File Status):** Shows the current state of the repository. It includes sections for **History**, **Staged(2)** (containing README.md and git.py), **Changed(1)** (containing package.json), and **Untracked(1)** (containing CONTRIBUTING.md).
- Middle Panel (Git Log):** Displays a detailed git log with commit history. Each commit includes the author, date, hash, and a brief description of the changes made.
- Right Panel (Notebook):** Shows an open notebook titled "Untitled.ipynb" with a code editor and a preview area.

Sample git log entries from the middle panel:

- John Doe 2a1ff23 18 minutes ago (working master origin/HEAD origin/master) Merge pull request #277 from JohnD-s/pull-frontend
- John Doe 23e3281 18 hours ago (upstream/master) Merge pull request #276 from JohnD-s/pull-frontend
- John Doe 862ffa5 19 hours ago (pull-frontend origin/pull-frontend) Unit tests for light/dark theme styles
- John Doe 0e52116 20 hours ago Dark theme icons for Git pull & push
- John Doe 4ad7e2c 21 hours ago Merge pull request #276 from JohnD-s/pull-frontend
- John Doe 4da7cf1 23 hours ago Fix titles on buttons
- John Doe 3356728 24 hours ago Merge pull request #275 from JohnD-s/tag-bugfix
- John Doe 61f99fe 2 days ago Fix typo in comment
- John Doe 960a6b2 4 days ago

## Notebook Instance Metadata

When you create a notebook instance, Amazon SageMaker AI creates a JSON file on the instance at the location `/opt/ml/metadata/resource-metadata.json` that contains the `ResourceName` and `ResourceArn` of the notebook instance. You can access this metadata from anywhere within the notebook instance, including in lifecycle configurations. For information about notebook instance lifecycle configurations, see [Customization of a SageMaker notebook instance using an LCC script](#).

 **Note**

The `resource-metadata.json` file can be modified with root access.

The `resource-metadata.json` file has the following structure:

```
{  
  "ResourceArn": "NotebookInstanceArn",  
  "ResourceName": "NotebookInstanceName"  
}
```

You can use this metadata from within the notebook instance to get other information about the notebook instance. For example, the following commands get the tags associated with the notebook instance:

```
NOTEBOOK_ARN=$(jq '.ResourceArn'  
                  /opt/ml/metadata/resource-metadata.json --raw-output)  
aws sagemaker list-tags --resource-arn $NOTEBOOK_ARN
```

The output looks like the following:

```
{  
  "Tags": [  
    {  
      "Key": "test",  
      "Value": "true"  
    }  
  ]  
}
```

## Monitor Jupyter Logs in Amazon CloudWatch Logs

Jupyter logs include important information such as events, metrics, and health information that provide actionable insights when running Amazon SageMaker notebooks. By importing Jupyter logs into CloudWatch Logs, customers can use CloudWatch Logs to detect anomalous behaviors, set alarms, and discover insights to keep the SageMaker AI notebooks running more smoothly. You can access the logs even when the Amazon EC2 instance that hosts the notebook is unresponsive, and use the logs to troubleshoot the unresponsive notebook. Sensitive information such as AWS account IDs, secret keys, and authentication tokens in presigned URLs are removed so that customers can share logs without leaking private information.

### To view Jupyter logs for a notebook instance:

1. Sign in to the AWS Management Console and open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Choose **Notebook instances**.
3. In the list of notebook instances, choose the notebook instance for which you want to view Jupyter logs by selecting the Notebook instance **Name**.  
This will bring you to the details page for that notebook instance.
4. Under **Monitor** on the notebook instance details page, choose **View logs**.
5. In the CloudWatch console, choose the log stream for your notebook instance. Its name is in the form *NotebookInstanceName/jupyter.log*.

For more information about monitoring CloudWatch logs for SageMaker AI, see [Log groups and streams that Amazon SageMaker AI sends to Amazon CloudWatch Logs](#).

## Amazon SageMaker Studio Lab

Amazon SageMaker Studio Lab is a free service that gives customers access to AWS compute resources, in an environment based on open-source JupyterLab. It is based on the same architecture and user interface as Amazon SageMaker Studio Classic, but with a subset of Studio Classic capabilities.

With Studio Lab, you can use AWS compute resources to create and run your Jupyter notebooks without signing up for an AWS account. Because Studio Lab is based on open-source JupyterLab, you can take advantage of open-source Jupyter extensions to run your Jupyter notebooks.

## Studio Lab compared to Amazon SageMaker Studio Classic

While Studio Lab provides free access to AWS compute resources, Amazon SageMaker Studio Classic provides the following advanced machine learning capabilities that Studio Lab does not support.

- Continuous integration and continuous delivery (Pipelines)
- Real-time predictions
- Large-scale distributed training
- Data preparation (Amazon SageMaker Data Wrangler)
- Data labeling (Amazon SageMaker Ground Truth)
- Feature Store
- Bias analysis (Clarify)
- Model deployment
- Model monitoring

Studio Classic also supports fine-grained access control and security by using AWS Identity and Access Management (IAM), Amazon Virtual Private Cloud (Amazon VPC), and AWS Key Management Service (AWS KMS). Studio Lab does not support these Studio Classic features, nor does it support the use of estimators and built-in SageMaker AI algorithms.

To export your Studio Lab projects for use with Studio Classic, see [Export an Amazon SageMaker Studio Lab environment to Amazon SageMaker Studio Classic](#).

The following topics give information about Studio Lab and how to use it

### Topics

- [Amazon SageMaker Studio Lab components overview](#)
- [Onboard to Amazon SageMaker Studio Lab](#)
- [Manage your account](#)
- [Launch your Amazon SageMaker Studio Lab project runtime](#)
- [Use Amazon SageMaker Studio Lab starter assets](#)
- [Studio Lab pre-installed environments](#)
- [Use the Amazon SageMaker Studio Lab project runtime](#)
- [Troubleshooting](#)

# Amazon SageMaker Studio Lab components overview

Amazon SageMaker Studio Lab consists of the following components. The following topics give more details about these components.

## Topics

- [Landing page](#)
- [Studio Lab account](#)
- [Project overview page](#)
- [Preview page](#)
- [Project](#)
- [Compute instance type](#)
- [Project runtime](#)
- [Session](#)

## Landing page

You can request an account and sign in to an existing account on your landing page. To navigate to the landing page, see the [Amazon SageMaker Studio Lab website](#). For more information about creating a Studio Lab account, see [Onboard to Amazon SageMaker Studio Lab](#).

The following screenshot shows the Studio Lab landing page interface for requesting a user account and signing in.

[Sign in](#)[Request account](#)

## Learn and experiment with machine learning

Quickly create data analytics, scientific computing, and machine learning projects with notebooks in your browser.

[Request free account](#)[Watch video](#)

## Studio Lab account

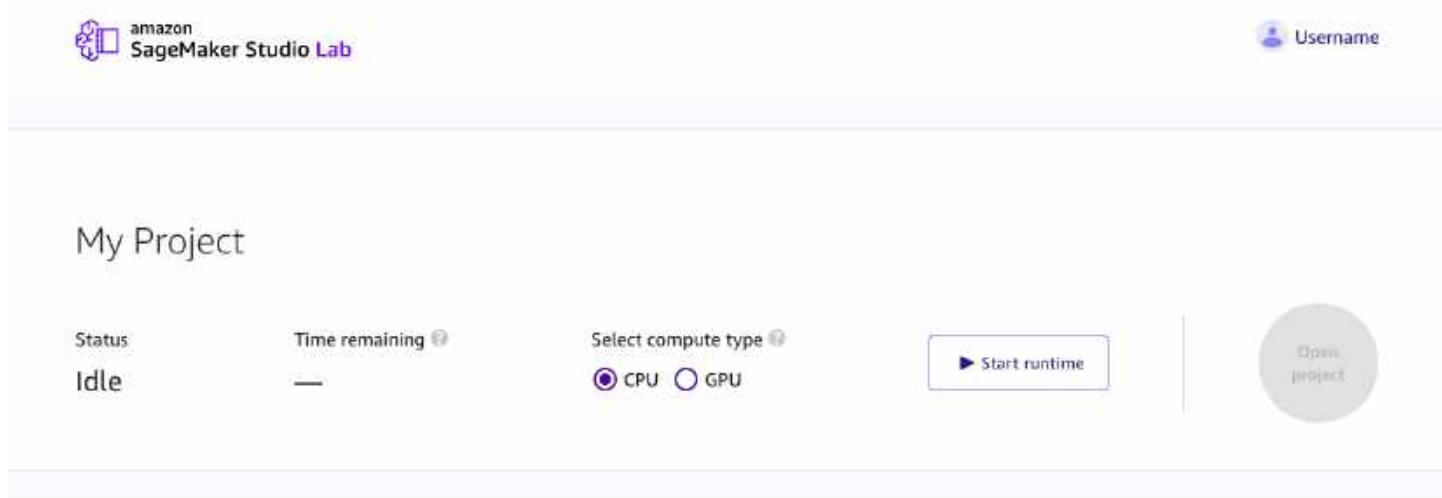
Your Studio Lab account gives you access to Studio Lab. For more information about creating a user account, see [Onboard to Amazon SageMaker Studio Lab](#).

## Project overview page

You can launch a compute instance and view information about your project on this page. To navigate to this page, you must sign in from the [Amazon SageMaker Studio Lab website](#). The URL takes the following format.

```
https://studiolab.sagemaker.aws/users/<YOUR_USER_NAME>
```

The following screenshot shows a project overview in the Studio Lab user interface.



## Preview page

On this page, you can access a read-only preview of a Jupyter notebook. You can not execute the notebook from preview, but you can copy that notebook into your project. For many customers, this may be the first Studio Lab page that customers see, as they may be opening a notebook from GitHub notebook. For more information on how to use GitHub resources, see [Use GitHub resources](#).

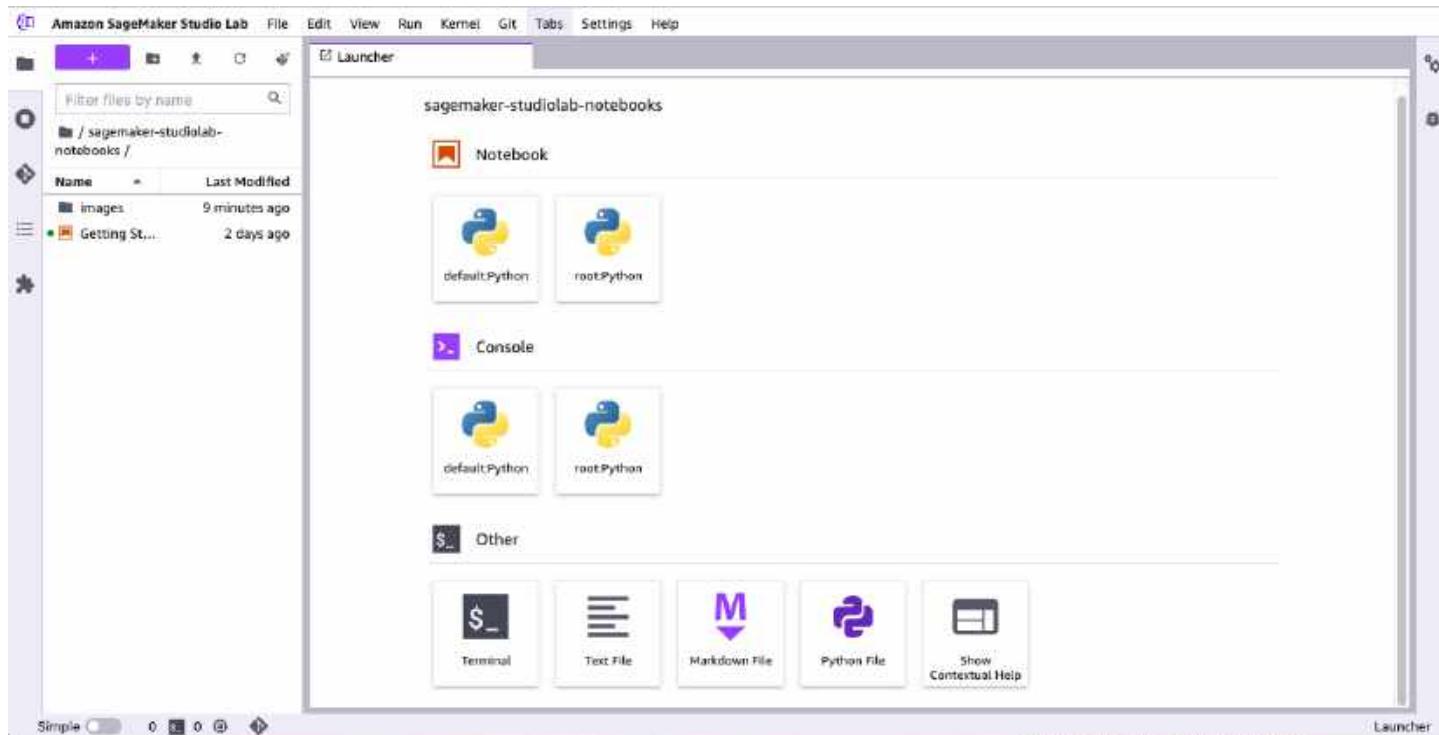
To copy the notebook preview to your Studio Lab project:

1. Sign in to your Studio Lab account. For more information about creating a Studio Lab account, see [Onboard to Amazon SageMaker Studio Lab](#).
2. Under **Notebook compute instance**, choose a compute instance type. For more information about compute instance types, see [Compute instance type](#).
3. Choose **Start runtime**. You might be asked to solve a CAPTCHA puzzle. For more information on CAPTCHA, see [What is a CAPTCHA puzzle?](#)
4. One time setup, for first time starting runtime using your Studio Lab account:
  - a. Enter a mobile phone number to associate with your Amazon SageMaker Studio Lab account and choose **Continue**.  
For information on supported countries and regions, see [Supported countries and regions \(SMS channel\)](#).
5. Choose **Copy to project**.

## Project

Your project contains all of your files and folders, including your Jupyter notebooks. You have full control over the files in your project. Your project also includes the JupyterLab-based user interface. From this interface, you can interact with your Jupyter notebooks, edit your source code files, integrate with GitHub, and connect to Amazon S3. For more information, see [Use the Amazon SageMaker Studio Lab project runtime](#).

The following screenshot shows a Studio Lab project with the file browser open and the Studio Lab Launcher displayed.



## Compute instance type

Your Amazon SageMaker Studio Lab project runtime is based on an EC2 instance. You are allotted 15 GB of storage and 16 GB of RAM. Availability of compute instances is not guaranteed and is subject to demand. If you require additional storage or compute resources, consider switching to Studio.

Amazon SageMaker Studio Lab offers the choice of a CPU (Central Processing Unit) and a GPU (Graphical Processing Unit). The following sections give information about these two options, including selection guidance.

### CPU

A central processing unit (CPU) is designed to handle a wide range of tasks efficiently, but is limited in how many tasks it can run concurrently. For machine learning, a CPU is recommended for compute intensive algorithms, such as time series, forecasting, and tabular data.

The CPU compute type has up to 4 hours at a time with a limit of 8 hours in a 24-hour period.

## GPU

A graphics processing unit (GPU) is designed to render high-resolution images and video concurrently. A GPU is recommended for deep learning tasks, especially for transformers and computer vision.

The GPU compute type has up to 4 hours at a time with a limit of 4 hours in a 24-hour period.

## Compute time

When compute time for Studio Lab reaches its time limit, the instance stops all running computations. Studio Lab does not support time limit increases.

Studio Lab automatically saves your environment when you update your environment and every time you create a new file. Custom-installed extensions and packages persist even after your runtime has ended.

File edits are periodically saved, but are not saved when your runtime ends. To ensure that you do not lose your progress, save your work manually. If you have content in your Studio Lab project that you don't want to lose, we recommend that you back up your content elsewhere. For more information about exporting your environment and files, see [Export an Amazon SageMaker Studio Lab environment to Amazon SageMaker Studio Classic](#).

During long computation, you do not need to keep your project open. For example, you can start training a model, then close your browser. The instance keeps running for up to the compute type limit in a 24-hour period. You can then sign in later to continue your work.

We recommend that you use checkpointing in your deep learning jobs. You can use saved checkpoints to restart a job from the previously saved checkpoint. For more information, see [File I/O](#).

## Project runtime

The project runtime is the period of time when your compute instance is running.

## Session

A user session begins every time you launch your project.

## Onboard to Amazon SageMaker Studio Lab

To onboard to Amazon SageMaker Studio Lab, follow the steps in this guide. In the following sections, you learn how to request a Studio Lab account, create your account, and sign in.

### Topics

- [Request a Studio Lab account](#)
- [Create a Studio Lab account](#)
- [Sign in to Studio Lab](#)

### Request a Studio Lab account

To use Studio Lab, you must first request approval to create a Studio Lab account. An AWS account cannot be used for onboarding to Studio Lab.

The following steps show how to request a Studio Lab account.

1. Navigate to the [Studio Lab landing page](#).
2. Select **Request account**.
3. Enter the required information into the form.
4. Select **Submit request**.
5. If you receive an email to verify your email address, follow the instructions in the email to complete this step.

Your account request must be approved before you can register for a Studio Lab account. Your request will be reviewed within five business days. When your account request is approved, you receive an email with a link to the Studio Lab account registration page. This link expires seven days after your request is approved. If the link expires, you must submit a new account request.

Note: Your account request is denied if your email has been associated with activity that violates our [Terms of Service](#) or other agreements.

## Referral codes

Studio Lab referral codes enable new account requests to be automatically approved to support machine learning events like workshops, hackathons, and classes. With a referral code, a trusted host can get their participants immediate access to Studio Lab. After an account has been created using a referral code, the account continues to exist after the expiration of the code.

To get a referral code, contact [Sales Support](#). To use a referral code, enter the code as part of the account request form.

## Create a Studio Lab account

After your request is approved, complete the following steps to create your Studio Lab account.

1. Select **Create account** in the account request approval email to open a new page.
2. From the new page, enter your **Email**, a **Password**, and a **Username**.
3. Select **Create account**.

You might be asked to solve a CAPTCHA puzzle. For more information on CAPTCHA, see [What is a CAPTCHA puzzle?](#)

## Sign in to Studio Lab

After you register for your account, you can sign in to Studio Lab.

1. Navigate to the [Studio Lab landing page](#).
2. Select **Sign in** to open a new page.
3. Enter your **Email** or **Username** and **Password**.
4. Select **Sign in** to open a new page to your project.

You might be asked to solve a CAPTCHA puzzle. For more information on CAPTCHA, see [What is a CAPTCHA puzzle?](#)

## Manage your account

The following topic gives information about managing your account, including changing your password, deleting your account, and getting information that we have collected. These topics

require that you sign in to your Amazon SageMaker Studio Lab account. For more information, see [Sign in to Studio Lab](#).

## Change your password

Follow these steps to change your Amazon SageMaker Studio Lab password.

1. Navigate to the Studio Lab project overview page. The URL takes the following format.

```
https://studiolab.sagemaker.aws/users/<YOUR_USER_NAME>
```

2. From the top-right corner, select your user name to open a dropdown menu.
3. From the dropdown menu, select **Change password** to open a new page.
4. Enter your current password into the **Enter your current password** field.
5. Enter your new password into the **Create a new password** and **Confirm your new password** fields.
6. Select **Submit**.

## Delete your account

Follow these steps to delete your Studio Lab account.

1. Navigate to the Studio Lab project overview page. The URL takes the following format.

```
https://studiolab.sagemaker.aws/users/<YOUR_USER_NAME>
```

2. From the top-right corner, select your user name to open a dropdown menu.
3. From the dropdown menu, select **Delete account** to open a new page.
4. Enter your password to confirm the deletion of your Studio Lab account.
5. Select **Delete**.

## Customer information

Studio Lab collects your email address, user name, encrypted password, project files, and metadata. When requesting an account, you can optionally choose to provide your first and last name, country, organization name, occupation, and the reason for your interest in this product.

We protect all customer personal data with encryption. For more information about how your personal information is handled, see the [Privacy Notice](#).

When you delete your account, all of your information is deleted immediately. If you have an inquiry about this, submit the [Amazon SageMaker Studio Lab Form](#). For information and support related to AWS compliance, see [Compliance support](#).

## Launch your Amazon SageMaker Studio Lab project runtime

The Amazon SageMaker Studio Lab project runtime lets you write and run code directly from your browser. It is based on JupyterLab and has an integrated terminal and console. For more information about JupyterLab, see the [JupyterLab Documentation](#).

The following topic gives information about how to manage your project runtime. These topics require that you sign in to your Amazon SageMaker Studio Lab account. For more information about signing in, see [Sign in to Studio Lab](#). For more information about your project, see [Amazon SageMaker Studio Lab components overview](#).

### Topics

- [Start your project runtime](#)
- [Stop your project runtime](#)
- [View remaining compute time](#)
- [Change your compute type](#)

## Start your project runtime

To use Studio Lab, you must start your project runtime. This runtime gives you access to the JupyterLab environment.

1. Navigate to the Studio Lab project overview page. The URL takes the following format.

```
https://studiolab.sagemaker.aws/users/<YOUR_USER_NAME>
```

2. Under **My Project**, select a compute type. For more information about compute types, see [Compute instance type](#).
3. Select **Start runtime**.

You might be asked to solve a CAPTCHA puzzle. For more information on CAPTCHA, see [What is a CAPTCHA puzzle?](#)

4. One time setup, for first time starting runtime using your Studio Lab account:
  - a. Enter a mobile phone number to associate with your Amazon SageMaker Studio Lab account and choose **Continue**.  
For information on supported countries and regions, see [Supported countries and regions \(SMS channel\)](#).
  - b. Enter the 6-digit code sent to the associated mobile phone number and choose **Verify**.
5. After the runtime is running, select **Open project** to open the project runtime environment in a new browser tab.

## Stop your project runtime

When you stop your project runtime, your files are not automatically saved. To ensure that you don't lose your work, save all of your changes before stopping your project runtime.

- Under **My Project**, select **Stop runtime**.

## View remaining compute time

Your project runtime has limited compute time based on the compute type that you select. For more information about compute time in Studio Lab, see [Compute instance type](#).

- Under **My Project**, view **Time remaining**.

## Change your compute type

You can switch your compute type based on your workflow. For more information about compute types, see [Compute instance type](#).

1. Save any project files before changing the compute type.
2. Navigate to the Studio Lab project overview page. The URL takes the following format.

```
https://studiolab.sagemaker.aws/users/<YOUR_USER_NAME>
```

3. Under **My Project**, select the desired compute type (CPU or GPU).
4. Confirm your choice by selecting **Restart** in the **Restart project runtime?** dialog box. Studio Lab stops your current project runtime, then starts a new project runtime with your updated compute type.
5. After your project runtime has started, select **Open project**. This opens your project runtime environment in a new browser tab. For information about using your project runtime environment, see [Use the Amazon SageMaker Studio Lab project runtime](#).

## Use Amazon SageMaker Studio Lab starter assets

Amazon SageMaker Studio Lab supports the following assets to help machine learning (ML) practitioners get started. This guide shows you how to clone notebooks for your project.

### Getting started notebook

Studio Lab comes with a starter notebook that gives general information and guides you through key workflows. When you launch your project runtime for the first time, this notebook automatically opens.

### Dive into Deep Learning

Dive into Deep Learning (D2L) is an interactive, open-source book that teaches the ideas, mathematical theory, and code that power machine learning. With over 150 Jupyter notebooks, D2L provides a comprehensive overview of deep learning principles. For more information about D2L, see the [D2L website](#).

The following procedure shows how to clone the D2L Jupyter notebooks to your instance.

1. Start and open the Studio Lab project runtime environment by following [Start your project runtime](#).
2. Once Studio Lab is open, choose the Git tab  
 on the left sidebar.  
)
3. Choose **Clone a Repository**.

If you do not see the **Clone a Repository** option, this may be because you are currently in a Git repository. Instead, use the following substeps.

- a. Choose the Folder tab  
 )  
on the left sidebar.
  - b. Beneath the file search bar, choose the folder icon to the left of the currently selected repository. When you hover over the folder icon, you will see the user directory (/home/studio-lab-user).
  - c. Once you are in the user directory, choose the Git tab on the left sidebar.
  - d. Choose **Clone a Repository**.
4. Under **Git repository URL (.git)** you will be asked to provide a URL.
  5. On a new browser tab, navigate to your Studio Lab project overview page. The URL takes the following format.

```
https://studiolab.sagemaker.aws/users/<YOUR_USER_NAME>
```

6. Under **New to machine learning?**, choose **Dive into Deep Learning**.
7. From the new **Dive into Deep Learning** browser tab, choose **GitHub** to open a new page with the example notebooks.
8. Choose **Code** and copy the GitHub repository's URL in the **HTTPS** tab.
9. Return to the Studio Lab open project browser tab, paste the D2L repository URL, and clone the repository.

## AWS Machine Learning University

The AWS Machine Learning University (MLU) provides access to the machine learning courses used to train Amazon's own developers. With AWS MLU, any developer can learn how to use machine learning with the learn-at-your-own-pace MLU Accelerator learning series. The MLU Accelerator series is designed to help developers begin their ML journey. It offers three-day foundational courses on these three subjects: Natural Language Processing, Tabular Data, and Computer Vision. For more information, see [Machine Learning University](#).

The following procedure shows how to clone the AWS MLU Jupyter notebooks to your instance.

1. Start and open the Studio Lab project runtime environment by following [Start your project runtime](#).

2. Once Studio Lab is open, choose the Git tab



)

on the left sidebar.

3. Choose **Clone a Repository**.

If you do not see the **Clone a Repository** option, this may be because you are currently in a Git repository. Instead, use the following substeps.

a. Choose the Folder tab



)

on the left sidebar.

b. Beneath the file search bar, choose the folder icon to the left of the currently selected repository. When you hover over the folder icon, you will see the user directory (/home/studio-lab-user).

c. Once you are in the user directory, choose the Git tab on the left sidebar.

d. Choose **Clone a Repository**.

4. Under **Git repository URL (.git)** you will be asked to provide a URL.

5. On a new browser tab, navigate to your Studio Lab project overview page. The URL takes the following format.

`https://studiolab.sagemaker.aws/users/<YOUR_USER_NAME>`

6. Under **New to machine learning?**, choose **AWS Machine Learning University**.

7. From the new **AWS Machine Learning University** browser tab, find a course that interests you by reading the **Course Summary** for each course.

8. Choose the corresponding GitHub repository of interest under **Course Content**, to open a new page with the example notebooks.

9. Choose **Code** and copy the GitHub repository's URL in the **HTTPS** tab.

10. Return to the Studio Lab open project browser tab, paste the MLU repository URL, and choose **Clone** to clone the repository.

## Roboflow

Roboflow gives you the tools to train, fine-tune, and label objects for computer vision applications. For more information, see <https://roboflow.com/>.

The following procedure shows how to clone the Roboflow Jupyter notebooks to your instance.

1. Navigate to the Studio Lab project overview page. The URL takes the following format.

```
https://studiolab.sagemaker.aws/users/<YOUR_USER_NAME>
```

2. Under **Resources and community**, find **Make AI Generated Images**.
3. Under **Make AI Generated Images** choose **Open notebook**.
4. Follow the tutorial under the Notebook preview.

## Studio Lab pre-installed environments

Amazon SageMaker Studio Lab uses conda environments to manage packages (or libraries) for your projects. This guide explains what conda environments are, how to interact with them, and the different pre-installed environments available in Studio Lab.

A conda environment is a directory that contains a collection of packages you have installed. It allows you to create isolated environments with specific package versions, preventing conflicts between projects with different dependencies.

You can interact with conda environments in Studio Lab in two ways:

- Terminal: Use the terminal to create, activate, and manage environments.
- JupyterLab Notebook: When opening a JupyterLab notebook, select the kernel with the environment name you wish to use, to use the packages installed in that environment.

For a walkthrough on managing environments, see [Manage your environment](#)

Studio Lab comes with several pre-installed environments that are either persistent or non-persistent memory environments. Any changes made to persistent memory environments will remain for your next session. Any changes to non-persistent memory environments will not remain for your next sessions, but the packages within will be updated and tested for compatibility by Amazon SageMaker AI. Here's an overview of each environment and its use case:

- **sagemaker-distribution**: A non-persistent environment managed by Amazon SageMaker AI. It contains popular packages for machine learning, data science, and visualization. This environment is regularly updated and tested for compatibility. Use this environment if you want a fully-managed setup with common packages pre-installed.

The `sagemaker-distribution` environment is closely related to the environment used in Amazon SageMaker Studio Classic, so after graduating from Studio Lab to Studio Classic the notebooks should run similarly. For information on exporting your environment from Studio Lab to Studio Classic, see [Export an Amazon SageMaker Studio Lab environment to Amazon SageMaker Studio Classic](#).

- `default`: A persistent environment with minimal packages pre-installed. Use this environment if you want to customize it significantly by installing additional packages.
- `studiolab`: A persistent environment where JupyterLab and related packages installed. Use this environment for configuring the JupyterLab user interface and installing Jupyter server extensions.
- `studiolab-safemode`: A non-persistent environment activated automatically when there's an issue with your project runtime. Use this environment for troubleshooting purposes. For information on troubleshooting, see [Troubleshooting](#).
- `base`: A non-persistent environment used for system tooling. This environment is not intended for customer use.

To view the packages in an environment, run the command `conda list`.

For more information on installing packages within your environment, see [Customize your environment](#).

If you plan to graduate from Studio Lab to Amazon SageMaker Studio Classic, see [Export an Amazon SageMaker Studio Lab environment to Amazon SageMaker Studio Classic](#).

For information on SageMaker images and their versions, see [Amazon SageMaker images available for use with Studio Classic](#).

## Use the Amazon SageMaker Studio Lab project runtime

The following topics give information about using the Amazon SageMaker Studio Lab project runtime. Before you can use the Studio Lab project runtime, you must onboard to Studio Lab by following the steps in [Onboard to Amazon SageMaker Studio Lab](#).

### Topics

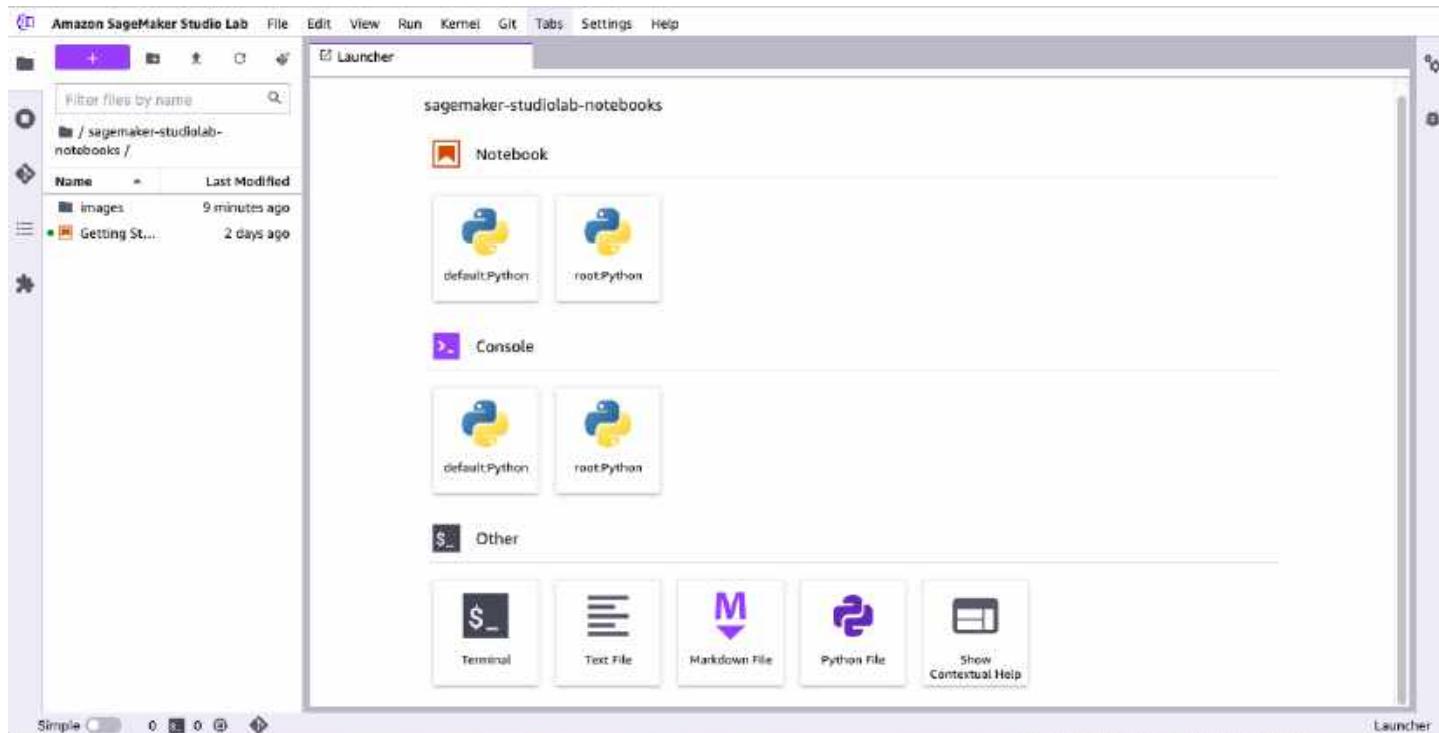
- [Amazon SageMaker Studio Lab UI overview](#)
- [Create or open an Amazon SageMaker Studio Lab notebook](#)

- [Use the Amazon SageMaker Studio Lab notebook toolbar](#)
- [Manage your environment](#)
- [Use external resources in Amazon SageMaker Studio Lab](#)
- [Get notebook differences](#)
- [Export an Amazon SageMaker Studio Lab environment to Amazon SageMaker Studio Classic](#)
- [Shut down Studio Lab resources](#)

## Amazon SageMaker Studio Lab UI overview

Amazon SageMaker Studio Lab extends the JupyterLab interface. Previous users of JupyterLab will notice similarities between the JupyterLab and Studio Lab UI, including the workspace. For an overview of the basic JupyterLab interface, see [The JupyterLab Interface](#).

The following image shows Studio Lab with the file browser open and the Studio Lab Launcher displayed.



You will find the *menu bar* at the top of the screen. The *left sidebar* contains icons to open file browsers, resource browsers, and tools. The *status bar* is located at the bottom-left corner of Studio Lab.

The main work area is divided horizontally into two panes. The left pane is the *file and resource browser*. The right pane contains one or more tabs for resources, such as notebooks and terminals.

## Topics

- [Left sidebar](#)
- [File and resource browser](#)
- [Main work area](#)

## Left sidebar

The left sidebar includes the following icons. When you hover over an icon, a tooltip displays the icon name. When you choose an icon, the file and resource browser displays the described functionality. For hierarchical entries, a selectable breadcrumb at the top of the browser shows your location in the hierarchy.

Icon	Description
	<p><b>File Browser</b></p> <p>Choose the <b>Upload Files</b> icon ( ) to add files to Studio Lab.</p> <p>Double-click a file to open the file in a new tab.</p> <p>To have adjacent files open, choose a tab that contains a notebook, Python, or text file, and then choose <b>New View for File</b>.</p> <p>Choose the plus (+) sign on the menu at the top of the file browser to open the Studio Lab Launcher.</p>
	<p><b>Running Terminals and Kernels</b></p> <p>You can see a list of all of the running terminals and kernels in your project. For more information, see <a href="#">Shut down Studio Lab resources</a>.</p>
	<p><b>Git</b></p>

Icon	Description
	You can connect to a Git repository and then access a full range of Git tools and operations. For more information, see <a href="#">Use external resources in Amazon SageMaker Studio Lab</a> .
	<b>Table of Contents</b> You can access the Table of Contents for your current Jupyter notebook.
	<b>Extension Manager</b> You can enable and manage third-party JupyterLab extensions.

## File and resource browser

The file and resource browser shows lists of your notebooks and files. On the menu at the top of the file browser, choose the plus (+) sign to open the Studio Lab Launcher. The Launcher allows you to create a notebook or open a terminal.

## Main work area

The main work area has multiple tabs that contain your open notebooks and terminals.

## Create or open an Amazon SageMaker Studio Lab notebook

When you create a notebook in Amazon SageMaker Studio Lab or open a notebook in Studio Lab, you must select a kernel for the notebook. The following topics describe how to create and open notebooks in Studio Lab.

For information about shutting down the notebook, see [Shut down Studio Lab resources](#).

## Topics

- [Open a Studio Lab notebook](#)
- [Create a notebook from the file menu](#)
- [Create a notebook from the Launcher](#)

## Open a Studio Lab notebook

Studio Lab can only open notebooks listed in the Studio Lab file browser. To clone a notebook into your file browser from an external repository, see [Use external resources in Amazon SageMaker Studio Lab](#).

### To open a notebook

1. In the left sidebar, choose the **File Browser** icon  to display the file browser.
2. Browse to a notebook file and double-click it to open the notebook in a new tab.

### Create a notebook from the file menu

#### To create a notebook from the File menu

1. From the Studio Lab menu, choose **File**, choose **New**, and then choose **Notebook**.
2. To use the default kernel, in the **Select Kernel** dialog box, choose **Select**. Otherwise, to select a different kernel, use the dropdown menu.

### Create a notebook from the Launcher

#### To create a notebook from the Launcher

1. Open the Launcher by using the keyboard shortcut **Ctrl + Shift + L**. Alternatively, you can open Launcher from the left sidebar: Choose the **File Browser** icon, and then choose the plus (+) icon.
2. To use the default kernel from the Launcher, under **Notebook**, choose **default:Python**. Otherwise, select a different kernel.

After you choose the kernel, your notebook launches and opens in a new Studio Lab tab.

To view the notebook's kernel session, in the left sidebar, choose the **Running Terminals and Kernels** icon .

You can stop the notebook's kernel session from this view.

## Use the Amazon SageMaker Studio Lab notebook toolbar

Amazon SageMaker Studio Lab notebooks extend the JupyterLab interface. For an overview of the basic JupyterLab interface, see [The JupyterLab Interface](#).

The following image shows the toolbar and an empty cell from a Studio Lab notebook.



When you hover over a toolbar icon, a tooltip displays the icon function. You can find additional notebook commands in the Studio Lab main menu. The toolbar includes the following icons:

Icon	Description
	<b>Save and checkpoint</b> Saves the notebook and updates the checkpoint file.
	<b>Insert cell</b> Inserts a code cell below the current cell. The current cell is noted by the blue vertical marker in the left margin.
	<b>Cut, copy, and paste cells</b> Cuts, copies, and pastes the selected cells.
	<b>Run cells</b> Runs the selected cells. The cell that follows the last-selected cell becomes the new-selected cell.
	<b>Interrupt kernel</b> Interrupts the kernel, which cancels the currently-running operation. The kernel remains active.
	<b>Restart kernel</b>

Icon	Description
	Restarts the kernel. Variables are reset. Unsaved information is not affected.
	<b>Restart kernel and re-run notebook</b>  Restarts the kernel. Variables are reset. Unsaved information is not affected. Then re-runs the entire notebook.
<b>Code</b>	<b>Cell type</b>  Displays or changes the current cell type. The cell types are: <ul style="list-style-type: none"><li>• Code – Code that the kernel runs.</li><li>• Markdown – Text rendered as markdown.</li><li>• Raw – Content, including Markdown markup, that's displayed as text.</li></ul>
	<b>Checkpoint diff</b>  Opens a new tab that displays the difference between the notebook and the checkpoint file. For more information, see <a href="#">Get notebook differences</a> .
	<b>Git diff</b>  Only enabled if the notebook is opened from a Git repository. Opens a new tab that displays the difference between the notebook and the last Git commit. For more information, see <a href="#">Get notebook differences</a> .
<b>default</b>	<b>Kernel</b>  Displays or changes the kernel that processes the cells in the notebook.  No Kernel indicates that the notebook was opened without specifying a kernel. You can edit the notebook, but you can't run any cells.

Icon	Description
	<p><b>Kernel busy status</b></p> <p>Displays a kernel's busy status by showing the circle's edge and its interior as the same color. The kernel is busy when it is starting and when it is processing cells. Additional kernel states are displayed in the status bar at the bottom-left corner of Studio Lab.</p>

## Manage your environment

Amazon SageMaker Studio Lab provides pre-installed environments for your Studio Lab notebook instances. Environments allow you to start up a Studio Lab notebook instance with the packages you want to use. This is done by installing packages in the environment and then selecting the environment as a Kernel.

Studio Lab has various environments pre-installed for you. You will typically want to use the `sagemaker-distribution` environment if you want to use a fully managed environment that already contains many popular packages used for machine learning (ML) engineers and data scientists. Otherwise you can use the default environment if you want persistent customization for your environment. For more information on the available pre-installed Studio Lab environments, see [Studio Lab pre-installed environments](#).

You can customize your environment by adding new packages (or libraries) to it. You can also create new environments from Studio Lab, import compatible environments, reset your environment to create space, and more.

The following commands are for running in a Studio Lab terminal. However, while installing packages it is highly recommended to install them within your Studio Lab Jupyter notebook. This ensures that the packages are installed in the intended environment. To run the commands in a Jupyter notebook, prefix the command with a % before running the cell. For example, the code snippet `pip list` in a terminal is the same as `%pip list` in a Jupyter notebook.

The following sections give information about your default conda environment, how to customize it, and how to add and remove conda environments. For a list of sample environments that you can install into Studio Lab, see [Creating Custom conda Environments](#). To use these sample environment YAML files with Studio Lab, see [Step 4: Install your Studio Lab conda environments in Studio Classic](#).

## Topics

- [Your default environment](#)
- [View environments](#)
- [Create, activate, and use new conda environments](#)
- [Using sample Studio Lab environments](#)
- [Customize your environment](#)
- [Refresh Studio Lab](#)

### Your default environment

Studio Lab uses conda environments to encapsulate the software packages that are needed to run notebooks. Your project contains a default conda environment, named `default`, with the [IPython kernel](#). This environment serves as the default kernel for your Jupyter notebooks.

### View environments

To view the environments in Studio Lab you can use a terminal or Jupyter notebook. The following command will be for a Studio Lab terminal. If you wish to run the corresponding commands in a Jupyter notebook, see [Manage your environment](#).

Open the Studio Lab terminal by opening the **File Browser** panel



),

choose the plus (+) sign on the menu at the top of the file browser to open the **Launcher**, then choose **Terminal**. From the Studio Lab terminal, list the conda environments by running the following.

```
conda env list
```

This command outputs a list of the conda environments and their locations in the file system. When you onboard to Studio Lab, you automatically activate the `studiolab` conda environment. The following is an example of listed environments after you onboard.

```
# conda environments:  
#  
default          /home/studio-lab-user/.conda/envs/default  
studiolab       * /home/studio-lab-user/.conda/envs/studiolab
```

studiolab-safemode	/opt/amazon/sagemaker/safemode-home/.conda/envs/studiolab-
safemode	
base	/opt/conda
sagemaker-distribution	/opt/conda/envs/sagemaker-distribution

The \* marks the activated environment.

## Create, activate, and use new conda environments

If you would like to maintain multiple environments for different use cases, you can create new conda environments in your project. The following sections show how to create and activate new conda environments. For a Jupyter notebook that shows how to create a custom environment, see [Setting up a Custom Environment in SageMaker Studio Lab](#).

### Note

Maintaining multiple environments counts against your available Studio Lab memory.

## Create conda environment

To create a conda environment, run the following conda command from your terminal. This example creates a new environment with Python 3.9.

```
conda create --name <ENVIRONMENT_NAME> python=3.9
```

Once the conda environment is created, you can view the environment in your environment list. For more information on how to view your environment list, see [View environments](#).

## Activate a conda environment

To activate any conda environment, run the following command in the terminal.

```
conda activate <ENVIRONMENT_NAME>
```

When you run this command, any packages installed using conda or pip are installed in the environment. For more information on installing packages, see [Customize your environment](#).

## Use a conda environment

1. To use your new conda environments with notebooks, make sure the `ipykernel` package is installed in the environment.

```
conda install ipykernel
```

2. Once the `ipykernel` package is installed in the environment, you can select the environment as the kernel for your notebook.

You may need to restart JupyterLab to see the environment available as a kernel. This can be done by choosing **Amazon SageMaker Studio Lab** in the top menu of your Studio Lab open project, and choosing **Restart JupyterLab....**

3. You can choose the kernel for an existing notebook or when you create a new one.

- For an existing notebook: open the notebook and choose the current kernel from the right side of the top menu. You can choose the kernel you wish to use from the drop-down menu.
- For a new notebook: open the Studio Lab launcher and choose the kernel under **Notebook**. This will open the notebook with the kernel you choose.

For an overview of the Studio Lab UI, see [Amazon SageMaker Studio Lab UI overview](#).

## Using sample Studio Lab environments

Studio Lab provides sample custom environments through the [SageMaker Studio Lab Examples](#) repository. The following shows how to clone and build these environments.

1. Clone the SageMaker Studio Lab Examples GitHub repository by following the instructions in [Use GitHub resources](#).
2. In Studio Lab choose the **File Browser** icon  on the left menu, so that the **File Browser** panel shows on the left.
3. Navigate to the `studio-lab-examples/custom-environments` directory in the File Browser.
4. Open the directory for the environment that you want to build.
5. Right click the `.yml` file in the folder, then select **Build conda Environment**.
6. You can now use the environment as a kernel after your conda environment has finished building. For instructions on how to use an existing environment as a kernel, see [Create, activate, and use new conda environments](#)

## Customize your environment

You can customize your environment by installing and removing extensions and packages as needed. Studio Lab comes with environments with packages pre-installed and using an existing environment may save you time and memory, as pre-installed packages do not count against your available Studio Lab memory. For more information on the available pre-installed Studio Lab environments, see [Studio Lab pre-installed environments](#).

Any installed extensions and packages installed on your default environment will persist in your project. That is, you do not need to install your packages for every project runtime session. However, extensions and packages installed on your `sagemaker-distribution` environment will not persist, so you will need to install new packages during your next session. Thus, it is highly recommended to install packages within your notebook to ensure that the packages are installed in the intended environment.

To view your environments, run the command `conda env list`.

To activate your environment, run the command `conda activate <ENVIRONMENT_NAME>`.

To view the packages in an environment, run the command `conda list`.

## Install packages

It is highly recommended to install your packages within your Jupyter notebook to ensure that your packages are installed in the intended environment. To install additional packages to your environment from a Jupyter notebook, run one of the following commands in a cell within your Jupyter notebook. These commands install packages in the currently activated environment.

- `%conda install <PACKAGE>`
- `%pip install <PACKAGE>`

We don't recommend using the `!pip` or `!conda` commands because they can behave in unexpected ways when you have multiple environments.

After you install new packages to your environment, you may need to restart the kernel to ensure that the packages work in your notebook. This can be done by choosing **Amazon SageMaker Studio Lab** in the top menu of your Studio Lab open project and choosing **Restart JupyterLab....**

## Remove packages

To remove a package, run the command

```
%conda remove <PACKAGE_NAME>
```

This command will also remove any package that depends on `<PACKAGE_NAME>`, unless a replacement can be found without that dependency.

To remove all of the packages in an environment, run the command

```
conda deactivate  
&& conda env remove --name  
<ENVIRONMENT_NAME>
```

## Refresh Studio Lab

To refresh Studio Lab, remove all of your environments and files.

1. List all conda environments.

```
conda env list
```

2. Activate the base environment.

```
conda activate base
```

3. Remove each environment in the list of conda environments, besides base.

```
conda remove --name <ENVIRONMENT_NAME> --all
```

4. Delete all of the files on your Studio Lab.

```
rm -rf .*
```

## Use external resources in Amazon SageMaker Studio Lab

With Amazon SageMaker Studio Lab, you can integrate external resources, such as Jupyter notebooks and data, from Git repositories and Amazon S3. You can also add an **Open in Studio Lab** button to your GitHub repo and notebooks. This button lets you clone your notebooks directly from Studio Lab.

The following topics show how to integrate external resources.

## Topics

- [Use GitHub resources](#)
- [Add an Open in Studio Lab button to your notebook](#)
- [Import files from your computer](#)
- [Connect to Amazon S3](#)

### Use GitHub resources

Studio Lab offers integration with GitHub. With this integration, you can clone notebooks and repositories directly to your Studio Lab project.

The following topics give information about how to use GitHub resources with Studio Lab.

#### Studio Lab sample notebooks

To get started with a repository of sample notebooks tailored for Studio Lab, see [Studio Lab Sample Notebooks](#).

This repository provides notebooks for the following use cases and others.

- Computer vision
- Connecting to AWS
- Creating custom environments
- Geospatial data analysis
- Natural language processing
- Using R

#### Clone a GitHub repo

To clone a GitHub repo to your Studio Lab project, follow these steps.

1. Start your Studio Lab project runtime. For more information on launching Studio Lab project runtime, see [Start your project runtime](#).
2. In Studio Lab, choose the **File Browser** icon  on the left menu, so that the **File Browser** panel shows on the left.

3. Navigate to your user directory by choosing the file icon beneath the file search bar.
4. Select the **Git** icon  from the left menu to open a new dropdown menu.
5. Choose **Clone a Repository**.
6. Paste the repository's URL under **Git repository URL (.git)**.
7. Select **Clone**.

## Clone individual notebooks from GitHub

To open a notebook in Studio Lab, you must have access to the repo that the notebook is in. The following examples describe Studio Lab permission-related behavior in various situations.

- If a repo is public, you can automatically clone the notebook into your project from the Studio Lab preview page.
- If a repo is private, you are prompted to sign in to GitHub from the Studio Lab preview page. If you have access to a private repo, you can clone the notebook into your project.
- If you don't have access to a private repo, you cannot clone the notebook from the Studio Lab preview page.

The following sections show two options for you to copy a GitHub notebook in your Studio Lab project. These options depend on whether the notebook has an **Open in Studio Lab** button.

### Option 1: Copy notebook with an Open in Studio Lab button

The following procedure shows how to copy a notebook that has an **Open in Studio Lab** button. If you want to add this button to your notebook, see [Add an Open in Studio Lab button to your notebook](#).

1. Sign in to Studio Lab following the steps in [Sign in to Studio Lab](#).
2. In a new browser tab, navigate to the GitHub notebook that you want to clone.
3. In the notebook, select the **Open in Studio Lab** button to open a new page in Studio Lab with a preview of the notebook.
4. If your project runtime is not already running, start it by choosing the **Start runtime** button at the top of the preview page. Wait for the runtime to start before proceeding to the next step.

5. After your project runtime has started, select **Copy to project** to open your project runtime in a new browser tab.
6. In the **Copy from GitHub?** dialog box, select **Copy notebook only**. This copies the notebook file to your project.

## Option 2: Clone any GitHub notebook

The following procedure shows how to copy any notebook from GitHub.

1. Navigate to the notebook in GitHub.
2. In the browser's address bar, modify the notebook URL, as follows.

```
# Original URL  
https://github.com/<PATH_TO_NOTEBOOK>  
  
# Modified URL  
https://studiolab.sagemaker.aws/import/github/<PATH_TO_NOTEBOOK>
```

3. Navigate to the modified URL. This opens a preview of the notebook in Studio Lab.
4. If your project runtime is not already running, start it by choosing the **Start runtime** button at the top of the preview page. Wait for the runtime to start before proceeding to the next step.
5. After your project runtime has started, select **Copy to project** to open your project runtime in a new browser tab.
6. In the **Copy from GitHub?** dialog box, select **Copy notebook only** to copy the notebook file to your project.

## Add an Open in Studio Lab button to your notebook

When you add the **Open in Studio Lab** button to your notebooks, others can clone your notebooks or repositories directly to their Studio Lab projects. If you are sharing your notebook within a public GitHub repository, your content will be publicly readable. Do not share private content, such as AWS access keys or AWS Identity and Access Management credentials, in your notebook.

To add the functional **Open in Studio Lab** button to your Jupyter notebook or repository, add the following markdown to the top of your notebook or repository.

```
[![Open In SageMaker Studio Lab](https://studiolab.sagemaker.aws/studiolab.svg)]  
(https://studiolab.sagemaker.aws/import/github/<PATH_TO_YOUR_NOTEBOOK_ON_GITHUB>)
```

## Import files from your computer

The following steps show how to import files from your computer to your Studio Lab project.

1. Open the Studio Lab project runtime.
2. Open the **File Browser** panel.
3. In the actions bar of the **File Browser** panel, select the **Upload Files** button.
4. Select the files that you want to upload from your local machine.
5. Select **Open**.

Alternatively, you can drag and drop files from your computer into the **File Browser** panel.

## Connect to Amazon S3

The AWS CLI enables AWS integration in your Studio Lab project. With this integration, you can pull resources from Amazon S3 to use with your Jupyter notebooks.

To use AWS CLI with Studio Lab, complete the following steps. For a notebook that outlines this integration, see [Using Studio Lab with AWS Resources](#).

1. Install the AWS CLI following the steps in [Installing or updating the latest version of the AWS CLI](#).
2. Configure your AWS credentials by following the steps in [Quick setup](#). The role for your AWS account must have permissions to access the Amazon S3 bucket that you are copying data from.
3. From your Jupyter notebook, clone resources from the Amazon S3 bucket, as needed. The following command shows how to clone all resources from an Amazon S3 path to your project. For more information, see the [AWS CLI Command Reference](#).

```
!aws s3 cp s3://<BUCKET_NAME>/<PATH_TO_RESOURCES>/ <PROJECT_DESTINATION_PATH>/ --recursive
```

## Get notebook differences

You can display the difference between the current notebook and the last checkpoint, or the last Git commit, using the Amazon SageMaker Studio Lab project UI.

## Topics

- [Get the difference between the last checkpoint](#)
- [Get the difference between the last commit](#)

### Get the difference between the last checkpoint

When you create a notebook, a hidden checkpoint file that matches the notebook is created. You can view changes between the notebook and the checkpoint file, or revert the notebook to match the checkpoint file.

To save the Studio Lab notebook and update the checkpoint file to match: Choose the **Save notebook and create checkpoint** icon



).

This is located on the Studio Lab menu's left side. The keyboard shortcut for **Save notebook and create checkpoint** is **Ctrl + s**.

To view changes between the Studio Lab notebook and the checkpoint file: Choose the **Checkpoint diff** icon



),

located in the center of the Studio Lab menu.

To revert the Studio Lab notebook to the checkpoint file: On the main Studio Lab menu, choose **File**, and then **Revert Notebook to Checkpoint**.

### Get the difference between the last commit

If a notebook is opened from a Git repository, you can view the difference between the notebook and the last Git commit.

To view the changes in the notebook from the last Git commit: Choose the **Git diff** icon



)

in the center of the notebook menu.

## Export an Amazon SageMaker Studio Lab environment to Amazon SageMaker Studio Classic

Amazon SageMaker Studio Classic offers many features for machine learning and deep learning work flows that are unavailable in Amazon SageMaker Studio Lab. This page shows how to

migrate a Studio Lab environment to Studio Classic to take advantage of more compute capacity, storage, and features. However, you may want to familiarize yourself with Studio Classic's prebuilt containers, which are optimized for the full MLOP pipeline. For more information, see [Amazon SageMaker Studio Lab](#)

To migrate your Studio Lab environment to Studio Classic, you must first onboard to Studio Classic following the steps in [Amazon SageMaker AI domain overview](#).

## Topics

- [Step 1: Export your Studio Lab conda environment](#)
- [Step 2: Save your Studio Lab artifacts](#)
- [Step 3: Import your Studio Lab artifacts to Studio Classic](#)
- [Step 4: Install your Studio Lab conda environments in Studio Classic](#)

### Step 1: Export your Studio Lab conda environment

You can export a conda environment and add libraries or packages to the environment by following the steps in [Manage your environment](#). The following example demonstrates using the default environment to be exported to Studio Classic.

1. Open the Studio Lab terminal by opening the **File Browser** panel



),

choose the plus (+) sign on the menu at the top of the file browser to open the **Launcher**, then choose **Terminal**. From the Studio Lab terminal, list the conda environments by running the following.

```
conda env list
```

This command outputs a list of the conda environments and their locations in the file system. When you onboard to Studio Lab, you automatically activate the studiolab conda environment.

```
# conda environments: #
    default                  /home/studio-lab-user/.conda/envs/default
    studiolab                * /home/studio-lab-user/.conda/envs/studiolab
    studiolab-safemode       /opt/amazon/sagemaker/safemode-home/.conda/
envs/studiolab-safemode
```

base

/opt/conda

We recommend that you do not export the studiolab, studiolab-safemode, and base environments. These environments are not usable in Studio Classic for the following reasons:

- studiolab: This sets up the JupyterLab environment for Studio Lab. Studio Lab runs a different major version of JupyterLab than Studio Classic, so it is not usable in Studio Classic.
  - studiolab-safemode: This also sets up the JupyterLab environment for Studio Lab. Studio Lab runs a different major version of JupyterLab than Studio Classic, so it is not usable in Studio Classic.
  - base: This environment comes with conda by default. The base environment in Studio Lab and the base environment in Studio Classic have incompatible versions of many packages.
2. For the conda environment that you want to migrate to Studio Classic, first activate the conda environment. The default environment is then changed when new libraries are installed or removed from it. To get the exact state of the environment, export it into a YAML file using the command line. The following command lines export the default environment into a YAML file, creating a file called myenv.yml.

```
conda activate default  
conda env export > ~/myenv.yml
```

## Step 2: Save your Studio Lab artifacts

Now that you have saved your environment to a YAML file, you can move the environment file to any platform.

Save to a local machine using Studio Lab GUI

 **Note**

Downloading a directory from the Studio Lab GUI by right-clicking on the directory is currently unavailable. If you wish to export a directory, please follow the steps using the **Save to Git repository** tab.

One option is to save the environment onto your local machine. To do this, use the following procedure.

1. In Studio Lab, choose the **File Browser** icon



)

on the left menu, so that the **File Browser** panel shows on the left.

2. Navigate to your user directory by choosing the file icon beneath the file search bar.
3. Choose (right-click) the myenv.yml file and then choose **Download**. You can repeat this process for other files you want to import to Studio Classic.

## Save to a Git repository

Another option is to save your environment to a Git repository. This option uses GitHub as an example. These steps require a GitHub account and repository. For more information, visit [GitHub](#). The following procedure shows how to synchronize your content with GitHub using the Studio Lab terminal.

1. From the Studio Lab terminal, navigate to your user directory and make a new directory to contain the files you want to export.

```
cd ~  
mkdir <NEW_DIRECTORY_NAME>
```

2. After you create a new directory, copy any file or directory you want to export to `<NEW_DIRECTORY_NAME>`.

Copy a file using the following code format:

```
cp <FILE_NAME> <NEW_DIRECTORY_NAME>
```

For example, replace `<FILE_NAME>` with `myenv.yml`.

Copy any directory using the following code format:

```
cp -r <DIRECTORY_NAME> <NEW_DIRECTORY_NAME>
```

For example, replace `<DIRECTORY_NAME>` with any directory name in your user directory.

3. Navigate to the new directory and initialize the directory as a Git repository using the following command. For more information, see the [git-init documentation](#).

```
cd <NEW_DIRECTORY_NAME>
git init
```

4. Using Git, add all relevant files and then commit your changes.

```
git add .
git commit -m "<COMMIT_MESSAGE>"
```

For example, replace *<COMMIT\_MESSAGE>* with Add Amazon SageMaker Studio Lab artifacts to GitHub repository to migrate to Amazon SageMaker Studio Classic .

5. Push the commit to your remote repository. This repository has the format `https://github.com/<GITHUB_USERNAME>/<REPOSITORY_NAME>.git` where *<GITHUB\_USERNAME>* is your GitHub user name and the *<REPOSITORY\_NAME>* is your remote repository name. Create a branch *<BRANCH\_NAME>* to push the content to the GitHub repository.

```
git branch -M <BRANCH_NAME>
git remote add origin https://github.com/<GITHUB_USERNAME>/<REPOSITORY_NAME>.git
git push -u origin <BRANCH_NAME>
```

### Step 3: Import your Studio Lab artifacts to Studio Classic

The following procedure shows how to import artifacts to Studio Classic. The instructions on using Feature Store through the console depends on if you have enabled Studio or Studio Classic as your default experience. For information on accessing Studio Classic through the console, see [Launch Studio Classic if Studio is your default experience](#).

From Studio Classic, you can import files from your local machine or from a Git repository. You can do this using the Studio Classic GUI or terminal. The following procedure uses the examples from [Step 2: Save your Studio Lab artifacts](#).

#### Import using the Studio Classic GUI

If you saved the files to your local machine, you can import the files to Studio Classic using the following steps.

1. Open the **File Browser** panel



)

at the top left of Studio Classic.

2. Choose the **Upload Files** icon



)

on the menu at the top of the **File Browser** panel.

3. Navigate to the file that you want to import, then choose **Open**.

**Note**

To import a directory into Studio Classic, first compress the directory on your local machine to a file. On a Mac, right-click the directory and choose **Compress " <DIRECTORY\_NAME> "**. In Windows, right-click the directory and choose **Send to**, and then choose **Compressed (zipped) folder**. After the directory is compressed, import the compressed file using the preceding steps. Unzip the compressed file by navigating to the Studio Classic terminal and running the command `<DIRECTORY_NAME>.zip`.

## Import using a Git repository

This example provides two options for how to clone a GitHub repository into Studio Classic. You can use the Studio Classic GUI by choosing the **Git**



)

tab on the left side of Studio Classic. Choose **Clone a Repository**, then paste your GitHub repository URL from [Step 2: Save your Studio Lab artifacts](#). Another option is to use the Studio Classic terminal by using the following procedure.

1. Open the Studio Classic **Launcher**. For more information on opening the **Launcher**, see [Amazon SageMaker Studio Classic Launcher](#).
2. In the **Launcher**, in the **Notebooks and compute resources** section, choose **Change environment**.
3. In Studio Classic, open the **Launcher**. To open the **Launcher**, choose **Amazon SageMaker Studio Classic** at the top-left corner of Studio Classic.

To learn about all the available ways to open the **Launcher**, see [Use the Amazon SageMaker Studio Classic Launcher](#).

4. In the **Change environment** dialog, use the **Image** dropdown list to select the **Data Science** image and choose **Select**. This image comes with conda pre-installed.
5. In the Studio Classic **Launcher**, choose **Open image terminal**.
6. From the image terminal, run the following command to clone your repository. This command creates a directory named after <REPOSITORY\_NAME> in your Studio Classic instance and clones your artifacts in that repository.

```
git clone https://github.com/<GITHUB_USERNAME>/<REPOSITORY_NAME>.git
```

## Step 4: Install your Studio Lab conda environments in Studio Classic

You can now recreate your conda environment by using your YAML file in your Studio Classic instance. Open the Studio Classic **Launcher**. For more information on opening the **Launcher**, see [Amazon SageMaker Studio Classic Launcher](#). From the **Launcher**, choose **Open image terminal**. In the terminal navigate to the directory that contains the YAML file, then run the following commands.

```
conda env create --file <ENVIRONMENT_NAME>.yml  
conda activate <ENVIRONMENT_NAME>
```

After these commands are complete, you can select your environment as the kernel for your Studio Classic notebook instances. To view the available environment, run `conda env list`. To activate your environment, run `conda activate <ENVIRONMENT_NAME>`.

## Shut down Studio Lab resources

You can view and shut down your running Amazon SageMaker Studio Lab resources from one location in your Studio Lab environment. The running resource types include terminals, and kernels. You can also shut down all resources of one resource type at the same time.

When you shut down all resources belonging to a resource type, the following occurs:

- **KERNELS** – All kernels, notebooks, and consoles are shut down.
- **TERMINALS** – All terminals are shut down.

## Shut down Studio Lab resources

1. Start your Studio Lab project runtime. For more information on launching Studio Lab project runtime, see [Start your project runtime](#).
2. Choose the **Running Terminals and Kernels** icon  on the left navigation pane.
3. Choose the X symbol to the right of the resource you wish to shut down. You can view the X symbol by hovering your cursor over a resource.
4. (Optional) You can shut down all the resources of a given resource type by choosing **Shutdown All** to the right of the resource type name.

## Troubleshooting

The guide shows common errors that might occur when using Amazon SageMaker Studio Lab. Each error contains a description, as well as a solution to the error.

### Note

You cannot share your password with multiple users or use Studio Lab to mine cryptocurrency. We don't recommend using Studio Lab for production tasks because of runtime limits.

### Can't access account

If you can't access your account, verify that you are using the correct email and password. If you have forgotten your password, use the following steps to reset your password. If you still cannot access your account, you must request and register for a new account using the instructions in [Onboard to Amazon SageMaker Studio Lab](#).

### Forgot password

If you forget your password, you must reset it using the following steps.

1. Navigate to the [Studio Lab landing page](#).
2. Select **Sign in**.

3. Select **Forgot password?** to open a new page.
4. Enter the email address that you used to sign up for an account.
5. Select **Send reset link** to send an email with a password reset link.
6. From the password reset email, select **Reset your password**.
7. Enter your new password.
8. Select **Submit**.

## Can't launch project runtime

If the Studio Lab project runtime does not launch, try launching it again. If this doesn't work, switch the instance type from CPU to GPU (or in reverse). For more information, see [Change your compute type](#).

## Runtime stopped running unexpectedly

If there is an issue with the environment used to run JupyterLab, then Studio Lab will automatically recreate the environment. Studio Lab does not support manual activation of this process.

## Conflicting versions

Because you can add packages and modify your environment as needed, you may run into conflicts between packages in your environment. If there are conflicts between packages in your environment, you must remove the conflicting package.

## Environment build fails

When you build an environment from a YAML file, a package-version conflict or file issue might cause a build to fail. To resolve this, remove the environment by running the following command. Do this before attempting to build it again.

```
conda remove --name <YOUR_ENVIRONMENT> --all
```

## Error message about allowing to download script from domain \*.awswaf.com

Studio Classic uses the web application firewall service AWS WAF to protect your resources, which uses JavaScript. If you are using a browser security plugin that prevents JavaScript from downloading, this error may pop up. To use Studio Classic, allow the JavaScript download from \*.awswaf.com as a trusted domain. For more information on AWS WAF, see [AWS WAF](#) from the AWS WAF, AWS Firewall Manager, and AWS Shield Advanced. Developer Guide.

## Disk space is full

If you run into a notification saying mentioning that your disk space is full or **File Load Error for <FILE\_NAME>** while attempting to open a file, you can remove files, directories, libraries, or environments to increase space. For more information on managing your libraries and environments, see [Manage your environment](#).

## Project runtime is in safe mode notification

If you run into a notification that **Project runtime is in safe mode**, you must free up some disk space to resume using the Studio Lab project runtime. Follow the instructions in the preceding troubleshoot item, **Disk space is full**. Once up to at least 500 MB of space has been cleared, you may restart the project runtime to use Studio Lab. This can be done by choosing **Amazon SageMaker Studio Lab** in the top menu of Studio Lab and choosing **Restart JupyterLab....**

### git Cannot import cv2

If you run into an error when importing cv2 after installing opencv-python, you must uninstall opencv-python and install opencv-python-headless as follows.

```
%pip uninstall opencv-python --yes  
%pip install opencv-python-headless
```

You can then import cv2 as expected.

## Studio Lab becomes unresponsive when opening large files

The Studio Lab IDE may fail to render when large files are opened, resulting in blocked access to Studio Lab resources. To resolve this, reset the Studio Lab workspace using the following procedure.

1. After you open the IDE, copy the URL in your browser's address bar. This URL should be in the `https://xxxxxx.studio.us-east-2.sagemaker.aws/studiolab/default/jupyter/lab` format. Close the tab.
2. In a new tab, paste the URL and remove anything after `https://xxxxxx.studio.us-east-2.sagemaker.aws/studiolab/default/jupyter/lab`.
3. Add `?reset` to the end of the URL, so it is in the `https://xxxxxx.studio.us-east-2.sagemaker.aws/studiolab/default/jupyter/lab?reset` format.
4. Navigate to the updated URL. This resets the saved UI state and makes the Studio Lab IDE responsive.

# Amazon SageMaker Canvas

Amazon SageMaker Canvas gives you the ability to use machine learning to generate predictions without needing to write any code. The following are some use cases where you can use SageMaker Canvas:

- Predict customer churn
- Plan inventory efficiently
- Optimize price and revenue
- Improve on-time deliveries
- Classify text or images based on custom categories
- Identify objects and text in images
- Extract information from documents

With Canvas, you can chat with popular large language models (LLMs), access Ready-to-use models, or build a custom model trained on your data.

Canvas chat is a functionality that leverages open-source and Amazon LLMs to help you boost your productivity. You can prompt the models to get assistance with tasks such as generating content, summarizing or categorizing documents, and answering questions. To learn more, see [Generative AI foundation models in SageMaker Canvas](#).

The [Ready-to-use models](#) in Canvas can extract insights from your data for a variety of use cases. You don't have to build a model to use Ready-to-use models because they are powered by Amazon AI services, including [Amazon Rekognition](#), [Amazon Textract](#), and [Amazon Comprehend](#). You only have to import your data and start using a solution to generate predictions.

If you want a model that is customized to your use case and trained with your data, you can [build a model](#). You can get predictions customized to your data by doing the following:

1. Import your data from one or more data sources.
2. Build a predictive model.
3. Evaluate the model's performance.
4. Generate predictions with the model.

Canvas supports the following types of custom models:

- Numeric prediction (also known as *regression*)
- Categorical prediction for 2 and 3+ categories (also known as *binary* and *multi-class classification*)
- Time series forecasting
- Single-label image prediction (also known as *image classification*)
- Multi-category text prediction (also known as *multi-class text classification*)

To learn more about pricing, see the [SageMaker Canvas pricing page](#). You can also see [Billing and cost in SageMaker Canvas](#) for more information.

SageMaker Canvas is currently available in the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (N. California)
- US West (Oregon)
- Asia Pacific (Mumbai)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Canada (Central)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Paris)
- Europe (Stockholm)
- South America (São Paulo)

## Topics

- [Are you a first-time SageMaker Canvas user?](#)
- [Getting started with using Amazon SageMaker Canvas](#)

- [Tutorial: Build an end-to-end machine learning workflow in SageMaker Canvas](#)
- [Amazon SageMaker Canvas setup and permissions management \(for IT administrators\)](#)
- [Generative AI assistance for solving ML problems in Canvas using Amazon Q Developer](#)
- [Data import](#)
- [Data preparation](#)
- [Generative AI foundation models in SageMaker Canvas](#)
- [Ready-to-use models](#)
- [Custom models](#)
- [Logging out of Amazon SageMaker Canvas](#)
- [Limitations and troubleshooting](#)
- [Billing and cost in SageMaker Canvas](#)

## Are you a first-time SageMaker Canvas user?

If you are a first-time user of SageMaker Canvas, we recommend that you begin by reading the following sections:

- For IT administrators – [Amazon SageMaker Canvas setup and permissions management \(for IT administrators\)](#)
- For analysts and individual users – [Getting started with using Amazon SageMaker Canvas](#)
- For an example of an end to end workflow – [Tutorial: Build an end-to-end machine learning workflow in SageMaker Canvas](#)

## Getting started with using Amazon SageMaker Canvas

This guide tells you how to get started with using SageMaker Canvas. If you're an IT administrator and would like more in-depth details, see [Amazon SageMaker Canvas setup and permissions management \(for IT administrators\)](#) to set up SageMaker Canvas for your users.

### Topics

- [Prerequisites for setting up Amazon SageMaker Canvas](#)
- [Step 1: Log in to SageMaker Canvas](#)
- [Step 2: Use SageMaker Canvas to get predictions](#)

## Prerequisites for setting up Amazon SageMaker Canvas

To set up a SageMaker Canvas application, onboard using one of the following setup methods:

1. **Onboard with the AWS console.** To onboard through the AWS console, you first create an Amazon SageMaker AI domain. SageMaker AI domains support the various machine learning (ML) environments such as Canvas and [SageMaker Studio](#). For more information about domains, see [Amazon SageMaker AI domain overview](#).
  - a. (Quick) [Use quick setup for Amazon SageMaker AI](#) – Choose this option if you'd like to quickly set up a domain. This grants your user all of the default Canvas permissions and basic functionality. Any additional features such as [document querying](#) can be enabled later by an admin. If you want to configure more granular permissions, we recommend that you choose the Advanced option instead.
  - b. (Standard) [Use custom setup for Amazon SageMaker AI](#) – Choose this option if you'd like to complete a more advanced setup of your domain. Maintain granular control over user permissions such as access to data preparation features, generative AI functionality, and model deployments.
2. **Onboard with AWS CloudFormation.** [AWS CloudFormation](#) automates the provisioning of resources and configurations so that you can set up Canvas for one or more user profiles at the same time. Use this option if you want to automate the onboarding process at scale and make sure that your applications are configured the same way every time. The following [CloudFormation template](#) provides a streamlined way to onboard to Canvas, ensuring that all required components are properly set up and allowing you to focus on building and deploying your machine learning models.

The following section describes how to onboard to Canvas by using the AWS console to create a domain.

 **Important**

For you to set up Amazon SageMaker Canvas, your version of Amazon SageMaker Studio must be 3.19.0 or later. For information about updating Amazon SageMaker Studio, see [Shut down and Update SageMaker Studio Classic](#).

## Onboard with the AWS console

If you're doing the quick domain setup, then you can follow the instructions in [Use quick setup for Amazon SageMaker AI](#), skip the rest of this section, and move on to [Step 1: Log in to SageMaker Canvas](#).

If you're doing the standard domain setup, then you can specify the Canvas features to which you'd like to grant your users access. Use the rest of this section as you complete the standard domain setup to help you configure the permissions that are specific to Canvas.

In the [Use custom setup for Amazon SageMaker AI](#) setup instructions, for **Step 2: Users and ML Activities**, you must select the Canvas permissions that you want to grant. In the **ML activities** section, you can select the following permissions policies to grant access to Canvas features. You can only select up to **8 ML activities** total when setting up your domain. The first two permissions in the following list are required to use Canvas, while the rest are for additional features.

- **Run Studio Applications** – These permissions are necessary to start up the Canvas application.
- **Canvas Core Access** – These permissions grant you access to the Canvas application and the basic functionality of Canvas, such as creating datasets, using basic data transforms, and building and analyzing models.
- (Optional) **Canvas Data Preparation (powered by Data Wrangler)** – These permissions grant you access to create data flows and use advanced transforms to prepare your data in Canvas. These permissions are also necessary for creating data processing jobs and data preparation job schedules.
- (Optional) **Canvas AI Services** – These permissions grant you access to the Ready-to-use models, foundation models, and Chat with Data features in Canvas.
- (Optional) **Kendra access** – This permission grants you access to the [document querying](#) feature, where you can query documents stored in an Amazon Kendra index using foundation models in Canvas.

If you select this option, then in the **Canvas Kendra Access** section, enter the IDs for your Amazon Kendra indexes to which you want to grant access.

- (Optional) **Canvas MLOps** – This permission grants you access to the [model deployment](#) feature in Canvas, where you can deploy models for use in production.

In the domain setup's **Step 3: Applications** section, choose **Configure Canvas** and then do the following:

1. For the **Canvas storage configuration**, specify where you want Canvas to store the application data, such as model artifacts, batch predictions, datasets, and logs. SageMaker AI creates a Canvas/ folder inside this bucket to store the data. For more information, see [Configure your Amazon S3 storage](#). For this section, do the following:
  - a. Select **System managed** if you want to set the location to the default SageMaker AI-created bucket that follows the pattern `s3://sagemaker-{Region}-{your-account-id}`.
  - b. Select **Custom S3** to specify your own Amazon S3 bucket as the storage location. Then, enter the Amazon S3 URI.
  - c. (Optional) For **Encryption key**, specify a KMS key for encrypting Canvas artifacts stored at the specified location.
2. (Optional) For **Amazon Q Developer**, do the following:
  - a. Turn on **Enable Amazon Q Developer in SageMaker Canvas for natural language ML** to give your users permissions to leverage generative AI assistance during their ML workflow in Canvas. This option only grants permissions to query Amazon Q Developer for help with predetermined tasks that can be completed in the Canvas application.
  - b. Turn on **Enable Amazon Q Developer chat for general AWS questions** to give your users permissions to make generative AI queries related to AWS services.
3. (Optional) Configure the **Large data processing** section if your users plan to process datasets larger than 5 GB in Canvas. For more detailed information about how to configure these options, see [Grant Users Permissions to Use Large Data across the ML Lifecycle](#).
4. (Optional) For the **ML Ops permissions configuration** section, do the following:
  - a. Leave the **Enable direct deployment of Canvas models** option turned on to give your users permissions to deploy their models from Canvas to a SageMaker AI endpoint. For more information about model deployment in Canvas, see [Deploy your models to an endpoint](#).
  - b. Leave the **Enable Model Registry registration permissions for all users** option turned on to give your users permissions to register their model version to the SageMaker AI model registry (it is turned on by default). For more information, see [Register a model version in the SageMaker AI model registry](#).
  - c. If you left the **Enable Model Registry registration permissions for all users** option turned on, then select either **Register to Model Registry only** or **Register and approve model in Model Registry**.

5. (Optional) For the **Local file upload configuration** section, turn on the **Enable local file upload** option to give your users permissions to upload files to Canvas from their local machines. Turning this option on attaches a cross-origin resource sharing (CORS) policy to the Amazon S3 bucket specified in the **Canvas storage configuration** (and overrides any existing CORS policy). To learn more about local file upload permissions, see [Grant Your Users Permissions to Upload Local Files](#).
6. (Optional) For the **OAuth settings** section, do the following:
  - a. Choose **Add OAuth configuration**.
  - b. For **Data source**, select your data source.
  - c. For **Secret setup**, select **Create a new secret** and enter the information you have from your identity provider. If you haven't done the initial OAuth setup with your data source yet, see [Set up connections to data sources with OAuth](#).
7. (Optional) For the **Canvas Ready-to-use models configuration**, do the following:
  - a. Leave the **Enable Canvas Ready-to-use models** option turned on to give your users permissions to generate predictions with Ready-to-use models in Canvas (it is turned on by default). This option also gives you permissions to chat with generative-AI powered models. For more information, see [Generative AI foundation models in SageMaker Canvas](#).
  - b. Leave the **Enable document query using Amazon Kendra** option turned on to give your users permissions to use foundation models for querying documents stored in an Amazon Kendra index. Then, from the dropdown menu, select the existing indexes to which you want to grant access. For more information, see [Generative AI foundation models in SageMaker Canvas](#).
  - c. For **Amazon Bedrock role**, select **Create and use a new execution role** to create a new IAM execution role that has a trust relationship with Amazon Bedrock. This IAM role is assumed by Amazon Bedrock to fine-tune large language models (LLMs) in Canvas. If you already have an execution role with a trust relationship, then select **Use an existing execution role** and choose your role from the dropdown. For more information about manually configuring permissions for your own execution role, see [Grant Users Permissions to Use Amazon Bedrock and Generative AI Features in Canvas](#).
8. Finish configuring the rest of the domain settings using the [Use custom setup for Amazon SageMaker AI](#) procedures.

### Note

If you encounter any issues with granting permissions through the console, such as permissions for Ready-to-use models, see the topic [Troubleshooting issues with granting permissions through the SageMaker AI console](#).

You should now have a SageMaker AI domain set up and all of the Canvas permissions configured.

You can edit the Canvas permissions for a domain or a specific user after the initial domain setup. Individual user settings override the domain settings. To learn how to edit your Canvas permissions in the domain settings, see [Edit domain settings](#).

### Give yourself permissions to use specific features in Canvas

The following information outlines the various permissions that you can grant to a Canvas user to allow the use of various features and functionalities within Canvas. Some of these permissions can be granted during the domain setup, but some require additional permissions or configuration. Refer to the specific permissions information for each feature that you want to enable:

- **Local file upload.** The permissions for local file upload are turned on by default in the Canvas base permissions when setting up your domain. If you can't upload local files from your machine to SageMaker Canvas, you can attach a CORS policy to the Amazon S3 bucket that you specified in the Canvas storage configuration. If you allowed SageMaker AI to use the default bucket, the bucket follows the naming pattern `s3://sagemaker-{Region}-{your-account-id}`. For more information, see [Grant Your Users Permissions to Upload Local Files](#).
- **Custom image and text prediction models.** The permissions for building custom image and text prediction models are turned on by default in the Canvas base permissions when setting up your domain. However, if you have a custom IAM configuration and don't want to attach the [AmazonSageMakerCanvasFullAccess](#) policy to your user's IAM execution role, then you must explicitly grant your user the necessary permissions. For more information, see [Grant Your Users Permissions to Build Custom Image and Text Prediction Models](#).
- **Ready-to-use models and foundation models.** You might want to use the Canvas Ready-to-use models to make predictions for your data. With the Ready-to-use models permissions, you can also chat with generative AI-powered models. The permissions are turned on by default when setting up your domain, or you can edit the permissions for a domain that you've already created. The Canvas Ready-to-use models permissions option adds the

[AmazonSageMakerCanvasAI ServicesAccess](#) policy to your execution role. For more information, see the [Get started](#) section of the Ready-to-use models documentation.

For more information about getting started with generative AI foundation models, see [Generative AI foundation models in SageMaker Canvas](#).

- **Fine-tune foundation models.** If you'd like to fine-tune foundation models in Canvas, you can either add the permissions when setting up your domain, or you can edit the permissions for the domain or user profile after creating your domain. You must add the [AmazonSageMakerCanvasAI ServicesAccess](#) policy to the AWS IAM role you chose when setting up the user profile, and you must also add a trust relationship with Amazon Bedrock to the role. For instructions on how to add these permissions to your IAM role, see [Grant Users Permissions to Use Amazon Bedrock and Generative AI Features in Canvas](#).
- **Send batch predictions to Amazon QuickSight.** You might want to [send batch predictions](#), or datasets of predictions you generate from a custom model, to Amazon QuickSight for analysis. In [QuickSight](#), you can build and publish predictive dashboards with your prediction results. For instructions on how to add these permissions to your Canvas user's IAM role, see [Grant Your Users Permissions to Send Predictions to Amazon QuickSight](#).
- **Deploy Canvas models to a SageMaker AI endpoint.** SageMaker AI Hosting offers *endpoints* which you can use to deploy your model for use in production. You can deploy models built in Canvas to a SageMaker AI endpoint and then make predictions programmatically in a production environment. For more information, see [Deploy your models to an endpoint](#).
- **Register model versions to the model registry.** You might want to register *versions* of your model to the [SageMaker AI model registry](#), which is a repository for tracking the status of updated versions of your model. A data scientist or MLOps team working in the SageMaker Model Registry can view the versions of your model that you've built and approve or reject them. Then, they can deploy your model version to production or kick off an automated workflow. Model registration permissions are turned on by default for your domain. You can manage permissions at the user profile level and grant or remove permissions to specific users. For more information, see [Register a model version in the SageMaker AI model registry](#).
- **Import data from Amazon Redshift.** If you want to import data from Amazon Redshift, you must give yourself additional permissions. You must add the `AmazonRedshiftFullAccess` managed policy to the AWS IAM role you chose when setting up the user profile. For instructions on how to add the policy to the role, see [Grant Users Permissions to Import Amazon Redshift Data](#).

**Note**

The necessary permissions to import through other data sources, such as Amazon Athena and SaaS platforms, are included in the [AmazonSageMakerFullAccess](#) and [AmazonSageMakerCanvasFullAccess](#) policies. If you followed the standard setup instructions, these policies should already be attached to your execution role. For more information about these data sources and their permissions, see [Connect to data sources](#).

## Step 1: Log in to SageMaker Canvas

When the initial setup is complete, you can access SageMaker Canvas with any of the following methods, depending on your use case:

- In the [SageMaker AI console](#), choose the **Canvas** in the left navigation pane. Then, on the **Canvas** page, select your user from the dropdown and launch the Canvas application.
- Open [SageMaker Studio](#), and in the Studio interface, go to the Canvas page and launch the Canvas application.
- Use your organization's SAML 2.0-based SSO methods, such as Okta or the IAM Identity Center.

When you log into SageMaker Canvas for the first time, SageMaker AI creates the application and a SageMaker AI *space* for you. The Canvas application's data is stored in the space. To learn more about spaces, see [Collaboration with shared spaces](#). The space consists of your user profile's applications and a shared directory for all of your applications' data. If you don't want to use the default space created by SageMaker AI and would prefer to create your own space for storing application data, see the page [Store SageMaker Canvas application data in your own SageMaker AI space](#).

## Step 2: Use SageMaker Canvas to get predictions

After you've logged in to Canvas, you can start building models and generating predictions for your data.

You can either use Canvas Ready-to-use models to make predictions without building a model, or you can build a custom model for your specific business problem. Review the following information to decide whether Ready-to-use models or custom models are best for your use case.

- **Ready-to-use models.** With Ready-to-use models, you can use pre-built models to extract insights from your data. The Ready-to-use models cover a variety of use cases, such as language detection and document analysis. To get started making predictions with Ready-to-use models, see [Ready-to-use models](#).
- **Custom models.** With custom models, you can build a variety of model types that are customized to make predictions for your data. Use custom models if you'd like to build a model that is trained on your business-specific data and if you'd like to use features such as [evaluating your model's performance](#). To get started with building a custom model, see [Custom models](#).

## Tutorial: Build an end-to-end machine learning workflow in SageMaker Canvas

This tutorial guides you through an end-to-end machine learning (ML) workflow using Amazon SageMaker Canvas. SageMaker Canvas is a visual no-code interface that you can use to prepare data and to train and deploy ML models. For the tutorial, you use a NYC taxi dataset to train a model that predicts the fare amount for a given trip. You get hands-on experience with key ML tasks such as assessing data quality and addressing data issues, splitting data into training and test sets, model training and evaluation, making predictions, and deploying your trained model—all within the SageMaker Canvas application.

### Important

This tutorial assumes that you or your administrator have created an AWS account. For information about creating an AWS account, see [Getting started: Are you a first time AWS User?](#)

## Setting up

An Amazon SageMaker AI domain is a centralized place to manage all your Amazon SageMaker AI environments and resources. A domain acts as a virtual boundary for your work in SageMaker AI, providing isolation and access control for your machine learning (ML) resources.

To get started with Amazon SageMaker Canvas, you or your administrator must navigate to the SageMaker AI console and create a Amazon SageMaker AI domain. A domain has the storage and compute resources needed for you to run SageMaker Canvas. Within the domain, you configure

SageMaker Canvas to access your Amazon S3 buckets and deploy models. Use the following procedure to set up a quick domain and create a SageMaker Canvas application.

## To set up SageMaker Canvas

1. Navigate to the [SageMaker AI console](#).
2. On the left-hand navigation, choose SageMaker Canvas.
3. Choose **Create a SageMaker AI domain**.
4. Choose **Set up**. The domain can take a few minutes to set up.

The preceding procedure used a quick domain set up. You can perform an advanced set up to control all aspects of the account configuration, including permissions, integrations, and encryption. For more information about a custom set up, see [Use custom setup for Amazon SageMaker AI](#).

By default, doing the quick domain set up provides you with permissions to deploy models. If you have custom permissions set up through a standard domain and you need manually grant model deployment permissions, see [Permissions management](#).

## Flow creation

Amazon SageMaker Canvas is a machine learning platform that enables users to build, train, and deploy machine learning models without extensive coding or machine learning expertise. One of the powerful features of Amazon SageMaker Canvas is the ability to import and work with large datasets from various sources, such as Amazon S3.

For this tutorial, we're using the NYC taxi dataset to predict the fare amount for each trip using a Amazon SageMaker Canvas Data Wrangler data flow. The following procedure outlines the steps for importing a modified version of the NYC taxi dataset into a data flow.

### Note

For improved processing, SageMaker Canvas imports a sample of your data. By default, it randomly samples 50,000 rows.

## To import the NYC taxi dataset

1. From the SageMaker Canvas home page, choose **Data Wrangler**.

2. Choose **Import data**.
3. Select **Tabular**.
4. Choose the toolbox next to data source.
5. Select **Amazon S3** from the dropdown.
6. For **Input S3 endpoint**, specify `s3://amazon-sagemaker-data-wrangler-documentation-artifacts/canvas-single-file-nyc-taxi-dataset.csv`
7. Choose **Go**.
8. Select the checkbox next to the dataset.
9. Choose **Preview data**.
10. Choose **Save**.

## Data Quality and Insights Report 1 (sample)

After importing a dataset into Amazon SageMaker Canvas, you can generate a Data Quality and Insights report on a sample of the data. Use it to provide valuable insights into the dataset. The report does the following:

- Assesses the dataset's completeness
- Identifies missing values and outliers

It can identify other potential issues that may impact model performance. It also evaluates the predictive power of each feature concerning the target variable, allowing you to identify the most relevant features for problem you're trying to solve.

We can use the insights from the report to predict the fare amount. By specifying the **Fare amount** column as the target variable and selecting **Regression** as the problem type, the report will analyze the dataset's suitability for predicting continuous values like fare prices. The report should reveal that features like **year** and **hour\_of\_day** have low predictive power for the chosen target variable, providing you with valuable insights.

Use the following procedure to get a Data Quality and Insights report on a 50,000 row sample from the dataset.

### To get a report on a sample

1. Choose **Get data insights** from the pop up window next to the **Data types** node.

2. For **Analysis name**, specify a name for the report.
3. For **Problem type**, choose **Regression**.
4. For **Target column**, choose **Fare amount**.
5. Choose **Create**.

You can review the Data Quality and Insights report on a sample of your data. The report indicates that the **year** and **hour\_of\_day** features are not predictive of the target variable, **Fare amount**.

At the top of the navigation, choose the name of the data flow to navigate back to it.

## Drop year and hour of day

We're using insights from the report to drop the **year** and **hour\_of\_day** columns to streamline the feature space and potentially improve model performance.

Amazon SageMaker Canvas provides a user-friendly interface and tools to perform such data transformations.

Use the following procedure to drop the **year** and **hour\_of\_day** columns from the NYC taxi dataset using the Data Wrangler tool in Amazon SageMaker Canvas.

1. Choose the icon next to **Data types**.
2. Choose **Add step**.
3. In the search bar, write **Drop column**.
4. Choose **Manage columns**.
5. Choose **Drop column**.
6. For **Columns to drop**, select the **year** and **hour\_of\_day** columns.
7. Choose **Preview** to view how your transform changes your data.
8. Choose **Add**.

You can use the preceding procedure as the basis to add all of the other transforms in SageMaker Canvas.

## Data Quality and Insights Report 2 (full dataset)

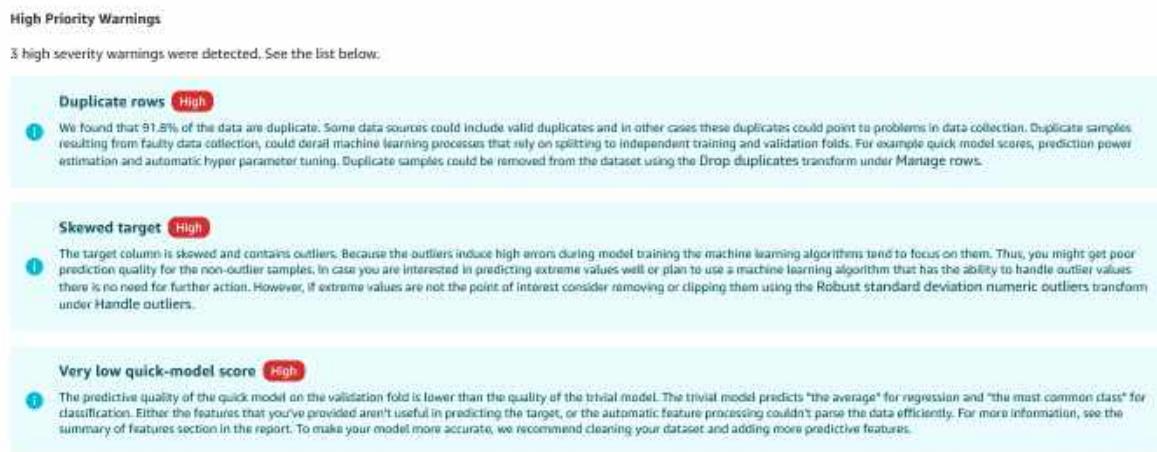
For the previous insights report, we used a sample of the NYC taxi dataset. For our second report, we're running a comprehensive analysis on the entire dataset to identify potential issues impacting model performance.

Use the following procedure to create a Data Quality and Insights report on an entire dataset.

### To get a report on the entire dataset

1. Choose the icon next to the **Drop columns** node.
2. Choose **Get data insights**.
3. For **Analysis name**, specify a name for the report.
4. For **Problem type**, choose **Regression**.
5. For **Target column**, choose **Fare amount**.
6. For **Data size**, choose **Full dataset**.
7. Choose **Create**.

The following is an image from the insights report:



It shows the following issues:

- Duplicate rows
- Skewed target

Duplicate rows can lead to data leakage, where the model is exposed to the same data during training and testing. They can lead to overly optimistic performance metrics. Removing duplicate rows ensures that the model is trained on unique instances, reducing the risk of data leakage and improving the model's ability to generalize.

A skewed target variable distribution, in this case, the **Fare amount** column, can cause imbalanced classes, where the model may become biased towards the majority class. This can lead to poor performance on minority classes, which is particularly problematic in scenarios where accurately predicting rare or underrepresented instances is important.

## Addressing data quality issues

To address these issues and prepare the dataset for modeling, you can search for the following transformations and apply them:

1. Drop duplicates using the **Manage rows** transform.
2. **Handle outliers** in the **Fare amount** column using the **Robust standard deviation numeric outliers**.
3. **Handle outliers** in the **Trip distance** and **Trip duration** columns using the **Standard deviation numeric outliers**.
4. Use the **Encode categorical** to encode the **Rate code id**, **Payment type**, **Extra flag**, and **Toll flag** columns as floats.

If you're not sure about how to apply a transform, see [Drop year and hour of day](#)

By addressing these data quality issues and applying appropriate transformations, you can improve the dataset's suitability for modeling.

## Verifying data quality and quick model accuracy

After applying the transforms to address data quality issues, such as removing duplicate rows, we create our final Data Quality and Insights report. This report helps verify that the applied transformations resolved the issues and that the dataset is now in a suitable state for modeling.

When reviewing the final Data Quality and Insights report, you should expect to see no major data quality issues flagged. The report should indicate that:

- The target variable is no longer skewed

- There are no outliers or duplicate rows

Additionally, the report should provide a quick model score based on a baseline model trained on the transformed dataset. This score serves as an initial indicator of the model's potential accuracy and performance.

Use the following procedure to create the Data Quality and Insights report.

### To create the Data Quality and Insights report

1. Choose the icon next to the **Drop columns** node.
2. Choose **Get data insights**.
3. For **Analysis name**, specify a name for the report.
4. For **Problem type**, choose **Regression**.
5. For **Target column**, choose **Fare amount**.
6. For **Data size**, choose **Full dataset**.
7. Choose **Create**.

### Split the data into training and test sets

To train a model and evaluate its performance, we use the **Split data** transform to split the data into training and test sets.

By default, SageMaker Canvas uses a Randomized split, but you can also use the following types of splits:

- Ordered
- Stratified
- Split by key

You can change the **Split percentage** or add splits.

For this tutorial, use all of the default settings in the split. You need to double click on the dataset to view its name. The training dataset has the name **Dataset (Train)**.

Next to the **Ordinal encode** node apply the **Split data** transform.

## Train model

After you split your data, you can train a model. This model learns from patterns in your data. You can use it to make predictions or uncover insights.

SageMaker Canvas has both quick builds and standard builds. Use a standard build to train best performing model on your data.

Before you start training a model, you must first export the training dataset as a SageMaker Canvas dataset.

### To export your dataset

1. Next to the node for the training dataset, choose the icon and select **Export**.
2. Select **SageMaker Canvas dataset**.
3. Choose **Export** to export the dataset.

After you've created a dataset, you can train a model on the SageMaker Canvas dataset that you've created. For information about training a model, see [Build a custom numeric or categorical prediction model](#).

## Evaluate model and make predictions

After training your machine learning model, it's crucial to evaluate its performance to ensure it meets your requirements and performs well on unseen data. Amazon SageMaker Canvas provides a user-friendly interface to assess your model's accuracy, review its predictions, and gain insights into its strengths and weaknesses. You can use the insights to make informed decisions about its deployment and potential areas for improvement.

Use the following procedure to evaluate a model before you deploy it.

### To evaluate a model

1. Choose **My Models**.
2. Choose the model you've created.
3. Under **Versions**, select the version corresponding to the model.

You can now view the model evaluation metrics.

After you evaluate the model, you can make predictions on new data. We're using the test dataset that we've created.

To use the test dataset for predictions we need to convert it into a SageMaker Canvas dataset. The SageMaker Canvas dataset is in a format that the model can interpret.

Use the following procedure to create a SageMaker Canvas dataset from the test dataset.

### To create a SageMaker Canvas dataset

1. Next to the **Dataset (Test)** dataset, choose the radio icon.
2. Select **Export**.
3. Select **SageMaker Canvas dataset**.
4. For **Dataset name**, specify a name for the dataset.
5. Choose **Export**.

Use the following procedure to make predictions. It assumes that you're still on the **Analyze** page.

### To make predictions on test dataset

1. Choose **Predict**.
2. Choose **Manual**.
3. Select the dataset that you've exported.
4. Choose **Generate predictions**.
5. When SageMaker Canvas has finished generating predictions, select the icon to the right of the dataset.
6. Choose **Preview** to view the predictions.

## Deploy a model

After you've evaluated your model, you can deploy it to an endpoint. You can submit requests to the endpoint to get predictions.

Use the following procedure to deploy a model. It assumes that you're still on the **Predict** page.

### To deploy a model

1. Choose **Deploy**.

2. Choose **Create deployment**.
3. Choose **Deploy**.

## Cleaning up

You've successfully completed the tutorial. To avoid incurring additional charges, delete the resources that you're not using.

Use the following procedure to delete the endpoint that you created. It assumes that you're still on the **Deploy** page.

### To delete an endpoint

1. Choose the radio button to the right of your deployment.
2. Select **Delete deployment**.
3. Choose **Delete**.

After deleting the deployment, delete the datasets that you've created within SageMaker Canvas. Use the following procedure to delete the datasets.

### To delete the datasets

1. Choose **Datasets** on the left-hand navigation.
2. Select the dataset that you've analyzed and the synthetic dataset used for predictions.
3. Choose **Delete**.

To avoid incurring additional charges, you must log out of SageMaker Canvas. For more information, see [Logging out of Amazon SageMaker Canvas](#).

## Amazon SageMaker Canvas setup and permissions management (for IT administrators)

The following pages explain how IT administrators can configure Amazon SageMaker Canvas and grant permissions to users within their organizations. You learn how to set up the storage configuration, manage data encryption and VPCs, control access to specific capabilities like generative AI foundation models, integrate with other AWS services like Amazon Redshift, and

more. By following these steps, you can tailor SageMaker Canvas for your users based on your organization's specific requirements.

You can also set up SageMaker Canvas for your users with AWS CloudFormation. For more information, see [AWS::SageMaker AI::App](#) in the *AWS CloudFormation User Guide*.

## Topics

- [Grant Your Users Permissions to Upload Local Files](#)
- [Set Up SageMaker Canvas for Your Users](#)
- [Configure your Amazon S3 storage](#)
- [Grant permissions for cross-account Amazon S3 storage](#)
- [Grant Users Permissions to Use Large Data across the ML Lifecycle](#)
- [Encrypt Your SageMaker Canvas Data with AWS KMS](#)
- [Store SageMaker Canvas application data in your own SageMaker AI space](#)
- [Grant Your Users Permissions to Build Custom Image and Text Prediction Models](#)
- [Grant Users Permissions to Use Amazon Bedrock and Generative AI Features in Canvas](#)
- [Update SageMaker Canvas for Your Users](#)
- [Request a Quota Increase](#)
- [Grant Users Permissions to Import Amazon Redshift Data](#)
- [Grant Your Users Permissions to Send Predictions to Amazon QuickSight](#)
- [Applications management](#)
- [Configure Amazon SageMaker Canvas in a VPC without internet access](#)
- [Set up connections to data sources with OAuth](#)

## Grant Your Users Permissions to Upload Local Files

If your users are uploading files from their local machines to SageMaker Canvas, you must attach a CORS (cross-origin resource sharing) configuration to the Amazon S3 bucket that they're using. When setting up or editing the SageMaker AI domain or user profile, you can specify either a custom Amazon S3 location or the default location, which is a SageMaker AI created Amazon S3 bucket with a name that uses the following pattern: `s3://sagemaker-{Region}-{your-account-id}`. SageMaker Canvas adds your users' data to the bucket whenever they upload a file.

To grant users permissions to upload local files to the bucket, you can attach a CORS configuration to it using either of the following procedures. You can use the first method when editing the settings of your domain, where you opt in to allow SageMaker AI to attach the CORS configuration to the bucket for you. You can also use the first method for editing a user profile within a domain. The second method is the manual method, where you can attach the CORS configuration to the bucket yourself.

## SageMaker AI domain settings method

To grant your users permissions to upload local files, you can edit the Canvas application configuration in the domain settings. This attaches a Cross-Origin Resource Sharing (CORS) configuration to the Canvas storage configuration's Amazon S3 bucket and grants all users in the domain permission to upload local files into SageMaker Canvas. By default, the permissions option is turned on when you set up a new domain, but you can turn this option on and off as needed.

### Note

If you have an existing CORS configuration on the storage configuration Amazon S3 bucket, turning on the local file upload option overwrites the existing configuration with the new configuration.

The following procedure shows how you can turn on this option by editing the domain settings in the SageMaker AI console.

1. Go to the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. In the left navigation pane, choose **Domains**.
3. From the list of domains, choose your domain.
4. On the domain details page, select the **App Configurations** tab.
5. Go to the **Canvas** section and choose **Edit**.
6. Turn on the **Enable local file upload** toggle. This attaches the CORS configuration and grants local file upload permissions.
7. Choose **Submit**.

Users in the specified domain should now have local file upload permissions.

You can also grant permissions to specific user profiles in a domain by following the preceding procedure and going into the user profile settings instead of the overall domain settings.

## Amazon S3 bucket method

If you want to manually attach the CORS configuration to the SageMaker AI Amazon S3 bucket, use the following procedure.

1. Sign in to <https://console.aws.amazon.com/s3/>.
2. Choose your bucket. If your domain uses the default SageMaker AI created bucket, the bucket's name uses the following pattern: s3://sagemaker-*{Region}*-*{your-account-id}*.
3. Choose **Permissions**.
4. Navigate to **Cross-origins resource sharing (CORS)**.
5. Choose **Edit**.
6. Add the following CORS policy:

```
[  
  {  
    "AllowedHeaders": [  
      "*"  
    ],  
    "AllowedMethods": [  
      "POST"  
    ],  
    "AllowedOrigins": [  
      "*"  
    ],  
    "ExposeHeaders": []  
  }  
]
```

7. Choose **Save changes**.

In the preceding procedure, the CORS policy must have "POST" listed under AllowedMethods.

After you've gone through the procedure, you should have:

- An IAM role assigned to each of your users.

- Amazon SageMaker Studio Classic runtime permissions for each of your users. SageMaker Canvas uses Studio Classic to run the commands from your users.
- If the users are uploading files from their local machines, a CORS policy attached to their Amazon S3 bucket.

If your users still can't upload the local files after you update the CORS policy, the browser might be caching the CORS settings from a previous upload attempt. If they're running into issues, instruct them to clear their browser cache and try again.

## Set Up SageMaker Canvas for Your Users

To set up Amazon SageMaker Canvas, do the following:

- Create an Amazon SageMaker AI domain.
- Create user profiles for the domain
- Set up Okta Single Sign On (Okta SSO) for your users.
- Activate link sharing for models.

Use Okta Single-Sign On (Okta SSO) to grant your users access to Amazon SageMaker Canvas. SageMaker Canvas supports SAML 2.0 SSO methods. The following sections guide you through procedures to set up Okta SSO.

To set up a domain, see [Use custom setup for Amazon SageMaker AI](#) and follow the instructions for setting up your domain using IAM authentication. You can use the following information to help you complete the procedure in the section:

- You can ignore the step about creating projects.
- You don't need to provide access to additional Amazon S3 buckets. Your users can use the default bucket that we provide when we create a role.
- To grant your users access to share their notebooks with data scientists, turn on **Notebook Sharing Configuration**.
- Use Amazon SageMaker Studio Classic version 3.19.0 or later. For information about updating Amazon SageMaker Studio Classic, see [Shut down and Update SageMaker Studio Classic](#).

Use the following procedure to set up Okta. For all of the following procedures, you specify the same IAM role for **IAM-role**.

## Add the SageMaker Canvas application to Okta

Set up the sign-on method for Okta.

1. Sign in to the Okta Admin dashboard.
2. Choose **Add application**. Search for **AWS Account Federation**.
3. Choose **Add**.
4. Optional: Change the name to **Amazon SageMaker Canvas**.
5. Choose **Next**.
6. Choose **SAML 2.0** as the **Sign-On** method.
7. Choose **Identity Provider Metadata** to open the metadata XML file. Save the file locally.
8. Choose **Done**.

## Set up ID federation in IAM

AWS Identity and Access Management (IAM) is the AWS service that you use to gain access to your AWS account. You gain access to AWS through an IAM account.

1. Sign in to the AWS console.
2. Choose **AWS Identity and Access Management (IAM)**.
3. Choose **Identity Providers**.
4. Choose **Create Provider**.
5. For **Configure Provider**, specify the following:
  - **Provider Type** – From the dropdown list, choose **SAML**.
  - **Provider Name** – Specify **Okta**.
  - **Metadata Document** – Upload the XML document that you've saved locally from step 7 of [Add the SageMaker Canvas application to Okta](#).
6. Find your identity provider under **Identity Providers**. Copy its **Provider ARN** value.
7. For **Roles**, choose the IAM role that you're using for Okta SSO access.
8. Under **Trust Relationship** for the IAM role, choose **Edit Trust Relationship**.
9. Modify the IAM trust relationship policy by specifying the **Provider ARN** value that you've copied and add the following policy:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "Federated": "arn:aws:iam::123456789012:saml-provider/Okta"  
            },  
            "Action": [  
                "sts:AssumeRoleWithSAML",  
                "sts:SetSourceIdentity",  
                "sts:TagSession"  
            ],  
            "Condition": {  
                "StringEquals": {  
                    "SAML:aud": "https://signin.aws.amazon.com/saml"  
                }  
            }  
        }  
    ]  
}
```

## 10. For Permissions, add the following policy:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AmazonSageMakerPresignedUrlPolicy",  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker>CreatePresignedDomainUrl",  
                "sagemaker>CreatePresignedDomainUrlWithPrincipalTag"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

## Configure SageMaker Canvas in Okta

Configure Amazon SageMaker Canvas in Okta using the following procedure.

To configure Amazon SageMaker Canvas to use Okta, follow the steps in this section. You must specify unique user names for each **SageMakerStudioProfileName** field. For example, you can use `user.login` as a value. If the username is different from the SageMaker Canvas profile name, choose a different uniquely identifying attribute. For example, you can use an employee's ID number for the profile name.

For an example of values that you can set for **Attributes**, see the code following the procedure.

1. Under **Directory**, choose **Groups**.
2. Add a group with the following pattern: `sagemaker#canvas#IAM-role#AWS-account-id`.
3. In Okta, open the **AWS Account Federation** application integration configuration.
4. Select **Sign On** for the AWS Account Federation application.
5. Choose **Edit** and specify the following:
  - SAML 2.0
  - **Default Relay State** – `https://Region.console.aws.amazon.com/sagemaker/home?region=Region#/studio/canvas/open/StudioId`. You can find the Studio Classic ID in the console: <https://console.aws.amazon.com/sagemaker/>
6. Choose **Attributes**.
7. In the **SageMakerStudioProfileName** fields, specify unique values for each username. The usernames must match the usernames that you've created in the AWS console.

```
Attribute 1:  
Name: https://aws.amazon.com/SAML/Attributes/  
PrincipalTag:SageMakerStudioUserProfileName  
Value: ${user.login}
```

```
Attribute 2:  
Name: https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys  
Value: {"SageMakerStudioUserProfileName"}
```

8. Select **Environment Type**. Choose **Regular AWS**.

- If your environment type isn't listed, you can set your ACS URL in the **ACS URL** field. If your environment type is listed, you don't need to enter your ACS URL
9. For **Identity Provider ARN**, specify the ARN you used in step 6 of the preceding procedure.
10. Specify a **Session Duration**.
11. Choose **Join all roles**.
12. Turn on **Use Group Mapping** by specifying the following fields:
- **App Filter** – okta
  - **Group Filter** – ^aws\#\S+\#(?IAM-role[\w\-\-]+)\#(?accountid\d+)\$
  - **Role Value Pattern** – arn:aws:iam::\$accountid:saml-provider/  
Okta,arn:aws:iam::\$accountid:role/IAM-role
13. Choose **Save/Next**.
14. Under **Assignments**, assign the application to the group that you've created.

## Add optional policies on access control in IAM

In IAM, you can apply the following policy to the administrator user who creates the user profiles.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "CreateSageMakerStudioUserProfilePolicy",  
            "Effect": "Allow",  
            "Action": "sagemaker>CreateUserProfile",  
            "Resource": "*",  
            "Condition": {  
                "ForAnyValue:StringEquals": {  
                    "aws:TagKeys": [  
                        "studiouserid"  
                    ]  
                }  
            }  
        }  
    ]  
}
```

If you choose to add the preceding policy to the admin user, you must use the following permissions from [Set up ID federation in IAM](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AmazonSageMakerPresignedUrlPolicy",  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker:CreatePresignedDomainUrl",  
                "sagemaker:CreatePresignedDomainUrlWithPrincipalTag"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {  
                    "sagemaker:ResourceTag/studiouserid": "${aws:PrincipalTag/  
SageMakerStudioUserProfileName}"  
                }  
            }  
        }  
    ]  
}
```

## Configure your Amazon S3 storage

When you set up your SageMaker Canvas application, the default storage location for model artifacts, datasets, and other application data is an Amazon S3 bucket that Canvas creates. This default Amazon S3 bucket follows the naming pattern `s3://sagemaker-{Region}-{your-account-id}` and exists in the same Region as your Canvas application. However, you can customize the storage location and specify your own Amazon S3 bucket for storing Canvas application data. You might want to use your own Amazon S3 bucket for storing application data for any of the following reasons:

- Your organization has internal naming conventions for Amazon S3 buckets.
- You want to enable cross-account access to model artifacts or other Canvas data.
- You want to be compliant with internal security guidelines, such as restricting users to specific Amazon S3 buckets or model artifacts.

- You want enhanced visibility and access to logs produced by Canvas, independent of the AWS console or SageMaker Studio Classic.

By specifying your own Amazon S3 bucket, you can have increased control over your own storage and be compliant with your organization.

To get started, you can either create a new SageMaker AI domain or user profile, or you can update an existing domain or user profile. Note that the user profile settings override the domain-level settings. For example, you can use the default bucket configuration at the domain level, but you can specify a custom Amazon S3 bucket for an individual user. After specifying your own Amazon S3 bucket for the domain or user profile, Canvas creates a subfolder called `Canvas/<UserProfileName>` under the input Amazon S3 URI and saves all artifacts generated in the Canvas application under this subfolder.

### **Important**

If you update an existing domain or user profile, you no longer have access to your Canvas artifacts from the previous location. Your files are still in the old Amazon S3 location, but you can no longer view them from Canvas. The new configuration takes effect the next time you log into the application.

For more information about granting cross-account access to your Amazon S3 bucket, see [Granting cross-account object permissions](#) in the *Amazon S3 User Guide*.

The following sections describe how to specify a custom Amazon S3 bucket for your Canvas storage configuration. If you're setting up a new SageMaker AI domain (or a new user in a domain), then use the [New domain setup method](#) or the [New user profile setup method](#). If you have an existing Canvas user profile and would like to update the profile's storage configuration, use the [Existing user method](#).

### **Before you begin**

If you're specifying an Amazon S3 URI from a different AWS account, or if you're using a bucket that is encrypted with AWS KMS, then you must configure permissions before proceeding. You must grant AWS IAM permissions to ensure that Canvas can download and upload objects to and from your bucket. For detailed information on how to grant the required permissions, see [Grant permissions for cross-account Amazon S3 storage](#).

Additionally, the final Amazon S3 URI for the training folder in your Canvas storage location must be 128 characters or less. The final Amazon S3 URI consists of your bucket path `s3://<your-bucket-name>/<folder-name>/` plus the path that Canvas adds to your bucket: `Canvas/<user-profile-name>/Training`. For example, an acceptable path that is less than 128 characters is `s3://<amzn-s3-demo-bucket>/<machine-learning>/Canvas/<user-1>/Training`.

## New domain setup method

If you're setting up a new domain and Canvas application, use this section to configure the storage location at the domain level. This configuration applies to all new users you create in the domain, unless you specify a different storage location for individual user profiles.

When doing a **Standard setup** for your domain, on the **Step 3: Configure Applications - Optional** page, use the following procedure for the **Canvas** section:

1. For the **Canvas storage configuration**, do the following:
  - a. Select **System managed** if you want to set the location to the default SageMaker AI bucket that follows the pattern `s3://sagemaker-{Region}-{your-account-id}`.
  - b. Select **Custom S3** to specify your own Amazon S3 bucket as the storage location. Then, enter the Amazon S3 URI.
  - c. (Optional) For **Encryption key**, specify a KMS key for encrypting Canvas artifacts stored at the specified location.
2. Finish setting up the domain and choose **Submit**.

Your domain is now configured to use the Amazon S3 location you specified for SageMaker Canvas application storage.

## New user profile setup method

If you're setting up a new user profile in your domain, use this section to configure the storage location for the user. This configuration overrides the domain-level configuration.

When adding a user profile to your domain, for **Step 2: Configure Applications**, use the following procedure for the **Canvas** section:

1. For the **Canvas storage configuration**, do the following:

- a. Select **System managed** if you want to set the location to the default SageMaker AI created bucket that follows the pattern `s3://sagemaker-{Region}-{your-account-id}`.
  - b. Select **Custom S3** to specify your own Amazon S3 bucket as the storage location. Then, enter the Amazon S3 URI.
  - c. (Optional) For **Encryption key**, specify a KMS key for encrypting Canvas artifacts stored at the specified location.
2. Finish setting up the user profile and choose **Submit**.

Your user profile is now configured to use the Amazon S3 location you specified for SageMaker Canvas application storage.

### Existing user method

If you have an existing Canvas user profile and would like to update the Amazon S3 storage location, you can edit the SageMaker AI domain or user profile settings. The change takes effect the next time you log into the Canvas application.

#### Note

When you change the storage location for an existing Canvas application, you lose access to your Canvas artifacts from the previous storage location. The artifacts are still stored in the old Amazon S3 location, but you can no longer view them from Canvas.

Remember that the user profile settings override the general domain settings, so you can update the Amazon S3 storage location for specific user profiles without changing it for all of the users. You can update the storage configuration for an existing domain or user by using the following procedures.

#### Update an existing domain

Use the following procedure to update the storage configuration for a domain.

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **Domains**.

4. From the list of domains, choose your domain.
5. On the **Domain details** page, choose the **App Configurations** tab.
6. Scroll down to the **Canvas** section and choose **Edit**.
7. The **Edit Canvas settings** page opens. For the **Canvas storage configuration** section, do the following:
  - a. Select **System managed** if you want to set the location to the default SageMaker AI created bucket that follows the pattern `s3://sagemaker-{Region}-{your-account-id}`.
  - b. Select **Custom S3** to specify your own Amazon S3 bucket as the storage location. Then, enter the Amazon S3 URI.
  - c. (Optional) For **Encryption key**, specify a KMS key for encrypting Canvas artifacts stored at the specified location.
8. Finish any other modifications you want to make to the domain, and then choose **Submit** to save your changes.

## Update an existing user profile

Use the following procedure to update the storage configuration for a user profile.

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of **domains**, choose your domain.
5. From the list of users in the domain, choose the user whose configuration you want to edit.
6. On the **User Details** page, choose **Edit**.
7. In the navigation pane, choose **Canvas settings**.
8. For the **Canvas storage configuration**, do the following:
  - a. Select **System managed** if you want to set the location to the default SageMaker AI bucket that follows the pattern `s3://sagemaker-{Region}-{your-account-id}`.
  - b. Select **Custom S3** to specify your own Amazon S3 bucket as the storage location. Then, enter the Amazon S3 URI.
  - c. (Optional) For **Encryption key**, specify a KMS key for encrypting Canvas artifacts stored at the specified location.

9. Finish any other modifications you want to make to the user profile, and then choose **Submit** to save your changes.

The storage location for your Canvas user profile should now be updated. The next time you log into the Canvas application, you receive a notification that the storage location has been updated. You lose access to any previous artifacts that you created in Canvas. You can still access the files in Amazon S3, but you can no longer view them in Canvas.

## Grant permissions for cross-account Amazon S3 storage

When setting up your SageMaker AI domain or user profile for users to access SageMaker Canvas, you specify an Amazon S3 storage location for Canvas artifacts. These artifacts include saved copies of your input datasets, model artifacts, predictions, and other application data. You can either use the default SageMaker AI created Amazon S3 bucket, or you can customize the storage location and specify your own bucket for storing Canvas application data.

You can specify an Amazon S3 bucket in another AWS account for storing your Canvas data, but first you must grant cross-account permissions so that Canvas can access the bucket.

The following sections describe how to grant permissions to Canvas for uploading and downloading objects to and from an Amazon S3 bucket in another account. There are additional permissions for when your bucket is encrypted with AWS KMS.

### Requirements

Before you begin, review the following requirements:

- Cross-account Amazon S3 buckets (and any associated AWS KMS keys) must be in the same AWS Region as the Canvas user domain or user profile.
- The final Amazon S3 URI for the training folder in your Canvas storage location must be 128 characters or less. The final S3 URI consists of your bucket path `s3://<your-bucket-name>/<folder-name>/` plus the path that Canvas adds to your bucket: `Canvas/<user-profile-name>/Training`. For example, an acceptable path that is less than 128 characters is `s3://<amzn-s3-demo-bucket>/<machine-learning>/Canvas/<user-1>/Training`.

## Permissions for cross-account Amazon S3 buckets

The following section outlines the basic steps for granting the necessary permissions so that Canvas can access your Amazon S3 bucket in another account. For more detailed instructions, see [Example 2: Bucket owner granting cross-account bucket permissions](#) in the *Amazon S3 User Guide*.

1. Create an Amazon S3 bucket, `bucketA`, in Account A.
2. The Canvas user exists in another account called Account B. In the following steps, we refer to the Canvas user's IAM role as `roleB` in Account B.

Give the IAM role `roleB` in Account B permission to download (GetObject) and upload (PutObject) objects to and from `bucketA` in Account A by attaching an IAM policy.

To limit access to a specific bucket folder, define the folder name in the resource element, such as `arn:aws:s3:::<bucketA>/FolderName/*`. For more information, see [How can I use IAM policies to grant user-specific access to specific folders?](#)

 **Note**

Bucket-level actions, such as `GetBucketCors` and `GetBucketLocation`, should be added on bucket-level resources, not folders.

The following example IAM policy grants the required permissions for `roleB` to access objects in `bucketA`:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "s3:GetObject",  
                "s3:PutObject",  
                "s3:DeleteObject"  
            ],  
            "Resource": [  
                "arn:aws:s3:::bucketA/FolderName/*",  
            ]  
        },  
    ],  
}
```

```
{  
    "Effect": "Allow",  
    "Action": [  
        "s3>ListBucket",  
        "s3:GetBucketCors",  
        "s3:GetBucketLocation"  
    ],  
    "Resource": [  
        "arn:aws:s3:::bucketA",  
    ]  
}  
}  
]
```

3. Configure the bucket policy for bucketA in Account A to grant permissions to the IAM role roleB in Account B.

 **Note**

Admins must also turn off **Block all public access** under the bucket **Permissions** section.

The following is an example bucket policy for bucketA to grant the necessary permissions to roleB:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "AWS": "arn:aws:iam::accountB:role/roleB"  
            },  
            "Action": [  
                "s3>DeleteObject",  
                "s3:GetObject",  
                "s3:PutObject"  
            ],  
            "Resource": "arn:aws:s3:::bucketA/FolderName/*"  
        },  
        {  
            "  
        }  
    ]  
}
```

```
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::accountB:role/roleB"
        },
        "Action": [
            "s3>ListBucket",
            "s3>GetBucketCors",
            "s3>GetBucketLocation"
        ],
        "Resource": "arn:aws:s3:::bucketA"
    }
]
```

After configuring the preceding permissions, your Canvas user profile in Account B can now use the Amazon S3 bucket in Account A as the storage location for Canvas artifacts.

## Permissions for cross-account Amazon S3 buckets encrypted with AWS KMS

The following procedure shows you how to grant the necessary permissions so that Canvas can access your Amazon S3 bucket in another account that is encrypted with AWS KMS. The steps are similar to the procedure above, but with additional permissions. For more information about granting cross-account KMS key access, see [Allowing users in other accounts to use a KMS key](#) in the *AWS KMS Developer Guide*.

1. Create an Amazon S3 bucket, `bucketA`, and an Amazon S3 KMS key `s3KmsInAccountA` in Account A.
2. The Canvas user exists in another account called Account B. In the following steps, we refer to the Canvas user's IAM role as `roleB` in Account B.

Give the IAM role `roleB` in Account B permission to do the following:

- Download (`GetObject`) and upload (`PutObject`) objects to and from `bucketA` in Account A.
- Access the AWS KMS key `s3KmsInAccountA` in Account A.

The following example IAM policy grants the required permissions for `roleB` to access objects in `bucketA` and use the KMS key `s3KmsInAccountA`:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "s3:GetObject",  
                "s3:PutObject",  
                "s3:DeleteObject"  
            ],  
            "Resource": [  
                "arn:aws:s3:::bucketA/FolderName/*"  
            ]  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "s3:GetBucketCors",  
                "s3:GetBucketLocation"  
            ],  
            "Resource": [  
                "arn:aws:s3:::bucketA"  
            ]  
        },  
        {  
            "Action": [  
                "kms:DescribeKey",  
                "kms>CreateGrant",  
                "kms:RetireGrant",  
                "kms:GenerateDataKey",  
                "kms:GenerateDataKeyWithoutPlainText",  
                "kms:Decrypt"  
            ],  
            "Effect": "Allow",  
            "Resource": "arn:aws:kms:{region}:accountA:key/s3KmsInAccountA"  
        }  
    ]  
}
```

3. Configure the bucket policy for bucketA and the key policy for s3KmsInAccountA in Account A to grant permissions to the IAM role roleB in Account B.

The following is an example bucket policy for bucketA to grant the necessary permissions to roleB:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "AWS": "arn:aws:iam::accountB:role/roleB"  
            },  
            "Action": [  
                "s3:DeleteObject",  
                "s3:GetObject",  
                "s3:PutObject"  
            ],  
            "Resource": "arn:aws:s3:::bucketA/FolderName/*"  
        },  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "AWS": "arn:aws:iam::accountB:role/roleB"  
            },  
            "Action": [  
                "s3:GetBucketCors",  
                "s3:GetBucketLocation"  
            ],  
            "Resource": "arn:aws:s3:::bucketA"  
        }  
    ]  
}
```

The following example is a key policy that you attach to the KMS key s3KmsInAccountA in Account A to grant roleB access. For more information about how to create and attach a key policy statement, see [Creating a key policy](#) in the *AWS KMS Developer Guide*.

```
{  
    "Sid": "Allow use of the key",  
    "Effect": "Allow",  
    "Principal": {  
        "AWS": [  
            "arn:aws:iam::accountB:role/roleB"  
        ]  
    },  
    "Action": "kms:Encrypt",  
    "Resource": "arn:aws:kms:us-east-1:123456789012:key/12345678-1234-1234-1234-123456789012"  
}
```

```
        "arn:aws:iam::accountB:role/roleB"
    ]
},
"Action": [
    "kms:DescribeKey",
    "kms>CreateGrant",
    "kms:RetireGrant",
    "kms:GenerateDataKey",
    "kms:GenerateDataKeyWithoutPlainText",
    "kms:Decrypt"
],
"Resource": "*"
}
```

After configuring the preceding permissions, your Canvas user profile in Account B can now use the encrypted Amazon S3 bucket in Account A as the storage location for Canvas artifacts.

## Grant Users Permissions to Use Large Data across the ML Lifecycle

Amazon SageMaker Canvas users working with datasets larger than 10 GB in CSV format or 2.5 GB in Parquet format require specific permissions for large data processing. These permissions are essential for managing large-scale data throughout the machine learning lifecycle. When datasets exceed the stated thresholds, or the application's local memory capacity, SageMaker Canvas uses Amazon EMR Serverless for efficient processing. This applies to:

- **Data Import:** Importing large datasets with random or stratified sampling.
- **Data Preparation:** Exporting processed data from Data Wrangler in Canvas to Amazon S3, to a new Canvas dataset, or to a Canvas model.
- **Model Building:** Training models on large datasets.
- **Inference:** Making predictions on large datasets.

By default, SageMaker Canvas uses EMR Serverless to run these remote jobs with the following app settings:

- **Pre-Initialized capacity:** Not configured
- **Application limits:** Maximum capacity of 400 vCPUs, max concurrent 16 vCPUs per account, 3000 GB memory, 20000 GB disk
- **Metastore configuration:** AWS Glue Data Catalog

- Application logs: AWS managed storage (enabled), using an AWS owned encryption key
- Application behavior: Auto-starts on job submission and auto-stops after the application is idle for 15 minutes

To enable these large data processing capabilities, users need the necessary permissions, which can be granted through the Amazon SageMaker AI domain settings. The method for granting these permissions depends on how your Amazon SageMaker AI domain was set up initially. We'll cover three main scenarios:

- Quick domain setup
- Custom domain setup (with public internet access/without VPC)
- Custom domain setup (with VPC and without public internet access)

Each scenario requires specific steps to ensure that users have the required permissions to leverage EMR Serverless for large data processing across the entire machine learning lifecycle in SageMaker Canvas.

### Scenario 1: Quick domain setup

If you used the **Quick setup** option when creating your SageMaker AI domain, follow these steps:

1. Navigate to the Amazon SageMaker AI domain settings:
  - a. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
  - b. In the left navigation pane, choose **Domains**.
  - c. Select your domain.
  - d. Choose the **App Configurations** tab.
  - e. Scroll to the **Canvas** section and choose **Edit**.
2. Enable large data processing:
  - a. In the **Large data processing configuration** section, turn on **Enable EMR Serverless for large data processing**.
  - b. Create or select an EMR Serverless role:
    - i. Choose **Create and use a new execution role** to create a new IAM role that has a trust relationship with EMR Serverless and the [AWS managed policy](#):

[AmazonSageMakerCanvasEMRServerlessExecutionRolePolicy](#) policy attached. This IAM role is assumed by Canvas to create EMR Serverless jobs.

- ii. Alternatively, if you already have an execution role with a trust relationship for EMR Serverless, then select **Use an existing execution role** and choose your role from the dropdown.
  - The existing role must have a name that begins with the prefix AmazonSageMakerCanvasEMRSExecutionAccess-.
  - The role you select should also have at least the permissions described in the [AWS managed policy: AmazonSageMakerCanvasEMRServerlessExecutionRolePolicy](#) policy.
  - The role should have an EMR Serverless trust policy, as shown below:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "EMRServerlessTrustPolicy",  
            "Effect": "Allow",  
            "Principal": {  
                "Service": "emr-serverless.amazonaws.com"  
            },  
            "Action": "sts:AssumeRole",  
            "Condition": {  
                "StringEquals": {  
                    "aws:SourceAccount": "<your-account-id>"  
                }  
            }  
        }  
    ]  
}
```

### 3. (Optional) Add Amazon S3 permissions for custom Amazon S3 buckets:

- a. The Canvas managed policy automatically grants read and write permissions for Amazon S3 buckets with sagemaker or SageMaker AI in their names. It also grants read permissions for objects in custom Amazon S3 buckets with the tag "SageMaker": "true".
- b. For custom Amazon S3 buckets without the required tag, add the following policy to your EMR Serverless role:

c.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "s3:GetObject",  
                "s3:PutObject",  
                "s3:DeleteObject"  
            ],  
            "Resource": [  
                "arn:aws:s3:::*"  
            ]  
        }  
    ]  
}
```

- d. We recommend that you scope down the permissions to specific Amazon S3 buckets that you want Canvas to access.
4. Save your changes and restart your SageMaker Canvas application.

## Scenario 2: Custom domain setup (with public internet access/without VPC)

If you created or use a custom domain, follow steps 1-3 from Scenario 1, and then do these additional steps:

1. Add permissions for the Amazon ECR `DescribeImages` operation to your Amazon SageMaker AI execution role, as Canvas utilizes public Amazon ECR Docker images for data preparation and model training:
  - a. Sign in to the AWS console and open the IAM console at <https://console.aws.amazon.com/iam/>.
  - b. Choose **Roles**.
  - c. In the search box, search for your SageMaker AI execution role by name and select it.
  - d. Add the following policy to your SageMaker AI execution role. This can be done either by adding it as a new inline policy or by appending the policy statement to an existing one. Note that an IAM role can have a maximum of 10 policies attached.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ECRDescribeImagesOperation",
            "Effect": "Allow",
            "Action": "ecr:DescribeImages",
            "Resource": [
                "arn:aws:ecr:*:*:repository/sagemaker-data-wrangler-emr-container",
                "arn:aws:ecr:*:*:repository/ap-dataprep-emr"
            ]
        }
    ]
}
```

2. Save your changes and restart your SageMaker Canvas application.

### Scenario 3: Custom domain setup (with VPC and without public internet access)

If you created or use a custom domain, follow all steps from Scenario 2, then follow these additional steps:

1. Ensure your VPC subnets are private:
  - Verify that the route table for your subnets doesn't have an entry mapping `0.0.0.0/0` to an Internet Gateway.
2. Add permissions for creating network interfaces:
  - a. When using SageMaker Canvas with EMR Serverless for large-scale data processing, EMR Serverless requires the ability to create Amazon EC2 ENIs to enable network communication between EMR Serverless applications and your VPC resources.
  - b. Add the following policy to your Amazon SageMaker AI execution role. This can be done either by adding it as a new inline policy or by appending the policy statement to an existing one. Note that an IAM role can have a maximum of 10 policies attached.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEC2ENICreation",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
        }
    ]
}
```

```
        "Resource": [
            "arn:aws:ec2:*:*:network-interface/*"
        ],
        "Condition": {
            "StringEquals": {
                "aws:CalledViaLast": "ops.emr-serverless.amazonaws.com"
            }
        }
    }
}
```

### 3. (Optional) Restrict ENI creation to specific subnets:

- a. To further secure your setup by restricting the creation of ENIs to certain subnets within your VPC, you can tag each subnet with specific conditions.
- b. Use the following IAM policy to ensure that EMR Serverless applications can only create Amazon EC2 ENIs within the allowed subnets and security groups:

```
{
    "Sid": "AllowEC2ENICreationInSubnetAndSecurityGroupWithEMRTags",
    "Effect": "Allow",
    "Action": [
        "ec2:CreateNetworkInterface"
    ],
    "Resource": [
        "arn:aws:ec2:*:*:subnet/*",
        "arn:aws:ec2:*:*:security-group/*"
    ],
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/KEY": "VALUE"
        }
    }
}
```

4. Follow the steps on the page [Configure Amazon SageMaker Canvas in a VPC without internet access](#) to set the VPC endpoint for Amazon S3, which is required by EMR Serverless and other AWS services that are used by SageMaker Canvas.
5. Save your changes and restart your SageMaker Canvas application.

By following these steps, you can enable large data processing in SageMaker Canvas for various domain setups, including those with custom VPC configurations. Remember to restart your SageMaker Canvas application after making these changes to apply the new permissions.

## Encrypt Your SageMaker Canvas Data with AWS KMS

You might have data that you want to encrypt while using Amazon SageMaker Canvas, such as your private company information or customer data. SageMaker Canvas uses AWS Key Management Service to protect your data. AWS KMS is a service that you can use to create and manage cryptographic keys for encrypting your data. For more information about AWS KMS, see [AWS Key Management Service](#) in the *AWS KMS Developer Guide*.

Amazon SageMaker Canvas provides you with several options for encrypting your data. SageMaker Canvas provides default encryption within the application for tasks such as building your model and generating insights. You can also choose to encrypt data stored in Amazon S3 to protect your data at rest. SageMaker Canvas supports importing encrypted datasets, so you can establish an encrypted workflow. The following sections describe how you can use AWS KMS encryption to protect your data while building models with SageMaker Canvas.

### Encrypt your data in SageMaker Canvas

With SageMaker Canvas, you can use two different AWS KMS encryption keys to encrypt your data in SageMaker Canvas, which you can specify when [setting up your domain](#) using the standard domain setup. These keys are specified in the following domain setup steps:

- **Step 3: Configure Applications - (Optional)** – When configuring the **Canvas storage configuration** section, you can specify an **Encryption key**. This is a KMS key that SageMaker Canvas uses for long-term storage of model objects and datasets, which are stored in the provided Amazon S3 bucket for your domain. If creating a Canvas application with the [CreateApp](#) API, use the `S3KMSKeyId` field to specify this key.
- **Step 6: Configure storage** – SageMaker Canvas uses one key for encrypting the Amazon SageMaker Studio private space that is created for your Canvas application, which includes temporary application storage, visualizations, and compute jobs (such as building models). You can use either the default AWS managed key or specify your own. If you specify your AWS KMS key, the data stored in the `/home/sagemaker-user` directory is encrypted with your key. If you don't specify an AWS KMS key, the data inside `/home/sagemaker-user` is encrypted with an AWS managed key. Regardless of whether you specify an AWS KMS key, all of the data outside of the working directory is encrypted with an AWS Managed Key. To learn more about the Studio space and your Canvas application storage, see [Store SageMaker Canvas application data in](#)

your own SageMaker AI space. If creating a Canvas application with the [CreateApp](#) API, use the KmsKeyId field to specify this key.

The preceding keys can be the same or different KMS keys.

## Prerequisites

To use your own KMS key for either of the previously described purposes, you must first grant your user's IAM role permission to use the key. Then, you can specify the KMS key when setting up your domain.

The simplest way to grant your role permission to use the key is to modify the key policy. Use the following procedure to grant your role the necessary permissions.

1. Open the [AWS KMS console](#).
2. In the **Key Policy** section, choose **Switch to policy view**.
3. Modify the key's policy to grant permissions for the kms:GenerateDataKey and kms:Decrypt actions to the IAM role. Additionally, if you're modifying the key policy that encrypts your Canvas application storage in the Studio space, grant the kms>CreateGrant action. You can add a statement that's similar to the following:

```
{  
    "Sid": "ExampleStmt",  
    "Action": [  
        "kms:CreateGrant", #this permission is only required for the key that encrypts  
        your SageMaker Canvas application storage  
        "kms:Decrypt",  
        "kms:GenerateDataKey"  
    ],  
    "Effect": "Allow",  
    "Principal": {  
        "AWS": "<arn:aws:iam::111122223333:role/Jane>"  
    },  
    "Resource": "*"  
}
```

4. Choose **Save changes**.

The less preferred method is to modify the user's IAM role to grant the user permissions to use or manage the KMS key. If you use this method, the KMS key policy must also allow access

management through IAM. To learn how to grant permission to a KMS key through the user's IAM role, see [Specifying KMS keys in IAM policy statements](#) in the *AWS KMS Developer Guide*.

## Encrypt your data in the SageMaker Canvas application

The first KMS key you can use in SageMaker Canvas is used for encrypting application data stored on Amazon Elastic Block Store (Amazon EBS) volumes and in the Amazon Elastic File System that SageMaker AI creates in your domain. SageMaker Canvas encrypts your data with this key in the underlying application and temporary storage systems created when using compute instances for building models and generating insights. SageMaker Canvas passes the key to other AWS services, such as Autopilot, whenever SageMaker Canvas initiates jobs with them to process your data.

You can specify this key by setting the `KmsKeyId` in the `CreateDomain` API call or while doing the standard domain setup in the console. If you don't specify your own KMS key, SageMaker AI uses a default AWS managed KMS key to encrypt your data in the SageMaker Canvas application.

To specify your own KMS key for use in the SageMaker Canvas application through the console, first set up your Amazon SageMaker AI domain using the **Standard setup**. Use the following procedure to complete the **Network and Storage Section** for the domain.

1. Fill out your desired Amazon VPC settings.
2. For **Encryption key**, choose **Enter a KMS key ARN**.
3. For **KMS ARN**, enter the ARN for your KMS key, which should have a format similar to the following: `arn:aws:kms:example-region-1:123456789098:key/111aa2bb-333c-4d44-5555-a111bb2c33dd`

## Encrypt your SageMaker Canvas data saved in Amazon S3

The second KMS key you can specify is used for data that SageMaker Canvas stores to Amazon S3. This KMS key is specified in the `S3KMSKeyId` field in the `CreateDomain` API call, or while doing the standard domain setup in the SageMaker AI console. SageMaker Canvas saves duplicates of your input datasets, application and model data, and output data to the Region's default SageMaker AI S3 bucket for your account. The naming pattern for this bucket is `s3://sagemaker-{Region}-{your-account-id}`, and SageMaker Canvas stores data in the `Canvas/` folder.

1. Turn on **Enable notebook resource sharing**.

2. For **S3 location for shareable notebook resources**, leave the default Amazon S3 path. Note that SageMaker Canvas does not use this Amazon S3 path; this Amazon S3 path is used for Studio Classic notebooks.
3. For **Encryption key**, choose **Enter a KMS key ARN**.
4. For **KMS ARN**, enter the ARN for your KMS key, which should have a format similar to the following: arn:aws:kms:us-east-1:111122223333:key/111aa2bb-333c-4d44-5555-a111bb2c33dd

## Import encrypted datasets from Amazon S3

Your users might have datasets that have been encrypted with a KMS key. While the preceding section shows you how to encrypt data in SageMaker Canvas and data stored to Amazon S3, you must grant your user's IAM role additional permissions if you want to import data from Amazon S3 that is already encrypted with AWS KMS.

To grant your user permissions to import encrypted datasets from Amazon S3 into SageMaker Canvas, add the following permissions to the IAM execution role that you've used for the user profile.

```
"kms:Decrypt",
"kms:GenerateDataKey"
```

To learn how to edit the IAM permissions for a role, see [Adding and removing IAM identity permissions](#) in the *IAM User Guide*. For more information about KMS keys, see [Key policies in AWS Key Management Service](#) in the *AWS KMS Developer Guide*.

## FAQs

Refer to the following FAQ items for answers to commonly asked questions about SageMaker Canvas AWS KMS support.

### Q: Does SageMaker Canvas retain my KMS key?

A: No. SageMaker Canvas may temporarily cache your key or pass it on to other AWS services (such as Autopilot), but SageMaker Canvas does not retain your KMS key.

**Q: I specified a KMS key when setting up my domain. Why did my dataset fail to import in SageMaker Canvas?**

A: Your user's IAM role may not have permissions to use that KMS key. To grant your user permissions, see the [Prerequisites](#). Another possible error is that you have a bucket policy on your Amazon S3 bucket that requires the use of a specific KMS key that doesn't match the KMS key you specified in your domain. Make sure that you specify the same KMS key for your Amazon S3 bucket and your domain.

**Q: How do I find the Region's default SageMaker AI Amazon S3 bucket for my account?**

A: The default Amazon S3 bucket follows the naming pattern `s3://sagemaker-{Region}-{your-account-id}`. The `Canvas/` folder in this bucket stores your SageMaker Canvas application data.

**Q: Can I change the default SageMaker AI Amazon S3 bucket used to store SageMaker Canvas data?**

A: No, SageMaker AI creates this bucket for you.

**Q: What does SageMaker Canvas store in the default SageMaker AI Amazon S3 bucket?**

A: SageMaker Canvas uses the default SageMaker AI Amazon S3 bucket to store duplicates of your input datasets, model artifacts, and model outputs.

**Q: What use cases are supported for using KMS keys with SageMaker Canvas?**

A: With SageMaker Canvas, you can use your own encryption keys with AWS KMS for building regression, binary and multi-class classification, and time series forecasting models, as well as for batch inference with your model.

**Store SageMaker Canvas application data in your own SageMaker AI space**

Your Amazon SageMaker Canvas application data, such as datasets that you import and your model artifacts, is stored in a *Amazon SageMaker Studio private space*. The space consists of a storage volume for your application data with 100 GB of storage per user profile, the type of the space (in this case, a Canvas application), and the image for your application's container. When you set up Canvas and launch your application for the first time, SageMaker AI creates a default private space that is assigned to your user profile and stores your Canvas data. You don't have to do any additional configuration to set up the space because SageMaker AI automatically creates the space

on your behalf. However, if you don't want to use the default space, you have the option to specify a space that you created yourself. This can be useful if you want to isolate your data. The following page shows you how to create and configure your own Studio space for storing Canvas application data.

### Note

You can only configure a custom Studio space for new Canvas applications. You can't modify the space configuration for existing Canvas applications.

## Before you begin

Your Amazon SageMaker AI domain or user profile must have at least 100 GB of storage in order to create and use the SageMaker Canvas application.

If you created your domain through the SageMaker AI console, enough storage is provisioned by default and you don't need to take any additional action. If you created your domain or user profile with the [CreateDomain](#) or [CreateUserProfile](#) APIs, then make sure that you set the MaximumEbsVolumeSizeInGb value to 100 GB or greater. To set a greater storage value, you can either create a new domain or user profile, or you can update an existing domain or user profile using the [UpdateDomain](#) or [UpdateUserProfile](#) APIs.

## Create a new space

First, create a new Studio space that is configured to store Canvas application data. This is the space that you specify when creating a new Canvas application in the next step.

To create a space, you can use the AWS SDK for Python (Boto3) or the AWS CLI.

### SDK for Python (Boto3)

The following example shows you how to use the AWS SDK for Python (Boto3) `create_space` method to create a space that you can use for Canvas applications. Make sure to specify these parameters:

- **DomainId:** Specify the ID for your SageMaker AI domain. To find your ID, you can go to the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/> and locate your domain in the **Domains** section.
- **SpaceName:** Specify a name for the new space.

- **EbsVolumeSizeInGb:** Specify the storage volume size for your space (in GB). The minimum value is 5 and the maximum is 16384.
- **SharingType:** Specify this field as **Private**. For more information, see [Amazon SageMaker Studio spaces](#).
- **OwnerUserName:** Specify the user profile name. To find user profile names associated with a domain, you can go to the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/> and locate your domain in the **Domains** section. In the domain's settings, you can view the user profiles.
- **AppType:** Specify this field as **Canvas**.

```
response = client.create_space(
    DomainId='<your-domain-id>',
    SpaceName='<your-new-space-name>',
    SpaceSettings={
        'AppType': 'Canvas',
        'SpaceStorageSettings': {
            'EbsStorageSettings': {
                'EbsVolumeSizeInGb': <storage-volume-size>
            }
        },
        'OwnershipSettings':{
            'OwnerUserName': '<your-user-profile>'
        },
        'SpaceSharingSettings':{
            'SharingType': 'Private'
        }
    }
)
```

## AWS CLI

The following example shows you how to use the AWS CLI `create-space` method to create a space that you can use for Canvas applications. Make sure to specify these parameters:

- **domain-id:** Specify the ID for your domain. To find your ID, you can go to the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/> and locate your domain in the **Domains** section.
- **space-name:** Specify a name for the new space.

- **EbsVolumeSizeinGb:** Specify the storage volume size for your space (in GB). The minimum value is 5 and the maximum is 16384.
- **SharingType:** Specify this field as **Private**. For more information, see [Amazon SageMaker Studio spaces](#).
- **OwnerUserName:** Specify the user profile name. To find user profile names associated with a domain, you can go to the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/> and locate your domain in the **Domains** section. In the domain's settings, you can view the user profiles.
- **AppType:** Specify this field as **Canvas**.

```
create-space
--domain-id <your-domain-id>
--space-name <your-new-space-name>
--space-settings '{
    "AppType": "Canvas",
    "SpaceStorageSettings": {
        "EbsStorageSettings": {"EbsVolumeSizeInGb": <storage-volume-size>}
    },
}
--ownership-settings '{"OwnerUserName": "<your-user-profile>"}'
--space-sharing-settings '{"SharingType": "Private"}'
```

You should now have a space. Keep track of your space's name for the next step.

## Create a new Canvas application

After creating a space, create a new Canvas application that specifies the space as its storage location.

To create a new Canvas application, you can use the AWS SDK for Python (Boto3) or the AWS CLI.

### **Important**

You must use the AWS SDK for Python (Boto3) or the AWS CLI to create your Canvas application. Specifying a custom space when creating Canvas applications through the SageMaker AI console isn't supported.

## SDK for Python (Boto3)

The following example shows you how to use the AWS SDK for Python (Boto3) `create_app` method to create a new Canvas application. Make sure to specify these parameters:

- `DomainId`: Specify the ID for your SageMaker AI domain.
- `SpaceName`: Specify the name of the space that you created in the previous step.
- `AppType`: Specify this field as `Canvas`.
- `AppName`: Specify `default` as the app name.

```
response = client.create_app(  
    DomainId='<your-domain-id>',  
    SpaceName='<your-space-name>',  
    AppType='Canvas',  
    AppName='default'  
)
```

## AWS CLI

The following example shows you how to use the AWS CLI `create-app` method to create a new Canvas application. Make sure to specify these parameters:

- `DomainId`: Specify the ID for your SageMaker AI domain.
- `SpaceName`: Specify the name of the space that you created in the previous step.
- `AppType`: Specify this field as `Canvas`.
- `AppName`: Specify `default` as the app name.

```
create-app  
--domain-id <your-domain-id>  
--space-name <your-space-name>  
--app-type Canvas  
--app-name default
```

You should now have a new Canvas application that uses a custom Studio space as the storage location for application data.

**A Important**

Any time you delete the Canvas application (or log out) and have to re-create the application, you must provide your space in the SpaceName field to make sure that Canvas uses your space.

The space is attached to the user profile you specified in the space configuration. You can delete your Canvas application without deleting the space, and the data stored in the space remains. The data stored in your space is only deleted if you delete your user profile, or if you directly delete the space.

## Grant Your Users Permissions to Build Custom Image and Text Prediction Models

**A Important**

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

In Amazon SageMaker Canvas, you can build [custom models](#) to meet your specific business need. Two of these custom model types are single-label image prediction and multi-category text prediction. The permissions to build these model types are included in the AWS Identity and Access Management (IAM) policy called [AmazonSageMakerCanvasFullAccess](#), which SageMaker AI attaches by default to your user's IAM execution role if you leave the [Canvas base permissions turned on](#). If you are using a custom IAM configuration, then you must explicitly add permissions to your user's IAM execution role so that they can build custom image and text prediction model types. To grant the necessary permissions to build image and text prediction models, read the following section to learn how to attach a least-permissions policy to your role.

To add the permissions to the user's IAM role, do the following:

1. Go to the [IAM console](#).
2. Choose **Roles**.
3. In the search box, search for the user's IAM role by name and select it.
4. On the page for the user's role, under **Permissions**, choose **Add permissions**.
5. Choose **Create inline policy**.
6. Select the **JSON** tab, and then paste the following least-permissions policy into the editor.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker:CreateAutoMLJobV2",  
                "sagemaker:DescribeAutoMLJobV2"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

7. Choose **Review policy**.
8. Enter a **Name** for the policy.
9. Choose **Create policy**.

For more information about AWS managed policies, see [Managed policies and inline policies](#) in the [IAM User Guide](#).

## Grant Users Permissions to Use Amazon Bedrock and Generative AI Features in Canvas

Generative AI features in Amazon SageMaker Canvas are powered by Amazon Bedrock foundation models, which are large language models (LLMs) that have the capability to understand and generate human-like text. This page describes how to grant the permissions necessary for the following features in SageMaker Canvas:

- [Chat with and compare Amazon Bedrock models](#): Access and start conversational chats with Amazon Bedrock models through SageMaker Canvas.
- [Use the Chat for data prep feature in Data Wrangler](#) : Use natural language to explore, visualize, and transform your data. This feature is powered by Anthropic Claude 2.
- [Fine-tune Amazon Bedrock foundation models](#): Fine-tune an Amazon Bedrock foundation model on your own data to receive customized responses.

In order to use these features, you must first request access to the specific Amazon Bedrock model that you want to use. Then, add the necessary AWS IAM permissions and a trust relationship with Amazon Bedrock to the user's execution role. To grant the permissions to the role, you can choose one of the following methods:

- Create a new Amazon SageMaker AI domain or user profile and turn on Amazon Bedrock permissions. For more information, see [Getting started with using Amazon SageMaker Canvas](#).
- Edit the settings for an existing Amazon SageMaker AI domain or user profile.
- Manually add permissions and a trust relationship to a domain's or user's IAM role.

## Step 1: Add Amazon Bedrock model access

Access to Amazon Bedrock models isn't granted by default, so you must go to the Amazon Bedrock console to request access to models for your AWS account.

To learn how to request access to a specific Amazon Bedrock model, following the procedure to **Add model access** on the page [Manage access to Amazon Bedrock foundation models](#) in the *Amazon Bedrock User Guide*.

## Step 2: Grant permissions to the user's IAM role

When setting up your Amazon SageMaker AI domain or user profile, the user's IAM execution role must have the [AmazonSageMakerCanvasBedrockAccess](#) policy attached, as well as a trust relationship with Amazon Bedrock, so that your user can access Amazon Bedrock models from SageMaker Canvas.

You can modify the domain settings and either create a new execution role (to which SageMaker AI attaches the required permissions for you) or specify an existing role.

Alternatively, you can manually modify the permissions for an existing IAM role through the IAM console.

Both methods are described in the following sections.

## Grant permissions through the domain settings

You can edit your domain or user profile settings to turn on the **Canvas Ready-to-use models configuration** setting and specify an Amazon Bedrock role.

To edit your domain settings and grant access to Amazon Bedrock models for Canvas users in the domain, do the following:

1. Go to the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. In the left navigation pane, choose **Domains**.
3. From the list of domains, choose your domain.
4. Choose the **App Configurations** tab.
5. In the **Canvas** section, choose **Edit**.
6. The **Edit Canvas settings** page opens. For the **Canvas Ready-to-use models configuration** section, do the following:
  - a. Turn on the **Enable Canvas Ready-to-use models option**.
  - b. For **Amazon Bedrock role**, select **Create and use a new execution role** to create a new IAM execution role that has the [AmazonSageMakerCanvasBedrockAccess](#) policy attached and a trust relationship with Amazon Bedrock. This IAM role is assumed by Amazon Bedrock when you access Amazon Bedrock models, use the chat for data prep feature, or fine-tune Amazon Bedrock models in Canvas. If you already have an execution role with a trust relationship, then select **Use an existing execution role** and choose your role from the dropdown.
7. Choose **Submit** to save your changes.

Your users should now have the necessary permissions to access Amazon Bedrock models, use the chat for data prep feature, and fine-tune Amazon Bedrock models in Canvas.

You can use the same procedure above for editing an individual user's settings, except go into the individual user's profile from the domain page and edit the user settings instead. Permissions granted to an individual user don't apply to other users in the domain, while permissions granted through the domain settings apply to all user profiles in the domain.

For more information on editing your domain settings, see [View and Edit domains](#).

## Grant permissions manually through IAM

You can manually grant users permissions to access and fine-tune Amazon Bedrock models in Canvas by adding permissions to the IAM role specified for the domain or user's profile. The IAM role must have the [AmazonSageMakerCanvasBedrockAccess](#) policy attached and a trust relationship with Amazon Bedrock.

The following section shows you how to attach the policy to your IAM role and create the trust relationship with Amazon Bedrock.

First, take note of your domain or user profile's IAM role. Note that permissions granted to an individual user don't apply to other users in the domain, while permissions granted through the domain apply to all user profiles in the domain.

To configure the IAM role and grant permissions to fine-tune foundation models in Canvas, do the following:

1. Go to the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the left navigation pane, choose **Roles**.
3. Search for the user's IAM role by name from the list of roles and select it.
4. On the **Permissions** tab, choose **Add permissions**. From the dropdown menu, choose **Attach policies**.
5. Search for the [AmazonSageMakerCanvasBedrockAccess](#) policy and select it.
6. Choose **Add permissions**.
7. Back on the IAM role's page, choose the **Trust relationships** tab.
8. Choose **Edit trust policy**.
9. In the policy editor, find the **Add a principal option** in the right panel and choose **Add**.
10. In the dialog box, for **Principal type**, select **AWS services**.
11. For **ARN**, enter **bedrock.amazonaws.com**.
12. Choose **Add principal**.
13. Choose **Update policy**.

You should now have an IAM role that has the [AmazonSageMakerCanvasBedrockAccess](#) policy attached and a trust relationship with Amazon Bedrock. For information about AWS managed policies, see [Managed policies and inline policies](#) in the *IAM User Guide*.

## Update SageMaker Canvas for Your Users

You can update to the latest version of Amazon SageMaker Canvas as either a user or an IT administrator. You can update Amazon SageMaker Canvas for a single user at a time.

To update the Amazon SageMaker Canvas application, you must delete the previous version.

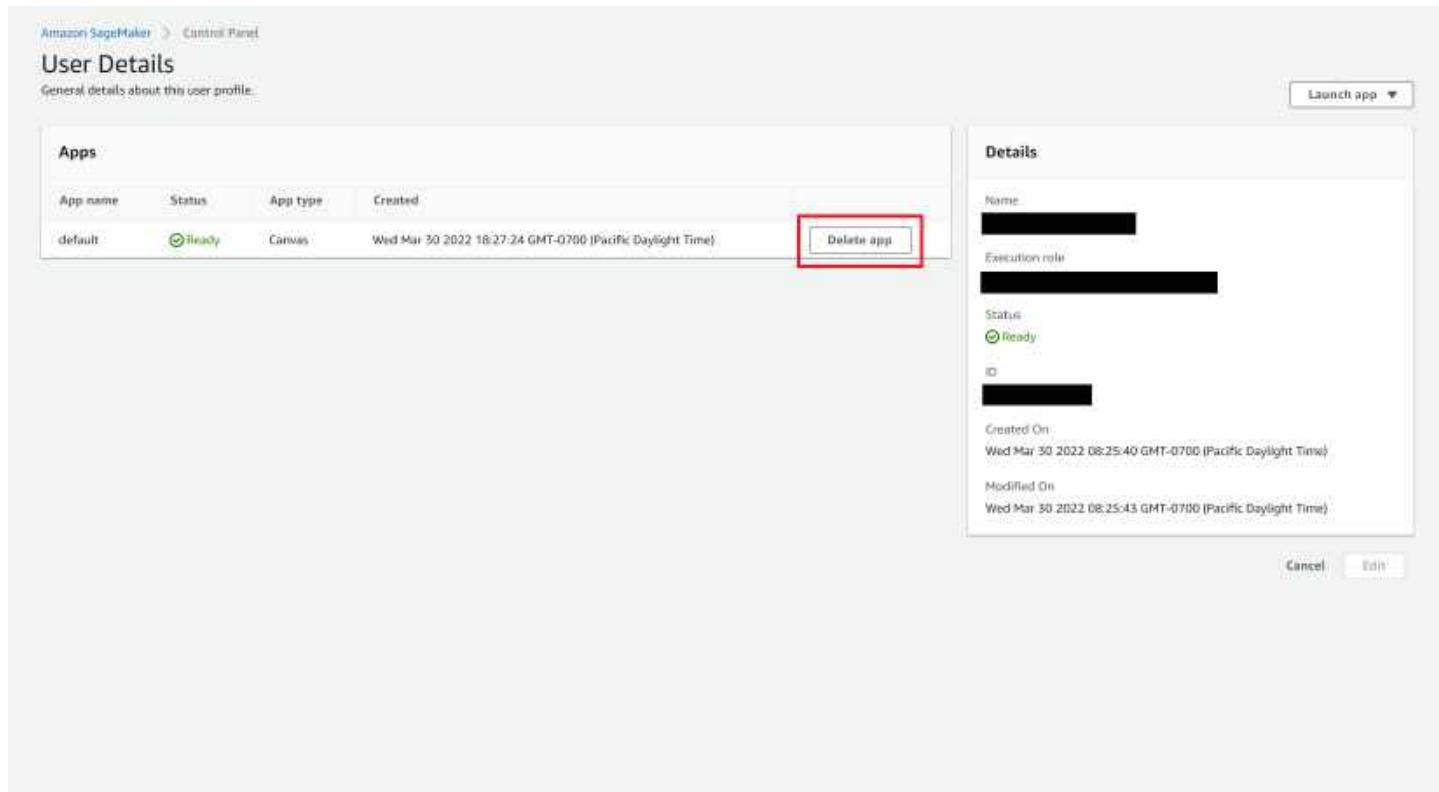
### **Important**

Deleting the previous version of Amazon SageMaker Canvas doesn't delete the data or models that the users have created.

Use the following procedure to log in to AWS, open Amazon SageMaker AI domain, and update Amazon SageMaker Canvas. The users can start using the SageMaker Canvas application when they log back in.

1. Sign in to the Amazon SageMaker AI console at [Amazon SageMaker Runtime](#).
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. On the **Domains** page, choose your domain.
5. From the list of **User profiles**, choose a user profile.
6. For the list of **Apps**, find the Canvas application (the **App type** says **Canvas**) and choose **Delete app**.
7. Complete the dialog box and choose **Confirm action**.

The following image shows the user profile page and highlights the **Delete app** action from the preceding procedure.



## Request a Quota Increase

Your users might use AWS resources in amounts that exceed those specified by their quotas. If your users are resource constrained and encounter errors in SageMaker Canvas, you can request a quota increase for them.

For more details about SageMaker AI quotas and how to request a quota increase, see [Quotas](#).

Amazon SageMaker Canvas uses the following services to process the requests of your users:

- Amazon SageMaker Autopilot
- Amazon SageMaker Studio Classic domain

For a list of the available quotas for SageMaker Canvas operations, see [Amazon SageMaker AI endpoints and quotas](#).

### Request an increase for instances to build custom models

When building a custom model, if you encounter an error during post-building analysis that tells you to increase your quota for m1.m5.2xlarge instances, use the following information to resolve the issue.

You must increase the SageMaker AI Hosting endpoint quota for the `m1.m5.2xlarge` instance type to a non-zero value in your AWS account. After building a model, SageMaker Canvas hosts the model on a SageMaker AI Hosting endpoint and uses the endpoint to generate the post-building analysis. If you don't increase the default account quota of 0 for `m1.m5.2xlarge` instances, SageMaker Canvas cannot complete this step and generates an error during post-building analysis.

For the procedure to increase the quota, see [Requesting a quota increase](#) in the *Service Quotas User Guide*.

## Grant Users Permissions to Import Amazon Redshift Data

Your users might have datasets stored in Amazon Redshift. Before users can import data from Amazon Redshift into SageMaker Canvas, you must add the `AmazonRedshiftFullAccess` managed policy to the IAM execution role that you've used for the user profile and add Amazon Redshift as a service principal to the role's trust policy. You must also associate the IAM execution role with your Amazon Redshift cluster. Complete the procedures in the following sections to give your users the required permissions to import Amazon Redshift data.

### Add Amazon Redshift permissions to your IAM role

You must grant Amazon Redshift permissions to the IAM role specified in your user profile.

To add the `AmazonRedshiftFullAccess` policy to the user's IAM role, do the following.

1. Sign in to the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Roles**.
3. In the search box, search for the user's IAM role by name and select it.
4. On the page for the user's role, under **Permissions**, choose **Add permissions**.
5. Choose **Attach policies**.
6. Search for the `AmazonRedshiftFullAccess` managed policy and select it.
7. Choose **Attach policies** to attach the policy to the role.

After attaching the policy, the role's **Permissions** section should now include `AmazonRedshiftFullAccess`.

To add Amazon Redshift as a service principal to the IAM role, do the following.

1. On the same page for the IAM role, under **Trust relationships**, choose **Edit trust policy**.

2. In the **Edit trust policy** editor, update the trust policy to add Amazon Redshift as a service principal. An IAM role that allows Amazon Redshift to access other AWS services on your behalf has a trust relationship as follows:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "Service": "redshift.amazonaws.com"  
            },  
            "Action": "sts:AssumeRole"  
        }  
    ]  
}
```

3. After editing the trust policy, choose **Update policy**.

You should now have an IAM role that has the policy `AmazonRedshiftFullAccess` attached to it and a trust relationship established with Amazon Redshift, giving users permission to import Amazon Redshift data into SageMaker Canvas. For more information about AWS managed policies, see [Managed policies and inline policies](#) in the *IAM User Guide*.

### Associate the IAM role with your Amazon Redshift cluster

In the settings for your Amazon Redshift cluster, you must associate the IAM role that you granted permissions to in the preceding section.

To associate an IAM role with your cluster, do the following.

1. Sign in to the Amazon Redshift console at <https://console.aws.amazon.com/redshiftv2/>.
2. On the navigation menu, choose **Clusters**, and then choose the name of the cluster that you want to update.
3. In the **Actions** dropdown menu, choose **Manage IAM roles**. The **Cluster permissions** page appears.
4. For **Available IAM roles**, enter either the ARN or the name of the IAM role, or choose the IAM role from the list.
5. Choose **Associate IAM role** to add it to the list of **Associated IAM roles**.

6. Choose **Save changes** to associate the IAM role with the cluster.

Amazon Redshift modifies the cluster to complete the change, and the IAM role to which you previously granted Amazon Redshift permissions is now associated with your Amazon Redshift cluster. Your users now have the required permissions to import Amazon Redshift data into SageMaker Canvas.

## Grant Your Users Permissions to Send Predictions to Amazon QuickSight

You must grant your SageMaker Canvas users permissions to send batch predictions to Amazon QuickSight. In Amazon QuickSight, users can create analyses and reports with a dataset and prepare dashboards to share their results. For more information about sending prediction to QuickSight for analysis, see [Send predictions to Amazon QuickSight](#).

To grant the necessary permissions to share batch predictions with users in QuickSight, you must add a permissions policy to the AWS Identity and Access Management (IAM) execution role that you've used for the user profile. The following section shows you how to attach a least-permissions policy to your role.

## Add the permissions policy to your IAM role

**To add the permissions policy, use the following procedure:**

1. Sign in to the IAM console at <https://console.aws.amazon.com/iam/>.
  2. Choose **Roles**.
  3. In the search box, search for the user's IAM role by name and select it.
  4. On the page for the user's role, under **Permissions**, choose **Add permissions**.
  5. Choose **Create inline policy**.
  6. Select the JSON tab, and then paste the following least-permissions policy into the editor.  
Replace the placeholders <*your-account-number*> with your own AWS account number.

```
        "quicksight>ListNamespaces",
        "quicksight>CreateDataSource",
        "quicksight>PassDataSet",
        "quicksight>PassDataSource"
    ],
    "Resource": [
        "arn:aws:quicksight:*:<your-account-number>:datasource/*",
        "arn:aws:quicksight:*:<your-account-number>:user/*",
        "arn:aws:quicksight:*:<your-account-number>:namespace/*",
        "arn:aws:quicksight:*:<your-account-number>:dataset/*"
    ]
}
]
```

7. Choose **Review policy**.
8. Enter a **Name** for the policy.
9. Choose **Create policy**.

You should now have a customer-managed IAM policy attached to your execution role that grants your Canvas users the necessary permissions to send batch predictions to users in QuickSight.

## Applications management

The following sections describe how you can manage your SageMaker Canvas applications. You can view, delete, or relaunch your applications from the **Domains** section of the SageMaker AI console.

### Topics

- [Check for active applications](#)
- [Delete an application](#)
- [Relaunch an application](#)

### Check for active applications

To check if you have any actively running SageMaker Canvas applications, use the following procedure.

1. Open the [SageMaker AI console](#).
2. On the left navigation pane, choose **Admin configurations**.

3. Under **Admin configurations**, choose **domains**.
4. On the **Domains** page, choose your domain.
5. On the **Domain details** page, under **User profiles**, select the user profile name for the Canvas application that you want to view.
6. Under **Apps**, find the application that says **Canvas** in the **App type** column.

The **Status** column displays the status of the application, such as **Ready**, **Pending**, or **Deleted**. If the application is **Ready**, then your SageMaker Canvas workspace instance is active. You can delete the application from the console or log out from the SageMaker Canvas interface.

## Delete an application

If you want to terminate your SageMaker Canvas workspace instance, you can either log out from the SageMaker Canvas application or delete your application from the SageMaker AI console. A *workspace instance* is dedicated for your use from when you start using SageMaker Canvas to the point when you stop using it. Deleting the application only terminates the workspace instance and stops workspace instance charges. Models and datasets aren't affected, but Quick build tasks automatically restart when you relaunch the application.

To delete your Canvas application through the AWS console, first close the browser tab in which your Canvas application was open. Then, use the following procedure to delete your SageMaker Canvas application.

1. Open the [SageMaker AI console](#).
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. On the **Domains** page, choose your domain.
5. On the **Domain details** page, under **User profiles**, select the user profile name for the Canvas application you want to view.
6. Under **Apps**, find the application that says **Canvas** in the **App type** column.
7. In the **Action** column, choose **Delete app**.
8. In the **Delete app** dialog box, select the **Yes, delete app** prompt, confirm the deletion by typing **delete** in the text field, and then choose **Delete**.

After you've successfully deleted the application, the **Status** column says **Deleted**. Otherwise, your application is still active.

You can also terminate the workspace instance by [logging out](#) from within the SageMaker Canvas application.

## Relaunch an application

If you delete or log out of your SageMaker Canvas application and want to relaunch the application, use the following procedure.

1. Navigate to the [SageMaker AI console](#).
2. In the navigation pane, choose **Canvas**.
3. On the SageMaker Canvas landing page, in the **Get Started** box, select your user profile from the dropdown.
4. Choose **Open Canvas** to open the application.

SageMaker Canvas begins launching the application.

You can also use the following secondary procedure if you encounter any issues with the previous procedure.

1. Open the [SageMaker AI console](#).
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. On the **Domains** page, choose your domain.
5. On the **Domain details** page, under **User profiles**, select the user profile name for the SageMaker Canvas application you want to view.
6. Choose **Launch** and select **Canvas** from the dropdown list.

SageMaker Canvas begins launching the application.

## Configure Amazon SageMaker Canvas in a VPC without internet access

The Amazon SageMaker Canvas application runs in a container in an AWS managed Amazon Virtual Private Cloud (VPC). If you want to further control access to your resources or run SageMaker Canvas without public internet access, you can configure your Amazon SageMaker AI domain and VPC settings. Within your own VPC, you can configure settings such as security groups (virtual firewalls that control inbound and outbound traffic from Amazon EC2 instances) and subnets (ranges of IP addresses in your VPC). To learn more about VPCs, see [How Amazon VPC works](#).

When the SageMaker Canvas application is running in the AWS managed VPC, it can interact with other AWS services using either an internet connection or through VPC endpoints created in a customer-managed VPC (without public internet access). SageMaker Canvas applications can access these VPC endpoints through a Studio Classic-created network interface that provides connectivity to the customer-managed VPC. The default behavior of the SageMaker Canvas application is to have internet access. When using an internet connection, the containers for the preceding jobs access AWS resources over the internet, such as the Amazon S3 buckets where you store training data and model artifacts.

However, if you have security requirements to control access to your data and job containers, we recommend that you configure SageMaker Canvas and your VPC so that your data and containers aren't accessible over the internet. SageMaker AI uses the VPC configuration settings you specify when setting up your domain for SageMaker Canvas.

If you want to configure your SageMaker Canvas application without internet access, you must configure your VPC settings when you onboard to [Amazon SageMaker AI domain](#), set up VPC endpoints, and grant the necessary AWS Identity and Access Management permissions. For information about configuring a VPC in Amazon SageMaker AI, see [Choose an Amazon VPC](#). The following sections describe how to run SageMaker Canvas in a VPC without public internet access.

## Configure Amazon SageMaker Canvas in a VPC without internet access

You can send traffic from SageMaker Canvas to other AWS services through your own VPC. If your own VPC doesn't have public internet access and you've set up your domain in **VPC only** mode, then SageMaker Canvas won't have public internet access as well. This includes all requests, such as accessing datasets in Amazon S3 or training jobs for standard builds, and the requests go through VPC endpoints in your VPC instead of the public internet. When you onboard to domain and [Choose an Amazon VPC](#), you can specify your own VPC as the default VPC for the domain, along with your desired security group and subnet settings. Then, SageMaker AI creates a network interface in your VPC that SageMaker Canvas uses to access VPC endpoints in your VPC.

Make sure that you set up one or more security groups in your VPC with inbound and outbound rules that allow [TCP traffic within the security group](#). This is required for connectivity between the Jupyter Server application and the Kernel Gateway applications. You must allow access to at least ports in the range 8192–65535. Also, make sure to create a distinct security group for each user profile and add inbound access from that same security group. We do not recommend reusing a domain level security group for user profiles. If the domain level security group allows inbound access to itself, all applications in the domain have access to all other applications in the domain. Note that the security group and subnet settings are set after you finish onboarding to domain.

When onboarding to domain, if you choose **Public internet only** as the network access type, the VPC is SageMaker AI managed and allows internet access.

You can change this behavior by choosing **VPC only** so that SageMaker AI sends all traffic to a network interface that SageMaker AI creates in your specified VPC. When you choose this option, you must provide the subnets, security groups, and VPC endpoints that are necessary to communicate with the SageMaker API and SageMaker AI Runtime, and various AWS services, such as Amazon S3 and Amazon CloudWatch, that are used by SageMaker Canvas. Note that you can only import data from Amazon S3 buckets located in the same Region as your VPC.

The following procedures show how you can configure these settings to use SageMaker Canvas without the internet.

### Step 1: Onboard to Amazon SageMaker AI domain

To send SageMaker Canvas traffic to a network interface in your own VPC instead of over the internet, specify the VPC you want to use when onboarding to [Amazon SageMaker AI domain](#). You must also specify at least two subnets in your VPC that SageMaker AI can use. Choose **Standard setup** and do the following procedure when configuring the **Network and Storage Section** for the domain.

1. Select your desired **VPC**.
2. Choose two or more **Subnets**. If you don't specify the subnets, SageMaker AI uses all of the subnets in the VPC.
3. Choose one or more **Security group(s)**.
4. Choose **VPC Only** to turn off direct internet access in the AWS managed VPC where SageMaker Canvas is hosted.

After disabling internet access, finish the onboarding process to set up your domain. For more information about the VPC settings for Amazon SageMaker AI domain, see [Choose an Amazon VPC](#).

### Step 2: Configure VPC endpoints and access

#### Note

In order to configure Canvas in your own VPC, you must enable private DNS hostnames for your VPC endpoints. For more information, see [Connect to SageMaker AI Through a VPC Interface Endpoint](#).

SageMaker Canvas only accesses other AWS services to manage and store data for its functionality. For example, it connects to Amazon Redshift if your users access an Amazon Redshift database. It can connect to an AWS service such as Amazon Redshift using an internet connection or a VPC endpoint. Use VPC endpoints if you want to set up connections from your VPC to AWS services that don't use the public internet.

A VPC endpoint creates a private connection to an AWS service that uses a networking path that is isolated from the public internet. For example, if you set up access to Amazon S3 using a VPC endpoint from your own VPC, then the SageMaker Canvas application can access Amazon S3 by going through the network interface in your VPC and then through the VPC endpoint that connects to Amazon S3. The communication between SageMaker Canvas and Amazon S3 is private.

For more information about configuring VPC endpoints for your VPC, see [AWS PrivateLink](#). If you are using Amazon Bedrock models in Canvas with a VPC, for more information about controlling access to your data, see [Protect jobs using a VPC](#) in the *Amazon Bedrock User Guide*.

The following are the VPC endpoints for each service you can use with SageMaker Canvas:

Service	Endpoint	Endpoint type
AWS Application Auto Scaling	com.amazo naws. <i>Region</i> .application-autoscaling	Interface
Amazon Athena	com.amazo naws. <i>Region</i> .athena	Interface
Amazon SageMaker AI	com.amazo naws. <i>Region</i> .sagemaker.api  com.amazo naws. <i>Region</i> .sagemaker.runtime  com.amazo naws. <i>Region</i> .notebook	Interface

Service	Endpoint	Endpoint type
Amazon SageMaker AI Data Science Assistant	com.amazo naws. <i>Region</i> .sagemaker-data-science-assistant	Interface
AWS Security Token Service	com.amazonaws. <i>Region</i> .sts	Interface
Amazon Elastic Container Registry (Amazon ECR)	com.amazo naws. <i>Region</i> .ecr.api  com.amazo naws. <i>Region</i> .ecr.dkr	Interface
Amazon Elastic Compute Cloud (Amazon EC2)	com.amazonaws. <i>Region</i> .ec2	Interface
Amazon Simple Storage Service (Amazon S3)	com.amazonaws. <i>Region</i> .s3	Gateway
Amazon Redshift	com.amazo naws. <i>Region</i> .redshift-data	Interface
AWS Secrets Manager	com.amazo naws. <i>Region</i> .secretsmanager	Interface
AWS Systems Manager	com.amazonaws. <i>Region</i> .ssm	Interface
Amazon CloudWatch	com.amazo naws. <i>Region</i> .monitoring	Interface
Amazon CloudWatch Logs	com.amazonaws. <i>Region</i> .logs	Interface
Amazon Forecast	com.amazo naws. <i>Region</i> .forecast  com.amazo naws. <i>Region</i> .forecastquery	Interface

Service	Endpoint	Endpoint type
Amazon Textract	com.amazonoaws.Region.textract	Interface
Amazon Comprehend	com.amazonoaws.Region.comprehend	Interface
Amazon Rekognition	com.amazonoaws.Region.rekognition	Interface
AWS Glue	com.amazonaws.Region.glue	Interface
AWS Application Auto Scaling	com.amazonoaws.Region.application-autoscaling	Interface
Amazon Relational Database Service (Amazon RDS)	com.amazonaws.Region.rds	Interface
Amazon Bedrock (see note after table)	com.amazonoaws.Region.bedrock-runtime	Interface
Amazon Kendra	com.amazonoaws.Region.kendra	Interface
Amazon EMR Serverless	com.amazonoaws.Region.emr-serverless	Interface
Amazon Q Developer (see note after table)	com.amazonaws.Region.q	Interface

### Note

The Amazon Q Developer VPC endpoint is currently available only in the US East (N. Virginia) region. To connect to it from other regions, you can choose one of the following options based on your security and infrastructure preferences:

- **Set up a NAT Gateway.** Configure a NAT Gateway in your VPC's private subnet to enable internet connectivity for the Q Developer endpoint. For more information, see [Setting up a NAT Gateway in a VPC Private Subnet](#).
- **Enable cross-region VPC endpoint access.** Set up cross-region VPC endpoint access for Q Developer. Use this option to connect securely without requiring internet access. For more information, see [Configuring Cross-Region VPC Endpoint Access](#).

### Note

For Amazon Bedrock, the interface endpoint service name `com.amazonaws.Region.bedrock` has been deprecated. Create a new VPC endpoint with the service name listed in the preceding table.

Additionally, you can't fine-tune foundation models from Canvas VPCs with no internet access. This is because Amazon Bedrock doesn't support VPC endpoints for model customization APIs. To learn more about fine-tuning foundation models in Canvas, see [Fine-tune foundation models](#).

You must also add an endpoint policy for Amazon S3 to control AWS principal access to your VPC endpoint. For information about how to update your VPC endpoint policy, see [Control access to VPC endpoints using endpoint policies](#).

The following are two VPC endpoint policies that you can use. Use the first policy if you only want to grant access to the basic functionality of Canvas, such as importing data and creating models. Use the second policy if you want to grant access to the additional [generative AI features](#) in Canvas.

#### Basic VPC endpoint policy

The following policy grants the necessary access to your VPC endpoint for basic operations in Canvas.

```
{  
    "Effect": "Allow",  
    "Action": [  
        "s3:GetObject",  
        "s3:PutObject",  
    ]  
}
```

```
        "s3>DeleteObject",
        "s3>CreateBucket",
        "s3>GetBucketCors",
        "s3>GetBucketLocation"
    ],
    "Resource": [
        "arn:aws:s3:::*SageMaker*",
        "arn:aws:s3:::*Sagemaker*",
        "arn:aws:s3:::*sagemaker*"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "s3>ListBucket",
        "s3>ListAllMyBuckets"
    ],
    "Resource": "*"
}
```

## Generative AI VPC endpoint policy

The following policy grants the necessary access to your VPC endpoint for basic operations in Canvas, as well as using generative AI foundation models.

```
{
    "Effect": "Allow",
    "Action": [
        "s3GetObject",
        "s3PutObject",
        "s3DeleteObject",
        "s3CreateBucket",
        "s3GetBucketCors",
        "s3GetBucketLocation"
    ],
    "Resource": [
        "arn:aws:s3:::*SageMaker*",
        "arn:aws:s3:::*Sagemaker*",
        "arn:aws:s3:::*sagemaker*",
        "arn:aws:s3:::*fmeval/datasets*",
        "arn:aws:s3:::*jumpstart-cache-prod*"
    ]
},
```

```
{  
    "Effect": "Allow",  
    "Action": [  
        "s3>ListBucket",  
        "s3>ListAllMyBuckets"  
    ],  
    "Resource": "*"  
}
```

### Step 3: Grant IAM permissions

The SageMaker Canvas user must have the necessary AWS Identity and Access Management permissions to allow connection to the VPC endpoints. The IAM role to which you give permissions must be the same one you used when onboarding to Amazon SageMaker AI domain. You can attach the SageMaker AI managed `AmazonSageMakerFullAccess` policy to the IAM role for the user to give the user the required permissions. If you require more restrictive IAM permissions and use custom policies instead, then give the user's role the `ec2:DescribeVpcEndpointServices` permission. SageMaker Canvas requires these permissions to verify the existence of the required VPC endpoints for standard build jobs. If it detects these VPC endpoints, then standard build jobs run by default in your VPC. Otherwise, they will run in the default AWS managed VPC.

For instructions on how to attach the `AmazonSageMakerFullAccess` IAM policy to your user's IAM role, see [Adding and removing IAM identity permissions](#).

To grant your user's IAM role the granular `ec2:DescribeVpcEndpointServices` permission, use the following procedure.

1. Sign in to the AWS Management Console and open the [IAM console](#).
2. In the navigation pane, choose **Roles**.
3. In the list, choose the name of the role to which you want to grant permissions.
4. Choose the **Permissions** tab.
5. Choose **Add permissions** and then choose **Create inline policy**.
6. Choose the **JSON** tab and enter the following policy, which grants the `ec2:DescribeVpcEndpointServices` permission:

```
{  
    "Version": "2012-10-17",  
    "Statement": [
```

```
{  
    "Sid": "VisualEditor0",  
    "Effect": "Allow",  
    "Action": "ec2:DescribeVpcEndpointServices",  
    "Resource": "*"  
}  
]  
}
```

7. Choose **Review policy**, and then enter a **Name** for the policy (for example, `VPCEndpointPermissions`).
8. Choose **Create policy**.

The user's IAM role should now have permissions to access the VPC endpoints configured in your VPC.

#### (Optional) Step 4: Override security group settings for specific users

If you are an administrator, you might want different users to have different VPC settings, or user-specific VPC settings. When you override the default VPC's security group settings for a specific user, these settings are passed on to the SageMaker Canvas application for that user.

You can override the security groups that a specific user has access to in your VPC when you set up a new user profile in Studio Classic. You can use the [CreateUserProfile](#) SageMaker API call (or [create\\_user\\_profile](#) with the [AWS CLI](#)), and then in the UserSettings, you can specify the SecurityGroups for the user.

## Set up connections to data sources with OAuth

The following section describes the steps you must take to set up OAuth connections to data sources from SageMaker Canvas. [OAuth](#) is a common authentication platform for granting access to resources without sharing passwords. With OAuth, you can quickly connect to your data from Canvas and import it for building models. Canvas currently supports OAuth for Snowflake and Salesforce Data Cloud.

### Note

You can only establish one OAuth connection for each data source.

## Set up OAuth for Salesforce Data Cloud

To set up OAuth for Salesforce Data Cloud, follow these general steps:

1. Sign in to Salesforce Data Cloud.
2. In Salesforce Data Cloud, create a new app connection and do the following:
  - a. Enable OAuth settings.
  - b. When prompted for a callback URL (or the URL of the resource accessing your data), specify the URL for your Canvas application. The Canvas application URL follows this format: `https://<domain-id>.studio.<region>.sagemaker.aws/canvas/default`
  - c. Copy the consumer key and secret.
  - d. Copy your authorization URL and token URL.

For more detailed instructions about performing the preceding tasks in Salesforce Data Cloud, see [Import data from Salesforce Data Cloud](#) in the Data Wrangler documentation for importing data from Salesforce Data Cloud.

After enabling access from Salesforce Data Cloud and getting your connection information, you must create an [AWS Secrets Manager](#) secret to store the information and add it to your Amazon SageMaker AI domain or user profile. Note that you can add a secret to both a domain and user profile, but Canvas looks for secrets in the user profile first.

To add a secret to your domain or user profile, do the following:

1. Go to the [Amazon SageMaker AI console](#).
2. Choose **domains** in the navigation pane.
3. From the list of **domains**, choose your domain.
  - a. If adding your secret to your domain, do the following:
    - i. Choose the domain.
    - ii. On the **domain settings** page, choose the **domain settings** tab.
    - iii. Choose **Edit**.
  - b. If adding the secret to your user profile, do the following:
    - i. Choose the user's domain.

- ii. On the **domain settings** page, choose the user profile.
  - iii. On the **User Details** page, choose **Edit**.
4. In the navigation pane, choose **Canvas settings**.
  5. For **OAuth settings**, choose **Add OAuth configuration**.
  6. For **Data source**, select **Salesforce Data Cloud**.
  7. For **Secret Setup**, select **Create a new secret**. Alternatively, if you already created an AWS Secrets Manager secret with your credentials, enter the ARN for the secret. If creating a new secret, do the following:
    - a. For **Identity Provider**, select **SALESFORCE**.
    - b. For **Client ID**, **Client Secret**, **Authorization URL**, and **Token URL**, enter all of the information you gathered from Salesforce Data Cloud in the previous procedure.
  8. Save your domain or user profile settings.

You should now be able to create a connection to your data in Salesforce Data Cloud from Canvas.

## Set up OAuth for Snowflake

To set up authentication for Snowflake, Canvas supports identity providers that you can use instead of having users directly enter their credentials into Canvas.

The following are links to the Snowflake documentation for the identity providers that Canvas supports:

- [Azure AD](#)
- [Okta](#)
- [Ping Federate](#)

The following process describes the general steps you must take. For more detailed instructions about performing these steps, you can refer to the [Setting up Snowflake OAuth Access](#) section in the Data Wrangler documentation for importing data from Snowflake.

To set up OAuth for Snowflake, do the following:

1. Register Canvas as an application with the identity provider. This requires specifying a redirect URL to Canvas, which should follow this format: `https://<domain-id>.studio.<region>.sagemaker.aws/canvas/default`

2. Within the identity provider, create a server or API that sends OAuth tokens to Canvas so that Canvas can access Snowflake. When setting up the server, use the authorization code and refresh token grant types, specify the access token lifetime, and set a refresh token policy. Additionally, within the External OAuth Security Integration for Snowflake, enable `external_oauth_any_role_mode`.
3. Get the following information from the identity provider: token URL, authorization URL, client ID, client secret. For Azure AD, also retrieve the OAuth scope credentials.
4. Store the information retrieved in the previous step in an AWS Secrets Manager secret.
  - a. For Okta and Ping Federate, the secret should look like the following format:

```
{"token_url": "https://identityprovider.com/oauth2/example-portion-of-URL-path/v2/token",
"client_id": "example-client-id", "client_secret": "example-client-secret",
"identity_provider": "OKTA" | "PING_FEDERATE",
"authorization_url": "https://identityprovider.com/oauth2/example-portion-of-URL-path/v2/authorize"}
```

- b. For Azure AD, the secret should also include the OAuth scope credentials as the `datasource_oauth_scope` field.

After configuring the identity provider and the secret, you must create an [AWS Secrets Manager](#) secret to store the information and add it to your Amazon SageMaker AI domain or user profile. Note that you can add a secret to both a domain and user profile, but Canvas looks for secrets in the user profile first.

To add a secret to your domain or user profile, do the following:

1. Go to the [Amazon SageMaker AI console](#).
2. Choose **domains** in the navigation pane.
3. From the list of **domains**, choose your domain.
  - a. If adding your secret to your domain, do the following:
    - i. Choose the domain.
    - ii. On the **domain settings** page, choose the **domain settings** tab.
    - iii. Choose **Edit**.
  - b. If adding the secret to your user profile, do the following:

- i. Choose the user's domain.
  - ii. On the **domain settings** page, choose the user profile.
  - iii. On the **User Details** page, choose **Edit**.
4. In the navigation pane, choose **Canvas settings**.
  5. For **OAuth settings**, choose **Add OAuth configuration**.
  6. For **Data source**, select **Snowflake**.
  7. For **Secret Setup**, select **Create a new secret**. Alternatively, if you already created an AWS Secrets Manager secret with your credentials, enter the ARN for the secret. If creating a new secret, do the following:
    - a. For **Identity Provider**, select **SNOWFLAKE**.
    - b. For **Client ID**, **Client Secret**, **Authorization URL**, and **Token URL**, enter all of the information you gathered from the identity provider in the previous procedure.
  8. Save your domain or user profile settings.

You should now be able to create a connection to your data in Snowflake from Canvas.

## Generative AI assistance for solving ML problems in Canvas using Amazon Q Developer

While using Amazon SageMaker Canvas, you can chat with Amazon Q Developer in natural language to leverage generative AI and solve problems. Q Developer is an assistant that helps you translate your goals into machine learning (ML) tasks and describes each step of the ML workflow. Q Developer helps Canvas users reduce the amount of time, effort, and data science expertise required to leverage ML and make data-driven decisions for their organizations.

Through a conversation with Q Developer, you can initiate actions in Canvas such as preparing data, building an ML model, making predictions, and deploying a model. Q Developer makes suggestions for next steps and provides you with context as you complete each step. It also informs you of results; for example, Canvas can transform your dataset according to best practices, and Q Developer can list the transforms that were used and why.

Amazon Q Developer is available in SageMaker Canvas at no additional cost to both Amazon Q Developer Pro Tier and Free Tier users. However, standard charges apply for resources such as the

SageMaker Canvas workspace instance and any resources used for building or deploying models. For more information about pricing, see [Amazon SageMaker Canvas pricing](#).

Use of Amazon Q is licensed to you under [MIT's 0 License](#) and subject to the [AWS Responsible AI Policy](#). When you use Q Developer from outside the US, Q Developer processes data across US regions. For more information, see [Cross region inference in Amazon Q Developer](#).

## How it works

Amazon Q Developer is a generative AI powered assistant available in SageMaker Canvas that you can query using natural language. Q Developer makes suggestions for each step of the machine learning workflow, explaining concepts and providing you with options and more details as needed. You can use Q Developer for help with regression, binary classification, and multi-class classification use cases.

For example, to predict customer churn, upload a dataset of historical customer churn information to Canvas through Q Developer. Q Developer suggests an appropriate ML model type and steps to fix dataset issues, build a model, and make predictions.

### Important

Amazon Q Developer is intended for conversations about machine learning problems within SageMaker Canvas. It guides users through Canvas actions and optionally answers questions about AWS services. Q Developer processes model inputs only in English. For more information about how you can use Q Developer, see [Amazon Q Developer features](#) in the [Amazon Q Developer User Guide](#).

## Supported regions

Amazon Q Developer is available within SageMaker Canvas in the following AWS Regions:

- US East (N. Virginia)
- US East (Ohio)
- US West (Oregon)
- Asia Pacific (Mumbai)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)

- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (Paris)

## Amazon Q Developer capabilities available in Canvas

The following list summarizes the Canvas tasks with which Q Developer can provide assistance:

- **Describe your objective** – Q Developer can suggest an ML model type and general approach to solve your problem.
- **Import and analyze datasets** – Tell Q Developer where your dataset is stored or upload a file to save it as a Canvas dataset. Prompt Q Developer to identify any issues in your dataset, such as outliers or missing values. Q Developer provides summary statistics about your dataset and lists any identified issues.

Q Developer supports queries about the following statistics for individual columns:

- Numeric columns – number of valid values, feature type, mean, median, minimum, maximum, standard deviation, 25th percentile, 75th percentile, number of outliers
- Categorical columns – number of missing values, number of valid values, feature type, most frequent, most frequent category, most frequent category count, least frequent, least frequent category, least frequent category count, categories
- **Fix dataset issues** – Prompt Q Developer to use Canvas's data transformation capabilities to create a revised version of your dataset. Canvas creates a Data Wrangler data flow and applies transforms according to data science best practices. For more information, see [Data preparation](#).

If you want to do more advanced data analysis or data preparation tasks than you can accomplish with Q Developer, then we recommend that you go to the Data Wrangler data flow interface.

- **Train a model** – Q Developer tells you the recommended ML model type for your problem and a proposed model building configuration. You can use the suggested default settings to do a quick build, or you can modify the configuration and do a standard build. When ready, prompt Q Developer to build your Canvas model.

All of the custom model types are supported. For more information about model types and quick versus standard builds, see [How custom models work](#).

- **Evaluate model accuracy** – After building a model, Q Developer provides a summary of how the model scores across various metrics. These metrics help you determine the usefulness and accuracy of your model. Q Developer can explain any concept or metric in detail.

To view full details and visualizations, open the model from the chat or the **My Models** page of Canvas. For more information, see [Model evaluation](#).

- **Get predictions for new data** – You can upload a new dataset and prompt Q Developer to help you open the prediction feature of Canvas.

Q Developer opens a new window in the application where you can either make a single prediction or make batch predictions with a new dataset. For more information, see [Predictions with custom models](#).

- **Deploy a model** – To deploy your model for production, ask Q Developer to help you deploy your model through Canvas. Q Developer opens a new window in which you can configure your deployment.

After deploying, view your deployment details either 1) on the **My Models** page of Canvas in the model's **Deploy** tab, or 2) on the **ML Ops** page in the **Deployments** tab. For more information, see [Deploy your models to an endpoint](#).

## Prerequisites

To use Amazon Q Developer to build ML models in SageMaker Canvas, complete the following prerequisites:

### Set up a Canvas application

Make sure that you have a Canvas application set up. For information about how to set up a Canvas application, see [Getting started with using Amazon SageMaker Canvas](#).

### Grant Q Developer permissions

To access Q Developer while using Canvas, you must attach the necessary permissions to the AWS IAM role used for your SageMaker AI domain or user profile. You can do this through the console or by manually attaching an AWS managed policy.

Permissions attached at the domain level apply to all user profiles in the domain, unless individual permissions are granted or revoked at the user profile level.

## SageMaker AI console method

You can grant permissions by editing the SageMaker AI domain or user profile settings.

To grant permissions through the domain settings in the SageMaker AI console, do the following:

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **Domains**.
4. From the list of domains, select your domain.
5. On the **Domain details** page, select the **App configurations** tab.
6. In the **Canvas** section, choose **Edit**.
7. On the **Edit Canvas settings** page, go to the **Amazon Q Developer** section and do the following:
  - a. Turn on **Enable Amazon Q Developer in SageMaker Canvas for natural language ML** to add the permissions to chat with Q Developer in Canvas to your domain's execution role.
  - b. (Optional) Turn on **Enable Amazon Q Developer chat for general AWS questions** if you want to ask Q Developer questions about various AWS services (for example: Describe how Athena works).

### Note

When making general AWS queries to Q Developer, your requests route through the US East (N. Virginia) AWS Region. To prevent your data from routing through US East (N. Virginia), turn off the **Enable Amazon Q Developer chat for general AWS questions** toggle.

## Manual method

Attach the [AmazonSageMakerCanvasSMDataScienceAssistantAccess](#) policy to the AWS IAM role used for your domain or user profile. For more information about how to do this, see [Adding and removing IAM identity permissions](#) in the *AWS IAM User Guide*.

### (Optional) Configure access to Q Developer from your VPC

If you have a VPC that is configured without public internet access, you can add a VPC endpoint for Q Developer. For more information, see [Configure Amazon SageMaker Canvas in a VPC without internet access](#).

## Getting started

To use Amazon Q Developer to build ML models in SageMaker Canvas, do the following:

1. Open your SageMaker Canvas application.
2. In the left navigation pane, choose **Amazon Q**.
3. Choose **Start a new conversation** to open a new chat.

When you start a new chat, Q Developer prompts you to state your problem or provide a dataset.



# Amazon Q

Your generative AI assistant



Hello,

I'm Amazon Q, your generative AI assistant. I can help you explore your data, and help you build ML models. Please tell me about yourself and the problem you are looking to solve today. Feel free to get started with a few examples below.

💡 I am a customer service director for an e-commerce platform and want to predict which customers are at risk of churning so we can proactively address their concerns

💡 I am a credit risk analyst at a bank and want to classify loan applicants into multiple risk categories (low risk, moderate risk, high risk) based on their financial characteristics and current economic indicators

💡 I am a healthcare administrator at a hospital and want to predict patient readmission rates to improve post-discharge care and reduce unplanned return visits

💡 I am a consumer electronics sales analyst and want to forecast product sales for the next 3 months.

[Learn more about what the assistant can do for you](#) ⓘ

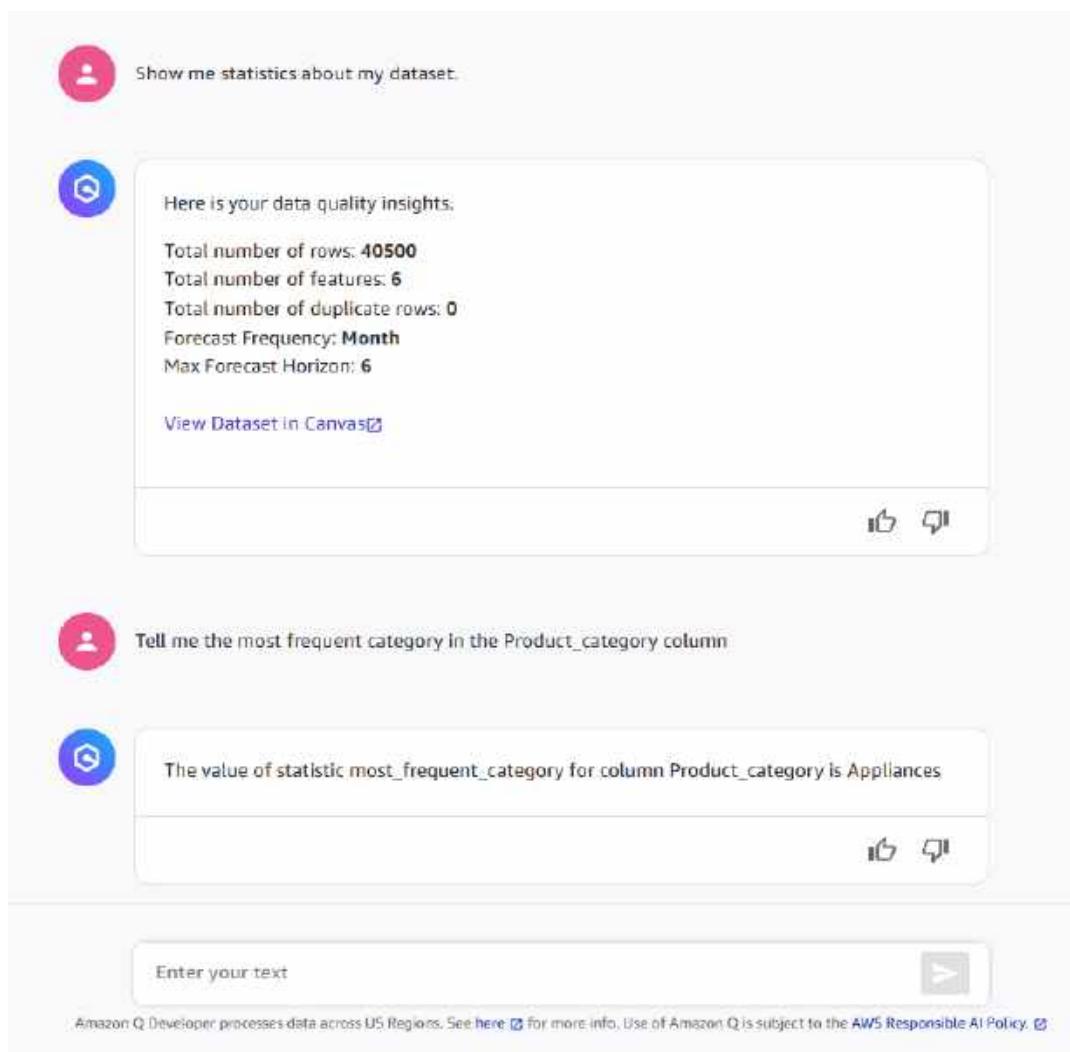
Tell me about a business problem you have



Amazon Q Developer processes data across US Regions. See [here](#) ⓘ for more info. Use of Amazon Q is subject to the [AWS Responsible AI Policy](#). ⓘ

After importing your data, you can ask Q Developer to provide you with summary statistics about your dataset, or you can ask questions about specific columns. For a list of the different statistics that Q Developer supports, see the preceding section [Amazon Q Developer capabilities available in](#)

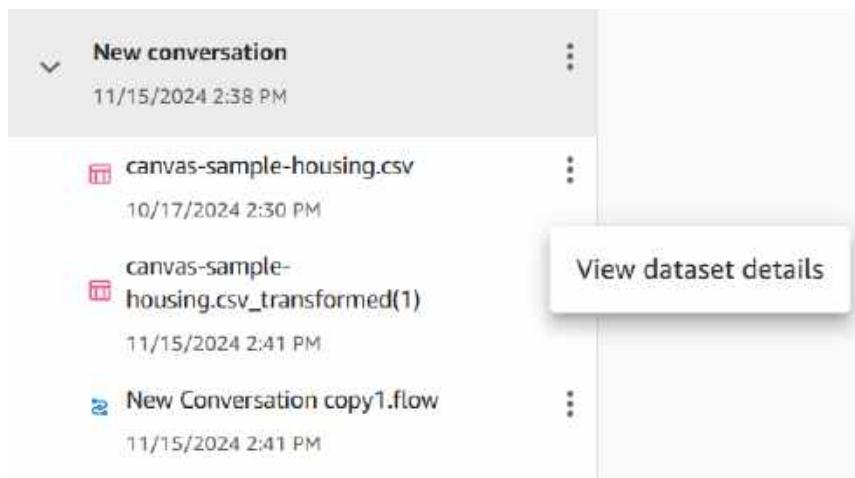
[Canvas](#). The following screenshot shows an example of asking for dataset statistics and the most frequent category in a product category column.



Q Developer tracks any Canvas artifacts you import or create during the conversation, such as transformed datasets and models. You can access them from the chat or other Canvas application tabs. For example, if Q Developer fixes issues in your dataset, you can access the new, transformed dataset from the following places:

- The artifacts sidebar in the Q Developer chat interface
- The **Datasets** page of Canvas, where you can view both your original and transformed datasets. The transformed dataset has the **Built by Amazon Q** label added to it.
- The **Data Wrangler** page of Canvas, where Q Developer creates a new data flow for your dataset

The following screenshot shows the original dataset and the transformed dataset in the sidebar of a chat.



When your data is ready, ask Q Developer to help build a Canvas model. Q Developer might prompt you to confirm a few fields and review the build configuration. If you use the default build configuration, then your model is built using a quick build. If you want to customize any part of your build configuration, such as selecting the algorithms used or changing the objective metric, then your model is built with a standard build.

The following screenshot shows how you can prompt Q Developer to initiate a Canvas model build with only a few prompts. This example uses the default configuration to start a quick build.

Here are the configurations that will be used to build the model. Let me know if this looks good to you, or feel free to ask me to change any of the parameters you see here.

**Model Type**  
Time series forecasting  
*Your model will forecast demand by using past values to predict future data values*

**Target Column**  
demand

**Item ID Column**  
item\_id

**Time stamp column**  
time\_stamp

**Objective metric**  
AverageWeightedQuantileLoss  
*The Weighted Quantile Loss (WQL) score measures the performance of models that predict quantiles. It assesses how well the predicted quantiles match the actual quantiles of a target variable.*

**Training method**  
Quick Build  
*Canvas builds a custom model using a single algorithm in about 15 minutes or less, balancing speed over accuracy.*

**Forecast quantiles**  
0.1, 0.5, 0.9

**Forecast length**  
1

↶ ↷

Your training job has been successfully launched!

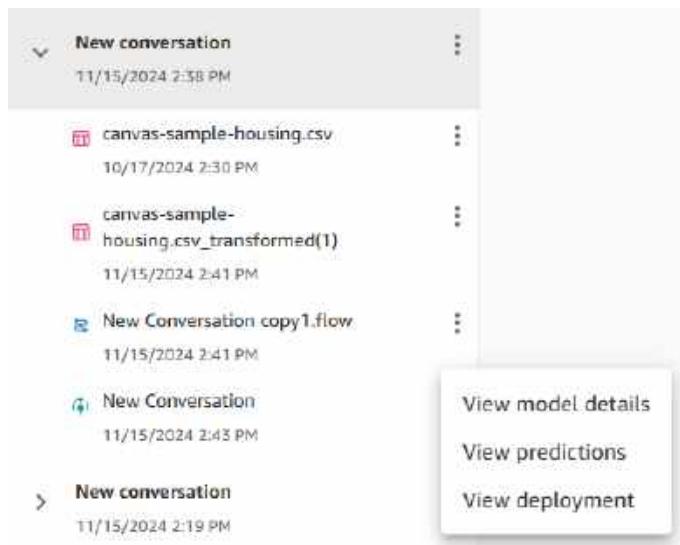
We are now processing your request, and the job is in progress. This may take a while depending on the size of the data and complexity of the model.

You will receive updates in the My Models tab as the job progresses. In the meantime, feel free to continue exploring or ask any questions.

Enter your text  ➤

Amazon Q Developer processes data across US Regions. See here [↗](#) for more info. Use of Amazon Q is subject to the AWS Responsible AI Policy. [↗](#)

After building your model, you can perform additional actions using either natural language in the chat or the artifacts sidebar menu. For example, you can view model details and metrics, make predictions, or deploy the model. The following screenshot shows the sidebar where you can choose these additional options.



You can also perform any of these actions by going to the **My Models** page of Canvas and selecting your model. From your model's page, you can navigate to the **Analyze**, **Predict**, and **Deploy** tabs to view model metrics and visualizations, make predictions, and manage deployments, respectively.

## Logging Q Developer conversations with AWS CloudTrail

AWS CloudTrail is a service that records actions taken by users, roles, or AWS services in Amazon SageMaker AI. CloudTrail captures API calls resulting from your interactions with Amazon Q Developer (a conversational AI assistant) while using SageMaker Canvas (a no-code ML interface). CloudTrail data shows request details, the IP address of the requester, who made the request, and when.

Your interactions with Q Developer are sent as `SendConversation` API calls to the SageMaker AI Data Science Assistant service, which is an internal service that Canvas leverages on the backend. The event source for `SendConversation` API calls is `sagemaker-data-science-assistant.amazonaws.com`.

**Note**

For privacy and security reasons, the content of your conversations is hidden in the logs, appearing as HIDDEN\_DUE\_TO\_SECURITY\_REASONS in the request and response elements.

To learn more about CloudTrail, see the [AWS CloudTrail User Guide](#). To learn more about CloudTrail in SageMaker AI, see [Logging Amazon SageMaker AI API calls using AWS CloudTrail](#).

The following is an example log file entry for the SendConversation API:

```
{  
    "eventVersion": "1.10",  
    "userIdentity": {  
        "type": "AssumedRole",  
        "principalId": "AROA123456789EXAMPLE:user-Isengard",  
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/user",  
        "accountId": "111122223333",  
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",  
        "sessionContext": {  
            "sessionIssuer": {  
                "type": "Role",  
                "principalId": "AROA123456789EXAMPLE",  
                "arn": "arn:aws:iam::111122223333:role/Admin",  
                "accountId": "111122223333",  
                "userName": "Admin"  
            },  
            "attributes": {  
                "creationDate": "2024-11-11T22:04:37Z",  
                "mfaAuthenticated": "false"  
            }  
        }  
    },  
    "eventTime": "2024-11-11T22:09:22Z",  
    "eventSource": "sagemaker-data-science-assistant.amazonaws.com",  
    "eventName": "SendConversation",  
    "awsRegion": "us-west-2",  
    "sourceIPAddress": "192.0.2.0",  
    "userAgent": "Boto3/1.33.13 md/Botocore#1.33.13 ua/2.0 os/  
linux#5.10.227-198.884.amzn2int.x86_64 md/arch#x86_64 lang/python#3.7.16 md/  
pyimpl#CPython cfg/retry-mode#legacy Botocore/1.33.13",  
}
```

```
"requestParameters": {
    "conversation": [
        {
            "utteranceId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "utterance": "HIDDEN_DUE_TO_SECURITY_REASONS",
            "timestamp": "Feb 4, 2020, 7:46:29 AM",
            "utteranceType": "User"
        }
    ],
    "utteranceId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
},
"responseElements": {
    "responseCode": "CHAT_RESPONSE",
    "conversationId": "1234567890abcdef0",
    "response": {
        "chat": {
            "body": "HIDDEN_DUE_TO_SECURITY_REASONS"
        }
    }
},
"requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"readOnly": false,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "123456789012",
"eventCategory": "Management",
"tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "gamma.us-west-2.data-science-assistant.sagemaker.aws.dev"
}
}
```

## Data import

Amazon SageMaker Canvas supports importing tabular, image, and document data. You can import datasets from your local machine, Amazon services such as Amazon S3 and Amazon Redshift, and external data sources. When importing datasets from Amazon S3, you can bring a dataset of any size. Use the datasets that you import to build models and make predictions for other datasets.

Each use case for which you can build a custom model accepts different types of input. For example, if you want to build a single-label image classification model, then you should import image data. For more information about the different model types and the data they accept, see [How custom models work](#). You can import data and build custom models in SageMaker Canvas for the following data types:

- **Tabular** (CSV, Parquet, or tables)
  - Categorical – Use categorical data to build custom categorical prediction models for 2 and 3+ category prediction.
  - Numeric – Use numeric data to build custom numeric prediction models.
  - Text – Use text data to build custom multi-category text prediction models.
  - Timeseries – Use timeseries data to build custom time series forecasting models.
- **Image** (JPG or PNG) – Use image data to build custom single-label image prediction models.
- **Document** (PDF, JPG, PNG, TIFF) – Document data is only supported for SageMaker Canvas Ready-to-use models. To learn more about Ready-to-use models that can make predictions for document data, see [Ready-to-use models](#).

You can import data into Canvas from the following data sources:

- Local files on your computer
- Amazon S3 buckets
- Amazon Redshift provisioned clusters (not Amazon Redshift Serverless)
- AWS Glue Data Catalog through Amazon Athena
- Amazon Aurora
- Amazon Relational Database Service (Amazon RDS)
- Salesforce Data Cloud
- Snowflake
- Databricks, SQLServer, MariaDB, and other popular databases through JDBC connectors
- Over 40 external SaaS platforms, such as SAP OData

For a full list of data sources from which you can import, see the following table:

Source	Type	Supported data types
Local file upload	Local	Tabular, Image, Document
Amazon Aurora	Amazon internal	Tabular
Amazon S3 bucket	Amazon internal	Tabular, Image, Document
Amazon RDS	Amazon internal	Tabular
Amazon Redshift provisioned clusters (not Redshift Serverless)	Amazon internal	Tabular
AWS Glue Data Catalog (through Amazon Athena)	Amazon internal	Tabular
<a href="#">Databricks</a>	External	Tabular
Snowflake	External	Tabular
<a href="#">Salesforce Data Cloud</a>	External	Tabular
SQLServer	External	Tabular
MySQL	External	Tabular
PostgreSQL	External	Tabular
MariaDB	External	Tabular
<a href="#">Amplitude</a>	External SaaS platform	Tabular
<a href="#">CircleCI</a>	External SaaS platform	Tabular
<a href="#">DocuSign Monitor</a>	External SaaS platform	Tabular
<a href="#">Domo</a>	External SaaS platform	Tabular
<a href="#">Datadog</a>	External SaaS platform	Tabular

Source	Type	Supported data types
<a href="#">Dynatrace</a>	External SaaS platform	Tabular
<a href="#">Facebook Ads</a>	External SaaS platform	Tabular
<a href="#">Facebook Page Insights</a>	External SaaS platform	Tabular
<a href="#">Google Ads</a>	External SaaS platform	Tabular
<a href="#">Google Analytics 4</a>	External SaaS platform	Tabular
<a href="#">Google Search Console</a>	External SaaS platform	Tabular
<a href="#">GitHub</a>	External SaaS platform	Tabular
<a href="#">GitLab</a>	External SaaS platform	Tabular
<a href="#">Infor Nexus</a>	External SaaS platform	Tabular
<a href="#">Instagram Ads</a>	External SaaS platform	Tabular
<a href="#">Jira Cloud</a>	External SaaS platform	Tabular
<a href="#">LinkedIn Ads</a>	External SaaS platform	Tabular
<a href="#">LinkedIn Ads</a>	External SaaS platform	Tabular
<a href="#">Mailchimp</a>	External SaaS platform	Tabular
<a href="#">Marketo</a>	External SaaS platform	Tabular
<a href="#">Microsoft Teams</a>	External SaaS platform	Tabular
<a href="#">Mixpanel</a>	External SaaS platform	Tabular
<a href="#">Okta</a>	External SaaS platform	Tabular
<a href="#">Salesforce</a>	External SaaS platform	Tabular
<a href="#">Salesforce Marketing Cloud</a>	External SaaS platform	Tabular

Source	Type	Supported data types
<a href="#">Salesforce Pardot</a>	External SaaS platform	Tabular
<a href="#">SAP OData</a>	External SaaS platform	Tabular
<a href="#">SendGrid</a>	External SaaS platform	Tabular
<a href="#">ServiceNow</a>	External SaaS platform	Tabular
<a href="#">Singular</a>	External SaaS platform	Tabular
<a href="#">Slack</a>	External SaaS platform	Tabular
<a href="#">Stripe</a>	External SaaS platform	Tabular
<a href="#">Trend Micro</a>	External SaaS platform	Tabular
<a href="#">Typeform</a>	External SaaS platform	Tabular
<a href="#">Veeva</a>	External SaaS platform	Tabular
<a href="#">Zendesk</a>	External SaaS platform	Tabular
<a href="#">Zendesk Chat</a>	External SaaS platform	Tabular
<a href="#">Zendesk Sell</a>	External SaaS platform	Tabular
<a href="#">Zendesk Sunshine</a>	External SaaS platform	Tabular
<a href="#">Zoom Meetings</a>	External SaaS platform	Tabular

For instructions on how to import data and information regarding input data requirements, such as the maximum file size for images, see [Create a dataset](#).

Canvas also provides several sample datasets in your application to help you get started. To learn more about the SageMaker AI-provided sample datasets you can experiment with, see [Use sample datasets](#).

After you import a dataset into Canvas, you can update the dataset at any time. You can do a manual update or you can set up a schedule for automatic dataset updates. For more information, see [Update a dataset](#).

For more information specific to each dataset type, see the following sections:

## Tabular

To import data from an external data source (such as a Snowflake database or a SaaS platform), you must authenticate and connect to the data source in the Canvas application. For more information, see [Connect to data sources](#).

If you want to import datasets larger than 5 GB from Amazon S3 into Canvas, you can achieve faster sampling by using Amazon Athena to query and sample the data from Amazon S3.

After creating datasets in Canvas, you can prepare and transform your data using the data preparation functionality of Data Wrangler. You can use Data Wrangler to handle missing values, transform your features, join multiple datasets into a single dataset, and more. For more information, see [Data preparation](#).

### Tip

As long as your data is arranged into tables, you can join datasets from various sources, such as Amazon Redshift, Amazon Athena, or Snowflake.

## Image

For information about how to edit an image dataset and perform tasks such as assigning or reassigning labels, adding images, or deleting images, see [Edit an image dataset](#).

## Create a dataset

### Note

If you're importing datasets larger than 5 GB into Amazon SageMaker Canvas, we recommend that you use the [Data Wrangler feature](#) in Canvas to create a data flow. Data Wrangler supports advanced data preparation features such as [joining](#) and [concatenating](#) data. After you create a data flow, you can export your data flow as a Canvas dataset and begin building a model. For more information, see [Export to create a model](#).

The following sections describe how to create a dataset in Amazon SageMaker Canvas. For custom models, you can create datasets for tabular and image data. For Ready-to-use models, you can use tabular and image datasets as well as document datasets. Choose your workflow based on the following information:

- For categorical, numeric, text, and timeseries data, see [Import tabular data](#).
- For image data, see [Import image data](#).
- For document data, see [Import document data](#).

A dataset can consist of multiple files. For example, you might have multiple files of inventory data in CSV format. You can upload these files together as a dataset as long as the schema (or column names and data types) of the files match.

Canvas also supports managing multiple versions of your dataset. When you create a dataset, the first version is labeled as V1. You can create a new version of your dataset by updating your dataset. You can do a manual update, or you can set up an automated schedule for updating your dataset with new data. For more information, see [Update a dataset](#).

When you import your data into Canvas, make sure that it meets the requirements in the following table. The limitations are specific to the type of model you're building.

Limit	2 category, 3+ category, numeric, and time series models	Text prediction models	Image prediction models	*Document data for Ready-to- use models
Supported file types	CSV and Parquet (local upload, Amazon S3, or databases)  JSON (databases)	CSV and Parquet (local upload, Amazon S3, or databases)	JPG, PNG	PDF, JPG, PNG, TIFF
Maximum file size	Local upload: 5 GB	Local upload: 5 GB	30 MB per image	5 MB per document

Limit	2 category, 3+ category, numeric, and time series models	Text prediction models	Image prediction models	*Document data for Ready-to- use models
	Data sources: PBs	Data sources: PBs		
Maximum number of files you can upload at a time	30	30	N/A	N/A
Maximum number of columns	1,000	1,000	N/A	N/A
Maximum number of entries (rows, images, or documents) for <b>Quick builds</b>	N/A	7500 rows	5000 images	N/A
Maximum number of entries (rows, images, or documents) for <b>Standard builds</b>	N/A	150,000 rows	180,000 images	N/A
Minimum number of entries (rows) for <b>Quick builds</b>	2 category: 500 rows  3+ category, numeric, time series: N/A	N/A	N/A	N/A
Minimum number of entries (rows, images, or documents) for <b>Standard builds</b>	250 rows	50 rows	50 images	N/A
Minimum number of entries (rows or images) per label	N/A	25 rows	25 rows	N/A

Limit	2 category, 3+ category, numeric, and time series models	Text prediction models	Image prediction models	*Document data for Ready-to- use models
Minimum number of labels	2 category: 2  3+ category: 3  Numeric, time series: N/A	2	2	N/A
Minimum sample size for random sampling	500	N/A	N/A	N/A
Maximum sample size for random sampling	200,000	N/A	N/A	N/A
Maximum number of labels	2 category: 2  3+ category, numeric, time series: N/A	1000	1000	N/A

\*Document data is currently only supported for [Ready-to-use models](#) that accept document data. You can't build a custom model with document data.

Also note the following restrictions:

- When importing data from an Amazon S3 bucket, make sure that your Amazon S3 bucket name doesn't contain a .. If your bucket name contains a .., you might experience errors when trying to import data into Canvas.
- For tabular data, Canvas disallows selecting any file with extensions other than .csv, .parquet, .parq, and .pqt for both local upload and Amazon S3 import. CSV files can

use any common or custom delimiter, and they must not have newline characters except when denoting a new row.

- For tabular data using Parquet files, note the following:
  - Parquet files can't include complex types like maps and lists.
  - The column names of Parquet files can't contain spaces.
  - If using compression, Parquet files must use either gzip or snappy compression types. For more information about the preceding compression types, see the [gzip documentation](#) and the [snappy documentation](#).
- For image data, if you have any unlabeled images, you must label them before building your model. For information about how to assign labels to images within the Canvas application, see [Edit an image dataset](#).
- If you set up automatic dataset updates or automatic batch prediction configurations, you can only create a total of 20 configurations in your Canvas application. For more information, see [How to manage automations](#).

After you import a dataset, you can view your datasets on the **Datasets** page at any time.

## Import tabular data

With tabular datasets, you can build categorical, numeric, time series forecasting, and text prediction models. Review the limitations table in the preceding **Import a dataset** section to ensure that your data meets the requirements for tabular data.

Use the following procedure to import a tabular dataset into Canvas:

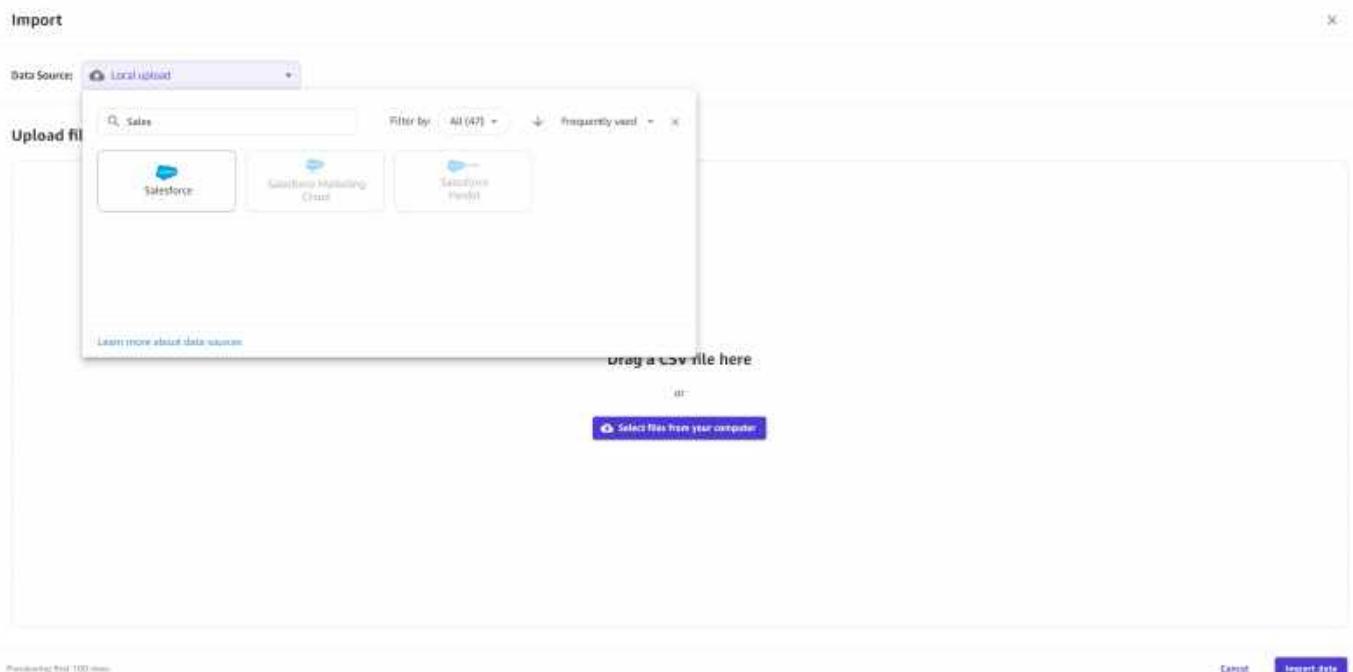
1. Open your SageMaker Canvas application.
2. In the left navigation pane, choose **Datasets**.
3. Choose **Import data**.
4. From the dropdown menu, choose **Tabular**.
5. In the popup dialog box, in the **Dataset name** field, enter a name for the dataset and choose **Create**.
6. On the **Create tabular dataset** page, open the **Data Source** dropdown menu.
7. Choose your data source:
  - To upload files from your computer, choose **Local upload**.

- To import data from another source, such as an Amazon S3 bucket or a Snowflake database, search for your data source in the **Search data source bar**. Then, choose the tile for your desired data source.

**Note**

You can only import data from the tiles that have an active connection. If you want to connect to a data source that is unavailable to you, contact your administrator. If you're an administrator, see [Connect to data sources](#).

The following screenshot shows the **Data Source** dropdown menu.



- (Optional) If you're connecting to an Amazon Redshift or Snowflake database for the first time, a dialog box appears to create a connection. Fill out the dialog box with your credentials and choose **Create connection**. If you already have a connection, choose your connection.
- From your data source, select your files to import. For local upload and importing from Amazon S3, you can select files. For Amazon S3 only, you also have the option to directly enter the S3 URI, alias, or ARN of your bucket or S3 access point in the **Input S3 endpoint** field, and then choose files to import. For database sources, you can drag-and-drop data tables from the left navigation pane.

10. (Optional) For tabular data sources that support SQL querying (such as Amazon Redshift, Amazon Athena, or Snowflake), you can choose **Edit in SQL** to make SQL queries before importing them.

The following screenshot shows the **Edit SQL** view for an Amazon Athena data source.

The screenshot shows the "Import" interface in Amazon SageMaker AI. At the top, there's a "Data Source" dropdown set to "Athena". Below it is a search bar and a sidebar with a tree view of data sources under "Amazon Data Catalog", including "sagemaker\_workshop\_2", "sagemakerflow", and "titanic". The main area is titled "Edit SQL" and contains the following SQL query:

```
SELECT "passengerid", "survived", "pclass", "name", "sex", "age", "sibsp", "parch", "ticket"
FROM "s3://titanic-dataset/titanic.csv"
```

Below the SQL editor is an "Import preview" table showing the first 10 rows of the dataset. The columns are: passengerid, survived, pclass, name, sex, age, sibsp, parch, fare, ticket. The data includes entries like:生存者率为0的女性乘客，名为Ismay, Mrs. John Bradley Thayer, 年龄2, 票价31.0328；生存者率为1的女性乘客，名为Cummings, Mrs. John Bradley Thayer, 年龄38, 票价13.75；等等。

11. Choose **Preview dataset** to preview your data before importing it.
12. In the **Import settings**, enter a **Dataset name** or use the default dataset name.
13. (Optional) For data that you import from Amazon S3, you are shown the **Advanced** settings and can fill out the following fields:
  - a. Toggle the **Use first row as header** option on if you want to use the first row of your dataset as the column names. If you selected multiple files, this applies to each file.
  - b. If you're importing a CSV file, for the **File encoding (CSV)** dropdown, select your dataset file's encoding. UTF-8 is the default.
  - c. For the **Delimiter** dropdown, select the delimiter that separates each cell in your data. The default delimiter is ,. You can also specify a custom delimiter.
  - d. Select **Multi-line detection** if you'd like Canvas to manually parse your entire dataset for multi-line cells. By default, this option is not selected and Canvas determines whether or not to use multi-line support by taking a sample of your data. However, Canvas might not detect any multi-line cells in the sample. If you have multi-line cells, we recommend that

you select the **Multi-line detection** option to force Canvas to check your entire dataset for multi-line cells.

#### 14. When you're ready to import your data, choose **Create dataset**.

While your dataset is importing into Canvas, you can see your datasets listed on the **Datasets** page. From this page, you can [View your dataset details](#).

When the **Status** of your dataset shows as Ready, Canvas successfully imported your data and you can proceed with [building a model](#).

If you have a connection to a data source, such as an Amazon Redshift database or a SaaS connector, you can return to that connection. For Amazon Redshift and Snowflake, you can add another connection by creating another dataset, returning to the **Import data** page, and choosing the **Data Source** tile for that connection. From the dropdown menu, you can open the previous connection or choose **Add connection**.

 **Note**

For SaaS platforms, you can only have one connection per data source.

### Import image data

With image datasets, you can build single-label image prediction custom models, which predict a label for an image. Review the limitations in the preceding **Import a dataset** section to ensure that your image dataset meets the requirements for image data.

 **Note**

You can only import image datasets from local file upload or an Amazon S3 bucket. Also, for image datasets, you must have at least 25 images per label.

Use the following procedure to import an image dataset into Canvas:

1. Open your SageMaker Canvas application.
2. In the left navigation pane, choose **Datasets**.
3. Choose **Import data**.

4. From the dropdown menu, choose **Image**.
5. In the popup dialog box, in the **Dataset name** field, enter a name for the dataset and choose **Create**.
6. On the **Import** page, open the **Data Source** dropdown menu.
7. Choose your data source. To upload files from your computer, choose **Local upload**. To import files from Amazon S3, choose **Amazon S3**.
8. From your computer or Amazon S3 bucket, select the images or folders of images that you want to upload.
9. When you're ready to import your data, choose **Import data**.

While your dataset is importing into Canvas, you can see your datasets listed on the **Datasets** page. From this page, you can [View your dataset details](#).

When the **Status** of your dataset shows as Ready, Canvas successfully imported your data and you can proceed with [building a model](#).

When you are building your model, you can edit your image dataset, and you can assign or re-assign labels, add images, or delete images from your dataset. For more information about how to edit your image dataset, see [Edit an image dataset](#).

## Import document data

The Ready-to-use models for expense analysis, identity document analysis, document analysis, and document queries support document data. You can't build a custom model with document data.

With document datasets, you can generate predictions for expense analysis, identity document analysis, document analysis, and document queries Ready-to-use models. Review the limitations table in the [Create a dataset](#) section to ensure that your document dataset meets the requirements for document data.

### Note

You can only import document datasets from local file upload or an Amazon S3 bucket.

Use the following procedure to import a document dataset into Canvas:

1. Open your SageMaker Canvas application.

2. In the left navigation pane, choose **Datasets**.
3. Choose **Import data**.
4. From the dropdown menu, choose **Document**.
5. In the popup dialog box, in the **Dataset name** field, enter a name for the dataset and choose **Create**.
6. On the **Import** page, open the **Data Source** dropdown menu.
7. Choose your data source. To upload files from your computer, choose **Local upload**. To import files from Amazon S3, choose **Amazon S3**.
8. From your computer or Amazon S3 bucket, select the document files that you want to upload.
9. When you're ready to import your data, choose **Import data**.

While your dataset is importing into Canvas, you can see your datasets listed on the **Datasets** page. From this page, you can [View your dataset details](#).

When the **Status** of your dataset shows as Ready, Canvas has successfully imported your data.

On the **Datasets** page, you can choose your dataset to preview it, which shows you up to the first 100 documents of your dataset.

## **View your dataset details**

For each of your datasets, you can view all of the files in a dataset, the dataset's version history, and any auto update configurations for the dataset. From the **Datasets** page, you can also initiate actions such as [Update a dataset](#) or [How custom models work](#).

To view the details for a dataset, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **Datasets**.
3. From the list of datasets, choose your dataset.

On the **Data** tab, you can see a preview of your data. If you choose **Dataset details**, you can see all of the files that are part of your dataset. Choose a file to see only the data from that file in the preview. For image datasets, the preview only shows you the first 100 images of your dataset.

On the **Version history** tab, you can see a list of all of the versions of your dataset. A new version is made whenever you update a dataset. To learn more about updating a dataset, see [Update a dataset](#). The following screenshot shows the **Version history** tab in the Canvas application.

Datasets / Sales_dataset <span style="border: 1px solid #ccc; border-radius: 50%; padding: 2px;">V1</span>						<a href="#">Update dataset</a>	<a href="#">+ Create a model</a>	...
Data	Version history	Auto updates				<a href="#">Dataset details</a>		
Version	Created	Type	Files	Cells (Columns x Rows)	Status			
V6	03/11/2021 12:13 PM	Automatic update	2	20,000 (12 x 1,250)	Ready			
V5	03/11/2021 12:13 PM	Automatic update	2	20,000 (12 x 1,250)	Ready			
V4	03/11/2021 12:13 PM	Automatic update	2	20,000 (12 x 1,250)	Ready			
V3	03/11/2021 12:13 PM	Automatic update	2	20,000 (12 x 1,250)	Ready			
V2	03/11/2021 12:13 PM	Manual update	2	20,000 (12 x 1,250)	Ready			
V1	03/11/2021 12:13 PM	Base data	2	20,000 (12 x 1,250)	Ready			

Rows per page: 25 ▾ 1-6 of 6 < >

On the **Auto updates** tab, you can enable auto updates for the dataset and set up a configuration to update your dataset on a regular schedule. To learn more about setting up auto updates for a dataset, see [Configure automatic updates for a dataset](#). The following screenshot shows the **Auto updates** tab with auto updates turned on and a list of auto update jobs that have been performed on the dataset.

Sales\_dataset V1

Update dataset  + Create a model

Data Version history Auto updates  Dataset details

Auto update enabled  Delete  Edit

Configuration created	Input dataset	Frequency	Starting time	Next job scheduled
3/30/2023 3:15 PM	customerchurn.csv	Hourly	04/01/2023 8:00 AM	04/01/2023 9:00 AM

Job history

Job created	Files	Cells (Columns x Rows)	Status
03/11/2021 12:13 PM	2	20,000 (12 x 1,250)	<span>Failed: {Dataset name} (V#) failed to auto update.</span>
03/11/2021 12:13 PM	2	20,000 (12 x 1,250)	<span>Failed: {Dataset name} (V#) failed to auto update.</span>
03/11/2021 12:13 PM	2	20,000 (12 x 1,250)	Ready
03/11/2021 12:13 PM	2	20,000 (12 x 1,250)	Ready
03/11/2021 12:13 PM	2	20,000 (12 x 1,250)	Ready

Rows per page: 25  1-6 of 6

## Update a dataset

After importing your initial dataset into Amazon SageMaker Canvas, you might have additional data that you want to add to your dataset. For example, you might get inventory data at the end of every week that you want to add to your dataset. Instead of importing your data multiple times, you can update your existing dataset and add or remove files from it.

### Note

You can only update datasets that you have imported through local upload or Amazon S3.

You can update your dataset either manually or automatically. For more information about automatic dataset updates, see [Configure automatic updates for a dataset](#).

Every time you update your dataset, Canvas creates a new version of your dataset. You can only use the latest version of your dataset to build a model or generate predictions. For more information about viewing the version history of your dataset, see [View your dataset details](#).

You can also use dataset updates with automated batch predictions, which starts a batch prediction job whenever you update your dataset. For more information, see [Batch predictions in SageMaker Canvas](#).

The following section describes how to do manual updates to your dataset.

## Manually update a dataset

To do a manual update, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **Datasets**.
3. From the list of datasets, choose the dataset you want to update.
4. Choose the **Update dataset** dropdown menu and choose **Manual update**. You are taken to the import data workflow.
5. From the **Data source** dropdown menu, choose either **Local upload** or **Amazon S3**.
6. The page shows you a preview of your data. From here, you can add or remove files from the dataset. If you're importing tabular data, the schema of the new files (column names and data types) must match the schema of the existing files. Additionally, your new files must not exceed the maximum dataset size or file size. For more information about these limitations, see [Import a dataset](#).

 **Note**

If you add a file with the same name as an existing file in your dataset, the new file overwrites the old version of the file.

7. When you're ready to save your changes, choose **Update dataset**.

You should now have a new version of your dataset.

On the **Datasets** page, you can choose the **Version history** tab to see all of the versions of your dataset and the history of both manual and automatic updates you've made.

## Configure automatic updates for a dataset

After importing your initial dataset into Amazon SageMaker Canvas, you might have additional data that you want to add to your dataset. For example, you might get inventory data at the end of every week that you want to add to your dataset. Instead of importing your data multiple times, you can update your existing dataset and add or remove files from it.

 **Note**

You can only update datasets that you have imported through local upload or Amazon S3.

With automatic dataset updates, you specify a location where Canvas checks for files at a frequency you specify. If you import new files during the update, the schema of the files must match the existing dataset exactly.

Every time you update your dataset, Canvas creates a new version of your dataset. You can only use the latest version of your dataset to build a model or generate predictions. For more information about viewing the version history of your dataset, see [View your dataset details](#).

You can also use dataset updates with automated batch predictions, which starts a batch prediction job whenever you update your dataset. For more information, see [Batch predictions in SageMaker Canvas](#).

The following section describes how to do automatic updates to your dataset.

An automatic update is when you set up a configuration for Canvas to update your dataset at a given frequency. We recommend that you use this option if you regularly receive new files of data that you want to add to your dataset.

When you set up the auto update configuration, you specify an Amazon S3 location where you upload your files and a frequency at which Canvas checks the location and imports files. Each instance of Canvas updating your dataset is referred to as a *job*. For each job, Canvas imports all of the files in the Amazon S3 location. If you have new files with the same names as existing files in your dataset, Canvas overwrites the old files with the new files.

For automatic dataset updates, Canvas doesn't perform schema validation. If the schema of files imported during an automatic update don't match the schema of the existing files or exceed the size limitations (see [Import a dataset](#) for a table of file size limitations), then you get errors when your jobs run.

### Note

You can only set up a maximum of 20 automatic configurations in your Canvas application. Additionally, Canvas only does automatic updates while you're logged in to your Canvas application. If you log out of your Canvas application, automatic updates pause until you log back in.

To configure automatic updates for your dataset, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **Datasets**.
3. From the list of datasets, choose the dataset you want to update.
4. Choose the **Update dataset** dropdown menu and choose **Automatic update**. You are taken to the **Auto update** tab for the dataset.
5. Turn on the **Auto update enabled** toggle.
6. For **Specify a data source**, enter the Amazon S3 path to a folder where you plan to regularly upload files.
7. For **Choose a frequency**, select **Hourly**, **Weekly**, or **Daily**.
8. For **Specify a starting time**, use the calendar and time picker to select when you want the first auto update job to start.
9. When you're ready to create the auto update configuration, choose **Save**.

Canvas begins the first job of your auto update cadence at the specified starting time.

## View your automatic dataset update jobs

To view the job history for your automatic dataset updates in Amazon SageMaker Canvas, on your dataset details page, choose the **Auto updates** tab.

Each automatic update to a dataset shows as a job in the **Auto updates** tab under the **Job history** section. For each job, you can see the following:

- **Job created** – The timestamp for when Canvas started updating the dataset.
- **Files** – The number of files in the dataset.
- **Cells (Columns x Rows)** – The number of columns and rows in the dataset.

- **Status** – The status of the dataset after the update. If the job was successful, the status is **Ready**. If the job failed for any reason, the status is **Failed**, and you can hover over the status for more details.

## Edit your automatic dataset update configuration

You might want to make changes to your auto update configuration for a dataset, such as changing the frequency of the updates. You might also want to turn off your automatic update configuration to pause the updates to your dataset.

To make changes to your auto update configuration for a dataset, go to the **Auto updates** tab of your dataset and choose **Edit** to make changes to the configuration.

To pause your dataset updates, turn off your automatic configuration. You can turn off auto updates by going to the **Auto updates** tab of your dataset and turning the **Enable auto updates** toggle off. You can turn this toggle back on at any time to resume the update schedule.

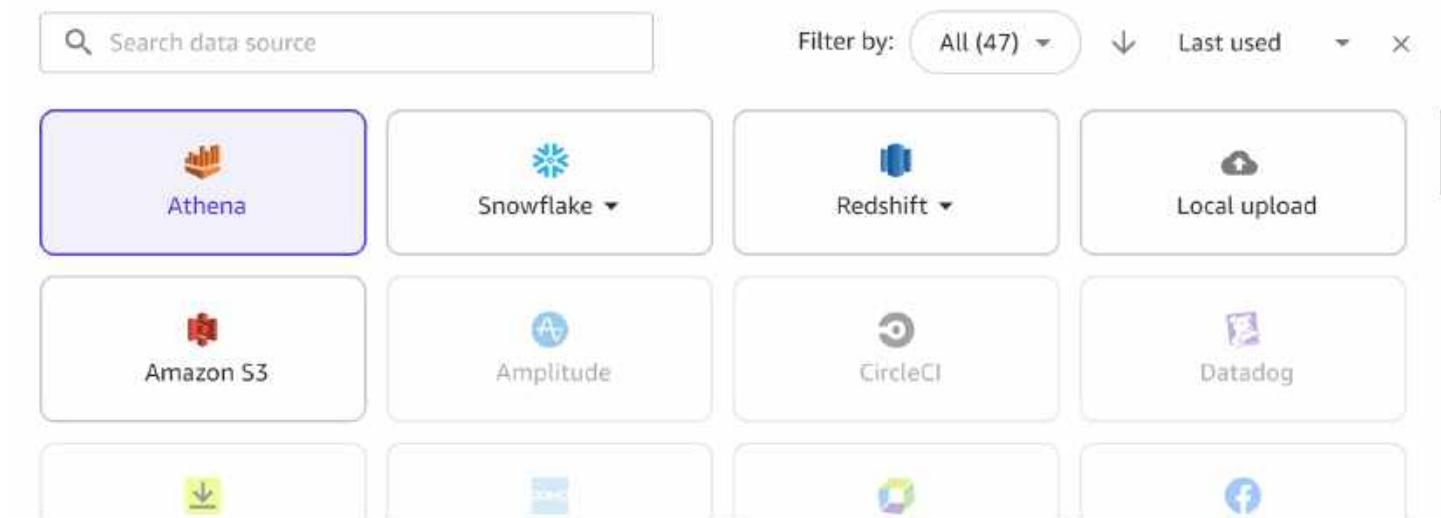
To learn how to delete your configuration, see [Delete an automatic configuration](#).

## Connect to data sources

In Amazon SageMaker Canvas, you can import data from a location outside of your local file system through an AWS service, a SaaS platform, or other databases using JDBC connectors. For example, you might want to import tables from a data warehouse in Amazon Redshift, or you might want to import Google Analytics data.

When you go through the **Import** workflow to import data in the Canvas application, you can choose your data source and then select the data that you want to import. For certain data sources, like Snowflake and Amazon Redshift, you must specify your credentials and add a connection to the data source.

The following screenshot shows the data sources toolbar in the **Import** workflow, with all of the available data sources highlighted. You can only import data from the data sources that are available to you. Contact your administrator if your desired data source isn't available.



[How to connect to data sources](#)

The following sections provide information about establishing connections to external data sources and importing data from them. Review the following section first to determine what permissions you need to import data from your data source.

## Permissions

Review the following information to ensure that you have the necessary permissions to import data from your data source:

- **Amazon S3:** You can import data from any Amazon S3 bucket as long as your user has permissions to access the bucket. For more information about using AWS IAM to control access to Amazon S3 buckets, see [Identity and access management in Amazon S3](#) in the *Amazon S3 User Guide*.
- **Amazon Athena:** If you have the [AmazonSageMakerFullAccess](#) policy and the [AmazonSageMakerCanvasFullAccess](#) policy attached to your user's execution role, then you can query your AWS Glue Data Catalog with Amazon Athena. If you're part of an Athena workgroup, make sure that the Canvas user has permissions to run Athena queries on the data. For more information, see [Using workgroups for running queries](#) in the *Amazon Athena User Guide*.
- **Amazon DocumentDB:** You can import data from any Amazon DocumentDB database as long as you have the credentials (username and password) to connect to the database and have the minimum base Canvas permissions attached to your user's execution role. For more information about Canvas permissions, see the [Prerequisites for setting up Amazon SageMaker Canvas](#).

- **Amazon Redshift:** To give yourself the necessary permissions to import data from Amazon Redshift, see [Grant Users Permissions to Import Amazon Redshift Data](#).
- **Amazon RDS:** If you have the [AmazonSageMakerCanvasFullAccess](#) policy attached to your user's execution role, then you'll be able to access your Amazon RDS databases from Canvas.
- **SaaS platforms:** If you have the [AmazonSageMakerFullAccess](#) policy and the [AmazonSageMakerCanvasFullAccess](#) policy attached to your user's execution role, then you have the necessary permissions to import data from SaaS platforms. See [Use SaaS connectors with Canvas](#) for more information about connecting to a specific SaaS connector.
- **JDBC connectors:** For database sources such as Databricks, MySQL or MariaDB, you must enable username and password authentication on the source database before attempting to connect from Canvas. If you're connecting to a Databricks database, you must have the JDBC URL that contains the necessary credentials.

## Connect to a database stored in AWS

You might want to import data that you've stored in AWS. You can import data from Amazon S3, use Amazon Athena to query a database in the AWS Glue Data Catalog, import data from [Amazon RDS](#), or make a connection to a provisioned Amazon Redshift database (not Redshift Serverless).

You can create multiple connections to Amazon Redshift. For Amazon Athena, you can access any databases that you have in your [AWS Glue Data Catalog](#). For Amazon S3, you can import data from a bucket as long as you have the necessary permissions.

Review the following sections for more detailed information.

### Connect to data in Amazon S3, Amazon Athena, or Amazon RDS

For Amazon S3, you can import data from an Amazon S3 bucket as long as you have permissions to access the bucket.

For Amazon Athena, you can access databases in your AWS Glue Data Catalog as long as you have permissions through your [Amazon Athena workgroup](#).

For Amazon RDS, if you have the [AmazonSageMakerCanvasFullAccess](#) policy attached to your user's role, then you'll be able to import data from your Amazon RDS databases into Canvas.

To import data from an Amazon S3 bucket, or to run queries and import data tables with Amazon Athena, see [Create a dataset](#). You can only import tabular data from Amazon Athena, and you can import tabular and image data from Amazon S3.

## Connect to an Amazon DocumentDB database

Amazon DocumentDB is a fully managed, serverless, document database service. You can import unstructured document data stored in an Amazon DocumentDB database into SageMaker Canvas as a tabular dataset, and then you can build machine learning models with the data.

### Important

Your SageMaker AI domain must be configured in **VPC only** mode to add connections to Amazon DocumentDB. You can only access Amazon DocumentDB clusters in the same Amazon VPC as your Canvas application. Additionally, Canvas can only connect to TLS-enabled Amazon DocumentDB clusters. For more information about how to set up Canvas in **VPC only** mode, see [Configure Amazon SageMaker Canvas in a VPC without internet access](#).

To import data from Amazon DocumentDB databases, you must have credentials to access the Amazon DocumentDB database and specify the username and password when creating a database connection. You can configure more granular permissions and restrict access by modifying the Amazon DocumentDB user permissions. To learn more about access control in Amazon DocumentDB, see [Database Access Using Role-Based Access Control](#) in the *Amazon DocumentDB Developer Guide*.

When you import from Amazon DocumentDB, Canvas converts your unstructured data into a tabular dataset by mapping the fields to columns in a table. Additional tables are created for each complex field (or nested structure) in the data, where the columns correspond to the sub-fields of the complex field. For more detailed information about this process and examples of schema conversion, see the [Amazon DocumentDB JDBC Driver Schema Discovery](#) GitHub page.

Canvas can only make a connection to a single database in Amazon DocumentDB. To import data from a different database, you must create a new connection.

You can import data from Amazon DocumentDB into Canvas by using the following methods:

- [Create a dataset](#). You can import your Amazon DocumentDB data and create a tabular dataset in Canvas. If you choose this method, make sure that you follow the [Import tabular data](#) procedure.
- [Create a data flow](#). You can create a data preparation pipeline in Canvas and add your Amazon DocumentDB database as a data source.

To proceed with importing your data, follow the procedure for one of the methods linked in the preceding list.

When you reach the step in either workflow to choose a data source (Step 6 for creating a dataset, or Step 8 for creating a data flow), do the following:

1. For **Data Source**, open the dropdown menu and choose **DocumentDB**.
2. Choose **Add connection**.
3. In the dialog box, specify your Amazon DocumentDB credentials:
  - a. Enter a **Connection name**. This is a name used by Canvas to identify this connection.
  - b. For **Cluster**, select the cluster in Amazon DocumentDB that stores your data. Canvas automatically populates the dropdown menu with Amazon DocumentDB clusters in the same VPC as your Canvas application.
  - c. Enter the **Username** for your Amazon DocumentDB cluster.
  - d. Enter the **Password** for your Amazon DocumentDB cluster.
  - e. Enter the name of the **Database** to which you want to connect.
  - f. The **Read preference** option determines which types of instances on your cluster Canvas reads the data from. Select one of the following:
    - **Secondary preferred** – Canvas defaults to reading from the cluster's secondary instances, but if a secondary instance isn't available, then Canvas reads from a primary instance.
    - **Secondary** – Canvas only reads from the cluster's secondary instances, which prevents the read operations from interfering with the cluster's regular read and write operations.
  - g. Choose **Add connection**. The following image shows the dialog box with the preceding fields for an Amazon DocumentDB connection.

Add a new DocumentDB connection X

---

Connection name (1)  
Create a name to identify your connection

Cluster ▼  
None

First part of the cluster endpoint used to construct the URL for connecting your database.

Username

Password (1)

Database

Read preference (1)  
 Secondary preferred  
 Secondary

---

Cancel Add connection

You should now have an Amazon DocumentDB connection, and you can use your Amazon DocumentDB data in Canvas to create either a dataset or a data flow.

## Connect to an Amazon Redshift database

You can import data from Amazon Redshift, a data warehouse where your organization keeps its data. Before you can import data from Amazon Redshift, the AWS IAM role you use must have the `AmazonRedshiftFullAccess` managed policy attached. For instructions on how to attach this policy, see [Grant Users Permissions to Import Amazon Redshift Data](#).

To import data from Amazon Redshift, you do the following:

1. Create a connection to an Amazon Redshift database.
2. Choose the data that you're importing.
3. Import the data.

You can use the Amazon Redshift editor to drag datasets onto the import pane and import them into SageMaker Canvas. For more control over the values returned in the dataset, you can use the following:

- SQL queries
- Joins

With SQL queries, you can customize how you import the values in the dataset. For example, you can specify the columns returned in the dataset or the range of values for a column.

You can use joins to combine multiple datasets from Amazon Redshift into a single dataset. You can drag your datasets from Amazon Redshift into the panel that gives you the ability to join the datasets.

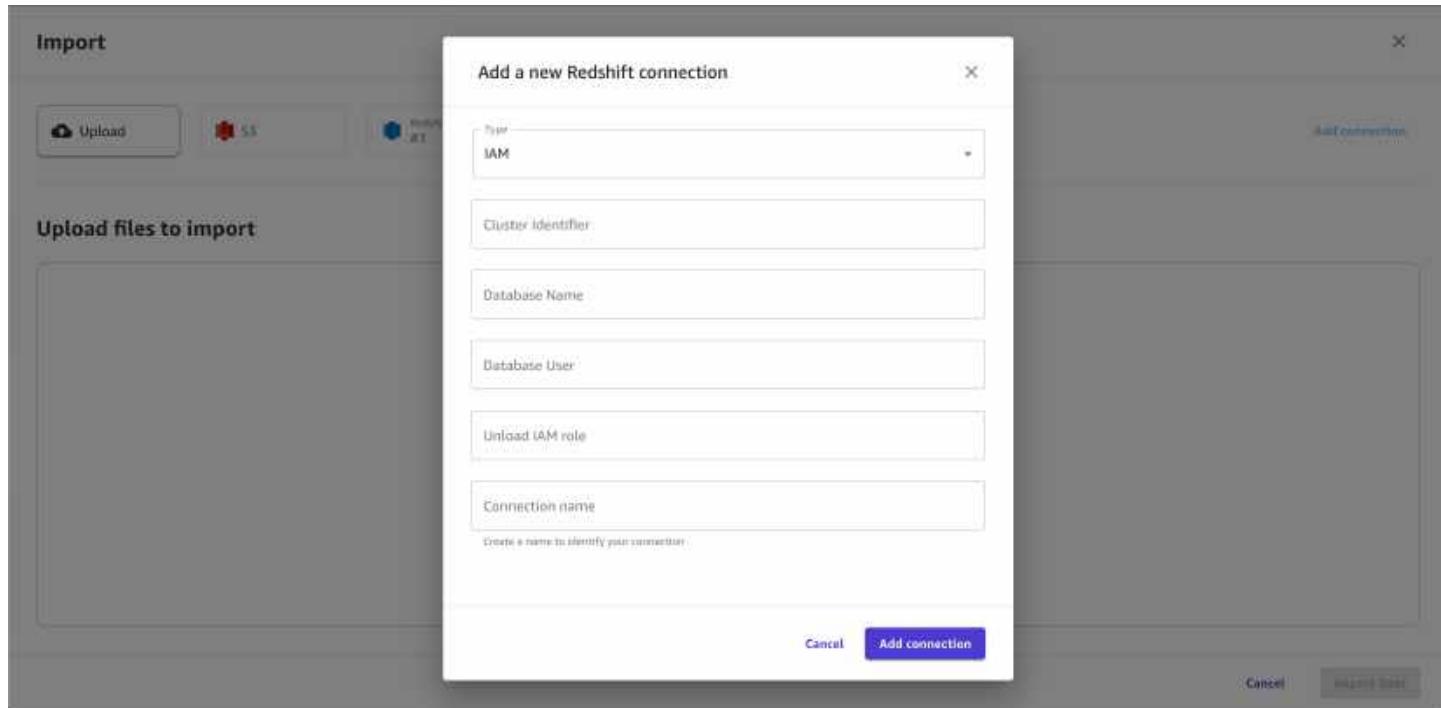
You can use the SQL editor to edit the dataset that you've joined and convert the joined dataset into a single node. You can join another dataset to the node. You can import the data that you've selected into SageMaker Canvas.

Use the following procedure to import data from Amazon Redshift.

1. In the SageMaker Canvas application, go to the **Datasets** page.
2. Choose **Import data**, and from the dropdown menu, choose **Tabular**.
3. Enter a name for the dataset and choose **Create**.
4. For **Data Source**, open the dropdown menu and choose **Redshift**.
5. Choose **Add connection**.
6. In the dialog box, specify your Amazon Redshift credentials:
  - a. For **Authentication method**, choose **IAM**.
  - b. Enter the **Cluster identifier** to specify to which cluster you want to connect. Enter only the cluster identifier and not the full endpoint of the Amazon Redshift cluster.
  - c. Enter the **Database name** of the database to which you want to connect.
  - d. Enter a **Database user** to identify the user you want to use to connect to the database.
  - e. For **ARN**, enter the IAM role ARN of the role that the Amazon Redshift cluster should assume to move and write data to Amazon S3. For more information about this role, see [Authorizing Amazon Redshift to access other AWS services on your behalf](#) in the *Amazon Redshift Management Guide*.

- f. Enter a **Connection name**. This is a name used by Canvas to identify this connection.
7. From the tab that has the name of your connection, drag the .csv file that you're importing to the **Drag and drop table to import** pane.
8. Optional: Drag additional tables to the import pane. You can use the GUI to join the tables. For more specificity in your joins, choose **Edit in SQL**.
9. Optional: If you're using SQL to query the data, you can choose **Context** to add context to the connection by specifying values for the following:
  - **Warehouse**
  - **Database**
  - **Schema**
10. Choose **Import data**.

The following image shows an example of fields specified for an Amazon Redshift connection.



The following image shows the page used to join datasets in Amazon Redshift.

**Import**

Upload S3 Test Add connection

**Test** Accessed 11/18/21 at 03:02:17 AM Edit in SQL

Search... 

date event listing sales return

**Import preview**

catid	eventname	listid	starttime	numtickets	priceperticket	sellerid	starttime
8	Return To Forever	121610	2008-01-01 12:09:48	2	99.00	3709	2008-01-01
6	The King and I	146839	2008-01-01 12:37:28	24	93.00	42967	2008-01-01
9	Hannah Montana	153855	2008-01-01 11:17:16	14	63.00	49537	2008-01-01
8	La Damnation de Faust	206280	2008-01-01 06:38:45	2	823.00	14754	2008-01-01

Cancel Import Data

The following image shows an SQL query being used to edit a join in Amazon Redshift.

**Import**

Upload S3 Test Add connection

**Test** Accessed 11/18/21 at 03:02:45 AM Edit SQL Convert to node

Search... 

date event listing sales return

**Edit SQL** Accessed 11/18/21 at 03:02:45 AM

```
WITH Cte7 AS (SELECT listid, sellerid, eventid, dateid, numtickets, priceperticket, totalprice, starttime FROM dev.public.listing),
      Unz AS (SELECT eventid, venueid, catid, dateid, eventname, starttime FROM dev.public.event)
SELECT
    catid,
    eventname,
    listid,
    starttime,
    numtickets,
    priceperticket,
    sellerid,
    starttime,
    totalprice,
    venueid,
```

Run SQL

**Import preview**

catid	eventname	listid	starttime	numtickets	priceperticket	sellerid	starttime
8	Return To Forever	121610	2008-01-01 12:09:48	2	99.00	3709	2008-01-01
6	The King and I	146839	2008-01-01 12:37:28	24	93.00	42967	2008-01-01
9	Hannah Montana	153855	2008-01-01 11:17:16	14	63.00	49537	2008-01-01
8	La Damnation de Faust	206280	2008-01-01 06:38:45	2	823.00	14754	2008-01-01

Cancel Import Data

## Connect to your data with JDBC connectors

With JDBC, you can connect to your databases from sources such as Databricks, SQLServer, MySQL, PostgreSQL, MariaDB, Amazon RDS, and Amazon Aurora.

You must make sure that you have the necessary credentials and permissions to create the connection from Canvas.

- For Databricks, you must provide a JDBC URL. The URL formatting can vary between Databricks instances. For information about finding the URL and the specifying the parameters within it, see [JDBC configuration and connection parameters](#) in the Databricks documentation. The following is an example of how a URL can be formatted:  
`jdbc:spark://aws-sagemaker-datawrangler.cloud.databricks.com:443/default;transportMode=http;ssl=1;httpPath$sql/protocolv1/o/3122619508517275/0909-200301-cut318;AuthMech=3;UID=token;PWD=personal-access-token`
- For other database sources, you must set up username and password authentication, and then specify those credentials when connecting to the database from Canvas.

Additionally, your data source must either be accessible through the public internet, or if your Canvas application is running in **VPC only** mode, then the data source must run in the same VPC. For more information about configuring an Amazon RDS database in a VPC, see [Amazon VPC VPCs and Amazon RDS](#) in the *Amazon RDS User Guide*.

After you've configured your data source credentials, you can sign in to the Canvas application and create a connection to the data source. Specify your credentials (or, for Databricks, the URL) when creating the connection.

## Connect to data sources with OAuth

Canvas supports using OAuth as an authentication method for connecting to your data in Snowflake and Salesforce Data Cloud. [OAuth](#) is a common authentication platform for granting access to resources without sharing passwords.

### Note

You can only establish one OAuth connection for each data source.

To authorize the connection, you must follow the initial setup described in [Set up connections to data sources with OAuth](#).

After setting up the OAuth credentials, you can do the following to add a Snowflake or Salesforce Data Cloud connection with OAuth:

1. Sign in to the Canvas application.
2. Create a tabular dataset. When prompted to upload data, choose Snowflake or Salesforce Data Cloud as your data source.
3. Create a new connection to your Snowflake or Salesforce Data Cloud data source. Specify OAuth as the authentication method and enter your connection details.

You should now be able to import data from your databases in Snowflake or Salesforce Data Cloud.

## Connect to a SaaS platform

You can import data from Snowflake and over 40 other external SaaS platforms. For a full list of the connectors, see the table on [Data import](#).

 **Note**

You can only import tabular data, such as data tables, from SaaS platforms.

## Use Snowflake with Canvas

Snowflake is a data storage and analytics service, and you can import your data from Snowflake into SageMaker Canvas. For more information about Snowflake, see the [Snowflake documentation](#).

You can import data from your Snowflake account by doing the following:

1. Create a connection to the Snowflake database.
2. Choose the data that you're importing by dragging and dropping the table from the left navigation menu into the editor.
3. Import the data.

You can use the Snowflake editor to drag datasets onto the import pane and import them into SageMaker Canvas. For more control over the values returned in the dataset, you can use the following:

- SQL queries
- Joins

With SQL queries, you can customize how you import the values in the dataset. For example, you can specify the columns returned in the dataset or the range of values for a column.

You can join multiple Snowflake datasets into a single dataset before you import into Canvas using SQL or the Canvas interface. You can drag your datasets from Snowflake into the panel that gives you the ability to join the datasets, or you can edit the joins in SQL and convert the SQL into a single node. You can join other nodes to the node that you've converted. You can then combine the datasets that you've joined into a single node and join the nodes to a different Snowflake dataset. Finally, you can import the data that you've selected into Canvas.

Use the following procedure to import data from Snowflake to Amazon SageMaker Canvas.

1. In the SageMaker Canvas application, go to the **Datasets** page.
2. Choose **Import data**, and from the dropdown menu, choose **Tabular**.
3. Enter a name for the dataset and choose **Create**.
4. For **Data Source**, open the dropdown menu and choose **Snowflake**.
5. Choose **Add connection**.
6. In the **Add a new Snowflake connection** dialog box, specify your Snowflake credentials. For the **Authentication method**, choose one of the following:
  - **Basic - username password** – Provide your Snowflake account ID, username, and password.
  - **ARN** – For improved protection of your Snowflake credentials, provide the ARN of an AWS Secrets Manager secret that contains your credentials. For more information, see [Create an AWS Secrets Manager secret](#) in the [AWS Secrets Manager User Guide](#).

Your secret should have your Snowflake credentials stored in the following JSON format:

```
{"accountid": "ID",  
 "username": "username",  
 "password": "password"}
```

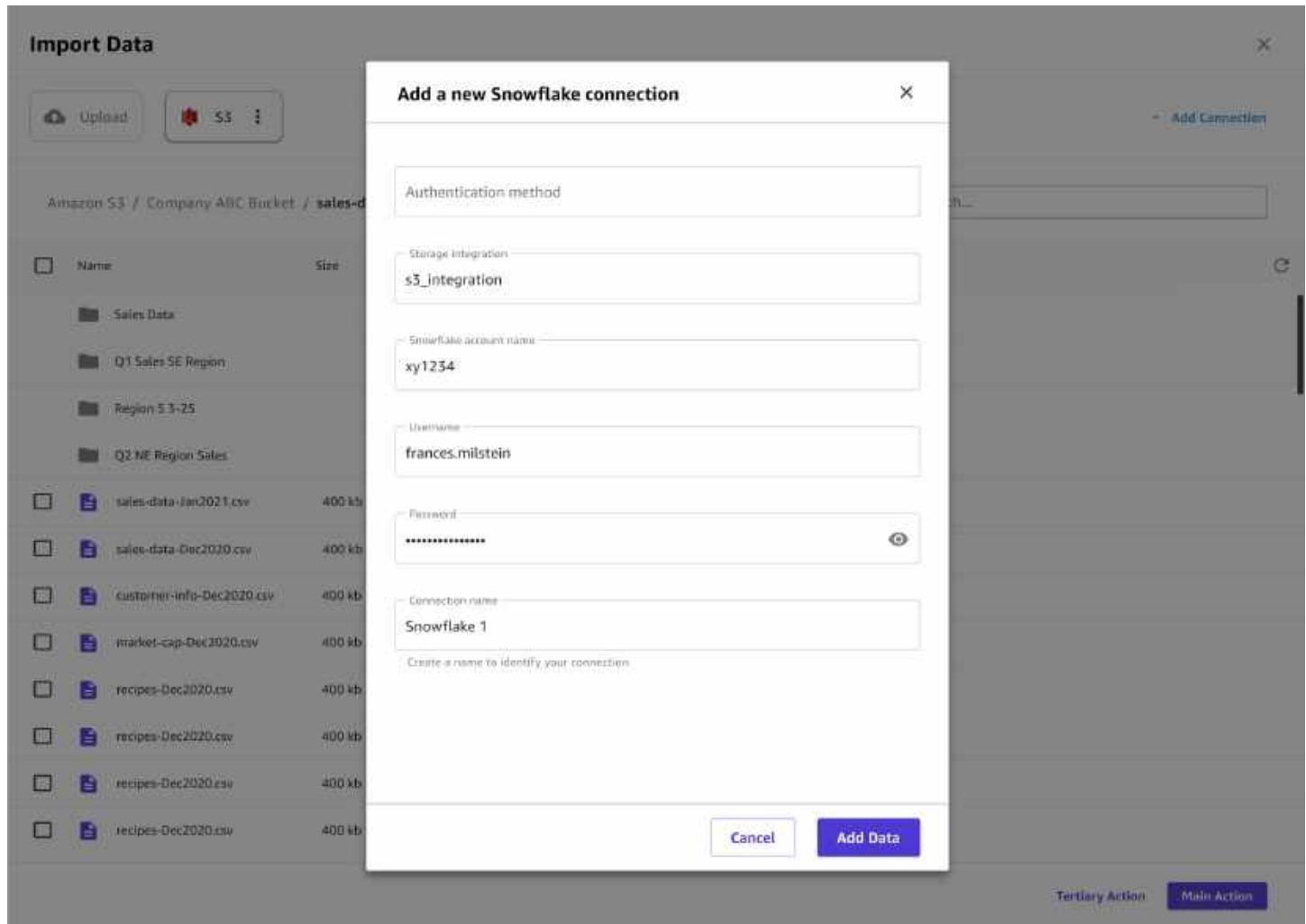
- **OAuth** – OAuth lets you authenticate without providing a password but requires additional setup. For more information about setting up OAuth credentials for Snowflake, see [Set up connections to data sources with OAuth](#).

7. Choose **Add connection**.
8. From the tab that has the name of your connection, drag the .csv file that you're importing to the **Drag and drop table to import** pane.
9. Optional: Drag additional tables to the import pane. You can use the user interface to join the tables. For more specificity in your joins, choose **Edit in SQL**.
10. Optional: If you're using SQL to query the data, you can choose **Context** to add context to the connection by specifying values for the following:
  - **Warehouse**
  - **Database**
  - **Schema**

Adding context to a connection makes it easier to specify future queries.

11. Choose **Import data**.

The following image shows an example of fields specified for a Snowflake connection.



The following image shows the page used to add context to a connection.

**Import Data**

Upload S3 Snowflake Crystal 1 Redshift Canvas Sales Add Connection

**Diamond 2** Context Edit SQL (Runned 8/9/21 at 11:34 AM) Cancel Convert to node

Warehouse: `!0.CustomerName, canvas_sales.OrderID`  
 Database: `IN Customers.CustomerID = canvas_sales.CustomerID`  
 Schema: `IN Customers.CustomerID = canvas_sales.CustomerID`

Search: `{table_name}`

Run SQL

**Import preview** New preview available Show dropped columns

<input checked="" type="checkbox"/> Sold	ABC	<input type="checkbox"/> Price	ABC	<input checked="" type="checkbox"/> Region	ABC	<input checked="" type="checkbox"/> Discount	ABC	<input checked="" type="checkbox"/> Fabric	ABC	<input checked="" type="checkbox"/> Age	ABC
Yes		29.99		Southwest		23		Yes		Yes	
Yes		29.99		Southwest		23		Yes		Yes	
Yes		29.99		Southwest		23		Yes		Yes	
Yes		29.99		Southwest		23		Yes		Yes	
Yes		29.99		Southwest		23		Yes		Yes	

Cancel Import data

The following image shows the page used to join datasets in Snowflake.

### Import Data

Upload S3 Snowflake Crystal 1 Redshift Canvas Sales Add Connection

Diamond 2 Context Last saved 8/9/21 at 11:34 AM Edit in SQL

Search

{database\_name}

{database\_name}

{database\_name}

{database\_name}

+ {schema\_name}

- {schema\_name}

{table\_name}

Import preview

Sold	ABC	Price	ABC	Region	ABC	Discount	ABC	Fabric	ABC	Age	ABC
Yes		29.99		Southwest		23		Yes		Yes	
Yes		29.99		Southwest		23		Yes		Yes	
Yes		29.99		Southwest		23		Yes		Yes	
Yes		29.99		Southwest		23		Yes		Yes	
Yes		29.99		Southwest		23		Yes		Yes	

Show dropped columns Cancel Import data

The following image shows a SQL query being used to edit a join in Snowflake.

The screenshot shows the "Import Data" interface in the Amazon SageMaker AI developer guide. At the top, there are connection buttons for "Upload", "S3", "Snowflake Crystal 1", and "Redshift Canvas Sales". A "Add Connection" button is also present. Below the connections, a "Diamond 2" section shows a search bar and a tree view of database, schema, and table names. An "Edit SQL" panel displays a complex JOIN query between multiple tables. A "Run SQL" button is located at the bottom right of the SQL panel. Below the SQL panel is an "Import preview" section containing a table with five rows of data. The table has columns for Sold, Price, Region, Discount, Fabric, Age, and ABC. The "Import data" button is at the bottom right of the preview table.

Sold	ABC	Price	Region	Discount	Fabric	Age	ABC
Yes	29.99		Southwest	23	Yes	Yes	
Yes	29.99		Southwest	23	Yes	Yes	
Yes	29.99		Southwest	23	Yes	Yes	
Yes	29.99		Southwest	23	Yes	Yes	
Yes	29.99		Southwest	23	Yes	Yes	

## Use SaaS connectors with Canvas

### Note

For SaaS platforms besides Snowflake, you can only have one connection per data source.

Before you can import data from a SaaS platform, your administrator must authenticate and create a connection to the data source. For more information about how administrators can create a connection with a SaaS platform, see [Managing Amazon AppFlow connections](#) in the *Amazon AppFlow User Guide*.

If you're an administrator getting started with Amazon AppFlow for the first time, see [Getting started](#) in the *Amazon AppFlow User Guide*.

To import data from a SaaS platform, you can follow the standard [Import tabular data procedure](#), which shows you how to import tabular datasets into Canvas.

## Sample datasets in Canvas

SageMaker Canvas provides sample datasets addressing unique use cases so you can start building, training, and validating models quickly without writing any code. The use cases associated with these datasets highlight the capabilities of SageMaker Canvas, and you can leverage these datasets to get started with building models. You can find the sample datasets in the **Datasets** page of your SageMaker Canvas application.

The following datasets are the samples that SageMaker Canvas provides by default. These datasets cover use cases such as predicting house prices, loan defaults, and readmission for diabetic patients; forecasting sales; predicting machine failures to streamline predictive maintenance in manufacturing units; and generating supply chain predictions for transportation and logistics. The datasets are stored in the `sample_dataset` folder in the default Amazon S3 bucket that SageMaker AI creates for your account in a Region.

- **canvas-sample-diabetic-readmission.csv:** This dataset contains historical data including over fifteen features with patient and hospital outcomes. You can use this dataset to predict whether high-risk diabetic patients are likely to get readmitted to the hospital within 30 days of discharge, after 30 days, or not at all. Use the **readmitted** column as the target column, and use the 3+ category prediction model type with this dataset. To learn more about how to build a model with this dataset, see the [SageMaker Canvas workshop page](#). This dataset was obtained from the [UCI Machine Learning Repository](#).
- **canvas-sample-housing.csv:** This dataset contains data on the characteristics tied to a given housing price. You can use this dataset to predict housing prices. Use the **median\_house\_value** column as the target column, and use the numeric prediction model type with this dataset. To learn more about building a model with this dataset, see the [SageMaker Canvas workshop page](#). This is the California housing dataset obtained from the [StatLib repository](#).
- **canvas-sample-loans.csv:** This dataset contains complete loan data for all loans issued from 2007–2011, including the current loan status and latest payment information. You can use this dataset to predict whether a customer will repay a loan. Use the **loan\_status** column as the target column, and use the 3+ category prediction model type with this dataset. To learn more about how to build a model with this dataset, see the [SageMaker Canvas workshop page](#). This data uses the LendingClub data obtained from [Kaggle](#).

- **canvas-sample-maintenance.csv:** This dataset contains data on the characteristics tied to a given maintenance failure type. You can use this dataset to predict which failure will occur in the future. Use the **Failure Type** column as the target column, and use the 3+ category prediction model type with this dataset. To learn more about how to build a model with this dataset, see the [SageMaker Canvas workshop page](#). This dataset was obtained from the [UCI Machine Learning Repository](#).
- **canvas-sample-shipping-logs.csv:** This dataset contains complete shipping data for all products delivered, including estimated time shipping priority, carrier, and origin. You can use this dataset to predict the estimated time of arrival of the shipment in number of days. Use the **ActualShippingDays** column as the target column, and use the numeric prediction model type with this dataset. To learn more about how to build a model with this data, see the [SageMaker Canvas workshop page](#). This is a synthetic dataset created by Amazon.
- **canvas-sample-sales-forecasting.csv:** This dataset contains historical time series sales data for retail stores. You can use this dataset to forecast sales for a particular retail store. Use the **sales** column as the target column, and use the time series forecasting model type with this dataset. To learn more about how to build a model with this dataset, see the [SageMaker Canvas workshop page](#). This is a synthetic dataset created by Amazon.

## Re-import a deleted sample dataset

Amazon SageMaker Canvas provides you with sample datasets for various use cases that highlight the capabilities of Canvas. To learn more about the sample datasets that are available, see [Sample datasets in Canvas](#). If you no longer wish to use the sample datasets, you can delete them from the **Datasets** page of your SageMaker Canvas application. However, these datasets are still stored in the Amazon S3 bucket that you specified as the [Canvas storage location](#), so you can always access them later.

If you used the default Amazon S3 bucket, the bucket name follows the pattern `sagemaker-{region}-{account ID}`. You can find the sample datasets in the directory path `Canvas/sample_dataset`.

If you delete a sample dataset from your SageMaker Canvas application and want to access the sample dataset again, use the following procedure.

1. Navigate to the **Datasets** page in your SageMaker Canvas application.
2. Choose **Import data**.

3. From the list of Amazon S3 buckets, select the bucket that is your Canvas storage location. If using the default SageMaker AI-created Amazon S3 bucket, it follows the naming pattern `sagemaker-{region}-{account ID}`.
4. Select the **Canvas** folder.
5. Select the **sample\_dataset** folder, which contains all of the sample datasets for SageMaker Canvas.
6. Select the dataset you want to import, and then choose **Import data**.

## Data preparation

### Note

Previously, Amazon SageMaker Data Wrangler was part of the SageMaker Studio Classic experience. Now, if you update to using the new Studio experience, you must use SageMaker Canvas to access Data Wrangler and receive the latest feature updates. If you have been using Data Wrangler in Studio Classic until now and want to migrate to Data Wrangler in Canvas, you might have to grant additional permissions so that you can create and use a Canvas application. For more information, see [\(Optional\) Migrate from Data Wrangler in Studio Classic to SageMaker Canvas](#).

To learn how to migrate your data flows from Data Wrangler in Studio Classic, see [\(Optional\) Migrate data from Studio Classic to Studio](#).

Use Amazon SageMaker Data Wrangler in Amazon SageMaker Canvas to prepare, featurize and analyze your data. You can integrate a Data Wrangler data preparation flow into your machine learning (ML) workflows to simplify and streamline data pre-processing and feature engineering using little to no coding. You can also add your own Python scripts and transformations to customize workflows.

- **Data Flow** – Create a data flow to define a series of ML data prep steps. You can use a flow to combine datasets from different data sources, identify the number and types of transformations you want to apply to datasets, and define a data prep workflow that can be integrated into an ML pipeline.
- **Transform** – Clean and transform your dataset using standard *transforms* like string, vector, and numeric data formatting tools. Featurize your data using transforms like text and date/time embedding and categorical encoding.

- **Generate Data Insights** – Automatically verify data quality and detect abnormalities in your data with Data Wrangler Data Quality and Insights Report.
- **Analyze** – Analyze features in your dataset at any point in your flow. Data Wrangler includes built-in data visualization tools like scatter plots and histograms, as well as data analysis tools like target leakage analysis and quick modeling to understand feature correlation.
- **Export** – Export your data preparation workflow to a different location. The following are example locations:
  - Amazon Simple Storage Service (Amazon S3) bucket
  - Amazon SageMaker Feature Store – Store the features and their data in a centralized store.
- **Automate data preparation** – Create machine learning workflows from your data flow.
  - Amazon SageMaker Pipelines – Build workflows that manage your SageMaker AI data preparation, model training, and model deployment jobs.
  - Serial inference pipeline – Create a serial inference pipeline from your data flow. Use it to make predictions on new data.
  - Python script – Store the data and their transformations in a Python script for your custom workflows.

## Create a data flow

Use a Data Wrangler flow in SageMaker Canvas, or *data flow*, to create and modify a data preparation pipeline. We recommend that you use Data Wrangler for datasets larger than 5 GB.

To get started, use the following procedure to import your data into a data flow.

1. Open SageMaker Canvas.
2. In the left-hand navigation, choose **Data Wrangler**.
3. Choose **Import and prepare**.
4. From the dropdown menu, choose either **Tabular** or **Image**.
5. For **Select a data source**, choose your data source and select the data that you want to import. You have the option to select up to 30 files or one folder. If you have a dataset already imported into Canvas, choose **Canvas dataset** as your source. Otherwise, connect to a data source such as Amazon S3 or Snowflake and browse through your data. For information about connecting to a data source or importing data, see the following pages:
  - [Data import](#)

- [Connect to data sources](#)
6. After selecting the data that you want to import, choose **Next**.
  7. (Optional) For the **Import settings** section when importing a tabular dataset, expand the **Advanced** dropdown menu. You can specify the following advanced settings for data flow imports:
    - **Sampling method** – Select the sampling method and sample size you'd like to use. For more information about how to change your sample, see the section [Edit the data flow sampling configuration](#).
    - **File encoding (CSV)** – Select your dataset file's encoding. UTF-8 is the default.
    - **Skip first rows** – Enter the number of rows you'd like to skip importing if you have redundant rows at the beginning of your dataset.
    - **Delimiter** – Select the delimiter that separates each item in your data. You can also specify a custom delimiter.
    - **Multi-line detection** – Select this option if you'd like Canvas to manually parse your entire dataset for multi-line cells. Canvas determines whether or not to use multi-line support by taking a sample of your data, but Canvas might not detect any multi-line cells in the sample. In this case, we recommend that you select the **Multi-line detection** option to force Canvas to check your entire dataset for multi-line cells.
8. Choose **Import**.

You should now have a new data flow, and you can begin adding transform steps and analyses.

## How the data flow UI works

To help you navigate your data flow, Data Wrangler has the following tabs in the top navigation pane:

- **Data flow** – This tab provides you with a visual view of your data flow step where you can add or remove transforms, and export data.
- **Data** – This tab gives you a preview of your data so that you can check the results of your transforms. You can also see an ordered list of your data flow steps and edit or reorder the steps.

**Note**

In this tab, you can only preview data visualizations (such as the distribution of values per column) for Amazon S3 data sources. Visualizations for other data sources, such as Amazon Athena, aren't supported.

- **Analyses** – In this tab, you can see separate sub-tabs for each analysis you create. For example, if you create a histogram and a Data Quality and Insights (DQI) report, Canvas creates a tab for each.

When you import a dataset, the original dataset appears on the data flow and is named **Source**. SageMaker Canvas automatically infers the types of each column in your dataset and creates a new dataframe named **Data types**. You can select this frame to update the inferred data types.

The datasets, transformations, and analyses that you use in the data flow are represented as *steps*. Each time you add a transform step, you create a new dataframe. When multiple transform steps (other than **Join** or **Concatenate**) are added to the same dataset, they are stacked.

Under the **Combine data** option, **Join** and **Concatenate** create standalone steps that contain the new joined or concatenated dataset.

## Edit the data flow sampling configuration

When importing tabular data into a Data Wrangler data flow, you can opt to take a sample of your dataset to speed up the data exploration and cleaning process. Running exploratory transforms on a sample of your dataset is often faster than running transforms on your entire dataset, and when you're ready to export your dataset and build a model, you can apply the transforms to the full dataset.

Canvas supports the following sampling methods:

- **FirstK** – Canvas selects the first  $K$  items from your dataset, where  $K$  is a number you specify. This sampling method is simple but can introduce bias if your dataset isn't randomly ordered.
- **Random** – Canvas selects items from the dataset at random, with each item having an equal probability of being chosen. This sampling method helps ensure that the sample is representative of the entire dataset.
- **Stratified** – Canvas divides the dataset into groups (or *strata*) based on one or more attributes (for example, age and income level). Then, a proportional number of items are randomly selected

from each group. This method ensures that all relevant subgroups are adequately represented in the sample.

You can edit your sampling configuration at any time to change the size of the sample used for data exploration.

To make changes to your sampling configuration, do the following:

1. In your data flow graph, select your data source node.
2. Choose **Sampling** on the bottom navigation bar.
3. The **Sampling** dialog box opens. For the **Sampling method** dropdown, select your desired sampling method.
4. For **Maximum sample size**, enter the number of rows you want to sample.
5. Choose **Update** to save your changes.

The changes to your sampling configuration should now be applied.

## Add a step to your data flow

In your Data Wrangler data flows, you can add steps that represent data transformations and analyses.

To add a step to your data flow, select **+** next to any dataset node or previously added step. Then, select one of the following options:

- **Edit data types** (For a **Data types** step only): If you have not added any transforms to a **Data types** step, you can double-click on the **Data types** step in your flow to open the **Data** tab and edit the data types that Data Wrangler inferred when importing your dataset.
- **Add transform**: Adds a new transform step. See [Transform data](#) to learn more about the data transformations you can add.
- **Get data insights**: Add analyses, such as histograms or custom visualizations. You can use this option to analyze your data at any point in the data flow. See [Perform exploratory data analysis \(EDA\)](#) to learn more about the analyses you can add.
- **Join**: Find this option under **Combine data** to join two datasets and add the resulting dataset to the data flow. To learn more, see [Join Datasets](#).
- **Concatenate**: Find this option under **Combine data** to concatenate two datasets and add the resulting dataset to the data flow. To learn more, see [Concatenate Datasets](#).

## Edit data flow steps

In Amazon SageMaker Canvas, you can edit individual steps in your data flows to transform your dataset without having to create a new data flow. The following page covers how to edit join and concatenate steps, as well as data source steps.

### Edit join and concatenate steps

Within your data flows, you have the flexibility to edit your join and concatenate steps. You can make necessary adjustments to your data processing workflow, ensuring that your data is properly combined and transformed without having to redo your entire data flow.

To edit a join or concatenate step in your data flow, do the following:

1. Open your data flow.
2. Choose the plus icon (+) next to the join or concatenate node that you want to edit.
3. From the context menu, choose **Edit**.
4. A side panel opens where you can edit the details of your join or concatenation. Modify your step fields, such as the type of join. To swap out a data node and select a different one to join or concatenate, choose the delete icon next to the node and then, in the data flow view, select the new node that you want to include in your transformation.

 **Note**

When swapping out a node during the editing process, you can only select steps that occur before the join or concatenate operation. You can swap either the left or right node, but you can only swap one node at a time. Additionally, you cannot select a source node as a replacement.

5. Choose **Preview** to view the result of the combining operation.
6. Choose **Update** to save your changes.

Your data flow should now be updated.

### Edit or replace a data source step

You might need to make changes to your data source or dataset without deleting the transforms and data flow steps applied to your original data. Within Data Wrangler, you can edit or replace

your data source configuration while keeping the steps of your data flow. When editing a data source, you can change the import settings, such as the sampling size or method and any advanced settings. You can also add more files with the same schema, or for query-based data sources such as Amazon Athena, you can edit the query. When replacing a data source, you have the option to select a different dataset, or even import the data from a different data source altogether, as long as the schema of the new data matches the original data.

To edit a data source configuration, do the following:

1. In the Canvas application, go to the **Data Wrangler** page.
2. Choose your data flow to view it.
3. In the **Data flow** tab that shows your data flow steps, find the **Source** node that you want to edit.
4. Choose the ellipsis icon next to the **Source** node.
5. From the context menu, choose **Edit**.
6. For Amazon S3 data sources and local upload, you have the option to select or upload more files with the same schema as your original data. For query-based data sources such as Amazon Athena, you can remove and select different tables in the visual query builder, or you can edit the SQL query directly. When you're done, choose **Next**.
7. For the **Import settings**, make any desired changes.
8. When you're done, choose **Save changes**.

Your data source should now be updated.

To replace a data source, do the following:

1. In the Canvas application, go to the **Data Wrangler** page.
2. Choose your data flow to view it.
3. In the **Data flow** tab that shows your data flow steps, find the **Source** node that you want to edit.
4. Choose the ellipsis icon next to the **Source** node.
5. From the context menu, choose **Replace**.
6. Go through the [create a data flow experience](#) to select another data source and data.
7. When you've selected your data and are ready to update the source node, choose **Save**.

You should now see the **Source** node updated in your data flow.

## Reorder steps in your data flow

After adding steps to your data flow, you have the option to reorder steps instead of deleting and re-adding them in the correct order. For example, you might decide to move a transform to impute missing values before a step to format strings.

### Note

You can't change the order of certain step types, such as defining your data source, changing data types, joining, concatenating, or splitting. Steps that can't be reordered are grayed out in the Canvas application UI.

To reorder your data flow steps, do the following:

1. While editing a data flow in Data Wrangler, choose the **Data** tab. A side panel called **Steps** lists your data flow steps in order.
2. Hover over a transform step and choose the **More options** icon ( ) next to that step.
3. From the context menu, choose **Reorder**.
4. Drag and drop your data flow steps into your desired order.
5. When you've finished, choose **Save**.

Your data flow steps and graph should now reflect the changes you've made.

## Delete a step from your data flow

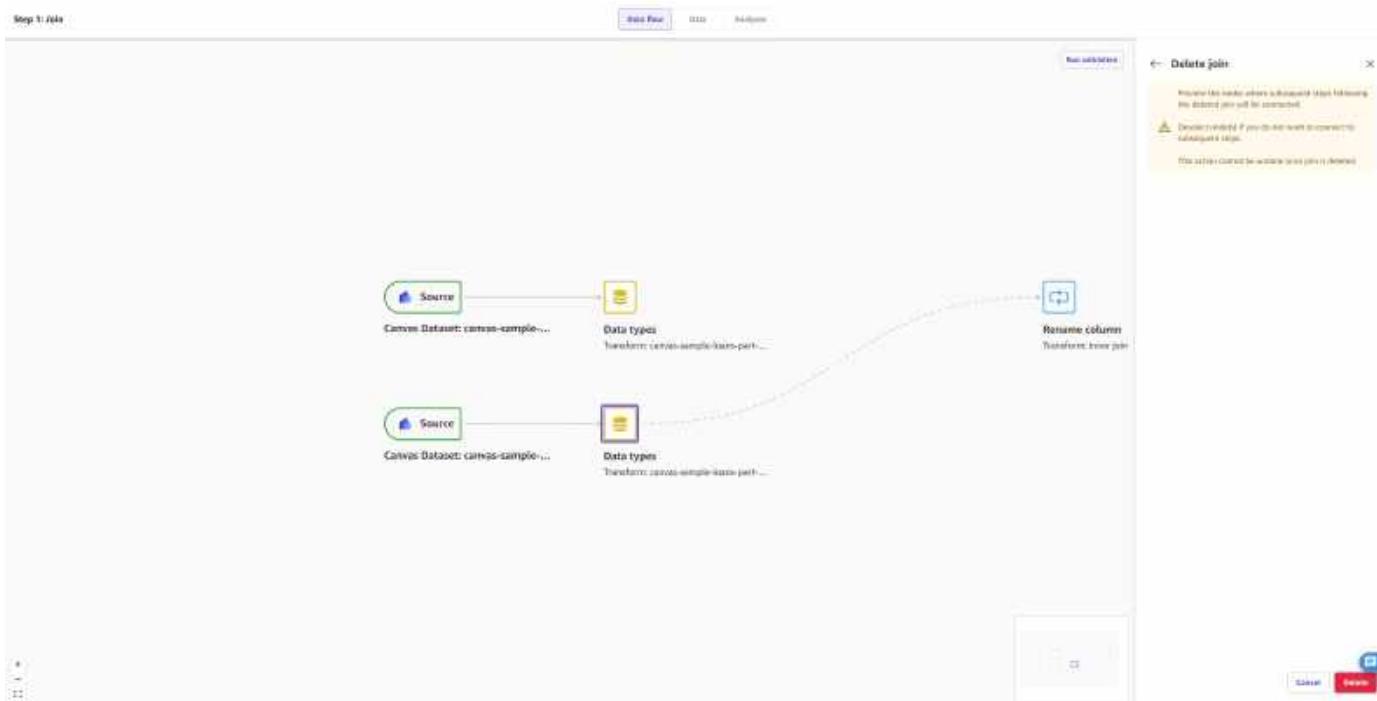
Within your data flows, you have the flexibility to delete your join and concatenate steps and choose whether or not to still apply any subsequent transforms to your data.

To delete a join or concatenate step from your data flow, do the following:

1. Open your data flow.
2. Choose the plus icon (+) next to the join or concatenate node that you want to delete.

3. In the context menu, choose **Delete**.
4. (Optional) If you have transformation steps following the join or concatenate step, then you can choose whether or not to keep the subsequent transformation steps and add them separately to each data node. In the **Delete join** side panel, choose a node to deselect it and remove any subsequent transformation steps. You can leave both nodes selected to keep all transformation steps, or you can deselect both nodes to discard all transformation steps.

The following screenshot shows this step with only the second of two data nodes selected. When the join is successfully deleted, then the subsequent **Rename column** transform is only kept by the second data node.



5. Choose **Delete**.

The join or concatenate step should now be removed from your data flow.

## Perform exploratory data analysis (EDA)

Data Wrangler includes built-in analyses that help you generate visualizations and data analyses in a few clicks. You can also create custom analyses using your own code.

You add an analysis to a dataframe by selecting a step in your data flow, and then choosing **Add analysis**. To access an analysis you've created, select the step that contains the analysis, and select the analysis.

Analyses are generated using a sample of up to 200,000 rows of your dataset, and you can configure the sample size. For more information about changing the sample size of your data flow, see [Edit the data flow sampling configuration](#).

### Note

Analyses are optimized for data with 1000 or fewer columns. You may experience some latency when generating analyses for data with additional columns.

You can add the following analysis to a dataframe:

- Data visualizations, including histograms and scatter plots.
- A quick summary of your dataset, including number of entries, minimum and maximum values (for numeric data), and most and least frequent categories (for categorical data).
- A quick model of the dataset, which can be used to generate an importance score for each feature.
- A target leakage report, which you can use to determine if one or more features are strongly correlated with your target feature.
- A custom visualization using your own code.

Use the following sections to learn more about these options.

### Get insights on data and data quality

Use the **Data Quality and Insights Report** to perform an analysis of the data that you've imported into Data Wrangler. We recommend that you create the report after you import your dataset. You can use the report to help you clean and process your data. It gives you information such as the number of missing values and the number of outliers. If you have issues with your data, such as target leakage or imbalance, the insights report can bring those issues to your attention.

Use the following procedure to create a Data Quality and Insights report. It assumes that you've already imported a dataset into your Data Wrangler flow.

### To create a Data Quality and Insights report

1. Choose the ellipsis icon next to a node in your Data Wrangler flow.

2. Select **Get data insights**.
3. For **Analysis type**, select **Data Quality and Insights Report**.
4. For **Analysis name**, specify a name for the insights report.
5. For **Problem type**, specify **Regression or Classification**.
6. For **Target column**, specify the target column.
7. For **Data size**, specify one of the following:
  - **Sampled dataset** – Uses the interactive sample from your data flow, which can contain up to 200,000 rows of your dataset. For information about how to edit the size of your sample, see [Edit the data flow sampling configuration](#).
  - **Full dataset** – Uses the full dataset from your data source to create the report.

 **Note**

Creating a Data Quality and Insights report on the full dataset uses an Amazon SageMaker processing job. A SageMaker Processing job provisions the additional compute resources required to get insights for all of your data. For more information about SageMaker Processing jobs, see [Data transformation workloads with SageMaker Processing](#).

8. Choose **Create**.

The following topics show the sections of the report:

### Topics

- [Summary](#)
- [Target column](#)
- [Quick model](#)
- [Feature summary](#)
- [Samples](#)
- [Definitions](#)

You can either download the report or view it online. To download the report, choose the download button at the top right corner of the screen.

## Summary

The insights report has a brief summary of the data that includes general information such as missing values, invalid values, feature types, outlier counts, and more. It can also include high severity warnings that point to probable issues with the data. We recommend that you investigate the warnings.

## Target column

When you create the Data Quality and Insights Report, Data Wrangler gives you the option to select a target column. A target column is a column that you're trying to predict. When you choose a target column, Data Wrangler automatically creates a target column analysis. It also ranks the features in the order of their predictive power. When you select a target column, you must specify whether you're trying to solve a regression or a classification problem.

For classification, Data Wrangler shows a table and a histogram of the most common classes. A class is a category. It also presents observations, or rows, with a missing or invalid target value.

For regression, Data Wrangler shows a histogram of all the values in the target column. It also presents observations, or rows, with a missing, invalid, or outlier target value.

## Quick model

The **Quick model** provides an estimate of the expected predicted quality of a model that you train on your data.

Data Wrangler splits your data into training and validation folds. It uses 80% of the samples for training and 20% of the values for validation. For classification, the sample is stratified split. For a stratified split, each data partition has the same ratio of labels. For classification problems, it's important to have the same ratio of labels between the training and classification folds. Data Wrangler trains the XGBoost model with the default hyperparameters. It applies early stopping on the validation data and performs minimal feature preprocessing.

For classification models, Data Wrangler returns both a model summary and a confusion matrix.

To learn more about the information that the classification model summary returns, see [Definitions](#).

A confusion matrix gives you the following information:

- The number of times the predicted label matches the true label.

- The number of times the predicted label doesn't match the true label.

The true label represents an actual observation in your data. For example, if you're using a model to detect fraudulent transactions, the true label represents a transaction that is actually fraudulent or non-fraudulent. The predicted label represents the label that your model assigns to the data.

You can use the confusion matrix to see how well the model predicts the presence or the absence of a condition. If you're predicting fraudulent transactions, you can use the confusion matrix to get a sense of both the sensitivity and the specificity of the model. The sensitivity refers to the model's ability to detect fraudulent transactions. The specificity refers to the model's ability to avoid detecting non-fraudulent transactions as fraudulent.

## Feature summary

When you specify a target column, Data Wrangler orders the features by their prediction power. Prediction power is measured on the data after it is split into 80% training and 20% validation folds. Data Wrangler fits a model for each feature separately on the training fold. It applies minimal feature preprocessing and measures prediction performance on the validation data.

It normalizes the scores to the range [0,1]. Higher prediction scores indicate columns that are more useful for predicting the target on their own. Lower scores point to columns that aren't predictive of the target column.

It's uncommon for a column that isn't predictive on its own to be predictive when it's used in tandem with other columns. You can confidently use the prediction scores to determine whether a feature in your dataset is predictive.

A low score usually indicates the feature is redundant. A score of 1 implies perfect predictive abilities, which often indicates target leakage. Target leakage usually happens when the dataset contains a column that isn't available at the prediction time. For example, it could be a duplicate of the target column.

## Samples

Data Wrangler provides information about whether your samples are anomalous or if there are duplicates in your dataset.

Data Wrangler detects anomalous samples using the *isolation forest algorithm*. The isolation forest associates an anomaly score with each sample (row) of the dataset. Low anomaly scores indicate

anomalous samples. High scores are associated with non-anomalous samples. Samples with a negative anomaly score are usually considered anomalous and samples with positive anomaly score are considered non-anomalous.

When you look at a sample that might be anomalous, we recommend that you pay attention to unusual values. For example, you might have anomalous values that result from errors in gathering and processing the data. The following is an example of the most anomalous samples according to the Data Wrangler's implementation of the isolation forest algorithm. We recommend using domain knowledge and business logic when you examine the anomalous samples.

Data Wrangler detects duplicate rows and calculates the ratio of duplicate rows in your data. Some data sources could include valid duplicates. Other data sources could have duplicates that point to problems in data collection. Duplicate samples that result from faulty data collection could interfere with machine learning processes that rely on splitting the data into independent training and validation folds.

The following are elements of the insights report that can be impacted by duplicated samples:

- Quick model
- Prediction power estimation
- Automatic hyperparameter tuning

You can remove duplicate samples from the dataset using the **Drop duplicates** transform under **Manage rows**. Data Wrangler shows you the most frequently duplicated rows.

## Definitions

The following are definitions for the technical terms that are used in the data insights report.

### Feature types

The following are the definitions for each of the feature types:

- **Numeric** – Numeric values can be either floats or integers, such as age or income. The machine learning models assume that numeric values are ordered and a distance is defined over them. For example, 3 is closer to 4 than to 10 and  $3 < 4 < 10$ .
- **Categorical** – The column entries belong to a set of unique values, which is usually much smaller than the number of entries in the column. For example, a column of length 100 could contain the unique values Dog, Cat, and Mouse. The values could be numeric, text, or a

combination of both. Horse, House, 8, Love, and 3.1 would all be valid values and could be found in the same categorical column. The machine learning model does not assume order or distance on the values of categorical features, as opposed to numeric features, even when all the values are numbers.

- **Binary** – Binary features are a special categorical feature type in which the cardinality of the set of unique values is 2.
- **Text** – A text column contains many non-numeric unique values. In extreme cases, all the elements of the column are unique. In an extreme case, no two entries are the same.
- **Datetime** – A datetime column contains information about the date or time. It can have information about both the date and time.

## Feature statistics

The following are definitions for each of the feature statistics:

- **Prediction power** – Prediction power measures how useful the column is in predicting the target.
- **Outliers** (in numeric columns) – Data Wrangler detects outliers using two statistics that are robust to outliers: median and robust standard deviation (RSTD). RSTD is derived by clipping the feature values to the range [5 percentile, 95 percentile] and calculating the standard deviation of the clipped vector. All values larger than median + 5 \* RSTD or smaller than median - 5 \* RSTD are considered to be outliers.
- **Skew** (in numeric columns) – Skew measures the symmetry of the distribution and is defined as the third moment of the distribution divided by the third power of the standard deviation. The skewness of the normal distribution or any other symmetric distribution is zero. Positive values imply that the right tail of the distribution is longer than the left tail. Negative values imply that the left tail of the distribution is longer than the right tail. As a rule of thumb, a distribution is considered skewed when the absolute value of the skew is larger than 3.
- **Kurtosis** (in numeric columns) – Pearson's kurtosis measures the heaviness of the tail of the distribution. It's defined as the fourth moment of the distribution divided by the square of the second moment. The kurtosis of the normal distribution is 3. Kurtosis values lower than 3 imply that the distribution is concentrated around the mean and the tails are lighter than the tails of the normal distribution. Kurtosis values higher than 3 imply heavier tails or outliers.
- **Missing values** – Null-like objects, empty strings and strings composed of only white spaces are considered missing.

- **Valid values for numeric features or regression target** – All values that you can cast to finite floats are valid. Missing values are not valid.
- **Valid values for categorical, binary, or text features, or for classification target** – All values that are not missing are valid.
- **Datetime features** – All values that you can cast to a datetime object are valid. Missing values are not valid.
- **Invalid values** – Values that are either missing or you can't properly cast. For example, in a numeric column, you can't cast the string "six" or a null value.

## Quick model metrics for regression

The following are the definitions for the quick model metrics:

- R2 or coefficient of determination) – R2 is the proportion of the variation in the target that is predicted by the model. R2 is in the range of [-infty, 1]. 1 is the score of the model that predicts the target perfectly and 0 is the score of the trivial model that always predicts the target mean.
- MSE or mean squared error – MSE is in the range [0, infty]. 0 is the score of the model that predicts the target perfectly.
- MAE or mean absolute error – MAE is in the range [0, infty] where 0 is the score of the model that predicts the target perfectly.
- RMSE or root mean square error – RMSE is in the range [0, infty] where 0 is the score of the model that predicts the target perfectly.
- Max error – The maximum absolute value of the error over the dataset. Max error is in the range [0, infty]. 0 is the score of the model that predicts the target perfectly.
- Median absolute error – Median absolute error is in the range [0, infty]. 0 is the score of the model that predicts the target perfectly.

## Quick model metrics for classification

The following are the definitions for the quick model metrics:

- **Accuracy** – Accuracy is the ratio of samples that are predicted accurately. Accuracy is in the range [0, 1]. 0 is the score of the model that predicts all samples incorrectly and 1 is the score of the perfect model.

- **Balanced accuracy** – Balanced accuracy is the ratio of samples that are predicted accurately when the class weights are adjusted to balance the data. All classes are given the same importance, regardless of their frequency. Balanced accuracy is in the range [0, 1]. 0 is the score of the model that predicts all samples wrong. 1 is the score of the perfect model.
- **AUC (binary classification)** – This is the area under the receiver operating characteristic curve. AUC is in the range [0, 1] where a random model returns a score of 0.5 and the perfect model returns a score of 1.
- **AUC (OVR)** – For multiclass classification, this is the area under the receiver operating characteristic curve calculated separately for each label using one versus rest. Data Wrangler reports the average of the areas. AUC is in the range [0, 1] where a random model returns a score of 0.5 and the perfect model returns a score of 1.
- **Precision** – Precision is defined for a specific class. Precision is the fraction of true positives out of all the instances that the model classified as that class. Precision is in the range [0, 1]. 1 is the score of the model that has no false positives for the class. For binary classification, Data Wrangler reports the precision of the positive class.
- **Recall** – Recall is defined for a specific class. Recall is the fraction of the relevant class instances that are successfully retrieved. Recall is in the range [0, 1]. 1 is the score of the model that classifies all the instances of the class correctly. For binary classification, Data Wrangler reports the recall of the positive class.
- **F1** – F1 is defined for a specific class. It's the harmonic mean of the precision and recall. F1 is in the range [0, 1]. 1 is the score of the perfect model. For binary classification, Data Wrangler reports the F1 for classes with positive values.

## Textual patterns

**Patterns** describe the textual format of a string using an easy to read format. The following are examples of textual patterns:

- "{digits:4-7}" describes a sequence of digits that have a length between 4 and 7.
- "{alnum:5}" describes an alpha-numeric string with a length of exactly 5.

Data Wrangler infers the patterns by looking at samples of non-empty strings from your data. It can describe many of the commonly used patterns. The **confidence** expressed as a percentage indicates how much of the data is estimated to match the pattern. Using the textual pattern, you can see which rows in your data you need to correct or drop.

The following describes the patterns that Data Wrangler can recognize:

Pattern	Textual Format
{alnum}	Alphanumeric strings
{any}	Any string of word characters
{digits}	A sequence of digits
{lower}	A lowercase word
{mixed}	A mixed-case word
{name}	A word beginning with a capital letter
{upper}	An uppercase word
{whitespace}	Whitespace characters

A word character is either an underscore or a character that might appear in a word in any language. For example, the strings 'Hello\_word' and 'écoute' both consist of word characters. 'H' and 'é' are both examples of word characters.

## Bias report

SageMaker Canvas provides the bias report in Data Wrangler to help uncover potential biases in your data. The bias report analyzes the relationship between the target column (label) and a column that you believe might contain bias (facet variable). For example, if you are trying to predict customer conversion, the facet variable may be the age of the customer. The bias report can help you determine whether or not your data is biased toward a certain age group.

To generate a bias report in Canvas, do the following:

1. In your data flow in Data Wrangler, choose the **More options** icon  next to a node in the flow.
2. From the context menu, choose **Get data insights**.

3. The **Create analysis** side panel opens. For the **Analysis type** dropdown menu, select **Bias Report**.
4. In the **Analysis name** field, enter a name for the bias report.
5. For the **Select the column your model predicts (target)** dropdown menu, select your target column.
6. For **Is your predicted column a value or threshold?**, select **Value** if your target column has categorical values or **Threshold** if it has numerical values.
7. For **Predicted value** (or **Predicted threshold**, depending on your selection in the previous step), enter the target column value or values that correspond to a positive outcome. For example, if predicting customer conversion, your value might be yes to indicate that a customer was converted.
8. For the **Select the column to analyze for bias** dropdown menu, select the column that you believe might contain bias, also known as the facet variable.
9. For **Is your column a value or threshold?**, select **Value** if the facet variable has categorical values or **Threshold** if it has numerical values.
10. For **Column value(s) to analyze for bias** (or **Column threshold to analyze for bias**, depending on your selection in the previous step), enter the value or values that you want to analyze for potential bias. For example, if you're checking for bias against customers over a certain age, use the beginning of that age range as your threshold.
11. For **Choose bias metrics**, select the bias metrics you'd like to include in your bias report. Hover over the info icons for more information about each metric.
12. (Optional) When prompted with the option **Would you like to analyze additional metrics?**, select **Yes** to view and include more bias metrics.
13. When you're ready to create the bias report, choose **Add**.

Once generated, the report gives you an overview of the bias metrics you selected. You can view the bias report at any time from the **Analyses** tab of your data flow.

## Histogram

Use histograms to see the counts of feature values for a specific feature. You can inspect the relationships between features using the **Color by** option.

You can use the **Facet by** feature to create histograms of one column, for each value in another column.

## Scatter plot

Use the **Scatter Plot** feature to inspect the relationship between features. To create a scatter plot, select a feature to plot on the **X axis** and the **Y axis**. Both of these columns must be numeric typed columns.

You can color scatter plots by an additional column.

Additionally, you can facet scatter plots by features.

## Table summary

Use the **Table Summary** analysis to quickly summarize your data.

For columns with numerical data, including log and float data, a table summary reports the number of entries (count), minimum (min), maximum (max), mean, and standard deviation (stddev) for each column.

For columns with non-numerical data, including columns with string, Boolean, or date/time data, a table summary reports the number of entries (count), least frequent value (min), and most frequent value (max).

## Quick model

Use the **Quick Model** visualization to quickly evaluate your data and produce importance scores for each feature. A [feature importance score](#) score indicates how useful a feature is at predicting a target label. The feature importance score is between [0, 1] and a higher number indicates that the feature is more important to the whole dataset. On the top of the quick model chart, there is a model score. A classification problem shows an F1 score. A regression problem has a mean squared error (MSE) score.

When you create a quick model chart, you select a dataset you want evaluated, and a target label against which you want feature importance to be compared. Data Wrangler does the following:

- Infers the data types for the target label and each feature in the dataset selected.
- Determines the problem type. Based on the number of distinct values in the label column, Data Wrangler determines if this is a regression or classification problem type. Data Wrangler sets a categorical threshold to 100. If there are more than 100 distinct values in the label column, Data Wrangler classifies it as a regression problem; otherwise, it is classified as a classification problem.

- Pre-processes features and label data for training. The algorithm used requires encoding features to vector type and encoding labels to double type.
- Trains a random forest algorithm with 70% of data. Spark's [RandomForestRegressor](#) is used to train a model for regression problems. The [RandomForestClassifier](#) is used to train a model for classification problems.
- Evaluates a random forest model with the remaining 30% of data. Data Wrangler evaluates classification models using an F1 score and evaluates regression models using an MSE score.
- Calculates feature importance for each feature using the Gini importance method.

## Target leakage

Target leakage occurs when there is data in a machine learning training dataset that is strongly correlated with the target label, but is not available in real-world data. For example, you may have a column in your dataset that serves as a proxy for the column you want to predict with your model.

When you use the **Target Leakage** analysis, you specify the following:

- **Target:** This is the feature about which you want your ML model to be able to make predictions.
- **Problem type:** This is the ML problem type on which you are working. Problem type can either be **classification** or **regression**.
- (Optional) **Max features:** This is the maximum number of features to present in the visualization, which shows features ranked by their risk of being target leakage.

For classification, the target leakage analysis uses the area under the receiver operating characteristic, or AUC - ROC curve for each column, up to **Max features**. For regression, it uses a coefficient of determination, or R2 metric.

The AUC - ROC curve provides a predictive metric, computed individually for each column using cross-validation, on a sample of up to around 1000 rows. A score of 1 indicates perfect predictive abilities, which often indicates target leakage. A score of 0.5 or lower indicates that the information on the column could not provide, on its own, any useful information towards predicting the target. Although it can happen that a column is uninformative on its own but is useful in predicting the target when used in tandem with other features, a low score could indicate the feature is redundant.

## Multicollinearity

Multicollinearity is a circumstance where two or more predictor variables are related to each other. The predictor variables are the features in your dataset that you're using to predict a target variable. When you have multicollinearity, the predictor variables are not only predictive of the target variable, but also predictive of each other.

You can use the **Variance Inflation Factor (VIF)**, **Principal Component Analysis (PCA)**, or **Lasso feature selection** as measures for the multicollinearity in your data. For more information, see the following.

### Variance Inflation Factor (VIF)

The Variance Inflation Factor (VIF) is a measure of collinearity among variable pairs. Data Wrangler returns a VIF score as a measure of how closely the variables are related to each other. A VIF score is a positive number that is greater than or equal to 1.

A score of 1 means that the variable is uncorrelated with the other variables. Scores greater than 1 indicate higher correlation.

Theoretically, you can have a VIF score with a value of infinity. Data Wrangler clips high scores to 50. If you have a VIF score greater than 50, Data Wrangler sets the score to 50.

You can use the following guidelines to interpret your VIF scores:

- A VIF score less than or equal to 5 indicates that the variables are moderately correlated with the other variables.
- A VIF score greater than or equal to 5 indicates that the variables are highly correlated with the other variables.

### Principle Component Analysis (PCA)

Principal Component Analysis (PCA) measures the variance of the data along different directions in the feature space. The feature space consists of all the predictor variables that you use to predict the target variable in your dataset.

For example, if you're trying to predict who survived on the *RMS Titanic* after it hit an iceberg, your feature space can include the passengers' age, gender, and the fare that they paid.

From the feature space, PCA generates an ordered list of variances. These variances are also known as singular values. The values in the list of variances are greater than or equal to 0. We can use them to determine how much multicollinearity there is in our data.

When the numbers are roughly uniform, the data has very few instances of multicollinearity. When there is a lot of variability among the values, we have many instances of multicollinearity. Before it performs PCA, Data Wrangler normalizes each feature to have a mean of 0 and a standard deviation of 1.

### Note

PCA in this circumstance can also be referred to as Singular Value Decomposition (SVD).

## Lasso feature selection

Lasso feature selection uses the L1 regularization technique to only include the most predictive features in your dataset.

For both classification and regression, the regularization technique generates a coefficient for each feature. The absolute value of the coefficient provides an importance score for the feature. A higher importance score indicates that it is more predictive of the target variable. A common feature selection method is to use all the features that have a non-zero lasso coefficient.

## Detect anomalies in time series data

You can use the anomaly detection visualization to see outliers in your time series data. To understand what determines an anomaly, you need to understand that we decompose the time series into a predicted term and an error term. We treat the seasonality and trend of the time series as the predicted term. We treat the residuals as the error term.

For the error term, you specify a threshold as the number of standard of deviations the residual can be away from the mean for it to be considered an anomaly. For example, you can specify a threshold as being 3 standard deviations. Any residual greater than 3 standard deviations away from the mean is an anomaly.

You can use the following procedure to perform an **Anomaly detection** analysis.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the **+**, and select **Add analysis**.

3. For **Analysis type**, choose **Time Series**.
4. For **Visualization**, choose **Anomaly detection**.
5. For **Anomaly threshold**, choose the threshold that a value is considered an anomaly.
6. Choose **Preview** to generate a preview of the analysis.
7. Choose **Add** to add the transform to the Data Wrangler data flow.

## Seasonal trend decomposition in time series data

You can determine whether there's seasonality in your time series data by using the Seasonal Trend Decomposition visualization. We use the STL (Seasonal Trend decomposition using LOESS) method to perform the decomposition. We decompose the time series into its seasonal, trend, and residual components. The trend reflects the long term progression of the series. The seasonal component is a signal that recurs in a time period. After removing the trend and the seasonal components from the time series, you have the residual.

You can use the following procedure to perform a **Seasonal-Trend decomposition** analysis.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the **+**, and select **Add analysis**.
3. For **Analysis type**, choose **Time Series**.
4. For **Visualization**, choose **Seasonal-Trend decomposition**.
5. For **Anomaly threshold**, choose the threshold that a value is considered an anomaly.
6. Choose **Preview** to generate a preview of the analysis.
7. Choose **Add** to add the transform to the Data Wrangler data flow.

## Create custom visualizations

You can add an analysis to your Data Wrangler flow to create a custom visualization. Your dataset, with all the transformations you've applied, is available as a [Pandas DataFrame](#). Data Wrangler uses the `df` variable to store the dataframe. You access the dataframe by calling the variable.

You must provide the output variable, `chart`, to store an [Altair](#) output chart. For example, you can use the following code block to create a custom histogram using the Titanic dataset.

```
import altair as alt
df = df.iloc[:30]
```

```
df = df.rename(columns={"Age": "value"})
df = df.assign(count=df.groupby('value').value.transform('count'))
df = df[['value', 'count']]
base = alt.Chart(df)
bar = base.mark_bar().encode(x=alt.X('value', bin=True, axis=None), y=alt.Y('count'))
rule = base.mark_rule(color='red').encode(
    x='mean(value):Q',
    size=alt.value(5))
chart = bar + rule
```

## To create a custom visualization:

1. Next to the node containing the transformation that you'd like to visualize, choose the **+**.
2. Choose **Add analysis**.
3. For **Analysis type**, choose **Custom Visualization**.
4. For **Analysis name**, specify a name.
5. Enter your code in the code box.
6. Choose **Preview** to preview your visualization.
7. Choose **Save** to add your visualization.

If you don't know how to use the Altair visualization package in Python, you can use custom code snippets to help you get started.

Data Wrangler has a searchable collection of visualization snippets. To use a visualization snippet, choose **Search example snippets** and specify a query in the search bar.

The following example uses the **Binned scatterplot** code snippet. It plots a histogram for 2 dimensions.

The snippets have comments to help you understand the changes that you need to make to the code. You usually need to specify the column names of your dataset in the code.

```
import altair as alt

# Specify the number of top rows for plotting
rows_number = 1000
df = df.head(rows_number)
# You can also choose bottom rows or randomly sampled rows
```

```
# df = df.tail(rows_number)
# df = df.sample(rows_number)

chart = (
    alt.Chart(df)
    .mark_circle()
    .encode(
        # Specify the column names for binning and number of bins for X and Y axis
        x=alt.X("col1:Q", bin=alt.Bin(maxbins=20)),
        y=alt.Y("col2:Q", bin=alt.Bin(maxbins=20)),
        size="count()",
    )
)

# :Q specifies that label column has quantitative type.
# For more details on Altair typing refer to
# https://altair-viz.github.io/user_guide/encoding.html#encoding-data-types
```

## Transform data

Amazon SageMaker Data Wrangler provides numerous ML data transforms to streamline cleaning and featurizing your data. Using the interactive data preparation tools in Data Wrangler, you can sample datasets of any size with a variety of sampling techniques and start exploring your data in a matter of minutes. After finalizing your data transforms on the sampled data, you can then scale the data flow to apply those transformations to the entire dataset.

When you add a transform, it adds a step to the data flow. Each transform you add modifies your dataset and produces a new dataframe. All subsequent transforms apply to the resulting dataframe.

Data Wrangler includes built-in transforms, which you can use to transform columns without any code. If you know how you want to prepare your data but don't know how to get started or which transforms to use, you can use the chat for data prep feature to interact conversationally with Data Wrangler and apply transforms using natural language. For more information, see [Chat for data prep](#).

You can also add custom transformations using PySpark, Python (User-Defined Function), pandas, and PySpark SQL. Some transforms operate in place, while others create a new output column in your dataset.

You can apply transforms to multiple columns at once. For example, you can delete multiple columns in a single step.

You can apply the **Process numeric** and **Handle missing** transforms only to a single column.

Use this page to learn more about the built-in and custom transforms offered by Data Wrangler.

## Join Datasets

You can join datasets directly in your data flow. When you join two datasets, the resulting joined dataset appears in your flow. The following join types are supported by Data Wrangler.

- **Left outer** – Include all rows from the left table. If the value for the column joined on a left table row does not match any right table row values, that row contains null values for all right table columns in the joined table.
- **Left anti** – Include rows from the left table that do not contain values in the right table for the joined column.
- **Left semi** – Include a single row from the left table for all identical rows that satisfy the criteria in the join statement. This excludes duplicate rows from the left table that match the criteria of the join.
- **Right outer** – Include all rows from the right table. If the value for the joined column in a right table row does not match any left table row values, that row contains null values for all left table columns in the joined table.
- **Inner** – Include rows from left and right tables that contain matching values in the joined column.
- **Full outer** – Include all rows from the left and right tables. If the row value for the joined column in either table does not match, separate rows are created in the joined table. If a row doesn't contain a value for a column in the joined table, null is inserted for that column.
- **Cartesian cross** – Include rows which combine each row from the first table with each row from the second table. This is a [Cartesian product](#) of rows from tables in the join. The result of this product is the size of the left table times the size of the right table. Therefore, we recommend caution in using this join between very large datasets.

Use the following procedure to join two datasets. You should have already imported two data sources into your data flow.

1. Select the **More options** icon  next to the left node that you want to join. The first node you select is always the left table in your join.
2. Hover over **Combine data**, and then choose **Join**.
3. Select the right node. The second node you select is always the right table in your join.
4. The **Join type** field is set to **Inner join** by default. Select the dropdown menu to change the join type.
5. For **Join keys**, verify the columns from the left and right tables that you want to use to join the data. You can add or remove additional join keys.
6. For **Name of join**, enter a name for the joined data, or use the default name.
7. (Optional) Choose **Preview** to preview the joined data.
8. Choose **Add** to complete the join.

 **Note**

If you receive a notice that Canvas didn't identify any matching rows when joining your data, we recommend that you either verify that you've selected the correct columns, or update your sample to try to find matching rows. You can choose a different sampling strategy or change the size of the sample. For information about how to edit the sample, see [Edit the data flow sampling configuration](#).

You should now see a join node added to your data flow.

## Concatenate Datasets

Concatenating combines two datasets by appending the rows from one dataset to another.

Use the following procedure to concatenate two datasets. You should have already imported two data sources into your data flow.

### To concatenate two datasets:

1. Select the **More options** icon 

next to the left node that you want to concatenate. The first node you select is always the left table in your concatenate operation.

2. Hover over **Combine data**, and then choose **Concatenate**.
3. Select the right node. The second node you select is always the right table in your concatenate.
4. (Optional) Select the checkbox next to **Remove duplicates after concatenation** to remove duplicate columns.
5. (Optional) Select the checkbox next to **Add column to indicate source dataframe** to add a column to the resulting dataframe that lists the source dataset for each record.
  - a. For **Indicator column name**, enter a name for the added column.
  - b. For **First dataset indicating string**, enter the value you want to use to mark records from the first dataset (or the left node).
  - c. For **Second dataset indicating string**, enter the value you want to use to mark records from the second dataset (or the right node).
6. For **Name of concatenate**, enter a name for the concatenation.
7. (Optional) Choose **Preview** to preview the concatenated data.
8. Choose **Add** to add the new dataset to your data flow.

You should now see a concatenate node added to your data flow.

## Balance Data

You can balance the data for datasets with an underrepresented category. Balancing a dataset can help you create better models for binary classification.

### Note

You can't balance datasets containing column vectors.

You can use the **Balance data** operation to balance your data using one of the following operators:

- *Random oversampling* – Randomly duplicates samples in the minority category. For example, if you're trying to detect fraud, you might only have cases of fraud in 10% of your data. For an equal proportion of fraudulent and non-fraudulent cases, this operator randomly duplicates fraud cases within the dataset 8 times.

- *Random undersampling* – Roughly equivalent to random oversampling. Randomly removes samples from the overrepresented category to get the proportion of samples that you desire.
- *Synthetic Minority Oversampling Technique (SMOTE)* – Uses samples from the underrepresented category to interpolate new synthetic minority samples. For more information about SMOTE, see the following description.

You can use all transforms for datasets containing both numeric and non-numeric features. SMOTE interpolates values by using neighboring samples. Data Wrangler uses the R-squared distance to determine the neighborhood to interpolate the additional samples. Data Wrangler only uses numeric features to calculate the distances between samples in the underrepresented group.

For two real samples in the underrepresented group, Data Wrangler interpolates the numeric features by using a weighted average. It randomly assigns weights to those samples in the range of [0, 1]. For numeric features, Data Wrangler interpolates samples using a weighted average of the samples. For samples A and B, Data Wrangler could randomly assign a weight of 0.7 to A and 0.3 to B. The interpolated sample has a value of  $0.7A + 0.3B$ .

Data Wrangler interpolates non-numeric features by copying from either of the interpolated real samples. It copies the samples with a probability that it randomly assigns to each sample. For samples A and B, it can assign probabilities 0.8 to A and 0.2 to B. For the probabilities it assigned, it copies A 80% of the time.

## Custom Transforms

The **Custom Transforms** group allows you to use Python (User-Defined Function), PySpark, pandas, or PySpark (SQL) to define custom transformations. For all three options, you use the variable `df` to access the dataframe to which you want to apply the transform. To apply your custom code to your dataframe, assign the dataframe with the transformations that you've made to the `df` variable. If you're not using Python (User-Defined Function), you don't need to include a return statement. Choose **Preview** to preview the result of the custom transform. Choose **Add** to add the custom transform to your list of **Previous steps**.

You can import the popular libraries with an `import` statement in the custom transform code block, such as the following:

- NumPy version 1.19.0
- scikit-learn version 0.23.2
- SciPy version 1.5.4

- pandas version 1.0.3
- PySpark version 3.0.0

### Important

**Custom transform** doesn't support columns with spaces or special characters in the name. We recommend that you specify column names that only have alphanumeric characters and underscores. You can use the **Rename column** transform in the **Manage columns** transform group to remove spaces from a column's name. You can also add a **Python (Pandas) Custom transform** similar to the following to remove spaces from multiple columns in a single step. This example changes columns named A column and B column to A\_column and B\_column respectively.

```
df.rename(columns={"A column": "A_column", "B column": "B_column"})
```

If you include print statements in the code block, the result appears when you select **Preview**. You can resize the custom code transformer panel. Resizing the panel provides more space to write code.

The following sections provide additional context and examples for writing custom transform code.

## Python (User-Defined Function)

The Python function gives you the ability to write custom transformations without needing to know Apache Spark or pandas. Data Wrangler is optimized to run your custom code quickly. You get similar performance using custom Python code and an Apache Spark plugin.

To use the Python (User-Defined Function) code block, you specify the following:

- **Input column** – The input column where you're applying the transform.
- **Mode** – The scripting mode, either pandas or Python.
- **Return type** – The data type of the value that you're returning.

Using the pandas mode gives better performance. The Python mode makes it easier for you to write transformations by using pure Python functions.

## PySpark

The following example extracts date and time from a timestamp.

```
from pyspark.sql.functions import from_unixtime, to_date, date_format
df = df.withColumn('DATE_TIME', from_unixtime('TIMESTAMP'))
df = df.withColumn( 'EVENT_DATE', to_date('DATE_TIME')).withColumn(
'EVENT_TIME', date_format('DATE_TIME', 'HH:mm:ss'))
```

## pandas

The following example provides an overview of the dataframe to which you are adding transforms.

```
df.info()
```

## PySpark (SQL)

The following example creates a new dataframe with four columns: *name*, *fare*, *pclass*, *survived*.

```
SELECT name, fare, pclass, survived FROM df
```

If you don't know how to use PySpark, you can use custom code snippets to help you get started.

Data Wrangler has a searchable collection of code snippets. You can use to code snippets to perform tasks such as dropping columns, grouping by columns, or modelling.

To use a code snippet, choose **Search example snippets** and specify a query in the search bar. The text you specify in the query doesn't have to match the name of the code snippet exactly.

The following example shows a **Drop duplicate rows** code snippet that can delete rows with similar data in your dataset. You can find the code snippet by searching for one of the following:

- Duplicates
- Identical
- Remove

The following snippet has comments to help you understand the changes that you need to make. For most snippets, you must specify the column names of your dataset in the code.

```
# Specify the subset of columns  
# all rows having identical values in these columns will be dropped  
  
subset = ["col1", "col2", "col3"]  
df = df.dropDuplicates(subset)  
  
# to drop the full-duplicate rows run  
# df = df.dropDuplicates()
```

To use a snippet, copy and paste its content into the **Custom transform** field. You can copy and paste multiple code snippets into the custom transform field.

## Custom Formula

Use **Custom formula** to define a new column using a Spark SQL expression to query data in the current dataframe. The query must use the conventions of Spark SQL expressions.

### Important

**Custom formula** doesn't support columns with spaces or special characters in the name. We recommend that you specify column names that only have alphanumeric characters and underscores. You can use the **Rename column** transform in the **Manage columns** transform group to remove spaces from a column's name. You can also add a **Python (Pandas) Custom transform** similar to the following to remove spaces from multiple columns in a single step. This example changes columns named A column and B column to A\_column and B\_column respectively.

```
df.rename(columns={"A column": "A_column", "B column": "B_column"})
```

You can use this transform to perform operations on columns, referencing the columns by name. For example, assuming the current dataframe contains columns named *col\_a* and *col\_b*, you can use the following operation to produce an **Output column** that is the product of these two columns with the following code:

```
col_a * col_b
```

Other common operations include the following, assuming a dataframe contains `col_a` and `col_b` columns:

- Concatenate two columns: `concat(col_a, col_b)`
- Add two columns: `col_a + col_b`
- Subtract two columns: `col_a - col_b`
- Divide two columns: `col_a / col_b`
- Take the absolute value of a column: `abs(col_a)`

For more information, see the [Spark documentation](#) on selecting data.

## Reduce Dimensionality within a Dataset

Reduce the dimensionality in your data by using Principal Component Analysis (PCA). The dimensionality of your dataset corresponds to the number of features. When you use dimensionality reduction in Data Wrangler, you get a new set of features called components. Each component accounts for some variability in the data.

The first component accounts for the largest amount of variation in the data. The second component accounts for the second largest amount of variation in the data, and so on.

You can use dimensionality reduction to reduce the size of the data sets that you use to train models. Instead of using the features in your dataset, you can use the principal components instead.

To perform PCA, Data Wrangler creates axes for your data. An axis is an affine combination of columns in your dataset. The first principal component is the value on the axis that has the largest amount of variance. The second principal component is the value on the axis that has the second largest amount of variance. The nth principal component is the value on the axis that has the nth largest amount of variance.

You can configure the number of principal components that Data Wrangler returns. You can either specify the number of principal components directly or you can specify the variance threshold percentage. Each principal component explains an amount of variance in the data. For example, you might have a principal component with a value of 0.5. The component would explain 50% of the variation in the data. When you specify a variance threshold percentage, Data Wrangler returns the smallest number of components that meet the percentage that you specify.

The following are example principal components with the amount of variance that they explain in the data.

- Component 1 – 0.5
- Component 2 – 0.45
- Component 3 – 0.05

If you specify a variance threshold percentage of 94 or 95, Data Wrangler returns Component 1 and Component 2. If you specify a variance threshold percentage of 96, Data Wrangler returns all three principal components.

You can use the following procedure to run PCA on your dataset.

To run PCA on your dataset, do the following.

1. Open your Data Wrangler data flow.
2. Choose the **+**, and select **Add transform**.
3. Choose **Add step**.
4. Choose **Dimensionality Reduction**.
5. For **Input Columns**, choose the features that you're reducing into the principal components.
6. (Optional) For **Number of principal components**, choose the number of principal components that Data Wrangler returns in your dataset. If specify a value for the field, you can't specify a value for **Variance threshold percentage**.
7. (Optional) For **Variance threshold percentage**, specify the percentage of variation in the data that you want explained by the principal components. Data Wrangler uses the default value of 95 if you don't specify a value for the variance threshold. You can't specify a variance threshold percentage if you've specified a value for **Number of principal components**.
8. (Optional) Deselect **Center** to not use the mean of the columns as the center of the data. By default, Data Wrangler centers the data with the mean before scaling.
9. (Optional) Deselect **Scale** to not scale the data with the unit standard deviation.
10. (Optional) Choose **Columns** to output the components to separate columns. Choose **Vector** to output the components as a single vector.
11. (Optional) For **Output column**, specify a name for an output column. If you're outputting the components to separate columns, the name that you specify is a prefix. If you're outputting the components to a vector, the name that you specify is the name of the vector column.

12. (Optional) Select **Keep input columns**. We don't recommend selecting this option if you plan on only using the principal components to train your model.
13. Choose **Preview**.
14. Choose **Add**.

## Encode Categorical

Categorical data is usually composed of a finite number of categories, where each category is represented with a string. For example, if you have a table of customer data, a column that indicates the country a person lives in is categorical. The categories would be *Afghanistan*, *Albania*, *Algeria*, and so on. Categorical data can be *nominal* or *ordinal*. Ordinal categories have an inherent order, and nominal categories do not. The highest degree obtained (*High school*, *Bachelors*, *Masters*, and so on) is an example of ordinal categories.

Encoding categorical data is the process of creating a numerical representation for categories. For example, if your categories are *Dog* and *Cat*, you may encode this information into two vectors,  $[1, 0]$  to represent *Dog*, and  $[0, 1]$  to represent *Cat*.

When you encode ordinal categories, you may need to translate the natural order of categories into your encoding. For example, you can represent the highest degree obtained with the following map: `{"High school": 1, "Bachelors": 2, "Masters":3}`.

Use categorical encoding to encode categorical data that is in string format into arrays of integers.

The Data Wrangler categorical encoders create encodings for all categories that exist in a column at the time the step is defined. If new categories have been added to a column when you start a Data Wrangler job to process your dataset at time  $t$ , and this column was the input for a Data Wrangler categorical encoding transform at time  $t-1$ , these new categories are considered *missing* in the Data Wrangler job. The option you select for **Invalid handling strategy** is applied to these missing values. Examples of when this can occur are:

- When you use a .flow file to create a Data Wrangler job to process a dataset that was updated after the creation of the data flow. For example, you may use a data flow to regularly process sales data each month. If that sales data is updated weekly, new categories may be introduced into columns for which an encode categorical step is defined.
- When you select **Sampling** when you import your dataset, some categories may be left out of the sample.

In these situations, these new categories are considered missing values in the Data Wrangler job.

You can choose from and configure an *ordinal* and a *one-hot encode*. Use the following sections to learn more about these options.

Both transforms create a new column named **Output column name**. You specify the output format of this column with **Output style**:

- Select **Vector** to produce a single column with a sparse vector.
- Select **Columns** to create a column for every category with an indicator variable for whether the text in the original column contains a value that is equal to that category.

## Ordinal Encode

Select **Ordinal encode** to encode categories into an integer between 0 and the total number of categories in the **Input column** you select.

**Invalid handing strategy:** Select a method to handle invalid or missing values.

- Choose **Skip** if you want to omit the rows with missing values.
- Choose **Keep** to retain missing values as the last category.
- Choose **Error** if you want Data Wrangler to throw an error if missing values are encountered in the **Input column**.
- Choose **Replace with NaN** to replace missing with NaN. This option is recommended if your ML algorithm can handle missing values. Otherwise, the first three options in this list may produce better results.

## One-Hot Encode

Select **One-hot encode** for **Transform** to use one-hot encoding. Configure this transform using the following:

- **Drop last category:** If True, the last category does not have a corresponding index in the one-hot encoding. When missing values are possible, a missing category is always the last one and setting this to True means that a missing value results in an all zero vector.
- **Invalid handing strategy:** Select a method to handle invalid or missing values.
  - Choose **Skip** if you want to omit the rows with missing values.

- Choose **Keep** to retain missing values as the last category.
- Choose **Error** if you want Data Wrangler to throw an error if missing values are encountered in the **Input column**.
- **Is input ordinal encoded:** Select this option if the input vector contains ordinal encoded data. This option requires that input data contain non-negative integers. If **True**, input  $i$  is encoded as a vector with a non-zero in the  $i$ th location.

## Similarity encode

Use similarity encoding when you have the following:

- A large number of categorical variables
- Noisy data

The similarity encoder creates embeddings for columns with categorical data. An embedding is a mapping of discrete objects, such as words, to vectors of real numbers. It encodes similar strings to vectors containing similar values. For example, it creates very similar encodings for "California" and "California".

Data Wrangler converts each category in your dataset into a set of tokens using a 3-gram tokenizer. It converts the tokens into an embedding using min-hash encoding.

The similarity encodings that Data Wrangler creates:

- Have low dimensionality
- Are scalable to a large number of categories
- Are robust and resistant to noise

For the preceding reasons, similarity encoding is more versatile than one-hot encoding.

To add the similarity encoding transform to your dataset, use the following procedure.

To use similarity encoding, do the following.

1. Sign in to the [Amazon SageMaker AI Console](#).
2. Choose **Open Studio Classic**.
3. Choose **Launch app**.

4. Choose **Studio**.
5. Specify your data flow.
6. Choose a step with a transformation.
7. Choose **Add step**.
8. Choose **Encode categorical**.
9. Specify the following:
  - **Transform – Similarity encode**
  - **Input column** – The column containing the categorical data that you're encoding.
  - **Target dimension** – (Optional) The dimension of the categorical embedding vector. The default value is 30. We recommend using a larger target dimension if you have a large dataset with many categories.
  - **Output style** – Choose **Vector** for a single vector with all of the encoded values. Choose **Column** to have the encoded values in separate columns.
  - **Output column** – (Optional) The name of the output column for a vector encoded output. For a column-encoded output, this is the prefix of the column names followed by listed number.

## Featurize Text

Use the **Featurize Text** transform group to inspect string-typed columns and use text embedding to featurize these columns.

This feature group contains two features, *Character statistics* and *Vectorize*. Use the following sections to learn more about these transforms. For both options, the **Input column** must contain text data (string type).

### Character Statistics

Use **Character statistics** to generate statistics for each row in a column containing text data.

This transform computes the following ratios and counts for each row, and creates a new column to report the result. The new column is named using the input column name as a prefix and a suffix that is specific to the ratio or count.

- **Number of words:** The total number of words in that row. The suffix for this output column is -  
stats\_word\_count.

- **Number of characters:** The total number of characters in that row. The suffix for this output column is `-stats_char_count`.
- **Ratio of upper:** The number of uppercase characters, from A to Z, divided by all characters in the column. The suffix for this output column is `-stats_capital_ratio`.
- **Ratio of lower:** The number of lowercase characters, from a to z, divided by all characters in the column. The suffix for this output column is `-stats_lower_ratio`.
- **Ratio of digits:** The ratio of digits in a single row over the sum of digits in the input column. The suffix for this output column is `-stats_digit_ratio`.
- **Special characters ratio:** The ratio of non-alphanumeric (characters like #\$&%:@) characters to over the sum of all characters in the input column. The suffix for this output column is `-stats_special_ratio`.

## Vectorize

Text embedding involves mapping words or phrases from a vocabulary to vectors of real numbers. Use the Data Wrangler text embedding transform to tokenize and vectorize text data into term frequency-inverse document frequency (TF-IDF) vectors.

When TF-IDF is calculated for a column of text data, each word in each sentence is converted to a real number that represents its semantic importance. Higher numbers are associated with less frequent words, which tend to be more meaningful.

When you define a **Vectorize** transform step, Data Wrangler uses the data in your dataset to define the count vectorizer and TF-IDF methods . Running a Data Wrangler job uses these same methods.

You configure this transform using the following:

- **Output column name:** This transform creates a new column with the text embedding. Use this field to specify a name for this output column.
- **Tokenizer:** A tokenizer converts the sentence into a list of words, or *tokens*.

Choose **Standard** to use a tokenizer that splits by white space and converts each word to lowercase. For example, "Good dog" is tokenized to ["good", "dog"].

Choose **Custom** to use a customized tokenizer. If you choose **Custom**, you can use the following fields to configure the tokenizer:

- **Minimum token length:** The minimum length, in characters, for a token to be valid. Defaults to 1. For example, if you specify 3 for minimum token length, words like a, at, in are dropped from the tokenized sentence.
- **Should regex split on gaps:** If selected, **regex** splits on gaps. Otherwise, it matches tokens. Defaults to True.
- **Regex pattern:** Regex pattern that defines the tokenization process. Defaults to '`\s+`'.
- **To lowercase:** If chosen, Data Wrangler converts all characters to lowercase before tokenization. Defaults to True.

To learn more, see the Spark documentation on [Tokenizer](#).

- **Vectorizer:** The vectorizer converts the list of tokens into a sparse numeric vector. Each token corresponds to an index in the vector and a non-zero indicates the existence of the token in the input sentence. You can choose from two vectorizer options, *Count* and *Hashing*.
  - **Count vectorize** allows customizations that filter infrequent or too common tokens. **Count vectorize parameters** include the following:
    - **Minimum term frequency:** In each row, terms (tokens) with smaller frequency are filtered. If you specify an integer, this is an absolute threshold (inclusive). If you specify a fraction between 0 (inclusive) and 1, the threshold is relative to the total term count. Defaults to 1.
    - **Minimum document frequency:** Minimum number of rows in which a term (token) must appear to be included. If you specify an integer, this is an absolute threshold (inclusive). If you specify a fraction between 0 (inclusive) and 1, the threshold is relative to the total term count. Defaults to 1.
    - **Maximum document frequency:** Maximum number of documents (rows) in which a term (token) can appear to be included. If you specify an integer, this is an absolute threshold (inclusive). If you specify a fraction between 0 (inclusive) and 1, the threshold is relative to the total term count. Defaults to 0.999.
    - **Maximum vocabulary size:** Maximum size of the vocabulary. The vocabulary is made up of all terms (tokens) in all rows of the column. Defaults to 262144.
    - **Binary outputs:** If selected, the vector outputs do not include the number of appearances of a term in a document, but rather are a binary indicator of its appearance. Defaults to False.

To learn more about this option, see the Spark documentation on [CountVectorizer](#).

- **Hashing** is computationally faster. **Hash vectorize parameters** includes the following:

- **Number of features during hashing:** A hash vectorizer maps tokens to a vector index according to their hash value. This feature determines the number of possible hash values. Large values result in fewer collisions between hash values but a higher dimension output vector.

To learn more about this option, see the Spark documentation on [FeatureHasher](#)

- **Apply IDF** applies an IDF transformation, which multiplies the term frequency with the standard inverse document frequency used for TF-IDF embedding. **IDF parameters** include the following:
  - **Minimum document frequency** : Minimum number of documents (rows) in which a term (token) must appear to be included. If **count\_vectorize** is the chosen vectorizer, we recommend that you keep the default value and only modify the **min\_doc\_freq** field in **Count vectorize parameters**. Defaults to 5.
  - **Output format:** The output format of each row.
    - Select **Vector** to produce a single column with a sparse vector.
    - Select **Flattened** to create a column for every category with an indicator variable for whether the text in the original column contains a value that is equal to that category. You can only choose flattened when **Vectorizer** is set as **Count vectorizer**.

## Transform Time Series

In Data Wrangler, you can transform time series data. The values in a time series dataset are indexed to specific time. For example, a dataset that shows the number of customers in a store for each hour in a day is a time series dataset. The following table shows an example of a time series dataset.

Hourly number of customers in a store

Number of customers	Time (hour)
4	09:00
10	10:00
14	11:00
25	12:00

Number of customers	Time (hour)
20	13:00
18	14:00

For the preceding table, the **Number of Customers** column contains the time series data. The time series data is indexed on the hourly data in the **Time (hour)** column.

You might need to perform a series of transformations on your data to get it in a format that you can use for your analysis. Use the **Time series** transform group to transform your time series data. For more information about the transformations that you can perform, see the following sections.

## Topics

- [Group by a Time Series](#)
- [Resample Time Series Data](#)
- [Handle Missing Time Series Data](#)
- [Validate the Timestamp of Your Time Series Data](#)
- [Standardizing the Length of the Time Series](#)
- [Extract Features from Your Time Series Data](#)
- [Use Lagged Features from Your Time Series Data](#)
- [Create a Datetime Range In Your Time Series](#)
- [Use a Rolling Window In Your Time Series](#)

## Group by a Time Series

You can use the group by operation to group time series data for specific values in a column.

For example, you have the following table that tracks the average daily electricity usage in a household.

Average daily household electricity usage

Household ID	Daily timestamp	Electricity usage (kWh)	Number of household occupants
household_0	1/1/2020	30	2
household_0	1/2/2020	40	2
household_0	1/4/2020	35	3
household_1	1/2/2020	45	3
household_1	1/3/2020	55	4

If you choose to group by ID, you get the following table.

Electricity usage grouped by household ID

Household ID	Electricity usage series (kWh)	Number of household occupants series
household_0	[30, 40, 35]	[2, 2, 3]
household_1	[45, 55]	[3, 4]

Each entry in the time series sequence is ordered by the corresponding timestamp. The first element of the sequence corresponds to the first timestamp of the series. For household\_0, 30 is the first value of the **Electricity Usage Series**. The value of 30 corresponds to the first timestamp of 1/1/2020.

You can include the starting timestamp and ending timestamp. The following table shows how that information appears.

Electricity usage grouped by household ID

Household ID	Electricity usage series (kWh)	Number of household occupants series	Start_time	End_time
household_0	[30, 40, 35]	[2, 2, 3]	1/1/2020	1/4/2020
household_1	[45, 55]	[3, 4]	1/2/2020	1/3/2020

You can use the following procedure to group by a time series column.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the +, and select **Add transform**.
3. Choose **Add step**.
4. Choose **Time Series**.
5. Under **Transform**, choose **Group by**.
6. Specify a column in **Group by this column**.
7. For **Apply to columns**, specify a value.
8. Choose **Preview** to generate a preview of the transform.
9. Choose **Add** to add the transform to the Data Wrangler data flow.

## Resample Time Series Data

Time series data usually has observations that aren't taken at regular intervals. For example, a dataset could have some observations that are recorded hourly and other observations that are recorded every two hours.

Many analyses, such as forecasting algorithms, require the observations to be taken at regular intervals. Resampling gives you the ability to establish regular intervals for the observations in your dataset.

You can either upsample or downsample a time series. Downsampling increases the interval between observations in the dataset. For example, if you downsample observations that are taken either every hour or every two hours, each observation in your dataset is taken every two hours. The hourly observations are aggregated into a single value using an aggregation method such as the mean or median.

Upsampling reduces the interval between observations in the dataset. For example, if you upsample observations that are taken every two hours into hourly observations, you can use an interpolation method to infer hourly observations from the ones that have been taken every two hours. For information on interpolation methods, see [pandas.DataFrame.interpolate](#).

You can resample both numeric and non-numeric data.

Use the **Resample** operation to resample your time series data. If you have multiple time series in your dataset, Data Wrangler standardizes the time interval for each time series.

The following table shows an example of downsampling time series data by using the mean as the aggregation method. The data is downsampled from every two hours to every hour.

Hourly temperature readings over a day before downsampling

Timestamp	Temperature (Celsius)
12:00	30
1:00	32
2:00	35
3:00	32
4:00	30

Temperature readings downsampled to every two hours

Timestamp	Temperature (Celsius)
12:00	30
2:00	33.5
4:00	35

You can use the following procedure to resample time series data.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the **+**, and select **Add transform**.
3. Choose **Add step**.
4. Choose **Resample**.
5. For **Timestamp**, choose the timestamp column.
6. For **Frequency unit**, specify the frequency that you're resampling.
7. (Optional) Specify a value for **Frequency quantity**.
8. Configure the transform by specifying the remaining fields.
9. Choose **Preview** to generate a preview of the transform.
10. Choose **Add** to add the transform to the Data Wrangler data flow.

## Handle Missing Time Series Data

If you have missing values in your dataset, you can do one of the following:

- For datasets that have multiple time series, drop the time series that have missing values that are greater than a threshold that you specify.
- Impute the missing values in a time series by using other values in the time series.

Imputing a missing value involves replacing the data by either specifying a value or by using an inferential method. The following are the methods that you can use for imputation:

- Constant value – Replace all the missing data in your dataset with a value that you specify.
- Most common value – Replace all the missing data with the value that has the highest frequency in the dataset.
- Forward fill – Use a forward fill to replace the missing values with the non-missing value that precedes the missing values. For the sequence: [2, 4, 7, NaN, NaN, NaN, 8], all of the missing values are replaced with 7. The sequence that results from using a forward fill is [2, 4, 7, 7, 7, 7, 8].
- Backward fill – Use a backward fill to replace the missing values with the non-missing value that follows the missing values. For the sequence: [2, 4, 7, NaN, NaN, NaN, 8], all of the missing values are replaced with 8. The sequence that results from using a backward fill is [2, 4, 7, 8, 8, 8, 8].
- Interpolate – Uses an interpolation function to impute the missing values. For more information on the functions that you can use for interpolation, see [pandas.DataFrame.interpolate](#).

Some of the imputation methods might not be able to impute all the missing value in your dataset. For example, a **Forward fill** can't impute a missing value that appears at the beginning of the time series. You can impute the values by using either a forward fill or a backward fill.

You can either impute missing values within a cell or within a column.

The following example shows how values are imputed within a cell.

Electricity usage with missing values

Household ID	Electricity usage series (kWh)
household_0	[30, 40, 35, NaN, NaN]
household_1	[45, NaN, 55]

Electricity usage with values imputed using a forward fill

Household ID	Electricity usage series (kWh)
household_0	[30, 40, 35, 35, 35]
household_1	[45, 45, 55]

The following example shows how values are imputed within a column.

Average daily household electricity usage with missing values

Household ID	Electricity usage (kWh)
household_0	30
household_0	40
household_0	NaN
household_1	NaN
household_1	NaN

Average daily household electricity usage with values imputed using a forward fill

Household ID	Electricity usage (kWh)
household_0	30
household_0	40
household_0	40
household_1	40
household_1	40

You can use the following procedure to handle missing values.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the **+**, and select **Add transform**.
3. Choose **Add step**.
4. Choose **Handle missing**.
5. For **Time series input type**, choose whether you want to handle missing values inside of a cell or along a column.
6. For **Impute missing values for this column**, specify the column that has the missing values.
7. For **Method for imputing values**, select a method.
8. Configure the transform by specifying the remaining fields.
9. Choose **Preview** to generate a preview of the transform.
10. If you have missing values, you can specify a method for imputing them under **Method for imputing values**.
11. Choose **Add** to add the transform to the Data Wrangler data flow.

## Validate the Timestamp of Your Time Series Data

You might have time stamp data that is invalid. You can use the **Validate time stamp** function to determine whether the timestamps in your dataset are valid. Your timestamp can be invalid for one or more of the following reasons:

- Your timestamp column has missing values.
- The values in your timestamp column are not formatted correctly.

If you have invalid timestamps in your dataset, you can't perform your analysis successfully. You can use Data Wrangler to identify invalid timestamps and understand where you need to clean your data.

The time series validation works in one of the two ways:

You can configure Data Wrangler to do one of the following if it encounters missing values in your dataset:

- Drop the rows that have the missing or invalid values.
- Identify the rows that have the missing or invalid values.
- Throw an error if it finds any missing or invalid values in your dataset.

You can validate the timestamps on columns that either have the `timestamp` type or the `string` type. If the column has the `string` type, Data Wrangler converts the type of the column to `timestamp` and performs the validation.

You can use the following procedure to validate the timestamps in your dataset.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the **+**, and select **Add transform**.
3. Choose **Add step**.
4. Choose **Validate timestamps**.
5. For **Timestamp Column**, choose the timestamp column.
6. For **Policy**, choose whether you want to handle missing timestamps.
7. (Optional) For **Output column**, specify a name for the output column.
8. If the date time column is formatted for the string type, choose **Cast to datetime**.
9. Choose **Preview** to generate a preview of the transform.
10. Choose **Add** to add the transform to the Data Wrangler data flow.

## Standardizing the Length of the Time Series

If you have time series data stored as arrays, you can standardize each time series to the same length. Standardizing the length of the time series array might make it easier for you to perform your analysis on the data.

You can standardize your time series for data transformations that require the length of your data to be fixed.

Many ML algorithms require you to flatten your time series data before you use them. Flattening time series data is separating each value of the time series into its own column in a dataset. The number of columns in a dataset can't change, so the lengths of the time series need to be standardized between you flatten each array into a set of features.

Each time series is set to the length that you specify as a quantile or percentile of the time series set. For example, you can have three sequences that have the following lengths:

- 3
- 4
- 5

You can set the length of all of the sequences as the length of the sequence that has the 50th percentile length.

Time series arrays that are shorter than the length you've specified have missing values added. The following is an example format of standardizing the time series to a longer length: [2, 4, 5, NaN, NaN, NaN].

You can use different approaches to handle the missing values. For information on those approaches, see [Handle Missing Time Series Data](#).

The time series arrays that are longer than the length that you specify are truncated.

You can use the following procedure to standardize the length of the time series.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the +, and select **Add transform**.
3. Choose **Add step**.

4. Choose **Standardize length**.
5. For **Standardize the time series length for the column**, choose a column.
6. (Optional) For **Output column**, specify a name for the output column. If you don't specify a name, the transform is done in place.
7. If the datetime column is formatted for the string type, choose **Cast to datetime**.
8. Choose **Cutoff quantile** and specify a quantile to set the length of the sequence.
9. Choose **Flatten the output** to output the values of the time series into separate columns.
10. Choose **Preview** to generate a preview of the transform.
11. Choose **Add** to add the transform to the Data Wrangler data flow.

## Extract Features from Your Time Series Data

If you're running a classification or a regression algorithm on your time series data, we recommend extracting features from the time series before running the algorithm. Extracting features might improve the performance of your algorithm.

Use the following options to choose how you want to extract features from your data:

- Use **Minimal subset** to specify extracting 8 features that you know are useful in downstream analyses. You can use a minimal subset when you need to perform computations quickly. You can also use it when your ML algorithm has a high risk of overfitting and you want to provide it with fewer features.
- Use **Efficient subset** to specify extracting the most features possible without extracting features that are computationally intensive in your analyses.
- Use **All features** to specify extracting all features from the time series.
- Use **Manual subset** to choose a list of features that you think explain the variation in your data well.

Use the following procedure to extract features from your time series data.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the **+**, and select **Add transform**.
3. Choose **Add step**.
4. Choose **Extract features**.

5. For **Extract features for this column**, choose a column.
6. (Optional) Select **Flatten** to output the features into separate columns.
7. For **Strategy**, choose a strategy to extract the features.
8. Choose **Preview** to generate a preview of the transform.
9. Choose **Add** to add the transform to the Data Wrangler data flow.

## Use Lagged Features from Your Time Series Data

For many use cases, the best way to predict the future behavior of your time series is to use its most recent behavior.

The most common uses of lagged features are the following:

- Collecting a handful of past values. For example, for time,  $t + 1$ , you collect  $t$ ,  $t - 1$ ,  $t - 2$ , and  $t - 3$ .
- Collecting values that correspond to seasonal behavior in the data. For example, to predict the occupancy in a restaurant at 1:00 PM, you might want to use the features from 1:00 PM on the previous day. Using the features from 12:00 PM or 11:00 AM on the same day might not be as predictive as using the features from previous days.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the **+**, and select **Add transform**.
3. Choose **Add step**.
4. Choose **Lag features**.
5. For **Generate lag features for this column**, choose a column.
6. For **Timestamp Column**, choose the column containing the timestamps.
7. For **Lag**, specify the duration of the lag.
8. (Optional) Configure the output using one of the following options:
  - **Include the entire lag window**
  - **Flatten the output**
  - **Drop rows without history**
9. Choose **Preview** to generate a preview of the transform.
10. Choose **Add** to add the transform to the Data Wrangler data flow.

## Create a Datetime Range In Your Time Series

You might have time series data that don't have timestamps. If you know that the observations were taken at regular intervals, you can generate timestamps for the time series in a separate column. To generate timestamps, you specify the value for the start timestamp and the frequency of the timestamps.

For example, you might have the following time series data for the number of customers at a restaurant.

Time series data on the number of customers at a restaurant

Number of customers
10
14
24
40
30
20

If you know that the restaurant opened at 5:00 PM and that the observations are taken hourly, you can add a timestamp column that corresponds to the time series data. You can see the timestamp column in the following table.

Time series data on the number of customers at a restaurant

Number of customers	Timestamp
10	1:00 PM
14	2:00 PM
24	3:00 PM

Number of customers	Timestamp
40	4:00 PM
30	5:00 PM
20	6:00 PM

Use the following procedure to add a datetime range to your data.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the **+**, and select **Add transform**.
3. Choose **Add step**.
4. Choose **Datetime range**.
5. For **Frequency type**, choose the unit used to measure the frequency of the timestamps.
6. For **Starting timestamp**, specify the start timestamp.
7. For **Output column**, specify a name for the output column.
8. (Optional) Configure the output using the remaining fields.
9. Choose **Preview** to generate a preview of the transform.
10. Choose **Add** to add the transform to the Data Wrangler data flow.

## Use a Rolling Window In Your Time Series

You can extract features over a time period. For example, for time,  $t$ , and a time window length of 3, and for the row that indicates the  $t$ th timestamp, we append the features that are extracted from the time series at times  $t - 3$ ,  $t - 2$ , and  $t - 1$ . For information on extracting features, see [Extract Features from Your Time Series Data](#).

You can use the following procedure to extract features over a time period.

1. Open your Data Wrangler data flow.
2. In your data flow, under **Data types**, choose the **+**, and select **Add transform**.
3. Choose **Add step**.
4. Choose **Rolling window features**.
5. For **Generate rolling window features for this column**, choose a column.

6. For **Timestamp Column**, choose the column containing the timestamps.
7. (Optional) For **Output Column**, specify the name of the output column.
8. For **Window size**, specify the window size.
9. For **Strategy**, choose the extraction strategy.
10. Choose **Preview** to generate a preview of the transform.
11. Choose **Add** to add the transform to the Data Wrangler data flow.

## Featurize Datetime

Use **Featurize date/time** to create a vector embedding representing a datetime field. To use this transform, your datetime data must be in one of the following formats:

- Strings describing datetime: For example, "January 1st, 2020, 12:44pm".
- A Unix timestamp: A Unix timestamp describes the number of seconds, milliseconds, microseconds, or nanoseconds from 1/1/1970.

You can choose to **Infer datetime format** and provide a **Datetime format**. If you provide a datetime format, you must use the codes described in the [Python documentation](#). The options you select for these two configurations have implications for the speed of the operation and the final results.

- The most manual and computationally fastest option is to specify a **Datetime format** and select **No** for **Infer datetime format**.
- To reduce manual labor, you can choose **Infer datetime format** and not specify a datetime format. It is also a computationally fast operation; however, the first datetime format encountered in the input column is assumed to be the format for the entire column. If there are other formats in the column, these values are NaN in the final output. Inferring the datetime format can give you unparsed strings.
- If you don't specify a format and select **No** for **Infer datetime format**, you get the most robust results. All the valid datetime strings are parsed. However, this operation can be an order of magnitude slower than the first two options in this list.

When you use this transform, you specify an **Input column** which contains datetime data in one of the formats listed above. The transform creates an output column named **Output column name**. The format of the output column depends on your configuration using the following:

- **Vector:** Outputs a single column as a vector.
- **Columns:** Creates a new column for every feature. For example, if the output contains a year, month, and day, three separate columns are created for year, month, and day.

Additionally, you must choose an **Embedding mode**. For linear models and deep networks, we recommend choosing **cyclic**. For tree-based algorithms, we recommend choosing **ordinal**.

## Format String

The **Format string** transforms contain standard string formatting operations. For example, you can use these operations to remove special characters, normalize string lengths, and update string casing.

This feature group contains the following transforms. All transforms return copies of the strings in the **Input column** and add the result to a new, output column.

Name	Function
Left pad	Left-pad the string with a given <b>Fill character</b> to the given <b>width</b> . If the string is longer than <b>width</b> , the return value is shortened to <b>width</b> characters.
Right pad	Right-pad the string with a given <b>Fill character</b> to the given <b>width</b> . If the string is longer than <b>width</b> , the return value is shortened to <b>width</b> characters.
Center (pad on either side)	Center-pad the string (add padding on both sides of the string) with a given <b>Fill character</b> to the given <b>width</b> . If the string is longer than <b>width</b> , the return value is shortened to <b>width</b> characters.
Prepend zeros	Left-fill a numeric string with zeros, up to a given <b>width</b> . If the string is longer than <b>width</b> , the return value is shortened to <b>width</b> characters.

Name	Function
Strip left and right	Returns a copy of the string with the leading and trailing characters removed.
Strip characters from left	Returns a copy of the string with leading characters removed.
Strip characters from right	Returns a copy of the string with trailing characters removed.
Lower case	Convert all letters in text to lowercase.
Upper case	Convert all letters in text to uppercase.
Capitalize	Capitalize the first letter in each sentence.
Swap case	Converts all uppercase characters to lowercase and all lowercase characters to uppercase characters of the given string, and returns it.
Add prefix or suffix	Adds a prefix and a suffix to the string column. You must specify at least one of <b>Prefix</b> and <b>Suffix</b> .
Remove symbols	Removes given symbols from a string. All listed characters are removed. Defaults to white space.

## Handle Outliers

Machine learning models are sensitive to the distribution and range of your feature values. Outliers, or rare values, can negatively impact model accuracy and lead to longer training times. Use this feature group to detect and update outliers in your dataset.

When you define a **Handle outliers** transform step, the statistics used to detect outliers are generated on the data available in Data Wrangler when defining this step. These same statistics are used when running a Data Wrangler job.

Use the following sections to learn more about the transforms this group contains. You specify an **Output name** and each of these transforms produces an output column with the resulting data.

### Robust standard deviation numeric outliers

This transform detects and fixes outliers in numeric features using statistics that are robust to outliers.

You must define an **Upper quantile** and a **Lower quantile** for the statistics used to calculate outliers. You must also specify the number of **Standard deviations** from which a value must vary from the mean to be considered an outlier. For example, if you specify 3 for **Standard deviations**, a value must fall more than 3 standard deviations from the mean to be considered an outlier.

The **Fix method** is the method used to handle outliers when they are detected. You can choose from the following:

- **Clip**: Use this option to clip the outliers to the corresponding outlier detection bound.
- **Remove**: Use this option to remove rows with outliers from the dataframe.
- **Invalidate**: Use this option to replace outliers with invalid values.

### Standard Deviation Numeric Outliers

This transform detects and fixes outliers in numeric features using the mean and standard deviation.

You specify the number of **Standard deviations** a value must vary from the mean to be considered an outlier. For example, if you specify 3 for **Standard deviations**, a value must fall more than 3 standard deviations from the mean to be considered an outlier.

The **Fix method** is the method used to handle outliers when they are detected. You can choose from the following:

- **Clip**: Use this option to clip the outliers to the corresponding outlier detection bound.
- **Remove**: Use this option to remove rows with outliers from the dataframe.
- **Invalidate**: Use this option to replace outliers with invalid values.

## Quantile Numeric Outliers

Use this transform to detect and fix outliers in numeric features using quantiles. You can define an **Upper quantile** and a **Lower quantile**. All values that fall above the upper quantile or below the lower quantile are considered outliers.

The **Fix method** is the method used to handle outliers when they are detected. You can choose from the following:

- **Clip**: Use this option to clip the outliers to the corresponding outlier detection bound.
- **Remove**: Use this option to remove rows with outliers from the dataframe.
- **Invalidate**: Use this option to replace outliers with invalid values.

## Min-Max Numeric Outliers

This transform detects and fixes outliers in numeric features using upper and lower thresholds. Use this method if you know threshold values that demark outliers.

You specify a **Upper threshold** and a **Lower threshold**, and if values fall above or below those thresholds respectively, they are considered outliers.

The **Fix method** is the method used to handle outliers when they are detected. You can choose from the following:

- **Clip**: Use this option to clip the outliers to the corresponding outlier detection bound.
- **Remove**: Use this option to remove rows with outliers from the dataframe.
- **Invalidate**: Use this option to replace outliers with invalid values.

## Replace Rare

When you use the **Replace rare** transform, you specify a threshold and Data Wrangler finds all values that meet that threshold and replaces them with a string that you specify. For example, you may want to use this transform to categorize all outliers in a column into an "Others" category.

- **Replacement string**: The string with which to replace outliers.
- **Absolute threshold**: A category is rare if the number of instances is less than or equal to this absolute threshold.

- **Fraction threshold:** A category is rare if the number of instances is less than or equal to this fraction threshold multiplied by the number of rows.
- **Max common categories:** Maximum not-rare categories that remain after the operation. If the threshold does not filter enough categories, those with the top number of appearances are classified as not rare. If set to 0 (default), there is no hard limit to the number of categories.

## Handle Missing Values

Missing values are a common occurrence in machine learning datasets. In some situations, it is appropriate to impute missing data with a calculated value, such as an average or categorically common value. You can process missing values using the **Handle missing values** transform group. This group contains the following transforms.

### Fill Missing

Use the **Fill missing** transform to replace missing values with a **Fill value** you define.

### Impute Missing

Use the **Impute missing** transform to create a new column that contains imputed values where missing values were found in input categorical and numerical data. The configuration depends on your data type.

For numeric data, choose an imputing strategy, the strategy used to determine the new value to impute. You can choose to impute the mean or the median over the values that are present in your dataset. Data Wrangler uses the value that it computes to impute the missing values.

For categorical data, Data Wrangler imputes missing values using the most frequent value in the column. To impute a custom string, use the **Fill missing** transform instead.

### Add Indicator for Missing

Use the **Add indicator for missing** transform to create a new indicator column, which contains a Boolean "false" if a row contains a value, and "true" if a row contains a missing value.

### Drop Missing

Use the **Drop missing** option to drop rows that contain missing values from the **Input column**.

### Manage Columns

You can use the following transforms to quickly update and manage columns in your dataset:

Name	Function
Drop Column	Delete a column.
Duplicate Column	Duplicate a column.
Rename Column	Rename a column.
Move Column	Move a column's location in the dataset. Choose to move your column to the start or end of the dataset, before or after a reference column, or to a specific index.

## Manage Rows

Use this transform group to quickly perform sort and shuffle operations on rows. This group contains the following:

- **Sort:** Sort the entire dataframe by a given column. Select the check box next to **Ascending order** for this option; otherwise, deselect the check box and descending order is used for the sort.
- **Shuffle:** Randomly shuffle all rows in the dataset.

## Manage Vectors

Use this transform group to combine or flatten vector columns. This group contains the following transforms.

- **Assemble:** Use this transform to combine Spark vectors and numeric data into a single column. For example, you can combine three columns: two containing numeric data and one containing vectors. Add all the columns you want to combine in **Input columns** and specify a **Output column name** for the combined data.
- **Flatten:** Use this transform to flatten a single column containing vector data. The input column must contain PySpark vectors or array-like objects. You can control the number of columns created by specifying a **Method to detect number of outputs**. For example, if you select **Length of first vector**, the number of elements in the first valid vector or array found in the column determines the number of output columns that are created. All other input vectors with too many items are truncated. Inputs with too few items are filled with NaNs.

You also specify an **Output prefix**, which is used as the prefix for each output column.

## Process Numeric

Use the **Process Numeric** feature group to process numeric data. Each scalar in this group is defined using the Spark library. The following scalars are supported:

- **Standard Scaler:** Standardize the input column by subtracting the mean from each value and scaling to unit variance. To learn more, see the Spark documentation for [StandardScaler](#).
- **Robust Scaler:** Scale the input column using statistics that are robust to outliers. To learn more, see the Spark documentation for [RobustScaler](#).
- **Min Max Scaler:** Transform the input column by scaling each feature to a given range. To learn more, see the Spark documentation for [MinMaxScaler](#).
- **Max Absolute Scaler:** Scale the input column by dividing each value by the maximum absolute value. To learn more, see the Spark documentation for [MaxAbsScaler](#).

## Sampling

After you've imported your data, you can use the **Sampling** transformer to take one or more samples of it. When you use the sampling transformer, Data Wrangler samples your original dataset.

You can choose one of the following sample methods:

- **Limit:** Samples the dataset starting from the first row up to the limit that you specify.
- **Randomized:** Takes a random sample of a size that you specify.
- **Stratified:** Takes a stratified random sample.

You can stratify a randomized sample to make sure that it represents the original distribution of the dataset.

You might be performing data preparation for multiple use cases. For each use case, you can take a different sample and apply a different set of transformations.

The following procedure describes the process of creating a random sample.

To take a random sample from your data.

1. Choose the **+** to the right of the dataset that you've imported. The name of your dataset is located below the **+**.
2. Choose **Add transform**.
3. Choose **Sampling**.
4. For **Sampling method**, choose the sampling method.
5. For **Approximate sample size**, choose the approximate number of observations that you want in your sample.
6. (Optional) Specify an integer for **Random seed** to create a reproducible sample.

The following procedure describes the process of creating a stratified sample.

To take a stratified sample from your data.

1. Choose the **+** to the right of the dataset that you've imported. The name of your dataset is located below the **+**.
2. Choose **Add transform**.
3. Choose **Sampling**.
4. For **Sampling method**, choose the sampling method.
5. For **Approximate sample size**, choose the approximate number of observations that you want in your sample.
6. For **Stratify column**, specify the name of the column that you want to stratify on.
7. (Optional) Specify an integer for **Random seed** to create a reproducible sample.

## Search and Edit

Use this section to search for and edit specific patterns within strings. For example, you can find and update strings within sentences or documents, split strings by delimiters, and find occurrences of specific strings.

The following transforms are supported under **Search and edit**. All transforms return copies of the strings in the **Input column** and add the result to a new output column.

Name	Function
Find substring	Returns the index of the first occurrence of the <b>Substring</b> for which you searched , You can start and end the search at <b>Start</b> and <b>End</b> respectively.
Find substring (from right)	Returns the index of the last occurrence of the <b>Substring</b> for which you searched. You can start and end the search at <b>Start</b> and <b>End</b> respectively.
Matches prefix	Returns a Boolean value if the string contains a given <b>Pattern</b> . A pattern can be a character sequence or regular expression. Optionally, you can make the pattern case sensitive.
Find all occurrences	Returns an array with all occurrences of a given pattern. A pattern can be a character sequence or regular expression.
Extract using regex	Returns a string that matches a given Regex pattern.
Extract between delimiters	Returns a string with all characters found between <b>Left delimiter</b> and <b>Right delimiter</b> .
Extract from position	Returns a string, starting from <b>Start position</b> in the input string, that contains all characters up to the start position plus <b>Length</b> .
Find and replace substring	Returns a string with all matches of a given <b>Pattern</b> (regular expression) replaced by <b>Replacement string</b> .
Replace between delimiters	Returns a string with the substring found between the first appearance of a <b>Left delimiter</b> and the last appearance of a <b>Right</b>

Name	Function
	<b>delimiter</b> replaced by <b>Replacement string</b> . If no match is found, nothing is replaced.
Replace from position	Returns a string with the substring between <b>Start position</b> and <b>Start position plus Length</b> replaced by <b>Replacement string</b> . If <b>Start position plus Length</b> is greater than the length of the replacement string, the output contains ....
Convert regex to missing	Converts a string to None if invalid and returns the result. Validity is defined with a regular expression in <b>Pattern</b> .
Split string by delimiter	Returns an array of strings from the input string, split by <b>Delimiter</b> , with up to <b>Max number of splits</b> (optional). The delimiter defaults to white space.

## Split data

Use the **Split data** transform to split your dataset into two or three datasets. For example, you can split your dataset into a dataset used to train your model and a dataset used to test it. You can determine the proportion of the dataset that goes into each split. For example, if you're splitting one dataset into two datasets, the training dataset can have 80% of the data while the testing dataset has 20%.

Splitting your data into three datasets gives you the ability to create training, validation, and test datasets. You can see how well the model performs on the test dataset by dropping the target column.

Your use case determines how much of the original dataset each of your datasets get and the method you use to split the data. For example, you might want to use a stratified split to make sure that the distribution of the observations in the target column are the same across datasets. You can use the following split transforms:

- Randomized split — Each split is a random, non-overlapping sample of the original dataset. For larger datasets, using a randomized split might be computationally expensive and take longer than an ordered split.
- Ordered split – Splits the dataset based on the sequential order of the observations. For example, for an 80/20 train-test split, the first observations that make up 80% of the dataset go to the training dataset. The last 20% of the observations go to the testing dataset. Ordered splits are effective in keeping the existing order of the data between splits.
- Stratified split – Splits the dataset to make sure that the number of observations in the input column have proportional representation. For an input column that has the observations 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, an 80/20 split on the column would mean that approximately 80% of the 1s, 80% of the 2s, and 80% of the 3s go to the training set. About 20% of each type of observation go to the testing set.
- Split by key – Avoids data with the same key occurring in more than one split. For example, if you have a dataset with the column 'customer\_id' and you're using it as a key, no customer id is in more than one split.

After you split the data, you can apply additional transformations to each dataset. For most use cases, they aren't necessary.

Data Wrangler calculates the proportions of the splits for performance. You can choose an error threshold to set the accuracy of the splits. Lower error thresholds more accurately reflect the proportions that you specify for the splits. If you set a higher error threshold, you get better performance, but lower accuracy.

For perfectly split data, set the error threshold to 0. You can specify a threshold between 0 and 1 for better performance. If you specify a value greater than 1, Data Wrangler interprets that value as 1.

If you have 10000 rows in your dataset and you specify an 80/20 split with an error of 0.001, you would get observations approximating one of the following results:

- 8010 observations in the training set and 1990 in the testing set
- 7990 observations in the training set and 2010 in the testing set

The number of observations for the testing set in the preceding example is in the interval between 8010 and 7990.

By default, Data Wrangler uses a random seed to make the splits reproducible. You can specify a different value for the seed to create a different reproducible split.

## Randomized split

Use the following procedure to perform a randomized split on your dataset.

To split your dataset randomly, do the following

1. Choose the **+** next to the node containing the dataset that you're splitting.
2. Choose **Add transform**.
3. Choose **Split data**.
4. (Optional) For **Splits**, specify the names and proportions of each split. The proportions must sum to 1.
5. (Optional) Choose the **+** to create an additional split.
  - Specify the names and proportions of all the splits. The proportions must sum to 1.
6. (Optional) Specify a value for **Error threshold** other than the default value.
7. (Optional) Specify a value for **Random seed**.
8. Choose **Preview**.
9. Choose **Add**.

## Ordered split

Use the following procedure to perform an ordered split on your dataset.

To make an ordered split in your dataset, do the following.

1. Choose the **+** next to the node containing the dataset that you're splitting.
2. Choose **Add transform**.
3. For **Transform**, choose **Ordered split**.
4. Choose **Split data**.
5. (Optional) For **Splits**, specify the names and proportions of each split. The proportions must sum to 1.
6. (Optional) Choose the **+** to create an additional split.

- Specify the names and proportions of all the splits. The proportions must sum to 1.
7. (Optional) Specify a value for **Error threshold** other than the default value.
  8. (Optional) For **Input column**, specify a column with numeric values. Uses the values of the columns to infer which records are in each split. The smaller values are in one split with the larger values in the other splits.
  9. (Optional) Select **Handle duplicates** to add noise to duplicate values and create a dataset of entirely unique values.
  10. (Optional) Specify a value for **Random seed**.
  11. Choose **Preview**.
  12. Choose **Add**.

## Stratified split

Use the following procedure to perform a stratified split on your dataset.

To make a stratified split in your dataset, do the following.

1. Choose the + next to the node containing the dataset that you're splitting.
2. Choose **Add transform**.
3. Choose **Split data**.
4. For **Transform**, choose **Stratified split**.
5. (Optional) For **Splits**, specify the names and proportions of each split. The proportions must sum to 1.
6. (Optional) Choose the + to create an additional split.
  - Specify the names and proportions of all the splits. The proportions must sum to 1.
7. For **Input column**, specify a column with up to 100 unique values. Data Wrangler can't stratify a column with more than 100 unique values.
8. (Optional) Specify a value for **Error threshold** other than the default value.
9. (Optional) Specify a value for **Random seed** to specify a different seed.
10. Choose **Preview**.
11. Choose **Add**.

## Split by column keys

Use the following procedure to split by the column keys in your dataset.

To split by the column keys in your dataset, do the following.

1. Choose the + next to the node containing the dataset that you're splitting.
2. Choose **Add transform**.
3. Choose **Split data**.
4. For **Transform**, choose **Split by key**.
5. (Optional) For **Splits**, specify the names and proportions of each split. The proportions must sum to 1.
6. (Optional) Choose the + to create an additional split.
  - Specify the names and proportions of all the splits. The proportions must sum to 1.
7. For **Key columns**, specify the columns with values that you don't want to appear in both datasets.
8. (Optional) Specify a value for **Error threshold** other than the default value.
9. Choose **Preview**.
10. Choose **Add**.

## Parse Value as Type

Use this transform to cast a column to a new type. The supported Data Wrangler data types are:

- Long
- Float
- Boolean
- Date, in the format dd-MM-yyyy, representing day, month, and year respectively.
- String

## Validate String

Use the **Validate string** transforms to create a new column that indicates that a row of text data meets a specified condition. For example, you can use a **Validate string** transform to verify that a

string only contains lowercase characters. The following transforms are supported under **Validate string**.

The following transforms are included in this transform group. If a transform outputs a Boolean value, True is represented with a 1 and False is represented with a 0.

Name	Function
String length	Returns True if a string length equals specified length. Otherwise, returns False.
Starts with	Returns True if a string starts will a specified prefix. Otherwise, returns False.
Ends with	Returns True if a string length equals specified length. Otherwise, returns False.
Is alphanumeric	Returns True if a string only contains numbers and letters. Otherwise, returns False.
Is alpha (letters)	Returns True if a string only contains letters. Otherwise, returns False.
Is digit	Returns True if a string only contains digits. Otherwise, returns False.
Is space	Returns True if a string only contains numbers and letters. Otherwise, returns False.
Is title	Returns True if a string contains any white spaces. Otherwise, returns False.
Is lowercase	Returns True if a string only contains lower case letters. Otherwise, returns False.
Is uppercase	Returns True if a string only contains upper case letters. Otherwise, returns False.

Name	Function
Is numeric	Returns True if a string only contains numbers. Otherwise, returns False.
Is decimal	Returns True if a string only contains decimal numbers. Otherwise, returns False.

## Unnest JSON Data

If you have a .csv file, you might have values in your dataset that are JSON strings. Similarly, you might have nested data in columns of either a Parquet file or a JSON document.

Use the **Flatten structured** operator to separate the first level keys into separate columns. A first level key is a key that isn't nested within a value.

For example, you might have a dataset that has a *person* column with demographic information on each person stored as JSON strings. A JSON string might look like the following.

```
{"seq": 1,"name": {"first": "Nathaniel","last": "Ferguson"},"age": 59,"city": "Posbotno","state": "WV"}
```

The **Flatten structured** operator converts the following first level keys into additional columns in your dataset:

- seq
- name
- age
- city
- state

Data Wrangler puts the values of the keys as values under the columns. The following shows the column names and values of the JSON.

```
seq, name, age, city, state  
1, {"first": "Nathaniel", "last": "Ferguson"}, 59, Posbotno, WV
```

For each value in your dataset containing JSON, the **Flatten structured** operator creates columns for the first-level keys. To create columns for nested keys, call the operator again. For the preceding example, calling the operator creates the columns:

- name\_first
- name\_last

The following example shows the dataset that results from calling the operation again.

```
seq, name, age, city, state, name_first, name_last  
1, {"first": "Nathaniel", "last": "Ferguson"}, 59, Posbotno, WV, Nathaniel, Ferguson
```

Choose **Keys to flatten on** to specify the first-level keys that want to extract as separate columns. If you don't specify any keys, Data Wrangler extracts all the keys by default.

## Explode Array

Use **Explode array** to expand the values of the array into separate output rows. For example, the operation can take each value in the array, [[1, 2, 3], [4, 5, 6], [7, 8, 9]] and create a new column with the following rows:

```
[1, 2, 3]  
[4, 5, 6]  
[7, 8, 9]
```

Data Wrangler names the new column, input\_column\_name\_flatten.

You can call the **Explode array** operation multiple times to get the nested values of the array into separate output columns. The following example shows the result of calling the operation multiple times on a dataset with a nested array.

Putting the values of a nested array into separate columns

<b>id</b>	<b>array</b>	<b>id</b>	<b>array_items</b>	<b>id</b>	<b>array_items_items</b>
1	[ [cat, dog], [bat, frog] ]	1	[cat, dog]	1	cat
2	[[rose, petunia], [lily, daisy]]	1	[bat, frog]	1	dog
		2	[rose, petunia]	1	bat
		2	[lily, daisy]	1	frog
			2	2	rose
			2	2	petunia
			2	2	lily
			2	2	daisy

## Transform Image Data

Use Data Wrangler to import and transform the images that you're using for your machine learning (ML) pipelines. After you've prepared your image data, you can export it from your Data Wrangler flow to your ML pipeline.

You can use the information provided here to familiarize yourself with importing and transforming image data in Data Wrangler. Data Wrangler uses OpenCV to import images. For more information about supported image formats, see [Image file reading and writing](#).

After you've familiarized yourself with the concepts of transforming your image data, go through the following tutorial, [Prepare image data with Amazon SageMaker Data Wrangler](#).

The following industries and use cases are examples where applying machine learning to transformed image data can be useful:

- Manufacturing – Identifying defects in items from the assembly line

- Food – Identifying spoiled or rotten food
- Medicine – Identifying lesions in tissues

When you work with image data in Data Wrangler, you go through the following process:

1. Import – Select the images by choosing the directory containing them in your Amazon S3 bucket.
2. Transform – Use the built-in transformations to prepare the images for your machine learning pipeline.
3. Export – Export the images that you've transformed to a location that can be accessed from the pipeline.

Use the following procedure to import your image data.

### To import your image data

1. Navigate to the **Create connection** page.
2. Choose **Amazon S3**.
3. Specify the Amazon S3 file path that contains the image data.
4. For **File type**, choose **Image**.
5. (Optional) Choose **Import nested directories** to import images from multiple Amazon S3 paths.
6. Choose **Import**.

Data Wrangler uses the open-source [imgaug](#) library for its built-in image transformations. You can use the following built-in transformations:

- **ResizeImage**
- **EnhanceImage**
- **CorruptImage**
- **SplitImage**
- **DropCorruptedImages**
- **DropImageDuplicates**

- **Brightness**
- **ColorChannels**
- **Grayscale**
- **Rotate**

Use the following procedure to transform your images without writing code.

### To transform the image data without writing code

1. From your Data Wrangler flow, choose the **+** next to the node representing the images that you've imported.
2. Choose **Add transform**.
3. Choose **Add step**.
4. Choose the transform and configure it.
5. Choose **Preview**.
6. Choose **Add**.

In addition to using the transformations that Data Wrangler provides, you can also use your own custom code snippets. For more information about using custom code snippets, see [Custom Transforms](#). You can import the OpenCV and imgaug libraries within your code snippets and use the transforms associated with them. The following is an example of a code snippet that detects edges within the images.

```
# A table with your image data is stored in the `df` variable
import cv2
import numpy as np
from pyspark.sql.functions import column

from sagemaker_dataprep.compute.operators.transforms.image.constants import
    DEFAULT_IMAGE_COLUMN, IMAGE_COLUMN_TYPE
from sagemaker_dataprep.compute.operators.transforms.image.decorators import
    BasicImageOperationDecorator, PandasUDFOperationDecorator

@BasicImageOperationDecorator
def my_transform(image: np.ndarray) -> np.ndarray:
```

```
# To use the code snippet on your image data, modify the following lines within the
# function
HYST_THRLD_1, HYST_THRLD_2 = 100, 200
edges = cv2.Canny(image,HYST_THRLD_1,HYST_THRLD_2)
return edges

@PandasUDFOperationDecorator(IMAGE_COLUMN_TYPE)
def custom_image_udf(image_row):
    return my_transform(image_row)

df = df.withColumn(DEFAULT_IMAGE_COLUMN,
custom_image_udf(column(DEFAULT_IMAGE_COLUMN)))
```

When you apply transformations in your Data Wrangler flow, Data Wrangler only applies them to a sample of the images in your dataset. To optimize your experience with the application, Data Wrangler doesn't apply the transforms to all of your images.

## Filter data

Use Data Wrangler to filter the data in your columns. When you filter the data in a column, you specify the following fields:

- **Column name** – The name of the column that you're using to filter the data.
- **Condition** – The type of filter that you're applying to values in the column.
- **Value** – The value or category in the column to which you're applying the filter.

You can filter on the following conditions:

- **=** – Returns values that match the value or category that you specify.
- **!=** – Returns values that don't match the value or category that you specify.
- **>=** – For **Long** or **Float** data, filters for values that are greater than or equal to the value that you specify.
- **<=** – For **Long** or **Float** data, filters for values that are less than or equal to the value that you specify.
- **>** – For **Long** or **Float** data, filters for values that are greater than the value that you specify.

- < – For **Long** or **Float** data, filters for values that are less than the value that you specify.

For a column that has the categories, male and female, you can filter out all the male values. You could also filter for all the female values. Because there are only male and female values in the column, the filter returns a column that only has female values.

You can also add multiple filters. The filters can be applied across multiple columns or the same column. For example, if you're creating a column that only has values within a certain range, you add two different filters. One filter specifies that the column must have values greater than the value that you provide. The other filter specifies that the column must have values less than the value that you provide.

Use the following procedure to add the filter transform to your data.

### To filter your data

1. From your Data Wrangler flow, choose the + next to the node with the data that you're filtering.
2. Choose **Add transform**.
3. Choose **Add step**.
4. Choose **Filter data**.
5. Specify the following fields:
  - **Column name** – The column that you're filtering.
  - **Condition** – The condition of the filter.
  - **Value** – The value or category in the column to which you're applying the filter.
6. (Optional) Choose + following the filter that you've created.
7. Configure the filter.
8. Choose **Preview**.
9. Choose **Add**.

### Chat for data prep

#### **Important**

For administrators:

- Chat for data prep requires the `AmazonSageMakerCanvasAISServicesAccess` policy. For more information, see [AWS managed policy: AmazonSageMakerCanvasAISServicesAccess](#)
- Chat for data prep requires access to Amazon Bedrock and the **Anthropic Claude** model within it. For more information, see [Add model access](#).
- You must run SageMaker Canvas data prep in the same AWS Region as the Region where you're running your model. Chat for data prep is available in the US East (N. Virginia), US West (Oregon), and Europe (Frankfurt) AWS Regions.

In addition to using the built-in transforms and analyses, you can use natural language to explore, visualize, and transform your data in a conversational interface. Within the conversational interface, you can use natural language queries to understand and prepare your data to build ML models.

The following are examples of some prompts that you can use:

- Summarize my data
- Drop column *example-column-name*
- Replace missing values with median
- Plot histogram of prices
- What is the most expensive item sold?
- How many distinct items were sold?
- Sort data by region

When you're transforming your data using your prompts, you can view a preview that shows how data is being transformed. You can choose to add it as step in your Data Wrangler flow based on what you see in the preview.

The responses to your prompts generate code for your transformations and analyses. You can modify the code to update the output from the prompt. For example, you can modify the code for an analysis to change the values of the axes of a graph.

Use the following procedure to start chatting with your data:

## To chat with your data

1. Open the SageMaker Canvas data flow.
2. Choose the speech bubble.

The screenshot shows the SageMaker Canvas interface with the "Data" tab selected. In the main area, there are three visualizations: a histogram for "ActualShippingDays" (long), a scatter plot for "ExpectedShippingDays" (long) vs "Carrier" (string), and a bar chart for "YShippingDistance". Below these, a text input field says "e.g. Help me understand my data with a summary" with a blue "▶" button. To the right, a sidebar titled "Steps" lists "1. S3 Source" and "2. Data types", which includes columns for "Column name" and "Type".

3. Specify a prompt.
4. (Optional) If an analysis has been generated by your query, choose **Add to analyses** to reference it for later.

The screenshot shows the SageMaker Canvas interface with the "Analyses" tab selected. It displays a scatter plot titled "plot total\_rooms vs median\_income" and a histogram for "total\_rooms". Below the plots, a "View code" button is visible. A sidebar titled "Steps" lists "1. S3 Source" and "2. Data types".

5. (Optional) If you've transformed your data using a prompt, do the following.
  - a. Choose **Preview** to view the results.
  - b. (Optional) Modify the code in the transform and choose **Update**.
  - c. (Optional) If you're happy with the results of the transform, choose **Add to steps** to add it to the steps panel on the right-hand navigation.

The screenshot shows the Data Wrangler interface for a data flow named 'canvas-data-prep.flow'. The current step is 'Step 3. Chat Transform: Remove population < 100'. The step details pane shows a transformation rule: 'remove rows where population is less than 100'. A note explains that this filters out rows where the population column is less than 100, keeping only rows with population greater than or equal to 100. Below this, there's a preview section with histograms for 'longitude (float)', 'latitude (float)', 'housing\_median\_age (float)', 'total\_rooms (float)', and 'total\_bedrooms (float)'. The histograms show the distribution of these variables. To the right, the 'Steps' panel shows the step has been added to the steps panel. The 'Steps' panel also includes a 'Name' field set to 'Chat Transform: Remove population < 100', a 'Required' dropdown set to 'Python (PySpark)', and an 'Example code snippet' section with the following code:

```
import pyspark.sql.functions as F
df = df.filter(F.col("population") >= 100)
```

After you've prepared your data using natural language, you can create a model using your transformed data. For more information about creating a model, see [How custom models work](#).

## How data processing works in Data Wrangler

While working with data interactively in an Amazon SageMaker Data Wrangler data flow, Amazon SageMaker Canvas only applies the transformations to a sample dataset for you to preview. After finishing your data flow in SageMaker Canvas, you can process all of your data and save it in a location that is suitable for your machine learning workflows.

There are several options for how to proceed after you've finished transforming your data in Data Wrangler:

- [Create a model](#). You can create a Canvas model, where you directly start creating a model with your prepared data. You can create a model either after processing your entire dataset, or by exporting just the sample data you worked with in Data Wrangler. Canvas saves your processed data (either the entire dataset or the sample data) as a Canvas dataset.

We recommend that you use your sample data for quick iterations, but that you use your entire data when you want to train your final model. When building tabular models, datasets larger than 5 GB are automatically downsampled to 5 GB, and for time series forecasting models, datasets larger than 30 GB are downsampled to 30 GB.

To learn more about creating a model, see [How custom models work](#).

- [Export the data](#). You can export your data for use in machine learning workflows. When you choose to export your data, you have several options:
  - You can save your data in the Canvas application as a dataset. For more information about the supported file types for Canvas datasets and additional requirements when importing data into Canvas, see [Create a dataset](#).
  - You can save your data to Amazon S3. Depending on the Canvas memory availability, your data is processed in the application and then exported to Amazon S3. If the size of your dataset exceeds what Canvas can process, then by default, Canvas uses an EMR Serverless job to scale to multiple compute instances, process your full dataset, and export it to Amazon S3. You can also manually configure a SageMaker Processing job to have more granular control over the compute resources used to process your data.
- [Export a data flow](#). You might want to save the code for your data flow so that you can modify or run your transformations outside of Canvas. Canvas provides you with the option to save your data flow transformations as Python code in a Jupyter notebook, which you can then export to Amazon S3 for use elsewhere in your machine learning workflows.

When you export your data from a data flow and save it either as a Canvas dataset or to Amazon S3, Canvas creates a new destination node in your data flow, which is a final node that shows you where your processed data is stored. You can add additional destination nodes to your flow if you'd like to perform multiple export operations. For example, you can export the data from different points in your data flow to only apply some of the transformations, or you can export transformed data to different Amazon S3 locations. For more information about how to add or edit a destination node, see [Add destination nodes](#) and [Edit a destination node](#).

For more information about setting up a schedule with Amazon EventBridge to automatically process and export your data on a schedule, see [Create a schedule to automatically process new data](#).

## Export to create a model

In just a few clicks from your data flow, you can export your transformed data and start creating an ML model in Canvas. Canvas saves your data as a Canvas dataset, and you're taken to the model build configuration page for a new model.

To create a Canvas model with your transformed data:

1. Navigate to your data flow.
2. Choose the ellipsis icon next to the node that you're exporting.
3. From the context menu, choose **Create model**.
4. In the **Export to create a model** side panel, enter a **Dataset name** for the new dataset.
5. Leave the **Process entire dataset** option selected to process and export your entire dataset before proceeding with building a model. Turn this option off to train your model using the interactive sample data you are working with in your data flow.
6. Enter a **Model name** to name the new model.
7. Select a **Problem type**, or the type of model that you want to build. For more information about the supported model types in SageMaker Canvas, see [How custom models work](#).
8. Select the **Target column**, or the value that you want the model to predict.
9. Choose **Export and create model**.

The **Build** tab for a new Canvas model should open, and you can finish configuring and training your model. For more information about how to build a model, see [Build a model](#).

## Export data

Export data to apply the transforms from your data flow to the full imported dataset. You can export any node in your data flow to the following locations:

- SageMaker Canvas dataset
- Amazon S3

If you want to train models in Canvas, you can export your full, transformed dataset as a Canvas dataset. If you want to use your transformed data in machine learning workflows external to SageMaker Canvas, you can export your dataset to Amazon S3.

## Export to a Canvas dataset

Use the following procedure to export a SageMaker Canvas dataset from a node in your data flow.

### To export a node in your flow as a SageMaker Canvas dataset

1. Navigate to your data flow.
2. Choose the ellipsis icon next to the node that you're exporting.
3. In the context menu, hover over **Export**, and then select **Export data to Canvas dataset**.
4. In the **Export to Canvas dataset** side panel, enter a **Dataset name** for the new dataset.
5. Leave the **Process entire dataset** option selected if you want SageMaker Canvas to process and save your full dataset. Turn this option off to only apply the transforms to the sample data you are working with in your data flow.
6. Choose **Export**.

You should now be able to go to the **Datasets** page of the Canvas application and see your new dataset.

## Export to Amazon S3

When exporting your data to Amazon S3, you can scale to transform and process data of any size. Canvas automatically processes your data locally if the application's memory can handle the size of your dataset. If your dataset size exceeds the local memory capacity of 5 GB, then Canvas initiates a remote job on your behalf to provision additional compute resources and process the data more quickly. By default, Canvas uses Amazon EMR Serverless to run these remote jobs. However, you can manually configure Canvas to use either EMR Serverless or a SageMaker Processing job with your own settings.

### Note

When running an EMR Serverless job, by default the job inherits the IAM role, KMS key settings, and tags of your Canvas application.

The following summarizes the options for remote jobs in Canvas:

- **EMR Serverless:** This is the default option that Canvas uses for remote jobs. EMR Serverless automatically provisions and scales compute resources to process your data so that you don't have to worry about choosing the right compute resources for your workload. For more information about EMR Serverless, see the [EMR Serverless User Guide](#).
- **SageMaker Processing:** SageMaker Processing jobs offer more advanced options and granular control over the compute resources used to process your data. For example, you can specify the type and count of the compute instances, configure the job in your own VPC and control network access, automate processing jobs, and more. For more information about automating processing jobs see [Create a schedule to automatically process new data](#). For more general information about SageMaker Processing jobs, see [Data transformation workloads with SageMaker Processing](#).

The following file types are supported when exporting to Amazon S3:

- CSV
- Parquet

To get started, review the following prerequisites.

### Prerequisites for EMR Serverless jobs

To create a remote job that uses EMR Serverless resources, you must have the necessary permissions. You can grant permissions either through the Amazon SageMaker AI domain or user profile settings, or you can manually configure your user's AWS IAM role. For instructions on how to grant users permissions to perform large data processing, see [Grant Users Permissions to Use Large Data across the ML Lifecycle](#).

If you don't want to configure these policies but still need to process large datasets through Data Wrangler, you can alternatively use a SageMaker Processing job.

Use the following procedures to export your data to Amazon S3. To configure a remote job, follow the optional advanced steps.

### To export a node in your flow to Amazon S3

1. Navigate to your data flow.

2. Choose the ellipsis icon next to the node that you're exporting.
3. In the context menu, hover over **Export**, and then select **Export data to Amazon S3**.
4. In the **Export to Amazon S3** side panel, you can change the **Dataset name** for the new dataset.
5. For the **S3 location**, enter the Amazon S3 location to which you want to export the dataset. You can enter the S3 URI, alias, or ARN of the S3 location or S3 access point. For more information access points, see [Managing data access with Amazon S3 access points](#) in the *Amazon S3 User Guide*.
6. (Optional) For the **Advanced settings**, specify values for the following fields:
  - a. **File type** – The file format of your exported data.
  - b. **Delimiter** – The delimiter used to separate values in the file.
  - c. **Compression** – The compression method used to reduce the file size.
  - d. **Number of partitions** – The number of dataset files that Canvas writes as the output of the job.
  - e. **Choose columns** – You can choose a subset of columns from the data to include in the partitions.
7. Leave the **Process entire dataset** option selected if you want Canvas to apply your data flow transforms to your entire dataset and export the result. If you deselect this option, Canvas only applies the transforms to the sample of your dataset used in the interactive Data Wrangler data flow.

 **Note**

If you only export a sample of your data, Canvas processes your data in the application and doesn't create a remote job for you.

8. Leave the **Auto job configuration** option selected if you want Canvas to automatically determine whether to run the job using Canvas application memory or an EMR Serverless job. If you deselect this option and manually configure your job, then you can choose to use either an EMR Serverless or a SageMaker Processing job. For instructions on how to configure an EMR Serverless or a SageMaker Processing job, see the section after this procedure before you export your data.
9. Choose **Export**.

The following procedures show how to manually configure the remote job settings for either EMR Serverless or SageMaker Processing when exporting your full dataset to Amazon S3.

## EMR Serverless

To configure an EMR Serverless job while exporting to Amazon S3, do the following:

1. In the **Export to Amazon S3** side panel, turn off the **Auto job configuration** option.
2. Select **EMR Serverless**.
3. For **Job name**, enter a name for your EMR Serverless job. The name can contain letters, numbers, hyphens, and underscores.
4. For **IAM role**, enter the user's IAM execution role. This role should have the required permissions to run EMR Serverless applications. For more information, see [Grant Users Permissions to Use Large Data across the ML Lifecycle](#).
5. (Optional) For **KMS key**, specify the key ID or ARN of an AWS KMS key to encrypt the job logs. If you don't enter a key, Canvas uses a default key for EMR Serverless.
6. (Optional) For **Monitoring configuration**, enter the name of an Amazon CloudWatch Logs log group to which you want to publish your logs.
7. (Optional) For **Tags**, add metadata tags to the EMR Serverless job consisting of key-value pairs. These tags can be used to categorize and search for jobs.
8. Choose **Export** to start the job.

## SageMaker Processing

To configure a SageMaker Processing job while exporting to Amazon S3, do the following:

1. In the **Export to Amazon S3** side panel, turn off the **Auto job configuration** option.
2. Select **SageMaker Processing**.
3. For **Job name**, enter a name for your SageMaker AI Processing job.
4. For **Instance type**, select the type of compute instance to run the processing job.
5. For **Instance count**, specify the number of compute instances to launch.
6. For **IAM role**, enter the user's IAM execution role. This role should have the required permissions for SageMaker AI to create and run processing jobs on your behalf. These permissions are granted if you have the [AmazonSageMakerFullAccess](#) policy attached to your IAM role.

7. For **Volume size**, enter the storage size in GB for the ML storage volume that is attached to each processing instance. Choose the size based on your expected input and output data size.
8. (Optional) For **Volume KMS key**, specify a KMS key to encrypt the storage volume. If you don't specify a key, the default Amazon EBS encryption key is used.
9. (Optional) For **KMS key**, specify a KMS key to encrypt input and output Amazon S3 data sources used by the processing job.
10. (Optional) For **Spark memory configuration**, do the following:
  - a. Enter **Driver memory in MB** for the Spark driver node that handles job coordination and scheduling.
  - b. Enter **Executor memory in MB** for the Spark executor nodes that run individual tasks in the job.
11. (Optional) For **Network configuration**, do the following:
  - a. For **Subnet configuration**, enter the IDs of the VPC subnets for the processing instances to be launched in. By default, the job uses the settings of your default VPC.
  - b. For **Security group configuration**, enter the IDs of the security groups to control inbound and outbound connectivity rules.
  - c. Turn on the **Enable inter-container traffic encryption** option to encrypt network communication between processing containers during the job.
12. (Optional) For **Associate schedules**, you can choose create an Amazon EventBridge schedule to have the processing job run on recurring intervals. Choose **Create new schedule** and fill out the dialog box. For more information about filling out this section and running processing jobs on a schedule, see [Create a schedule to automatically process new data](#).
13. (Optional) Add **Tags** as key-value pairs so that you can categorize and search for processing jobs.
14. Choose **Export** to start the processing job.

After exporting your data, you should find the fully processed dataset in the specified Amazon S3 location.

## Export a data flow

Exporting your data flow translates the operations that you've made in Data Wrangler and exports it into a Jupyter notebook of Python code that you can modify and run. This can be helpful for integrating the code for your data transformations into your machine learning pipelines.

You can choose any data node in your data flow and export it. Exporting the data node exports the transformation that the node represents and the transformations that precede it.

### To export a data flow as a Jupyter notebook

1. Navigate to your data flow.
2. Choose the ellipsis icon next to the node that you want to export.
3. In the context menu, hover over **Export**, and then hover over **Export via Jupyter notebook**.
4. Choose one of the following:
  - **SageMaker Pipelines**
  - **Amazon S3**
  - **SageMaker AI Inference Pipeline**
  - **SageMaker AI Feature Store**
  - **Python Code**
5. The **Export data flow as notebook** dialog box opens. Select one of the following:
  - **Download a local copy**
  - **Export to S3 location**
6. If you selected **Export to S3 location**, enter the Amazon S3 location to which you want to export the notebook.
7. Choose **Export**.

Your Jupyter notebook should either download to your local machine, or you can find it saved in the Amazon S3 location you specified.

### Add destination nodes

A destination node in SageMaker Canvas specifies where to store your processed and transformed data. When you choose to export your transformed data to Amazon S3, Canvas uses the specified

destination node location, applying all the transformations you've configured in your data flow. For more information about export jobs to Amazon S3, see the preceding section [Export to Amazon S3](#).

By default, choosing to export your data to Amazon S3 adds a destination node to your data flow. However, you can add multiple destination nodes to your flow, allowing you to simultaneously export different sets of transformations or variations of your data to different Amazon S3 locations. For example, you can create one destination node that exports the data after applying all transformations, and another destination node that exports the data after only certain initial transformations, such as a join operation. This flexibility enables you to export and store different versions or subsets of your transformed data in separate S3 locations for various use cases.

Use the following procedure to add a destination node to your data flow.

### To add a destination node

1. Navigate to your data flow.
2. Choose the ellipsis icon next to the node where you want to place the destination node.
3. In the context menu, hover over **Export**, and then select **Add destination**.
4. In the **Export destination** side panel, enter a **Dataset name** to name the output.
5. For **Amazon S3 location**, enter the Amazon S3 location to which you want to export the output. You can enter the S3 URI, alias, or ARN of the S3 location or S3 access point. For more information access points, see [Managing data access with Amazon S3 access points](#) in the *Amazon S3 User Guide*.
6. For **Export settings**, specify the following fields:
  - a. **File type** – The file format of the exported data.
  - b. **Delimiter** – The delimiter used to separate values in the file.
  - c. **Compression** – The compression method used to reduce the file size.
7. For **Partitioning**, specify the following fields:
  - a. **Number of partitions** – The number of dataset files that SageMaker Canvas writes as the output of the job.
  - b. **Choose columns** – You can choose a subset of columns from the data to include in the partitions.
8. Choose **Add** if you want to simply add a destination node to your data flow, or choose **Add** and then choose **Export** if you want to add the node and initiate an export job.

You should now see a new destination node in your flow.

## Edit a destination node

A *destination node* in a Amazon SageMaker Canvas data flow specifies the Amazon S3 location where your processed and transformed data is stored, applying all the configured transformations in your data flow. You can edit the configuration of an existing destination node and then choose to re-run the job to overwrite the data in the specified Amazon S3 location. For more information about adding a new destination node, see [Add destination nodes](#).

Use the following procedure to edit a destination node in your data flow and initiate an export job.

### To edit a destination node

1. Navigate to your data flow.
2. Choose the ellipsis icon next to the destination node that you want to edit.
3. In the context menu, choose **Edit**.
4. The **Edit destination** side panel opens. From this panel, you can edit details such as the dataset name, the Amazon S3 location, and the export and partitioning settings.
5. (Optional) In **Additional nodes to export**, you can select more destination nodes to process when you run the export job.
6. Leave the **Process entire dataset** option selected if you want Canvas to apply your data flow transforms to your entire dataset and export the result. If you deselect this option, Canvas only applies the transforms to the sample of your dataset used in the interactive Data Wrangler data flow.
7. Leave the **Auto job configuration** option selected if you want Canvas to automatically determine whether to run the job using Canvas application memory or an EMR Serverless job. If you deselect this option and manually configure your job, then you can choose to use either an EMR Serverless or a SageMaker Processing job. For instructions on how to configure an EMR Serverless or a SageMaker Processing job, see the preceding section [Export to Amazon S3](#).
8. When you're done making changes, choose **Update**.

Saving changes to your destination node configuration doesn't automatically re-run a job or overwrite data that has already been processed and exported. Export your data again to run a job with the new configuration. If you decide to export your data again with a job, Canvas uses the updated destination node configuration to transform and output the data to the specified location, overwriting any existing data.

## Create a schedule to automatically process new data

### Note

The following section only applies to SageMaker Processing jobs. If you used the default Canvas settings or EMR Serverless to create a remote job to apply transforms to your full dataset, this section doesn't apply.

If you're processing data periodically, you can create a schedule to run the processing job automatically. For example, you can create a schedule that runs a processing job automatically when you get new data. For more information about processing jobs, see [Export to Amazon S3](#).

When you create a job, you must specify an IAM role that has permissions to create the job. You can use the [AmazonSageMakerCanvasDataPrepFullAccess](#) policy to add permissions.

Add the following trust policy to the role to allow EventBridge to assume it.

```
{  
    "Effect": "Allow",  
    "Principal": {  
        "Service": "events.amazonaws.com"  
    },  
    "Action": "sts:AssumeRole"  
}
```

### Important

When you create a schedule, Data Wrangler creates an eventRule in EventBridge. You incur charges for both the event rules that you create and the instances used to run the processing job.

For information about EventBridge pricing, see [Amazon EventBridge pricing](#). For information about processing job pricing, see [Amazon SageMaker Pricing](#).

You can set a schedule using one of the following methods:

- [CRON expressions](#)

**Note**

Data Wrangler doesn't support the following expressions:

- LW#
- Abbreviations for days
- Abbreviations for months

- [RATE expressions](#)

- Recurring – Set an hourly or daily interval to run the job.
- Specific time – Set specific days and times to run the job.

The following sections provide procedures on scheduling jobs when filling out the SageMaker AI Processing job settings while [exporting your data to Amazon S3](#). All of the following instructions begin in the **Associate schedules** section of the SageMaker Processing job settings.

## CRON

Use the following procedure to create a schedule with a CRON expression.

1. In the **Export to Amazon S3** side panel, make sure you've turned off the **Auto job configuration** toggle and have the **SageMaker Processing** option selected.
2. In the **SageMaker Processing** job settings, open the **Associate schedules** section and choose **Create new schedule**.
3. The **Create new schedule** dialog box opens. For **Schedule Name**, specify the name of the schedule.
4. For **Run Frequency**, choose **CRON**.
5. For each of the **Minutes**, **Hours**, **Days of month**, **Month**, and **Day of week** fields, enter valid CRON expression values.
6. Choose **Create**.
7. (Optional) Choose **Add another schedule** to run the job on an additional schedule.

**Note**

You can associate a maximum of two schedules. The schedules are independent and don't affect each other unless the times overlap.

8. Choose one of the following:
  - **Schedule and run now** – The job runs immediately and subsequently runs on the schedules.
  - **Schedule only** – The job only runs on the schedules that you specify.
9. Choose **Export** after you've filled out the rest of the export job settings.

## RATE

Use the following procedure to create a schedule with a RATE expression.

1. In the **Export to Amazon S3** side panel, make sure you've turned off the **Auto job configuration** toggle and have the **SageMaker Processing** option selected.
2. In the **SageMaker Processing** job settings, open the **Associate schedules** section and choose **Create new schedule**.
3. The **Create new schedule** dialog box opens. For **Schedule Name**, specify the name of the schedule.
4. For **Run Frequency**, choose **Rate**.
5. For **Value**, specify an integer.
6. For **Unit**, select one of the following:
  - **Minutes**
  - **Hours**
  - **Days**
7. Choose **Create**.
8. (Optional) Choose **Add another schedule** to run the job on an additional schedule.

**Note**

You can associate a maximum of two schedules. The schedules are independent and don't affect each other unless the times overlap.

## 9. Choose one of the following:

- **Schedule and run now** – The job runs immediately and subsequently runs on the schedules.
- **Schedule only** – The job only runs on the schedules that you specify.

10. Choose **Export** after you've filled out the rest of the export job settings.**Recurring**

Use the following procedure to create a schedule that runs a job on a recurring basis.

1. In the **Export to Amazon S3** side panel, make sure you've turned off the **Auto job configuration** toggle and have the **SageMaker Processing** option selected.
2. In the **SageMaker Processing** job settings, open the **Associate schedules** section and choose **Create new schedule**.
3. The **Create new schedule** dialog box opens. For **Schedule Name**, specify the name of the schedule.
4. For **Run Frequency**, choose **Recurring**.
5. For **Every x hours**, specify the hourly frequency that the job runs during the day. Valid values are integers in the inclusive range of **1** and **23**.
6. For **On days**, select one of the following options:
  - **Every Day**
  - **Weekends**
  - **Weekdays**
  - **Select Days**
- (Optional) If you've selected **Select Days**, choose the days of the week to run the job.

**Note**

The schedule resets every day. If you schedule a job to run every five hours, it runs at the following times during the day:

- 00:00
- 05:00
- 10:00
- 15:00
- 20:00

7. Choose **Create**.
8. (Optional) Choose **Add another schedule** to run the job on an additional schedule.

**Note**

You can associate a maximum of two schedules. The schedules are independent and don't affect each other unless the times overlap.

9. Choose one of the following:
  - **Schedule and run now** – The job runs immediately and subsequently runs on the schedules.
  - **Schedule only** – The job only runs on the schedules that you specify.
10. Choose **Export** after you've filled out the rest of the export job settings.

## Specific time

Use the following procedure to create a schedule that runs a job at specific times.

1. In the **Export to Amazon S3** side panel, make sure you've turned off the **Auto job configuration** toggle and have the **SageMaker Processing** option selected.
2. In the **SageMaker Processing** job settings, open the **Associate schedules** section and choose **Create new schedule**.

3. The **Create new schedule** dialog box opens. For **Schedule Name**, specify the name of the schedule.
  4. For **Run Frequency**, choose **Start time**.
  5. For **Start time**, enter a time in UTC format (for example, **09:00**). The start time defaults to the time zone where you are located.
  6. For **On days**, select one of the following options:
    - **Every Day**
    - **Weekends**
    - **Weekdays**
    - **Select Days**  
  - (Optional) If you've selected **Select Days**, choose the days of the week to run the job.
7. Choose **Create**.
  8. (Optional) Choose **Add another schedule** to run the job on an additional schedule.

 **Note**

You can associate a maximum of two schedules. The schedules are independent and don't affect each other unless the times overlap.

9. Choose one of the following:
  - **Schedule and run now** – The job runs immediately and subsequently runs on the schedules.
  - **Schedule only** – The job only runs on the schedules that you specify.
10. Choose **Export** after you've filled out the rest of the export job settings.

You can use the SageMaker AI AWS Management Console to view the jobs that are scheduled to run. Your processing jobs run within Pipelines. Each processing job has its own pipeline. It runs as a processing step within the pipeline. You can view the schedules that you've created within a pipeline. For information about viewing a pipeline, see [View the details of a pipeline](#).

Use the following procedure to view the jobs that you've scheduled.

To view the jobs you've scheduled, do the following.

1. Open Amazon SageMaker Studio Classic.
2. Open Pipelines
3. View the pipelines for the jobs that you've created.

The pipeline running the job uses the job name as a prefix. For example, if you've created a job named `housing-data-feature-engineering`, the name of the pipeline is `canvas-data-prep-housing-data-feature-engineering`.

4. Choose the pipeline containing your job.
5. View the status of the pipelines. Pipelines with a **Status of Succeeded** have run the processing job successfully.

To stop the processing job from running, do the following:

To stop a processing job from running, delete the event rule that specifies the schedule. Deleting an event rule stops all the jobs associated with the schedule from running. For information about deleting a rule, see [Disabling or deleting an Amazon EventBridge rule](#).

You can stop and delete the pipelines associated with the schedules as well. For information about stopping a pipeline, see [StopPipelineExecution](#). For information about deleting a pipeline, see [DeletePipeline](#).

## Automate data preparation in SageMaker Canvas

After you transform your data in data flow, you can export the transforms to your machine learning workflows. When you export your transforms, SageMaker Canvas creates a Jupyter notebook. You must run the notebook within Amazon SageMaker Studio Classic. For information about getting started with Studio Classic, contact your administrator.

## Automate data preparation using Pipelines

When you want to build and deploy large-scale machine learning (ML) workflows, you can use Pipelines to create workflows that manage and deploy SageMaker AI jobs. With Pipelines, you can build workflows that manage your SageMaker AI data preparation, model training, and model deployment jobs. You can use the first-party algorithms that SageMaker AI offers by using Pipelines. For more information on Pipelines, see [SageMaker Pipelines](#).

When you export one or more steps from your data flow to Pipelines, Data Wrangler creates a Jupyter notebook that you can use to define, instantiate, run, and manage a pipeline.

## Use a Jupyter Notebook to Create a Pipeline

Use the following procedure to create a Jupyter notebook to export your Data Wrangler flow to Pipelines.

Use the following procedure to generate a Jupyter notebook and run it to export your Data Wrangler flow to Pipelines.

1. Choose the + next to the node that you want to export.
2. Choose **Export data flow**.
3. Choose **Pipelines (via Jupyter Notebook)**.
4. Download the Jupyter notebook or copy it to an Amazon S3 location. We recommend copying it to an Amazon S3 location that you can access within Studio Classic. Contact your administrator if you need guidance on a suitable location.
5. Run the Jupyter notebook.

You can use the Jupyter notebook that Data Wrangler produces to define a pipeline. The pipeline includes the data processing steps that are defined by your Data Wrangler flow.

You can add additional steps to your pipeline by adding steps to the steps list in the following code in the notebook:

```
pipeline = Pipeline(  
    name=pipeline_name,  
    parameters=[instance_type, instance_count],  
    steps=[step_process], #Add more steps to this list to run in your Pipeline  
)
```

For more information on defining pipelines, see [Define SageMaker AI Pipeline](#).

## Automate data preparation using an inference endpoint

Use your Data Wrangler flow to process data at the time of inference by creating a SageMaker AI serial inference pipeline from your Data Wrangler flow. An inference pipeline is a series of steps that results in a trained model making predictions on new data. A serial inference pipeline within Data Wrangler transforms the raw data and provides it to the machine learning model for a prediction. You create, run, and manage the inference pipeline from a Jupyter notebook within Studio Classic. For more information about accessing the notebook, see [Use a Jupyter notebook to create an inference endpoint](#).

Within the notebook, you can either train a machine learning model or specify one that you've already trained. You can either use Amazon SageMaker Autopilot or XGBoost to train the model using the data that you've transformed in your Data Wrangler flow.

The pipeline provides the ability to perform either batch or real-time inference. You can also add the Data Wrangler flow to SageMaker Model Registry. For more information about hosting models, see [Multi-model endpoints](#).

### **Important**

You can't export your Data Wrangler flow to an inference endpoint if it has the following transformations:

- Join
- Concatenate
- Group by

If you must use the preceding transforms to prepare your data, use the following procedure.

#### **To prepare your data for inference with unsupported transforms**

1. Create a Data Wrangler flow.
2. Apply the preceding transforms that aren't supported.
3. Export the data to an Amazon S3 bucket.
4. Create a separate Data Wrangler flow.
5. Import the data that you've exported from the preceding flow.
6. Apply the remaining transforms.
7. Create a serial inference pipeline using the Jupyter notebook that we provide.

For information about exporting your data to an Amazon S3 bucket see [Export data](#).

For information about opening the Jupyter notebook used to create the serial inference pipeline, see [Use a Jupyter notebook to create an inference endpoint](#).

Data Wrangler ignores transforms that remove data at the time of inference. For example, Data Wrangler ignores the [Handle Missing Values](#) transform if you use the **Drop missing** configuration.

If you've refit transforms to your entire dataset, the transforms carry over to your inference pipeline. For example, if you used the median value to impute missing values, the median value from refitting the transform is applied to your inference requests. You can either refit the transforms from your Data Wrangler flow when you're using the Jupyter notebook or when you're exporting your data to an inference pipeline. .

The serial inference pipeline supports the following data types for the input and output strings. Each data type has a set of requirements.

## Supported datatypes

- **text/csv** – the datatype for CSV strings
  - The string can't have a header.
  - Features used for the inference pipeline must be in the same order as features in the training dataset.
  - There must be a comma delimiter between features.
  - Records must be delimited by a newline character.

The following is an example of a validly formatted CSV string that you can provide in an inference request.

```
abc,0.0,"Doe, John",12345\ndef,1.1,"Doe, Jane",67890
```

- **application/json** – the datatype for JSON strings
  - The features used in the dataset for the inference pipeline must be in the same order as the features in the training dataset.
  - The data must have a specific schema. You define schema as a single `instances` object that has a set of features. Each `features` object represents an observation.

The following is an example of a validly formatted JSON string that you can provide in an inference request.

```
{  
  "instances": [  
    {  
      "Feature1": 1,  
      "Feature2": 2  
    }  
  ]  
}
```

```
{  
    "features": ["abc", 0.0, "Doe, John", 12345]  
},  
{  
    "features": ["def", 1.1, "Doe, Jane", 67890]  
}  
]  
}
```

## Use a Jupyter notebook to create an inference endpoint

Use the following procedure to export your Data Wrangler flow to create an inference pipeline.

To create an inference pipeline using a Jupyter notebook, do the following.

1. Choose the **+** next to the node that you want to export.
2. Choose **Export data flow**.
3. Choose **SageMaker AI Inference Pipeline (via Jupyter Notebook)**.
4. Download the Jupyter notebook or copy it to an Amazon S3 location. We recommend copying it to an Amazon S3 location that you can access within Studio Classic. Contact your administrator if you need guidance on a suitable location.
5. Run the Jupyter notebook.

When you run the Jupyter notebook, it creates an inference flow artifact. An inference flow artifact is a Data Wrangler flow file with additional metadata used to create the serial inference pipeline. The node that you're exporting encompasses all of the transforms from the preceding nodes.

### **Important**

Data Wrangler needs the inference flow artifact to run the inference pipeline. You can't use your own flow file as the artifact. You must create it by using the preceding procedure.

## Automate data preparation using Python Code

To export all steps in your data flow to a Python file that you can manually integrate into any data processing workflow, use the following procedure.

Use the following procedure to generate a Jupyter notebook and run it to export your Data Wrangler flow to Python code.

1. Choose the **+** next to the node that you want to export.
2. Choose **Export data flow**.
3. Choose **Python Code**.
4. Download the Jupyter notebook or copy it to an Amazon S3 location. We recommend copying it to an Amazon S3 location that you can access within Studio Classic. Contact your administrator if you need guidance on a suitable location.
5. Run the Jupyter notebook.

You might need to configure the Python script to make it run in your pipeline. For example, if you're running a Spark environment, make sure that you are running the script from an environment that has permission to access AWS resources.

## Generative AI foundation models in SageMaker Canvas

Amazon SageMaker Canvas provides generative AI foundation models that you can use to start conversational chats. These content generation models are trained on large amounts of text data to learn the statistical patterns and relationships between words, and they can produce coherent text that is statistically similar to the text on which they were trained. You can use this capability to increase your productivity by doing the following:

- Generate content, such as document outlines, reports, and blogs
- Summarize text from large corpuses of text, such as earnings call transcripts, annual reports, or chapters of user manuals
- Extract insights and key takeaways from large passages of text, such as meeting notes or narratives
- Improve text and catch grammatical errors or typos

The foundation models are a combination of Amazon SageMaker JumpStart and [Amazon Bedrock](#) large language models (LLMs). Canvas offers the following models:

Model	Type	Description
Amazon Titan	Amazon Bedrock model	<p>Amazon Titan is a powerful, general-purpose language model that you can use for tasks such as summarization, text generation (such as creating a blog post), classification, open-ended Q&amp;A, and information extraction. It is pretrained on large datasets, making it suitable for complex tasks and reasoning. To continue supporting best practices in the responsible use of AI, Amazon Titan foundation models are built to detect and remove harmful content in the data, reject inappropriate content in the user input, and filter model outputs that contain inappropriate content (such as hate speech, profanity, and violence).</p>
Anthropic Claude Instant	Amazon Bedrock model	<p>Anthropic's Claude Instant is a faster and more cost-effective yet still very capable model. This model can handle a range of tasks including casual dialogue, text analysis, summarization, and document question answering. Just like Claude-2, Claude Instant can support</p>

Model	Type	Description
		up to 100,000 tokens in each prompt, equivalent to about 200 pages of information.
Anthropic Claude-2	Amazon Bedrock model	Claude-2 is Anthropic's most powerful model, which excels at a wide range of tasks from sophisticated dialogue and creative content generation to detailed instruction following. Claude-2 can take up to 100,000 tokens in each prompt, equivalent to about 200 pages of information. It can generate longer responses compared to its prior version. It supports use cases such as question answering, information extraction, removing PII, content generation, multiple-choice classification, roleplay, comparing text, summarization, and document Q&A with citation.

Model	Type	Description
Falcon-7B-Instruct	JumpStart model	Falcon-7B-Instruct has 7 billion parameters and was fine-tuned on a mixture of chat and instruct datasets. It is suitable as a virtual assistant and performs best when following instructions or engaging in conversation. Since the model was trained on large amounts of English-language web data, it carries the stereotypes and biases commonly found online and is not suitable for languages other than English. Compared to Falcon-40B-Instruct, Falcon-7B-Instruct is a slightly smaller and more compact model.

Model	Type	Description
Falcon-40B-Instruct	JumpStart model	<p>Falcon-40B-Instruct has 40 billion parameters and was fine-tuned on a mixture of chat and instruct datasets. It is suitable as a virtual assistant and performs best when following instructions or engaging in conversation. Since the model was trained on large amounts of English-language web data, it carries the stereotypes and biases commonly found online and is not suitable for languages other than English.</p> <p>Compared to Falcon-7B-Instruct, Falcon-40B-Instruct is a slightly larger and more powerful model.</p>

Model	Type	Description
Jurassic-2 Mid	Amazon Bedrock model	<p>Jurassic-2 Mid is a high-performance text generation model trained on a massive corpus of text (current up to mid 2022). It is highly versatile, general-purpose, and capable of composing human-like text and solving complex tasks such as question answering, text classification, and many others. This model offers zero-shot instruction capabilities, allowing it to be directed with only natural language and without the use of examples. It performs up to 30% faster than its predecessor, the Jurassic-1 model.</p> <p>Jurassic-2 Mid is AI21's mid-sized model, carefully designed to strike the right balance between exceptional quality and affordability.</p>

Model	Type	Description
Jurassic-2 Ultra	Amazon Bedrock model	<p>Jurassic-2 Ultra is a high-performance text generation model trained on a massive corpus of text (current up to mid 2022). It is highly versatile, general-purpose, and capable of composing human-like text and solving complex tasks such as question answering, text classification, and many others. This model offers zero-shot instruction capabilities, allowing it to be directed with only natural language and without the use of examples. It performs up to 30% faster than its predecessor, the Jurassic-1 model.</p> <p>Compared to Jurassic-2 Mid, Jurassic-2 Ultra is a slightly larger and more powerful model.</p>

Model	Type	Description
Llama-2-7b-Chat	JumpStart model	Llama-2-7b-Chat is a foundation model by Meta that is suitable for engaging in meaningful and coherent conversations, generating new content, and extracting answers from existing notes. Since the model was trained on large amounts of English-language internet data, it carries the biases and limitations commonly found online and is best-suited for tasks in English.

Model	Type	Description
Llama-2-13B-Chat	Amazon Bedrock model	<p>Llama-2-13B-Chat by Meta was fine-tuned on conversational data after initial training on internet data. It is optimized for natural dialog and engaging chat abilities, making it well-suited as a conversational agent. Compared to the smaller Llama-2-7b-Chat, Llama-2-13B-Chat has nearly twice as many parameters, allowing it to remember more context and produce more nuanced conversational responses. Like Llama-2-7b-Chat, Llama-2-13B-Chat was trained on English-language data and is best-suited for tasks in English.</p>

Model	Type	Description
Llama-2-70B-Chat	Amazon Bedrock model	<p>Like Llama-2-7b-Chat and Llama-2-13B-Chat, the Llama-2-70B-Chat model by Meta is optimized for engaging in natural and meaningful dialog. With 70 billion parameters, this large conversational model can remember more extensive context and produce highly coherent responses when compared to the more compact model versions. However, this comes at the cost of slower responses and higher resource requirements. Llama-2-70B-Chat was trained on large amounts of English-language internet data and is best-suited for tasks in English.</p>

Model	Type	Description
Mistral-7B	JumpStart model	Mistral-7B by Mistral.AI is an excellent general purpose language model suitable for a wide range of natural language (NLP) tasks like text generation, summarization, and question answering. It utilizes grouped-query attention (GQA) which allows for faster inference speeds, making it perform comparably to models with twice or three times as many parameters. It was trained on a mixture of text data including books, websites, and scientific papers in the English language, so it is best-suited for tasks in English.

Model	Type	Description
Mistral-7B-Chat	JumpStart model	Mistral-7B-Chat is a conversational model by Mistral.AI based on Mistral-7B. While Mistral-7B is best for general NLP tasks, Mistral-7B-Chat has been further fine-tuned on conversational data to optimize its abilities for natural, engaging chat. As a result, Mistral-7B-Chat generates more human-like responses and remembers the context of previous responses. Like Mistral-7B, this model is best-suited for English language tasks.
MPT-7B-Instruct	JumpStart model	MPT-7B-Instruct is a model for long-form instruction following tasks and can assist you with writing tasks including text summarization and question-answering to save you time and effort. This model was trained on large amounts of fine-tuned data and can handle larger inputs, such as complex documents. Use this model when you want to process large bodies of text or want the model to generate long responses.

The foundation models from Amazon Bedrock are currently only available in the US East (N. Virginia) and US West (Oregon) Regions. Additionally, when using foundation models from Amazon Bedrock, you are charged based on the volume of input tokens and output tokens, as specified by each model provider. For more information, see the [Amazon Bedrock pricing page](#). The JumpStart foundation models are deployed on SageMaker AI Hosting instances, and you are charged for the duration of usage based on the instance type used. For more information about the cost of different instance types, see the Amazon SageMaker AI Hosting: Real-Time Inference section on the [SageMaker pricing page](#).

Document querying is an additional feature that you can use to query and get insights from documents stored in indexes using Amazon Kendra. With this functionality, you can generate content from the context of those documents and receive responses that are specific to your business use case, as opposed to responses that are generic to the large amounts of data on which the foundation models were trained. For more information about indexes in Amazon Kendra, see the [Amazon Kendra Developer Guide](#).

If you would like to get responses from any of the foundation models that are customized to your data and use case, you can fine-tune foundation models. To learn more, see [Fine-tune foundation models](#).

If you'd like to get predictions from an Amazon SageMaker JumpStart foundation model through an application or website, you can deploy the model to a SageMaker AI *endpoint*. SageMaker AI endpoints host your model, and you can send requests to the endpoint through your application code to receive predictions from the model. For more information, see [Deploy your models to an endpoint](#).

## Complete the prerequisites for foundation models in SageMaker Canvas

The following sections outline the prerequisites for interacting with foundation models and using the document query feature in Canvas. The rest of the content on this page assumes that you've met the prerequisites for foundation models. The document query feature requires additional permissions.

### Prerequisites for foundation models

The permissions you need for interacting with models are included in the Canvas Ready-to-use models permissions. To use the generative AI-powered models in Canvas, you must turn on the **Canvas Ready-to-use models configuration** permissions when setting up your Amazon SageMaker AI domain. For more information, see [Prerequisites for setting up Amazon SageMaker Canvas](#). The **Canvas Ready-to-use models configuration** attaches the

AmazonSageMakerCanvasAISServicesAccess policy to your Canvas user's AWS Identity and Access Management (IAM) execution role. If you encounter any issues with granting permissions, see the topic [Troubleshooting issues with granting permissions through the SageMaker AI console](#).

If you've already set up your domain, you can edit your domain settings and turn on the permissions. For instructions on how to edit your domain settings, see [Edit domain settings](#). When editing the settings for your domain, go to the **Canvas settings** and turn on the **Enable Canvas Ready-to-use models** option.

Certain JumpStart foundation models also require that you request a SageMaker AI instance quota increase. Canvas hosts the models that you're currently interacting with on these instances, but the default quota for your account may be insufficient. If you run into an error while running any of the following models, request a quota increase for the associated instance types:

- Falcon-40B – ml.g5.12xlarge, ml.g5.24xlarge
- Falcon-13B – ml.g5.2xlarge, ml.g5.4xlarge, ml.g5.8xlarge
- MPT-7B-Instruct – ml.g5.2xlarge, ml.g5.4xlarge, ml.g5.8xlarge

For the preceding instances types, request an increase from 0 to 1 for the endpoint usage quota. For more information about how to increase an instance quota for your account, see [Requesting a quota increase](#) in the *Service Quotas User Guide*.

## Prerequisites for document querying

### Note

Document querying is supported in the following AWS Regions: US East (N. Virginia), US East (Ohio), US West (Oregon), Europe (Ireland), Asia Pacific (Singapore), Asia Pacific (Sydney), Asia Pacific (Tokyo), and Asia Pacific (Mumbai).

The document querying feature requires that you already have an Amazon Kendra index that stores your documents and document metadata. For more information about Amazon Kendra, see the [Amazon Kendra Developer Guide](#). To learn more about the quotas for querying indexes, see [Quotas](#) in the *Amazon Kendra Developer Guide*.

You must also make sure that your Canvas user profile has the necessary permissions for document querying. The [AmazonSageMakerCanvasFullAccess](#) policy must be attached to the AWS IAM

execution role for the SageMaker AI domain that hosts your Canvas application (this policy is attached by default to all new and existing Canvas user profiles). You must also specifically grant document querying permissions and specify access to one or more Amazon Kendra indexes.

If your Canvas administrator is setting up a new domain or user profile, have them set up the domain by following the instructions in [Prerequisites for setting up Amazon SageMaker Canvas](#). While setting up the domain, they can turn on the document querying permissions through the **Canvas Ready-to-use models configuration**.

The Canvas administrator can manage document querying permissions at the user profile level as well. For example, if the administrator wants to grant document querying permissions to some user profiles but remove permissions for others, they can edit the permissions for a specific user.

The following procedure shows how to turn on document querying permissions for a specific user profile:

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the user profile's domain.
5. On the **domain details** page, choose the **User profile** whose permissions you want to edit.
6. On the **User Details** page, choose **Edit**.
7. In the left navigation pane, choose **Canvas settings**.
8. In the **Canvas Ready-to-use models configuration** section, turn on the **Enable document query using Amazon Kendra** toggle.
9. In the dropdown, select one or more Amazon Kendra indexes to which you want to grant access.
10. Choose **Submit** to save the changes to your domain settings.

You should now be able to use Canvas foundation models to query documents in the specified Amazon Kendra indexes.

## Start a new conversation to generate, extract, or summarize content

To get started with generative AI foundation models in Canvas, you can initiate a new chat session with one of the models. For JumpStart models, you are charged while the model is active, so

you must start up models when you want to use them and shut them down when you are done interacting. If you do not shut down a JumpStart model, Canvas shuts it down after 2 hours of inactivity. For Amazon Bedrock models (such as Amazon Titan), you are charged by prompt; the models are already active and don't need to be started up or shut down. You are charged directly for use of these models by Amazon Bedrock.

To open a chat with a model, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **Ready-to-use models**.
3. Choose **Generate, extract and summarize content**.
4. On the welcome page, you'll receive a recommendation to start up the default model. You can start the recommended model, or you can choose **Select another model** from the dropdown to choose a different one.
5. If you selected a JumpStart foundation model, you have to start it up before it is available for use. Choose **Start up the model**, and then the model is deployed to a SageMaker AI instance. It might take several minutes for this to complete. When the model is ready, you can enter prompts and ask the model questions.

If you selected a foundation model from Amazon Bedrock, you can start using it instantly by entering a prompt and asking questions.

Depending on the model, you can perform various tasks. For example, you can enter a passage of text and ask the model to summarize it. Or, you can ask the model to come up with a short summary of the market trends in your domain.

The model's responses in a chat are based on the context of your previous prompts. If you want to ask a new question in the chat that is unrelated to the previous conversation topic, we recommend that you start a new chat with the model.

## Extract information from documents with document querying

### Note

This section assumes that you've completed the section above [Prerequisites for document querying](#).

Document querying is a feature that you can use while interacting with foundation models in Canvas. With document querying, you can access a corpus of documents stored in an Amazon Kendra *index*, which holds the contents of your documents and is structured in a way to make documents searchable. You can ask specific questions that are targeted to the data in your Amazon Kendra index, and the foundation model returns answers to your questions. For example, you can query an internal knowledge base of IT information and ask questions such as "How do I connect to my company's network?" For more information about setting up an index, see the [Amazon Kendra Developer Guide](#).

When using the document query feature, the foundation models restrict their responses to the content of the documents in your index with a technique called Retrieval Augmented Generation (RAG). This technique bundles the most relevant information from the index along with the user's prompt and sends it to the foundation model to get a response. Responses are limited to what can be found in your index, preventing the model from giving you incorrect responses based on external data. For more information about this process, see the blog post [Quickly build high-accuracy Generative AI applications on enterprise data](#).

To get started, in a chat with a foundation model in Canvas, turn on the **Document query** toggle at the top of the page. From the dropdown, select the Amazon Kendra index that you want to query. Then, you can begin asking questions related to the documents in your index.

### **Important**

Document querying supports the [Compare model outputs](#) feature. Any existing chat history is overwritten when you start a new chat to compare model outputs.

## Start up models

### **Note**

The following section describe starting up models, which only applies to the JumpStart foundation models, such as Falcon-40B-Instruct. You can access Amazon Bedrock models, such as Amazon Titan, instantly at any time.

You can start up as many JumpStart models as you like. Each active JumpStart model incurs charges on your account, so we recommend that you don't start up more models than you are currently using.

To start up another model, you can do the following:

1. On the **Generate, extract and summarize content** page, choose **New chat**.
2. Choose the model from the dropdown menu. If you want to choose a model not displayed in the dropdown, choose **Start up another model**, and then select the model that you want to start up.
3. Choose **Start up model**.

The model should begin starting up, and within a few minutes you can chat with the model.

## Shut down models

We highly recommend that you shut down models that you aren't using. The models automatically shut down after 2 hours of inactivity. However, to manually shut down a model, you can do the following:

1. On the **Generate, extract and summarize content** page, open the chat for the model that you want to shut down.
2. On the chat page, choose the **More options** icon (⋮).
3. Choose **Shut down model**.
4. In the **Shut down model** confirmation box, choose **Shut down**.

The model begins shutting down. If your chat compares two or more models, you can shut down an individual model from the chat page by choosing the model's **More options** icon (⋮) and then choosing **Shut down model**.

## Compare model outputs

You might want to compare the output of different models side by side to see which model output you prefer. This can help you decide which model is best suited to your use case. You can compare up to three models in chats.

**Note**

Each individual model incurs charges on your account.

You must start a new chat to add models for comparison. To compare the output of models side by side in a chat, do the following:

1. In a chat, choose **New chat**.
2. Choose **Compare**, and use the dropdown menu to select the model that you want to add. To add a third model, choose **Compare** again to add another model.

**Note**

If you want to use a JumpStart model that isn't currently active, you are prompted to start up the model.

When the models are active, you see the two models side by side in the chat. You can submit your prompt, and each model responds in the same chat, as shown in the following screenshot.

The screenshot shows the Amazon SageMaker Canvas interface. At the top, there are navigation links: 'Ready-to-use models' (selected), 'Generate, extract and summarize content', and 'ready-to-use model'. Below this is a header bar with a '+' button for 'New chat', the user icon 'Claude 2', and the model name 'Falcon-7B-Instruct'. A message input field at the bottom has the placeholder 'Ask me anything...'. The main area is a conversation window:

**Claude 2:** Write a report outlining the potential benefits and ROI of implementing a customer relationship management (CRM) system.

**Falcon-7B-Instruct:**

Here is a draft report on the potential benefits and ROI of implementing a CRM system:

**Potential Benefits and ROI of Implementing a CRM System**

**Introduction**

Customer relationship management (CRM) systems are software tools that help businesses manage relationships and interactions with customers and prospects. Implementing a CRM can provide numerous benefits that drive return on investment (ROI) through increased sales, improved customer retention and loyalty, and operational efficiencies. This report outlines the key benefits a business can realize from CRM implementation and how they translate to measurable ROI.

**Increased Sales**

One of the biggest potential benefits of CRM is increased sales revenue. By tracking every customer interaction in one centralized database, sales teams can better capitalize on cross-sell and upsell opportunities. The CRM provides insights into which customers are ready to buy more or upgrade their purchases. Sales reps can focus their efforts on these qualified leads to drive more sales.

According to research, businesses see an average 10% or more increase in annual revenue after implementing a CRM. For a company with \$10 million in yearly sales, a CRM could help generate an additional \$1 million or more.

**Improved Customer Retention**

It costs 5 times more to attract a new customer than to keep an existing one. A CRM helps retain more customers by tracking satisfaction levels, purchase history, and support needs. Sales and marketing teams can proactively reach out to at-risk customers and provide tailored promotions to encourage repeat purchases.

A customer relationship management (CRM) system can be a powerful tool for businesses of all sizes. It can help businesses to streamline their customer interactions, improve customer satisfaction, and increase their ROI. A CRM system can help businesses to better manage customer relationships by providing a centralized platform for customer data, including contact information and communication history. This can help businesses to better understand their customers and their needs, which can lead to improved customer service and increased customer loyalty.

A CRM system can also help businesses to improve their sales and marketing strategies. By tracking customer interactions and analyzing customer data, businesses can identify opportunities to increase sales and improve customer relationships.

Finally, a CRM system can help businesses to better understand their ROI. By tracking customer interactions and analyzing customer data, businesses can identify areas where they can save money and increase their ROI.

Overall, a CRM system can be a powerful tool for businesses of all sizes. It can help businesses to streamline their customer interactions, improve customer satisfaction, and increase their ROI.

When you're done interacting, make sure to shut down any JumpStart models individually to avoid incurring further charges.

## Fine-tune foundation models

The foundation models that you can access through Amazon SageMaker Canvas can help you with a range of general purpose tasks. However, if you have a specific use case and would like to customized responses based on your own data, you can *fine-tune* a foundation model.

To fine-tune a foundation model, you provide a dataset that consists of sample prompts and model responses. Then, you train the foundation model on the data. Finally, the fine-tuned foundation model is able to provide you with more specific responses.

The following list contains the foundation models that you can fine-tune in Canvas:

- Titan Express
- Falcon-7B
- Falcon-7B-Instruct
- Falcon-40B-Instruct

- Falcon-40B
- Flan-T5-Large
- Flan-T5-XL
- Flan-T5-Xxl
- MPT-7B
- MPT-7B-Instruct

You can access more detailed information about each foundation model in the Canvas application while fine-tuning a model. For more information, see [Fine-tune the model](#).

This topic describes how to fine-tune foundation models in Canvas.

## Before you begin

Before fine-tuning a foundation model, make sure that you have the permissions for Ready-to-use models in Canvas and an AWS Identity and Access Management execution role that has a trust relationship with Amazon Bedrock, which allows Amazon Bedrock to assume your role while fine-tuning foundation models.

While setting up or editing your Amazon SageMaker AI domain, you must 1) turn on the Canvas Ready-to-use models configuration permissions, and 2) create or specify an Amazon Bedrock role, which is an IAM execution role to which SageMaker AI attaches a trust relationship with Amazon Bedrock. For more information about configuring these settings, see [Prerequisites for setting up Amazon SageMaker Canvas](#).

You can configure the Amazon Bedrock role manually if you would rather use your own IAM execution role (instead of letting SageMaker AI create one on your behalf). For more information about configuring your own IAM execution role's trust relationship with Amazon Bedrock, see [Grant Users Permissions to Use Amazon Bedrock and Generative AI Features in Canvas](#).

You must also have a dataset that is formatted for fine-tuning large language models (LLMs). The following is a list of requirements for your dataset:

- The dataset must be tabular and contain at least two columns of text data—one input column (which contains example prompts to the model) and one output column (which contains example responses from the model).

An example is the following:

Input	Output
What are your shipping terms?	We offer free shipping on all orders over \$50. Orders under \$50 have a shipping fee of \$5.99.
How can I return an item?	To return an item, please visit our returns center and follow the instructions. You must provide your order number and the reason for the return.
I'm having trouble with my product. What can I do?	Please contact our customer support team and we will be happy to help you troubleshoot the issue.

- We recommend that the dataset has at least 100 text pairs (rows of corresponding input and output items). This ensures that the foundation model has enough data for fine-tuning and increases the accuracy of its responses.
- Each input and output item should contain a maximum of 512 characters. Anything longer is reduced to 512 characters when fine-tuning the foundation model.

When fine-tuning an Amazon Bedrock model, you must adhere to the Amazon Bedrock quotas. For more information, see [Model customization quotas](#) in the *Amazon Bedrock User Guide*.

For more information about general dataset requirements and limitations in Canvas, see [Create a dataset](#).

## Fine-tune a foundation model

You can fine-tune a foundation model by using any of the following methods in the Canvas application:

- While in a **Generate, extract and summarize content** chat with a foundation model, choose the **Fine-tune model** icon ).

- While in a chat with a foundation model, if you've re-generated the response two or more times, then Canvas offers you the option to **Fine-tune model**. The following screenshot shows you what this looks like.

Not happy with the model's response? You can fine-tune it to get the responses you want.

[Learn more about fine-tuning a model.](#)



- On the **My models** page, you can create a new model by choosing **New model**, and then select **Fine-tune foundation model**.
- On the **Ready-to-use models** home page, you can choose **Create your own model**, and then in the **Create new model** dialog box, choose **Fine-tune foundation model**.
- While browsing your datasets in the **Data Wrangler** tab, you can select a dataset and choose **Create a model**. Then, choose **Fine-tune foundation model**.

After you've begun to fine-tune a model, do the following:

### Select a dataset

On the **Select** tab of fine-tuning a model, you choose the data on which you'd like to train the foundation model.

Either select an existing dataset or create a new dataset that meets the requirements listed in the [Before you begin](#) section. For more information about how to create a dataset, see [Create a dataset](#).

When you've selected or created a dataset and you're ready to move on, choose **Select dataset**.

### Fine-tune the model

After selecting your data, you're now ready to begin training and fine-tune the model.

On the **Fine-tune** tab, do the following:

- (Optional) Choose **Learn more about our foundation models** to access more information about each model and help you decide which foundation model or models to deploy.
- For **Select up to 3 base models**, open the dropdown menu and check up to 3 foundation models (up to 2 JumpStart models and 1 Amazon Bedrock model) that you'd like to fine-tune during the training job. By fine-tuning multiple foundation models, you can compare

their performance and ultimately choose the one best suited to your use case as the default model. For more information about default models, see [View model candidates in the model leaderboard](#).

3. For **Select Input column**, select the column of text data in your dataset that contains the example model prompts.
4. For **Select Output column**, select the column of text data in your dataset that contains the example model responses.
5. (Optional) To configure advanced settings for the training job, choose **Configure model**. For more information about the advanced model building settings, see [Advanced model building configurations](#).

In the **Configure model** pop-up window, do the following:

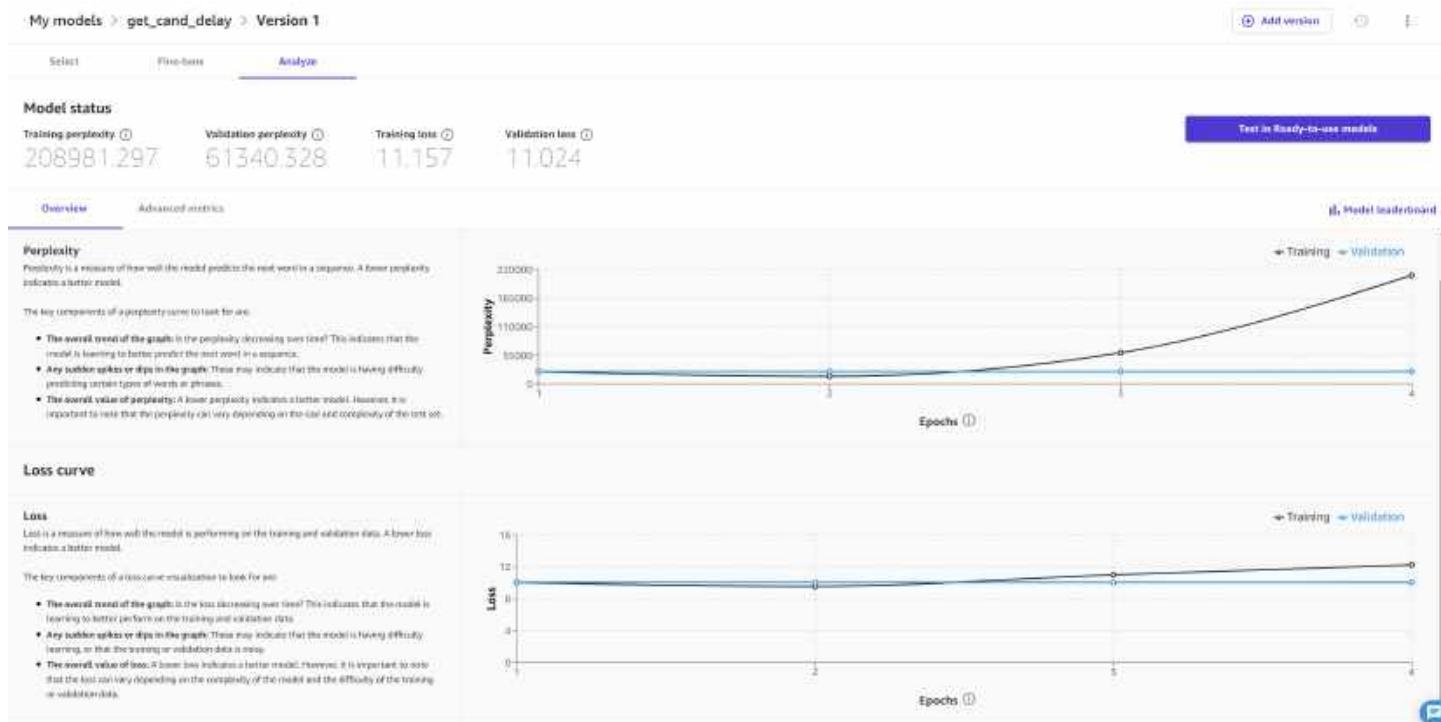
- a. For **Hyperparameters**, you can adjust the **Epoch count**, **Batch size**, **Learning rate**, and **Learning rate warmup steps** for each model you selected. For more information about these parameters, see the [Hyperparameters section in the JumpStart documentation](#).
  - b. For **Data split**, you can specify percentages for how to divide your data between the **Training set** and **Validation set**.
  - c. For **Max job runtime**, you can set the maximum amount of time that Canvas runs the build job. This feature is only available for JumpStart foundation models.
  - d. After configuring the settings, choose **Save**.
6. Choose **Fine-tune** to begin training the foundation models you selected.

After the fine-tuning job begins, you can leave the page. When the model shows as **Ready** on the **My models** page, it's ready for use, and you can now analyze the performance of your fine-tuned foundation model.

## Analyze the fine-tuned foundation model

On the **Analyze** tab of your fine-tuned foundation model, you can see the model's performance.

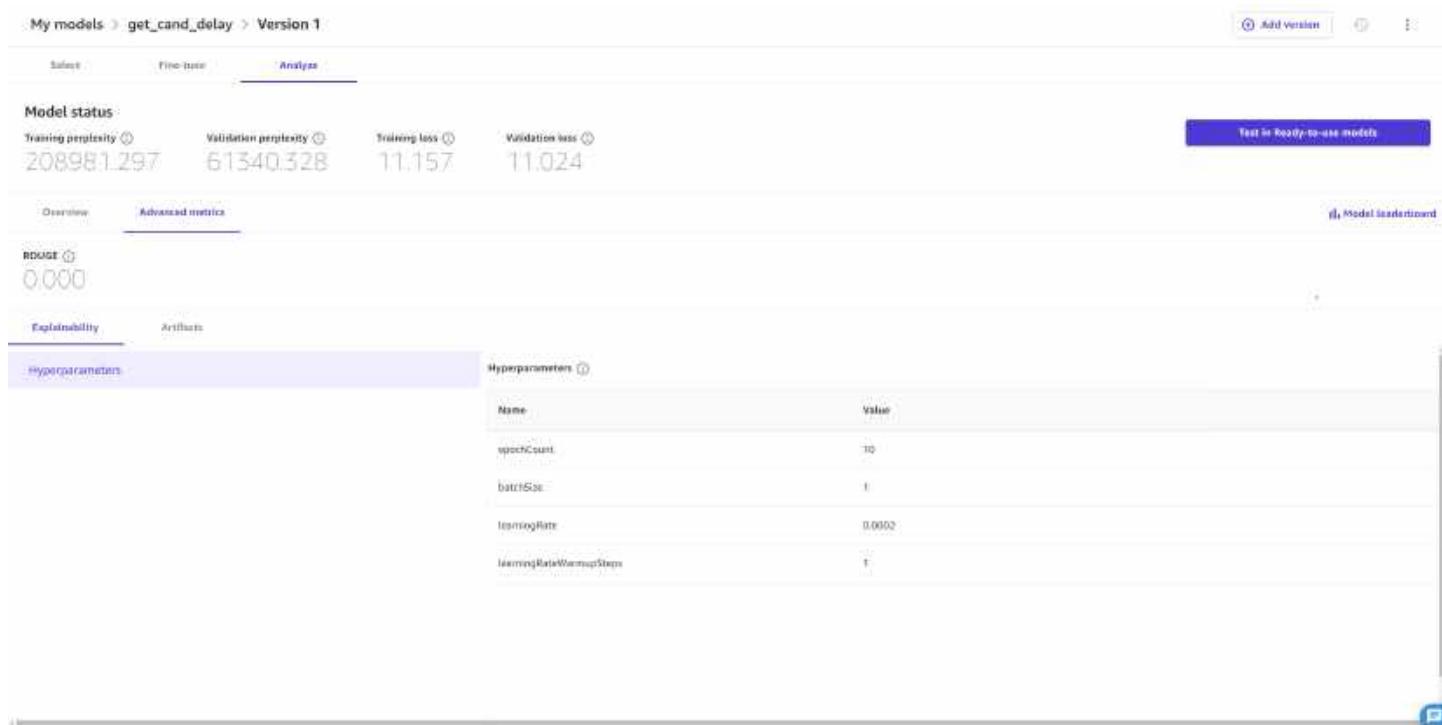
The **Overview** tab on this page shows you the perplexity and loss scores, along with analyses that visualize the model's improvement over time during training. The following screenshot shows the **Overview** tab.



On this page, you can see the following visualizations:

- The **Perplexity Curve** measures how well the model predicts the next word in a sequence, or how grammatical the model's output is. Ideally, as the model improves during training, the score decreases and results in a curve that lowers and flattens over time.
- The **Loss Curve** quantifies the difference between the correct output and the model's predicted output. A loss curve that decreases and flattens over time indicates that the model is improving its ability to make accurate predictions.

The **Advanced metrics** tab shows you the hyperparameters and additional metrics for your model. It looks like the following screenshot:



Name	Value
epochCount	10
batchSize	1
learningRate	0.0002
learningRateWarmupSteps	1

The **Advanced metrics** tab contains the following information:

- The **Explainability** section contains the **Hyperparameters**, which are the values set before the job to guide the model's fine-tuning. If you didn't specify custom hyperparameters in the model's advanced settings in the [Fine-tune the model](#) section, then Canvas selects default hyperparameters for you.

For JumpStart models, you can also see the advanced metric [ROUGE \(Recall-Oriented Understudy for Gisting Evaluation\)](#), which evaluates the quality of summaries generated by the model. It measures how well the model can summarize the main points of a passage.

- The **Artifacts** section provides you with links to artifacts generated during the fine-tuning job. You can access the training and validation data saved in Amazon S3, as well as the link to the model evaluation report (to learn more, see the following paragraph).

To get more model evaluation insights, you can download a report that is generated using [SageMaker Clarify](#), which is a feature that can help you detect bias in your model and data. First, generate the report by choosing **Generate evaluation report** at the bottom of the page. After the report has generated, you can download the full report by choosing **Download report** or by returning to the **Artifacts** section.

You can also access a Jupyter notebook that shows you how to replicate your fine-tuning job in Python code. You can use this to replicate or make programmatic changes to your fine-tuning job or get a deeper understanding of how Canvas fine-tunes your model. To learn more about model notebooks and how to access them, see [Download a model notebook](#).

For more information about how to interpret the information in the **Analyze** tab of your fine-tuned foundation model, see the topic [Model evaluation](#).

After analyzing the **Overview** and **Advanced metrics** tabs, you can also choose to open the **Model leaderboard**, which shows you the list of the base models trained during the build. The model with the lowest loss score is considered the best performing model and is selected as the **Default model**, which is the model whose analysis you see in the **Analyze** tab. You can only test and deploy the default model. For more information about the model leaderboard and how to change the default model, see [View model candidates in the model leaderboard](#).

## Test a fine-tuned foundation model in a chat

After analyzing the performance of a fine-tuned foundation model, you might want to test it out or compare its responses with the base model. You can test a fine-tuned foundation model in a chat in the **Generate, extract and summarize content** feature.

Start a chat with a fine-tuned model by choosing one of the following methods:

- On the fine-tuned model's **Analyze** tab, choose **Test in Ready-to-use foundation models**.
- On the Canvas **Ready-to-use models** page, choose **Generate, extract and summarize content**. Then, choose **New chat** and select the version of the model that you want to test.

The model starts up in a chat, and you can interact with it like any other foundation model. You can add more models to the chat and compare their outputs. For more information about the functionality of chats, see [Generative AI foundation models in SageMaker Canvas](#).

## Operationalize fine-tuned foundation models

After fine-tuning your model in Canvas, you can do the following:

- Register the model to the SageMaker Model Registry for integration into your organizations MLOps processes. For more information, see [Register a model version in the SageMaker AI model registry](#).

- Deploy the model to a SageMaker AI endpoint and send requests to the model from your application or website to get predictions (or *inference*). For more information, see [Deploy your models to an endpoint](#).

 **Important**

You can only register and deploy JumpStart based fine-tuned foundation models, not Amazon Bedrock based models.

## Ready-to-use models

With Amazon SageMaker Canvas Ready-to-use models, you can make predictions on your data without writing a single line of code or having to build a model—all you have to bring is your data. The Ready-to-use models use pre-built models to generate predictions without requiring you to spend the time, expertise, or cost required to build a model, and you can choose from a variety of use cases ranging from language detection to expense analysis.

Canvas integrates with existing AWS services, such as [Amazon Textract](#), [Amazon Rekognition](#), and [Amazon Comprehend](#), to analyze your data and make predictions or extract insights. You can use the predictive power of these services from within the Canvas application to get high quality predictions for your data.

Canvas supports the following Ready-to-use models types:

Ready-to-use model	Description	Supported data type
Sentiment analysis	Detect sentiment in lines of text, which can be positive, negative, neutral, or mixed. Currently, you can only do sentiment analysis for English language text.	Plain text or tabular (CSV, Parquet)
Entities extraction	Extract entities, which are real-world objects such as people, places, and commercial items, or units	Plain text or tabular (CSV, Parquet)

Ready-to-use model	Description	Supported data type
	such as dates and quantities, from text.	
Language detection	Determine the dominant language in text such as English, French, or German.	Plain text or tabular (CSV, Parquet)
Personal information detection	Detect personal information that could be used to identify an individual, such as addresses, bank account numbers, and phone numbers, from text.	Plain text or tabular (CSV, Parquet)
Object detection in images	Detect objects, concepts, scenes, and actions in your images.	Image (JPG, PNG)
Text detection in images	Detect text in your images.	Image (JPG, PNG)
Expense analysis	Extract information from invoices and receipts, such as date, number, item prices, total amount, and payment terms.	Document (PDF, JPG, PNG, TIFF)
Identity document analysis	Extract information from passports, driver licenses, and other identity documentation issued by the US Government.	Document (PDF, JPG, PNG, TIFF)
Document analysis	Analyze documents and forms for relationships among detected text.	Document (PDF, JPG, PNG, TIFF)

Ready-to-use model	Description	Supported data type
Document queries	Extract information from structured documents such as paystubs, bank statements, W-2s, and mortgage application forms by asking questions using natural language.	Document (PDF)

## Get started

To get started with Ready-to-use models, review the following information.

### Prerequisites

To use Ready-to-use models in Canvas, you must turn on the **Canvas Ready-to-use models configuration** permissions when [setting up your Amazon SageMaker AI domain](#). The **Canvas Ready-to-use models configuration** attaches the [AmazonSageMakerCanvasAI ServicesAccess](#) policy to your Canvas user's AWS Identity and Access Management (IAM) execution role. If you encounter any issues with granting permissions, see the topic [Troubleshooting issues with granting permissions through the SageMaker AI console](#).

If you've already set up your domain, you can edit your domain settings and turn on the permissions. For instructions on how to edit your domain settings, see [Edit domain settings](#). When editing the settings for your domain, go to the **Canvas settings** and turn on the **Enable Canvas Ready-to-use models** option.

### (Optional) Opt out of AI services data storage

Certain AWS AI services store and use your data to make improvements to the service. You can opt out of having your data stored or used for service improvements. To learn more about how to opt out, see [AI services opt-out policies](#) in the *AWS Organizations User Guide*.

### How to use Ready-to-use models

To get started with Ready-to-use models, do the following:

- 1. (Optional) Import your data.** You can import a tabular, image, or document dataset to generate batch predictions, or a dataset of predictions, with Ready-to-use models. To get started with importing a dataset, see [Create a data flow](#).

- 2. Generate predictions.** You can generate single or batch predictions with your chosen Ready-to-use model. To get started with making predictions, see [Make predictions for text data](#).

## Make predictions for text data

The following procedures describe how to make both single and batch predictions for text datasets. Each Ready-to-use model supports both **Single predictions** and **Batch predictions** for your dataset. A **Single prediction** is when you only need to make one prediction. For example, you have one image from which you want to extract text, or one paragraph of text for which you want to detect the dominant language. A **Batch prediction** is when you'd like to make predictions for an entire dataset. For example, you might have a CSV file of customer reviews for which you'd like to analyze the customer sentiment, or you might have image files in which you'd like to detect objects.

You can use these procedures for the following Ready-to-use model types: sentiment analysis, entities extraction, language detection, and personal information detection.

 **Note**

For sentiment analysis, you can only use English language text.

### Single predictions

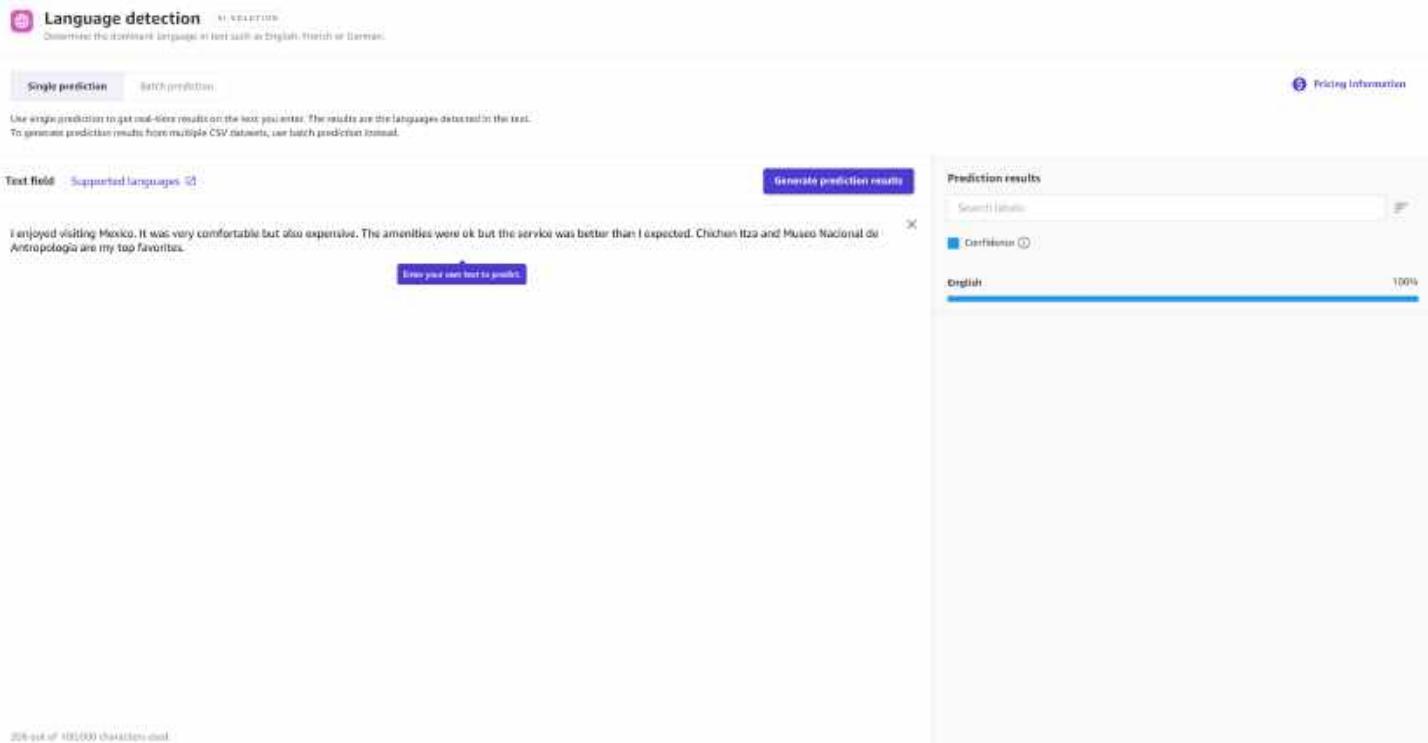
To make a single prediction for Ready-to-use models that accept text data, do the following:

1. In the left navigation pane of the Canvas application, choose **Ready-to-use models**.
2. On the **Ready-to-use models** page, choose the Ready-to-use model for your use case. For text data, it should be one of the following: **Sentiment analysis**, **Entities extraction**, **Language detection**, or **Personal information detection**.
3. On the **Run predictions** page for your chosen Ready-to-use model, choose **Single prediction**.
4. For **Text field**, enter the text for which you'd like to get a prediction.
5. Choose **Generate prediction results** to get your prediction.

In the right pane **Prediction results**, you receive an analysis of your text in addition to a **Confidence** score for each result or label. For example, if you chose language detection and

entered a passage of text in French, you might get French with a 95% confidence score and traces of other languages, like English, with a 5% confidence score.

The following screenshot shows the results for a single prediction using language detection where the model is 100% confident that the passage is English.



## Batch predictions

To make batch predictions for Ready-to-use models that accept text data, do the following:

1. In the left navigation pane of the Canvas application, choose **Ready-to-use models**.
2. On the **Ready-to-use models** page, choose the Ready-to-use model for your use case. For text data, it should be one of the following: **Sentiment analysis**, **Entities extraction**, **Language detection**, or **Personal information detection**.
3. On the **Run predictions** page for your chosen Ready-to-use model, choose **Batch prediction**.
4. Choose **Select dataset** if you've already imported your dataset. If not, choose **Import new dataset**, and then you are directed through the import data workflow.
5. From the list of available datasets, select your dataset and choose **Generate predictions** to get your predictions.

After the prediction job finishes running, on the **Run predictions** page, you see an output dataset listed under **Predictions**. This dataset contains your results, and if you select the **More options** icon (), you can **Preview** the output data. Then, you can choose **Download** to download the results.

## Make predictions for image data

The following procedures describe how to make both single and batch predictions for image datasets. Each Ready-to-use model supports both **Single predictions** and **Batch predictions** for your dataset. A **Single prediction** is when you only need to make one prediction. For example, you have one image from which you want to extract text, or one paragraph of text for which you want to detect the dominant language. A **Batch prediction** is when you'd like to make predictions for an entire dataset. For example, you might have a CSV file of customer reviews for which you'd like to analyze the customer sentiment, or you might have image files in which you'd like to detect objects.

You can use these procedures for the following Ready-to-use model types: object detection images and text detection in images.

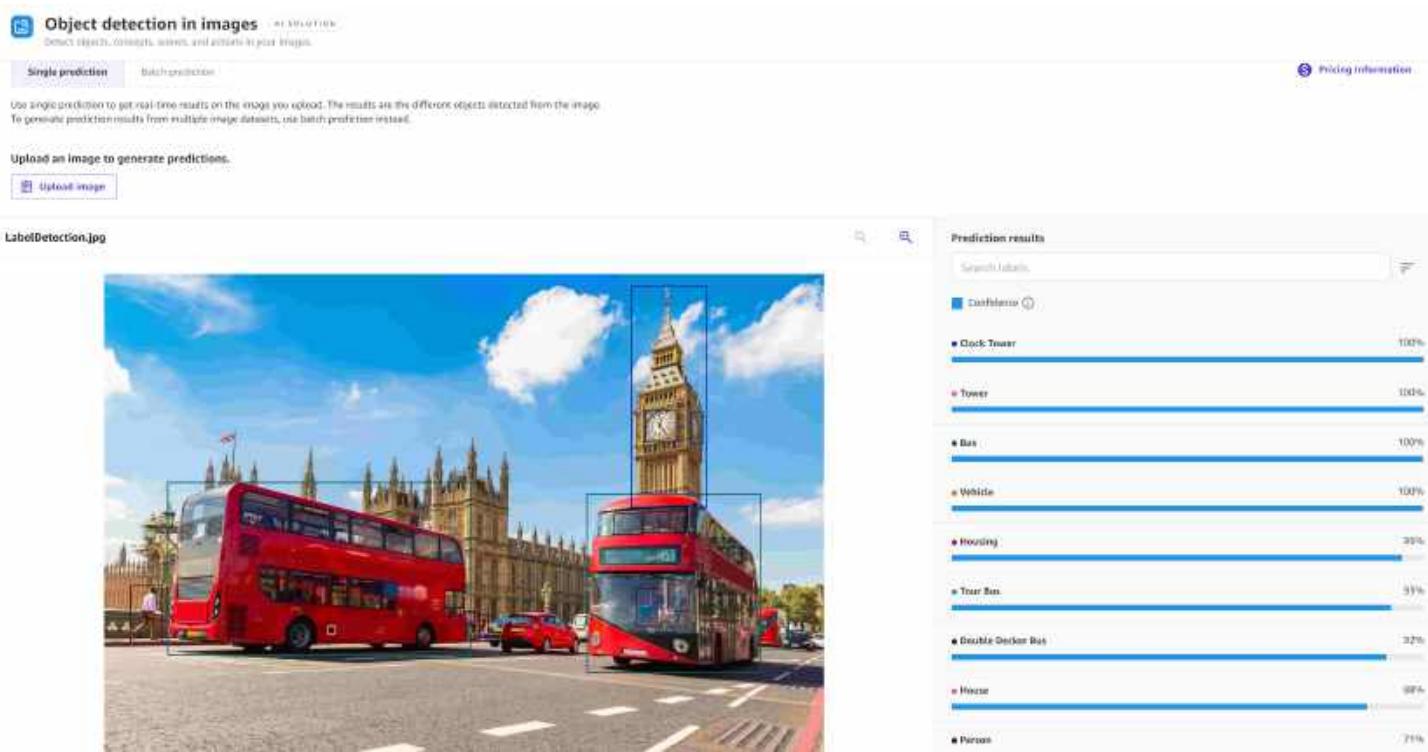
### Single predictions

To make a single prediction for Ready-to-use models that accept image data, do the following:

1. In the left navigation pane of the Canvas application, choose **Ready-to-use models**.
2. On the **Ready-to-use models** page, choose the Ready-to-use model for your use case. For image data, it should be one of the following: **Object detection images** or **Text detection in images**.
3. On the **Run predictions** page for your chosen Ready-to-use model, choose **Single prediction**.
4. Choose **Upload image**.
5. You are prompted to select an image to upload from your local computer. Select the image from your local files, and then the prediction results generate.

In the right pane **Prediction results**, you receive an analysis of your image in addition to a **Confidence** score for each object or text detected. For example, if you chose object detection in images, you receive a list of objects in the image along with a confidence score of how certain the model is that each object was accurately detected, such as 93%.

The following screenshot shows the results for a single prediction using the object detection in images solution, where the model predicts objects such as a clock tower and bus with 100% confidence.



## Batch predictions

To make batch predictions for Ready-to-use models that accept image data, do the following:

1. In the left navigation pane of the Canvas application, choose **Ready-to-use models**.
2. On the **Ready-to-use models** page, choose the Ready-to-use model for your use case. For image data, it should be one of the following: **Object detection images** or **Text detection in images**.
3. On the **Run predictions** page for your chosen Ready-to-use model, choose **Batch prediction**.
4. Choose **Select dataset** if you've already imported your dataset. If not, choose **Import new dataset**, and then you are directed through the import data workflow.
5. From the list of available datasets, select your dataset and choose **Generate predictions** to get your predictions.

After the prediction job finishes running, on the **Run predictions** page, you see an output dataset listed under **Predictions**. This dataset contains your results, and if you select the **More options** icon ( ! ),

you can choose **View prediction results** to preview the output data. Then, you can choose **Download prediction** and download the results as a CSV or a ZIP file.

## Make predictions for document data

The following procedures describe how to make both single and batch predictions for document datasets. Each Ready-to-use model supports both **Single predictions** and **Batch predictions** for your dataset. A **Single prediction** is when you only need to make one prediction. For example, you have one image from which you want to extract text, or one paragraph of text for which you want to detect the dominant language. A **Batch prediction** is when you'd like to make predictions for an entire dataset. For example, you might have a CSV file of customer reviews for which you'd like to analyze the customer sentiment, or you might have image files in which you'd like to detect objects.

You can use these procedures for the following Ready-to-use model types: expense analysis, identity document analysis, and document analysis.

### Note

For document queries, only single predictions are currently supported.

### Single predictions

To make a single prediction for Ready-to-use models that accept document data, do the following:

1. In the left navigation pane of the Canvas application, choose **Ready-to-use models**.
2. On the **Ready-to-use models** page, choose the Ready-to-use model for your use case. For document data, it should be one of the following: **Expense analysis**, **Identity document analysis**, or **Document analysis**.
3. On the **Run predictions** page for your chosen Ready-to-use model, choose **Single prediction**.
4. If your Ready-to-use model is identity document analysis or document analysis, complete the following actions. If you're doing expense analysis or document queries, skip this step and go to Step 5 or Step 6, respectively.
  - a. Choose **Upload document**.
  - b. You are prompted to upload a PDF, JPG, or PNG file from your local computer. Select the document from your local files, and then the prediction results will generate.

5. If your Ready-to-use model is expense analysis, do the following:
  - a. Choose **Upload invoice or receipt**.
  - b. You are prompted to upload a PDF, JPG, PNG, or TIFF file from your local computer. Select the document from your local files, and then the prediction results will generate.
6. If your Ready-to-use model is document queries, do the following:
  - a. Choose **Upload document**.
  - b. You are prompted to upload a PDF file from your local computer. Select the document from your local files. Your PDF must be 1–100 pages long.

 **Note**

If you're in the Asia Pacific (Seoul), Asia Pacific (Singapore), Asia Pacific (Sydney), or Europe (Frankfurt) regions, then the maximum PDF size for document queries is 20 pages.

- c. In the right side pane, enter queries to search for information in the document. The number of characters you can have in a single query is from 1–200. You can add up to 15 queries at a time.
- d. Choose **Submit queries**, and then the results generate with answers to your queries. You are billed once for each submissions of queries you make.

In the right pane **Prediction results**, you'll receive an analysis of your document.

The following information describes the results for each type of solution:

- For expense analysis, the results are categorized into **Summary fields**, which include fields such as the total on a receipt, and **Line item fields**, which include fields such as individual items on a receipt. The identified fields are highlighted on the document image in the output.
- For identity document analysis, the output shows you the fields that the Ready-to-use model identified, such as first and last name, address, or date of birth. The identified fields are highlighted on the document image in the output.
- For document analysis, the results are categorized into **Raw text**, **Forms**, **Tables**, and **Signatures**. **Raw text** includes all of the extracted text, while **Forms**, **Tables**, and **Signatures** only include information on the form that falls into those categories. For example, **Tables** only includes

information extracted from tables in the document. The identified fields are highlighted on the document image in the output.

- For document queries, Canvas returns answers to each of your queries. You can open the collapsible query dropdown to view a result, along with a confidence score for the prediction. If Canvas finds multiple answers in the document, then you might have more than one result for each query.

The following screenshot shows the results for a single prediction using the document analysis solution.

The screenshot shows the Amazon SageMaker Canvas interface for Document analysis. At the top, there's a navigation bar with tabs for "Single prediction" (selected), "Batch prediction", and "Pricing information". Below the navigation bar, there's a note about single predictions and batch predictions. A "Upload a document to generate predictions" button is present, with a "Upload document" link next to it. The main area displays a "Paystub.jpg" file and its "Prediction results". The results are categorized into "Raw text", "Forms", "Tables", and "Signatures". Under "Raw text", there are search fields for "Search label" and "Segment by line" or "Segment by word". The "Tables" section contains several tables with extracted data, such as "CO. FILE/DEPT CLOCK NUMBER", "ANY COMPANY 1200", "Period Ending", "7/16/2008", "475 ANY AVENUE", "Paydate", "7/21/2008", "ANYTOWNSUSA 10101", "Social Security Number: 847-23-4321", "Toussie Harold Dennis Hartnett", "JOHN UTILITIES", "Emergency/Residence", "121 MAIN STREET", "Federal: 1.5% Additional Tax", "ANITOWN, USA 12345", "State: 2", "Local: 2", "Earnings", "1000", "Hours", "Overtime", "year-to-date", "Other Benefits and", "Regular", "10.00", "32.00", "320.00", "16.00", "20.00", "Information", "This period", "Total to Date", "Overtime", "1133", "100", "10.00", "180.00", "Group Term Life", "0.51", "27.00", "Holiday", "10.00", "8.00", "90.00", "4,100.00", "Leave Accr. Paid", "840.00", "Tuition", "37.43", "1,900.00", "Gross Pay", "4,412.43", "25,826.00", "Voc. Adv.", "40.00", "Sick Adv.", "10.00", "Deductions", "Statutory", "10.00", "Overtime", "Federal Income Tax", "40.00", "2,111.20", "Social Security Tax", "20.00", "1,418.00", "Medicare Tax", "0.50", "541.12", "Reportable Netpay", "NY State Income Tax", "4.43", "418.00", "EFFECTIVE THIS PAY PERIOD YOUR REGULAR", "NYC Income Tax", "4.34".

## Batch predictions

To make batch predictions for Ready-to-use models that accept document data, do the following:

- In the left navigation pane of the Canvas application, choose **Ready-to-use models**.
- On the **Ready-to-use models** page, choose the Ready-to-use model for your use case. For image data, it should be one of the following: **Expense analysis**, **Identity document analysis**, or **Document analysis**.
- On the **Run predictions** page for your chosen Ready-to-use model, choose **Batch prediction**.

4. Choose **Select dataset** if you've already imported your dataset. If not, choose **Import new dataset**, and then you are directed through the import data workflow.
5. From the list of available datasets, select your dataset and choose **Generate predictions**. If your use case is document analysis, continue to Step 6.
6. (Optional) If your use case is Document analysis, another dialog box called **Select features to include in batch prediction** appears. You can select **Forms**, **Tables**, and **Signatures** to group the results by those features. Then, choose **Generate predictions**.

After the prediction job finishes running, on the **Run predictions** page, you see an output dataset listed under **Predictions**. This dataset contains your results, and if you select the **More options** icon (), you can choose **View prediction results** to preview the analysis of your document data.

The following information describes the results for each type of solution:

- For expense analysis, the results are categorized into **Summary fields**, which include fields such as the total on a receipt, and **Line item fields**, which include fields such as individual items on a receipt. The identified fields are highlighted on the document image in the output.
- For identity document analysis, the output shows you the fields that the Ready-to-use model identified, such as first and last name, address, or date of birth. The identified fields are highlighted on the document image in the output.
- For document analysis, the results are categorized into **Raw text**, **Forms**, **Tables**, and **Signatures**. **Raw text** includes all of the extracted text, while **Forms**, **Tables**, and **Signatures** only include information on the form that falls into those categories. For example, **Tables** only includes information extracted from tables in the document. The identified fields are highlighted on the document image in the output.

After previewing your results, you can choose **Download prediction** and download the results as a ZIP file.

## Custom models

In Amazon SageMaker Canvas, you can train custom machine learning models tailored to your specific data and use case. By training a custom model on your data, you are able to capture characteristics and trends that are specific and most representative of your data. For example, you

might want to create a custom time series forecasting model that you train on inventory data from your warehouse to manage your logistics operations.

Canvas supports training a range of model types. After training a custom model, you can evaluate the model's performance and accuracy. Once satisfied with a model, you can make predictions on new data, and you also have the option to share the custom model with data scientists for further analysis or to deploy it to a SageMaker AI hosted endpoint for real-time inference, all from within the Canvas application.

You can train a Canvas custom model on the following types of datasets:

- Tabular (including numeric, categorical, timeseries, and text data)
- Image

The following table shows the types of custom models that you can build in Canvas, along with their supported data types and data sources.

Model type	Example use case	Supported data types	Supported data sources
Numeric prediction	Predicting house prices based on features like square footage	Numeric	Local upload, Amazon S3, SaaS connectors
2 category prediction	Predicting whether or not a customer is likely to churn	Binary or categorical	Local upload, Amazon S3, SaaS connectors
3+ category prediction	Predicting patient outcomes after being discharged from the hospital	Categorical	Local upload, Amazon S3, SaaS connectors
Time series forecasting	Predicting your inventory for the next quarter	Timeseries	Local upload, Amazon S3, SaaS connectors

Model type	Example use case	Supported data types	Supported data sources
Single-label image prediction	Predicting types of manufacturing defects in images	Image (JPG, PNG)	Local upload, Amazon S3
Multi-category text prediction	Predicting categories of products, such as clothing, electronics, or household goods, based on product descriptions	Source column: text Target column: binary or categorical	Local upload, Amazon S3

## Get started

To get started with building and generating predictions from a custom model, do the following:

- Determine your use case and type of model that you want to build. For more information about the custom model types, see [How custom models work](#). For more information about the data types and sources supported for custom models, see [Data import](#).
- [Import your data](#) into Canvas. You can build a custom model with any tabular or image dataset that meets the input requirements. For more information about the input requirements, see [Create a dataset](#).

To learn more about sample datasets provided by SageMaker AI with which you can experiment, see [Sample datasets in Canvas](#).

- [Build](#) your custom model. You can do a **Quick build** to get your model and start making predictions more quickly, or you can do a **Standard build** for greater accuracy.

For numeric, categorical, and time series forecasting model types, you can clean and prepare your data with the [Data Wrangler feature](#). In Data Wrangler, you can create a data flow and use various data preparation techniques, such as applying advanced transforms or joining datasets. For image prediction models, you can [Edit an image dataset](#) to update your labels or add and delete images. Note that you can't use these features for multi-category text prediction models.

- [Evaluate your model's performance](#) and determine how well it might perform on real-world data.
- [Make single or batch predictions](#) with your model.

## How custom models work

Use Amazon SageMaker Canvas to build a custom model on the dataset that you've imported. Use the model that you've built to make predictions on new data. SageMaker Canvas uses the information in the dataset to build up to 250 models and choose the one that performs the best.

When you begin building a model, Canvas automatically recommends one or more *model types*. Model types fall into one of the following categories:

- **Numeric prediction** – This is known as *regression* in machine learning. Use the numeric prediction model type when you want to make predictions for numeric data. For example, you might want to predict the price of houses based on features such as the house's square footage.
- **Categorical prediction** – This is known as *classification* in machine learning. When you want to categorize data into groups, use the categorical prediction model types:
  - **2 category prediction** – Use the 2 category prediction model type (also known as *binary classification* in machine learning) when you have two categories that you want to predict for your data. For example, you might want to determine whether a customer is likely to churn.
  - **3+ category prediction** – Use the 3+ category prediction model type (also known as *multi-class classification* in machine learning) when you have three or more categories that you want to predict for your data. For example, you might want to predict a customer's loan status based on features such as previous payments.
- **Time series forecasting** – Use time series forecasts when you want to make predictions over a period of time. For example, you might want to predict the number of items you'll sell in the next quarter. For information about time series forecasts, see [Time Series Forecasts in Amazon SageMaker Canvas](#).
- **Image prediction** – Use the single-label image prediction model type (also known as *single-label image classification* in machine learning) when you want to assign labels to images. For example, you might want to classify different types of manufacturing defects in images of your product.
- **Text prediction** – Use the multi-category text prediction model type (also known as *multi-class text classification* in machine learning) when you want to assign labels to passages of text. For example, you might have a dataset of customer reviews for a product, and you want to determine whether customers liked or disliked the product. You might have your model predict whether a given passage of text is Positive, Negative, or Neutral.

For a table of the supported input data types for each model type, see [Custom models](#).

For each tabular data model that you build (which includes numeric, categorical, time series forecasting, and text prediction models), you choose the **Target column**. The **Target column** is the column that contains the information that you want to predict. For example, if you're building a model to predict whether people have cancelled their subscriptions, the **Target column** contains data points that are either a yes or a no about someone's cancellation status.

For image prediction models, you build the model with a dataset of images that have been assigned labels. For the unlabeled images that you provide, the model predicts a label. For example, if you're building a model to predict whether an image is a cat or a dog, you provide images labeled as cats or dogs when building the model. Then, the model can accept unlabeled images and predict them as either cats or dogs.

### What happens when you build a model

To build your model, you can choose either a **Quick build** or a **Standard build**. The **Quick build** has a shorter build time, but the **Standard build** generally has a higher accuracy.

For tabular and time series forecasting models, Canvas uses *downsampling* to reduce the size of datasets larger than 5 GB or 30 GB, respectively. Canvas downsamples with the stratified sampling method. The table below lists the size of the downsample by model type. To control the sampling process, you can use Data Wrangler in Canvas to sample using your preferred sampling technique. For time series data, you can resample to aggregate data points. For more information about sampling, see [Sampling](#). For more information about resampling time series data, see [Resample Time Series Data](#).

If you choose to do a **Quick build** on a dataset with more than 50,000 rows, then Canvas samples your data down to 50,000 rows for a shorter model training time.

The following table summarizes key characteristics of the model building process, including average build times for each model and build type, the size of the downsample when building models with large datasets, and the minimum and maximum number of data points you should have for each build type.

Limit	Numeric and categorical prediction	Time series forecasting	Image prediction	Text prediction
<b>Quick build time</b>	2-20 minutes	2-20 minutes	15-30 minutes	15-30 minutes

Limit	Numeric and categorical prediction	Time series forecasting	Image prediction	Text prediction
<b>Standard build time</b>	2-4 hours	2-4 hours	2-5 hours	2-5 hours
Downsample size (the reduced size of a large dataset after Canvas downsamples)	5 GB	30 GB	N/A	N/A
Minimum number of entries (rows) for <b>Quick builds</b>	2 category: 500 rows  3+ category, numeric, time series: N/A	N/A	N/A	N/A
Minimum number of entries (rows, images, or documents) for <b>Standard builds</b>	250	50	50	N/A
Maximum number of entries (rows, images, or documents) for <b>Quick builds</b>	N/A	N/A	5000	7500
Maximum number of entries (rows, images, or documents) for <b>Standard builds</b>	N/A	150,000	180,000	N/A
Maximum number of columns	1,000	1,000	N/A	N/A

Canvas predicts values by using the information in the rest of the dataset, depending on the model type:

- For categorical prediction, Canvas puts each row into one of the categories listed in the **Target column**.

- For numeric prediction, Canvas uses the information in the dataset to predict the numeric values in the **Target column**.
- For time series forecasting, Canvas uses historical data to predict values for the **Target column** in the future.
- For image prediction, Canvas uses images that have been assigned labels to predict labels for unlabeled images.
- For text prediction, Canvas analyzes text data that has been assigned labels to predict labels for passages of unlabeled text.

## Additional features to help you build your model

Before building your model, you can use Data Wrangler in Canvas to prepare your data using 300+ built-in transforms and operators. Data Wrangler supports transforms for both tabular and image datasets. Additionally, you can connect to data sources outside of Canvas, create jobs to apply transforms to your entire dataset, and export your fully prepared and cleaned data for use in ML workflows outside of Canvas. For more information, see [Data preparation](#).

To see visualizations and analytics to explore your data and determine which features to include in your model, you can use Data Wrangler's built-in analyses. You can also access a **Data Quality and Insights Report** that highlights potential issues with your dataset and provides recommendations for how to fix them. For more information, see [Perform exploratory data analysis \(EDA\)](#).

In addition to the more advanced data preparation and exploration functionality provided through Data Wrangler, Canvas provides some basic features that you can use:

- To filter your data and access a set of basic data transforms, see [Prepare data for model building](#).
- To access simple visualizations and analytics for feature exploration, see [Data exploration and analysis](#).
- To learn more about additional features such as previewing your model, validating your dataset, and changing the size of the random sample used to build your model, see [Preview your model](#).

For tabular datasets with multiple columns (such as datasets for building categorical, numeric, or time series forecasting model types), you might have rows with missing data points. While Canvas builds the model, it automatically adds missing values. Canvas uses the values in your dataset to perform a mathematical approximation for the missing values. For the highest model accuracy, we recommend adding in the missing data if you can find it. Note that the missing data feature is not supported for text prediction or image prediction models.

## Get started

To get started with building a custom model, see [Build a model](#) and follow the procedure for the type of model that you want to build.

## Preview your model

### Note

The following functionality is only available for custom models built with tabular datasets. Multi-category text prediction models are also excluded.

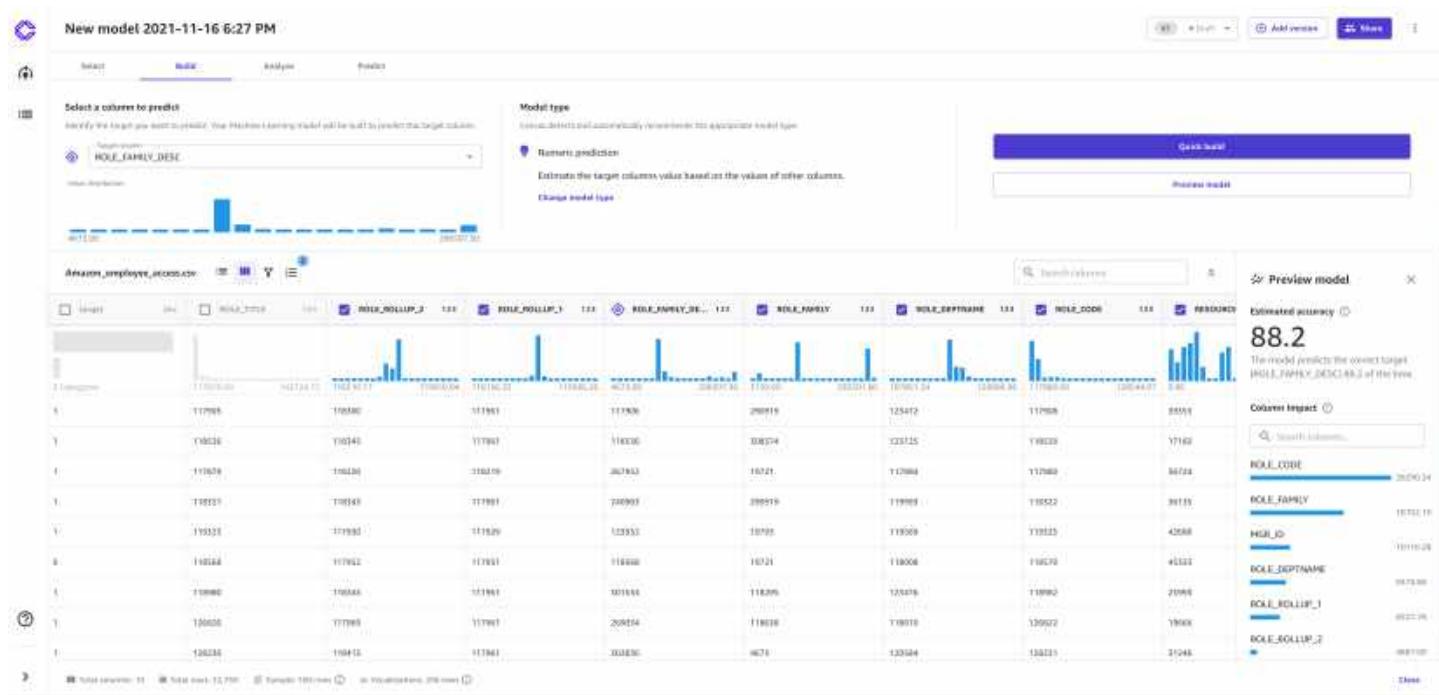
SageMaker Canvas provides you with a tool to preview your model before you begin building. This gives you an estimated accuracy score and also gives you a preliminary idea of how each column might impact the model.

To preview the model score, when you're on the **Build** tab of your model, choose **Preview model**.

The model preview generates an **Estimated accuracy** prediction of how well the model might analyze your data. The accuracy of a **Quick build** or a **Standard build** represents how well the model can perform on real data and is generally higher than the **Estimated accuracy**.

The model preview also provides you with the **Column Impact** scores, which can indicate the importance of each column to the model's predictions.

The following screenshot shows a model preview in the Canvas application.



Amazon SageMaker Canvas automatically handles missing values in your dataset while it builds the model. It infers the missing values by using adjacent values that are present in the dataset.

If you're satisfied with your model preview and want to proceed with building a model, then see [Build a model](#).

## Data validation

Before you build your model, SageMaker Canvas checks your dataset for issues that might cause your build to fail. If SageMaker Canvas finds any issues, then it warns you on the **Build** page before you attempt to build a model.

You can choose **Validate data** to see a list of the issues with your dataset. You can then use the SageMaker Canvas [Data Wrangler data preparation features](#), or your own tools, to fix your dataset before starting a build. If you don't fix the issues with your dataset, then your build fails.

If you make changes to your dataset to fix the issues, you have the option to re-validate your dataset before attempting a build. We recommend that you re-validate your dataset before building.

The following table shows the issues that SageMaker Canvas checks for in your dataset and how to resolve them.

Issue	Resolution
Wrong model type for your data	Try another model type or use a different dataset.
Missing values in your target column	Replace the missing values, drop rows with missing values, or use a different dataset.
Too many unique labels in your target column	Verify that you've used the correct column for your target column, or use a different dataset.
Too many non-numeric values in your target column	Choose a different target column, select another model type, or use a different dataset.
One or more column names contain double underscores	Rename the columns to remove any double underscores, and try again.
None of the rows in your dataset are complete	Replace the missing values, or use a different dataset.
Too many unique labels for the number of rows in your data	Check that you're using the right target column, increase the number of rows in your dataset, consolidate similar labels, or use a different dataset.

## Random sample

SageMaker Canvas uses the random sampling method to sample your dataset. The random sample method means that each row has an equal chance of being picked for the sample. You can choose a column in the preview to get summary statistics for the random sample, such as the mean and the mode.

By default, SageMaker Canvas uses a random sample size of 20,000 rows from your dataset for datasets with more than 20,000 rows. For datasets smaller than 20,000 rows, the default sample size is the number of rows in your dataset. You can increase or decrease the sample size by choosing **Random sample** in the **Build** tab of the SageMaker Canvas application. You can use the slider to select your desired sample size, and then choose **Update** to change the sample size. The maximum sample size you can choose for a dataset is 40,000 rows, and the minimum sample size is

500 rows. If you choose a large sample size, the dataset preview and summary statistics might take a few moments to reload.

The **Build** page shows a preview of 100 rows from your dataset. If the sample size is the same size as your dataset, then the preview uses the first 100 rows of your dataset. Otherwise, the preview uses the first 100 rows of the random sample.

## Build a model

The following sections show you how to build a model for each of the main types of custom models.

- To build numeric prediction, 2 category prediction, or 3+ category prediction models, see [Build a custom numeric or categorical prediction model](#).
- To build single-label image prediction models, see [Build a custom image prediction model](#).
- To build multi-category text prediction models, see [Build a custom text prediction model](#).
- To build time series forecasting models, see [Build a time series forecasting model](#).

### Note

If you encounter an error during post-building analysis that tells you to increase your quota for `m1.m5.2xlarge` instances, see [Request a Quota Increase](#).

## Build a custom numeric or categorical prediction model

Numeric and categorical prediction models support both **Quick builds** and **Standard builds**.

To build a numeric or categorical prediction model, use the following procedure:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **My models**.
3. Choose **New model**.
4. In the **Create new model** dialog box, do the following:
  - a. Enter a name in the **Model name** field.
  - b. Select the **Predictive analysis** problem type.

- c. Choose **Create**.
5. For **Select dataset**, select your dataset from the list of datasets. If you haven't already imported your data, choose **Import** to be directed through the import data workflow.
6. When you're ready to begin building your model, choose **Select dataset**.
7. On the **Build** tab, for the **Target column** dropdown list, select the target for your model that you would like to predict.
8. For **Model type**, Canvas automatically detects the problem type for you. If you want to change the type or configure advanced model settings, choose **Configure model**.

When the **Configure model** dialog box opens, do the following:

- a. For **Model type**, choose the model type that you want to build.
- b. After you choose the model type, there are additional **Advanced settings**. For more information about each of the advanced settings, see [Advanced model building configurations](#). To configure the advanced settings, do the following:
  - i. (Optional) For the **Objective metric** dropdown menu, select the metric that you want Canvas to optimize while building your model. If you don't select a metric, Canvas chooses one for you by default. For descriptions of the available metrics, see [Metrics reference](#).
  - ii. For **Training method**, choose **Auto**, **Ensemble**, or **Hyperparameter optimization (HPO) mode**.
  - iii. For **Algorithms**, select the algorithms that you want to include for building model candidates.
  - iv. For **Data split**, specify in percentages how you want to split your data between the **Training set** and the **Validation set**. The training set is used for building the model, while the validation set is used for testing accuracy of model candidates.
  - v. For **Max candidates and runtime**, do the following:
    - A. Set the **Max candidates** value, or the maximum number of model candidates that Canvas can generate. Note that **Max candidates** is only available in HPO mode.
    - B. Set the hour and minute values for **Max job runtime**, or the maximum amount of time that Canvas can spend building your model. After the maximum time, Canvas stops building and selects the best model candidate.
- c. After configuring the advanced settings, choose **Save**.

## 9. Select or deselect columns in your data to include or drop them from your build.

### Note

If you make batch predictions with your model after building, Canvas adds dropped columns to your prediction results. However, Canvas does not add the dropped columns to your batch predictions for time series models.

10. (Optional) Use the visualization and analytics tools that Canvas provides to visualize your data and determine which features you might want to include in your model. For more information, see [Explore and analyze your data](#).
11. (Optional) Use data transformations to clean, transform, and prepare your data for model building. For more information, see [Prepare your data with advanced transformations](#). You can view and remove your transforms by choosing **Model recipe** to open the **Model recipe** side panel.
12. (Optional) For additional features such as previewing the accuracy of your model, validating your dataset, and changing the size of the random sample that Canvas takes from your dataset, see [Preview your model](#).
13. After reviewing your data and making any changes to your dataset, choose **Quick build** or **Standard build** to begin a build for your model. The following screenshot shows the **Build** page and the **Quick build** and **Standard build** options.

The screenshot shows the Amazon SageMaker Canvas interface for the 'titanic-model' dataset. At the top, there are tabs for 'Select', 'Build', 'Analyze', and 'Predict'. The 'Build' tab is selected. A message at the top says 'No items have been found in your dataset'. On the left, a sidebar titled 'Select a column to predict' shows 'Survived' as the target column. Below it, a 'Value distribution' chart shows the count of rows for each value. In the center, under 'Model type', it says 'SageMaker Canvas automatically recommends the appropriate model type for your analysis.' It shows '2 category prediction' where 'Survived' is classified into two categories. On the right, a dropdown menu for 'Build type' is open, showing 'Quick build' (selected) and 'Standard build'. The 'Quick build' section indicates it's suitable for training and testing with up to 2,000 instances. The main area displays a table of dataset statistics for columns like 'Survived', 'Pclass', 'Sex', 'Age', etc. At the bottom, there are status bars for 'Total instances: 891' and 'Total rows: 891'.

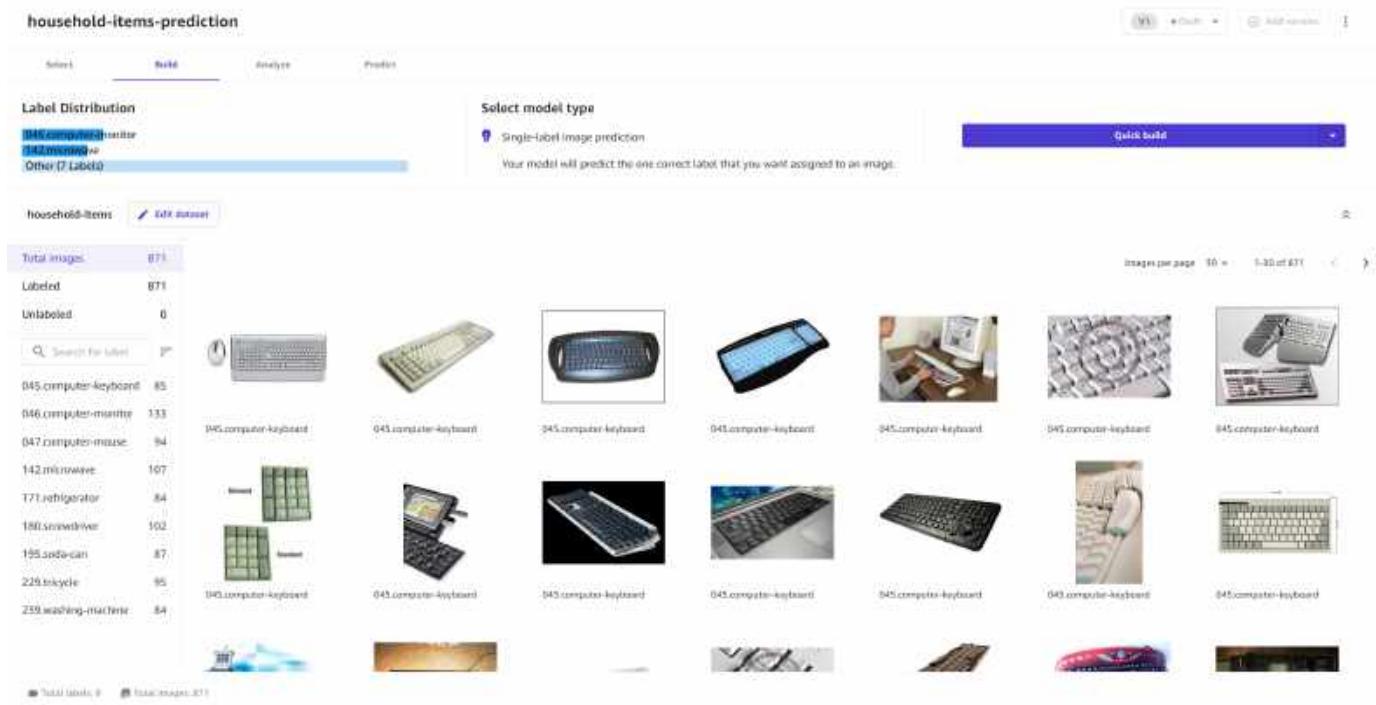
After your model begins building, you can leave the page. When the model shows as **Ready** on the **My models** page, it's ready for analysis and predictions.

## Build a custom image prediction model

Single-label image prediction models support both **Quick builds** and **Standard builds**.

To build a single-label image prediction model, use the following procedure:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **My models**.
3. Choose **New model**.
4. In the **Create new model** dialog box, do the following:
  - a. Enter a name in the **Model name** field.
  - b. Select the **Image analysis** problem type.
  - c. Choose **Create**.
5. For **Select dataset**, select your dataset from the list of datasets. If you haven't already imported your data, choose **Import** to be directed through the import data workflow.
6. When you're ready to begin building your model, choose **Select dataset**.
7. On the **Build** tab, you see the **Label distribution** for the images in your dataset. The **Model type** is set to **Single-label image prediction**.
8. On this page, you can preview your images and edit the dataset. If you have any unlabeled images, choose **Edit dataset** and [Assign labels to unlabeled images](#). You can also perform other tasks when you [Edit an image dataset](#), such as renaming labels and adding images to the dataset.
9. After reviewing your data and making any changes to your dataset, choose **Quick build** or **Standard build** to begin a build for your model. The following screenshot shows the **Build** page of an image prediction model that is ready to be built.



After your model begins building, you can leave the page. When the model shows as **Ready** on the **My models** page, it's ready for analysis and predictions.

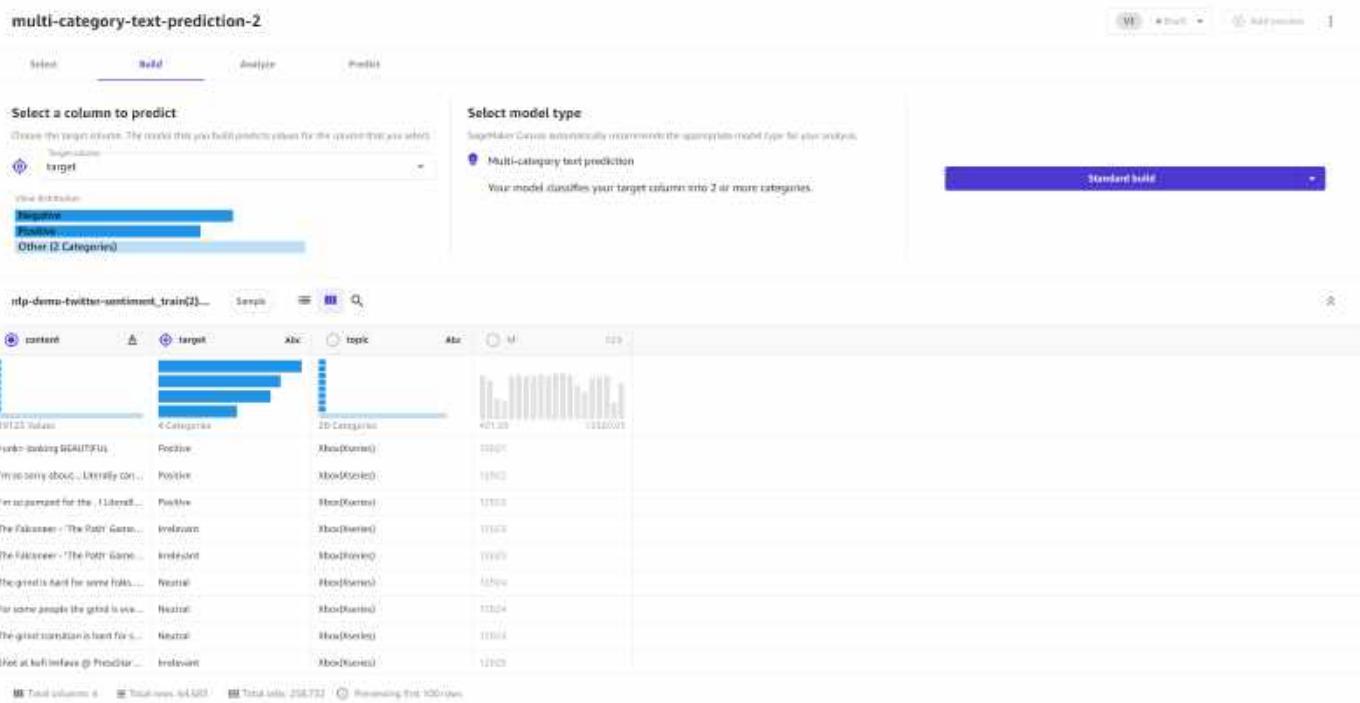
## Build a custom text prediction model

Multi-category text prediction models support both **Quick builds** and **Standard builds**.

To build a text prediction model, use the following procedure:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **My models**.
3. Choose **New model**.
4. In the **Create new model** dialog box, do the following:
  - a. Enter a name in the **Model name** field.
  - b. Select the **Text analysis** problem type.
  - c. Choose **Create**.
5. For **Select dataset**, select your dataset from the list of datasets. If you haven't already imported your data, choose **Import** to be directed through the import data workflow.
6. When you're ready to begin building your model, choose **Select dataset**.

7. On the **Build** tab, for the **Target column** dropdown list, select the target for your model that you would like to predict. The target column must have a binary or categorical data type, and there must be at least 25 entries (or rows of data) for each unique label in the target column.
8. For **Model type**, confirm that the model type is automatically set to **Multi-category text prediction**.
9. For the training column, select your source column of text data. This should be the column containing the text that you want to analyze.
10. Choose **Quick build** or **Standard build** to begin building your model. The following screenshot shows the **Build** page of a text prediction model that is ready to be built.



After your model begins building, you can leave the page. When the model shows as **Ready** on the **My models** page, it's ready for analysis and predictions.

## Build a time series forecasting model

Time series forecasting models support both **Quick builds** and **Standard builds**.

To build a time series forecasting model, use the following procedure:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **My models**.

3. Choose **New model**.
4. In the **Create new model** dialog box, do the following:
  - a. Enter a name in the **Model name** field.
  - b. Select the **Time series forecasting** problem type.
  - c. Choose **Create**.
5. For **Select dataset**, select your dataset from the list of datasets. If you haven't already imported your data, choose **Import** to be directed through the import data workflow.
6. When you're ready to begin building your model, choose **Select dataset**.
7. On the **Build** tab, for the **Target column** dropdown list, select the target for your model that you would like to predict.
8. In the **Model type** section, choose **Configure model**.
9. The **Configure model** box opens. For the **Time series configuration** section, fill out the following fields:
  - a. For **Item ID column**, choose a column in your dataset that uniquely identifies each row.
  - b. (Optional) For **Group column**, choose one or more categorical columns that you want to use for grouping your forecasting values.
  - c. For **Time stamp column**, select the column with timestamps (in datetime format). For more information about the accepted datetime formats, see [Time Series Forecasts in Amazon SageMaker Canvas](#).
  - d. For the **Forecast length** field, enter the period of time for which you want to forecast values. Canvas automatically detects the units of time in your data.
  - e. (Optional) Turn on the **Use holiday schedule** toggle to select a holiday schedule from various countries and make your forecasts with holiday data more accurate.
10. In the **Configure model** box, there are additional settings in the **Advanced** section. For more information about each of the advanced settings, see [Advanced model building configurations](#). To configure the **Advanced** settings, do the following:
  - a. For the **Objective metric** dropdown menu, select the metric that you want Canvas to optimize while building your model. If you don't select a metric, Canvas chooses one for you by default. For descriptions of the available metrics, see [Metrics reference](#).
  - b. If you're running a standard build, you'll see the **Algorithms** section. This section is for selecting the time series forecasting algorithms that you'd like to use for building your

model. You can select a subset of the available algorithms, or you can select all of them if you aren't sure which ones to try.

When you run your standard build, Canvas builds an ensemble model that combines all of the algorithms together to optimize prediction accuracy.

 **Note**

If you're running a quick build, Canvas uses a single tree-based learning algorithm to train your model, and you don't have to select any algorithms.

- c. For **Forecast quantiles**, enter up to 5 comma-separated quantile values to specify the upper and lower bounds of your forecast.
  - d. After configuring the **Advanced** settings, choose **Save**.
11. Select or deselect columns in your data to include or drop them from your build.
-  **Note**
- If you make batch predictions with your model after building, Canvas adds dropped columns to your prediction results. However, Canvas does not add the dropped columns to your batch predictions for time series models.
12. (Optional) Use the visualization and analytics tools that Canvas provides to visualize your data and determine which features you might want to include in your model. For more information, see [Explore and analyze your data](#).
  13. (Optional) Use data transformations to clean, transform, and prepare your data for model building. For more information, see [Prepare your data with advanced transformations](#). You can view and remove your transforms by choosing **Model recipe** to open the **Model recipe** side panel.
  14. (Optional) For additional features such as previewing the accuracy of your model, validating your dataset, and changing the size of the random sample that Canvas takes from your dataset, see [Preview your model](#).
  15. After reviewing your data and making any changes to your dataset, choose **Quick build** or **Standard build** to begin a build for your model.

After your model begins building, you can leave the page. When the model shows as **Ready** on the **My models** page, it's ready for analysis and predictions.

## Advanced model building configurations

Amazon SageMaker Canvas supports various advanced settings that you can configure when building a model. The following page lists all of the advanced settings along with additional information about their options and configurations.

### Note

The following advanced settings are currently only supported for numeric, categorical, and time series forecasting model types.

## Advanced numeric and categorical prediction model settings

Canvas supports the following advanced settings for numeric and categorical prediction model types.

### Objective metric

The objective metric is the metric that you want Canvas to optimize while building your model. If you don't select a metric, Canvas chooses one for you by default. For descriptions of the available metrics, see the [Metrics reference](#).

### Training method

Canvas can automatically select the training method based on the dataset size, or you can select it manually. The following training methods are available for you to choose from:

- **Ensembling** – SageMaker AI leverages the AutoGluon library to train several base models. To find the best combination for your dataset, ensemble mode runs 5–10 trials with different model and meta parameter settings. Then, these models are combined using a stacking ensemble method to create an optimal predictive model. For a list of algorithms supported by ensemble mode for tabular data, see the following [Algorithms](#) section.
- **Hyperparameter optimization (HPO)** – SageMaker AI finds the best version of a model by tuning hyperparameters using Bayesian optimization or multi-fidelity optimization while running training jobs on your dataset. HPO mode selects the algorithms that are most relevant to your dataset and selects the best range of hyperparameters to tune your models. To tune your

models, HPO mode runs up to 100 trials (default) to find the optimal hyperparameters settings within the selected range. If your dataset size is less than 100 MB, SageMaker AI uses Bayesian optimization. SageMaker AI chooses multi-fidelity optimization if your dataset is larger than 100 MB.

For a list of algorithms supported by HPO mode for tabular data, see the following [Algorithms](#) section.

- **Auto** – SageMaker AI automatically chooses either ensembling mode or HPO mode based on your dataset size. If your dataset is larger than 100 MB, SageMaker AI chooses HPO mode. Otherwise, it chooses ensembling mode.

## Algorithms

In **Ensembling** mode, Canvas supports the following machine learning algorithms:

- [LightGBM](#) – An optimized framework that uses tree-based algorithms with gradient boosting. This algorithm uses trees that grow in breadth, rather than depth, and is highly optimized for speed.
- [CatBoost](#) – A framework that uses tree-based algorithms with gradient boosting. Optimized for handling categorical variables.
- [XGBoost](#) – A framework that uses tree-based algorithms with gradient boosting that grows in depth, rather than breadth.
- [Random Forest](#) – A tree-based algorithm that uses several decision trees on random sub-samples of the data with replacement. The trees are split into optimal nodes at each level. The decisions of each tree are averaged together to prevent overfitting and improve predictions.
- [Extra Trees](#) – A tree-based algorithm that uses several decision trees on the entire dataset. The trees are split randomly at each level. The decisions of each tree are averaged to prevent overfitting and to improve predictions. Extra trees add a degree of randomization in comparison to the random forest algorithm.
- [Linear Models](#) – A framework that uses a linear equation to model the relationship between two variables in observed data.
- Neural network torch – A neural network model that's implemented using [Pytorch](#).
- Neural network fast.ai – A neural network model that's implemented using [fast.ai](#).

In **HPO mode**, Canvas supports the following machine learning algorithms:

- [XGBoost](#) – A supervised learning algorithm that attempts to accurately predict a target variable by combining an ensemble of estimates from a set of simpler and weaker models.
- Deep learning algorithm – A multilayer perceptron (MLP) and feedforward artificial neural network. This algorithm can handle data that is not linearly separable.

## Data split

You have the option to specify how you want to split your dataset between the training set (the portion of your dataset used for building the model) and the validation set, (the portion of your dataset used for verifying the model's accuracy). For example, a common split ratio is 80% training and 20% validation, where 80% of your data is used to build the model while 20% is saved for measuring model performance. If you don't specify a custom ratio, then Canvas splits your dataset automatically.

## Max candidates



This feature is only available in the HPO training mode.

You can specify the maximum number of model candidates that Canvas generates while building your model. We recommend that you use the default number of candidates, which is 100, to build the most accurate models. The maximum number you can specify is 250. Decreasing the number of model candidates may impact your model's accuracy.

## Max job runtime

You can specify the maximum job runtime, or the maximum amount of time that Canvas spends building your model. After the time limit, Canvas stops building and selects the best model candidate.

The maximum time that you can specify is 720 hours. We highly recommend that you keep the maximum job runtime greater than 30 minutes to ensure that Canvas has enough time to generate model candidates and finish building your model.

## Advanced time series forecasting model settings

For time series forecasting models, Canvas supports the Objective metric, which is listed in the previous section.

Time series forecasting models also support the following advanced setting:

## Algorithm selection

When you build a time series forecasting model, Canvas uses an *ensemble* (or a combination) of statistical and machine learning algorithms to deliver highly accurate time series forecasts. By default, Canvas selects the optimal combination of all the available algorithms based on the time series in your dataset. However, you have the option to specify one or more algorithms to use for your forecasting model. In this case, Canvas determines the best blend using only your selected algorithms. If you're uncertain about which algorithm to select for training your model, we recommend that you choose all of the available algorithms.

### Note

Algorithm selection is only supported for standard builds. If you don't select any algorithms in the advanced settings, then by default SageMaker AI runs a quick build and trains model candidates using a single tree-based learning algorithm. For more information about the difference between quick builds and standard builds, see [How custom models work](#).

Canvas supports the following time series forecasting algorithms:

- [\*\*Autoregressive Integrated Moving Average \(ARIMA\)\*\*](#) – A simple stochastic time series model that uses statistical analysis to interpret the data and make future predictions. This algorithm is useful for simple datasets with fewer than 100 time series.
- [\*\*Convolutional Neural Network - Quantile Regression \(CNN-QR\)\*\*](#) – A proprietary, supervised learning algorithm that trains one global model from a large collection of time series and uses a quantile decoder to make predictions. CNN-QR works best with large datasets containing hundreds of time series.
- [\*\*DeepAR+\*\*](#) – A proprietary, supervised learning algorithm for forecasting scalar time series using recurrent neural networks (RNNs) to train a single model jointly over all of the time series. DeepAR+ works best with large datasets containing hundreds of feature time series.
- [\*\*Non-Parametric Time Series \(NPTS\)\*\*](#) – A scalable, probabilistic baseline forecaster that predicts the future value distribution of a given time series by sampling from past observations. NPTS is useful when working with sparse or intermittent time series (for example, forecasting demand for individual items where the time series has many 0s or low counts).
- [\*\*Exponential Smoothing \(ETS\)\*\*](#) – A forecasting method that produces forecasts which are weighted averages of past observations where the weights of older observations exponentially decrease.

The algorithm is useful for simple datasets with fewer than 100 time series and datasets with seasonality patterns.

- [Prophet](#) – An additive regression model that works best with time series that have strong seasonal effects and several seasons of historical data. The algorithm is useful for datasets with non-linear growth trends that approach a limit.

## Forecast quantiles

For time series forecasting, SageMaker AI trains 6 model candidates with your target time series. Then, SageMaker AI combines these models using a stacking ensemble method to create an optimal forecasting model for a given objective metric. Each forecasting model generates a probabilistic forecast by producing forecasts at quantiles between P1 and P99. These quantiles are used to account for forecast uncertainty. By default, forecasts are generated for 0.1 (p10), 0.5 (p50), and 0.9 (p90). You can choose to specify up to five of your own quantiles from 0.01 (p1) to 0.99 (p99), by increments of 0.01 or higher.

## Edit an image dataset

In Amazon SageMaker Canvas, you can edit your image datasets and review your labels before building a model. You might want to perform tasks such as assigning labels to unlabeled images or adding more images to the dataset. These tasks can all be done in the Canvas application, providing you with one place to modify your dataset and build a model.

### Note

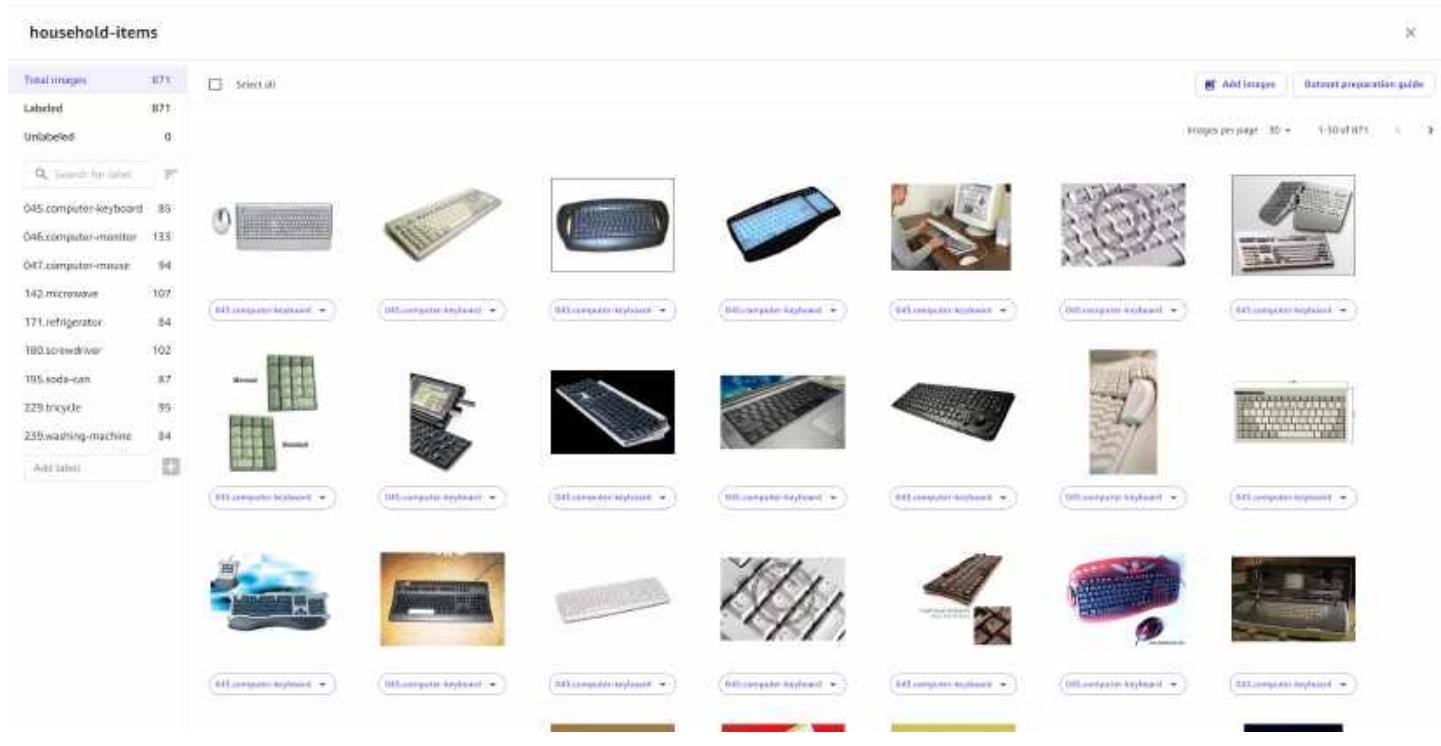
Before building a model, you must assign labels to all images in your dataset. Also, you must have at least 25 images per label and a minimum of two labels. For more information about assigning labels, see the section on this page called **Assign labels to unlabeled images**. If you can't determine a label for an image, you should delete it from your dataset. For more information about deleting images, see the section on this page [Add or delete images from the dataset](#).

To begin editing your image dataset, you should be on the **Build** tab while building your single-label image prediction model.

A new page opens that shows the images in your dataset along with their labels. This page categorizes your image dataset into **Total images**, **Labeled images**, and **Unlabeled images**. You

can also review the **Dataset preparation guide** for best practices on building a more accurate image prediction model.

The following screenshot shows the page for editing your image dataset.



From this page, you can do the following actions.

### View the properties for each image (label, size, dimensions)

To view an individual image, you can search for it by file name in the search bar. Then, choose the image to open the full view. You can view the image properties and reassign the image's label. Choose **Save** when you're doing viewing the image.

### Add, rename, or delete labels in the dataset

Canvas lists the labels for your dataset in the left navigation pane. You can add new labels to the dataset by entering a label in the **Add label** text field.

To rename or delete a label from your dataset, choose the **More options** icon

( ! )

next to the label and select either **Rename** or **Delete**. If you rename the label, you can enter the new label name and choose **Confirm**. If you delete the label, the label is removed from all images in your dataset that have that label. Any images with that label are left unlabeled.

## Assign labels to unlabeled images

To view the unlabeled images in your dataset, choose **Unlabeled** in the left navigation pane. For each image, select it and open the label titled **Unlabeled** and select a label to assign to the image from the dropdown list. You can also select more than one image and perform this action, and all selected images are assigned the label you chose.

## Reassign labels to images

You can reassign labels to images by selecting the image (or multiple images at a time) and opening the dropdown titled with the current label. Select your desired label, and the image or images are updated with the new label.

## Sort your images by label

You can view all the images for a given label by choosing the label in the left navigation pane.

## Add or delete images from the dataset

You can add more images to your dataset by choosing **Add images** in the top navigation pane. You'll be taken through the workflow to import more images. The images you import are added to your existing dataset.

You can delete images from your dataset by selecting them and then choosing **Delete** in the top navigation pane.

### Note

After making any changes to your dataset, choose **Save dataset** to make sure that you don't lose your changes.

## Data exploration and analysis

### Note

You can only use SageMaker Canvas visualizations and analytics for models built on tabular datasets. Multi-category text prediction models are also excluded.

In Amazon SageMaker Canvas, you can explore the variables in your dataset using visualizations and analytics and create in-application visualizations and analytics. You can use these explorations to uncover relationships between your variables before building your model.

For more information about visualization techniques in Canvas, see [Explore your data using visualization techniques](#).

For more information about analytics in Canvas, see [Explore your data using analytics](#).

## Explore your data using visualization techniques

### Note

You can only use SageMaker Canvas visualizations for models built on tabular datasets. Multi-category text prediction models are also excluded.

With Amazon SageMaker Canvas, you can explore and visualize your data to gain advanced insights into your data before building your ML models. You can visualize using scatter plots, bar charts, and box plots, which can help you understand your data and discover the relationships between features that could affect the model accuracy.

In the **Build** tab of the SageMaker Canvas application, choose **Data visualizer** to begin creating your visualizations.

You can change the visualization sample size to adjust the size of the random sample taken from your dataset. A sample size that is too large might affect the performance of your data visualizations, so we recommend that you choose an appropriate sample size. To change the sample size, use the following procedure.

1. Choose **Visualization sample**.
2. Use the slider to select your desired sample size.
3. Choose **Update** to confirm the change to your sample size.

### Note

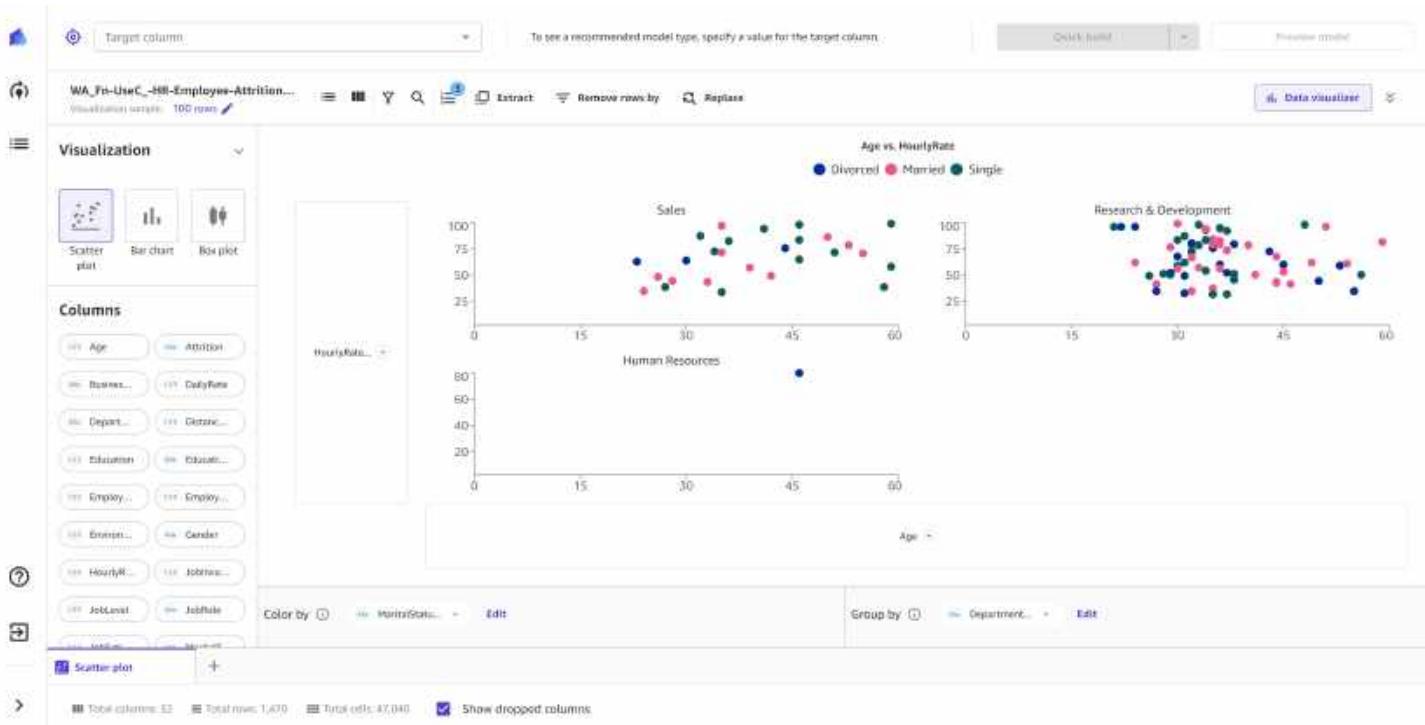
Certain visualization techniques require columns of a specific data type. For example, you can only use numeric columns for the x and y-axes of scatter plots.

## Scatter plot

To create a scatter plot with your dataset, choose **Scatter plot** in the **Visualization** panel. Choose the features you want to plot on the x and y-axes from the **Columns** section. You can drag and drop the columns onto the axes or, once an axis has been dropped, you can choose a column from the list of supported columns.

You can use **Color by** to color the data points on the plot with a third feature. You can also use **Group by** to group the data into separate plots based on a fourth feature.

The following image shows a scatter plot that uses **Color by** and **Group by**. In this example, each data point is colored by the `MaritalStatus` feature, and grouping by the `Department` feature results in a scatter plot for the data points of each department.

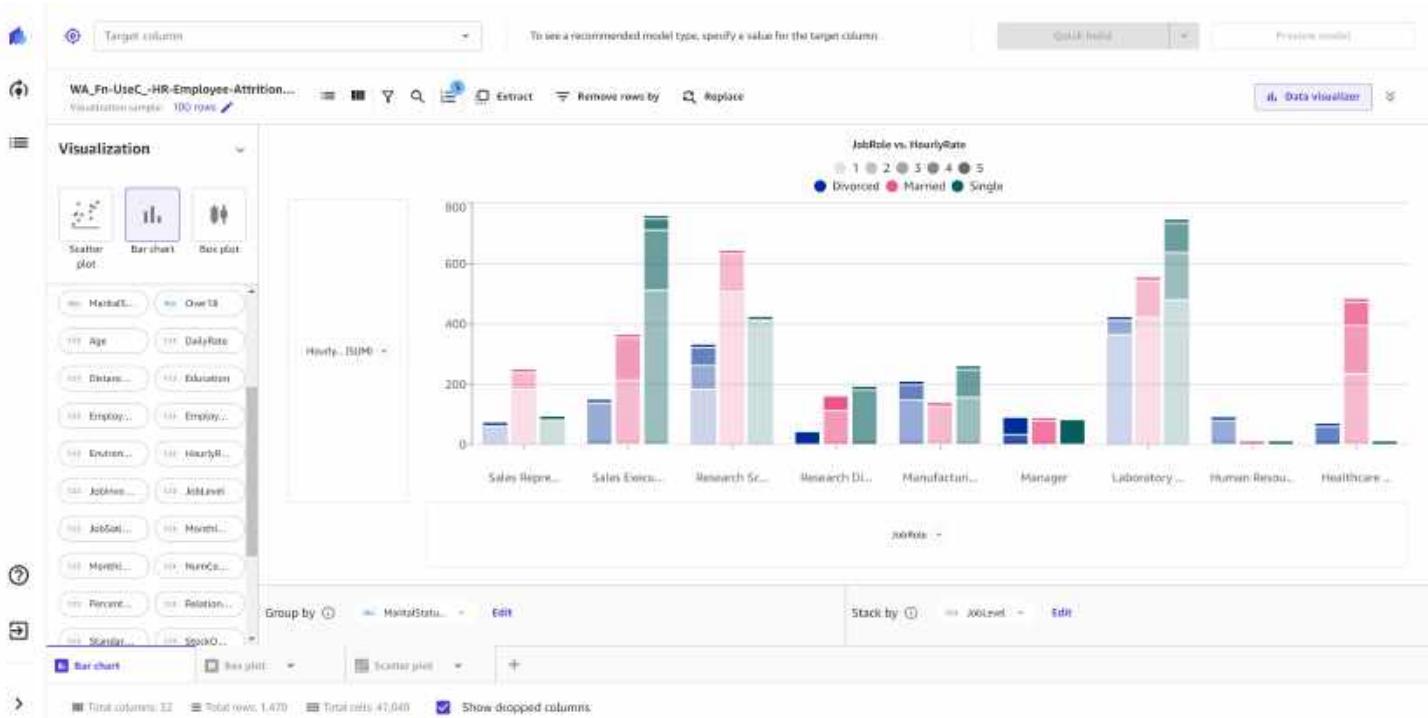


## Bar chart

To create a bar chart with your dataset, choose **Bar chart** in the **Visualization** panel. Choose the features you want to plot on the x and y-axes from the **Columns** section. You can drag and drop the columns onto the axes or, once an axis has been dropped, you can choose a column from the list of supported columns.

You can use **Group by** to group the bar chart by a third feature. You can use **Stack by** to vertically shade each bar based on the unique values of a fourth feature.

The following image shows a bar chart that uses **Group by** and **Stack by**. In this example, the bar chart is grouped by the MaritalStatus feature and stacked by the JobLevel feature. For each JobRole on the x axis, there is a separate bar for the unique categories in the MaritalStatus feature, and every bar is vertically stacked by the JobLevel feature.

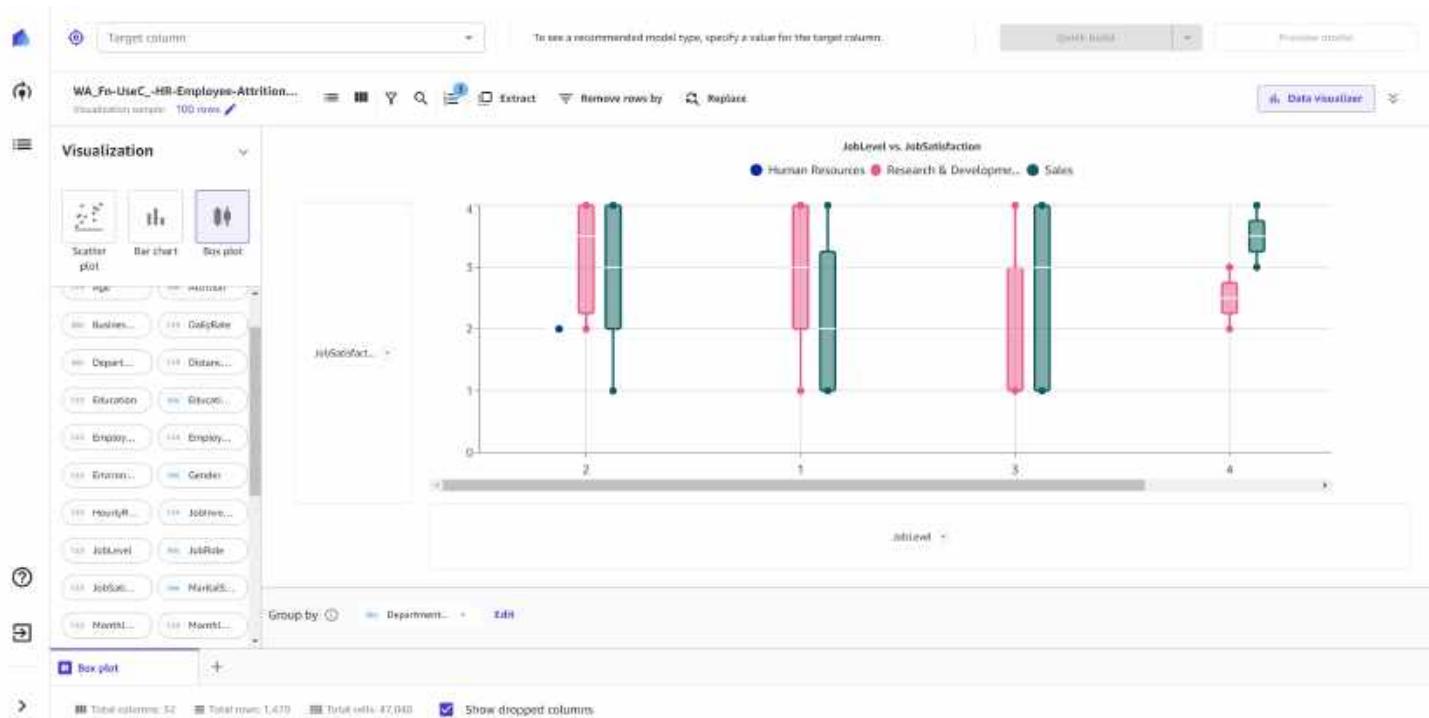


## Box plot

To create a box plot with your dataset, choose **Box plot** in the **Visualization** panel. Choose the features you want to plot on the x and y-axes from the **Columns** section. You can drag and drop the columns onto the axes or, once an axis has been dropped, you can choose a column from the list of supported columns.

You can use **Group by** to group the box plots by a third feature.

The following image shows a box plot that uses **Group by**. In this example, the x and y-axes show JobLevel and JobSatisfaction, respectively, and the colored box plots are grouped by the Department feature.



## Explore your data using analytics

### **Note**

You can only use SageMaker Canvas analytics for models built on tabular datasets. Multi-category text prediction models are also excluded.

With analytics in Amazon SageMaker Canvas, you can explore your dataset and gain insight on all of your variables before building a model. You can determine the relationships between features in your dataset using correlation matrices. You can use this technique to summarize your dataset into a matrix that shows the correlations between two or more values. This helps you identify and visualize patterns in a given dataset for advanced data analysis.

The matrix shows the correlation between each feature as positive, negative, or neutral. You might want to include features that have a high correlation with each other when building your model. Features that have little to no correlation might be irrelevant to your model, and you can drop those features when building your model.

To get started with correlation matrices in SageMaker Canvas, see the following section.

## Create a correlation matrix

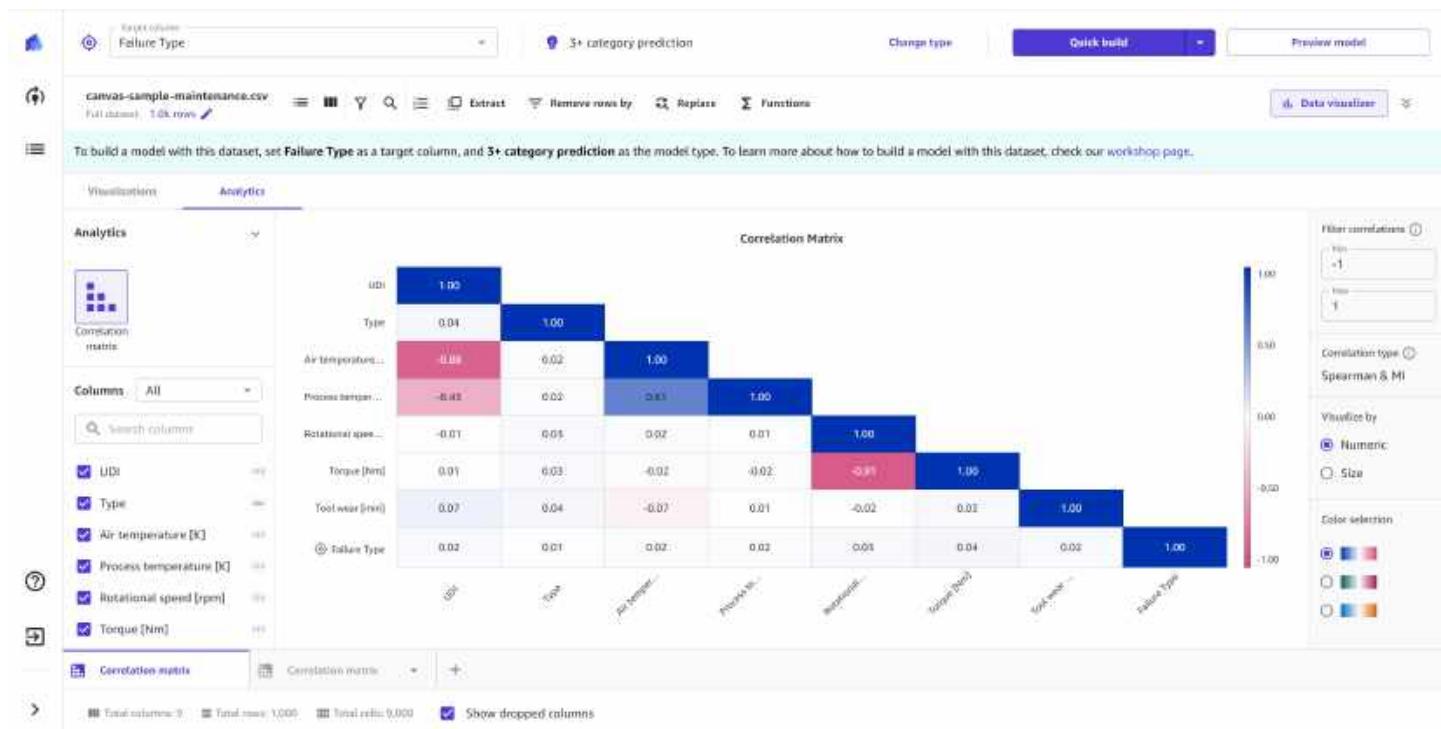
You can create a correlation matrix when you are preparing to build a model in the **Build** tab of the SageMaker Canvas application.

For instructions on how to begin creating a model, see [Build a model](#).

After you've started preparing a model in the SageMaker Canvas application, do the following:

1. In the **Build** tab, choose **Data visualizer**.
2. Choose **Analytics**.
3. Choose **Correlation matrix**.

You should see a visualization similar to the following screenshot, which shows up to 15 columns of the dataset organized into a correlation matrix.



After you've created the correlation matrix, you can customize it by doing the following:

### 1. Choose your columns

For **Columns**, you can select the columns that you want to include in the matrix. You can compare up to 15 columns from your dataset.

### Note

You can use numeric, categorical, or binary column types for a correlation matrix. The correlation matrix doesn't support datetime or text data column types.

To add or remove columns from the correlation matrix, select and deselect columns from the **Columns** panel. You can also drag and drop columns from the panel directly onto the matrix. If your dataset has a lot of columns, you can search for the columns you want in the **Search columns** bar.

To filter the columns by data type, choose the dropdown list and select **All**, **Numeric**, or **Categorical**. Selecting **All** shows you all of the columns from your dataset, whereas the **Numeric** and **Categorical** filters only show you the numeric or categorical columns in your dataset. Note that binary column types are included in the numeric or categorical filters.

For the best data insights, include your target column in the correlation matrix. When you include your target column in the correlation matrix, it appears as the last feature on the matrix with a target symbol.

## 2. Choose your correlation type

SageMaker Canvas supports different *correlation types*, or methods for calculating the correlation between your columns.

To change the correlation type, use the **Columns** filter mentioned in the preceding section to filter for your desired column type and columns. You should see the **Correlation type** in the side panel. For numeric comparisons, you have the option to select either **Pearson** or **Spearman**. For categorical comparisons, the correlation type is set as **MI**. For categorical and mixed comparisons, the correlation type is set as **Spearman & MI**.

For matrices that only compare numeric columns, the correlation type is either Pearson or Spearman. The Pearson measure evaluates the linear relationship between two continuous variables. The Spearman measure evaluates the monotonic relationship between two variables. For both Pearson and Spearman, the scale of correlation ranges from -1 to 1, with either end of the scale indicating a perfect correlation (a direct 1:1 relationship) and 0 indicating no correlation. You might want to select Pearson if your data has more linear relationships (as revealed by a [scatter plot visualization](#)). If your data is not linear, or contains a mixture of linear and monotonic relationships, then you might want to select Spearman.

For matrices that only compare categorical columns, the correlation type is set to Mutual Information Classification (MI). The MI value is a measure of the mutual dependence between two random variables. The MI measure is on a scale of 0 to 1, with 0 indicating no correlation and 1 indicating a perfect correlation.

For matrices that compare a mix of numeric and categorical columns, the correlation type **Spearman & MI** is a combination of the Spearman and MI correlation types. For correlations between two numeric columns, the matrix shows the Spearman value. For correlations between a numeric and categorical column or two categorical columns, the matrix shows the MI value.

Lastly, remember that correlation does not necessarily indicate causation. A strong correlation value only indicates that there is a relationship between two variables, but the variables might not have a causal relationship. Carefully review your columns of interest to avoid bias when building your model.

### 3. Filter your correlations

In the side panel, you can use the **Filter correlations** feature to filter for the range of correlation values that you want to include in the matrix. For example, if you want to filter for features that only have positive or neutral correlation, you can set the **Min** to 0 and the **Max** to 1 (valid values are -1 to 1).

For Spearman and Pearson comparisons, you can set the **Filter correlations** range anywhere from -1 to 1, with 0 meaning that there is no correlation. -1 and 1 mean that the variables have a strong negative or positive correlation, respectively.

For MI comparisons, the correlation range only goes from 0 to 1, with 0 meaning that there is no correlation and 1 meaning that the variables have a strong correlation, either positive or negative.

Each feature has a perfect correlation (1) with itself. Therefore, you might notice that the top row of the correlation matrix is always 1. If you want to exclude these values, you can use the filter to set the **Max** less than 1.

Keep in mind that if your matrix compares a mix of numeric and categorical columns and uses the **Spearman & MI** correlation type, then the *categorical x numeric* and *categorical x categorical* correlations (which use the MI measure) are on a scale of 0 to 1, whereas the *numeric x numeric* correlations (which use the Spearman measure) are on a scale of -1 to 1. Review your correlations of interest carefully to ensure that you know the correlation type being used to calculate each value.

## 4. Choose the visualization method

In the side panel, you can use **Visualize by** to change the visualization method of the matrix. Choose the **Numeric** visualization method to show the correlation (Pearson, Spearman, or MI) value, or choose the **Size** visualization method to visualize the correlation with differently sized and colored dots. If you choose **Size**, you can hover over a specific dot on the matrix to see the actual correlation value.

## 5. Choose a color palette

In the side panel, you can use **Color selection** to change the color palette used for the scale of negative to positive correlation in the matrix. Select one of the alternative color palettes to change the colors used in the matrix.

## Prepare data for model building

### Note

You can now do advanced data preparation in SageMaker Canvas with Data Wrangler, which provides you with a natural language interface and over 300 built-in transformations. For more information, see [Data preparation](#).

Your machine learning dataset might require data preparation before you build your model. You might want to clean your data due to various issues, which might include missing values or outliers, and perform feature engineering to improve the accuracy of your model. Amazon SageMaker Canvas provides ML data transforms with which you can clean, transform, and prepare your data for model building. You can use these transforms on your datasets without any code. SageMaker Canvas adds the transforms you use to the **Model recipe**, which is a record of the data preparation done on your data before building the model. Any data transforms you use only modify the input data for model building and do not modify your original data source.

The preview of your dataset shows the first 100 rows of the dataset. If your dataset has more than 20,000 rows, Canvas takes a random sample of 20,000 rows and previews the first 100 rows from that sample. You can only search for and specify values from the previewed rows, and the filter functionality only filters the previewed rows and not the entire dataset.

The following transforms are available in SageMaker Canvas for you to prepare your data for building.

### Note

You can only use advanced transformations for models built on tabular datasets. Multi-category text prediction models are also excluded.

## Drop columns

You can exclude a column from your model build by dropping it in the **Build** tab of the SageMaker Canvas application. Deselect the column you want to drop, and it isn't included when building the model.

### Note

If you drop columns and then make [batch predictions](#) with your model, SageMaker Canvas adds the dropped columns back to the output dataset available for you to download. However, SageMaker Canvas does not add the dropped columns back for time series models.

## Filter rows

The filter functionality filters the previewed rows (the first 100 rows of your dataset) according to conditions that you specify. Filtering rows creates a temporary preview of the data and does not impact the model building. You can filter to preview rows that have missing values, contain outliers, or meet custom conditions in a column you choose.

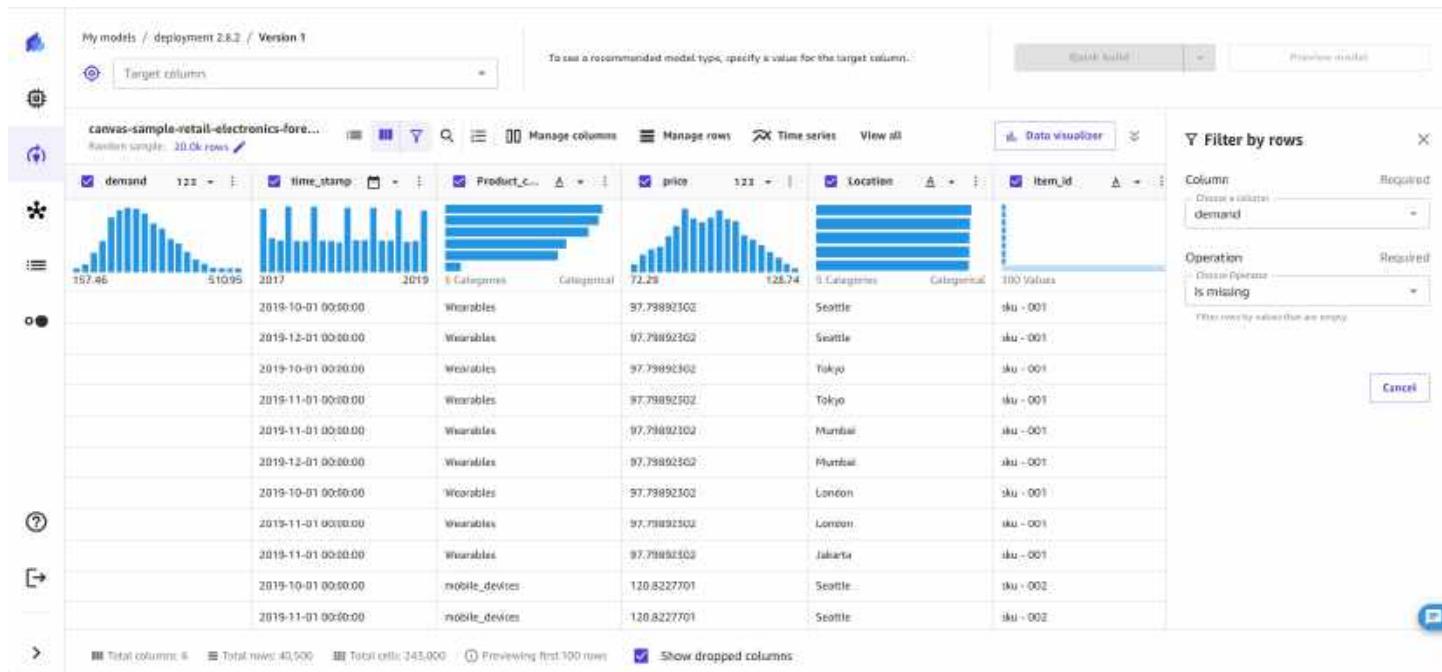
### Filter rows by missing values

Missing values are a common occurrence in machine learning datasets. If you have rows with null or empty values in certain columns, you might want to filter for and preview those rows.

To filter missing values from your previewed data, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Filter by rows** (▽).
2. Choose the **Column** you want to check for missing values.
3. For the **Operation**, choose **Is missing**.

SageMaker Canvas filters for rows that contain missing values in the **Column** you selected and provides a preview of the filtered rows.



## Filter rows by outliers

Outliers, or rare values in the distribution and range of your data, can negatively impact model accuracy and lead to longer building times. SageMaker Canvas enables you to detect and filter rows that contain outliers in numeric columns. You can choose to define outliers with either standard deviations or a custom range.

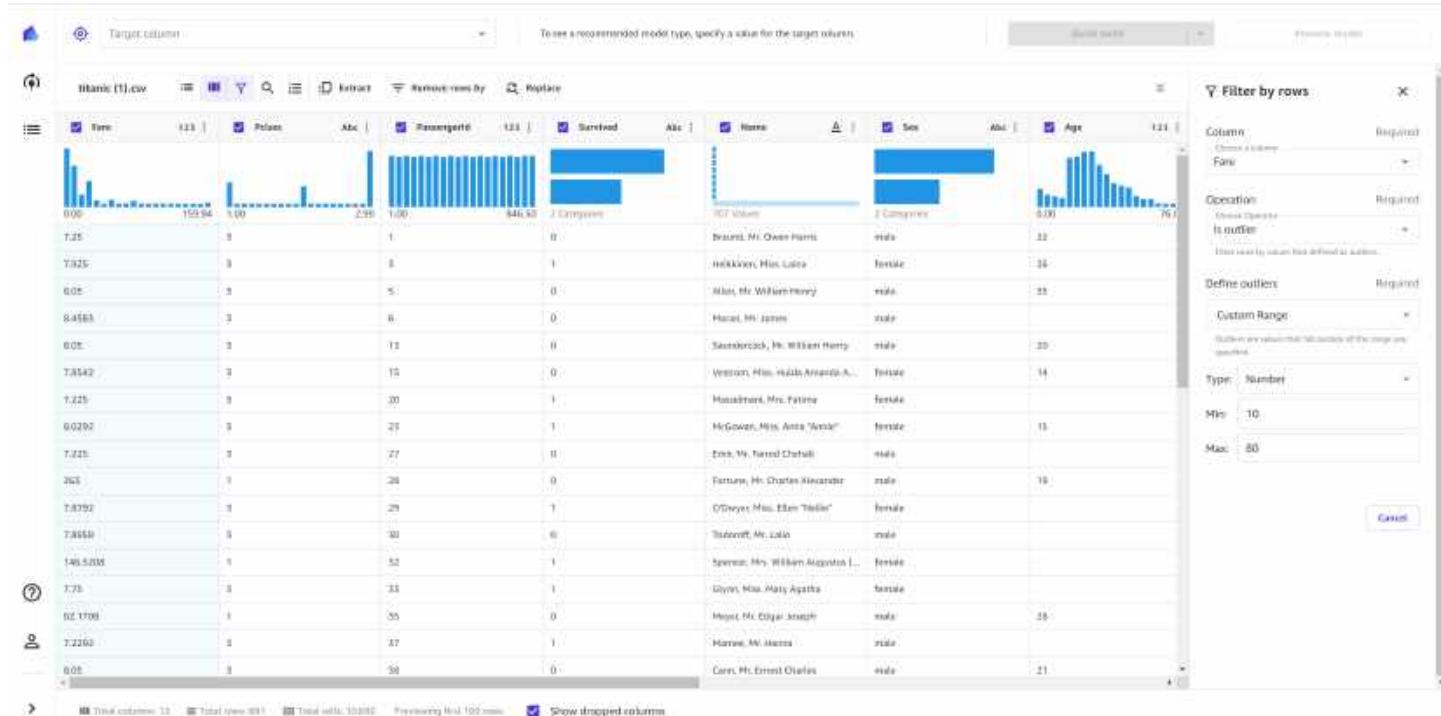
To filter for outliers in your data, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Filter by rows** ( ).
2. Choose the **Column** you want to check for outliers.
3. For the **Operation**, choose **Is outlier**.
4. Set the **Outlier range** to either **Standard deviation** or **Custom range**.
5. If you choose **Standard deviation**, specify a **SD** (standard deviation) value from 1–3. If you choose **Custom range**, select either **Percentile** or **Number**, and then specify the **Min** and **Max** values.

The **Standard deviation** option detects and filters for outliers in numeric columns using the mean and standard deviation. You specify the number of standard deviations a value must vary from the

mean to be considered an outlier. For example, if you specify 3 for **SD**, a value must fall more than 3 standard deviations from the mean to be considered an outlier.

The **Custom range** option detects and filters for outliers in numeric columns using minimum and maximum values. Use this method if you know your threshold values that delimit outliers. You can set the **Type** of the range to either **Percentile** or **Number**. If you choose **Percentile**, the **Min** and **Max** values should be the minimum and maximum of the percentile range (0-100) that you want to allow. If you choose **Number**, the **Min** and **Max** values should be the minimum and maximum numeric values that you want to filter in the data.



## Filter rows by custom values

You can filter for rows with values that meet custom conditions. For example, you might want to preview rows that have a price value greater than 100 before removing them. With this functionality, you can filter rows that exceed the threshold you set and preview the filtered data.

To use the custom filter functionality, do the following.

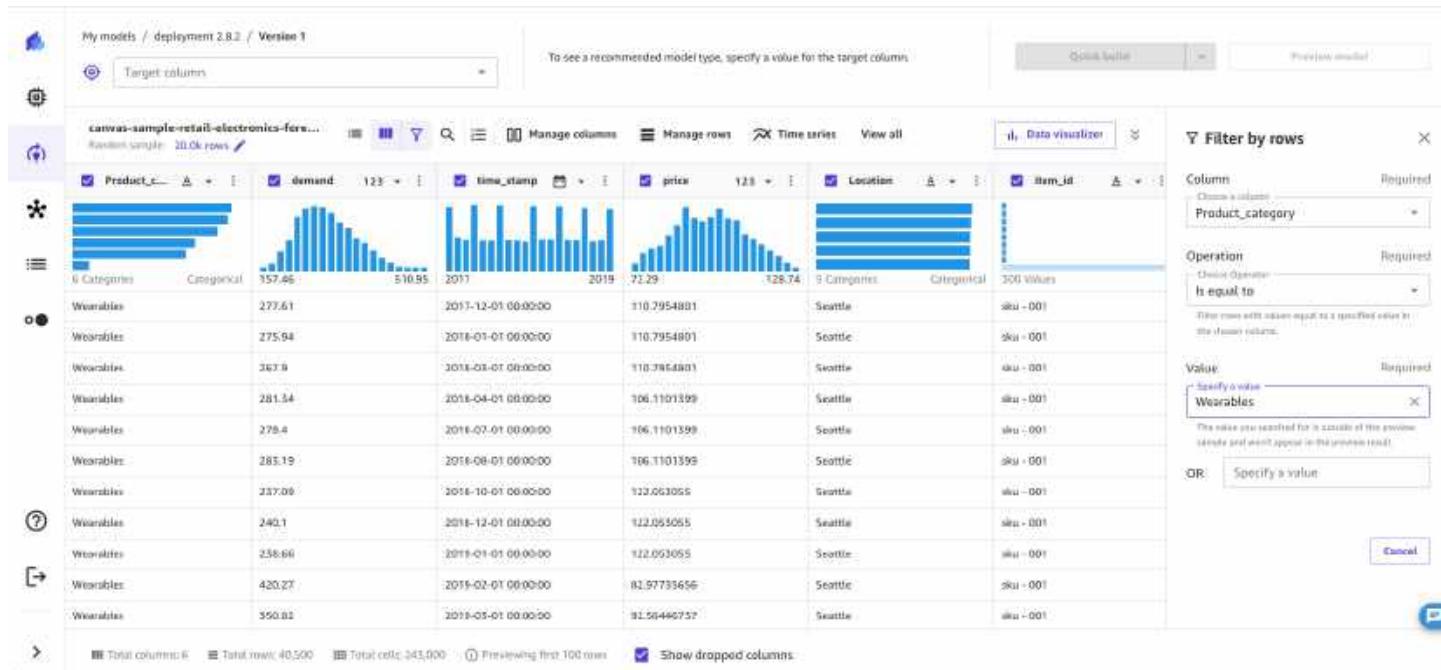
1. In the **Build** tab of the SageMaker Canvas application, choose **Filter by rows** (Y).
2. Choose the **Column** you want to check.
3. Select the type of **Operation** you want to use, and then specify the values for the selected condition.

For the **Operation**, you can choose one of the following options. Note that the available operations depend on the data type of the column you choose. For example, you cannot create a `is greater than` operation for a column containing text values.

Operation	Supported data type	Supported feature type	Function
Is equal to	Numeric, Text	Binary, Categorical	Filters rows where the value in <b>Column</b> equals the values you specify.
Is not equal to	Numeric, Text	Binary, Categorical	Filters rows where the value in <b>Column</b> doesn't equal the values you specify.
Is less than	Numeric	N/A	Filters rows where the value in <b>Column</b> is less than the value you specify.
Is less than or equal to	Numeric	N/A	Filters rows where the value in <b>Column</b> is less than or equal to the value you specify.
Is greater than	Numeric	N/A	Filters rows where the value in <b>Column</b> is greater than the value you specify.
Is greater than or equal to	Numeric	N/A	Filters rows where the value in <b>Column</b> is greater than or equal to the value you specify.
Is between	Numeric	N/A	Filters rows where the value in <b>Column</b> is between or equal to two values you specify.
Contains	Text	Categorical	Filters rows where the value in <b>Column</b> contains a values you specify.

Operation	Supported data type	Supported feature type	Function
Starts with	Text	Categorical	Filters rows where the value in <b>Column</b> begins with a value you specify.
Ends with	Categorical	Categorical	Filters rows where the value in <b>Column</b> ends with a value you specify.

After you set the filter operation, SageMaker Canvas updates the preview of the dataset to show you the filtered data.



## Functions and operators

You can use mathematical functions and operators to explore and distribute your data. You can use the SageMaker Canvas supported functions or create your own formula with your existing data and create a new column with the result of the formula. For example, you can add the corresponding values of two columns and save the result to a new column.

You can nest statements to create more complex functions. The following are some examples of nested functions that you might use.

- To calculate BMI, you could use the function `weight / (height ^ 2)`.
- To classify ages, you could use the function `Case(age < 18, 'child', age < 65, 'adult', 'senior')`.

You can specify functions in the data preparation stage before you build your model. To use a function, do the following.

- In the **Build** tab of the SageMaker Canvas application, choose **View all** and then choose **Custom formula** to open the **Custom formula** panel.
- In the **Custom formula** panel, you can choose a **Formula** to add to your **Model Recipe**. Each formula is applied to all of the values in the columns you specify. For formulas that accept two or more columns as arguments, use columns with matching data types; otherwise, you get an error or null values in the new column.
- After you've specified a **Formula**, add a column name in the **New Column Name** field. SageMaker Canvas uses this name for the new column that is created.
- (Optional) Choose **Preview** to preview your transform.
- To add the function to your **Model Recipe**, choose **Add**.

SageMaker Canvas saves the result of your function to a new column using the name you specified in **New Column Name**. You can view or remove functions from the **Model Recipe** panel.

SageMaker Canvas supports the following operators for functions. You can use either the text format or the in-line format to specify your function.

Operator	Description	Supported data types	Text format	In-line format
Add	Returns the sum of the values	Numeric	<code>Add(sales1, sales2)</code>	<code>sales1 + sales2</code>
Subtract	Returns the difference between the values	Numeric	<code>Subtract(sales1, sales2)</code>	<code>sales1 - sales2</code>

Operator	Description	Supported data types	Text format	In-line format
Multiply	Returns the product of the values	Numeric	Multiply(sales1, sales2)	sales1 * sales2
Divide	Returns the quotient of the values	Numeric	Divide(sales1, sales2)	sales1 / sales2
Mod	Returns the result of the modulo operator (the remainder after dividing the two values)	Numeric	Mod(sales1, sales2)	sales1 % sales2
Abs	Returns the absolute value of the value	Numeric	Abs(sales1)	N/A
Negate	Returns the negative of the value	Numeric	Negate(c1)	-c1
Exp	Returns e (Euler's number) raised to the power of the value	Numeric	Exp(sales1)	N/A
Log	Returns the logarithm (base 10) of the value	Numeric	Log(sales1)	N/A
Ln	Returns the natural logarithm (base e) of the value	Numeric	Ln(sales1)	N/A
Pow	Returns the value raised to a power	Numeric	Pow(sales1, 2)	sales1 ^ 2
If	Returns a true or false label based on a condition you specify	Boolean, Numeric, Text	If(sales1 >7000, 'truelabel', 'falselabel')	N/A

Operator	Description	Supported data types	Text format	In-line format
Or	Returns a Boolean value of whether one of the specified values or conditions is true or not	Boolean	Or(fullprice, discount)	fullprice    discount
And	Returns a Boolean value of whether two of the specified values or conditions are true or not	Boolean	And(sales1,sales2)	sales1 && sales2
Not	Returns a Boolean value that is the opposite of the specified value or conditions	Boolean	Not(sales1)	!sales1
Case	Returns a Boolean value based on conditional statements (returns c1 if cond1 is true, returns c2 if cond2 is true, else returns c3)	Boolean, Numeric, Text	Case(cond1, c1, cond2, c2, c3)	N/A
Equal	Returns a Boolean value of whether two values are equal	Boolean, Numeric, Text	N/A	c1 = c2 c1 == c2
Not equal	Returns a Boolean value of whether two values are not equal	Boolean, Numeric, Text	N/A	c1 != c2
Less than	Returns a Boolean value of whether c1 is less than c2	Boolean, Numeric, Text	N/A	c1 < c2
Greater than	Returns a Boolean value of whether c1 is greater than c2	Boolean, Numeric, Text	N/A	c1 > c2

Operator	Description	Supported data types	Text format	In-line format
Less than or equal	Returns a Boolean value of whether c1 is less than or equal to c2	Boolean, Numeric, Text	N/A	c1 <= c2
Greater than or equal	Returns a Boolean value of whether c1 is greater than or equal to c2	Boolean, Numeric, Text	N/A	c1 >= c2

SageMaker Canvas also supports aggregate operators, which can perform operations such as calculating the sum of all the values or finding the minimum value in a column. You can use aggregate operators in combination with standard operators in your functions. For example, to calculate the difference of values from the mean, you could use the function `Abs(height - avg(height))`. SageMaker Canvas supports the following aggregate operators.

Aggregate operator	Description	Format	Example
sum	Returns the sum of all the values in a column	sum	sum(c1)
minimum	Returns the minimum value of a column	min	min(c2)
maximum	Returns the maximum value of a column	max	max(c3)
average	Returns the average value of a column	avg	avg(c4)
std	Returns the sample standard deviation of a column	std	std(c1)
stddev	Returns the standard deviation of the values in a column	stddev	stddev(c1)

Aggregate operator	Description	Format	Example
variance	Returns the unbiased variance of the values in a column	variance	variance(c1)
approx_count_distinct	Returns the approximate number of distinct items in a column	approx_count_distinct	approx_count_distinct(c1)
count	Returns the number of items in a column	count	count(c1)
first	Returns the first value of a column	first	first(c1)
last	Returns the last value of a column	last	last(c1)
stddev_pop	Returns the population standard deviation of a column	stddev_pop	stddev_pop(c1)
variance_pop	Returns the population variance of the values in a column	variance_pop	variance_pop(c1)

## Manage rows

With the Manage rows transform, you can perform sort, random shuffle, and remove rows of data from the dataset.

### Sort rows

To sort the rows in a dataset by a given column, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Manage rows** and then choose **Sort rows**.
2. For **Sort Column**, choose the column you want to sort by.
3. For **Sort Order**, choose either **Ascending** or **Descending**.
4. Choose **Add** to add the transform to the **Model recipe**.

## Shuffle rows

To randomly shuffle the rows in a dataset, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Manage rows** and then choose **Shuffle rows**.
2. Choose **Add** to add the transform to the **Model recipe**.

## Drop duplicate rows

To remove duplicate rows in a dataset, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Manage rows** and then choose **Drop duplicate rows**.
2. Choose **Add** to add the transform to the **Model recipe**.

## Remove rows by missing values

Missing values are a common occurrence in machine learning datasets and can impact model accuracy. Use this transform if you want to drop rows with null or empty values in certain columns.

To remove rows that contain missing values in a specified column, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Manage rows**.
2. Choose **Drop rows by missing values**.
3. Choose **Add** to add the transform to the **Model recipe**.

SageMaker Canvas drops rows that contain missing values in the **Column** you selected. After removing the rows from the dataset, SageMaker Canvas adds the transform in the **Model recipe** section. If you remove the transform from the **Model recipe** section, the rows return to your dataset.

The screenshot shows the SageMaker Canvas interface. On the left, there's a sidebar with various icons. In the center, a data preview table is shown with columns: demand, time\_stamp, Product\_id, price, Location, and item\_id. The table has 14 rows of sample data. At the top right, there's a 'Manage rows' tab which is currently selected. A modal window titled 'Drop rows by missing values' is open on the right side. It has a dropdown menu 'Column' set to 'Choose a column' and a value 'demand'. Below this are 'Preview', 'Cancel', and 'Add' buttons. The status bar at the bottom shows 'Total columns: 6', 'Total rows: 40,300', 'Total size: 243.000', and 'Reviewing first 100 rows'.

## Remove rows by outliers

Outliers, or rare values in the distribution and range of your data, can negatively impact model accuracy and lead to longer building times. With SageMaker Canvas, you can detect and remove rows that contain outliers in numeric columns. You can choose to define outliers with either standard deviations or a custom range.

To remove outliers from your data, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Manage rows**.
2. Choose **Drop rows by outlier values**.
3. Choose the **Column** you want to check for outliers.
4. Set the **Operator** to **Standard deviation**, **Custom numeric range**, or **Custom quantile range**.
5. If you choose **Standard deviation**, specify a **Standard deviations** (standard deviation) value from 1–3. If you choose **Custom numeric range** or **Custom quantile range**, specify the **Min** and **Max** values (numbers for numeric ranges, or percentiles between 0–100% for quantile ranges).
6. Choose **Add** to add the transform to the **Model recipe**.

The **Standard deviation** option detects and removes outliers in numeric columns using the mean and standard deviation. You specify the number of standard deviations a value must vary from the

mean to be considered an outlier. For example, if you specify 3 for **Standard deviations**, a value must fall more than 3 standard deviations from the mean to be considered an outlier.

The **Custom numeric range** and **Custom quantile range** options detect and remove outliers in numeric columns using minimum and maximum values. Use this method if you know your threshold values that delimit outliers. If you choose a numeric range, the **Min** and **Max** values should be the minimum and maximum numeric values that you want to allow in the data. If you choose a quantile range, the **Min** and **Max** values should be the minimum and maximum of the percentile range (0–100) that you want to allow.

After removing the rows from the dataset, SageMaker Canvas adds the transform in the **Model recipe** section. If you remove the transform from the **Model recipe** section, the rows return to your dataset.

The screenshot shows the SageMaker Canvas interface with the 'My models / deployment 2.8.2 / Version 1' header. In the center, there's a table view of a dataset named 'canvas-sample-retail-electronics-Forecast'. The table has columns: price, time\_stamp, Product\_c..., Location, item\_id, and demand. A tooltip says 'To see a recommended model type, specify a value for the target column.' Below the table, a 'Manage rows' button is highlighted. To the right, a 'Drop rows by outlier values' dialog is open. It includes fields for 'Column' (set to 'price'), 'Operator' (set to 'Standard deviation'), and 'Standard deviations' (set to '1'). A note below says 'The values should be integers and greater than 0 and less than 4.' At the bottom of the dialog are 'Preview', 'Cancel', and 'Add' buttons.

## Remove rows by custom values

You can remove rows with values that meet custom conditions. For example, you might want to exclude all of the rows with a price value greater than 100 when building your model. With this transform, you can create a rule that removes all rows that exceed the threshold you set.

To use the custom remove transform, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Manage rows**.
2. Choose **Drop rows by formula**.

3. Choose the **Column** you want to check.
4. Select the type of **Operation** you want to use, and then specify the values for the selected condition.
5. Choose **Add** to add the transform to the **Model recipe**.

For the **Operation**, you can choose one of the following options. Note that the available operations depend on the data type of the column you choose. For example, you cannot create a `is greater than` operation for a column containing text values.

Operation	Supported data type	Supported feature type	Function
Is equal to	Numeric, Text	Binary, Categorical	Removes rows where the value in <b>Column</b> equals the values you specify.
Is not equal to	Numeric, Text	Binary, Categorical	Removes rows where the value in <b>Column</b> doesn't equal the values you specify.
Is less than	Numeric	N/A	Removes rows where the value in <b>Column</b> is less than the value you specify.
Is less than or equal to	Numeric	N/A	Removes rows where the value in <b>Column</b> is less than or equal to the value you specify.
Is greater than	Numeric	N/A	Removes rows where the value in <b>Column</b> is greater than the value you specify.
Is greater than or equal to	Numeric	N/A	Removes rows where the value in <b>Column</b> is greater than or equal to the value you specify.

Operation	Supported data type	Supported feature type	Function
Is between	Numeric	N/A	Removes rows where the value in <b>Column</b> is between or equal to two values you specify.
Contains	Text	Categorical	Removes rows where the value in <b>Column</b> contains a values you specify.
Starts with	Text	Categorical	Removes rows where the value in <b>Column</b> begins with a value you specify.
Ends with	Text	Categorical	Removes rows where the value in <b>Column</b> ends with a value you specify.

After removing the rows from the dataset, SageMaker Canvas adds the transform in the **Model recipe** section. If you remove the transform from the **Model recipe** section, the rows return to your dataset.

## Rename columns

With the rename columns transform, you can rename columns in your data. When you rename a column, SageMaker Canvas changes the column name in the model input.

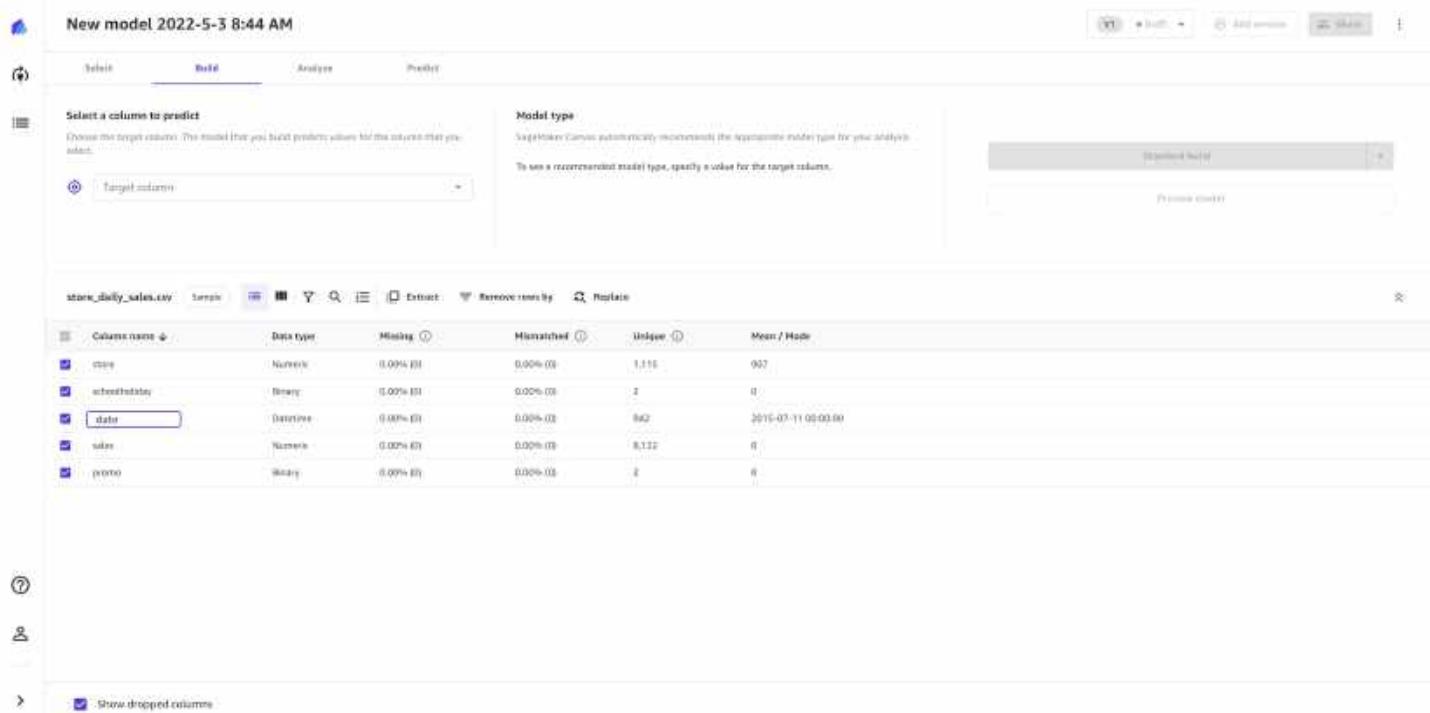
You can rename a column in your dataset by double-clicking on the column name in the **Build** tab of the SageMaker Canvas application and entering a new name.

Pressing the **Enter** key submits the change, and clicking anywhere outside the input cancels the change. You can also rename a column by clicking the **More options** icon

(), located at the end of the row in list view or at the end of the header cell in grid view, and choosing **Rename**.

Your column name can't be longer than 32 characters or have double underscores (`__`), and you can't rename a column to the same name as another column. You also can't rename a dropped column.

The following screenshot shows how to rename a column by double-clicking the column name.



The screenshot shows the SageMaker Canvas interface in the 'Build' tab. At the top, there's a header bar with tabs for 'Select', 'Build', 'Analyzer', and 'Predict'. Below the header, there's a 'Select a column to predict' section where 'Target column' is set to 'date'. To the right, there's a 'Model type' section with a note about automatically matching the appropriate model type for analysis. The main area displays a table for 'store\_daily\_sales.csv' with columns: 'date', 'store', 'achieveables', 'sales', and 'proto'. The 'date' column is highlighted with a blue border, indicating it's selected for renaming. The table includes headers for 'Column name', 'Data type', 'Missing', 'Mismatched', 'Unique', and 'Mean / Mode'. At the bottom left, there's a checkbox for 'Show dropped columns'.

When you rename a column, SageMaker Canvas adds the transform in the **Model recipe** section. If you remove the transform from the **Model recipe** section, the column reverts to its original name.

## Manage columns

With the following transforms, you can change the data type of columns and replace missing values or outliers for specific columns. SageMaker Canvas uses the updated data types or values when building your model but doesn't change your original dataset. Note that if you've dropped a column from your dataset using the [Drop columns](#) transform, you can't replace values in that column.

### Replace missing values

Missing values are a common occurrence in machine learning datasets and can impact model accuracy. You can choose to drop rows that have missing values, but your model is more accurate if you choose to replace the missing values instead. With this transform, you can replace missing values in numeric columns with the mean or median of the data in a column, or you can also specify a custom value with which to replace missing values. For non-numeric columns, you can replace missing values with the mode (most common value) of the column or a custom value.

Use this transform if you want to replace the null or empty values in certain columns. To replace missing values in a specified column, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Manage columns**.
2. Choose **Replace missing values**.
3. Choose the **Column** in which you want to replace missing values.
4. Set **Mode** to **Manual** to replace missing values with values that you specify. With the **Automatic (default)** setting, SageMaker Canvas replaces missing values with imputed values that best fit your data. This imputation method is done automatically for each model build, unless you specify the **Manual** mode.
5. Set the **Replace with** value:
  - If your column is numeric, then select **Mean**, **Median**, or **Custom**. **Mean** replaces missing values with the mean for the column, and **Median** replaces missing values with the median for the column. If you choose **Custom**, then you must specify a custom value that you want to use to replace missing values.
  - If your column is non-numeric, then select **Mode** or **Custom**. **Mode** replaces missing values with the mode, or the most common value, for the column. For **Custom**, specify a custom value that you want to use to replace missing values.
6. Choose **Add** to add the transform to the **Model recipe**.

After replacing the missing values in the dataset, SageMaker Canvas adds the transform in the **Model recipe** section. If you remove the transform from the **Model recipe** section, the missing values return to the dataset.

The screenshot shows the SageMaker Canvas interface. On the left, there's a data preview table titled "Source" with columns: demand, time\_stamp, Product\_categ..., price, Location, and Item\_id. The table contains several rows of data. On the right, a modal dialog titled "Replace missing values" is open. It has fields for "Column" (set to "demand"), "Mode" (set to "Manual"), and "Replace with" (set to "Custom" with value "0"). There are also sections for "Identify a value" and "Preview". At the bottom of the dialog are "Cancel" and "Add" buttons.

## Replace outliers

Outliers, or rare values in the distribution and range of your data, can negatively impact model accuracy and lead to longer building times. SageMaker Canvas enables you to detect outliers in numeric columns and replace the outliers with values that lie within an accepted range in your data. You can choose to define outliers with either standard deviations or a custom range, and you can replace outliers with the minimum and maximum values in the accepted range.

To replace outliers in your data, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Manage columns**.
2. Choose **Replace outlier values**.
3. Choose the **Column** in which you want to replace outliers.
4. For **Define outliers**, choose **Standard deviation**, **Custom numeric range**, or **Custom quantile range**.
5. If you choose **Standard deviation**, specify a **Standard deviations** (standard deviation) value from 1–3. If you choose **Custom numeric range** or **Custom quantile range**, specify the **Min** and **Max** values (numbers for numeric ranges, or percentiles between 0–100% for quantile ranges).

6. For **Replace with**, select **Min/max range**.
7. Choose **Add** to add the transform to the **Model recipe**.

The **Standard deviation** option detects outliers in numeric columns using the mean and standard deviation. You specify the number of standard deviations a value must vary from the mean to be considered an outlier. For example, if you specify 3 for **Standard deviations**, a value must fall more than 3 standard deviations from the mean to be considered an outlier. SageMaker Canvas replaces outliers with the minimum value or maximum value in the accepted range. For example, if you configure the standard deviations to only include values from 200–300, then SageMaker Canvas changes a value of 198 to 200 (the minimum).

The **Custom numeric range** and **Custom quantile range** options detect outliers in numeric columns using minimum and maximum values. Use this method if you know your threshold values that delimit outliers. If you choose a numeric range, the **Min** and **Max** values should be the minimum and maximum numeric values that you want to allow. SageMaker Canvas replaces any values that fall outside of the minimum and maximum to the minimum and maximum values. For example, if your range only allows values from 1–100, then SageMaker Canvas changes a value of 102 to 100 (the maximum). If you choose a quantile range, the **Min** and **Max** values should be the minimum and maximum of the percentile range (0–100) that you want to allow.

After replacing the values in the dataset, SageMaker Canvas adds the transform in the **Model recipe** section. If you remove the transform from the **Model recipe** section, the original values return to the dataset.

The screenshot shows the Amazon SageMaker Canvas interface. On the left, there's a sidebar with various icons for model management, deployment, and data exploration. The main area has a header "My models / deployment 2.8.2 / Version 1" and a sub-header "Target column". A message says "To see a recommended model type, specify a value for the target column." Below this is a "Data visualizer" section with a "Source" tab selected. It displays a table with columns: demand, time\_stamp, Product\_id, price, Location, and Item\_id. The table contains 20 rows of sample data. To the right of the table is a "Replace outlier values" panel. It includes a dropdown for "Column" set to "demand", an "Operator" dropdown set to "Standard deviation", and a "Standard deviations" input field set to "3". A note below says "Outliers are values that fall outside of the standard deviation you specified." At the bottom of the panel are "Replace with" dropdowns set to "Delete a row" and "Min/max range", and buttons for "Cancel" and "Add". At the very bottom of the interface, there are summary statistics: "Total columns: 6", "Total rows: 40,500", "Total crfs: 143,000", and a link to "Previewing first 100 rows". There's also a checked checkbox for "Show dropped columns".

## Change data type

SageMaker Canvas provides you with the ability to change the *data type* of your columns between numeric, text, and datetime, while also displaying the associated *feature type* for that data type. A *data type* refers to the format of the data and how it is stored, while the *feature type* refers to the characteristic of the data used in machine learning algorithms, such as binary or categorical. This gives you the flexibility to manually change the type of data in your columns based on the features. The ability to choose the right data type ensures data integrity and accuracy prior to building models. These data types are used when building models.

### Note

Currently, changing the feature type (for example, from binary to categorical) is not supported.

The following table lists all of the supported data types in Canvas.

Data type	Description	Example
Numeric	Numeric data represents numerical values	1, 2, 3 1.1, 1.2, 1.3
Text	Text data represents sequences of characters, like names or descriptions	A, B, C, D apple, banana, orange 1A!, 2A!, 3A!
Datetime	Datetime data represents dates and times in timestamp format	2019-07-01 01:00:00, 2019-07-01 02:00:00, 2019-07-01 03:00:00

The following table lists all of the supported feature types in Canvas.

Feature type	Description	Example
Binary	Binary features represent two possible values	0, 1, 0, 1, 0 (2 distinct values) true, false, true (2 distinct values)
Categorical	Categorical features represent distinct categories or groups	apple, banana, orange, apple (3 distinct values) A, B, C, D, E, A, D, C (5 distinct values)

To modify data type of a column in a dataset, do the following.

1. In the **Build** tab of the SageMaker Canvas application, go to the **Column view** or **Grid view** and select the **Data type** dropdown for the specific column.
2. In the **Data type** dropdown, choose the data type to convert to. The following screenshot shows the dropdown menu.

To build a model with this dataset, inner join with `canvas-sample-product-descriptions.csv` in Join data, set `ActualShippingDays` as a target column, and `Numeric prediction` as the model type. To learn more about how to build a model with this dataset, check our workshop top page.

Column name	Data type	Feature type	Missing	Mismatched	Unique	Mode
ShippingDistro	Text	Categorical	0.00% (0)	0.00% (0)	428	
KShippingDistro	Text	Categorical	0.00% (0)	0.00% (0)	421	
ShippingPriority	Datetime	Categorical	0.00% (0)	0.00% (0)	4	Normal
ShippingOrigin	Text	Categorical	0.00% (0)	0.00% (0)	8	Normal
ProductID	Text	Text	100% (1)	0.00% (0)	12	4F771B6-1831-44e6-...
OrderID	Text	Text	0.00% (0)	0.00% (0)	1,000	06E72B8-362d-44e...
OrderDate_year	Numeric	Binary	0.00% (0)	0.00% (0)	2	2,011
OrderDate_week_of_year	Numeric	Binary	0.00% (0)	0.00% (0)	55	1
OrderDate_month	Numeric	Binary	0.00% (0)	0.00% (0)	12	1
OrderDate_hour	Numeric	Binary	0.00% (0)	0.00% (0)	1	0
OrderDate_day_of_year	Numeric	Binary	0.00% (0)	0.00% (0)	366	390
OrderDate	Datetime	Text	0.00% (0)	0.00% (0)	561	2020-01-01 00:00:00

3. For **Column**, choose or verify the column you want to change the data type for.
4. For **New data type**, choose or verify the new data type you want to convert to.
5. If the **New data type** is Datetime or Numeric, choose one of the following options under **Handle invalid values**:
  - a. **Replace with empty value** – Invalid values are substituted with an empty value
  - b. **Delete rows** – Rows with an invalid value are removed from the dataset
  - c. **Replace with custom value** – Invalid values are substituted with the **Custom Value** that you specify.
6. Choose **Add** to add the transform to the **Model recipe**.

The data type for your column should now be updated.

## Prepare time series data

Use the following functionalities to prepare your time series data for building time series forecasting models.

### Resample time series data

By resampling time-series data, you can establish regular intervals for the observations in your time series dataset. This is particularly useful when working with time series data containing irregularly spaced observations. For instance, you can use resampling to transform a dataset with observations recorded every one hour, two hour and three hour intervals into a regular one hour

interval between observations. Forecasting algorithms require the observations to be taken at regular intervals.

To resample time series data, do the following.

1. In the **Build** tab of the SageMaker Canvas application, choose **Time series**.
2. Choose **Resample**.
3. For **Timestamp column**, choose the column you want to apply the transform to. You can only select columns of the **Datetime** type.
4. In the **Frequency settings** section, choose a **Frequency** and **Rate**. **Frequency** is the unit of frequency and **Rate** is the interval of the unit of frequency to be applied to the column. For example, choosing **Calendar Day** for **Frequency value** and 1 for **Rate** sets the interval to increase every 1 calendar day, such as 2023-03-26 00:00:00, 2023-03-27 00:00:00, 2023-03-28 00:00:00. See the table after this procedure for a complete list of **Frequency value**.
5. Choose **Add** to add the transform to the **Model recipe**.

The following table lists all of the **Frequency** types you can select when resampling time series data.

Frequency	Description	Example values (assuming Rate is 1)
Business Day	Resample observations in the datetime column to 5 business days of the week (Monday, Tuesday, Wednesday, Thursday, Friday)	2023-03-24 00:00:00
		2023-03-27 00:00:00
		2023-03-28 00:00:00
		2023-03-29 00:00:00
		2023-03-30 00:00:00
		2023-03-31 00:00:00
		2023-04-03 00:00:00
Calendar Day	Resample observations in the datetime column	2023-03-26 00:00:00

Frequency	Description	Example values (assuming Rate is 1)
	to all 7 days of the week (Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday)	2023-03-27 00:00:00 2023-03-28 00:00:00 2023-03-29 00:00:00 2023-03-30 00:00:00 2023-03-31 00:00:00 2023-04-01 00:00:00
Week	Resample observations in the datetime column to the first day of each week	2023-03-13 00:00:00 2023-03-20 00:00:00 2023-03-27 00:00:00 2023-04-03 00:00:00
Month	Resample observations in the datetime column to the first day of each month	2023-03-01 00:00:00 2023-04-01 00:00:00 2023-05-01 00:00:00 2023-06-01 00:00:00
Annual Quarter	Resample observations in the datetime column to the last day of each quarter	2023-03-31 00:00:00 2023-06-30 00:00:00 2023-09-30 00:00:00 2023-12-31 00:00:00

Frequency	Description	Example values (assuming Rate is 1)
Year	Resample observations in the datetime column to the last day of each year	2022-12-31 0:00:00 2023-12-31 0:00:00 2024-12-31 0:00:00
Hour	Resample observations in the datetime column to each hour of each day	2023-03-24 0:00:00 2023-03-24 1:00:00 2023-03-24 2:00:00 2023-03-24 3:00:00
Minute	Resample observations in the datetime column to each minute of each hour	2023-03-24 0:00:00 2023-03-24 0:01:00 2023-03-24 0:02:00 2023-03-24 0:03:00
Second	Resample observations in the datetime column to each second of each minute	2023-03-24 0:00:00 2023-03-24 0:00:01 2023-03-24 0:00:02 2023-03-24 0:00:03

When applying the resampling transform, you can use the **Advanced** option to specify how the resulting values of the rest of the columns (other than the timestamp column) in your dataset are modified. This can be achieved by specifying the resampling methodology, which can either be downsampling or upsampling for both numeric and non-numeric columns.

*Downsampling* increases the interval between observations in the dataset. For example, if you downsample observations that are taken either every hour or every two hours, each observation in your dataset is taken every two hours. The values of other columns of the hourly observations

are aggregated into a single value using a combination method. The following tables show an example of downsampling time series data by using mean as the combination method. The data is downsampled from every two hours to every hour.

The following table shows the hourly temperature readings over a day before downsampling.

Timestamp	Temperature (Celsius)
12:00 pm	30
1:00 am	32
2:00 am	35
3:00 am	32
4:00 am	30

The following table shows the temperature readings after downsampling to every two hours.

Timestamp	Temperature (Celsius)
12:00 pm	30
2:00 am	33.5
2:00 am	35
4:00 am	32.5

To downsample time series data, do the following:

1. Expand the **Advanced** section under the **Resample** transform.
2. Choose **Non-numeric combination** to specify the combination method for non-numeric columns. See the table below for a complete list of combination methods.
3. Choose **Numeric combination** to specify the combination method for numeric columns. See the table below for a complete list of combination methods.

If you don't specify combination methods, the default values are **Most Common** for **Non-numeric combination** and **Mean** for **Numeric combination**. The following table lists the methods for numeric and non-numeric combination.

Downsampling methodology	Combination method	Description
Non-numeric combination	Most Common	Aggregate values in the non-numeric column by the most commonly occurring value
Non-numeric combination	Last	Aggregate values in the non-numeric column by the last value in the column
Non-numeric combination	First	Aggregate values in the non-numeric column by the first value in the column
Numeric combination	Mean	Aggregate values in the numeric column by taking the mean of all the values in the column
Numeric combination	Median	Aggregate values in the numeric column by taking the median of all the values in the column
Numeric combination	Min	Aggregate values in the numeric column by taking the minimum of all the values in the column
Numeric combination	Max	Aggregate values in the numeric column by taking the maximum of all the values in the column

Downsampling methodology	Combination method	Description
Numeric combination	Sum	Aggregate values in the numeric column by adding all the values in the column
Numeric combination	Quantile	Aggregate values in the numeric column by taking the quantile of all the values in the column

*Upsampling* reduces the interval between observations in the dataset. For example, if you upsample observations that are taken every two hours into hourly observations, the values of other columns of the hourly observations are interpolated from the ones that have been taken every two hours.

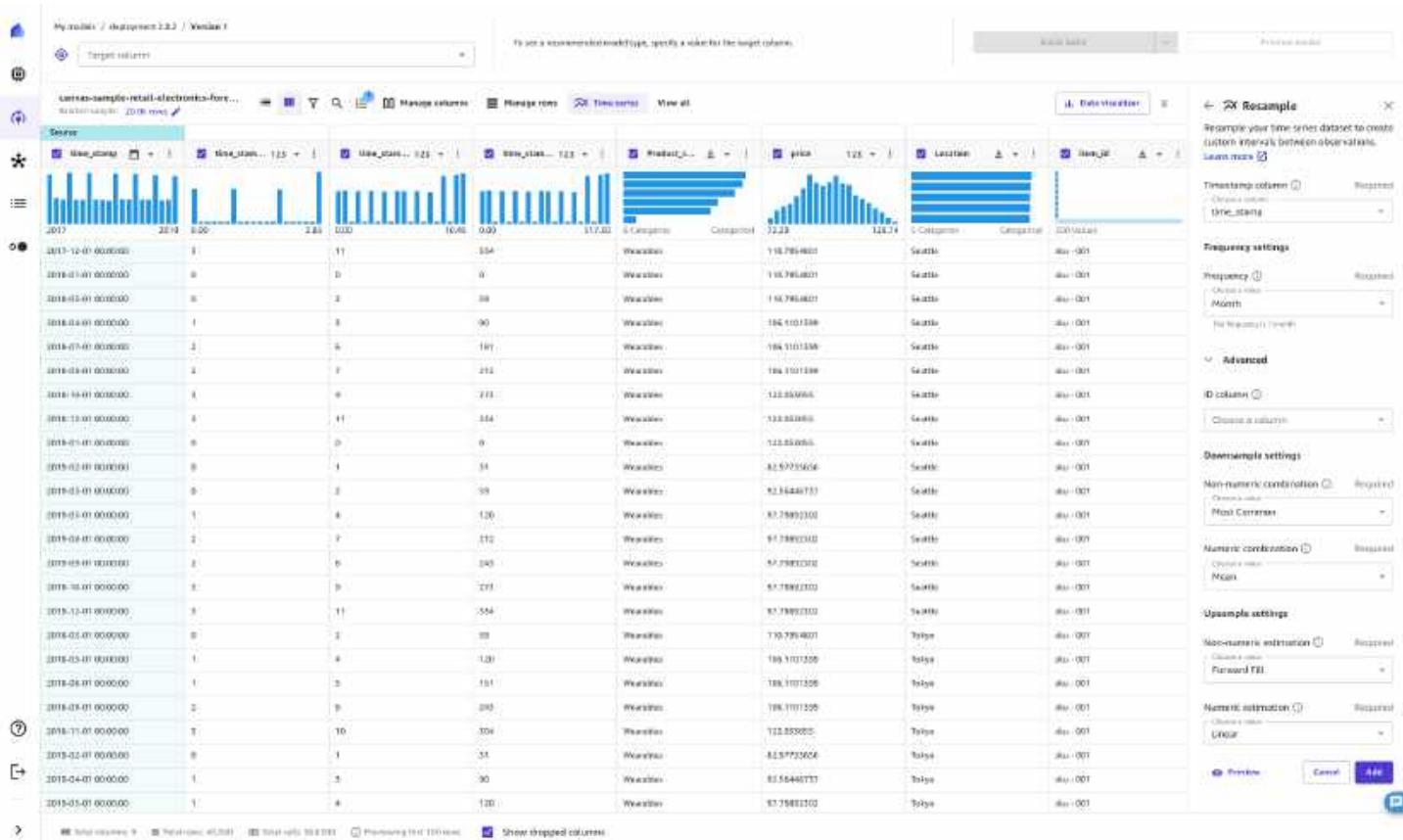
To upsample time series data, do the following:

1. Expand the **Advanced** section under the **Resample** transform.
2. Choose **Non-numeric estimation** to specify the estimation method for non-numeric columns. See the table after this procedure for a complete list of methods.
3. Choose **Numeric estimation** to specify the estimation method for numeric columns. See the table below for a complete list of methods.
4. (Optional) Choose **ID Column** to specify the column that has the IDs of the observations of the time series. Specify this option if your dataset has two time series. If you have a column representing only one time series, don't specify a value for this field. For example, you can have a dataset that has the columns `id` and `purchase`. The `id` column has the following values: [1, 2, 2, 1]. The `purchase` column has the following values [\$2, \$3, \$4, \$1]. Therefore, the dataset has two time series—one time series is 1: [\$2, \$1], and the other time series is 2: [\$3, \$4].

If you don't specify estimation methods, the default values are **Forward Fill** for **Non-numeric estimation** and **Linear** for **Numeric estimation**. The following table lists the methods for estimation.

Upsampling methodology	Estimation method	Description
Non-numeric estimation	Forward Fill	Interpolate values in the non-numeric column by taking the consecutive values after all the values in the column
Non-numeric estimation	Backward Fill	Interpolate values in the non-numeric column by taking the consecutive values before all the values in the column
Non-numeric estimation	Keep Missing	Interpolate values in the non-numeric column by showing empty values
Numeric estimation	Linear, Time, Index, Zero, S-Linear, Nearest, Quadratic, Cubic, Barycentric, Polynomia l, Krogh, Piecewise Polynomia l, Spline, P-chip, Akima, Cubic Spline, From Derivatives	Interpolate values in the numeric column by using the specified interpolator. For information on interpolation methods, see <a href="#">pandas.DataFrame.interpolate</a> in the pandas documentation.

The following screenshot shows the **Advanced** settings with the fields for downsampling and upsampling filled out.



## Use datetime extraction

With the datetime extraction transform, you can extract values from a datetime column to a separate column. For example, if you have a column containing dates of purchases, you can extract the month value to a separate column and use the new column when building your model. You can also extract multiple values to separate columns with a single transform.

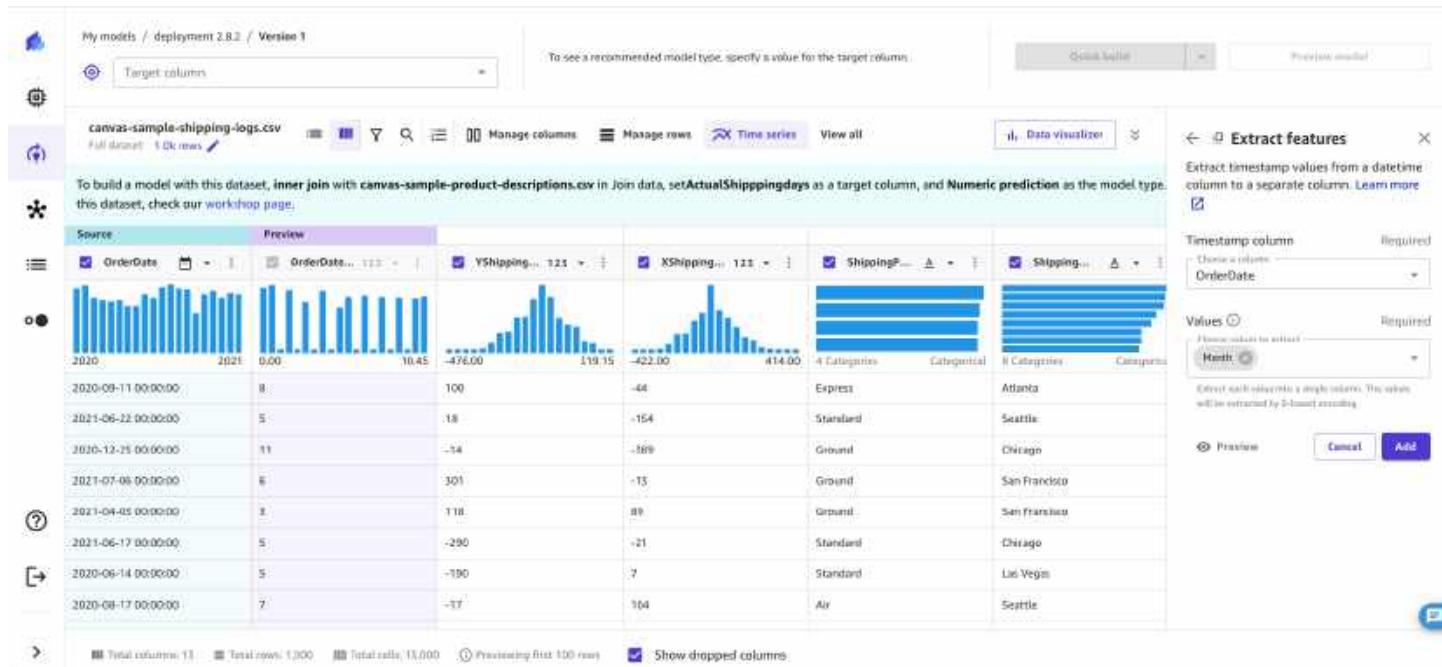
Your datetime column must use a supported timestamp format. For a list of the formats that SageMaker Canvas supports, see [Time Series Forecasts in Amazon SageMaker Canvas](#). If your dataset does not use one of the supported formats, update your dataset to use a supported timestamp format and re-import it to Amazon SageMaker Canvas before building your model.

To perform a datetime extraction, do the following.

1. In the **Build** tab of the SageMaker Canvas application, on the transforms bar, choose **View all**.
  2. Choose **Extract features**.
  3. Choose the **Timestamp column** from which you want to extract values.

4. For **Values**, select one or more values to extract from the column. The values you can extract from a timestamp column are **Year**, **Month**, **Day**, **Hour**, **Week of year**, **Day of year**, and **Quarter**.
5. (Optional) Choose **Preview** to preview the transform results.
6. Choose **Add** to add the transform to the **Model recipe**.

SageMaker Canvas creates a new column in the dataset for each of the values you extract. Except for **Year** values, SageMaker Canvas uses a 0-based encoding for the extracted values. For example, if you extract the **Month** value, January is extracted as 0, and February is extracted as 1.



You can see the transform listed in the **Model recipe** section. If you remove the transform from the **Model recipe** section, the new columns are removed from the dataset.

## Model evaluation

After you've built your model, you can evaluate how well your model performed on your data before using it to make predictions. You can use information, such as the model's accuracy when predicting labels and advanced metrics, to determine whether your model can make sufficiently accurate predictions for your data.

The section [Evaluate your model's performance](#) describes how to view and interpret the information on your model's **Analyze** page. The section [Use advanced metrics in your analyses](#)

contains more detailed information about the **Advanced metrics** used to quantify your model's accuracy.

You can also view more advanced information for specific *model candidates*, which are all of the model iterations that Canvas runs through while building your model. Based on the advanced metrics for a given model candidate, you can select a different candidate to be the default, or the version that is used for making predictions and deploying. For each model candidate, you can view the **Advanced metrics** information to help you decide which model candidate you'd like to select as the default. You can view this information by selecting the model candidate from the **Model leaderboard**. For more information, see [View model candidates in the model leaderboard](#).

Canvas also provides the option to download a Jupyter notebook so that you can view and run the code used to build your model. This is useful if you'd like to make adjustments to the code or learn more about how your model was built. For more information, see [Download a model notebook](#).

## Evaluate your model's performance

Amazon SageMaker Canvas provides overview and scoring information for the different types of model. Your model's score can help you determine how accurate your model is when it makes predictions. The additional scoring insights can help you quantify the differences between the actual and predicted values.

To view the analysis of your model, do the following:

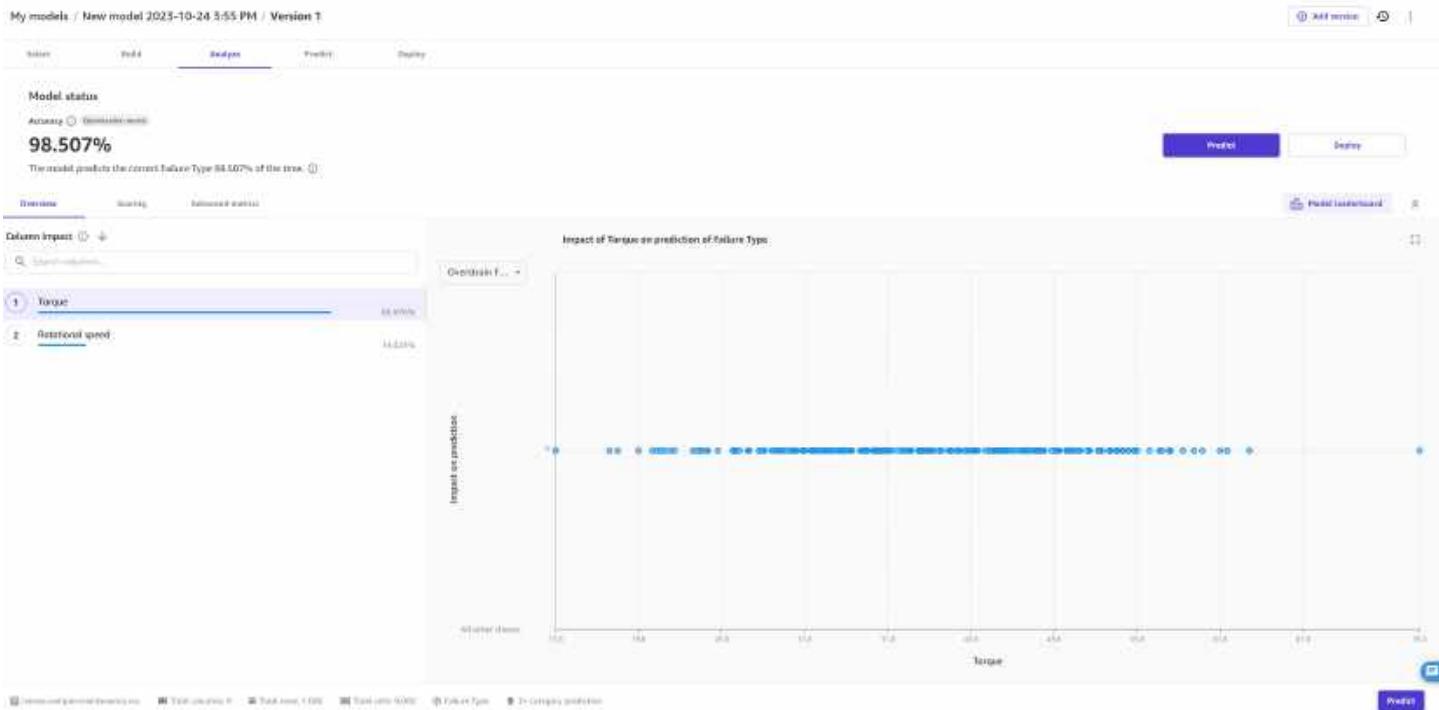
1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **My models**.
3. Choose the model that you built.
4. In the top navigation pane, choose the **Analyze** tab.
5. Within the **Analyze** tab, you can view the overview and scoring information for your model.

The following sections describe how to interpret the scoring for each model type.

## Evaluate categorical prediction models

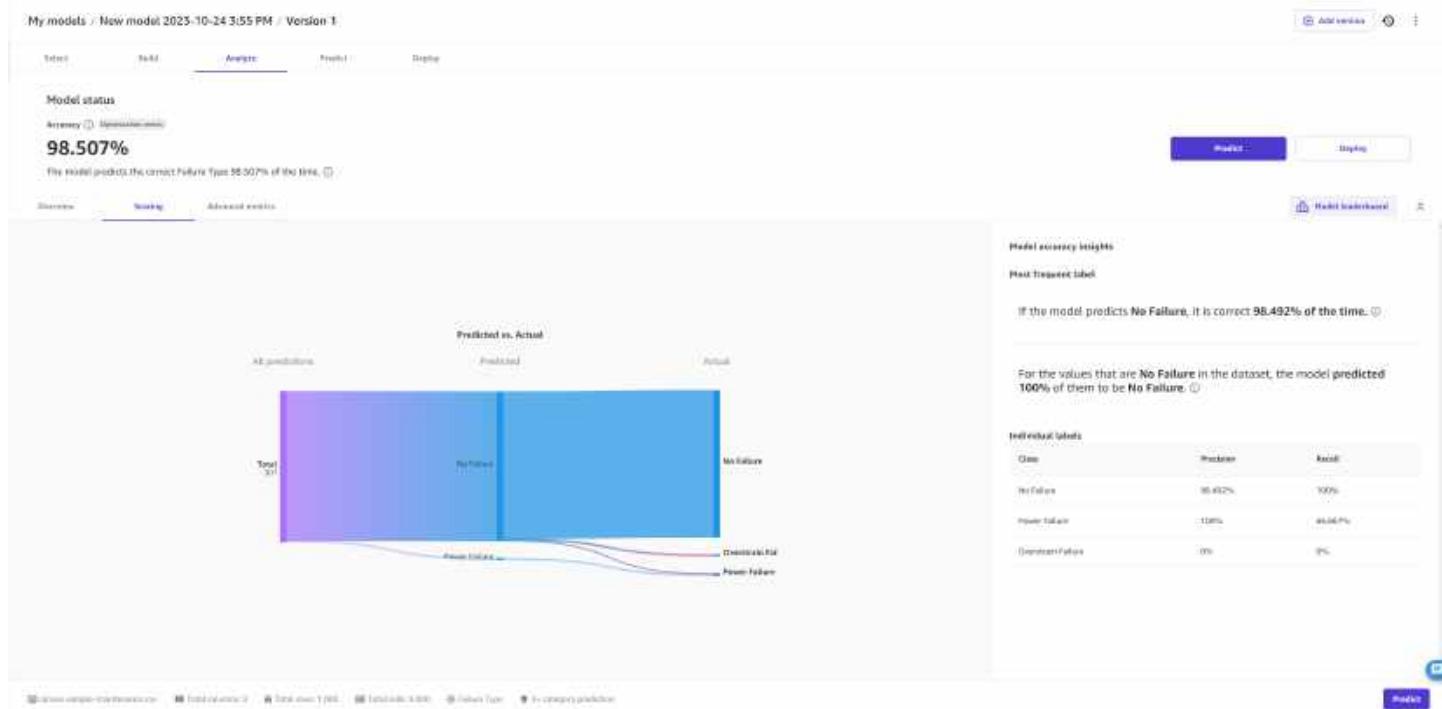
The **Overview** tab shows you the column impact for each column. **Column impact** is a percentage score that indicates how much weight a column has in making predictions in relation to the other columns. For a column impact of 25%, Canvas weighs the prediction as 25% for the column and 75% for the other columns.

The following screenshot shows the **Accuracy** score for the model, along with the **Optimization metric**, which is the metric that you choose to optimize when building the model. In this case, the **Optimization metric is Accuracy**. You can specify a different optimization metric if you build a new version of your model.



The **Scoring** tab for a categorical prediction model gives you the ability to visualize all the predictions. Line segments extend from the left of the page, indicating all the predictions the model has made. In the middle of the page, the line segments converge on a perpendicular segment to indicate the proportion of each prediction to a single category. From the predicted category, the segments branch out to the actual category. You can get a visual sense of how accurate the predictions were by following each line segment from the predicted category to the actual category.

The following image gives you an example **Scoring** section for a **3+ category prediction** model.



You can also view the **Advanced metrics** tab for more detailed information about your model's performance, such as the advanced metrics, error density plots, or confusion matrices. To learn more about the **Advanced metrics** tab, see [Use advanced metrics in your analyses](#).

## Evaluate numeric prediction models

The **Overview** tab shows you the column impact for each column. **Column impact** is a percentage score that indicates how much weight a column has in making predictions in relation to the other columns. For a column impact of 25%, Canvas weighs the prediction as 25% for the column and 75% for the other columns.

The following screenshot shows the **RMSE** score for the model on the **Overview** tab, which in this case is the **Optimization metric**. The **Optimization metric** is the metric that you choose to optimize when building the model. You can specify a different optimization metric if you build a new version of your model.

Select      Build      **Analyze**      Predict

Model status

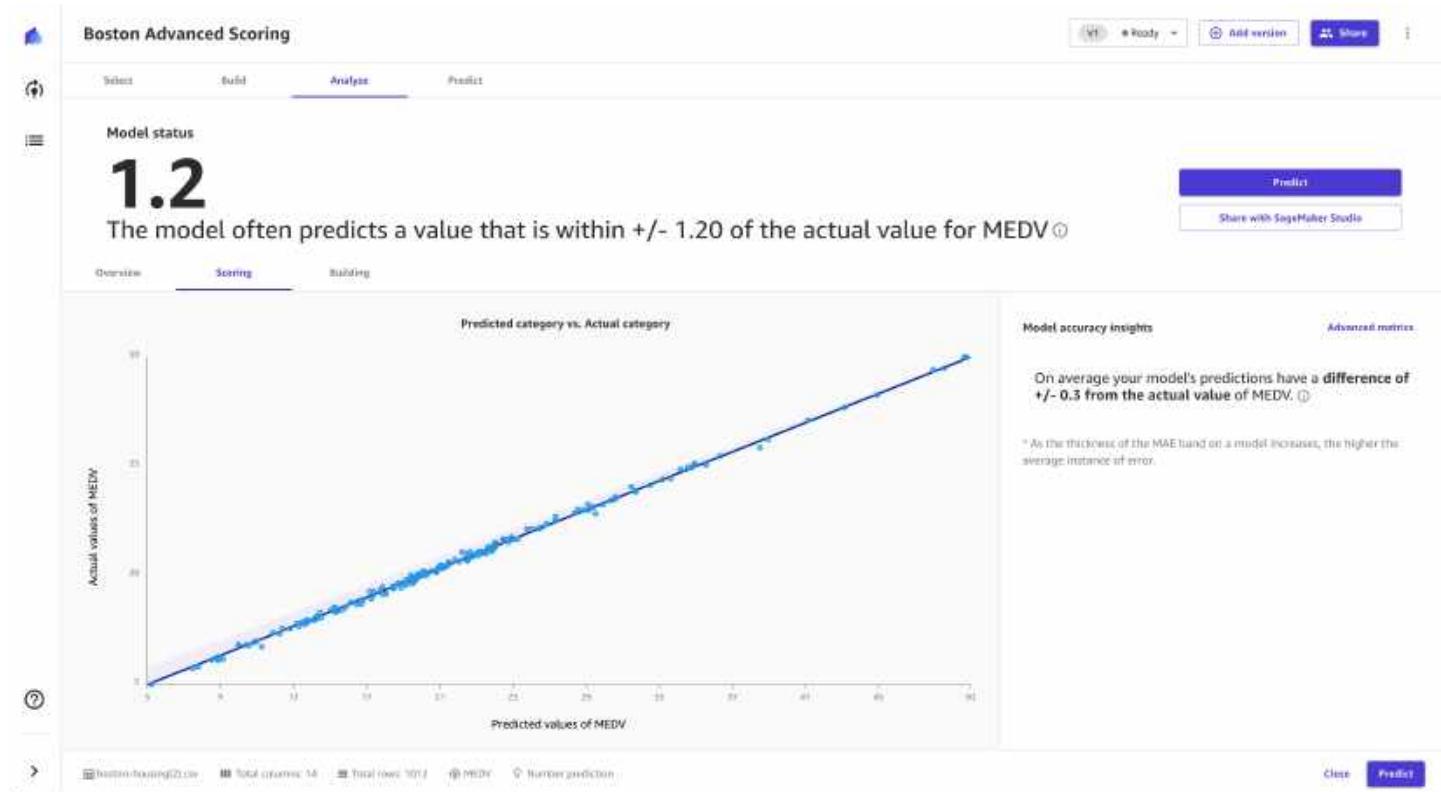
RMSE ⓘ Optimization metric  
43344.19

The model often predicts a value that is within +/- 43344.19 of the actual value for median\_house\_value ⓘ

Overview      Scoring

The **Scoring** tab for numeric prediction shows a line to indicate the model's predicted value in relation to the data used to make predictions. The values of the numeric prediction are often +/- the RMSE (root mean squared error) value. The value that the model predicts is often within the range of the RMSE. The width of the purple band around the line indicates the RMSE range. The predicted values often fall within the range.

The following image shows the **Scoring** section for numeric prediction.



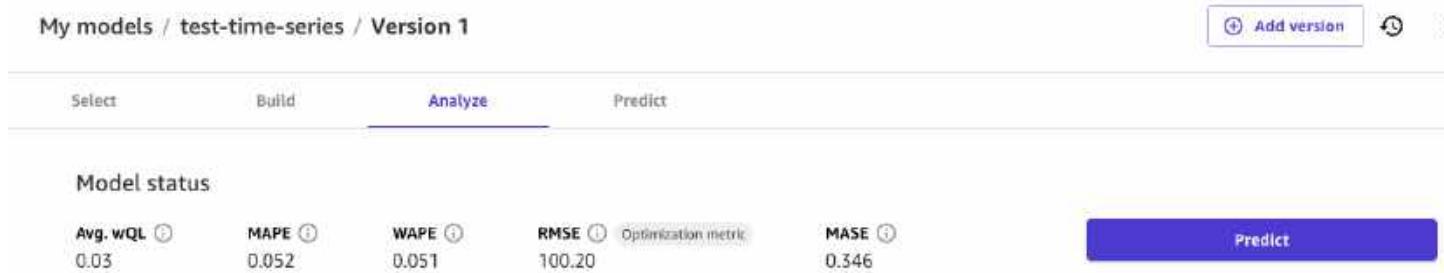
You can also view the **Advanced metrics** tab for more detailed information about your model's performance, such as the advanced metrics, error density plots, or confusion matrices. To learn more about the **Advanced metrics** tab, see [Use advanced metrics in your analyses](#).

## Evaluate time series forecasting models

On the **Analyze** page for time series forecasting models, you can see an overview of the model's metrics. You can hover over each metric for more information, or you can see [Use advanced metrics in your analyses](#) for more information about each metric.

In the **Column impact** section, you can see the score for each column. **Column impact** is a percentage score that indicates how much weight a column has in making predictions in relation to the other columns. For a column impact of 25%, Canvas weighs the prediction as 25% for the column and 75% for the other columns.

The following screenshot shows the time series metrics scores for the model, along with the **Optimization metric**, which is the metric that you choose to optimize when building the model. In this case, the **Optimization metric** is **RMSE**. You can specify a different optimization metric if you build a new version of your model. These metrics scores are taken from your backtest results, which are available for download in the **Artifacts** tab.



The screenshot shows the 'My models / test-time-series / Version 1' interface. The 'Analyze' tab is selected. At the top, there are tabs for 'Select', 'Build', 'Analyze', and 'Predict'. Below the tabs, the 'Model status' section displays several metrics: Avg. wQI (0.03), MAPE (0.052), WAPE (0.051), RMSE (100.20) labeled as the 'Optimization metric', and MASE (0.346). A large blue 'Predict' button is visible on the right. The overall interface is clean and modern, typical of the SageMaker Canvas web application.

The **Artifacts** tab provides access to several key resources that you can use to dive deeper into your model's performance and continue iterating upon it:

- **Shuffled training and validation splits** – This section includes links to the artifacts generated when your dataset was split into training and validation sets, enabling you to review the data distribution and potential biases.
- **Backtest results** – This section includes a link to the forecasted values for your validation dataset, which is used to generate accuracy metrics and evaluation data for your model.
- **Accuracy metrics** – This section lists the advanced metrics that evaluate your model's performance, such as Root Mean Squared Error (RMSE). For more information about each metric, see [Metrics for time series forecasts](#).
- **Explainability report** – This section provides a link to download the explainability report, which offers insights into the model's decision-making process and the relative importance of input columns. This report can help you identify potential areas for improvement.

On the **Analyze** page, you can also choose the **Download** button to directly download the backtest results, accuracy metrics, and explainability report artifacts to your local machine.

## Evaluate image prediction models

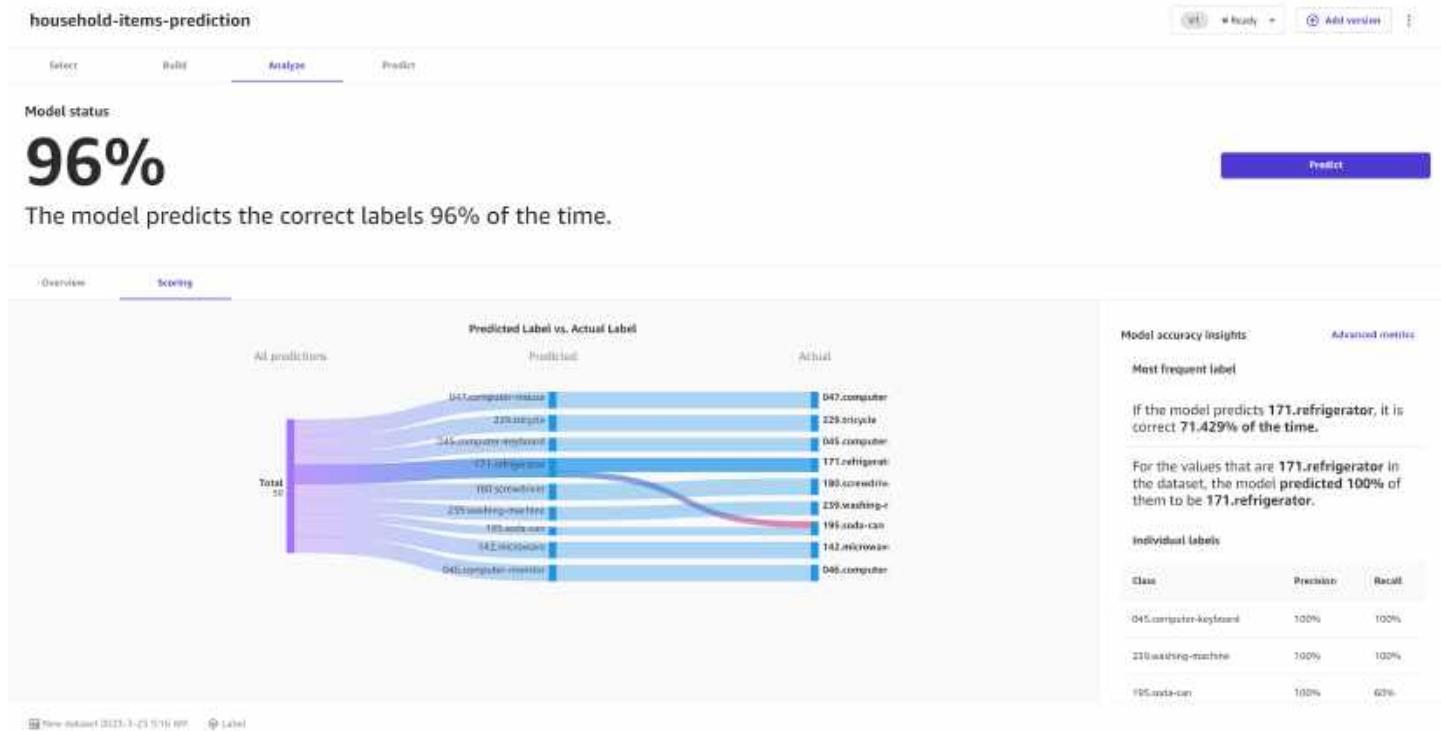
The **Overview** tab shows you the **Per label performance**, which gives you an overall accuracy score for the images predicted for each label. You can choose a label to see more specific details, such as the **Correctly predicted** and **Incorrectly predicted** images for the label.

You can turn on the **Heatmap** toggle to see a heatmap for each image. The heatmap shows you the areas of interest that have the most impact when your model is making predictions. For more information about heatmaps and how to use them to improve your model, choose the **More info** icon next to the **Heatmap** toggle.

The **Scoring** tab for single-label image prediction models shows you a comparison of what the model predicted as the label versus what the actual label was. You can select up to 10 labels at a time. You can change the labels in the visualization by choosing the labels dropdown menu and selecting or deselecting labels.

You can also view insights for individual labels or groups of labels, such as the three labels with the highest or lowest accuracy, by choosing the **View scores for** dropdown menu in the **Model accuracy insights** section.

The following screenshot shows the **Scoring** information for a single-label image prediction model.



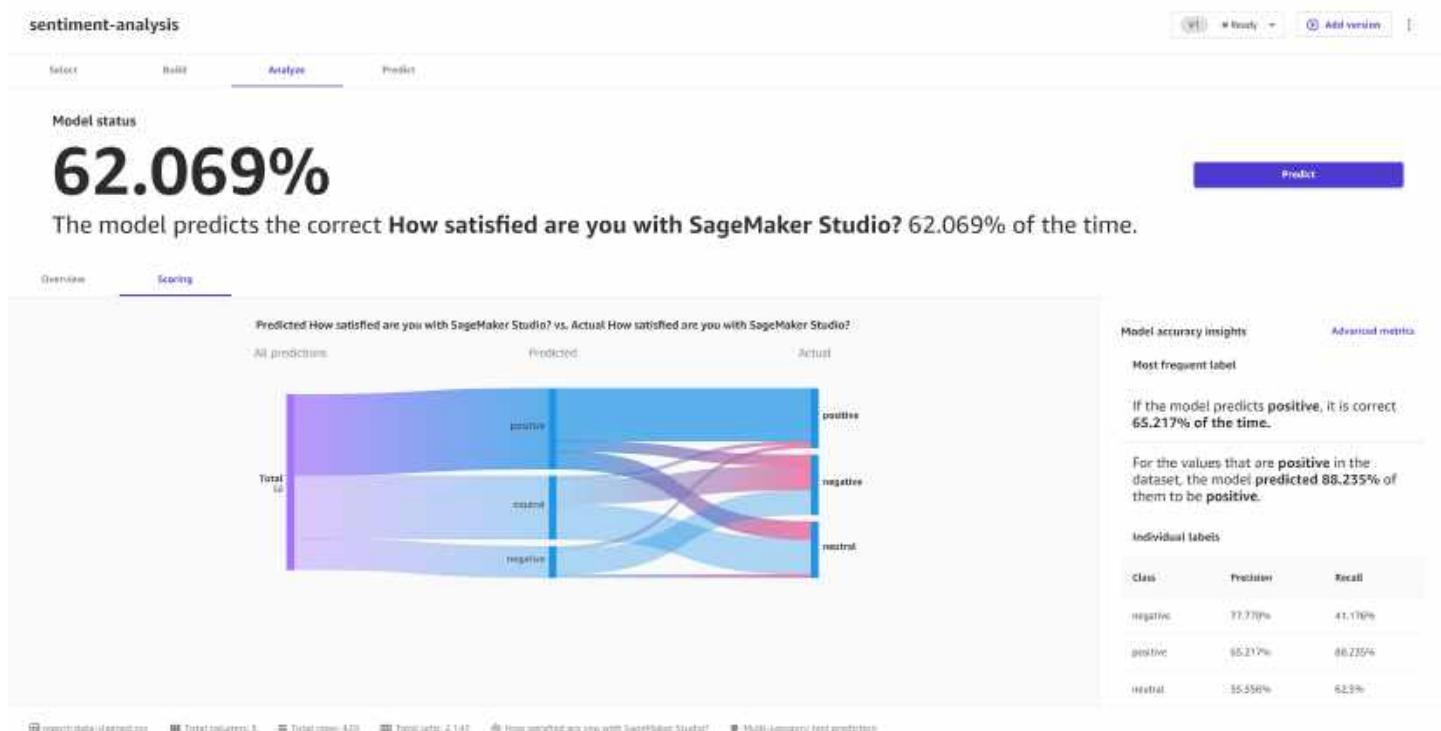
## Evaluate text prediction models

The **Overview** tab shows you the **Per label performance**, which gives you an overall accuracy score for the passages of text predicted for each label. You can choose a label to see more specific details, such as the **Correctly predicted** and **Incorrectly predicted** passages for the label.

The **Scoring** tab for multi-category text prediction models shows you a comparison of what the model predicted as the label versus what the actual label was.

In the **Model accuracy insights** section, you can see the **Most frequent category**, which tells you the category that the model predicted most frequently and how accurate those predictions were. If your model predicts a label of **Positive** correctly 99% of the time, then you can be fairly confident that your model is good at predicting positive sentiment in text.

The following screenshot shows the **Scoring** information for a multi-category text prediction model.



## Use advanced metrics in your analyses

The following section describes how to find and interpret the advanced metrics for your model in Amazon SageMaker Canvas.

### Note

Advanced metrics are only currently available for numeric and categorical prediction models.

To find the **Advanced metrics** tab, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **My models**.
3. Choose the model that you built.
4. In the top navigation pane, choose the **Analyze** tab.
5. Within the **Analyze** tab, choose the **Advanced metrics** tab.

In the **Advanced metrics** tab, you can find the **Performance** tab. The page looks like the following screenshot.

The screenshot shows the 'Performance' tab within the 'Advanced metrics' section of the SageMaker Canvas interface. At the top, it displays the 'Model status' as 'Accuracy (98.507%)'. Below this, there are four average metric sections: 'Average F1' (59.247%), 'Average accuracy' (98.507%), 'Average precision' (65.164%), and 'Average recall' (55.556%). A note states, 'The model predicts the correct value 98.507% of the time.' To the right, there are buttons for 'Predict' and 'Deploy'. Further down, there's a 'Metrics table' section with columns for 'Metric name' and 'Value'. The table lists various metrics for a regression model, including 'Mean absolute error' (0.99574649939771), 'Root mean square error' (0.00000002044068), 'R-squared' (0.99740617719888), 'Adjusted R-squared' (0.991401133142158), 'Root mean square residual' (0.00000002044068), 'Log loss' (0.4163507875429744), and 'Information criterion' (0.8914311854587282). At the bottom, there are links for 'View details' and 'View metrics'.

At the top, you can see an overview of the metrics scores, including the **Optimization metric**, which is the metric that you selected (or that Canvas selected by default) to optimize when building the model.

The following sections describe more detailed information for the **Performance** tab within the **Advanced metrics**.

## Performance

In the **Performance** tab, you'll see a **Metrics table**, along with visualizations that Canvas creates based on your model type. For categorical prediction models, Canvas provides a *confusion matrix*, whereas for numeric prediction models, Canvas provides you with *residuals* and *error density* charts.

In the **Metrics table**, you are provided with a full list of your model's scores for each advanced metric, which is more comprehensive than the scores overview at the top of the page. The metrics shown here depend on your model type. For a reference to help you understand and interpret each metric, see [Metrics reference](#).

To understand the visualizations that might appear based on your model type, see the following options:

- **Confusion matrix** – Canvas uses confusion matrices to help you visualize when a model makes predictions correctly. In a confusion matrix, your results are arranged to compare the predicted values against the actual values. The following example explains how a confusion matrix works for a 2 category prediction model that predicts positive and negative labels:
  - True positive – The model correctly predicted positive when the true label was positive.
  - True negative – The model correctly predicted negative when the true label was negative.
  - False positive – The model incorrectly predicted positive when the true label was negative.
  - False negative – The model incorrectly predicted negative when the true label was positive.
- **Precision recall curve** – The precision recall curve is a visualization of the model's precision score plotted against the model's recall score. Generally, a model that can make perfect predictions would have precision and recall scores that are both 1. The precision recall curve for a decently accurate model is fairly high in both precision and recall.
- **Residuals** – Residuals are the difference between the actual values and the values predicted by the model. A residuals chart plots the residuals against the corresponding values to visualize their distribution and any patterns or outliers. A normal distribution of residuals around zero indicates that the model is a good fit for the data. However, if the residuals are significantly skewed or have outliers, it may indicate that the model is overfitting the data or that there are other issues that need to be addressed.
- **Error density** – An error density plot is a representation of the distribution of errors made by a model. It shows the probability density of the errors at each point, helping you to identify any areas where the model may be overfitting or making systematic errors.

## View model candidates in the model leaderboard

When you do a [Standard build](#) for tabular and time series forecasting models in Amazon SageMaker Canvas, SageMaker AI trains multiple *model candidates* (different iterations of the model) and by default selects the one with the highest value for the optimization metric. For tabular models, Canvas builds up to 250 different model candidates using various algorithms and hyperparameter settings. For time series forecasting models, Canvas builds 7 different models—one for each of the [supported forecasting algorithms](#) and one ensemble model that averages the predictions of the other models to try to optimize accuracy.

The default model candidate is the only version that you can use in Canvas for actions like making predictions, registering to the model registry, or deploying to an endpoint. However, you might want to review all of the model candidates and select a different candidate to be the default

model. You can view all of the model candidates and more details about each candidate on the **Model leaderboard** in Canvas.

To view the **Model leaderboard**, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **My models**.
3. Choose the model that you built.
4. In the top navigation pane, choose the **Analyze** tab.
5. Within the **Analyze** tab, choose **Model leaderboard**.

The **Model leaderboard** page opens, which for tabular models looks like the following screenshot.

Model name	Accuracy	F1 Optimization	Precision	Recall
XGBoost_01 <small>(Default model)</small>	98.232%	83.245%	79.653%	75.568%
XGBoost_02	98.212%	84.122%	78.375%	75.113%
ExtraTrees_01	97.127%	83.125%	78.122%	75.265%
ExtraTrees_02	97.115%	86.924%	78.156%	
LinearLearner_01	96.398%	85.356%	78.339%	74.319%
LinearLearner_02	96.113%	82.412%	78.107%	74.106%
LinearLearner_05	95.365%	83.122%	77.226%	73.513%
XGBoost_123	95.092%	82.056%	76.165%	73.615%
XGBoost_5B	94.469%	82.055%	75.592%	74.365%
ExtraTrees_98	94.122%	81.122%	75.135%	74.293%
ExtraTrees_109	93.824%	80.357%	75.287%	74.106%
ExtraTrees_122	93.812%	80.323%	76.275%	74.102%
ExtraTrees_109	93.785%	80.185%	77.532%	74.098%

For time series forecasting models, you see 7 models, which include one for each of the time series forecasting algorithms supported by Canvas and one ensemble model. For more information about the algorithms, see [Advanced time series forecasting model settings](#).

In the preceding screenshot, you can see that the first model candidate listed is marked as the **Default model**. This is the model candidate with which you can make predictions or deploy to endpoints.

To view more detailed metrics information about the model candidates to compare them, you can choose the **More options** icon

( ! )

and choose **View model details**.

### **Important**

Loading the model details for non-default model candidates may take a few minutes (typically less than 10 minutes), and SageMaker AI Hosting charges apply. For more information, see [SageMaker AI Pricing](#).

The model candidate opens in the **Analyze** tab, and the metrics shown are specific to that model candidate. When you're done reviewing the model candidate's metrics, you can go back or exit the view to return to the **Model leaderboard**.

If you'd like to set the **Default model** to a different candidate, you can choose the **More options** icon

( ! )

and choose **Change to the default model**. Changing the default model for a model trained using HPO mode might take several minutes.

### **Note**

If your model is already deployed in production, [registered to the model registry](#), or has [automations](#) set up, you must delete your deployment, model registration, or automations before changing the default model.

## **Metrics reference**

The following sections describe the metrics that are available in Amazon SageMaker Canvas for each model type.

## Metrics for numeric prediction

The following list defines the metrics for numeric prediction in SageMaker Canvas and gives you information about how you can use them.

- InferenceLatency – The approximate amount of time between making a request for a model prediction to receiving it from a real-time endpoint to which the model is deployed. This metric is measured in seconds and is only available for models built with the **Ensembling** mode.
- MAE – Mean absolute error. On average, the prediction for the target column is  $+/- \{MAE\}$  from the actual value.

Measures how different the predicted and actual values are when they're averaged over all values. MAE is commonly used in numeric prediction to understand model prediction error. If the predictions are linear, MAE represents the average distance from a predicted line to the actual value. MAE is defined as the sum of absolute errors divided by the number of observations. Values range from 0 to infinity, with smaller numbers indicating a better model fit to the data.

- MAPE – Mean absolute percent error. On average, the prediction for the target column is  $+/- \{MAPE\} \%$  from the actual value.

MAPE is the mean of the absolute differences between the actual values and the predicted or estimated values, divided by the actual values and expressed as a percentage. A lower MAPE indicates better performance, as it means that the predicted or estimated values are closer to the actual values.

- MSE – Mean squared error, or the average of the squared differences between the predicted and actual values.

MSE values are always positive. The better a model is at predicting the actual values, the smaller the MSE value is.

- R2 – The percentage of the difference in the target column that can be explained by the input column.

Quantifies how much a model can explain the variance of a dependent variable. Values range from one (1) to negative one (-1). Higher numbers indicate a higher fraction of explained variability. Values close to zero (0) indicate that very little of the dependent variable can be explained by the model. Negative values indicate a poor fit and that the model is outperformed by a constant function (or a horizontal line).

- RMSE – Root mean squared error, or the standard deviation of the errors.

Measures the square root of the squared difference between predicted and actual values, and is averaged over all values. It is used to understand model prediction error, and it's an important metric to indicate the presence of large model errors and outliers. Values range from zero (0) to infinity, with smaller numbers indicating a better model fit to the data. RMSE is dependent on scale, and should not be used to compare datasets of different types.

## Metrics for categorical prediction

This section defines the metrics for categorical prediction in SageMaker Canvas and gives you information about how you can use them.

The following is a list of available metrics for 2-category prediction:

- Accuracy – The percentage of correct predictions.

Or, the ratio of the number of correctly predicted items to the total number of predictions. Accuracy measures how close the predicted class values are to the actual values. Values for accuracy metrics vary between zero (0) and one (1). A value of 1 indicates perfect accuracy, and 0 indicates complete inaccuracy.

- AUC – A value between 0 and 1 that indicates how well your model is able to separate the categories in your dataset. A value of 1 indicates that it was able to separate the categories perfectly.
- BalancedAccuracy – Measures the ratio of accurate predictions to all predictions.

This ratio is calculated after normalizing true positives (TP) and true negatives (TN) by the total number of positive (P) and negative (N) values. It is defined as follows:  $0.5 * ((TP/P) + (TN/N))$ , with values ranging from 0 to 1. The balanced accuracy metric gives a better measure of accuracy when the number of positives or negatives differ greatly from each other in an imbalanced dataset, such as when only 1% of email is spam.

- F1 – A balanced measure of accuracy that takes class balance into account.

It is the harmonic mean of the precision and recall scores, defined as follows:  $F1 = 2 * (precision * recall) / (precision + recall)$ . F1 scores vary between 0 and 1. A score of 1 indicates the best possible performance, and 0 indicates the worst.

- InferenceLatency – The approximate amount of time between making a request for a model prediction to receiving it from a real-time endpoint to which the model is deployed. This metric is measured in seconds and is only available for models built with the **Ensembling** mode.

- LogLoss – Log loss, also known as cross-entropy loss, is a metric used to evaluate the quality of the probability outputs, rather than the outputs themselves. Log loss is an important metric to indicate when a model makes incorrect predictions with high probabilities. Values range from 0 to infinity. A value of 0 represents a model that perfectly predicts the data.
- Precision – Of all the times that {category x} was predicted, the prediction was correct {precision}% of the time.

Precision measures how well an algorithm predicts the true positives (TP) out of all of the positives that it identifies. It is defined as follows: Precision = TP/(TP+FP), with values ranging from zero (0) to one (1). Precision is an important metric when the cost of a false positive is high. For example, the cost of a false positive is very high if an airplane safety system is falsely deemed safe to fly. A false positive (FP) reflects a positive prediction that is actually negative in the data.

- Recall – The model correctly predicted {recall}% to be {category x} when {target\_column} was actually {category x}.

Recall measures how well an algorithm correctly predicts all of the true positives (TP) in a dataset. A true positive is a positive prediction that is also an actual positive value in the data. Recall is defined as follows: Recall = TP/(TP+FN), with values ranging from 0 to 1. Higher scores reflect a better ability of the model to predict true positives (TP) in the data. Note that it is often insufficient to measure only recall, because predicting every output as a true positive yields a perfect recall score.

The following is a list of available metrics for 3+ category prediction:

- Accuracy – The percentage of correct predictions.

Or, the ratio of the number of correctly predicted items to the total number of predictions. Accuracy measures how close the predicted class values are to the actual values. Values for accuracy metrics vary between zero (0) and one (1). A value of 1 indicates perfect accuracy, and 0 indicates complete inaccuracy.

- BalancedAccuracy – Measures the ratio of accurate predictions to all predictions.

This ratio is calculated after normalizing true positives (TP) and true negatives (TN) by the total number of positive (P) and negative (N) values. It is defined as follows:  $0.5 * ((TP/P) + (TN/N))$ , with values ranging from 0 to 1. The balanced accuracy metric gives a better measure of accuracy

when the number of positives or negatives differ greatly from each other in an imbalanced dataset, such as when only 1% of email is spam.

- **F1macro** – The F1macro score applies F1 scoring by calculating the precision and recall, and then taking their harmonic mean to calculate the F1 score for each class. Then, the F1macro averages the individual scores to obtain the F1macro score. F1macro scores vary between 0 and 1. A score of 1 indicates the best possible performance, and 0 indicates the worst.
- **InferenceLatency** – The approximate amount of time between making a request for a model prediction to receiving it from a real-time endpoint to which the model is deployed. This metric is measured in seconds and is only available for models built with the **Ensembling** mode.
- **LogLoss** – Log loss, also known as cross-entropy loss, is a metric used to evaluate the quality of the probability outputs, rather than the outputs themselves. Log loss is an important metric to indicate when a model makes incorrect predictions with high probabilities. Values range from 0 to infinity. A value of 0 represents a model that perfectly predicts the data.
- **PrecisionMacro** – Measures precision by calculating precision for each class and averaging scores to obtain precision for several classes. Scores range from zero (0) to one (1). Higher scores reflect the model's ability to predict true positives (TP) out of all of the positives that it identifies, averaged across multiple classes.
- **RecallMacro** – Measures recall by calculating recall for each class and averaging scores to obtain recall for several classes. Scores range from 0 to 1. Higher scores reflect the model's ability to predict true positives (TP) in a dataset, whereas a true positive reflects a positive prediction that is also an actual positive value in the data. It is often insufficient to measure only recall, because predicting every output as a true positive will yield a perfect recall score.

Note that for 3+ category prediction, you also receive the average F1, Accuracy, Precision, and Recall metrics. The scores for these metrics are just the metric scores averaged for all categories.

## Metrics for image and text prediction

The following is a list of available metrics for image prediction and text prediction.

- **Accuracy** – The percentage of correct predictions.

Or, the ratio of the number of correctly predicted items to the total number of predictions. Accuracy measures how close the predicted class values are to the actual values. Values for accuracy metrics vary between zero (0) and one (1). A value of 1 indicates perfect accuracy, and 0 indicates complete inaccuracy.

- **F1** – A balanced measure of accuracy that takes class balance into account.

It is the harmonic mean of the precision and recall scores, defined as follows:  $F1 = 2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$ . F1 scores vary between 0 and 1. A score of 1 indicates the best possible performance, and 0 indicates the worst.

- Precision – Of all the times that {category x} was predicted, the prediction was correct {precision}% of the time.

Precision measures how well an algorithm predicts the true positives (TP) out of all of the positives that it identifies. It is defined as follows:  $\text{Precision} = \text{TP}/(\text{TP}+\text{FP})$ , with values ranging from zero (0) to one (1). Precision is an important metric when the cost of a false positive is high. For example, the cost of a false positive is very high if an airplane safety system is falsely deemed safe to fly. A false positive (FP) reflects a positive prediction that is actually negative in the data.

- Recall – The model correctly predicted {recall}% to be {category x} when {target\_column} was actually {category x}.

Recall measures how well an algorithm correctly predicts all of the true positives (TP) in a dataset. A true positive is a positive prediction that is also an actual positive value in the data. Recall is defined as follows:  $\text{Recall} = \text{TP}/(\text{TP}+\text{FN})$ , with values ranging from 0 to 1. Higher scores reflect a better ability of the model to predict true positives (TP) in the data. Note that it is often insufficient to measure only recall, because predicting every output as a true positive yields a perfect recall score.

Note that for image and text prediction models where you are predicting 3 or more categories, you also receive the *average* F1, Accuracy, Precision, and Recall metrics. The scores for these metrics are just the metric scores average for all categories.

## Metrics for time series forecasts

The following defines the advanced metrics for time series forecasts in Amazon SageMaker Canvas and gives you information about how you can use them.

- Average Weighted Quantile Loss (wQL) – Evaluates the forecast by averaging the accuracy at the P10, P50, and P90 quantiles. A lower value indicates a more accurate model.
- Weighted Absolute Percent Error (WAPE) – The sum of the absolute error normalized by the sum of the absolute target, which measures the overall deviation of forecasted values from observed values. A lower value indicates a more accurate model, where WAPE = 0 is a model with no errors.

- Root Mean Square Error (RMSE) – The square root of the average squared errors. A lower RMSE indicates a more accurate model, where  $\text{RMSE} = 0$  is a model with no errors.
- Mean Absolute Percent Error (MAPE) – The percentage error (percent difference of the mean forecasted value versus the actual value) averaged over all time points. A lower value indicates a more accurate model, where  $\text{MAPE} = 0$  is a model with no errors.
- Mean Absolute Scaled Error (MASE) – The mean absolute error of the forecast normalized by the mean absolute error of a simple baseline forecasting method. A lower value indicates a more accurate model, where  $\text{MASE} < 1$  is estimated to be better than the baseline and  $\text{MASE} > 1$  is estimated to be worse than the baseline.

## Predictions with custom models

Use the custom model that you've built in SageMaker Canvas to make predictions for your data. The following sections show you how to make predictions for numeric and categorical prediction models, time series forecasts, image prediction models, and text prediction models.

Numeric and categorical prediction, image prediction, and text prediction custom models support making the following types of predictions for your data:

- **Single predictions** — A **Single prediction** is when you only need to make one prediction. For example, you have one image or passage of text that you want to classify.
- **Batch predictions** — A **Batch prediction** is when you'd like to make predictions for an entire dataset. You can make batch predictions for datasets that are 1 TB+. For example, you have a CSV file of customer reviews for which you'd like to predict the customer sentiment, or you have a folder of image files that you'd like to classify. You should make predictions with a dataset that matches your input dataset. Canvas provides you with the ability to do manual batch predictions, or you can configure automatic batch predictions that run whenever you update a dataset.

For each prediction or set of predictions, SageMaker Canvas returns the following:

- The predicted values
- The probability of the predicted value being correct

## Get started

Choose one of the following workflows to make predictions with your custom model:

- [Batch predictions in SageMaker Canvas](#)
- [Make single predictions](#)

After generating predictions with your model, you can also do the following:

- [Update your model by adding versions.](#) If you want to try to improve the prediction accuracy of your model, you can build new versions of your model. You can choose to clone your original model building configuration and dataset, or you can change your configuration and select a different dataset. After adding a new version, you can review and compare versions to choose the best one.
- [Register a model version in the SageMaker AI model registry.](#) You can register versions of your model to the SageMaker Model Registry, which is a feature for tracking and managing the status of model versions and machine learning pipelines. A data scientist or MLOps team user with access to the SageMaker Model Registry can review your model versions and approve or reject them before deploying them to production.
- [Send your batch predictions to Amazon QuickSight.](#) In Amazon QuickSight, you can build and publish dashboards with your batch prediction datasets. This can help you analyze and share results generated by your custom model.

## Make single predictions

### Note

This section describes how to get single predictions from your model inside the Canvas application. For information about making real-time invocations in a production environment by deploying your model to an endpoint, see [Deploy your models to an endpoint](#).

Make single predictions if you want to get a prediction for a single data point. You can use this feature to get real-time predictions or to experiment with changing individual values to see how they impact the prediction outcome. Note that single predictions rely on an Asynchronous Inference endpoint, which shuts down after being idle (or not receiving any prediction requests) for two hours.

Choose one of the following procedures based on your model type.

## Make single predictions with numeric and categorical prediction models

To make a single prediction for a numeric or categorical prediction model, do the following:

1. In the left navigation pane of the Canvas application, choose **My models**.
2. On the **My models** page, choose your model.
3. After opening your model, choose the **Predict** tab.
4. On the **Run predictions** page, choose **Single prediction**.
5. For each **Column** field, which represents the columns of your input data, you can change the **Value**. Select the dropdown menu for the **Value** you want to change. For numeric fields, you can enter a new number. For fields with labels, you can select a different label.
6. When you're ready to generate the prediction, in the right **Prediction** pane, choose **Update**.

In the right **Prediction** pane, you'll see the prediction result. You can **Copy** the prediction result chart, or you can also choose **Download** to either download the prediction result chart as an image or to download the values and prediction as a CSV file.

## Make single predictions with time series forecasting models

To make a single prediction for a time series forecasting model, do the following:

1. In the left navigation pane of the Canvas application, choose **My models**.
2. On the **My models** page, choose your model.
3. After opening your model, choose the **Predict** tab.
4. Choose **Single prediction**.
5. For **Item**, select the item for which you want to forecast values.
6. If you used a group by column to train the model, then select the group by category for the item.

The prediction result loads in the pane below, showing you a chart with the forecast for each quantile. Choose **Schema view** to see the numeric predicted values. You can also choose **Download** to download the prediction results as either an image or a CSV file.

## Make single predictions with image prediction models

To make a single prediction for a single-label image prediction model, do the following:

1. In the left navigation pane of the Canvas application, choose **My models**.
2. On the **My models** page, choose your model.
3. After opening your model, choose the **Predict** tab.
4. On the **Run predictions** page, choose **Single prediction**.
5. Choose **Import image**.
6. You'll be prompted to upload an image. You can upload an image from your local computer or from an Amazon S3 bucket.
7. Choose **Import** to import your image and generate the prediction.

In the right **Prediction results** pane, the model lists the possible labels for the image along with a **Confidence** score for each label. For example, the model might predict the label **Sea** for an image, with a confidence score of 96%. The model may have predicted the image as a **Glacier** with only a confidence score of 4%. Therefore, you can determine that your model is fairly confident in predicting images of the sea.

## Make single predictions with text prediction models

To make a single prediction for a multi-category text prediction model, do the following:

1. In the left navigation pane of the Canvas application, choose **My models**.
2. On the **My models** page, choose your model.
3. After opening your model, choose the **Predict** tab.
4. On the **Run predictions** page, choose **Single prediction**.
5. For the **Text field**, enter the text for which you'd like to get a prediction.
6. Choose **Generate prediction results** to get your prediction.

In the right **Prediction results** pane, you receive an analysis of your text in addition to a **Confidence** score for each possible label. For example, if you entered a good review for a product, you might get **Positive** with a confidence score of 85%, while the confidence score for **Neutral** might be 10% and the confidence score for **Negative** only 5%.

## Batch predictions in SageMaker Canvas

Make batch predictions when you have an entire dataset for which you'd like to generate predictions. Amazon SageMaker Canvas supports batch predictions for datasets up to PBs in size.

There are two types of batch predictions you can make:

- [Manual batch predictions](#) are when you have a dataset for which you want to make one-time predictions.
- *Automatic* batch predictions are when you set up a configuration that runs whenever a specific dataset is updated. For example, if you've configured weekly updates to a SageMaker Canvas dataset of inventory data, you can set up automatic batch predictions that run whenever you update the dataset. After setting up an automated batch predictions workflow, see [How to manage automations](#) for more information about viewing and editing the details of your configuration. For more information about setting up automatic dataset updates, see [Configure automatic updates for a dataset](#).

 **Note**

You can only set up automatic batch predictions for datasets imported through local upload or Amazon S3. Additionally, automatic batch predictions can only run while you're logged in to the Canvas application. If you log out of Canvas, the automatic batch prediction job resumes when you log back in.

To get started, review the [Batch prediction dataset requirements](#), and then choose one of the following manual or automatic batch prediction workflows.

## Topics

- [Batch prediction dataset requirements](#)
- [Make manual batch predictions](#)
- [Make automatic batch predictions](#)
- [Edit your automatic batch prediction configuration](#)
- [Delete your automatic batch prediction configuration](#)
- [View your batch prediction jobs](#)

## Batch prediction dataset requirements

For batch predictions, make sure that your datasets meet the requirements outlined in [Create a dataset](#). If your dataset is larger than 5 GB, then Canvas uses Amazon EMR Serverless to process your data and split it into smaller batches. After your data has been split, Canvas uses SageMaker

AI Batch Transform to make predictions. You may see charges from both of these services after running batch predictions. For more information, see [Canvas pricing](#).

You might not be able to make predictions on some datasets if they have incompatible *schemas*. A *schema* is an organizational structure. For a tabular dataset, the schema is the names of the columns and the data type of the data in the columns. An incompatible schema might happen for one of the following reasons:

- The dataset that you're using to make predictions has fewer columns than the dataset that you're using to build the model.
- The data types in the columns you used to build the dataset might be different from the data types in dataset that you're using to make predictions.
- The dataset that you're using to make predictions and the dataset that you've used to build the model have column names that don't match. The column names are case sensitive. Column1 is not the same as column1.

To ensure that you can successfully generate batch predictions, match the schema of your batch predictions dataset to the dataset you used to train the model.

 **Note**

For batch predictions, if you dropped any columns when building your model, Canvas adds the dropped columns back to the prediction results. However, Canvas does not add the dropped columns to your batch predictions for time series models.

## Make manual batch predictions

Choose one of the following procedures to make manual batch predictions based on your model type.

### Make manual batch predictions with numeric, categorical, and time series forecasting models

To make manual batch predictions for numeric, categorical, and time series forecasting model types, do the following:

1. In the left navigation pane of the Canvas application, choose **My models**.
2. On the **My models** page, choose your model.

3. After opening your model, choose the **Predict** tab.
4. On the **Run predictions** page, choose **Batch prediction**.
5. Choose **Select dataset** to pick a dataset for generating predictions.
6. From the list of available datasets, select your dataset, and then choose **Start Predictions** to get your predictions.

After the prediction job finishes running, there is an output dataset listed on the same page in the **Predictions** section. This dataset contains your results, and if you select the **More options** icon (),

you can choose **Preview** to preview the output data. You can see the input data matched to the prediction and the probability that the prediction is correct. Then, you can choose **Download prediction** to download the results as a file.

## Make manual batch predictions with image prediction models

To make manual batch predictions for a single-label image prediction model, do the following:

1. In the left navigation pane of the Canvas application, choose **My models**.
2. On the **My models** page, choose your model.
3. After opening your model, choose the **Predict** tab.
4. On the **Run predictions** page, choose **Batch prediction**.
5. Choose **Select dataset** if you've already imported your dataset. If not, choose **Import new dataset**, and then you'll be directed through the import data workflow.
6. From the list of available datasets, select your dataset and choose **Generate predictions** to get your predictions.

After the prediction job finishes running, on the **Run predictions** page, you see an output dataset listed under **Predictions**. This dataset contains your results, and if you select the **More options** icon ()

you can choose **View prediction results** to see the output data. You can see the images along with their predicted labels and confidence scores. Then, you can choose **Download prediction** to download the results as a CSV or a ZIP file.

## Make manual batch predictions with text prediction models

To make manual batch predictions for a multi-category text prediction model, do the following:

1. In the left navigation pane of the Canvas application, choose **My models**.
2. On the **My models** page, choose your model.
3. After opening your model, choose the **Predict** tab.
4. On the **Run predictions** page, choose **Batch prediction**.
5. Choose **Select dataset** if you've already imported your dataset. If not, choose **Import new dataset**, and then you'll be directed through the import data workflow. The dataset you choose must have the same source column as the dataset with which you built the model.
6. From the list of available datasets, select your dataset and choose **Generate predictions** to get your predictions.

After the prediction job finishes running, on the **Run predictions** page, you see an output dataset listed under **Predictions**. This dataset contains your results, and if you select the **More options** icon (),

you can choose **Preview** to see the output data. You can see the images along with their predicted labels and confidence scores. Then, you can choose **Download prediction** to download the results.

## Make automatic batch predictions

To set up a schedule for automatic batch predictions, do the following:

1. In the left navigation pane of Canvas, choose **My models**.
2. Choose your model.
3. Choose the **Predict** tab.
4. Choose **Batch prediction**.
5. For **Generate predictions**, choose **Automatic**.
6. The **Automate batch predictions** dialog box pops up. Choose **Select dataset** and choose the dataset for which you want to automate predictions. Note that you can only select a dataset that was imported through local upload or Amazon S3.
7. After selecting a dataset, choose **Set up**.

Canvas runs a batch predictions job for the dataset after you set up the configuration. Then, every time you [Update a dataset](#), either manually or automatically, another batch predictions job runs.

After the prediction job finishes running, on the **Run predictions** page, you see an output dataset listed under **Predictions**. This dataset contains your results, and if you select the **More options** icon

( ! ),

you can choose **Preview** to preview the output data. You can see the input data matched to the prediction and the probability that the prediction is correct. Then, you can choose **Download** to download the results.

The following sections describe how to view, update, and delete your automatic batch prediction configuration through the **Datasets** page in the Canvas application. You can only set up a maximum of 20 automatic configurations in Canvas. For more information about viewing your automated batch predictions job history or making changes to your automatic configuration through the **Automations** page, see [How to manage automations](#).

## Edit your automatic batch prediction configuration

You might want to make changes to your auto update configuration for a dataset, such as changing the frequency of the updates. You might also want to turn off your automatic update configuration to pause the updates to your dataset.

When you edit a batch prediction configuration, you can change the target dataset but not the frequency (since automatic batch predictions occur whenever the dataset is updated).

To edit your auto update configuration, do the following:

1. Go to the **Predict** tab of your model.
2. Under **Predictions**, choose the **Configuration** tab.
3. Find your configuration and choose the **More options** icon ( ! ).
4. From the dropdown menu, choose **Update configuration**.
5. The **Automate batch prediction** dialog box opens. You can select another dataset and choose **Set up** to save your changes.

Your automatic batch predictions configuration is now updated.

To pause your automatic batch predictions, turn off your automatic configuration by doing the following:

1. Go to the **Predict** tab of your model.
2. Under **Predictions**, choose the **Configuration** tab.
3. Find your configuration from the list and turn off the **Auto update** toggle.

Automatic batch predictions are now paused. You can turn the toggle back on at any time to resume the update schedule.

## Delete your automatic batch prediction configuration

To learn how to delete your automatic batch prediction configuration, see [Delete an automatic configuration](#).

You can also delete your configuration by doing the following:

1. Go to the **Predict** tab of your model.
2. Under **Predictions**, choose the **Configuration** tab.
3. Find your configuration from the list and choose the **More options** icon ().
4. From the dropdown menu, choose **Delete configuration**.

Your configuration should now be deleted.

## View your batch prediction jobs

To view the statuses and history of your batch prediction jobs, go to the **Predict** tab of your model.

Each batch prediction job shows up in the **Predict** tab of your model. Under **Predictions**, you can see the **All jobs** tab and the **Configuration** tabs:

- **All jobs** – In this tab, you can see all of the manual and automatic batch prediction jobs for this model. You can filter the jobs by configuration name. For each job, you can see the following fields:
  - **Status** – The current status of your batch prediction job. If the status is **Failed** or **Partially failed**, you can hover over the status to view a more detailed error message to help you troubleshoot.
  - **Input dataset** – The name of your Canvas input dataset, including the dataset version.
  - **Prediction type** – Whether the prediction job was automatic or manual.
  - **Rows** – The number of rows predicted.
  - **Configuration name** – The name of the batch prediction job configuration.
  - **QuickSight** – Describes whether you've sent the batch predictions to Amazon QuickSight.
  - **Created** – The creation time of the batch prediction job.

## If you choose the **More options** icon

( ! ),

you can choose **View details**, **Preview prediction**, **Download prediction**, or **Send to Amazon QuickSight**. If you choose **View details**, a page opens that shows you the full details of the batch prediction job, including the status, the input and output data configurations, information about the instances used to complete the job and access to the Amazon CloudWatch logs. The page looks like the following screenshot.

The screenshot shows the configuration page for a batch prediction job named "Sales-predictor-batch-inference". The left sidebar includes links for Home, Data Wrangler, Datasets, My Models (which is selected), ML Ops, Ready-to-use, and Gen AI. The main content area is divided into several sections:

- Job summary:** Displays the job name "Sales-predictor-batch-inference", status "Ready", configuration name "SalesPredictorConfig", and creation date "04/26/2024 10:45 PM". It also shows the input dataset "Sales\_data", prediction type "Manual", instance type "ml.m5.4xlarge", and instance count "2". A "View logs" link is provided.
- Input data configuration:** Shows S3 data type "S3 Prefix" set to "text/csv", split type "Line", compression type "None", and content type "text/csv". The S3 URI field contains "s3://".
- Output data configuration:** Shows output data encryption key as "-", accept "text/csv", assemble with "Line", and S3 output path "s3://".
- Environment variables:** Lists Region (North America) and Team (Sales).

- Configuration** – In this tab, you can see all of the automatic batch prediction configurations you've created for this model. For each configuration, you can see fields such as the timestamp for when it was **Created**, the **Input dataset** it

tracks for updates, and the **Next job scheduled**, which is the time when the next automatic prediction job is scheduled to start. If you choose the **More options** icon (), you can choose **View all jobs** to see the job history and in progress jobs for the configuration.

## Send predictions to Amazon QuickSight

### Note

You can send batch predictions to Amazon QuickSight for numeric and categorical prediction and time series forecasting models. Single-label image prediction and multi-category text prediction models are excluded.

Once you generate batch predictions with custom tabular models in SageMaker Canvas, you can send those predictions as CSV files to Amazon QuickSight, which is a business intelligence (BI) service to build and publish predictive dashboards.

For example, if you built a 2 category prediction model to determine whether a customer will churn, you can create a visual, predictive dashboard in Amazon QuickSight to show the percentage of customers that are expected to churn. To learn more about Amazon QuickSight, see the [Amazon QuickSight User Guide](#).

The following sections show you how to send your batch predictions to Amazon QuickSight for analysis.

### Before you begin

Your user must have the necessary AWS Identity and Access Management (IAM) permissions to send your predictions to Amazon QuickSight. Your administrator can set up the IAM permissions for your user. For more information, see [Grant Your Users Permissions to Send Predictions to Amazon QuickSight](#).

Your Amazon QuickSight account must contain the default namespace, which is set up when you first create your Amazon QuickSight account. Contact your administrator to help you get access to Amazon QuickSight. For more information, see [Setting up for Amazon QuickSight](#) in the *Amazon QuickSight User Guide*.

Your Amazon QuickSight account must be created in the same Region as your Canvas application. If your Amazon QuickSight account's home Region differs from your Canvas application's Region, you must either [close](#) and recreate your Amazon QuickSight account, or [set up a Canvas application](#) in the same Region as your Amazon QuickSight account. You can check your Amazon QuickSight home Region by doing the following (assuming you already have an Amazon QuickSight account):

1. Open your [Amazon QuickSight console](#).
2. When the page loads, your Amazon QuickSight home Region is appended to the URL in the following format: `https://<your-home-region>.quicksight.aws.amazon.com/`.

You must know the usernames of the Amazon QuickSight users to whom you want to send your predictions. You can send predictions to yourself or other users who have the right permissions. Any users to whom you send predictions must be in the default [namespace](#) of your Amazon QuickSight account and have the Author or Admin role in Amazon QuickSight.

Additionally, Amazon QuickSight must have access to the SageMaker AI default Amazon S3 bucket for your domain, which is named with the following format: `sagemaker-{REGION}-{ACCOUNT_ID}`. The Region should be the same as your Amazon QuickSight account's home Region and your Canvas application's Region. To learn how to give Amazon QuickSight access to the batch predictions stored in your Amazon S3 bucket, see the topic [I can't connect to Amazon S3](#) in the *Amazon QuickSight User Guide*.

## Supported data formats

Before sending your predictions, check that the data format of your batch predictions is compatible with Amazon QuickSight.

- To learn more about the accepted data formats for timeseries data, see [Supported date formats](#) in the *Amazon QuickSight User Guide*.
- To learn more about data values that might prevent you from sending to Amazon QuickSight, see [Unsupported values in data](#) in the *Amazon QuickSight User Guide*.

Also note that Amazon QuickSight uses the character " as a text qualifier, so if your Canvas data contains any " characters, make sure that you close all matching quotes. Any mismatching quotes can cause issues with sending your dataset to Amazon QuickSight.

## Send your batch predictions to Amazon QuickSight

Use the following procedure to send your predictions to Amazon QuickSight:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **My models**.
3. On the **My models** page, choose your model.
4. Choose the **Predict** tab.
5. Under **Predictions**, select the dataset (or datasets) of batch predictions that you'd like to share. You can share up to 5 datasets of batch predictions at a time.
6. After you select your dataset, choose **Send to Amazon QuickSight**.

 **Note**

The **Send to Amazon QuickSight** button doesn't activate unless you select one or more datasets.

Alternatively, you can preview your predictions by choosing the **More options** icon (⋮) and then **View prediction results**. From the dataset preview, you can choose **Send to Amazon QuickSight**. The following screenshot shows you the **Send to Amazon QuickSight** button in a dataset preview.

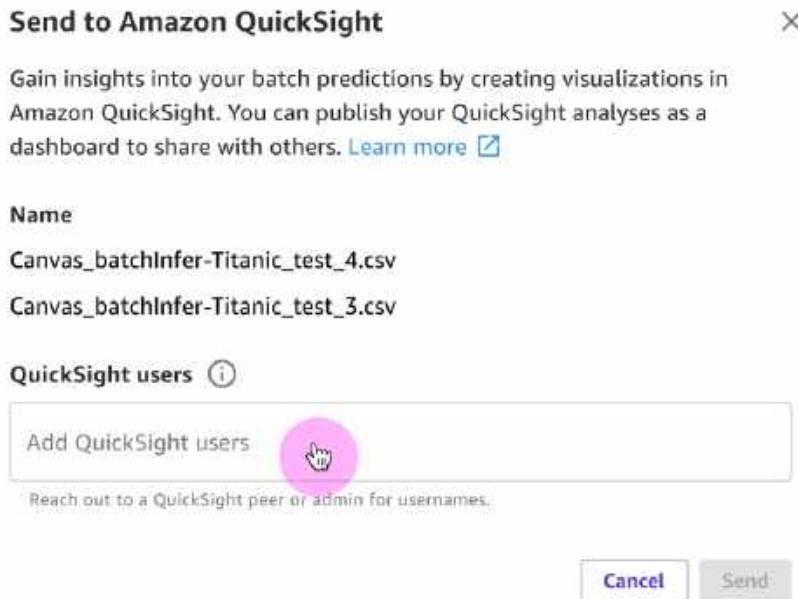
Canvas_batchInfer-Titanic_test_2									X
Prediction & probability		Input dataset 							
Survived	Probability	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	
Yes	81.4%	7892-POOKP	Female	0	Yes	No	28	Yes	
Yes	80.2%	9237-HQITU	Female	0	No	No	2	Yes	
Yes	78.6%	9305-CDSKC	Female	0	No	No	8	Yes	
Yes	77.6%	4190-MFLUW	Female	0	Yes	Yes	10	Yes	
Yes	76.1%	0280-XJGEX	Male	0	No	No	49	Yes	
Yes	50.3%	5668-QPYBK	Male	0	No	No	2	Yes	
No	90.1%	3655-SNQYZ	Female	0	Yes	Yes	69	Yes	
No	88.3%	5129-JLPIS	Male	0	No	No	25	Yes	
No	84.3%	5575-GNVDE	Male	0	No	No	34	Yes	
No	81.1%	9959-WOKFT	Male	0	No	Yes	71	Yes	
No	79.3%	8091-TTVAX	Male	0	Yes	No	58	Yes	
No	72.0%	6588-TABGU	Male	0	No	Yes	62	Yes	
Nn	71.9%	7745-CFFCRW	Male	n	Nn	Nn	45	Nn	

 Send to Amazon QuickSight
 Download CSV

## 7. In the **Send to Amazon QuickSight** dialog box, do the following:

- For **QuickSight users**, enter the name of the Amazon QuickSight users to whom you want to send your predictions. If you want to send them to yourself, enter your own username. You can only send predictions to users in the default namespace of the Amazon QuickSight account, and the user must have the Author or Admin role in Amazon QuickSight.
- Choose **Send**.

The following screenshot shows the **Send to Amazon QuickSight** dialog box:



After you send your batch predictions, the **QuickSight** field for the datasets you sent shows as Sent. In the confirmation box that confirms your predictions were sent, you can choose **Open Amazon QuickSight** to open your Amazon QuickSight application. If you're done using Canvas, you should [log out](#) of the Canvas application.

The Amazon QuickSight users that you've sent datasets to can open their Amazon QuickSight application and view the Canvas datasets that have been shared with them. Then, they can create predictive dashboards with the data. For more information, see [Getting started with Amazon QuickSight data analysis](#) in the *Amazon QuickSight User Guide*.

By default, all of the users to whom you send predictions have owner permissions for the dataset in Amazon QuickSight. Owners are able to create analyses, refresh, edit, delete, and re-share datasets. The changes that owners make to a dataset change the dataset for all users with access. To change the permissions, go to the dataset in Amazon QuickSight and manage its permissions. For more information, see [Viewing and editing the permissions users that a dataset is shared with](#) in the *Amazon QuickSight User Guide*.

## Download a model notebook

### Note

The model notebook feature is available for quick build and standard build tabular models, and fine-tuned foundation models. Model notebooks aren't supported for image prediction, text prediction, or time series forecasting models.

If you'd like to generate a model notebook for a tabular model built before this feature was launched, you must rebuild the model to generate a notebook.

For eligible models that you successfully build in Amazon SageMaker Canvas, a Jupyter notebook containing a report of all the model building steps is generated. This Jupyter notebook contains Python code that you can run locally or run in an environment like Amazon SageMaker Studio Classic to replicate the steps necessary to build your model. The notebook can be useful if you'd like to experiment with the code or see the backend details of how Canvas builds models.

To access the model notebook, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **My models**.
3. Choose the model and version that you built.
4. On the model version's page, choose the **More options** icon ( ) in the header.
5. From the dropdown menu, choose **View Notebook**.
6. A popup appears with the notebook content. You can choose **Download** and then do one of the following:
  - a. Choose **Download** to save the notebook content to your local device.
  - b. Choose **Copy S3 URI** to copy the Amazon S3 location where the notebook is stored. The notebook is stored in the Amazon S3 bucket specified in your **Canvas storage configuration**, which is configured in the [Prerequisites for setting up Amazon SageMaker Canvas](#) section.

You should now be able to view the notebook either locally or as an object in Amazon S3. You can upload the notebook to an IDE to edit and run the code, or you can share the notebook with others in your organization to review.

## Send your model to Amazon QuickSight

If you use Amazon QuickSight and want to leverage SageMaker Canvas in your Amazon QuickSight visualizations, you can build an Amazon SageMaker Canvas model and use it as a *predictive field* in your Amazon QuickSight dataset. A *predictive field* is a field in your Amazon QuickSight dataset that can make predictions for a given column in your dataset, similar to how Canvas users make single or batch predictions with a model. To learn more about how to integrate Canvas predictive abilities into your Amazon QuickSight datasets, see [SageMaker Canvas integration](#) in the [Amazon QuickSight User Guide](#).

The following steps explain how you can add a predictive field to your Amazon QuickSight dataset using a Canvas model:

1. Open the Canvas application and build a model with your dataset.
2. After building the model in Canvas, send the model to Amazon QuickSight. A schema file automatically downloads to your local machine when you send the model to Amazon QuickSight. You upload this schema file to Amazon QuickSight in the next step.
3. Open Amazon QuickSight and choose a dataset with the same schema as the dataset you used to build your model. Add a predictive field to the dataset and do the following:
  - a. Specify the model sent from Canvas.
  - b. Upload the schema file that was downloaded in Step 2.
4. Save and publish your changes, and then generate predictions for the new dataset. Amazon QuickSight uses the model to fill in the target column with predictions.

In order to send a model from Canvas to Amazon QuickSight, you must meet the following prerequisites:

- You must have both Canvas and Amazon QuickSight set up. Your Amazon QuickSight account must be created in the same AWS Region as your Canvas application. If your Amazon QuickSight account's home Region differs from your Canvas application's Region, you must either [close](#) and recreate your Amazon QuickSight account, or [set up a Canvas application](#) in the same Region as your Amazon QuickSight account. Your Amazon QuickSight account must also contain the

default namespace, which you set up when you first create your Amazon QuickSight account. Contact your administrator to help you get access to Amazon QuickSight. For more information, see [Setting up for Amazon QuickSight](#) in the *Amazon QuickSight User Guide*.

- Your user must have the necessary AWS Identity and Access Management (IAM) permissions to send your predictions to Amazon QuickSight. Your administrator can set up the IAM permissions for your user. For more information, see [Grant Your Users Permissions to Send Predictions to Amazon QuickSight](#).
- Amazon QuickSight must have access to the Amazon S3 bucket that you've specified for Canvas application storage. For more information, see [Configure your Amazon S3 storage](#).

## Time Series Forecasts in Amazon SageMaker Canvas

### Note

Time series forecasting models are only supported for tabular datasets.

Amazon SageMaker Canvas gives you the ability to use machine learning time series forecasts. Time series forecasts give you the ability to make predictions that can vary with time.

You can make a time series forecast for the following examples:

- Forecasting your inventory in the coming months.
- The number of items sold in the next four months.
- The effect of reducing the price on sales during the holiday season.
- Item inventory in the next 12 months.
- The number of customers entering a store in the next several hours.
- Forecasting how a 10% reduction in the price of a product affects sales over a time period.

To make a time series forecast, your dataset must have the following:

- A timestamp column with all values having the `datetime` type.
- A target column that has the values that you're using to forecast future values.
- An item ID column that contains unique identifiers for each item in your dataset, such as SKU numbers.

The datetime values in the timestamp column must use one of the following formats:

- YYYY-MM-DD HH:MM:SS
- YYYY-MM-DDTHH:MM:SSZ
- YYYY-MM-DD
- MM/DD/YY
- MM/DD/YY HH:MM
- MM/DD/YYYY
- YYYY/MM/DD HH:MM:SS
- YYYY/MM/DD
- DD/MM/YYYY
- DD/MM/YY
- DD-MM-YY
- DD-MM-YYYY

You can make forecasts for the following intervals:

- 1 min
- 5 min
- 15 min
- 30 min
- 1 hour
- 1 day
- 1 week
- 1 month
- 1 year

### Future values in your input dataset

Canvas automatically detects columns in your dataset that might potentially contain future values. If present, these values can enhance the accuracy of predictions. Canvas marks these specific columns with a Future values label. Canvas infers the relationship between the data in these

columns and the target column that you are trying to predict, and utilizes that relationship to generate more accurate forecasts.

For example, you can forecast the amount of ice cream sold by a grocery store. To make a forecast, you must have a timestamp column and a column that indicates how much ice cream the grocery store sold. For a more accurate forecast, your dataset can also include the price, the ambient temperature, the flavor of the ice cream, or a unique identifier for the ice cream.

Ice cream sales might increase when the weather is warmer. A decrease in the price of the ice cream might result in more units sold. Having a column with ambient temperature data and a column with pricing data can improve your ability to forecast the number of units of ice cream the grocery store sells.

While providing future values is optional, it helps you to perform what-if analyses directly in the Canvas application, showing you how changes in future values could alter your predictions.

## Handling missing values

You might have missing data for different reasons. The reason for your missing data might inform how you want Canvas to impute it. For example, your organization might use an automatic system that only tracks when a sale happens. If you're using a dataset that comes from this type of automatic system, you have missing values in the target column.

### Important

If you have missing values in the target column, we recommend using a dataset that doesn't have them. SageMaker Canvas uses the target column to forecast future values. Missing values in the target column can greatly reduce the accuracy of the forecast.

For missing values in the dataset, Canvas automatically imputes the missing values for you by filling the target column with 0 and other numeric columns with the median value of the column.

However, you can select your own filling logic for the target column and other numeric columns in your datasets. Target columns have different filling guidelines and restrictions than the rest of the numeric columns. Target columns are filled up to the end of the historical period, whereas numeric columns are filled across both historical and future periods all the way to the end of the forecast horizon. Canvas only fills future values in a numeric column if your data has at least one record with a future timestamp and a value for that specific column.

You can choose one of the following filling logic options to impute missing values in your data:

- zero – Fill with 0.
- NaN – Fill with NaN, or not a number. This is only supported for the target column.
- mean – Fill with the mean value from the data series.
- median – Fill with the median value from the data series.
- min – Fill with the minimum value from the data series.
- max – Fill with the maximum value from the data series.

When choosing a filling logic, you should consider how your model interprets the logic. For example, in a retail scenario, recording zero sales of an available item is different from recording zero sales of an unavailable item, as the latter scenario doesn't necessarily imply a lack of customer interest in the unavailable item. In this case, filling with 0 in the target column of the dataset might cause the model to be under-biased in its predictions and infer a lack of customer interest in unavailable items. Conversely, filling with NaN might cause the model to ignore true occurrences of zero items being sold of available items.

## Types of forecasts

You can make one of the following types of forecasts:

- **Single item**
- **All items**

For a forecast on all the items in your dataset, SageMaker Canvas returns a forecast for the future values for each item in your dataset.

For a single item forecast, you specify the item and SageMaker Canvas returns a forecast for the future values. The forecast includes a line graph that plots the predicted values over time.

## Topics

- [Additional options for forecasting insights](#)

## Additional options for forecasting insights

In Amazon SageMaker Canvas, you can use the following optional methods to get more insights from your forecast:

- Group column
- Holiday schedule
- What-if scenario

You can specify a column in your dataset as a **Group column**. Amazon SageMaker Canvas groups the forecast by each value in the column. For example, you can group the forecast on columns containing price data or unique item identifiers. Grouping a forecast by a column lets you make more specific forecasts. For example, if you group a forecast on a column containing item identifiers, you can see the forecast for each item.

Overall sales of items might be impacted by the presence of holidays. For example, in the United States, the number of items sold in both November and December might differ greatly from the number of items sold in January. If you use the data from November and December to forecast the sales in January, your results might be inaccurate. Using a holiday schedule prevents you getting inaccurate results. You can use a holiday schedule for 251 countries.

For a forecast on a single item in your dataset, you can use what-if scenarios. A what-if scenario gives you the ability to change values in your data and change the forecast. For example, you can answer the following questions by using a what-if scenario, "What if I lowered prices? How would that affect the number of items sold?"

## Adding model versions in Amazon SageMaker Canvas

In Amazon SageMaker Canvas, you can update the models that you've built by adding *versions*. Each model that you build has a version number. The first model is version 1 or V1. You can use model versions to see changes in prediction accuracy when you update your data or use [advanced transformations](#).

When viewing your model, SageMaker Canvas shows you the model history so that you can compare all of the model versions that you built. You can also delete versions that are no longer useful to you. By creating multiple model versions and evaluating their accuracy, you can iteratively improve your model performance.

 **Note**

Text prediction and image prediction models only support one model version.

To add a model version, you can either clone an existing version or create a new version.

Cloning an existing version copies over the current model configuration, including the model recipe and the input dataset. Alternatively, you can create a new version if you want to configure a new model recipe or choose a different dataset.

If you create a new version and select a different dataset, you must choose a dataset with the same target column and schema as the dataset from version 1.

Before you can add a new version, you must successfully build at least one model version. Then, you can [register a model version in the SageMaker Model Registry](#). Use the registry for tracking model versions and for collaborating with Studio Classic users on production model approvals.

If you did a quick build for your first model version, you have the option to run a standard build when you add a version. Standard builds generally have higher accuracy. Therefore, if you feel confident in your quick build configuration, you can run a standard build to create a final version of your model. To learn more about the differences between quick builds and standard builds, see [How custom models work](#).

The following procedures show you how to add model versions; the procedure is different depending on whether you are adding a version of the same build type or a different build type (quick versus standard). Use the procedure **To add a new model version** to add versions of the same build type. To add a standard build model version after running a quick build, follow the procedure **To run a standard build**.

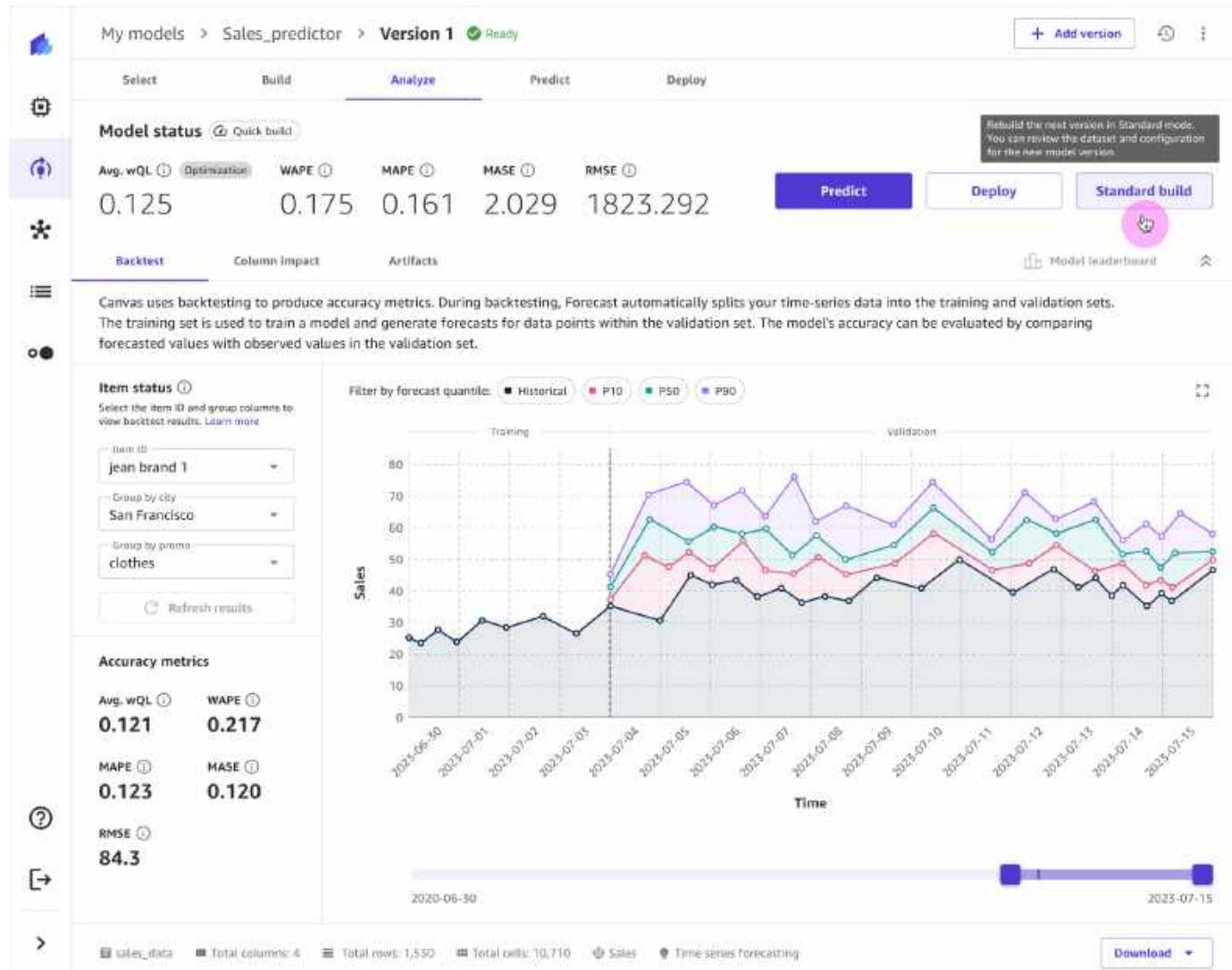
## To add a new model version

1. Open your SageMaker Canvas application. For more information, see [Getting started with using Amazon SageMaker Canvas](#).
2. In the left navigation pane, choose **My models**.
3. On the **My models** page, choose your model. To find your model, you can choose **Filter by problem type**.
4. After your model opens, choose the **Add version** button in the top panel.
5. From the dropdown menu, select one of the following options:
  - a. **Add a new version from scratch** – When you select this option, the **Build** tab opens with the draft for a new model version. You can select a different dataset (as long as the schema matches the schema of the first model version's dataset) and configure a new model recipe. For more information about building a model version, see [Build a model](#).

- b. **Clone an existing version with configurations** – A dialog box prompts you to select the version that you want to clone. After you've selected your desired version, choose **Clone**. The **Build** tab opens with the draft for a new model version. Any model recipe configurations are copied over from the cloned version. For more information about building a model version, see [Build a model](#).

## To run a standard build

1. Open your SageMaker Canvas application. For more information, see [Getting started with using Amazon SageMaker Canvas](#).
2. In the left navigation pane, choose **My models**.
3. On the **My models** page, choose your model. You can choose **Filter by problem type** to find your model more easily.
4. After your model opens, choose the **Analyze** tab.
5. Choose **Standard build**.



On the model draft page that opens to the **Build** tab, you can modify your model configuration and start a build. For more information about building a model version, see [Build a model](#).

You should now have a new model version build in progress. For more information about building a model, see [How custom models work](#).

After building a model version, you can return to your model details page at any time to view all of the versions or add more versions. The following image shows the **Versions** page for a model.

Version	Status	Created	Dataset	Model score	F1	Precision	Recall	AUC	Shared	Model registry
V1	Ready	05/04/2023 4:59 AM	train-test	79.215%	85.258%	82.145%	89.904%	0.794	—	Not Registered
V2	Building	05/04/2023 4:57 AM	train-test	85.140%	86.486%	84.956%	88.073%	0.852	—	Registered

On the **Versions** page, you can view the following information for each of your model versions:

- **Status** – This field tells you whether your model is currently building (In building), done building (Ready), failed to build (Failed), or still being edited (In draft).
- **Model score, F1, Precision, Recall, and AUC** – If you turn on the **Show advanced metrics** toggle on this page, you can see these model metrics. These metrics indicate the accuracy and performance of your model. For more information, see [Evaluate your model](#).
- **Shared** – This field states whether you shared the model version with SageMaker Studio Classic users.
- **Model registry** – This field states whether you registered the version to a model registry. For more information, see [Register a model version in the SageMaker AI model registry](#).

## MLOps

After building a model in SageMaker Canvas that you feel confident about, you might want to integrate your model with the machine learning operations (MLOps) processes in your organization. MLOps includes common tasks such as deploying a model for use in production or setting up continuous integration and continuous deployment (CI/CD) pipelines.

The following topics describe how you can use features within Canvas to use a Canvas-built model in production.

### Topics

- [Register a model version in the SageMaker AI model registry](#)
- [Deploy your models to an endpoint](#)
- [View your deployments](#)
- [Update a deployment configuration](#)
- [Test your deployment](#)

- [Invoke your endpoint](#)
- [Delete a model deployment](#)

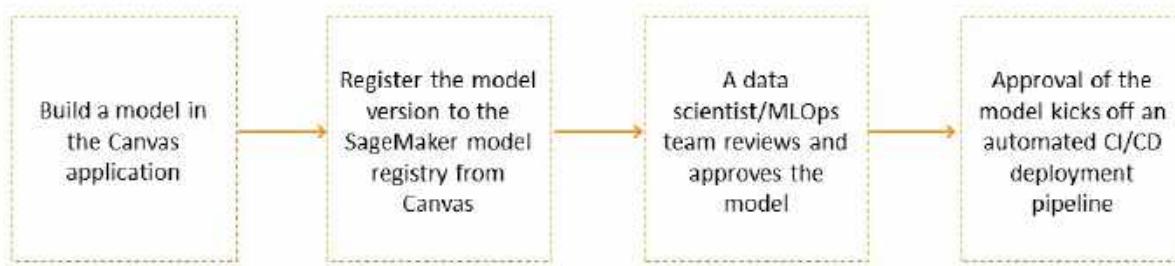
## Register a model version in the SageMaker AI model registry

With SageMaker Canvas, you can build multiple iterations, or versions, of your model to improve it over time. You might want to build a new version of your model if you acquire better training data or if you want to attempt to improve the model's accuracy. For more information about adding versions to your model, see [Update a model](#).

After you've [built a model](#) that you feel confident about, you might want to evaluate its performance and have it reviewed by a data scientist or MLOps engineer in your organization before using it in production. To do this, you can register your model versions to the [SageMaker Model Registry](#). The SageMaker Model Registry is a repository that data scientists or engineers can use to catalog machine learning (ML) models and manage model versions and their associated metadata, such as training metrics. They can also manage and log the approval status of a model.

After you register your model versions to the SageMaker Model Registry, a data scientist or your MLOps team can access the SageMaker Model Registry through [SageMaker Studio Classic](#), which is a web-based integrated development environment (IDE) for working with machine learning models. In the SageMaker Model Registry interface in Studio Classic, the data scientist or MLOps team can evaluate your model and update its approval status. If the model doesn't perform to their requirements, the data scientist or MLOps team can update the status to Rejected. If the model does perform to their requirements, then the data scientist or MLOps team can update the status to Approved. Then, they can [deploy your model to an endpoint](#) or [automate model deployment](#) with CI/CD pipelines. You can use the SageMaker AI model registry feature to seamlessly integrate models built in Canvas with the MLOps processes in your organization.

The following diagram summarizes an example of registering a model version built in Canvas to the SageMaker Model Registry for integration into an MLOps workflow.



You can register tabular, image, and text model versions to the SageMaker Model Registry. This includes time series forecasting models and JumpStart based [fine-tuned foundation models](#).

**Note**

Currently, you can't register Amazon Bedrock based fine-tuned foundation models built in Canvas to the SageMaker Model Registry.

The following sections show you how to register a model version to the SageMaker Model Registry from Canvas.

## Permissions management

By default, you have permissions to register model versions to the SageMaker Model Registry. SageMaker AI grants these permissions for all new and existing Canvas user profiles through the [AmazonSageMakerCanvasFullAccess](#) policy, which is attached to the AWS IAM execution role for the SageMaker AI domain that hosts your Canvas application.

If your Canvas administrator is setting up a new domain or user profile, when they're setting up the domain and following the prerequisite instructions in the [Getting started guide](#), SageMaker AI turns on the model registration permissions through the **ML Ops permissions configuration** option, which is enabled by default.

The Canvas administrator can manage model registration permissions at the user profile level as well. For example, if the administrator wants to grant model registration permissions to some user profiles but remove permissions for others, they can edit the permissions for a specific user.

The following procedure shows how to turn off model registration permissions for a specific user profile:

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the user profile's domain.
5. On the **domain details** page, choose the **User profile** whose permissions you want to edit.
6. On the **User Details** page, choose **Edit**.
7. In the left navigation pane, choose **Canvas settings**.
8. In the **ML Ops permissions configuration** section, turn off the **Enable Model Registry registration permissions** toggle.
9. Choose **Submit** to save the changes to your domain settings.

The user profile should no longer have model registration permissions.

## Register a model version to the SageMaker AI model registry

SageMaker Model Registry tracks all of the model versions that you build to solve a particular problem in a *model group*. When you build a SageMaker Canvas model and register it to SageMaker Model Registry, it gets added to a model group as a new model version. For example, if you build and register four versions of your model, then a data scientist or MLOps team working in the SageMaker Model Registry interface can view the model group and review all four versions of the model in one place.

When registering a Canvas model to the SageMaker Model Registry, a model group is automatically created and named after your Canvas model. Optionally, you can rename it to a name of your choice, or use an existing model group in the SageMaker Model Registry. For more information about creating a model group, see [Create a Model Group](#).

 **Note**

Currently, you can only register models built in Canvas to the SageMaker Model Registry in the same account.

To register a model version to the SageMaker Model Registry from the Canvas application, use the following procedure:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **My models**.
3. On the **My models** page, choose your model. You can **Filter by problem type** to find your model more easily.
4. After choosing your model, the **Versions** page opens, listing all of the versions of your model. You can turn on the **Show advanced metrics** toggle to view the advanced metrics, such as **Recall** and **Precision**, to compare your model versions and determine which one you'd like to register.
5. From the list of model versions, for the the version that you want to register, choose the **More options** icon ( ! ). Alternatively, you can double click on the version that you need to register, and then on the version details page, choose the **More options** icon ( ! ).
6. In the dropdown list, choose **Add to Model Registry**. The **Add to Model Registry** dialog box opens.
7. In the **Add to Model Registry** dialog box, do the following:
  - a. (Optional) In the **SageMaker Studio Classic model group** section, for the **Model group name** field, enter the name of the model group to which you want to register your version. You can specify the name for a new model group that SageMaker AI creates for you, or you can specify an existing model group. If you don't specify this field, Canvas registers your version to a default model group with the same name as your model.
  - b. Choose **Add**.

Your model version should now be registered to the model group in the SageMaker Model Registry. When you register a model version to a model group in the SageMaker Model Registry, all subsequent versions of the Canvas model are registered to the same model group (if you choose to register them). If you register your versions to a different model group, you need to go to the SageMaker Model Registry and [delete the model group](#). Then, you can re-register your model versions to the new model group.

To view the status of your models, you can return to the **Versions** page for your model in the Canvas application. This page shows you the **Model Registry** status of each version. If the status is Registered, then the model has been successfully registered.

If you want to view the details of your registered model version, for the **Model Registry** status, you can hover over the **Registered** field to see the **Model registry details** pop-up box. These details contain more info, such as the following:

- The **Model package group name** is the model group that your version is registered to in the SageMaker Model Registry.
- The **Approval status**, which can be Pending Approval, Approved, or Rejected. If a Studio Classic user approves or rejects your version in the SageMaker Model Registry, then this status is updated on your model versions page when you refresh the page.

The following screenshot shows the **Model registry details** box, along with an **Approval status** of Approved for this particular model version.

## Model Registry details

Model package group name	(i)	canvas-test-cv-v1
Model Registry version	(i)	Version 1
Model Registry account ID	(i)	[REDACTED]
Approval status	(i)	 Approved

## Deploy your models to an endpoint

In Amazon SageMaker Canvas, you can deploy your models to an endpoint to make predictions. SageMaker AI provides the ML infrastructure for you to host your model on an endpoint with the compute instances that you choose. Then, you can *invoke* the endpoint (send a prediction request) and get a real-time prediction from your model. With this functionality, you can use your model in production to respond to incoming requests, and you can integrate your model with existing applications and workflows.

To get started, you should have a model that you'd like to deploy. You can deploy custom model versions that you've built, Amazon SageMaker JumpStart foundation models, and fine-tuned JumpStart foundation models. For more information about building a model in Canvas, see [How custom models work](#). For more information about JumpStart foundation models in Canvas, see [Generative AI foundation models in SageMaker Canvas](#).

Review the following **Permissions management** section, and then begin creating new deployments in the **Deploy a model** section.

## Permissions management

By default, you have permissions to deploy models to SageMaker AI Hosting endpoints. SageMaker AI grants these permissions for all new and existing Canvas user profiles through the [AmazonSageMakerCanvasFullAccess](#) policy, which is attached to the AWS IAM execution role for the SageMaker AI domain that hosts your Canvas application.

If your Canvas administrator is setting up a new domain or user profile, when they're setting up the domain and following the prerequisite instructions in the [Prerequisites for setting up Amazon SageMaker Canvas](#), SageMaker AI turns on the model deployment permissions through the **Enable direct deployment of Canvas models** option, which is enabled by default.

The Canvas administrator can manage model deployment permissions at the user profile level as well. For example, if the administrator doesn't want to grant model deployment permissions to all user profiles when setting up a domain, they can grant permissions to specific users after creating the domain.

The following procedure shows how to modify the model deployment permissions for a specific user profile:

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **Domains**.
4. From the list of domains, select the user profile's domain.
5. On the **Domain details** page, select the **User profiles** tab.
6. Choose your **User profile**.
7. On the user profile's page, select the **App Configurations** tab.
8. In the **Canvas** section, choose **Edit**.

9. In the **ML Ops configuration** section, turn on the **Enable direct deployment of Canvas models** toggle to enable deployment permissions.
10. Choose **Submit** to save the changes to your domain settings.

The user profile should now have model deployment permissions.

After granting permissions to the domain or user profile, make sure that the user logs out of their Canvas application and logs back in to apply the permission changes.

## Deploy a model

To get started with deploying your model, you create a new deployment in Canvas and specify the model version that you want to deploy along with the ML infrastructure, such as the type and number of compute instances that you would like to use for hosting the model.

Canvas suggests a default type and number of instances based on your model type, or you can learn more about the various SageMaker AI instance types on the [Amazon SageMaker pricing page](#). You are charged based on the SageMaker AI instance pricing while your endpoint is active.

When deploying JumpStart foundation models, you also have the option to specify the length of the deployment time. You can deploy the model to an endpoint indefinitely (meaning the endpoint is active until you delete the deployment). Or, if you only need the endpoint for a short period of time and would like to reduce costs, you can deploy the model to an endpoint for a specified amount of time, after which SageMaker AI shuts down the endpoint for you.

### Note

If you deploy a model for a specified amount of time, stay logged in to the Canvas application for the duration of the endpoint. If you log out of or delete the application, then Canvas is unable to shut down the endpoint at the specified time.

After your model is deployed to a SageMaker AI Hosting [real-time inference endpoint](#), you can begin making predictions by *invoking* the endpoint.

There are several different ways for you to deploy a model from the Canvas application. You can access the model deployment option through any of the following methods:

- On the **My models** page of the Canvas application, choose the model that you want to deploy. Then, from the model's **Versions** page, choose the **More options** icon

- ( ! ) next to a model version and select **Deploy**.
- When on the details page for a model version, on the **Analyze** tab, choose the **Deploy** option.
  - When on the details page for a model version, on the **Predict** tab, choose the **More options** icon ( ! ) at the top of the page and select **Deploy**.
- On the **ML Ops** page of the Canvas application, choose the **Deployments** tab and then choose **Create deployment**.
  - For JumpStart foundation models and fine-tuned foundation models, go to the **Ready-to-use models** page of the Canvas application. Choose **Generate, extract and summarize content**. Then, find the JumpStart foundation model or fine-tuned foundation model that you want to deploy. Choose the model, and on the model's chat page, choose the **Deploy** button.

All of these methods open the **Deploy model** side panel, where you specify the deployment configuration for your model. To deploy the model from this panel, do the following:

- (Optional) If you're creating a deployment from the **ML Ops** page, you'll have the option to **Select model and version**. Use the dropdown menus to select the model and model version that you want to deploy.
- Enter a name in the **Deployment name** field.
- (For JumpStart foundation models and fine-tuned foundation models only) Choose a **Deployment length**. Select **Indefinite** to leave the endpoint active until you shut it down, or select **Specify length** and then enter the period of time for which you want the endpoint to remain active.
- For **Instance type**, SageMaker AI detects a default instance type and number that is suitable for your model. However, you can change the instance type that you would like to use for hosting your model.

 **Note**

If you run out of the instance quota for the chosen instance type on your AWS account, you can request a quota increase. For more information about the default quotas and how to request an increase, see [Amazon SageMaker AI endpoints and quotas](#) in the [AWS General Reference guide](#).

5. For **Instance count**, you can set the number of active instances that are used for your endpoint. SageMaker AI detects a default number that is suitable for your model, but you can change this number.
6. When you're ready to deploy your model, choose **Deploy**.

Your model should now be deployed to an endpoint.

## View your deployments

You might want to check the status or details of a model deployment in Amazon SageMaker Canvas. For example, if your deployment failed, you might want to check the details to troubleshoot.

You can view your Canvas model deployments from the Canvas application or from the Amazon SageMaker AI console.

To view deployment details from Canvas, choose one of the following procedures:

To view your deployment details from the **ML Ops** page, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation pane, choose **ML Ops**.
3. Choose the **Deployments** tab.
4. Choose your deployment by name from the list.

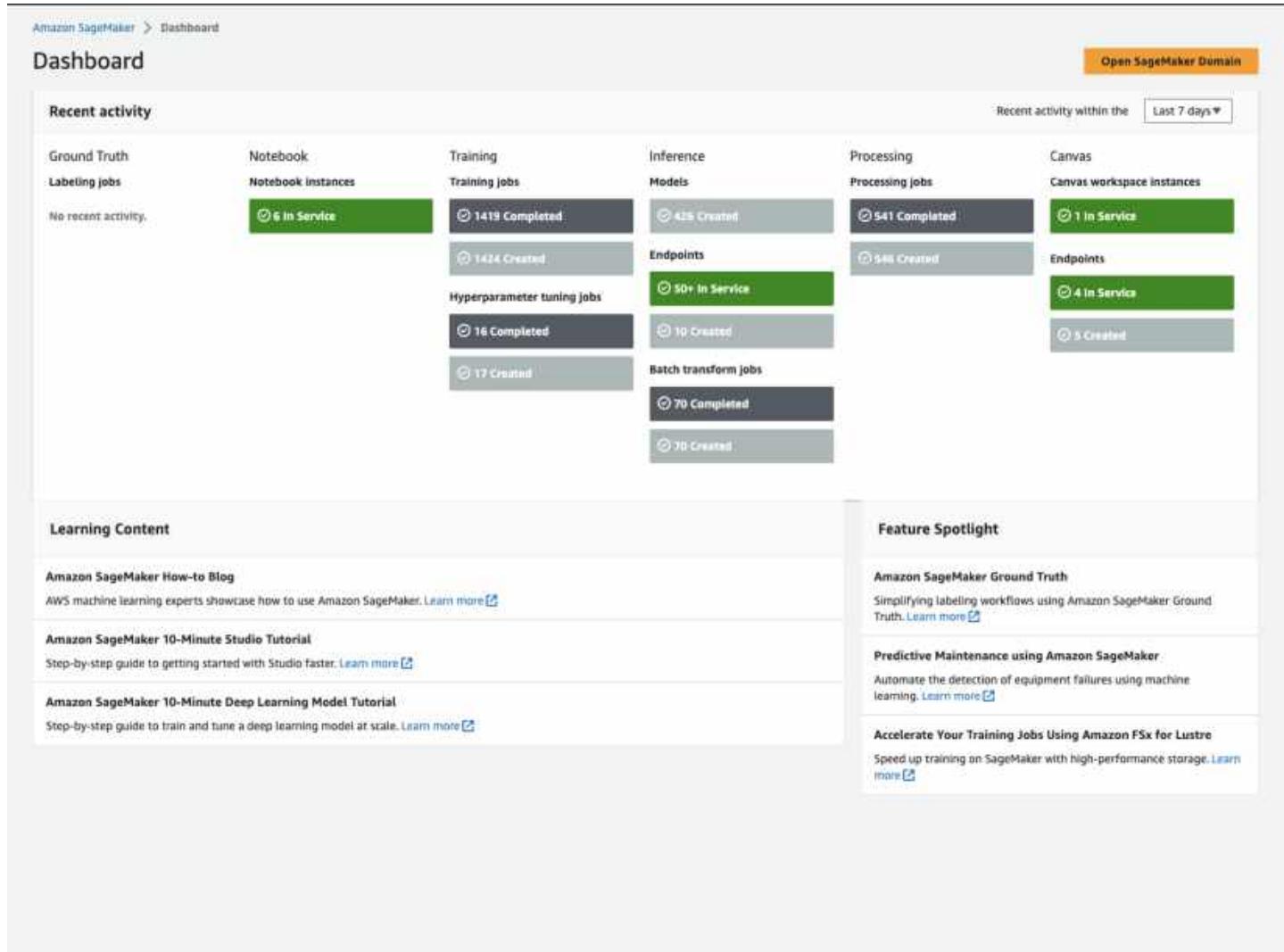
To view your deployment details from a model version's page, do the following:

1. In the SageMaker Canvas application, go to your model version's details page.
2. Choose the **Deploy** tab.
3. On the **Deployments** section that lists all of the deployment configurations associated with that model version, find your deployment.
4. Choose the **More options** icon (⋮), and then select **View details** to open the details page.

The details page for your deployment opens, and you can view information such as the time of the most recent prediction, the endpoint's status and configuration, and the model version that is currently deployed to the endpoint.

You can also view your currently active Canvas workspace instances and active endpoints from the **SageMaker AI dashboard** in the [SageMaker AI console](#). Your Canvas endpoints are listed alongside any other SageMaker AI Hosting endpoints that you've created, and you can filter them by searching for endpoints with the Canvas tag.

The following screenshot shows the SageMaker AI dashboard. In the **Canvas** section, you can see that one workspace instance is in service and four endpoints are active.



## Update a deployment configuration

You can update the deployment configuration for models that you've deployed to endpoints in Amazon SageMaker Canvas. For example, you can deploy an updated model version to the

endpoint, or you can update the instance type or number of instances behind the endpoint based on your capacity needs.

There are several different ways for you to update your deployment from the Canvas application. You can use any of the following methods:

- On the **ML Ops** page of the Canvas application, you can choose the **Deployments** tab and select the deployment that you want to update. Then, choose **Update configuration**.
- When on the details page for a model version, on the **Deploy** tab, you can view the deployments for that version. Next to the deployment, choose the **More options** icon ( ! ) and then choose **Update configuration**.

Both of the preceding methods open the **Update configuration** side panel, where you can make changes to your deployment configuration. To update the configuration, do the following:

1. For the **Select version** dropdown menu, you can select a different model version to deploy to the endpoint.

 **Note**

When updating a deployment configuration, you can only choose a different model version to deploy. To deploy a different model, create a new deployment.

2. For **Instance type**, you can select a different instance type for hosting your model.
3. For **Instance count**, you can change the number of active instances that are used for your endpoint.
4. Choose **Save**.

Your deployment configuration should now be updated.

## Test your deployment

You can test a model deployment by invoking the endpoint, or making single prediction requests, through the Amazon SageMaker Canvas application. You can use this functionality to confirm that your endpoint responds to requests before invoking your endpoint programmatically in a production environment.

## Test a custom model deployment

You can test a custom model deployment by accessing it through the **ML Ops** page and making a single invocation, which returns a prediction along with the probability that the prediction is correct.

### Note

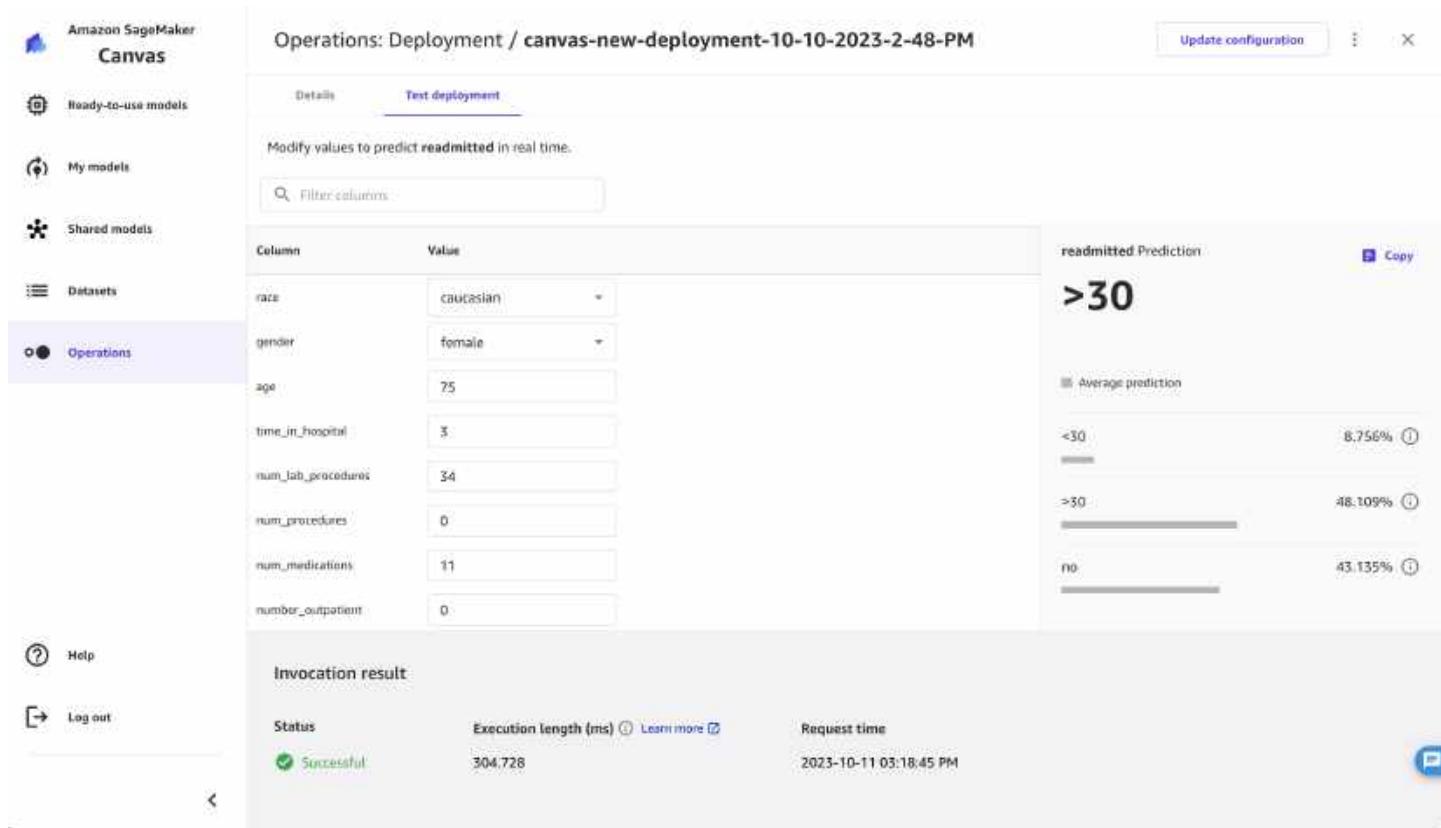
Execution length is an estimate of the time taken to invoke and get a response from the endpoint in Canvas. For detailed latency metrics, see [SageMaker AI Endpoint Invocation Metrics](#).

To test your endpoint through the Canvas application, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation panel, choose **ML Ops**.
3. Choose the **Deployments** tab.
4. From the list of deployments, choose the one with the endpoint that you want to invoke.
5. On the deployment's details page, choose the **Test deployment** tab.
6. On the deployment testing page, you can modify the **Value** fields to specify a new data point. For time series forecasting models, you specify the **Item ID** for which you want to make a forecast.
7. After modifying the values, choose **Update** to get the prediction result.

The prediction loads, along with the **Invocation result** fields which indicate whether or not the invocation was successful and how long the request took to process.

The following screenshot shows a prediction performed in the Canvas application on the **Test deployment** tab.



For all model types except numeric prediction and time series forecasting, the prediction returns the following fields:

- **predicted\_label** – the predicted output
- **probability** – the probability that the predicted label is correct
- **labels** – the list of all the possible labels
- **probabilities** – the probabilities corresponding to each label (the order of this list matches the order of the labels)

For numeric prediction models, the prediction only contains the **score** field, which is the predicted output of the model, such as the predicted price of a house.

For time series forecasting models, the prediction is a graph showing the forecasts by quantile. You can choose **Schema view** to see the forecasted numeric values for each quantile.

You can continue making single predictions through the deployment testing page, or you can see the following section [Invoke your endpoint](#) to learn how to invoke your endpoint programmatically from applications.

## Test a JumpStart foundation model deployment

You can chat with a deployed JumpStart foundation model through the Canvas application to test its functionality before invoking it through code.

To chat with a deployed JumpStart foundation model, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation panel, choose **ML Ops**.
3. Choose the **Deployments** tab.
4. From the list of deployments, find the one that you want to invoke and choose its **More options** icon ().
5. From the context menu, choose **Test deployment**.
6. A new **Generate, extract and summarize content** chat opens with the JumpStart foundation model, and you can begin typing prompts. Note that prompts from this chat are sent as requests to your SageMaker AI Hosting endpoint.

## Invoke your endpoint

### Note

We recommend that you [test your model deployment in Amazon SageMaker Canvas](#) before invoking a SageMaker AI endpoint programmatically.

You can use your Amazon SageMaker Canvas models that you've deployed to a SageMaker AI endpoint in production with your applications. Invoke the endpoint programmatically the same way that you invoke any other [SageMaker AI real-time endpoint](#). Invoking an endpoint programmatically returns a response object which contains the same fields described in [Test your deployment](#).

For more detailed information about how to programmatically invoke endpoints, see [Invoke models for real-time inference](#).

The following Python examples show you how to invoke your endpoint based on the model type.

## JumpStart foundation models

The following example shows you how to invoke a JumpStart foundation model that you've deployed to an endpoint.

```
import boto3
import pandas as pd

client = boto3.client("runtime.sagemaker")
body = pd.DataFrame(
    [['feature_column1', 'feature_column2'],
     ['feature_column1', 'feature_column2']])
.to_csv(header=False, index=False).encode("utf-8")

response = client.invoke_endpoint(
    EndpointName="endpoint_name",
    ContentType="text/csv",
    Body=body,
    Accept="application/json"
)
```

## Numeric and categorical prediction models

The following example shows you how to invoke numeric or categorical prediction models.

```
import boto3
import pandas as pd

client = boto3.client("runtime.sagemaker")
body = pd.DataFrame(['feature_column1', 'feature_column2'], ['feature_column1',
    'feature_column2']).to_csv(header=False, index=False).encode("utf-8")

response = client.invoke_endpoint(
    EndpointName="endpoint_name",
    ContentType="text/csv",
    Body=body,
    Accept="application/json"
)
```

## Time series forecasting models

The following example shows you how to invoke time series forecasting models. For a complete example of how to test invoke a time series forecasting model, see [Time-Series Forecasting with Amazon SageMaker Autopilot](#).

```
import boto3
import pandas as pd

csv_path = './real-time-payload.csv'
data = pd.read_csv(csv_path)

client = boto3.client("runtime.sagemaker")

body = data.to_csv(index=False).encode("utf-8")

response = client.invoke_endpoint(
    EndpointName="endpoint_name",
    ContentType="text/csv",
    Body=body,
    Accept="application/json"
)
```

## Image prediction models

The following example shows you how to invoke image prediction models.

```
import boto3
client = boto3.client("runtime.sagemaker")
with open("example_image.jpg", "rb") as file:
    body = file.read()
    response = client.invoke_endpoint(
        EndpointName="endpoint_name",
        ContentType="application/x-image",
        Body=body,
        Accept="application/json"
)
```

## Text prediction models

The following example shows you how to invoke text prediction models.

```
import boto3
```

```
import pandas as pd

client = boto3.client("runtime.sagemaker")
body = pd.DataFrame([["Example text 1"], ["Example text 2"]]).to_csv(header=False,
index=False).encode("utf-8")

response = client.invoke_endpoint(
    EndpointName="endpoint_name",
    ContentType="text/csv",
    Body=body,
    Accept="application/json"
)
```

## Delete a model deployment

You can delete your model deployments from the Amazon SageMaker Canvas application. This action also deletes the endpoint from the SageMaker AI console and shuts down any endpoint-related resources.

### Note

Optionally, you can delete your endpoint through the [SageMaker AI console](#) or using the SageMaker AI DeleteEndpoint API. For more information, see [Delete Endpoints and Resources](#). However, when you delete the endpoint through the SageMaker AI console or APIs instead of the Canvas application, the list of deployments in Canvas isn't automatically updated. You must also delete the deployment from the Canvas application to remove it from the list.

To delete a deployment in Canvas, do the following:

1. Open the SageMaker Canvas application.
2. In the left navigation panel, choose **ML Ops**.
3. Choose the **Deployments** tab.
4. From the list of deployments, choose the one that you want to delete.
5. At the top of the deployment details page, choose the **More options** icon (.
6. Choose **Delete deployment**.
7. In the **Delete deployment** dialog box, choose **Delete**.

Your deployment and SageMaker AI Hosting endpoint should now be deleted from both Canvas and the SageMaker AI console.

## How to manage automations

In SageMaker Canvas, you can create automations that update your dataset or generate predictions from your model on a schedule. For example, you might receive new shipping data on a daily basis. You can set up an automatic update for your dataset and automatic batch predictions that run whenever the dataset is updated. Using these features, you can set up an automated workflow and reduce the amount of time you spend manually updating datasets and making predictions.

### Note

You can only set up a maximum of 20 automatic configurations in your Canvas application. Automations are only active while you're logged in to the Canvas application. If you log out of Canvas, your automatic jobs pause until you log back in.

The following sections describe how to view, edit, and delete configurations for existing automations. To learn how to set up automations, see the following topics:

- To set up automatic dataset updates, see [Update a dataset](#).
- To set up automatic batch predictions, see [Batch predictions in SageMaker Canvas](#).

## Topics

- [View your automations](#)
- [Edit your automatic configurations](#)
- [Delete an automatic configuration](#)

## View your automations

You can also view all of your auto update jobs by going to the left navigation pane of Canvas and choosing **ML Ops**. The **ML Operations** page combines automations for both automatic dataset updates and automatic batch predictions. On the **Automations** tab, you can see the following sub-tabs:

- **All jobs** – You can see every instance of a **Dataset update** or **Batch prediction** job that Canvas has done. For each job, you can see fields such as the associated **Input dataset**, the

**Configuration name** of the associated auto update configuration, and the **Status** showing whether the job was successful or not. You can filter the jobs by configuration name:

- For dataset update jobs, you can choose the latest version of the dataset, or the most recent job, to preview the dataset.

- For batch prediction jobs, you can choose the **More options** icon

(

)

to preview or download the predictions for that job. You can also choose **View details** to see more details about your prediction job. For more information about batch prediction job details, see [View your batch prediction jobs](#).

- **Configuration** – You can see all of the **Dataset update** and **Batch prediction** configurations you've created. For each configuration, you can see fields such as the associated **Input dataset** and the **Frequency** of the jobs. You can also turn off or turn on the **Auto update** toggle to pause or resume automatic updates. If you choose the **More options** icon

(

for a specific configuration, you can choose to **View all jobs** for the configuration, **Update configuration**, or **Delete configuration**.

## Edit your automatic configurations

After setting up a configuration, you might want to make changes to it. For automatic dataset updates, you can update the Amazon S3 location for Canvas to import data, the frequency of the updates, and the starting time. For automatic batch predictions, you can change the dataset that the configuration tracks for updates. You can also turn off the automation to temporarily pause updates until you choose to resume them.

The following sections show you how to update each type of configuration.

### Note

You can't change the frequency for automatic batch predictions because automatic batch predictions run every time the target dataset is updated.

## Topics

- [Edit your automatic dataset update configuration](#)
- [Edit your automatic batch prediction configuration](#)

## Edit your automatic dataset update configuration

You might want to make changes to your auto update configuration for a dataset, such as changing the frequency of the updates. You might also want to turn off your automatic update configuration to pause the updates to your dataset.

To make changes to your auto update configuration for a dataset, do the following:

1. In the left navigation pane of Canvas, choose **ML Ops**.
2. Choose the **Automations** tab.
3. Choose the **Configuration** tab.
4. For your auto update configuration, choose the **More options** icon ().
5. In the dropdown menu, choose **Update configuration**. You are taken to the **Auto updates** tab of the dataset.
6. Make your changes to the configuration. When you're done making changes, choose **Save**.

To pause your dataset updates, turn off your automatic configuration. One way to turn off auto updates is by doing the following:

1. In the left navigation pane of Canvas, choose **ML Ops**.
2. Choose the **Automations** tab.
3. Choose the **Configuration** tab.
4. Find your configuration from the list and turn off the **Auto update** toggle.

Automatic updates for your dataset are now paused. You can turn this toggle back on at any time to resume the update schedule.

## Edit your automatic batch prediction configuration

When you edit a batch prediction configuration, you can change the target dataset but not the frequency (since automatic batch predictions occur whenever the dataset is updated).

To make changes to your automatic batch predictions configuration, do the following:

1. In the left navigation pane of Canvas, choose **ML Ops**.
2. Choose the **Automations** tab.

3. Choose the **Configuration** tab.
4. For your auto update configuration, choose the **More options** icon ().
5. In the dropdown menu, choose **Update configuration**. You are taken to the **Auto updates** tab of the dataset.
6. The **Automate batch prediction** dialog box opens. You can select another dataset and choose **Set up** to save your changes.

Your automatic batch predictions configuration is now updated.

To pause your automatic batch predictions, turn off your automatic configuration. Use the following procedure to turn off your configuration:

1. In the left navigation pane of Canvas, choose **ML Ops**.
2. Choose the **Automations** tab.
3. Choose the **Configuration** tab.
4. Find your configuration from the list and turn off the **Auto update** toggle.

Automatic batch predictions for your dataset are now paused. You can turn this toggle back on at any time to resume the update schedule.

### Delete an automatic configuration

You might want to delete a configuration to stop your automated workflow in SageMaker Canvas.

To delete a configuration for automatic dataset updates or automatic batch predictions, do the following:

1. In the left navigation pane of Canvas, choose **ML Ops**.
2. Choose the **Automations** tab.
3. Choose the **Configuration** tab.
4. Find your auto update configuration, and choose the **More options** icon ().
5. Choose **Delete configuration**.
6. In the dialog box that pops up, choose **Delete**.

Your auto update configuration is now deleted.

## Logging out of Amazon SageMaker Canvas

After completing your work in Amazon SageMaker Canvas, you can log out or configure your application to automatically terminate the *workspace instance*. A workspace instance is dedicated for your use every time you launch a Canvas application, and you are billed for as long as the instance runs. Logging out or terminating the workspace instance stops the workspace instance billing. For more information, see [SageMaker Pricing](#).

The following sections describe how to log out of your Canvas application and how to configure your application to automatically shut down on a schedule.

### Log out of Canvas

When you log out of Canvas, your models and datasets aren't affected. Any quick or standard model builds or [large data processing jobs](#) continue running even if you log out.

To log out, choose the **Log out** button



)

on the left panel of the SageMaker Canvas application.

You can also log out from the SageMaker Canvas application by closing your browser tab and then [deleting the application](#) in the console.

After you log out, SageMaker Canvas tells you to relaunch in a different tab. Logging in takes around 1 minute. If you have an administrator who set up SageMaker Canvas for you, use the instructions they gave you to log back in. If don't have an administrator, see the procedure for accessing SageMaker Canvas in [Prerequisites for setting up Amazon SageMaker Canvas](#).

### Automatically shut down Canvas

If you're a Canvas administrator, you might want to regularly shut down applications to reduce costs. You can either create a schedule to shut down active Canvas applications, or you can create an automation to shut down Canvas applications as soon as they're *idle* (meaning the user hasn't been active for 2 hours).

You can create these solutions using AWS Lambda functions that call the DeleteApp API and delete Canvas applications given certain conditions. For more information about these solutions and access to AWS CloudFormation templates that you can use, see the blog [Optimizing costs for Amazon SageMaker Canvas with automatic shutdown of idle apps](#).

**Note**

You might experience missing [Amazon CloudWatch](#) metrics if there was an error when setting up your idle shut down schedule or a CloudWatch error. We recommend that you add a CloudWatch alarm that monitors for missing metrics. If you encounter this issue, reach out to Support for help.

## Limitations and troubleshooting

The following section outlines troubleshooting help and limitations that apply when using Amazon SageMaker Canvas. You can use these this topic to help troubleshoot any issues you encounter.

### Troubleshooting issues with granting permissions through the SageMaker AI console

If you're having trouble granting Canvas base permissions or Ready-to-use models permissions to your user, your user might have an AWS IAM execution role with more than one trust relationship to other AWS services. A trust relationship is a policy attached to your role that defines which principals (users, roles, accounts, or services) can assume the role. For example, you might encounter an issue granting additional Canvas permissions to your user if their execution role has a trust relationship to both Amazon SageMaker AI and Amazon Forecast.

You can fix this problem by choosing one of the following options.

#### 1. Remove all but one trusted service from the role.

This solution requires you to edit the trust relationship for your user profile's IAM role and remove all AWS services except SageMaker AI.

To edit the trust relationship for your IAM execution role, do the following:

1. Go to the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, choose **Roles**. The console displays the roles for your account.
3. Choose the name of the role that you want to modify, and select the **Trust relationships** tab on the details page.
4. Choose **Edit trust policy**.

5. In the **Edit trust policy editor**, paste the following, and then choose **Update Policy**.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "Service": [  
                    "sagemaker.amazonaws.com"  
                ]  
            },  
            "Action": "sts:AssumeRole"  
        }  
    ]  
}
```

You can also update this policy document using the IAM CLI. For more information, see [update-trust](#) in the *IAM Command Line Reference*.

You can now retry granting the Canvas base permissions or the Ready-to-use models permissions to your user.

## 2. Use a different role with one or fewer trusted services.

This solution requires you to specify a different IAM role for your user profile. Use this option if you already have an IAM role that you can substitute.

To specify a different execution role for your user, do the following:

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. From the list of domains, select the domain that you want to view a list of user profiles for.
5. On the **domain details** page, choose the **User profiles** tab.
6. Choose the user whose permissions you want to edit. On the **User details** page, choose **Edit**.
7. On the **General settings** page, choose the **Execution role** dropdown list and select the role that you want to use.
8. Choose **Submit** to save your changes to the user profile.

Your user should now be using an execution role with only one trusted service (SageMaker AI).

You can retry granting the Canvas base permissions or the Ready-to-use models permissions to your user.

### 3. Manually attach the AWS managed policy to the execution role instead of using the toggle in the SageMaker AI domain settings.

Instead of using the toggle in the domain or user profile settings, you can manually attach the AWS managed policies that grant a user the correct permissions.

To grant a user Canvas base permissions, attach the [AmazonSageMakerCanvasFullAccess](#) policy. To grant a user Ready-to-use models permissions, attach the [AmazonSageMakerCanvasAI ServicesAccess](#) policy.

Use the following procedure to attach an AWS managed policy to your role:

1. Go to the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Roles**.
3. In the search box, search for the user's IAM role by name and select it.
4. On the page for the user's role, under **Permissions**, choose **Add permissions**.
5. From the dropdown menu, choose **Attach policies**.
6. Search for and select the policy or policies that you want to attach to the user's execution role:
  - a. To grant the Canvas base permissions, search for and select the [AmazonSageMakerCanvasFullAccess](#) policy.
  - b. To grant the Ready-to-use models permissions, search for and select the [AmazonSageMakerCanvasAI ServicesAccess](#) policy.
7. Choose **Add permissions** to attach the policy to the role.

After attaching an AWS managed policy to the user's role through the IAM console, your user should now have the Canvas base permissions or Ready-to-use models permissions.

## Troubleshooting issues with creating a Canvas application due to space failure

When creating a new Canvas application, if you encounter an error stating `Unable to create app <app-arn> because space <space-arn> is not in InService state`, this indicates that the underlying Amazon SageMaker Studio space creation has failed. A Studio

*space* is the underlying storage that hosts your Canvas application data. For more general information about Studio spaces, see [Amazon SageMaker Studio spaces](#). For more information about configuring spaces in Canvas, see [Store SageMaker Canvas application data in your own SageMaker AI space](#).

To determine the root cause of your why space creation failed, you can use the [DescribeSpace](#) API to check the FailureReason field. For more information about the possible statuses of spaces and what they mean, see [Amazon SageMaker AI domain entities and statuses](#).

To resolve this issue, find your domain in the SageMaker AI console and delete the failed space listed in the error message you received. For detailed steps on how to find and delete a space, see the page [Stop and delete your Studio running applications and spaces](#) and follow the instructions to **Delete a Studio space**. Deleting the space also deletes any applications associated with the space. After deleting the space, you can try to create your Canvas application again. The space should now provision successfully, allowing Canvas to launch.

## Billing and cost in SageMaker Canvas

To track the costs associated with your SageMaker Canvas application, you can use the AWS Billing and Cost Management service. Billing and Cost Management provides tools to help you gather information related to your cost and usage, analyze your cost drivers and usage trends, and take action to budget your spending. For more information, see [What is AWS Billing and Cost Management?](#)

Billing in SageMaker Canvas consists of the following components:

- Workspace instance charges – You are charged for the number of hours that you are logged in to or using SageMaker Canvas. We recommend that you log out or create a schedule to shut down any Canvas applications that you’re not actively using to reduce costs. For more information, see [Logging out of Amazon SageMaker Canvas](#).
- AWS service charges – You are charged for building and making predictions with custom models, or for making predictions with Ready-to-use models:
  - Training charges – For all model types, you are charged based on your resource usage while the model builds. These resources include any compute instances that Canvas spins up. You may see these charges on your account as Hosting, Training, Processing, or Batch Transform jobs.
  - Prediction charges – You are charged for the resources used to generate predictions, depending on the type of custom model that you built or the type of Ready-to-use model you used.

The [Ready-to-use models](#) in Canvas leverage other AWS services to generate predictions. When you use a Ready-to-use model, you are charged by the respective service, and their pricing conditions apply:

- For sentiment analysis, entities extraction, language detection, and personal information detection, you're charged with [Amazon Comprehend pricing](#).
- For object detection in images and text detection in images, you're charged with [Amazon Rekognition pricing](#).
- For expense analysis, identity document analysis, and document analysis, you're charged with [Amazon Textract pricing](#).

For more information, see [SageMaker Canvas pricing](#).

To help you track your costs in Billing and Cost Management, you can assign custom tags to your SageMaker Canvas app and users. You can track the costs your apps incur, and by tagging individual user profiles, you can track costs based on the user profile. For more information about tags, see [Using Cost Allocation Tags](#).

You can add tags to your SageMaker Canvas app and users by doing the following:

- If you are setting up your Amazon SageMaker AI domain and SageMaker Canvas for the first time, follow the [Getting Started](#) instructions and add tags when creating your domain or users. You can add tags either through the **General settings** in the domain console setup, or through the APIs ([CreateDomain](#) or [CreateUserProfile](#)). SageMaker AI adds the tags specified in your domain or UserProfile to any SageMaker Canvas apps or users you create after you create the domain.
- If you want to add tags to apps in an existing domain, you must add tags to either the domain or the UserProfile. You can add tags through either the console or the [AddTags](#) API. If you add tags through the console, then you must delete and relaunch your SageMaker Canvas app in order for the tags to propagate to the app. If you use the API, the tags are added directly to the app. For more information about deleting and relaunching a SageMaker Canvas app, see [Manage apps](#).

After you add tags to your domain, it might take up to 24 hours for the tags to appear in the AWS Billing and Cost Management console for activation. After they appear in the console, it takes another 24 hours for the tags to activate.

On the **Cost explorer** page, you can group and filter your costs by tags and usage types to separate your Workspace instance charges from your Training charges. The charges for each are listed as the following:

- Workspace instance charges: Charges show up under the usage type REGION-Canvas:Session-Hrs (Hrs).
- Training charges: Charges show up under the usage types for SageMaker AI Hosting, Training, Processing, or Batch Transform jobs.

## Amazon SageMaker geospatial capabilities

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. If prior to November 30, 2023 you created a Amazon SageMaker AI domain, Studio Classic remains the default experience. domains created after November 30, 2023 default to the new Studio experience.

Amazon SageMaker geospatial features and resources are *only* available in Studio Classic. To learn more about setting up a domain and getting started with Studio, see [Getting started with Amazon SageMaker geospatial](#).

Amazon SageMaker geospatial capabilities makes it easier for data scientists and machine learning (ML) engineers to build, train, and deploy ML models faster using geospatial data. You have access to open-source and third-party data, processing, and visualization tools to make it more efficient to prepare geospatial data for ML. You can increase your productivity by using purpose-built algorithms and pre-trained ML models to speed up model building and training, and use built-in visualization tools to explore prediction outputs on an interactive map and then collaborate across teams on insights and results.

### Note

Currently, SageMaker geospatial capabilities are only supported in the US West (Oregon) Region.

If you don't see the SageMaker geospatial UI available in your current Studio Classic instance check to make sure you are currently in the US West (Oregon) Region.

## Why use SageMaker geospatial capabilities?

You can use SageMaker geospatial capabilities to make predictions on geospatial data faster than do-it-yourself solutions. SageMaker geospatial capabilities make it easier to access geospatial data from your existing customer data lakes, open-source datasets, and other SageMaker geospatial data providers. SageMaker geospatial capabilities minimize the need for building custom infrastructure and data preprocessing functions by offering purpose-built algorithms for efficient data preparation, model training, and inference. You can also create and share custom visualizations and data with your company from Amazon SageMaker Studio Classic. SageMaker geospatial capabilities offer pre-trained models for common uses in agriculture, real estate, insurance, and financial services.

## How can I use SageMaker geospatial capabilities?

You can use SageMaker geospatial capabilities in two ways.

- Through the SageMaker geospatial UI, as a part of Amazon SageMaker Studio Classic UI.
- Through a Studio Classic notebook instance that uses the **Geospatial 1.0** image.

### SageMaker AI has the following geospatial capabilities

- Use a purpose built SageMaker geospatial image that supports both CPU and GPU-based notebook instances, and also includes commonly used open-source libraries found in geospatial machine learning workflows.
- Use the Amazon SageMaker Processing and the SageMaker geospatial container to run large-scale workloads with your own datasets, including soil, weather, climate, LiDAR, and commercial aerial and satellite imagery.
- Run an [Earth Observation job](#) for raster data processing.
- Run a [Vector Enrichment job](#) to convert latitude and longitude into human readable addresses, and match noisy GPS traces to specific roads.
- Use built-in [visualization tools right in Studio Classic to interactively view geospatial data or model predictions on a map.](#)

You can also use data from a collection of geospatial data providers. Currently, the data collections available include:

- [USGS Landsat](#)

- [Sentinel-1](#)
- [Sentinel-2](#)
- [Copernicus DEM](#)
- [National Agriculture Imagery Program](#)

## Are you a first-time user of SageMaker geospatial?

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. New domains created after November 30, 2023 default to the Studio experience. Access to SageMaker geospatial is limited to Studio Classic, to learn more see [Accessing SageMaker geospatial](#).

If you're a first-time user of AWS or Amazon SageMaker AI, we recommend that you do the following:

- 1. Create an AWS account.**

To learn about setting up an AWS account and getting started with SageMaker AI, see [Complete Amazon SageMaker AI prerequisites](#).

- 2. Create a user role and execution role that work with SageMaker geospatial.**

As a managed service, Amazon SageMaker geospatial capabilities performs operations on your behalf on the AWS hardware that SageMaker AI manages. A SageMaker AI execution role can perform only the operations that users grant. To work with SageMaker geospatial capabilities, you must set up a user role and an execution role. For more information, see [SageMaker geospatial capabilities roles](#).

- 3. Update your trust policy to include SageMaker geospatial.**

SageMaker geospatial defines an additional service principal. To learn how to create or update your SageMaker AI execution role's trust policy, see [Adding the SageMaker geospatial service principal to an existing SageMaker AI execution role](#).

- 4. Set up an Amazon SageMaker AI domain to access Amazon SageMaker Studio Classic.**

To use SageMaker geospatial, a domain is required. For domains created before November 30, 2023 the default experience is Studio Classic. Domains created after November 30, 2023 default to the Studio experience. To learn more about accessing Studio Classic from Studio, see [Accessing SageMaker geospatial](#).

## 5. Remember, shut down resources.

When you have finished using SageMaker geospatial capabilities, shut down the instance it runs on to avoid incurring additional charges. For more information, see [Shut down resources from Amazon SageMaker Studio Classic](#).

### Topics

- [Getting started with Amazon SageMaker geospatial](#)
- [Using a processing jobs for custom geospatial workloads](#)
- [Earth Observation Jobs](#)
- [Vector Enrichment Jobs](#)
- [Visualization Using SageMaker geospatial capabilities](#)
- [Amazon SageMaker geospatial Map SDK](#)
- [SageMaker geospatial capabilities FAQ](#)
- [SageMaker geospatial Security and Permissions](#)
- [Types of compute instances](#)
- [Data collections](#)

## Getting started with Amazon SageMaker geospatial

SageMaker geospatial provides a purpose built **Image** and **Instance type** for Amazon SageMaker Studio Classic notebooks. You can use either CPU or GPU enabled notebooks with the SageMaker geospatial **Image**. You can also visualize your geospatial data using a purpose built visualizer. Furthermore, SageMaker geospatial also provides APIs that allow you to query raster data collections. You can also use pre-trained models to analyze geospatial data, reverse geocoding, and map matching.

### Important

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. If prior to November 30, 2023 you created a Amazon SageMaker AI domain, Studio Classic remains the default experience. domains created after November 30, 2023 default to the new Studio experience.

To access and get started using Amazon SageMaker geospatial, do the following:

## Topics

- [Accessing SageMaker geospatial](#)
- [Create an Amazon SageMaker Studio Classic notebook using the geospatial image](#)
- [Access the Sentinel-2 raster data collection and create an earth observation job to perform land segmentation](#)

## Accessing SageMaker geospatial

### Note

Currently, SageMaker geospatial capabilities are only supported in the US West (Oregon) Region and in Studio Classic.

If you don't see the SageMaker geospatial UI available in your current Studio Classic instance check to make sure you are currently in the US West (Oregon) Region.

A domain is required to access SageMaker geospatial. If you created a domain prior to November 30, 2023 the default experience is Studio Classic.

If you created a domain after November 30, 2023 or if you have migrated to Studio, then you can use the following procedure to activate Studio Classic from within Studio to use SageMaker geospatial features.

To learn more about creating a domain, see [Onboard to Amazon SageMaker AI domain](#).

## To access Studio Classic from Studio

1. Launch Amazon SageMaker Studio.
2. Under **Applications**, choose **Studio Classic**.
3. Then, choose **Create Studio Classic space**.
4. On the **Create Studio Classic space** page, enter a **Name**.
5. Disable the **Share with my domain** option. SageMaker geospatial is not available in shared domains.
6. Then choose **Create space**.

When successful the **Status** changes to **Updating**. When your Studio Classic application is ready to be used the status changes to **Stopped**.

To start your Studio Classic application, choose **Run**.

## Create an Amazon SageMaker Studio Classic notebook using the geospatial image

### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the Studio Classic application. For information about using the updated Studio experience, see [Amazon SageMaker Studio](#).

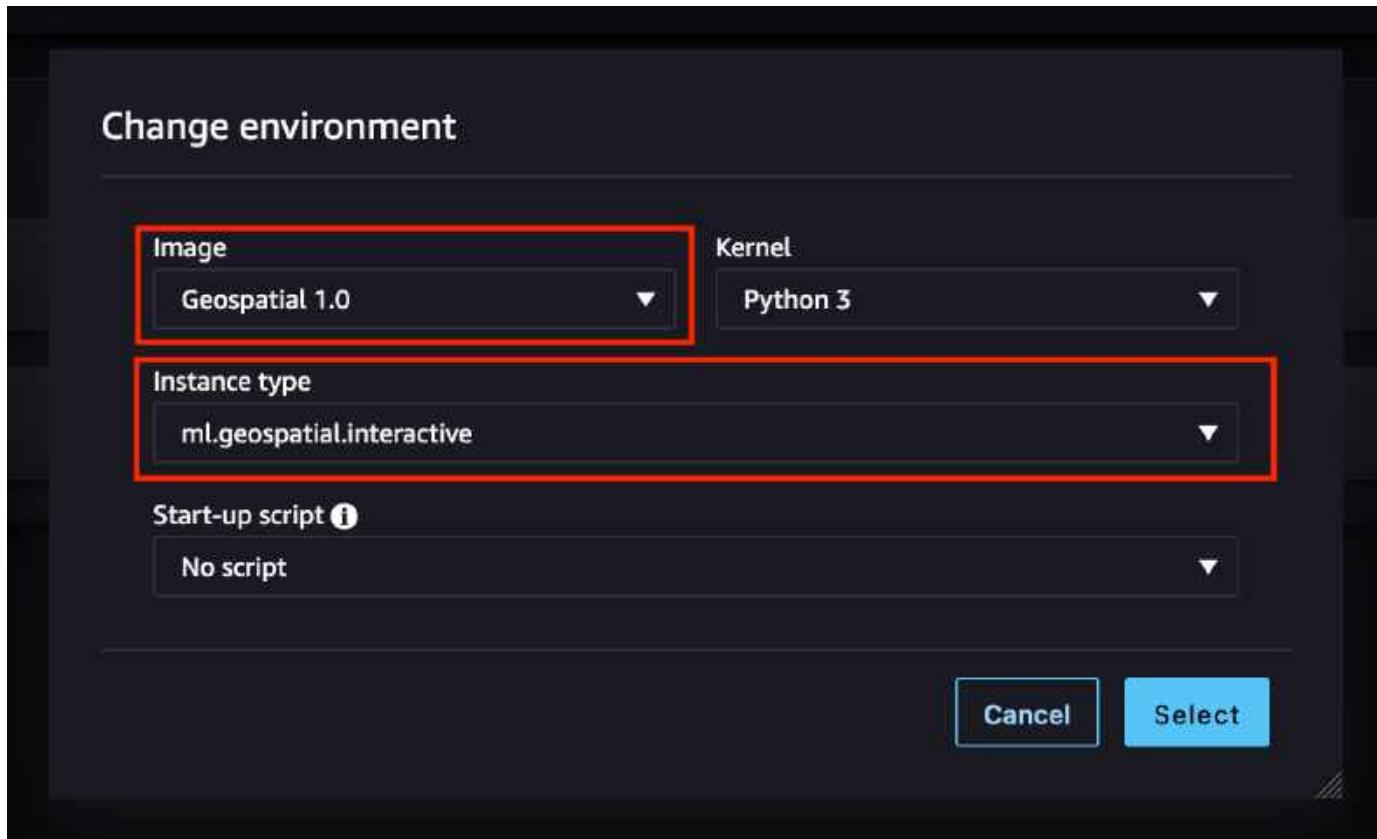
### **Note**

Currently, SageMaker geospatial is only supported in the US West (Oregon) Region. If you don't see SageMaker geospatial available in your current domain or notebook instance, make sure that you're currently in the US West (Oregon) Region.

Use the following procedure to create Studio Classic notebook with the SageMaker geospatial image. If your default studio experience is Studio, see [Accessing SageMaker geospatial](#) to learn about starting a Studio Classic application.

### To create a Studio Classic notebook with the SageMaker geospatial image

1. Launch Studio Classic
2. Choose **Home** in the menu bar.
3. Under **Quick actions**, choose **Open Launcher**.
4. When the **Launcher** dialog box opens. Choose **Change environment** under **Notebooks and compute resources**.
5. When, the **Change environment** dialog box opens. Choose the **Image** dropdown and choose or type **Geospatial 1.0**.



6. Next, choose an **Instance type** from the dropdown.

SageMaker geospatial supports two types of notebook instances: CPU and GPU. The supported CPU instance is called **ml.geospatial.interactive**. Any of the G5-family of GPU instances can be used with the Geospatial 1.0 image.

**Note**

If you receive a `ResourceLimitExceeded` error when attempting to start a GPU based instance, you need to request a quota increase. To get started on a Service Quotas quota increase request, see [Requesting a quota increase](#) in the *Service Quotas User Guide*

7. Choose **Select**.
8. Choose **Create notebook**.

After creating a notebook, to learn more about SageMaker geospatial, try the [SageMaker geospatial tutorial](#). It shows you how to process Sentinel-2 image data and perform land segmentation on it using the earth observation jobs API.

## Access the Sentinel-2 raster data collection and create an earth observation job to perform land segmentation

This Python-based tutorial uses the SDK for Python (Boto3) and an Amazon SageMaker Studio Classic notebook. To complete this demo successfully, make sure that you have the required AWS Identity and Access Management (IAM) permissions to use SageMaker geospatial and Studio Classic. SageMaker geospatial requires that you have a user, group, or role which can access Studio Classic. You must also have a SageMaker AI execution role that specifies the SageMaker geospatial service principal, `sagemaker-geospatial.amazonaws.com` in its trust policy.

To learn more about these requirements, see [SageMaker geospatial IAM roles](#).

This tutorial shows you how to use SageMaker geospatial API to complete the following tasks:

- Find the available raster data collections with `list_raster_data_collections`.
- Search a specified raster data collection by using `search_raster_data_collection`.
- Create an earth observation job (EOJ) by using `start_earth_observation_job`.

### Using `list_raster_data_collections` to find available data collections

SageMaker geospatial supports multiple raster data collections. To learn more about the available data collections, see [Data collections](#).

This demo uses satellite data that's collected from [Sentinel-2 Cloud-Optimized GeoTIFF](#) satellites. These satellites provide global coverage of Earth's land surface every five days. In addition to collecting surface images of Earth, the Sentinel-2 satellites also collect data across a variety of spectralbands.

To search an area of interest (AOI), you need the ARN that's associated with the Sentinel-2 satellite data. To find the available data collections and their associated ARNs in your AWS Region, use the `list_raster_data_collections` API operation.

Because the response can be paginated, you must use the `get_paginator` operation to return all of the relevant data:

```
import boto3
import sagemaker
import sagemaker_geospatial_map
import json
```

```
## SageMaker Geospatial is currently only available in US-WEST-2
session = boto3.Session(region_name='us-west-2')
execution_role = sagemaker.get_execution_role()

## Creates a SageMaker Geospatial client instance
geospatial_client = session.client(service_name="sagemaker-geospatial")

# Creates a reusable Paginator for the list_raster_data_collections API operation
paginator = geospatial_client.getPaginator("list_raster_data_collections")

# Create a PageIterator from the paginator class
page_iterator = paginator.paginate()

# Use the iterator to iterate through the results of list_raster_data_collections
results = []
for page in page_iterator:
    results.append(page['RasterDataCollectionSummaries'])

print(results)
```

This is a sample JSON response from the `list_raster_data_collections` API operation. It's truncated to include only the data collection (Sentinel-2) that's used in this code example. For more details about a specific raster data collection, use `get_raster_data_collection`:

```
{
    "Arn": "arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-collection/public/nmqj48dcu3g7ayw8",
    "Description": "Sentinel-2a and Sentinel-2b imagery, processed to Level 2A (Surface Reflectance) and converted to Cloud-Optimized GeoTIFFs",
    "DescriptionPageUrl": "https://registry.opendata.aws/sentinel-2-l2a-cogs",
    "Name": "Sentinel 2 L2A COGs",
    "SupportedFilters": [
        {
            "Maximum": 100,
            "Minimum": 0,
            "Name": "EoCloudCover",
            "Type": "number"
        },
        {
            "Maximum": 90,
            "Minimum": 0,
            "Name": "ViewOffNadir",
            "Type": "number"
        }
    ]
}
```

```
        },
        {
            "Name": "Platform",
            "Type": "string"
        }
    ],
    "Tags": {},
    "Type": "PUBLIC"
}
```

After running the previous code sample, you get the ARN of the Sentinel-2 raster data collection, `arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-collection/public/nmqj48dcu3g7ayw8`. In the [next section](#), you can query the Sentinel-2 data collection using the `search_raster_data_collection` API.

## Searching the Sentinel-2 raster data collection using `search_raster_data_collection`

In the preceding section, you used `list_raster_data_collections` to get the ARN for the Sentinel-2 data collection. Now you can use that ARN to search the data collection over a given area of interest (AOI), time range, properties, and the available UV bands.

To call the `search_raster_data_collection` API you must pass in a Python dictionary to the `RasterDataCollectionQuery` parameter. This example uses `AreaOfInterest`, `TimeRangeFilter`, `PropertyFilters`, and `BandFilter`. For ease, you can specify the Python dictionary using the variable `search_rdc_query` to hold the search query parameters:

```
search_rdc_query = {
    "AreaOfInterest": {
        "AreaOfInterestGeometry": {
            "PolygonGeometry": {
                "Coordinates": [
                    [
                        # coordinates are input as longitude followed by latitude
                        [-114.529, 36.142],
                        [-114.373, 36.142],
                        [-114.373, 36.411],
                        [-114.529, 36.411],
                        [-114.529, 36.142],
                    ]
                ]
            }
        }
    }
}
```

```
},
  "TimeRangeFilter": {
    "StartTime": "2022-01-01T00:00:00Z",
    "EndTime": "2022-07-10T23:59:59Z"
  },
  "PropertyFilters": {
    "Properties": [
      {
        "Property": {
          "EoCloudCover": {
            "LowerBound": 0,
            "UpperBound": 1
          }
        }
      }
    ],
    "LogicalOperator": "AND"
  },
  "BandFilter": [
    "visual"
  ]
}
```

In this example, you query an `AreaOfInterest` that includes [Lake Mead](#) in Utah. Furthermore, Sentinel-2 supports multiple types of image bands. To measure the change in the surface of the water, you only need the `visual` band.

After you create the query parameters, you can use the `search_raster_data_collection` API to make the request.

The following code sample implements a `search_raster_data_collection` API request. This API does not support pagination using the `getPaginator` API. To make sure that the full API response has been gathered the code sample uses a `while` loop to check that `NextToken` exists. The code sample then uses `.extend()` to append the satellite image URLs and other response metadata to the `items_list`.

To learn more about the `search_raster_data_collection`, see [SearchRasterDataCollection](#) in the [Amazon SageMaker AI API Reference](#).

```
search_rdc_response = sm_geo_client.search_raster_data_collection(
    Arn='arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-collection/
public/nmqj48dcu3g7ayw8',
```

```
RasterDataCollectionQuery=search_rdc_query
)

## items_list is the response from the API request.
items_list = []

## Use the python .get() method to check that the 'NextToken' exists, if null returns
None breaking the while loop
while search_rdc_response.get('NextToken'):
    items_list.extend(search_rdc_response['Items'])
    search_rdc_response = sm_geo_client.search_raster_data_collection(
        Arn='arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-
collection/public/nmqj48dcu3g7ayw8',
        RasterDataCollectionQuery=search_rdc_query,
        NextToken=search_rdc_response['NextToken']
    )

## Print the number of observation return based on the query
print (len(items_list))
```

The following is a JSON response from your query. It has been truncated for clarity. Only the **"BandFilter": ["visual"]** specified in the request is returned in the Assets key-value pair:

```
{
    'Assets': {
        'visual': {
            'Href': 'https://sentinel-cogs.s3.us-west-2.amazonaws.com/sentinel-s2-l2a-
cogs/15/T/UH/2022/6/S2A_15TUH_20220623_0_L2A/TCI.tif'
        }
    },
    'DateTime': datetime.datetime(2022, 6, 23, 17, 22, 5, 926000, tzinfo = tzlocal()),
    'Geometry': {
        'Coordinates': [
            [
                [-114.529, 36.142],
                [-114.373, 36.142],
                [-114.373, 36.411],
                [-114.529, 36.411],
                [-114.529, 36.142],
            ]
        ],
        'Type': 'Polygon'
```

```
},
'Id': 'S2A_15TUH_20220623_0_L2A',
'Properties': {
    'EoCloudCover': 0.046519,
    'Platform': 'sentinel-2a'
}
}
```

Now that you have your query results, in the next section you can visualize the results by using `matplotlib`. This is to verify that results are from the correct geographical region.

## Visualizing your `search_raster_data_collection` using `matplotlib`

Before you start the earth observation job (EOJ), you can visualize a result from our query with `matplotlib`. The following code sample takes the first item, `items_list[0]["Assets"]["visual"]["Href"]`, from the `items_list` variable created in the previous code sample and prints an image using `matplotlib`.

```
# Visualize an example image.
import os
from urllib import request
import tifffile
import matplotlib.pyplot as plt

image_dir = "./images/lake_mead"
os.makedirs(image_dir, exist_ok=True)

image_dir = "./images/lake_mead"
os.makedirs(image_dir, exist_ok=True)

image_url = items_list[0]["Assets"]["visual"]["Href"]
img_id = image_url.split("/")[-2]
path_to_image = image_dir + "/" + img_id + "_TCI.tif"
response = request.urlretrieve(image_url, path_to_image)
print("Downloaded image: " + img_id)

tci = tifffile.imread(path_to_image)
plt.figure(figsize=(6, 6))
plt.imshow(tci)
plt.show()
```

After checking that the results are in the correct geographical region, you can start the Earth Observation Job (EOJ) in the next step. You use the EOJ to identify the water bodies from the satellite images by using a process called land segmentation.

## Starting an earth observation job (EOJ) that performs land segmentation on a series of Satellite images

SageMaker geospatial provides multiple pre-trained models that you can use to process geospatial data from raster data collections. To learn more about the available pre-trained models and custom operations, see [Types of Operations](#).

To calculate the change in the water surface area, you need to identify which pixels in the images correspond to water. Land cover segmentation is a semantic segmentation model supported by the `start_earth_observation_job` API. Semantic segmentation models associate a label with every pixel in each image. In the results, each pixel is assigned a label that's based on the class map for the model. The following is the class map for the land segmentation model:

```
{  
    0: "No_data",  
    1: "Saturated_or_defective",  
    2: "Dark_area_pixels",  
    3: "Cloud_shadows",  
    4: "Vegetation",  
    5: "Not_vegetated",  
    6: "Water",  
    7: "Unclassified",  
    8: "Cloud_medium_probability",  
    9: "Cloud_high_probability",  
    10: "Thin_cirrus",  
    11: "Snow_ice"  
}
```

To start an earth observation job, use the `start_earth_observation_job` API. When you submit your request, you must specify the following:

- `InputConfig (dict)` – Used to specify the coordinates of the area that you want to search, and other metadata that's associated with your search.
- `JobConfig (dict)` – Used to specify the type of EOJ operation that you performed on the data. This example uses **LandCoverSegmentationConfig**.

- **ExecutionRoleArn (string)** – The ARN of the SageMaker AI execution role with the necessary permissions to run the job.
- **Name (string)** – A name for the earth observation job.

The `InputConfig` is a Python dictionary. Use the following variable `ejj_input_config` to hold the search query parameters. Use this variable when you make the `start_earth_observation_job` API request. w.

```
# Perform land cover segmentation on images returned from the Sentinel-2 dataset.
ejj_input_config = {
    "RasterDataCollectionQuery": {
        "RasterDataCollectionArn": "arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-collection/public/nmqj48dcu3g7ayw8",
        "AreaOfInterest": {
            "AreaOfInterestGeometry": {
                "PolygonGeometry": {
                    "Coordinates": [
                        [
                            [
                                [-114.529, 36.142],
                                [-114.373, 36.142],
                                [-114.373, 36.411],
                                [-114.529, 36.411],
                                [-114.529, 36.142]
                            ]
                        ]
                    ]
                }
            },
            "TimeRangeFilter": {
                "StartTime": "2021-01-01T00:00:00Z",
                "EndTime": "2022-07-10T23:59:59Z",
            },
            "PropertyFilters": {
                "Properties": [{"Property": {"EoCloudCover": {"LowerBound": 0,
"UpperBound": 1}}}],
                "LogicalOperator": "AND",
            },
        }
    }
}
```

The JobConfig is a Python dictionary that is used to specify the EOJ operation that you want performed on your data:

```
eoj_config = {"LandCoverSegmentationConfig": {}}
```

With the dictionary elements now specified, you can submit your start\_earth\_observation\_job API request using the following code sample:

```
# Gets the execution role arn associated with current notebook instance
execution_role_arn = sagemaker.get_execution_role()

# Starts an earth observation job
response = sm_geo_client.start_earth_observation_job(
    Name="lake-mead-landcover",
    InputConfig=eoj_input_config,
    JobConfig=eoj_config,
    ExecutionRoleArn=execution_role_arn,
)

print(response)
```

The start an earth observation job returns an ARN along with other metadata.

To get a list of all ongoing and current earth observation jobs use the list\_earth\_observation\_jobs API. To monitor the status of a single earth observation job use the get\_earth\_observation\_job API. To make this request, use the ARN created after submitting your EOJ request. To learn more, see [GetEarthObservationJob](#) in the *Amazon SageMaker AI API Reference*.

To find the ARNs associated with your EOJs use the list\_earth\_observation\_jobs API operation. To learn more, see [ListEarthObservationJobs](#) in the *Amazon SageMaker AI API Reference*.

```
# List all jobs in the account
sg_client.list_earth_observation_jobs()["EarthObservationJobSummaries"]
```

The following is an example JSON response:

```
{
    'Arn': 'arn:aws:sagemaker-geospatial:us-west-2:111122223333:earth-observation-job/futg3vuq935t',
```

```
'CreationTime': datetime.datetime(2023, 10, 19, 4, 33, 54, 21481, tzinfo =  
tzlocal()),  
'DurationInSeconds': 3493,  
'Name': 'lake-mead-landcover',  
'OperationType': 'LAND_COVER_SEGMENTATION',  
'Status': 'COMPLETED',  
'Tags': {}  
}, {  
    'Arn': 'arn:aws:sagemaker-geospatial:us-west-2:111122223333:earth-observation-job/  
wu8j9x42zw3d',  
    'CreationTime': datetime.datetime(2023, 10, 20, 0, 3, 27, 270920, tzinfo =  
tzlocal()),  
    'DurationInSeconds': 1,  
    'Name': 'mt-shasta-landcover',  
    'OperationType': 'LAND_COVER_SEGMENTATION',  
    'Status': 'INITIALIZING',  
    'Tags': {}  
}
```

After the status of your EOJ job changes to COMPLETED, proceed to the next section to calculate the change in Lake Mead's surface area.

## Calculating the change in the Lake Mead surface area

To calculate the change in Lake Mead's surface area, first export the results of the EOJ to Amazon S3 by using `export_earth_observation_job`:

```
sagemaker_session = sagemaker.Session()  
s3_bucket_name = sagemaker_session.default_bucket() # Replace with your own bucket if  
needed  
s3_bucket = session.resource("s3").Bucket(s3_bucket_name)  
prefix = "export-lake-mead-eoj" # Replace with the S3 prefix desired  
export_bucket_and_key = f"s3://{s3_bucket_name}/{prefix}/"  
  
eoj_output_config = {"S3Data": {"S3Uri": export_bucket_and_key}}  
export_response = sm_geo_client.export_earth_observation_job(  
    Arn="arn:aws:sagemaker-geospatial:us-west-2:111122223333:earth-observation-  
job/7xgwzijebypn",  
    ExecutionRoleArn=execution_role_arn,  
    OutputConfig=eoj_output_config,  
    ExportSourceImages=False,  
)
```

To see the status of your export job, use `get_earth_observation_job`:

```
export_job_details =  
    sm_geo_client.get_earth_observation_job(Arn=export_response["Arn"])
```

To calculate the changes in Lake Mead's water level, download the land cover masks to the local SageMaker notebook instance and download the source images from our previous query. In the class map for the land segmentation model, the water's class index is 6.

To extract the water mask from a Sentinel-2 image, follow these steps. First, count the number of pixels marked as water (class index 6) in the image. Second, multiply the count by the area that each pixel covers. Bands can differ in their spatial resolution. For the land cover segmentation model all bands are down sampled to a spatial resolution equal to 60 meters.

```
import os  
from glob import glob  
import cv2  
import numpy as np  
import tifffile  
import matplotlib.pyplot as plt  
from urllib.parse import urlparse  
from botocore import UNSIGNED  
from botocore.config import Config  
  
# Download land cover masks  
mask_dir = "./masks/lake_mead"  
os.makedirs(mask_dir, exist_ok=True)  
image_paths = []  
for s3_object in s3_bucket.objects.filter(Prefix=prefix).all():  
    path, filename = os.path.split(s3_object.key)  
    if "output" in path:  
        mask_name = mask_dir + "/" + filename  
        s3_bucket.download_file(s3_object.key, mask_name)  
        print("Downloaded mask: " + mask_name)  
  
# Download source images for visualization  
for tci_url in tci_urls:  
    url_parts = urlparse(tci_url)  
    img_id = url_parts.path.split("/")[-2]  
    tci_download_path = image_dir + "/" + img_id + "_TCI.tif"  
    cogs_bucket = session.resource(  
        "s3", config=Config(signature_version=UNSIGNED, region_name="us-west-2"))
```

```
.).Bucket(url_parts.hostname.split(".")[0])
cogs_bucket.download_file(url_parts.path[1:], tci_download_path)
print("Downloaded image: " + img_id)

print("Downloads complete.")

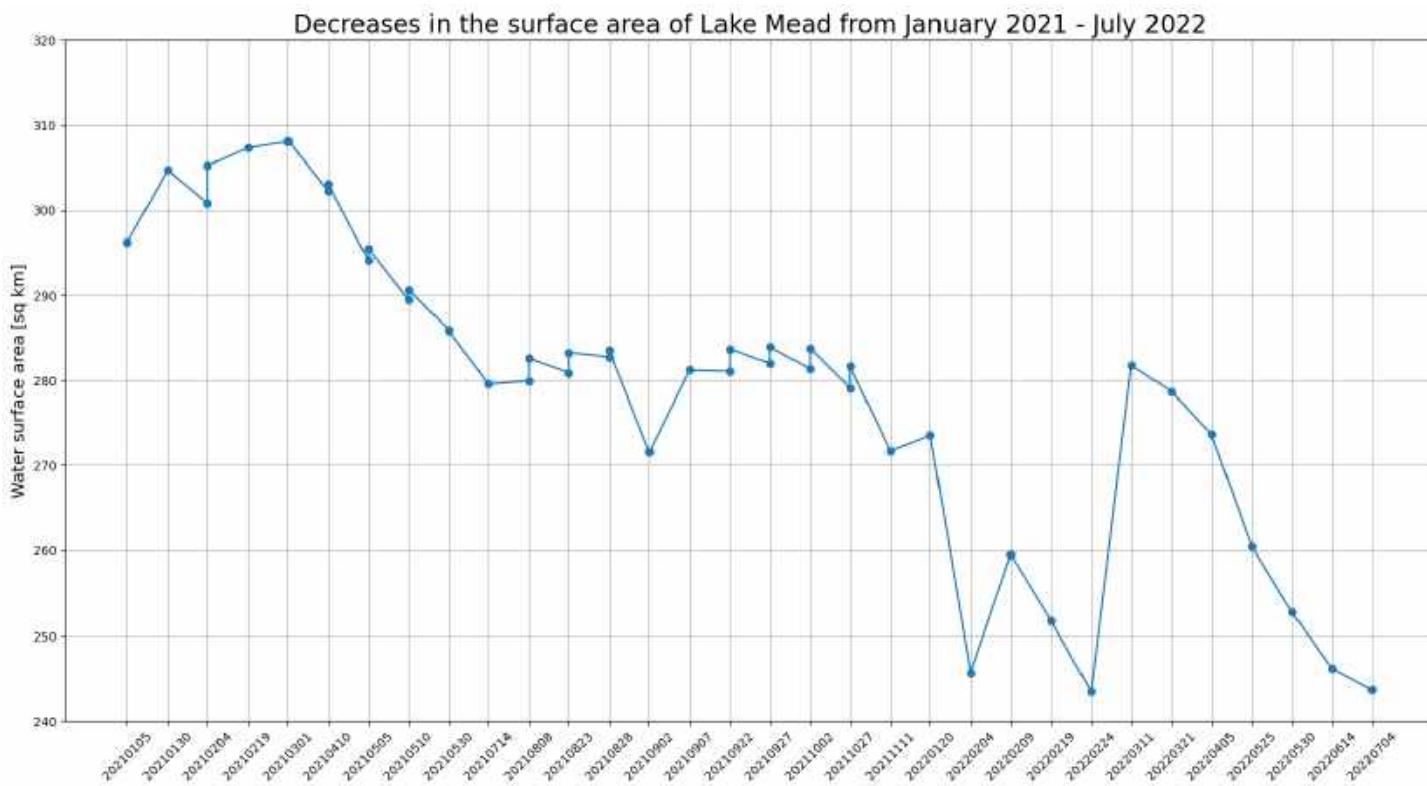
image_files = glob("images/lake_mead/*.tif")
mask_files = glob("masks/lake_mead/*.tif")
image_files.sort(key=lambda x: x.split("SQA_")[1])
mask_files.sort(key=lambda x: x.split("SQA_")[1])
overlay_dir = "./masks/lake_mead_overlay"
os.makedirs(overlay_dir, exist_ok=True)
lake_areas = []
mask_dates = []

for image_file, mask_file in zip(image_files, mask_files):
    image_id = image_file.split("/")[-1].split("_TCI")[0]
    mask_id = mask_file.split("/")[-1].split(".tif")[0]
    mask_date = mask_id.split("_")[2]
    mask_dates.append(mask_date)
    assert image_id == mask_id
    image = tifffile.imread(image_file)
    image_ds = cv2.resize(image, (1830, 1830), interpolation=cv2.INTER_LINEAR)
    mask = tifffile.imread(mask_file)
    water_mask = np.isin(mask, [6]).astype(np.uint8) # water has a class index 6
    lake_mask = water_mask[1000:, :1100]
    lake_area = lake_mask.sum() * 60 * 60 / (1000 * 1000) # calculate the surface area
    lake_areas.append(lake_area)
    contour, _ = cv2.findContours(water_mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    combined = cv2.drawContours(image_ds, contour, -1, (255, 0, 0), 4)
    lake_crop = combined[1000:, :1100]
    cv2.putText(lake_crop, f"{mask_date}", (10,50), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 0, 0), 3, cv2.LINE_AA)
    cv2.putText(lake_crop, f"{lake_area} [sq km]", (10,100), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 0, 0), 3, cv2.LINE_AA)
    overlay_file = overlay_dir + '/' + mask_date + '.png'
    cv2.imwrite(overlay_file, cv2.cvtColor(lake_crop, cv2.COLOR_RGB2BGR))

# Plot water surface area vs. time.
plt.figure(figsize=(20,10))
plt.title('Lake Mead surface area for the 2021.02 - 2022.07 period.', fontsize=20)
plt.xticks(rotation=45)
plt.ylabel('Water surface area [sq km]', fontsize=14)
plt.plot(mask_dates, lake_areas, marker='o')
```

```
plt.grid('on')
plt.ylim(240, 320)
for i, v in enumerate(lake_areas):
    plt.text(i, v+2, "%d" %v, ha='center')
plt.show()
```

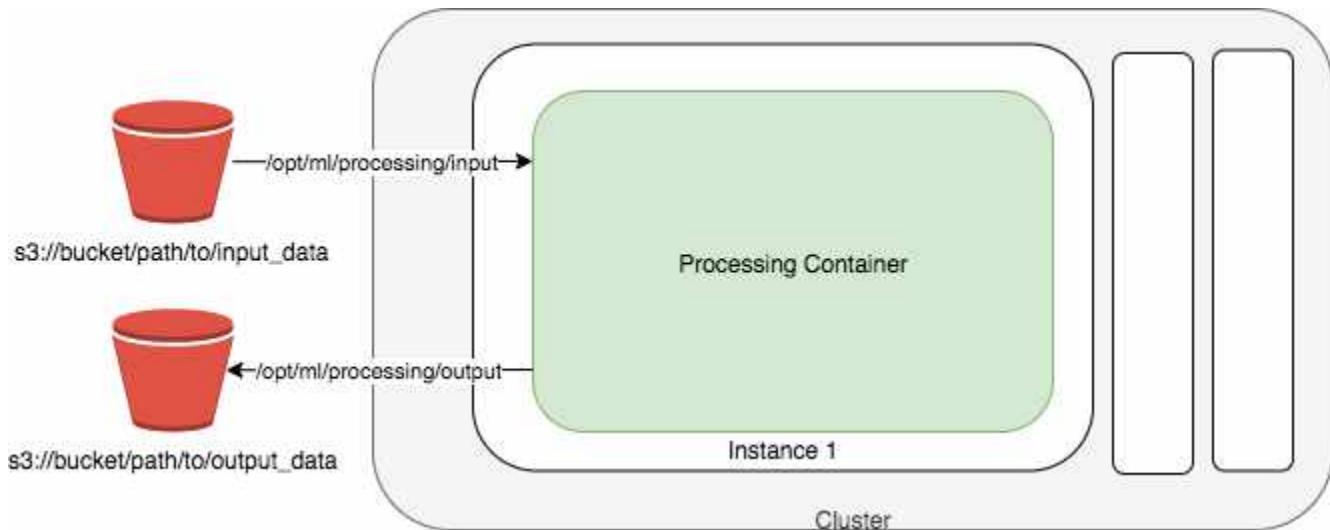
Using matplotlib, you can visualize the results with a graph. The graph shows that the surface area of Lake Mead decreased from January 2021–July 2022.



## Using a processing jobs for custom geospatial workloads

With [Amazon SageMaker Processing](#), you can use a simplified, managed experience on SageMaker AI to run your data processing workloads with the purpose-built geospatial container.

The underlying infrastructure for a Amazon SageMaker Processing job is fully managed by SageMaker AI. During a processing job, cluster resources are provisioned for the duration of your job, and cleaned up when a job completes.



The preceding diagram shows how SageMaker AI spins up a geospatial processing job. SageMaker AI takes your geospatial workload script, copies your geospatial data from Amazon Simple Storage Service(Amazon S3), and then pulls the specified geospatial container. The underlying infrastructure for the processing job is fully managed by SageMaker AI. Cluster resources are provisioned for the duration of your job, and cleaned up when a job completes. The output of the processing job is stored in the bucket you specified.

### ⚠ Path naming constraints

The local paths inside a Processing jobs container must begin with **/opt/ml/processing/**.

SageMaker geospatial provides a purpose-built container, `081189585635.dkr.ecr.us-west-2.amazonaws.com/sagemaker-geospatial-v1-0:latest` that can be specified when running a processing job.

## Topics

- [Overview: Run processing jobs using ScriptProcessor and a SageMaker geospatial container](#)
- [Using ScriptProcessor to calculate the Normalized Difference Vegetation Index \(NDVI\) using Sentinel-2 satellite data](#)

## Overview: Run processing jobs using `ScriptProcessor` and a SageMaker geospatial container

SageMaker geospatial provides a purpose-built processing container, `081189585635.dkr.ecr.us-west-2.amazonaws.com/sagemaker-geospatial-v1-0:latest`. You can use this container when running a job with Amazon SageMaker Processing. When you create an instance of the [ScriptProcessor](#) class that is available through the *Amazon SageMaker Python SDK for Processing*, specify this `image_uri`.

### Note

If you receive a `ResourceLimitExceeded` error when attempting to start a processing job, you need to request a quota increase. To get started on a Service Quotas quota increase request, see [Requesting a quota increase](#) in the *Service Quotas User Guide*

### Prerequisites for using `ScriptProcessor`

1. You have created a Python script that specifies your geospatial ML workload.
2. You have granted the SageMaker AI execution role access to any Amazon S3 buckets that are needed.
3. Prepare your data for import into the container. Amazon SageMaker Processing jobs support either setting the `s3_data_type` equal to "ManifestFile" or to "S3Prefix".

The following procedure show you how to create an instance of `ScriptProcessor` and submit a Amazon SageMaker Processing job using the SageMaker geospatial container.

### To create a `ScriptProcessor` instance and submit a Amazon SageMaker Processing job using a SageMaker geospatial container

1. Instantiate an instance of the `ScriptProcessor` class using the SageMaker geospatial image:

```
from sagemaker.processing import ScriptProcessor, ProcessingInput, ProcessingOutput  
  
sm_session = sagemaker.session.Session()  
execution_role_arn = sagemaker.get_execution_role()  
  
# purpose-built geospatial container
```

```
image_uri = '081189585635.dkr.ecr.us-west-2.amazonaws.com/sagemaker-geospatial-v1-0:latest'

script_processor = ScriptProcessor(
    command=['python3'],
    image_uri=image_uri,
    role=execution_role_arn,
    instance_count=4,
    instance_type='ml.m5.4xlarge',
    sagemaker_session=sm_session
)
```

Replace `execution_role_arn` with the ARN of the SageMaker AI execution role that has access to the input data stored in Amazon S3 and any other AWS services that you want to call in your processing job. You can update the `instance_count` and the `instance_type` to match the requirements of your processing job.

## 2. To start a processing job, use the `.run( )` method:

```
# Can be replaced with any S3 compliant string for the name of the folder.
s3_folder = geospatial-data-analysis

# Use .default_bucket() to get the name of the S3 bucket associated with your current
# SageMaker session
s3_bucket = sm_session.default_bucket()

s3_manifest_uri = f's3://{s3_bucket}/{s3_folder}/manifest.json'
s3_prefix_uri = f's3://{s3_bucket}/{s3_folder}/image-prefix

script_processor.run(
    code='preprocessing.py',
    inputs=[
        ProcessingInput(
            source=s3_manifest_uri | s3_prefix_uri ,
            destination='/opt/ml/processing/input_data/',
            s3_data_type= "ManifestFile" | "S3Prefix",
            s3_data_distribution_type= "ShardedByS3Key" | "FullyReplicated"
        )
    ],
    outputs=[
        ProcessingOutput(
            source='/opt/ml/processing/output_data/',
            destination=s3_output_prefix_url
    )
)
```

```
)  
]  
)
```

- Replace *preprocessing.py* with the name of your own Python data processing script.
- A processing job supports two methods for formatting your input data. You can either create a manifest file that points to all of the input data for your processing job, or you can use a common prefix on each individual data input. If you created a manifest file set `s3_manifest_uri` equal to "ManifestFile". If you used a file prefix set `s3_manifest_uri` equal to "S3Prefix". You specify the path to your data using `source`.
- You can distribute your processing job data two ways:
  - Distribute your data to all processing instances by setting `s3_data_distribution_type` equal to `FullyReplicated`.
  - Distribute your data in shards based on the Amazon S3 key by setting `s3_data_distribution_type` equal to `ShardedByS3Key`. When you use `ShardedByS3Key` one shard of data is sent to each processing instance.

You can use a script to process SageMaker geospatial data. That script can be found in [Step 3: Writing a script that can calculate the NDVI](#). To learn more about the `.run()` API operation, see [run in the Amazon SageMaker Python SDK for Processing](#).

To monitor the progress of your processing job, the `ProcessingJobs` class supports a [describe](#) method. This method returns a response from the `DescribeProcessingJob` API call. To learn more, see [DescribeProcessingJob in the Amazon SageMaker AI API Reference](#).

The next topic show you how to create an instance of the `ScriptProcessor` class using the SageMaker geospatial container, and then how to use it to calculate the Normalized Difference Vegetation Index (NDVI) with Sentinel-2 images.

## Using `ScriptProcessor` to calculate the Normalized Difference Vegetation Index (NDVI) using Sentinel-2 satellite data

The following code samples show you how to calculate the normalized difference vegetation index of a specific geographical area using the purpose-built geospatial image within a Studio Classic notebook and run a large-scale workload with Amazon SageMaker Processing using [ScriptProcessor](#) from the SageMaker AI Python SDK.

This demo also uses an Amazon SageMaker Studio Classic notebook instance that uses the geospatial kernel and instance type. To learn how to create a Studio Classic geospatial notebook instance, see [Create an Amazon SageMaker Studio Classic notebook using the geospatial image](#).

You can follow along with this demo in your own notebook instance by copying and pasting the following code snippets:

1. [Use search\\_raster\\_data\\_collection to query a specific area of interest \(AOI\) over a given a time range using a specific raster data collection, Sentinel-2.](#)
2. [Create a manifest file that specifies what data will be processed during the processing job.](#)
3. [Write a data processing Python script calculating the NDVI.](#)
4. [Create a ScriptProcessor instance and start the Amazon SageMaker Processing job.](#)
5. [Visualizing the results of your processing job.](#)

## Query the Sentinel-2 raster data collection using SearchRasterDataCollection

With `search_raster_data_collection` you can query supported raster data collections. This example uses data that's pulled from Sentinel-2 satellites. The area of interest (`AreaOfInterest`) specified is rural northern Iowa, and the time range (`TimeRangeFilter`) is January 1, 2022 to December 30, 2022. To see the available raster data collections in your AWS Region use `list_raster_data_collections`. To see a code example using this API, see [ListRasterDataCollections](#) in the *Amazon SageMaker AI Developer Guide*.

In following code examples you use the ARN associated with Sentinel-2 raster data collection, `arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-collection/public/nmqj48dcu3g7ayw8`.

A `search_raster_data_collection` API request requires two parameters:

- You need to specify an `Arn` parameter that corresponds to the raster data collection that you want to query.
- You also need to specify a `RasterDataCollectionQuery` parameter, which takes in a Python dictionary.

The following code example contains the necessary key-value pairs needed for the `RasterDataCollectionQuery` parameter saved to the `search_rdc_query` variable.

```
search_rdc_query = {
    "AreaOfInterest": {
        "AreaOfInterestGeometry": {
            "PolygonGeometry": {
                "Coordinates": [
                    [
                        [
                            -94.50938680498298,
                            43.22487436936203
                        ],
                        [
                            [
                                -94.50938680498298,
                                42.843474642037194
                            ],
                            [
                                [
                                    -93.86520004156142,
                                    42.843474642037194
                                ],
                                [
                                    [
                                        -93.86520004156142,
                                        43.22487436936203
                                    ],
                                    [
                                        [
                                            -94.50938680498298,
                                            43.22487436936203
                                        ]
                                    ]
                                ]
                            ]
                        ]
                    }
                }
            },
            "TimeRangeFilter": {"StartTime": "2022-01-01T00:00:00Z", "EndTime": "2022-12-30T23:59:59Z"}
        }
    }
}
```

To make the `search_raster_data_collection` request, you must specify the ARN of the Sentinel-2 raster data collection: `arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-collection/public/nmqj48dcu3g7ayw8`. You also must need to pass in the Python dictionary that was defined previously, which specifies query parameters.

```
## Creates a SageMaker Geospatial client instance
sm_geo_client= session.create_client(service_name="sagemaker-geospatial")
```

```
search_rdc_response1 = sm_geo_client.search_raster_data_collection(  
    Arn='arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-collection/  
public/nmqj48dcu3g7ayw8',  
    RasterDataCollectionQuery=search_rdc_query  
)
```

The results of this API can not be paginated. To collect all the satellite images returned by the `search_raster_data_collection` operation, you can implement a while loop. This checks for `NextToken` in the API response:

```
## Holds the list of API responses from search_raster_data_collection  
items_list = []  
while search_rdc_response1.get('NextToken') and search_rdc_response1['NextToken'] !=  
None:  
    items_list.extend(search_rdc_response1['Items'])  
  
    search_rdc_response1 = sm_geo_client.search_raster_data_collection(  
        Arn='arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-collection/  
public/nmqj48dcu3g7ayw8',  
        RasterDataCollectionQuery=search_rdc_query,  
        NextToken=search_rdc_response1['NextToken']  
)
```

The API response returns a list of URLs under the `Assets` key corresponding to specific image bands. The following is a truncated version of the API response. Some of the image bands were removed for clarity.

```
{  
    'Assets': {  
        'aot': {  
            'Href': 'https://sentinel-cogs.s3.us-west-2.amazonaws.com/sentinel-s2-l2a-  
cogs/15/T/UH/2022/12/S2A_15TUH_20221230_0_L2A/A0T.tif'  
        },  
        'blue': {  
            'Href': 'https://sentinel-cogs.s3.us-west-2.amazonaws.com/sentinel-s2-l2a-  
cogs/15/T/UH/2022/12/S2A_15TUH_20221230_0_L2A/B02.tif'  
        },  
        'swir22-jp2': {  
            'Href': 's3://sentinel-s2-l2a/tiles/15/T/UH/2022/12/30/0/B12.jp2'  
        },  
        'visual-jp2': {  
            'Href': 's3://sentinel-s2-l2a/tiles/15/T/UH/2022/12/30/0/TCI.jp2'
```

```
        },
        'wvp-jp2': {
            'Href': 's3://sentinel-s2-l2a/tiles/15/T/UH/2022/12/30/0/WVP.jp2'
        }
    },
    'DateTime': datetime.datetime(2022, 12, 30, 17, 21, 52, 469000, tzinfo =
tzlocal()),
    'Geometry': {
        'Coordinates': [
            [
                [-95.46676936182894, 43.32623760511659],
                [-94.11293433656887, 43.347431265475954],
                [-94.09532154452742, 42.35884880571144],
                [-95.42776890002203, 42.3383710796791],
                [-95.46676936182894, 43.32623760511659]
            ]
        ],
        'Type': 'Polygon'
    },
    'Id': 'S2A_15TUH_20221230_0_L2A',
    'Properties': {
        'EoCloudCover': 62.384969,
        'Platform': 'sentinel-2a'
    }
}
```

In the [next section](#), you create a manifest file using the 'Id' key from the API response.

## Create an input manifest file using the Id key from the search\_raster\_data\_collection API response

When you run a processing job, you must specify a data input from Amazon S3. The input data type can either be a manifest file, which then points to the individual data files. You can also add a prefix to each file that you want processed. The following code example defines the folder where your manifest files will be generated.

Use SDK for Python (Boto3) to get the default bucket and the ARN of the execution role that is associated with your Studio Classic notebook instance:

```
sm_session = sagemaker.session.Session()
s3 = boto3.resource('s3')
# Gets the default excution role associated with the notebook
```

```
execution_role_arn = sagemaker.get_execution_role()

# Gets the default bucket associated with the notebook
s3_bucket = sm_session.default_bucket()

# Can be replaced with any name
s3_folder = "script-processor-input-manifest"
```

Next, you create a manifest file. It will hold the URLs of the satellite images that you wanted processed when you run your processing job later in step 4.

```
# Format of a manifest file
manifest_prefix = []
manifest_prefix['prefix'] = 's3://' + s3_bucket + '/' + s3_folder + '/'
manifest = [manifest_prefix]

print(manifest)
```

The following code sample returns the S3 URI where your manifest files will be created.

```
[{'prefix': 's3://sagemaker-us-west-2-111122223333/script-processor-input-manifest/'}]
```

All the response elements from the `search_raster_data_collection` response are not needed to run the processing job.

The following code snippet removes the unnecessary elements '`Properties`', '`Geometry`', and '`DateTime`'. The '`Id`' key-value pair, '`Id': 'S2A_15TUH_20221230_0_L2A'`', contains the year and the month. The following code example parses that data to create new keys in the Python dictionary `dict_month_items`. The values are the assets that are returned from the `SearchRasterDataCollection` query.

```
# For each response get the month and year, and then remove the metadata not related to
# the satelite images.
dict_month_items = {}
for item in items_list:
    # Example ID being split: 'S2A_15TUH_20221230_0_L2A'
    yyyyymm = item['Id'].split("_")[2][:6]
    if yyyyymm not in dict_month_items:
        dict_month_items[yyyyymm] = []
```

```
# Removes unneeded metadata elements for this demo
item.pop('Properties', None)
item.pop('Geometry', None)
item.pop('DateTime', None)

# Appends the response from search_raster_data_collection to newly created key
above
dict_month_items[yyyymm].append(item)
```

This code example uploads the `dict_month_items` to Amazon S3 as a JSON object using the [`.upload\_file\(\)`](#) API operation:

```
## key_ is the yyyymm timestamp formatted above
## value_ is the reference to all the satellite images collected via our searchRDC
query
for key_, value_ in dict_month_items.items():
    filename = f'manifest_{key_}.json'
    with open(filename, 'w') as fp:
        json.dump(value_, fp)
    s3.meta.client.upload_file(filename, s3_bucket, s3_folder + '/' + filename)
    manifest.append(filename)
    os.remove(filename)
```

This code example uploads a parent `manifest.json` file that points to all the other manifests uploaded to Amazon S3. It also saves the path to a local variable: `s3_manifest_uri`. You'll use that variable again to specify the source of the input data when you run the processing job in step 4.

```
with open('manifest.json', 'w') as fp:
    json.dump(manifest, fp)
s3.meta.client.upload_file('manifest.json', s3_bucket, s3_folder + '/' +
    'manifest.json')
os.remove('manifest.json')

s3_manifest_uri = f's3://{s3_bucket}/{s3_folder}/manifest.json'
```

Now that you created the input manifest files and uploaded them, you can write a script that processes your data in the processing job. It processes the data from the satellite images, calculates the NDVI, and then returns the results to a different Amazon S3 location.

## Write a script that calculates the NDVI

Amazon SageMaker Studio Classic supports the use of the `%%writefile` cell magic command. After running a cell with this command, its contents will be saved to your local Studio Classic directory. This is code specific to calculating NDVI. However, the following can be useful when you write your own script for a processing job:

- In your processing job container, the local paths inside the container must begin with `/opt/ml/processing/`. In this example, `input_data_path = '/opt/ml/processing/input_data/'` and `processed_data_path = '/opt/ml/processing/output_data/'` are specified in that way.
- With Amazon SageMaker Processing, a script that a processing job runs can upload your processed data directly to Amazon S3. To do so, make sure that the execution role associated with your `ScriptProcessor` instance has the necessary requirements to access the S3 bucket. You can also specify an `outputs` parameter when you run your processing job. To learn more, see the [.run\(\) API operation](#) in the *Amazon SageMaker Python SDK*. In this code example, the results of the data processing are uploaded directly to Amazon S3.
- To manage the size of the Amazon EBS container attached to your processing job use the `volume_size_in_gb` parameter. The container's default size is 30 GB. You can also optionally use the Python library [Garbage Collector](#) to manage storage in your Amazon EBS container.

The following code example loads the arrays into the processing job container. When arrays build up and fill in the memory, the processing job crashes. To prevent this crash, the following example contains commands that remove the arrays from the processing job's container.

```
%%writefile compute_ndvi.py

import os
import pickle
import sys
import subprocess
import json
import rioxarray

if __name__ == "__main__":
    print("Starting processing")

    input_data_path = '/opt/ml/processing/input_data/'
    input_files = []
```

```
for current_path, sub_dirs, files in os.walk(input_data_path):
    for file in files:
        if file.endswith(".json"):
            input_files.append(os.path.join(current_path, file))

print("Received {} input_files: {}".format(len(input_files), input_files))

items = []
for input_file in input_files:
    full_file_path = os.path.join(input_data_path, input_file)
    print(full_file_path)
    with open(full_file_path, 'r') as f:
        items.append(json.load(f))

items = [item for sub_items in items for item in sub_items]

for item in items:
    red_uri = item["Assets"]["red"]["Href"]
    nir_uri = item["Assets"]["nir"]["Href"]

    red = rioxarray.open_rasterio(red_uri, masked=True)
    nir = rioxarray.open_rasterio(nir_uri, masked=True)

    ndvi = (nir - red) / (nir + red)

    file_name = 'ndvi_' + item["Id"] + '.tif'
    output_path = '/opt/ml/processing/output_data'
    output_file_path = f'{output_path}/{file_name}'

    ndvi.rio.to_raster(output_file_path)
    print("Written output:", output_file_path)
```

You now have a script that can calculate the NDVI. Next, you can create an instance of the `ScriptProcessor` and run your Processing job.

## Creating an instance of the `ScriptProcessor` class

This demo uses the `ScriptProcessor` class that is available via the Amazon SageMaker Python SDK. First, you need to create an instance of the class, and then you can start your Processing job by using the `.run()` method.

```
from sagemaker.processing import ScriptProcessor, ProcessingInput, ProcessingOutput
```

```
image_uri = '081189585635.dkr.ecr.us-west-2.amazonaws.com/sagemaker-geospatial-v1-0:latest'

processor = ScriptProcessor(
    command=['python3'],
    image_uri=image_uri,
    role=execution_role_arn,
    instance_count=4,
    instance_type='ml.m5.4xlarge',
    sagemaker_session=sm_session
)

print('Starting processing job.')
```

When you start your Processing job, you need to specify a [ProcessingInput](#) object. In that object, you specify the following:

- The path to the manifest file that you created in step 2, `s3_manifest_uri`. This is the source of the input data to the container.
- The path to where you want the input data to be saved in the container. This must match the path that you specified in your script.
- Use the `s3_data_type` parameter to specify the input as "ManifestFile".

```
s3_output_prefix_url = f"s3://{s3_bucket}/{s3_folder}/output"

processor.run(
    code='compute_ndvi.py',
    inputs=[
        ProcessingInput(
            source=s3_manifest_uri,
            destination='/opt/ml/processing/input_data/',
            s3_data_type="ManifestFile",
            s3_data_distribution_type="ShardedByS3Key"
        ),
    ],
    outputs=[
        ProcessingOutput(
            source='/opt/ml/processing/output_data/',
            destination=s3_output_prefix_url,
            s3_upload_mode="Continuous"
    )
)
```

```
)  
]  
)
```

The following code example uses the [.describe\(\) method](#) to get details of your Processing job.

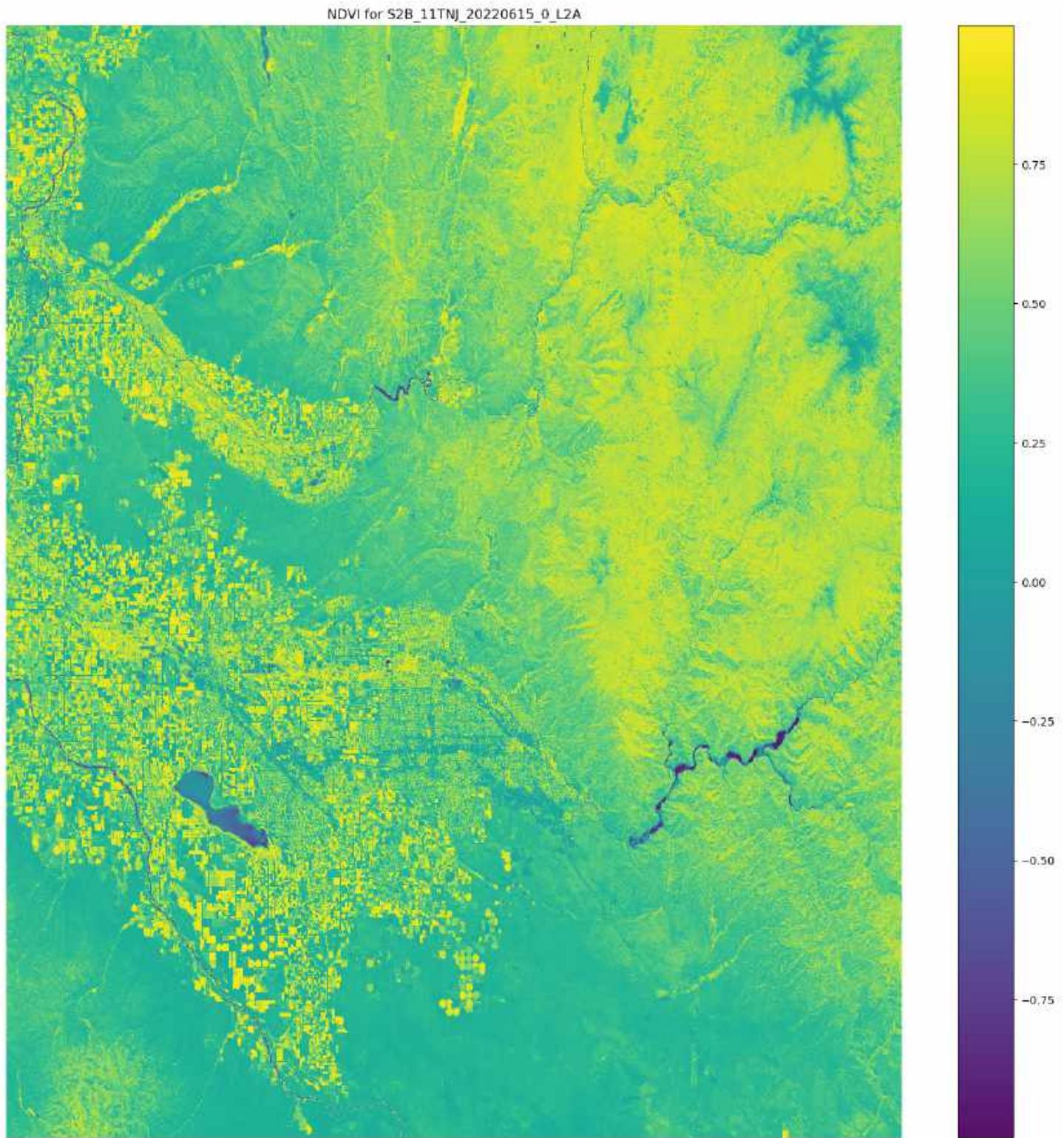
```
preprocessing_job_descriptor = processor.jobs[-1].describe()  
s3_output_uri = preprocessing_job_descriptor["ProcessingOutputConfig"]["Outputs"][0]  
["S3Output"]["S3Uri"]  
print(s3_output_uri)
```

## Visualizing your results using matplotlib

With the [Matplotlib](#) Python library, you can plot raster data. Before you plot the data, you need to calculate the NDVI using sample images from the Sentinel-2 satellites. The following code example opens the image arrays using the `.open_rasterio()` API operation, and then calculates the NDVI using the `nir` and `red` image bands from the Sentinel-2 satellite data.

```
# Opens the python arrays  
import rioxarray  
  
red_uri = items[25]["Assets"]["red"]["Href"]  
nir_uri = items[25]["Assets"]["nir"]["Href"]  
  
red = rioxarray.open_rasterio(red_uri, masked=True)  
nir = rioxarray.open_rasterio(nir_uri, masked=True)  
  
# Calculates the NDVI  
ndvi = (nir - red)/ (nir + red)  
  
# Common plotting library in Python  
import matplotlib.pyplot as plt  
  
f, ax = plt.subplots(figsize=(18, 18))  
ndvi.plot(cmap='viridis', ax=ax)  
ax.set_title("NDVI for {}".format(items[25]["Id"]))  
ax.set_axis_off()  
plt.show()
```

The output of the preceding code example is a satellite image with the NDVI values overlaid on it. An NDVI value near 1 indicates lots of vegetation is present, and values near 0 indicate no vegetation is present.



This completes the demo of using ScriptProcessor.

## Earth Observation Jobs

Using an Earth Observation job (EOJ), you can acquire, transform, and visualize geospatial data to make predictions. You can choose an operation based on your use case from a wide range of operations and models. You get the flexibility of choosing your area of interest, selecting the data providers, and setting time-range based and cloud-cover-percentage-based filters. After SageMaker AI creates an EOJ for you, you can visualize the inputs and outputs of the job using the visualization functionality. An EOJ has various use cases that include comparing deforestation over time and diagnosing plant health. You can create an EOJ by using a SageMaker notebook with a SageMaker geospatial image. You can also access the SageMaker geospatial UI as a part of Amazon SageMaker Studio Classic UI to view the list of all your jobs. You can also use the UI to pause or stop an ongoing job. You can choose a job from the list of available EOJ to view the **Job summary**, the **Job details** as well as visualize the **Job output**.

### Topics

- [Create an Earth Observation Job Using a Amazon SageMaker Studio Classic Notebook with a SageMaker geospatial Image](#)
- [Types of Operations](#)

## Create an Earth Observation Job Using a Amazon SageMaker Studio Classic Notebook with a SageMaker geospatial Image

To use a SageMaker Studio Classic notebook with a SageMaker geospatial image:

1. From the **Launcher**, choose **Change environment** under **Notebooks and compute resources**.
2. Next, the **Change environment** dialog opens.
3. Select the **Image** dropdown and choose **Geospatial 1.0**. The **Instance type** should be **ml.geospatial.interactive**. Do not change the default values for other settings.
4. Choose **Select**.
5. Choose **Create notebook**.

You can initiate an EOJ using a Amazon SageMaker Studio Classic notebook with a SageMaker geospatial image using the code provided below.

```
import boto3
import sagemaker
```

```
import sagemaker_geospatial_map

session = boto3.Session()
execution_role = sagemaker.get_execution_role()
sg_client = session.client(service_name="sagemaker-geospatial")
```

The following is an example showing how to create an EOJ in the US West (Oregon) Region.

```
#Query and Access Data
search_rdc_args = {
    "Arn": "arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-collection/public/nmqj48dcu3g7ayw8", # sentinel-2 L2A COG
    "RasterDataCollectionQuery": {
        "AreaOfInterest": {
            "AreaOfInterestGeometry": {
                "PolygonGeometry": {
                    "Coordinates": [
                        [
                            [
                                [-114.529, 36.142],
                                [-114.373, 36.142],
                                [-114.373, 36.411],
                                [-114.529, 36.411],
                                [-114.529, 36.142],
                            ]
                        ]
                    ]
                }
            }
        },
        "TimeRangeFilter": {
            "StartTime": "2021-01-01T00:00:00Z",
            "EndTime": "2022-07-10T23:59:59Z",
        },
        "PropertyFilters": {
            "Properties": [{"Property": {"EoCloudCover": {"LowerBound": 0,
"UpperBound": 1}}}],
            "LogicalOperator": "AND",
        },
        "BandFilter": ["visual"],
    },
}
tci_urls = []
data_manifests = []
```

```
while search_rdc_args.get("NextToken", True):
    search_result = sg_client.search_raster_data_collection(**search_rdc_args)
    if search_result.get("NextToken"):
        data_manifests.append(search_result)
    for item in search_result["Items"]:
        tci_url = item["Assets"]["visual"]["Href"]
        print(tci_url)
        tci_urls.append(tci_url)

    search_rdc_args["NextToken"] = search_result.get("NextToken")

# Perform land cover segmentation on images returned from the sentinel dataset.
eoj_input_config = {
    "RasterDataCollectionQuery": {
        "RasterDataCollectionArn": "arn:aws:sagemaker-geospatial:us-west-2:378778860802:raster-data-collection/public/nmqj48dcu3g7ayw8",
        "AreaOfInterest": {
            "AreaOfInterestGeometry": {
                "PolygonGeometry": {
                    "Coordinates": [
                        [
                            [-114.529, 36.142],
                            [-114.373, 36.142],
                            [-114.373, 36.411],
                            [-114.529, 36.411],
                            [-114.529, 36.142],
                        ]
                    ]
                }
            }
        },
        "TimeRangeFilter": {
            "StartTime": "2021-01-01T00:00:00Z",
            "EndTime": "2022-07-10T23:59:59Z",
        },
        "PropertyFilters": {
            "Properties": [{"Property": {"EoCloudCover": {"LowerBound": 0,
"UpperBound": 1}}}],
            "LogicalOperator": "AND",
        },
    }
}
eoj_config = {"LandCoverSegmentationConfig": {}}
```

```
response = sg_client.start_earth_observation_job(  
    Name="lake-mead-landcover",  
    InputConfig=eoj_input_config,  
    JobConfig=eoj_config,  
    ExecutionRoleArn=execution_role,  
)
```

After your EOJ is created, the Arn is returned to you. You use the Arn to identify a job and perform further operations. To get the status of a job, you can run `sg_client.get_earth_observation_job(Arn = response['Arn'])`.

The following example shows how to query the status of an EOJ until it is completed.

```
eoj_arn = response["Arn"]  
job_details = sg_client.get_earth_observation_job(Arn=eoj_arn)  
{k: v for k, v in job_details.items() if k in ["Arn", "Status", "DurationInSeconds"]}  
# List all jobs in the account  
sg_client.list_earth_observation_jobs()["EarthObservationJobSummaries"]
```

After the EOJ is completed, you can visualize the EOJ outputs directly in the notebook. The following example shows you how an interactive map can be rendered.

```
map = sagemaker_geospatial_map.create_map({  
    'is_raster': True  
})  
map.set_sagemaker_geospatial_client(sg_client)  
# render the map  
map.render()
```

The following example shows how the map can be centered on an area of interest and the input and output of the EOJ can be rendered as separate layers within the map.

```
# visualize the area of interest  
config = {"label": "Lake Mead AOI"}  
aoi_layer = map.visualize_eoj_aoi(Arn=eoj_arn, config=config)  
  
# Visualize input.  
time_range_filter = {  
    "start_date": "2022-07-01T00:00:00Z",  
    "end_date": "2022-07-10T23:59:59Z",  
}
```

```
config = {"label": "Input"}  
  
input_layer = map.visualize_eoj_input(  
    Arn=eoj_arn, config=config, time_range_filter=time_range_filter  
)  
# Visualize output, EOJ needs to be in completed status.  
time_range_filter = {  
    "start_date": "2022-07-01T00:00:00Z",  
    "end_date": "2022-07-10T23:59:59Z",  
}  
config = {"preset": "singleBand", "band_name": "mask"}  
output_layer = map.visualize_eoj_output(  
    Arn=eoj_arn, config=config, time_range_filter=time_range_filter  
)
```

You can use the `export_earth_observation_job` function to export the EOJ results to your Amazon S3 bucket. The export function makes it convenient to share results across teams. SageMaker AI also simplifies dataset management. We can simply share the EOJ results using the job ARN, instead of crawling thousands of files in the S3 bucket. Each EOJ becomes an asset in the data catalog, as results can be grouped by the job ARN. The following example shows how you can export the results of an EOJ.

```
sagemaker_session = sagemaker.Session()  
s3_bucket_name = sagemaker_session.default_bucket() # Replace with your own bucket if needed  
s3_bucket = session.resource("s3").Bucket(s3_bucket_name)  
prefix = "eoj_lakemead" # Replace with the S3 prefix desired  
export_bucket_and_key = f"s3://{s3_bucket_name}/{prefix}/"  
  
eoj_output_config = {"S3Data": {"S3Uri": export_bucket_and_key}}  
export_response = sg_client.export_earth_observation_job(  
    Arn=eoj_arn,  
    ExecutionRoleArn=execution_role,  
    OutputConfig=eoj_output_config,  
    ExportSourceImages=False,  
)
```

You can monitor the status of your export job using the following snippet.

```
# Monitor the export job status  
export_job_details = sg_client.get_earth_observation_job(Arn=export_response["Arn"])
```

```
{k: v for k, v in export_job_details.items() if k in ["Arn", "Status",  
"DurationInSeconds"]}
```

You are not charged the storage fees after you delete the EOJ.

For an example that showcases how to run an EOJ, see this [blog post](#).

For more example notebooks on SageMaker geospatial capabilities, see this [GitHub repository](#).

## Types of Operations

When you create an EOJ, you select an operation based on your use case. Amazon SageMaker geospatial capabilities provide a combination of purpose-built operations and pre-trained models. You can use these operations to understand the impact of environmental changes and human activities over time or identify cloud and cloud-free pixels.

### Cloud Masking

Identify clouds in satellite images is an essential pre-processing step in producing high-quality geospatial data. Ignoring cloud pixels can lead to errors in analysis, and over-detection of cloud pixels can decrease the number of valid observations. Cloud masking has the ability to identify cloudy and cloud-free pixels in satellite images. An accurate cloud mask helps get satellite images for processing and improves data generation. The following is the class map for cloud masking.

```
{  
0: "No_cloud",  
1: "cloud"  
}
```

### Cloud Removal

Cloud removal for Sentinel-2 data uses an ML-based semantic segmentation model to identify clouds in the image. Cloudy pixels can be replaced by pixels from other timestamps. USGS Landsat data contains landsat metadata that is used for cloud removal.

### Temporal Statistics

Temporal statistics calculate statistics for geospatial data through time. The temporal statistics currently supported include mean, median, and standard deviation. You can calculate these

statistics by using GROUPBY and set it to either all or yearly. You can also mention the TargetBands.

## Zonal Statistics

Zonal statistics performs statistical operations over a specified area on the image.

## Resampling

Resampling is used to upscale and downscale the resolution of a geospatial image. The value attribute in resampling represents the length of a side of the pixel.

## Geomosaic

Geomosaic allows you to stitch smaller images into a large image.

## Band Stacking

Band stacking takes more than one image band as input and stacks them into a single GeoTIFF. The OutputResolution attribute determines the resolution of the output image. Based on the resolutions of the input images, you can set it to lowest, highest or average.

## Band Math

Band Math, also known as Spectral Index, is a process of transforming the observations from multiple spectral bands to a single band, indicating the relative abundance of features of interests. For instance, Normalized Difference Vegetation Index (NDVI) and Enhanced Vegetation Index (EVI) are helpful for observing the presence of green vegetation features.

## Land Cover Segmentation

Land Cover segmentation is a semantic segmentation model that has the capability to identify the physical material, such as vegetation, water, and bare ground, at the earth surface. Having an accurate way to map the land cover patterns helps you understand the impact of environmental change and human activities over time. Land Cover segmentation is often used for region planning, disaster response, ecological management, and environmental impact assessment. The following is the class map for Land Cover segmentation.

```
{  
  0: "No_data",  
  1: "Saturated_or_defective",
```

```
2: "Dark_area_pixels",
3: "Cloud_shadows",
4: "Vegetation",
5: "Not_vegetated",
6: "Water",
7: "Unclassified",
8: "Cloud_medium_probability",
9: "Cloud_high_probability",
10: "Thin_cirrus",
11: "Snow_ice"
}
```

## Availability of EOJ Operations

The availability of operations depends on whether you are using the SageMaker geospatial UI or the Amazon SageMaker Studio Classic notebooks with a SageMaker geospatial image. Currently, notebooks support all functionalities. To summarize, the following geospatial operations are supported by SageMaker AI:

Operations	Description	Availability
Cloud Masking	Identify cloud and cloud-free pixels to get improved and accurate satellite imagery.	UI, Notebook
Cloud Removal	Remove pixels containing parts of a cloud from satellite imagery.	Notebook
Temporal Statistics	Calculate statistics through time for a given GeoTIFF.	Notebook
Zonal Statistics	Calculate statistics on user-defined regions.	Notebook
Resampling	Scale images to different resolutions.	Notebook
Geomosaic	Combine multiple images for greater fidelity.	Notebook

Operations	Description	Availability
Band Stacking	Combine multiple spectral bands to create a single image.	Notebook
Band Math / Spectral Index	Obtain a combination of spectral bands that indicate the abundance of features of interest.	UI, Notebook
Land Cover Segmentation	Identify land cover types such as vegetation and water in satellite imagery.	UI, Notebook

## Vector Enrichment Jobs

A Vector Enrichment Job (VEJ) performs operations on your vector data. Currently, you can use a VEJ to do reverse geocoding or map matching.

### Reverse Geocoding

With a reverse geocoding VEJ, you can convert geographic coordinates (latitude, longitude) to human-readable addresses powered by Amazon Location Service. When you upload a CSV file containing the longitude and latitude coordinates, it returns the address number, country, label, municipality, neighborhood, postal code and region of that location. The output file consists of your input data along with columns containing these values appended at the end. These jobs are optimized to accept tens of thousands of GPS traces.

### Map Matching

Map matching allows you to snap GPS coordinates to road segments. The input should be a CSV file containing the trace ID (route), longitude, latitude and the timestamp attributes. There can be multiple GPS co-ordinates per route. The input can contain multiple routes too. The output is a GeoJSON file that contains links of the predicted route. It also has the snap points provided in the input. These jobs are optimized to accept tens of thousands of drives in one request. Map matching is supported by [OpenStreetMap](#). Map matching fails if the names in the input source field don't match the ones in MapMatchingConfig. The error message you receive contains

the field names present in the input file and the expected field name that is not found in MapMatchingConfig.

The input CSV file for a VEJ must contain the following:

- A header row
- Latitude and longitude in separate columns
- The ID and Timestamp columns can be in numeric or string format. All other column data must be in numeric format only
- No miss matching quotes

For the timestamp column, SageMaker geospatial capabilities supports epoch time in seconds and milliseconds (long integer). The string formats supported are as follows:

- "dd.MM.yyyy HH:mm:ss z"
- "yyyy-MM-dd'T'HH:mm:ss.SSS'Z""
- "yyyy-MM-dd'T'HH:mm:ss"
- "yyyy-MM-dd hh:mm:ss a"
- "yyyy-MM-dd HH:mm:ss"
- "yyyyMMddHHmmss"

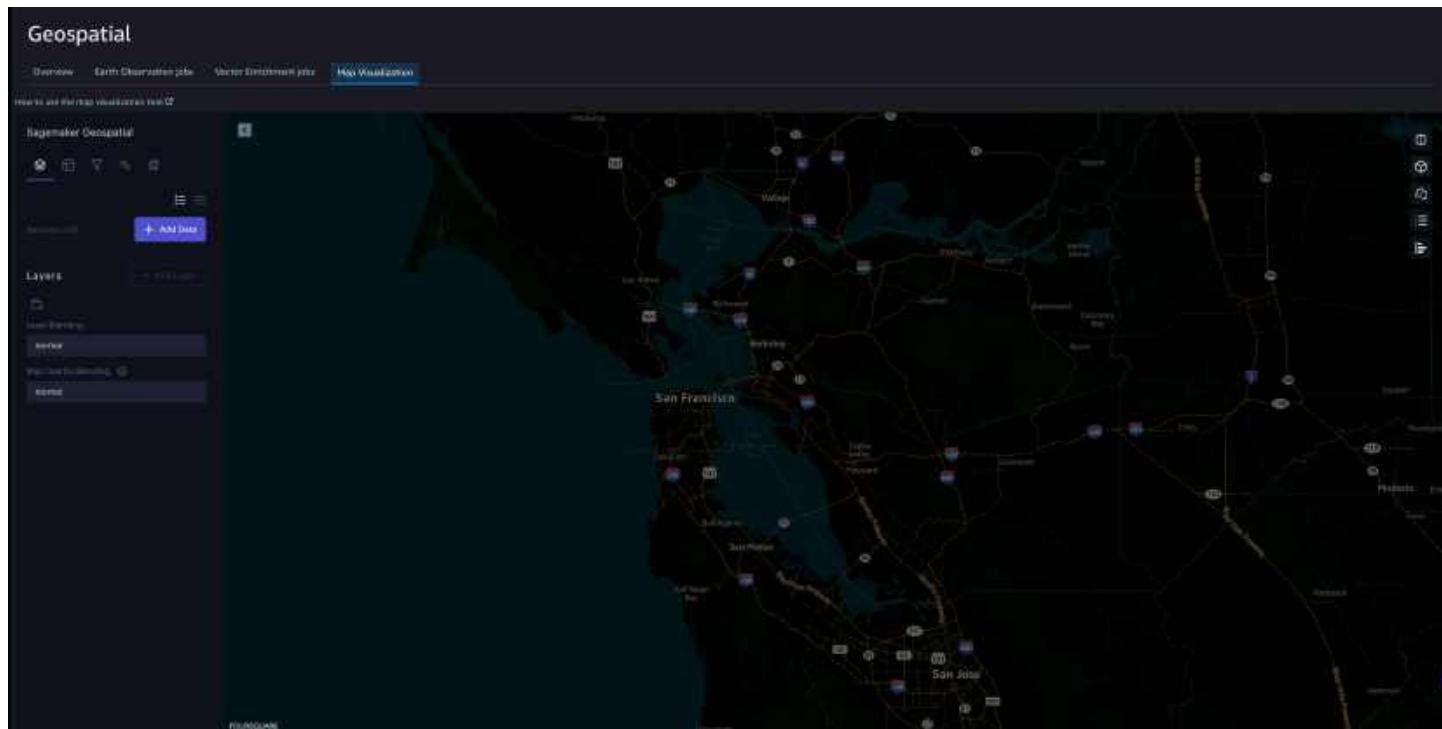
While you need to use an Amazon SageMaker Studio Classic notebook to execute a VEJ, you can view all the jobs you create using the UI. To use the visualization in the notebook, you first need to export your output to your S3 bucket. The VEJ actions you can perform are as follows.

- [StartVectorEnrichmentJob](#)
- [GetVectorEnrichmentJob](#)
- [ListVectorEnrichmentJobs](#)
- [StopVectorEnrichmentJob](#)
- [DeleteVectorEnrichmentJob](#)

## Visualization Using SageMaker geospatial capabilities

Using the visualization functionalities provided by Amazon SageMaker geospatial you can visualize geospatial data, the inputs to your EOJ or VEJ jobs as well as the outputs exported from your

Amazon S3 bucket. The visualization tool is powered by [Foursquare Studio](#). The following image depicts the visualization tool supported by SageMaker geospatial capabilities.



You can use the left navigation panel to add data, layers, filters, and columns. You can also make modifications to how you interact with the map.

## Dataset

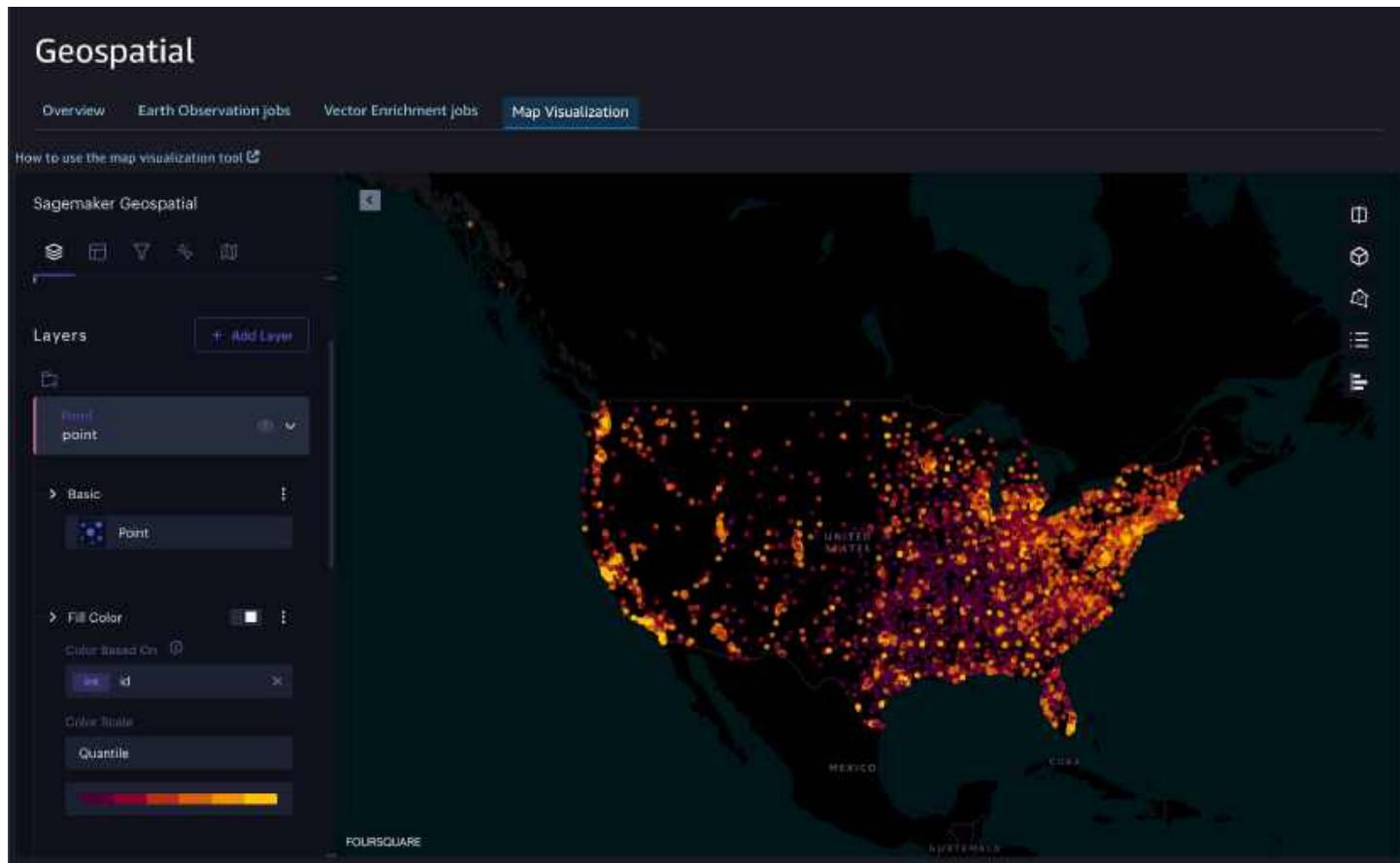
The source of data used for visualization is called a **Dataset**. To add data for visualization, choose **Add Data** in the left navigation panel. You can either upload the data from your Amazon S3 bucket or your local machine. The data formats supported are CSV, JSON and GeoJSON. You can add multiple datasets to your map. After you upload the dataset, you can see it loaded on the map screen.

## Layers

In the layer panel, a layer is created and populated automatically when you add a dataset. If your map consists of more than one dataset, you can select which dataset belongs to a layer. You can create new layers and group them. SageMaker geospatial capabilities support various layer types, including point, arc, icon, and polygon.

You can choose any data point in a layer to have an **Outline**. You can also further customize the data points. For example, you can choose the layer type as **Point** and then **Fill Color** based on any column of your dataset. You can also change the radius of the points.

The following image shows the layers panel supported by SageMaker geospatial capabilities.



## Columns

You can view the columns present in your dataset by using the **Columns** tab in the left navigation panel.

## Filters

You can use filters to limit the data points that display on the map.

## Interactions

In the **Interactions** panel, you can customize how you interact with the map. For example, you can choose what metrics to display when you hover the tooltip over a data point.

## Base map

Currently, SageMaker AI only supports the Amazon Dark base map.

## Split Map Modes

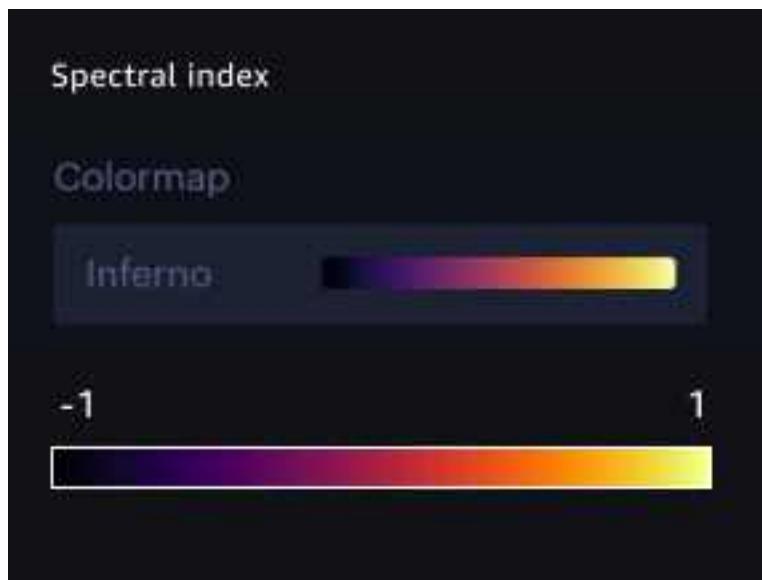
You can have a **Single Map**, **Dual Maps** or **Swipe Maps**. With **Dual Maps**, you can compare the same map side-by-side using different layers. Use **Swipe Maps** to overlay two maps on each other and use the sliding separator to compare them. You can choose the split map mode by choosing the **Split Mode** button on the top right corner of your map.

## Legends for EOJ in the SageMaker geospatial UI

The output visualization of an EOJ depends on the operation you choose to create it. The legend is based on the default color scale. You can view the legend by choosing the **Show legend** button on the top right corner of your map.

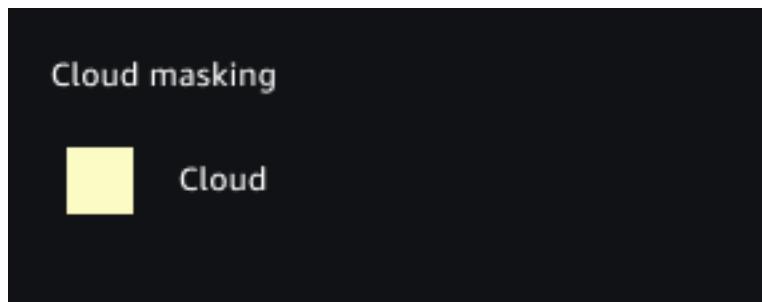
### Spectral Index

When you visualize the output for an EOJ that uses the spectral index operation, you can map the category based on the color from the legend as shown.



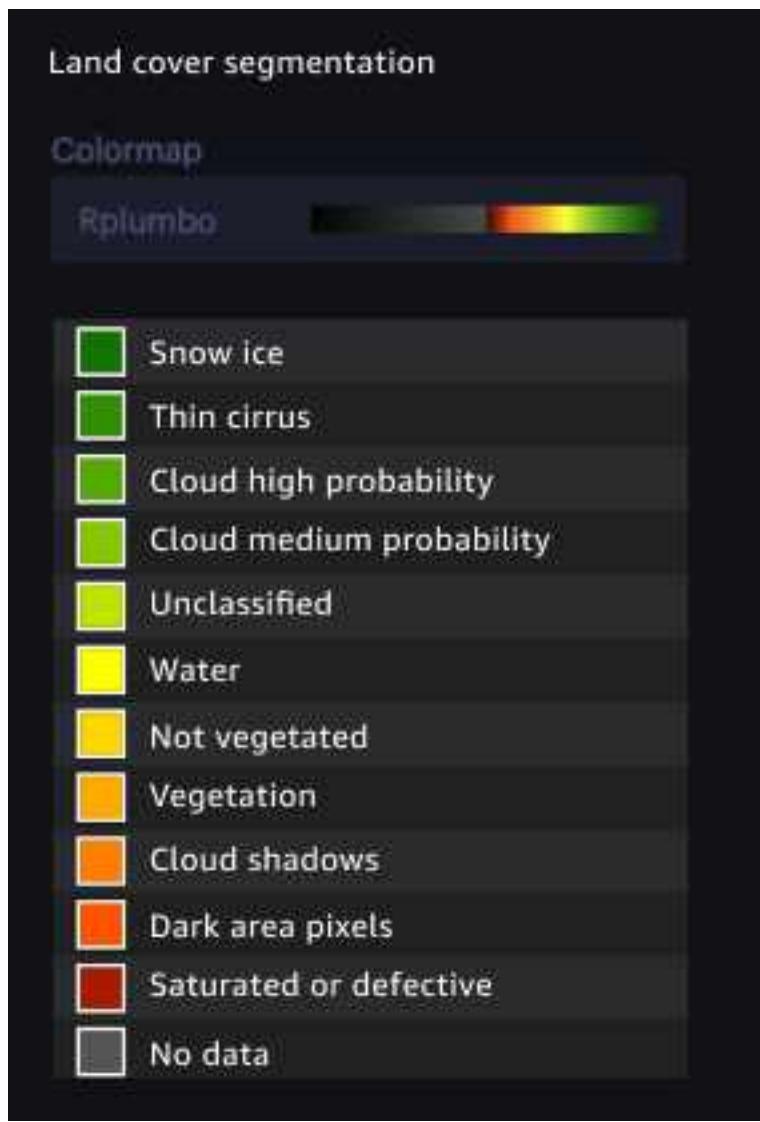
### Cloud Masking

When you visualize the output for an EOJ that uses the cloud masking operation, you can map the category based on the color from the legend as shown.



## Land Cover Segmentation

When you visualize the output for an EOJ that uses the Land Cover Segmentation operation, you can map the category based on the color from the legend as shown.



## Amazon SageMaker geospatial Map SDK

You can use Amazon SageMaker geospatial capabilities to visualize maps within the SageMaker geospatial UI as well as SageMaker notebooks with a geospatial image. These visualizations are supported by the map visualization library called [Foursquare Studio](#).

You can use the APIs provided by the SageMaker geospatial map SDK to visualize your geospatial data, including the input, output, and Aoi for EOJ.

### Topics

- [add\\_dataset API](#)
- [update\\_dataset API](#)
- [add\\_layer API](#)
- [update\\_layer API](#)
- [visualize\\_eoj\\_aoi API](#)
- [visualize\\_eoj\\_input API](#)
- [visualize\\_eoj\\_output API](#)

## add\_dataset API

Adds a raster or vector dataset object to the map.

### Request syntax

```
Request =  
    add_dataset(  
        self,  
        dataset: Union[Dataset, Dict, None] = None,  
        *,  
        auto_create_layers: bool = True,  
        center_map: bool = True,  
        **kwargs: Any,  
    ) -> Optional[Dataset]
```

### Request parameters

The request accepts the following parameters.

#### Positional arguments

Argument	Type	Description
dataset	Union[Dataset, Dict, None]	Data used to create a dataset, in CSV, JSON, or GeoJSON format (for local datasets) or a UUID string.

## Keyword arguments

Argument	Type	Description
auto_create_layers	Boolean	Whether to attempt to create new layers when adding a dataset. Default value is False.
center_map	Boolean	Whether to center the map on the created dataset. Default value is True.
id	String	Unique identifier of the dataset. If you do not provide it, a random ID is generated.
label	String	Dataset label which is displayed.
color	Tuple[float, float, float]	Color label of the dataset.
metadata	Dictionary	Object containing tileset metadata (for tiled datasets).

## Response

This API returns the [Dataset](#) object that was added to the map.

## update\_dataset API

Updates an existing dataset's settings.

### Request syntax

```
Request =
    update_dataset(
        self,
        dataset_id: str,
        values: Union[_DatasetUpdateProps, dict, None] = None,
```

```
    **kwargs: Any,  
) -> Dataset
```

## Request parameters

The request accepts the following parameters.

### Positional arguments

Argument	Type	Description
dataset_id	String	The identifier of the dataset to be updated.
values	Union[ <a href="#">DatasetUpdateProps</a> , dict, None]	The values to update.

### Keyword arguments

Argument	Type	Description
label	String	Dataset label which is displayed.
color	<a href="#">RGBColor</a>	Color label of the dataset.

## Response

This API returns the updated dataset object for interactive maps, or None for non-interactive HTML environments.

## add\_layer API

Adds a new layer to the map. This function requires at least one valid layer configuration.

### Request syntax

```
Request =
```

```
add_layer(  
    self,  
    layer: Union[LayerCreationProps, dict, None] = None,  
    **kwargs: Any  
) -> Layer
```

## Request parameters

The request accepts the following parameters.

### Arguments

Argument	Type	Description
layer	Union[ <a href="#">LayerCreationProps</a> , dict, None]	A set of properties used to create a layer.

## Response

The layer object that was added to the map.

## update\_layer API

Update an existing layer with given values.

### Request syntax

```
Request =  
    update_layer(  
        self,  
        layer_id: str,  
        values: Union[LayerUpdateProps, dict, None],  
        **kwargs: Any  
) -> Layer
```

## Request parameters

The request accepts the following parameters.

### Arguments

Positional argument	Type	Description
layer_id	String	The ID of the layer to be updated.
values	Union[ <a href="#">LayerUpdateProps</a> , dict, None]	The values to update.

## Keyword arguments

Argument	Type	Description
type	<a href="#">LayerType</a>	The type of layer.
data_id	String	Unique identifier of the dataset this layer visualizes.
fields	Dict [string, Optional[string]]	Dictionary that maps fields that the layer requires for visualization to appropriate dataset fields.
label	String	Canonical label of this layer.
is_visible	Boolean	Whether the layer is visible or not.
config	<a href="#">LayerConfig</a>	Layer configuration specific to its type.

## Response

Returns the updated layer object.

## visualize\_eoj\_aoi API

Visualize the AoI of the given job ARN.

## Request parameters

The request accepts the following parameters.

### Arguments

Argument	Type	Description
Arn	String	The ARN of the job.
config	Dictionary  config = { label: <string> custom label of the added AOL layer, default AOL }	An option to pass layer properties.

## Response

Reference of the added input layer object.

## visualize\_eoj\_input API

Visualize the input of the given EOJ ARN.

### Request parameters

The request accepts the following parameters.

### Arguments

Argument	Type	Description
Arn	String	The ARN of the job.
time_range_filter	Dictionary  time_range_filter = {  start_date: <string> date in ISO format	An option to provide the start and end time. Defaults to the raster data collection search start and end date.

Argument	Type	Description
	end_date: <string> date in ISO format }	
config	Dictionary config = { label: <string> custom label of the added output layer, default Input }	An option to pass layer properties.

## Response

Reference of the added input layer object.

## visualize\_eoj\_output API

Visualize the output of the given EOJ ARN.

### Request parameters

The request accepts the following parameters.

#### Arguments

Argument	Type	Description
Arn	String	The ARN of the job.
time_range_filter	Dictionary time_range_filter = { start_date: <string> date in ISO format end_date: <string> date in ISO format	An option to provide the start and end time. Defaults to the raster data collection search start and end date.

Argument	Type	Description
	}	
config	Dictionary  config = {  label: <string> custom label of the added output layer, default Output  preset: <string> singleBand or trueColor,  band_name: <string>, only required for 'singleBand' preset. Allowed bands for a EOJ  }	An option to pass layer properties.

## Response

Reference of the added output Layer object.

To learn more about visualizing your geospatial data, refer to [Visualization Using Amazon SageMaker geospatial](#).

## SageMaker geospatial capabilities FAQ

Use the following FAQ items to find answers to commonly asked questions about SageMaker geospatial capabilities.

### 1. What regions are Amazon SageMaker geospatial capabilities available in?

Currently, SageMaker geospatial capabilities are only supported in the US West (Oregon) Region. To view SageMaker geospatial, choose the name of the currently displayed Region in the navigation bar of the console. Then choose the US West (Oregon) Region.

## 2. What AWS Identity and Access Management permissions and policies are required to use SageMaker geospatial?

To use SageMaker geospatial you need a user, group, or role that can access SageMaker AI. You also need to create a SageMaker AI execution role so that SageMaker geospatial can perform operations on your behalf. To learn more, see [SageMaker geospatial capabilities roles](#).

## 3. I have an existing SageMaker AI execution role. Do I need to update it?

Yes. To use SageMaker geospatial you must specify an additional service principal in your IAM trust policy: `sagemaker-geospatial.amazonaws.com`. To learn about specifying a service principal in a trust relationship, see [Adding the SageMaker geospatial service principal to an existing SageMaker AI execution role](#) in the *Amazon SageMaker AI Developer Guide*.

## 4. Can I use SageMaker geospatial capabilities through my VPC environment?

Yes, you can use SageMaker geospatial through a VPN. To learn more, see [Use Amazon SageMaker geospatial capabilities in Your Amazon Virtual Private Cloud](#).

## 5. Why can't I see the SageMaker geospatial map visualizer, image or instance type when I navigate to Amazon SageMaker Studio Classic?

Verify that you are launching Amazon SageMaker Studio Classic in the US West (Oregon) Region and that you are not using a shared space.

## 6. Why can't I see the SageMaker geospatial image or instance type when I try to create a notebook instance in Studio Classic?

Verify that you are launching Amazon SageMaker Studio Classic in the US West (Oregon) Region and that you are not using a shared space. To learn more, see [Create an Amazon SageMaker Studio Classic notebook using the geospatial image](#).

## 7. What bands supported for various raster data collections?

Use the `GetRasterDataCollection` API response and refer to the `ImageSourceBands` field to find the bands supported for that particular data collection.

# SageMaker geospatial Security and Permissions

Use the topics on this page to learn about SageMaker geospatial capabilities security features. Additionally, learn how to use SageMaker geospatial capabilities in an Amazon Virtual Private Cloud as well as protect your data at rest using encryption.

For more information about IAM users and roles, see [Identities \(Users, Groups, and Roles\)](#) in the IAM User Guide.

To learn more about using IAM with SageMaker AI, see [AWS Identity and Access Management for Amazon SageMaker AI](#).

## Topics

- [Configuration and Vulnerability Analysis in SageMaker geospatial](#)
- [Security Best Practices for SageMaker geospatial capabilities](#)
- [Use Amazon SageMaker geospatial capabilities in Your Amazon Virtual Private Cloud](#)
- [Use AWS KMS Permissions for Amazon SageMaker geospatial capabilities](#)

## Configuration and Vulnerability Analysis in SageMaker geospatial

Configuration and IT controls are a shared responsibility between AWS and you, our customer. AWS handles basic security tasks like guest operating system (OS) and database patching, firewall configuration, and disaster recovery. These procedures have been reviewed and certified by the appropriate third parties. For more details, see the following resources:

- [Shared Responsibility Model](#).
- [Amazon Web Services: Overview of Security Processes](#).

## Security Best Practices for SageMaker geospatial capabilities

Amazon SageMaker geospatial capabilities provide a number of security features to consider as you develop and implement your own security policies. The following best practices are general guidelines and don't represent a complete security solution. Because these best practices might not be appropriate or sufficient for your environment, treat them as helpful considerations rather than prescriptions.

### Apply principle of least privilege

Amazon SageMaker geospatial capabilities provide granular access policy for applications using IAM roles. We recommend that the roles be granted only the minimum set of privileges required by the job. We also recommend auditing the jobs for permissions on a regular basis and upon any change to your application.

### Role-based access control (RBAC) permissions

Administrators should strictly control Role-based access control (RBAC) permissions for Amazon SageMaker geospatial capabilities.

## Use temporary credentials whenever possible

Where possible, use temporary credentials instead of long-term credentials, such as access keys. For scenarios in which you need IAM users with programmatic access and long-term credentials, we recommend that you rotate access keys. Regularly rotating long-term credentials helps you familiarize yourself with the process. This is useful in case you are ever in a situation where you must rotate credentials, such as when an employee leaves your company. We recommend that you use IAM access last used information to rotate and remove access keys safely. For more information, see [Rotating access keys](#) and [Security best practices in IAM](#).

## Use AWS CloudTrail to view and log API calls

AWS CloudTrail tracks anyone making API calls in your AWS account. API calls are logged whenever anyone uses the Amazon SageMaker geospatial capabilities API, the Amazon SageMaker geospatial capabilities console or Amazon SageMaker geospatial capabilities AWS CLI commands. Enable logging and specify an Amazon S3 bucket to store the logs.

Your trust, privacy, and the security of your content are our highest priorities. We implement responsible and sophisticated technical and physical controls designed to prevent unauthorized access to, or disclosure of, your content and ensure that our use complies with our commitments to you. For more information, see [AWS Data Privacy FAQ](#).

## Use Amazon SageMaker geospatial capabilities in Your Amazon Virtual Private Cloud

The following topic gives information on how to use SageMaker notebooks with a SageMaker geospatial image in a Amazon SageMaker AI domain with VPC only mode. For more information on VPCs in Amazon SageMaker Studio Classic see [Choose an Amazon VPC](#).

### VPC only communication with the internet

By default, SageMaker AI domain uses two Amazon VPC. One of the Amazon VPC is managed by Amazon SageMaker AI and provides direct internet access. You specify the other Amazon VPC, which provides encrypted traffic between the domain and your Amazon Elastic File System (Amazon EFS) volume.

You can change this behavior so that SageMaker AI sends all traffic over your specified Amazon VPC. If VPC only has been chosen as the network access mode during the SageMaker AI domain

creation, the following requirements need to be considered to still allow usage of SageMaker Studio Classic notebooks within the created SageMaker AI domain.

## Requirements to use VPC only mode

### Note

In order to use the visualization components of SageMaker geospatial capabilities, the browser you use to access the SageMaker Studio Classic UI needs to be connected to the internet.

When you choose `VpcOnly`, follow these steps:

1. You must use private subnets only. You cannot use public subnets in `VpcOnly` mode.
2. Ensure your subnets have the required number of IP addresses needed. The expected number of IP addresses needed per user can vary based on use case. We recommend between 2 and 4 IP addresses per user. The total IP address capacity for a Studio Classic domain is the sum of available IP addresses for each subnet provided when the domain is created. Ensure that your estimated IP address usage does not exceed the capacity supported by the number of subnets you provide. Additionally, using subnets distributed across many availability zones can aid in IP address availability. For more information, see [VPC and subnet sizing for IPv4](#).

### Note

You can configure only subnets with a default tenancy VPC in which your instance runs on shared hardware. For more information on the tenancy attribute for VPCs, see [Dedicated Instances](#).

3. Set up one or more security groups with inbound and outbound rules that together allow the following traffic:
  - [NFS traffic over TCP on port 2049](#) between the domain and the Amazon EFS volume.
  - [TCP traffic within the security group](#). This is required for connectivity between the JupyterServer app and the KernelGateway apps. You must allow access to at least ports in the range 8192–65535.
4. If you want to allow internet access, you must use a [NAT gateway](#) with access to the internet, for example through an [internet gateway](#).

5. If you don't want to allow internet access, [create interface VPC endpoints](#) (AWS PrivateLink) to allow Studio Classic to access the following services with the corresponding service names. You must also associate the security groups for your VPC with these endpoints.

 **Note**

Currently, SageMaker geospatial capabilities are only supported in the US West (Oregon) Region.

- SageMaker API : com.amazonaws.us-west-2.sagemaker.api
- SageMaker AI runtime: com.amazonaws.us-west-2.sagemaker.runtime. This is required to run Studio Classic notebooks with a SageMaker geospatial image.
- Amazon S3: com.amazonaws.us-west-2.s3.
- To use SageMaker Projects: com.amazonaws.us-west-2.servicecatalog.
- SageMaker geospatial capabilities: com.amazonaws.us-west-2.sagemaker-geospatial

If you use the [SageMaker Python SDK](#) to run remote training jobs, you must also create the following Amazon VPC endpoints.

- AWS Security Token Service: com.amazonaws.*region*.sts
- Amazon CloudWatch: com.amazonaws.*region*.logs. This is required to allow SageMaker Python SDK to get the remote training job status from Amazon CloudWatch.

 **Note**

For a customer working within VPC mode, company firewalls can cause connection issues with SageMaker Studio Classic or between JupyterServer and the KernelGateway. Make the following checks if you encounter one of these issues when using SageMaker Studio Classic from behind a firewall.

- Check that the Studio Classic URL is in your networks allowlist.
- Check that the websocket connections are not blocked. Jupyter uses websocket under the hood. If the KernelGateway application is InService, JupyterServer may not be able

to connect to the KernelGateway. You should see this problem when opening System Terminal as well.

## Use AWS KMS Permissions for Amazon SageMaker geospatial capabilities

You can protect your data at rest using encryption for SageMaker geospatial capabilities. By default, it uses server-side encryption with an Amazon SageMaker geospatial owned key. SageMaker geospatial capabilities also supports an option for server-side encryption with a customer managed KMS key.

### Server-Side Encryption with Amazon SageMaker geospatial managed key (Default)

SageMaker geospatial capabilities encrypts all your data, including computational results from your Earth Observation jobs (EOJ) and Vector Enrichment jobs (VEJ) along with all your service metadata. There is no data that is stored within SageMaker geospatial capabilities unencrypted. It uses a default AWS owned key to encrypt all your data.

### Server-Side Encryption with customer managed KMS key (Optional)

SageMaker geospatial capabilities supports the use of a symmetric customer managed key that you create, own, and manage to add a second layer of encryption over the existing AWS owned encryption. Because you have full control of this layer of encryption, you can perform such tasks as:

- Establishing and maintaining key policies
- Establishing and maintaining IAM policies and grants
- Enabling and disabling key policies
- Rotating key cryptographic material
- Adding tags
- Creating key aliases
- Scheduling keys for deletion

For more information, see [Customer managed keys](#) in the *AWS Key Management Service Developer Guide*.

## How SageMaker geospatial capabilities uses grants in AWS KMS

SageMaker geospatial capabilities requires a grant to use your customer managed key. When you create an EOJ or an VEJ encrypted with a customer managed key, SageMaker geospatial capabilities creates a grant on your behalf by sending a CreateGrant request to AWS KMS. Grants in AWS KMS are used to give SageMaker geospatial capabilities access to a KMS key in a customer account. You can revoke access to the grant, or remove the service's access to the customer managed key at any time. If you do, SageMaker geospatial capabilities won't be able to access any of the data encrypted by the customer managed key, which affects operations that are dependent on that data.

### Create a customer managed key

You can create a symmetric customer managed key by using the AWS Management Console, or the AWS KMS APIs.

#### To create a symmetric customer managed key

Follow the steps for [Creating symmetric encryption KMS keys](#) in the AWS Key Management Service Developer Guide.

### Key policy

Key policies control access to your customer managed key. Every customer managed key must have exactly one key policy, which contains statements that determine who can use the key and how they can use it. When you create your customer managed key, you can specify a key policy. For more information, see [Determining access to AWS KMS keys](#) in the *AWS Key Management Service Developer Guide*.

To use your customer managed key with your SageMaker geospatial capabilities resources, the following API operations must be permitted in the key policy. The principal for these operations should be the Execution Role you provide in the SageMaker geospatial capabilities request. SageMaker geospatial capabilities assumes the provided Execution Role in the request to perform these KMS operations.

- [kms:CreateGrant](#)
- kms:GenerateDataKey
- kms:Decrypt
- kms:GenerateDataKeyWithoutPlaintext

The following are policy statement examples you can add for SageMaker geospatial capabilities:

## CreateGrant

```
"Statement" : [
  {
    "Sid" : "Allow access to Amazon SageMaker geospatial capabilities",
    "Effect" : "Allow",
    "Principal" : {
      "AWS" : "<Customer provided Execution Role ARN>"
    },
    "Action" : [
      "kms:CreateGrant",
      "kms:Decrypt",
      "kms:GenerateDataKey",
      "kms:GenerateDataKeyWithoutPlaintext"
    ],
    "Resource" : "*",
  },
]
```

For more information about specifying permissions in a policy, see [AWS KMS permissions](#) in the *AWS Key Management Service Developer Guide*. For more information about troubleshooting, see [Troubleshooting key access](#) in the *AWS Key Management Service Developer Guide*.

If your key policy does not have your account root as key administrator, you need to add the same KMS permissions on your execution role ARN. Here is a sample policy you can add to the execution role:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "kms:CreateGrant",
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyWithoutPlaintext"
      ],
      "Resource": [
        "<KMS key Arn>"
      ],
    }
  ]
}
```

```
        "Effect": "Allow"
    }
]
}
```

## Monitoring your encryption keys for SageMaker geospatial capabilities

When you use an AWS KMS customer managed key with your SageMaker geospatial capabilities resources, you can use AWS CloudTrail or Amazon CloudWatch Logs to track requests that SageMaker geospatial sends to AWS KMS.

Select a tab in the following table to see examples of AWS CloudTrail events to monitor KMS operations called by SageMaker geospatial capabilities to access data encrypted by your customer managed key.

### CreateGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:SageMaker-Geospatial-StartEOJ-KMSAccess",
        "arn": "arn:aws:sts::111122223333:assumed-role/SageMakerGeospatialCustomerRole/SageMaker-Geospatial-StartEOJ-KMSAccess",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAIOSFODNN7EXAMPLE3",
                "arn": "arn:aws:sts::111122223333:assumed-role/SageMakerGeospatialCustomerRole",
                "accountId": "111122223333",
                "userName": "SageMakerGeospatialCustomerRole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-03-17T18:02:06Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "arn:aws:iam::111122223333:root"
    }
}
```

```
},
  "eventTime": "2023-03-17T18:02:06Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "CreateGrant",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "172.12.34.56",
  "userAgent": "ExampleDesktop/1.0 (V1; OS)",
  "requestParameters": {
    "retiringPrincipal": "sagemaker-geospatial.us-west-2.amazonaws.com",
    "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
    "operations": [
      "Decrypt"
    ],
    "granteePrincipal": "sagemaker-geospatial.us-west-2.amazonaws.com"
  },
  "responseElements": {
    "grantId":
"0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
    "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
  },
  "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
  "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
  "readOnly": false,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}
```

## GenerateDataKey

```
{
  "eventVersion": "1.08",
```

```
"userIdentity": {  
    "type": "AWSService",  
    "invokedBy": "sagemaker-geospatial.amazonaws.com"  
},  
"eventTime": "2023-03-24T00:29:45Z",  
"eventSource": "kms.amazonaws.com",  
"eventName": "GenerateDataKey",  
"awsRegion": "us-west-2",  
"sourceIPAddress": "sagemaker-geospatial.amazonaws.com",  
"userAgent": "sagemaker-geospatial.amazonaws.com",  
"requestParameters": {  
    "encryptionContext": {  
        "aws:s3:arn": "arn:aws:s3:::axis-earth-observation-  
job-378778860802/111122223333/numpyintp64/output/  
consolidated/32PPR/2022-01-04T09:58:03Z/S2B_32PPR_20220104_0_L2A_msavi.tif"  
    },  
    "keyId": "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",  
    "keySpec": "AES_256"  
},  
"responseElements": null,  
"requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",  
"eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",  
"readOnly": true,  
"resources": [  
    {  
        "accountId": "111122223333",  
        "type": "AWS::KMS::Key",  
        "ARN": "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"  
    }  
],  
"eventType": "AwsApiCall",  
"managementEvent": true,  
"recipientAccountId": "111122223333",  
"eventCategory": "Management"  
}
```

## Decrypt

```
{  
    "eventVersion": "1.08",  
    "userIdentity": {
```

```
        "type": "AWSService",
        "invokedBy": "sagemaker-geospatial.amazonaws.com"
    },
    "eventTime": "2023-03-28T22:04:24Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "sagemaker-geospatial.amazonaws.com",
    "userAgent": "sagemaker-geospatial.amazonaws.com",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "aws:s3:arn": "arn:aws:s3:::axis-earth-observation-
job-378778860802/111122223333/numpyeintp64/output/
consolidated/32PPR/2022-01-04T09:58:03Z/S2B_32PPR_20220104_0_L2A_msavi.tif"
        },
        "responseElements": null,
        "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
        "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
        "readOnly": true,
        "resources": [
            {
                "accountId": "111122223333",
                "type": "AWS::KMS::Key",
                "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
            }
        ],
        "eventType": "AwsApiCall",
        "managementEvent": true,
        "recipientAccountId": "111122223333",
        "eventCategory": "Management"
    }
}
```

## GenerateDataKeyWithoutPlainText

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:SageMaker-Geospatial-StartE0J-
KMSAccess",
```

```
        "arn": "arn:aws:sts::111122223333:assumed-role/  
SageMakerGeospatialCustomerRole/SageMaker-Geospatial-StartE0J-KMSAccess",  
        "accountId": "111122223333",  
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",  
        "sessionContext": {  
            "sessionIssuer": {  
                "type": "Role",  
                "principalId": "AKIAIOSFODNN7EXAMPLE3",  
                "arn": "arn:aws:sts::111122223333:assumed-role/  
SageMakerGeospatialCustomerRole",  
                "accountId": "111122223333",  
                "userName": "SageMakerGeospatialCustomerRole"  
            },  
            "webIdFederationData": {},  
            "attributes": {  
                "creationDate": "2023-03-17T18:02:06Z",  
                "mfaAuthenticated": "false"  
            }  
        },  
        "invokedBy": "arn:aws:iam::111122223333:root"  
    },  
    "eventTime": "2023-03-28T22:09:16Z",  
    "eventSource": "kms.amazonaws.com",  
    "eventName": "GenerateDataKeyWithoutPlaintext",  
    "awsRegion": "us-west-2",  
    "sourceIPAddress": "172.12.34.56",  
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",  
    "requestParameters": {  
        "keySpec": "AES_256",  
        "keyId": "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"  
    },  
    "responseElements": null,  
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",  
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",  
    "readOnly": true,  
    "resources": [  
        {  
            "accountId": "111122223333",  
            "type": "AWS::KMS::Key",  
            "ARN": "arn:aws:kms:us-  
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"  
        }  
    ],
```

```
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111122223333",
"eventCategory": "Management"
}
```

## Types of compute instances

SageMaker geospatial capabilities offer three types of compute instances.

- **SageMaker Studio Classic geospatial notebook instances** – SageMaker geospatial supports both CPU and GPU-based notebook instances in Studio Classic. Notebook instances are used to build, train, and deploy ML models. For a list of available notebook instance types that work with the geospatial image, see [Supported notebook instance types](#).
- **SageMaker geospatial jobs instances** – Run processing jobs to transform satellite image data.
- **SageMaker geospatial model inference types** – Make predictions by using pre-trained ML models on satellite imagery.

The instance type is determined by the operations that you run.

The following table shows the available SageMaker geospatial specific operations and instance types that you can use.

Operations	Instance
Temporal Statistics	ml.geospatial.jobs
Zonal Statistics	ml.geospatial.jobs
Resampling	ml.geospatial.jobs
Geomosaic	ml.geospatial.jobs
Band Stacking	ml.geospatial.jobs
Band Math	ml.geospatial.jobs
Cloud Removal with Landsat8	ml.geospatial.jobs

Operations	Instance
Cloud Removal with Sentinel-2	ml.geospatial.models
Cloud Masking	ml.geospatial.models
Land Cover Segmentation	ml.geospatial.models

## SageMaker geospatial supported notebook instance types

SageMaker geospatial supports both CPU and GPU-based notebook instances in Studio Classic. If when starting a GPU enabled notebook instance you receive a ResourceLimitExceeded error, you need to request a quota increase. To get started on a Service Quotas quota increase request, see [Requesting a quota increase](#) in the *Service Quotas User Guide*.

Supported Studio Classic notebook instance types

Name	Instance type
ml.geospatial.interactive	CPU
ml.g5.xlarge	GPU
ml.g5.2xlarge	GPU
ml.g5.4xlarge	GPU
ml.g5.8xlarge	GPU
ml.g5.16xlarge	GPU
ml.g5.12xlarge	GPU
ml.g5.24xlarge	GPU
ml.g5.48xlarge	GPU

You are charged different rates for each type of compute instance that you use. For more information about pricing, see [Geospatial ML with Amazon SageMaker AI](#).

## SageMaker geospatial libraries

The SageMaker geospatial specific **Instance type, ml.geospatial.interactive** contains the following Python libraries.

Geospatial libraries available on the geospatial instance type

Library name	Version available
numpy	1.23.4
scipy	1.11.2
pandas	1.4.4
gdal	3.2.2
fiona	1.8.22
geopandas	0.11.1
shapley	1.8.4
seaborn	0.11.2
notebook	1.8.22
scikit-image	0.11.2
rasterio	6.4.12
scikit-learn	0.19.2
ipyleaflet	1.0.1
rtree	0.17.2
opencv	4.6.0.66
supy	2022.4.7
SNAP toolbox	9.0

Library name	Version available
cdsapi	0.6.1
arosics	1.8.1
rasterstats	0.18.0
rioxarray	0.14.1
pyroSAR	0.20.0
eo-learn	1.4.1
deepforest	1.2.7
scrapy	2.8.0
netCDF4	1.6.3
xarray[complete]	0.20.1
Orfeotoolbox	OTB-8.1.1
pytorch	2.0.1
pytorch-cuda	11.8
torchvision	0.15.2
torchaudio	2.0.2
pytorch-lightning	2.0.6
tensorflow	2.13.0

## Data collections

Amazon SageMaker geospatial supports the following raster data collections. Of the following data collections, you can use the USGS Landsat and the Sentinel-2 Cloud-Optimized GeoTIFF data

collections when starting an Earth Observation Job (EOJ). To learn more about the EOJs, see [Earth Observation Jobs](#).

- [Copernicus Digital Elevation Model \(DEM\) – GLO-30](#)
- [Copernicus Digital Elevation Model \(DEM\) – GLO-90](#)
- [Sentinel-2 Cloud-Optimized GeoTIFFs](#)
- [Sentinel-1](#)
- [National Agriculture Imagery Program \(NAIP\) on AWS](#)
- [USGS Landsat 8](#)

To find the list of available raster data collections in your AWS Regions, use `ListRasterDataCollections`. In the [ListRasterDataCollections response](#), you get a [RasterDataCollectionMetadata](#) object that contains details about the available raster data collections.

### **Example Example – Calling the `ListRasterDataCollections` API using the AWS SDK for Python (Boto3)**

When you use the SDK for Python (Boto3) and SageMaker geospatial, you must create a geospatial client, `geospatial_client`. Use the following Python snippet to make a call to the `list_raster_data_collections` API:

```
import boto3
import sagemaker
import sagemaker_geospatial_map
import json

## SageMaker Geospatial Capabilities is currently only available in US-WEST-2
session = boto3.Session(region_name='us-west-2')
execution_role = sagemaker.get_execution_role()

## Creates a SageMaker Geospatial client instance
geospatial_client = session.client(service_name="sagemaker-geospatial")

# Creates a reusable Paginator for the list_raster_data_collections API operation
paginator = geospatial_client.getPaginator("list_raster_data_collections")

# Create a PageIterator from the Paginator
page_iterator = paginator.paginate()
```

```
# Use the iterator to iterate through the results of list_raster_data_collections
results = []
for page in page_iterator:
    results.append(page['RasterDataCollectionSummaries'])

print (results)
```

In the JSON response, you will receive the following, which has been truncated for clarity:

```
{
    "Arn": "arn:aws:sagemaker-geospatial:us-west-2:555555555555:raster-data-collection/public/dxxbpqwv9041ny8",
    "Description": "Copernicus DEM is a Digital Surface Model which represents the surface of the Earth including buildings, infrastructure, and vegetation. GLO-30 is instance of Copernicus DEM that provides limited worldwide coverage at 30 meters.",
    "DescriptionPageUrl": "https://registry.opendata.aws/copernicus-dem/",
    "Name": "Copernicus DEM GLO-30",
    "Tags": {},
    "Type": "PUBLIC"
}
```

## Image band information from the USGS Landsat and Sentinel-2 data collections

Image band information from the USGS Landsat 8 and Sentinel-2 data collections are provided in the following table.

### USGS Landsat

Band name	Wave length range (nm)	Units	Valid range	Fill value	Spatial resolution
coastal	435 - 451	Unitless	1 - 65455	0 (No Data)	30m
blue	452 - 512	Unitless	1 - 65455	0 (No Data)	30m
green	533 - 590	Unitless	1 - 65455	0 (No Data)	30m
red	636 - 673	Unitless	1 - 65455	0 (No Data)	30m
nir	851 - 879	Unitless	1 - 65455	0 (No Data)	30m

<b>Band name</b>	<b>Wave length range (nm)</b>	<b>Units</b>	<b>Valid range</b>	<b>Fill value</b>	<b>Spatial resolution</b>
swir16	1566 - 1651	Unitless	1 - 65455	0 (No Data)	30m
swir22	2107 - 2294	Unitless	1 - 65455	0 (No Data)	30m
qa_aerosol	NA	Bit Index	0 - 255	1	30m
qa_pixel	NA	Bit Index	1 - 65455	1 (bit 0)	30m
qa_radsat	NA	Bit Index	1 - 65455	NA	30m
t	10600 - 11190	Scaled Kelvin	1 - 65455	0 (No Data)	30m (scaled from 100m)
atran	NA	Unitless	0 - 10000	-9999 (No Data)	30m
cdist	NA	Kilometers	0 - 24000	-9999 (No Data)	30m
drad	NA	W/(m <sup>2</sup> sr μm)/DN	0 - 28000	-9999 (No Data)	30m
urad	NA	W/(m <sup>2</sup> sr μm)/DN	0 - 28000	-9999 (No Data)	30m
trad	NA	W/(m <sup>2</sup> sr μm)/DN	0 - 28000	-9999 (No Data)	30m
emis	NA	Emissivity coefficient	1 - 10000	-9999 (No Data)	30m
emsd	NA	Emissivity coefficient	1 - 10000	-9999 (No Data)	30m

## Sentinel-2

<b>Band name</b>	<b>Wave length range (nm)</b>	<b>Scale</b>	<b>Valid range</b>	<b>Fill value</b>	<b>Spatial resolution</b>
coastal	443	0.0001	NA	0 (No Data)	60m
blue	490	0.0001	NA	0 (No Data)	10m
green	560	0.0001	NA	0 (No Data)	10m
red	665	0.0001	NA	0 (No Data)	10m
rededge1	705	0.0001	NA	0 (No Data)	20m
rededge2	740	0.0001	NA	0 (No Data)	20m
rededge3	783	0.0001	NA	0 (No Data)	20m
nir	842	0.0001	NA	0 (No Data)	10m
nir08	865	0.0001	NA	0 (No Data)	20m
nir08	865	0.0001	NA	0 (No Data)	20m
nir09	940	0.0001	NA	0 (No Data)	60m
swir16	1610	0.0001	NA	0 (No Data)	20m
swir22	2190	0.0001	NA	0 (No Data)	20m
aot	Aerosol optical thickness	0.001	NA	0 (No Data)	10m
wvp	Scene-average water vapor	0.001	NA	0 (No Data)	10m
scl	Scene classification data	NA	1 - 11	0 (No Data)	20m

# RStudio on Amazon SageMaker AI

RStudio is an integrated development environment for R, with a console, syntax-highlighting editor that supports direct code execution, and tools for plotting, history, debugging and workspace management. Amazon SageMaker AI supports RStudio as a fully-managed integrated development environment (IDE) integrated with Amazon SageMaker AI domain through Posit Workbench. RStudio allows customers to create data science insights using an R environment. With RStudio integration, you can launch an RStudio environment in the domain to run your RStudio workflows on SageMaker AI resources. For more information about Posit Workbench, see the [Posit website](#). This page gives information about important RStudio concepts.

SageMaker AI integrates RStudio through the creation of a RStudioServerPro app.

The following are supported by RStudio on SageMaker AI.

- R developers use the RStudio IDE interface with popular developer tools from the R ecosystem. Users can launch new RStudio sessions, write R code, install dependencies from RStudio Package Manager, and publish Shiny apps using RStudio Connect.
- R developers can quickly scale underlying compute resources to run large scale data processing and statistical analysis.
- Platform administrators can set up user identities, authorization, networking, storage, and security for their data science teams through AWS IAM Identity Center and AWS Identity and Access Management integration. This includes connection to private Amazon Virtual Private Cloud (Amazon VPC) resources and internet-free mode with AWS PrivateLink.
- Integration with AWS License Manager.

For information on the onboarding steps to create a domain with RStudio enabled, see [Amazon SageMaker AI domain overview](#).

## Region availability

The following table gives information about the AWS Regions that RStudio on SageMaker AI is supported in.

Region name	Region
US East (Ohio)	us-east-2

Region name	Region
US East (N. Virginia)	us-east-1
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Asia Pacific (Mumbai)	ap-south-1
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Singapore)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
Europe (Frankfurt)	eu-central-1
Europe (Ireland)	eu-west-1
Europe (London)	eu-west-2
Europe (Paris)	eu-west-3
Europe (Stockholm)	eu-north-1
South America (São Paulo)	sa-east-1

## RStudio components

- *RStudioServerPro*: The RStudioServerPro app is a multiuser app that is a shared resource among all user profiles in the domain. Once an RStudio app is created in a domain, the admin can give permissions to users in the domain.
- *RStudio user*: RStudio users are users within the domain that are authorized to use the RStudio license.

- *RStudio admin*: An RStudio on Amazon SageMaker AI admin can access the RStudio administrative dashboard. RStudio on Amazon SageMaker AI admins differ from "stock" Posit Workbench admins because they do not have root access to the instance running the RStudioServerPro app and can't modify the RStudio configuration file.
- *RStudio Server*: The RStudio Server instance is responsible for serving the RStudio UI to all authorized Users. This instance is launched on an Amazon SageMaker AI instance.
- *RSession*: An RSession is a browser-based interface to the RStudio IDE running on an Amazon SageMaker AI instance. Users can create and interact with their RStudio projects through the RSession.
- *RSessionGateway*: The RSessionGateway app is used to support an RSession.
- *RStudio administrative dashboard*: This dashboard gives information on the RStudio users in the Amazon SageMaker AI domain and their sessions. This dashboard can only be accessed by users that have RStudio admin authorization.

## Differences from Posit Workbench

RStudio on Amazon SageMaker AI has some significant differences from [Posit Workbench](#).

- When using RStudio on SageMaker AI, users don't have access to the RStudio configuration files. Amazon SageMaker AI manages the configuration file and sets defaults. You can modify the RStudio Connect and RStudio Package Manager URLs when creating your RStudio-enabled Amazon SageMaker AI domain.
- Project sharing, realtime collaboration, and Job Launcher are not currently supported when using RStudio on Amazon SageMaker AI.
- When using RStudio on SageMaker AI, the RStudio IDE runs on Amazon SageMaker AI instances for on-demand containerized compute resources.
- RStudio on SageMaker AI only supports the RStudio IDE and does not support other IDEs supported by a Posit Workbench installation.
- RStudio on SageMaker AI only supports the RStudio version specified in [RStudio Versioning](#).

## RStudio on Amazon SageMaker AI management

The following topics give information on managing RStudio on Amazon SageMaker AI. This includes information about your RStudio environment configuration, user sessions, and necessary

resources. For information on how to use RStudio on SageMaker AI, see [RStudio on Amazon SageMaker AI user guide](#).

For information about creating a Amazon SageMaker AI domain with RStudio enabled, see [Amazon SageMaker AI domain overview](#).

For information about the AWS Regions that RStudio on SageMaker AI is supported in, see [Supported Regions and Quotas](#).

## Topics

- [Get an RStudio license](#)
- [RStudio Versioning](#)
- [Network and Storage](#)
- [RStudioServerPro instance type](#)
- [Add an RStudio Connect URL](#)
- [Update the RStudio Package Manager URL](#)
- [Create an Amazon SageMaker AI domain with RStudio using the AWS CLI](#)
- [Add RStudio support to an existing domain](#)
- [Custom images with RStudio on SageMaker AI](#)
- [Create a user to use RStudio](#)
- [Log in to RStudio as another user](#)
- [Terminate sessions for another user](#)
- [Use the RStudio administrative dashboard](#)
- [Shut down RStudio](#)
- [Billing and cost](#)
- [Diagnose issues and get support](#)

## Get an RStudio license

RStudio on Amazon SageMaker AI is a paid product and requires that each user is appropriately licensed. Licenses for RStudio on Amazon SageMaker AI may be obtained from RStudio PBC directly, or by purchasing a subscription to Posit Workbench on AWS Marketplace. For existing customers of Posit Workbench Enterprise, licenses are issued at no additional cost. To use an

RStudio license with Amazon SageMaker AI, you must first have a valid RStudio license registered with AWS License Manager. For licenses purchased directly through Rstudio PBC, a license grant for your AWS Account must be created. Contact RStudio for direct license purchases or to enable existing licenses in AWS License Manager. For more information about registering a license with AWS License Manager, see [Seller issued licenses in AWS License Manager](#).

The following topics show how to acquire and validate a license granted by RStudio PBC.

## Get an RStudio license

1. If you don't have an RStudio license, you may purchase one from the AWS Marketplace or from RStudio PBC directly.
  - To purchase a subscription from the AWS Marketplace, complete the steps to [subscribe with a SaaS contract](#) by searching for **Posit Platform (RStudio on SageMaker)**. To fulfill the license, you will be redirected to an external form outside the AWS Marketplace. You must provide additional information, including your company name and email address. If you can't access that form to provide a company name and a contact email, create a ticket with Posit Support at <https://support.posit.co/hc/en-us/requests/new> with details about your purchase.
  - To purchase from RStudio PBC directly, navigate to [RStudio Pricing](#) or contact [sales@rstudio.com](mailto:sales@rstudio.com). When buying or updating an RStudio license, you must provide the AWS Account that will host your Amazon SageMaker AI domain.

If you have an existing RStudio license, contact your RStudio Sales representative or [sales@rstudio.com](mailto:sales@rstudio.com) to add RStudio on Amazon SageMaker AI to your existing Posit Workbench Enterprise license, or to convert your Posit Workbench Standard license. The RStudio Sales representative will send you the appropriate electronic order form.

2. RStudio grants a Posit Workbench license to your AWS Account through AWS License Manager in the US East (N. Virginia) Region. Although the RStudio license is granted in the US East (N. Virginia) Region, your license can be consumed in any AWS Region that RStudio on Amazon SageMaker AI is supported in. You can expect the license grant process to complete within three business days after you share your AWS account ID with RStudio.
3. When this license is granted, you receive an email from your RStudio Sales representative with instructions to accept your license grant.

## Validate your RStudio license to be used with Amazon SageMaker AI

1. Log into the AWS License Manager console in the same region as your Amazon SageMaker AI domain. If you are using AWS License Manager for the first time, AWS License Manager prompts you to grant permission to use AWS License Manager.
2. Select **Start using AWS License manager**.
3. Select I grant AWS License Manager the required permissions and select **Grant Permissions**.
4. Navigate to **Granted Licenses** on the left panel.
5. Select the license grant with RSW-SageMaker as the Product name and select **View**.
6. From the license detail page, select **Accept & activate license**.

## RStudio administrative dashboard

You can use the RStudio administrative dashboard to see the number of users on the license following the steps in [Use the RStudio administrative dashboard](#).

## RStudio Versioning

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

This guide provides information about the 2024.04.2+764.pro1 version update for RStudio on SageMaker AI. Starting September 04, 2024, new domains with RStudio support are created with Posit Workbench version 2024.04.2+764.pro1. This applies to the RStudioServerPro applications and default RSessionGateway applications.

The following sections give information about the 2024.04.2+764.pro1 release.

## Latest version updates

The latest RStudio version is 2024.04.2+764.pro1. This version includes the following changes:

- R versions supported:
  - 4.4.0
  - 4.3.3
  - 4.2.3
  - 4.2.1
  - 4.1.3
  - 4.0.2

For more information about the changes in this release, see <https://docs.posit.co/ide/news/>.

### Note

To ensure compatibility, we recommend using RSessions with a prefix that matches the current Posit Workbench version.

If you see the following warning, there is a version mismatch between the RSession and the Posit Workbench version used in RStudio on SageMaker AI. To resolve this issue, update the RStudio version for the domain. For information about updating the RStudio version, see [Upgrade to the new version](#).

Session version 2023.03.3-547.pro5 does not match server version  
2024.04.2+764.pro1 - this is an unsupported configuration, and you may  
experience unexpected issues as a result.

## Versioning

There are currently two versions of Posit Workbench supported by SageMaker AI.

- Latest version supported: 2024.04.2+764.pro1
- Previous version supported: 2023.03.3-547.pro5

**Note**

SageMaker AI will support version 2023.03.3-547.pro5 until October 2024.

Version 2022.02.2-485.pro2 is deprecated and is no longer supported. We recommend updating to the latest version.

The default Posit Workbench version that SageMaker AI selects depends on the creation date of the domain.

- For domains created after September 04, 2024, version 2024.04.2+764.pro1 is the default selected version.
- For domains created after February 27, 2024 and before September 04, 2024, version 2023.03.3-547.pro5 is the default selected version. You can update your domains to the latest version (2024.04.2+764.pro1) by setting it as the default version for the domain. For more information, see [Upgrade to the new version](#).
- For domains created before February 27, 2024, version 2023.03.3-547.pro5 is the default selected version. You can update your domains to the latest version (2024.04.2+764.pro1) by setting it as the default version for the domain. For more information, see [Upgrade to the new version](#).

**Note**

The default RSessionGateway application version matches the current version of the RStudioServerPro application.

The following table lists the image ARNs for both versions for each AWS Region. These ARNs are passed as part of an update-domain command to set the desired version.

Region	2023.03.3-547.pro5 Image ARN	2024.04.2+764.pro1 Image ARN
us-east-1	arn:aws:sagemaker:us-east-1:081325390199:image/rstudio-workbench-2023.03	arn:aws:sagemaker:us-east-1:081325390199:image/rstudio-workbench-2024.04

Region	<b>2023.03.3-547.pro5</b> Image ARN	<b>2024.04.2+764.pro1</b> Image ARN
us-east-2	arn:aws:sagemaker:us-east-2:429704687514:image/rstudio-workbench-2023.03	arn:aws:sagemaker:us-east-2:429704687514:image/rstudio-workbench-2024.04
us-west-1	arn:aws:sagemaker:us-west-1:742091327244:image/rstudio-workbench-2023.03	arn:aws:sagemaker:us-west-1:742091327244:image/rstudio-workbench-2024.04
us-west-2	arn:aws:sagemaker:us-west-2:236514542706:image/rstudio-workbench-2023.03	arn:aws:sagemaker:us-west-2:236514542706:image/rstudio-workbench-2024.04
af-south-1	arn:aws:sagemaker:af-south-1:559312083959:image/rstudio-workbench-2023.03	arn:aws:sagemaker:af-south-1:559312083959:image/rstudio-workbench-2024.04
ap-east-1	arn:aws:sagemaker:ap-east-1:493642496378:image/rstudio-workbench-2023.03	arn:aws:sagemaker:ap-east-1:493642496378:image/rstudio-workbench-2024.04
ap-south-1	arn:aws:sagemaker:ap-south-1:394103062818:image/rstudio-workbench-2023.03	arn:aws:sagemaker:ap-south-1:394103062818:image/rstudio-workbench-2024.04
ap-northeast-2	arn:aws:sagemaker:ap-northeast-2:806072073708:image/rstudio-workbench-2023.03	arn:aws:sagemaker:ap-northeast-2:806072073708:image/rstudio-workbench-2024.04
ap-southeast-1	arn:aws:sagemaker:ap-southeast-1:492261229750:image/rstudio-workbench-2023.03	arn:aws:sagemaker:ap-southeast-1:492261229750:image/rstudio-workbench-2024.04
ap-southeast-2	arn:aws:sagemaker:ap-southeast-2:452832661640:image/rstudio-workbench-2023.03	arn:aws:sagemaker:ap-southeast-2:452832661640:image/rstudio-workbench-2024.04

Region	2023.03.3-547.pro5 Image ARN	2024.04.2+764.pro1 Image ARN
ap-northeast-1	arn:aws:sagemaker:ap-northeast-1:102112518831:image/rstudio-workbench-2023.03	arn:aws:sagemaker:ap-northeast-1:102112518831:image/rstudio-workbench-2024.04
ca-central-1	arn:aws:sagemaker:ca-central-1:310906938811:image/rstudio-workbench-2023.03	arn:aws:sagemaker:ca-central-1:310906938811:image/rstudio-workbench-2024.04
eu-central-1	arn:aws:sagemaker:eu-central-1:936697816551:image/rstudio-workbench-2023.03	arn:aws:sagemaker:eu-central-1:936697816551:image/rstudio-workbench-2024.04
eu-west-1	arn:aws:sagemaker:eu-west-1:470317259841:image/rstudio-workbench-2023.03	arn:aws:sagemaker:eu-west-1:470317259841:image/rstudio-workbench-2024.04
eu-west-2	arn:aws:sagemaker:eu-west-2:712779665605:image/rstudio-workbench-2023.03	arn:aws:sagemaker:eu-west-2:712779665605:image/rstudio-workbench-2024.04
eu-west-3	arn:aws:sagemaker:eu-west-3:615547856133:image/rstudio-workbench-2023.03	arn:aws:sagemaker:eu-west-3:615547856133:image/rstudio-workbench-2024.04
eu-north-1	arn:aws:sagemaker:eu-north-1:243637512696:image/rstudio-workbench-2023.03	arn:aws:sagemaker:eu-north-1:243637512696:image/rstudio-workbench-2024.04
eu-south-1	arn:aws:sagemaker:eu-south-1:592751261982:image/rstudio-workbench-2023.03	arn:aws:sagemaker:eu-south-1:592751261982:image/rstudio-workbench-2024.04
sa-east-1	arn:aws:sagemaker:sa-east-1:782484402741:image/rstudio-workbench-2023.03	arn:aws:sagemaker:sa-east-1:782484402741:image/rstudio-workbench-2024.04

## Changes to BYOI Images

If you use a BYOI image with RStudio and update your RStudioServerPro version to 2024.04.2+764.pro1, you must upgrade your custom images to use the 2024.04.2+764.pro1 release and redeploy your existing RSessions. If you attempt to load a non-compatible image in an RSession of a domain using the 2024.04.2+764.pro1 version, the RSession fails because it cannot parse parameters that it receives. To prevent failure, update all of the deployed custom images in your existing RStudioServerPro application.

The RSW\_VERSION in the Dockerfile must be consistent with the Posit Workbench version used in RStudio on SageMaker AI. You can validate the current version in Posit Workbench. To do so, use the version name that's located in the lower left corner of the Posit Workbench launcher page.

```
ARG RSW_VERSION=2024.04.2+764.pro1
ENV RSTUDIO_FORCE_NON_ZERO_EXIT_CODE="1"
ARG RSW_NAME=rstudio-workbench
ARG OS_CODE_NAME=jammy
ARG RSW_DOWNLOAD_URL=https://s3.amazonaws.com/rstudio-ide-build/server/${OS_CODE_NAME}/
amd64
RUN RSW_VERSION_URL=`echo -n "${RSW_VERSION}" | sed 's/+/-/g'` \
    && curl -o rstudio-workbench.deb ${RSW_DOWNLOAD_URL}/${RSW_NAME}-
${RSW_VERSION_URL}-amd64.deb \
    && gdebi -n ./rstudio-workbench.deb
```

## Upgrade to the new version

Existing domains using version 2023.03.3-547.pro5 can upgrade to 2024.04.2+764.pro1 version in one of two ways:

- Create a new domain from the AWS CLI with RStudio enabled.
- Update an existing domain to use the 2024.04.2+764.pro1 version.

The following procedure shows how to delete the RStudio application for an existing domain, set the default version to 2024.04.2+764.pro1, and then create an RStudio application.

1. Delete the RStudioServerPro application and all RSessionGateway applications associated with your existing domain. For information about how to find your domain ID, see [View domains](#). For more information about deleting applications, see [Shut down RStudio](#).

```
aws sagemaker delete-app \
```

```
--region region \
--domain-id domainId \
--user-profile-name domain-shared \
--app-type RStudioServerPro \
--app-name default
```

2. If your domain is using RStudio version 2023.03.3-547.pro5, update the domain to set 2024.04.2+764.pro1 as the default Posit Workbench version. The SageMakerImageArn value in the following update-domain command specifies the RStudio 2024.04.2+764.pro1 version as the default. This ARN must match the Region that your domain is in. For a list of all available ARNs, see [Versioning](#).

Pass an execution role ARN for the domain that provides permissions to update the domain.

```
aws sagemaker update-domain \
--region region \
--domain-id domainId \
--domain-settings-for-update "{\"RStudioServerProDomainSettingsForUpdate\": {\"DefaultResourceSpec\": {\"SageMakerImageArn\": \"arn-for-2024.04.2+764.pro1-version\", \"InstanceType\": \"system\"}, \"DomainExecutionRoleArn\": \"execution-role-arn\"}}"
```

3. Create a new RStudioServerPro application in the existing domain.

```
aws sagemaker create-app \
--region region
--domain-id domainId \
--user-profile-name domain-shared \
--app-type RStudioServerPro \
--app-name default
```

Your RStudioServerPro application is now updated to version 2024.04.2+764.pro1. You can now relaunch your RSessionGateway applications.

## Downgrade to the existing version

You can manually downgrade the version of your existing RStudio application to the 2023.03.3-547.pro5 version.

## To downgrade to the existing version

1. Delete the RStudioServerPro application that's associated with your existing domain. For information about how to find your domain ID, see [View domains](#).

```
aws sagemaker delete-app \
--domain-id domainId \
--user-profile-name domain-shared \
--app-type RStudioServerPro \
--app-name default
```

2. Pass the corresponding 2023.03.3-547.pro5 ARN for your Region as part of the update-domain command. For a list of all available ARNs, see [Versioning](#). You must also pass an execution role ARN for the domain that provides permissions to update the domain.

```
aws sagemaker update-domain \
--region region \
--domain-id domainId \
--domain-settings-for-update "{\"RStudioServerProDomainSettingsForUpdate\": {\"DefaultResourceSpec\": {\"SageMakerImageArn\": \"arn-for-2023.03.3-547.pro5-version\", \"InstanceType\": \"system\"}, \"DomainExecutionRoleArn\": \"execution-role-arn\"}}"
```

3. Create a new RStudioServerPro application in the existing domain. The RStudio version defaults to 2023.03.3-547.pro5.

```
aws sagemaker create-app \
--domain-id domainId \
--user-profile-name domain-shared \
--app-type RStudioServerPro \
--app-name default
```

Your RStudioServerPro application is now downgraded to version 2023.03.3-547.pro5.

## Network and Storage

The following topic describes network access and data storage considerations for your RStudio instance. For general information about network access and data storage when using Amazon SageMaker AI, see [Data Protection in Amazon SageMaker AI](#).

### Amazon EFS volume

RStudio on Amazon SageMaker AI shares an Amazon EFS volume with the Amazon SageMaker Studio Classic application in the domain. When the RStudio application is added to a domain, SageMaker AI creates a folder named shared in the Amazon EFS directory. If this shared folder is deleted or changed manually, then the RStudio application may no longer function. For more information about the Amazon EFS volume, see [Manage Your Amazon EFS Storage Volume in SageMaker Studio Classic](#).

## Installed packages and scripts

Packages that you install from within RStudio are scoped to the user profile level. This means that the installed package persists through RSession shut down, restarts, and across RSessions for each user profile that they are installed in. R Scripts that are saved in RSessions behave the same way. Both packages and R Scripts are saved in the user's Amazon EFS volume.

## Encryption

RStudio on Amazon SageMaker AI supports encryption at rest.

## Use RStudio in VPC-only mode

RStudio on Amazon SageMaker AI supports [AWS PrivateLink](#) integration. With this integration, you can use RStudio on SageMaker AI in VPC-only mode without direct access to the internet. When you use RStudio in VPC-only mode, your security groups are automatically managed by the service. This includes connectivity between your RServer and your RSessions.

The following are required to use RStudio in VPC-only mode. For more information on selecting a VPC, see [Choose an Amazon VPC](#).

- A private subnet with either access the internet to make a call to Amazon SageMaker AI & License Manager, or Amazon Virtual Private Cloud (Amazon VPC) endpoints for both Amazon SageMaker AI & License Manager.
- The domain cannot have any more than two associated Security Groups.
- A Security Group ID for use with the domain in domain Settings. This must allow all outbound access.
- A Security Group ID for use with the Amazon VPC endpoint. This security group must allow inbound traffic from the domain Security Group ID.
- Amazon VPC Endpoint for `sagemaker.api` and AWS License Manager. This must be in the same Amazon VPC as the private subnet.

## RStudioServerPro instance type

When deciding which Amazon EC2 instance type to use for your RStudioServerPro app, the main factor to consider is bandwidth. Bandwidth is important because the RStudioServerPro instance is responsible for serving the RStudio UI to all users. This includes UI heavy workflows, such as generating figures, animations, and displaying many data rows. Therefore, there may be some UI performance degradation depending on the workload across all users. The following are the available instance types to use for your RStudioServerPro. For pricing information about these instances, see [Amazon SageMaker Pricing](#).

- **system:** This instance type is recommended for Domains with low UI use.

 **Note**

The system value is translated to `m1.t3.medium`.

- `m1.c5.4xlarge`: This instance type is recommended for Domains with moderate UI use.
- `m1.c5.9xlarge`: This instance type is recommended for Domains with heavy UI use.

## Changing RStudio instance type

To change the instance type of your RStudioServerPro, pass the new instance type as part of a call to the `update-domain` CLI command. You then need to delete the existing RStudioServerPro app using the `delete-app` CLI command and create a new RStudioServerPro app using the `create-app` CLI command.

## Add an RStudio Connect URL

RStudio Connect is a publishing platform for Shiny applications, R Markdown reports, dashboards, plots, and more. RStudio Connect makes it easy to surface machine learning and data science insights by making hosting content simple and scalable. If you have an RStudio Connect server, then you can set the server as the default place where apps are published. For more information about RStudio Connect, see [RStudio Connect](#).

When you onboard to RStudio on Amazon SageMaker AI domain, an RStudio Connect server is not created. You can create an RStudio Connect server on an Amazon EC2 instance to use Connect with Amazon SageMaker AI domain. For information about how to set up your RStudio Connect server, see [Host RStudio Connect and Package Manager for ML development in RStudio on Amazon SageMaker AI](#).

## Add an RStudio Connect URL

If you have an RStudio Connect URL, you can update the default URL so that your RStudio Users can publish to it.

1. Navigate to the **domains** page.
2. Select the desired domain.
3. Choose **domain Settings**.
4. Under **General Settings**, select **Edit**.
5. From the new page, select **RStudio Settings** on the left side.
6. Under **RStudio Connect URL**, enter the RStudio Connect URL to add.
7. Select **Submit**.

### CLI

You can set a default RStudio Connect URL when you create your domain. The only way to update your RStudio Connect URL from the AWS CLI is to delete your domain and create a new one with the updated RStudio Connect URL.

## Update the RStudio Package Manager URL

RStudio Package Manager is a repository management server used to organize and centralize packages across your organization. For more information on RStudio Package Manager, see [RStudio Package Manager](#). If you don't supply your own Package Manager URL, Amazon SageMaker AI domain uses the default Package Manager repository when you onboard RStudio following the steps in [Amazon SageMaker AI domain overview](#). For more information, see [Host RStudio Connect and Package Manager for ML development in RStudio on Amazon SageMaker AI](#). The following procedure shows how to update the Package Manager URL.

### Update Package Manager URL

You can update the Package Manager URL used for your RStudio-enabled domain as follows.

1. Navigate to the **domains** page.
2. Select the desired domain.
3. Choose **domain Settings**.
4. Under **General Settings**, select **Edit**.

5. From the new page, select **RStudio Settings** on the left side.
6. Under **RStudio Package Manager**, enter your RStudio Package Manager URL.
7. Select **Submit**.

## CLI

The only way to update your Package Manager URL from the AWS CLI is to delete your domain and create a new one with the updated Package Manager URL.

## Create an Amazon SageMaker AI domain with RStudio using the AWS CLI

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

The following topic shows how to onboard to Amazon SageMaker AI domain with RStudio enabled using the AWS CLI. To onboard using the AWS Management Console, see [Amazon SageMaker AI domain overview](#).

### Prerequisites

- Install and configure [AWS CLI version 2](#)
- Configure the [AWS CLI](#) with IAM credentials

### Create DomainExecution role

To launch the RStudio App, you must provide a DomainExecution role. This role is used to determine whether RStudio needs to be launched as part of Amazon SageMaker AI domain

creation. This role is also used by Amazon SageMaker AI to access the RStudio License and push RStudio logs.

 **Note**

The DomainExecution role should have at least AWS License Manager permissions to access RStudio License, and CloudWatch permissions to push logs in your account.

The following procedure shows how to create the DomainExecution role with the AWS CLI.

1. Create a file named `assume-role-policy.json` with the following content.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Action": "sts:AssumeRole",  
            "Effect": "Allow",  
            "Principal": {  
                "Service": [  
                    "sagemaker.amazonaws.com"  
                ]  
            }  
        }  
    ]  
}
```

2. Create the DomainExecution role. `<REGION>` should be the AWS Region to launch your domain in.

```
aws iam create-role --region <REGION> --role-name DomainExecution --assume-role-policy-document file://assume-role-policy.json
```

3. Create a file named `domain-setting-policy.json` with the following content. This policy allows the RStudioServerPro app to access necessary resources and allows Amazon SageMaker AI to automatically launch an RStudioServerPro app when the existing RStudioServerPro app is in a Deleted or Failed status.

```
{  
    "Version": "2012-10-17",
```

```
"Statement": [
    {
        "Sid": "VisualEditor0",
        "Effect": "Allow",
        "Action": [
            "license-manager:ExtendLicenseConsumption",
            "license-manager>ListReceivedLicenses",
            "license-manager:GetLicense",
            "license-manager:CheckoutLicense",
            "license-manager:CheckInLicense",
            "logs>CreateLogDelivery",
            "logs>CreateLogGroup",
            "logs>CreateLogStream",
            "logs>DeleteLogDelivery",
            "logs>Describe*",
            "logs>GetLogDelivery",
            "logs>GetLogEvents",
            "logs>ListLogDeliveries",
            "logs>PutLogEvents",
            "logs>PutResourcePolicy",
            "logs>UpdateLogDelivery",
            "sagemaker>CreateApp"
        ],
        "Resource": "*"
    }
]
```

4. Create the domain setting policy that is attached to the DomainExecution role. Be aware of the PolicyArn from the response, you will need to enter that ARN in the following steps.

```
aws iam create-policy --region <REGION> --policy-name domain-setting-policy --policy-document file://domain-setting-policy.json
```

5. Attach domain-setting-policy to the DomainExecution role. Use the PolicyArn returned in the previous step.

```
aws iam attach-role-policy --role-name DomainExecution --policy-arn <POLICY_ARN>
```

## Create Amazon SageMaker AI domain with RStudio App

The RStudioServerPro app is launched automatically when you create a Amazon SageMaker AI domain using the `create-domain` CLI command with the `RStudioServerProDomainSettings` parameter specified. When launching the RStudioServerPro App, Amazon SageMaker AI checks for a valid RStudio license in the account and fails domain creation if the license is not found.

The creation of a Amazon SageMaker AI domain differs based on the authentication method and the network type. These options must be used together, with one authentication method and one network connection type selected. For more information about the requirements to create a new domain, see [CreateDomain](#).

The following authentication methods are supported.

- IAM Auth
- SSO Auth

The following network connection types are supported:

- PublicInternet
- VPCOnly

### Authentication methods

#### IAM Auth Mode

The following shows how to create a Amazon SageMaker AI domain with RStudio enabled and an IAM Auth Network Type. For more information about AWS Identity and Access Management, see [What is IAM?](#).

- `DomainExecutionRoleArn` should be the ARN for the role created in the previous step.
- `ExecutionRole` is the ARN of the role given to users in the Amazon SageMaker AI domain.
- `vpc-id` should be the ID of your Amazon Virtual Private Cloud. `subnet-ids` should be a space-separated list of subnet IDs. For information about `vpc-id` and `subnet-ids`, see [VPCs and subnets](#).
- `RStudioPackageManagerUrl` and `RStudioConnectUrl` are optional and should be set to the URLs of your RStudio Package Manager and RStudio Connect server, respectively.

- app-network-access-type should be either PublicInternetOnly or VPCOnly.

```
aws sagemaker create-domain --region <REGION> --domain-name <DOMAIN_NAME> \
    --auth-mode IAM \
    --default-user-settings ExecutionRole=<DEFAULT_USER_EXECUTIONROLE> \
    --domain-settings
RStudioServerProDomainSettings={RStudioPackageManagerUrl=<<PACKAGE_MANAGER_URL>>, RStudioConnectUrl=<<CONNECT_URL>>} \
    --vpc-id <VPC_ID> \
    --subnet-ids <SUBNET_IDS> \
    --app-network-access-type <NETWORK_ACCESS_TYPE>
```

## Authentication using IAM Identity Center

The following shows how to create a Amazon SageMaker AI domain with RStudio enabled and an SSO Auth Network Type. AWS IAM Identity Center must be enabled for the region that the domain is launched on. For more information about IAM Identity Center, see [What is AWS IAM Identity Center?](#).

- DomainExecutionRoleArn should be the ARN for the role created in the previous step.
- ExecutionRole is the ARN of the role given to users in the Amazon SageMaker AI domain.
- vpc-id should be the ID of your Amazon Virtual Private Cloud. subnet-ids should be a space-separated list of subnet IDs. For information about vpc-id and subnet-ids, see [VPCs and subnets](#).
- RStudioPackageManagerUrl and RStudioConnectUrl are optional and should be set to the URLs of your RStudio Package Manager and RStudio Connect server, respectively.
- app-network-access-type should be either PublicInternetOnly or VPCOnly.

```
aws sagemaker create-domain --region <REGION> --domain-name <DOMAIN_NAME> \
    --auth-mode SSO \
    --default-user-settings ExecutionRole=<DEFAULT_USER_EXECUTIONROLE> \
    --domain-settings
RStudioServerProDomainSettings={RStudioPackageManagerUrl=<<PACKAGE_MANAGER_URL>>, RStudioConnectUrl=<<CONNECT_URL>>} \
    --vpc-id <VPC_ID> \
    --subnet-ids <SUBNET_IDS> \
    --app-network-access-type <NETWORK_ACCESS_TYPE>
```

## Connection types

### PublicInternet/Direct Internet network type

The following shows how to create a Amazon SageMaker AI domain with RStudio enabled and a PublicInternet Network Type.

- DomainExecutionRoleArn should be the ARN for the role created in the previous step.
- ExecutionRole is the ARN of the role given to users in the Amazon SageMaker AI domain.
- vpc-id should be the ID of your Amazon Virtual Private Cloud. subnet-ids should be a space-separated list of subnet IDs. For information about vpc-id and subnet-ids, see [VPCs and subnets](#).
- RStudioPackageManagerUrl and RStudioConnectUrl are optional and should be set to the URLs of your RStudio Package Manager and RStudio Connect server, respectively.
- auth-mode should be either SSO or IAM.

```
aws sagemaker create-domain --region <REGION> --domain-name <DOMAIN_NAME> \
    --auth-mode <AUTH_MODE> \
    --default-user-settings ExecutionRole=<DEFAULT_USER_EXECUTIONROLE> \
    --domain-settings
RStudioServerProDomainSettings={RStudioPackageManagerUrl=<<PACKAGE_MANAGER_URL>>,RStudioConnectUrl=<<CONNECT_URL>>} \
    --vpc-id <VPC_ID> \
    --subnet-ids <SUBNET_IDS> \
    --app-network-access-type PublicInternetOnly
```

### VPCOnly mode

The following shows how to launch a Amazon SageMaker AI domain with RStudio enabled and a VPCOnly Network Type. For more information about using the VPCOnly network access type, see [Connect Studio notebooks in a VPC to external resources](#).

- DomainExecutionRoleArn should be the ARN for the role created in the previous step.
- ExecutionRole is the ARN of the role given to users in the Amazon SageMaker AI domain.
- vpc-id should be the ID of your Amazon Virtual Private Cloud. subnet-ids should be a space-separated list of subnet IDs. Your private subnet must be able to either access the internet to make a call to Amazon SageMaker AI, and AWS License Manager or have Amazon VPC endpoints for both Amazon SageMaker AI and AWS License Manager. For information about Amazon VPC

- endpoints, see [Interface Amazon VPC endpoints](#). For information about vpc-id and subnet-ids, see [VPCs and subnets](#).
- SecurityGroups must allow outbound access to the Amazon SageMaker AI and AWS License Manager endpoints.
  - auth-mode should be either SSO or IAM.

### Note

When using Amazon Virtual Private Cloud endpoints, the security group attached to your Amazon Virtual Private Cloud endpoints must allow inbound traffic from the security group you pass as part of the domain-setting parameter of the create-domain CLI call.

With RStudio, Amazon SageMaker AI manages security groups for you. This means that Amazon SageMaker AI manages security group rules to ensure RSessions can access RStudioServerPro Apps. Amazon SageMaker AI creates one security group rule per user profile.

```
aws sagemaker create-domain --region <REGION> --domain-name <DOMAIN_NAME> \
    --auth-mode <AUTH_MODE> \
    --default-user-settings
    SecurityGroups=<USER_SECURITY_GROUP>,ExecutionRole=<DEFAULT_USER_EXECUTIONROLE> \
    --domain-settings
    SecurityGroupIds=<DOMAIN_SECURITY_GROUP>,RStudioServerProDomainSettings={DomainExecutionRoleAr
\ \
    --vpc-id <VPC_ID> \
    --subnet-ids "<SUBNET_IDS>" \
    --app-network-access-type VPCOnly --app-security-group-management Service
```

Note: The RStudioServerPro app is launched by a special user profile named domain-shared. As a result, this app is not returned as part of list-app API calls by any other user profiles.

You may have to increase the Amazon VPC quota in your account to increase the number of users. For more information, see [Amazon VPC quotas](#).

### Verify domain creation

Use the following command to verify that your domain has been created with a Status of InService. Your domain-id is appended to the domains ARN. For example, arn:aws:sagemaker:<REGION>:<ACCOUNT\_ID>:domain/<DOMAIN\_ID>.

```
aws sagemaker describe-domain --domain-id <DOMAIN_ID> --region <REGION>
```

## Add RStudio support to an existing domain

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

If you have added an RStudio License through AWS License Manager, you can create a new Amazon SageMaker AI domain with support for RStudio on SageMaker AI. If you have an existing domain that does not support RStudio, you can add RStudio support to that domain without having to delete and recreate the domain.

The following topic outlines how to add this support.

### Prerequisites

You must complete the following steps before you update your current domain to add support for RStudio on SageMaker AI.

- Install and configure [AWS CLI version 2](#)
- Configure the [AWS CLI](#) with IAM credentials
- Create a domain execution role following the steps in [Create a SageMaker AI Domain with RStudio using the AWS CLI](#). This domain-level IAM role is required by the RStudioServerPro app. The role requires access to AWS License Manager for verifying a valid Posit Workbench license and Amazon CloudWatch Logs for publishing server logs.
- Bring your RStudio license to AWS License Manager following the steps in [RStudio license](#).

- (Optional) If you want to use RStudio in VPCOnly mode, complete the steps in [RStudio in VPC-Only](#).
- Ensure that the security groups you have configured for each [UserProfile](#) in your domain meet the account-level quotas. When configuring the default user profile during domain creation, you can use the `DefaultUserSettings` parameter of the [CreateDomain](#) API to add `SecurityGroups` that are inherited by all the user profiles created in the domain. You can also provide additional security groups for a specific user as part of the `UserSettings` parameter of the [CreateUserProfile](#) API. If you have added security groups this way, you must ensure that the total number of security groups per user profile doesn't exceed the maximum quota of 2 in VPCOnly mode and 4 in PublicInternetOnly mode. If the resulting total number of security groups for any user profile exceeds the quota, you can combine multiple security groups' rules into one security group.

## Add RStudio support to an existing domain

After you have completed the prerequisites, you can add RStudio support to your existing domain. The following steps outline how to update your existing domain to add support for RStudio.

### Step 1: Delete all apps in the domain

To add support for RStudio in your domain, SageMaker AI must update the underlying security groups for all existing user profiles. To complete this, you must delete and recreate all existing apps in the domain. The following procedure shows how to delete all of the apps.

1. List all of the apps in the domain.

```
aws sagemaker \
  list-apps \
  --domain-id-equals <DOMAIN_ID>
```

2. Delete each app for each user profile in the domain.

```
// JupyterServer apps
aws sagemaker \
  delete-app \
  --domain-id <DOMAIN_ID> \
  --user-profile-name <USER_PROFILE> \
  --app-type JupyterServer \
  --app-name <APP_NAME>
```

```
// KernelGateway apps
aws sagemaker \
    delete-app \
    --domain-id <DOMAIN_ID> \
    --user-profile-name <USER_PROFILE> \
    --app-type KernelGateway \
    --app-name <APP_NAME>
```

## Step 2 - Update all user profiles with the new list of security groups

This is a one-time action that you must complete for all of the existing user profiles in your domain when you have refactored your existing security groups. This prevents you from hitting the quota for the maximum number of security groups. The `UpdateUserProfile` API call fails if the user has any apps that are in [InService](#) status. Delete all apps, then call `UpdateUserProfile` API to update the security groups.

### Note

The following requirement for VPCOnly mode outlined in [Connect Amazon SageMaker Studio Classic Notebooks in a VPC to External Resources](#) is no longer needed when adding RStudio support because AppSecurityGroupManagement is managed by the SageMaker AI service:

[“TCP traffic within the security group”](#). This is required for connectivity between the JupyterServer app and the KernelGateway apps. You must allow access to at least ports in the range 8192–65535.”

```
aws sagemaker \
    update-user-profile \
    --domain-id <DOMAIN_ID> \
    --user-profile-name <USER_PROFILE> \
    --user-settings "{\"SecurityGroups\": [\"<SECURITY_GROUP>\", \
    \"<SECURITY_GROUP>\"]}]}
```

## Step 3 - Activate RStudio by calling the `UpdateDomain` API

1. Call the [UpdateDomain](#) API to add support for RStudio on SageMaker AI. The `defaultusersettings` parameter is only needed if you have refactored the default security groups for your user profiles.

- For VPCOnly mode:

```
aws sagemaker \
    update-domain \
    --domain-id <DOMAIN_ID> \
    --app-security-group-management Service \
    --domain-settings-for-update
RStudioServerProDomainSettingsForUpdate={DomainExecutionRoleArn=<DOMAIN_EXECUTION_ROLE_A
\
    --default-user-settings "{\"SecurityGroups\": [\"<SECURITY_GROUP>\",
\ "<SECURITY_GROUP>\"]}"
```

- For PublicInternetOnly mode:

```
aws sagemaker \
    update-domain \
    --domain-id <DOMAIN_ID> \
    --domain-settings-for-update
RStudioServerProDomainSettingsForUpdate={DomainExecutionRoleArn=<DOMAIN_EXECUTION_ROLE_A
    --default-user-settings "{\"SecurityGroups\": [\"<SECURITY_GROUP>\",
\ "<SECURITY_GROUP>\"]}"
```

2. Verify that the domain status is InService. After the domain status is InService, support for RStudio on SageMaker AI is added.

```
aws sagemaker \
    describe-domain \
    --domain-id <DOMAIN_ID>
```

3. Verify that the RStudioServerPro app's status is InService using the following command.

```
aws sagemaker list-apps --user-profile-name domain-shared
```

#### Step 4 - Add RStudio access for existing users

As part of the update in Step 3, SageMaker AI marks the RStudio [AccessStatus](#) of all existing user profiles in the domain as DISABLED by default. This prevents exceeding the number of users allowed by your current license. To add access for existing users, there is a one-time opt-in step. Perform the opt-in by calling the [UpdateUserProfile](#) API with the following [RStudioServerProAppSettings](#):

- AccessStatus = ENABLED
- *Optional* - UserGroup = R\_STUDIO\_USER or R\_STUDIO\_ADMIN

```
aws sagemaker \
    update-user-profile \
    --domain-id <DOMAIN_ID> \
    --user-profile-name <USER_PROFILE> \
    --user-settings "{\"RStudioServerProAppSettings\": {\"AccessStatus\": \"ENABLED \
\"}}"
```

 **Note**

By default, the number of users that can have access to RStudio is 60.

## Step 5 – Deactivate RStudio access for new users

Unless otherwise specified when calling `UpdateDomain`, RStudio support is added by default for all new user profiles created after you have added support for RStudio on SageMaker AI. To deactivate access for a new user profile, you must explicitly set the `AccessStatus` parameter to `DISABLED` as part of the `CreateUserProfile` API call. If the `AccessStatus` parameter is not specified as part of the `CreateUserProfile` API, the default access status is `ENABLED`.

```
aws sagemaker \
    create-user-profile \
    --domain-id <DOMAIN_ID> \
    --user-profile-name <USER_PROFILE> \
    --user-settings "{\"RStudioServerProAppSettings\": {\"AccessStatus\": \"DISABLED \
\"}}"
```

## Custom images with RStudio on SageMaker AI

A SageMaker image is a file that identifies language packages and other dependencies that are required to run RStudio on Amazon SageMaker AI. SageMaker AI uses these images to create an environment where you run RStudio. Amazon SageMaker AI provides a built-in RStudio image for you to use. If you need different functionality, you can bring your own custom images. This page gives information about key concepts for using custom images with RStudio on SageMaker AI. The process to bring your own image to use with RStudio on SageMaker AI takes three steps:

1. Build a custom image from a Dockerfile and push it to a repository in Amazon Elastic Container Registry (Amazon ECR).
2. Create a SageMaker image that points to a container image in Amazon ECR and attach it to your Amazon SageMaker AI domain.
3. Launch a new session in RStudio with your custom image.

You can create images and image versions, and attach image versions to your domain, using the SageMaker AI control panel, the [AWS SDK for Python \(Boto3\)](#), and the [AWS Command Line Interface \(AWS CLI\)](#). You can also create images and image versions using the SageMaker AI console, even if you haven't onboarded to a domain.

The following topics show how to bring your own image to RStudio on SageMaker AI by creating, attaching, and launching a custom image.

## Key terminology

The following section defines key terms for bringing your own image to use with RStudio on SageMaker AI.

- **Dockerfile:** A Dockerfile is a file that identifies the language packages and other dependencies for your Docker image.
- **Docker image:** The Docker image is a built Dockerfile. This image is checked into Amazon ECR and serves as the basis of the SageMaker AI image.
- **SageMaker image:** A SageMaker image is a holder for a set of SageMaker image versions based on Docker images.
- **Image version:** An image version of a SageMaker image represents a Docker image that is compatible with RStudio and stored in an Amazon ECR repository. Each image version is immutable. These image versions can be attached to a domain and used with RStudio on SageMaker AI.

## Complete prerequisites

You must complete the following prerequisites before bringing your own image to use with RStudio on Amazon SageMaker AI.

- If you have an existing domain with RStudio that was created before April 7, 2022, you must delete your RStudioServerPro application and recreate it. For information about how to delete an application, see [Shut down and Update SageMaker Studio Classic](#).
- Install the Docker application. For information about setting up Docker, see [Orientation and setup](#).
- Create a local copy of an RStudio-compatible Dockerfile that works with SageMaker AI. For information about creating a sample RStudio dockerfile, see [Use a custom image to bring your own development environment to RStudio on Amazon SageMaker AI](#).
- Use an AWS Identity and Access Management execution role that has the [AmazonSageMakerFullAccess](#) policy attached. If you have onboarded to domain, you can get the role from the **domain Summary** section of the SageMaker AI control panel.

Add the following permissions to access the Amazon Elastic Container Registry (Amazon ECR) service to your execution role.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "VisualEditor0",  
            "Effect": "Allow",  
            "Action": [  
                "ecr:CreateRepository",  
                "ecr:BatchGetImage",  
                "ecr:CompleteLayerUpload",  
                "ecr:DescribeImages",  
                "ecr:DescribeRepositories",  
                "ecr:UploadLayerPart",  
                "ecr>ListImages",  
                "ecr:InitiateLayerUpload",  
                "ecr:BatchCheckLayerAvailability",  
                "ecr:PutImage"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

- Install and configure AWS CLI with the following (or higher) version. For information about installing the AWS CLI, see [Installing or updating the latest version of the AWS CLI](#).

```
AWS CLI v1 >= 1.23.6  
AWS CLI v2 >= 2.6.2
```

## Custom RStudio image specifications

In this guide, you'll learn custom RStudio image specifications to use when you bring your own image. There are two sets of requirements that you must satisfy with your custom RStudio image to use it with Amazon SageMaker AI. These requirements are imposed by RStudio PBC and the Amazon SageMaker Studio Classic platform. If either of these sets of requirements aren't satisfied, then your custom image won't function properly.

### RStudio PBC requirements

RStudio PBC requirements are laid out in the [Using Docker images with RStudio Workbench / RStudio Server Pro, Launcher, and Kubernetes](#) article. Follow the instructions in this article to create the base of your custom RStudio image.

For instructions about how to install multiple R versions in your custom image, see [Installing multiple versions of R on Linux](#).

### Amazon SageMaker Studio Classic requirements

Amazon SageMaker Studio Classic imposes the following set of installation requirements for your RStudio image.

- You must use an RStudio base image of at least 2023.03.2-454.pro2. For more information, see [RStudio Versioning](#).
- You must install the following packages:

```
yum install -y sudo \  
openjdk-11-jdk \  
libpng-dev \  
&& yum clean all \  
&& /opt/R/${R_VERSION}/bin/R -e "install.packages('reticulate', repos='https://  
packagemanager.rstudio.com/cran/_linux_/centos7/latest')" \  
&& /opt/python/${PYTHON_VERSION}/bin/pip install --upgrade \  
'boto3>1.0<2.0' \  
'awscli>1.0<2.0' \  
'sagemaker[local]<3'
```

- You must provide default values for the RSTUDIO\_CONNECT\_URL and RSTUDIO\_PACKAGE\_MANAGER\_URL environment values.

```
ENV RSTUDIO_CONNECT_URL "YOUR_CONNECT_URL"  
ENV RSTUDIO_PACKAGE_MANAGER_URL "YOUR_PACKAGE_MANAGER_URL"  
ENV RSTUDIO_FORCE_NON_ZERO_EXIT_CODE 1
```

The following general specifications apply to the image that is represented by an RStudio image version.

## Running the image

ENTRYPOINT and CMD instructions are overridden so that the image is run as an RSession application.

## Stopping the image

The DeleteApp API issues the equivalent of a docker stop command. Other processes in the container won't get the SIGKILL/SIGTERM signals.

## File system

The /opt/.sagemakerinternal and /opt/ml directories are reserved. Any data in these directories might not be visible at runtime.

## User data

Each user in a SageMaker AI domain gets a user directory on a shared Amazon Elastic File System volume in the image. The location of the current user's directory on the Amazon Elastic File System volume is /home/sagemaker-user.

## Metadata

A metadata file is located at /opt/ml/metadata/resource-metadata.json. No additional environment variables are added to the variables defined in the image. For more information, see [Get App Metadata](#).

## GPU

On a GPU instance, the image is run with the --gpus option. Only the CUDA toolkit should be included in the image, not the NVIDIA drivers. For more information, see [NVIDIA User Guide](#).

## Metrics and logging

Logs from the RSession process are sent to Amazon CloudWatch in the customer's account. The name of the log group is /aws/sagemaker/studio. The name of the log stream is \$domainID/\$userProfileName/RSession/\$appName.

## Image size

Image size is limited to 25 GB. To view the size of your image, run `docker image ls`.

## Create a custom RStudio image

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

This topic describes how you can create a custom RStudio image using the SageMaker AI console and the AWS CLI. If you use the AWS CLI, you must run the steps from your local machine. The following steps do not work from within Amazon SageMaker Studio Classic.

When you create an image, SageMaker AI also creates an initial image version. The image version represents a container image in [Amazon Elastic Container Registry \(ECR\)](#). The container image must satisfy the requirements to be used in RStudio. For more information, see [Custom RStudio image specifications](#).

For information about testing your image locally and resolving common issues, see the [SageMaker Studio Custom Image Samples repo](#).

## Topics

- [Add a SageMaker AI-compatible RStudio Docker container image to Amazon ECR](#)
- [Create a SageMaker image from the console](#)
- [Create an image from the AWS CLI](#)

## Add a SageMaker AI-compatible RStudio Docker container image to Amazon ECR

Use the following steps to add a Docker container image to Amazon ECR:

- Create an Amazon ECR repository.
- Authenticate to Amazon ECR.
- Build a SageMaker AI-compatible RStudio Docker image.
- Push the image to the Amazon ECR repository.

### Note

The Amazon ECR repository must be in the same AWS Region as your domain.

## To build and add a Docker image to Amazon ECR

1. Create an Amazon ECR repository using the AWS CLI. To create the repository using the Amazon ECR console, see [Creating a repository](#).

```
aws ecr create-repository \
    --repository-name rstudio-custom \
    --image-scanning-configuration scanOnPush=true
```

Response:

```
{
  "repository": {
    "repositoryArn": "arn:aws:ecr:us-east-2:acct-id:repository/rstudio-custom",
    "registryId": "acct-id",
    "repositoryName": "rstudio-custom",
    "repositoryUri": "acct-id.dkr.ecr.us-east-2.amazonaws.com/rstudio-custom",
    ...
  }
}
```

```
}
```

2. Authenticate to Amazon ECR using the repository URI returned as a response from the `create-repository` command. Make sure that the Docker application is running. For more information, see [Registry Authentication](#).

```
aws ecr get-login-password | \
    docker login --username AWS --password-stdin <repository-uri>
```

Response:

```
Login Succeeded
```

3. Build the Docker image. Run the following command from the directory that includes your Dockerfile.

```
docker build .
```

4. Tag your built image with a unique tag.

```
docker tag <image-id> "<repository-uri>:<tag>"
```

5. Push the container image to the Amazon ECR repository. For more information, see [ImagePush](#) and [Pushing an image](#).

```
docker push <repository-uri>:<tag>
```

Response:

```
The push refers to repository [<account-id>.dkr.ecr.us-east-2.amazonaws.com/
rstudio-custom]
r: digest: <digest> size: 3066
```

## Create a SageMaker image from the console

### To create an image

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.

3. Under **Admin configurations**, choose **Images**.
4. On the **Custom images** page, choose **Create image**.
5. For **Image source**, enter the registry path to the container image in Amazon ECR. The path is in the following format:

*acct-id.dkr.ecr.region.amazonaws.com/repo-name[:tag] or [@digest]*

6. Choose **Next**.
7. Under **Image properties**, enter the following:
  - Image name – The name must be unique to your account in the current AWS Region.
  - (Optional) Image display name – The name displayed in the domain user interface. When not provided, Image name is displayed.
  - (Optional) Description – A description of the image.
  - IAM role – The role must have the [AmazonSageMakerFullAccess](#) policy attached. Use the dropdown menu to choose one of the following options:
    - Create a new role – Specify any additional Amazon Simple Storage Service (Amazon S3) buckets that you want your notebooks users to access. If you don't want to allow access to additional buckets, choose **None**.

SageMaker AI attaches the [AmazonSageMakerFullAccess](#) policy to the role. The role allows your notebook users to access the Amazon S3 buckets listed next to the check marks.

- Enter a custom IAM role ARN – Enter the Amazon Resource Name (ARN) of your IAM role.
  - Use existing role – Choose one of your existing roles from the list.
  - (Optional) Image tags – Choose **Add new tag**. You can add up to 50 tags. Tags are searchable using the SageMaker AI console or the SageMaker AI Search API.
8. Under **Image type**, select RStudio image.
  9. Choose **Submit**.

The new image is displayed in the **Custom images** list and briefly highlighted. After the image has been successfully created, you can choose the image name to view its properties or choose **Create version** to create another version.

## To create another image version

1. Choose **Create version** on the same row as the image.
2. For **Image source**, enter the registry path to the Amazon ECR image. The image shouldn't be the same image as used in a previous version of the SageMaker AI image.

To use the custom image in RStudio, you must attach it to your domain. For more information, see [Attach a custom SageMaker image](#).

## Create an image from the AWS CLI

This section shows how to create a custom Amazon SageMaker image using the AWS CLI.

Use the following steps to create a SageMaker image:

- Create an Image.
- Create an ImageVersion.
- Create a configuration file.
- Create an AppImageConfig.

## To create the SageMaker image entities

1. Create a SageMaker image. The role ARN must have at least the `AmazonSageMakerFullAccessPolicy` policy attached.

```
aws sagemaker create-image \
--image-name rstudio-custom-image \
--role-arn arn:aws:iam::<acct-id>:role/service-role/<execution-role>
```

Response:

```
{  
    "ImageArn": "arn:aws:sagemaker:us-east-2:acct-id:image/rstudio-custom-image"  
}
```

2. Create a SageMaker image version from the image. Pass the unique tag value that you chose when you pushed the image to Amazon ECR.

```
aws sagemaker create-image-version \
```

```
--image-name rstudio-custom-image \
--base-image <repository-uri>:<tag>
```

Response:

```
{  
    "ImageVersionArn": "arn:aws:sagemaker:us-east-2:acct-id:image-version/rstudio-image/1"  
}
```

3. Check that the image version was successfully created.

```
aws sagemaker describe-image-version \
--image-name rstudio-custom-image \
--version 1
```

Response:

```
{  
    "ImageVersionArn": "arn:aws:sagemaker:us-east-2:acct-id:image-version/rstudio-custom-image/1",  
    "ImageVersionStatus": "CREATED"  
}
```

### Note

If the response is "ImageVersionStatus": "CREATED\_FAILED", the response also includes the failure reason. A permissions issue is a common cause of failure. You also can check your Amazon CloudWatch Logs. The name of the log group is /aws/sagemaker/studio. The name of the log stream is \$domainID/\$userProfileName/KernelGateway/\$appName.

4. Create a configuration file, named `app-image-config-input.json`. The app image config is used to configuration for running a SageMaker image as a Kernel Gateway application.

```
{  
    "AppImageConfigName": "rstudio-custom-config"  
}
```

5. Create the AppImageConfig using the file that you created in the previous step.

```
aws sagemaker create-app-image-config \
--cli-input-json file://app-image-config-input.json
```

Response:

```
{  
    "AppImageConfigArn": "arn:aws:sagemaker:us-east-2:acct-id:app-image-config/r-  
    image-config"  
}
```

## Attach a custom SageMaker image

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

This guide shows how to attach a custom RStudio image to your Amazon SageMaker AI domain using the SageMaker AI console or the AWS Command Line Interface (AWS CLI).

To use a custom SageMaker image, you must attach a custom RStudio image to your domain. When you attach an image version, it appears in the RStudio Launcher and is available in the **Select image** dropdown list. You use the dropdown to change the image used by RStudio.

There is a limit to the number of image versions that you can attach. After you reach the limit, you must first detach a version so that you can attach a different version of the image.

## Topics

- [Attach an image version to your domain using the console](#)
- [Attach an existing image version to your domain using the AWS CLI](#)

## Attach an image version to your domain using the console

You can attach a custom SageMaker image version to your domain using the SageMaker AI console's control panel. You can also create a custom SageMaker image, and an image version, and then attach that version to your domain.

### To attach an existing image

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. Select the desired domain.
5. Choose **Environment**.
6. Under **Custom SageMaker Studio Classic images attached to domain**, choose **Attach image**.
7. For **Image source**, choose **Existing image or New image**.

If you select **Existing image**, choose an image from the Amazon SageMaker image store.

If you select **New image**, provide the Amazon ECR registry path for your Docker image. The path must be in the same AWS Region as the domain. The Amazon ECR repo must be in the same account as your domain, or cross-account permissions for SageMaker AI must be enabled.

8. Choose an existing image from the list.
9. Choose a version of the image from the list.
10. Choose **Next**.
11. Enter values for **Image name**, **Image display name**, and **Description**.
12. Choose the IAM role. For more information, see [Create a custom RStudio image](#).
13. (Optional) Add tags for the image.
14. (Optional) Choose **Add new tag**, then add a configuration tag.
15. For **Image type**, select **RStudio Image**.
16. Choose **Submit**.

Wait for the image version to be attached to the domain. After the version is attached, it appears in the **Custom images** list and is briefly highlighted.

## Attach an existing image version to your domain using the AWS CLI

Two methods are presented to attach the image version to your domain using the AWS CLI. In the first method, you create a new domain with the version attached. This method is simpler but you must specify the Amazon Virtual Private Cloud (Amazon VPC) information and execution role that's required to create the domain.

If you have already onboarded to the domain, you can use the second method to attach the image version to your current domain. In this case, you don't need to specify the Amazon VPC information and execution role. After you attach the version, delete all of the applications in your domain and relaunch RStudio.

### Attach the SageMaker image to a new domain

To use this method, you must specify an execution role that has the [AmazonSageMakerFullAccess](#) policy attached.

Use the following steps to create the domain and attach the custom SageMaker AI image:

- Get your default VPC ID and subnet IDs.
- Create the configuration file for the domain, which specifies the image.
- Create the domain with the configuration file.

### To add the custom SageMaker image to your domain

1. Get your default VPC ID.

```
aws ec2 describe-vpcs \
--filters Name=isDefault,Values=true \
--query "Vpcs[0].VpcId" --output text
```

Response:

```
vpc-xxxxxxxx
```

2. Get your default subnet IDs using the VPC ID from the previous step.

```
aws ec2 describe-subnets \
--filters Name=vpc-id,Values=<vpc-id> \
--query "Subnets[*].SubnetId" --output json
```

Response:

```
[  
    "subnet-b55171dd",  
    "subnet-8a5f99c6",  
    "subnet-e88d1392"  
]
```

3. Create a configuration file named `create-domain-input.json`. Insert the VPC ID, subnet IDs, `ImageName`, and `AppImageConfigName` from the previous steps. Because `ImageVersionNumber` isn't specified, the latest version of the image is used, which is the only version in this case. Your execution role must satisfy the requirements in [Complete prerequisites](#).

```
{  
    "DomainName": "domain-with-custom-r-image",  
    "VpcId": "<vpc-id>",  
    "SubnetIds": [  
        "<subnet-ids>"  
    ],  
    "DomainSettings": {  
        "RStudioServerProDomainSettings": {  
            "DomainExecutionRoleArn": "<execution-role>"  
        }  
    },  
    "DefaultUserSettings": {  
        "ExecutionRole": "<execution-role>",  
        "RSessionAppSettings": {  
            "CustomImages": [  
                {  
                    "AppImageConfigName": "rstudio-custom-config",  
                    "ImageName": "rstudio-custom-image"  
                }  
            ]  
        }  
    },  
    "AuthMode": "IAM"
```

}

#### 4. Create the domain with the attached custom SageMaker image.

```
aws sagemaker create-domain \
--cli-input-json file://create-domain-input.json
```

Response:

```
{  
    "DomainArn": "arn:aws:sagemaker:region:acct-id:domain/domain-id",  
    "Url": "https://domain-id.studio.region.sagemaker.aws/..."  
}
```

### Attach the SageMaker image to an existing domain

This method assumes that you've already onboarded to domain. For more information, see [Amazon SageMaker AI domain overview](#).

#### Note

You must delete all of the applications in your domain to update the domain with the new image version. For information about deleting these applications, see [Delete an Amazon SageMaker AI domain](#).

Use the following steps to add the SageMaker image to your current domain.

- Get your DomainID from the SageMaker AI console.
- Use the DomainID to get the DefaultUserSettings for the domain.
- Add the ImageName and AppImageConfig as a CustomImage to the DefaultUserSettings.
- Update your domain to include the custom image.

### To add the custom SageMaker image to your domain

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.

3. Under **Admin configurations**, choose **domains**.
4. Select the desired domain.
5. Choose **domain settings**.
6. Under **General Settings**, find the **domain ID**. The ID is in the following format: d-xxxxxxxxxxxx.
7. Use the domain ID to get the description of the domain.

```
aws sagemaker describe-domain \
--domain-id <d-xxxxxxxxxxxx>
```

Response:

```
{  
    "DomainId": "d-xxxxxxxxxxxx",  
    "DefaultUserSettings": {  
        "KernelGatewayAppSettings": {  
            "CustomImages": [  
                ],  
                ...  
            }  
        }  
    }
```

8. Save the **DefaultUserSettings** section of the response to a file named `update-domain-input.json`.
9. Insert the `ImageName` and `AppImageConfigName` from the previous steps as a custom image. Because `ImageVersionNumber` isn't specified, the latest version of the image is used, which is the only version in this case.

```
{  
    "DefaultUserSettings": {  
        "RSessionAppSettings": {  
            "CustomImages": [  
                {  
                    "ImageName": "rstudio-custom-image",  
                    "AppImageConfigName": "rstudio-custom-config"  
                }  
            ]  
        }  
    }
```

```
 }  
 }
```

10. Use the domain ID and default user settings file to update your domain.

```
aws sagemaker update-domain \  
  --domain-id <d-xxxxxxxxxxxx> \  
  --cli-input-json file://update-domain-input.json
```

Response:

```
{  
  "DomainArn": "arn:aws:sagemaker:region:acct-id:domain/domain-id"  
}
```

11. Delete the RStudioServerPro application. You must restart the RStudioServerPro domain-shared application for the RStudio Launcher UI to pick up the latest changes.

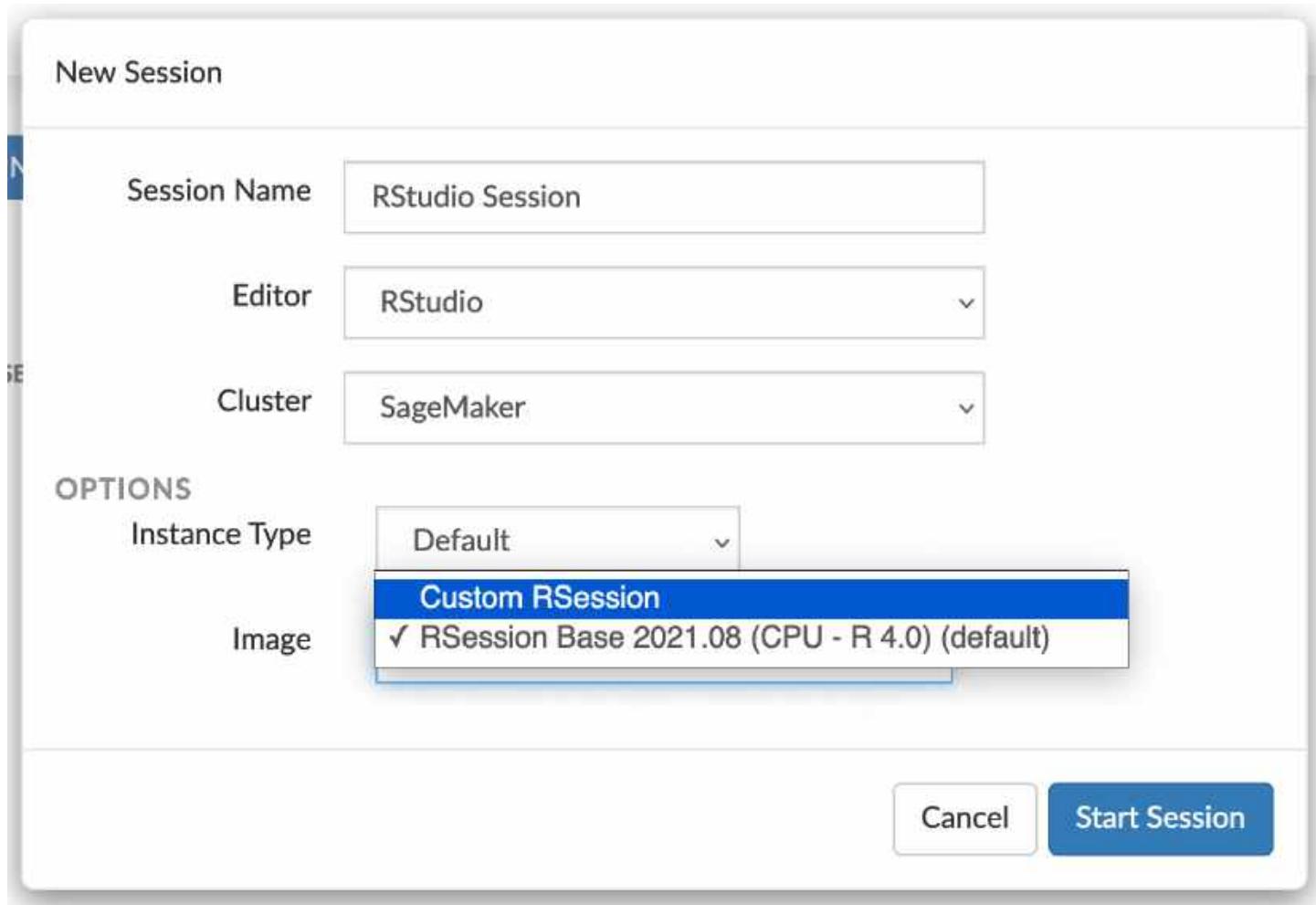
```
aws sagemaker delete-app \  
  --domain-id <d-xxxxxxxxxxxx> --user-profile-name domain-shared \  
  --app-type RStudioServerPro --app-name default
```

12. Create a new RStudioServerPro application. You must create this application using the AWS CLI.

```
aws sagemaker create-app \  
  --domain-id <d-xxxxxxxxxxxx> --user-profile-name domain-shared \  
  --app-type RStudioServerPro --app-name default
```

## Launch a custom SageMaker image in RStudio

You can use your custom image when launching an RStudio applicaton from the console. After you create your custom SageMaker image and attach it to your domain, the image appears in the image selector dialog box of the RStudio Launcher. To launch a new RStudio app, follow the steps in [Launch RSessions from the RStudio Launcher](#) and select your custom image as shown in the following image.



## Clean up image resources

This guide shows how to clean up RStudio image resources that you created in the previous sections. To delete an image, complete the following steps using either the SageMaker AI console or the AWS CLI, as shown in this guide.

- Detach the image and image versions from your Amazon SageMaker AI domain.
- Delete the image, image version, and app image config.

After you've completed these steps, you can delete the container image and repository from Amazon ECR. For more information about how to delete the container image and repository, see [Deleting a repository](#).

## Clean up resources from the SageMaker AI console

When you detach an image from a domain, all versions of the image are detached. When an image is detached, all users of the domain lose access to the image versions.

### To detach an image

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. Select the desired domain.
5. Choose **Environment**.
6. Under **Custom images attached to domain**, choose the image and then choose **Detach**.
7. (Optional) To delete the image and all versions from SageMaker AI, select **Also delete the selected images ....** This does not delete the associated images from Amazon ECR.
8. Choose **Detach**.

## Clean up resources from the AWS CLI

### To clean up resources

1. Detach the image and image versions from your domain by passing an empty custom image list to the domain. Open the update-domain-input.json file that you created in [Attach the SageMaker image to your current domain](#).
2. Delete the RSessionAppSettings custom images and then save the file. Do not modify the KernelGatewayAppSettings custom images.

```
{  
    "DomainId": "d-xxxxxxxxxxxxx",  
    "DefaultUserSettings": {  
        "KernelGatewayAppSettings": {  
            "CustomImages": [  
                ],  
                ...  
            },  
            "RSessionAppSettings": {  
                "CustomImages": [  
                    ],  
                    ...  
                },  
                "StudioAppSettings": {  
                    "CustomImages": [  
                        ],  
                        ...  
                    },  
                    "TrainingAppSettings": {  
                        "CustomImages": [  
                            ],  
                            ...  
                        }  
                }  
            }  
        }  
    }  
}
```

```
        "DefaultResourceSpec": {  
            }  
            ...  
        }  
    }  
}
```

### 3. Use the domain ID and default user settings file to update your domain.

```
aws sagemaker update-domain \  
    --domain-id <d-xxxxxxxxxxxx> \  
    --cli-input-json file://update-domain-input.json
```

Response:

```
{  
    "DomainArn": "arn:aws:sagemaker:us-east-2:acct-id:domain/d-xxxxxxxxxxxx"  
}
```

### 4. Delete the app image config.

```
aws sagemaker delete-app-image-config \  
    --app-image-config-name rstudio-image-config
```

### 5. Delete the SageMaker image, which also deletes all image versions. The container images in Amazon ECR that are represented by the image versions are not deleted.

```
aws sagemaker delete-image \  
    --image-name rstudio-image
```

## Create a user to use RStudio

### **⚠ Important**

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can

occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

After your RStudio-enabled Amazon SageMaker AI domain is running, you can add user profiles (UserProfiles) to the domain. The following topics show how to create user profiles that are authorized to use RStudio, as well as update an existing user profile. For information on how to delete an RStudio App, UserProfile, or domain, follow the steps in [Delete an Amazon SageMaker AI domain](#).

 **Note**

The limit for the total number of UserProfiles in a Amazon SageMaker AI domain is 60.

There are two types of users:

- Unauthorized: This user cannot access the RStudio app.
- Authorized: This user can access the RStudio app and use one of the RStudio license seats. By default, a new user is Authorized if the domain is enabled for RStudio.

Changing a user's authorization status is only valid from Unauthorized to Authorized. If a user is authorized, they can be given one of the following levels of access to RStudio.

- RStudio User: This is a standard RStudio user and can access RStudio.
- RStudio Admin: The admin of your Amazon SageMaker AI domain has the ability to create users, add existing users, and update the permissions of existing users. Admins can also access the RStudio Administrative dashboard. However, this admin is not able to update parameters that are managed by Amazon SageMaker AI.

## Methods to create a user

The following topics show how to create a user in your RStudio-enabled Amazon SageMaker AI domain.

## Create user console

To create a user in your RStudio-enabled Amazon SageMaker AI domain from the console, complete the steps in [Add user profiles](#).

## Create user CLI

The following command shows how to add users to a Amazon SageMaker AI domain with IAM authentication. A User can belong to either the R\_STUDIO\_USER or R\_STUDIO\_ADMIN User group.

```
aws sagemaker create-user-profile --region <REGION> \
--domain-id <DOMAIN-ID> \
--user-profile-name <USER_PROFILE_NAME-ID> \
--user-settings RStudioServerProAppSettings={UserGroup=<USER-GROUP>}
```

The following command shows how to add users to a Amazon SageMaker AI domain with authentication using IAM Identity Center. A user can belong to either the R\_STUDIO\_USER or R\_STUDIO\_ADMIN User group.

```
aws sagemaker create-user-profile --region <REGION> \
--domain-id <DOMAIN-ID> \
--user-profile-name <USER_PROFILE_NAME-ID> \
--user-settings RStudioServerProAppSettings={UserGroup=<USER-GROUP>} \
--single-sign-on-user-identifier UserName \
--single-sign-on-user-value <USER-NAME>
```

## Log in to RStudio as another user

The following topic demonstrates how to log in to RStudio on Amazon SageMaker AI as another user.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. On the left navigation pane, choose **Admin configurations**.
3. Under **Admin configurations**, choose **domains**.
4. Select the domain containing the user profile.
5. Select a user name from the list of users. This opens a new page with details about the user profile and the apps that are running.
6. Select **Launch**.
7. From the dropdown, select **RStudio** to launch an RStudio instance.

## Terminate sessions for another user

The following topic demonstrates how to terminate sessions for another user in RStudio on Amazon SageMaker AI.

1. From the list of running apps, identify the app you want to delete.
2. Click the respective **Delete app** button for the app you are deleting.

## Use the RStudio administrative dashboard

This topic shows how to access and use the RStudio administrative dashboard. With the RStudio administrative dashboard, admins can manage users and RSessions, as well as view information about RStudio Server instance utilization and Amazon CloudWatch Logs.

### Launch the RStudio administrative dashboard

The R\_STUDIO\_ADMIN authorization allows the user to access the RStudio administrative dashboard. An R\_STUDIO\_ADMIN user can access the RStudio administrative dashboard by replacing workspaces with admin in their RStudio URL manually. The following shows how to modify the URL to access the RStudio administrative dashboard.

For example, the following RStudio URL:

```
https://<DOMAIN-ID>.studio.us-east-2.sagemaker.aws/rstudio/default/s/<SESSION-ID>/  
workspaces
```

Can be converted to:

```
https://<DOMAIN-ID>.studio.us-east-2.sagemaker.aws/rstudio/default/s/<SESSION-ID>/admin
```

### Dashboard tab

This tab gives an overview of your RStudio Server instance utilization, as well as information on the number of active RSessions.

### Sessions tab

This tab gives information on the active RSessions, such as the user that launched the RSessions, the time that the RSessions have been running, and their resource utilization.

## Users tab

This tab gives information on the RStudio authorized users in the domain, such as the time that the last RSession was launched and their resource utilization.

## Stats tab

This tab gives information on the utilization of your RStudio Server instance.

## Logs tab

This tab displays Amazon CloudWatch Logs for the RStudio Server instance. For more information about logging events with Amazon CloudWatch Logs, see [What is Amazon CloudWatch Logs?](#)

## Shut down RStudio

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

To shut down and restart your Posit Workbench and the associated RStudioServerPro app, you must first shut down all of your existing RSessions. You can shut down the RSessionGateway apps from within RStudio. You can then shut down the RStudioServerPro app using the AWS CLI. After the RStudioServerPro app is shut down, you must reopen RStudio through the SageMaker AI console.

Any unsaved notebook information is lost in the process. The user data in the Amazon EFS volume isn't impacted.

### Note

If you are using a custom image with RStudio, ensure that your docker image is using an RStudio version that is compatible with the version of Posit Workbench being used by SageMaker AI after you restart your RStudioServerPro app.

The following topics show how to shut down the RSessionGateway and RStudioServerPro apps and restart them.

## Suspend your RSessions

Complete the following procedure to suspend all of your RSessions.

1. From the RStudio Launcher, identify the RSession that you want to suspend.
2. Select **Suspend** for the session.
3. Repeat this for all RSessions.

## Delete your RSessions

Complete the following procedure to shut down all of your RSessions.

1. From the RStudio Launcher, identify the RSession that you want to delete.
2. Select **Quit** for the session. This opens a new **Quit Session** window.
3. From the **Quit Session** window, select **Force Quit**, to end all child processes in the session.
4. Select **Quit Session** to confirm deletion of the session.
5. Repeat this for all RSessions.

## Delete your RStudioServerPro app

Run the following commands from the AWS CLI to delete and restart your RStudioServerPro app.

1. Delete the RStudioServerPro application by using your current domain id.

```
aws sagemaker delete-app \
--domain-id <domainId> \
--user-profile-name domain-shared \
```

```
--app-type RStudioServerPro \
--app-name default
```

## 2. Re-create the RStudioServerPro application.

```
aws sagemaker create-app \
--domain-id <domainId> \
--user-profile-name domain-shared \
--app-type RStudioServerPro \
--app-name default
```

## Billing and cost

To track the costs associated with your RStudio environment, you can use the AWS Billing and Cost Management service. AWS Billing and Cost Management provides useful tools to help you gather information related to your cost and usage, analyze your cost drivers and usage trends, and take action to budget your spending. For more information, see [What is AWS Billing and Cost Management?](#). The following describes components required to run RStudio on Amazon SageMaker AI and how each component factors into billing for your RStudio instance.

- RStudio License –You must purchase an RStudio license. There is no additional charge for using your RStudio license with Amazon SageMaker AI. For more information about your RStudio license, see [Get an RStudio license](#).
- RSession - These are RStudio working sessions launched by end users. You are charged while the RSession is running.
- RStudio Server - A multi-tenant server manages all the RSessions. You can choose the instance type to run RStudio Server on, and pay the related costs. The default instance, "system", is free, but you can choose to pay for higher tiers. For more information about the available instance types for your RStudio Server, see [RStudioServerPro instance type](#).

## Tracking billing at user level

To track billing at the user level using Cost Allocation Tags, see [Using Cost Allocation Tags](#).

## Diagnose issues and get support

The following sections describe how to diagnose issues with RStudio on Amazon SageMaker AI. To get support for RStudio on Amazon SageMaker AI, contact Amazon SageMaker AI support.

For help with purchasing an RStudio license or modifying the number of license seats, contact [sales@rstudio.com](mailto:sales@rstudio.com).

## Upgrade your version

If you receive a warning that there is a version mismatch between your RSession and RStudioServerPro apps, then you must upgrade the version of your RStudioServerPro app. For more information, see [RStudio Versioning](#).

## View Metrics and Logs

You can monitor your workflow performance while using RStudio on Amazon SageMaker AI. View data logs and information about metrics with the RStudio administrative dashboard or Amazon CloudWatch.

### View your RStudio logs from the RStudio administrative dashboard

You can view metrics and logs directly from the RStudio administrative dashboard.

1. Log in to your **Amazon SageMaker AI domain**.
2. Navigate to the RStudio administrative dashboard following the steps in [Use the RStudio administrative dashboard](#).
3. Select the **Logs** tab.

### View your RStudio logs from Amazon CloudWatch Logs

Amazon CloudWatch monitors your AWS resources and the applications that you run on AWS in real time. You can use Amazon CloudWatch to collect and track metrics, which are variables that you can measure for your resources and applications. To ensure that your RStudio apps have permissions for Amazon CloudWatch, you must include the permissions described in [Amazon SageMaker AI domain overview](#). You don't need to do any setup to gather Amazon CloudWatch Logs.

The following steps show how to view Amazon CloudWatch Logs for your RSession.

These logs can be found in the /aws/sagemaker/studio log stream from the AWS CloudWatch console.

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. Select Logs from the left side. From the dropdown menu, select Log groups.

3. On the Log groups screen, search for aws/sagemaker/studio. Select the Log group.
4. On the aws/sagemaker/studio Log group screen, navigate to the Log streams tab.
5. To find the logs for your domain, search Log streams using the following format:

```
<DomainId>/domain-shared/rstudioserverpro/default
```

## RStudio on Amazon SageMaker AI user guide

With RStudio support in Amazon SageMaker AI, you can put your production workflows in place and take advantage of SageMaker AI features. The following topics show how to launch an RStudio session and complete key workflows. For information about managing RStudio on SageMaker AI, see [RStudio on Amazon SageMaker AI management](#).

For information about the onboarding steps to create an Amazon SageMaker AI domain with RStudio enabled, see [Amazon SageMaker AI domain overview](#).

For information about the AWS Regions that RStudio on SageMaker AI is supported in, see [Supported Regions and Quotas](#).

### Topics

- [Collaborate in RStudio](#)
- [Base R image](#)
- [RSession application colocation](#)
- [Launch RSessions from the RStudio Launcher](#)
- [Suspend your RSessions](#)
- [Delete your RSessions](#)
- [RStudio Connect](#)
- [Amazon SageMaker AI feature integration with RStudio on Amazon SageMaker AI](#)

## Collaborate in RStudio

To share your RStudio project, you can connect RStudio to your Git repo. For information on setting this up, see [Version Control with Git and SVN](#).

Note: Project sharing and realtime collaboration are not currently supported when using RStudio on Amazon SageMaker AI.

## Base R image

When launching your RStudio instance, the Base R image serves as the basis of your instance. This image extends the [r-session-complete](#) Docker image.

This Base R image includes the following:

- R v4.0 or higher
- awscli, sagemaker, and boto3 Python packages
- [Reticulate](#) package for R SDK integration

## RSession application colocation

Users can create multiple RSession applications on the same instance. Each instance type supports up to four colocated RSession applications. This applies to each user independently. For example, if two users create applications, then SageMaker AI allocates different underlying instances to each user. Each of these instances would support 4 RSession applications.

Customers only pay for the instance type used regardless of how many Rsession applications are running on the instance. If a user creates an RSession with a different associated instance type, then a new underlying instance is created.

## Launch RSessions from the RStudio Launcher

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

The following sections show how to use the RStudio Launcher to launch RSessions. They also include information about how to open the RStudio Launcher when using RStudio on Amazon SageMaker AI.

## Open RStudio Launcher

Open the RStudio launcher using the following set of procedures that matches your environment.

### Open RStudio Launcher from the Amazon SageMaker AI Console

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation, select **RStudio**.
3. Under **Get Started**, select the domain and user profile to launch.
4. Choose **Launch RStudio**.

### Open RStudio Launcher from Amazon SageMaker Studio

1. Navigate to Studio following the steps in [Launch Amazon SageMaker Studio](#).
2. Under **Applications**, select **RStudio**.
3. From the RStudio landing page, choose **Launch application**.

### Open RStudio Launcher from the AWS CLI

The procedure to open the RStudio Launcher using the AWS CLI differs depending on the method used to manage your users.

#### IAM Identity Center

1. Use the AWS access portal to open your Amazon SageMaker AI domain.
2. Modify the URL path to “/rstudio/default” as follows.

```
#Studio URL  
https://<domain-id>.studio.<region>.sagemaker.aws/jupyter/default/lab  
  
#modified URL  
https://<domain-id>.studio.<region>.sagemaker.aws/rstudio/default
```

#### IAM

To open the RStudio Launcher from the AWS CLI in IAM mode, complete the following procedure.

1. Create a presigned URL using the following command.

```
aws sagemaker create-presigned-domain-url --region <REGION> \  
--domain-id <DOMAIN-ID> \  
--user-profile-name <USER-PROFILE-NAME>
```

2. Append *&redirect=RStudioServerPro* to the generated URL.

3. Navigate to the updated URL.

## Launch RSessions

After you've launched the RStudio Launcher, you can create a new RSesssion.

1. Select **New Session**.
2. Enter a **Session Name**.
3. Select an instance type that your RSesssion runs on. This defaults to `ml.t3.medium`.
4. Select an Image that your RSesssion uses as the kernel.
5. Select Start Session.
6. After your session has been created, you can start it by selecting the name.

### Note

If you receive a warning that there is a version mismatch between your RSesssion and RStudioServerPro apps, then you must upgrade the version of your RStudioServerPro app. For more information, see [RStudio Versioning](#).

## Suspend your RSessions

The following procedure demonstrates how to suspend an RSesssion from the RStudio Launcher when using RStudio on Amazon SageMaker AI. For information about accessing the RStudio Launcher, see [Launch RSessions from the RStudio Launcher](#).

1. From the RStudio Launcher, identify the RSesssion that you want to suspend.
2. Select **Suspend** for the session.

## Delete your RSessions

The following procedure demonstrates how to delete an RSession from the RStudio Launcher when using RStudio on Amazon SageMaker AI. For information about accessing the RStudio Launcher, see [Launch RSessions from the RStudio Launcher](#).

1. From the RStudio Launcher, identify the RSession that you want to delete.
2. Select **Quit** for the session. This opens a new **Quit Session** window.
3. From the **Quit Session** window, select **Force Quit**, to end all child processes in the session.
4. Select **Quit Session** to confirm deletion of the session.

## RStudio Connect

RStudio Connect enables data scientists to publish insights, dashboard and web applications from RStudio on Amazon SageMaker AI. For more information, see [Host RStudio Connect and Package Manager for ML development in RStudio on Amazon SageMaker AI](#).

For more information on RStudio Connect, see the [RStudio Connect User Guide](#).

## Amazon SageMaker AI feature integration with RStudio on Amazon SageMaker AI

One of the benefits of using RStudio on Amazon SageMaker AI is the integration of Amazon SageMaker AI features. This includes integration with Amazon SageMaker Studio Classic and Reticulate. The following gives information about these integrations and examples for using them.

### Use Amazon SageMaker Studio Classic and RStudio on Amazon SageMaker AI

Your Amazon SageMaker Studio Classic and RStudio instances share the same Amazon EFS file system. This means that files that you import and create using Studio Classic can be accessed using RStudio and vice versa. This allows you to work on the same files using both Studio Classic and RStudio without having to move your files between the two. For more information on this workflow, see the [Announcing Fully Managed RStudio on Amazon SageMaker AI for Data Scientists](#) blog.

### Use Amazon SageMaker SDK with reticulate

The [reticulate](#) package is used as an R interface to [Amazon SageMaker Python SDK](#) to make API calls to Amazon SageMaker. The reticulate package translates between R and Python objects, and Amazon SageMaker AI provides a serverless data science environment to train and deploy

Machine Learning (ML) models at scale. For general information about the reticulate package, see [R Interface to Python](#).

For a blog that outlines how to use the reticulate package with Amazon SageMaker AI, see [Using R with Amazon SageMaker AI](#).

The following examples show how to use reticulate for specific use cases.

- For a notebook that describes how to use reticulate to do batch transform to make predictions, see [Batch Transform Using R with Amazon SageMaker AI](#).
- For a notebook that describes how to use reticulate to conduct hyperparameter tuning and generate predictions, see [Hyperparameter Optimization Using R with Amazon SageMaker AI](#).

## Code Editor in Amazon SageMaker Studio

Code Editor, based on [Code-OSS, Visual Studio Code - Open Source](#), helps you write, test, debug, and run your analytics and machine learning code. Code Editor extends and is fully integrated with Amazon SageMaker Studio. It also supports integrated development environment (IDE) extensions available in the [Open VSX Registry](#). The following page gives information about Code Editor and key details for using it.

Code Editor has the [AWS Toolkit for VS Code](#) extension pre-installed, which enables connections to AWS services such as [Amazon CodeWhisperer](#), a general purpose, machine learning-powered code generator that provides code recommendations in real time. For more information about extensions, see [Code Editor Connections and Extensions](#).

### **Important**

As of November 30, 2023, the previous Amazon SageMaker Studio experience is now named Amazon SageMaker Studio Classic. The following section is specific to using the updated Studio experience. For information about using the Studio Classic application, see [Amazon SageMaker Studio Classic](#).

To launch Code Editor, create a Code Editor private space. The Code Editor space uses a single Amazon Elastic Compute Cloud (Amazon EC2) instance for your compute and a single Amazon Elastic Block Store (Amazon EBS) volume for your storage. Everything in your space such as your code, Git profile, and environment variables are stored on the same Amazon EBS volume. The

volume has 3000 IOPS and a throughput of 125 MBps. Your administrator has configured the default Amazon EBS storage settings for your space.

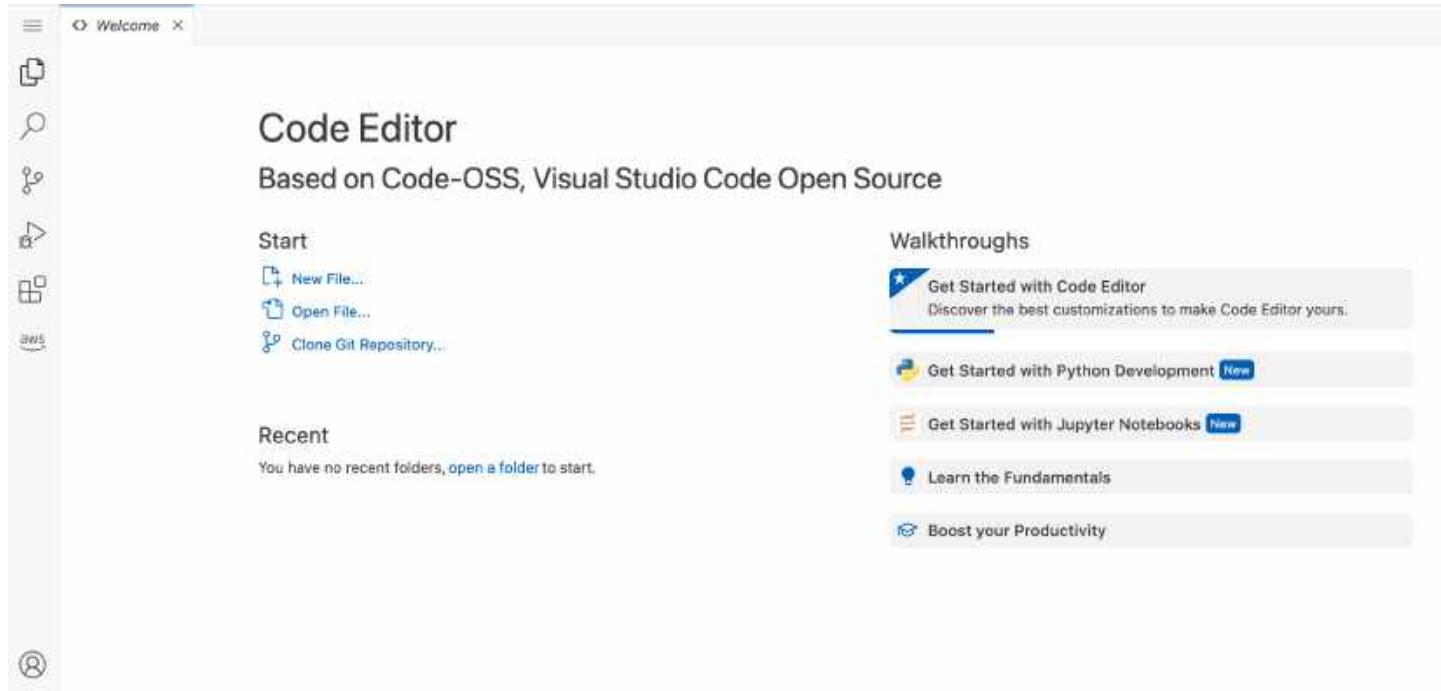
The default storage size is 5 GB, but your administrator can increase the amount of space you get. For more information, see [Change the default storage size](#).

The working directory of your users within the storage volume is `/home/sagemaker-user`. If you specify your own AWS KMS key to encrypt the volume, everything in the working directory is encrypted using your customer managed key. If you don't specify an AWS KMS key, the data inside `/home/sagemaker-user` is encrypted with an AWS managed key. Regardless of whether you specify an AWS KMS key, all of the data outside of the working directory is encrypted with an AWS Managed Key.

You can scale your compute up or down by changing the Amazon EC2 instance type that runs your Code Editor application. Before you change the associated instance type, you must first stop your Code Editor space. For more information, see [Code Editor application instances and images](#).

Your administrator might provide you with a lifecycle configuration to customize your environment. You can specify the lifecycle configuration when you create the space. For more information, see [Code Editor lifecycle configurations](#).

You can also bring your own file storage system if you have an Amazon EFS volume.



## Topics

- [Using the Code Editor](#)
- [Code Editor administrator guide](#)

## Using the Code Editor

The topics in this section provide guides for using Code Editor, including how to launch, add connections to AWS services, shut down resources, and more. After creating a Code Editor space, you can access your Code Editor session directly through the browser.

Within your Code Editor environment, you can do the following:

- Access all artifacts persisted in your home directory
- Clone your GitHub repositories and commit changes
- Access the SageMaker Python SDK

You can return to Studio to review any assets created in your Code Editor environment such as experiments, pipelines, or training jobs.

### Topics

- [Check the version of Code Editor](#)
- [Code Editor application instances and images](#)
- [Launch a Code Editor application in Studio](#)
- [Launch a Code Editor application using the AWS CLI](#)
- [Clone a repository in Code Editor](#)
- [Code Editor Connections and Extensions](#)
- [Shut down Code Editor resources](#)

## Check the version of Code Editor

The following steps show how to check the version of your Code Editor application.

### To check the Code Editor application version

1. Launch and run a Code Editor space and navigate to the Code Editor application UI. For more information, see [Launch a Code Editor application in Studio](#).

2. In the upper-left corner of the Code Editor UI, choose the menu button



).

Then, choose **Help**. Then, choose **About**.

## Code Editor application instances and images

Only some instances are compatible with Code Editor applications. You can choose the instance type that is compatible with your use case from the **Instance** dropdown menu.

The **Fast launch** instances start up much faster than the other instances. For more information about fast launch instance types in Studio, [Instance types available for use with Studio Classic](#).

### Note

If you use a GPU instance type when configuring your Code Editor application, you must also use a GPU-based image. The Code Editor space UI automatically selects a compatible image when you select your instance type.

Within a space, your data is stored in an Amazon EBS volume that persists independently from the life of an instance. You won't lose your data when you change instances. If your Code Editor space is Running, you must stop your space before changing instance types.

The following table lists the ARNs of the available Code Editor CPU and GPU images for each Region.

Region	CPU	GPU
us-east-1	arn:aws:sagemaker:us-east-1:885854791233:image/sagemaker-distribution-cpu	arn:aws:sagemaker:us-east-1:885854791233:image/sagemaker-distribution-gpu
us-east-2	arn:aws:sagemaker:us-east-2:37914896644:image/sagemaker-distribution-cpu	arn:aws:sagemaker:us-east-2:37914896644:image/sagemaker-distribution-gpu

Region	CPU	GPU
us-west-1	arn:aws:sagemaker:us-west-1:053634841547:image/sagemaker-distribution-cpu	arn:aws:sagemaker:us-west-1:053634841547:image/sagemaker-distribution-gpu
us-west-2	arn:aws:sagemaker:us-west-2:542918446943:image/sagemaker-distribution-cpu	arn:aws:sagemaker:us-west-2:542918446943:image/sagemaker-distribution-gpu
af-south-1	arn:aws:sagemaker:af-south-1:238384257742:image/sagemaker-distribution-cpu	arn:aws:sagemaker:af-south-1:238384257742:image/sagemaker-distribution-gpu
ap-east-1	arn:aws:sagemaker:ap-east-1:523751269255:image/sagemaker-distribution-cpu	arn:aws:sagemaker:ap-east-1:523751269255:image/sagemaker-distribution-gpu
ap-south-1	arn:aws:sagemaker:ap-south-1:245090515133:image/sagemaker-distribution-cpu	arn:aws:sagemaker:ap-south-1:245090515133:image/sagemaker-distribution-gpu
ap-northeast-2	arn:aws:sagemaker:ap-northeast-2:064688005998:image/sagemaker-distribution-cpu	arn:aws:sagemaker:ap-northeast-2:064688005998:image/sagemaker-distribution-gpu
ap-southeast-1	arn:aws:sagemaker:ap-southeast-1:022667117163:image/sagemaker-distribution-cpu	arn:aws:sagemaker:ap-southeast-1:022667117163:image/sagemaker-distribution-gpu
ap-southeast-2	arn:aws:sagemaker:ap-southeast-2:648430277019:image/sagemaker-distribution-cpu	arn:aws:sagemaker:ap-southeast-2:648430277019:image/sagemaker-distribution-gpu
ap-northeast-1	arn:aws:sagemaker:ap-northeast-1:010972774902:image/sagemaker-distribution-cpu	arn:aws:sagemaker:ap-northeast-1:010972774902:image/sagemaker-distribution-gpu

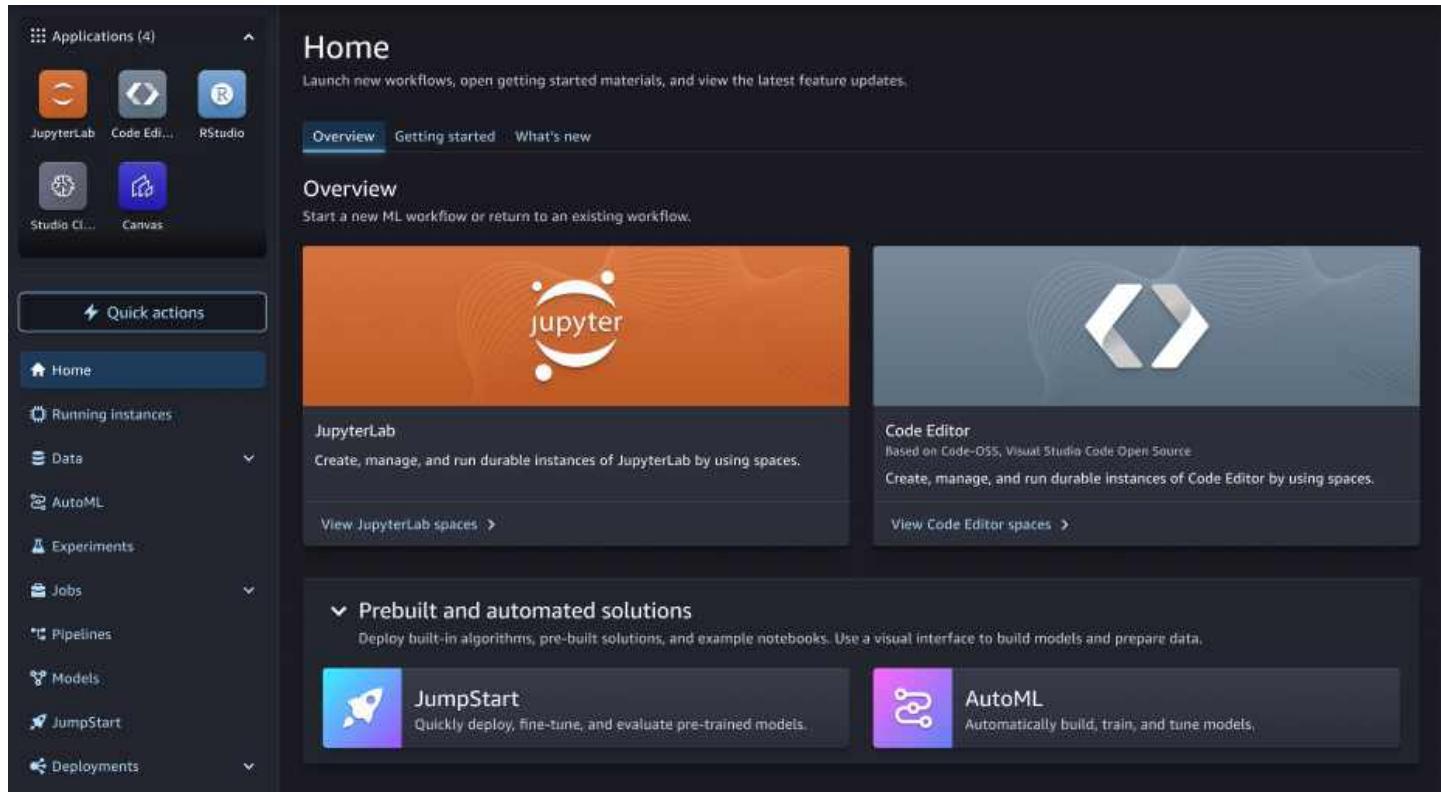
Region	CPU	GPU
ca-central-1	arn:aws:sagemaker:ca-centra l-1:481561238223:image/ sagemaker-distribution-cpu	arn:aws:sagemaker:ca-centra l-1:481561238223:image/ sagemaker-distribution-gpu
eu-central-1	arn:aws:sagemaker:eu-centra l-1:545423591354:image/ sagemaker-distribution-cpu	arn:aws:sagemaker:eu-centra l-1:545423591354:image/ sagemaker-distribution-gpu
eu-west-1	arn:aws:sagemaker:eu-west-1 :819792524951:image/ sagemaker-distribution-cpu	arn:aws:sagemaker:eu-west-1 :819792524951:image/ sagemaker-distribution-gpu
eu-west-2	arn:aws:sagemaker:eu-west-2 :021081402939:image/ sagemaker-distribution-cpu	arn:aws:sagemaker:eu-west-2 :021081402939:image/ sagemaker-distribution-gpu
eu-west-3	arn:aws:sagemaker:eu-west-3 :856416204555:image/ sagemaker-distribution-cpu	arn:aws:sagemaker:eu-west-3 :856416204555:image/ sagemaker-distribution-gpu
eu-north-1	arn:aws:sagemaker:eu-north- 1:175620155138:image/ sagemaker-distribution-cpu	arn:aws:sagemaker:eu-north- 1:175620155138:image/ sagemaker-distribution-gpu
eu-south-1	arn:aws:sagemaker:eu-south- 1:810671768855:image/ sagemaker-distribution-cpu	arn:aws:sagemaker:eu-south- 1:810671768855:image/ sagemaker-distribution-gpu
sa-east-1	arn:aws:sagemaker:sa-east-1 :567556641782:image/ sagemaker-distribution-cpu	arn:aws:sagemaker:sa-east-1 :567556641782:image/ sagemaker-distribution-gpu
ap-northeast-3	arn:aws:sagemaker:ap-northe ast-3:564864627153:image/ sagemaker-distribution-cpu	arn:aws:sagemaker:ap-northe ast-3:564864627153:image/ sagemaker-distribution-gpu

Region	CPU	GPU
ap-southeast-3	arn:aws:sagemaker:ap-southeast-3:370607712162:image/sagemaker-distribution-cpu	arn:aws:sagemaker:ap-southeast-3:370607712162:image/sagemaker-distribution-gpu
me-south-1	arn:aws:sagemaker:me-south-1:523774347010:image/sagemaker-distribution-cpu	arn:aws:sagemaker:me-south-1:523774347010:image/sagemaker-distribution-gpu
me-central-1	arn:aws:sagemaker:me-central-1:358593528301:image/sagemaker-distribution-cpu	arn:aws:sagemaker:me-central-1:358593528301:image/sagemaker-distribution-gpu
il-central-1	arn:aws:sagemaker:il-central-1:080319125002:image/sagemaker-distribution-cpu	arn:aws:sagemaker:il-central-1:080319125002:image/sagemaker-distribution-gpu
cn-north-1	arn:aws:sagemaker:cn-north-1:674439102856:image/sagemaker-distribution-cpu	arn:aws:sagemaker:cn-north-1:674439102856:image/sagemaker-distribution-gpu
cn-northwest-1	arn:aws:sagemaker:cn-northwest-1:651871951035:image/sagemaker-distribution-cpu	arn:aws:sagemaker:cn-northwest-1:651871951035:image/sagemaker-distribution-gpu
us-gov-west-1	arn:aws:sagemaker:us-gov-west-1:300992924816:image/sagemaker-distribution-cpu	arn:aws:sagemaker:us-gov-west-1:300992924816:image/sagemaker-distribution-gpu
us-gov-east-1	arn:aws:sagemaker:us-gov-east-1:300993876623:image/sagemaker-distribution-cpu	arn:aws:sagemaker:us-gov-east-1:300993876623:image/sagemaker-distribution-gpu

If you encounter instance limits, contact your administrator. To get more storage and compute for a user, administrators can request an increase to a user's AWS quotas. For more information about requesting a quota increase, see [Amazon SageMaker AI endpoints and quotas](#).

## Launch a Code Editor application in Studio

To configure and access your Code Editor integrated development environment through Studio, you must create a Code Editor space. For more information about spaces in Studio, see [Amazon SageMaker Studio spaces](#).



The following procedure shows how to create and run a Code Editor space.

### To create and run a Code Editor space

1. Launch the updated Studio experience. For more information, see [Launch Amazon SageMaker Studio](#).
2. Do one of the following:
  - Within the updated Amazon SageMaker Studio UI, select **Code Editor** from the **Applications** menu.
  - Within the updated Amazon SageMaker Studio UI, choose **View Code Editor spaces** in the **Overview** section of the Studio homepage.
3. In the upper-right corner of the Code Editor landing page, choose **Create Code Editor space**.
4. Enter a name for your Code Editor space. The name must be 1–62 characters in length using letters, numbers, and dashes only.

5. Choose **Create space**.
6. After the space is created, you have some options before you choose to run the space:
  - You can edit the **Storage (GB)**, **Lifecycle Configuration**, or **Attach custom EFS file system** settings. Options for these settings are available based on administrator specification.
  - From the **Instance** dropdown menu, you can choose the instance type most compatible with your use case. From the **Image** dropdown menu, you can choose a SageMaker Distribution image or a custom image provided by your administrator.

 **Note**

Switching between sagemaker-distribution images changes the underlying version of Code Editor being used, which may cause incompatibilities due to browser caching. You should clear the browser cache when switching between images.

If you use a GPU instance type when configuring your Code Editor application, you must also use a GPU-based image. Within a space, your data is stored in an Amazon EBS volume that persists independently from the life of an instance. You won't lose your data when you change instances.

 **Important**

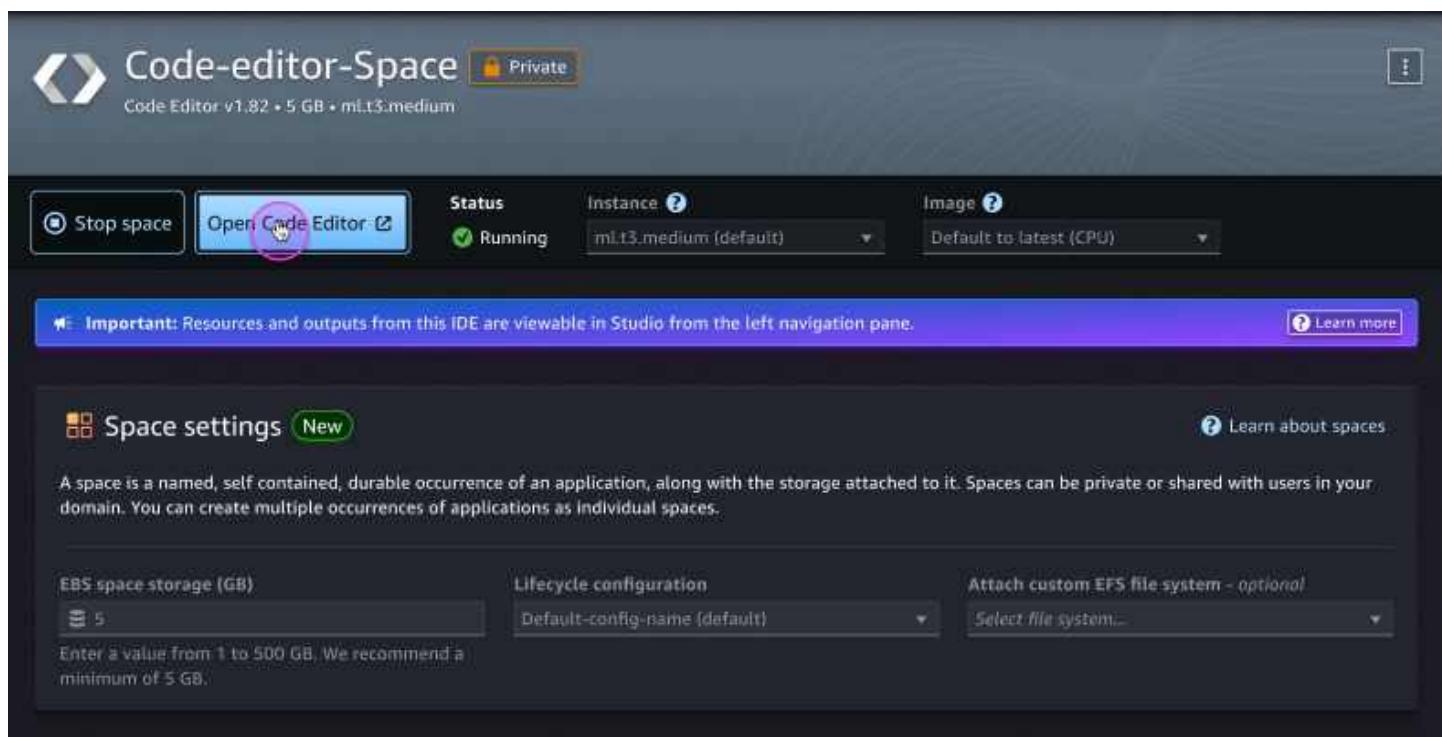
Custom IAM policies that allow Studio users to create spaces must also grant permissions to list images (`sagemaker: ListImage`) to view custom images. To add the permission, see [Add or remove identity permissions](#) in the *AWS Identity and Access Management User Guide*.

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker AI resources already include permissions to list images while creating those resources.

 **Note**

To update space settings, you must first stop your space. If your Code Editor uses an instance with NVMe instance stores, any data stored on the NVMe store is deleted when the space is stopped.

7. After updating your settings, choose **Run Space** in the space detail page.
8. After the status of the space is Running, choose **Open Code Editor** to go to your Code Editor session.



## Launch a Code Editor application using the AWS CLI

To configure and access your Code Editor integrated development environment through the AWS Command Line Interface (AWS CLI), you must create a Code Editor space. Be sure to meet the [Complete prerequisites](#) before going through the following steps. Use the following procedure to create and run a Code Editor space.

### To create and run a Code Editor space

1. Access a space using AWS Identity and Access Management (IAM) or AWS IAM Identity Center authentication. For more information about accessing spaces using the AWS CLI, see *Accessing spaces using the AWS Command Line Interface* in [Amazon SageMaker Studio spaces](#).
2. Create an application and specify CodeEditor as the app-type using the following command.

If you use a GPU instance type when creating your Code Editor application, you must also use a GPU-based image.

```
aws sagemaker create-app \
--domain-id domain-id \
--space-name space-name \
--app-type CodeEditor \
--app-name default \
--resource-spec "SageMakerImageArn=arn:aws:sagemaker:region:account-id:image/sagemaker-distribution-cpu"
```

For more information about available Code Editor image ARNs, see [Code Editor application instances and images](#).

3. After the Code Editor application is in service, launch the application using a presigned URL. You can use the `describe-app` API to check if your application is in service. Use the `create-presigned-domain-url` API to create a presigned URL:

```
aws sagemaker create-presigned-domain-url \
--domain-id domain-id \
--space-name space-name \
--user-profile-name user-profile-name \
--session-expiration-duration-in-seconds 43200 \
--landing-uri app:CodeEditor:
```

4. Open the generated URL to start working in your Code Editor application.

## Clone a repository in Code Editor

You can navigate through folders and clone a repository in the **Explorer** window of the Code Editor application UI.

To clone a repository, go through the following steps:

### To clone a repository

1. Open your Code Editor application in the browser, and choose the **Exploration** button



)

in the left navigation pane.

2. Choose **Clone Repository** in the **Explorer** window. Then, provide a repository URL or pick a repository source in the prompt.

3. Choose a folder to clone your repository into. Note that the default Code Editor folder is /home/sagemaker-user/. Cloning your repository may take some time.
4. To open the cloned repository, choose either **Open in New Window** or **Open**.
5. To return to the Code Editor application UI homepage, choose **Cancel**.
6. Within the repository, a prompt asks if you trust the authors of the files in your new repository. You have two choices:
  - a. To trust the folder and enable all features, choose **Yes, I trust the authors**.
  - b. To browse the repository content in *restricted mode*, choose **No, I don't trust the authors**.

In restricted mode, tasks are not allowed to run, debugging is disabled, workspace settings are not applied, and extensions have limited functionality.

To exit restricted mode, trust the authors of all files in your current folder or its parent folder, and enable all features, choose **Manage** in the **Restricted Mode** banner.

## Code Editor Connections and Extensions

Code Editor supports IDE connections to AWS services as well as extensions available in the [Open VSX Registry](#).

### Connections to AWS

Code Editor environments are integrated with the [AWS Toolkit for VS Code](#) to add connections to AWS services. To get started with connections to AWS services, you must have valid AWS Identity and Access Management (IAM) credentials. For more information, see [Authentication and access for the AWS Toolkit for Visual Studio Code](#).

Within your Code Editor environment, you can add connections to:

- [AWS Explorer](#) – View, modify, and deploy AWS resources in Amazon S3, CloudWatch, and more.

Accessing certain features within AWS Explorer requires certain AWS permissions. For more information, see [Authentication and access for the AWS Toolkit for Visual Studio Code](#).

- [Amazon CodeWhisperer](#) – Build applications faster with AI-powered code suggestions.

To use Amazon CodeWhisperer with Code Editor, you must add the following permissions to your SageMaker AI execution role.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "CodeWhispererPermissions",  
      "Effect": "Allow",  
      "Action": ["codewhisperer:GenerateRecommendations"],  
      "Resource": "*"  
    }  
  ]  
}
```

For more information, see [Creating IAM policies](#) and [Adding and removing IAM identity permissions](#) in the *IAM User Guide*.

## Extensions

Code Editor supports IDE extensions available in the [Open VSX Registry](#).

To get started with extensions in your Code Editor environment, choose the **Extensions** icon



)

in the left navigation pane. Here, you can configure connections to AWS by installing the AWS Toolkit. For more information, see [Installing the AWS Toolkit for Visual Studio Code](#).

In the search bar, you can search directly for additional extensions through the [Open VSX Registry](#), such as the AWS Toolkit, Jupyter, Python, and more.

## Shut down Code Editor resources

When you're finished using a Code Editor space, you can use Studio to stop it. That way, you stop incurring costs for the space.

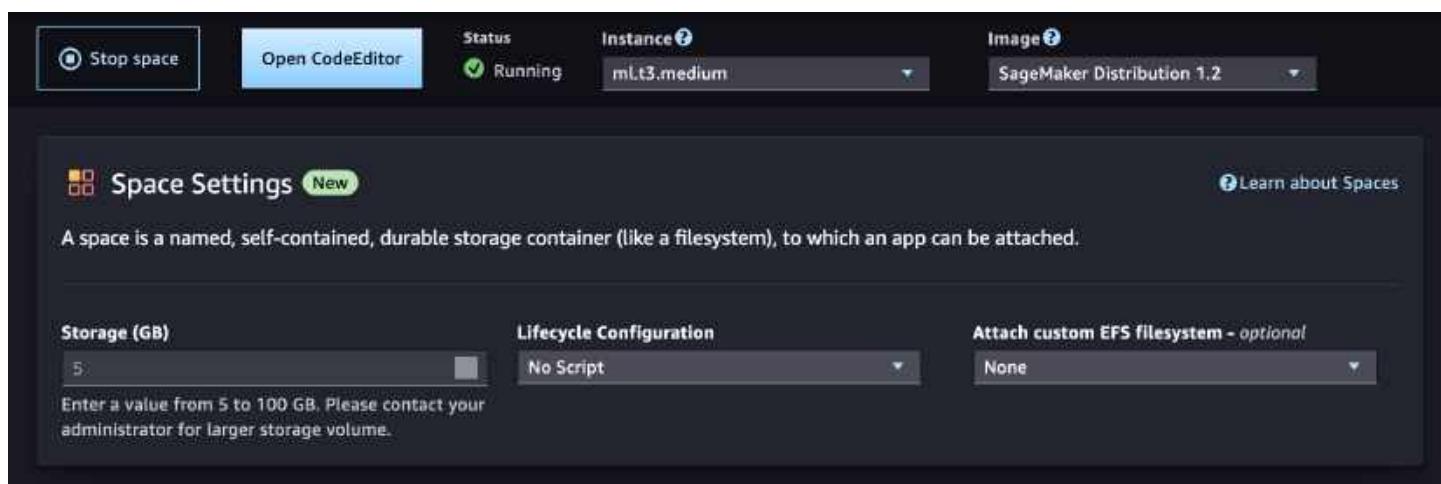
Alternatively, you can delete unused Code Editor resources by using the AWS CLI.

### Stop your Code Editor space using Studio

To stop your Code Editor space in Studio use the following steps:

## To stop your Code Editor space in Studio

1. Return to the Code Editor landing page by doing one of the following:
  - a. In the navigation bar in the upper-left corner, choose **Code Editor**.
  - b. Alternatively, in the left navigation pane, choose **Code Editor** in the **Applications** menu.
2. Find the name of the Code Editor space you created. If the status of your space is **Running**, choose **Stop** in the **Action** column. You can also stop your space directly in the space detail page by choosing **Stop space**. The space may take some time to stop.



Additional resources such as SageMaker AI endpoints, Amazon EMR (Amazon EMR) clusters and Amazon Simple Storage Service (Amazon S3) buckets created from Studio are not automatically deleted when your space instance shuts down. To stop accruing charges from resources, delete any additional resources. For more information, see [Delete unused resources](#).

### Delete Code Editor resources using the AWS CLI

You can delete your Code Editor application and space using the AWS Command Line Interface (AWS CLI).

- [DeleteApp](#)
- [DeleteSpace](#)

# Code Editor administrator guide

You can use Code Editor with an On-Demand Instance for faster start-up time, and configurable storage. You can launch a Code Editor application through Amazon SageMaker Studio or through the AWS CLI. You can also edit Code Editor default settings within the domain console. For more information, see [Edit domain settings](#). The following topics outline how administrators can configure Code Editor, based on Code-OSS, Visual Studio Code - Open Source by changing storage options, customizing environments, and managing user access, as well as giving information about the prerequisites needed to use Code Editor.

## Topics

- [Complete prerequisites](#)
- [Give your users access to private spaces](#)
- [Change the default storage size](#)
- [Code Editor lifecycle configurations](#)
- [Environment customization using custom images](#)

## Complete prerequisites

To use Code Editor, based on Code-OSS, Visual Studio Code - Open Source, you must complete the following prerequisites.

1. You must first onboard to Amazon SageMaker AI domain and create a user profile. For more information, see [Amazon SageMaker AI domain overview](#).
2. If you are interacting with your Code Editor application using the AWS CLI, you must also complete the following prerequisites.
  - a. Update the AWS CLI by following the steps in [Installing the current AWS CLI Version](#).
  - b. From your local machine, run `aws configure` and provide your AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).
3. (Optional) To get more storage and compute for your application, you can request an increase to your AWS quotas. For more information about requesting a quota increase, see [Amazon SageMaker AI endpoints and quotas](#).

# Give your users access to private spaces

## **Important**

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

This section provides a policy that grants user access to private spaces. You can also use the policy to restrict private spaces and applications that are associated with them to the owner associated with the user profile.

You must provide your users with permissions to the following:

- Private spaces
  - The user profile required for access to the private spaces

To provide permissions, attach the following policy to the IAM roles of your users.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "sagemaker>CreateApp",  
        "sagemaker>DeleteApp"  
      ],  
      "Resource": "arn:aws:sagemaker:{}Region:{}AccountId}:app/*",  
      "Condition": {
```

```
    "Null": {
        "sagemaker:OwnerUserProfileArn": "true"
    }
},
{
    "Sid": "SMStudioCreatePresignedDomainUrlForUserProfile",
    "Effect": "Allow",
    "Action": [
        "sagemaker>CreatePresignedDomainUrl"
    ],
    "Resource": "arn:aws:sagemaker:{Region}:{AccountId}:user-profile/${sagemaker:DomainId}/${sagemaker:UserProfileName}"
},
{
    "Sid": "SMStudioAppPermissionsListAndDescribe",
    "Effect": "Allow",
    "Action": [
        "sagemaker>ListApps",
        "sagemaker>ListDomains",
        "sagemaker>ListUserProfiles",
        "sagemaker>ListSpaces",
        "sagemaker>DescribeApp",
        "sagemaker>DescribeDomain",
        "sagemaker>DescribeUserProfile",
        "sagemaker>DescribeSpace"
    ],
    "Resource": "*"
},
{
    "Sid": "SMStudioAppPermissionsTagOnCreate",
    "Effect": "Allow",
    "Action": [
        "sagemaker>AddTags"
    ],
    "Resource": "arn:aws:sagemaker:{Region}:{AccountId}://*",
    "Condition": {
        "Null": {
            "sagemaker:TaggingAction": "false"
        }
    }
},
{
    "Sid": "SMStudioRestrictSharedSpacesWithoutOwners",
```

```
"Effect": "Allow",
"Action": [
    "sagemaker>CreateSpace",
    "sagemaker>UpdateSpace",
    "sagemaker>DeleteSpace"
],
"Resource": "arn:aws:sagemaker:{Region}:{AccountId}:space/
${sagemaker:DomainId}/*",
"Condition": {
    "Null": {
        "sagemaker:OwnerUserProfileArn": "true"
    }
},
{
    "Sid": "SMStudioRestrictSpacesToOwnerUserProfile",
    "Effect": "Allow",
    "Action": [
        "sagemaker>CreateSpace",
        "sagemaker>UpdateSpace",
        "sagemaker>DeleteSpace"
    ],
    "Resource": "arn:aws:sagemaker:{Region}:{AccountId}:space/
${sagemaker:DomainId}/*",
    "Condition": {
        "ArnLike": {
            "sagemaker:OwnerUserProfileArn": "arn:aws:sagemaker:$AWS Region:
$111122223333:user-profile/${sagemaker:DomainId}/${sagemaker:UserProfileName}"
        },
        "StringEquals": {
            "sagemaker:SpaceSharingType": [
                "Private",
                "Shared"
            ]
        }
    }
},
{
    "Sid": "SMStudioRestrictCreatePrivateSpaceAppsToOwnerUserProfile",
    "Effect": "Allow",
    "Action": [
        "sagemaker>CreateApp",
        "sagemaker>DeleteApp"
    ],
}
```

```
"Resource": "arn:aws:sagemaker:{{Region}}:{{AccountId}}:app/${{sagemaker:DomainId}}/*",
  "Condition": {
    "ArnLike": {
      "sagemaker:OwnerUserProfileArn": "arn:aws:sagemaker:${{aws:Region}}:${{aws:PrincipalAccount}}:user-profile/${{sagemaker:DomainId}}/${{sagemaker:UserProfileName}}"
    },
    "StringEquals": {
      "sagemaker:SpaceSharingType": [
        "Private"
      ]
    }
  },
},
]
```

## Change the default storage size

You can change the default storage settings of your users. You can also change the default storage settings based on your organizational requirements and the needs of your users.

To change the storage size of your users, do the following:

1. Update the Amazon EBS storage settings in the domain.
2. Create a user profile and specify the storage settings within it.

Use the following AWS Command Line Interface (AWS CLI) command to update the domain.

```
aws --region $REGION sagemaker update-domain \
--domain-id $DOMAIN_ID \
--default-user-settings '{
  "SpaceStorageSettings": {
    "DefaultEbsStorageSettings": {
      "DefaultEbsVolumeSizeInGb": 5,
      "MaximumEbsVolumeSizeInGb": 100
    }
  }
}'
```

Use the following AWS CLI command to create the user profile and specify the default storage settings.

```
aws --region $REGION sagemaker create-user-profile \
--domain-id $DOMAIN_ID \
--user-profile-name $USER_PROFILE_NAME \
--user-settings '{
    "SpaceStorageSettings": {
        "DefaultEbsStorageSettings": {
            "DefaultEbsVolumeSizeInGb": 5,
            "MaximumEbsVolumeSizeInGb": 100
        }
    }
}'
```

Use the following AWS CLI commands to update the default storage settings in the user profile.

```
aws --region $REGION sagemaker update-user-profile \
--domain-id $DOMAIN_ID \
--user-profile-name $USER_PROFILE_NAME \
--user-settings '{
    "SpaceStorageSettings": {
        "DefaultEbsStorageSettings": {
            "DefaultEbsVolumeSizeInGb": 25,
            "MaximumEbsVolumeSizeInGb": 200
        }
    }
}'
```

## Code Editor lifecycle configurations

You can use Code Editor lifecycle configurations to automate customization for your Studio environment. This customization includes installing custom packages, configuring extensions, preloading datasets, and setting up source code repositories.

The following instructions use the AWS Command Line Interface (AWS CLI) to create, attach, debug, and detach lifecycle configurations for the CodeEditor application type:

- [Create and attach lifecycle configurations in Studio](#)
- [Debug lifecycle configurations in Studio](#)
- [Detach lifecycle configurations in Studio](#)

## Create and attach lifecycle configurations in Studio

The following section provides AWS CLI commands to create a lifecycle configuration, attach a lifecycle configuration when creating a new user profile, and attach a lifecycle configuration when updating a user profile. For prerequisites and general steps on creating and attaching lifecycle configurations in Studio, see [Lifecycle configuration creation](#).

When creating your Studio lifecycle configuration with the `create-studio-lifecycle-config` command, be sure to specify that the `studio-lifecycle-config-app-type` is `CodeEditor`. The following example shows how to create a new Studio lifecycle configuration for your Code Editor application.

```
aws sagemaker create-studio-lifecycle-config \
--studio-lifecycle-config-name my-code-editor-lcc \
--studio-lifecycle-config-content $LCC_CONTENT \
--studio-lifecycle-config-app-type CodeEditor
```

Note the ARN of the newly created lifecycle configuration that is returned. When attaching a lifecycle configuration, provide this ARN within the `LifecycleConfigArns` list of `CodeEditorAppSettings`.

You can attach a lifecycle configuration when creating a user profile or domain. The following example shows how to create a new user profile with the lifecycle configuration attached. You can also create a new domain with a lifecycle configuration attached by using the [create-domain](#) command.

```
# Create a new UserProfile
aws sagemaker create-user-profile \
--domain-id domain-id \
--user-profile-name user-profile-name \
--user-settings '{
  "CodeEditorAppSettings": {
    "LifecycleConfigArns": [
      lifecycle-configuration-arn-list
    ]
  }
}'
```

You can alternatively attach a lifecycle configuration when updating a user profile or domain. The following example shows how to update a user profile with the lifecycle configuration attached.

You can also update a new domain with a lifecycle configuration attached by using the [update-domain](#) command.

```
# Update a UserProfile
aws sagemaker update-user-profile \
--domain-id domain-id \
--user-profile-name user-profile-name \
--user-settings '{
  "CodeEditorAppSettings": {
    "LifecycleConfigArns": [
      lifecycle-configuration-arn-list
    ]
  }
}'
```

## Debug lifecycle configurations in Studio

To debug lifecycle configuration scripts for Code Editor, you must use Studio. For instructions about debugging lifecycle configurations in Studio, see [Debug lifecycle configurations](#). To find the logs for a specific application, search the log streams using the following format:

```
domain-id/space-name/CodeEditor/default/LifecycleConfigOnStart
```

## Detach lifecycle configurations in Studio

To detach lifecycle configurations for Code Editor, you can use either the console or the AWS CLI. For steps on detaching lifecycle configurations from the Studio console, see [Detach lifecycle configurations](#).

To detach a lifecycle configuration using the AWS CLI, remove the desired lifecycle configuration from the list of lifecycle configurations attached to the resource. Then pass the list as part of the respective command:

- [update-user-profile](#)
- [update-domain](#)

For example, the following command removes all lifecycle configurations for the Code Editor application attached to the domain.

```
aws sagemaker update-domain --domain-id domain-id \
```

```
--default-user-settings '{  
"CodeEditorAppSettings": {  
 "LifecycleConfigArns":  
 []  
 }  
}'
```

## Create a lifecycle configuration to clone repositories into a Code Editor application

This section shows how to clone a repository and create a Code Editor application with the lifecycle configuration attached.

1. From your local machine, create a file named `my-script.sh` with the following content:

```
#!/bin/bash  
set -eux
```

2. Clone the repository of your choice in your lifecycle configuration script.

```
export REPOSITORY_URL="https://github.com/aws-samples/sagemaker-studio-lifecycle-config-examples.git"  
git -C /home/sagemaker-user clone $REPOSITORY_URL
```

3. After finalizing your script, create and attach your lifecycle configuration. For more information, see [Create and attach lifecycle configurations in Studio](#).
4. Create your Code Editor application with the lifecycle configuration attached.

```
aws sagemaker create-app \  
--domain-id domain-id \  
--space-name space-name \  
--app-type CodeEditor \  
--app-name default \  
--resource-spec "SageMakerImageArn=arn:aws:sagemaker:region:image-account-id:image/sagemaker-distribution-cpu,LifecycleConfigArn=arn:aws:sagemaker:region:user-account-id:studio-lifecycle-config/my-code-editor-lcc,InstanceType=ml.t3.large"
```

For more information about available Code Editor image ARNs, see [Code Editor application instances and images](#).

## Create a lifecycle configuration to install Code Editor extensions

This section shows how to create a lifecycle configuration to install extensions from the [Open VSX Registry](#) in your Code Editor environment.

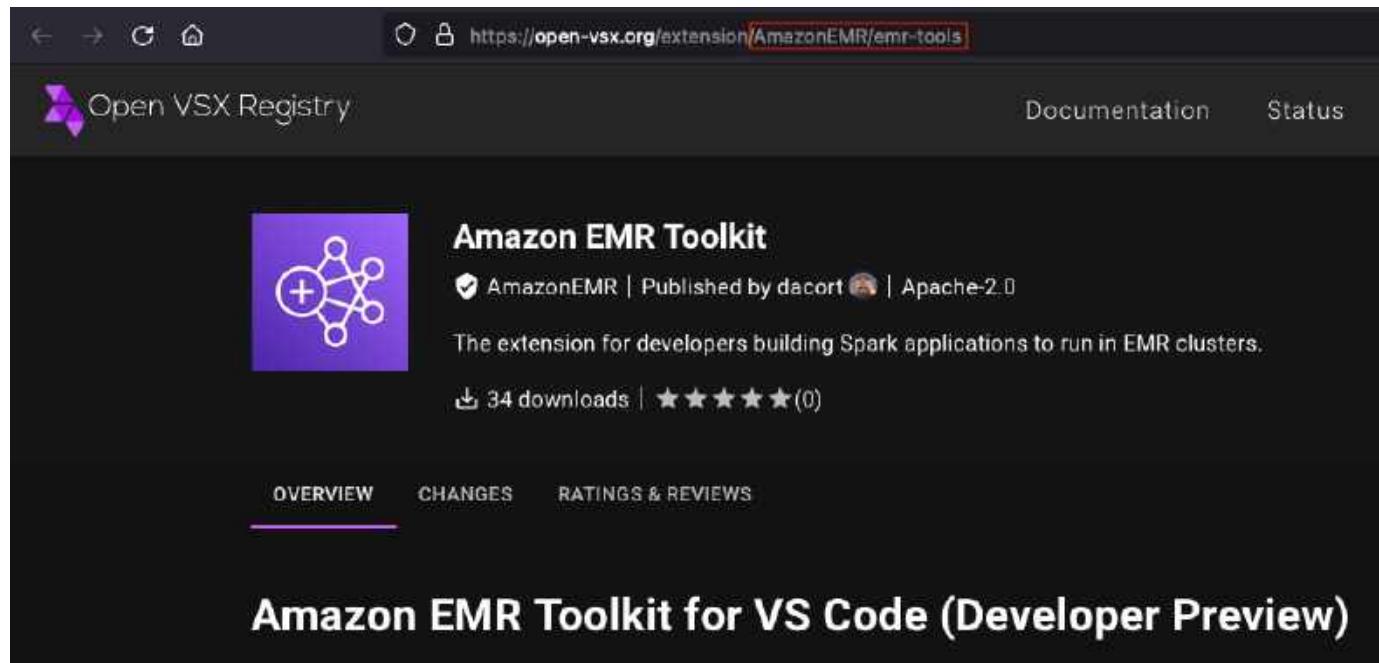
1. From your local machine, create a file named my-script.sh with the following content:

```
#!/bin/bash  
set -eux
```

2. Within the script, install the [Open VSX Registry](#) extension of your choice:

```
sagemaker-code-editor --install-extension AmazonEMR.emr-tools --extensions-dir /  
opt/amazon/sagemaker/sagemaker-code-editor-server-data/extensions
```

You can retrieve the extension name from the URL of the extension in the [Open VSX Registry](#). The extension name to use in the sagemaker-code-editor command should contain all text that follows <https://open-vsx.org/extension/> in the URL. Replace all instances of a slash (/) with a period (.). For example, AmazonEMR/emr-tools should be AmazonEMR.emr-tools.



3. After finalizing your script, create and attach your lifecycle configuration. For more information, see [Create and attach lifecycle configurations in Studio](#).
4. Create your Code Editor application with the lifecycle configuration attached:

```
aws sagemaker create-app \
--domain-id domain-id \
--space-name space-name \
--app-type CodeEditor \
--app-name default \
--resource-spec "SageMakerImageArn=arn:aws:sagemaker:region:image-account-id:image/sagemaker-distribution-cpu,LifecycleConfigArn=arn:aws:sagemaker:region:user-account-id:studio-lifecycle-config/my-code-editor-lcc,InstanceType=ml.t3.large"
```

For more information about available Code Editor image ARNs, see [Code Editor application instances and images](#). For more information about connections and extensions, see [Code Editor Connections and Extensions](#).

## Environment customization using custom images

If you need functionality that is different than what's provided by SageMaker distribution, you can bring your own image with your custom extensions and packages. You can also use it to personalize the Code Editor UI for your own branding or compliance needs.

For requirements for your image, see [Dockerfile specifications](#).

For a tutorial that helps you create an image that your users can access to run their Code Editor environment, see [Provide users with access to custom images](#).

### Topics

- [Dockerfile specifications](#)
- [Provide users with access to custom images](#)

### Dockerfile specifications

The image that you specify in your Dockerfile must match the specifications in the following sections to create the image successfully.

### Running the image

- **Entrypoint** – We recommend embedding the entry point into the image using the Docker CMD or Entrypoint instructions. You can also configure ContainerEntrypoint and

ContainerArguments that are passed to the container at runtime. For more information, see [CodeEditorAppImageConfig](#).

- EnvVariables – With Studio, you can configure ContainerEnvironment variables that are made available to a container. The environment variable is overwritten with the environment variables from SageMaker AI. To provide you with a better experience, the environment variables are usually AWS\_ and SageMaker AI\_namespaced to give priority to platform environments.

The following are the environment variables:

- AWS\_REGION
- AWS\_DEFAULT\_REGION
- AWS\_CONTAINER\_CREDENTIALS\_RELATIVE\_URI
- SAGEMAKER\_SPACE\_NAME

## Specifications for the user and file system

- WorkingDirectory – The Amazon EBS volume for your space is mounted on the path /home/sagemaker-user. You can't change the mount path. Use the WORKDIR instruction to set the working directory of your image to a folder within /home/sagemaker-user.
- UID – The user ID of the Docker container. UID=1000 is a supported value. You can add sudo access to your users. The IDs are remapped to prevent a process running in the container from having more privileges than necessary.
- GID – The group ID of the Docker container. GID=100 is a supported value. You can add sudo access to your users. The IDs are remapped to prevent a process running in the container from having more privileges than necessary.
- Meta data directories – The /opt/.sagemakerinternal and /opt/ml directories that are used by AWS. The meta data file in /opt/ml contains meta data about resources such as DomainId.

Use the following command to show the file system contents:

```
cat /opt/ml/metadata/resource-metadata.json
>{"AppType":"CodeEditor","DomainId":"example-domain-id","UserProfileName":"example-user-profile-name","ResourceArn":"arn:aws:sagemaker:AWS
Region:111122223333;app/domain-ID/user-ID/CodeEditor/
default","ResourceName":"default","AppImageVersion":"current"}
```

- Logging directories – `/var/log/studio` are reserved for the logging directories of Code Editor and the extensions associated with it. We recommend that you don't use the folders in creating your image.

## Health check and URL for applications

- Base URL – The base URL for the BYOI application must be `codeeditor/default`. You can only have one application and it must always be named default.
- Health check endpoint – You must host your Code Editor server at `0.0.0.0 port 8888` for SageMaker AI to detect it.
- Authentication – You must pass `--without-connection-token` when opening `sagemaker-code-editor` to allow SageMaker AI to authenticate your users.

### Note

If you are using Amazon SageMaker Distribution as the base image, these requirements are already taken care of as part of the included `entrypoint-code-editor` script.

## Dockerfile samples

The following is a sample Dockerfile that meets the specifications listed in the previous sections to create an image from scratch using a [micromamba](#) base environment:

```
FROM mambaorg/micromamba:latest
ARG NB_USER="sagemaker-user"
ARG NB_UID=1000
ARG NB_GID=100

USER root

RUN micromamba install -y --name base -c conda-forge sagemaker-code-editor

USER $NB_UID

CMD eval "$(micromamba shell hook --shell=bash)"; \
    micromamba activate base; \
    sagemaker-code-editor --host 0.0.0.0 --port 8888 \
    --without-connection-token \
```

```
--base-path "/CodeEditor/default"
```

The following is a sample Dockerfile that meets the specifications listed in the previous sections to create an image based on [Amazon SageMaker AI Distribution](#):

```
FROM public.ecr.aws/sagemaker/sagemaker-distribution:latest-cpu
ARG NB_USER="sagemaker-user"
ARG NB_UID=1000
ARG NB_GID=100
ENV MAMBA_USER=$NB_USER

USER root

# install scrapy in the base environment
RUN micromamba install -y --name base -c conda-forge scrapy

# download VSCodeVim
RUN \
  wget https://github.com/VSCodeVim/Vim/releases/download/v1.27.2/vim-1.27.2.vsix \
  -P /tmp/exts/ --no-check-certificate

# Install the extension
RUN \
  extensionloc=/opt/amazon/sagemaker/sagemaker-code-editor-server-data/extensions \
  && sagemaker-code-editor \
  --install-extension "/tmp/exts/vim-1.27.2.vsix" \
  --extensions-dir "${extensionloc}"

USER $MAMBA_USER
ENTRYPOINT ["entrypoint-code-editor"]
```

## Provide users with access to custom images

This documentation provides step-by-step instructions to provide your users with access to custom images for their Code Editor environments. You can use the information on this page to create custom environments for your user's workflows. The process involves utilizing:

- Docker
- AWS Command Line Interface
- Amazon Elastic Container Registry
- Amazon SageMaker AI AWS Management Console

After following the guidance on this page, Code Editor users on the Amazon SageMaker AI domain will have access to the custom image and environment from their Code Editor spaces to empower their machine learning workflows.

**⚠️ Important**

This page assumes that you have the AWS Command Line Interface and Docker installed on your local machine.

To have your users successfully run their image within Code Editor, you must do the following:

**To have your users successfully run the image**

1. Create the Dockerfile
2. Build the image from the Dockerfile
3. Upload the image to Amazon Elastic Container Registry
4. Attach the image to your Amazon SageMaker AI domain
5. Have your users access the image from their Code Editor space

**Step 1: Create the Dockerfile**

Create a Dockerfile to define the steps needed to create the environment needed to run the application in your user's container.

**⚠️ Important**

Your Dockerfile must meet the specifications provided in [Dockerfile specifications](#).

For sample Dockerfiles in the correct format, see [Dockerfile samples](#).

**Step 2: Build the image**

In the same directory as your Dockerfile, build your image using the following command:

```
docker build -t username/imagename:tag your-account-id.dkr.ecr.AWS  
Region.amazonaws.com/your-repository-name:tag
```

### **⚠️ Important**

Your image must be tagged in the following format: `123456789012.dkr.ecr.your-region.amazonaws.com/your-repository-name:tag`

You won't be able to push it to an Amazon Elastic Container Registry repository otherwise.

## **Step 3: Push the image to the Amazon Elastic Container Registry repository**

After you've built your image, log in to your Amazon ECR repository using the following command:

```
aws ecr get-login-password --region AWS Region | docker login --username AWS --password-stdin 123456789012.dkr.ecr.AWS Region.amazonaws.com
```

After you've logged in, push your Dockerfile using the following command:

```
docker push 123456789012.dkr.ecr.AWS Region.amazonaws.com/your-repository-name:tag
```

## **Step 4: Attach image to the Amazon SageMaker AI domain of your users**

After you've pushed the image, you must access it from your Amazon SageMaker AI domain using either the SageMaker AI console or the AWS CLI.

### **Attach the image using the SageMaker AI console**

Use the following procedure to attach the image to a SageMaker domain through the SageMaker AI console :

1. Open the [SageMaker AI console](#).
2. Under **Admin configurations**, choose **Domains**.
3. From the list of **domains**, select a domain.
4. Open the **Environment** tab.
5. For **Custom images for personal Studio apps**, choose **Attach image**.
6. Specify the image source. You can create a new image or choose an existing image.
7. Choose **Next**.
8. Choose **Code Editor** as the application type.

## 9. Choose **Submit**.

### Attach the image using the AWS CLI

Use the following procedure to attach the image to a SageMaker domain through the AWS CLI :

1. Create a SageMaker image. The AmazonSageMakerFullAccess policy must be attached to your role as you use the following AWS CLI commands.

```
aws sagemaker create-image \
--image-name code-editor-custom-image \
--role-arn arn:aws:iam::account-id:role/service-role/execution-role
```

2. Create a SageMaker image version from the image. Pass the unique tag value that you chose when you pushed the image to Amazon ECR.

```
aws sagemaker create-image-version \
--image-name code-editor-custom-image \
--base-image repository-uri:tag
```

3. Create a configuration file called `app-image-config-input.json`. The application image configuration is used as configuration for running a SageMaker image as a Code Editor application. You may also specify your [ContainerConfig](#) arguments here.

```
{  
    "AppImageConfigName": "code-editor-app-image-config",  
    "CodeEditorAppImageConfig":  
    {  
        "ContainerConfig":  
        {}  
    }  
}
```

4. Create the AppImageConfig using the application image configuration file that you created.

```
aws sagemaker create-app-image-config \
--cli-input-json file://app-image-config-input.json
```

5. Create a configuration file, named `updateDomain.json`. Be sure to specify your domain ID.

```
{
```

```
"DomainId": "domain-id",
"DefaultUserSettings": {
    "CodeEditorAppSettings": {
        "CustomImages": [
            {
                "ImageName": "code-editor-custom-image",
                "AppImageConfigName": "code-editor-app-image-config"
            }
        ]
    }
}
```

6. Call the `UpdateDomain` command with the configuration file as input.

 **Note**

You must delete all of the applications in your domain before updating the domain with the new image. Note that you only need to delete applications; you **do not** need to delete user profiles or shared spaces. For instructions on deleting applications, choose one of the following options.

- If you use the SageMaker AI console, run through Step 1 to 5d and Step 6 to 7d of the [Delete a domain \(Console\)](#) section.
- If you use the AWS CLI, run through Step 1 to 3 of the [Delete a domain \(AWS CLI\)](#) section.

```
aws sagemaker update-domain --cli-input-json file://updateDomain.json
```

## Step 5: Have your users access the image from their Code Editor space

Your users can now select the image that you've attached to their domain from their Code Editor space.

For more information on selecting a custom image, see [Launch a Code Editor application in Studio](#).

# Amazon SageMaker HyperPod

SageMaker HyperPod helps you provision resilient clusters for running machine learning (ML) workloads and developing state-of-the-art models such as large language models (LLMs), diffusion models, and foundation models (FMs). It accelerates development of FMs by removing undifferentiated heavy-lifting involved in building and maintaining large-scale compute clusters powered by thousands of accelerators such as AWS Trainium and NVIDIA A100 and H100 Graphical Processing Units (GPUs). When accelerators fail, the resiliency features of SageMaker HyperPod monitor the cluster instances automatically detect and replace the faulty hardware on the fly so that you can focus on running ML workloads.

To get started, check [the section called “Prerequisites”](#), set up [the section called “IAM for HyperPod”](#), and choose one of the following orchestrator options supported by SageMaker HyperPod.

## Slurm support in SageMaker HyperPod

SageMaker HyperPod provides support for running machine learning workloads on resilient clusters by integrating with Slurm, an open-source workload manager. Slurm support in SageMaker HyperPod enables seamless cluster orchestration through Slurm cluster configuration, allowing you to set up head, login, and worker nodes on the SageMaker HyperPod clusters. This integration also facilitates Slurm-based job scheduling for running ML workloads on the cluster, as well as direct access to cluster nodes for job scheduling. With HyperPod's lifecycle configuration support, you can customize the computing environment of the clusters to meet your specific requirements. Additionally, by leveraging the Amazon SageMaker AI distributed training libraries, you can optimize the clusters' performance on AWS computing and network resources. To learn more, see [the section called “Orchestrating HyperPod clusters with Slurm”](#).

## Amazon EKS support in SageMaker HyperPod

SageMaker HyperPod also integrates with Amazon EKS to enable large-scale training of foundation models on long-running and resilient compute clusters. This allows cluster admin users to provision HyperPod clusters and attach them to an EKS control plane, enabling dynamic capacity management, direct access to cluster instances, and resiliency capabilities. For data scientists, Amazon EKS support in HyperPod allows running containerized workloads for training foundation models, inference on the EKS cluster, and leveraging the job auto-resume capability for Kubeflow PyTorch training. The architecture involves a 1-to-1 mapping between an EKS cluster (control plane) and a HyperPod cluster (worker nodes) within a VPC, providing a tightly integrated solution

for running large-scale ML workloads. To learn more, see [the section called “Orchestrating HyperPod clusters with Amazon EKS”](#).

## AWS Regions supported by SageMaker HyperPod

SageMaker HyperPod is available in the following AWS Regions.

- us-east-1
- us-east-2
- us-west-1
- us-west-2
- eu-central-1
- eu-north-1
- eu-west-1
- eu-west-2
- ap-south-1
- ap-southeast-1
- ap-southeast-2
- ap-southeast-4
- ap-northeast-1
- sa-east-1

### Topics

- [Prerequisites for using SageMaker HyperPod](#)
- [AWS Identity and Access Management for SageMaker HyperPod](#)
- [SageMaker HyperPod recipes](#)
- [Orchestrating SageMaker HyperPod clusters with Slurm](#)
- [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#)
- [HyperPod in Studio](#)
- [SageMaker HyperPod references](#)
- [Amazon SageMaker HyperPod release notes](#)
- [Amazon SageMaker HyperPod AMI releases](#)

# Prerequisites for using SageMaker HyperPod

The following sections walk you through prerequisites before getting started with SageMaker HyperPod.

## Topics

- [SageMaker HyperPod quotas](#)
- [Setting up SageMaker HyperPod with a custom Amazon VPC](#)
- [Setting up SageMaker HyperPod clusters across multiple AZs](#)
- [Setting up AWS Systems Manager and Run As for cluster user access control](#)
- [\(Optional\) Setting up SageMaker HyperPod with Amazon FSx for Lustre](#)

## SageMaker HyperPod quotas

You can create SageMaker HyperPod clusters given the quotas for *cluster usage* in your AWS account.

### Important

To learn more about SageMaker HyperPod pricing, see [the section called “SageMaker HyperPod pricing”](#) and [Amazon SageMaker Pricing](#).

## View Amazon SageMaker HyperPod quotas using the AWS Management Console

Look up the default and applied values of a *quota*, also referred to as a *limit*, for *cluster usage*, which is used for SageMaker HyperPod.

1. Open the [Service Quotas console](#).
2. In the left navigation pane, choose **AWS services**.
3. From the **AWS services** list, search for and select **Amazon SageMaker AI**.
4. In the **Service quotas** list, you can see the service quota name, applied value (if it's available), AWS default quota, and whether the quota value is adjustable.
5. In the search bar, type **cluster usage**. This shows quotas for cluster usage, applied quotas, and the default quotas.

## Request a Amazon SageMaker HyperPod quota increase using the AWS Management Console

Increase your quotas at the account or resource level.

1. To increase the quota of instances for *cluster usage*, select the quota that you want to increase.
2. If the quota is adjustable, you can request a quota increase at either the account level or resource level based on the value listed in the **Adjustability** column.
3. For **Increase quota value**, enter the new value. The new value must be greater than the current value.
4. Choose **Request**.
5. To view any pending or recently resolved requests in the console, navigate to the **Request history** tab from the service's details page, or choose **Dashboard** from the navigation pane. For pending requests, choose the status of the request to open the request receipt. The initial status of a request is **Pending**. After the status changes to **Quota requested**, you see the case number with AWS Support. Choose the case number to open the ticket for your request.

To learn more about requesting a quota increase in general, see [Requesting a Quota Increase](#) in the *AWS Service Quotas User Guide*.

## Setting up SageMaker HyperPod with a custom Amazon VPC

To set up a SageMaker HyperPod cluster with a custom Amazon VPC, review the following prerequisites.

### Note

VPC configuration is mandatory for Amazon EKS orchestration. For Slurm orchestration, VPC setup is optional.

- Validate [Elastic Network Interface](#) (ENI) capacity in your AWS account before creating a SageMaker HyperPod cluster with a custom VPC. The ENI limit is controlled by Amazon EC2 and varies by AWS Region. SageMaker HyperPod cannot automatically request quota increases.

### To verify your current ENI quota:

1. Open the [Service Quotas console](#).
2. In the **Manage quotas** section, use the **AWS Services** drop-down list to search for **VPC**.

3. Choose to view the quotas of **Amazon Virtual Private Cloud (Amazon VPC)**.
4. Look for the service quota **Network interfaces per Region** or the **Quota code L-DF5E4CA3**.

If your current ENI limit is insufficient for your SageMaker HyperPod cluster needs, request a quota increase. Ensuring adequate ENI capacity beforehand helps prevent cluster deployment failures.

- When using a custom VPC to connect a SageMaker HyperPod cluster with AWS resources, provide the VPC name, ID, AWS Region, subnet IDs, and security group IDs during cluster creation.

 **Note**

When your Amazon VPC and subnets support IPv6 in the [VPCCConfig](#) of the cluster or at the Instance group level using the `OverrideVPCCConfig` attribute of [ClusterInstanceGroupSpecification](#), network communications differ based on the cluster orchestration platform:

- Slurm-orchestrated clusters automatically configure nodes with dual IPv6 and IPv4 addresses, allowing immediate IPv6 network communications. No additional configuration is required beyond the VPCCConfig IPv6 settings.
- In EKS-orchestrated clusters, nodes receive dual-stack addressing, but pods can only use IPv6 when the Amazon EKS cluster is explicitly IPv6-enabled. You must create a new IPv6 Amazon EKS cluster - existing IPv4 Amazon EKS clusters cannot be converted to IPv6. For information about deploying an IPv6 Amazon EKS cluster, see [Amazon EKS IPv6 Cluster Deployment](#).

Additional resources for IPv6 configuration:

- For information about adding IPv6 support to your VPC, see to [IPv6 Support for VPC](#).
- For information about creating a new IPv6-compatible VPC, see [Amazon VPC Creation Guide](#).
- To configure SageMaker HyperPod with a custom Amazon VPC, see [Custom Amazon VPC setup for SageMaker HyperPod](#).

- Make sure that all resources are deployed in the same AWS Region as the SageMaker HyperPod cluster. Configure security group rules to allow inter-resource communication within the VPC. For example, when creating a VPC in us-west-2, provision subnets across one or more Availability Zones (such as us-west-2a or us-west-2b), and create a security group allowing intra-group traffic.

**Note**

SageMaker HyperPod supports multi-Availability Zone deployment. For more information, see [the section called “Setting up SageMaker HyperPod clusters across multiple AZs”](#).

- Establish Amazon Simple Storage Service (Amazon S3) connectivity for VPC-deployed SageMaker HyperPod instance groups by creating a VPC endpoint. Without internet access, instance groups cannot store or retrieve lifecycle scripts, training data, or model artifacts. We recommend that you create a custom IAM policy restricting Amazon S3 bucket access to the private VPC. For more information, see [Endpoints for Amazon S3](#) in the *AWS PrivateLink Guide*.
- For HyperPod clusters using Elastic Fabric Adapter (EFA)-enabled instances, configure the security group to allow all inbound and outbound traffic to and from the security group itself. Specifically, avoid using `0.0.0.0/0` for outbound rules, as this may cause EFA health check failures. For more information about EFA security group preparation guidelines, see [Step 1: Prepare an EFA-enabled security group](#) in the *Amazon EC2 User Guide*.

## Setting up SageMaker HyperPod clusters across multiple AZs

You can configure your SageMaker HyperPod clusters across multiple Availability Zones (AZs) to improve reliability and availability.

**Note**

Elastic Fabric Adapter (EFA) traffic cannot cross AZs or VPCs. This does not apply to normal IP traffic from the ENA device of an EFA interface. For more information, see [EFA limitations](#).

- **Default behavior**

HyperPod deploys all cluster instances in a single Availability Zone. The VPC configuration determines the deployment AZ:

- For Slurm-orchestrated clusters, VPC configuration is optional. When no VPC configuration is provided, HyperPod defaults to one subnet from the platform VPC.
- For EKS-orchestrated clusters, VPC configuration is required.

- For both Slurm and EKS orchestrators, when [VpcConfig](#) is provided, HyperPod selects a subnet from the provided VpcConfig's subnet list. All instance groups inherit the subnet's AZ.

 **Note**

Once you create a cluster, you cannot modify its VpcConfig settings.

To learn more about configuring VPCs for HyperPod clusters, see the preceding section, [Setting up SageMaker HyperPod with a custom Amazon VPC](#).

- **Multi-AZ configuration**

You can set up your HyperPod cluster across multiple AZs when creating a cluster or when adding a new instance group to an existing cluster. To configure multi-AZ deployments, you can override the default VPC settings of the cluster by specifying different subnets and security groups, potentially across different Availability Zones, for individual instance groups within your cluster.

SageMaker HyperPod API users can use the `OverrideVpcConfig` property within the [ClusterInstanceGroupSpecification](#) when working with the [CreateCluster](#) or [UpdateCluster](#) APIs.

The `OverrideVpcConfig` field:

- Cannot be modified after the instance group is created.
- Is optional. If not specified, the cluster level [VpcConfig](#) is used as default.
- For Slurm-orchestrated clusters, can only be specified when cluster level `VpcConfig` is provided. If no `VpcConfig` is specified at cluster level, `OverrideVpcConfig` cannot be used for any instance group.
- Contains two required fields:
  - Subnets - accepts between 1 and 16 subnet IDs
  - SecurityGroupIds - accepts between 1 and 5 security group IDs

For more information about creating or updating a SageMaker HyperPod cluster using the SageMaker HyperPod console UI or the AWS CLI:

- Slurm orchestration: See [Operating Slurm-orchestrated HyperPod clusters](#).
- EKS orchestration. See [Operating EKS-orchestrated HyperPod clusters](#).

**Note**

When running workloads across multiple AZs, be aware that network communication between AZs introduces additional latency. Consider this impact when designing latency-sensitive applications.

## Setting up AWS Systems Manager and Run As for cluster user access control

the section called "SageMaker HyperPod DLAMI" comes with [AWS Systems Manager](#) (SSM) out of the box to help you manage access to your SageMaker HyperPod cluster instance groups. This section describes how to create operating system (OS) users in your SageMaker HyperPod clusters and associate them with IAM users and roles. This is useful to authenticate SSM sessions using the credentials of the OS user account.

**Note**

Granting users access to HyperPod cluster nodes allows them to install and operate user-managed software on the nodes. Ensure that you maintain the principle of least-privilege permissions for users.

### Enabling Run As in your AWS account

As an AWS account admin or a cloud administrator, you can manage access to SageMaker HyperPod clusters at an IAM role or user level by using the [Run As feature in SSM](#). With this feature, you can start each SSM session using the OS user associated to the IAM role or user.

To enable Run As in your AWS account, follow the steps in [Turn on Run As support for Linux and macOS managed nodes](#). If you already created OS users in your cluster, make sure that you associate them with IAM roles or users by tagging them as guided in **Option 2** of step 5 under [To turn on Run As support for Linux and macOS managed nodes](#).

### (Optional) Setting up SageMaker HyperPod with Amazon FSx for Lustre

To start using SageMaker HyperPod and mapping data paths between the cluster and your FSx for Lustre file system, select one of the AWS Regions supported by SageMaker HyperPod. After choosing the AWS Region you prefer, you also should determine which Availability Zone (AZ) to use.

If you use SageMaker HyperPod compute nodes in AZs different from the AZs where your FSx for Lustre file system is set up within the same AWS Region, there might be communication and network overhead. We recommend that you use the same physical AZ as the one for the SageMaker HyperPod service account to avoid any cross-AZ traffic between SageMaker HyperPod clusters and your FSx for Lustre file system. Also, make sure that you have configured it with your VPC. If you want to use Amazon FSx as the main file system for storage, you must configure SageMaker HyperPod clusters with your VPC.

## AWS Identity and Access Management for SageMaker HyperPod

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use Amazon EKS resources. IAM is an AWS service that you can use with no additional charge.

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

Let's assume that there are two main layers of SageMaker HyperPod users: *cluster admin users* and *data scientist users*.

- **Cluster admin users** – Are responsible for creating and managing SageMaker HyperPod clusters. This includes configuring the HyperPod clusters and managing user access to them.
  - Create and configure SageMaker HyperPod clusters with Slurm or Amazon EKS.
  - Create and configure IAM roles for data scientist users and HyperPod cluster resources.

- For SageMaker HyperPod orchestration with Amazon EKS, create and configure [EKS access entries](#), [role-based access control \(RBAC\)](#), and Pod Identity to fulfill data science use cases.
- **Data scientist users** – Focus on ML model training. They use the open-source orchestrator or the SageMaker HyperPod CLI to submit and manage training jobs.
  - Assume and use the IAM Role provided by cluster admin users.
  - Interact with the open-source orchestrator CLIs supported by SageMaker HyperPod (Slurm or Kubernetes) or the SageMaker HyperPod CLI to check clusters capacity, connect to cluster, and submit workloads.

Set up IAM roles for cluster admins by attaching the right permissions or policies to operate SageMaker HyperPod clusters. Cluster admins also should create IAM roles to provide to SageMaker HyperPod resources to assume to run and communicate with necessary AWS resources, such as Amazon S3, Amazon CloudWatch, and AWS Systems Manager (SSM). Finally, the AWS account admin or the cluster admins should grant scientists permissions to access the SageMaker HyperPod clusters and run ML workloads.

Depending on which orchestrator you choose, permissions needed for the cluster admin and scientists may vary. You can also control the scope of permissions for various actions in the roles using the condition keys per service. Use the following Service Authorization References for adding detailed scope for the services related to SageMaker HyperPod.

- [Amazon Elastic Compute Cloud](#)
- [Amazon Elastic Container Registry](#) (for SageMaker HyperPod cluster orchestration with Amazon EKS)
- [Amazon Elastic Kubernetes Service](#) (for SageMaker HyperPod cluster orchestration with Amazon EKS)
- [Amazon FSx](#)
- [AWS IAM Identity Center](#) (successor to [AWS Single Sign-On](#))
- [AWS Identity and Access Management \(IAM\)](#)
- [Amazon Simple Storage Service](#)
- [Amazon SageMaker AI](#)
- [AWS Systems Manager](#)

## Topics

- [IAM users for cluster admin](#)
- [IAM users for scientists](#)
- [IAM role for SageMaker HyperPod](#)

## IAM users for cluster admin

Cluster administrators (admins) operate and configure SageMaker HyperPod clusters, performing the tasks in [the section called “SageMaker HyperPod operation”](#). The following policy example includes the minimum set of permissions for cluster administrators to run the SageMaker HyperPod core APIs and manage SageMaker HyperPod clusters within your AWS account.

### Slurm

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker:CreateCluster",  
                "sagemaker>ListClusters"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker>DeleteCluster",  
                "sagemaker:DescribeCluster",  
                "sagemaker:DescribeClusterNode",  
                "sagemaker>ListClusterNodes",  
                "sagemaker:UpdateCluster",  
                "sagemaker:UpdateClusterSoftware",  
                "sagemaker:BatchDeleteClusterNodes"  
            ],  
            "Resource": "arn:aws:sagemaker:region:account-id:cluster/*"  
        }  
    ]  
}
```

## Amazon EKS

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam:PassRole",  
            "Resource": <execution-role-arn>  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker>CreateCluster",  
                "sagemaker>DeleteCluster",  
                "sagemaker>DescribeCluster",  
                "sagemaker>DescribeCluterNode",  
                "sagemaker>ListClusterNodes",  
                "sagemaker>ListClusters",  
                "sagemaker>UpdateCluster",  
                "sagemaker>UpdateClusterSoftware",  
                "sagemaker>BatchDeleteClusterNodes",  
                "eks>DescribeCluster",  
                "eks>CreateAccessEntry",  
                "eks>DescribeAccessEntry",  
                "eks>DeleteAccessEntry",  
                "eks>AssociateAccessPolicy",  
                "iam>CreateServiceLinkedRole"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

To grant permissions to access the SageMaker AI console, use the sample policy provided at [Permissions required to use the Amazon SageMaker AI console](#).

To grant permissions to access the Amazon EC2 Systems Manager console, use the sample policy provided at [Using the AWS Systems Manager console](#) in the *AWS Systems Manager User Guide*.

You might also consider attaching the [AmazonSageMakerFullAccess](#) policy to the role; however, note that the AmazonSageMakerFullAccess policy grants permissions to the entire SageMaker API calls, features, and resources.

For guidance on IAM users in general, see [IAM users](#) in the *AWS Identity and Access Management User Guide*.

## IAM users for scientists

Scientists log into and run ML workloads on SageMaker HyperPod cluster nodes provisioned by cluster admins. For scientists in your AWS account, you should grant the permission "ssm:StartSession" to run the SSM start-session command. The following is a policy example for IAM users.

### Slurm

Add the following policy to grant SSM session permissions to connect to an SSM target for all resources. This allows you to access HyperPod clusters.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "ssm:StartSession",  
                "ssm:TerminateSession"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

### Amazon EKS

Grant the following IAM role permissions for data scientists to run hyperpod list-clusters and hyperpod connect-cluster commands among the HyperPod CLI commands. To learn more about the HyperPod CLI, see [the section called “Running jobs on HyperPod clusters”](#). It also includes SSM session permissions to connect to an SSM target for all resources. This allows you to access HyperPod clusters.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "DescribeHyperpodClusterPermissions",  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker:DescribeCluster"  
            ],  
            "Resource": "<hyperpod-cluster-arn>"  
        },  
        {  
            "Sid": "UseEksClusterPermissions",  
            "Effect": "Allow",  
            "Action": [  
                "eks:DescribeCluster",  
            ],  
            "Resource": "<eks-cluster-arn>"  
        },  
        {  
            "Sid": "ListClustersPermission",  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker>ListClusters"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "ssm:StartSession",  
                "ssm:TerminateSession"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

To grant data scientists IAM users or roles access to Kubernetes APIs in the cluster, see also [Grant IAM users and roles access to Kubernetes APIs in the Amazon EKS User Guide](#).

## IAM role for SageMaker HyperPod

For SageMaker HyperPod clusters to run and communicate with necessary AWS resources, you need to create an IAM role for HyperPod cluster to assume.

Start with attaching the managed role [the section called "AmazonSageMakerHyperPodServiceRolePolicy"](#). Given this AWS managed policy, SageMaker HyperPod cluster instance groups assume the role to communicate with Amazon CloudWatch, Amazon S3, and AWS Systems Manager Agent (SSM Agent). This managed policy is the minimum requirement for SageMaker HyperPod resources to run properly, so you must provide an IAM role with this policy to all instance groups.

 **Tip**

Depending on your preference on designing the level of permissions for multiple instance groups, you can also set up multiple IAM roles and attach them to different instance groups. When you set up your cluster user access to specific SageMaker HyperPod cluster nodes, the nodes assume the role with the selective permissions you manually attached. When you set up the access for scientists to specific cluster nodes through [AWS Systems Manager](#) (see also [the section called "Setting up AWS Systems Manager and Run As for cluster user access control"](#)), the cluster nodes assume the role with the selective permissions you manually attach.

After you are done with creating IAM roles, make notes of their names and ARNs. You use the roles when creating a SageMaker HyperPod cluster, granting the correct permissions required for each instance group to communicate with necessary AWS resources.

### Slurm

For HyperPod orchestrated with Slurm, you must attach the following managed policy to the SageMaker HyperPod IAM role.

- [AmazonSageMakerClusterInstanceRolePolicy](#)

### (Optional) Additional permissions for using SageMaker HyperPod with Amazon Virtual Private Cloud

If you want to use your own Amazon Virtual Private Cloud (VPC) instead of the default SageMaker AI VPC, you should add the following additional permissions to the IAM role for SageMaker HyperPod.

```
{  
    "Effect": "Allow",  
    "Action": [  
        "ec2:CreateNetworkInterface",  
        "ec2:CreateNetworkInterfacePermission",  
        "ec2>DeleteNetworkInterface",  
        "ec2>DeleteNetworkInterfacePermission",  
        "ec2:DescribeNetworkInterfaces",  
        "ec2:DescribeVpcs",  
        "ec2:DescribeDhcpOptions",  
        "ec2:DescribeSubnets",  
        "ec2:DescribeSecurityGroups",  
        "ec2:DetachNetworkInterface"  
    ],  
    "Resource": "*"  
}  
  
{  
    "Effect": "Allow",  
    "Action": "ec2:CreateTags",  
    "Resource": [  
        "arn:aws:ec2:*:*:network-interface/*"  
    ]  
}
```

The following list breaks down which permissions are needed to enable SageMaker HyperPod cluster functionalities when you configure the cluster with your own Amazon VPC.

- The following ec2 permissions are required to enable configuring a SageMaker HyperPod cluster with your VPC.

```
{  
    "Effect": "Allow",  
    "Action": [  
        "ec2:CreateNetworkInterface",  
        "ec2:CreateNetworkInterfacePermission",  
        "ec2>DeleteNetworkInterface",  
        "ec2>DeleteNetworkInterfacePermission",  
        "ec2:DescribeNetworkInterfaces",  
        "ec2:DescribeVpcs",  
        "ec2:DescribeDhcpOptions",  
        "ec2:DescribeSubnets",  
        "ec2:DescribeSecurityGroups",  
        "ec2:DetachNetworkInterface"  
    ],  
    "Resource": "*"  
}  
  
{  
    "Effect": "Allow",  
    "Action": "ec2:CreateTags",  
    "Resource": [  
        "arn:aws:ec2:*:*:network-interface/*"  
    ]  
}
```

```
        "ec2:DescribeVpcs",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups"
    ],
    "Resource": "*"
}
```

- The following ec2 permission is required to enable the [SageMaker HyperPod auto-resume functionality](#).

```
{
    "Effect": "Allow",
    "Action": [
        "ec2:DetachNetworkInterface"
    ],
    "Resource": "*"
}
```

- The following ec2 permission allows SageMaker HyperPod to create tags on the network interfaces within your account.

```
{
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": [
        "arn:aws:ec2:*:*:network-interface/*"
    ]
}
```

## Amazon EKS

For HyperPod orchestrated with Amazon EKS, you must attach the following managed policies to the SageMaker HyperPod IAM role.

- [AmazonSageMakerClusterInstanceRolePolicy](#)

In addition to the managed policies, attach the following permission policy to the role.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "ec2:AssignPrivateIpAddresses",  
                "ec2>CreateNetworkInterface",  
                "ec2>CreateNetworkInterfacePermission",  
                "ec2>DeleteNetworkInterface",  
                "ec2>DeleteNetworkInterfacePermission",  
                "ec2:DescribeNetworkInterfaces",  
                "ec2:DescribeVpcs",  
                "ec2:DescribeDhcpOptions",  
                "ec2:DescribeSubnets",  
                "ec2:DescribeSecurityGroups",  
                "ec2:DetachNetworkInterface",  
                "ec2:ModifyNetworkInterfaceAttribute",  
                "ec2:UnassignPrivateIpAddresses",  
                "ecr:BatchGetImage",  
                "ecr:GetAuthorizationToken",  
                "ecr:GetDownloadUrlForLayer",  
                "eks-auth:AssumeRoleForPodIdentity"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "ec2>CreateTags"  
            ],  
            "Resource": [  
                "arn:aws:ec2:*:*:network-interface/*"  
            ]  
        }  
    ]  
}
```

### Note

The "eks-auth:AssumeRoleForPodIdentity" permission is optional. It's required if you plan to use EKS Pod identity.

## SageMaker HyperPod service-linked role

For Amazon EKS support in SageMaker HyperPod, HyperPod creates a service-linked role with [the section called “AmazonSageMakerHyperPodServiceRolePolicy”](#) to monitor and support resiliency on your EKS cluster such as replacing nodes and restarting jobs.

### IAM policies for Amazon EKS

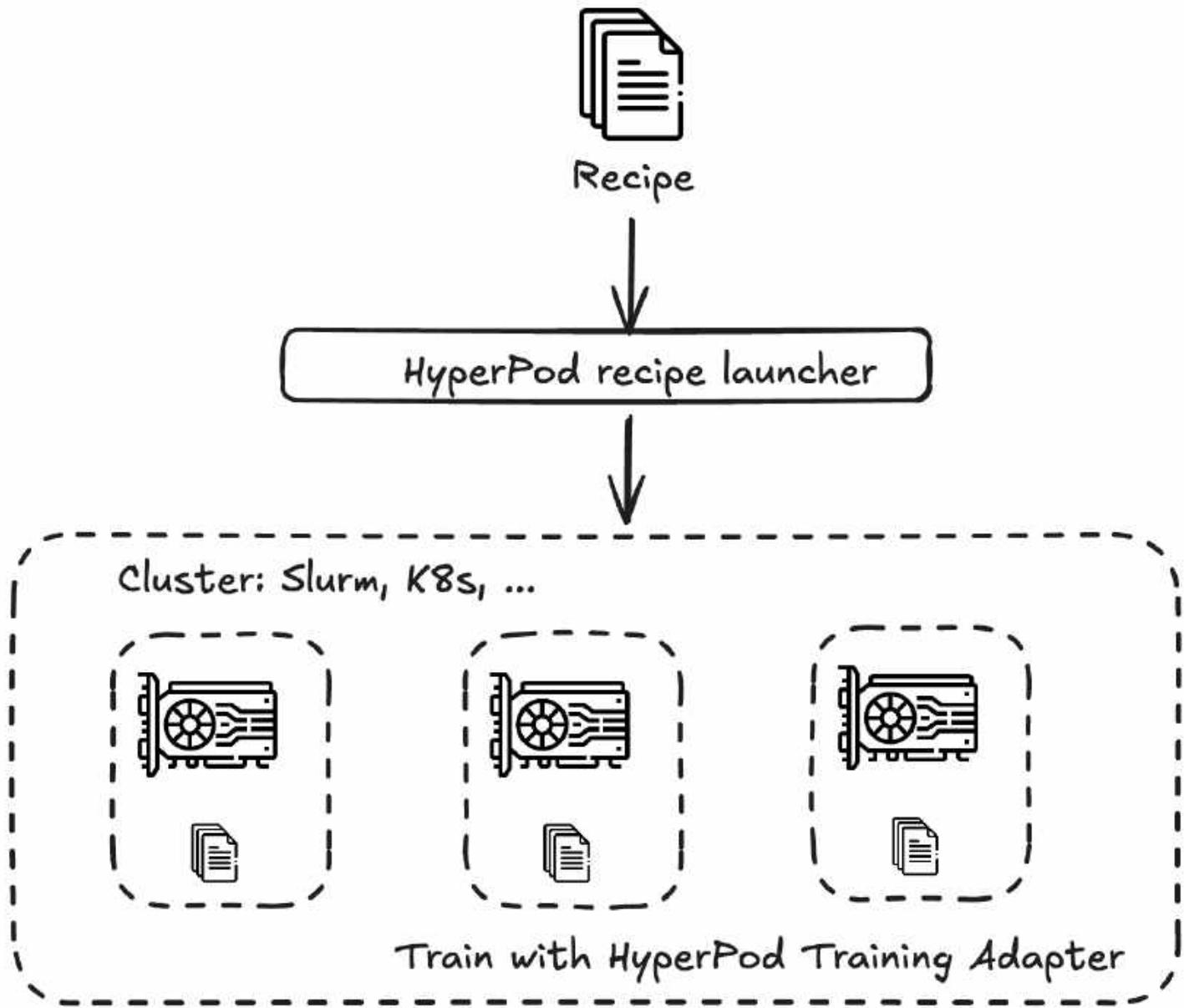
## SageMaker HyperPod recipes

Use Amazon SageMaker HyperPod recipes to get started with training and fine-tuning publicly available foundation models. To view the available recipes, see [SageMaker HyperPod recipes](#).

The recipes are pre-configured training configurations for the following model families:

- [Llama 3.1](#)
- [Llama 3.2](#)
- [Mistral](#)
- [Mixtral](#)

You can run recipes within SageMaker HyperPod or as SageMaker training jobs. You use the Amazon SageMaker HyperPod training adapter as the framework to help you run end-to-end training workflows. The training adapter is built on the [NVIDIA NeMo framework](#) and [NeuronX Distributed Training package](#). If you're familiar with using NeMo, the process of using the training adapter is the same. The training adapter runs the recipe on your cluster.



You can also train your own model by defining your own custom recipe.

The following tables outline the predefined recipes and launch scripts that SageMaker HyperPod currently supports.

## Available pre-training models, recipes, and launch scripts

Model	Size	Sequence	Nodes	Instance	Accelerator	Recipe	Script
Llama3.2	11b	8192	4	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.2	90b	8192	32	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.2	1b	8192	1	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.2	3b	8192	1	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	70b	16384	32	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	70b	16384	64	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	70b	8192	32	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	70b	8192	64	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3	70b	8192	16	ml.trn1.32xlarge	AWS TRN	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	8b	16384	16	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	8b	16384	32	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	8b	8192	16	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>

Model	Size	Sequence	Nodes	Instance	Accelerator	Recipe	Script
Llama3.1	8b	8192	32	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3	8b	8192	4	ml.trn1.3 2xlarge	AWS TRN	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	8b	8192	16	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	N/A
Mistral	7b	16384	16	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Mistral	7b	16384	32	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Mistral	7b	8192	16	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Mistral	7b	8192	32	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Mixtral	22b	16384	32	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Mixtral	22b	16384	64	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Mixtral	22b	8192	32	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Mixtral	22b	8192	64	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Mixtral	7b	16384	16	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>

Model	Size	Sequence	Nodes	Instance	Accelerator	Recipe	Script
Mixtral	7b	16384	32	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Mixtral	7b	8192	16	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Mixtral	7b	8192	32	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>

## Available fine-tuning models, recipes, and launch scripts

Model	Method	Size	Sequence length	Nodes	Instance	Accelerator	Recipe	Script
Llama3.1	QLoRA	405b	131072	2	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	LoRA	405b	16384	6	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	QLoRA	405b	16384	2	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	LoRA	405b	16384	6	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	QLoRA	405b	8192	2	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	SFT	70b	16384	16	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	LoRA	70b	16384	2	ml.p5.48x large	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>

Model	Method	Size	Sequence length	Nodes	Instance	Accelerator	Recipe	Script
Llama3.1	SFT	70b	8192	10	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	LoRA	70b	8192	1	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	SFT	8b	16384	1	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	LoRA	8b	16384	1	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	SFT	8b	8192	1	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	LoRA	8b	8192	1	ml.p5.48xlarge	Nvidia H100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	SFT	70b	8192	32	ml.p4d.24xlarge	Nvidia A100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	LoRA	70b	8192	20	ml.p4d.24xlarge	Nvidia A100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	SFT	8b	8192	4	ml.p4d.24xlarge	Nvidia A100	<a href="#">link</a>	<a href="#">link</a>
Llama3.1	LoRA	8b	8192	1	ml.p4d.24xlarge	Nvidia A100	<a href="#">link</a>	<a href="#">link</a>
Llama3	SFT	8b	8192	1	ml.trn1.32xlarge	AWS TRN	<a href="#">link</a>	<a href="#">link</a>

To get started with a tutorial, see [Tutorials](#).

## Topics

- [Tutorials](#)
- [Default Configurations](#)
- [Cluster-Specific Configurations](#)
- [Special Considerations](#)
- [Advanced Settings](#)
- [Appendix](#)

## Tutorials

The following quick-start tutorials help you get started with using the recipes for training:

- SageMaker HyperPod with Slurm Orchestration
  - [HyperPod Slurm cluster pre-training tutorial \(GPU\)](#)
  - [HyperPod Slurm cluster PEFT-Lora tutorial \(GPU\)](#)
  - [Trainium Slurm cluster pre-training tutorial](#)
- SageMaker HyperPod with K8s Orchestration
  - [Kubernetes cluster pre-training tutorial \(GPU\)](#)
  - [Trainium SageMaker training jobs pre-training tutorial](#)
- SageMaker training jobs
  - [SageMaker training jobs pre-training tutorial \(GPU\)](#)
  - [Trainium SageMaker training jobs pre-training tutorial](#)

### HyperPod Slurm cluster pre-training tutorial (GPU)

The following tutorial sets up Slurm environment and starts a training job on a Llama 8 billion parameter model.

#### Prerequisites

Before you start setting up your environment to run the recipe, make sure you have:

- Set up a HyperPod GPU Slurm cluster.
  - Your HyperPod Slurm cluster must have Nvidia Enroot and Pyxis enabled (these are enabled by default).

- A shared storage location. It can be an Amazon FSx file system or an NFS system that's accessible from the cluster nodes.
- Data in one of the following formats:
  - JSON
  - JSONGZ (Compressed JSON)
  - ARROW
- (Optional) You must get a HuggingFace token if you're using the model weights from HuggingFace for pre-training or fine-tuning. For more information about getting the token, see [User access tokens](#).

## HyperPod GPU Slurm environment setup

To initiate a training job on a HyperPod GPU Slurm cluster, do the following:

1. SSH into the head node of your Slurm cluster.
2. After you log in, set up the virtual environment. Make sure you're using Python 3.9 or greater.

```
#set up a virtual environment
python3 -m venv ${PWD}/venv
source venv/bin/activate
```

3. Clone the SageMaker HyperPod recipes and SageMaker HyperPod adapter repositories to a shared storage location.

```
git clone https://github.com/aws/sagemaker-hyperpod-training-adapter-for-nemo.git
git clone --recursive https://github.com/aws/sagemaker-hyperpod-recipes.git
cd sagemaker-hyperpod-recipes
pip3 install -r requirements.txt
```

4. Create a squash file using Enroot. To find the most recent release of the SMP container, see [Release notes for the SageMaker model parallelism library](#). To gain a deeper understanding of how to use the Enroot file, see [Build AWS-optimized Nemo-Launcher image](#).

```
REGION="region"  
IMAGE="658645717510.dkr.ecr.${REGION}.amazonaws.com/smdistributed-  
modelparallel:2.4.1-gpu-py311-cu121"  
aws ecr get-login-password --region ${REGION} | docker login --username AWS --  
password-stdin 658645717510.dkr.ecr.${REGION}.amazonaws.com
```

```
enroot import -o $PWD/smdistributed-modelparallel.sqsh dockerd://${IMAGE}  
mv $PWD/smdistributed-modelparallel.sqsh "/fsx/<any-path-in-the-shared-filesystem>"
```

5. To use the Enroot squash file to start training, use the following example to modify the `recipes_collection/config.yaml` file.

```
container: /fsx/path/to/your/smdistributed-modelparallel.sqsh
```

## Launch the training job

After you install the dependencies, start a training job from the `sagemaker-hyperpod-recipes/launcher_scripts` directory. You get the dependencies by cloning the [SageMaker HyperPod recipes repository](#):

First, pick your training recipe from Github, the model name is specified as part of the recipe. We use the `launcher_scripts/llama/run_hf_llama3_8b_seq16k_gpu_p5x16_pretrain.sh` script to launch a Llama 8b with sequence length 8192 pre-training recipe, `llama/hf_llama3_8b_seq16k_gpu_p5x16_pretrain`, in the following example.

- IMAGE: The container from the environment setup section.
- (Optional) You can provide the HuggingFace token if you need pre-trained weights from HuggingFace by setting the following key-value pair:

```
recipes.model.hf_access_token=<your_hf_token>
```

```
#!/bin/bash  
IMAGE="${YOUR_IMAGE}"  
SAGEMAKER_TRAINING_LAUNCHER_DIR="${SAGEMAKER_TRAINING_LAUNCHER_DIR:-$PWD}"  
  
TRAIN_DIR="${YOUR_TRAIN_DIR}" # Location of training dataset  
VAL_DIR="${YOUR_VAL_DIR}" # Location of validation dataset  
  
# experiment output directory  
EXP_DIR="${YOUR_EXP_DIR}"  
  
HYDRA_FULL_ERROR=1 python3 "${SAGEMAKER_TRAINING_LAUNCHER_DIR}/main.py" \  
  recipes=training/llama/hf_llama3_8b_seq16k_gpu_p5x16_pretrain \  
  base_results_dir="${SAGEMAKER_TRAINING_LAUNCHER_DIR}/results" \  
  recipes.run.name="hf_llama3_8b" \  
  
```

```
recipes.exp_manager.exp_dir="$EXP_DIR" \
recipes.model.data.train_dir="$TRAIN_DIR" \
recipes.model.data.val_dir="$VAL_DIR" \
container="${IMAGE}" \
+cluster.container_mounts.0="/fsx:/fsx"
```

After you've configured all the required parameters in the launcher script, you can run the script using the following command.

```
bash launcher_scripts/llama/run_hf_llama3_8b_seq16k_gpu_p5x16_pretrain.sh
```

For more information about the Slurm cluster configuration, see [Run a training job on HyperPod Slurm](#).

## HyperPod Slurm cluster PEFT-Lora tutorial (GPU)

The following tutorial sets up Slurm environment and starts a parameter-efficient fine-tuning (PEFT) job on a Llama 8 billion parameter model.

### Prerequisites

Before you start setting up your environment, make sure you have:

- Set up HyperPod GPU Slurm cluster
  - Your HyperPod Slurm cluster must have Nvidia Enroot and Pyxis enabled (these are enabled by default).
- A shared storage location. It can be an Amazon FSx file system or NFS system that's accessible from the cluster nodes.
- Data in one of the following formats:
  - JSON
  - JSONGZ (Compressed JSON)
  - ARROW
- (Optional) If you need the pre-trained weights from HuggingFace or if you're training a Llama 3.2 model, you must get the HuggingFace token before you start training. For more information about getting the token, see [User access tokens](#).

## Set up the HyperPod GPU Slurm environment

To initiate a training job on a Slurm cluster, do the following:

- SSH into the head node of your Slurm cluster.
- After you log in, set up the virtual environment. Make sure you're using Python 3.9 or greater.

```
#set up a virtual environment
python3 -m venv ${PWD}/venv
source venv/bin/activate
```

- Clone the SageMaker HyperPod recipes and SageMaker HyperPod adapter repositories to a shared storage location. The shared storage location can be an Amazon FSx file system or NFS system that's accessible from the cluster nodes.

```
git clone https://github.com/aws/sagemaker-hyperpod-training-adapter-for-nemo.git
git clone --recursive https://github.com/aws/sagemaker-hyperpod-recipes.git
cd sagemaker-hyperpod-recipes
pip3 install -r requirements.txt
```

- Create a squash file using Enroot. To find the most recent release of the SMP container, see [Release notes for the SageMaker model parallelism library](#). For more information about using the Enroot file, see [Build AWS-optimized Nemo-Launcher image](#).

```
REGION="<region>"
IMAGE="658645717510.dkr.ecr.${REGION}.amazonaws.com/smdistributed-
modelparallel:2.4.1-gpu-py311-cu121"
aws ecr get-login-password --region ${REGION} | docker login --username AWS --
password-stdin 658645717510.dkr.ecr.${REGION}.amazonaws.com
enroot import -o $PWD/smdistributed-modelparallel.sqsh dockerd://${IMAGE}
mv $PWD/smdistributed-modelparallel.sqsh "/fsx/<any-path-in-the-shared-filesystem>"
```

- To use the Enroot squash file to start training, use the following example to modify the `recipes_collection/config.yaml` file.

```
container: /fsx/path/to/your/smdistributed-modelparallel.sqsh
```

## Launch the training job

To launch a PEFT job for the Llama 8 billion parameter model with a sequence length of 8192 on a single Slurm compute node, set the launch script, `launcher_scripts/llama/run_hf_llama3_8b_seq8k_gpu_lora.sh`, to the following:

- IMAGE: The container from the environment setup section.
- HF\_MODEL\_NAME\_OR\_PATH: Define the name or the path of the pre-trained weights in the `hf_model_name_or_path` parameter of the recipe.
- (Optional) You can provide the HuggingFace token if you need pre-trained weights from HuggingFace by setting the following key-value pair:

```
recipes.model.hf_access_token=${HF_ACCESS_TOKEN}
```

```
#!/bin/bash
IMAGE="${YOUR_IMAGE}"
SAGEMAKER_TRAINING_LAUNCHER_DIR="${SAGEMAKER_TRAINING_LAUNCHER_DIR:-$PWD}"

TRAIN_DIR="${YOUR_TRAIN_DIR}" # Location of training dataset
VAL_DIR="${YOUR_VAL_DIR}" # Location of validation dataset

# experiment output directory
EXP_DIR="${YOUR_EXP_DIR}"
HF_ACCESS_TOKEN="${YOUR_HF_TOKEN}"
HF_MODEL_NAME_OR_PATH="${YOUR_HF_MODEL_NAME_OR_PATH}"

# Add hf_model_name_or_path and turn off synthetic_data
HYDRA_FULL_ERROR=1 python3 ${SAGEMAKER_TRAINING_LAUNCHER_DIR}/main.py \
    recipes=fine-tuning/llama/hf_llama3_8b_seq8k_gpu_lora \
    base_results_dir=${SAGEMAKER_TRAINING_LAUNCHER_DIR}/results \
    recipes.run.name="hf_llama3_lora" \
    recipes.exp_manager.exp_dir="$EXP_DIR" \
    recipes.model.data.train_dir="$TRAIN_DIR" \
    recipes.model.data.val_dir="$VAL_DIR" \
    recipes.model.hf_model_name_or_path="$HF_MODEL_NAME_OR_PATH" \
    container="${IMAGE}" \
    +cluster.container_mounts.0="/fsx:/fsx" \
    recipes.model.hf_access_token="${HF_ACCESS_TOKEN}"
```

After you've configured all the required parameters in the preceding script, you can initiate the training job by running it.

```
bash launcher_scripts/llama/run_hf_llama3_8b_seq8k_gpu_lora.sh
```

For more information about the Slurm cluster configuration, see [Run a training job on HyperPod Slurm](#).

### Trainium Slurm cluster pre-training tutorial

The following tutorial sets up a Trainium environment on a Slurm cluster and starts a training job on a Llama 8 billion parameter model.

#### Prerequisites

Before you start setting up your environment, make sure you have:

- Set up a SageMaker HyperPod Trainium Slurm cluster.
- A shared storage location. It can be an Amazon FSx file system or NFS system that's accessible from the cluster nodes.
- Data in one of the following formats:
  - JSON
  - JSONGZ (Compressed JSON)
  - ARROW
- (Optional) You must get a HuggingFace token if you're using the model weights from HuggingFace for pre-training or fine-tuning. For more information about getting the token, see [User access tokens](#).

### Set up the Trainium environment on the Slurm Cluster

To initiate a training job on a Slurm cluster, do the following:

- SSH into the head node of your Slurm cluster.
- After you log in, set up the Neuron environment. For information about setting up Neuron, see [Neuron setup steps](#). We recommend relying on the Deep learning AMI's that come pre-installed with Neuron's drivers, such as [Ubuntu 20 with DLAMI Pytorch](#).

- Clone the SageMaker HyperPod recipes repository to a shared storage location in the cluster. The shared storage location can be an Amazon FSx file system or NFS system that's accessible from the cluster nodes.

```
git clone --recursive https://github.com/aws/sagemaker-hyperpod-recipes.git  
cd sagemaker-hyperpod-recipes  
pip3 install -r requirements.txt
```

- Go through the following tutorial: [HuggingFace Llama3-8B Pretraining](#)
- Prepare a model configuration. The model configurations available in the Neuron repo. For the model configuration used the in this tutorial, see [llama3 8b model config](#)

## Launch the training job in Trainium

To launch a training job in Trainium, specify a cluster configuration and a Neuron recipe.

For example, to launch a llama3 8b pre-training job in Trainium, set the launch script, launcher\_scripts/llama/run\_hf\_llama3\_8b\_seq8k\_trn1x4\_pretrain.sh, to the following:

- MODEL\_CONFIG: The model config from the environment setup section
- (Optional) You can provide the HuggingFace token if you need pre-trained weights from HuggingFace by setting the following key-value pair:

```
recipes.model.hf_access_token=<your_hf_token>
```

```
#!/bin/bash

#Users should set up their cluster type in /recipes_collection/config.yaml

SAGEMAKER_TRAINING_LAUNCHER_DIR=${SAGEMAKER_TRAINING_LAUNCHER_DIR:-"$(pwd)"}

COMPILE=0
TRAIN_DIR="${TRAIN_DIR}" # Location of training dataset
MODEL_CONFIG="${MODEL_CONFIG}" # Location of config.json for the model

HYDRA_FULL_ERROR=1 python3 "${SAGEMAKER_TRAINING_LAUNCHER_DIR}/main.py" \
    base_results_dir="${SAGEMAKER_TRAINING_LAUNCHER_DIR}/results" \
    instance_type="trn1.32xlarge" \
    recipes.run.compile="$COMPILE" \
```

```
recipes.run.name="hf-llama3-8b" \
recipes.trainer.num_nodes=4 \
recipes=training/llama/hf_llama3_8b_seq8k_trn1x4_pretrain \
recipes.data.train_dir="$TRAIN_DIR" \
recipes.model.model_config="$MODEL_CONFIG"
```

To launch the training job, run the following command:

```
bash launcher_scripts/llama/run_hf_llama3_8b_seq8k_trn1x4_pretrain.sh
```

For more information about the Slurm cluster configuration, see [Run a training job on HyperPod Slurm](#).

## Kubernetes cluster pre-training tutorial (GPU)

There are two ways to launch a training job in a GPU Kubernetes cluster:

- (Recommended) [HyperPod command-line tool](#)
- The NeMo style launcher

### Prerequisites

Before you start setting up your environment, make sure you have:

- A HyperPod GPU Kubernetes cluster is setup properly.
- A shared storage location. It can be an Amazon FSx file system or NFS system that's accessible from the cluster nodes.
- Data in one of the following formats:
  - JSON
  - JSONGZ (Compressed JSON)
  - ARROW
- (Optional) You must get a HuggingFace token if you're using the model weights from HuggingFace for pre-training or fine-tuning. For more information about getting the token, see [User access tokens](#).

## GPU Kubernetes environment setup

To set up a GPU Kubernetes environment, do the following:

- Set up the virtual environment. Make sure you're using Python 3.9 or greater.

```
python3 -m venv ${PWD}/venv  
source venv/bin/activate
```

- Install dependencies using one of the following methods:

- (Recommended): [HyperPod command-line tool](#) method:

```
# install HyperPod command line tools  
git clone https://github.com/aws/sagemaker-hyperpod-cli  
cd sagemaker-hyperpod-cli  
pip3 install .
```

- SageMaker HyperPod recipes method:

```
# install SageMaker HyperPod Recipes.  
git clone --recursive git@github.com:aws/sagemaker-hyperpod-recipes.git  
cd sagemaker-hyperpod-recipes  
pip3 install -r requirements.txt
```

- [Set up kubectl and eksctl](#)
- [Install Helm](#)
- Connect to your Kubernetes cluster

```
aws eks update-kubeconfig --region "${CLUSTER_REGION}" --name "${CLUSTER_NAME}"  
hyperpod connect-cluster --cluster-name "${CLUSTER_NAME}" [--region  
"${CLUSTER_REGION}"] [--namespace <namespace>]
```

## Launch the training job with the SageMaker HyperPod CLI

We recommend using the SageMaker HyperPod command-line interface (CLI) tool to submit your training job with your configurations. The following example submits a training job for the hf\_llama3\_8b\_seq16k\_gpu\_p5x16\_pretrain model.

- `your_training_container`: A Deep Learning container. To find the most recent release of the SMP container, see [Release notes for the SageMaker model parallelism library](#).

- (Optional) You can provide the HuggingFace token if you need pre-trained weights from HuggingFace by setting the following key-value pair:

```
"recipes.model.hf_access_token": "<your_hf_token>"
```

```
hyperpod start-job --recipe training/llama/hf_llama3_8b_seq16k_gpu_p5x16_pretrain \
--persistent-volume-claims fsx-claim:data \
--override-parameters \
'{
"recipes.run.name": "hf-llama3-8b",
"recipes.exp_manager.exp_dir": "/data/<your_exp_dir>",
"container": "658645717510.dkr.ecr.<region>.amazonaws.com/smdistributed-
modelparallel:2.4.1-gpu-py311-cu121",
"recipes.model.data.train_dir": "<your_train_data_dir>",
"recipes.model.data.val_dir": "<your_val_data_dir>",
"cluster": "k8s",
"cluster_type": "k8s"
}'
```

After you've submitted a training job, you can use the following command to verify if you submitted it successfully.

```
kubectl get pods
NAME                      READY   STATUS        RESTARTS   AGE
hf-llama3-<your-alias>-worker-0   0/1     running      0          36s
```

If the STATUS is PENDING or ContainerCreating, run the following command to get more details.

```
kubectl describe pod <name of pod>
```

After the job STATUS changes to Running, you can examine the log by using the following command.

```
kubectl logs <name of pod>
```

The STATUS becomes Completed when you run `kubectl get pods`.

## Launch the training job with the recipes launcher

Alternatively, you can use the SageMaker HyperPod recipes to submit your training job. Using the recipes involves updating `k8s.yaml`, `config.yaml`, and running the launch script.

- In `k8s.yaml`, update `persistent_volume_claims`. It mounts the Amazon FSx claim to the `/data` directory of each computing pod

```
persistent_volume_claims:  
  - claimName: fsx-claim  
    mountPath: data
```

- In `config.yaml`, update `repo_url_or_path` under `git`.

```
git:  
  repo_url_or_path: <training_adapter_repo>  
  branch: null  
  commit: null  
  entry_script: null  
  token: null
```

- Update `launcher_scripts/llama/run_hf_llama3_8b_seq16k_gpu_p5x16_pretrain.sh`
  - `your_contrainer`: A Deep Learning container. To find the most recent release of the SMP container, see [Release notes for the SageMaker model parallelism library](#).
  - (Optional) You can provide the HuggingFace token if you need pre-trained weights from HuggingFace by setting the following key-value pair:

```
recipes.model.hf_access_token=<your_hf_token>
```

```
#!/bin/bash  
#Users should setup their cluster type in /recipes_collection/config.yaml  
REGION="<region>"  
IMAGE="658645717510.dkr.ecr.${REGION}.amazonaws.com/smdistributed-  
modelparallel:2.4.1-gpu-py311-cu121"  
SAGEMAKER_TRAINING_LAUNCHER_DIR=${SAGEMAKER_TRAINING_LAUNCHER_DIR:-$(pwd)}  
EXP_DIR="<your_exp_dir>" # Location to save experiment info including logging,  
# checkpoints, ect  
TRAIN_DIR="<your_training_data_dir>" # Location of training dataset  
VAL_DIR="<your_val_data_dir>" # Location of validation dataset
```

```
HYDRA_FULL_ERROR=1 python3 "${SAGEMAKER_TRAINING_LAUNCHER_DIR}/main.py" \
recipes=training/hf_llama3_8b_seq8k_gpu_p5x16_pretrain \
base_results_dir="${SAGEMAKER_TRAINING_LAUNCHER_DIR}/results" \
recipes.run.name="hf-llama3" \
recipes.exp_manager.exp_dir="$EXP_DIR" \
cluster=k8s \
cluster_type=k8s \
container="${IMAGE}" \
recipes.model.data.train_dir=$TRAIN_DIR \
recipes.model.data.val_dir=$VAL_DIR
```

- Launch the training job

```
bash launcher_scripts/llama/run_hf_llama3_8b_seq16k_gpu_p5x16_pretrain.sh
```

After you've submitted the training job, you can use the following command to verify if you submitted it successfully.

```
kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
hf-llama3-<your-alias>-worker-0	0/1	running	0	36s

If the STATUS is PENDING or ContainerCreating, run the following command to get more details.

```
kubectl describe pod <name-of-pod>
```

After the job STATUS changes to Running, you can examine the log by using the following command.

```
kubectl logs <name of pod>
```

The STATUS will turn to Completed when you run `kubectl get pods`.

For more information about the k8s cluster configuration, see [Run a training job on HyperPod k8s](#).

## Trainium Kubernetes cluster pre-training tutorial

You can use one of the following methods to start a training job in a Trainium Kubernetes cluster.

- (Recommended) [HyperPod command-line tool](#)
- The NeMo style launcher

### Prerequisites

Before you start setting up your environment, make sure you have:

- Set up a HyperPod Trainium Kubernetes cluster
- A shared storage location that can be an Amazon FSx file system or NFS system that's accessible from the cluster nodes.
- Data in one of the following formats:
  - JSON
  - JSONGZ (Compressed JSON)
  - ARROW
- (Optional) You must get a HuggingFace token if you're using the model weights from HuggingFace for pre-training or fine-tuning. For more information about getting the token, see [User access tokens](#).

## Set up your Trainium Kubernetes environment

To set up the Trainium Kubernetes environment, do the following:

1. Complete the steps in the following tutorial: [HuggingFace Llama3-8B Pretraining](#) starting from **Download the dataset**.
2. Prepare a model configuration. They're available in the Neuron repo. For this tutorial, you can use the llama3 8b model config.
3. Virtual environment setup. Make sure you're using Python 3.9 or greater.

```
python3 -m venv ${PWD}/venv  
source venv/bin/activate
```

4. Install the dependencies

- (Recommended) Use the following HyperPod command-line tool

```
# install HyperPod command line tools
git clone https://github.com/aws/sagemaker-hyperpod-cli
cd sagemaker-hyperpod-cli
pip3 install .
```

- If you're using SageMaker HyperPod recipes, specify the following

```
# install SageMaker HyperPod Recipes.
git clone --recursive git@github.com:aws/sagemaker-hyperpod-recipes.git
cd sagemaker-hyperpod-recipes
pip3 install -r requirements.txt
```

## 5. [Set up kubectl and eksctl](#)

## 6. [Install Helm](#)

## 7. Connect to your Kubernetes cluster

```
aws eks update-kubeconfig --region "${CLUSTER_REGION}" --name "${CLUSTER_NAME}"
hyperpod connect-cluster --cluster-name "${CLUSTER_NAME}" [--region
"${CLUSTER_REGION}"] [--namespace <namespace>]
```

## 8. Container: The [Neuron container](#)

### Launch the training job with the SageMaker HyperPod CLI

We recommend using the SageMaker HyperPod command-line interface (CLI) tool to submit your training job with your configurations. The following example submits a training job for the hf\_llama3\_8b\_seq8k\_trn1x4\_pretrain Trainium model.

- your\_neuron\_container: The [Neuron container](#).
- your\_model\_config: The model configuration from the environment setup section
- (Optional) You can provide the HuggingFace token if you need pre-trained weights from HuggingFace by setting the following key-value pair:

```
"recipes.model.hf_access_token": "<your_hf_token>"
```

```
hyperpod start-job --recipe training/llama/hf_llama3_8b_seq8k_trn1x4_pretrain \
--persistent-volume-claims fsx-claim:data \
--override-parameters \
'{
  "cluster": "k8s",
  "cluster_type": "k8s",
  "container": "<your_neuron_contrainer>",
  "recipes.run.name": "hf-llama3",
  "recipes.run.compile": 0,
  "recipes.model.model_config": "<your_model_config>",
  "instance_type": "trn1.32xlarge",
  "recipes.data.train_dir": "<your_train_data_dir>"
}'
```

After you've submitted a training job, you can use the following command to verify if you submitted it successfully.

```
kubectl get pods
NAME                      READY   STATUS    RESTARTS   AGE
hf-llama3-<your-alias>-worker-0   0/1     running   0          36s
```

If the STATUS is PENDING or ContainerCreating, run the following command to get more details.

```
kubectl describe pod <name of pod>
```

After the job STATUS changes to Running, you can examine the log by using the following command.

```
kubectl logs <name of pod>
```

The STATUS will turn to Completed when you run `kubectl get pods`.

## Launch the training job with the recipes launcher

Alternatively, use SageMaker HyperPod recipes to submit your training job. To submit the training job using a recipe, update `k8s.yaml` and `config.yaml`. Run the bash script for the model to launch it.

- In `k8s.yaml`, update `persistent_volume_claims` to mount the Amazon FSx claim to the `/data` directory in the compute nodes

```
persistent_volume_claims:  
  - claimName: fsx-claim  
    mountPath: data
```

- Update launcher\_scripts/llama/run\_hf\_llama3\_8b\_seq8k\_trn1x4\_pretrain.sh
  - your\_neuron\_contrainer: The container from the environment setup section
  - your\_model\_config: The model config from the environment setup section

(Optional) You can provide the HuggingFace token if you need pre-trained weights from HuggingFace by setting the following key-value pair:

```
recipes.model.hf_access_token=<your_hf_token>
```

```
#!/bin/bash  
#Users should set up their cluster type in /recipes_collection/config.yaml  
IMAGE="your\_neuron\_contrainer"  
MODEL_CONFIG="your\_model\_config"  
SAGEMAKER_TRAINING_LAUNCHER_DIR=${SAGEMAKER_TRAINING_LAUNCHER_DIR:-"$(pwd)"}  
TRAIN_DIR="your\_training\_data\_dir" # Location of training dataset  
VAL_DIR="your\_val\_data\_dir" # Location of talidation dataset  
  
HYDRA_FULL_ERROR=1 python3 "${SAGEMAKER_TRAINING_LAUNCHER_DIR}/main.py" \  
  recipes=training/llama/hf_llama3_8b_seq8k_trn1x4_pretrain \  
  base_results_dir="${SAGEMAKER_TRAINING_LAUNCHER_DIR}/results" \  
  recipes.run.name="hf-llama3-8b" \  
  instance_type=trn1.32xlarge \  
  recipes.model.model_config="$MODEL_CONFIG" \  
  cluster=k8s \  
  cluster_type=k8s \  
  container="${IMAGE}" \  
  recipes.data.train_dir=$TRAIN_DIR \  
  recipes.data.val_dir=$VAL_DIR
```

- Launch the job

```
bash launcher_scripts/llama/run_hf_llama3_8b_seq8k_trn1x4_pretrain.sh
```

After you've submitted a training job, you can use the following command to verify if you submitted it successfully.

```
kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
hf-llama3-<your-alias>-worker-0	0/1	running	0	36s

If the STATUS is at PENDING or ContainerCreating, run the following command to get more details.

```
kubectl describe pod <name of pod>
```

After the job STATUS changes to Running, you can examine the log by using the following command.

```
kubectl logs <name of pod>
```

The STATUS will turn to Completed when you run `kubectl get pods`.

For more information about the k8s cluster configuration, see [Trainium Kubernetes cluster pre-training tutorial](#).

## SageMaker training jobs pre-training tutorial (GPU)

This tutorial guides you through the process of setting up and running a pre-training job using SageMaker training jobs with GPU instances.

- Set up your environment
- Launch a training job using SageMaker HyperPod recipes

Before you begin, make sure you have following prerequisites.

### Prerequisites

Before you start setting up your environment, make sure you have:

- Amazon FSx file system or an Amazon S3 bucket where you can load the data and output the training artifacts.
- Requested a Service Quota for 1x ml.p4d.24xlarge and 1x ml.p5.48xlarge on Amazon SageMaker AI. To request a service quota increase, do the following:
  1. On the AWS Service Quotas console, navigate to AWS services,

2. Choose **Amazon SageMaker AI**.
  3. Choose one ml.p4d.24xlarge and one ml.p5.48xlarge instance.
- Create an AWS Identity and Access Management(IAM) role with the following managed policies to give SageMaker AI permissions to run the examples.
    - AmazonSageMakerFullAccess
    - AmazonEC2FullAccess
  - Data in one of the following formats:
    - JSON
    - JSONGZ (Compressed JSON)
    - ARROW
  - (Optional) You must get a HuggingFace token if you're using the model weights from HuggingFace for pre-training or fine-tuning. For more information about getting the token, see [User access tokens](#).

## GPU SageMaker training jobs environment setup

Before you run a SageMaker training job, configure your AWS credentials and preferred region by running the `aws configure` command. As an alternative to the `configure` command, you can provide your credentials through environment variables such as `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY`, and `AWS_SESSION_TOKEN`. For more information, see [SageMaker AI Python SDK](#).

We strongly recommend using a SageMaker AI Jupyter notebook in SageMaker AI JupyterLab to launch a SageMaker training job. For more information, see [SageMaker JupyterLab](#).

- (Optional) Set up the virtual environment and dependencies. If you are using a Jupyter notebook in Amazon SageMaker Studio, you can skip this step. Make sure you're using Python 3.9 or greater.

```
# set up a virtual environment
python3 -m venv ${PWD}/venv
source venv/bin/activate
# install dependencies after git clone.

git clone --recursive git@github.com:aws/sagemaker-hyperpod-recipes.git
cd sagemaker-hyperpod-recipes
```

```
pip3 install -r requirements.txt  
# Set the aws region.  
  
aws configure set <your_region>
```

- Install SageMaker AI Python SDK

```
pip3 install --upgrade sagemaker
```

- Container: The GPU container is set automatically by the SageMaker AI Python SDK. You can also provide your own container.

 **Note**

If you're running a Llama 3.2 multi-modal training job, the `transformers` version must be `4.45.2` or greater.

Append `transformers==4.45.2` to `requirements.txt` in `source_dir` only when you're using the SageMaker AI Python SDK. For example, append it if you're using it in a notebook in SageMaker AI JupyterLab.

If you are using HyperPod recipes to launch using cluster type `sm_jobs`, this will be done automatically.

## Launch the training job using a Jupyter Notebook

You can use the following Python code to run a SageMaker training job with your recipe. It leverages the PyTorch estimator from the [SageMaker AI Python SDK](#) to submit the recipe. The following example launches the `llama3-8b` recipe on the SageMaker AI Training platform.

```
import os  
import sagemaker,boto3  
from sagemaker.debugger import TensorBoardOutputConfig  
  
from sagemaker.pytorch import PyTorch  
  
sagemaker_session = sagemaker.Session()  
role = sagemaker.get_execution_role()
```

```
bucket = sagemaker_session.default_bucket()
output = os.path.join(f"s3://{bucket}", "output")
output_path = "<s3-URI>"

overrides = {
    "run": {
        "results_dir": "/opt/ml/model",
    },
    "exp_manager": {
        "exp_dir": "",
        "explicit_log_dir": "/opt/ml/output/tensorboard",
        "checkpoint_dir": "/opt/ml/checkpoints",
    },
    "model": {
        "data": {
            "train_dir": "/opt/ml/input/data/train",
            "val_dir": "/opt/ml/input/data/val",
        },
    },
}
}

tensorboard_output_config = TensorBoardOutputConfig(
    s3_output_path=os.path.join(output, 'tensorboard'),
    container_local_output_path=overrides["exp_manager"]["explicit_log_dir"]
)

estimator = PyTorch(
    output_path=output_path,
    base_job_name=f"llama-recipe",
    role=role,
    instance_type="ml.p5.48xlarge",
    training_recipe="training/llama/hf_llama3_8b_seq8k_gpu_p5x16_pretrain",
    recipe_overrides=recipe_overrides,
    sagemaker_session=sagemaker_session,
    tensorboard_output_config=tensorboard_output_config,
)
estimator.fit(inputs={"train": "s3 or fsx input", "val": "s3 or fsx input"}, wait=True)
```

The preceding code creates a PyTorch estimator object with the training recipe and then fits the model using the `fit()` method. Use the `training_recipe` parameter to specify the recipe you want to use for training.

**Note**

If you're running a Llama 3.2 multi-modal training job, the transformers version must be 4.45.2 or greater.

Append `transformers==4.45.2` to `requirements.txt` in `source_dir` only when you're using SageMaker AI Python SDK directly. For example, you must append the version to the text file when you're using a Jupyter notebook.

When you deploy the endpoint for a SageMaker training job, you must specify the image URI that you're using. If don't provide the image URI, the estimator uses the training image as the image for the deployment. The training images that SageMaker HyperPod provides don't contain the dependencies required for inference and deployment. The following is an example of how an inference image can be used for deployment:

```
from sagemaker import image_uris
container=image_uris.retrieve(framework='pytorch',region='us-
west-2',version='2.0',py_version='py310',image_scope='inference',
instance_type='ml.p4d.24xlarge')
predictor =
estimator.deploy(initial_instance_count=1,instance_type='ml.p4d.24xlarge',image_uri=container)
```

**Note**

Running the preceding code on Sagemaker notebook instance might need more than the default 5GB of storage that SageMaker AI JupyterLab provides. If you run into space not available issues, create a new notebook instance where you use a different notebook instance and increase the storage of the notebook.

## Launch the training job with the recipes launcher

Update the `./recipes_collection/cluster/sm_jobs.yaml` file to look like the following:

```
sm_jobs_config:
  output_path: <s3_output_path>
tensorboard_config:
  output_path: <s3_output_path>
```

```
  container_logs_path: /opt/ml/output/tensorboard # Path to logs on the container
  wait: True # Whether to wait for training job to finish
  inputs: # Inputs to call fit with. Set either s3 or file_system, not both.
    s3: # Dictionary of channel names and s3 URIs. For GPUs, use channels for train
        and validation.
      train: <s3_train_data_path>
      val: null
  additional_estimator_kwargs: # All other additional args to pass to estimator. Must
  be int, float or string.
  max_run: 180000
  enable_remote_debug: True
  recipe_overrides:
    exp_manager:
      explicit_log_dir: /opt/ml/output/tensorboard
    data:
      train_dir: /opt/ml/input/data/train
    model:
      model_config: /opt/ml/input/data/train/config.json
  compiler_cache_url: "<compiler_cache_url>"
```

Update `./recipes_collection/config.yaml` to specify `sm_jobs` in the cluster and `cluster_type`.

```
defaults:
- _self_
- cluster: sm_jobs # set to `slurm`, `k8s` or `sm_jobs`, depending on the desired
  cluster
- recipes: training/llama/hf_llama3_8b_seq8k_trn1x4_pretrain
cluster_type: sm_jobs # bcm, bcp, k8s or sm_jobs. If bcm, k8s or sm_jobs, it must
  match - cluster above.
```

Launch the job with the following command

```
python3 main.py --config-path recipes_collection --config-name config
```

For more information about configuring SageMaker training jobs, see [Run a training job on SageMaker training jobs](#).

## Trainium SageMaker training jobs pre-training tutorial

This tutorial guides you through the process of setting up and running a pre-training job using SageMaker training jobs with AWS Trainium instances.

- Set up your environment
- Launch a training job

Before you begin, make sure you have the following prerequisites.

### Prerequisites

Before you start setting up your environment, make sure you have:

- Amazon FSx file system or S3 bucket where you can load the data and output the training artifacts.
- Request a Service Quota for the `ml.trn1.32xlarge` instance on Amazon SageMaker AI. To request a service quota increase, do the following:

#### To request a service quota increase for `ml.trn1.32xlarge` instance

1. Navigate to the AWS Service Quotas console.
  2. Choose AWS services.
  3. Select JupyterLab.
  4. Specify one instance for `ml.trn1.32xlarge`.
- Create an AWS Identity and Access Management (IAM) role with the `AmazonSageMakerFullAccess` and `AmazonEC2FullAccess` managed policies. These policies provide Amazon SageMaker AI with permissions to run the examples.
  - Data in one of the following formats:
    - JSON
    - JSONGZ (Compressed JSON)
    - ARROW
  - (Optional) If you need the pre-trained weights from HuggingFace or if you're training a Llama 3.2 model, you must get the HuggingFace token before you start training. For more information about getting the token, see [User access tokens](#).

## Set up your environment for Trainium SageMaker training jobs

Before you run a SageMaker training job, use the `aws configure` command to configure your AWS credentials and preferred region . As an alternative, you can also provide your credentials through environment variables such as the `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY`, and `AWS_SESSION_TOKEN`. For more information, see [SageMaker AI Python SDK](#).

We strongly recommend using a SageMaker AI Jupyter notebook in SageMaker AI JupyterLab to launch a SageMaker training job. For more information, see [SageMaker JupyterLab](#).

- (Optional) If you are using Jupyter notebook in Amazon SageMaker Studio, you can skip running the following command. Make sure to use a version  $\geq$  python 3.9

```
# set up a virtual environment
python3 -m venv ${PWD}/venv
source venv/bin/activate
# install dependencies after git clone.

git clone --recursive git@github.com:aws/sagemaker-hyperpod-recipes.git
cd sagemaker-hyperpod-recipes
pip3 install -r requirements.txt
```

- Install SageMaker AI Python SDK

```
pip3 install --upgrade sagemaker
```

- If you are running a llama 3.2 multi-modal training job, the `transformers` version must be 4.45.2 or greater.
  - Append `transformers==4.45.2` to `requirements.txt` in `source_dir` only when you're using the SageMaker AI Python SDK.
  - If you are using HyperPod recipes to launch using `sm_jobs` as the cluster type, you don't have to specify the `transformers` version.
- Container: The Neuron container is set automatically by SageMaker AI Python SDK.

## Launch the training job with a Jupyter Notebook

You can use the following Python code to run a SageMaker training job using your recipe. It leverages the PyTorch estimator from the [SageMaker AI Python SDK](#) to submit the recipe. The following example launches the llama3-8b recipe as a SageMaker AI Training Job.

- `compiler_cache_url`: Cache to be used to save the compiled artifacts, such as an Amazon S3 artifact.

```
import os
import sagemaker,boto3
from sagemaker.debugger import TensorBoardOutputConfig

from sagemaker.pytorch import PyTorch

sagemaker_session = sagemaker.Session()
role = sagemaker.get_execution_role()

recipe_overrides = {
    "run": {
        "results_dir": "/opt/ml/model",
    },
    "exp_manager": {
        "explicit_log_dir": "/opt/ml/output/tensorboard",
    },
    "data": {
        "train_dir": "/opt/ml/input/data/train",
    },
    "model": {
        "model_config": "/opt/ml/input/data/train/config.json",
    },
    "compiler_cache_url": "<compiler_cache_url>"
}

tensorboard_output_config = TensorBoardOutputConfig(
    s3_output_path=os.path.join(output, 'tensorboard'),
    container_local_output_path=overrides["exp_manager"]["explicit_log_dir"]
)

estimator = PyTorch(
    output_path=output_path,
    base_job_name=f"llama-trn",
    role=role,
    instance_type="ml.trn1.32xlarge",
    sagemaker_session=sagemaker_session,
    training_recipe="training/llama/hf_llama3_70b_seq8k_trn1x16_pretrain",
    recipe_overrides=recipe_overrides,
)
```

```
estimator.fit(inputs={"train": "your-inputs"}, wait=True)
```

The preceding code creates a PyTorch estimator object with the training recipe and then fits the model using the `fit()` method. Use the `training_recipe` parameter to specify the recipe you want to use for training.

## Launch the training job with the recipes launcher

- Update `./recipes_collection/cluster/sm_jobs.yaml`
  - `compiler_cache_url`: The URL used to save the artifacts. It can be an Amazon S3 URL.

```
sm_jobs_config:  
    output_path: <s3_output_path>  
    wait: True  
    tensorboard_config:  
        output_path: <s3_output_path>  
        container_logs_path: /opt/ml/output/tensorboard # Path to logs on the container  
    wait: True # Whether to wait for training job to finish  
    inputs: # Inputs to call fit with. Set either s3 or file_system, not both.  
        s3: # Dictionary of channel names and s3 URIs. For GPUs, use channels for train  
        and validation.  
            train: <s3_train_data_path>  
            val: null  
    additional_estimator_kwargs: # All other additional args to pass to estimator.  
    Must be int, float or string.  
        max_run: 180000  
        image_uri: <your_image_uri>  
        enable_remote_debug: True  
        py_version: py39  
    recipe_overrides:  
        model:  
            exp_manager:  
                exp_dir: <exp_dir>  
            data:  
                train_dir: /opt/ml/input/data/train  
                val_dir: /opt/ml/input/data/val
```

- Update `./recipes_collection/config.yaml`

```
defaults:  
    - _self_
```

```
- cluster: sm_jobs  
- recipes: training/hf_llama3_8b_seq8k_trn1x4_pretrain  
cluster_type: sm_jobs # bcm, bcp, k8s or sm_jobs. If bcm, k8s or sm_jobs, it must  
match - cluster above.  
  
instance_type: ml.trn1.32xlarge  
base_results_dir: ~/sm_job/hf_llama3_8B # Location to store the results, checkpoints  
and logs.
```

- Launch the job with `main.py`

```
python3 main.py --config-path recipes_collection --config-name config
```

For more information about configuring SageMaker training jobs, see [SageMaker training jobs pre-training tutorial \(GPU\)](#).

## Default Configurations

This section outlines the essential components and settings required to initiate and customize your Large Language Model (LLM) training processes using SageMaker HyperPod. This section covers the key repositories, configuration files, and recipe structures that form the foundation of your training jobs. Understanding these default configurations is crucial for effectively setting up and managing your LLM training workflows, whether you're using pre-defined recipes or customizing them to suit your specific needs.

### Topics

- [Github Repositories](#)
- [General configuration](#)

## Github Repositories

To launch a training job, you utilize files from two distinct GitHub repositories:

- [SageMaker HyperPod recipes](#)
- [SageMaker HyperPod training adapter for NeMo](#)

These repositories contain essential components for initiating, managing, and customizing Large Language Model (LLM) training processes. You use the scripts from the repositories to set up and run the training jobs for your LLMs.

## SageMaker HyperPod recipe repository

Use the [SageMaker HyperPod recipes](#) repository to get a recipe.

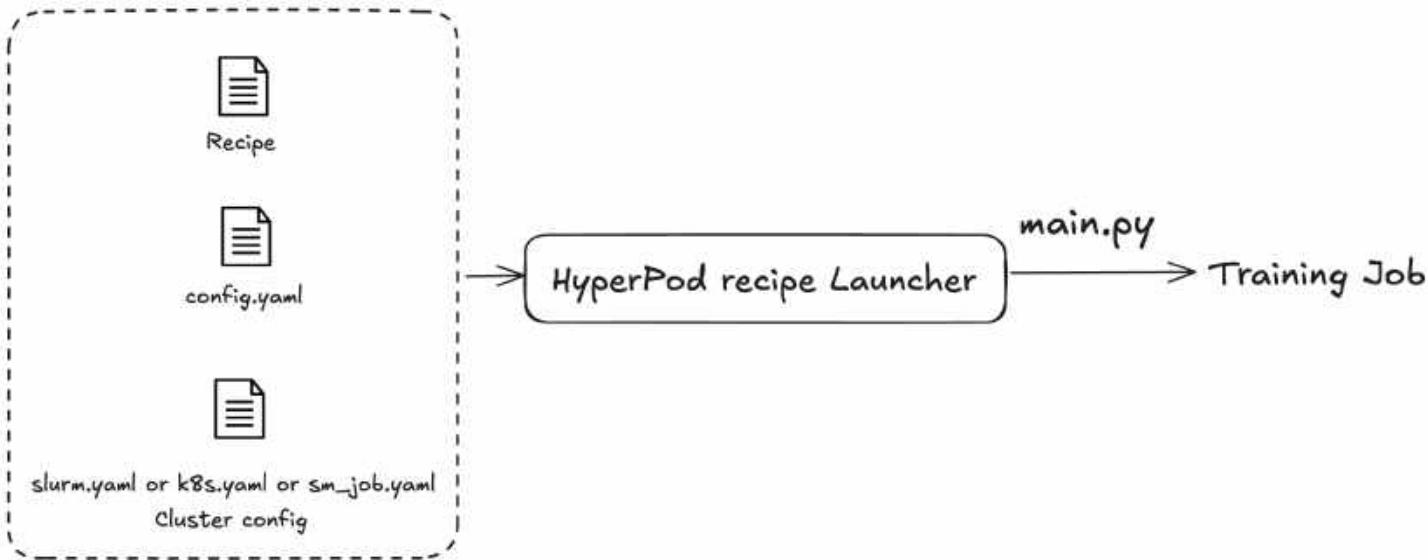
1. `main.py`: This file serves as the primary entry point for initiating the process of submitting a training job to either a cluster or a SageMaker training job.
2. `launcher_scripts`: This directory contains a collection of commonly used scripts designed to facilitate the training process for various Large Language Models (LLMs).
3. `recipes_collection`: This folder houses a compilation of pre-defined LLM recipes provided by the developers. Users can leverage these recipes in conjunction with their custom data to train LLM models tailored to their specific requirements.

You use the SageMaker HyperPod recipes to launch training or fine-tuning jobs. Regardless of the cluster you're using, the process of submitting the job is the same. For example, you can use the same script to submit a job to a Slurm or Kubernetes cluster. The launcher dispatches a training job based on three configuration files:

1. General Configuration (`config.yaml`): Includes common settings such as the default parameters or environment variables used in the training job.
2. Cluster Configuration (`cluster`): For training jobs using clusters only. If you're submitting a training job to a Kubernetes cluster, you might need to specify information such as volume, label, or restart policy. For Slurm clusters, you might need to specify the Slurm job name. All the parameters are related to the specific cluster that you're using.
3. Recipe (`recipes`): Recipes contain the settings for your training job, such as the model types, sharding degree, or dataset paths. For example, you can specify Llama as your training model and train it using model or data parallelism techniques like Fully Sharded Distributed Parallel (FSDP) across eight machines. You can also specify different checkpoint frequencies or paths for your training job.

After you've specified a recipe, you run the launcher script to specify an end-to-end training job on a cluster based on the configurations through the `main.py` entry point. For each recipe that you use, there are accompanying shell scripts located in the `launch_scripts` folder. These examples guide you through submitting and initiating training jobs. The following figure illustrates how a

SageMaker HyperPod recipe launcher submits a training job to a cluster based on the preceding. Currently, the SageMaker HyperPod recipe launcher is built on top of the Nvidia NeMo Framework Launcher. For more information, see [NeMo Launcher Guide](#).



## SageMaker HyperPod recipe adapter

The SageMaker HyperPod training adapter is a training framework. You can use it to manage the entire lifecycle of your training jobs. Use the adapter to distribute the pre-training or fine-tuning of your models across multiple machines. The adapter uses different parallelism techniques to distribute the training. It also handles the implementation and management of saving the checkpoints. For more details, see [Advanced Settings](#).

Use the [SageMaker HyperPod recipe adapter repository](#) to use the recipe adapter.

1. **src:** This directory contains the implementation of Large-scale Language Model (LLM) training, encompassing various features such as model parallelism, mixed-precision training, and checkpointing management.
2. **examples:** This folder provides a collection of examples demonstrating how to create an entry point for training an LLM model, serving as a practical guide for users.

## General configuration

The config.yaml file specifies the training recipe and the cluster. It also includes runtime configurations such as environment variables for the training job.

defaults:

```
- _self_
- cluster: slurm
- recipes: training/llama/hf_llama3_8b_seq8192_gpu
instance_type: p5.48xlarge
git:
  repo_url_or_path: null
  branch: null
  commit: null
  entry_script: null
  token: null
env_vars:
  NCCL_DEBUG: WARN
```

You can modify the following parameters in config.yaml:

1. **defaults**: Specify your default settings, such as the default cluster or default recipes.
2. **instance\_type**: Modify the Amazon EC2 instance type to match the instance type that you're using.
3. **git**: Specify the location of the SageMaker HyperPod recipe adapter repository for the training job.
4. **env\_vars**: You can specify the environment variables to be passed into your runtime training job. For example, you can adjust the logging level of NCCL by specifying the NCCL\_DEBUG environment variable.

The recipe is the core configuration that defines your training job architecture. This file includes many important pieces of information for your training job, such as the following:

- Whether to use model parallelism
- The source of your datasets
- Mixed precision training
- Checkpointing-related configurations

You can use the recipes as-is. You can also use the following information to modify them.

## run

The following is the basic run information for running your training job.

```
run:  
  name: llama-8b  
  results_dir: ${base_results_dir}/${.name}  
  time_limit: "6-00:00:00"  
  model_type: hf
```

1. **name:** Specify the name for your training job in the configuration file.
2. **results\_dir:** You can specify the directory where the results of your training job are stored.
3. **time\_limit:** You can set a maximum training time for your training job to prevent it from occupying hardware resources for too long.
4. **model\_type:** You can specify the type of model you are using. For example, you can specify `hf` if your model is from HuggingFace.

## exp\_manager

The `exp_manager` configures the experiment. With the `exp_manager`, you can specify fields such as the output directory or checkpoint settings. The following is an example of how you can configure the `exp_manager`.

```
exp_manager:  
  exp_dir: null  
  name: experiment  
  create_tensorboard_logger: True
```

1. **exp\_dir:** The experiment directory includes the standard output and standard error files for your training job. By default, it uses your current directory.
2. **name:** The experiment name used to identify your experiment under the `exp_dir`.
3. **create\_tensorboard\_logger:** Specify `True` or `False` to enable or disable the TensorBoard logger.

## Checkpointing

Here are three types of checkpointing we support:

- Auto checkpointing

- Manual checkpointing
- Full checkpointing

## Auto checkpointing

If you're saving or loading checkpoints that are automatically managed by the SageMaker HyperPod recipe adapter, you can enable `auto_checkpoint`. To enable `auto_checkpoint`, set `enabled` to `True`. You can use auto checkpointing for both training and fine-tuning. You can use auto checkpointing for both shared file systems and Amazon S3.

```
exp_manager
  checkpoint_dir: ${recipes.exp_manager.exp_dir}/checkpoints/
  auto_checkpoint:
    enabled: True
```

Auto checkpoint is saving the `local_state_dict` asynchronously with an automatically computed optimal saving interval.

### Note

Under this checkpointing mode, the auto saved checkpointing doesn't support re-sharding between training runs. To resume from the latest auto saved checkpoint, you must preserve the same shard degrees. You don't need to specify extra information to auto resume.

## Manual checkpointing

You can modify `checkpoint_callback_params` to asynchronously save an intermediate checkpoint in `shared_state_dict`. For example, you can specify the following configuration to enable sharded checkpointing every 10 steps and keep the latest 3 checkpoints.

Sharded checkpointing allows you to change the shard degrees between training runs and load the checkpoint by setting `resume_from_checkpoint`.

### Note

- If it is a PEFT fine tuning, sharded checkpointing doesn't support Amazon S3.
- Auto and manual checkpointing are mutually exclusive.

- Only FSDP shard degrees and replication degrees changes are allowed.

```
exp_manager:  
  checkpoint_callback_params:  
    # Set save_top_k = 0 to disable sharded checkpointing  
    save_top_k: 3  
    every_n_train_steps: 10  
    monitor: "step"  
    mode: "max"  
    save_last: False  
  resume_from_checkpoint: ${recipes.exp_manager.exp_dir}/checkpoints/
```

To learn more about checkpointing, see [Checkpointing using SMP](#).

## Full checkpointing

The exported full\_state\_dict checkpoint can be used for inference or fine tuning. You can load a full checkpoint through hf\_model\_name\_or\_path. Under this mode, only the model weights are saved.

To export the full\_state\_dict model, you can set the following parameters.

### Note

Currently, full checkpointing isn't supported for Amazon S3 checkpointing. You can't set the S3 path for exp\_manager.checkpoint\_dir if you're enabling full checkpointing. However, you can set exp\_manager.export\_full\_model.final\_export\_dir to a specific directory on your local filesystem while setting exp\_manager.checkpoint\_dir to an Amazon S3 path.

```
exp_manager:  
  export_full_model:  
    # Set every_n_train_steps = 0 to disable full checkpointing  
    every_n_train_steps: 0  
    save_last: True  
    final_export_dir : null
```

## model

Define various aspects of your model architecture and training process. This includes settings for model parallelism, precision, and data handling. Below are the key components you can configure within the model section:

### model parallelism

After you've specified the recipe, you define the model that you're training. You can also define the model parallelism. For example, you can define tensor\_model\_parallel\_degree. You can enable other features like training with FP8 precision. For example, you can train a model with tensor parallelism and context parallelism:

```
model:  
  model_type: llama_v3  
  # Base configs  
  train_batch_size: 4  
  val_batch_size: 1  
  seed: 12345  
  grad_clip: 1.0  
  
  # Model parallelism  
  tensor_model_parallel_degree: 4  
  expert_model_parallel_degree: 1  
  context_parallel_degree: 2
```

To gain a better understanding of different types of model parallelism techniques, you can refer to the following approaches:

1. [the section called "Tensor parallelism"](#)
2. [the section called "Expert parallelism"](#)
3. [the section called "Context parallelism"](#)
4. [the section called "Hybrid sharded data parallelism"](#)

## FP8

To enable FP8 (8-bit floating-point precision), you can specify the FP8-related configuration in the following example:

```
model:
```

```
# FP8 config
fp8: True
fp8_amax_history_len: 1024
fp8_amax_compute_algo: max
```

It's important to note that the FP8 data format is currently supported only on the P5 instance type. If you are using an older instance type, such as P4, disable the FP8 feature for your model training process. For more information about FP8, see [Mixed precision training](#).

## data

You can specify your custom datasets for your training job by adding the data paths under data. The data module in our system supports the following data formats:

1. JSON
2. JSONGZ (Compressed JSON)
3. ARROW

However, you are responsible for preparing your own pre-tokenized dataset. If you're an advanced user with specific requirements, there is also an option to implement and integrate a customized data module. For more information on HuggingFace datasets, see [Datasets](#).

```
model:
  data:
    train_dir: /path/to/your/train/data
    val_dir: /path/to/your/val/data
    dataset_type: hf
    use_synthetic_data: False
```

You can specify how you're training the model. By default, the recipe uses pre-training instead of fine-tuning. The following example configures the recipe to run a fine-tuning job with LoRA (Low-Rank Adaptation).

```
model:
  # Fine tuning config
  do_finetune: True
  # The path to resume from, needs to be HF compatible
  hf_model_name_or_path: null
  hf_access_token: null
  # PEFT config
```

```
peft:  
  peft_type: lora  
  rank: 32  
  alpha: 16  
  dropout: 0.1
```

For information about the recipes, see [SageMaker HyperPod recipes](#).

## Cluster-Specific Configurations

SageMaker HyperPod offers flexibility in running training jobs across different cluster environments. Each environment has its own configuration requirements and setup process. This section outlines the steps and configurations needed for running training jobs in SageMaker HyperPod Slurm, SageMaker HyperPod k8s, and SageMaker training jobs. Understanding these configurations is crucial for effectively leveraging the power of distributed training in your chosen environment.

You can use a recipe in the following cluster environments:

- SageMaker HyperPod Slurm Orchestration
- SageMaker HyperPod Amazon Elastic Kubernetes Service Orchestration
- SageMaker training jobs

To launch a training job in a cluster, set and install the corresponding cluster configuration and environment.

### Topics

- [Run a training job on HyperPod Slurm](#)
- [Run a training job on HyperPod k8s](#)
- [Run a SageMaker training job](#)

### Run a training job on HyperPod Slurm

SageMaker HyperPod Recipes supports submitting a training job to a GPU/Trainium slurm cluster. Before you submit the training job, update the cluster configuration. Use one of the following methods to update the cluster configuration:

- `Modify slurm.yaml`

- Override it through the command line

After you've updated the cluster configuration, install the environment.

## Configure the cluster

To submit a training job to a Slurm cluster, specify the Slurm-specific configuration. Modify `slurm.yaml` to configure the Slurm cluster. The following is an example of a Slurm cluster configuration. You can modify this file for your own training needs:

```
job_name_prefix: 'sagemaker-'
slurm_create_submission_file_only: False
stderr_to_stdout: True
srun_args:
  # - "--no-container-mount-home"
slurm_docker_cfg:
  docker_args:
    # - "--runtime=nvidia"
  post_launch_commands:
container_mounts:
  - "/fsx:/fsx"
```

1. `job_name_prefix`: Specify a job name prefix to easily identify your submissions to the Slurm cluster.
2. `slurm_create_submission_file_only`: Set this configuration to True for a dry run to help you debug.
3. `stderr_to_stdout`: Specify whether you're redirecting your standard error (stderr) to standard output (stdout).
4. `srun_args`: Customize additional srun configurations, such as excluding specific compute nodes. For more information, see the srun documentation.
5. `slurm_docker_cfg`: The SageMaker HyperPod recipe launcher launches a Docker container to run your training job. You can specify additional Docker arguments within this parameter.
6. `container_mounts`: Specify the volumes you're mounting into the container for the recipe launcher, for your training jobs to access the files in those volumes.

## Run a training job on HyperPod k8s

SageMaker HyperPod Recipes supports submitting a training job to a GPU/Trainium Kubernetes cluster. Before you submit the training job do one of the following:

- Modify the `k8s.yaml` cluster configuration file
- Override the cluster configuration through the command line

After you've done either of the preceding steps, install the corresponding environment.

### Configure the cluster using `k8s.yaml`

To submit a training job to a Kubernetes cluster, you specify Kubernetes-specific configurations. The configurations include the cluster namespace or the location of the persistent volume.

```
pullPolicy: Always
restartPolicy: Never
namespace: default
persistent_volume_claims:
  - null
```

1. `pullPolicy`: You can specify the pull policy when you submit a training job. If you specify "Always," the Kubernetes cluster always pulls your image from the repository. For more information, see [Image pull policy](#).
2. `restartPolicy`: Specify whether to restart your training job if it fails.
3. `namespace`: You can specify the Kubernetes namespace where you're submitting the training job.
4. `persistent_volume_claims`: You can specify a shared volume for your training job for all training processes to access the files in the volume.

## Run a SageMaker training job

SageMaker HyperPod Recipes supports submitting a SageMaker training job. Before you submit the training job, you must update the cluster configuration, `sm_job.yaml`, and install corresponding environment.

## Use your recipe as a SageMaker training job

You can use your recipe as a SageMaker training job if you aren't hosting a cluster. You must modify the SageMaker training job configuration file, `sm_job.yaml`, to run your recipe.

```
sm_jobs_config:  
  output_path: null  
  tensorboard_config:  
    output_path: null  
    container_logs_path: null  
  wait: True  
  inputs:  
    s3:  
      train: null  
      val: null  
    file_system:  
      directory_path: null  
  additional_estimator_kwargs:  
    max_run: 1800
```

1. `output_path`: You can specify where you're saving your model to an Amazon S3 URL.
2. `tensorboard_config`: You can specify a TensorBoard related configuration such as the output path or TensorBoard logs path.
3. `wait`: You can specify whether you're waiting for the job to be completed when you submit your training job.
4. `inputs`: You can specify the paths for your training and validation data. The data source can be from a shared filesystem such as Amazon FSx or an Amazon S3 URL.
5. `additional_estimator_kwargs`: Additional estimator arguments for submitting a training job to the SageMaker training job platform. For more information, see [Algorithm Estimator](#).

## Special Considerations

When you're using a Amazon SageMaker HyperPod recipes, there are some factors that can impact the process of model training.

- The `transformers` version must be `4.45.2` or greater for Llama 3.2. If you're using a Slurm or K8s workflow, the version is automatically updated.
- Mixtral does not support 8-bit floating point precision (FP8)

- Amazon EC2 p4 instance does not support FP8

## Advanced Settings

The SageMaker HyperPod recipe adapter is built on top of the Nvidia Nemo and Pytorch-lightning frameworks. If you've already used these frameworks, integrating your custom models or features into the SageMaker HyperPod recipe adapter is a similar process. In addition to modifying the recipe adapter, you can change your own pre-training or fine-tuning script. For guidance on writing your custom training script, see [examples](#).

### Use the SageMaker HyperPod adapter to create your own model

Within the recipe adapter, you can customize the following files in the following locations:

1. `collections/data`: Contains a module responsible for loading datasets. Currently, it only supports datasets from HuggingFace. If you have more advanced requirements, the code structure allows you to add custom data modules within the same folder.
2. `collections/model`: Includes the definitions of various language models. Currently, it supports common large language models like Llama, Mixtral, and Mistral. You have the flexibility to introduce your own model definitions within this folder.
3. `collections/parts`: This folder contains strategies for training models in a distributed manner. One example is the Fully Sharded Data Parallel (FSDP) strategy, which allows for sharding a large language model across multiple accelerators. Additionally, the strategies support various forms of model parallelism. You also have the option to introduce your own customized training strategies for model training.
4. `utils`: Contains various utilities aimed at facilitating the management of a training job. It serves as a repository where for your own tools. You can use your own tools for tasks such as troubleshooting or benchmarking. You can also add your own personalized PyTorch Lightning callbacks within this folder. You can use PyTorch Lightning callbacks to seamlessly integrate specific functionalities or operations into the training lifecycle.
5. `conf`: Contains the configuration schema definitions used for validating specific parameters in a training job. If you introduce new parameters or configurations, you can add your customized schema to this folder. You can use the customized schema to define the validation rules. You can validate data types, ranges, or any other parameter constraint. You can also define your own custom schema to validate the parameters.

## Appendix

Use the following information to get information about monitoring and analyzing training results.

### Monitor training results

Monitoring and analyzing training results is essential for developers to assess convergence and troubleshoot issues. SageMaker HyperPod recipes offer Tensorboard integration to analyze training behavior. To address the challenges of profiling large distributed training jobs, these recipes also incorporate VizTracer. VizTracer is a low-overhead tool for tracing and visualizing Python code execution. For more information about VizTracer, see [VizTracer](#).

The following sections guide you through the process of implementing these features in your SageMaker HyperPod recipes.

### Tensorboard

Tensorboard is a powerful tool for visualizing and analyzing the training process. To enable Tensorboard, modify your recipe by setting the following parameter:

```
exp_manager:  
  exp_dir: null  
  name: experiment  
  create_tensorboard_logger: True
```

After you enable the Tensorboard logger, the training logs are generated and stored within the experiment directory. The experiment directed is defined in `exp_manager.exp_dir`. To access and analyze these logs locally, use the following procedure:

#### To access and analyze logs

1. Download the Tensorboard experiment folder from your training environment to your local machine.
2. Open a terminal or command prompt on your local machine.
3. Navigate to the directory containing the downloaded experiment folder.
4. Launch Tensorboard with the following the command.

```
tensorboard --port=<port> --bind_all --logdir experiment.
```

5. Open your web browser and visit `http://localhost:8008`.

You can now see the status and visualizations of your training jobs within the Tensorboard interface. Seeing the status and visualizations helps you monitor and analyze the training process. Monitoring and analyzing the training process helps you gain insights into the behavior and performance of your models. For more information about how you monitor and analyze the training with Tensorboard, see the [NVIDIA NeMo Framework User Guide](#).

## VizTracer

To enable VizTracer, you can modify your recipe by setting the `model.viztracer.enabled` parameter to true. For example, you can update your `llama` recipe to enable VizTracer by adding the following configuration:

```
model:  
  viztracer:  
    enabled: true
```

After the training has completed, your VizTracer profile is in the experiment folder `exp_dir/result.json`. To analyze your profile, you can download it and open it using the `vizviewer` tool:

```
vizviewer --port <port> result.json
```

This command launches the `vizviewer` on port 9001. You can view your VizTracer by specifying `http://localhost:<port>` in your browser. After you open VizTracer, you begin analyzing the training. For more information about using VizTracer, see VizTracer documentation.

## SageMaker JumpStart versus SageMaker HyperPod

While SageMaker JumpStart provides fine-tuning capabilities, the SageMaker HyperPod recipes provide the following:

- Additional fine-grained control over the training loop
- Recipe customization for your own models and data
- Support for model parallelism

Use the SageMaker HyperPod recipes when you need access to the model's hyperparameters, multi-node training, and customization options for the training loop.

For more information about fine-tuning your models in SageMaker JumpStart, see [Fine-tune publicly available foundation models with the JumpStartEstimator class](#)

# Orchestrating SageMaker HyperPod clusters with Slurm

Slurm support in SageMaker HyperPod helps you provision resilient clusters for running machine learning (ML) workloads and developing state-of-the-art models such as large language models (LLMs), diffusion models, and foundation models (FMs). It accelerates development of FMs by removing undifferentiated heavy-lifting involved in building and maintaining large-scale compute clusters powered by thousands of accelerators such as AWS Trainium and NVIDIA A100 and H100 Graphical Processing Units (GPUs). When accelerators fail, the resiliency features of SageMaker HyperPod monitors the cluster instances automatically detect and replace the faulty hardware on the fly so that you can focus on running ML workloads. Additionally, with lifecycle configuration support in SageMaker HyperPod, you can customize your computing environment to best suit your needs and configure it with the Amazon SageMaker AI distributed training libraries to achieve optimal performance on AWS.

## Operating clusters

You can create, configure, and maintain SageMaker HyperPod clusters graphically through the console user interface (UI) and programmatically through the AWS command line interface (CLI) or AWS SDK for Python (Boto3). With Amazon VPC, you can secure the cluster network and also take advantage of configuring your cluster with resources in your VPC, such as Amazon FSx for Lustre, which offers the fastest throughput. You can also give different IAM roles to cluster instance groups, and limit actions that your cluster resources and users can operate. To learn more, see [the section called "SageMaker HyperPod operation"](#).

## Configuring your ML environment

SageMaker HyperPod runs [the section called "SageMaker HyperPod DLAMI"](#), which sets up an ML environment on the HyperPod clusters. You can configure additional customizations to the DLAMI by providing lifecycle scripts to support your use case. To learn more about how to set up lifecycle scripts, see [the section called "Getting started with SageMaker HyperPod"](#) and [the section called "Customize SageMaker HyperPod clusters using lifecycle scripts"](#).

## Scheduling jobs

After you successfully create a HyperPod cluster, cluster users can log into the cluster nodes (such as head or controller node, log-in node, and worker node) and schedule jobs for running machine learning workloads. To learn more, see [the section called "Jobs on HyperPod clusters"](#).

## Resiliency against hardware failures

SageMaker HyperPod runs health checks on cluster nodes and provides a workload auto-resume functionality. With the cluster resiliency features of HyperPod, you can resume your workload from the last checkpoint you saved, after faulty nodes are replaced with healthy ones in clusters with more than 16 nodes. To learn more, see [the section called “Cluster resiliency”](#).

## Logging and managing clusters

You can find SageMaker HyperPod resource utilization metrics and lifecycle logs in Amazon CloudWatch, and manage SageMaker HyperPod resources by tagging them. Each CreateCluster API run creates a distinct log stream, named in <cluster-name>-<timestamp> format. In the log stream, you can check the host names, the name of failed lifecycle scripts, and outputs from the failed scripts such as `stdout` and `stderr`. For more information, see [the section called “Cluster management”](#).

## Compatible with SageMaker AI tools

Using SageMaker HyperPod, you can configure clusters with AWS optimized collective communications libraries offered by SageMaker AI, such as the [SageMaker AI distributed data parallelism \(SMDDP\) library](#). The SMDDP library implements the AllGather operation optimized to the AWS compute and network infrastructure for the most performant SageMaker AI machine learning instances powered by NVIDIA A100 GPUs. To learn more, see [the section called “Run distributed training workloads with Slurm on HyperPod”](#).

## Topics

- [Tutorial for getting started with SageMaker HyperPod](#)
- [SageMaker HyperPod operation](#)
- [Customize SageMaker HyperPod clusters using lifecycle scripts](#)
- [Jobs on SageMaker HyperPod clusters](#)
- [SageMaker HyperPod cluster resources monitoring](#)
- [SageMaker HyperPod cluster resiliency](#)
- [SageMaker HyperPod cluster management](#)
- [SageMaker HyperPod FAQ](#)

## Tutorial for getting started with SageMaker HyperPod

Get started with creating your first SageMaker HyperPod cluster and learn the cluster operation functionalities of SageMaker HyperPod. You can create a SageMaker HyperPod cluster through

the SageMaker AI console UI or the AWS CLI commands. This tutorial shows how to create a new SageMaker HyperPod cluster with Slurm, which is a popular workload scheduler software. After you go through this tutorial, you will know how to log into the cluster nodes using the AWS Systems Manager commands (aws ssm). After you complete this tutorial, see also [the section called “SageMaker HyperPod operation”](#) to learn more about the SageMaker HyperPod basic operations, and [the section called “Jobs on HyperPod clusters”](#) to learn how to schedule jobs on the provisioned cluster.

### Tip

To find practical examples and solutions, see also the [SageMaker HyperPod workshop](#).

## Topics

- [Using the SageMaker HyperPod console UI](#)
- [Using the AWS CLI commands for the SageMaker HyperPod APIs](#)

## Using the SageMaker HyperPod console UI

Create your first SageMaker HyperPod cluster using the SageMaker HyperPod console UI.

### Create your first SageMaker HyperPod cluster with Slurm

The following tutorial demonstrates how to create a new SageMaker HyperPod cluster and set it up with Slurm through the SageMaker AI console UI. Following the tutorial, you'll create a HyperPod cluster with three Slurm nodes, my-controller-group, my-login-group, and worker-group-1.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Choose **HyperPod Clusters** in the left navigation pane and then **Cluster Management**.
3. On the **SageMaker HyperPod Clusters** page, choose **Create cluster**.
4. In **Step 1: Cluster settings**, specify a name for the new cluster. Skip the **Tags** section.
5. In **Step 2: Instance groups**, add instance groups. Each instance group can be configured differently, and you can create a heterogeneous cluster that consists of multiple instance groups with various instance types. For lifecycle configuration scripts to run on the instance group during cluster creation, you can start with using the sample lifecycle scripts provided in the [Awsome Distributed Training GitHub repository](#).

- a. For **Instance group name**, specify a name for the instance group. For this tutorial, create three instance groups named `my-controller-group`, `my-login-group`, and `worker-group-1`.
- b. For **Select instance type**, choose the instance for the instance group. For this tutorial, select `ml.c5.xlarge` for `my-controller-group`, `ml.m5.4xlarge` for `my-login-group`, and `ml.trn1.32xlarge` for `worker-group-1`.

Ensure that you choose the instance type with sufficient quotas in your account, or request additional quotas by following at [the section called "SageMaker HyperPod quotas"](#).

- c. For **Quantity**, specify an integer not exceeding the instance quota for cluster usage. For this tutorial, enter **1** for all three groups.
- d. For **S3 path to lifecycle script files**, enter the Amazon S3 path in which your lifecycle scripts are stored. If you don't have lifecycle scripts, go through the following substeps to use the base lifecycle scripts provided by the SageMaker HyperPod service team.
  - i. Clone the [Awsome Distributed Training GitHub repository](#).

```
git clone https://github.com/aws-samples/awsome-distributed-training/
```

- ii. Under [1.architectures/5.sagemaker\\_hyperpods/LifecycleScripts/base-config](#), you can find a set of base lifecycle scripts. To learn more about the lifecycle scripts, see also [the section called "Customize SageMaker HyperPod clusters using lifecycle scripts"](#).
- iii. Write a Slurm configuration file and save it as `provisioning_params.json`. In the file, specify basic Slurm configuration parameters to properly assign Slurm nodes to the SageMaker HyperPod cluster instance groups. For example, the `provisioning_params.json` should be similar to the following based on the HyperPod cluster instance group configured through the previous steps 5a, 5b, and 5c.

```
{  
    "version": "1.0.0",  
    "workload_manager": "slurm",  
    "controller_group": "my-controller-group",  
    "login_group": "my-login-group",  
    "worker_groups": [  
        {
```

```
        "instance_group_name": "worker-group-1",
        "partition_name": "partition-1"
    }
]
}
```

- iv. Upload the scripts to your Amazon S3 bucket. Create an S3 bucket with a path in the following format: s3://sagemaker-<unique-s3-bucket-name>/<lifecycle-script-directory>/src. You can create this bucket using the Amazon S3 console.

 **Note**

You must prefix sagemaker- to the S3 bucket path, because the [???](#) with `AmazonSageMakerClusterInstanceRolePolicy` only allows principals to access S3 buckets with this specific prefix.

- e. For **Directory path to your on-create lifecycle script**, enter the file name of the lifecycle script under **S3 path to lifecycle script files**.
- f. For **IAM role**, choose the IAM role you created using the `AmazonSageMakerClusterInstanceRolePolicy` from the section [the section called "IAM role for SageMaker HyperPod"](#).
- g. Under **Advanced configuration**, you can set up the following optional configurations.
  - i. (Optional) For **Threads per core**, specify 1 for disabling multi-threading and 2 for enabling multi-threading. To find which instance type supports multi-threading, see the reference table of [CPU cores and threads per CPU core per instance type](#) in the *Amazon Elastic Compute Cloud User Guide*.
  - ii. (Optional) For **Additional instance storage configs**, specify an integer between 1 and 16384 to set the size of an additional Elastic Block Store (EBS) volume in gigabytes (GB). The EBS volume is attached to each instance of the instance group. The default mount path for the additional EBS volume is /opt/sagemaker. After the cluster is successfully created, you can SSH into the cluster instances (nodes) and verify if the EBS volume is mounted correctly by running the `df -h` command. Attaching an additional EBS volume provides stable, off-instance, and independently persisting storage, as described in the [Amazon EBS volumes](#) section in the *Amazon Elastic Block Store User Guide*.
6. In **Step 3: Advanced configuration**, set up network settings within, and in and out of, the cluster. Select your own VPC if you already have one that gives SageMaker AI access to your

VPC. If you don't have one but want to create a new VPC, follow the instructions at [Create a VPC](#) in the *Amazon Virtual Private Cloud User Guide*. You can leave it as **no VPC** to use the default SageMaker AI VPC.

7. In **Step 4: Review and create**, review the configuration you've set from step 1 to 3 and finish submitting the cluster creation request.
8. The new cluster should appear under **Clusters** in the main pane of the SageMaker HyperPod console. You can check the status of it displayed under the **Status** column.
9. After the status of the cluster turns to **InService**, you can start logging into the cluster nodes. To access the cluster nodes and start running ML workloads, see [the section called "Jobs on HyperPod clusters"](#).

## Delete the cluster and clean resources

After you have successfully tested creating a SageMaker HyperPod cluster, it continues running in the **InService** state until you delete the cluster. We recommend that you delete any clusters created using on-demand SageMaker AI instances when not in use to avoid incurring continued service charges based on on-demand pricing. In this tutorial, you have created a cluster that consists of two instance groups. One of them uses a C5 instance, so make sure you delete the cluster by following the instructions at [the section called "Delete a SageMaker HyperPod cluster"](#).

However, if you have created a cluster with reserved compute capacity, the status of the clusters does not affect service billing.

To clean up the lifecycle scripts from the S3 bucket used for this tutorial, go to the S3 bucket you used during cluster creation and remove the files entirely.

If you have tested running any workloads on the cluster, make sure if you have uploaded any data or if your job saved any artifacts to different S3 buckets or file system services such as Amazon FSx for Lustre and Amazon Elastic File System. To prevent any incurring charges, delete all artifacts and data from the storage or file system.

## Using the AWS CLI commands for the SageMaker HyperPod APIs

Create your first SageMaker HyperPod cluster using the AWS CLI commands for HyperPod.

### Create your first SageMaker HyperPod cluster with Slurm

The following tutorial demonstrates how to create a new SageMaker HyperPod cluster and set it up with Slurm through the [AWS CLI commands for SageMaker HyperPod](#). Following the tutorial, you'll

create a HyperPod cluster with three Slurm nodes, `my-controller-group`, `my-login-group`, and `worker-group-1`.

1. First, prepare and upload lifecycle scripts to an Amazon S3 bucket. During cluster creation, HyperPod runs them in each instance group. Upload lifecycle scripts to Amazon S3 using the following command.

```
aws s3 sync \  
~/local-dir-to-lifecycle-scripts/* \  
s3://sagemaker-<unique-s3-bucket-name>/<lifecycle-script-directory>/src
```

 **Note**

The S3 bucket path should start with a prefix `sagemaker-`, because the [???](#) with `AmazonSageMakerClusterInstanceRolePolicy` only allows access to Amazon S3 buckets that starts with the specific prefix.

If you are starting from scratch, use sample lifecycle scripts provided in the [Awsome Distributed Training GitHub repository](#). The following sub-steps show how to download, what to modify, and how to upload the sample lifecycle scripts to an Amazon S3 bucket.

- a. Download a copy of the lifecycle script samples to a directory on your local computer.

```
git clone https://github.com/aws-samples/awsome-distributed-training/
```

- b. Go into the directory [1.architectures/5.sagemaker\\_hypopods/LifecycleScripts/base-config](#), where you can find a set of lifecycle scripts.

```
cd awsome-distributed-training/1.architectures/5.sagemaker_hypopods/  
LifecycleScripts/base-config
```

To learn more about the lifecycle script samples, see [the section called “Customize SageMaker HyperPod clusters using lifecycle scripts”](#).

- c. Write a Slurm configuration file and save it as `provisioning_params.json`. In the file, specify basic Slurm configuration parameters to properly assign Slurm nodes to the SageMaker HyperPod cluster instance groups. In this tutorial, set up three Slurm nodes

named `my-controller-group`, `my-login-group`, and `worker-group-1`, as shown in the following example configuration `provisioning_params.json`.

```
{  
    "version": "1.0.0",  
    "workload_manager": "slurm",  
    "controller_group": "my-controller-group",  
    "login_group": "my-login-group",  
    "worker_groups": [  
        {  
            "instance_group_name": "worker-group-1",  
            "partition_name": "partition-1"  
        }  
    ]  
}
```

- d. Upload the scripts to `s3://sagemaker-<unique-s3-bucket-name>/<lifecycle-script-directory>/src`. You can do so by using the Amazon S3 console, or by running the following AWS CLI Amazon S3 command.

```
aws s3 sync \  
~/local-dir-to-lifecycle-scripts/* \  
s3://sagemaker-<unique-s3-bucket-name>/<lifecycle-script-directory>/src
```

2. Prepare a [CreateCluster](#) request file in JSON format and save as `create_cluster.json`. The following request template aligns with the Slurm node configuration defined in the `provisioning_params.json` in Step 1.c. For `ExecutionRole`, provide the ARN of the IAM role you created with the managed `AmazonSageMakerClusterInstanceRolePolicy` in [the section called “Prerequisites”](#).

```
{  
    // Required: Specify the name of the cluster.  
    "ClusterName": "my-hyperpod-cluster",  
    // Required: Configure instance groups to be launched in the cluster  
    "InstanceGroups": [  
        {  
            // Required: Specify the basic configurations to set up a controller  
            // node.  
            "InstanceGroupName": "my-controller-group",  
            "InstanceType": "ml.c5.xlarge",  
            "InstanceCount": 1,  
            "LifeCycleConfig": {  
                "PreLaunchCommands": [  
                    {  
                        "Command": "echo 'Hello, world!' > /tmp/hello.txt",  
                        "FailureAction": "CONTINUE",  
                        "Order": 1  
                    }  
                ]  
            }  
        }  
    ]  
}
```

```
        "SourceS3Uri": "s3://sagemaker-<unique-s3-bucket-name>/<lifecycle-script-directory>/src",
        "OnCreate": "on_create.sh"
    },
    "ExecutionRole": "${ROLE}",
    // Optional: Configure an additional storage per instance group.
    "InstanceStorageConfigs": [
        {
            // Attach an additional EBS volume to each instance within the
            instance group.
            // The default mount path for the additional EBS volume is /opt/
            sagemaker.

            "EbsVolumeConfig":{
                // Specify an integer between 1 and 16384 in gigabytes (GB).
                "VolumeSizeInGB": integer,
            }
        }
    ]
},
{
    "InstanceGroupName": "my-login-group",
    "InstanceType": "ml.m5.4xlarge",
    "InstanceCount": 1,
    "LifeCycleConfig": {
        "SourceS3Uri": "s3://sagemaker-<unique-s3-bucket-name>/<lifecycle-script-directory>/src",
        "OnCreate": "on_create.sh"
    },
    "ExecutionRole": "${ROLE}"
},
{
    "InstanceGroupName": "worker-group-1",
    "InstanceType": "ml.trn1.32xlarge",
    "InstanceCount": 1,
    "LifeCycleConfig": {
        "SourceS3Uri": "s3://sagemaker-<unique-s3-bucket-name>/<lifecycle-script-directory>/src",
        "OnCreate": "on_create.sh"
    },
    "ExecutionRole": "${ROLE}"
}
]
```

3. Run the following command to create the cluster.

```
aws sagemaker create-cluster --cli-input-json file://complete/path/to/  
create_cluster.json
```

This should return the ARN of the created cluster.

If you receive an error due to resource limits, ensure that you change the instance type to one with sufficient quotas in your account, or request additional quotas by following at [the section called "SageMaker HyperPod quotas".](#)

4. Run `describe-cluster` to check the status of the cluster.

```
aws sagemaker describe-cluster --cluster-name my-hyperpod-cluster
```

After the status of the cluster turns to **InService**, proceed to the next step.

5. Run `list-cluster-nodes` to check the details of the cluster nodes.

```
aws sagemaker list-cluster-nodes --cluster-name my-hyperpod-cluster
```

This returns a response, and the `InstanceId` is what your cluster users need for logging (`aws ssm`) into them. For more information about logging into the cluster nodes and running ML workloads, see [the section called "Jobs on HyperPod clusters".](#)

## Delete the cluster and clean resources

After you have successfully tested creating a SageMaker HyperPod cluster, it continues running in the **InService** state until you delete the cluster. We recommend that you delete any clusters created using on-demand SageMaker AI capacity when not in use to avoid incurring continued service charges based on on-demand pricing. In this tutorial, you have created a cluster that consists of two instance groups. One of them uses a C5 instance, so make sure you delete the cluster by running the following command.

```
aws sagemaker delete-cluster --cluster-name my-hyperpod-cluster
```

To clean up the lifecycle scripts from the Amazon S3 bucket used for this tutorial, go to the Amazon S3 bucket you used during cluster creation and remove the files entirely.

If you have tested running any model training workloads on the cluster, also check if you have uploaded any data or if your job has saved any artifacts to different Amazon S3 buckets or file system services such as Amazon FSx for Lustre and Amazon Elastic File System. To prevent incurring charges, delete all artifacts and data from the storage or file system.

## SageMaker HyperPod operation

This section provides guidance on managing SageMaker HyperPod through the SageMaker AI console UI or the AWS Command Line Interface (CLI). You'll learn how to perform various tasks related to SageMaker HyperPod, whether you prefer a visual interface or working with commands.

### Topics

- [Using the SageMaker HyperPod console UI](#)
- [Using the AWS CLI](#)

### Using the SageMaker HyperPod console UI

The following topics provide guidance on how to manage SageMaker HyperPod through the console UI.

### Topics

- [Create a SageMaker HyperPod cluster](#)
- [Browse your SageMaker HyperPod clusters](#)
- [View details of each SageMaker HyperPod cluster](#)
- [Edit a SageMaker HyperPod cluster](#)
- [Delete a SageMaker HyperPod cluster](#)

### Create a SageMaker HyperPod cluster

See the following instructions on creating a new SageMaker HyperPod cluster through the SageMaker HyperPod console UI.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Choose **HyperPod Clusters** in the left navigation pane and then **Cluster Management**.
3. In the SageMaker HyperPod landing page, choose **Create HyperPod cluster**.
4. From the drop-down menu of **Create HyperPod cluster**, choose **Orchestrated by Slurm**.

5. In **Step 1: Cluster settings**, set up basic information for the cluster.
  - a. For **Cluster name**, specify a name for the new cluster.
  - b. For **Tags**, add key and value pairs to the new cluster and manage the cluster as an AWS resource. To learn more, see [Tagging your AWS resources](#).
6. In **Step 2: Advanced configuration**, configure optional network settings within the cluster and in-and-out of the cluster. Select your own VPC if you already have one that gives SageMaker AI access to your resources under the VPC. If you want to create a new VPC, see [Create a default VPC](#) or [Create a VPC](#) in the *Amazon Virtual Private Cloud User Guide*. If you don't make any selections, SageMaker automatically uses the default VPC associated with your account.

 **Note**

If you want to use your own VPC, you should add additional permissions to the IAM role for SageMaker HyperPod clusters. To learn more, see [the section called “Setting up SageMaker HyperPod with a custom Amazon VPC”](#).

7. In **Step 3: Instance groups**, choose **Create instance group**. Each instance group can be configured differently, and you can create a heterogeneous cluster that consists of multiple instance groups with various instance types. In the **Create an instance group** configuration pop-up window, fill the instance group configuration information.
  - a. For **Instance group name**, specify a name for the instance group.
  - b. For **Select instance type**, choose the instance for the instance group.
  - c. For **Quantity**, specify an integer not exceeding the instance quota for cluster usage. To view your current quotas or request a quota increase, see [the section called “SageMaker HyperPod quotas”](#).
  - d. For **S3 path to lifecycle script files**, enter the Amazon S3 path in which your lifecycle scripts are stored or use the **Browse S3** option.
  - e. For **Directory path to your on-create lifecycle script**, enter the file name of the lifecycle script under **S3 path to lifecycle script files**.
  - f. For **IAM role**, choose the IAM role you have created for SageMaker HyperPod resources, following the section [the section called “IAM for HyperPod”](#).
  - g. Under **Advanced configuration**, you can set up the following optional configurations.

- i. (Optional) For **Threads per core**, specify 1 for disabling multi-threading and 2 for enabling multi-threading. To find which instance type supports multi-threading, see the reference table of [CPU cores and threads per CPU core per instance type](#) in the *Amazon EC2 User Guide*.
  - ii. (Optional) For **Additional instance storage configs**, specify an integer between 1 and 16384 to set the size of an additional Elastic Block Store (EBS) volume in gigabytes (GB). The EBS volume is attached to each instance of the instance group. The default mount path for the additional EBS volume is /opt/sagemaker. After the cluster is successfully created, you can SSH into the cluster instances (nodes) and verify if the EBS volume is mounted correctly by running the df -h command. Attaching an additional EBS volume provides stable, off-instance, and independently persisting storage, as described in the [Amazon EBS volumes](#) section in the *Amazon Elastic Block Store User Guide*.
8. In **Step 4: Review and create**, review the configuration you have set from **Step 1 to Step 3** and finish submitting the cluster creation request.
  9. After the status of the cluster turns to **InService**, you can start logging into the cluster nodes. To access the cluster nodes and start running ML workloads, see [the section called "Jobs on HyperPod clusters"](#).

## Browse your SageMaker HyperPod clusters

Under **Clusters** in the main pane of the SageMaker HyperPod console on the SageMaker HyperPod console main page, all created clusters should appear listed under the **Clusters** section, which provides a summary view of clusters, their ARNs, status, and creation time.

## View details of each SageMaker HyperPod cluster

Under **Clusters** on the console main page, the cluster **Names** are activated as links. Choose the cluster name link to see details of each cluster.

## Edit a SageMaker HyperPod cluster

1. Under **Clusters** in the main pane of the SageMaker HyperPod console, choose the cluster you want to update.
2. Select your cluster, and choose **Edit**.

3. In the **Edit <your-cluster>** page, you can edit the configurations of existing instance groups, add more instance groups, delete instance groups, and change tags for the cluster. After making changes, choose **Submit**.
  - a. In the **Configure instance groups** section, you can add more instance groups by choosing **Create instance group**.
  - b. In the **Configure instance groups** section, you can choose **Edit** to change its configuration or **Delete** to remove the instance group permanently.

 **Important**

When deleting an instance group, consider the following points:

- Your SageMaker HyperPod cluster must always maintain at least one instance group.
- Ensure all critical data is backed up before removal
- The removal process cannot be undone.

 **Note**

Deleting an instance group will terminate all compute resources associated with that group.

- c. In the **Tags** section, you can update tags for the cluster.

## Delete a SageMaker HyperPod cluster

1. Under **Clusters** in the main pane of the SageMaker HyperPod console, choose the cluster you want to delete.
2. Select your cluster, and choose **Delete**.
3. In the pop-up window for cluster deletion, review the cluster information carefully to confirm that you chose the right cluster to delete.
4. After you reviewed the cluster information, choose **Yes, delete cluster**.
5. In the text field to confirm this deletion, type **delete**.
6. Choose **Delete** on the lower right corner of the pop-up window to finish sending the cluster deletion request.

## Using the AWS CLI

The following topics provide guidance on writing SageMaker HyperPod API request files in JSON format and run them using the AWS CLI commands.

### Topics

- [Create a new cluster](#)
- [Describe a cluster](#)
- [List details of cluster nodes](#)
- [Describe details of a cluster node](#)
- [List clusters](#)
- [Update cluster configuration](#)
- [Update the SageMaker HyperPod platform software of a cluster](#)
- [Scale down a cluster](#)
- [Delete a cluster](#)

### Create a new cluster

1. Prepare lifecycle configuration scripts and upload them to an S3 bucket, such as `s3://sagemaker-amzn-s3-demo-bucket/lifecycle-script-directory/src/`. The following step 2 assumes that there's an entry point script named `on_create.sh` in the specified S3 bucket.

 **Important**

Make sure that you set the S3 path to start with `s3://sagemaker-`. The [the section called "IAM role for SageMaker HyperPod"](#) has the managed [AmazonSageMakerClusterInstanceRolePolicy](#) attached, which allows access to S3 buckets with the specific prefix `sagemaker-`.

2. Prepare a [CreateCluster](#) API request file in JSON format. You should configure instance groups to match with the Slurm cluster you design in the `provisioning_params.json` file that'll be used during cluster creating as part of running a set of lifecycle scripts. To learn more, see [the section called "Customize SageMaker HyperPod clusters using lifecycle scripts"](#). The following template has two instance groups to meet the minimum requirement for a Slurm cluster: one controller (head) node and one compute (worker)

node. For ExecutionRole, provide the ARN of the IAM role you created with the managed AmazonSageMakerClusterInstanceRolePolicy from the section [the section called “IAM role for SageMaker HyperPod”](#).

```
// create_cluster.json
{
    "ClusterName": "your-hyperpod-cluster",
    "InstanceGroups": [
        {
            "InstanceGroupName": "controller-group",
            "InstanceType": "ml.m5.xlarge",
            "InstanceCount": 1,
            "LifeCycleConfig": {
                "SourceS3Uri": "s3://amzn-s3-demo-bucket-sagemaker/lifecycle-
script-directory/src/",
                "OnCreate": "on_create.sh"
            },
            "ExecutionRole": "arn:aws:iam::111122223333:role/iam-role-for-cluster",
            // Optional: Configure an additional storage per instance group.
            "InstanceStorageConfigs": [
                {
                    // Attach an additional EBS volume to each instance within the
                    instance group.
                    // The default mount path for the additional EBS volume is /opt/
                    sagemaker.
                    "EbsVolumeConfig": {
                        // Specify an integer between 1 and 16384 in gigabytes (GB).
                        "VolumeSizeInGB": integer,
                    }
                }
            ]
        },
        {
            "InstanceGroupName": "worker-group-1",
            "InstanceType": "ml.p4d.xlarge",
            "InstanceCount": 1,
            "LifeCycleConfig": {
                "SourceS3Uri": "s3://amzn-s3-demo-bucket-sagemaker/lifecycle-
script-directory/src/",
                "OnCreate": "on_create.sh"
            },
            "ExecutionRole": "arn:aws:iam::111122223333:role/iam-role-for-cluster"
        }
    ]
}
```

```
],
  // Optional
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ],
  // Optional
  "VpcConfig": {
    "SecurityGroupIds": [ "string" ],
    "Subnets": [ "string" ]
  }
}
```

Depending on how you design the cluster structure through your lifecycle scripts, you can configure up to 20 instance groups under the `InstanceGroups` parameter.

For the `Tags` request parameter, you can add custom tags for managing the SageMaker HyperPod cluster as an AWS resource. You can add tags to your cluster in the same way you add them in other AWS services that support tagging. To learn more about tagging AWS resources in general, see [Tagging AWS Resources User Guide](#).

For the `VpcConfig` request parameter, specify the information of a VPC you want to use. For more information, see [the section called "Setting up SageMaker HyperPod with a custom Amazon VPC"](#).

### 3. Run the [create-cluster](#) command as follows.

```
aws sagemaker create-cluster \
--cli-input-json file://complete/path/to/create_cluster.json
```

This should return the ARN of the new cluster.

## Describe a cluster

Run [describe-cluster](#) to check the status of the cluster. You can specify either the name or the ARN of the cluster.

```
aws sagemaker describe-cluster --cluster-name your-hyperpod-cluster
```

After the status of the cluster turns to **InService**, proceed to the next step. Using this API, you can also retrieve failure messages from running other HyperPod API operations.

## List details of cluster nodes

Run [list-cluster-nodes](#) to check the key information of the cluster nodes.

```
aws sagemaker list-cluster-nodes --cluster-name your-hyperpod-cluster
```

This returns a response, and the InstanceId is what you need to use for logging (using aws ssm) into them.

## Describe details of a cluster node

Run [describe-cluster-node](#) to retrieve details of a cluster node. You can get the cluster node ID from list-cluster-nodes output. You can specify either the name or the ARN of the cluster.

```
aws sagemaker describe-cluster-node \  
  --cluster-name your-hyperpod-cluster \  
  --node-id i-11122233344455aa
```

## List clusters

Run [list-clusters](#) to list all clusters in your account.

```
aws sagemaker list-clusters
```

You can also add additional flags to filter the list of clusters down. To learn more about what this command runs at low level and additional flags for filtering, see the [ListClusters](#) API reference.

## Update cluster configuration

Run [update-cluster](#) to update the configuration of a cluster.

### Note

You can use the UpdateCluster API to scale down or remove entire instance groups from your SageMaker HyperPod cluster. For additional instructions on how to scale down or delete instance groups, see [the section called “Scale down a cluster”](#).

1. Create an `UpdateCluster` request file in JSON format. Make sure that you specify the right cluster name and instance group name to update. You can change the instance type, the number of instances, the lifecycle configuration entrypoint script, and the path to the script.
  - a. For `ClusterName`, specify the name of the cluster you want to update.
  - b. For `InstanceGroupName`
    - i. To update an existing instance group, specify the name of the instance group you want to update.
    - ii. To add a new instance group, specify a new name not existing in your cluster.
  - c. For `InstanceType`
    - i. To update an existing instance group, you must match the instance type you initially specified to the group.
    - ii. To add a new instance group, specify an instance type you want to configure the group with.
  - d. For `InstanceCount`
    - i. To update an existing instance group, specify an integer that corresponds to your desired number of instances. You can provide a higher or lower value (down to 0) to scale the instance group up or down.
    - ii. To add a new instance group, specify an integer greater or equal to 1.
  - e. For `LifeCycleConfig`, you can change both `SourceS3Uri` and `OnCreate` values as you want to update the instance group.
  - f. For `ExecutionRole`
    - i. For updating an existing instance group, keep using the same IAM role you attached during cluster creation.
    - ii. For adding a new instance group, specify an IAM role you want to attach.
  - g. For `TreadsPerCore`
    - i. For updating an existing instance group, keep using the same value you specified during cluster creation.
    - ii. For adding a new instance group, you can choose any value from the allowed options per instance type. For more information, search the instance type and see the **Valid treads per core** column in the reference table at [CPU cores and threads per CPU core per instance type](#) in the *Amazon EC2 User Guide*.

The following code snippet is a JSON request file template you can use. For more information about the request syntax and parameters of this API, see the [UpdateCluster](#) API reference.

```
// update_cluster.json
{
    // Required
    "ClusterName": "name-of-cluster-to-update",
    // Required
    "InstanceGroups": [
        {
            "InstanceGroupName": "name-of-instance-group-to-update",
            "InstanceType": "ml.m5.xlarge",
            "InstanceCount": 1,
            "LifeCycleConfig": {
                "SourceS3Uri": "s3://amzn-s3-demo-bucket-sagemaker/lifecycle-script-directory/src/",
                "OnCreate": "on_create.sh"
            },
            "ExecutionRole": "arn:aws:iam::111122223333:role/iam-role-for-cluster",
            // Optional: Configure an additional storage per instance group.
            "InstanceStorageConfigs": [
                {
                    // Attach an additional EBS volume to each instance within the
                    // instance group.
                    // The default mount path for the additional EBS volume is /opt/
                    // sagemaker.
                    "EbsVolumeConfig": {
                        // Specify an integer between 1 and 16384 in gigabytes (GB).
                        "VolumeSizeInGB": integer,
                    }
                }
            ]
        },
        // add more blocks of instance groups as needed
        { ... }
    ]
}
```

2. Run the following update-cluster command to submit the request.

```
aws sagemaker update-cluster \
--cli-input-json file://complete/path/to/update_cluster.json
```

## Update the SageMaker HyperPod platform software of a cluster

Run [update-cluster-software](#) to update existing clusters with software and security patches provided by the SageMaker HyperPod service. For `--cluster-name`, specify either the name or the ARN of the cluster to update.

### Important

Note that you must back up your work before running this API. The patching process replaces the root volume with the updated AMI, which means that your previous data stored in the instance root volume will be lost. Make sure that you back up your data from the instance root volume to Amazon S3 or Amazon FSx for Lustre. For more information, see [the section called “Use the backup script provided by SageMaker HyperPod”](#).

```
aws sagemaker update-cluster-software --cluster-name your-hyperpod-cluster
```

This command calls the [UpdateClusterSoftware](#) API. After the API call, SageMaker HyperPod updates the cluster instances to use the latest [the section called “SageMaker HyperPod DLAMI”](#) and runs your lifecycle scripts in the S3 bucket that you specified during cluster creation or update. The SageMaker HyperPod service team regularly rolls out new [the section called “SageMaker HyperPod DLAMI”](#)s for enhancing security and improving user experiences. We recommend that you always keep updating to the latest SageMaker HyperPod DLAMI. For future SageMaker HyperPod DLAMI updates for security patching, follow up with [the section called “HyperPod release notes”](#).

### Tip

If the security patch fails, you can retrieve failure messages by running the [DescribeCluster](#) API as instructed at [the section called “Describe a cluster”](#).

### Note

You can only run this API programmatically. The patching functionality is not implemented in the SageMaker HyperPod console UI.

## Use the backup script provided by SageMaker HyperPod

SageMaker HyperPod provides a script to back up and restore your data at

[1.architectures/5.sagemaker-hyperpod/patching-backup.sh](https://github.com/awsm-AI/Awesome-Distributed-Training/tree/main/architectures/5.sagemaker-hyperpod/patching-backup.sh) in the *Awsome Distributed Training GitHub repository*. The script provides the following two functions.

### To back up data to an S3 bucket before patching

```
sudo bash patching-backup.sh --create <s3-buckup-bucket-path>
```

After you run the command, the script checks squeue if there are queued jobs, stops Slurm if there's no job in the queue, backs up mariadb, and copies local items on disc defined under LOCAL\_ITEMS. You can add more files and directories to LOCAL\_ITEMS.

```
# Define files and directories to back up.  
LOCAL_ITEMS=(  
    "/var/spool/slurmd"  
    "/var/spool/slurmctld"  
    "/etc/systemd/system/slurmctld.service"  
    "/home/ubuntu/backup_slurm_acct_db.sql"  
    # ... Add more items as needed  
)
```

Also, you can add custom code to the provided script to back up any applications for your use case.

### To restore data from an S3 bucket after patching

```
sudo bash patching-backup.sh --restore <s3-buckup-bucket-path>
```

## Scale down a cluster

You can scale down the number of instances or delete instance groups in your SageMaker HyperPod cluster to optimize resource allocation or reduce costs.

You scale down by either using the `UpdateCluster` API operation to randomly terminate instances from your instance group down to a specified number, or by terminating specific instances using the `BatchDeleteClusterNodes` API operation. You can also completely remove entire instance groups using the `UpdateCluster` API. For more information about how to scale down using these methods, see [Scale down a SageMaker HyperPod cluster](#).

### Note

You cannot remove instances that are configured as Slurm controller nodes. Attempting to delete a Slurm controller node results in a validation error with the error code NODE\_ID\_IN\_USE.

## Delete a cluster

Run [delete-cluster](#) to delete a cluster. You can specify either the name or the ARN of the cluster.

```
aws sagemaker delete-cluster --cluster-name your-hyperpod-cluster
```

## Customize SageMaker HyperPod clusters using lifecycle scripts

SageMaker HyperPod offers always up-and-running compute clusters, which are highly customizable as you can write lifecycle scripts to tell SageMaker HyperPod how to set up the cluster resources. The following topics are best practices for preparing lifecycle scripts to set up SageMaker HyperPod clusters with open source workload manager tools.

The following topics discuss in-depth best practices for preparing lifecycle scripts to set up Slurm configurations on SageMaker HyperPod.

### High-level overview

The following procedure is the main flow of provisioning a HyperPod cluster and setting it up with Slurm. The steps are put in order of a **bottom-up** approach.

1. Plan how you want to create Slurm nodes on a HyperPod cluster. For example, if you want to configure two Slurm nodes, you'll need to set up two instance groups in a HyperPod cluster.
2. Prepare a `provisioning_parameters.json` file, which is a [the section called “Configuration form for provisioning Slurm nodes on HyperPod”](#). `provisioning_parameters.json` should contain Slurm node configuration information to be provisioned on the HyperPod cluster. This should reflect the design of Slurm nodes from Step 1.
3. Prepare a set of lifecycle scripts to set up Slurm on HyperPod to install software packages and set up an environment in the cluster for your use case. You should structure the lifecycle scripts to collectively run in order in a central Python script (`lifecycle_script.py`), and write an entrypoint shell script (`on_create.sh`) to run the Python script. The entrypoint shell script is what you need to provide to a HyperPod cluster creation request later in Step 5.

Also, note that you should write the scripts to expect `resource_config.json` that will be generated by HyperPod during cluster creation. `resource_config.json` contains HyperPod cluster resource information such as IP addresses, instance types, and ARNs, and is what you need to use for configuring Slurm.

#### 4. Collect all the files from the previous steps into a folder.

```
### lifecycle_files // your local folder

    ### provisioning_parameters.json
    ### on_create.sh
    ### lifecycle_script.py
    ### ... // more setup scripts to be fed into lifecycle_script.py
```

#### 5. Upload all the files to an S3 bucket. Copy and keep the S3 bucket path. Note that you should create an S3 bucket path starting with `sagemaker-` because you need to choose an [the section called “IAM role for SageMaker HyperPod”](#) attached with [AmazonSageMakerClusterInstanceRolePolicy](#), which only allows S3 bucket paths starting with the prefix `sagemaker-`. The following command is an example command to upload all the files to an S3 bucket.

```
aws s3 cp --recursive ./lifecycle_files s3://sagemaker-hyperpod-lifecycle/src
```

#### 6. Prepare a HyperPod cluster creation request.

- Option 1: If you use the AWS CLI, write a cluster creation request in JSON format (`create_cluster.json`) following the instructions at [the section called “Create a new cluster”](#).
- Option 2: If you use the SageMaker AI console UI, fill the **Create a cluster** request form in the HyperPod console UI following the instructions at [the section called “Create a SageMaker HyperPod cluster”](#).

At this stage, make sure that you create instance groups in the same structure that you planned in Step 1 and 2. Also, make sure that you specify the S3 bucket from Step 5 in the request forms.

#### 7. Submit the cluster creation request. HyperPod provisions a cluster based on the request, and then creates a `resource_config.json` file in the HyperPod cluster instances, and sets up Slurm on the cluster running the lifecycle scripts.

The following topics walk you through and dive deep into details on how to organize configuration files and lifecycle scripts to work properly during HyperPod cluster creation.

## Topics

- [Start with base lifecycle scripts provided by HyperPod](#)
- [What particular configurations HyperPod manages in Slurm configuration files](#)
- [Mount Amazon FSx for Lustre to a HyperPod cluster](#)
- [Validate the JSON configuration files before creating a Slurm cluster on HyperPod](#)
- [Validate runtime before running production workloads on a Slurm cluster on HyperPod](#)
- [Develop lifecycle scripts interactively on a HyperPod cluster node](#)
- [Update a cluster with new or updated lifecycle scripts](#)

### Start with base lifecycle scripts provided by HyperPod

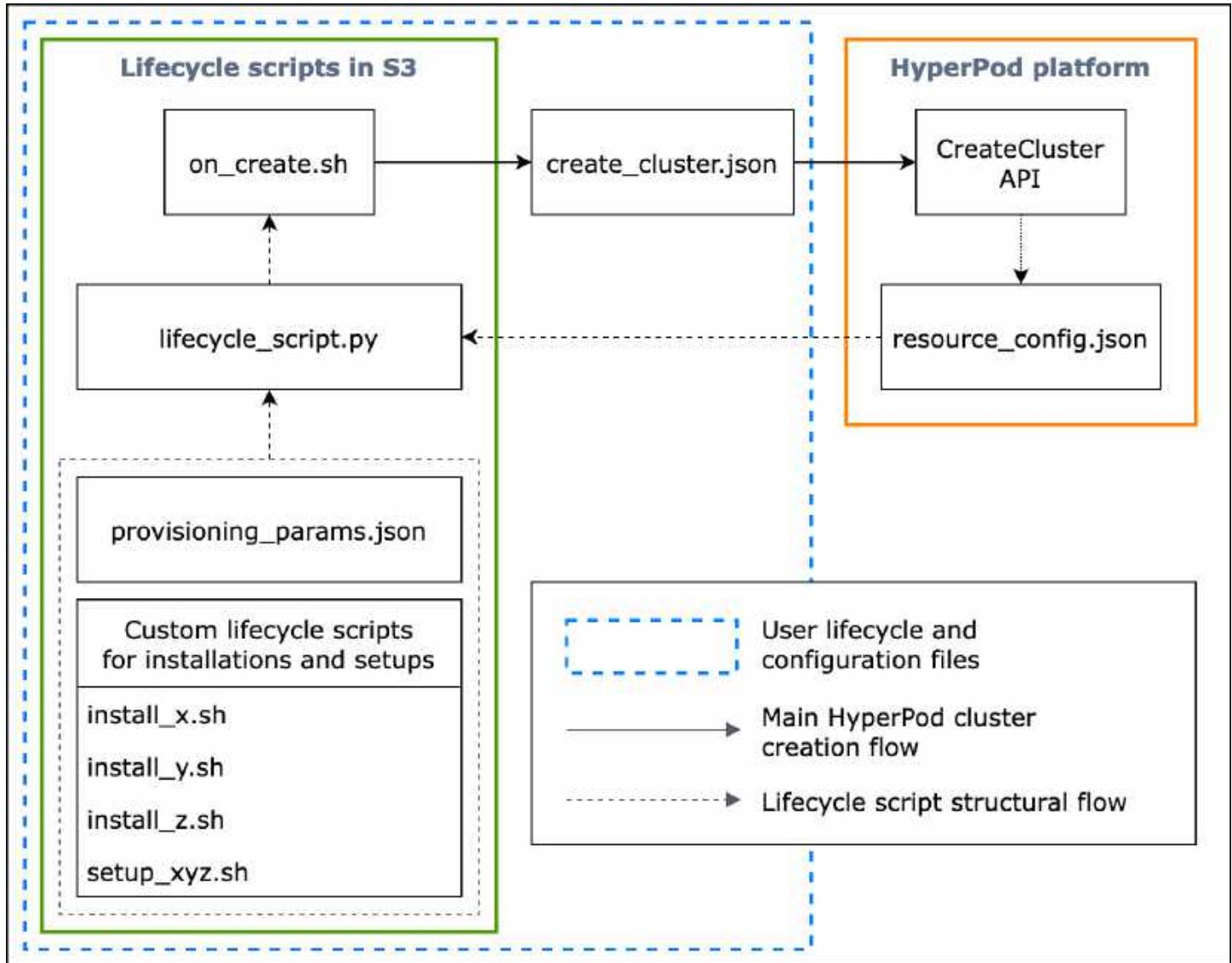
This section walks you through every component of the basic flow of setting up Slurm on HyperPod in a **top-down** approach. It starts from preparing a HyperPod cluster creation request to run the CreateCluster API, and dives deep into the hierarchical structure down to lifecycle scripts. Use the sample lifecycle scripts provided in the [Awsome Distributed Training GitHub repository](#). Clone the repository by running the following command.

```
git clone https://github.com/aws-samples/awsome-distributed-training/
```

The base lifecycle scripts for setting up a Slurm cluster on SageMaker HyperPod are available at [1.architectures/5.sagemaker\\_hyperpods/LifecycleScripts/base-config](#).

```
cd awsome-distributed-training/1.architectures/5.sagemaker_hyperpods/LifecycleScripts/  
base-config
```

The following flowchart shows a detailed overview of how you should design the base lifecycle scripts. The descriptions below the diagram and the procedural guide explain how they work during the HyperPod CreateCluster API call.



**Figure:** A detailed flow chart of HyperPod cluster creation and the structure of lifecycle scripts. (1) The dashed arrows are directed to where the boxes are "called into" and shows the flow of configuration files and lifecycle scripts preparation. It starts from preparing `provisioning_parameters.json` and lifecycle scripts. These are then coded in `lifecycle_script.py` for a collective execution in order. And the execution of the `lifecycle_script.py` script is done by the `on_create.sh` shell script, which to be run in the HyperPod instance terminal. (2) The solid arrows show the main HyperPod cluster creation flow and how the boxes are "called into" or "submitted to". `on_create.sh` is required for cluster creation request, either in `create_cluster.json` or the **Create a cluster** request form in the console UI. After you submit the request, HyperPod runs the `CreateCluster API` based on the given configuration information from the request and the lifecycle scripts. (3) The dotted arrow indicates that the HyperPod platform creates `resource_config.json` in the cluster instances during cluster

*resource provisioning. `resource_config.json` contains HyperPod cluster resource information such as the cluster ARN, instance types, and IP addresses. It is important to note that you should prepare the lifecycle scripts to expect the `resource_config.json` file during cluster creation. For more information, see the procedural guide below.*

The following procedural guide explains what happens during HyperPod cluster creation and how the base lifecycle scripts are designed.

1. `create_cluster.json` – To submit a HyperPod cluster creation request, you prepare a `CreateCluster` request file in JSON format. In this best practices example, we assume that the request file is named `create_cluster.json`. Write `create_cluster.json` to provision a HyperPod cluster with instance groups. The best practice is to add the same number of instance groups as the number of Slurm nodes you plan to configure on the HyperPod cluster. Make sure that you give distinctive names to the instance groups that you'll assign to Slurm nodes you plan to set up.

Also, you are required to specify an S3 bucket path to store your entire set of configuration files and lifecycle scripts to the field name `InstanceGroups.LifeCycleConfig.SourceS3Uri` in the `CreateCluster` request form, and specify the file name of an endpoint shell script (assume that it's named `on_create.sh`) to `InstanceGroups.LifeCycleConfig.OnCreate`.

 **Note**

If you are using the **Create a cluster** submission form in the HyperPod console UI, the console manages filling and submitting the `CreateCluster` request on your behalf, and runs the `CreateCluster` API in the backend. In this case, you don't need to create `create_cluster.json`; instead, make sure that you specify the correct cluster configuration information to the **Create a cluster** submission form.

2. `on_create.sh` – For each instance group, you need to provide an endpoint shell script, `on_create.sh`, to run commands, run scripts to install software packages, and set up the HyperPod cluster environment with Slurm. The two things you need to prepare are a `provisioning_parameters.json` required by HyperPod for setting up Slurm and a set of lifecycle scripts for installing software packages. This script should be written to find and run the following files as shown in the sample script at [on\\_create.sh](#).

**Note**

Make sure that you upload the entire set of lifecycle scripts to the S3 location you specify in `create_cluster.json`. You should also place your `provisioning_parameters.json` in the same location.

- a. `provisioning_parameters.json` – This is a [the section called “Configuration form for provisioning Slurm nodes on HyperPod”](#). The `on_create.sh` script finds this JSON file and defines environment variable for identifying the path to it. Through this JSON file, you can configure Slurm nodes and storage options such as Amazon FSx for Lustre for Slurm to communicate with. In `provisioning_parameters.json`, make sure that you assign the HyperPod cluster instance groups using the names you specified in `create_cluster.json` to the Slurm nodes appropriately based on how you plan to set them up.

The following diagram shows an example of how the two JSON configuration files `create_cluster.json` and `provisioning_parameters.json` should be written to assign HyperPod instance groups to Slurm nodes. In this example, we assume a case of setting up three Slurm nodes: controller (management) node, log-in node (which is optional), and compute (worker) node.

**Tip**

To help you validate these two JSON files, the HyperPod service team provides a validation script, [`validate-config.py`](#). To learn more, see [the section called “Validate the JSON configuration files before creating a Slurm cluster on HyperPod”](#).

<code>create_cluster.json</code> for HyperPod cluster resource config	<code>provisioning_params.json</code> for Slurm config
<pre>{   "ClusterName": "your-hyperpod-cluster",   "InstanceGroups": [     {       "InstanceGroupName": "controller-machine",       "InstanceType": "ml.c5.xlarge",       "InstanceCount": 1,       "LifeCycleConfig": {         "SourceS3Uri": "s3://sagemaker-&lt;unique-s3-bucket-path&gt;/src",         "OnCreate": "on_create.sh"       },       "ExecutionRole": "\${ROLE}",       "ThreadsPerCore": 1     },     {       "InstanceGroupName": "login-group",       "InstanceType": "ml.m5.4xlarge",       "InstanceCount": 1,       "LifeCycleConfig": {         "SourceS3Uri": "s3://sagemaker-&lt;unique-s3-bucket-path&gt;/src",         "OnCreate": "on_create.sh"       },       "ExecutionRole": "\${ROLE}",       "ThreadsPerCore": 1     },     {       "InstanceGroupName": "compute-nodes",       "InstanceType": "ml.trn1.32xlarge",       "InstanceCount": 4,       "LifeCycleConfig": {         "SourceS3Uri": "s3://sagemaker-&lt;unique-s3-bucket-path&gt;/src",         "OnCreate": "on_create.sh"       },       "ExecutionRole": "\${ROLE}",       "ThreadsPerCore": 1     }   ],   "VpcConfig": {     "SecurityGroupIds": [ "string" ],     "Subnets": [ "string" ]   } }</pre>	<pre>{   "version": "1.0.0",   "workload_manager": "slurm",   "controller_group": "controller-machine",   "login_group": "login-group",   "worker_groups": [     {       "instance_group_name": "compute-nodes",       "partition_name": "dev"     }   ],   "fsx_dns_name": "fs-12345678a90b01cde. fsx.us-west-2.amazonaws.com",   "fsx_mountname": "1abcdefg" }</pre>

**Figure:** Direct comparison between `create_cluster.json` for HyperPod cluster creation and `provisioning_params.json` for Slurm configuration. The number of instance groups in `create_cluster.json` should match with the number of nodes you want to configure as Slurm nodes. In case of the example in the figure, three Slurm nodes will be configured on a HyperPod cluster of three instance groups. You should assign the HyperPod cluster instance groups to Slurm nodes by specifying the instance group names accordingly.

- b. `resource_config.json` – During cluster creation, the `lifecycle_script.py` script is written to expect a `resource_config.json` file from HyperPod. This file contains information about the cluster, such as instance types and IP addresses.

When you run the `CreateCluster` API, HyperPod creates a resource configuration file at `/opt/ml/config/resource_config.json` based on the `create_cluster.json` file. The file path is saved to the environment variable named `SAGEMAKER_RESOURCE_CONFIG_PATH`.

## ⚠️ Important

The `resource_config.json` file is auto-generated by the HyperPod platform, and you DO NOT need to create it. The following code is to show an example of `resource_config.json` that would be created from the cluster creation based on `create_cluster.json` in the previous step, and to help you understand what happens in the backend and how an auto-generated `resource_config.json` would look.

```
{  
  
    "ClusterConfig": {  
        "ClusterArn": "arn:aws:sagemaker:us-west-2:111122223333:cluster/  
        abcde01234yz",  
        "ClusterName": "your-hyperpod-cluster"  
    },  
    "InstanceGroups": [  
        {  
            "Name": "controller-machine",  
            "InstanceType": "ml.c5.xlarge",  
            "Instances": [  
                {  
                    "InstanceName": "controller-machine-1",  
                    "AgentIpAddress": "111.222.333.444",  
                    "CustomerIpAddress": "111.222.333.444",  
                    "InstanceId": "i-12345abcedfg67890"  
                }  
            ]  
        },  
        {  
            "Name": "login-group",  
            "InstanceType": "ml.m5.xlarge",  
            "Instances": [  
                {  
                    "InstanceName": "login-group-1",  
                    "AgentIpAddress": "111.222.333.444",  
                    "CustomerIpAddress": "111.222.333.444",  
                    "InstanceId": "i-12345abcedfg67890"  
                }  
            ]  
        }  
    ]  
}
```

```
        ],
      },
      {
        "Name": "compute-nodes",
        "InstanceType": "ml.trn1.32xlarge",
        "Instances": [
          {
            "InstanceName": "compute-nodes-1",
            "AgentIpAddress": "111.222.333.444",
            "CustomerIpAddress": "111.222.333.444",
            "InstanceId": "i-12345abcdedfg67890"
          },
          {
            "InstanceName": "compute-nodes-2",
            "AgentIpAddress": "111.222.333.444",
            "CustomerIpAddress": "111.222.333.444",
            "InstanceId": "i-12345abcdedfg67890"
          },
          {
            "InstanceName": "compute-nodes-3",
            "AgentIpAddress": "111.222.333.444",
            "CustomerIpAddress": "111.222.333.444",
            "InstanceId": "i-12345abcdedfg67890"
          },
          {
            "InstanceName": "compute-nodes-4",
            "AgentIpAddress": "111.222.333.444",
            "CustomerIpAddress": "111.222.333.444",
            "InstanceId": "i-12345abcdedfg67890"
          }
        ]
      }
    ]
```

- c. `lifecycle_script.py` – This is the main Python script that collectively runs lifecycle scripts setting up Slurm on the HyperPod cluster while being provisioned. This script reads in `provisioning_parameters.json` and `resource_config.json` from the paths that are specified or identified in `on_create.sh`, passes the relevant information to each lifecycle script, and then runs the lifecycle scripts in order.

Lifecycle scripts are a set of scripts that you have a complete flexibility to customize to install software packages and set up necessary or custom configurations during

cluster creation, such as setting up Slurm, creating users, installing Conda or Docker. The sample [lifecycle\\_script.py](#) script is prepared to run other base lifecycle scripts in the repository, such as launching Slurm deamons ([start\\_slurm.sh](#)), mounting Amazon FSx for Lustre ([mount\\_fsx.sh](#)), and setting up MariaDB accounting ([setup\\_mariadb\\_accounting.sh](#)) and RDS accounting ([setup\\_rds\\_accounting.sh](#)). You can also add more scripts, package them under the same directory, and add code lines to `lifecycle_script.py` to let HyperPod run the scripts. For more information about the base lifecycle scripts, see also [3.1 Lifecycle scripts](#) in the *Awsome Distributed Training GitHub repository*.

 **Note**

HyperPod runs [the section called “SageMaker HyperPod DLAMI”](#) on each instance of a cluster, and the AMI has pre-installed software packages complying compatibilities between them and HyperPod functionalities. Note that if you reinstall any of the pre-installed packages, you are responsible for installing compatible packages and note that some HyperPod functionalities might not work as expected.

In addition to the default setups, more scripts for installing the following software are available under the [utils](#) folder. The `lifecycle_script.py` file is already prepared to include code lines for running the installation scripts, so see the following items to search those lines and uncomment to activate them.

- i. The following code lines are for installing [Docker](#), [Enroot](#), and [Pyxis](#). These packages are required to run Docker containers on a Slurm cluster.

To enable this installation step, set the `enable_docker_enroot_pyxis` parameter to `True` in the [config.py](#) file.

```
# Install Docker/Enroot/Pyxis
if Config.enable_docker_enroot_pyxis:
    ExecuteBashScript("./utils/install_docker.sh").run()
    ExecuteBashScript("./utils/install_enroot_pyxis.sh").run(node_type)
```

- ii. You can integrate your HyperPod cluster with [Amazon Managed Service for Prometheus](#) and [Amazon Managed Grafana](#) to export metrics about the HyperPod cluster and cluster nodes to Amazon Managed Grafana dashboards. To export metrics and use the [Slurm dashboard](#), the [NVIDIA DCGM Exporter dashboard](#), and the [EFA Metrics dashboard](#) on

Amazon Managed Grafana, you need to install the [Slurm exporter for Prometheus](#), the [NVIDIA DCGM exporter](#), and the [EFA node exporter](#). For more information about installing the exporter packages and using Grafana dashboards on an Amazon Managed Grafana workspace, see [the section called “HyperPod cluster resources monitoring”](#).

To enable this installation step, set the `enable_observability` parameter to `True` in the [`config.py`](#) file.

```
# Install metric exporting software and Prometheus for observability

if Config.enable_observability:
    if node_type == SlurmNodeType.COMPUTE_NODE:
        ExecuteBashScript("./utils/install_docker.sh").run()
        ExecuteBashScript("./utils/install_dcgm_exporter.sh").run()
        ExecuteBashScript("./utils/install_efa_node_exporter.sh").run()

    if node_type == SlurmNodeType.HEAD_NODE:
        wait_for_scontrol()
        ExecuteBashScript("./utils/install_docker.sh").run()
        ExecuteBashScript("./utils/install_slurm_exporter.sh").run()
        ExecuteBashScript("./utils/install_prometheus.sh").run()
```

3. Make sure that you upload all configuration files and setup scripts from **Step 2** to the S3 bucket you provide in the `CreateCluster` request in **Step 1**. For example, assume that your `create_cluster.json` has the following.

```
"LifeCycleConfig": {

    "SourceS3URI": "s3://sagemaker-hyperpod-lifecycle/src",
    "OnCreate": "on_create.sh"
}
```

Then, your `s3://sagemaker-hyperpod-lifecycle/src` should contain `on_create.sh`, `lifecycle_script.py`, `provisioning_parameters.json`, and all other setup scripts. Assume that you have prepared the files in a local folder as follows.

```
### lifecycle_files // your local folder
### provisioning_parameters.json
### on_create.sh
### lifecycle_script.py
```

```
### ... // more setup scripts to be fed into lifecycle_script.py
```

To upload the files, use the S3 command as follows.

```
aws s3 cp --recursive ./lifecycle_scripts s3://sagemaker-hyperpod-lifecycle/src
```

## What particular configurations HyperPod manages in Slurm configuration files

When you create a Slurm cluster on HyperPod, the HyperPod agent sets up the [slurm.conf](#) and [gres.conf](#) files at /opt/slurm/etc/ to manage the Slurm cluster based on your HyperPod cluster creation request and lifecycle scripts. The following list shows which specific parameters the HyperPod agent handles and overwrites.

### Important

We strongly recommend that you **do not** change these parameters managed by HyperPod.

- In [slurm.conf](#), HyperPod sets up the following basic parameters: ClusterName, SlurmctldHost, PartitionName, and NodeName.

Also, to enable the [the section called “Auto-resume”](#) functionality, HyperPod requires the TaskPlugin and SchedulerParameters parameters set as follows. The HyperPod agent sets up these two parameters with the required values by default.

```
TaskPlugin=task/none  
SchedulerParameters=permit_job_expansion
```

- In [gres.conf](#), HyperPod manages NodeName for GPU nodes.

## Mount Amazon FSx for Lustre to a HyperPod cluster

To mount an Amazon FSx for Lustre shared file system to your HyperPod cluster, set up the following.

1. Use your Amazon VPC.

- a. For HyperPod cluster instances to communicate within your VPC, make sure that you attach the [the section called "Setting up SageMaker HyperPod with a custom Amazon VPC"](#) to the IAM role for SageMaker HyperPod.
- b. In `create_cluster.json`, include the following VPC information.

```
"VpcConfig": {  
    "SecurityGroupIds": [ "string" ],  
    "Subnets": [ "string" ]  
}
```

For more tips about setting up Amazon VPC, see [the section called "Prerequisites"](#).

2. To finish configuring Slurm with Amazon FSx for Lustre, specify the Amazon FSx DNS name and Amazon FSx mount name in `provisioning_parameters.json` as shown in the figure in the [the section called "Start with base lifecycle scripts provided by HyperPod"](#) section. You can find the Amazon FSx information either from the Amazon FSx for Lustre console in your account or by running the following AWS CLI command, `aws fsx describe-file-systems`.

```
"fsx_dns_name": "fs-12345678a90b01cde.fsx.us-west-2.amazonaws.com",  
"fsx_mountname": "1abcdefg"
```

## Validate the JSON configuration files before creating a Slurm cluster on HyperPod

To validate the JSON configuration files before submitting a cluster creation request, use the configuration validation script [validate-config.py](#). This script parses and compares your HyperPod cluster configuration JSON file and Slurm configuration JSON file, and identifies if there's any resource misconfiguration between the two files and also across Amazon EC2, Amazon VPC, and Amazon FSx resources. For example, to validate the `create_cluster.json` and `provisioning_parameters.json` files from the [the section called "Start with base lifecycle scripts provided by HyperPod"](#) section, run the validation script as follows.

```
python3 validate-config.py --cluster-config create_cluster.json --provisioning-parameters provisioning_parameters.json
```

The following is an example output of a successful validation.

```
## Validated instance group name worker-group-1 is correct ...
```

```
## Validated subnet subnet-012345abcdef67890 ...
## Validated security group sg-012345abcdef67890 ingress rules ...
## Validated security group sg-012345abcdef67890 egress rules ...
## Validated FSx Lustre DNS name fs-012345abcdef67890.fsx.us-east-1.amazonaws.com
## Validated FSx Lustre mount name abcdefgh
# Cluster Validation succeeded
```

## Validate runtime before running production workloads on a Slurm cluster on HyperPod

To check the runtime before running any production workloads on a Slurm cluster on HyperPod, use the runtime validation script [hyperpod-precheck.py](#). This script checks if the Slurm cluster has all packages installed for running Docker, if the cluster has a properly mounted FSx for Lustre file system and a user directory sharing the file system, and if the Slurm deamon is running on all compute nodes.

To run the script on multiple nodes at once, use `srun` as shown in the following example command of running the script on a Slurm cluster of 8 nodes.

```
# The following command runs on 8 nodes
srun -N 8 python3 hyperpod-precheck.py
```

### Note

To learn more about the validation script such as what runtime validation functions the script provides and guidelines to resolve issues that don't pass the validations, see [Runtime validation before running workloads](#) in the *Awsome Distributed Training GitHub repository*.

## Develop lifecycle scripts interactively on a HyperPod cluster node

This section explains how you can interactively develop lifecycle scripts without repeatedly creating and deleting a HyperPod cluster.

1. Create a HyperPod cluster with the base lifecycle scripts.
2. Log in to a cluster node.
3. Develop a script (`configure_xyz.sh`) by editing and running it repeatedly on the node.
  - a. HyperPod runs the lifecycle scripts as the root user, so we recommend that you run the `configure_xyz.sh` as the root user while developing to make sure that the script is tested under the same condition while run by HyperPod.

4. Integrate the script into `lifecycle_script.py` by adding a code line similar to the following.

```
ExecuteBashScript("./utils/configure_xyz.sh").run()
```

5. Upload the updated lifecycle scripts to the S3 bucket that you initially used for uploading the base lifecycle scripts.

6. Test the integrated version of `lifecycle_script.py` by creating a new HyperPod cluster.

### Update a cluster with new or updated lifecycle scripts

There are three ways to update the HyperPod cluster software.

- The `UpdateClusterSoftware` API for patching the HyperPod software re-runs the lifecycle scripts on the entire instance group.
- The `UpdateCluster` API only runs the lifecycle scripts for new instance groups.
- You can also run lifecycle scripts directly in the HyperPod instances.

#### Note

HyperPod runs [the section called “SageMaker HyperPod DLAMI”](#) on each instance of a cluster, and the AMI has pre-installed software packages complying compatibilities between them and HyperPod functionalities. Note that if you reinstall any of the pre-installed packages, you are responsible for installing compatible packages and note that some HyperPod functionalities might not work as expected.

### Jobs on SageMaker HyperPod clusters

The following topics provide procedures and examples of accessing compute nodes and running ML workloads on provisioned SageMaker HyperPod clusters. Depending on how you have set up the environment on your HyperPod cluster, there are many ways to run ML workloads on HyperPod clusters. Examples of running ML workloads on HyperPod clusters are also provided in the [Awesome Distributed Training GitHub repository](#). The following topics walk you through how to log in to the provisioned HyperPod clusters and get you started with running sample ML workloads.

**Tip**

To find practical examples and solutions, see also the [SageMaker HyperPod workshop](#).

**Topics**

- [Access your SageMaker HyperPod cluster nodes](#)
- [Schedule a Slurm job on a SageMaker HyperPod cluster](#)
- [Run Docker containers on a Slurm compute node on HyperPod](#)
- [Run distributed training workloads with Slurm on HyperPod](#)

## Access your SageMaker HyperPod cluster nodes

You can access your **InService** cluster through AWS Systems Manager (SSM) by running the AWS CLI command `aws ssm start-session` with the SageMaker HyperPod cluster host name in format of `sagemaker-cluster:[cluster-id]_[instance-group-name]-[instance-id]`. You can retrieve the cluster ID, the instance ID, and the instance group name from the [SageMaker HyperPod console](#) or by running `describe-cluster` and `list-cluster-nodes` from the [AWS CLI commands for SageMaker HyperPod](#). For example, if your cluster ID is `aa11bbbbbb222`, the cluster node name is `controller-group`, and the cluster node ID is `i-111222333444555aa`, the SSM `start-session` command should be the following.

**Note**

Granting users access to HyperPod cluster nodes allows them to install and operate user-managed software on the nodes. Ensure that you maintain the principle of least-privilege permissions for users.

If you haven't set up AWS Systems Manager, follow the instructions provided at [the section called "Setting up AWS Systems Manager and Run As for cluster user access control"](#).

```
$ aws ssm start-session \
--target sagemaker-cluster:aa11bbbbbb222_controller-group-i-111222333444555aa \
--region us-west-2
Starting session with SessionId: s0011223344aabccdd
root@ip-111-22-333-444:/usr/bin#
```

Note that this initially connects you as the root user. Before running jobs, switch to the ubuntu user by running the following command.

```
root@ip-111-22-333-444:/usr/bin# sudo su - ubuntu  
ubuntu@ip-111-22-333-444:/usr/bin#
```

For advanced settings for practical use of HyperPod clusters, see the following topics.

## Topics

- [Additional tips for accessing your SageMaker HyperPod cluster nodes](#)
- [Set up a multi-user environment through the Amazon FSx shared space](#)
- [Set up a multi-user environment by integrating HyperPod clusters with Active Directory](#)

## Additional tips for accessing your SageMaker HyperPod cluster nodes

### Use the `easy-ssh.sh` script provided by HyperPod for simplifying the connection process

To make the previous process into a single line command, the HyperPod team provides the [`easy-ssh.sh`](#) script that retrieves your cluster information, aggregates them into the SSM command, and connects to the compute node. You don't need to manually look for the required HyperPod cluster information as this script runs `describe-cluster` and `list-cluster-nodes` commands and parses the information needed for completing the SSM command. The following example commands show how to run the [`easy-ssh.sh`](#) script. If it runs successfully, you'll be connected to the cluster as the root user. It also prints a code snippet to set up SSH by adding the HyperPod cluster as a remote host through an SSM proxy. By setting up SSH, you can connect your local development environment such as Visual Studio Code with the HyperPod cluster.

```
$ chmod +x easy-ssh.sh  
$ ./easy-ssh.sh -c <node-group> <cluster-name>  
Cluster id: <cluster_id>  
Instance id: <instance_id>  
Node Group: <node-group>  
Add the following to your ~/.ssh/config to easily connect:  
  
$ cat <<EOF >> ~/.ssh/config  
Host <cluster-name>  
    User ubuntu
```

```
ProxyCommand sh -c "aws ssm start-session --target sagemaker-cluster:<cluster_id>_<node-group>-<instance_id> --document-name AWS-StartSSHSession --parameters 'portNumber=%p'"  
EOF
```

Add your ssh keypair and then you can do:

```
$ ssh <cluster-name>  
  
aws ssm start-session --target sagemaker-cluster:<cluster_id>_<node-group>-<instance_id>  
  
Starting session with SessionId: s0011223344aabccdd  
root@ip-111-22-333-444:/usr/bin#
```

Note that this initially connects you as the root user. Before running jobs, switch to the ubuntu user by running the following command.

```
root@ip-111-22-333-444:/usr/bin# sudo su - ubuntu  
ubuntu@ip-111-22-333-444:/usr/bin#
```

## Set up for easy access with SSH by using the HyperPod compute node as a remote host

To further simplify access to the compute node using SSH from a local machine, the easy-ssh.sh script outputs a code snippet of setting up the HyperPod cluster as a remote host as shown in the previous section. The code snippet is auto-generated to help you directly add to the `~/.ssh/config` file on your local device. The following procedure shows how to set up for easy access using SSH through the SSM proxy, so that you or your cluster users can directly run `ssh <cluster-name>` to connect to the HyperPod cluster node.

1. On your local device, add the HyperPod compute node with a user name as a remote host to the `~/.ssh/config` file. The following command shows how to append the auto-generated code snippet from the easy-ssh.sh script to the `~/.ssh/config` file. Make sure that you copy it from the auto-generated output of the easy-ssh.sh script that has the correct cluster information.

```
$ cat <<EOF >> ~/.ssh/config  
Host <cluster-name>  
User ubuntu
```

```
ProxyCommand sh -c "aws ssm start-session --target sagemaker-cluster:<cluster_id>_<node-group>-<instance_id> --document-name AWS-StartSSHSession --parameters 'portNumber=%p'"  
EOF
```

2. On the HyperPod cluster node, add the public key on your local device to the `~/.ssh/authorized_keys` file on the HyperPod cluster node.
  - a. Print the public key file on your local machine.

```
$ cat ~/.ssh/id_rsa.pub
```

This should return your key. Copy the output of this command.

(Optional) If you don't have a public key, create one by running the following command.

```
$ ssh-keygen -t rsa -q -f "$HOME/.ssh/id_rsa" -N ""
```

- b. Connect to the cluster node and switch to the user to add the key. The following command is an example of accessing as the `ubuntu` user. Replace `ubuntu` to the user name for which you want to set up the easy access with SSH.

```
$ ./easy-ssh.sh -c <node-group> <cluster-name>  
$ sudo su - ubuntu  
ubuntu@ip-111-22-333-444:/usr/bin#
```

- c. Open the `~/.ssh/authorized_keys` file and add the public key at the end of the file.

```
ubuntu@ip-111-22-333-444:/usr/bin# vim ~/.ssh/authorized_keys
```

After you finish setting up, you can connect to the HyperPod cluster node as the user by running a simplified SSH command as follows.

```
$ ssh <cluster-name>  
ubuntu@ip-111-22-333-444:/usr/bin#
```

Also, you can use the host for remote development from an IDE on your local device, such as [Visual Studio Code Remote - SSH](#).

## Set up a multi-user environment through the Amazon FSx shared space

You can use the Amazon FSx shared space to manage a multi-user environment in a Slurm cluster on SageMaker HyperPod. If you have configured your Slurm cluster with Amazon FSx during the HyperPod cluster creation, this is a good option for setting up workspace for your cluster users. Create a new user and setup the home directory for the user on the Amazon FSx shared file system.

### Tip

To allow users to access your cluster through their user name and dedicated directories, you should also associate them with IAM roles or users by tagging them as guided in **Option 2** of step 5 under the procedure **To turn on Run As support for Linux and macOS managed nodes** provided at [Turn on Run As support for Linux and macOS managed nodes](#) in the AWS Systems Manager User Guide. See also [the section called “Setting up AWS Systems Manager and Run As for cluster user access control”](#).

### To set up a multi-user environment while creating a Slurm cluster on SageMaker HyperPod

The SageMaker HyperPod service team provides a script [add\\_users.sh](#) as part of the base lifecycle script samples.

1. Prepare a text file named `shared_users.txt` that you need to create in the following format. The first column is for user names, the second column is for unique user IDs, and the third column is for the user directories in the Amazon FSx shared space.

```
username1,uid1,/fsx/username1  
username2,uid2,/fsx/username2  
...  
...
```

2. Make sure that you upload the `shared_users.txt` and [add\\_users.sh](#) files to the S3 bucket for HyperPod lifecycle scripts. While the cluster creation, cluster update, or cluster software update is in progress, the [add\\_users.sh](#) reads in the `shared_users.txt` and sets up the user directories properly.

### To create new users and add to an existing Slurm cluster running on SageMaker HyperPod

1. On the head node, run the following command to save a script that helps create a user. Make sure that you run this with sudo permissions.

```
$ cat > create-user.sh << EOL
#!/bin/bash

set -x

# Prompt user to get the new user name.
read -p "Enter the new user name, i.e. 'sean':"
" USER

# create home directory as /fsx/<user>
# Create the new user on the head node
sudo useradd \$USER -m -d /fsx/\$USER --shell /bin/bash;
user_id=\$(id -u \$USER)

# add user to docker group
sudo usermod -aG docker \$USER

# setup SSH Keypair
sudo -u \$USER ssh-keygen -t rsa -q -f "/fsx/\$USER/.ssh/id_rsa" -N ""
sudo -u \$USER cat /fsx/\$USER/.ssh/id_rsa.pub | sudo -u \$USER tee /fsx/\$USER/.ssh/
authorized_keys

# add user to compute nodes
read -p "Number of compute nodes in your cluster, i.e. 8:"
" NUM_NODES
srun -N \$NUM_NODES sudo useradd -u \$user_id \$USER -d /fsx/\$USER --shell /bin/
bash;

# add them as a sudoer
read -p "Do you want this user to be a sudoer? (y/N):"
" SUDO
if [ "\$SUDO" = "y" ]; then
    sudo usermod -aG sudo \$USER
    sudo srun -N \$NUM_NODES sudo usermod -aG sudo \$USER
    echo -e "If you haven't already you'll need to run:\n\nsudo visudo /
etc/sudoers\n\nChange the line:\n\n%sudo  ALL=(ALL:ALL) ALL\nTo\n\n%sudo
ALL=(ALL:ALL) NOPASSWD: ALL\n\nOn each node."
fi
EOL
```

2. Run the script with the following command. You'll be prompted for adding the name of a user and the number of compute nodes that you want to allow the user to access.

```
$ bash create-user.sh
```

3. Test the user by running the following commands.

```
$ sudo su - <user> && ssh $(srun hostname)
```

4. Add the user information to the `shared_users.txt` file, so the user will be created on any new compute nodes or new clusters.

## Set up a multi-user environment by integrating HyperPod clusters with Active Directory

In practical use cases, HyperPod clusters are typically used by multiple users: machine learning (ML) researchers, software engineers, data scientists, and cluster administrators. They edit their own files and run their own jobs without impacting each other's work. To set up a multi-user environment, use the Linux user and group mechanism to statically create multiple users on each instance through lifecycle scripts. However, the drawback to this approach is that you need to duplicate user and group settings across multiple instances in the cluster to keep a consistent configuration across all instances when you make updates such as adding, editing, and removing users.

To solve this, you can use [Lightweight Directory Access Protocol \(LDAP\)](#) and [LDAP over TLS/SSL \(LDAPS\)](#) to integrate with a directory service such as [AWS Directory Service for Microsoft Active Directory](#). To learn more about setting up Active Directory and a multi-user environment in a HyperPod cluster, see the blog post [Integrate HyperPod clusters with Active Directory for seamless multi-user login](#).

## Schedule a Slurm job on a SageMaker HyperPod cluster

You can launch training jobs using the standard Slurm `sbatch` or `srun` commands. For example, to launch an 8-node training job, you can run `srun -N 8 --exclusive train.sh` SageMaker HyperPod supports training in a range of environments, including `conda`, `venv`, `docker`, and `enroot`. You can configure an ML environment by running lifecycle scripts on your SageMaker HyperPod clusters. You also have an option to attach a shared file system such as Amazon FSx, which can also be used as a virtual environment.

The following example shows how to run a job for training Llama-2 with the Fully Sharded Data Parallelism (FSDP) technique on a SageMaker HyperPod cluster with an Amazon FSx shared file system. You can also find more examples from the [Awsome Distributed Training GitHub repository](#).

**Tip**

All SageMaker HyperPod examples are available in the `3.test_cases` folder of the [Awsome Distributed Training GitHub repository](#).

1. Clone the [Awsome Distributed Training GitHub repository](#), and copy the training job examples to your Amazon FSx file system.

```
$ TRAINING_DIR=/fsx/users/my-user/fsdp  
$ git clone https://github.com/aws-samples/awsome-distributed-training/
```

2. Run the `create_conda_env.sh` script. This creates a conda environment on your Amazon FSx file system. Make sure that the file system is accessible to all nodes in the cluster.
3. Build the virtual Conda environment by launching a single node slurm job as follows.

```
$ srun -N 1 /path_to/create_conda_env.sh
```

4. After the environment is built, you can launch a training job by pointing to the environment path on the shared volume. You can launch both single-node and multi-node training jobs with the same setup. To launch a job, create a job launcher script (also called an entry point script) as follows.

```
#!/usr/bin/env bash  
set -ex  
  
ENV_PATH=/fsx/users/my_user/pytorch_env  
TORCHRUN=$ENV_PATH/bin/torchrn  
TRAINING_SCRIPT=/fsx/users/my_user/pt_train.py  
  
WORLD_SIZE_JOB=$SLURM_NTASKS  
RANK_NODE=$SLURM_NODEID  
PROC_PER_NODE=8  
MASTER_ADDR=(`scontrol show hostnames \$SLURM_JOB_NODELIST | head -n 1`)  
MASTER_PORT=$((expr 10000 + $(echo -n \$SLURM_JOBID | tail -c 4)))  
  
DIST_ARGS="--nproc_per_node=\$PROC_PER_NODE \  
          --nnodes=\$WORLD_SIZE_JOB \  
          --node_rank=\$RANK_NODE \  
          --master_addr=\$MASTER_ADDR \  
          --master_port=\$MASTER_PORT \  
          "
```

```
$TORCHRUN $DIST_ARGS $TRAINING_SCRIPT
```

### Tip

If you want to make your training job more resilient against hardware failures by using the auto-resume capability of SageMaker HyperPod, you need to properly set up the environment variable MASTER\_ADDR in the entrypoint script. To learn more, see [the section called “Auto-resume”](#).

This tutorial assumes that this script is saved as /fsx/users/my\_user/train.sh.

- With this script in the shared volume at /fsx/users/my\_user/train.sh, run the following srun command to schedule the Slurm job.

```
$ cd /fsx/users/my_user/  
$ srun -N 8 train.sh
```

## Run Docker containers on a Slurm compute node on HyperPod

To run Docker containers with Slurm on SageMaker HyperPod, you need to use [Enroot](#) and [Pyxis](#). The Enroot package helps convert Docker images into a runtime that Slurm can understand, while the Pyxis enables scheduling the runtime as a Slurm job through an srun command, srun --container-image=*docker/image:tag*.

### Tip

The Docker, Enroot, and Pyxis packages should be installed during cluster creation as part of running the lifecycle scripts as guided in [the section called “Start with base lifecycle scripts provided by HyperPod”](#). Use the [base lifecycle scripts](#) provided by the HyperPod service team when creating a HyperPod cluster. Those base scripts are set up to install the packages by default. In the [config.py](#) script, there's the Config class with the boolean type parameter for installing the packages set to True (enable\_docker\_enroot\_pyxis=True). This is called by and parsed in the [lifecycle\\_script.py](#) script, which calls `install_docker.sh` and `install_enroot_pyxis.sh` scripts from the [utils](#) folder. The installation scripts are

where the actual installations of the packages take place. Additionally, the installation scripts identify if they can detect NVMe store paths from the instances they are run on and set up the root paths for Docker and Enroot to /opt/dlami/nvme. The default root volume of any fresh instance is mounted to /tmp only with a 100GB EBS volume, which runs out if the workload you plan to run involves training of LLMs and thus large size Docker containers. If you use instance families such as P and G with local NVMe storage, you need to make sure that you use the NVMe storage attached at /opt/dlami/nvme, and the installation scripts take care of the configuration processes.

## To check if the root paths are set up properly

On a compute node of your Slurm cluster on SageMaker HyperPod, run the following commands to make sure that the lifecycle script worked properly and the root volume of each node is set to /opt/dlami/nvme/\*. The following commands shows examples of checking the Enroot runtime path and the data root path for 8 compute nodes of a Slurm cluster.

```
$ srun -N 8 cat /etc/enroot/enroot.conf | grep "ENROOT_RUNTIME_PATH"
ENROOT_RUNTIME_PATH      /opt/dlami/nvme/tmp/enroot/user-$(id -u)
... // The same or similar lines repeat 7 times
```

```
$ srun -N 8 cat /etc/docker/daemon.json
{
  "data-root": "/opt/dlami/nvme/docker/data-root"
}
... // The same or similar lines repeat 7 times
```

After you confirm that the runtime paths are properly set to /opt/dlami/nvme/\*, you're ready to build and run Docker containers with Enroot and Pyxis.

## To test Docker with Slurm

1. On your compute node, try the following commands to check if Docker and Enroot are properly installed.

```
$ docker --help
$ enroot --help
```

2. Test if Pyxis and Enroot installed correctly by running one of the [NVIDIA CUDA Ubuntu](#) images.

```
$ srun --container-image=nvidia/cuda:XX.Y.Z-base-ubuntuXX.YY nvidia-smi
pyxis: importing docker image: nvidia/cuda:XX.Y.Z-base-ubuntuXX.YY
pyxis: imported docker image: nvidia/cuda:XX.Y.Z-base-ubuntuXX.YY
DAY MMM DD HH:MM:SS YYYY
+-----+
| NVIDIA-SMI 470.141.03    Driver Version: 470.141.03    CUDA Version: XX.YY      |
|-----+-----+-----+
| GPU  Name      Persistence-M| Bus-Id      Disp.A | Volatile Uncorr. ECC  | |
| Fan  Temp  Perf  Pwr:Usage/Cap|          Memory-Usage | GPU-Util  Compute M.  |
|                   |                               |           | MIG M.  |
|=====+=====+=====+=====+=====+=====+=====+=====+=====+
|   0  Tesla T4          Off  | 00000000:00:1E.0 Off |                  0 | |
| N/A   40C     P0    27W /  70W |          0MiB / 15109MiB |      0%     Default  |
|                   |                               |           | N/A  |
+-----+-----+-----+
+-----+
| Processes:                               |
| GPU  GI  CI      PID  Type  Process name          GPU Memory  |
|       ID  ID
|=====+=====+=====+=====+=====+=====+=====+=====+
|   No running processes found
+-----+
```

You can also test it by creating a script and running an `sbatch` command as follows.

```
$ cat <<EOF >> container-test.sh
#!/bin/bash
#SBATCH --container-image=nvidia/cuda:XX.Y.Z-base-ubuntuXX.YY
nvidia-smi
EOF

$ sbatch container-test.sh
pyxis: importing docker image: nvidia/cuda:XX.Y.Z-base-ubuntuXX.YY
pyxis: imported docker image: nvidia/cuda:XX.Y.Z-base-ubuntuXX.YY
DAY MMM DD HH:MM:SS YYYY
+-----+
| NVIDIA-SMI 470.141.03    Driver Version: 470.141.03    CUDA Version: XX.YY      |
|-----+-----+-----+
| GPU  Name      Persistence-M| Bus-Id      Disp.A | Volatile Uncorr. ECC  | |
| Fan  Temp  Perf  Pwr:Usage/Cap|          Memory-Usage | GPU-Util  Compute M.  |
|                   |                               |           | MIG M.  |
|=====+=====+=====+=====+=====+=====+=====+=====+=====+
```

```
|=====+=====+=====+=====+=====+|  
| 0 Tesla T4          Off | 00000000:00:1E.0 Off |          0 |  
| N/A   40C    P0     27W / 70W |      0MiB / 15109MiB |      0%     Default |  
|                           |                           |          N/A |  
+-----+-----+-----+  
  
+-----+  
| Processes:          |  
| GPU   GI   CI       PID   Type   Process name           GPU Memory |  
|        ID   ID             |                           | Usage           |  
+=====+=====+=====+=====+=====+=====+  
| No running processes found  
+-----+
```

## To run a test Slurm job with Docker

After you have completed setting up Slurm with Docker, you can bring any pre-built Docker images and run using Slurm on SageMaker HyperPod. The following is a sample use case that walks you through how to run a training job using Docker and Slurm on SageMaker HyperPod. It shows an example job of model-parallel training of the Llama 2 model with the SageMaker AI model parallelism (SMP) library.

1. If you want to use one of the pre-built ECR images distributed by SageMaker AI or DLC, make sure that you give your HyperPod cluster the permissions to pull ECR images through the [the section called "IAM role for SageMaker HyperPod"](#). If you use your own or an open source Docker image, you can skip this step. Add the following permissions to the [the section called "IAM role for SageMaker HyperPod"](#). In this tutorial, we use the [SMP Docker image](#) pre-packaged with the SMP library .

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "ecr:BatchCheckLayerAvailability",  
        "ecr:BatchGetImage",  
        "ecr-public:*",  
        "ecr:GetDownloadUrlForLayer",  
        "ecr:GetAuthorizationToken",  
        "sts:*"
```

```
        ],
        "Resource": "*"
    }
]
}
```

2. On the compute node, clone the repository and go to the folder that provides the example scripts of training with SMP.

```
$ git clone https://github.com/aws-samples/awsome-distributed-training/
$ cd awsome-distributed-training/3.test_cases/17.SM-modelparallelv2
```

3. In this tutorial, run the sample script [docker\\_build.sh](#) that pulls the SMP Docker image, build the Docker container, and runs it as an Enroot runtime. You can modify this as you want.

```
$ cat docker_build.sh
#!/usr/bin/env bash

region=us-west-2
dlc_account_id=658645717510
aws ecr get-login-password --region $region | docker login --username AWS --password-
stdin $dlc_account_id.dkr.ecr.$region.amazonaws.com

docker build -t smpv2 .
enroot import -o smpv2.sqsh docker://smpv2:latest
```

```
$ bash docker_build.sh
```

4. Create a batch script to launch a training job using sbatch. In this tutorial, the provided sample script [launch\\_training\\_enroot.sh](#) launches a model-parallel training job of the 70-billion-parameter Llama 2 model with a synthetic dataset on 8 compute nodes. A set of training scripts are provided at [3.test\\_cases/17.SM-modelparallelv2/scripts](#), and `launch_training_enroot.sh` takes `train_external.py` as the entrypoint script.

### Important

To use the a Docker container on SageMaker HyperPod, you must mount the `/var/log` directory from the host machine, which is the HyperPod compute node in this case, onto the `/var/log` directory in the container. You can set it up by adding the following variable for Enroot.

```
"${{HYPERPOD_PATH:="/var/log/aws/clusters":"/var/log/aws/clusters"}"
```

```
$ cat launch_training_enroot.sh
#!/bin/bash

# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0

#SBATCH --nodes=8 # number of nodes to use, 2 p4d(e) = 16 A100 GPUs
#SBATCH --job-name=smpv2_llama # name of your job
#SBATCH --exclusive # job has exclusive use of the resource, no sharing
#SBATCH --wait-all-nodes=1

set -ex;

#####
##### User Variables #####
#####

#####
model_type=llama_v2
model_size=70b

# Toggle this to use synthetic data
use_synthetic_data=1

# To run training on your own data set Training/Test Data path -> Change this to
# the tokenized dataset path in Fsx. Acceptable formats are huggingface (arrow) and
# Jsonlines.
# Also change the use_synthetic_data to 0

export TRAINING_DIR=/fsx/path_to_data
export TEST_DIR=/fsx/path_to_data
export CHECKPOINT_DIR=$(pwd)/checkpoints

# Variables for Enroot
: "${IMAGE:=$(pwd)/smpv2.sqsh}"
: "${{{HYPERPOD_PATH:="/var/log/aws/clusters":"/var/log/aws/clusters"}]}" # This is
# needed for validating its hyperpod cluster
```

```
: "${TRAIN_DATA_PATH:=$TRAINING_DIR:$TRAINING_DIR}"
: "${TEST_DATA_PATH:=$TEST_DIR:$TEST_DIR}"
: "${CHECKPOINT_PATH:=$CHECKPOINT_DIR:$CHECKPOINT_DIR}"

#####
## Environment Variables ##
#####

#export NCCL_SOCKET_IFNAME=en
export NCCL_ASYNC_ERROR_HANDLING=1

export NCCL_PROTO="simple"
export NCCL_SOCKET_IFNAME="^lo,docker"
export RDMAV_FORK_SAFE=1
export FI_EFA_USE_DEVICE_RDMA=1
export NCCL_DEBUG_SUBSYS=off
export NCCL_DEBUG="INFO"
export SM_NUM_GPUS=8
export GPU_NUM_DEVICES=8
export FI_EFA_SET_CUDA_SYNC_MEMOPS=0

# async runtime error ...
export CUDA_DEVICE_MAX_CONNECTIONS=1

#####
## Command and Options ##
#####

if [ "$model_size" == "7b" ]; then
    HIDDEN_WIDTH=4096
    NUM_LAYERS=32
    NUM_HEADS=32
    LLAMA_INTERMEDIATE_SIZE=11008
    DEFAULT_SHARD_DEGREE=8
# More Llama model size options
elif [ "$model_size" == "70b" ]; then
    HIDDEN_WIDTH=8192
    NUM_LAYERS=80
    NUM_HEADS=64
    LLAMA_INTERMEDIATE_SIZE=28672
    # Reduce for better perf on p4de
    DEFAULT_SHARD_DEGREE=64
```

```
fi

if [ -z "$shard_degree" ]; then
    SHARD_DEGREE=$DEFAULT_SHARD_DEGREE
else
    SHARD_DEGREE=$shard_degree
fi

if [ -z "$LLAMA_INTERMEDIATE_SIZE" ]; then
    LLAMA_ARGS=""
else
    LLAMA_ARGS="--llama_intermediate_size $LLAMA_INTERMEDIATE_SIZE "
fi

if [ $use_synthetic_data == 1 ]; then
    echo "using synthetic data"
    declare -a ARGS=(
        --container-image $IMAGE
        --container-mounts $HYPERPOD_PATH,$CHECKPOINT_PATH
    )
else
    echo "using real data...."
    declare -a ARGS=(
        --container-image $IMAGE
        --container-mounts $HYPERPOD_PATH,$TRAIN_DATA_PATH,$TEST_DATA_PATH,
$CHECKPOINT_PATH
    )
fi

declare -a TORCHRUN_ARGS=(
    # change this to match the number of gpus per node:
    --nproc_per_node=8 \
    --nnodes=$SLURM_JOB_NUM_NODES \
    --rdzv_id=$SLURM_JOB_ID \
    --rdzv_backend=c10d \
    --rdzv_endpoint=$(hostname) \
)
srun -l "${ARGS[@]}" torchrun "${TORCHRUN_ARGS[@]}" /path_to/train_external.py \
    --train_batch_size 4 \
    --max_steps 100 \
```

```
--hidden_width $HIDDEN_WIDTH \
--num_layers $NUM_LAYERS \
--num_heads $NUM_HEADS \
${LLAMA_ARGS} \
--shard_degree $SHARD_DEGREE \
--model_type $model_type \
--profile_nsys 1 \
--use_smp_implementation 1 \
--max_context_width 4096 \
--tensor_parallel_degree 1 \
--use_synthetic_data $use_synthetic_data \
--training_dir $TRAINING_DIR \
--test_dir $TEST_DIR \
--dataset_type hf \
--checkpoint_dir $CHECKPOINT_DIR \
--checkpoint_freq 100 \
```

```
$ sbatch Launch_training_enroot.sh
```

To find the downloadable code examples, see [Run a model-parallel training job using the SageMaker AI model parallelism library, Docker and Enroot with Slurm in the Awsome Distributed Training GitHub repository](#). For more information about distributed training with a Slurm cluster on SageMaker HyperPod, proceed to the next topic at [the section called “Run distributed training workloads with Slurm on HyperPod”](#).

## Run distributed training workloads with Slurm on HyperPod

SageMaker HyperPod is specialized for workloads of training large language models (LLMs) and foundation models (FMs). These workloads often require the use of multiple parallelism techniques and optimized operations for ML infrastructure and resources. Using SageMaker HyperPod, you can use the following SageMaker AI distributed training frameworks:

- The [SageMaker AI distributed data parallelism \(SMDDP\) library](#) that offers collective communication operations optimized for AWS.
- The [SageMaker AI model parallelism \(SMP\) library](#) that implements various model parallelism techniques.

## Topics

- [Using SMDDP on a SageMaker HyperPod](#)

- [Using SMP on a SageMaker HyperPod cluster](#)

## Using SMDDP on a SageMaker HyperPod

The [SMDDP library](#) is a collective communication library that improves compute performance of distributed data parallel training. The SMDDP library works with the following open source distributed training frameworks:

- [PyTorch distributed data parallel \(DDP\)](#)
- [PyTorch fully sharded data parallelism \(FSDP\)](#)
- [DeepSpeed](#)
- [Megatron-DeepSpeed](#)

The SMDDP library addresses communications overhead of the key collective communication operations by offering the following for SageMaker HyperPod.

- The library offers AllGather optimized for AWS. AllGather is a key operation used in sharded data parallel training, which is a memory-efficient data parallelism technique offered by popular libraries. These include the SageMaker AI model parallelism (SMP) library, DeepSpeed Zero Redundancy Optimizer (ZeRO), and PyTorch Fully Sharded Data Parallelism (FSDP).
- The library performs optimized node-to-node communication by fully utilizing the AWS network infrastructure and the SageMaker AI ML instance topology.

## To run sample data-parallel training jobs

Explore the following distributed training samples implementing data parallelism techniques using the SMDDP library.

- [awsome-distributed-training/3.test\\_cases/12.SM-dataparallel-FSDP](#)
- [awsome-distributed-training/3.test\\_cases/13.SM-dataparallel-deepspeed](#)

## To set up an environment for using the SMDDP library on SageMaker HyperPod

The following are training environment requirements for using the SMDDP library on SageMaker HyperPod.

- PyTorch v2.0.1 and later

- CUDA v11.8 and later
- libstdc++ runtime version greater than 3
- Python v3.10.x and later
- m1.p4d.24xlarge and m1.p4de.24xlarge, which are supported instance types by the SMDDP library
- imdsv2 enabled on training host

Depending on how you want to run the distributed training job, there are two options to install the SMDDP library:

- A direct installation using the SMDDP binary file.
- Using the SageMaker AI Deep Learning Containers (DLCs) pre-installed with the SMDDP library.

Docker images pre-installed with the SMDDP library or the URLs to the SMDDP binary files are listed at [Supported Frameworks](#) in the SMDDP library documentation.

### To install the SMDDP library on the SageMaker HyperPod DLAMI

- `pip install --no-cache-dir https://smdataparallel.s3.amazonaws.com/binary/pytorch/<pytorch-version>/cuXYZ/YYYY-MM-DD/smdistributed_dataparallel-X.Y.Z-cp310-cp310-linux_x86_64.whl`

#### Note

If you work in a Conda environment, ensure that you install PyTorch using `conda install` instead of `pip`.

```
conda install pytorch==X.Y.Z torchvision==X.Y.Z torchaudio==X.Y.Z pytorch-cuda=X.Y.Z -c pytorch -c nvidia
```

### To use the SMDDP library on a Docker container

- The SMDDP library is pre-installed on the SageMaker AI Deep Learning Containers (DLCs). To find the list of SageMaker AI framework DLCs for PyTorch with the SMDDP library, see [Supported Frameworks](#) in the SMDDP library documentation. You can also bring your own Docker container

with required dependencies installed to use the SMDDP library. To learn more about setting up a custom Docker container to use the SMDDP library, see also [the section called “Create your own docker container with the library”](#).

### **Important**

To use the SMDDP library in a Docker container, mount the `/var/log` directory from the host machine onto `/var/log` in the container. This can be done by adding the following option when running your container.

```
docker run <OTHER_OPTIONS> -v /var/log:/var/log ...
```

To learn how to run data-parallel training jobs with SMDDP in general, see [the section called “Distributed training with the SMDDP library”](#).

## Using SMP on a SageMaker HyperPod cluster

The [SageMaker AI model parallelism \(SMP\) library](#) offers various [state-of-the-art model parallelism techniques](#), including:

- fully sharded data parallelism
- expert parallelism
- mixed precision training with FP16/BF16 and FP8 data types
- tensor parallelism

The SMP library is also compatible with open source frameworks such as PyTorch FSDP, NVIDIA Megatron, and NVIDIA Transformer Engine.

## To run a sample model-parallel training workload

The SageMaker AI service teams provide sample training jobs implementing model parallelism with the SMP library at [awsome-distributed-training/3.test\\_cases/17.SM-modelparallelv2](#).

## SageMaker HyperPod cluster resources monitoring

To achieve comprehensive observability into your SageMaker HyperPod cluster resources and software components, integrate the cluster with [Amazon Managed Service for Prometheus](#) and [Amazon Managed Grafana](#). The integration with Amazon Managed Service for Prometheus enables the export of metrics related to your HyperPod cluster resources, providing insights into their performance, utilization, and health. The integration with Amazon Managed Grafana enables the visualization of these metrics through various Grafana dashboards that offer intuitive interface for monitoring and analyzing the cluster's behavior. By leveraging these services, you gain a centralized and unified view of your HyperPod cluster, facilitating proactive monitoring, troubleshooting, and optimization of your distributed training workloads.

### Tip

To find practical examples and solutions, see also the [SageMaker HyperPod workshop](#).

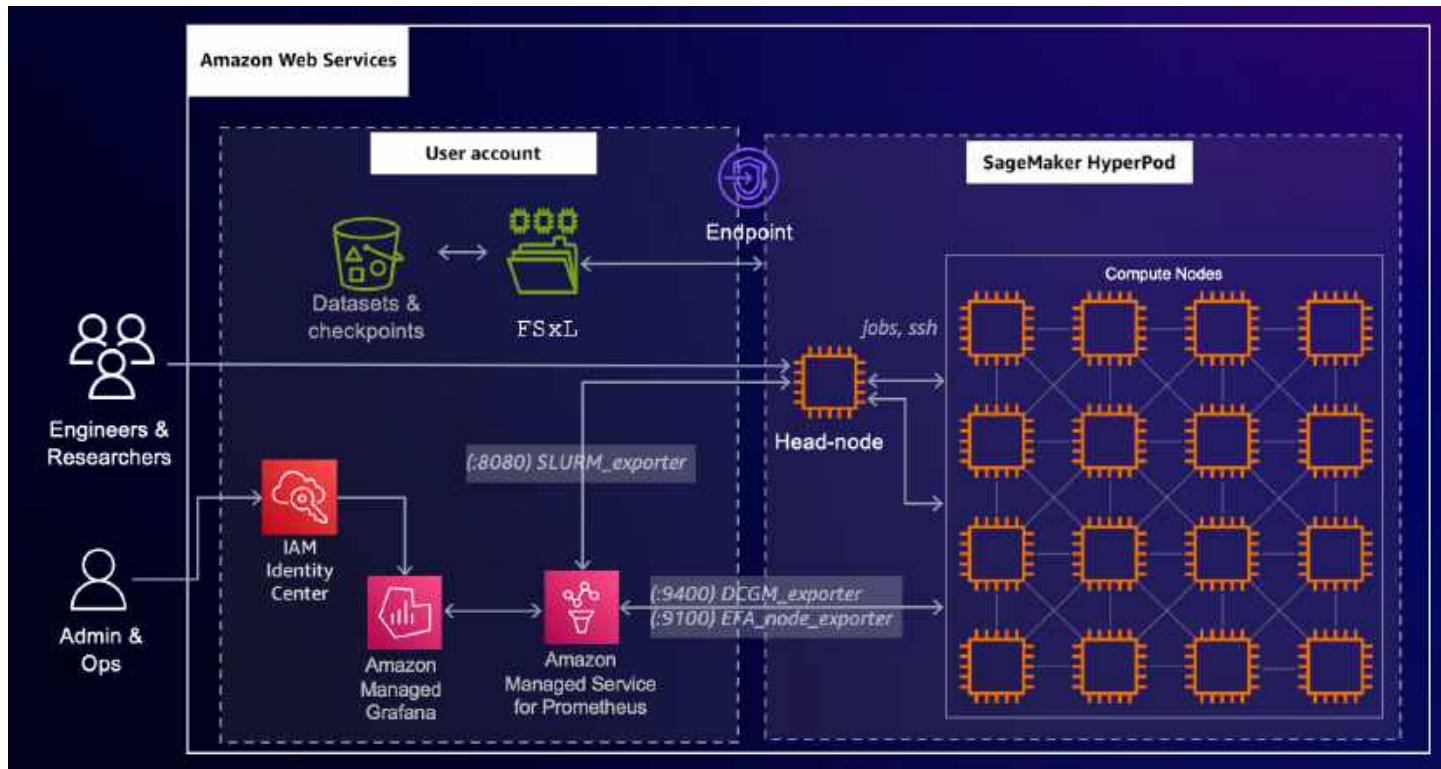


Figure: This architecture diagram shows an overview of configuring SageMaker HyperPod with Amazon Managed Service for Prometheus and Amazon Managed Grafana.

Proceed to the following topics to set up for SageMaker HyperPod cluster observability.

## Topics

- [Complete prerequisites for SageMaker HyperPod cluster observability](#)
- [Install metrics exporter packages on your HyperPod cluster](#)
- [Validate Prometheus setup on the head node of a HyperPod cluster](#)
- [Set up an Amazon Managed Grafana workspace](#)
- [Exported metrics reference](#)
- [Amazon SageMaker HyperPod Slurm metrics](#)

### Complete prerequisites for SageMaker HyperPod cluster observability

Before proceeding with the steps to [the section called “Install metrics exporter packages on your HyperPod cluster”](#), ensure that the following prerequisites are met.

#### Enable IAM Identity Center

To enable observability for your SageMaker HyperPod cluster, you must first enable IAM Identity Center. This is a prerequisite for deploying an AWS CloudFormation stack that sets up the Amazon Managed Grafana workspace and Amazon Managed Service for Prometheus. Both of these services also require the IAM Identity Center for authentication and authorization, ensuring secure user access and management of the monitoring infrastructure.

For detailed guidance on enabling IAM Identity Center, see the [Enabling IAM Identity Center](#) section in the *AWS IAM Identity Center User Guide*.

After successfully enabling IAM Identity Center, set up a user account that will serve as the administrative user throughout the following configuration procedures.

#### Create and deploy an AWS CloudFormation stack for SageMaker HyperPod observability

Create and deploy a CloudFormation stack for SageMaker HyperPod observability to monitor HyperPod cluster metrics in real time using Amazon Managed Service for Prometheus and Amazon Managed Grafana. To deploy the stack, note that you also should enable your [IAM Identity Center](#) beforehand.

Use the sample CloudFormation script [cluster-observability.yaml](#) that helps you set up Amazon VPC subnets, Amazon FSx for Lustre file systems, Amazon S3 buckets, and IAM roles required to create a HyperPod cluster observability stack.

## Install metrics exporter packages on your HyperPod cluster

In the [base configuration lifecycle scripts](#) that the SageMaker HyperPod team provides also includes installation of various metric exporter packages. To activate the installation step, the only thing you need to do is to set the parameter `enable_observability=True` in the [config.py](#) file. The lifecycle scripts are designed to bootstrap your cluster with the following open-source metric exporter packages.

Name	Script deployment target node	Exporter description
<a href="#">Slurm exporter for Prometheus</a>	Head (controller) node	Exports Slurm Accounting metrics.
<a href="#">Elastic Fabric Adapter (EFA) node exporter</a>	Compute node	Exports metrics from cluster nodes and EFA. The package is a fork of the <a href="#">Prometheus node exporter</a> .
<a href="#">NVIDIA Data Center GPU Management (DCGM) exporter</a>	Compute node	Exports NVIDIA DCGM metrics about health and performance of NVIDIA GPUs.

With `enable_observability=True` in the [config.py](#) file, the following installation step is activated in the [lifecycle\\_script.py](#) script.

```
# Install metric exporting software and Prometheus for observability
if Config.enable_observability:
    if node_type == SlurmNodeType.COMPUTE_NODE:
        ExecuteBashScript("./utils/install_docker.sh").run()
        ExecuteBashScript("./utils/install_dcgm_exporter.sh").run()
        ExecuteBashScript("./utils/install_efa_node_exporter.sh").run()

    if node_type == SlurmNodeType.HEAD_NODE:
        wait_for_scontrol()
        ExecuteBashScript("./utils/install_docker.sh").run()
        ExecuteBashScript("./utils/install_slurm_exporter.sh").run()
        ExecuteBashScript("./utils/install_prometheus.sh").run()
```

On the compute nodes, the script installs the NVIDIA Data Center GPU Management (DCGM) exporter and the Elastic Fabric Adapter (EFA) node exporter. The DCGM exporter is an exporter for Prometheus that collects metrics from NVIDIA GPUs, enabling monitoring of GPU usage, performance, and health. The EFA node exporter, on the other hand, gathers metrics related to the EFA network interface, which is essential for low-latency and high-bandwidth communication in HPC clusters.

On the head node, the script installs the Slurm exporter for Prometheus and the [Prometheus open-source software](#). The Slurm exporter provides Prometheus with metrics related to Slurm jobs, partitions, and node states.

Note that the lifecycle scripts are designed to install all the exporter packages as docker containers, so the Docker package also should be installed on both the head and compute nodes. The scripts for these components are conveniently provided in the [utils](#) folder of the *Awsome Distributed Training GitHub repository*.

After you have successfully set up your HyperPod cluster installed with the exporter packages, proceed to the next topic to finish setting up Amazon Managed Service for Prometheus and Amazon Managed Grafana.

## Validate Prometheus setup on the head node of a HyperPod cluster

After you have successfully set up your HyperPod cluster installed with the exporter packages, check if Prometheus is properly set up on the head node of your HyperPod cluster.

1. Connect to the head node of your cluster. For instructions on accessing a node, see [the section called “Access your SageMaker HyperPod cluster nodes”](#).
2. Run the following command to verify the Prometheus config and service file created by the lifecycle script `install_prometheus.sh` is running on the controller node. The output should show the Active status as **active (running)**.

```
$ sudo systemctl status prometheus
● prometheus service - Prometheus Exporter
  Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; preset:disabled)
  Active: active (running) since DAY YYYY-MM-DD HH:MM:SS UTC; Ss ago
    Main PID: 12345 (prometheus)
       Tasks: 7 (limit: 9281)
      Memory: 35M
        CPU: 234ms
       CGroup: /system.slice/prometheus.service
```

```
-12345 /usr/bin/prometheus--config.file=/etc/prometheus/prometheus.yml
```

3. Validate the Prometheus configuration file as follows. The output must be similar to the following, with three exporter configured with the right compute node IP addresses.

```
$ cat /etc/prometheus/prometheus.yml
global:
  scrape_interval: 15s
  evaluation_interval: 15s
  scrape_timeout: 15s

scrape_configs:
  - job_name: 'slurm_exporter'
    static_configs:
      - targets:
          - 'localhost:8080'
  - job_name: 'dcgm_exporter'
    static_configs:
      - targets:
          - '<ComputeNodeIP>:9400'
          - '<ComputeNodeIP>:9400'
  - job_name: 'efa_node_exporter'
    static_configs:
      - targets:
          - '<ComputeNodeIP>:9100'
          - '<ComputeNodeIP>:9100'

remote_write:
  - url: <AMPReoteWriteURL>
    queue_config:
      max_samples_per_send: 1000
      max_shards: 200
      capacity: 2500
    sigv4:
      region: <Region>
```

4. To test if Prometheus is exporting Slurm, DCGM, and EFA metrics properly, run the following curl command for Prometheus on port :9090 on the head node.

```
$ curl -s http://localhost:9090/metrics | grep -E 'slurm|dcgm|efa'
```

With the metrics exported to Amazon Managed Service for Prometheus Workspace through the Prometheus remote write configuration from the controller node, you can proceed to the next topic to set up Amazon Managed Grafana dashboards to display the metrics.

## Set up an Amazon Managed Grafana workspace

Create a new Amazon Managed Grafana workspace or update an existing Amazon Managed Grafana workspace with Amazon Managed Service for Prometheus as the data source.

### Topics

- [Create a Grafana workspace and set Amazon Managed Service for Prometheus as a data source](#)
- [Open the Grafana workspace and finish setting up the data source](#)
- [Import open-source Grafana dashboards](#)

### Create a Grafana workspace and set Amazon Managed Service for Prometheus as a data source

To visualize metrics from Amazon Managed Service for Prometheus, create an Amazon Managed Grafana workspace and set it up to use Amazon Managed Service for Prometheus as a data source.

1. To create a Grafana workspace, follow the instructions at [Creating a workspace](#) in the *Amazon Managed Service for Prometheus User Guide*.
  - a. In Step 13, select Amazon Managed Service for Prometheus as the data source.
  - b. In Step 17, you can add the admin user and also other users in your IAM Identity Center.

For more information, see also the following resources.

- [Set up Amazon Managed Grafana for use with Amazon Managed Service for Prometheus](#) in the *Amazon Managed Service for Prometheus User Guide*
- [Use AWS data source configuration to add Amazon Managed Service for Prometheus as a data source](#) in the *Amazon Managed Grafana User Guide*

### Open the Grafana workspace and finish setting up the data source

After you have successfully created or updated an Amazon Managed Grafana workspace, select the workspace URL to open the workspace. This prompts you to enter a user name and the password

of the user that you have set up in IAM Identity Center. You should log in using the admin user to finish setting up the workspace.

1. In the workspace **Home** page, choose **Apps**, **AWS Data Sources**, and **Data sources**.
2. In the **Data sources** page, and choose the **Data sources** tab.
3. For **Service**, choose Amazon Managed Service for Prometheus.
4. In the **Browse and provision data sources** section, choose the AWS region where you provisioned an Amazon Managed Service for Prometheus workspace.
5. From the list of data sources in the selected Region, choose the one for Amazon Managed Service for Prometheus. Make sure that you check the resource ID and the resource alias of the Amazon Managed Service for Prometheus workspace that you have set up for HyperPod observability stack.

## Import open-source Grafana dashboards

After you've successfully set up your Amazon Managed Grafana workspace with Amazon Managed Service for Prometheus as the data source, you'll start collecting metrics to Prometheus, and then should start seeing the various dashboards showing charts, information, and more. The Grafana open source software provides various dashboards, and you can import them into Amazon Managed Grafana.

### To import open-source Grafana dashboards to Amazon Managed Grafana

1. In the **Home** page of your Amazon Managed Grafana workspace, choose **Dashboards**.
2. Choose the drop down menu button with the UI text **New**, and select **Import**.
3. Paste the URL to the [Slurm Dashboard](#).

`https://grafana.com/grafana/dashboards/4323-slurm-dashboard/`

4. Select **Load**.
5. Repeat the previous steps to import the following dashboards.

- a. [Node Exporter Full Dashboard](#)

`https://grafana.com/grafana/dashboards/1860-node-exporter-full/`

- b. [NVIDIA DCGM Exporter Dashboard](#)

<https://grafana.com/grafana/dashboards/12239-nvidia-dcm-exporter-dashboard/>

c. [EFA Metrics Dashboard](#)

<https://grafana.com/grafana/dashboards/20579-efa-metrics-dev/>

d. [FSx for Lustre Metrics Dashboard](#)

<https://grafana.com/grafana/dashboards/20906-fsx-lustre/>

## Exported metrics reference

The following sections present comprehensive lists of metrics exported from SageMaker HyperPod to Amazon Managed Service for Prometheus upon the successful configuration of the AWS CloudFormation stack for SageMaker HyperPod observability. You can start monitoring these metrics visualized in the Amazon Managed Grafana dashboards.

### Slurm exporter dashboard

Provides visualized information of Slurm clusters on SageMaker HyperPod.

#### Types of metrics

- **Cluster Overview:** Displaying the total number of nodes, jobs, and their states.
- **Job Metrics:** Visualizing job counts and states over time.
- **Node Metrics:** Showing node states, allocation, and available resources.
- **Partition Metrics:** Monitoring partition-specific metrics such as CPU, memory, and GPU utilization.
- **Job Efficiency:** Calculating job efficiency based on resources utilized.

#### List of metrics

Metric name	Description
slurm_job_count	Total number of jobs in the Slurm cluster

Metric name	Description
slurm_job_state_count	Count of jobs in each state (e.g., running, pending, completed)
slurm_node_count	Total number of nodes in the Slurm cluster
slurm_node_state_count	Count of nodes in each state (e.g., idle, alloc, mix)
slurm_partition_node_count	Count of nodes in each partition
slurm_partition_job_count	Count of jobs in each partition
slurm_partition_alloc_cpus	Total number of allocated CPUs in each partition
slurm_partition_free_cpus	Total number of available CPUs in each partition
slurm_partition_alloc_memory	Total allocated memory in each partition
slurm_partition_free_memory	Total available memory in each partition
slurm_partition_alloc_gpus	Total allocated GPUs in each partition
slurm_partition_free_gpus	Total available GPUs in each partition

## Node exporter dashboard

Provides visualized information of system metrics collected by the [Prometheus node exporter](#) from the HyperPod cluster nodes.

### Types of metrics

- **System overview:** Displaying CPU load averages and memory usage.
- **Memory metrics:** Visualizing memory utilization including total memory, free memory, and swap space.
- **Disk usage:** Monitoring disk space utilization and availability.

- **Network traffic:** Showing network bytes received and transmitted over time.
- **File system metrics:** Analyzing file system usage and availability.
- **Disk I/O metrics:** Visualizing disk read and write activity.

## List of metrics

For a complete list of metrics exported, see the [Node exporter](#) and [procfs](#) GitHub repositories. The following table shows a subset of the metrics that provides insights into system resource utilization such as CPU load, memory usage, disk space, and network activity.

Metric name	Description
node_load1	1-minute load average
node_load5	5-minute load average
node_load15	15-minute load average
node_memory_MemTotal	Total system memory
node_memory_MemFree	Free system memory
node_memory_MemAvailable	Available memory for allocation to processes
node_memory_Buffers	Memory used by the kernel for buffering
node_memory_Cached	Memory used by the kernel for caching file system data
node_memory_SwapTotal	Total swap space available
node_memory_SwapFree	Free swap space
node_memory_SwapCached	Memory that once was swapped out, is swapped back in but still in swap
node_filesystem_avail_bytes	Available disk space in bytes
node_filesystem_size_bytes	Total disk space in bytes

Metric name	Description
node_filesystem_free_bytes	Free disk space in bytes
node_network_receive_bytes	Network bytes received
node_network_transmit_bytes	Network bytes transmitted
node_disk_read_bytes	Disk bytes read
node_disk_written_bytes	Disk bytes written

## NVIDIA DCGM exporter dashboard

Provides visualized information of NVIDIA GPU metrics collected by the [NVIDIA DCGM exporter](#).

### Types of metrics

- GPU Overview:** Displaying GPU utilization, temperatures, power usage, and memory usage.
- Temperature Metrics:** Visualizing GPU temperatures over time.
- Power Usage:** Monitoring GPU power draw and power usage trends.
- Memory Utilization:** Analyzing GPU memory usage including used, free, and total memory.
- Fan Speed:** Showing GPU fan speeds and variations.
- ECC Errors:** Tracking GPU memory ECC errors and pending errors.

### List of metrics

The following table shows a list of the metrics that provides insights into the NVIDIA GPU health and performance, including clock frequencies, temperatures, power usage, memory utilization, fan speeds, and error metrics.

Metric name	Description
DCGM_FI_DEV_SM_CLOCK	SM clock frequency (in MHz)
DCGM_FI_DEV_MEM_CLOCK	Memory clock frequency (in MHz)
DCGM_FI_DEV_MEMORY_TEMP	Memory temperature (in C)

Metric name	Description
DCGM_FI_DEV_GPU_TEMP	GPU temperature (in C)
DCGM_FI_DEV_POWER_USAGE	Power draw (in W)
DCGM_FI_DEV_TOTAL_ENERGY_CO NSUMPTION	Total energy consumption since boot (in mJ)
DCGM_FI_DEV_PCIE_REPLAY_COUNTER	Total number of PCIe retries
DCGM_FI_DEV_MEM_COPY_UTIL	Memory utilization (in %)
DCGM_FI_DEV_ENC_UTIL	Encoder utilization (in %)
DCGM_FI_DEV_DEC_UTIL	Decoder utilization (in %)
DCGM_FI_DEV_XID_ERRORS	Value of the last XID error encountered
DCGM_FI_DEV_FB_FREE	Frame buffer memory free (in MiB)
DCGM_FI_DEV_FB_USED	Frame buffer memory used (in MiB)
DCGM_FI_DEV_NVLINK_BANDWIDT H_TOTAL	Total number of NVLink bandwidth counters for all lanes
DCGM_FI_DEV_VGPU_LICENSE_STATUS	vGPU License status
DCGM_FI_DEV_UNCORRECTABLE_R EMAPPED_ROWS	Number of remapped rows for uncorrectable errors
DCGM_FI_DEV_CORRECTABLE_R EMAPPED_ROWS	Number of remapped rows for correctable errors
DCGM_FI_DEV_ROW_REMAP_FAILURE	Whether remapping of rows has failed

## EFA metrics dashboard

Provides visualized information of the metrics from [Amazon Elastic Fabric Adapter \(EFA\)](#) equipped on P instances collected by the [EFA node exporter](#).

## Types of metrics

- **EFA error metrics:** Visualizing errors such as allocation errors, command errors, and memory map errors.
- **EFA network traffic:** Monitoring received and transmitted bytes, packets, and work requests.
- **EFA RDMA performance:** Analyzing RDMA read and write operations, including bytes transferred and error rates.
- **EFA port lifespan:** Displaying the lifespan of EFA ports over time.
- **EFA keep-alive packets:** Tracking the number of keep-alive packets received.

## List of metrics

The following table shows a list of the metrics that provides insights into various aspects of EFA operation, including errors, completed commands, network traffic, and resource utilization.

Metric name	Description
node_amazonefa_info	Non-numeric data from /sys/class/infinib and/, value is always 1.
node_amazonefa_lifespan	Lifespan of the port
node_amazonefa_rdma_read_bytes	Number of bytes read with RDMA
node_amazonefa_rdma_read_re_sp_bytes	Number of read response bytes with RDMA
node_amazonefa_rdma_read_wr_err	Number of read write errors with RDMA
node_amazonefa_rdma_read_wrs	Number of read rs with RDMA
node_amazonefa_rdma_write_bytes	Number of bytes written with RDMA
node_amazonefa_rdma_write_rcv_bytes	Number of bytes written and received with RDMA
node_amazonefa_rdma_write_wr_err	Number of bytes written with error RDMA
node_amazonefa_rdma_write_wrs	Number of bytes written wrs RDMA

Metric name	Description
node_amazonefa_recv_bytes	Number of bytes received
node_amazonefa_recv_wrs	Number of bytes received wrs
node_amazonefa_rx_bytes	Number of bytes received
node_amazonefa_rx_drops	Number of packets dropped
node_amazonefa_rx_pkts	Number of packets received
node_amazonefa_send_bytes	Number of bytes sent
node_amazonefa_send_wrs	Number of wrs sent
node_amazonefa_tx_bytes	Number of bytes transmitted
node_amazonefa_tx_pkts	Number of packets transmitted

## FSx for Lustre metrics dashboard

Provides visualized information of the [metrics from Amazon FSx for Lustre](#) file system collected by [Amazon CloudWatch](#).

### Note

The Grafana FSx for Lustre dashboard utilizes Amazon CloudWatch as its data source, which differs from the other dashboards that you have configured to use Amazon Managed Service for Prometheus. To ensure accurate monitoring and visualization of metrics related to your FSx for Lustre file system, configure the FSx for Lustre dashboard to use Amazon CloudWatch as the data source, specifying the same AWS Region where your FSx for Lustre file system is deployed.

## Types of metrics

- **DataReadBytes:** The number of bytes for file system read operations.
- **DataWriteBytes:** The number of bytes for file system write operations.

- **DataReadOperations:** The number of read operations.
- **DataWriteOperations:** The number of write operations.
- **MetadataOperations:** The number of meta data operations.
- **FreeDataStorageCapacity:** The amount of available storage capacity.

## Amazon SageMaker HyperPod Slurm metrics

Amazon SageMaker HyperPod provides a set of Amazon CloudWatch metrics that you can use to monitor the health and performance of your HyperPod clusters. These metrics are collected from the Slurm workload manager running on your HyperPod clusters and are available in the /aws/sagemaker/Clusters CloudWatch namespace.

### Cluster level metrics

The following cluster-level metrics are available for HyperPod. These metrics use the ClusterId dimension to identify the specific HyperPod cluster.

CloudWatch metric name	Notes	Amazon EKS Container Insights metric name
cluster_node_count	Total number of nodes in the cluster	cluster_node_count
cluster_idle_node_count	Number of idle nodes in the cluster	N/A
cluster_failed_node_count	Number of failed nodes in the cluster	cluster_failed_node_count
cluster_cpu_count	Total CPU cores in the cluster	node_cpu_limit
cluster_idle_cpu_count	Number of idle CPU cores in the cluster	N/A
cluster_gpu_count	Total GPUs in the cluster	node_gpu_limit
cluster_idle_gpu_count	Number of idle GPUs in the cluster	N/A

CloudWatch metric name	Notes	Amazon EKS Container Insights metric name
cluster_running_task_count	Number of running Slurm jobs in the cluster	N/A
cluster_pending_task_count	Number of pending Slurm jobs in the cluster	N/A
cluster_preempted_task_count	Number of preempted Slurm jobs in the cluster	N/A
cluster_avg_task_wait_time	Average wait time for Slurm jobs in the cluster	N/A
cluster_max_task_wait_time	Maximum wait time for Slurm jobs in the cluster	N/A

## Instance level metrics

The following instance-level metrics are available for HyperPod. These metrics also use the `ClusterId` dimension to identify the specific HyperPod cluster.

CloudWatch metric name	Notes	Amazon EKS Container Insights metric name
node_gpu_utilization	Average GPU utilization across all instances	node_gpu_utilization
node_gpu_memory_utilization	Average GPU memory utilization across all instances	node_gpu_memory_utilization
node_cpu_utilization	Average CPU utilization across all instances	node_cpu_utilization
node_memory_utilization	Average memory utilization across all instances	node_memory_utilization

## SageMaker HyperPod cluster resiliency

SageMaker HyperPod provides the following cluster resiliency features.

### Topics

- [Cluster health check](#)
- [Auto-resume](#)
- [How to replace a faulty node not being auto-resumed by HyperPod](#)

### Cluster health check

This section describes the set of health checks that SageMaker HyperPod uses to regularly monitor cluster instance health for issues with devices such as accelerators (GPU and Trainium cores) and networking (EFA).

Category	Utility name	Instance type compatibility	Description
Accelerator	DCGM policies	GPU	Each instance in the cluster continuously monitors all GPU-related policies including XID errors with <a href="#">NVIDIA DCGM</a> .
Accelerator	NVIDIA SMI	GPU	<a href="#">nvidia-smi</a> utility is a well-known CLI to manage and monitor GPUs. The built-in health checker parses the output from nvidia-smi to determine the health of the instance.
Accelerator	Neuron sysfs	Trainium	For Trainium-powered instances

Category	Utility name	Instance type compatibility	Description
			, the health of the Neuron devices is determined by reading counters from <a href="#">Neuron sysfs</a> propagated directly by the Neuron driver.
Network	EFA	GPU and Trainium	To aid in the diagnostic of Elastic Fabric Adaptor (EFA) devices, the EFA health checker runs a series of connectivity tests using all available EFA cards within the instance.
Stress	<a href="#">DCGM diagnostics</a> level 2	GPU	<a href="#">DCGM diagnostics</a> level 2 is used to exercise the GPUs in the system and put them under pressure to get a thorough insight of the health.
Stress	CPU stress	GPU and Trainium	CPU health is determined using the <a href="#">Linux stress</a> tool, which runs multiple threads to achieve 100% CPU utilization and perform I/O operations.

## Auto-resume

This section describes how to run a training job with the SageMaker HyperPod auto-resume functionality, which provides a zero-touch resiliency infrastructure to automatically recover a training job from the last saved checkpoint in the event of a hardware failure for clusters with more than 16 nodes.

With the auto-resume functionality, if a job fails due to a hardware failure or any transient issues in-between training, SageMaker HyperPod auto-resume starts the node replacement workflow and restarts the job after the faulty nodes are replaced.

### Note

When [Generic Resources \(GRES\)](#) are attached to a Slurm node, Slurm typically doesn't permit changes in the node allocation, such as replacing nodes, and thus doesn't allow to resume a failed job. Unless explicitly forbidden, the HyperPod auto-resume functionality automatically re-queues any faulty job associated with the GRES-enabled nodes. This process involves stopping the job, placing it back into the job queue, and then restarting the job from the beginning.

## Using the SageMaker HyperPod auto-resume functionality with Slurm

When you use SageMaker HyperPod auto-resume with Slurm, you should run the job inside an exclusive allocation acquired either by using `salloc` or `sbatch`. In any case, you need to modify the entrypoint script to make sure that all setup steps run in a single `srun` command when resuming the job. Through the entrypoint script, it is important to set up the environment on the replaced node to be consistent with the environment that the job step was running before it was stopped. The following procedure shows how to prepare an entrypoint script to keep the environment consistent and run it as a single `srun` command.

### Tip

If you use `sbatch`, you can keep the batch script simple by creating a separate script for setting up the environment and using a single `srun` command.

1. Create a script using the following code example and save it as `train_auto_resume.sh`. This script deploys training environment setups assuming that there is no manual configuration

previously made to the replaced node. This ensures that the environment is node-agnostic, so that when a node is replaced, the same environment is provisioned on the node before resuming the job.

### Note

The following code example shows how to discover the Slurm node list associated with the job. Do not use the `$SLURM_JOB_NODELIST` environment variable provided by Slurm, because its value might be outdated after SageMaker HyperPod auto-resumes the job. The following code example shows how to define a new `NODE_LIST` variable to replace `SLURM_JOB_NODELIST`, and then set up the `MASTER_NODE` and `MASTER_ADDR` variables off of the `NODE_LIST` variable.

```
#!/bin/bash

# Filename: train_auto_resume.sh
# Sample containerized script to launch a training job with a single srun which can
# be auto-resumed.

# Place your training environment setup here.
# Example: Install conda, docker, activate virtual env, etc.

# Get the list of nodes for a given job
NODE_LIST=$(scontrol show jobid=$SLURM_JOBID | \ # Show details of the SLURM job
            awk -F= '/NodeList=/\{print $2\}' | \ # Extract NodeList field
            grep -v Exc)                         # Exclude nodes marked as excluded

# Determine the master node from the node list
MASTER_NODE=$(scontrol show hostname $NODE_LIST | \ # Convert node list to hostnames
              head -n 1)                           # Select the first hostname as
                                             # master node

# Get the master node address
MASTER_ADDR=$(scontrol show node=$MASTER_NODE | \ # Show node information
              awk -F= '/NodeAddr=/\{print $2\}' | \ # Extract NodeAddr
              awk '{print $1}')                  # Print the first part of NodeAddr

# Torchrun command to launch the training job
torchrun_cmd="torchrun --nnodes=$SLURM_NNODES \\"
```

```
--nproc_per_node=1 \
--node_rank=$SLURM_NODE \
--master-addr=$MASTER_ADDR \
--master_port=1234 \
<your_training_script.py>"'

# Execute the torchrun command in the 'pytorch' Conda environment,
# streaming output live
/opt/conda/bin/conda run --live-stream -n pytorch $torchrun_cmd
```

### Tip

You can use the preceding script to add more commands for installing any additional dependencies for your job. However, we recommend that you keep the dependency installation scripts to the [set of lifecycle scripts](#) that are used during cluster creation. If you use a virtual environment hosted on a shared directory, you can also utilize this script to activate the virtual environment.

2. Launch the job with SageMaker HyperPod auto-resume enabled by adding the flag `--auto-resume=1` to indicate that the `srun` command should be automatically retried in case of hardware failure.

### Note

If you have set up a resource allocation using `sbatch` or `salloc`, you can run multiple `srun` commands within the allocation. In the event of a failure, the SageMaker HyperPod auto-resume functionality only operates in the current [job step](#) of the `srun` command with the flag `--auto-resume=1`. In other words, activating auto-resume in an `srun` command doesn't apply to other `srun` commands launched within a resource allocation session.

The following are `srun` command examples with `auto-resume` enabled.

## Using `sbatch`

Because most of the logic for setting up the environment is already in `train_auto_resume.sh`, the batch script should be simple and similar to the following code example. Assume that the following batch script is saved as `batch.sh`.

```
#!/bin/bash
#SBATCH --nodes 2
#SBATCH --exclusive
srun --auto-resume=1 train_auto_resume.sh
```

Run the preceding batch script using the following command.

```
sbatch batch.sh
```

## Using salloc

Start by acquiring an exclusive allocation, and run the `srun` command with the `--auto-resume` flag and the entrypoint script.

```
salloc -N 2 --exclusive
srun --auto-resume=1 train_auto_resume.sh
```

## How to replace a faulty node not being auto-resumed by HyperPod

The HyperPod auto-resume functionality monitors if the state of your Slurm nodes turns to `fail` or `down`. You can check the state of Slurm nodes by running `sinfo`.

If you have a node stuck with an issue but not being fixed by the HyperPod auto-resume functionality, we recommend you to run the following command to change the state of the node to `fail`.

```
scontrol update node=<ip-ipv4> state=fail reason="Action:Replace"
```

In the preceding command example, replace `<ip-ipv4>` with the Slurm node name (host name) of the faulty instance you want to replace.

After running this command, the node should go into the `fail` state, waits for the currently running jobs to finish, is replaced with a healthy instance, and is recovered with the same host name. This process takes time depending on the available instances in your Availability Zone and the time it takes to run your lifecycle scripts. During the update and replacement processes, avoid changing the state of the node manually again or restarting the Slurm controller; doing so can lead

to a replacement failure. If the node does not get recovered nor turn to the idle state after a long time, contact [AWS Support](#).

If the faulty node is continuously stuck in the fail state, the last resort you might try is to manually force change the node state to down. This requires administrator privileges (sudo permissions).

### Warning

Proceed carefully before you run the following command as it forces kill all jobs, and you might lose all unsaved work.

```
scontrol update node=<ip-ipv4> state=down reason="Action:Replace"
```

## SageMaker HyperPod cluster management

The following topics discuss logging and managing SageMaker HyperPod clusters.

### Logging SageMaker HyperPod events

All events and logs from SageMaker HyperPod are saved to Amazon CloudWatch under the log group name /aws/sagemaker/Clusters/[ClusterName]/[ClusterID]. Every call to the CreateCluster API creates a new log group. The following list contains all of the available log streams collected in each log group.

Log Group Name	Log Stream Name
/aws/sagemaker/Clusters/[ClusterName]/[ClusterID]	LifecycleConfig/[instance-group-name]/[instance-id]

### Logging SageMaker HyperPod at instance level

You can access the LifecycleScript logs published to CloudWatch during cluster instance configuration. Every instance within the created cluster generates a separate log stream, distinguishable by the LifecycleConfig/[instance-group-name]/[instance-id] format.

All logs that are written to /var/log/provision/provisioning.log are uploaded to the preceding CloudWatch stream. Sample LifecycleScripts at

[1.architectures/5.sagemaker\\_hyperpods/LifecycleScripts/base-config](#) redirect their `stdout` and `stderr` to this location. If you are using your custom scripts, write your logs to the `/var/log/provision/provisioning.log` location for them to be available in CloudWatch.

## Tagging resources

AWS Tagging system helps manage, identify, organize, search for, and filter resources. SageMaker HyperPod supports tagging, so you can manage the clusters as an AWS resource. During cluster creation or editing an existing cluster, you can add or edit tags for the cluster. To learn more about tagging in general, see [Tagging your AWS resources](#).

### Using the SageMaker HyperPod console UI

When you are [creating a new cluster](#) and [editing a cluster](#), you can add, remove, or edit tags.

### Using the SageMaker HyperPod APIs

When you write a [CreateCluster](#) or [UpdateCluster](#) API request file in JSON format, edit the Tags section.

### Using the AWS CLI tagging commands for SageMaker AI

#### To tag a cluster

Use [aws sagemaker add-tags](#) as follows.

```
aws sagemaker add-tags --resource-arn cluster_ARN --tags Key=string,Value=string
```

#### To untag a cluster

Use [aws sagemaker delete-tags](#) as follows.

```
aws sagemaker delete-tags --resource-arn cluster_ARN --tag-keys "tag_key"
```

#### To list tags for a resource

Use [aws sagemaker list-tags](#) as follows.

```
aws sagemaker list-tags --resource-arn cluster_ARN
```

## SageMaker HyperPod FAQ

Use the following frequently asked questions to troubleshoot problems with using SageMaker HyperPod.

### Q. Why can I not find log groups of my SageMaker HyperPod cluster in Amazon CloudWatch?

By default, agent logs and instance start-up logs are sent to the HyperPod platform account's CloudWatch. In case of user lifecycle scripts, lifecycle configuration logs are sent to your account's CloudWatch.

If you use the [sample lifecycle scripts](#) provided by the HyperPod service team, you can expect to find the lifecycle configuration logs written to `/var/log/provision/provisioning.log`, and you wouldn't encounter this problem.

However, if you use custom paths for collecting logs from lifecycle provisioning and can't find the log groups appearing in your account's CloudWatch, it might be due to mismatches in the log file paths specified in your lifecycle scripts and what the CloudWatch agent running on the HyperPod cluster instances looks for. In this case, it means that you need to properly set up your lifecycle scripts to send logs to the CloudWatch agent, and also set up the CloudWatch agent configuration accordingly. To resolve the problem, choose one of the following options.

- **Option 1:** Update your lifecycle scripts to write logs to `/var/log/provision/provisioning.log`.
- **Option 2:** Update the CloudWatch agent to look for your custom paths for logging lifecycle provisioning.
  1. Each HyperPod cluster instance contains a CloudWatch agent configuration file in JSON format at `/opt/aws/amazon-cloudwatch-agent/sagemaker_cwagent_config.json`. In the configuration file, find the field name `logs.logs_collected.files.collect_list.file_path`. With the default setup by HyperPod, the key-value pair should be "`file_path": "/var/log/provision/provisioning.log"` as documented at [the section called "Logging SageMaker HyperPod at instance level"](#). The following code snippet shows how the JSON file looks with the HyperPod default configuration.

```
"logs": {  
    "logs_collected": {  
        "files": {
```

```
        "collect_list": [
            {
                "file_path": "/var/log/provision/provisioning.log",
                "log_group_name": "/aws/sagemaker/Clusters/[ClusterName]/
[ClusterID]",
                "log_stream_name": "LifecycleConfig/[InstanceGroupName]/
{instance_id}",
                "retention_in_days": -1
            }
        ]
    },
    "force_flush_interval": 3
}
```

2. Replace the value for the "file\_path" field name with the custom path you use in your lifecycle scripts. For example, if you have set up your lifecycle scripts to write to /var/log/custom-provision/custom-provisioning.log, update the value to match with it as follows.

```
"file_path": "/var/log/custom-provision/custom-provisioning.log"
```

3. Restart the CloudWatch agent with the configuration file to finish applying the custom path. For example, the following CloudWatch command shows how to restart the CloudWatch agent with the CloudWatch agent configuration file from step 1. For more information, see also [Troubleshooting the CloudWatch agent](#).

```
sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl \
-a fetch-config -m ec2 -s -c \
file:/opt/aws/amazon-cloudwatch-agent/sagemaker_cwagent_config.json
```

## Q. What particular configurations does HyperPod manage in Slurm configuration files such as `slurm.conf` and `gres.conf`?

When you create a Slurm cluster on HyperPod, the HyperPod agent sets up the [slurm.conf](#) and [gres.conf](#) files at /opt/slurm/etc/ to manage the Slurm cluster based on your HyperPod cluster creation request and lifecycle scripts. The following list shows what specific parameters the HyperPod agent handles and overwrites.

## ⚠️ Important

We strongly recommend that you DON'T change these parameters managed by HyperPod.

- In [`slurm.conf`](#), HyperPod sets up the following basic parameters: ClusterName, SlurmctldHost, PartitionName, and NodeName.

Also, to enable the [the section called "Auto-resume"](#) functionality, HyperPod requires the TaskPlugin and SchedulerParameters parameters set as follows. The HyperPod agent sets up these two parameters with the required values by default.

```
TaskPlugin=task/none  
SchedulerParameters=permit_job_expansion
```

- In [`gres.conf`](#), HyperPod manages NodeName for GPU nodes.

## Q. How do I run Docker on Slurm nodes on HyperPod?

To help you run Docker on your Slurm nodes running on HyperPod, the HyperPod service team provides setup scripts that you can include as part of the lifecycle configuration for cluster creation. To learn more, see [the section called "Start with base lifecycle scripts provided by HyperPod"](#) and [the section called "Run Docker containers on a Slurm compute node on HyperPod"](#).

## Q. Why does my parallel training job fail when I use NVIDIA Collective Communications Library (NCCL) on SageMaker HyperPod platform on the Slurm framework?

By default, the Linux OS sets the #RemoveIPC=yes flag. Slurm and mpirun jobs that use NCCL generate inter-process communication (IPC) resources under non-root user sessions. These user sessions might log out during the job process.

When you run jobs with Slurm or mpirun, if systemd detects that the user isn't logged in, it cleans up the IPC resources. Slurm and mpirun jobs can run without the user being logged in, but that requires that you disable cleanup at the systemd level and set it up at the Slurm level instead. For more information, see [Systemd in the NCCL documentation](#).

To disable cleanup at the systemd level, complete the following steps.

1. Set the flag #RemoveIPC=no in the file `/etc/systemd/logind.conf` if you're running training jobs that use Slurm and NCCL.

2. By default, Slurm doesn't clean up shared resources. We recommend that you set up a Slurm epilog script to clean up shared resources. This cleanup is useful when you have a lot of shared resources and want to clean them up after training jobs. The following is an example script.

```
#!/bin/bash
: <<'SUMMARY'
Script: epilog.sh
```

Use this script with caution, as it can potentially delete unnecessary resources and cause issues if you don't use it correctly.

Note: You must save this script in a shared in a shared location that is accessible to all nodes in the cluster, such as /fsx volume.

Workers must be able to access the script to run the script after jobs.

#### SUMMARY

```
# Define the log directory and create it if it doesn't exist
LOG_DIR="/<PLACEHOLDER>/epilogue" #NOTE: Update PLACEHOLDER to be a shared value
path, such as /fsx/epilogue.
mkdir -p "$LOG_DIR"

# Name the log file using the Slurm job name and job ID
log_file="$LOG_DIR/epilogue-${SLURM_JOB_NAME}_${SLURM_JOB_ID}.log"

logging() {
    echo "[$(date)] $1" | tee -a "$log_file"
}

# Slurm epilogue script to clean up IPC resources
logging "Starting IPC cleanup for Job ${SLURM_JOB_ID}"

# Clean up shared memory segments by username
for seg in $(ipcs -m | awk -v owner="${SLURM_JOB_USER}" '$3 == owner {print $2}'); do
    if ipcrm -m "$seg"; then
        logging "Removed shared memory segment $seg"
    else
        logging "Failed to remove shared memory segment $seg"
    fi
done

# Clean up semaphores by username
for sem in $(ipcs -s | awk -v user="${SLURM_JOB_USER}" '$3 == user {print $2}'); do
```

```
if ipcrm -s "$sem"; then
    logging "Removed semaphore $sem"
else
    logging "Failed to remove semaphore $sem"
fi
done

# Clean up NCCL IPC
NCCL_IPC_PATH="/dev/shm/nccl-*"
for file in $NCCL_IPC_PATH; do
    if [ -e "$file" ]; then
        if rm "$file"; then
            logging "Removed NCCL IPC file $file"
        else
            logging "Failed to remove NCCL IPC file $file"
        fi
    fi
done
logging "IPC cleanup completed for Job $SLURM_JOB_ID"
exit 0
```

For more information about the Epilog parameter, see [Slurm documentation](#).

3. In the `slurm.conf` file from the controller node, add in a line to point to the epilog script you created.

```
Epilog="/path/to/epilog.sh" #For example: /fsx/epilogue/epilog.sh
```

4. Run the following commands to change permissions of the script and to make it executable.

```
chown slurm:slurm /path/to/epilog.sh
chmod +x /path/to/epilog.sh
```

5. To apply all of your changes, run `scontrol reconfigure`.

## Q. How do I use local NVMe store of P instances for launching Docker or Enroot containers with Slurm?

Because the default root volume of your head node usually is limited by 100GB EBS volume, you need to set up Docker and Enroot to use local NVMe instance store. To learn how to set up NVMe store and use it for launching Docker containers, see [the section called “Run Docker containers on a Slurm compute node on HyperPod”](#).

## Q. How to set up EFA security groups?

If you want to create a HyperPod cluster with EFA-enabled instances, make sure that you set up a security group to allow all inbound and outbound traffic to and from the security group itself. To learn more, see [Step 1: Prepare an EFA-enabled security group](#) in the *Amazon EC2 User Guide*.

## Q. How do I monitor my HyperPod cluster nodes? Is there any CloudWatch metrics exported from HyperPod?

To gain observability into the resource utilization of your HyperPod cluster, we recommend that you integrate the HyperPod cluster with Amazon Managed Grafana and Amazon Managed Service for Prometheus. With various open-source Grafana dashboards and exporter packages, you can export and visualize metrics related to the HyperPod cluster resources. To learn more about setting up SageMaker HyperPod with Amazon Managed Grafana and Amazon Managed Service for Prometheus, see [the section called “HyperPod cluster resources monitoring”](#). Note that SageMaker HyperPod currently doesn't support the exportation of system metrics to Amazon CloudWatch.

## Q. Can I add an additional storage to the HyperPod cluster nodes? The cluster instances have limited local instance store.

If the default instance storage is insufficient for your workload, you can configure additional storage per instance. Starting from the [release on June 20, 2024](#), you can add an additional Amazon Elastic Block Store (EBS) volume to each instance in your SageMaker HyperPod cluster. Note that this capability cannot be applied to existing instance groups of SageMaker HyperPod clusters created before June 20, 2024. You can utilize this capability by patching existing SageMaker HyperPod clusters created before June 20, 2024 and adding new instance groups to them. This capability is fully effective for any SageMaker HyperPod clusters created after June 20, 2024.

# Orchestrating SageMaker HyperPod clusters with Amazon EKS

SageMaker HyperPod is a SageMaker AI-managed service that enables large-scale training of foundation models on long-running and resilient compute clusters, integrating with Amazon EKS for orchestrating the HyperPod compute resources. You can run uninterrupted training jobs spanning weeks or months at scale using Amazon EKS clusters with HyperPod resiliency features that check for various hardware failures and automatically recover faulty nodes.

Key features for cluster admin users include the following.

- Provisioning resilient HyperPod clusters and attaching them to an EKS control plane

- Enabling dynamic capacity management, such as adding more nodes, updating software, and deleting clusters
- Enabling access to the cluster instances directly through `kubectl` or SSM/SSH
- Offering [resiliency capabilities](#), including basic health checks, deep health checks, a health-monitoring agent, and support for PyTorch job auto-resume
- Integrating with observability tools such as [Amazon CloudWatch Container Insights](#), [Amazon Managed Service for Prometheus](#), and [Amazon Managed Grafana](#)

For data scientist users, EKS support in HyperPod enables the following.

- Running containerized workloads for training foundation models on the HyperPod cluster
- Running inference on the EKS cluster, leveraging the integration between HyperPod and EKS
- Leveraging the job auto-resume capability for [Kubeflow PyTorch training \(PyTorchJob\)](#)

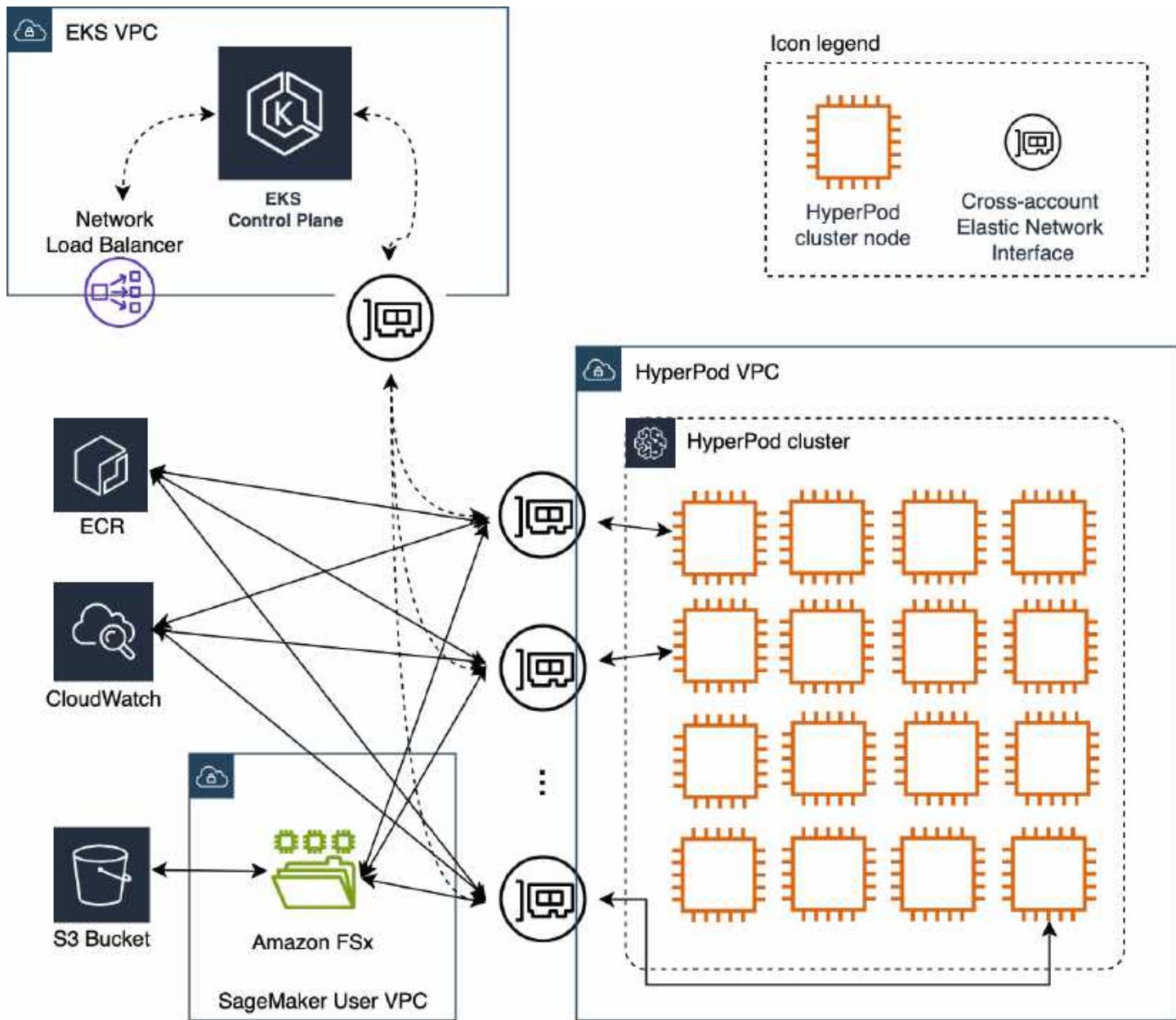
 **Note**

Amazon EKS enables user-managed orchestration of tasks and infrastructure on SageMaker HyperPod through the Amazon EKS Control Plane. Ensure that user access to the cluster through the Kubernetes API Server endpoint follows the principle of least-privilege, and that network egress from the HyperPod cluster is secured.

To learn more about securing access to the Amazon EKS API Server, see [Control network access to cluster API server endpoint](#).

To learn more about securing network access on HyperPod, see [Setting up SageMaker HyperPod with a custom Amazon VPC](#).

The high-level architecture of Amazon EKS support in HyperPod involves a 1-to-1 mapping between an EKS cluster (control plane) and a HyperPod cluster (worker nodes) within a VPC, as shown in the following diagram.



## Managing SageMaker HyperPod clusters orchestrated by Amazon EKS

This section provides guidance on managing SageMaker HyperPod through the SageMaker AI console UI or the AWS Command Line Interface (CLI). It explains how to perform various tasks related to SageMaker HyperPod, whether you prefer a visual interface or working with commands.

### Topics

- [Getting started with Amazon EKS support in SageMaker HyperPod](#)
- [Install packages on the Amazon EKS cluster using Helm](#)

- [Setting up Kubernetes role-based access control](#)
- [Managing SageMaker HyperPod clusters using the SageMaker HyperPod console UI](#)
- [Managing SageMaker HyperPod clusters using the AWS CLI](#)
- [Configure storage for SageMaker HyperPod clusters orchestrated by Amazon EKS](#)

## Getting started with Amazon EKS support in SageMaker HyperPod

In addition to the general [the section called “Prerequisites”](#) for SageMaker HyperPod, check the following requirements and considerations for orchestrating SageMaker HyperPod clusters using Amazon EKS.

### Requirements

#### Note

Before creating a HyperPod cluster, you need a running Amazon EKS cluster configured with VPC and installed using Helm.

- If using the SageMaker AI console, you can create an Amazon EKS cluster within the HyperPod cluster console page. For more information, see [the section called “Create a SageMaker HyperPod cluster”](#).
- If using AWS CLI, you should create an Amazon EKS cluster before creating a HyperPod cluster to associate with. For more information, see [Create an Amazon EKS cluster](#) in the Amazon EKS User Guide.

When provisioning your Amazon EKS cluster, consider the following:

#### 1. Kubernetes version support

- SageMaker HyperPod supports Kubernetes versions 1.28, 1.29, 1.30, and 1.31.

#### 2. Amazon EKS cluster authentication mode

- The authentication mode of an Amazon EKS cluster supported by SageMaker HyperPod are API and API\_AND\_CONFIG\_MAP.

#### 3. Networking

- SageMaker HyperPod requires the Amazon VPC Container Network Interface (CNI) plug-in version 1.18.3 or later.

**Note**

[AWS VPC CNI plugin for Kubernetes](#) is the only CNI supported by SageMaker HyperPod.

- The [type of the subnet](#) in your VPC must be private for HyperPod clusters.

#### 4. IAM roles

- Ensure the necessary IAM roles for HyperPod are set up as guided in the [the section called "IAM for HyperPod"](#) section.

#### 5. Amazon EKS cluster add-ons

- You can continue using the various add-ons provided by Amazon EKS such as [Kube-proxy](#), [CoreDNS](#), the [Amazon VPC Container Network Interface \(CNI\)](#) plugin, Amazon EKS pod identity, the GuardDuty agent, the Amazon FSx Container Storage Interface (CSI) driver, the Mountpoint for Amazon S3 CSI driver, the AWS Distro for OpenTelemetry, and the CloudWatch Observability agent.

### Considerations for configuring SageMaker HyperPod clusters with Amazon EKS

- You must use distinct IAM roles based on the type of your nodes. For HyperPod nodes, use a role based on [IAM role for SageMaker HyperPod](#). For Amazon EKS nodes, see [Amazon EKS node IAM role](#).
- You can't mount additional EBS volumes directly to Pods running on HyperPod cluster nodes. Instead, you need to utilize [InstanceStorageConfigs](#) to provision and mount additional EBS volumes to the HyperPod nodes. It's important to note that you can only attach additional EBS volumes to new instance groups while creating or updating a HyperPod cluster. Once you have configured instance groups with these additional EBS volumes, in your Amazon EKS Pod configuration file, you'll need to set the [local path](#) to /opt/sagemaker to properly mount the volumes to your Amazon EKS Pods.
- You can deploy the [Amazon EBS CSI \(Container Storage Interface\)](#) controller on HyperPod nodes. However, the Amazon EBS CSI node DaemonSet, which facilitates the mounting and unmounting of EBS volumes, can only run on non-HyperPod instances.
- If you use instance-type labels for defining scheduling constraints, ensure that you use the SageMaker AI ML instance types prefixed with m1... For example, for P5 instances, use m1.p5.48xlarge instead of p5.48xlarge.

## Considerations for configuring network for SageMaker HyperPod clusters with Amazon EKS

- Each HyperPod cluster instance supports one Elastic Network Interface (ENI). For the maximum number of Pods per instance type, refer to the following table.

Instance type	Max number of pods
ml.p4d.24xlarge	49
ml.p4de.24xlarge	49
ml.p5.48xlarge	49
ml.trn1.32xlarge	49
ml.trn1n.32xlarge	49
ml.g5.xlarge	14
ml.g5.2xlarge	14
ml.g5.4xlarge	29
ml.g5.8xlarge	29
ml.g5.12xlarge	49
ml.g5.16xlarge	29
ml.g5.24xlarge	49
ml.g5.48xlarge	49
ml.c5.large	9
ml.c5.xlarge	14
ml.c5.2xlarge	14
ml.c5.4xlarge	29
ml.c5.9xlarge	29

Instance type	Max number of pods
ml.c5.12xlarge	29
ml.c5.18xlarge	49
ml.c5.24xlarge	49
ml.c5n.large	9
ml.c5n.2xlarge	14
ml.c5n.4xlarge	29
ml.c5n.9xlarge	29
ml.c5n.18xlarge	49
ml.m5.large	9
ml.m5.xlarge	14
ml.m5.2xlarge	14
ml.m5.4xlarge	29
ml.m5.8xlarge	29
ml.m5.12xlarge	29
ml.m5.16xlarge	49
ml.m5.24xlarge	49
ml.t3.medium	5
ml.t3.large	11
ml.t3.xlarge	14
ml.t3.2xlarge	14

Instance type	Max number of pods
ml.g6.xlarge	14
ml.g6.2xlarge	14
ml.g6.4xlarge	29
ml.g6.8xlarge	29
ml.g6.12xlarge	29
ml.g6.16xlarge	49
ml.g6.24xlarge	49
ml.g6.48xlarge	49
ml.gr6.4xlarge	29
ml.gr6.8xlarge	29
ml.g6e.xlarge	14
ml.g6e.2xlarge	14
ml.g6e.4xlarge	29
ml.g6e.8xlarge	29
ml.g6e.12xlarge	29
ml.g6e.16xlarge	49
ml.g6e.24xlarge	49
ml.g6e.48xlarge	49
ml.p5e.48xlarge	49

- Only Pods with `hostNetwork = true` have access to the Amazon EC2 Instance Metadata Service (IMDS) by default. Use the Amazon EKS Pod identity or the [IAM roles for service accounts \(IRSA\)](#) to manage access to the AWS credentials for Pods.
- EKS-orchestrated HyperPod clusters support dual IP addressing modes, allowing configuration with IPv4 or IPv6 for IPv6 Amazon EKS clusters in IPv6-enabled VPC and subnet environments. For more information, see [the section called “Setting up SageMaker HyperPod with a custom Amazon VPC”](#).

## Considerations for using the HyperPod cluster resiliency features

- Node auto-replacement is not supported for CPU instances.
- The HyperPod health monitoring agent needs to be installed for node auto-recovery to work. The agent can be installed using Helm. For more information, see [the section called “Install packages on the Amazon EKS cluster using Helm”](#).
- The HyperPod deep health check and health monitoring agent supports GPU and Trn instances.
- SageMaker AI applies the following taint to nodes when they are undergoing deep health checks:

```
effect: NoSchedule  
key: sagemaker.amazonaws.com/node-health-status  
value: Unschedulable
```

### Note

You cannot add custom taints to nodes in instance groups with DeepHealthChecks turned on.

Once your Amazon EKS cluster is running, configure your cluster using the Helm package manager as instructed in [the section called “Install packages on the Amazon EKS cluster using Helm”](#) before creating your HyperPod cluster.

## Install packages on the Amazon EKS cluster using Helm

Before creating a SageMaker HyperPod cluster and attaching it to an Amazon EKS cluster, you should install packages using [Helm](#), a package manager for Kubernetes. Helm is an open-source tool for setting up a installation process for Kubernetes clusters. It enables the automation and

streamlining of dependency installations and simplifies various setups needed for preparing the Amazon EKS cluster as the orchestrator (control plane) for a SageMaker HyperPod cluster.

The SageMaker HyperPod service team provides a Helm chart package, which bundles key dependencies such as device/EFA plug-ins, plug-ins, [Kubeflow Training Operator](#), and associated permission configurations.

### **Important**

This helm installation step is a required step. Failure to configure your Amazon EKS cluster using the provided Helm chart may result in the SageMaker HyperPod cluster not functioning correctly or the creation process failing entirely. The aws-hyperpod namespace name cannot be modified.

1. [Install Helm](#) on your local machine.
2. Download the Helm charts provided by SageMaker HyperPod located at `helm_chart/HyperPodHelmChart` in the [SageMaker HyperPod CLI repository](#).

```
git clone https://github.com/aws/sagemaker-hyperpod-cli.git
cd sagemaker-hyperpod-cli/helm_chart
```

3. Update the dependencies of the Helm chart, preview the changes that will be made to your Kubernetes cluster, and install the Helm chart.

```
helm dependencies update HyperPodHelmChart
```

```
helm install hyperpod-dependencies HyperPodHelmChart --dry-run
```

```
helm install hyperpod-dependencies HyperPodHelmChart
```

In summary, the Helm installation sets up various components for your Amazon EKS cluster, including job scheduling and queueing (Kueue), storage management, MLflow integration, and Kubeflow. Additionally, the charts install the following components for integrating with the SageMaker HyperPod cluster resiliency features, which are required components.

- **Health monitoring agent** – This installs the health-monitoring agent provided by SageMaker HyperPod. This is required if you want to get your HyperPod cluster be monitored. Health-monitoring agents are provided as Docker images as follows. In the provided values.yaml in the Helm charts, the image is preset. The agent support GPU-based instances and Trainium-accelerator-based instances (trn1, trn1n, inf2). It is installed to the aws-hyperpod namespace.

```
590183648699.dkr.ecr.us-west-2.amazonaws.com/hyperpod-health-monitoring-agent:1.0.230.0_1.0.19.0
```

- **Deep health check** – This sets up a ClusterRole, a ServiceAccount (deep-health-check-service-account) in the aws-hyperpod namespace, and a ClusterRoleBinding to enable the SageMaker HyperPod deep health check feature. For more information about the Kubernetes RBAC file for deep health check, see the configuration file [deep-health-check-rbac.yaml](#) in the SageMaker HyperPod CLI GitHub repository.
- **job-auto-restart** - This sets up a ClusterRole, a ServiceAccount (job-auto-restart) in the aws-hyperpod namespace, and a ClusterRoleBinding, to enable the auto-restart feature for PyTorch training jobs in SageMaker HyperPod. For more information about the Kubernetes RBAC file for job-auto-restart, see the configuration file [job-auto-restart-rbac.yaml](#) in the SageMaker HyperPod CLI GitHub repository.
- **Kubeflow MPI operator** – The [MPI Operator](#) is a Kubernetes operator that simplifies running distributed Machine Learning (ML) and High-Performance Computing (HPC) workloads using the Message Passing Interface (MPI) on Kubernetes clusters. It installs MPI Operator v0.5. It is installed to the mpi-operator namespace.
- **nvidia-device-plugin** – This is a Kubernetes device plug-in that allows you to automatically expose NVIDIA GPUs for consumption by containers in your Amazon EKS cluster. It allows Kubernetes to allocate and provide access to the requested GPUs for that container. Required when using an instance type with GPU.
- **neuron-device-plugin** – This is a Kubernetes device plug-in that allows you to automatically expose AWS Inferentia chips for consumption by containers in your Amazon EKS cluster. It allows Kubernetes to access and utilize the AWS Inferentia chips on the cluster nodes. Required when using a Neuron instance type.
- **aws-efa-k8s-device-plugin** – This is a Kubernetes device plug-in that enables the use of AWS Elastic Fabric Adapter (EFA) on Amazon EKS clusters. EFA is a network device that provides low-latency and high-throughput communication between instances in a cluster. Required when using an EFA supported instance type.

For more information about the installation procedure using the provided Helm charts, see the [README file in the SageMaker HyperPod CLI repository](#).

## Setting up Kubernetes role-based access control

Cluster admin users also need to set up [Kubernetes role-based access control \(RBAC\)](#) for data scientist users to use the [SageMaker HyperPod CLI](#) to run workloads on HyperPod clusters orchestrated with Amazon EKS.

### Option 1: Set up RBAC using Helm chart

The SageMaker HyperPod service team provides a Helm sub-chart for setting up RBAC. To learn more, see [the section called “Install packages on the Amazon EKS cluster using Helm”](#).

### Option 2: Set up RBAC manually

Create ClusterRole and ClusterRoleBinding with the minimum privilege, and create Role and RoleBinding with mutation permissions.

#### To create ClusterRole & ClusterRoleBinding for data scientist IAM role

Create a cluster-level configuration file cluster\_level\_config.yaml as follows.

```
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: hyperpod-scientist-user-cluster-role
rules:
- apiGroups: []
  resources: ["pods"]
  verbs: ["list"]
- apiGroups: []
  resources: ["nodes"]
  verbs: ["list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: hyperpod-scientist-user-cluster-role-binding
subjects:
- kind: Group
  name: hyperpod-scientist-user-cluster-level
  apiGroup: rbac.authorization.k8s.io
roleRef:
```

```
kind: ClusterRole
name: hyperpod-scientist-user-cluster-role # this must match the name of the Role or
ClusterRole you wish to bind to
apiGroup: rbac.authorization.k8s.io
```

Apply the configuration to the EKS cluster.

```
kubectl apply -f cluster_level_config.yaml
```

## To create Role and RoleBinding in namespace

This is the namespace training operator that run training jobs and Resiliency will monitor by default. Job auto-resume can only support in kubeflow namespace or namespace prefixed aws-hyperpod.

Create a role configuration file `namespace_level_role.yaml` as follows. This example creates a role in the kubeflow namespace

```
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  namespace: kubeflow
  name: hyperpod-scientist-user-namespace-level-role
###
# 1) add/list/describe/delete pods
# 2) get/list/watch/create/patch/update/delete/describe kubeflow pytorch job
# 3) get pod log
###
rules:
- apiGroups: []
  resources: ["pods"]
  verbs: ["create", "get"]
- apiGroups: []
  resources: ["nodes"]
  verbs: ["get", "list"]
- apiGroups: []
  resources: ["pods/log"]
  verbs: ["get", "list"]
- apiGroups: []
  resources: ["pods/exec"]
  verbs: ["get", "create"]
- apiGroups: ["kubeflow.org"]
```

```
resources: ["pytorchjobs", "pytorchjobs/status"]
verbs: ["get", "list", "create", "delete", "update", "describe"]
- apiGroups: [""]
  resources: ["configmaps"]
  verbs: ["create", "update", "get", "list", "delete"]
- apiGroups: [""]
  resources: ["secrets"]
  verbs: ["create", "get", "list", "delete"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  namespace: kubeflow
  name: hyperpod-scientist-user-namespace-level-role-binding
subjects:
- kind: Group
  name: hyperpod-scientist-user-namespace-level
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: hyperpod-scientist-user-namespace-level-role # this must match the name of the
Role or ClusterRole you wish to bind to
  apiGroup: rbac.authorization.k8s.io
```

Apply the configuration to the EKS cluster.

```
kubectl apply -f namespace_level_role.yaml
```

## Create an access entry for Kubernetes groups

After you have set up RBAC using one of the two options above, use the following sample command replacing the necessary information.

```
aws eks create-access-entry \
--cluster-name <eks-cluster-name> \
--principal-arn arn:aws:iam::<AWS_ACCOUNT_ID_SCIENTIST_USER>:role/ScientistUserRole
\
--kubernetes-groups '['hyperpod-scientist-user-namespace-level","hyperpod-
scientist-user-cluster-level"]'
```

For the principal-arn parameter, you need to use the [the section called “IAM users for scientists”](#).

## Managing SageMaker HyperPod clusters using the SageMaker HyperPod console UI

The following topics provide guidance on how to manage SageMaker HyperPod in the SageMaker AI console.

### Topics

- [Create a SageMaker HyperPod cluster](#)
- [Browse, view, and edit SageMaker HyperPod clusters](#)
- [SageMaker HyperPod task governance](#)
- [Delete a SageMaker HyperPod cluster](#)

### Create a SageMaker HyperPod cluster

See the following instructions on creating a new SageMaker HyperPod cluster using the SageMaker HyperPod console UI.

1. Open the Amazon SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. Choose **HyperPod Clusters** in the left navigation pane and then **Cluster Management**.
3. In the SageMaker HyperPod landing page, choose **Create HyperPod cluster**.
4. From the drop-down menu of **Create HyperPod cluster**, choose **Orchestrated by Amazon EKS**.
5. From the Amazon EKS cluster list, choose the EKS cluster with which you want to configure the new HyperPod cluster.
  1. If you need to create a new EKS cluster, choose **Create EKS cluster**. You can create it from the EKS cluster list page without having to open the Amazon EKS console.

 **Note**

The VPC subnet you choose for HyperPod has to be private.

2. After submitting a new EKS cluster creation request, wait until the EKS cluster becomes Active.
3. Install the Helm chart as instructed in [the section called “Install packages on the Amazon EKS cluster using Helm”](#).

4. After the EKS cluster creation has completed, choose **Create HyperPod cluster** and then **Orchestrated by EKS** again. You should be able to find and select the new EKS cluster. To proceed, choose **Select**.
6. On the **Configure a new HyperPod cluster** page, set up the basic information for the cluster such as name, options to enable the HyperPod cluster resiliency features, and tags.
7. For **Cluster name**, specify a name for the new cluster.
8. For **Cluster resiliency - node recovery**, specify **Automatic** to enable automatic node recovery. SageMaker HyperPod replaces or reboots instances (nodes) when issues are found by the health-monitoring agent.
9. For **Tags**, add key and value pairs to the new cluster and manage the cluster as an AWS resource. To learn more, see [Tagging your AWS resources](#).
10. In **Step 2: Advanced configuration**, configure network settings within the cluster and in-and-out of the cluster. For orchestration of SageMaker HyperPod cluster with Amazon EKS, the VPC is automatically set to the one configured with the EKS cluster you selected.
11. In **Step 3: Configure instance groups**, choose **Create instance group**. Each instance group can be configured differently, and you can create a heterogeneous cluster that consists of multiple instance groups with various instance types. In the **Create an instance group** configuration pop-up window, fill the instance group configuration information.

Create an instance group pop-up page, configure a new instance group following the UI guidance.

- a. For **Instance group name**, specify a name for the instance group.
- b. For **Select instance type**, choose the instance for the instance group.
- c. For **Quantity**, specify an integer not exceeding the instance quota for cluster usage.
- d. Prepare a lifecycle configuration script and upload to an Amazon S3 bucket, such as `s3://amzn-s3-demo-bucket/Lifecycle-scripts/base-config/`.

For a quick start, download the sample script [on\\_create.sh](#) from the AWSome Distributed Training GitHub repository, and upload it to the S3 bucket. This script sets up the logging file `/var/log/provision/provisioning.log` required for CloudWatch to gather logs from Pod containers. You can also include additional setup instructions, a series of setup scripts, or commands to be executed during the HyperPod cluster provisioning stage.

- e. For **S3 bucket URI for lifecycle scripts**, enter the Amazon S3 path in which the lifecycle scripts are stored.
  - f. For **Directory path to the entrypoint script in the base Amazon S3 path**, enter the file name of the lifecycle script under **Amazon S3 path to lifecycle script files**. If you use the provided sample script, enter `on_create.sh`.
  - g. For IAM role, choose the IAM role you have created for SageMaker HyperPod resources, following the section [the section called “IAM role for SageMaker HyperPod”](#).
  - h. Under **Advanced configuration**, you can set up the following optional configurations.
    - i. (Optional) For **Threads per core**, specify 1 for disabling multi-threading and 2 for enabling multi-threading. To find which instance type supports multi-threading, see the reference table of [CPU cores and threads per CPU core per instance type](#) in the *Amazon EC2 User Guide*.
    - ii. (Optional) For **Additional instance storage configs**, specify an integer between 1 and 16384 to set the size of an additional Elastic Block Store (EBS) volume in gigabytes (GB). The EBS volume is attached to each instance of the instance group. The default mount path for the additional EBS volume is `/opt/sagemaker`. After the cluster is successfully created, you can SSH into the cluster instances (nodes) and verify if the EBS volume is mounted correctly by running the `df -h` command. Attaching an additional EBS volume provides stable, off-instance, and independently persisting storage, as described in the [Amazon EBS volumes](#) section in the *Amazon Elastic Block Store User Guide*.
12. For **Deep health check**, select the advanced health checks you want to run on the instances. To learn more, see [the section called “Deep health checks”](#).
  13. In **Step 4: Review and create**, review the configuration you have set from **Step 1 to Step 3** and finish submitting the cluster creation request.
  14. After the status of the cluster turns to **InService**, you can start logging into the cluster nodes. To access the cluster nodes and start running ML workloads, see [the section called “Jobs on HyperPod clusters”](#).

## Browse, view, and edit SageMaker HyperPod clusters

Use the following instructions to browse, view, and edit SageMaker HyperPod clusters orchestrated by Amazon EKS in the SageMaker AI console.

## Topics

- [To browse your SageMaker HyperPod clusters](#)
- [To view details of each SageMaker HyperPod cluster](#)
- [To edit a SageMaker HyperPod cluster](#)

## To browse your SageMaker HyperPod clusters

Under **Clusters** on the SageMaker HyperPod page in the SageMaker AI console, all created clusters should be listed under the **Clusters** section, which provides a summary view of clusters, their ARNs, status, and creation time.

## To view details of each SageMaker HyperPod cluster

Under **Clusters** on the SageMaker HyperPod page in the SageMaker AI console, the cluster names are activated as links. Choose the cluster name link to see details of each cluster.

## To edit a SageMaker HyperPod cluster

1. Under **Clusters** in the main pane of the SageMaker HyperPod console, choose the cluster you want to update.
2. Select your cluster, and choose **Edit**.
3. In the **Edit <your-cluster>** page, you can edit the configurations of existing instance groups, add more instance groups, delete instance groups, and change tags for the cluster. After making changes, choose **Submit**.
  - a. In the **Configure instance groups** section, you can add more instance groups by choosing **Create instance group**.
  - b. In the **Configure instance groups** section, you can choose **Edit** to change its configuration or **Delete** to remove the instance group permanently.

### Important

When deleting an instance group, consider the following points:

- Your SageMaker HyperPod cluster must always maintain at least one instance group.
- Ensure all critical data is backed up before removal.
- The removal process cannot be undone.

**Note**

Deleting an instance group will terminate all compute resources associated with that group.

- c. In the **Tags** section, you can update tags for the cluster.

## SageMaker HyperPod task governance

SageMaker HyperPod task governance is a robust management system designed to streamline resource allocation and ensure efficient utilization of compute resources across teams and projects for your Amazon EKS clusters. This provides administrators with the capability to set:

- Priority levels for various tasks
- Compute allocation for each team
- How each team lends and borrows idle compute
- If a team preempts their own tasks

HyperPod task governance also provides Amazon EKS cluster Observability, offering real-time visibility into cluster capacity. This includes compute availability and usage, team allocation and utilization, and task run and wait time information, setting you up for informed decision-making and proactive resource management.

The following sections cover how to set up, understand key concepts, and use HyperPod task governance for your Amazon EKS clusters.

### Topics

- [Setup for SageMaker HyperPod task governance](#)
- [Dashboard](#)
- [Tasks](#)
- [Policies](#)
- [Example HyperPod task governance AWS CLI commands](#)
- [Troubleshoot](#)
- [Attribution document for Amazon SageMaker HyperPod task governance](#)

## Setup for SageMaker HyperPod task governance

The following section provides information on how to get set up with the Amazon CloudWatch Observability EKS and SageMaker HyperPod task governance add-ons.

If you have not already done so, see [IAM users for cluster admin](#) for the example minimum permission policy for HyperPod cluster administrators. This includes permissions run the SageMaker HyperPod core APIs and manage SageMaker HyperPod clusters within your AWS account, performing the tasks in [SageMaker HyperPod operation](#).

### Topics

- [Dashboard setup](#)
- [Task governance setup](#)

### Dashboard setup

Use the following information to get set up with Amazon SageMaker HyperPod Amazon CloudWatch Observability EKS add-on. This sets you up with a detailed visual dashboard that provides a view into metrics for your EKS cluster hardware, team allocation, and tasks.

If you are having issues setting up, please see [Troubleshoot](#) for known troubleshooting solutions.

### Topics

- [HyperPod Amazon CloudWatch Observability EKS add-on prerequisites](#)
- [HyperPod Amazon CloudWatch Observability EKS add-on setup](#)

### HyperPod Amazon CloudWatch Observability EKS add-on prerequisites

The following section includes the prerequisites needed before installing the Amazon EKS Observability add-on.

- If you have not already done so, follow the instructions in [IAM users for cluster admin](#) to ensure that you have the minimum permission for HyperPod cluster administrative tasks.
- Attach the CloudWatchAgentServerPolicy IAM policy to your worker nodes. To do so, enter the following command. Replace *my-worker-node-role* with the IAM role used by your Kubernetes worker nodes.

```
aws iam attach-role-policy \
```

```
--role-name my-worker-node-role \
--policy-arn arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy
```

## HyperPod Amazon CloudWatch Observability EKS add-on setup

Use the following options to set up the Amazon SageMaker HyperPod Amazon CloudWatch Observability EKS add-on.

Setup using the SageMaker AI console

The following permissions are required for setup and visualizing the HyperPod task governance dashboard. This section expands upon the permissions listed in [IAM users for cluster admin](#).

To manage task governance, use the sample policy:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sagemaker>ListClusters",
        "sagemaker>DescribeCluster",
        "sagemaker>ListComputeQuotas",
        "sagemaker>CreateComputeQuota",
        "sagemaker>UpdateComputeQuota",
        "sagemaker>DescribeComputeQuota",
        "sagemaker>DeleteComputeQuota",
        "sagemaker>ListClusterSchedulerConfigs",
        "sagemaker>DescribeClusterSchedulerConfig",
        "sagemaker>CreateClusterSchedulerConfig",
        "sagemaker>UpdateClusterSchedulerConfig",
        "sagemaker>DeleteClusterSchedulerConfig",
        "eks>ListAddons",
        "eks>CreateAddon",
        "eks>DescribeAddon",
        "eks>DescribeCluster",
        "eks>DescribeAccessEntry",
        "eks>ListAssociatedAccessPolicies",
        "eks>AssociateAccessPolicy",
        "eks>DisassociateAccessPolicy"
      ],
    }
  ]
}
```

```
        "Resource": "*"
    }
]
}
```

To grant permissions to manage Amazon CloudWatch Observability Amazon EKS and view the HyperPod cluster dashboard through the SageMaker AI console, use the sample policy below:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks>ListAddons",
                "eks>CreateAddon",
                "eks>UpdateAddon",
                "eks>DescribeAddon",
                "eks>DescribeAddonVersions",
                "sagemaker>DescribeCluster",
                "sagemaker>DescribeClusterNode",
                "sagemaker>ListClusterNodes",
                "sagemaker>ListClusters",
                "sagemaker>ListComputeQuotas",
                "sagemaker>DescribeComputeQuota",
                "sagemaker>ListClusterSchedulerConfigs",
                "sagemaker>DescribeClusterSchedulerConfig",
                "eks>DescribeCluster",
                "cloudwatch:GetMetricData",
                "eks>AccessKubernetesApi"
            ],
            "Resource": "*"
        }
    ]
}
```

Navigate to the **Dashboard** tab in the SageMaker HyperPod console to install the Amazon CloudWatch Observability EKS. To ensure task governance related metrics are included in the **Dashboard**, enable the Kueue metrics checkbox. Enabling the Kueue metrics enables CloudWatch **Metrics** costs, after free-tier limit is reached. For more information, see [Metrics in Amazon CloudWatch Pricing](#).

## Setup using the EKS AWS CLI

Use the following EKS AWS CLI command to install the add-on:

```
aws eks create-addon --cluster-name cluster-name
--addon-name amazon-cloudwatch-observability
--configuration-values "configuration json"
```

Below is an example of the JSON of the configuration values:

```
{
  "agent": {
    "config": {
      "logs": {
        "metrics_collected": {
          "kubernetes": {
            "kueue_container_insights": true,
            "enhanced_container_insights": true
          },
          "application_signals": { }
        }
      },
      "traces": {
        "traces_collected": {
          "application_signals": { }
        }
      }
    },
    "traces": {
      "traces_collected": {
        "application_signals": { }
      }
    }
  }
}
```

## Setup using the EKS Console UI

1. Navigate to the [EKS console](#).
2. Choose your cluster.
3. Choose **Add-ons**.
4. Find the **Amazon CloudWatch Observability** add-on and install. Install version >= 2.4.0 for the add-on.
5. Include the following JSON, Configuration values:

```
{
```

```
"agent": {
    "config": {
        "logs": {
            "metrics_collected": {
                "kubernetes": {
                    "kueue_container_insights": true,
                    "enhanced_container_insights": true
                },
                "application_signals": { }
            },
            "traces": {
                "traces_collected": {
                    "application_signals": { }
                }
            }
        },
        "traces": {
            "traces_collected": {
                "application_signals": { }
            }
        }
    },
    "traces": {
        "traces_collected": {
            "application_signals": { }
        }
    }
}
```

Once the EKS Observability add-on has been successfully installed, you can view your EKS cluster metrics under the HyperPod console **Dashboard** tab.

## Task governance setup

This section includes information on how to set up the Amazon SageMaker HyperPod task governance EKS add-on. This includes granting permissions that allows you to set task prioritization, compute allocation for teams, how idle compute is shared, and task preemption for teams.

If you are having issues setting up, please see [Troubleshoot](#) for known troubleshooting solutions.

## Topics

- [Kueue Settings](#)
- [HyperPod Task governance prerequisites](#)
- [HyperPod task governance setup](#)

## Kueue Settings

HyperPod task governance EKS add-on installs [Kueue](#) for your HyperPod EKS clusters. Kueue is a kubernetes-native system that manages quotas and how jobs consume them.

EKS HyperPod task governance add-on version	Version of Kueue that is installed as part of the add-on	Version of kube-rbac-proxy that is installed as part of the add-on
v1.0.0	v0.8.1	v0.18.1

HyperPod task governance leverages Kueue for Kubernetes-native job queueing, scheduling, and quota management, and is installed with the HyperPod task governance EKS add-on. When installed, HyperPod creates and modifies SageMaker AI-managed Kubernetes resources such as `KueueManagerConfig`, `ClusterQueues`, `LocalQueues`, `WorkloadPriorityClasses`, `ResourceFlavors`, and `ValidatingAdmissionPolicies`. While Kubernetes administrators have the flexibility to modify the state of these resources, it is possible that any changes made to a SageMaker AI-managed resource may be updated and overwritten by the service.

The following information outlines the configuration settings utilized by the HyperPod task governance add-on for setting up Kueue.

```
apiVersion: config.kueue.x-k8s.io/v1beta1
  kind: Configuration
  health:
    healthProbeBindAddress: :8081
  metrics:
    bindAddress: :8080
    enableClusterQueueResources: true
  webhook:
    port: 9443
  manageJobsWithoutQueueName: false
  leaderElection:
    leaderElect: true
    resourceName: c1f6bfd2.kueue.x-k8s.io
  controller:
    groupKindConcurrency:
      Job.batch: 5
      Pod: 5
      Workload.kueue.x-k8s.io: 5
```

```
LocalQueue.kueue.x-k8s.io: 1
ClusterQueue.kueue.x-k8s.io: 1
ResourceFlavor.kueue.x-k8s.io: 1

clientConnection:
  qps: 50
  burst: 100
integrations:
  frameworks:
    - "batch/job"
    - "kubeflow.org/mpijob"
    - "ray.io/rayjob"
    - "ray.io/raycluster"
    - "jobset.x-k8s.io/jobset"
    - "kubeflow.org/mxjob"
    - "kubeflow.org/paddlejob"
    - "kubeflow.org/pytorchjob"
    - "kubeflow.org/tfjob"
    - "kubeflow.org/xgboostjob"
    - "pod"
podOptions:
  namespaceSelector:
    matchExpressions:
      - key: kubernetes.io/metadata.name
        operator: NotIn
        values: [ kube-system, kueue-system ]
fairSharing:
  enable: true
  preemptionStrategies: [LessThanOrEqualToFinalShare, LessThanInitialShare]
resources:
  excludeResourcePrefixes: []
```

For more information about each configuration entry, see [Configuration](#) in the Kueue documentation.

## HyperPod Task governance prerequisites

- If you have not already done so, see [IAM users for cluster admin](#) for the example minimum permission policy for HyperPod cluster administrators. This includes permissions run the SageMaker HyperPod core APIs and manage SageMaker HyperPod clusters within your AWS account, performing the tasks in [SageMaker HyperPod operation](#).
- You will need to have your Kubernetes version  $\geq 1.30$ . For instructions, see [Update existing clusters to the new Kubernetes version](#).

- If you already have Kueue installed in their clusters, uninstall Kueue before installing the EKS add-on.
- A HyperPod node must already exist in the EKS cluster before installing the HyperPod task governance add-on.

## HyperPod task governance setup

The following provides information on how to get set up with HyperPod task governance.

### Setup using the SageMaker AI console

The following provides information on how to get set up with HyperPod task governance using the SageMaker HyperPod console.

You already have all of the following permissions attached if you have already granted permissions to manage Amazon CloudWatch Observability EKS and view the HyperPod cluster dashboard through the SageMaker AI console in the [HyperPod Amazon CloudWatch Observability EKS add-on setup](#). If you have not set this up, use the sample policy below to grant permissions to manage the HyperPod task governance add-on and view the HyperPod cluster dashboard through the SageMaker AI console.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "eks>ListAddons",  
                "eks>CreateAddon",  
                "eks>UpdateAddon",  
                "eks>DescribeAddon",  
                "eks>DescribeAddonVersions",  
                "sagemaker>DescribeCluster",  
                "sagemaker>DescribeClusterNode",  
                "sagemaker>ListClusterNodes",  
                "sagemaker>ListClusters",  
                "eks>DescribeCluster",  
                "eks>AccessKubernetesApi"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

```
]  
}
```

Navigate to the **Dashboard** tab in the SageMaker HyperPod console to install the Amazon SageMaker HyperPod task governance Add-on.

### Setup using the Amazon EKS AWS CLI

Use the example [`create-addon`](#) EKS AWS CLI command to set up the HyperPod task governance Amazon EKS API and console UI using the AWS CLI:

```
aws eks create-addon --region region --cluster-name cluster-name --addon-name  
amazon-sagemaker-hyperpod-taskgovernance
```

You can view the **Policies** tab in the HyperPod SageMaker AI console if the install was successful. You can also use the following example [`describe-addon`](#) EKS AWS CLI command to check the status.

```
aws eks describe-addon --region region --cluster-name cluster-name --addon-name amazon-  
sagemaker-hyperpod-taskgovernance
```

## Dashboard

Amazon SageMaker HyperPod task governance provides a comprehensive dashboard view of your Amazon EKS cluster utilization metrics, including hardware, team, and task metrics. The following provides information on your HyperPod EKS cluster dashboard.

The dashboard provides a comprehensive view of cluster utilization metrics, including hardware, team, and task metrics. You will need to install the EKS add-on to view the dashboard. For more information, see [Dashboard setup](#).

In the [Amazon SageMaker AI console](#), under **HyperPod Clusters**, you can navigate to the HyperPod console and view your list of HyperPod clusters in your Region. Choose your cluster and navigate to the **Dashboard** tab. The dashboard contains the following metrics. You can download the data for a section by choosing the corresponding **Export**.

### Utilization

Provides health of the EKS cluster point-in-time and trend-based metrics for critical compute resources. By default, **All Instance Groups** are shown. Use the dropdown menu to filter your instance groups. The metrics included in this section are:

- Number of total, running, and pending recovery instances. The number of pending recovery instances refer to the number of instances that need attention for recovery.
- GPUs, GPU memory, vCPUs, and vCPUs memory.
- GPU utilization, GPU memory utilization, vCPU utilization, and vCPU memory utilization.
- An interactive graph of your GPU and vCPU utilization.

## Teams

Provides information into team-specific resource management. This includes:

- Instance and GPU allocation.
- GPU utilization rates.
- Borrowed GPU statistics.
- Task status (running or pending).
- A bar chart view of GPU utilization versus compute allocation across teams.
- Team detailed GPU and vCPU-related information. By default, the information displayed includes **All teams**. You can filter by team and instances by choosing the dropdown menus. In the interactive plot you can filter by time.

## Tasks

### Note

To view your HyperPod EKS cluster tasks in the dashboard:

- Configure Kubernetes Role-Based Access Control (RBAC) for data scientist users in the designated HyperPod namespace to authorize task execution on Amazon EKS-orchestrated clusters. Namespaces follow the format `hyperpod-ns-team-name`. To establish RBAC permissions, refer to the [team role creation instructions](#).

- Ensure that your job is submitted with the appropriate namespace and priority class labels. For a comprehensive example, see [Submit a job to SageMaker AI-managed queue and namespace](#).

Provides information on task-related metrics. This includes number of running, pending, and preempted tasks, and run and wait time statistics. By default, the information displayed includes **All teams**. You can filter by team by choosing the dropdown menu. In the interactive plot you can filter by time.

## Tasks

The following provides information on Amazon SageMaker HyperPod EKS cluster tasks. Tasks are operations or jobs that are sent to the cluster. These can be machine learning operations, like training, running experiments, or inference. The viewable task details list include status, run time, and how much compute is being used per task.

In the [Amazon SageMaker AI console](#), under **HyperPod Clusters**, you can navigate to the HyperPod console and view your list of HyperPod clusters in your Region. Choose your cluster and navigate to the **Tasks** tab.

For the **Tasks** tab to be viewable from anyone besides the administrator, the administrator needs to [add an access entry to the EKS cluster for the IAM role](#).

### Note

To view your HyperPod EKS cluster tasks in the dashboard:

- Configure Kubernetes Role-Based Access Control (RBAC) for data scientist users in the designated HyperPod namespace to authorize task execution on Amazon EKS-orchestrated clusters. Namespaces follow the format `hyperpod-ns-team-name`. To establish RBAC permissions, refer to the [team role creation instructions](#).
- Ensure that your job is submitted with the appropriate namespace and priority class labels. For a comprehensive example, see [Submit a job to SageMaker AI-managed queue and namespace](#).

For EKS clusters, kubeflow (PyTorch, MPI, TensorFlow) tasks are shown. By default, PyTorch tasks are shown. You can filter for PyTorch, MPI, TensorFlow tasks by choosing the dropdown menu or

using the search field. The information that is shown for each task includes the task name, status, namespace, priority class, and creation time.

## Policies

Amazon SageMaker HyperPod task governance simplifies how your Amazon EKS cluster resources are allocated and how tasks are prioritized. The following provides information on HyperPod EKS cluster policies. For information on how to set up task governance, see [Task governance setup](#).

The policies are divided up into **Compute prioritization** and **Compute allocation**. The policy concepts below will be organized in the context of these policies.

**Compute prioritization**, or cluster policy, determines how idle compute is borrowed and how tasks are prioritized by teams.

- **Idle compute allocation** defines how idle compute is allocated across teams. That is, how unused compute can be borrowed from teams. When choosing an **Idle compute allocation**, you can choose between:
  - **First-come first-serve**: When applied, teams are not prioritized against each other and each incoming task is equally likely to obtain over-quota resources. Tasks are prioritized based on order of submission. This means a user may be able to use 100% of the idle compute if they request it first.
  - **Fair-share**: When applied, teams borrow idle compute based on their assigned **Fair-share weight**. These weights are defined in **Compute allocation**. For more information on how this can be used, see [Sharing idle compute resources examples](#).
- **Task prioritization** defines how tasks are queued as compute becomes available. When choosing a **Task prioritization**, you can choose between:
  - **First-come first-serve**: When applied, tasks are queued in the order they are requested.
  - **Task ranking**: When applied, tasks are queued in the order defined by their prioritization. If this option is chosen, you must add priority classes along with the weights at which they should be prioritized. Tasks of the same priority class will be executed on a first-come first-serve basis. When enabled in Compute allocation, tasks are preempted from lower priority tasks by higher priority tasks within the team.

When data scientists submit jobs to the cluster, they use the priority class name in the YAML file. The priority class is in the format *priority-class-name*-priority. For an example, see [Submit a job to SageMaker AI-managed queue and namespace](#).

- **Priority classes:** These classes establish a relative priority for tasks when borrowing capacity. When a task is running using borrowed quota, it may be preempted by another task of higher priority than it, if no more capacity is available for the incoming task. If **Preemption** is enabled in the **Compute allocation**, a higher priority task may also preempt tasks within its own team.

**Compute allocation**, or compute quota, defines a team's compute allocation and what weight (or priority level) a team is given for fair-share idle compute allocation.

- **Team name:** The team name. A corresponding **Namespace** will be created, of type hyperpods-*team-name*.
- **Members:** Members of the team namespace. You will need to set up a Kubernetes role-based access control (RBAC) for data scientist users that you want to be part of this team, to run tasks on HyperPod clusters orchestrated with Amazon EKS. To set up a Kubernetes RBAC, use the instructions in [create team role](#).
- **Fair-share weight:** This is the level of prioritization assigned to the team when **Fair-share** is applied for **Idle compute allocation**. The highest priority has a weight of 100 and the lowest priority has a weight of 0. Higher weight enables a team to access unutilized resources within shared capacity sooner. A zero weight signifies the lowest priority, implying this team will always be at a disadvantage compared to other teams.

The fair-share weight provides a comparative edge to this team when vying for available resources against others. Admission prioritizes scheduling tasks from teams with the highest weights and the lowest borrowing. For example, if Team A has a weight of 10 and Team B has a weight of 5, Team A would have priority in accessing unutilized resources as it would have jobs that are scheduled earlier than Team B.

- **Task preemption:** Compute is taken over from a task based on priority. By default, the team loaning idle compute will preempt tasks from other teams.
- **Lending and borrowing:** How idle compute is being lent by the team and if the team can borrow from other teams.
  - **Borrow limit:** The limit of idle compute that a team is allowed to borrow. A team can borrow up to 500% of allocated compute. The value you provide here is interpreted as a percentage. For example, a value of 500 will be interpreted as 500%.

For information on how these concepts are used, such as priority classes and name spaces, see [Example HyperPod task governance AWS CLI commands](#).

## Sharing idle compute resources examples

The total reserved quota should not surpass the cluster's available capacity for that resource, to ensure proper quota management. For example, if a cluster comprises 20 m1 . c5 . 2xlarge instances, the cumulative quota assigned to teams should remain under 20.

If the **Compute allocation** policies for teams allow for **Lend and Borrow** or **Lend**, the idle capacity is shared between these teams. For example, Team A and Team B have **Lend and Borrow** enabled. Team A has a quota of 6 but is using only 2 for its jobs, and Team B has a quota of 5 and is using 4 for its jobs. A job that is submitted to Team B requiring 4 resources. 3 will be borrowed from Team A.

If any team's **Compute allocation** policy is set to **Don't Lend**, the team would not be able to borrow any additional capacity beyond its own allocations.

To maintain a pool or a set of resources that all teams can borrow from, you can set up a dedicated team with resources that bridge the gap between other teams' allocations and the total cluster capacity. Ensure that this cumulative resource allocation includes the appropriate instance types and does not exceed the total cluster capacity. To ensure that these resources can be shared among teams, enable the participating teams to have their compute allocations set to **Lend and Borrow** or **Lend** for this common pool of resources. Every time new teams are introduced, quota allocations are changed, or there are any changes to the cluster capacity, revisit the quota allocations of all the teams and ensure the cumulative quota remains at or below cluster capacity.

### Topics

- [Create policies](#)
- [Edit policies](#)

### Create policies

You can create your **Cluster policy** and **Compute allocation** configurations in the **Policies** tab. For following provides instructions on how to create the following configurations.

- Create your **Cluster policy** to update how tasks are prioritized and idle compute is allocated.
- Create **Compute allocation** to create a new compute allocation policy for a team.

**Note**

When you create a **Compute allocation** you will need to set up a Kubernetes role-based access control (RBAC) for data scientist users in the corresponding namespace to run tasks on HyperPod clusters orchestrated with Amazon EKS. The namespaces have the format `hyperpod-ns-team-name`. To set up a Kubernetes RBAC, use the instructions in [create team role](#).

For information about the HyperPod task governance EKS cluster policy concepts, see [Policies](#).

## Create HyperPod task governance policies

This procedure assumes that you have already created an Amazon EKS cluster set up with HyperPod. If you have not already done so, see [Create a SageMaker HyperPod cluster](#).

1. Navigate to the [Amazon SageMaker AI console](#).
2. On the left navigation pane, under **HyperPod Clusters**, choose **Cluster Management**.
3. Choose your Amazon EKS cluster listed under **SageMaker HyperPod clusters**.
4. Choose the **Policies** tab.
5. To create your **Cluster policy**:
  - a. Choose the corresponding **Edit** to update how tasks are prioritized and idle compute is allocated.
  - b. After you have made your changes, choose **Submit**.
6. To create a **Compute allocation**:
  - a. Choose the corresponding **Create**. This takes you to the compute allocation creation page.
  - b. After you have made your changes, choose **Submit**.

## Edit policies

You can edit your **Cluster policy** and **Compute allocation** configurations in the **Policies** tab. The following provides instructions on how to edit the following configurations.

- Edit your **Cluster policy** to update how tasks are prioritized and idle compute is allocated.
- Edit **Compute allocation** to create a new compute allocation policy for a team.

**Note**

When you create a **Compute allocation** you will need to set up a Kubernetes role-based access control (RBAC) for data scientist users in the corresponding namespace to run tasks on HyperPod clusters orchestrated with Amazon EKS. The namespaces have the format `hyperpod-ns-team-name`. To set up a Kubernetes RBAC, use the instructions in [create team role](#).

For more information about the HyperPod task governance EKS cluster policy concepts, see [Policies](#).

## Edit HyperPod task governance policies

This procedure assumes that you have already created an Amazon EKS cluster set up with HyperPod. If you have not already done so, see [Create a SageMaker HyperPod cluster](#).

1. Navigate to the [Amazon SageMaker AI console](#).
2. On the left navigation pane, under **HyperPod Clusters**, choose **Cluster Management**.
3. Choose your Amazon EKS cluster listed under **SageMaker HyperPod clusters**.
4. Choose the **Policies** tab.
5. To create your **Cluster policy**:
  - a. Choose the corresponding **Edit** to update how tasks are prioritized and idle compute is allocated.
  - b. After you have made your changes, choose **Submit**.
6. To edit your **Compute allocation**:
  - a. Choose the configuration you wish to edit under **Compute allocation**. This takes you to the configuration details page.
  - b. If you wish to edit these configurations, choose **Edit**.
  - c. After you have made your changes, choose **Submit**.

## Example HyperPod task governance AWS CLI commands

You can use HyperPod with EKS through Kubectl or through HyperPod custom CLI. You can use these commands through Studio or AWS CLI. The following provides SageMaker HyperPod task

governance examples, on how to view cluster details using the HyperPod AWS CLI commands. For more information, including how to install, see the [HyperPod CLI Github repository](#).

## Topics

- [Get cluster accelerator device quota information](#)
- [Submit a job to SageMaker AI-managed queue and namespace](#)
- [List jobs](#)
- [Get job detailed information](#)
- [Suspend and unsuspend jobs](#)
- [Debugging jobs](#)

### Get cluster accelerator device quota information

The following example command gets the information on the cluster accelerator device quota.

```
hyperpod get-clusters -n hyperpod-ns-test-team
```

The namespace in this example, hyperpod-ns-test-team, is created in Kubernetes based on the team name provided, test-team, when the compute allocation is created. For more information, see [Edit policies](#).

Example response:

```
[  
  {  
    "Cluster": "hyperpod-eks-test-cluster-id",  
    "InstanceType": "ml.g5.xlarge",  
    "TotalNodes": 2,  
    "AcceleratorDevicesAvailable": 1,  
    "NodeHealthStatus=Schedulable": 2,  
    "DeepHealthCheckStatus=Passed": "N/A",  
    "Namespaces": {  
      "hyperpod-ns-test-team": {  
        "TotalAcceleratorDevices": 1,  
        "AvailableAcceleratorDevices": 1  
      }  
    }  
  }  
]
```

## Submit a job to SageMaker AI-managed queue and namespace

The following example command submits a job to your HyperPod cluster. If you have access to only one team, the HyperPod AWS CLI will automatically assign the queue for you in this case. Otherwise if multiple queues are discovered, we will display all viable options for you to select.

```
hyperpod start-job --job-name hyperpod-cli-test --job-kind kubeflow/PyTorchJob --image docker.io/kubeflowkatib/pytorch-mnist-cpu:v1beta1-bc09cf0 --entry-script /opt/pytorch-mnist/mnist.py --pull-policy IfNotPresent --instance-type ml.g5.xlarge --node-count 1 --tasks-per-node 1 --results-dir ./result --priority training-priority
```

The priority classes are defined in the **Cluster policy**, which defines how tasks are prioritized and idle compute is allocated. When a data scientist submits a job, they use one of the priority class names with the format *priority-class-name*-priority. In this example, **training-priority** refers to the priority class named “training”. For more information on policy concepts, see [Policies](#).

If a priority class is not specified, the job is treated as a low priority job, with a task ranking value of 0.

If a priority class is specified, but does not correspond to one of the priority classes defined in the **Cluster policy**, the submission fails and an error message provides the defined set of priority classes.

You can also submit the job using a YAML configuration file using the following command:

```
hyperpod start-job --config-file ./yaml-configuration-file-name.yaml
```

The following is an example YAML configuration file that is equivalent to submitting a job as discussed above.

```
defaults:
  - override hydra/job_logging: stdout
hydra:
  run:
    dir: .
    output_subdir: null
training_cfg:
  entry_script: /opt/pytorch-mnist/mnist.py
  script_args: []
  run:
```

```
name: hyperpod-cli-test
nodes: 1
ntasks_per_node: 1
cluster:
  cluster_type: k8s
  instance_type: ml.g5.xlarge
  custom_labels:
    kueue.x-k8s.io/priority-class: training-priority
  cluster_config:
    label_selector:
      required:
        sagemaker.amazonaws.com/node-health-status:
          - Schedulable
    preferred:
      sagemaker.amazonaws.com/deep-health-check-status:
        - Passed
    weights:
      - 100
  pullPolicy: IfNotPresent
base_results_dir: ./result
container: docker.io/kubeflowkatib/pytorch-mnist-cpu:v1beta1-bc09cf
env_vars:
  NCCL_DEBUG: INFO
```

Alternatively, you can submit a job using `kubectl` to ensure the task appears in the **Dashboard** tab. The following is an example `kubectl` command.

```
kubectl apply -f ./yaml-configuration-file-name.yaml
```

When submitting the job, include your queue name and priority class labels. For example, with the queue name `hyperpod-ns-team-name-localqueue` and priority class `priority-class-name-priority`, you must include the following labels:

- `kueue.x-k8s.io/queue-name: hyperpod-ns-team-name-localqueue`
- `kueue.x-k8s.io/priority-class: priority-class-name-priority`

The following YAML configuration snippet demonstrates how to add labels to your original configuration file to ensure your task appears in the **Dashboard** tab:

```
metadata:
  name: job-name
```

```
namespace: hyperpod-ns-team-name
labels:
  kueue.x-k8s.io/queue-name: hyperpod-ns-team-name-localqueue
  kueue.x-k8s.io/priority-class: priority-class-name-priority
```

## List jobs

The following command lists the jobs and their details.

```
hyperpod list-jobs
```

Example response:

```
{
  "jobs": [
    {
      "Name": "hyperpod-cli-test",
      "Namespace": "hyperpod-ns-test-team",
      "CreationTime": "2024-11-18T21:21:15Z",
      "Priority": "training",
      "State": "Succeeded"
    }
  ]
}
```

## Get job detailed information

The following command provides a job's details. If no namespace is specified, HyperPod AWS CLI will fetch a namespace managed by SageMaker AI that you have access to.

```
hyperpod get-job --job-name hyperpod-cli-test
```

Example response:

```
{
  "Name": "hyperpod-cli-test",
  "Namespace": "hyperpod-ns-test-team",
  "Label": {
    "app": "hyperpod-cli-test",
    "app.kubernetes.io/managed-by": "Helm",
    "kueue.x-k8s.io/priority-class": "training"
  },
}
```

```
"CreationTimestamp": "2024-11-18T21:21:15Z",
"Status": {
    "completionTime": "2024-11-18T21:25:24Z",
    "conditions": [
        {
            "lastTransitionTime": "2024-11-18T21:21:15Z",
            "lastUpdateTime": "2024-11-18T21:21:15Z",
            "message": "PyTorchJob hyperpod-cli-test is created.",
            "reason": "PyTorchJobCreated",
            "status": "True",
            "type": "Created"
        },
        {
            "lastTransitionTime": "2024-11-18T21:21:17Z",
            "lastUpdateTime": "2024-11-18T21:21:17Z",
            "message": "PyTorchJob hyperpod-ns-test-team/hyperpod-cli-test is
running.",
            "reason": "PyTorchJobRunning",
            "status": "False",
            "type": "Running"
        },
        {
            "lastTransitionTime": "2024-11-18T21:25:24Z",
            "lastUpdateTime": "2024-11-18T21:25:24Z",
            "message": "PyTorchJob hyperpod-ns-test-team/hyperpod-cli-test
successfully completed.",
            "reason": "PyTorchJobSucceeded",
            "status": "True",
            "type": "Succeeded"
        }
    ],
    "replicaStatuses": {
        "Worker": {
            "selector": "training.kubeflow.org/job-name=hyperpod-cli-
test,training.kubeflow.org/operator-name=pytorchjob-controller,training.kubeflow.org/
replica-type=worker",
            "succeeded": 1
        }
    },
    "startTime": "2024-11-18T21:21:15Z"
},
"ConsoleURL": "https://us-west-2.console.aws.amazon.com/sagemaker/home?region=us-
west-2#/cluster-management/hyperpod-eks-test-cluster-id"
```

{}

## Suspend and unsuspend jobs

If you want to remove some submitted job from the scheduler, HyperPod AWS CLI provides suspend command to temporarily remove the job from orchestration. The suspended job will no longer be scheduled unless the job is manually unsuspended by the unsuspend command.

To temporarily suspend a job:

```
hyperpod patch-job suspend --job-name hyperpod-cli-test
```

To add a job back to the queue:

```
hyperpod patch-job unsuspend --job-name hyperpod-cli-test
```

## Debugging jobs

The HyperPod AWS CLI also provides other commands for you to debug job submission issues. For example `list-pods` and `get-logs` in the HyperPod AWS CLI Github repository.

## Troubleshoot

The following page contains known solutions for troubleshooting your HyperPod EKS clusters.

### Topics

- [Dashboard tab](#)
- [Tasks tab](#)
- [Policies](#)

## Dashboard tab

### The EKS add-on fails to install

For the EKS add-on installation to succeed, you will need to have a Kubernets version  $\geq 1.30$ . To update, see [Update Kubernetes version](#).

For the EKS add-on installation to succeed, all of the nodes need to be in **Ready** status and all of the pods need to be in **Running** status.

To check the status of your nodes, use the [list-cluster-nodes](#) AWS CLI command or navigate to your EKS cluster in the [EKS console](#) and view the status of your nodes. Resolve the issue for each node or reach out to your administrator. If the node status is **Unknown**, delete the node. Once all nodes statuses are **Ready**, retry installing the EKS add-on in HyperPod from the [Amazon SageMaker AI console](#).

To check the status of your pods, use the [Kubernetes CLI](#) command `kubectl get pods -n cloudwatch-agent` or navigate to your EKS cluster in the [EKS console](#) and view the status of your pods with the namespace `cloudwatch-agent`. Resolve the issue for the pods or reach out to your administrator to resolve the issues. Once all pod statuses are **Running**, retry installing the EKS add-on in HyperPod from the [Amazon SageMaker AI console](#).

For more troubleshooting, see [Troubleshooting the Amazon CloudWatch Observability EKS add-on](#).

## Tasks tab

If you see the error message about how the **Custom Resource Definition (CRD) is not configured on the cluster**, grant `EKSAdminViewPolicy` and `ClusterAccessRole` policies to your domain execution role.

- For information on how to get your execution role, see [Get your execution role](#).
- To learn how to attach policies to an IAM user or group, see [Adding and removing IAM identity permissions](#).

## Policies

The following lists solutions to errors relating to policies using the HyperPod APIs or console.

- If the policy is in `CreateFailed` or `CreateRollbackFailed` status, you need to delete the failed policy and create a new one.
- If the policy is in `UpdateFailed` status, retry the update with the same policy ARN.
- If the policy is in `UpdateRollbackFailed` status, you need to delete the failed policy and then create a new one.
- If the policy is in `DeleteFailed` or `DeleteRollbackFailed` status, retry the delete with the same policy ARN.
  - If you ran into an error while trying to delete the **Compute prioritization**, or cluster policy, using the HyperPod console, try to delete the `cluster-scheduler-config` using the API. To check the status of the resource, go to the details page of a compute allocation.

To see more details into the failure, use the describe API.

## Attribution document for Amazon SageMaker HyperPod task governance

In the following you can learn about attributions and third-party licenses for material used in Amazon SageMaker HyperPod task governance.

### Topics

- [base-files](#)
- [netbase](#)
- [golang-lru](#)

### [base-files](#)

This is the Debian prepackaged version of the Debian Base System Miscellaneous files. These files were written by Ian Murdock <imurdock@debian.org> and Bruce Perens <bruce@pixar.com>.

This package was first put together by Bruce Perens <Bruce@Pixar.com>, from his own sources.

The GNU Public Licenses in /usr/share/common-licenses were taken from ftp.gnu.org and are copyrighted by the Free Software Foundation, Inc.

The Artistic License in /usr/share/common-licenses is the one coming from Perl and its SPDX name is "Artistic License 1.0 (Perl)".

Copyright © 1995-2011 Software in the Public Interest.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

On Debian systems, the complete text of the GNU General

Public License can be found in `/usr/share/common-licenses/GPL'.

## netbase

Format: <https://www.debian.org/doc/packaging-manuals/copyright-format/1.0/>

Comment:

This package was created by Peter Tobias [tobias@et-inf.fho-emden.de](mailto:tobias@et-inf.fho-emden.de) on  
Wed, 24 Aug 1994 21:33:28 +0200 and maintained by Anthony Towns  
[ajt@debian.org](mailto:ajt@debian.org) until 2001.

It is currently maintained by Marco d'Itri <[md@linux.it](mailto:md@linux.it)>.

Files: \*

Copyright:

Copyright © 1994-1998 Peter Tobias  
Copyright © 1998-2001 Anthony Towns  
Copyright © 2002-2022 Marco d'Itri

License: GPL-2

This program is free software; you can redistribute it and/or modify  
it under the terms of the GNU General Public License, version 2, as  
published by the Free Software Foundation.

.

This program is distributed in the hope that it will be useful,  
but WITHOUT ANY WARRANTY; without even the implied warranty of  
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the  
GNU General Public License for more details.

.

You should have received a copy of the GNU General Public License along  
with this program; if not, write to the Free Software Foundation,  
Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

.

On Debian systems, the complete text of the GNU General Public License  
version 2 can be found in '/usr/share/common-licenses/GPL-2'.

## golang-lru

Copyright © 2014 HashiCorp, Inc.

Mozilla Public License, version 2.0

1. Definitions

1.1. "Contributor"

means each individual or legal entity that creates, contributes to the creation of, or owns Covered Software.

#### 1.2. "Contributor Version"

means the combination of the Contributions of others (if any) used by a Contributor and that particular Contributor's Contribution.

#### 1.3. "Contribution"

means Covered Software of a particular Contributor.

#### 1.4. "Covered Software"

means Source Code Form to which the initial Contributor has attached the notice in Exhibit A, the Executable Form of such Source Code Form, and Modifications of such Source Code Form, in each case including portions thereof.

#### 1.5. "Incompatible With Secondary Licenses"

means

- a. that the initial Contributor has attached the notice described in Exhibit B to the Covered Software; or
- b. that the Covered Software was made available under the terms of version 1.1 or earlier of the License, but not also under the terms of a Secondary License.

#### 1.6. "Executable Form"

means any form of the work other than Source Code Form.

#### 1.7. "Larger Work"

means a work that combines Covered Software with other material, in a separate file or files, that is not Covered Software.

#### 1.8. "License"

means this document.

#### 1.9. "Licensable"

means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently, any and all of the rights conveyed by this License.

#### 1.10. "Modifications"

means any of the following:

- a. any file in Source Code Form that results from an addition to, deletion from, or modification of the contents of Covered Software; or
- b. any new file in Source Code Form that contains any Covered Software.

#### 1.11. "Patent Claims" of a Contributor

means any patent claim(s), including without limitation, method, process, and apparatus claims, in any patent Licensable by such Contributor that would be infringed, but for the grant of the License, by the making, using, selling, offering for sale, having made, import, or transfer of either its Contributions or its Contributor Version.

#### 1.12. "Secondary License"

means either the GNU General Public License, Version 2.0, the GNU Lesser General Public License, Version 2.1, the GNU Affero General Public License, Version 3.0, or any later versions of those licenses.

#### 1.13. "Source Code Form"

means the form of the work preferred for making modifications.

#### 1.14. "You" (or "Your")

means an individual or a legal entity exercising rights under this License. For legal entities, "You" includes any entity that controls, is controlled by, or is under common control with You. For purposes of this definition, "control" means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

## 2. License Grants and Conditions

## 2.1. Grants

Each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license:

- a. under intellectual property rights (other than patent or trademark) Licensable by such Contributor to use, reproduce, make available, modify, display, perform, distribute, and otherwise exploit its Contributions, either on an unmodified basis, with Modifications, or as part of a Larger Work; and
- b. under Patent Claims of such Contributor to make, use, sell, offer for sale, have made, import, and otherwise transfer either its Contributions or its Contributor Version.

## 2.2. Effective Date

The licenses granted in Section 2.1 with respect to any Contribution become effective for each Contribution on the date the Contributor first distributes such Contribution.

## 2.3. Limitations on Grant Scope

The licenses granted in this Section 2 are the only rights granted under this License. No additional rights or licenses will be implied from the distribution or licensing of Covered Software under this License.

Notwithstanding Section 2.1(b) above, no patent license is granted by a Contributor:

- a. for any code that a Contributor has removed from Covered Software; or
- b. for infringements caused by: (i) Your and any other third party's modifications of Covered Software, or (ii) the combination of its Contributions with other software (except as part of its Contributor Version); or
- c. under Patent Claims infringed by Covered Software in the absence of its Contributions.

This License does not grant any rights in the trademarks, service marks, or logos of any Contributor (except as may be necessary to comply with the notice requirements in Section 3.4).

## 2.4. Subsequent Licenses

No Contributor makes additional grants as a result of Your choice to distribute the Covered Software under a subsequent version of this License (see Section 10.2) or under the terms of a Secondary License (if permitted under the terms of Section 3.3).

## 2.5. Representation

Each Contributor represents that the Contributor believes its Contributions are its original creation(s) or it has sufficient rights to grant the rights to its Contributions conveyed by this License.

## 2.6. Fair Use

This License is not intended to limit any rights You have under applicable copyright doctrines of fair use, fair dealing, or other equivalents.

## 2.7. Conditions

Sections 3.1, 3.2, 3.3, and 3.4 are conditions of the licenses granted in Section 2.1.

## 3. Responsibilities

### 3.1. Distribution of Source Form

All distribution of Covered Software in Source Code Form, including any Modifications that You create or to which You contribute, must be under the terms of this License. You must inform recipients that the Source Code Form of the Covered Software is governed by the terms of this License, and how they can obtain a copy of this License. You may not attempt to alter or restrict the recipients' rights in the Source Code Form.

### 3.2. Distribution of Executable Form

If You distribute Covered Software in Executable Form then:

- a. such Covered Software must also be made available in Source Code Form, as described in Section 3.1, and You must inform recipients of the Executable Form how they can obtain a copy of such Source Code Form by reasonable means in a timely manner, at a charge no more than the cost

- of distribution to the recipient; and
- b. You may distribute such Executable Form under the terms of this License, or sublicense it under different terms, provided that the license for the Executable Form does not attempt to limit or alter the recipients' rights in the Source Code Form under this License.

### 3.3. Distribution of a Larger Work

You may create and distribute a Larger Work under terms of Your choice, provided that You also comply with the requirements of this License for the Covered Software. If the Larger Work is a combination of Covered Software with a work governed by one or more Secondary Licenses, and the Covered Software is not Incompatible With Secondary Licenses, this License permits You to additionally distribute such Covered Software under the terms of such Secondary License(s), so that the recipient of the Larger Work may, at their option, further distribute the Covered Software under the terms of either this License or such Secondary License(s).

### 3.4. Notices

You may not remove or alter the substance of any license notices (including copyright notices, patent notices, disclaimers of warranty, or limitations of liability) contained within the Source Code Form of the Covered Software, except that You may alter any license notices to the extent required to remedy known factual inaccuracies.

### 3.5. Application of Additional Terms

You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Software. However, You may do so only on Your own behalf, and not on behalf of any Contributor. You must make it absolutely clear that any such warranty, support, indemnity, or liability obligation is offered by You alone, and You hereby agree to indemnify every Contributor for any liability incurred by such Contributor as a result of warranty, support, indemnity or liability terms You offer. You may include additional disclaimers of warranty and limitations of liability specific to any jurisdiction.

## 4. Inability to Comply Due to Statute or Regulation

If it is impossible for You to comply with any of the terms of this License

with respect to some or all of the Covered Software due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be placed in a text file included with all distributions of the Covered Software under this License. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it.

## 5. Termination

- 5.1. The rights granted under this License will terminate automatically if You fail to comply with any of its terms. However, if You become compliant, then the rights granted under this License from a particular Contributor are reinstated (a) provisionally, unless and until such Contributor explicitly and finally terminates Your grants, and (b) on an ongoing basis, if such Contributor fails to notify You of the non-compliance by some reasonable means prior to 60 days after You have come back into compliance. Moreover, Your grants from a particular Contributor are reinstated on an ongoing basis if such Contributor notifies You of the non-compliance by some reasonable means, this is the first time You have received notice of non-compliance with this License from such Contributor, and You become compliant prior to 30 days after Your receipt of the notice.
- 5.2. If You initiate litigation against any entity by asserting a patent infringement claim (excluding declaratory judgment actions, counter-claims, and cross-claims) alleging that a Contributor Version directly or indirectly infringes any patent, then the rights granted to You by any and all Contributors for the Covered Software under Section 2.1 of this License shall terminate.
- 5.3. In the event of termination under Sections 5.1 or 5.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or Your distributors under this License prior to termination shall survive termination.

## 6. Disclaimer of Warranty

Covered Software is provided under this License on an "as is" basis, without warranty of any kind, either expressed, implied, or statutory, including, without limitation, warranties that the Covered Software is free of defects, merchantable, fit for a particular purpose or non-infringing. The entire risk as to the quality and performance of the Covered Software

is with You. Should any Covered Software prove defective in any respect, You (not any Contributor) assume the cost of any necessary servicing, repair, or correction. This disclaimer of warranty constitutes an essential part of this License. No use of any Covered Software is authorized under this License except under this disclaimer.

## 7. Limitation of Liability

Under no circumstances and under no legal theory, whether tort (including negligence), contract, or otherwise, shall any Contributor, or anyone who distributes Covered Software as permitted above, be liable to You for any direct, indirect, special, incidental, or consequential damages of any character including, without limitation, damages for lost profits, loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses, even if such party shall have been informed of the possibility of such damages. This limitation of liability shall not apply to liability for death or personal injury resulting from such party's negligence to the extent applicable law prohibits such limitation. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so this exclusion and limitation may not apply to You.

## 8. Litigation

Any litigation relating to this License may be brought only in the courts of a jurisdiction where the defendant maintains its principal place of business and such litigation shall be governed by laws of that jurisdiction, without reference to its conflict-of-law provisions. Nothing in this Section shall prevent a party's ability to bring cross-claims or counter-claims.

## 9. Miscellaneous

This License represents the complete agreement concerning the subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not be used to construe this License against a Contributor.

## 10. Versions of the License

### 10.1. New Versions

Mozilla Foundation is the license steward. Except as provided in Section 10.3, no one other than the license steward has the right to modify or publish new versions of this License. Each version will be given a distinguishing version number.

#### 10.2. Effect of New Versions

You may distribute the Covered Software under the terms of the version of the License under which You originally received the Covered Software, or under the terms of any subsequent version published by the license steward.

#### 10.3. Modified Versions

If you create software not governed by this License, and you want to create a new license for such software, you may create and use a modified version of this License if you rename the license and remove any references to the name of the license steward (except to note that such modified license differs from this License).

#### 10.4. Distributing Source Code Form that is Incompatible With Secondary Licenses If You choose to distribute Source Code Form that is Incompatible With Secondary Licenses under the terms of this version of the License, the notice described in Exhibit B of this License must be attached.

##### Exhibit A - Source Code Form License Notice

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at <http://mozilla.org/MPL/2.0/>.

If it is not possible or desirable to put the notice in a particular file, then You may include the notice in a location (such as a LICENSE file in a relevant directory) where a recipient would be likely to look for such a notice.

You may add additional accurate notices of copyright ownership.

##### Exhibit B - "Incompatible With Secondary Licenses" Notice

This Source Code Form is "Incompatible With Secondary Licenses", as defined by the Mozilla Public License, v. 2.0.

## Delete a SageMaker HyperPod cluster

Use the following instructions to delete SageMaker HyperPod clusters orchestrated by Amazon EKS in the SageMaker AI console.

1. Under **Clusters** in the main pane of the SageMaker HyperPod console, choose the cluster you want to delete.
2. Select your cluster, and choose **Delete**.
3. In the pop-up window for cluster deletion, review the cluster information carefully to confirm that you chose the right cluster to delete.
4. After you reviewed the cluster information, choose **Yes, delete cluster**.
5. In the text field to confirm this deletion, type **delete**.
6. Choose **Delete** on the lower right corner of the pop-up window to finish sending the cluster deletion request.

## Managing SageMaker HyperPod clusters using the AWS CLI

The following topics provide guidance on writing SageMaker HyperPod API request files in JSON format and run them using the AWS CLI commands.

### Topics

- [Create a SageMaker HyperPod cluster](#)
- [Retrieve SageMaker HyperPod cluster details](#)
- [Update SageMaker HyperPod cluster configuration](#)
- [Update the SageMaker HyperPod platform software](#)
- [Access SageMaker HyperPod cluster nodes](#)
- [Scale down a SageMaker HyperPod cluster](#)
- [Delete a SageMaker HyperPod cluster](#)

## Create a SageMaker HyperPod cluster

Learn how to create SageMaker HyperPod clusters orchestrated by Amazon EKS using the AWS CLI.

1. Before creating an SageMaker HyperPod cluster:
  - a. Ensure that you have an existing Amazon EKS cluster up and running. For detailed instructions about how to set up an Amazon EKS cluster, see [Create an Amazon EKS cluster](#) in the *Amazon EKS User Guide*.
  - b. Install the Helm chart as instructed in [the section called “Install packages on the Amazon EKS cluster using Helm”](#).
2. Prepare a lifecycle configuration script and upload to an Amazon S3 bucket, such as `s3://amzn-s3-demo-bucket/Lifecycle-scripts/base-config/`.

For a quick start, download the sample script [on\\_create.sh](#) from the AWSome Distributed Training GitHub repository, and upload it to the S3 bucket. This script sets up the logging file `/var/log/provision/provisioning.log` required for CloudWatch to gather logs from Pod containers. You can also include additional setup instructions, a series of setup scripts, or commands to be executed during the HyperPod cluster provisioning stage.

 **Important**

If you create an [the section called “IAM role for SageMaker HyperPod”](#) attaching only the managed [AmazonSageMakerClusterInstanceRolePolicy](#), your cluster has access to Amazon S3 buckets with the specific prefix `sagemaker-`.

3. Prepare a [CreateCluster](#) API request file in JSON format. For `ExecutionRole`, provide the ARN of the IAM role you created with the managed `AmazonSageMakerClusterInstanceRolePolicy` from the section [the section called “IAM role for SageMaker HyperPod”](#).

 **Note**

Ensure that your SageMaker HyperPod cluster is deployed within the same Virtual Private Cloud (VPC) as your Amazon EKS cluster. The subnets and security groups specified in the SageMaker HyperPod cluster configuration must allow network connectivity and communication with the Amazon EKS cluster's API server endpoint.

```
// create_cluster.json
{
    "ClusterName": "string",
    "InstanceGroups": [
        {
            "InstanceGroupName": "string",
            "InstanceType": "string",
            "InstanceCount": number,
            "LifeCycleConfig": {
                "SourceS3Uri": "s3://amzn-s3-demo-bucket-sagemaker/<Lifecycle-script-directory>/src/",
                "OnCreate": "on_create.sh"
            },
            "ExecutionRole": "string",
            "ThreadsPerCore": number,
            "OnStartDeepHealthChecks": [
                "InstanceStress", "InstanceConnectivity"
            ]
        }
    ],
    "VpcConfig": {
        "SecurityGroupIds": ["string"],
        "Subnets": ["string"]
    },
    "Tags": [
        {
            "Key": "string",
            "Value": "string"
        }
    ],
    "Orchestrator": {
        "Eks": {
            "ClusterArn": "string",
        }
    },
    "NodeRecovery": "Automatic"
}
```

Note the following when configuring to create a new SageMaker HyperPod cluster associating with an EKS cluster.

- You can configure up to 20 instance groups under the `InstanceGroups` parameter.
- For `Orchestrator.Eks.ClusterArn`, specify the ARN of the EKS cluster you want to use as the orchestrator.

- For OnStartDeepHealthChecks, add InstanceStress and InstanceConnectivity to enable [the section called “Deep health checks”](#).
  - For NodeRecovery, specify Automatic to enable automatic node recovery. SageMaker HyperPod replaces or reboots instances (nodes) when issues are found by the health-monitoring agent.
  - For the Tags parameter, you can add custom tags for managing the SageMaker HyperPod cluster as an AWS resource. You can add tags to your cluster in the same way you add them in other AWS services that support tagging. To learn more about tagging AWS resources in general, see [Tagging AWS Resources User Guide](#).
  - For the VpcConfig parameter, specify the information of the VPC used in the EKS cluster. The subnets must be private.
4. Run the [create-cluster](#) command as follows.

**⚠ Important**

When running the `create-cluster` command with the `--cli-input-json` parameter, you must include the `file://` prefix before the complete path to the JSON file. This prefix is required to ensure that the AWS CLI recognizes the input as a file path. Omitting the `file://` prefix results in a parsing parameter error.

```
aws sagemaker create-cluster \
--cli-input-json file://complete/path/to/create_cluster.json
```

This should return the ARN of the new cluster.

## Retrieve SageMaker HyperPod cluster details

Learn how to retrieve SageMaker HyperPod cluster details using the AWS CLI.

### Describe a cluster

Run [describe-cluster](#) to check the status of the cluster. You can specify either the name or the ARN of the cluster.

```
aws sagemaker describe-cluster --cluster-name your-hyperpod-cluster
```

After the status of the cluster turns to **InService**, proceed to the next step. Using this API, you can also retrieve failure messages from running other HyperPod API operations.

## List details of cluster nodes

Run [list-cluster-nodes](#) to check the key information of the cluster nodes.

```
aws sagemaker list-cluster-nodes --cluster-name your-hyperpod-cluster
```

This returns a response, and the InstanceId is what you need to use for logging (using aws ssm) into them.

## Describe details of a cluster node

Run [describe-cluster-node](#) to retrieve details of a cluster node. You can get the cluster node ID from list-cluster-nodes output. You can specify either the name or the ARN of the cluster.

```
aws sagemaker describe-cluster-node \  
  --cluster-name your-hyperpod-cluster \  
  --node-id i-11122233344455aa
```

## List clusters

Run [list-clusters](#) to list all clusters in your account.

```
aws sagemaker list-clusters
```

You can also add additional flags to filter the list of clusters down. To learn more about what this command runs at low level and additional flags for filtering, see the [ListClusters](#) API reference.

## Update SageMaker HyperPod cluster configuration

Run [update-cluster](#) to update the configuration of a cluster.

### Note

Important considerations:

- You cannot change the EKS cluster information that your HyperPod cluster is associated after the cluster is created.

- If deep health checks are running on the cluster, this API will not function as expected. You might encounter an error message stating that deep health checks are in progress. To update the cluster, you should wait until the deep health checks finish.

1. Create an [UpdateCluster](#) API request file in JSON format. Make sure that you specify the right cluster name and instance group name to update. For each instance group, you can change the instance type, the number of instances, the lifecycle configuration entrypoint script, and the path to the script.

 **Note**

You can use the `UpdateCluster` to scale down or remove entire instance groups from your SageMaker HyperPod cluster. For additional instructions on how to scale down or delete instance groups, see [the section called “Scale down a SageMaker HyperPod cluster”](#).

- a. For `ClusterName`, specify the name of the cluster you want to update.
- b. For `InstanceGroupName`
  - i. To update an existing instance group, specify the name of the instance group you want to update.
  - ii. To add a new instance group, specify a new name not existing in your cluster.
- c. For `InstanceType`
  - i. To update an existing instance group, you must match the instance type you initially specified to the group.
  - ii. To add a new instance group, specify an instance type you want to configure the group with.
- d. For `InstanceCount`
  - i. To update an existing instance group, specify an integer that corresponds to your desired number of instances. You can provide a higher or lower value (down to 0) to scale the instance group up or down.
  - ii. To add a new instance group, specify an integer greater or equal to 1.
- e. For `LifeCycleConfig`, you can change the values for both `SourceS3Uri` and `OnCreate` as you want to update the instance group.

f. For ExecutionRole

- i. For updating an existing instance group, keep using the same IAM role you attached during cluster creation.
- ii. For adding a new instance group, specify an IAM role you want to attach.

g. For ThreadsPerCore

- i. For updating an existing instance group, keep using the same value you specified during cluster creation.
  - ii. For adding a new instance group, you can choose any value from the allowed options per instance type. For more information, search the instance type and see the **Valid threads per core** column in the reference table at [CPU cores and threads per CPU core per instance type](#) in the *Amazon EC2 User Guide*.
- h. For OnStartDeepHealthChecks, add InstanceStress and InstanceConnectivity to enable [the section called “Deep health checks”](#).
- i. For NodeRecovery, specify Automatic to enable automatic node recovery. SageMaker HyperPod replaces or reboots instances (nodes) when issues are found by the health-monitoring agent.

The following code snippet is a JSON request file template you can use. For more information about the request syntax and parameters of this API, see the [UpdateCluster](#) API reference.

```
// update_cluster.json
{
    // Required
    "ClusterName": "name-of-cluster-to-update",
    // Required
    "InstanceGroups": [
        {
            "InstanceGroupName": "string",
            "InstanceType": "string",
            "InstanceCount": number,
            "LifeCycleConfig": {
                "SourceS3Uri": "string",
                "OnCreate": "string"
            },
            "ExecutionRole": "string",
            "ThreadsPerCore": number,
            "OnStartDeepHealthChecks": [
                "InstanceStress", "InstanceConnectivity"
            ]
        }
    ]
}
```

```
  }],
  "NodeRecovery": "Automatic"
}
```

## 2. Run the following update-cluster command to submit the request.

```
aws sagemaker update-cluster \
--cli-input-json file://complete/path/to/update_cluster.json
```

## Update the SageMaker HyperPod platform software

When you create your SageMaker HyperPod cluster, SageMaker HyperPod selects an Amazon Machine Image (AMI) corresponding to the Kubernetes version of your Amazon EKS cluster.

Run [update-cluster-software](#) to update existing clusters with software and security patches provided by the SageMaker HyperPod service. For `--cluster-name`, specify either the name or the ARN of the cluster to update.

### Important

- When this API is called, SageMaker HyperPod doesn't drain or redistribute the jobs (Pods) running on the nodes. Make sure to check if there are any jobs running on the nodes before calling this API.
- The patching process replaces the root volume with the updated AMI, which means that your previous data stored in the instance root volume will be lost. Make sure that you back up your data from the instance root volume to Amazon S3 or Amazon FSx for Lustre.
- All cluster nodes experience downtime (nodes appear as `<NotReady>` in the output of `kubectl get node`) while the patching is in progress. We recommend that you terminate all workloads before patching and resume them after the patch completes.

If the security patch fails, you can retrieve failure messages by running the [DescribeCluster](#) API as instructed at [the section called “Describe a cluster”](#).

```
aws sagemaker update-cluster-software --cluster-name your-hyperpod-cluster
```

When calling the `UpdateClusterSoftware` API, SageMaker HyperPod updates the Kubernetes version of the nodes by selecting the latest [the section called “SageMaker HyperPod DLAMI”](#) based on the Kubernetes version of your Amazon EKS cluster. It then runs the lifecycle scripts in the Amazon S3 bucket that you specified during the cluster creation or update.

You can verify the kubelet version of a node by running the `kubectl describe node` command.

The Kubernetes version of SageMaker HyperPod cluster nodes does not automatically update when you update your Amazon EKS cluster version. After updating the Kubernetes version for your Amazon EKS cluster, you must use the `UpdateClusterSoftware` API to update your SageMaker HyperPod cluster nodes to the same Kubernetes version.

It is recommended to update your SageMaker HyperPod cluster after updating your Amazon EKS nodes, and avoid having more than one version difference between the Amazon EKS cluster version and the SageMaker HyperPod cluster nodes version.

The SageMaker HyperPod service team regularly rolls out new [the section called “SageMaker HyperPod DLAMI”](#)s for enhancing security and improving user experiences. We recommend that you always keep updating to the latest SageMaker HyperPod DLAMI. For future SageMaker HyperPod DLAMI updates for security patching, follow up with [the section called “HyperPod release notes”](#).

#### Note

You can only run this API programmatically. The patching functionality is not implemented in the SageMaker HyperPod console UI.

## Access SageMaker HyperPod cluster nodes

You can directly access the nodes of a SageMaker HyperPod cluster in service using the AWS CLI commands for AWS Systems Manager (SSM). Run `aws ssm start-session` with the host name of the node in format of `sagemaker-cluster:[cluster-id]_[instance-group-name]-[instance-id]`. You can retrieve the cluster ID, the instance ID, and the instance group name from the [SageMaker HyperPod console](#) or by running `describe-cluster` and `list-cluster-nodes` from the [AWS CLI commands for SageMaker HyperPod](#). For example, if your cluster ID is `aa11bbbbbb222`, the cluster node name is `controller-group`, and the cluster node ID is `i-111222333444555aa`, the SSM `start-session` command should be the following.

**Note**

If you haven't set up AWS Systems Manager, follow the instructions provided at [the section called "Setting up AWS Systems Manager and Run As for cluster user access control"](#).

```
$ aws ssm start-session \
  --target sagemaker-cluster:aa11bbbb222_controller-group-i-111222333444555aa \
  --region us-west-2
Starting session with SessionId: s0011223344aabccdd
root@ip-111-22-333-444:/usr/bin#
```

## Scale down a SageMaker HyperPod cluster

You can scale down the number of instances running on your Amazon SageMaker HyperPod cluster. You might want to scale down a cluster for various reasons, such as reduced resource utilization or cost optimization.

The following page outlines two main approaches to scaling down:

- **Scale down at the instance group level:** This approach uses the `UpdateCluster` API, with which you can:
  - Scale down the instance counts for specific instance groups independently. SageMaker AI handles the termination of nodes in a way that reaches the new target instance counts you've set for each group. See [the section called "Scale down an instance group"](#).
  - Completely delete instance groups from your cluster. See [the section called "Delete instance groups"](#).
- **Scale down at the instance level:** This approach uses the `BatchDeleteClusterNodes` API, with which you can specify the individual nodes you want to terminate. See [the section called "Scale down at the instance level"](#).

**Note**

When scaling down at the instance level with `BatchDeleteClusterNodes`, you can only terminate a maximum of 99 instances at a time. `UpdateCluster` supports terminating any number of instances.

## Important considerations

- When scaling down a cluster, you should ensure that the remaining resources are sufficient to handle your workload and that any necessary data migration or rebalancing is properly handled to avoid disruptions.
- Make sure to back up your data to Amazon S3 or an FSx for Lustre file system before invoking the API on a worker node group. This can help prevent any potential data loss from the instance root volume. For more information about backup, see [Use the backup script provided by SageMaker HyperPod](#).
- To invoke this API on an existing cluster, you must first patch the cluster by running the [UpdateClusterSoftware](#) API. For more information about patching a cluster, see [Update the SageMaker HyperPod platform software of a cluster](#).
- Metering/billing for on-demand instances will automatically be stopped after scale down. To stop metering for scaled-down reserved instances, you should reach out to your AWS account team for support.
- You can use the released capacity from the scaled-down reserved instances to scale up another SageMaker HyperPod cluster.

## Scale down at the instance group level

The [UpdateCluster](#) operation allows you to make changes to the configuration of your SageMaker HyperPod cluster, such as scaling down the number of instances of an instance group or removing entire instance groups. This can be useful when you want to adjust the resources allocated to your cluster based on changes in your workload, optimize costs, or change the instance type of an instance group.

### Scale down an instance group

Use this approach when you have an instance group that is idle and it's safe to terminate any of the instances for scaling down. When you submit an `UpdateCluster` request to scale down, HyperPod randomly chooses instances for termination and scales down to the specified number of nodes for the instance group.

#### Note

When you scale the number of instances in an instance group down to 0, all the instances within that group will be terminated. However, the instance group itself will still exist as

part of the SageMaker HyperPod cluster. You can scale the instance group back up at a later time, using the same instance group configuration. Alternatively, you can choose to remove an instance group permanently. For more information, see [the section called “Delete instance groups”](#).

## To scale down with `UpdateCluster`

1. Follow the steps outlined in [Update SageMaker HyperPod cluster configuration](#). When you reach step 1.d where you specify the **InstanceCount** field, enter a number that is smaller than the current number of instances to scale down the cluster.
2. Run the [update-cluster](#) AWS CLI command to submit your request.

The following is an example of an `UpdateCluster` JSON object. Consider the case where your instance group currently has 2 running instances. If you set the **InstanceCount** field to 1, as shown in the example, then HyperPod randomly selects one of the instances and terminates it.

```
{  
  "ClusterName": "name-of-cluster-to-update",  
  "InstanceGroups": [  
    {  
      "InstanceGroupName": "training-instances",  
      "InstanceType": "instance-type",  
      "InstanceCount": 1,  
      "LifeCycleConfig": {  
        "SourceS3Uri": "s3://amzn-s3-demo-bucket/training-script.py",  
        "OnCreate": "s3://amzn-s3-demo-bucket/setup-script.sh"  
      },  
      "ExecutionRole": "arn:aws:iam::123456789012:role/SageMakerRole",  
      "ThreadsPerCore": number-of-threads,  
      "OnStartDeepHealthChecks": [  
        "InstanceStress",  
        "InstanceConnectivity"  
      ]  
    }  
  ],  
  "NodeRecovery": "Automatic"  
}
```

## Delete instance groups

You can use the [UpdateCluster](#) operation to remove entire instance groups from your SageMaker HyperPod cluster when they are no longer needed. This goes beyond simple scaling down, allowing you to completely eliminate specific instance groups from your cluster's configuration.

### Note

When removing an instance group:

- All instances within the targeted group are terminated.
- The entire group configuration is deleted from the cluster.
- Any workloads running on that instance group are stopped.

## To delete instance groups with [UpdateCluster](#)

1. When following the steps outlined in [Update SageMaker HyperPod cluster configuration](#):
  - a. Set the optional `InstanceGroupsToDelete` parameter in your `UpdateCluster` JSON and pass the comma-separated list of instance group names that you want to delete.
  - b. When you specify the `InstanceGroups` list, ensure that the specifications of the instance groups you are removing are no longer listed in the `InstanceGroups` list.
2. Run the [update-cluster](#) AWS CLI command to submit your request.

### Important

- Your SageMaker HyperPod cluster must always maintain at least one instance group.
- Ensure all critical data is backed up before removal.
- The removal process cannot be undone.

The following is an example of an `UpdateCluster` JSON object. Consider the case where a cluster currently has 3 instance groups, a *training*, a *prototype-training*, and an *inference-serving* group. You want to delete the *prototype-training* group.

```
{  
  "ClusterName": "name-of-cluster-to-update",  
  "InstanceGroups": [  
    {  
      "InstanceGroupName": "training",  
      "InstanceType": "instance-type",  
      "InstanceCount": ,  
      "LifeCycleConfig": {  
        "SourceS3Uri": "s3://amzn-s3-demo-bucket/training-script.py",  
        "OnCreate": "s3://amzn-s3-demo-bucket/setup-script.sh"  
      },  
      "ExecutionRole": "arn:aws:iam::123456789012:role/SageMakerRole",  
      "ThreadsPerCore": number-of-threads,  
      "OnStartDeepHealthChecks": [  
        "InstanceStress",  
        "InstanceConnectivity"  
      ],  
    },  
    {  
      "InstanceGroupName": "inference-serving",  
      "InstanceType": "instance-type",  
      "InstanceCount": 2,  
      [...]  
    },  
  ],  
  "InstanceGroupsToDelete": [ "prototype-training" ],  
  "NodeRecovery": "Automatic"  
}
```

## Scale down at the instance level

The `BatchDeleteClusterNodes` operation allows you to scale down a SageMaker HyperPod cluster by specifying the individual nodes you want to terminate. `BatchDeleteClusterNodes` provides more granular control for targeted node removal and cluster optimization. For example, you might use `BatchDeleteClusterNodes` to delete targeted nodes for maintenance, rolling upgrades, or rebalancing resources geographically.

### API request and response

When you submit a `BatchDeleteClusterNodes` request, SageMaker HyperPod deletes nodes by their instance IDs. The API accepts a request with the cluster name and a list of node IDs to be deleted.

The response includes two sections:

- Failed: A list of errors of type [BatchDeleteClusterNodesError](#) - one per instance ID.
- Successful: The list of instance IDs successfully terminated.

## Validation and error handling

The API performs various validations, such as:

- Verifying the node ID format (prefix of i- and Amazon EC2 instance ID structure).
- Checking the node list length, with a limit of 99 or fewer node IDs in a single BatchDeleteClusterNodes request.
- Ensuring a valid SageMaker HyperPod cluster with the input cluster-name is present and that no cluster-level operations (update, system update, patching, or deletion) are in progress.
- Handling cases where instances are not found, have invalid status, or are in use.

## API Response Codes

- The API returns a 200 status code for successful (e.g., all input nodes succeeded validation) or partially successful requests (e.g., some input nodes fail validation).
- If all of these validations fail (e.g., all input nodes fail validation), the API will return a 400 Bad Request response with the appropriate error messages and error codes.

## Example

The following is an example of **scaling down a cluster at the instance level** using the AWS CLI:

```
aws sagemaker batch-delete-cluster-nodes --cluster-name "cluster-name" --node-ids '["i-11111222233333", "i-11111222233333"]'
```

## Delete a SageMaker HyperPod cluster

Run [delete-cluster](#) to delete a cluster. You can specify either the name or the ARN of the cluster.

```
aws sagemaker delete-cluster --cluster-name your-hyperpod-cluster
```

This API only cleans up the SageMaker HyperPod resources and doesn't delete any resources of the associated EKS cluster. This includes the Amazon EKS cluster, EKS Pod identities, Amazon

FSx volumes, and EKS add-ons. This also includes the initial configuration you added to your EKS cluster. If you want to clean up all resources, make sure that you also clean up the EKS resources separately.

Make sure that you first delete the SageMaker HyperPod resources, followed by the EKS resources. Performing the deletion in the reverse order may result in lingering resources.

### **Important**

When this API is called, SageMaker HyperPod doesn't drain or redistribute the jobs (Pods) running on the nodes. Make sure to check if there are any jobs running on the nodes before calling this API.

## Configure storage for SageMaker HyperPod clusters orchestrated by Amazon EKS

Cluster admin needs to configure storage for data scientist users to manage input and output data and storing checkpoints during training on SageMaker HyperPod clusters.

### Handling large datasets (input/output data)

- **Data access and management:** Data scientists often work with large datasets that are required for training machine learning models. Specifying storage parameters in the job submission allows them to define where these datasets are located (e.g., Amazon S3 buckets, persistent volumes in Kubernetes) and how they are accessed during the job execution.
- **Performance optimization:** The efficiency of accessing input data can significantly impact the performance of the training job. By optimizing storage parameters, data scientists can ensure that data is read and written efficiently, reducing I/O bottlenecks.

### Storing checkpoints

- **Checkpointing in training:** During long-running training jobs, it's common practice to save checkpoints—intermediate states of the model. This allows data scientists to resume training from a specific point in case of a failure, rather than starting from scratch.
- **Data recovery and experimentation:** By specifying the storage location for checkpoints, data scientists can ensure that these checkpoints are securely stored, potentially in a distributed storage system that offers redundancy and high availability. This is crucial for recovering from interruptions and for experimenting with different training strategies.

**Tip**

For a hands-on experience and guidance on how to set up storage for SageMaker HyperPod cluster orchestrated with Amazon EKS, see the following sections in the [Amazon EKS Support in SageMaker HyperPod workshop](#).

- [Set up Amazon FSx for Lustre on SageMaker HyperPod](#)
- [Set up Amazon S3 on SageMaker HyperPod](#) using [Mountpoint for Amazon S3](#)

## Cluster resiliency features for SageMaker HyperPod cluster orchestration with Amazon EKS

SageMaker HyperPod provides the following cluster resiliency features.

### Topics

- [SageMaker HyperPod health-monitoring agent](#)
- [Basic health checks](#)
- [Deep health checks](#)
- [Automatic node recovery](#)
- [Resilience-related Kubernetes labels by SageMaker HyperPod](#)
- [Manually quarantine, replace, or reboot a node](#)
- [Suggested resilience configurations](#)

### SageMaker HyperPod health-monitoring agent

SageMaker HyperPod health-monitoring agent continuously monitors the health status of each GPU-based or Trainium-based instances. When it detects any instance or GPU failures, the agent marks the instance as unhealthy.

#### Health checks done by the SageMaker HyperPod health-monitoring agent

The SageMaker HyperPod health-monitoring agent checks the following.

#### NVIDIA GPUs

- [DCGM policy violation notifications](#)

- Errors in the nvidia-smi output
- Various errors in the logs generated by the Amazon Elastic Compute Cloud (EC2) platform

## AWS Trainium

- Errors in the output from the [AWS Neuron monitor](#)
- Outputs generated by the Neuron node problem detector (For more information about the AWS Neuron node problem detector, see [Node problem detection and recovery for AWS Neuron nodes within Amazon EKS clusters](#).)
- Various errors in the logs generated by the Amazon EC2 platform

## Logs generated by the SageMaker HyperPod health-monitoring agent

The SageMaker HyperPod health-monitoring agent is an out-of-the-box health check feature and continuously runs on all HyperPod clusters. The health monitoring agent publishes detected health events on GPU or Trn instances to CloudWatch under the Cluster log group /aws/sagemaker/Clusters/.

The detection logs from the HyperPod health monitoring agent are created as separate log streams named SagemakerHealthMonitoringAgent for each node. You can query the detection logs using CloudWatch log insights as follows.

```
fields @timestamp, @message
| filter @message like /HealthMonitoringAgentDetectionEvent/
```

This should return an output similar to the following.

```
2024-08-21T11:35:35.532-07:00
{"level":"info","ts":"2024-08-21T18:35:35Z","msg":"NPD caught event: %v","details":
":",
{"severity":"warn","timestamp":"2024-08-22T20:59:29Z","reason":"XidHardwareFailure","message":"
condition NvidiaErrorReboot is now: True, reason: XidHardwareFailure,
message: \"NVRM: Xid (PCI:0000:b9:00): 71, pid=<unknown>, name=<unknown>,
NVLink: fatal error detected on link 6(0x10000, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0)\\"", "HealthMonitoringAgentDetectionEvent":"HealthEvent"}
2024-08-21T11:35:35.532-07:00
 {"level":"info","ts":"2024-08-21T18:35:35Z","msg":"NPD caught event: %v","details":
":",
 {"severity":"warn","timestamp":"2024-08-22T20:59:29Z","reason":"XidHardwareFailure","message":"
```

```
condition NvidiaErrorReboot is now: True, reason: XidHardwareFailure,
message: \"NVRM: Xid (PCI:0000:b9:00): 71, pid=<unknown>, name=<unknown>,
NVLink: fatal error detected on link 6(0x10000, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0)\\"}, "HealthMonitoringAgentDetectionEvent": "HealthEvent"}
```

## Basic health checks

SageMaker HyperPod performs a set of *basic health checks* on cluster instances during the creation and update of HyperPod clusters. These basic health checks are orchestrator-agnostic, so these checks are applicable regardless of the underlying orchestration platforms supported by SageMaker HyperPod (Amazon EKS or Slurm).

The basic health checks monitor cluster instances for issues related to devices such as accelerators (GPU and Trainium cores) and network devices (Elastic Fabric Adapter, or EFA). To find the list of basic cluster health checks, see [Cluster health checks](#).

## Deep health checks

SageMaker HyperPod performs *deep health checks* on cluster instances during the creation and update of HyperPod clusters. The deep health checks ensure the reliability and stability of the SageMaker HyperPod clusters by thoroughly testing the underlying hardware and infrastructure components before allowing the clusters to be used for training machine learning models. This proactive approach helps identify and mitigate potential issues early in the cluster lifecycle.

### List of deep health checks done by SageMaker HyperPod

SageMaker HyperPod runs the following deep health checks.

#### Instance-level deep health checks

Category	Utility name	Instance type compatibility	Description
Accelerator	GPU/NVLink count	GPU	Verifies GPU/NVLink counts.
Accelerator	<a href="#">DCGM diagnostics level 4</a>	GPU	Assesses the health and functionality of NVIDIA GPUs by running DCGM

Category	Utility name	Instance type compatibility	Description
			(NVIDIA Data Center GPU Manager) diagnostics at level 4, including additional memory tests.
Accelerator	Neuron sysfs	Trainium	For Trainium-powered instances, the health of the Neuron devices is determined by reading counters from <a href="#">Neuron sysfs</a> propagated directly by the Neuron driver.
Accelerator	Neuron hardware check	Trainium	Runs a training workload to produce numbers. it then verifies aiming to test the hardware.
Accelerator	NCCOM local test	Trainium	Evaluates the performance of collective communication operations on single Trainium nodes
Network	EFA	GPU and Trainium	Runs latency and bandwidth benchmarking on the attached EFA device.

## Cluster-level deep health checks

Category	Utility name	Instance type compatibility	Description
Accelerator	NCCL test	GPU	Verifies the performance of collective communication operations on multiple NVIDIA GPUs
Accelerator	NCCOM cluster test	Trainium	Verifies the performance of collective communication operations on multiple Trainium nodes

## Logs from the deep health checks

The following are example logs from the SageMaker HyperPod deep health checks.

### Cluster-level logs

The cluster-level deep health check logs are stored in your CloudWatch log group at `/aws/sagemaker/Clusters/<cluster_name>/<cluster_id>`

The log streams are logged at `DeepHealthCheckResults/<log_stream_id>`.

As an example shown below, the deep health check output logs show the instance ID that failed the checks with the cause of the failure.

```
{
  "level": "error",
  "ts": "2024-06-18T21:15:22Z",
  "msg": "Encountered FaultyInstance. Replace the Instance. Region: us-west-2,
  InstanceType: p4d.24xlarge. ERROR:Bandwidth has less than threshold: Expected minimum
  threshold :80, NCCL Test output Bw: 30"
}
```

## Instance-level logs

The instance-level deep health check logs are stored at `/var/log/aws/clusterscat/sagemaker-deep-health-check.log` on each node. SSH into the node and open the log file by running the following command.

```
cat /var/log/aws/clusterscat/sagemaker-deep-health-check.log
```

The following is an example output of the hardware stress, [NVIDIA DCGM](#) stress, and EFA connectivity test.

```
# Hardware Stress Test output

2024-08-20T21:53:58Z info Executing Hardware stress check with command: stress-ng, and
args: [--cpu 32 --vm 2 --hdd 1 --fork 8 --switch 4 --timeout 60 --metrics]

2024-08-20T21:54:58Z info stress-ng success

2024-08-20T21:54:58Z     info     GpuPci Count check success

# DCGM Stress Test

2024-08-20T22:25:02Z     info     DCGM diagnostic health summary: dcgmCheckLevel:
0 dcgmVersion: 3.3.7 gpuDriverVersion: 535.183.01, gpuDeviceIds: [2237]
replacementRequired: false rebootRequired:false

# EFA Loopback Test

2024-08-20T22:26:28Z     info     EFA Loopback check passed for device: rdmap0s29 .
Output summary is MaxBw: 58.590000, AvgBw: 32.420000, MaxTypicalLat: 30.870000,
MinTypicalLat: 20.080000, AvgLat: 21.630000
```

The following is an example output of the NCCL connectivity test.

#	size	count	type	redop	root	time	algbw	busbw	#wrong
time	algbw	busbw	#wrong						
#	(B)	(elements)				(us)	(GB/s)	(GB/s)	
(us)	(GB/s)	(GB/s)							
	8	2	float	sum	-1	353.9	0.00	0.00	0
304.2	0.00	0.00	0						

	16		4	float	sum	-1	352.8	0.00	0.00	0
422.9	0.00	0.00	0							
	32		8	float	sum	-1	520.0	0.00	0.00	0
480.3	0.00	0.00	0							
	64		16	float	sum	-1	563.0	0.00	0.00	0
416.1	0.00	0.00	0							
	128		32	float	sum	-1	245.1	0.00	0.00	0
308.4	0.00	0.00	0							
	256		64	float	sum	-1	310.8	0.00	0.00	0
304.9	0.00	0.00	0							
	512		128	float	sum	-1	304.9	0.00	0.00	0
300.8	0.00	0.00	0							
	1024		256	float	sum	-1	509.3	0.00	0.00	0
495.4	0.00	0.00	0							
	2048		512	float	sum	-1	530.3	0.00	0.00	0
420.0	0.00	0.00	0							
	4096		1024	float	sum	-1	391.2	0.01	0.01	0
384.5	0.01	0.01	0							
	8192		2048	float	sum	-1	328.5	0.02	0.02	0
253.2	0.03	0.03	0							
	16384		4096	float	sum	-1	497.6	0.03	0.03	0
490.9	0.03	0.03	0							
	32768		8192	float	sum	-1	496.7	0.07	0.07	0
425.0	0.08	0.08	0							
	65536		16384	float	sum	-1	448.0	0.15	0.15	0
501.0	0.13	0.13	0							
	131072		32768	float	sum	-1	577.4	0.23	0.23	0
593.4	0.22	0.22	0							
	262144		65536	float	sum	-1	757.8	0.35	0.35	0
721.6	0.36	0.36	0							
	524288		131072	float	sum	-1	1057.1	0.50	0.50	0
1019.1	0.51	0.51	0							
	1048576		262144	float	sum	-1	1460.5	0.72	0.72	0
1435.6	0.73	0.73	0							
	2097152		524288	float	sum	-1	2450.6	0.86	0.86	0
2583.1	0.81	0.81	0							
	4194304		1048576	float	sum	-1	4344.5	0.97	0.97	0
4419.3	0.95	0.95	0							
	8388608		2097152	float	sum	-1	8176.5	1.03	1.03	0
8197.8	1.02	1.02	0							
	16777216		4194304	float	sum	-1	15312	1.10	1.10	0
15426	1.09	1.09	0							
	33554432		8388608	float	sum	-1	30149	1.11	1.11	0
29941	1.12	1.12	0							

67108864	16777216	float	sum	-1	57819	1.16	1.16	0
58635	1.14	1.14	0					
134217728	33554432	float	sum	-1	115699	1.16	1.16	0
115331	1.16	1.16	0					
268435456	67108864	float	sum	-1	227507	1.18	1.18	0
228047	1.18	1.18	0					
536870912	134217728	float	sum	-1	453751	1.18	1.18	0
456595	1.18	1.18	0					
1073741824	268435456	float	sum	-1	911719	1.18	1.18	0
911808	1.18	1.18	0					
2147483648	536870912	float	sum	-1	1804971	1.19	1.19	0
1806895	1.19	1.19	0					

2024-08-20T16:22:43.831-07:00

# Out of bounds values : 0 OK

2024-08-20T16:22:43.831-07:00

# Avg bus bandwidth : 0.488398

2024-08-20T23:22:43Z info Nccl test successful. Summary: NcclMaxAlgoBw: 1.190000, NcclAvgAlgoBw: 0.488398, NcclThresholdAlgoBw: 1.180000, NcclOutOfBoundError: OK, NcclOperations: all\_reduce\_perf, NcclTotalDevices: 2, NcclNodes: 2, NcclClusterMessage:

## Automatic node recovery

During cluster creation or update, cluster admin users can select the node (instance) recovery option between Automatic (Recommended) and None at the cluster level. If set to Automatic, SageMaker HyperPod reboots or replaces faulty nodes automatically.

### **⚠ Important**

We recommend setting the Automatic option.

Automatic node recovery runs when issues are found from health-monitoring agent, basic health checks, and deep health checks. If set to None, the health monitoring agent will label the instances when a fault is detected, but it will not automatically initiate any repair or recovery actions on the affected nodes. This option is not recommended.

## Resilience-related Kubernetes labels by SageMaker HyperPod

*Labels* are key-value pairs that are attached to [Kubernetes objects](#). SageMaker HyperPod introduces the following labels for the health checks it provides.

### Node health status labels

The node-health-status labels represent the status of the node health and to be used as part of node selector filter in healthy nodes.

Label	Description
sagemaker.amazonaws.com/node-health-status: Schedulable	The node has passed basic health checks and is available for running workloads. This health check is the same as the <a href="#">currently available SageMaker HyperPod resiliency features for Slurm clusters</a> .
sagemaker.amazonaws.com/node-health-status: Unschedulable	The node is running deep health checks and is not available for running workloads.
sagemaker.amazonaws.com/node-health-status: UnschedulablePendingReplacement	The node has failed deep health checks or health-monitoring agent checks and requires a replacement. If automatic node recovery is enabled, the node will be automatically replaced by SageMaker HyperPod.
sagemaker.amazonaws.com/node-health-status: UnschedulablePendingReboot	The node has failed deep health checks or health-monitoring agent checks and requires a reboot. If automatic node recovery is enabled, the node will be automatically rebooted by SageMaker HyperPod.

### Deep health check labels

The deep-health-check-status labels represent the progress of deep health check on a specific node. Helpful for Kubernetes users to quickly filter for progress of overall deep health checks.

Label	Description
sagemaker.amazonaws.com/deep-health-check-status: InProgress	The node is running deep health checks and is not available for running workloads.
sagemaker.amazonaws.com/deep-health-check-status: Passed	The node has successfully completed deep health checks and health-monitoring agent checks, and is available for running workloads.
sagemaker.amazonaws.com/deep-health-check-status: Failed	The node has failed deep health checks or health-monitoring agent checks and requires a reboot or replacement. If automatic node recovery is enabled, the node will be automatically rebooted or replaced by SageMaker HyperPod.

## Fault type and reason labels

The following describes the fault-type and fault-reason labels.

- **fault-type** labels represent high-level fault categories when health checks fail. These are populated for failures identified during both deep health and health-monitoring agent checks.
- **fault-reason** labels represent the detailed fault reason associated with a fault-type.

## How SageMaker HyperPod labels

The following topics cover how labeling is done depending on various cases.

### Topics

- [When a node is added to a SageMaker HyperPod cluster with deep health check config disabled](#)
- [When a node is added to a SageMaker HyperPod cluster with deep health check config enabled](#)
- [When there are any compute failures on nodes](#)

## When a node is added to a SageMaker HyperPod cluster with deep health check config disabled

When a new node added into cluster, and if deep health check is not enabled for the instance group, SageMaker HyperPod runs the same health checks as the [currently available SageMaker HyperPod health checks for Slurm clusters](#).

If the health check passes, the nodes will be marked with the following label.

```
sagemaker.amazonaws.com/node-health-status: Schedulable
```

If the health check doesn't pass, the nodes will be terminated and replaced. This behavior is the same as the way SageMaker HyperPod health check works for Slurm clusters.

## When a node is added to a SageMaker HyperPod cluster with deep health check config enabled

When a new node is added into a SageMaker HyperPod cluster, and if the deep health check test is enabled for the instance group, HyperPod first taints the node and starts the ~2-hour deep health check/stress test on the node. There are 3 possible outputs of the node labels after the deep health check.

1. When the deep health check test passes

```
sagemaker.amazonaws.com/node-health-status: Schedulable
```

2. When the deep health check test fails, and the instance needs to be replaced

```
sagemaker.amazonaws.com/node-health-status: UnschedulablePendingReplacement
```

3. When the deep health check test fails, and the instance needs to be rebooted to rerun the deep health check

```
sagemaker.amazonaws.com/node-health-status: UnschedulablePendingReboot
```

If an instance fails the deep health check test, the instance will always be replaced. If the deep health check tests succeeds, the taint on the node will be removed.

## When there are any compute failures on nodes

The SageMaker HyperPod health monitor agent also continuously monitors the health status of each node. When it detects any failures (such as GPU failure and driver crash), the agent marks the node with one of the following labels.

1. When the node is unhealthy and needs to be replaced

```
sagemaker.amazonaws.com/node-health-status: UnschedulablePendingReplacement
```

2. When the node is unhealthy and needs to be rebooted

```
sagemaker.amazonaws.com/node-health-status: UnschedulablePendingReboot
```

The health monitor agent also taints the node when it detects any node health issues.

## Manually quarantine, replace, or reboot a node

Learn how to manually quarantine, replace, and reboot a faulty node in SageMaker HyperPod clusters orchestrated with Amazon EKS.

### To quarantine a node and force delete a training pod

```
kubectl cordon <node-name>
```

After quarantine, force ejecting the Pod. This is useful when you see a pod is stuck in termination for more than 30min or kubectl describe pod shows 'Node is not ready' in Events

```
kubectl delete pods <pod-name> --grace-period=0 --force
```

### To replace a node

Label the node to replace with sagemaker.amazonaws.com/node-health-status=UnschedulablePendingReplacement, which triggers the SageMaker HyperPod [the section called "Automatic node recovery"](#). Note that you also need to activate automatic node recovery during cluster creation or update.

```
kubectl label nodes <node-name> \
sagemaker.amazonaws.com/node-health-status=UnschedulablePendingReplacement
```

## To reboot a node

Label the node to reboot with `sagemaker.amazonaws.com/node-health-status=UnschedulablePendingReboot`, which triggers the SageMaker HyperPod [the section called “Automatic node recovery”](#). Note that you also need to activate automatic node recovery during cluster creation or update.

```
kubectl label nodes <node-name> \
  sagemaker.amazonaws.com/node-health-status=UnschedulablePendingReboot
```

After the labels `UnschedulablePendingReplacement` or `UnschedulablePendingReboot` are applied, you should be able to see the node is terminated or reboot in few minutes.

## Suggested resilience configurations

When the deep health checks are enabled, whenever a new instance is added to the HyperPod cluster (either during create-cluster or through automatic node replacement), the new instance goes through the deep health check process (instance level stress tests) for about a couple of hours. The following are suggested resilience config combinations depending on possible cases.

1. **Case:** When you have additional spare nodes within a cluster as back-up resources (not using the full capacity), or if you can wait for about 2 hours for the deep health check process to get the less error-prone instances.

**Recommendation:** Enable the deep health check config throughout the cluster lifecycle. Node auto-recovery config is enabled by default.

2. **Case:** When you don't have additional backup nodes (capacity is fully used for some training load). You want to get the replacement nodes as soon as possible to resume the training job.

**Recommendation:** Enable the deep health check during cluster creation, then turn-off the deep health check config after the cluster is created. Node auto recovery config is enabled by default.

3. **Case:** When you don't have additional backup nodes, and you don't want to wait for the ~2 hour deep health check process (small clusters).

**Recommendation:** disable the deep health check config throughout the cluster life cycle. Node auto recovery config is enabled by default.

If you want to resume the training job from a failure immediately, make sure that you have additional spare nodes as backup resources in the cluster.

## Running jobs on SageMaker HyperPod clusters orchestrated by Amazon EKS

The following topics provide procedures and examples of accessing compute nodes and running ML workloads on provisioned SageMaker HyperPod clusters orchestrated with Amazon EKS. Depending on how you have set up the environment on your HyperPod cluster, there are many ways to run ML workloads on HyperPod clusters.

### Tip

For a hands-on experience and guidance on how to set up and use a SageMaker HyperPod cluster orchestrated with Amazon EKS, we recommend taking this [Amazon EKS Support in SageMaker HyperPod](#) workshop.

Data scientist users can train foundational models using the EKS cluster set as the orchestrator for the SageMaker HyperPod cluster. Scientists leverage the [SageMaker HyperPod CLI](#) and the native kubectl commands to find available SageMaker HyperPod clusters, submit training jobs (Pods), and manage their workloads. The SageMaker HyperPod CLI enables job submission using a training job schema file, and provides capabilities for job listing, description, cancellation, and execution. Scientists can use [Kubeflow Training Operator](#) according to compute quotas managed by HyperPod, and [SageMaker AI-managed MLflow](#) to manage ML experiments and training runs.

### Topics

- [Install the SageMaker HyperPod CLI](#)
- [SageMaker HyperPod CLI commands](#)
- [Run jobs using the SageMaker HyperPod CLI](#)
- [Run jobs using kubectl](#)

### Install the SageMaker HyperPod CLI

SageMaker HyperPod provides the [SageMaker HyperPod command line interface \(CLI\)](#) package.

1. Check if the version of Python on your local machine is between 3.8 and 3.11.
2. Check the prerequisites in the README markdown file in the [SageMaker HyperPod CLI](#) package.
3. Clone the SageMaker HyperPod CLI package from GitHub.

```
git clone https://github.com/aws/sagemaker-hyperpod-cli.git
```

#### 4. Install the SageMaker HyperPod CLI.

```
cd sagemaker-hyperpod-cli && pip install .
```

#### 5. Test if the SageMaker HyperPod CLI is successfully installed by running the following command.

```
hyperpod --help
```

 **Note**

If you are a data scientist and want to use the SageMaker HyperPod CLI, make sure that your IAM role is set up properly by your cluster admins following the instructions at [the section called “IAM users for scientists”](#) and [the section called “Setting up Kubernetes role-based access control”](#).

### SageMaker HyperPod CLI commands

The following table summarizes the SageMaker HyperPod CLI commands.

 **Note**

For a complete CLI reference, see [README](#) in the [SageMaker HyperPod CLI GitHub repository](#).

SageMaker HyperPod CLI command	Entity	Description
hyperpod get-clusters	cluster/access	Lists all clusters to which the user has been enabled with IAM permissions to submit training workloadsGives the current snapshot of whole available instances which are not running any workloads or jobs along with maximum

SageMaker HyperPod CLI command	Entity	Description
		capacity, grouping by health check statuses (ex: BurnInPassed)
hyperpod connect-cluster	cluster/access	Configures kubectl to operate on the specified HyperPod cluster and namespace
hyperpod start-job	job	Submits the job to targeted cluster-Job name will be unique at namespace level-Users will be able to override yaml spec by passing them as CLI arguments
hyperpod get-job	job	Display metadata of the submitted job
hyperpod list-jobs	job	Lists all jobs in the connected cluster/namespace to which the user has been added with IAM permissions to submit training workloads
hyperpod cancel-job	job	Stops and deletes the job and gives up underlying compute resources. This job cannot be resumed again. A new job needs to be started, if needed.
hyperpod list-pods	pod	Lists all the pods in the given job in a namespace

SageMaker HyperPod CLI command	Entity	Description
hyperpod get-log	pod	Retrieves the logs of a particular pod in a specified job
hyperpod exec	pod	Run the bash command in the shell of the specified pod(s) and publishes the output
hyperpod --help	utility	lists all supported commands

## Run jobs using the SageMaker HyperPod CLI

To run jobs, make sure that you installed Kubeflow Training Operator in the EKS clusters. For more information, see [the section called “Install packages on the Amazon EKS cluster using Helm”](#).

Run the `hyperpod get-cluster` command to get the list of available HyperPod clusters.

```
hyperpod get-clusters
```

Run the `hyperpod connect-cluster` to configure the SageMaker HyperPod CLI with the EKS cluster orchestrating the HyperPod cluster.

```
hyperpod connect-cluster --cluster-name <hyperpod-cluster-name>
```

Use the `hyperpod start-job` command to run a job. The following command shows the command with required options.

```
hyperpod start-job \
--job-name <job-name>
--image <docker-image-uri>
--entry-script <entrypoint-script>
--instance-type <ml.instance.type>
--node-count <integer>
```

The `hyperpod start-job` command also comes with various options such as job auto-resume and job scheduling.

## Enabling job auto-resume

The hyperpod `start-job` command also has the following options to specify job auto-resume. For enabling job auto-resume to work with the SageMaker HyperPod node resiliency features, you must set the value for the `restart-policy` option to `OnFailure`. The job must be running under the `kubeflow` namespace or a namespace prefixed with `hyperpod`.

- `--auto-resume <bool>` #Optional, enable job auto resume after fails, default is false
- `--max-retry <int>` #Optional, if auto-resume is true, max-retry default value is 1 if not specified
- `--restart-policy <enum>` #Optional, PyTorchJob restart policy. Available values are `Always`, `OnFailure`, `Never` or `ExitCode`. The default value is `OnFailure`.

```
hyperpod start-job \
    ... // required options \
    --auto-resume true \
    --max-retry 3 \
    --restart-policy OnFailure
```

## Running jobs with scheduling options

The hyperpod `start-job` command has the following options to set up the job with queuing mechanisms.

### Note

You need [Kueue](#) installed in the EKS cluster. If you haven't installed, follow the instructions in [Setup for SageMaker HyperPod task governance](#).

- `--scheduler-type <enum>` #Optional, Specify the scheduler type. The default is Kueue.
- `--queue-name <string>` #Optional, Specify the name of the [Local Queue](#) or [Cluster Queue](#) you want to submit with the job. The queue should be created by cluster admins using `CreateComputeQuota`.
- `--priority <string>` #Optional, Specify the name of the [Workload Priority Class](#), which should be created by cluster admins.

```
hyperpod start-job \
```

```
... // required options  
--scheduler-type Kueue \  
--queue-name high-priority-queue \  
--priority high
```

## Running jobs from a configuration file

As an alternative, you can create a job configuration file containing all the parameters required by the job and then pass this config file to the hyperpod `start-job` command using the `--config-file` option. In this case:

1. Create your job configuration file with the required parameters. Refer to the job configuration file in the SageMaker HyperPod CLI GitHub repository for a [baseline configuration file](#).
2. Start the job using the configuration file as follows.

```
hyperpod start-job --config-file /path/to/test_job.yaml
```

### Tip

For a complete list of parameters of the `hyperpod start-job` command, see the [Submitting a Job](#) section in the `README.md` of the SageMaker HyperPod CLI GitHub repository.

## Run jobs using `kubectl`

### Note

Training job auto resume requires Kubeflow Training Operator release version `1.7.0`, `1.8.0`, or `1.8.1`.

Note that you should install Kubeflow Training Operator in the clusters using a Helm chart. For more information, see [the section called “Install packages on the Amazon EKS cluster using Helm”](#). Verify if the Kubeflow Training Operator control plane is properly set up by running the following command.

```
kubectl get pods -n kubeflow
```

This should return an output similar to the following.

NAME	READY	STATUS	RESTARTS	AGE
training-operator-658c68d697-46zmn	1/1	Running	0	90s

## To submit a training job

To run a training jobs, prepare the job configuration file and run the [kubectl apply](#) command as follows.

```
kubectl apply -f /path/to/training_job.yaml
```

## To describe a training job

To retrieve the details of the job submitted to the EKS cluster, use the following command. It returns job information such as the job submission time, completion time, job status, configuration details.

```
kubectl get -o yaml training-job -n kubeflow
```

## To stop a training job and delete EKS resources

To stop a training job, use kubectl delete. The following is an example of stopping the training job created from the configuration file pytorch\_job\_simple.yaml.

```
kubectl delete -f /path/to/training_job.yaml
```

This should return the following output.

```
pytorchjob.kubeflow.org "training-job" deleted
```

## To enable job auto-resume

SageMaker HyperPod supports job auto-resume functionality for Kubernetes jobs, integrating with the Kubeflow Training Operator control plane.

Ensure that there are sufficient nodes in the cluster that have passed the SageMaker HyperPod health check. The nodes should have the taint sagemaker.amazonaws.com/node-health-status set to Schedulable. It is recommended to include a node selector in the job YAML file to select nodes with the appropriate configuration as follows.

[sagemaker.amazonaws.com/node-health-status](https://sagemaker.amazonaws.com/node-health-status): Schedulable

The following code snippet is an example of how to modify a Kubeflow PyTorch job YAML configuration to enable the job auto-resume functionality. You need to add two annotations and set `restartPolicy` to `OnFailure` as follows.

```
apiVersion: "kubeflow.org/v1"
kind: PyTorchJob
metadata:
  name: pytorch-simple
  namespace: kubeflow
  annotations: { // config for job auto resume
    sagemaker.amazonaws.com/enable-job-auto-resume: "true"
    sagemaker.amazonaws.com/job-max-retry-count: "2"
  }
spec:
  pytorchReplicaSpecs:
    .....
    Worker:
      replicas: 10
      restartPolicy: OnFailure
      template:
        spec:
          nodeSelector:
            sagemaker.amazonaws.com/node-health-status: Schedulable
```

## To check the job auto-resume status

Run the following command to check the status of job auto-resume.

```
kubectl describe pytorchjob -n kubeflow <job-name>
```

Depending on the failure patterns, you might see two patterns of Kubeflow training job restart as follows.

### Pattern 1:

Start Time:	2024-07-11T05:53:10Z		
Events:			
Type	Reason	Age	From
Message			

```

-----
Normal  SuccessfulCreateService  9m45s          pytorchjob-controller
Created service: pt-job-1-worker-0
Normal  SuccessfulCreateService  9m45s          pytorchjob-controller
Created service: pt-job-1-worker-1
Normal  SuccessfulCreateService  9m45s          pytorchjob-controller
Created service: pt-job-1-master-0
Warning PyTorchJobRestarting    7m59s          pytorchjob-controller
PyTorchJob pt-job-1 is restarting because 1 Master replica(s) failed.
Normal  SuccessfulCreatePod    7m58s (x2 over 9m45s) pytorchjob-controller
Created pod: pt-job-1-worker-0
Normal  SuccessfulCreatePod    7m58s (x2 over 9m45s) pytorchjob-controller
Created pod: pt-job-1-worker-1
Normal  SuccessfulCreatePod    7m58s (x2 over 9m45s) pytorchjob-controller
Created pod: pt-job-1-master-0
Warning PyTorchJobRestarting    7m58s          pytorchjob-controller
PyTorchJob pt-job-1 is restarting because 1 Worker replica(s) failed.

```

## Pattern 2:

### Events:

Type	Reason	Age	From	Message
-----	-----	-----	-----	-----
Normal	SuccessfulCreatePod	19m	pytorchjob-controller	Created pod: pt-job-2-worker-0
Normal	SuccessfulCreateService	19m	pytorchjob-controller	Created service: pt-job-2-worker-0
Normal	SuccessfulCreatePod	19m	pytorchjob-controller	Created pod: pt-job-2-master-0
Normal	SuccessfulCreateService	19m	pytorchjob-controller	Created service: pt-job-2-master-0
Normal	SuccessfulCreatePod	4m48s	pytorchjob-controller	Created pod: pt-job-2-worker-0
Normal	SuccessfulCreatePod	4m48s	pytorchjob-controller	Created pod: pt-job-2-master-0

## Observability for SageMaker HyperPod cluster orchestrated by Amazon EKS

To achieve comprehensive observability into your SageMaker HyperPod cluster resources and software components, integrate the cluster with [Amazon CloudWatch Container Insights](#), [Amazon Managed Service for Prometheus](#), and [Amazon Managed Grafana](#).

The integration with Amazon Managed Service for Prometheus enables the export of metrics related to your HyperPod cluster resources, providing insights into their performance, utilization, and health. The integration with Amazon Managed Grafana enables the visualization of these metrics through various Grafana dashboards that offer intuitive interface for monitoring and analyzing the cluster's behavior. By leveraging these services, you gain a centralized and unified view of your HyperPod cluster, facilitating proactive monitoring, troubleshooting, and optimization of your distributed training workloads.

### Tip

To find practical examples and solutions, see also the [Observability](#) section in the [Amazon EKS Support in SageMaker HyperPod workshop](#).

Proceed to the following topics to set up for SageMaker HyperPod cluster observability.

### Topics

- [Model observability for training jobs on SageMaker HyperPod clusters orchestrated by Amazon EKS](#)
- [Cluster observability](#)

## Model observability for training jobs on SageMaker HyperPod clusters orchestrated by Amazon EKS

SageMaker HyperPod clusters orchestrated with Amazon EKS can integrate with the [MLflow application on Amazon SageMaker Studio](#). Cluster admins set up the MLflow server and connect it with the SageMaker HyperPod clusters. Data scientists can gain insights into the model

### To set up an MLflow server using AWS CLI

An MLflow tracking server should be created by cluster admin.

1. Create a SageMaker AI MLflow tracking server, following the instructions at [Create a tracking server using the AWS CLI](#).
2. Make sure that the `eks-auth:AssumeRoleForPodIdentity` permission exists in the IAM execution role for SageMaker HyperPod.
3. If the `eks-pod-identity-agent` add-on is not already installed on your EKS cluster, install the add-on on the EKS cluster.

```
aws eks create-addon \
--cluster-name <eks_cluster_name> \
--addon-name eks-pod-identity-agent \
--addon-version vx.y.z-eksbuild.1
```

#### 4. Create a trust-relationship.json file for a new role for Pod to call MLflow APIs.

```
cat >trust-relationship.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
EOF
```

Run the following code to create the role and attach the trust relationship.

```
aws iam create-role --role-name hyperpod-mlflow-role \
--assume-role-policy-document file://trust-relationship.json \
--description "allow pods to emit mlflow metrics and put data in s3"
```

#### 5. Create the following policy that grants Pod access to call all sagemaker-mlflow operations and to put model artifacts in S3. S3 permission already exists within the tracking server but if the model artifacts is too big direct call to s3 is made from the MLflow code to upload the artifacts.

```
cat >hyperpod-mlflow-policy.json <<EOF
{
    "Version": "2012-10-17",
```

```
"Statement": [  
    {  
        "Effect": "Allow",  
        "Action": [  
            "sagemaker-mlflow:AccessUI",  
            "sagemaker-mlflow>CreateExperiment",  
            "sagemaker-mlflow:SearchExperiments",  
            "sagemaker-mlflow:GetExperiment",  
            "sagemaker-mlflow:GetExperimentByName",  
            "sagemaker-mlflow>DeleteExperiment",  
            "sagemaker-mlflow:RestoreExperiment",  
            "sagemaker-mlflow:UpdateExperiment",  
            "sagemaker-mlflow>CreateRun",  
            "sagemaker-mlflow>DeleteRun",  
            "sagemaker-mlflow:RestoreRun",  
            "sagemaker-mlflow:GetRun",  
            "sagemaker-mlflow:LogMetric",  
            "sagemaker-mlflow:LogBatch",  
            "sagemaker-mlflow:LogModel",  
            "sagemaker-mlflow:LogInputs",  
            "sagemaker-mlflow:SetExperimentTag",  
            "sagemaker-mlflow:SetTag",  
            "sagemaker-mlflow>DeleteTag",  
            "sagemaker-mlflow:LogParam",  
            "sagemaker-mlflow:GetMetricHistory",  
            "sagemaker-mlflow:SearchRuns",  
            "sagemaker-mlflow>ListArtifacts",  
            "sagemaker-mlflow:UpdateRun",  
            "sagemaker-mlflow>CreateRegisteredModel",  
            "sagemaker-mlflow:GetRegisteredModel",  
            "sagemaker-mlflow:RenameRegisteredModel",  
            "sagemaker-mlflow:UpdateRegisteredModel",  
            "sagemaker-mlflow>DeleteRegisteredModel",  
            "sagemaker-mlflow:GetLatestModelVersions",  
            "sagemaker-mlflow>CreateModelVersion",  
            "sagemaker-mlflow:GetModelVersion",  
            "sagemaker-mlflow:UpdateModelVersion",  
            "sagemaker-mlflow>DeleteModelVersion",  
            "sagemaker-mlflow:SearchModelVersions",  
            "sagemaker-mlflow:GetDownloadURIForModelVersionArtifacts",  
            "sagemaker-mlflow:TransitionModelVersionStage",  
            "sagemaker-mlflow:SearchRegisteredModels",  
            "sagemaker-mlflow:SetRegisteredModelTag",  
            "sagemaker-mlflow>DeleteRegisteredModelTag",  
        ]  
    }  
]
```

```
        "sagemaker-mlflow>DeleteModelVersionTag",
        "sagemaker-mlflow>DeleteRegisteredModelAlias",
        "sagemaker-mlflow>SetRegisteredModelAlias",
        "sagemaker-mlflow>GetModelVersionByAlias"
    ],
    "Resource": "arn:aws:sagemaker:us-west-2:111122223333:mlflow-tracking-
server/<ml tracking server name>"  
},  
{  
    "Effect": "Allow",  
    "Action": [  
        "s3:PutObject"  
    ],  
    "Resource": "arn:aws:s3:::<mlflow-s3-bucket_name>"  
}  
]  
]  
}  
EOF
```

### Note

The ARNs should be the one from the MLflow server and the S3 bucket set up with the MLflow server during the server you created following the instructions [Set up MLflow infrastructure](#).

6. Attach the `mlflow-metrics-emit-policy` policy to the `hyperpod-mlflow-role` using the policy document saved in the previous step.

```
aws iam put-role-policy \
--role-name hyperpod-mlflow-role \
--policy-name mlflow-metrics-emit-policy \
--policy-document file://hyperpod-mlflow-policy.json
```

7. Create a Kubernetes service account for Pod to access the MLflow server.

```
cat >mlflow-service-account.yaml <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  name: mlflow-service-account
  namespace: kubeflow
```

EOF

Run the following command to apply to the EKS cluster.

```
kubectl apply -f mlflow-service-account.yaml
```

## 8. Create a Pod identity association.

```
aws eks create-pod-identity-association \
--cluster-name EKS_CLUSTER_NAME \
--role-arn arn:aws:iam::111122223333:role/hyperpod-mlflow-role \
--namespace kubeflow \
--service-account mlflow-service-account
```

## To collect metrics from training jobs to the MLflow server

Data scientists need to set up the training script and docker image to emit metrics to the MLflow server.

### 1. Add the following lines at the beginning of your training script.

```
import mlflow

# Set the Tracking Server URI using the ARN of the Tracking Server you created
mlflow.set_tracking_uri(os.environ['MLFLOW_TRACKING_ARN'])
# Enable autologging in MLflow
mlflow.autolog()
```

### 2. Build a Docker image with the training script and push to Amazon ECR. Get the ARN of the ECR container. For more information about building and pushing a Docker image, see [Pushing a Docker image in the ECR User Guide](#).

#### Tip

Make sure that you add installation of mlflow and sagemaker-mlflow packages in the Docker file. To learn more about the installation of the packages, requirements, and compatible versions of the packages, see [Install MLflow and the SageMaker AI MLflow plugin](#).

3. Add a service account in the training job Pods to give them access to hyperpod-mlflow-role. This allows Pods to call MLflow APIs. Run the following SageMaker HyperPod CLI job submission template. Create this with file name mlflow-test.yaml.

```
defaults:
  - override hydra/job_logging: stdout

hydra:
  run:
    dir: .
  output_subdir: null

training_cfg:
  entry_script: ./train.py
  script_args: []
  run:
    name: test-job-with-mlflow # Current run name
    nodes: 2 # Number of nodes to use for current training
    # ntasks_per_node: 1 # Number of devices to use per node
  cluster:
    cluster_type: k8s # currently k8s only
    instance_type: ml.c5.2xlarge
    cluster_config:
      # name of service account associated with the namespace
      service_account_name: mlflow-service-account
      # persistent volume, usually used to mount FSx
      persistent_volume_claims: null
      namespace: kubeflow
      # required node affinity to select nodes with SageMaker HyperPod
      # labels and passed health check if burn-in enabled
      label_selector:
        required:
          sagemaker.amazonaws.com/node-health-status:
            - Schedulable
        preferred:
          sagemaker.amazonaws.com/deep-health-check-status:
            - Passed
        weights:
          - 100
      pullPolicy: IfNotPresent # policy to pull container, can be Always, IfNotPresent and Never
      restartPolicy: OnFailure # restart policy
```

```
base_results_dir: ./result # Location to store the results, checkpoints and logs.  
container: 111122223333.dkr.ecr.us-west-2.amazonaws.com/tag # container to use  
  
env_vars:  
  NCCL_DEBUG: INFO # Logging level for NCCL. Set to "INFO" for debug information  
  MLFLOW_TRACKING_ARN: arn:aws:sagemaker:us-west-2:111122223333:mlflow-tracking-server/  
    tracking-server-name
```

4. Start the job using the YAML file as follows.

```
hyperpod start-job --config-file /path/to/mlflow-test.yaml
```

5. Generate a pre-signed URL for the MLflow tracking server. You can open the link on your browser and start tracking your training job.

```
aws sagemaker create-presigned-mlflow-tracking-server-url \  
  --tracking-server-name "tracking-server-name" \  
  --session-expiration-duration-in-seconds 1800 \  
  --expires-in-seconds 300 \  
  --region region
```

## Cluster observability

To gain visibility into the cluster resource utilization, set up Amazon CloudWatch Container Insights and Amazon Managed Grafana to extract metrics and visualize them on various dashboards.

### Topics

- [Amazon CloudWatch Container Insights](#)
- [Set up an Amazon Managed Grafana workspace](#)

## Amazon CloudWatch Container Insights

Use [Amazon CloudWatch Container Insights](#) to collect, aggregate, and summarize metrics and logs from the containerized applications and micro-services on the EKS cluster associated with a HyperPod cluster.

Amazon CloudWatch Insights collects metrics for compute resources, such as CPU, memory, disk, and network. Container Insights also provides diagnostic information, such as container restart

failures, to help you isolate issues and resolve them quickly. You can also set CloudWatch alarms on metrics that Container Insights collects.

To find a complete list of metrics, see [Amazon EKS and Kubernetes Container Insights metrics](#) in the *Amazon EKS User Guide*.

## Install CloudWatch Container Insights

Cluster admin users should set up CloudWatch Container Insights following the instructions at [Install the CloudWatch agent by using the Amazon CloudWatch Observability EKS add-on or the Helm chart](#) in the *CloudWatch User Guide*. For more information about Amazon EKS add-on, see also [Install the Amazon CloudWatch Observability EKS add-on](#) in the *Amazon EKS User Guide*.

After the installation has completed, verify that the CloudWatch Observability add-on is visible in the EKS cluster add-on tab. It might take about a couple of minutes until the dashboard loads.

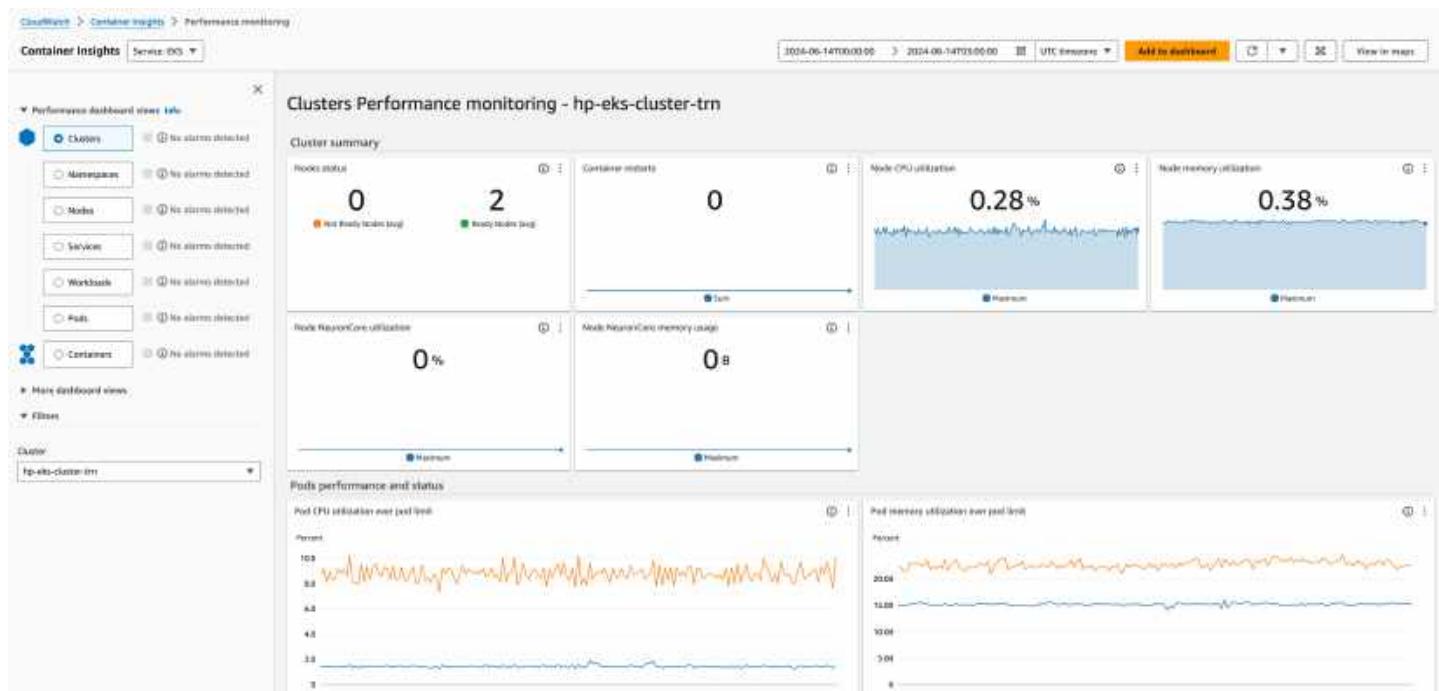
### Note

SageMaker HyperPod requires the CloudWatch Insight v2.0.1-eksbuild.1 or later.



## Access CloudWatch container insights dashboard

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. Choose **Insights**, and then choose **Container Insights**.
3. Select the EKS cluster set up with the HyperPod cluster you're using.
4. View the Pod/Cluster level metrics.



## Access CloudWatch container insights logs

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. Choose **Logs**, and then choose **Log groups**.

When you have the HyperPod clusters integrated with Amazon CloudWatch Container Insights, you can access the relevant log groups in the following format: /aws/containerinsights /<eks-cluster-name>/\*. Within this log group, you can find and explore various types of logs such as Performance logs, Host logs, Application logs, and Data plane logs.

## Set up an Amazon Managed Grafana workspace

You can integrate SageMaker HyperPod with Amazon Managed Grafana and Amazon Managed Service for Prometheus to gain comprehensive cluster observability and visualize in various Grafana dashboards: the Kubernetes cluster monitoring dashboard, the NVIDIA DCGM exporter dashboard, and the FSx for Lustre metrics dashboard, and the EFA metrics dashboard.

## HyperPod in Studio

You can launch machine learning workloads on Amazon SageMaker HyperPod clusters and view HyperPod cluster information in Amazon SageMaker Studio. The increased visibility into cluster

details and hardware metrics can help your team identify the right candidate for your pre-training or fine-tuning workloads.

A set of commands are available to help you get started when you launch Studio IDEs on a HyperPod cluster. You can work on your training scripts, use Docker containers for the training scripts, and submit jobs to the cluster, all from within the Studio IDEs. The following sections provide information on how to set this up, how to discover clusters and monitor their tasks, how to view cluster information, and how to connect to HyperPod clusters in IDEs within Studio.

## Topics

- [Set up HyperPod in Studio](#)
- [HyperPod tabs in Studio](#)
- [Connect to HyperPod clusters and submit tasks to clusters](#)
- [Troubleshoot](#)

## Set up HyperPod in Studio

You need to set up the clusters depending on your choice of the cluster orchestrator to access your clusters through Amazon SageMaker Studio. In the following sections, choose the setup that matches with your orchestrator.

The instructions assume that you already have your cluster set up. For information on the cluster orchestrators and how to set up, start with the HyperPod orchestrator pages:

- [Orchestrating SageMaker HyperPod clusters with Slurm](#)
- [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#)

## Topics

- [Set up a Slurm cluster in Studio](#)
- [Set up an Amazon EKS cluster in Studio](#)

## Set up a Slurm cluster in Studio

The following instructions describe how to set up a HyperPod Slurm cluster in Studio.

1. Create a domain or have one ready. For information on creating a domain, see [Guide to getting set up with Amazon SageMaker AI](#).

2. (Optional) Create and attach a custom FSx for Lustre volume to your domain.
  - a. Ensure that your FSx Lustre file system exists in the same VPC as your intended domain, and is in one of the subnets present in the domain.
  - b. You can follow the instructions in [Adding a custom file system to a domain](#).
3. (Optional) We recommend that you add tags to your clusters to ensure a more smooth workflow. For information on how to add tags, see [Edit a SageMaker HyperPod cluster](#) to update your cluster using the SageMaker AI console.
  - a. Tag your FSx for Lustre file system to your Studio domain. This will help you identify the file system while launching your Studio spaces. To do so, add the following tag to your cluster to identify it with the FSx filesystem ID, `fs-id`.

Tag Key = "hyperpod-cluster-filesystem", Tag Value = "fs-id".
  - b. Tag your [Amazon Managed Grafana](#) workspace to your Studio domain. This will be used to quickly link to your Grafana workspace directly from your cluster in Studio. To do so, add the following tag to your cluster to identify it with your Grafana workspace ID, `ws-id`.

Tag Key = "grafana-workspace", Tag Value = "ws-id".
4. Add the following permission to your execution role.

For information on SageMaker AI execution roles and how to edit them, see [Understanding domain space permissions and execution roles](#).

To learn how to attach policies to an IAM user or group, see [Adding and removing IAM identity permissions](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "ssm:StartSession",  
                "ssm:TerminateSession"  
            ],  
            "Resource": "*"  
        },  
        {
```

```
        "Effect": "Allow",
        "Action": [
            "sagemaker:CreateCluster",
            "sagemaker>ListClusters"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "cloudwatch:PutMetricData",
            "cloudwatch:GetMetricData"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "sagemaker:DescribeCluster",
            "sagemaker:DescribeClusterNode",
            "sagemaker>ListClusterNodes",
            "sagemaker:UpdateCluster",
            "sagemaker:UpdateClusterSoftware"
        ],
        "Resource": "arn:aws:sagemaker:region:account-id:cluster/*"
    }
]
```

5. Add a tag to this IAM role, with Tag Key = "SSMSessionRunAs" and Tag Value = "os\_user". The os\_user here is the same user that you setup for the Slurm cluster. Manage access to SageMaker HyperPod clusters at an IAM role or user level by using the Run As feature in [AWS Systems Manager Agent \(SSM Agent\)](#). With this feature, you can start each SSM session using the operating system (OS) user associated to the IAM role or user.

For information on how to add tags to your execution role, see [Tag IAM roles](#).

6. [Turn on Run As support for Linux and macOS managed nodes](#). The Run As settings are account wide and is required for all SSM sessions to start successfully.
7. (Optional) [Restrict task view in Studio for Slurm clusters](#). For information on viewable tasks in Studio, see [Tasks](#).

In Amazon SageMaker Studio you can navigate to view your clusters in HyperPod clusters (under Compute).

## Restrict task view in Studio for Slurm clusters

You can restrict users to view Slurm tasks that are authorized to view, without requiring manual input of namespaces or additional permissions checks. The restriction is applied based on the users' IAM role, providing a streamlined and secure user experience. The following section provides information on how to restrict task view in Studio for Slurm clusters. For information on viewable tasks in Studio, see [Tasks](#).

All Studio users can view, manage, and interact with all Slurm cluster tasks by default. To restrict this, you can manage access to SageMaker HyperPod clusters at an IAM role or user level by using the **Run As** feature in [AWS Systems Manager Agent \(SSM Agent\)](#).

You can do this by tagging IAM roles with specific identifiers, such as their username or group. When a user accesses Studio, the Session Manager uses the Run As feature to execute commands as a specific Slurm user account that matches their IAM role tags. The Slurm configuration can be set up to limit task visibility based on the user account. The Studio UI will automatically filter tasks visible to that specific user account when commands are executed through the Run As feature. Once set up, each user assuming the role with the specified identifiers will have those Slurm tasks filtered based on the Slurm configuration. For information on how to add tags to your execution role, see [Tag IAM roles](#).

## Set up an Amazon EKS cluster in Studio

The following instructions describe how to set up an Amazon EKS cluster in Studio.

1. Create a domain or have one ready. For information on creating a domain, see [Guide to getting set up with Amazon SageMaker AI](#).
2. Add the following permission to your execution role.

For information on SageMaker AI execution roles and how to edit them, see [Understanding domain space permissions and execution roles](#).

To learn how to attach policies to an IAM user or group, see [Adding and removing IAM identity permissions](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [
```

```
{  
    "Sid": "DescribeHyperpodClusterPermissions",  
    "Effect": "Allow",  
    "Action": [  
        "sagemaker:DescribeCluster"  
    ],  
    "Resource": "hyperpod-cluster-arn"  
},  
{  
    "Effect": "Allow",  
    "Action": "ec2:Describe*",  
    "Resource": "*"  
},  
{  
    "Effect": "Allow",  
    "Action": [  
        "ecr:CompleteLayerUpload",  
        "ecr:GetAuthorizationToken",  
        "ecr:UploadLayerPart",  
        "ecr:InitiateLayerUpload",  
        "ecr:BatchCheckLayerAvailability",  
        "ecr:PutImage"  
    ],  
    "Resource": "*"  
},  
{  
    "Effect": "Allow",  
    "Action": [  
        "cloudwatch:PutMetricData",  
        "cloudwatch:GetMetricData"  
    ],  
    "Resource": "*"  
},  
{  
    "Sid": "UseEksClusterPermissions",  
    "Effect": "Allow",  
    "Action": [  
        "eks:DescribeCluster",  
        "eks:AccessKubernetesApi",  
        "eks:DescribeAddon"  
    ],  
    "Resource": "eks-cluster-arn"  
},  
{
```

```
        "Sid": "ListClustersPermission",
        "Effect": "Allow",
        "Action": [
            "sagemaker>ListClusters"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "ssm:StartSession",
            "ssm:TerminateSession"
        ],
        "Resource": "*"
    }
]
```

3. [Grant IAM users access to Kubernetes with EKS access entries.](#)

- a. Navigate to the Amazon EKS cluster associated with your HyperPod cluster.
  - b. Choose the **Access** tab and [create an access entry](#) for the execution role you created.
    - i. In step 1, Select the execution role you created above in the **IAM** principal dropdown.
    - ii. In step 2, select a policy name and select an access scope that you want the users to have access to.
4. (Optional) To ensure a more smooth experience, we recommend that you add tags to your clusters. For information on how to add tags, see [Edit a SageMaker HyperPod cluster](#) to update your cluster using the SageMaker AI console.
- Tag your [Amazon Managed Grafana](#) workspace to your Studio domain. This will be used to quickly link to your Grafana workspace directly from your cluster in Studio. To do so, add the following tag to your cluster to identify it with your Grafana workspace ID, ws-id.

Tag Key = "grafana-workspace", Tag Value = "ws-id".

5. (Optional) [Restrict task view in Studio for EKS clusters](#). For information on viewable tasks in Studio, see [Tasks](#).

## Restrict task view in Studio for EKS clusters

You can restrict Kubernetes namespace permissions for users, so that they will only have access to view tasks belonging to a specified namespace. The following provides information on how to restrict the task view in Studio for EKS clusters. For information on viewable tasks in Studio, see [Tasks](#).

Users will have visibility to all EKS cluster tasks by default. You can restrict users' visibility for EKS cluster tasks to specified namespaces, ensuring that users can access the resources they need while maintaining strict access controls. You will need to provide the namespace for the user to display jobs of that namespace once the following is set up.

Once the restriction is applied, you will need to provide the namespace to the users assuming the role. Studio will only display the jobs of the namespace once the user provides inputs namespace they have permissions to view in the **Tasks** tab.

The following configuration allows administrators to grant specific, limited access to data scientists for viewing tasks within the cluster. This configuration grants the following permissions:

- List and get pods
- List and get events
- Get Custom Resource Definitions (CRDs)

### YAML Configuration

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: pods-events-crd-cluster-role
rules:
- apiGroups: []
  resources: ["pods"]
  verbs: ["get", "list"]
- apiGroups: []
  resources: ["events"]
  verbs: ["get", "list"]
- apiGroups: ["apiextensions.k8s.io"]
  resources: ["customresourcedefinitions"]
  verbs: ["get"]
---
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: pods-events-crd-cluster-role-binding
subjects:
- kind: Group
  name: pods-events-crd-cluster-level
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: pods-events-crd-cluster-role
  apiGroup: rbac.authorization.k8s.io
```

1. Save the YAML configuration to a file named `cluster-role.yaml`.
2. Apply the configuration using [kubectl](#):

```
  kubectl apply -f cluster-role.yaml
```

3. Verify the configuration:

```
  kubectl get clusterrole pods-events-crd-cluster-role
  kubectl get clusterrolebinding pods-events-crd-cluster-role-binding
```

4. Assign users to the `pods-events-crd-cluster-level` group through your identity provider or IAM.

## HyperPod tabs in Studio

In Amazon SageMaker Studio you can navigate to one of your clusters in **HyperPod clusters** (under **Compute**) and view your list of clusters. The displayed clusters contain information like tasks, hardware metrics, settings, and metadata details. This visibility can help your team identify the right candidate for your pre-training or finetuning workloads. The following sections provide information on each type of information.

### Tasks

Amazon SageMaker HyperPod provides a view of your cluster tasks. Tasks are operations or jobs that are sent to the cluster. These can be machine learning operations, like training, running experiments, or inference. The following section provides information on your HyperPod cluster tasks.

In Amazon SageMaker Studio, you can navigate to one of your clusters in **HyperPod clusters** (under **Compute**) and view the **Tasks** information on your cluster. If you are having any issues with viewing tasks, see [Troubleshoot](#).

The task table includes:

For Slurm clusters

For Slurm clusters, the tasks currently in the Slurm job scheduler queue are shown in the table. The information shown for each task includes the task name, status, job ID, partition, run time, nodes, created by, and actions.

For a list and details about past jobs, use the [`sacct`](#) command in JupyterLab or a Code Editor terminal. The `sacct` command is used to view *historical information* about jobs that have *finished* or are *complete* in the system. It provides accounting information, including job resources usage like memory and exit status.

By default, all Studio users can view, manage, and interact with all available Slurm tasks. To restrict the viewable tasks to Studio users, see [Restrict task view in Studio for Slurm clusters](#).

For Amazon EKS clusters

For Amazon EKS clusters, kubeflow (PyTorch, MPI, TensorFlow) tasks are shown in the table. PyTorch tasks are shown by default. You can sort for PyTorch, MPI, and TensorFlow under **Task type**. The information that is shown for each task includes the task name, status, namespace, priority class, and creation time.

By default, all users can view jobs across all namespaces. To restrict the viewable Kubernetes namespaces available to Studio users, see [Restrict task view in Studio for EKS clusters](#). If a user cannot view the tasks and is asked to provide a namespace, they need to get that information from the administrator.

## Metrics

Amazon SageMaker HyperPod provides a view of your Slurm or Amazon EKS cluster utilization metrics. The following provides information on your HyperPod cluster metrics.

You will need to install the Amazon EKS add-on to view the following metrics. For more information, see [Install the Amazon CloudWatch Observability EKS add-on](#).

In Amazon SageMaker Studio, you can navigate to one of your clusters in **HyperPod clusters** (under **Compute**) and view the **Metrics** details on your cluster. Metrics provides a comprehensive view of cluster utilization metrics, including hardware, team, and task metrics. This includes compute availability and usage, team allocation and utilization, and task run and wait time information.

## Settings

Amazon SageMaker HyperPod provides a view of your cluster settings. The following provides information on your HyperPod cluster settings.

In Amazon SageMaker Studio you can navigate to one of your clusters in **HyperPod clusters** (under **Compute**) and view the **Settings** information on your cluster. The information includes the following:

- **Instances** details, including instance ID, status, instance type, and instance group
- **Instance groups** details, including instance group name, type, counts, and compute information
- **Orchestration** details, including the orchestrator, version, and certification authority
- **Cluster resiliency** details
- **Security** details, including subnets and security groups

## Details

Amazon SageMaker HyperPod provides a view of your cluster metadata details. The following paragraph provides information on how to get your HyperPod cluster details.

In Amazon SageMaker Studio, you can navigate to one of your clusters in **HyperPod clusters** (under **Compute**) and view the **Details** on your cluster. This includes the tags, logs, and metadata.

## Connect to HyperPod clusters and submit tasks to clusters

You can launch machine learning workloads on HyperPod clusters within Amazon SageMaker Studio IDEs. When you launch Studio IDEs on a HyperPod cluster, a set of commands are available to help you get started. You can work on your training scripts, use Docker containers for the training scripts, and submit jobs to the cluster, all from within the Studio IDEs. The following section provides information on how to connect your cluster to Studio IDEs.

In Amazon SageMaker Studio you can navigate to one of your clusters in **HyperPod clusters** (under **Compute**) and view your list of clusters. You can connect your cluster to an IDE listed under **Actions**.

You can also choose your custom file system from the list of options. For information on how to get this set up, see [Set up HyperPod in Studio](#).

Alternatively, you can create a space and launch an IDE using the AWS CLI. Use the following commands to do so. The following example creates a Private JupyterLab space for *user-profile-name* with the *fs-id* FSx for Lustre file system attached.

1. Create a space using the [create-space](#) AWS CLI.

```
aws sagemaker create-space \
--region your-region \
--ownership-settings "OwnerUserName=user-profile-name" \
--space-sharing-settings "SharingType=Private" \
--space-settings
  "AppType=JupyterLab,CustomFileSystems=[{FSxLustreFileSystem={FileSystemId=fs-id}]}"
```

2. Create the app using the [create-app](#) AWS CLI.

```
aws sagemaker create-app \
--region your-region \
--space-name space-name \
--resource-spec '{"ec2InstanceType":"'"instance-type"'", "appEnvironmentArn":"'"image-arn"'"}'
```

Once you have your applications open, you can submit tasks directly to the clusters you are connected to.

## Troubleshoot

The following section lists troubleshooting solutions for HyperPod in Studio.

### Topics

- [Tasks tab](#)
- [Metrics tab](#)

### Tasks tab

If you get Custom Resource Definition (CRD) is not configured on the cluster while in the **Tasks** tab.

- Grant `EKSAdminViewPolicy` and `ClusterAccessRole` policies to your domain execution role.

For information on how to add tags to your execution role, see [Tag IAM roles](#).

To learn how to attach policies to an IAM user or group, see [Adding and removing IAM identity permissions](#).

If the tasks grid for Slurm metrics doesn't stop loading in the **Tasks** tab.

- Ensure that RunAs enabled in your [AWS Session Manager](#) preferences and the role you are using has the `SSMSessionRunAs` tag attached.
  - To enable RunAs, navigate to the **Preference** tab in the [Systems Manager console](#).
  - [Turn on Run As support for Linux and macOS managed nodes](#)

For restricted task view in Studio for EKS clusters:

- If your execution role doesn't have permissions to list namespaces for EKS clusters.
  - See [Restrict task view in Studio for EKS clusters](#).
- If users are experiencing issues with access for EKS clusters.
  1. Verify RBAC is enabled by running the following AWS CLI command.

```
kubectl api-versions | grep rbac
```

This should return `rbac.authorization.k8s.io/v1`.

2. Check if the `ClusterRole` and `ClusterRoleBinding` exist by running the following commands.

```
kubectl get clusterrole pods-events-crd-cluster-role  
kubectl get clusterrolebinding pods-events-crd-cluster-role-binding
```

3. Verify user group membership. Ensure the user is correctly assigned to the `pods-events-crd-cluster-level` group in your identity provider or IAM.

- If user can't see any resources.
  - Verify group membership and ensure the `ClusterRoleBinding` is correctly applied.
- [If users can see resources in all namespaces](#).

- If namespace restriction is required, consider using Role and RoleBinding instead of ClusterRole and ClusterRoleBinding.
- If configuration appears correct, but permissions aren't applied.
  - Check if there are any NetworkPolicies or PodSecurityPolicies interfering with access.

## Metrics tab

If there are no Amazon CloudWatch metrics are displayed in the **Metrics** tab.

- The Metrics section of HyperPod cluster details uses CloudWatch to fetch the data. In order to see the metrics in this section, you need to have enabled [Cluster observability](#). Contact your administrator to configure metrics.

## SageMaker HyperPod references

Find more information and references about using SageMaker HyperPod in the following topics.

### Topics

- [SageMaker HyperPod pricing](#)
- [SageMaker HyperPod APIs](#)
- [SageMaker HyperPod forms](#)
- [SageMaker HyperPod DLAMI](#)
- [SageMaker HyperPod API permissions reference](#)
- [SageMaker HyperPod commands in AWS CLI](#)
- [SageMaker HyperPod Python modules in AWS SDK for Python \(Boto3\)](#)

## SageMaker HyperPod pricing

The following topics provide information about SageMaker HyperPod pricing. To find more details on price per hour for using SageMaker HyperPod instances, see also [Amazon SageMaker Pricing](#).

### Capacity requests

You can allocate on-demand or reserved compute capacity with SageMaker AI for use on SageMaker HyperPod. On-demand cluster creation allocates available capacity from the SageMaker AI on-demand capacity pool. Alternatively, you can request reserved capacity to ensure access by submitting a ticket for a quota increase. Inbound capacity requests are prioritized by SageMaker AI and you receive an estimated time for capacity allocation.

## Service billing

When you provision a compute capacity on SageMaker HyperPod, you are billed for the duration of the capacity allocation. SageMaker HyperPod billing appears in your anniversary bills with a line item for the type of capacity allocation (on-demand, reserved), the instance type, and the time spent on using the instance.

To submit a ticket for a quota increase, see [the section called “SageMaker HyperPod quotas”](#).

## SageMaker HyperPod APIs

The following list is a full set of SageMaker HyperPod APIs for submitting action requests in JSON format to SageMaker AI through AWS CLI or AWS SDK for Python (Boto3).

- [BatchDeleteClusterNodes](#)
- [CreateCluster](#)
- [DeleteCluster](#)
- [DescribeCluster](#)
- [DescribeClusterNode](#)
- [ListClusterNodes](#)
- [ListClusters](#)
- [UpdateCluster](#)
- [UpdateClusterSoftware](#)

## SageMaker HyperPod forms

To configure the Slurm workload manager tool on HyperPod, you should create a Slurm configuration file required by HyperPod using the provided form.

## Configuration form for provisioning Slurm nodes on HyperPod

The following code is the Slurm configuration form you should prepare to properly set up Slurm nodes on your HyperPod cluster. You should complete this form and upload it as part of a set of lifecycle scripts during cluster creation. To learn how this form should be prepared throughout HyperPod cluster creation processes, see [the section called “Customize SageMaker HyperPod clusters using lifecycle scripts”](#).

```
// Save as provisioning_params.json.  
{  
    "version": "1.0.0",  
    "workload_manager": "slurm",  
    "controller_group": "string",  
    "login_group": "string",  
    "worker_groups": [  
        {  
            "instance_group_name": "string",  
            "partition_name": "string"  
        }  
    ],  
    "fsx_dns_name": "string",  
    "fsx_mountname": "string"  
}
```

- **version** – Required. This is the version of the HyperPod provisioning parameter form. Keep it to **1.0.0**.
- **workload\_manager** – Required. This is for specifying which workload manager to be configured on the HyperPod cluster. Keep it to **slurm**.
- **controller\_group** – Required. This is for specifying the name of the HyperPod cluster instance group you want to assign to Slurm controller (head) node.
- **login\_group** – Optional. This is for specifying the name of the HyperPod cluster instance group you want to assign to Slurm login node.
- **worker\_groups** – Required. This is for setting up Slurm worker (compute) nodes on the HyperPod cluster.
  - **instance\_group\_name** – Required. This is for specifying the name of the HyperPod instance group you want to assign to Slurm worker (compute) node.
  - **partition\_name** – Required. This is for specifying the partition name to the node.

- `fsx_dns_name` – Optional. If you want to set up your Slurm nodes on the HyperPod cluster to communicate with Amazon FSx, specify the FSx DNS name.
- `fsx_mountname` – Optional. If you want to set up your Slurm nodes on the HyperPod cluster to communicate with Amazon FSx, specify the FSx mount name.

## SageMaker HyperPod DLAMI

SageMaker HyperPod runs a DLAMI based on:

- [AWS Deep Learning Base GPU AMI \(Ubuntu 20.04\)](#) for orchestration with Slurm.
- Amazon Linux 2 based AMI for orchestration with Amazon EKS.

The SageMaker HyperPod DLAMI is bundled with additional packages to support open source tools such as Slurm, Kubernetes, dependencies, and SageMaker HyperPod cluster software packages to support resiliency features such as cluster health check and auto-resume. To follow up with HyperPod software updates that the HyperPod service team distributes through DLAMIs, see [the section called "HyperPod release notes"](#).

## SageMaker HyperPod API permissions reference

### Important

Custom IAM policies that allow Amazon SageMaker Studio or Amazon SageMaker Studio Classic to create Amazon SageMaker resources must also grant permissions to add tags to those resources. The permission to add tags to resources is required because Studio and Studio Classic automatically tag any resources they create. If an IAM policy allows Studio and Studio Classic to create resources but does not allow tagging, "AccessDenied" errors can occur when trying to create resources. For more information, see [Provide permissions for tagging SageMaker AI resources](#).

[AWS managed policies for Amazon SageMaker AI](#) that give permissions to create SageMaker resources already include permissions to add tags while creating those resources.

When you are setting up access control for allowing to run SageMaker HyperPod API operations and writing a permissions policy that you can attach to IAM users for cloud administrators, use the following table as a reference.

Amazon SageMaker API Operations	Required Permissions (API Actions)	Resources
CreateCluster	sagemaker:CreateCluster	arn:aws:sagemaker: <i>region:account-id</i> :cluster/ <i>cluster-id</i>
DeleteCluster	sagemaker>DeleteCluster	arn:aws:sagemaker: <i>region:account-id</i> :cluster/ <i>cluster-id</i>
DescribeCluster	sagemaker:DescribeCluster	arn:aws:sagemaker: <i>region:account-id</i> :cluster/ <i>cluster-id</i>
DescribeClusterNode	sagemaker:DescribeClusterNode	arn:aws:sagemaker: <i>region:account-id</i> :cluster/ <i>cluster-id</i>
ListClusterNodes	sagemaker>ListClusterNodes	arn:aws:sagemaker: <i>region:account-id</i> :cluster/ <i>cluster-id</i>
ListClusters	sagemaker>ListClusters	arn:aws:sagemaker: <i>region:account-id</i> :cluster/ <i>cluster-id</i>
UpdateCluster	sagemaker:UpdateCluster	arn:aws:sagemaker: <i>region:account-id</i> :cluster/ <i>cluster-id</i>

UpdateClusterSoftware	sagemaker:UpdateClusterSoftware	arn:aws:sagemaker: <i>region:account-id</i> :cluster/ <i>cluster-id</i>
-----------------------	---------------------------------	--

For a complete list of permissions and resource types for SageMaker APIs, see [Actions, resources, and condition keys for Amazon SageMaker AI](#) in the *AWS Service Authorization Reference*.

## SageMaker HyperPod commands in AWS CLI

The following are the AWS CLI commands for SageMaker HyperPod to run the core [HyperPod API operations](#).

- [batch-delete-cluster-nodes](#)
- [create-cluster](#)
- [delete-cluster](#)
- [describe-cluster](#)
- [describe-cluster-node](#)
- [list-cluster-nodes](#)
- [list-clusters](#)
- [update-cluster](#)
- [update-cluster-software](#)

## SageMaker HyperPod Python modules in AWS SDK for Python (Boto3)

The following are the methods of the AWS SDK for Python (Boto3) client for SageMaker AI to run the core [HyperPod API operations](#).

- [batch\\_delete\\_cluster\\_nodes](#)
- [create\\_cluster](#)
- [delete\\_cluster](#)
- [describe\\_cluster](#)
- [describe\\_cluster\\_node](#)
- [list\\_cluster\\_nodes](#)

- [list\\_clusters](#)
- [update\\_cluster](#)
- [update\\_cluster\\_software](#)

## Amazon SageMaker HyperPod release notes

This topic covers release notes that track update, fixes, and new features for Amazon SageMaker HyperPod. If you are looking for general feature releases, updates, and improvements for Amazon SageMaker HyperPod, you might find this page helpful.

The HyperPod AMI releases are documented separately to include information of the key components including general AMI releases, versions, and dependencies. If you are looking for these information related to HyperPod AMI releases, see [the section called “HyperPod AMI releases”](#).

### SageMaker HyperPod release notes: February 20, 2025

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Slurm](#) and [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#).

#### New features and improvements

- Added support for deleting instance groups from your SageMaker HyperPod cluster. For more information, see [the section called “Delete instance groups”](#) from EKS-orchestrated clusters and [the section called “Scale down a cluster”](#) for Slurm-orchestrated clusters.

### SageMaker HyperPod release notes: February 18, 2025

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Slurm](#) and [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#).

#### New features

- This release of SageMaker HyperPod incorporates a security update from the Nvidia container toolkit (from version 1.17.3 to version 1.17.4). For more information, see [v1.17.4 release note](#).

**Note**

For all container workloads in the Nvidia container toolkit version 1.17.4, the mounting of CUDA compatibility libraries is now disabled. To ensure compatibility with multiple CUDA versions on container workflows, update your LD\_LIBRARY\_PATH to include your CUDA compatibility libraries. You can find the specific steps in [???](#).

For information about related AMI releases, see [the section called “February 18, 2025”](#) and [the section called “February 18, 2025”](#).

## SageMaker HyperPod release notes: February 06, 2025

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Slurm](#) and [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#).

### New features and improvements

- Enhanced SageMaker HyperPod multi-AZ support: You can specify different subnets and security groups, cutting across different Availability Zones, for individual instance groups within your cluster. For more information about SageMaker HyperPod multi-AZ support, see [the section called “Setting up SageMaker HyperPod clusters across multiple AZs”](#).

## SageMaker HyperPod release notes: January 22, 2025

### AMI releases

- [???](#)

## SageMaker HyperPod release notes: January 09, 2025

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#) and [Orchestrating SageMaker HyperPod clusters with Slurm](#).

### New features and improvements

- Added IPv6 support: Clusters can use IPv6 addressing when configured with IPv6-enabled VPC and subnets. For more information, see [the section called “Setting up SageMaker HyperPod with a custom Amazon VPC”](#).

## SageMaker HyperPod release notes: December 21, 2024

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#) and [Orchestrating SageMaker HyperPod clusters with Slurm](#).

### New features

- SageMaker HyperPod now supports the following instance types for both Slurm and Amazon EKS clusters.
  - New instance types: C6gn, C6i, M6i, R6i.
  - New Trainium instance types: Trn1 and Trn1n.

### Improvements

- Enhanced error logging visibility when Slurm interrupts jobs, and prevented unnecessary job step termination during Slurm-initiated job cancellations.
- Updated base DLAMI for p5en for both Slurm and Amazon EKS clusters.

### AMI releases

- [???](#)
- [???](#)

## SageMaker HyperPod release notes: December 13, 2024

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#) and [Orchestrating SageMaker HyperPod clusters with Slurm](#).

### New feature

- SageMaker HyperPod releases a set of Amazon CloudWatch metrics to monitor the health and performance of SageMaker HyperPod Slurm clusters. These metrics are related to CPU, GPU, memory utilization, and cluster instance information such as node counts and failed nodes. This

monitoring feature is enabled by default, and the metrics can be accessed under the /aws/sagemaker/Clusters CloudWatch namespace. You can also set up CloudWatch alarms based on these metrics to proactively detect and address potential issues within their Slurm-based HyperPod clusters. For more information, see [the section called “Amazon SageMaker HyperPod Slurm metrics”](#).

## AMI releases

- [the section called “December 13, 2024”](#)

## SageMaker HyperPod release notes: November 24, 2024

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#) and [Orchestrating SageMaker HyperPod clusters with Slurm](#).

### New features

- Added support for configuring SageMaker HyperPod clusters across multiple Availability Zones. For more information about SageMaker HyperPod multi-AZ support, see [Setting up SageMaker HyperPod clusters across multiple AZs](#).

## AMI releases

- [the section called “November 24, 2024”](#)
- [the section called “November 24, 2024”](#)

## SageMaker HyperPod release notes: November 15, 2024

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#) and [Orchestrating SageMaker HyperPod clusters with Slurm](#). For more information, see and [the section called “November 15, 2024”](#).

### New features and improvements

- Added support for trn1 and trn1n instance types for both Amazon EKS and Slurm orchestrated clusters.
- Improved log management for Slurm clusters:

- Implemented log rotation: weekly or daily based on size.
- Set log retention to 3 weeks.
- Compressed logs to reduce storage impact.
- Continued uploading logs to CloudWatch for long-term retention.

 **Note**

Some logs are still stored in syslogs.

- Adjusted Fluent Bit settings to prevent tracking issues with files containing long lines.

## Bug fixes

- Prevented unintended truncation with Slurm controller node updates in configuration file `slurm.config`.

## AMI releases

- [the section called “November 15, 2024”](#)
- [the section called “November 15, 2024”](#)

## SageMaker HyperPod release notes: November 11, 2024

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#) and [Orchestrating SageMaker HyperPod clusters with Slurm](#).

### New feature

- SageMaker HyperPod AMI now supports G6e instance types.

## AMI releases

- [the section called “November 11, 2024”](#)
- [the section called “November 11, 2024”](#)

## SageMaker HyperPod release notes: October 31, 2024

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#) and [Orchestrating SageMaker HyperPod clusters with Slurm](#).

### New features

- Added scaling down SageMaker HyperPod clusters at the instance group level and instance level for both Amazon EKS and Slurm orchestrated clusters. For more information about scaling down Amazon EKS clusters, see [Scale down a SageMaker HyperPod cluster](#). For more information about scaling down Slurm clusters, see *Scale down a cluster* in [Using the AWS CLI](#).
- SageMaker HyperPod now supports the P5e instance type for both Amazon EKS and Slurm orchestrated clusters.

## SageMaker HyperPod release notes: October 21, 2024

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#) and [Orchestrating SageMaker HyperPod clusters with Slurm](#).

### New feature

- SageMaker HyperPod now supports the P5e[n], G6, Gr6, and Trn2[n] instance types for both Slurm and Amazon EKS clusters.

### AMI releases

- [the section called “October 21, 2024”](#)
- [the section called “October 21, 2024”](#)

## SageMaker HyperPod release notes: September 10, 2024

SageMaker HyperPod releases the following for [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#) and [Orchestrating SageMaker HyperPod clusters with Slurm](#).

### New features

- Added Amazon EKS support in SageMaker HyperPod. To learn more, see [the section called “Orchestrating HyperPod clusters with Amazon EKS”](#).

- Added support for managing SageMaker HyperPod clusters through AWS CloudFormation and Terraform. For more information about managing HyperPod clusters through AWS CloudFormation, see [CloudFormation documentation for AWS::SageMaker::Cluster](#). To learn about managing HyperPod clusters through Terraform, see [Terraform documentation for awscs\\_sagemaker\\_cluster](#).

## AMI releases

- [the section called “September 10, 2024”](#)
- [the section called “September 10, 2024”](#)

## SageMaker HyperPod release notes: August 20, 2024

SageMaker HyperPod releases the following for [the section called “Orchestrating HyperPod clusters with Slurm”](#).

### New features

- Enhanced the [SageMaker HyperPod auto-resume functionality](#), extending the resiliency capability for Slurm nodes attached with Generic REsources (GRES).

When [Generic Resources \(GRES\)](#) are attached to a Slurm node, Slurm typically doesn't permit changes in the node allocation, such as replacing nodes, and thus doesn't allow to resume a failed job. Unless explicitly forbidden, the HyperPod auto-resume functionality automatically requeues any faulty job associated with the GRES-enabled nodes. This process involves stopping the job, placing it back into the job queue, and then restarting the job from the beginning.

### Other changes

- Pre-packaged [slurmrestd](#) in the SageMaker HyperPod AMI.
- Changed the default values for ResumeTimeout and UnkillableStepTimeout from 60 seconds to 300 seconds in `slurm.conf` to improve system responsiveness and job handling.
- Made minor improvements on health checks for NVIDIA Data Center GPU Manager (DCGM) and The NVIDIA System Management Interface (`nvidia-smi`).

### Bug fixes

- The HyperPod auto-resume plug-in can use idle nodes to resume a job.

## SageMaker HyperPod release notes: June 20, 2024

SageMaker HyperPod releases the following for [the section called “Orchestrating HyperPod clusters with Slurm”](#).

### New features

- Added a new capability of attaching additional storage to SageMaker HyperPod cluster instances. With this capability, you can configure supplementary storage at the instance group configuration level during the cluster creation or update processes, either through the SageMaker HyperPod console or the [CreateCluster](#) and [UpdateCluster](#) APIs. The additional EBS volume is attached to each instance within a SageMaker HyperPod cluster and mounted to /opt/sagemaker. To learn more about implementing it in your SageMaker HyperPod cluster, see the updated documentation on the following pages.
  - [the section called “Getting started with SageMaker HyperPod”](#)
  - [the section called “SageMaker HyperPod operation”](#)

Note that you need to update the HyperPod cluster software to use this capability. After patching the HyperPod cluster software, you can utilize this capability for existing SageMaker HyperPod clusters created before June 20, 2024 by adding new instance groups. This capability is fully effective for any SageMaker HyperPod clusters created after June 20, 2024.

### Upgrade steps

- Run the following command to call the [UpdateClusterSoftware](#) API to update your existing HyperPod clusters with the latest HyperPod DLAMI. To find more instructions, see [the section called “Update the SageMaker HyperPod platform software of a cluster”](#).

#### Important

Back up your work before running this API. The patching process replaces the root volume with the updated AMI, which means that your previous data stored in the instance root volume will be lost. Make sure that you back up your data from the instance root volume to Amazon S3 or Amazon FSx for Lustre. For more information, see [the section called “Use the backup script provided by SageMaker HyperPod”](#).

```
aws sagemaker update-cluster-software --cluster-name your-cluster-name
```

### Note

Note that you should run the AWS CLI command to update your HyperPod cluster.

Updating the HyperPod software through SageMaker HyperPod console UI is currently not available.

## SageMaker HyperPod release notes: April 24, 2024

SageMaker HyperPod releases the following for [the section called “Orchestrating HyperPod clusters with Slurm”](#).

### Bug fixes

- Fixed a bug with the ThreadsPerCore parameter in the [ClusterInstanceGroupSpecification](#) API. With the fix, the [CreateCluster](#) and [UpdateCluster](#) APIs properly take and apply the user input through ThreadsPerCore. This fix is effective on HyperPod clusters created after April 24, 2024. If you had issues with this bug and want to get this fix applied to your cluster, you need to create a new cluster. Make sure that you back up and restore your work while moving to a new cluster following the instructions at [the section called “Use the backup script provided by SageMaker HyperPod”](#).

## SageMaker HyperPod release notes: March 27, 2024

SageMaker HyperPod releases the following for [the section called “Orchestrating HyperPod clusters with Slurm”](#).

### HyperPod software patch

The HyperPod service team distributes software patches through [the section called “SageMaker HyperPod DLAMI”](#). See the following details about the latest HyperPod DLAMI.

- In this release of the HyperPod DLAMI, Slurm is built with REST service (slurmestd) with JSON, YAML, and JWT support.
- Upgraded [Slurm](#) to v23.11.3.

## Improvements

- Increased auto-resume service timeout to 60 minutes.
- Improved instance replacement process to not restart the Slurm controller.
- Improved error messages from running lifecycle scripts, such as download errors and instance health check errors on instance start-up.

## Bug fixes

- Fixed a bug with chrony service that caused an issue with time synchronization.
- Fixed a bug with parsing `slurm.conf`.
- Fixed an issue with [NVIDIA go-dcgm](#) library.

## SageMaker HyperPod release notes: March 14, 2024

SageMaker HyperPod releases the following for [the section called “Orchestrating HyperPod clusters with Slurm”](#).

### Improvements

- HyperPod now properly supports passing partition names provided through `provisioning_params.json` and creates partitions appropriately based on provided inputs. For more information about `provisioning_params.json`, see [the section called “SageMaker HyperPod forms”](#) and [the section called “Customize SageMaker HyperPod clusters using lifecycle scripts”](#).

### AMI releases

- [the section called “March 14, 2024”](#)

## SageMaker HyperPod release notes: February 15, 2024

SageMaker HyperPod releases the following for [the section called “Orchestrating HyperPod clusters with Slurm”](#).

### New features

- Added a new `UpdateClusterSoftware` API for SageMaker HyperPod security patching. When security patches become available, we recommend you to update existing SageMaker HyperPod clusters in your account by running `aws sagemaker update-cluster-software --cluster-name your-cluster-name`. To follow up with future security patches, keep tracking this Amazon SageMaker HyperPod release notes page. To learn how the `UpdateClusterSoftware` API works, see [the section called “Update the SageMaker HyperPod platform software of a cluster”](#).

## SageMaker HyperPod release notes: November 29, 2023

SageMaker HyperPod releases the following for [the section called “Orchestrating HyperPod clusters with Slurm”](#).

### New features

- Launched Amazon SageMaker HyperPod at AWS re:Invent 2023.

### AMI releases

- [the section called “November 29, 2023”](#)

## Amazon SageMaker HyperPod AMI releases

Amazon SageMaker HyperPod Amazon Machine Images (AMIs) are specialized machine images for distributed machine learning workloads and high-performance computing. These AMIs enhance base images with essential components including GPU drivers and AWS Neuron accelerator support.

Key components added to HyperPod AMIs include:

- Advanced orchestration tools:
  - [Orchestrating SageMaker HyperPod clusters with Slurm](#)
  - [Orchestrating SageMaker HyperPod clusters with Amazon EKS](#)
- Cluster management dependencies
- Built-in resiliency features:
  - cluster health check

- auto-resume capabilities
- Support for HyperPod cluster management and configuration

These enhancements are built upon the following base Deep Learning AMIs (DLAMIs):

- [AWS Deep Learning AMIs Base GPU AMI \(Ubuntu 20.04\)](#) for orchestration with Slurm.
- Amazon Linux 2 based AMI for orchestration with Amazon EKS.

Choose your HyperPod AMIs based on your orchestration preference:

- For Slurm orchestration, see [the section called “AMI releases for Slurm”](#).
- For Amazon EKS orchestration, see [the section called “AMI releases for Amazon EKS”](#).

For information about Amazon SageMaker HyperPod feature releases, see [the section called “HyperPod release notes”](#).

## SageMaker HyperPod AMI releases for Slurm

The following release notes track the latest updates for Amazon SageMaker HyperPod AMI releases for Slurm orchestration. These HyperPod AMIs are built upon [AWS Deep Learning Base GPU AMI \(Ubuntu 20.04\)](#). The HyperPod service team distributes software patches through [the section called “SageMaker HyperPod DLAMI”](#). For HyperPod AMI releases for Amazon EKS orchestration, see [the section called “AMI releases for Amazon EKS”](#). For information about Amazon SageMaker HyperPod feature releases, see [the section called “HyperPod release notes”](#).

### Note

To update existing HyperPod clusters with the latest DLAMI, see [the section called “Update the SageMaker HyperPod platform software of a cluster”](#).

## SageMaker HyperPod AMI releases for Slurm: February 18, 2025

### Improvements for Slurm

- Upgraded Slurm version to 24.11.
- Upgraded Elastic Fabric Adapter (EFA) version from 1.37.0 to 1.38.0.

- The EFA now includes the AWS OFI NCCL plugin. You can find this plugin in the /opt/amazon/ofi-nccl directory, rather than the original /opt/aws-ofi-nccl/ location. If you need to update your LD\_LIBRARY\_PATH environment variable, make sure to modify the path to point to the new /opt/amazon/ofi-nccl location for the OFI NCCL plugin.
- Removed the emacs package from these DLAMIs. You can install emacs from GNU emacs.

## Amazon SageMaker HyperPod DLAMI for Slurm support

Installed the latest version of neoron SDK 2.19

- **aws-neuronx-collectives/unknown:** 2.23.135.0-3e70920f2 amd64
- **aws-neuronx-dkms/unknown:** 2.19.64.0 amd64
- **aws-neuronx-runtime-lib/unknown:** 2.23.112.0-9b5179492 amd64
- **aws-neuronx-tools/unknown:** 2.20.204.0 amd64

## SageMaker HyperPod AMI releases for Slurm: December 21, 2024

### SageMaker HyperPod DLAMI for Slurm support

Deep Learning Slurm AMI

- **NVIDIA driver:** 550.127.05
- **EFA driver:** 2.13.0-1
- Installed the latest version of AWS Neuron SDK
  - **aws-neuronx-collectives:** 2.22.33.0
  - **aws-neuronx-dkms:** 2.18.20.0
  - **aws-neuronx-oci-hook:** 2.5.8.0
  - **aws-neuronx-runtime-lib:** 2.22.19.0
  - **aws-neuronx-tools:** 2.19.0.0

## SageMaker HyperPod AMI releases for Slurm: November 24, 2024

### AMI general updates

- Released in MEL (Melbourne) Region.

- Updated SageMaker HyperPod base DLAMI to the following versions:
  - Slurm: 2024-11-22.

## SageMaker HyperPod AMI releases for Slurm: November 15, 2024

### AMI general updates

- Installed latest libnvidia-nscq-xxx package.

## SageMaker HyperPod DLAMI for Slurm support

### Deep Learning Slurm AMI

- **NVIDIA driver:** 550.127.05
- **EFA driver:** 2.13.0-1
- Installed the latest version of AWS Neuron SDK
  - **aws-neuronx-collectives:** v2.22.33.0-d2128d1aa
  - **aws-neuronx-dkms:** v2.17.17.0
  - **aws-neuronx-oci-hook:** v2.4.4.0
  - **aws-neuronx-runtime-lib:** v2.21.41.0
  - **aws-neuronx-tools:** v2.18.3.0

## SageMaker HyperPod AMI releases for Slurm: November 11, 2024

### AMI general updates

- Updated SageMaker HyperPod base DLAMI to the following version:
  - Slurm: 2024-10-23.

## SageMaker HyperPod AMI releases for Slurm: October 21, 2024

### AMI general updates

- Updated SageMaker HyperPod base DLAMI to the following versions:
  - Slurm: 2024-09-27.

## SageMaker HyperPod AMI releases for Slurm: September 10, 2024

### SageMaker HyperPod DLAMI for Slurm support

#### Deep Learning Slurm AMI

- Installed the NVIDIA driver v550.90.07
- Installed the EFA driver v2.10
- Installed the latest version of AWS Neuron SDK
  - **aws-neuronx-collectives:** v2.21.46.0
  - **aws-neuronx-dkms:** v2.17.17.0
  - **aws-neuronx-oci-hook:** v2.4.4.0
  - **aws-neuronx-runtime-lib:** v2.21.41.0
  - **aws-neuronx-tools:** v2.18.3.0

## SageMaker HyperPod AMI releases for Slurm: March 14, 2024

### HyperPod DLAMI for Slurm software patch

- Upgraded [Slurm](#) to v23.11.1
- Added [OpenPMIx](#) v4.2.6 for enabling [Slurm with PMIx](#).
- Built upon the [AWS Deep Learning Base GPU AMI \(Ubuntu 20.04\)](#) released on 2023-10-26
- A complete list of pre-installed packages in this HyperPod DLAMI in addition to the base AMI
  - [Slurm](#): v23.11.1
  - [OpenPMIx](#) : v4.2.6
  - Munge: v0.5.15
  - aws-neuronx-dkms: v2.\*
  - aws-neuronx-collectives: v2.\*
  - aws-neuronx-runtime-lib: v2.\*
  - aws-neuronx-tools: v2.\*
  - SageMaker HyperPod software packages to support features such as cluster health check and auto-resume

### Upgrade steps

- Run the following command to call the [UpdateClusterSoftware](#) API to update your existing HyperPod clusters with the latest HyperPod DLAMI. To find more instructions, see [the section called “Update the SageMaker HyperPod platform software of a cluster”](#).

 **Important**

Back up your work before running this API. The patching process replaces the root volume with the updated AMI, which means that your previous data stored in the instance root volume will be lost. Make sure that you back up your data from the instance root volume to Amazon S3 or Amazon FSx for Lustre. For more information, see [the section called “Use the backup script provided by SageMaker HyperPod”](#).

```
aws sagemaker update-cluster-software --cluster-name your-cluster-name
```

 **Note**

Note that you should run the AWS CLI command to update your HyperPod cluster. Updating the HyperPod software through SageMaker HyperPod console UI is currently not available.

## SageMaker HyperPod AMI release for Slurm: November 29, 2023

### HyperPod DLAMI for Slurm software patch

The HyperPod service team distributes software patches through [the section called “SageMaker HyperPod DLAMI”](#). See the following details about the latest HyperPod DLAMI.

- Built upon the [AWS Deep Learning Base GPU AMI \(Ubuntu 20.04\)](#) released on 2023-10-18
- A complete list of pre-installed packages in this HyperPod DLAMI in addition to the base AMI
  - [Slurm](#): v23.02.3
  - Munge: v0.5.15
  - aws-neuronx-dkms: v2.\*
  - aws-neuronx-collectives: v2.\*
  - aws-neuronx-runtime-lib: v2.\*

- aws-neuronx-tools: v2.\*
- SageMaker HyperPod software packages to support features such as cluster health check and auto-resume

## SageMaker HyperPod AMI releases for Amazon EKS

The following release notes track the latest updates for Amazon SageMaker HyperPod AMI releases for Amazon EKS orchestration. Each release note includes a summarized list of packages pre-installed or pre-configured in the SageMaker HyperPod DLAMIs for Amazon EKS support. Each DLAMI is built on Amazon Linux 2 (AL2) and supports a specific Kubernetes version. For HyperPod DLAMI releases for Slurm orchestration, see [the section called “AMI releases for Slurm”](#). For information about Amazon SageMaker HyperPod feature releases, see [the section called “HyperPod release notes”](#).

### SageMaker HyperPod AMI releases for Amazon EKS: February 18, 2025

#### Improvements for K8s

- Upgraded Nvidia container toolkit from version 1.17.3 to version 1.17.4.
- Fixed the issue where customers were unable to connect to nodes after a reboot.
- Upgraded Elastic Fabric Adapter (EFA) version from 1.37.0 to 1.38.0.
- The EFA now includes the AWS OFI NCCL plugin, which is located in the /opt/amazon/ofi-nccl directory instead of the original /opt/aws-ofi-nccl/ path. If you need to update your LD\_LIBRARY\_PATH environment variable, make sure to modify the path to point to the new /opt/amazon/ofi-nccl location for the OFI NCCL plugin.
- Removed the emacs package from these DLAMIs. You can install emacs from GNU emacs.

#### SageMaker HyperPod DLAMI for Amazon EKS support

Installed the latest version of neuron SDK

- **aws-neuronx-dkms.noarch:** 2.19.64.0-dkms @neuron
- **aws-neuronx-oci-hook.x86\_64:** 2.4.4.0-1 @neuron
- **aws-neuronx-tools.x86\_64:** 2.18.3.0-1 @neuron
- **aws-neuronx-collectives.x86\_64:** 2.23.135.0\_3e70920f2-1 neuron
- **aws-neuronx-gpsimd-customop.x86\_64:** 0.2.3.0-1 neuron

- **aws-neuronx-gpsimd-customop-lib.x86\_64**
- **aws-neuronx-gpsimd-tools.x86\_64:** 0.13.2.0\_94ba34927-1 neuron
- **aws-neuronx-k8-plugin.x86\_64:** 2.23.45.0-1 neuron
- **aws-neuronx-k8-scheduler.x86\_64:** 2.23.45.0-1 neuron
- **aws-neuronx-runtime-lib.x86\_64:** 2.23.112.0\_9b5179492-1 neuron
- **aws-neuronx-tools.x86\_64:** 2.20.204.0-1 neuron
- **tensorflow-model-server-neuronx.x86\_64**

## SageMaker HyperPod AMI releases for Amazon EKS: January 22, 2025

### AMI general updates

- New SageMaker HyperPod AMI for Amazon EKS 1.31.2.

### SageMaker HyperPod DLAMI for Amazon EKS support

The AMIs include the following:

#### Deep Learning EKS AMI 1.31

- **Amazon EKS Components**
  - Kubernetes Version: 1.31.2
  - Containerd Version: 1.7.23
  - Runc Version: 1.1.14
  - AWS IAM Authenticator: 0.6.26
- **Amazon SSM Agent:** 3.3.987
- **Linux Kernel:** 5.10.230
- **OSS Nvidia driver:** 550.127.05
- **NVIDIA CUDA:** 12.4
- **EFA Installer:** 1.37.0
- **GDRCopy:** 2.4.1-1
- **Nvidia container toolkit:** 1.17.3
- **AWS OFI NCCL:** 1.13.0

- **aws-neuronx-tools:** 2.18.3
- **aws-neuronx-runtime-lib:** 2.23.112.0
- **aws-neuronx-oci-hook:** 2.4.4.0-1
- **aws-neuronx-dkms:** 2.18.20.0
- **aws-neuronx-collectives:** 2.23.133.0

## SageMaker HyperPod AMI releases for Amazon EKS: December 21, 2024

### SageMaker HyperPod DLAMI for Amazon EKS support

The AMIs include the following:

K8s v1.28

- **Amazon EKS Components**
  - Kubernetes Version: 1.28.15
  - Containerd Version: 1.7.23
  - Runc Version: 1.1.14
  - AWS IAM Authenticator: 0.6.26
- **Amazon SSM Agent:** 3.3.987
- **Linux Kernel:** 5.10.228
- **OSS NVIDIA driver:** 550.127.05
- **NVIDIA CUDA:** 12.4
- **EFA Installer:** 1.37.0
- **GDRCopy:** 2.4
- **NVIDIA container toolkit:** 1.17.3
- **AWS OFI NCCL:** 1.13.0
- **aws-neuronx-tools:** 2.18.3.0-1
- **aws-neuronx-runtime-lib:** 2.23.112.0
- **aws-neuronx-oci-hook:** 2.4.4.0-1
- **aws-neuronx-dkms:** 2.18.20.0
- **aws-neuronx-collectives:** 2.23.135.0

## K8s v1.29

- **Amazon EKS Components**

- Kubernetes Version: 1.29.10
- Containerd Version: 1.7.23
- Runc Version: 1.1.14
- AWS IAM Authenticator: 0.6.26

- **Amazon SSM Agent:** 3.3.987

- **Linux Kernel:** 5.15.0
- **OSS Nvidia driver:** 550.127.05
- **NVIDIA CUDA:** 12.4
- **EFA Installer:** 1.37.0
- **GDRCopy:** 2.4
- **Nvidia container toolkit:** 1.17.3
- **AWS OFI NCCL:** 1.13.0
- **aws-neuronx-tools:** 2.18.3.0-1
- **aws-neuronx-runtime-lib:** 2.23.112.0
- **aws-neuronx-oci-hook:** 2.4.4.0-1
- **aws-neuronx-dkms:** 2.18.20.0
- **aws-neuronx-collectives:** 2.23.135.0

## K8s v1.30

- **Amazon EKS Components**

- Kubernetes Version: 1.30.6
- Containerd Version: 1.7.23
- Runc Version: 1.1.14
- AWS IAM Authenticator: 0.6.26

- **Amazon SSM Agent:** 3.3.987.0

- **Linux Kernel:** 5.10.228

- **OSS Nvidia driver:** 550.127.05

- **NVIDIA CUDA:** 12.4
- **EFA Installer:** 1.37.0
- **GDRCopy:** 2.4
- **Nvidia container toolkit:** 1.17.3
- **AWS OFI NCCL:** 1.13.0
- **aws-neuronx-tools:** 2.18.3.0-1
- **aws-neuronx-runtime-lib:** 2.23.112.0
- **aws-neuronx-oci-hook:** 2.4.4.0-1
- **aws-neuronx-dkms:** 2.18.20.0
- **aws-neuronx-collectives:** 2.23.135.0

## SageMaker HyperPod AMI releases for Amazon EKS: December 13, 2024

### SageMaker HyperPod DLAMI for Amazon EKS upgrade

- Updated SSM Agent to version 3.3.1311.0.

## SageMaker HyperPod AMI releases for Amazon EKS: November 24, 2024

### AMI general updates

- Released in MEL (Melbourne) Region.
- Updated SageMaker HyperPod base DLAMI to the following versions:
  - Kubernetes: 2024-11-01.

## SageMaker HyperPod AMI releases for Amazon EKS: November 15, 2024

### SageMaker HyperPod DLAMI for Amazon EKS support

The AMIs include the following:

#### Deep Learning EKS AMI 1.28

- **Amazon EKS Components**
  - Kubernetes Version: 1.28.15

- Containerd Version: 1.7.23
- Runc Version: 1.1.14
- AWS IAM Authenticator: 0.6.26
- **Amazon SSM Agent:** 3.3.987
- **Linux Kernel:** 5.10.228
- **OSS NVIDIA driver:** 550.127.05
- **NVIDIA CUDA:** 12.4
- **EFA Installer:** 1.34.0
- **GDRCopy:** 2.4
- **NVIDIA container toolkit:** 1.17.3
- **AWS OFI NCCL:** 1.11.0
- **aws-neuronx-tools:** 2.18.3.0-1
- **aws-neuronx-runtime-lib:** 2.22.19.0
- **aws-neuronx-oci-hook:** 2.4.4.0-1
- **aws-neuronx-dkms:** 2.18.20.0
- **aws-neuronx-collectives:** 2.22.33.0

## Deep Learning EKS AMI 1.29

- **Amazon EKS Components**
  - Kubernetes Version: 1.29.10
  - Containerd Version: 1.7.23
  - Runc Version: 1.1.14
  - AWS IAM Authenticator: 0.6.26
- **Amazon SSM Agent:** 3.3.987
- **Linux Kernel:** 5.10.228
- **OSS Nvidia driver:** 550.127.05
- **NVIDIA CUDA:** 12.4
- **EFA Installer:** 1.34.0
- **GDRCopy:** 2.4
- **Nvidia container toolkit:** 1.17.3

- **AWS OFI NCCL:** 1.11.0
- **aws-neuronx-tools:** 2.18.3.0-1
- **aws-neuronx-runtime-lib:** 2.22.19.0
- **aws-neuronx-oci-hook:** 2.4.4.0-1
- **aws-neuronx-dkms:** 2.18.20.0
- **aws-neuronx-collectives:** 2.22.33.0

## Deep Learning EKS AMI 1.30

- **Amazon EKS Components**
  - Kubernetes Version: 1.30.6
  - Containerd Version: 1.7.23
  - Runc Version: 1.1.14
  - AWS IAM Authenticator: 0.6.26
- **Amazon SSM Agent:** 3.3.987
- **Linux Kernel:** 5.10.228
- **OSS Nvidia driver:** 550.127.05
- **NVIDIA CUDA:** 12.4
- **EFA Installer:** 1.34.0
- **GDRCopy:** 2.4
- **Nvidia container toolkit:** 1.17.3
- **AWS OFI NCCL:** 1.11.0
- **aws-neuronx-tools:** 2.18.3.0-1
- **aws-neuronx-runtime-lib:** 2.22.19.0
- **aws-neuronx-oci-hook:** 2.4.4.0-1
- **aws-neuronx-dkms:** 2.18.20.0
- **aws-neuronx-collectives:** 2.22.33.0

**SageMaker HyperPod AMI releases for Amazon EKS: November 11, 2024**

### AMI general updates

- Updated SageMaker HyperPod DLAMI with Amazon EKS versions 1.28.13, 1.29.8, 1.30.4.

## SageMaker HyperPod AMI releases for Amazon EKS: October 21, 2024

### AMI general updates

- Updated SageMaker HyperPod base DLAMI to the following versions:
  - Amazon EKS: 1.28.11, 1.29.6, 1.30.2.

## SageMaker HyperPod AMI releases for Amazon EKS: September 10, 2024

### SageMaker HyperPod DLAMI for Amazon EKS support

The AMIs include the following:

#### Deep Learning EKS AMI 1.28

- **Amazon EKS Components**
  - Kubernetes Version: 1.28.11
  - Containerd Version: 1.7.20
  - Runc Version: 1.1.11
  - AWS IAM Authenticator: 0.6.21
- **Amazon SSM Agent:** 3.3.380
- **Linux Kernel:** 5.10.223
- **OSS NVIDIA driver:** 535.183.01
- **NVIDIA CUDA:** 12.2
- **EFA Installer:** 1.32.0
- **GDRCopy:** 2.4
- **NVIDIA container toolkit:** 1.16.1
- **AWS OFI NCCL:** 1.9.1
- **aws-neuronx-tools:** 2.18.3.0-1
- **aws-neuronx-runtime-lib:** 2.21.41.0
- **aws-neuronx-oci-hook:** 2.4.4.0-1
- **aws-neuronx-dkms:** 2.17.17.0
- **aws-neuronx-collectives:** 2.21.46.0

## Deep Learning EKS AMI 1.29

- **Amazon EKS Components**

- Kubernetes Version: 1.29.6
- Containerd Version: 1.7.20
- Runc Version: 1.1.11
- AWS IAM Authenticator: 0.6.21
- **Amazon SSM Agent:** 3.3.380
- **Linux Kernel:** 5.10.223
- **OSS Nvidia driver:** 535.183.01
- **NVIDIA CUDA:** 12.2
- **EFA Installer:** 1.32.0
- **GDRCopy:** 2.4
- **Nvidia container toolkit:** 1.16.1
- **AWS OFI NCCL:** 1.9.1
- **aws-neuronx-tools:** 2.18.3.0-1
- **aws-neuronx-runtime-lib:** 2.21.41.0
- **aws-neuronx-oci-hook:** 2.4.4.0-1
- **aws-neuronx-dkms:** 2.17.17.0
- **aws-neuronx-collectives:** 2.21.46.0

## Deep Learning EKS AMI 1.30

- **Amazon EKS Components**

- Kubernetes Version: 1.30.2
- Containerd Version: 1.7.20
- Runc Version: 1.1.11
- AWS IAM Authenticator: 0.6.21
- **Amazon SSM Agent:** 3.3.380
- **Linux Kernel:** 5.10.223
- **OSS Nvidia driver:** 535.183.01
- **NVIDIA CUDA:** 12.2

- **EFA Installer:** 1.32.0
- **GDRCopy:** 2.4
- **Nvidia container toolkit:** 1.16.1
- **AWS OFI NCCL:** 1.9.1
- **aws-neuronx-tools:** 2.18.3.0-1
- **aws-neuronx-runtime-lib:** 2.21.41.0
- **aws-neuronx-oci-hook:** 2.4.4.0-1
- **aws-neuronx-dkms:** 2.17.17.0
- **aws-neuronx-collectives:** 2.21.46.0

## Generative AI in SageMaker notebook environments

[Jupyter AI](#) is an open-source extension of JupyterLab integrating generative AI capabilities into Jupyter notebooks. Through the Jupyter AI chat interface and magic commands, users experiment with code generated from natural language instructions, explain existing code, ask questions about their local files, generate entire notebooks, and more. The extension connects Jupyter notebooks with large language models (LLMs) that users can use to generate text, code, or images, and to ask questions about their own data. Jupyter AI supports generative model providers such as AI21, Anthropic, AWS (JumpStart and Amazon Bedrock), Cohere, and OpenAI.

You can also use Amazon Q Developer as an out of the box solution. Instead of having to manually set up a connection to a model, you can start using Amazon Q Developer with minimal configuration. When you enable Amazon Q Developer, it becomes the default solution provider within Jupyter AI. For more information about using Amazon Q Developer, see [SageMaker JupyterLab](#).

The extension's package is included in [Amazon SageMaker Distribution version 1.2 and onwards](#). Amazon SageMaker Distribution is a Docker environment for data science and scientific computing used as the default image of JupyterLab notebook instances. Users of different IPython environments can install Jupyter AI manually.

In this section, we provide an overview of Jupyter AI capabilities and demonstrate how to configure models provided by JumpStart or Amazon Bedrock from [JupyterLab](#) or [Studio Classic](#) notebooks. For more in-depth information on the Jupyter AI project, refer to its [documentation](#). Alternatively, you can refer to the blog post [Generative AI in Jupyter](#) for an overview and examples of key Jupyter AI capabilities.

Before using Jupyter AI and interacting with your LLMs, make sure that you satisfy the following prerequisites:

- For models hosted by AWS, you should have the ARN of your SageMaker AI endpoint or have access to Amazon Bedrock. For other model providers, you should have the API key used to authenticate and authorize requests to your model. Jupyter AI supports a wide range of model providers and language models, refer to the list of its [supported models](#) to stay updated on the latest available models. For information on how to deploy a model in JumpStart, see [Deploy a Model](#) in the JumpStart documentation. You need to request access to [Amazon Bedrock](#) to use it as your model provider.
- Ensure that Jupyter AI libraries are present in your environment. If not, install the required package by following the instructions in [Jupyter AI installation](#).
- Familiarize yourself with the capabilities of Jupyter AI in [Access Jupyter AI Features](#).
- Configure the target models you wish to use by following the instructions in [Configure your model provider](#).

After completing the prerequisite steps, you can proceed to [Use Jupyter AI in JupyterLab or Studio Classic](#).

## Topics

- [Jupyter AI installation](#)
- [Access Jupyter AI Features](#)
- [Configure your model provider](#)
- [Use Jupyter AI in JupyterLab or Studio Classic](#)

## Jupyter AI installation

To use Jupyter AI, you must first install the Jupyter AI package. For [Amazon SageMaker AI Distribution](#) users, we recommend selecting the SageMaker Distribution image version 1.2 or later. No further installation is necessary. Users of JupyterLab in Studio can choose the version of their Amazon SageMaker Distribution when creating a space.

For users of other IPython environments, the version of the recommended Jupyter AI package depends on the version of JupyterLab they are using.

The Jupyter AI distribution consists of two packages.

- **jupyter\_ai**: This package provides a JupyterLab extension and a native chat user interface (UI). It acts as a conversational assistant using the large language model of your choice.
- **jupyter\_ai\_magics**: This package provides the IPython `%%ai` and `%ai` magic commands with which you can invoke a large language model (LLM) from your notebook cells.

### Note

Installing `jupyter_ai` also installs `jupyter_ai_magics`. However, you can install `jupyter_ai_magics` independently without JupyterLab or `jupyter_ai`. The magic commands `%%ai` and `%ai` work in any IPython kernel environment. If you only install `jupyter_ai_magics`, you can't use the chat UI.

For users of JupyterLab 3, in particular Studio Classic users, we recommend installing `jupyter-ai` [version 1.5.x](#) or any later 1.x version. However, we highly recommend using Jupyter AI with JupyterLab 4. The `jupyter-ai` version compatible with JupyterLab 3 may not allow users to set additional model parameters such as temperature, top-k and top-p sampling, max tokens or max length, or user acceptance license agreements.

For users of JupyterLab 4 environments that do not use SageMaker Distribution, we recommend installing `jupyter-ai` [version 2.5.x](#) or any later 2.x version.

See the installation instructions in the *Installation* section of [Jupyter AI documentation](#).

## Access Jupyter AI Features

You can access Jupyter AI capabilities through two distinct methods: using the chat UI or using magic commands within notebooks.

### From the chat user interface AI assistant

The chat interface connects you with JupyterNaut, a conversational agent that uses the language model of your choice.

After launching a JupyterLab application installed with Jupyter AI, you can access the chat interface by choosing the chat icon



)

in the left navigation panel. First-time users are prompted to configure their model. See [Configure your model provider in the chat UI](#) for configuration instructions.

### Using the chat UI, you can:

- **Answer questions:** For instance, you can ask Jupyternaut to create a Python function that adds CSV files to an Amazon S3 bucket. Subsequently, you can refine your answer with a follow-up question, such as adding a parameter to the function to choose the path where the files are written.
- **Interact with files in JupyterLab:** You can include a portion of your notebook in your prompt by selecting it. Then, you can either replace it with the model's suggested answer or manually copy the answer to your clipboard.
- **Generate entire notebooks from prompts:** By starting your prompt with /generate, you trigger a notebook generation process in the background without interrupting your use of Jupyternaut. A message containing the link to the new file is displayed upon completion of the process.
- **Learn from and ask questions about local files:** Using the /learn command, you can teach an embedding model of your choice about local files and then ask questions about those files using the /ask command. Jupyter AI stores the embedded content in a local [FAISS vector database](#), then uses retrieval-augmented generation (RAG) to provide answers based on what it has learned. To erase all previously learned information from your embedding model, use / learn -d.

 **Note**

Amazon Q developer doesn't have the capability to generate notebooks from scratch.

For a complete list of features and detailed instructions on their usage, see the [Jupyter AI chat interface](#) documentation. To learn about how to configure access to a model in Jupyternaut, see [Configure your model provider in the chat UI](#).

### From notebook cells

Using %%ai and %ai magic commands, you can interact with the language model of your choice from your notebook cells or any IPython command line interface. The %%ai command applies your instructions to the entire cell, whereas %ai apply them to the specific line.

The following example illustrates an `%%ai` magic command invoking an Anthropic Claude model to output an HTML file containing the image of a white square with black borders.

```
%%ai anthropic:claude-v1.2 -f html  
Create a square using SVG with a black border and white fill.
```

To learn about the syntax of each command, use `%ai help`. To list the providers and models supported by the extension, run `%ai list`.

For a complete list of features and detailed instructions on their usage, see the Jupyter AI [magic commands](#) documentation. In particular, you can customize the output format of your model using the `-f` or `--format` parameter, allow variable interpolation in prompts, including special `In` and `Out` variables, and more.

To learn about how to configure the access to a model, see [Configure your model provider in a notebook](#).

## Configure your model provider

### Note

In this section, we assume that the language and embedding models that you plan to use are already deployed. For models provided by AWS, you should already have the ARN of your SageMaker AI endpoint or access to Amazon Bedrock. For other model providers, you should have the API key used to authenticate and authorize requests to your model.

Jupyter AI supports a wide range of model providers and language models, refer to the list of its [supported models](#) to stay updated on the latest available models. For information on how to deploy a model provided by JumpStart, see [Deploy a Model](#) in the JumpStart documentation. You need to request access to [Amazon Bedrock](#) to use it as your model provider.

The configuration of Jupyter AI varies depending on whether you are using the chat UI or magic commands.

## Configure your model provider in the chat UI

### Note

You can configure several LLMs and embedding models following the same instructions. However, you must configure at least one **Language model**.

### To configure your chat UI

1. In JupyterLab, access the chat interface by choosing the chat icon



)

in the left navigation panel.

2. Choose the configuration icon



)

in the top right corner of the left pane. This opens the Jupyter AI configuration panel.

3. Fill out the fields related to your service provider.

- **For models provided by JumpStart or Amazon Bedrock**

- In the **language model** dropdown list, select `sagemaker-endpoint` for models deployed with JumpStart or `bedrock` for models managed by Amazon Bedrock.

- The parameters differ based on whether your model is deployed on SageMaker AI or Amazon Bedrock.

- For models deployed with JumpStart:

- Enter the name of your endpoint in **Endpoint name**, and then the AWS Region in which your model is deployed in **Region name**. To retrieve the ARN of the SageMaker AI endpoints, navigate to <https://console.aws.amazon.com/sagemaker/> and then choose **Inference** and **Endpoints** in the left menu.

- Paste the JSON of the **Request schema** tailored to your model, and the corresponding **Response path** for parsing the model's output.

**Note**

You can find the request and response format of various of JumpStart foundation models in the following [example notebooks](#). Each notebook is named after the model it demonstrates.

- For models managed by Amazon Bedrock: Add the AWS profile storing your AWS credentials on your system (optional), and then the AWS Region in which your model is deployed in [Region name](#).
- (Optional) Select an [embedding model](#) to which you have access. Embedding models are used to capture additional information from local documents, enabling the text generation model to respond to questions within the context of those documents.
- Choose **Save Changes** and navigate to the left arrow icon



in the top left corner of the left pane. This opens the Jupyter AI chat UI. You can start interacting with your model.

- **For models hosted by third-party providers**

- In the **language model** dropdown list, select your provider ID. You can find the details of each provider, including their ID, in Jupyter AI [list of model providers](#).
- (Optional) Select an [embedding model](#) to which you have access. Embedding models are used to capture additional information from local documents, enabling the text generation model to respond to questions within the context of those documents.
- Insert your models' API keys.
- Choose **Save Changes** and navigate to the left arrow icon



in the top left corner of the left pane. This opens the Jupyter AI chat UI. You can start interacting with your model.

The following snapshot is an illustration of the chat UI configuration panel set to invoke a Flan-t5-small model provided by JumpStart and deployed in SageMaker AI.

## Language model

Language model

SageMaker endpoint :: \*

Endpoint name

hf-text2text-flan-t5-small

Specify an endpoint name as the model ID. In addition, you must specify a region name, request schema, and response path. For more information, see the documentation about [SageMaker endpoints deployment](#) and about [using magic commands with SageMaker endpoints](#).

Region name (required)

us-west-2

Request schema (required)

{"inputs": "<prompt>"}

Response path (required)

[0].["generated\_text"]

## Embedding model

Embedding model

None

## API Keys

### Input

When writing a message, press Enter to:

- Send the message
- Start a new line (use Shift+Enter to send)

**Save Changes**

## Pass extra model parameters and custom parameters to your request

Your model may need extra parameters, like a customized attribute for user agreement approval or adjustments to other model parameters such as temperature or response length. We recommend configuring these settings as a start up option of your JupyterLab application using a Lifecycle Configuration. For information on how to create a Lifecycle Configuration and attach it to your domain, or to a user profile from the [SageMaker AI console](#), see [Create and associate a lifecycle configuration](#). You can choose your LCC script when creating a space for your JupyterLab application.

Use the following JSON schema to configure your [extra parameters](#):

```
{  
  "AiExtension": {  
    "model_parameters": {  
      "<provider_id>:<model_id>": { Dictionary of model parameters which is unpacked  
        and passed as-is to the provider.  
      }  
    }  
  }  
}
```

The following script is an example of a JSON configuration file that you can use when creating a JupyterLab application LCC to set the maximum length of an [AI21 Labs Jurassic-2 model](#) deployed on Amazon Bedrock. Increasing the length of the model's generated response can prevent the systematic truncation of your model's response.

```
#!/bin/bash  
set -eux  
  
mkdir -p /home/sagemaker-user/.jupyter  
  
json='{"AiExtension": {"model_parameters": {"bedrock:ai21.j2-mid-v1": {"model_kwargs":  
  {"maxTokens": 200}}}}}'  
# equivalent to %%ai bedrock:ai21.j2-mid-v1 -m {"model_kwargs": {"maxTokens": 200}}  
  
# File path  
file_path="/home/sagemaker-user/.jupyter/jupyter_ai_config.json"  
  
#jupyter --paths
```

```
# Write JSON to file
echo "$json" > "$file_path"

# Confirmation message
echo "JSON written to $file_path"

restart-jupyter-server

# Waiting for 30 seconds to make sure the Jupyter Server is up and running
sleep 30
```

The following script is an example of a JSON configuration file for creating a JupyterLab application LCC used to set additional model parameters for an [Anthropic Claude model](#) deployed on Amazon Bedrock.

```
#!/bin/bash
set -eux

mkdir -p /home/sagemaker-user/.jupyter

json='{"AiExtension": {"model_parameters": {"bedrock:anthropic.claude-v2": {"model_kwargs": {"temperature":0.1,"top_p":0.5,"top_k":250,"max_tokens_to_sample":2}}}}}'
# equivalent to %ai bedrock:anthropic.claude-v2 -m {"model_kwargs": {"temperature":0.1,"top_p":0.5,"top_k":250,"max_tokens_to_sample":2000} }

# File path
file_path="/home/sagemaker-user/.jupyter/jupyter_ai_config.json"

#jupyter --paths

# Write JSON to file
echo "$json" > "$file_path"

# Confirmation message
echo "JSON written to $file_path"

restart-jupyter-server

# Waiting for 30 seconds to make sure the Jupyter Server is up and running
sleep 30
```

Once you have attached your LCC to your domain, or user profile, add your LCC to your space when launching your JupyterLab application. To ensure that your configuration file is updated by the LCC, run more `~/.jupyter/jupyter_ai_config.json` in a terminal. The content of the file should correspond to the content of the JSON file passed to the LCC.

## Configure your model provider in a notebook

**To invoke a model via Jupyter AI within JupyterLab or Studio Classic notebooks using the `%%ai` and `%ai` magic commands**

1. Install the client libraries specific to your model provider in your notebook environment. For example, when using OpenAI models, you need to install the `openai` client library. You can find the list of the client libraries required per provider in the *Python package(s)* column of the Jupyter AI [Model providers list](#).

 **Note**

For models hosted by AWS, `boto3` is already installed in the SageMaker AI Distribution image used by JupyterLab, or any Data Science image used with Studio Classic.

2. • **For models hosted by AWS**

Ensure that your execution role has the permission to invoke your SageMaker AI endpoint for models provided by JumpStart or that you have access to Amazon Bedrock.

- **For models hosted by third-party providers**

Export your provider's API key in your notebook environment using environment variables. You can use the following magic command. Replace the `provider_API_key` in the command by the environment variable found in the *Environment variable* column of the Jupyter AI [Model providers list](#) for your provider.

```
%env provider_API_key=your_API_key
```

## Use Jupyter AI in JupyterLab or Studio Classic

You can use Jupyter AI in JupyterLab or Studio Classic by invoking language models from either the chat UI or from notebook cells. The following sections give information about the steps needed to complete this.

## Use language models from the chat UI

Compose your message in the chat UI text box to start interacting with your model. To clear the message history, use the `/clear` command.

 **Note**

Clearing the message history does not erase the chat context with the model provider.

## Use language models from notebook cells

Before using the `%%ai` and `%ai` commands to invoke a language model, load the IPython extension by running the following command in a JupyterLab or Studio Classic notebook cell.

```
%load_ext jupyter_ai_magics
```

- **For models hosted by AWS:**

- To invoke a model deployed in SageMaker AI, pass the string `sagemaker-endpoint: endpoint-name` to the `%%ai` magic command with the required parameters below, then add your prompt in the following lines.

The following table lists the required and optional parameters when invoking models hosted by SageMaker AI or Amazon Bedrock.

Parameter Name	Parameter	Short Version	Description
Request schema	<code>--request-schema</code>	<code>-q</code>	<b>Required:</b> The JSON object the endpoint expects, with the prompt being substituted into any value that matches the string literal <code>&lt;prompt&gt;</code> .

Parameter Name	Parameter	Short Version	Description
Region name	--region-name	-n	<b>Required:</b> The AWS Region where the model is deployed.
Response path	--response-path	-p	<b>Required:</b> A JSONPath string used to extract the language model's output from the JSON response of the endpoint.

Parameter Name	Parameter	Short Version	Description
Extra model parameters	--model-parameters	-m	<b>Optional:</b> A JSON value specifying additional parameters to be passed to the model. The accepted value is parsed into a dictionary, unpacked, and directly passed to the provider class. This is useful when the endpoint or the model requires custom parameters. For example, in Llama 2 models when accepting the End User License Agreement (EULA) is necessary, you can pass the EULA acceptance to the endpoint using -m {"endpoint_kwargs": {"CustomAttribute": "accept_eula": true}} . Alternatively, you can use the -m parameter to pass extra model

Parameter Name	Parameter	Short Version	Description
			parameters, such as setting the maximum number of tokens for a model's generated response. For example, when working with an AI21 Labs Jurassic model: -m {"model_kwargs":{"maxTokens":256}} .
Output format	--format	-f	<b>Optional:</b> The IPython display used to render the output. It can be any of the following values [code   html   image   json   markdown   math   md   text], provided that the invoked model supports the specified format.

The following command invokes a [Llama2-7b](#) model hosted by SageMaker AI.

```
%%ai sagemaker-endpoint:jumpstart-dft-meta-textgeneration-llama-2-7b -q
{"inputs":"<prompt>","parameters":
{"max_new_tokens":64,"top_p":0.9,"temperature":0.6,"return_full_text":false}}
```

```
-n us-east-2 -p [0].generation -m {"endpoint_kwargs":  
{"CustomAttributes":"accept_eula=true"}} -f text  
Translate English to French:  
sea otter => loutre de mer  
peppermint => menthe poivrée  
plush girafe => girafe peluche  
cheese =>
```

The following example invokes a Flan-t5-small model hosted by SageMaker AI.

```
%%ai sagemaker-endpoint:hf-text2text-flan-t5-small --request-  
schema={"inputs":"<prompt>","parameters":{"num_return_sequences":4}} --region-  
name=us-west-2 --response-path=[0]["generated_text"] -f text  
What is the atomic number of Hydrogen?
```

- To invoke a model deployed in Amazon Bedrock, pass the string `bedrock:provider-id:provider-name` to the `%%ai` magic command with any optional parameter defined in the list of [parameters for invoking models hosted by JumpStart or Amazon Bedrock](#), then add your prompt in the following lines.

The following example invokes an [AI21 Labs Jurassic-2 model](#) hosted by Amazon Bedrock.

```
%%ai bedrock:ai21.j2-mid-v1 -m {"model_kwargs":{"maxTokens":256}} -f code  
Write a function in python implementing a bubble sort.
```

#### For models hosted by third-party providers

To invoke a model hosted by third-party providers, pass the string `provider-id:provider-name` to the `%%ai` magic command with an optional [Output format](#), then add your prompt in the following lines. You can find the details of each provider, including their ID, in the Jupyter AI [list of model providers](#).

The following command asks an Anthropic Claude model to output an HTML file containing the image of a white square with black borders.

```
%%ai anthropic:claude-v1.2 -f html  
Create a square using SVG with a black border and white fill.
```

# Amazon Q Developer

Amazon Q Developer is a generative AI conversational assistant that helps you write better code. Amazon Q Developer is available in the following IDEs within Amazon SageMaker Studio:

- JupyterLab
- Code Editor, based on Code-OSS, Visual Studio Code - Open Source

Use the following sections to set up Amazon Q Developer and use it within your environment.

## Topics

- [Set up Amazon Q Developer for your users](#)
- [Use Amazon Q to Expedite Your Machine Learning Workflows](#)

## Set up Amazon Q Developer for your users

Amazon Q Developer is a generative AI conversational assistant. You can set up Amazon Q Developer within a new domain or an existing domain. Use the following information to set up Amazon Q Developer.

With Amazon Q Developer, your users can:

- Receive step-by-step guidance on using SageMaker AI features independently or in combination with other AWS services.
- Get sample code to get started on your ML tasks such as data preparation, training, inference, and MLOps.
- Receive troubleshooting assistance to debug and resolve errors encountered while running code.

### Note

Amazon Q Developer in Studio doesn't use user content to improve the service, regardless of whether you use the Free-tier or Pro-tier subscription. For IDE-level telemetry sharing, Amazon Q might track your users' usage, such as the number of questions asked and whether recommendations were accepted or rejected. This telemetry data doesn't include

personally identifiable information such as the users' IP address. For more information on data protection and instructions for opting out, see [Opt out of data sharing in the IDE](#).

You can set up Amazon Q Developer with either a Pro or Free tier subscription. The Pro tier is a paid subscription service with higher usage limits and other features. For more information about the differences between the tiers, see [Understanding tiers of service for Amazon Q Developer](#).

**⚠️ Important**

Code Editor, based on Code-OSS, Visual Studio Code - Open Source only supports using a Free tier subscription.

For information about subscribing to Amazon Q Developer Pro, see [Subscribing to Amazon Q Developer Pro](#).

### Set up instructions for Amazon Q Developer Free Tier:

To set up Amazon Q Developer Free Tier, use the following procedure:

#### To set up Amazon Q Developer Free Tier

1. Add the following policy to the IAM role that you've used to create your JupyterLab or Code Editor space:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "q:SendMessage"  
      ],  
      "Resource": [  
        "*"  
      ]  
    },  
    {  
      "Sid": "AmazonQDeveloperPermissions",  
      "Effect": "Allow",  
      "Action": [  
        "q:SendMessage"  
      ],  
      "Resource": [  
        "*"  
      ]  
    }  
  ]  
}
```

```
"Action": [
    "codewhisperer:GenerateRecommendations"
],
"Resource": "*"
}
]
}
```

2. Navigate to Amazon SageMaker Studio.
3. Open your JupyterLab or Code Editor space.
4. Navigate to the **Launcher** and choose **Terminal**.
5. In JupyterLab, do the following:
  - a. Specify `restart-jupyter-server`.
  - b. Restart your browser and navigate back to Amazon SageMaker Studio.

## Set up instructions for Amazon Q Developer Pro tier:

### Prerequisites

To set up Amazon Q Pro, you must have:

- An Amazon SageMaker AI domain set up for your organization with IAM Identity Center configured as the means of access.
- An Amazon Q Developer Pro subscription.

If you're updating a domain that you've already set up for your organization, you need to update it to use Amazon Q Developer. You can use either the AWS Management Console or the AWS Command Line Interface to update a domain.

You must use the ARN of your Amazon Q Developer profile. You can find the Q Profile ARN on the [Q Developer Settings](#) page.

You can use the following AWS Command Line Interface command to update your domain:

```
aws --region AWS Region sagemaker update-domain --domain-id domain-id --domain-settings-for-update "AmazonQSettings={Status=ENABLED,QProfileArn=Q-Profile-ARN}"
```

You can also use the following procedure to update the domain within the AWS Management Console.

1. Navigate to the [Amazon SageMaker AI](#) console.
2. Choose domains.
3. Select **App Configurations**.
4. For **Amazon Q Developer for SageMaker AI Applications**, choose **Edit**.
5. Select **Enable Amazon Q Developer on this domain**.
6. Provide the Q Profile ARN.
7. Choose **Submit**.

You must use the ARN of your Amazon Q Developer profile. You can find the ARN of the Q Profile on the [Amazon Q account details](#) page of the [Amazon Q Developer](#) console.

The **Set up for organizations** is an advanced setup for the Amazon SageMaker AI domain that lets you use IAM Identity Center. For information about how you can set up the domain and information about setting up IAM Identity Center, see [Use custom setup for Amazon SageMaker AI](#).

When setting up Amazon Q Developer in a new domain, you can either use the AWS Management Console or the following AWS Command Line Interface command from your local machine:

```
aws --region AWS Region sagemaker create-domain --domain-id domain-id --domain-name "example-domain-name" --vpc-id example-vpc-id --subnet-ids example-subnet-ids --auth-mode SSO --default-user-settings "ExecutionRole=arn:aws:iam::111122223333:role/IAM-role", --domain-settings "AmazonQSettings={status=ENABLED,qProfileArn=Q-profile-ARN}" --query example-domain-ARN--output text
```

You can use the following AWS CLI command to disable Amazon Q Developer:

```
aws --region AWS Region sagemaker update-domain --domain-id domain-id --domain-settings-for-update "AmazonQSettings={Status=DISABLED,QProfileArn=Q-Profile-ARN}"
```

You can set up Amazon Q Developer within a new domain or an existing domain. Use the following information to set up Amazon Q Developer.

We recommend using the latest version of the AWS Command Line Interface. For information about updating the AWS CLI, see [Install or update to the latest version of the AWS Command Line Interface](#).

If you need to establish a connection between Amazon Q Developer and your VPC, see [Creating an interface VPC endpoint for Amazon Q](#).

### Note

Amazon Q Developer has the following limitations:

- It doesn't support shared spaces.
- Amazon Q Developer detects whether a code suggestion might be too similar to publicly available code. The reference tracker can flag suggestions with repository URLs and licenses, or filter them out. This allows you to review the referenced code and its usage before you adopt it. All references are logged for you to review later to ensure that your code flow is not disturbed and that you can keep coding without interruption.

For more information about code references, see [Using code references - Amazon Q Developer](#) and [AI Coding Assistant - Amazon Q Developer FAQs](#).

- Amazon Q processes all user interaction data within the US East (N. Virginia) AWS Region. For more information about how Amazon Q processes data and the AWS Regions it supports, see [Supported Regions for Amazon Q Developer](#).
- Amazon Q only works within Amazon SageMaker Studio. It is not supported within Amazon SageMaker Studio Classic.
- On JupyterLab, Amazon Q works within SageMaker AI Distribution Images version 2.0 and above. On Code Editor, Amazon Q works within SageMaker AI Distribution Images version 2.2.1 and above.

- Amazon Q Developer in JupyterLab works within the Jupyter AI extension. You can't use other 3P models within the extension while you're using Amazon Q.

## Use Amazon Q to Expedite Your Machine Learning Workflows

Amazon Q Developer is your AI-powered companion for machine learning development. With Amazon Q Developer, you can:

- Receive step-by-step guidance on using SageMaker AI features independently or in combination with other AWS services.
- Get sample code to get started on your ML tasks such as data preparation, training, inference, and MLOps.

To use Amazon Q Developer, choose the **Q** from the left-hand navigation of your JupyterLab or Code Editor environment.

If you don't see the **Q** icon, your administrator needs to set it up for you. For more information about setting up Amazon Q Developer, see [Set up Amazon Q Developer for your users](#).

Amazon Q automatically provides suggestions to help you write your code. You can also ask for suggestions through the chat interface.

## Amazon SageMaker Partner AI Apps overview

With Amazon SageMaker Partner AI Apps, users get access to generative AI and machine learning (ML) development applications built, published, and distributed by industry-leading application providers. Partner AI Apps are certified to run on SageMaker AI. With Partner AI Apps, users can accelerate and improve how they build solutions based on foundation models (FM) and classic ML models without compromising the security of their sensitive data. The data stays completely within their trusted security configuration and is never shared with a third party.

## How it works

Partner AI Apps are full application stacks that include an Amazon Elastic Kubernetes Service cluster and an array of accompanying services that can include Application Load Balancer, Amazon Relational Database Service, Amazon Simple Storage Service buckets, Amazon Simple Queue Service queues, and Redis caches.

These service applications can be shared across all users in a SageMaker AI domain and are provisioned by an admin. After provisioning the application by purchasing a subscription through the AWS Marketplace, the admin can give users in the SageMaker AI domain permissions to access the Partner AI App directly from Amazon SageMaker Studio, Amazon SageMaker Unified Studio (preview), or using a pre-signed URL. For information about launching an application from Studio, see [Launch Amazon SageMaker Studio](#).

Partner AI Apps offers the following benefits for administrators and users.

- Administrators use the SageMaker AI console to browse, discover, select, and provision the Partner AI Apps for use by their data science and ML teams. After the Partner AI Apps are deployed, SageMaker AI runs them on service-managed AWS accounts. This significantly reduces the operational overhead associated with building and operating these applications, and contributes to the security and privacy of customer data.
- Data scientists and ML developers can access Partner AI Apps from within their ML development environment in Amazon SageMaker Studio or Amazon SageMaker Unified Studio (preview). They can use the Partner AI Apps to analyze their data, experiments, and models created on SageMaker AI. This minimizes context switching and helps accelerate building foundation models and bringing new generative AI capabilities to market.

## Integration with AWS services

Partner AI Apps uses the existing AWS Identity and Access Management (IAM) configuration for authorization and authentication. As a result, users don't need to provide separate credentials to access each Partner AI App from Amazon SageMaker Studio. For more information about authorization and authentication with Partner AI Apps, see [Set up Partner AI Apps](#).

Partner AI Apps also integrates with Amazon CloudWatch to provide operational monitoring and management. Customers can also browse Partner AI Apps, and get details about them, such as features, customer experience, and pricing, from the AWS Management Console. For information about Amazon CloudWatch, see [How Amazon CloudWatch works](#).

## Supported types

Partner AI Apps support the following types:

- Comet
- Deepchecks

- Fiddler
- Lakera Guard

When the admin launches a Partner AI App, they must select the configuration of the instance cluster that the Partner AI App is launched with. This configuration is known as the Partner AI App's tier. A Partner AI App's tier can be one of the following values:

- small
- medium
- large

The following sections give information about each of the Partner AI App types, and details about the Partner AI App's tier values.

## Comet overview

Comet provides an end-to-end model evaluation platform for AI developers, with LLM evaluations, experiment tracking, and production monitoring.

We recommend the following Partner AI App tiers based on the workload:

- small – Recommended for up to 5 users and 20 running jobs.
- medium – Recommended for up to 50 users and 100 running jobs.
- large – Recommended for up to 500 users and more than 100 running jobs.

 **Note**

SageMaker AI does not support viewing the Comet UI as part of the output of a Jupyter notebook.

## Deepchecks overview

AI application developers and stakeholders can use Deepchecks to continuously validate LLM-based applications including characteristics, performance metrics, and potential pitfalls throughout the entire lifecycle from pre-deployment and internal experimentation to production.

We recommend the following Partner AI App tiers based on the speed desired for the workload:

- **small** – Processes 200 tokens per second.
- **medium** – Processes 500 tokens per second.
- **large** – Processes 1300 tokens per second.

## Fiddler overview

The Fiddler AI Observability Platform facilitates validating, monitoring, and analyzing ML models in production, including tabular, deep learning, computer vision, and natural language processing models.

We recommend the following Partner AI App tiers based on the speed desired for the workload:

- **small** – Processing 10MM events across 5 models, 100 features, and 20 iterations takes about 53 minutes.
- **medium** – Processing 10MM events across 5 models, 100 features, and 20 iterations takes about 23 minutes.
- **large** – Processing 10MM events across 5 models, 100 features, and 100 iterations takes about 27 minutes.

## Lakera Guard overview

Lakera Guard is a low-latency AI application firewall to secure generative AI applications from gen AI-specific threats.

We recommend the following Partner AI App tiers based on the workload:

- **small** – Recommended for up to 20 Robotic Process Automations (RPAs).
- **medium** – Recommended for up to 100 RPAs.
- **large** – Recommended for up to 200 RPAs.

# Set up Partner AI Apps

The following topics describe the permissions needed to start using Amazon SageMaker Partner AI Apps. The permissions required are split into two parts, depending on the user permissions level:

- **Administrative permissions** – Permissions for administrators setting up data scientist and machine learning (ML) developer environments.
  - AWS Marketplace
  - Partner AI Apps management
  - AWS License Manager
- **User permissions** – Permissions for data scientists and machine learning developers.
  - User authorization
  - Identity propagation
  - SDK access

## Prerequisites

Admins can complete the following prerequisites to set up Partner AI Apps.

- (Optional) Onboard to a SageMaker AI domain. Partner AI Apps can be accessed directly from a SageMaker AI domain. For more information, see [Amazon SageMaker AI domain overview](#).
  - If using Partner AI Apps in a SageMaker AI domain in VPC-only mode, admins must create an endpoint with the following format to connect to the Partner AI Apps. For more information about using Studio in VPC-only mode, see [Connect Amazon SageMaker Studio in a VPC to External Resources](#).

```
aws.sagemaker.<region>.partner-app
```

- (Optional) If admins are interacting with the domain using the AWS CLI, they must also complete the following prerequisites.
  1. Update the AWS CLI by following the steps in [Installing the current AWS CLI Version](#).
  2. From the local machine, run `aws configure` and provide AWS credentials. For information about AWS credentials, see [Understanding and getting your AWS credentials](#).

## Administrative permissions

The administrator must add the following permissions to enable Partner AI Apps in SageMaker AI.

- Permission to complete AWS Marketplace subscription for Partner AI Apps
- Set up Partner AI App execution role

## AWS Marketplace subscription for Partner AI Apps

Admins must complete the following steps to add permissions for AWS Marketplace. For information about using AWS Marketplace, see [Getting started as a buyer using AWS Marketplace](#).

1. Grant permissions for AWS Marketplace. Partner AI Apps administrators require these permissions to purchase subscriptions to Partner AI Apps from AWS Marketplace. To get access to AWS Marketplace, admins must attach the `AWSMarketplaceManageSubscriptions` managed policy to the IAM role that they're using to access the SageMaker AI console and purchase the app. For details about the `AWSMarketplaceManageSubscriptions` managed policy, see [AWS managed policies for AWS Marketplace buyers](#). For information about attaching managed policies, see [Adding and removing IAM identity permissions](#).
  2. Grant permissions for SageMaker AI to run operations on the admins behalf using other AWS services. Admins must grant SageMaker AI permissions to use these services and the resources that they act upon. The following policy definition demonstrates how to grant the required Partner AI Apps permissions. These permissions are needed in addition to the existing permissions for the admin role. For more information, see [How to use SageMaker AI execution roles](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sagemaker>CreatePartnerApp",
        "sagemaker>DeletePartnerApp",
        "sagemaker>UpdatePartnerApp",
        "sagemaker>DescribePartnerApp",
        "sagemaker>ListPartnerApps",
        "sagemaker>CreatePartnerAppPresignedUrl",
        "sagemaker>CreatePartnerApp",
        "sagemaker>AddTags",
        "sagemaker>ListTags",
        "sagemaker>DeleteTags"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sagemaker>CreateEndpointConfig",
        "sagemaker>DeleteEndpointConfig",
        "sagemaker>UpdateEndpointConfig",
        "sagemaker>DescribeEndpointConfig",
        "sagemaker>ListEndpointConfigs",
        "sagemaker>CreateEndpoint",
        "sagemaker>DeleteEndpoint",
        "sagemaker>UpdateEndpoint",
        "sagemaker>DescribeEndpoint",
        "sagemaker>ListEndpoints"
      ],
      "Resource": "*"
    }
  ]
}
```

```
        "iam:PassRole"
    ],
    "Resource": "arn:aws:iam::*:role/*",
    "Condition": {
        "StringEquals": {
            "iam:PassedToService": "sagemaker.amazonaws.com"
        }
    }
}
]
```

## Set up Partner AI App execution role

1. Partner AI Apps require an execution role to interact with resources in the AWS account. Admins can create this execution role using the AWS CLI. The Partner AI App uses this role to complete actions related to Partner AI App functionality.

```
aws iam create-role --role-name PartnerAiAppExecutionRole --assume-role-policy-document '{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "sagemaker.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}'
```

2. Create the AWS License Manager service-linked role by following the steps in [Create a service-linked role for License Manager](#).
3. Grant permissions for the Partner AI App to access License Manager using the AWS CLI. These permissions are required to access the licenses for Partner AI App. This allows the Partner AI App to verify access to the Partner AI App license.

```
aws iam put-role-policy --role-name PartnerAiAppExecutionRole --policy-name LicenseManagerPolicy --policy-document '{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": [  
            "license-manager:CheckoutLicense",  
            "license-manager:CheckInLicense",  
            "license-manager:ExtendLicenseConsumption",  
            "license-manager:GetLicense",  
            "license-manager:GetLicenseUsage"  
        ],  
        "Resource": "*"  
    }  
}'
```

4. If the Partner AI App requires access to an Amazon S3 bucket, then add Amazon S3 permissions to the execution role. For more information, see [Required permissions for Amazon S3 API operations](#).

## User permissions

After admins have completed the administrative permissions settings, they must make sure that users have the permissions needed to access the Partner AI Apps.

1. Grant permissions for SageMaker AI to run operations on your behalf using other AWS services. Admins must grant SageMaker AI permissions to use these services and the resources that they act upon. Admins grant SageMaker AI these permissions using an IAM execution role. For more information about IAM roles, see [IAM roles](#). The following policy definition demonstrates how to grant the required Partner AI Apps permissions. This policy can be added to the execution role of the user profile. For more information, see [How to use SageMaker AI execution roles](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "sagemaker:DescribePartnerApp",
```

```
        "sagemaker>ListPartnerApps",
        "sagemaker>CreatePartnerAppPresignedUrl"
    ],
    "Resource": "arn:aws:sagemaker:*:*:partner-app/app-*"
}
]
```

2. (Optional) If launching Partner AI Apps from Studio, add the `sts:TagSession` trust policy to the role used to launch Studio or the Partner AI Apps directly as follows. This makes sure that the identity can be propagated properly.

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "sagemaker.amazonaws.com"
    },
    "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
    ]
}
```

3. (Optional) If using the SDK of a Partner AI App to access functionality in SageMaker AI, add the following `CallPartnerAppApi` permission to the role used to run the SDK code. If running the SDK code from Studio, add the permission to the Studio execution role. If running the code from anywhere other than Studio, add the permission to the IAM role used with the notebook. This gives the user access the Partner AI App functionality from the Partner AI App's SDK.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Action": [
                "sagemaker:CallPartnerAppApi"
            ],
            "Resource": [
                "arn:aws:sagemaker:region:account:partner-app/app"
            ]
        }
    ]
}
```

```
        }  
    ]  
}
```

## Manage user authorization and authentication

To provide access to Partner AI Apps to members of their team, admins must make sure that the identity of their users is propagated to the Partner AI Apps. This propagation makes sure users can properly access the Partner AI Apps' UI and perform authorized Partner AI App actions.

Partner AI Apps support the following identity sources:

- AWS IAM Identity Center
- External identity providers (IdPs)
- IAM Session-based identity

The following sections gives information about the identity sources that Partner AI Apps support, as well as important details related to that identity source.

### IAM Identity Center

If a user is authenticated into Studio using IAM Identity Center and launches an application from Studio, the IAM Identity Center UserName is automatically propagated as the user identity for a Partner AI App. This is not the case if the user launches the Partner AI App directly using the `CreatePartnerAppPresignedUrl` API.

### External identity providers (IdPs)

If using SAML for AWS account federation, admins have two options to carry over the IdP identity as the user identity for a Partner AI App. For information about setting up AWS account federation, see [How to Configure SAML 2.0 for AWS account Federation](#).

- **Principal Tag** – Admins can configure the IdP-specific IAM Identity Center application to pass identity information from the landing session using the AWS session `PrincipalTag` with the following `Name` attribute. When using SAML, the landing role session uses an IAM role. To use the `PrincipalTag`, admins must add the `sts:TagSession` permission to this landing role, as well as the Studio execution role. For more information about `PrincipalTag`, see [Configure SAML assertions for the authentication response](#).

<https://aws.amazon.com/SAML/Attributes/PrincipalTag:SageMakerPartnerAppUser>

- **Landing session name** – Admins can propagate the landing session name as the identity for the Partner AI App. To do this, they must set the `EnableIamSessionBasedIdentity` opt-in flag for each Partner AI App. For more information, see [EnableIamSessionBasedIdentity](#).

## IAM session-based identity

### Important

We do not recommend using this method for production accounts. For production accounts, use an identity provider for increased security.

SageMaker AI supports the following options for identity propagation when using an IAM session-based identity. All of the options, except using a session tag with AWS STS, require setting the `EnableIamSessionBasedIdentity` opt-in flag for each application. For more information, see [EnableIamSessionBasedIdentity](#).

When propagating identities, SageMaker AI verifies whether an AWS STS Session tag is being used. If one is not used, then SageMaker AI propagates the IAM username or AWS STS session name.

- **AWS STS Session tag** – Admins can set a `SageMakerPartnerAppUser` session tag for the launcher IAM session. When admins launch a Partner AI App using the SageMaker AI console or the AWS CLI, the `SageMakerPartnerAppUser` session tag is automatically passed as the user identity for the Partner AI App. The following example shows how to set the `SageMakerPartnerAppUser` session tag using the AWS CLI. The value of the key is added as a principal tag.

```
aws sts assume-role \
--role-arn arn:aws:iam::account:role/iam-role-used-to-launch-partner-ai-app \
--role-session-name session_name \
--tags Key=SageMakerPartnerAppUser,Value=user-name
```

When giving users access to a Partner AI App using `CreatePartnerAppPresignedUrl`, we recommend verifying the value for the `SageMakerPartnerAppUser` key. This helps to prevent unintended access to Partner AI App resources. The following trust policy verifies that the session

tag exactly matches the associated IAM user. Admins can use any principal tag for this purpose. It should be configured on the role that is launching Studio or the Partner AI App.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "RoleTrustPolicyRequireUsernameForSessionName",  
            "Effect": "Allow",  
            "Action": [  
                "sts:AssumeRole",  
                "sts:TagSession"  
            ],  
            "Principal": {  
                "AWS": "arn:aws:iam::account:root"  
            },  
            "Condition": {  
                "StringLike": {  
                    "aws:RequestTag/SageMakerPartnerAppUser": "${aws:username}"  
                }  
            }  
        }  
    ]  
}
```

- **Authenticated IAM user** – The username of the user is automatically propagated as the Partner AI App user.
- **AWS STS session name** – If no SageMakerPartnerAppUser session tag is configured when using AWS STS, SageMaker AI returns an error when users launch a Partner AI App. To avoid this error, admins must set the EnableIamSessionBasedIdentity opt-in flag for each Partner AI App. For more information, see [EnableIamSessionBasedIdentity](#).

When the EnableIamSessionBasedIdentity opt-in flag is enabled, use the [IAM role trust policy](#) to make sure that the IAM session name is or contains the IAM username. This makes sure that users don't gain access by impersonating other users. The following trust policy verifies that the session name exactly matches the associated IAM user. Admins can use any principal tag for this purpose. It should be configured on the role that is launching Studio or the Partner AI App.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {
```

```
{  
    "Sid": "RoleTrustPolicyRequireUsernameForSessionName",  
    "Effect": "Allow",  
    "Action": "sts:AssumeRole",  
    "Principal": {  
        "AWS": "arn:aws:iam::account:root"  
    },  
    "Condition": {  
        "StringEquals": {  
            "sts:RoleSessionName": "${aws:username}"  
        }  
    }  
}  
]  
}
```

Admins must also add the `sts:TagSession` trust policy to the role that is launching Studio or the Partner AI App. This makes sure that the identity can be propagated properly.

```
{  
    "Effect": "Allow",  
    "Principal": {  
        "Service": "sagemaker.amazonaws.com"  
    },  
    "Action": [  
        "sts:AssumeRole",  
        "sts:TagSession"  
    ]  
}
```

After setting the credentials, admins can give their users access to Studio or the Partner AI App from the AWS CLI using either the `CreatePresignedDomainUrl` or `CreatePartnerAppPresignedUrl` API calls, respectively.

Users can also then launch Studio from the SageMaker AI console, and launch Partner AI Apps from Studio.

## EnableIamSessionBasedIdentity

`EnableIamSessionBasedIdentity` is an opt-in flag. When the `EnableIamSessionBasedIdentity` flag is set, SageMaker AI passes IAM session information

as the Partner AI App user identity. For more information about AWS STS sessions, see [Use temporary credentials with AWS resources](#).

## Access control

To control access to Partner AI Apps, use an IAM policy attached to the user profile's execution role. To launch a Partner AI App directly from Studio or using the AWS CLI, the user profile's execution role must have a policy that gives permissions for the `CreatePartnerAppPresignedUrl` API. Remove this permission from the user profile's execution role to make sure they can't launch Partner AI Apps.

## Root admin users

The Comet and Fiddler Partner AI Apps require at least one root admin user. Root admin users have permissions to add both normal and admin users and manage resources. The usernames provided as root admin users must be consistent with the usernames from the identity source.

While root admin users are persisted in SageMaker AI, normal admin users are not and exist only within the Partner AI App until the Partner AI App is terminated.

Admins can update root admin users using the `UpdatePartnerApp` API call. When root admin users are updated, the updated list of root admin users is passed to the Partner AI App. The Partner AI App makes sure that all usernames in the list are granted root admin privileges. If a root admin user is removed from the list, the user still retains normal admin permissions until either:

- The user is removed from the application.
- Another admin user revokes admin permissions for the user.

### Note

Fiddler doesn't support updating admin users. Only Comet supports updates to root admin users.

To delete a root admin user, you must first update the list of root admin users using the `UpdatePartnerApp` API. Then, remove or revoke the admin permissions through the Partner AI App's UI.

If you remove a root admin user from the Partner AI App's UI without updating the list of root admin users with the `UpdatePartnerApp` API, the change is temporary. When SageMaker AI

sends the next Partner AI App update request, SageMaker AI sends the root admin list that still includes the user to the Partner AI App. This overrides the deletion completed from the Partner AI App UI.

## Partner AI App provisioning

After admins have set up the required permissions, they can explore and provision Amazon SageMaker Partner AI Apps for users in the domain.

Admins can view all of the available Partner AI Apps, as well as the Partner AI Apps that they have provisioned from the [Amazon SageMaker AI console](#). From the **Partner AI Apps** page, admins can view details about the pricing model for each Partner AI App and make them available to users. Admins can make them available by navigating to the AWS Marketplace to subscribe to that Partner AI App.

Admins can provision new apps from the Partner AI Apps page. They can also view the Partner AI Apps that they have already provisioned from the **My Apps** tab.

### Note

Applications that admins provision can be accessed by all users that admins give proper permissions to in an AWS account. Partner AI Apps are not restricted to a specific domain or user.

## Status

When admins view a Partner AI App that they have provisioned, they can also see the status of their application with one of the following values.

- **Deployed** – The application is ready for use. Admins can update the application configuration and delete the application.
- **Error** – There was an issue with the application deployment. Admins can troubleshoot and configure the application again to deploy it.
- **Not deployed** – The application has been subscribed to, but not deployed. Admins can configure the application to deploy it.

## Options

When admins configure an application, they can decide the following options:

- **App name** – A unique name for the application.
- **App maintenance schedule** – Partner AI Apps undergo maintenance on a weekly basis. With this option, admins choose both the day of the week and the time that this maintenance happens.
- **STS identity propagation** – Use this option to pass the AWS Security Token Service (AWS STS) launcher IAM session name as the Partner AI App user identity. For more information, see [Set up Partner AI Apps](#).
- **Admin management** – Some Partner AI Apps support adding up to five admins that have full rights to manage the Partner AI App functionality. This only applies to Comet and Fiddler. For more information, see [Set up Partner AI Apps](#).
- **Execution role** – The role that the Partner AI App uses to access resources and perform actions. For more information, see [Set up Partner AI Apps](#).
- **App version** – The version of the Partner AI App that admins want to use.
- **Tier selection** – The infrastructure deployment tier for the Partner AI App. The tier size impacts the speed and capabilities of the application. For more information, see [Set up Partner AI Apps](#).
- **Lakera S3 bucket policy** – This is only required by the Lakera-guard app to access an Amazon S3 bucket.

## Set up the Amazon SageMaker Partner AI Apps SDKs

The following topic outlines the process needed to install and use the application-specific SDKs with Amazon SageMaker Partner AI Apps. To install and use SDKs for applications, you must specify environment variables specific to Partner AI Apps, so the application's SDK can pick up environment variables and trigger authorization. The following sections give information about the steps needed to complete this for each of the supported application types.

### Comet

Comet offers two products:

- Opik is an source LLM evaluation framework.
- Comet's ML platform can be used to track, compare, explain, and optimize models across the complete ML lifecycle.

Comet supports the use of two different SDKs based on the product that you are interacting with. Complete the following procedure to install and use the Comet or Opik SDKs. For more information about the Comet SDK, see [Quickstart](#). For more information about the Opik SDK, see [Open source LLM evaluation framework](#).

1. Launch the environment that you are using the Comet or Opik SDKs with Partner AI Apps in. For information about launching a JupyterLab application, see [Create a space](#). For information about launching a Code Editor, based on Code-OSS, Visual Studio Code - Open Source application, see [Launch a Code Editor application in Studio](#).
2. Launch a Jupyter notebook or Code Editor space.
3. From the development environment, install the compatible Comet, Opik, and SageMaker Python SDK versions. To be compatible:
  - The SageMaker Python SDK version must be at least 2.237.0.
  - The Comet SDK version must be the latest version.
  - The Opik SDK version must match the version used by your Opik application. Verify the Opik version used in the Opik web application UI. The exception to this is that the Opik SDK version must be at least 1.2.0 when the Opik application version is 1.1.5.

 **Note**

SageMaker JupyterLab comes with SageMaker Python SDK installed. However, you may need to upgrade the SageMaker Python SDK if the version is lower than 2.237.0.

```
%pip install sagemaker>=2.237.0 comet_ml  
##or  
%pip install sagemaker>=2.237.0 opik=<compatible-version>
```

4. Set the following environment variables for the application resource ARN. These environment variables are used to communicate with the Comet and Opik SDKs. To retrieve these values, navigate to the details page for the application in Amazon SageMaker Studio.

```
os.environ['AWS_PARTNER_APP_AUTH'] = 'true'  
os.environ['AWS_PARTNER_APP_ARN'] = '<partner-app-ARN>'
```

- For the Comet application, the SDK URL is automatically included as part of the API key set in the following step. You may instead set the COMET\_URL\_OVERRIDE environment variable to manually override the SDK URL.

```
os.environ['COMET_URL_OVERRIDE'] = '<comet-url>'
```

- For the Opik application, the SDK URL is automatically included as part of the API key set in the following step. You may instead set the OPIK\_URL\_OVERRIDE environment variable to manually override the SDK URL. To get the Opik workspace name, see the Opik application and navigate to the user's workspace.

```
os.environ['OPIK_URL_OVERRIDE'] = '<opik-url>'  
os.environ['OPIK_WORKSPACE'] = '<workspace-name>'
```

- Set the environment variable that identifies the API key for Comet or Opik. This is used to verify the connection from SageMaker to the application when the Comet and Opik SDKs are used. This API key is application-specific and is not managed by SageMaker. To get this key, you must log into the application and retrieve the API key. The Opik API key is the same as the Comet API key.

```
os.environ['COMET_API_KEY'] = '<API-key>'  
os.environ["OPIK_API_KEY"] = os.environ["COMET_API_KEY"]
```

## Fiddler

Complete the following procedure to install and use the Fiddler Python Client. For information about the Fiddler Python Client, see [About Client 3.x](#).

- Launch the notebook environment that you are using the Fiddler Python Client with Partner AI Apps in. For information about launching a JupyterLab application, see [Create a space](#). For information about launching a Code Editor, based on Code-OSS, Visual Studio Code - Open Source application, see [Launch a Code Editor application in Studio](#).
- Launch a Jupyter notebook or Code Editor space.
- From the development environment, install the Fiddler Python Client and SageMaker Python SDK versions. To be compatible:
  - The SageMaker Python SDK version must be at least 2.237.0.

- The Fiddler Python Client version must be compatible with the version of Fiddler used in the application. After verifying the Fiddler version from the UI, see the Fiddler [Compatibility Matrix](#) for the compatible Fiddler Python Client version.

 **Note**

SageMaker JupyterLab comes with SageMaker Python SDK installed. However, you may need to upgrade the SageMaker Python SDK if the version is lower than 2.237.0.

```
%pip install sagemaker>=2.237.0 fiddler-client=<compatible-version>
```

- Set the following environment variables for the application resource ARN and the SDK URL. These environment variables are used to communicate with the Fiddler Python Client. To retrieve these values, navigate to the details page for the Fiddler application in Amazon SageMaker Studio.

```
os.environ['AWS_PARTNER_APP_AUTH'] = 'true'  
os.environ['AWS_PARTNER_APP_ARN'] = '<partner-app-ARN>'  
os.environ['AWS_PARTNER_APP_URL'] = '<partner-app-URL>'
```

- Set the environment variable that identifies the API key for the Fiddler application. This is used to verify the connection from SageMaker to the Fiddler application when the Fiddler Python Client is used. This API key is application-specific and is not managed by SageMaker. To get this key, you must log into the Fiddler application and retrieve the API key.

```
os.environ['FIDDLER_KEY'] = '<API-key>'
```

## Deepchecks

Complete the following procedure to install and use Deepchecks Python SDK.

- Launch the notebook environment that you are using the Deepchecks Python SDK with Partner AI Apps in. For information about launching a JupyterLab application, see [Create a space](#). For information about launching a Code Editor, based on Code-OSS, Visual Studio Code - Open Source application, see [Launch a Code Editor application in Studio](#).
- Launch a Jupyter notebook or Code Editor space.

- From the development environment, install the compatible Deepchecks Python SDK and SageMaker Python SDK versions. Partner AI Apps is running version `0.21.15` of Deepchecks. To be compatible:

- The SageMaker Python SDK version must be at least `2.237.0`.
- The Deepchecks Python SDK must use the minor version `0.21`.

 **Note**

SageMaker JupyterLab comes with SageMaker Python SDK installed. However, you may need to upgrade the SageMaker Python SDK if the version is lower than `2.237.0`.

```
%pip install sagemaker>=2.237.0 deepchecks-llm-client>=0.21,<0.22
```

- Set the following environment variables for the application resource ARN and the SDK URL. These environment variables are used to communicate with the Deepchecks Python SDK. To retrieve these values, navigate to the details page for the application in Amazon SageMaker Studio.

```
os.environ['AWS_PARTNER_APP_AUTH'] = 'true'  
os.environ['AWS_PARTNER_APP_ARN'] = '<partner-app-ARN>'  
os.environ['AWS_PARTNER_APP_URL'] = '<partner-app-URL>'
```

- Set the environment variable that identifies the API key for the Deepchecks application. This is used to verify the connection from SageMaker to the Deepchecks application when the Deepchecks Python SDK is used. This API key is application-specific and is not managed by SageMaker. To get this key, see [Setup: Python SDK Installation & API Key Retrieval](#).

```
os.environ['DEEPCHECKS_API_KEY'] = '<API-key>'
```

## Lakera

Lakera does not offer an SDK. Instead, you can interact with the Lakera Guard API through HTTP requests to the available endpoints in any programming language. For more information, see [Lakera Guard API](#).

To use the SageMaker Python SDK with Lakera, complete the following steps:

1. Launch the environment that you are using Partner AI Apps in. For information about launching a JupyterLab application, see [Create a space](#). For information about launching a Code Editor, based on Code-OSS, Visual Studio Code - Open Source application, see [Launch a Code Editor application in Studio](#).
2. Launch a Jupyter notebook or Code Editor space.
3. From the development environment, install the compatible SageMaker Python SDK version. The SageMaker Python SDK version must be at least 2.237.0

 **Note**

SageMaker JupyterLab comes with SageMaker Python SDK installed. However, you may need to upgrade the SageMaker Python SDK if the version is lower than 2.237.0.

```
%pip install sagemaker>=2.237.0
```

4. Set the following environment variables for the application resource ARN and the SDK URL. To retrieve these values, navigate to the details page for the application in Amazon SageMaker Studio.

```
os.environ['AWS_PARTNER_APP_ARN'] = '<partner-app-ARN>'  
os.environ['AWS_PARTNER_APP_URL'] = '<partner-app-URL>'
```

## Partner AI Apps in Studio

After the admin has added the required permissions and authorized users, users can view the Amazon SageMaker Partner AI App in Amazon SageMaker Studio. From Studio, users can launch apps that have been approved for use by their administrator.

### Browsing and selecting

To browse the available Partner AI Apps, users must navigate to Studio. For information about launching Studio, see [Launch Amazon SageMaker Studio](#).

After users have launched Studio, they can view all of the available Partner AI Apps by selecting the **Partner AI Apps** section in the left navigation. The **Partner AI Apps** page lists all of the Partner

AI Apps, and gives information about whether the Partner AI Apps have been deployed by the admin. If the desired Partner AI Apps haven't been deployed, users can reach out to the admin to request that they deploy the Partner AI Apps for use in the SageMaker AI domain.

If the application has been deployed, users can open the Partner AI App UI to start using it or view details of the Partner AI App.

When users view the details of the application, they see the value of the following.

- ARN – This is the resource ARN of the Partner AI App.
- SDK URL – This is the URL of the Partner AI App that the Partner AI App SDK uses to support app-specific tasks such as logging model experiment tracking data from a JupyterLab notebook in Studio.

Users can use these values to write code that uses the Partner AI App SDK for app-specific tasks.

Each Partner AI App's details page includes a sample notebook. To get started, users can launch the sample notebook in a JupyterLab space in the Studio environment.

# Data labeling with a human-in-the-loop

To train a machine learning model, you need a large, high-quality, labeled dataset. You can label your data using Amazon SageMaker Ground Truth. Choose from one of the Ground Truth [built-in task types](#) or create your own [custom labeling workflow](#). To improve the accuracy of your data labels and reduce the total cost of labeling your data, use Ground Truth enhanced data labeling features like [automated data labeling](#) and [annotation consolidation](#).

## Topics

- [Training data labeling using humans with Amazon SageMaker Ground Truth](#)
- [Use Amazon SageMaker Ground Truth Plus to Label Data](#)
- [Workforces](#)
- [Crowd HTML Elements Reference](#)
- [Using Amazon Augmented AI for Human Review](#)

## Training data labeling using humans with Amazon SageMaker Ground Truth

To train a machine learning model, you need a large, high-quality, labeled dataset. Ground Truth helps you build high-quality training datasets for your machine learning models. With Ground Truth, you can use workers from either Amazon Mechanical Turk, a vendor company that you choose, or an internal, private workforce along with machine learning to enable you to create a labeled dataset. You can use the labeled dataset output from Ground Truth to train your own models. You can also use the output as a training dataset for an Amazon SageMaker AI model.

Depending on your ML application, you can choose from one of the Ground Truth built-in task types to have workers generate specific types of labels for your data. You can also build a custom labeling workflow to provide your own UI and tools to workers labeling your data. To learn more about the Ground Truth built in task types, see [Built-in Task Types](#). To learn how to create a custom labeling workflow, see [Custom labeling workflows](#).

In order to automate labeling your training dataset, you can optionally use *automated data labeling*, a Ground Truth process that uses machine learning to decide which data needs to be labeled by humans. Automated data labeling may reduce the labeling time and manual effort

required. For more information, see [Automate data labeling](#). To create a custom labeling workflow, see [Custom labeling workflows](#).

Use either pre-built or custom tools to assign the labeling tasks for your training dataset. A *labeling UI template* is a webpage that Ground Truth uses to present tasks and instructions to your workers. The SageMaker AI console provides built-in templates for labeling data. You can use these templates to get started , or you can build your own tasks and instructions by using our HTML 2.0 components. For more information, see [Custom labeling workflows](#).

Use the workforce of your choice to label your dataset. You can choose your workforce from:

- The Amazon Mechanical Turk workforce of over 500,000 independent contractors worldwide.
- A private workforce that you create from your employees or contractors for handling data within your organization.
- A vendor company that you can find in the AWS Marketplace that specializes in data labeling services.

For more information, see [Workforces](#).

You store your datasets in Amazon S3 buckets. The buckets contain three things: The data to be labeled, an input manifest file that Ground Truth uses to read the data files, and an output manifest file. The output file contains the results of the labeling job. For more information, see [Use input and output data](#).

Events from your labeling jobs appear in Amazon CloudWatch under the /aws/sagemaker/ LabelingJobs group. CloudWatch uses the labeling job name as the name for the log stream.

## Are You a First-time User of Ground Truth?

If you are a first-time user of Ground Truth, we recommend that you do the following:

1. **Read [Getting started: Create a bounding box labeling job with Ground Truth](#)**—This section walks you through setting up your first Ground Truth labeling job.
2. **Explore other topics**—Depending on your needs, do the following:
  - **Explore built-in task types**— Use built-in task types to streamline the process of creating a labeling job. See [Built-in Task Types](#) to learn more about Ground Truth built-in task types.
  - **Manage your labeling workforce**—Create new work teams and manage your existing workforce. For more information, see [Workforces](#).

- **Learn about streaming labeling jobs**— Create a streaming labeling job and send new dataset objects to workers in real time using a perpetually running labeling job. Workers continuously receive new data objects to label as long as the labeling job is active and new objects are being sent to it. To learn more, see [Ground Truth streaming labeling jobs](#).
3. To learn more about available operations to automate Ground Truth operations, see the [SageMaker AI service API reference](#).

## Getting started: Create a bounding box labeling job with Ground Truth

To get started using Amazon SageMaker Ground Truth, follow the instructions in the following sections. The sections here explain how to use the console to create a bounding box labeling job, assign a public or private workforce, and send the labeling job to your workforce. You can also learn how to monitor the progress of a labeling job.

This video shows you how to setup and use Amazon SageMaker Ground Truth. (Length: 9:37)

If you want to create a custom labeling workflow, see [Custom labeling workflows](#) for instructions.

Before you create a labeling job, you must upload your dataset to an Amazon S3 bucket. For more information, see [Use input and output data](#).

### Topics

- [Before You Begin](#)
- [Create a Labeling Job](#)
- [Select Workers](#)
- [Configure the Bounding Box Tool](#)
- [Monitoring Your Labeling Job](#)

### Before You Begin

Before you begin using the SageMaker AI console to create a labeling job, you must set up the dataset for use. Do this:

1. Save two images at publicly available HTTP URLs. The images are used when creating instructions for completing a labeling task. The images should have an aspect ratio of around 2:1. For this exercise, the content of the images is not important.

2. Create an Amazon S3 bucket to hold the input and output files. The bucket must be in the same Region where you are running Ground Truth. Make a note of the bucket name because you use it during step 2.

Ground Truth requires all S3 buckets that contain labeling job input image data have a CORS policy attached. To learn more about this change, see [CORS Requirement for Input Image Data](#).

3. You can create an IAM role or let SageMaker AI create a role with the [AmazonSageMakerFullAccess](#) IAM policy. Refer to [Creating IAM roles](#) and assign the following permissions policy to the user that is creating the labeling job:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "sagemakergroundtruth",  
            "Effect": "Allow",  
            "Action": [  
                "cognito-idp:CreateGroup",  
                "cognito-idp:CreateUserPool",  
                "cognito-idp:CreateUserPoolDomain",  
                "cognito-idp:AdminCreateUser",  
                "cognito-idp:CreateUserPoolClient",  
                "cognito-idp:AdminAddUserToGroup",  
                "cognito-idp:DescribeUserPoolClient",  
                "cognito-idp:DescribeUserPool",  
                "cognito-idp:UpdateUserPool"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

## Create a Labeling Job

In this step you use the console to create a labeling job. You tell Amazon SageMaker Ground Truth the Amazon S3 bucket where the manifest file is stored and configure the parameters for the job. For more information about storing data in an Amazon S3 bucket, see [Use input and output data](#).

## To create a labeling job

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/>.
2. From the left navigation, choose **Labeling jobs**.
3. Choose **Create labeling job** to start the job creation process.
4. In the **Job overview** section, provide the following information:
  - **Job name** – Give the labeling job a name that describes the job. This name is shown in your job list. The name must be unique in your account in an AWS Region.
  - **Label attribute name** – Leave this unchecked as the default value is the best option for this introductory job.
  - **Input data setup** – Select **Automated data setup**. This option allows you to automatically connect to your input data in S3.
  - **S3 location for input datasets** – Enter the S3 location where you added the images in step 1.
  - **S3 location for output datasets** – The location where your output data is written in S3.
  - **Data type** – Use the drop down menu to select **Image**. Ground Truth will use all images found in the S3 location for input datasets as input for your labeling job.
  - **IAM role** – Create or choose an IAM role with the AmazonSageMakerFullAccess IAM policy attached.
5. In the **Task type** section, for the **Task category** field, choose **Image**.
6. In the **Task selection** choose **Bounding box**.
7. Choose **Next** to move on to configuring your labeling job.

## Select Workers

In this step you choose a workforce for labeling your dataset. It is recommended that you create a private workforce to test Amazon SageMaker Ground Truth. Use email addresses to invite the members of your workforce. If you create a private workforce in this step you won't be able to import your Amazon Cognito user pool later. If you want to create a private workforce using an Amazon Cognito user pool, see [Manage a Private Workforce \(Amazon Cognito\)](#) and use the Mechanical Turk workforce instead in this tutorial.

**Tip**

To learn about the other workforce options you can use with Ground Truth, see [Workforces](#).

**To create a private workforce:**

1. In the **Workers** section, choose **Private**.
2. If this is your first time using a private workforce, in the **Email addresses** field, enter up to 100 email addresses. The addresses must be separated by a comma. You should include your own email address so that you are part of the workforce and can see data object labeling tasks.
3. In the **Organization name** field, enter the name of your organization. This information is used to customize the email sent to invite a person to your private workforce. You can change the organization name after the user pool is created through the console.
4. In the **Contact email** field enter an email address that members of the workforce use to report problems with the task.

If you add yourself to the private workforce, you will receive an email that looks similar to the following. **Amazon, Inc.** is replaced by the organization you enter in step 3 of the preceding procedure. Select the link in the email to log in using the temporary password provided. If prompted, change your password. When you successfully log in, you see the worker portal where your labeling tasks appear.

**[EXTERNAL] You're invited by Amazon, Inc. to work on a labeling project.****N**

To: [no-reply@verificationemail.com](mailto:no-reply@verificationemail.com) <[no-reply@verificationemail.com](mailto:no-reply@verificationemail.com)>

Thursday, February 11, 2021 at 10:34 AM

**CAUTION:** This email originated from outside of the organization. Do not click links or open attachments unless you can confirm the sender and know the content is safe.

**You're invited to work on a labeling project.**

You will need this user name and temporary password to log in the first time.

User name: [REDACTED]

Temporary password: [REDACTED]

Open the link below to log in:

[REDACTED]

After you log in with your temporary password, you are required to create a new one. If you have any questions, please contact [REDACTED].

**Tip**

You can find the link to your private workforce's worker portal in the **Labeling workforces** section of the Ground Truth area of the SageMaker AI console. To see the link, select the **Private** tab. The link is under the **Labeling portal sign-in URL** header in **Private workforce summary**.

If you choose to use the Amazon Mechanical Turk workforce to label the dataset, you are charged for labeling tasks completed on the dataset.

## To use the Amazon Mechanical Turk workforce:

1. In the **Workers** section, choose **Public**.
2. Set a **Price per task**.
3. If applicable, choose **The dataset does not contain adult content** to acknowledge that the sample dataset has no adult content. This information enables Amazon SageMaker Ground Truth to warn external workers on Mechanical Turk that they might encounter potentially offensive content in your dataset.
4. Choose the check box next to the following statement to acknowledge that the sample dataset does not contain any personally identifiable information (PII). This is a requirement to use Mechanical Turk with Ground Truth. If your input data does contain PII, use the private workforce for this tutorial.

**You understand and agree that the Amazon Mechanical Turk workforce consists of independent contractors located worldwide and that you should not share confidential information, personal information or protected health information with this workforce.**

## Configure the Bounding Box Tool

Finally you configure the bounding box tool to give instructions to your workers. You can configure a task title that describes the task and provides high-level instructions for the workers. You can provide both quick instructions and full instructions. Quick instructions are displayed next to the image to be labeled. Full instructions contain detailed instructions for completing the task. In this example, you only provide quick instructions. You can see an example of full instructions by choosing **Full instructions** at the bottom of the section.

### To configure the bounding box tool

1. In the **Task description** field type in brief instructions for the task. For example:

**Draw a box around any *objects* in the image.**

Replace *objects* with the name of an object that appears in your images.

2. In the **Labels** field, type a category name for the objects that the worker should draw a bounding box around. For example, if you are asking the worker to draw boxes around football players, you could use "Football Player" in this field.

3. The **Short instructions** section enables you to create instructions that are displayed on the page with the image that your workers are labeling. We suggest that you include an example of a correctly drawn bounding box and an example of an incorrectly drawn box. To create your own instructions, use these steps:
  - a. Select the text between **GOOD EXAMPLE** and the image placeholder. Replace it with the following text:

**Draw the box around the object with a small border.**
  - b. Select the first image placeholder and delete it.
  - c. Choose the image button and then enter the HTTPS URL of one of the images that you created in step 1. It is also possible to embed images directly in the short instructions section, however this section has a quota of 100 kilobytes (including text). If your images and text exceed 100 kilobytes, you receive an error.
  - d. Select the text between **BAD EXAMPLE** and the image placeholder. Replace it with the following text:

**Don't make the bounding box too large or cut into the object.**
  - e. Select the second image placeholder and delete it.
  - f. Choose the image button and then enter the HTTPS URL of the other image that you created in step 1.
4. Select **Preview** to preview the worker UI. The preview opens in a new tab, and so if your browser blocks pop ups you may need to manually enable the tab to open. When you add one or more annotations to the preview and then select **Submit** you can see a preview of the output data your annotation would created.
5. After you have configured and verified your instructions, select **Create** to create the labeling job.

If you used a private workforce, you can navigate to the worker portal that you logged into in [Select Workers](#) of this tutorial to see your labeling tasks. The tasks may take a few minutes to appear.

Now that you've created a labeling job, you can [monitor it, or stop it](#).

## Monitoring Your Labeling Job

After you create your labeling job, you see a list of all the jobs that you have created. You can use this list to monitor the status of your labeling jobs. The list has the following fields:

- **Name** – The name that you assigned the job when you created it.
- **Status** – The completion status of the job. The status can be one of Complete, Failed, In progress, or Stopped.
- **Labeled objects/total** – Shows the total number of objects in the labeling job and how many of them have been labeled.
- **Creation time** – The date and time that you created the job.

You can also clone, chain, or stop a job. Select a job and then select one of the following from the **Actions** menu:

- **Clone** – Creates a new labeling job with the configuration copied from the selected job. You can clone a job when you want to change to the job and run it again. For example, you can clone a job that was sent to a private workforce so that you can send it to the Amazon Mechanical Turk workforce. Or you can clone a job to rerun it against a new dataset stored in the same location as the original job.
- **Chain** – Creates a new labeling job that can build upon the data and models (if any) of a stopped, failed, or completed job. For more information about the use cases and how to use it, see [Chaining labeling jobs](#).
- **Stop** – Stops a running job. You cannot restart a stopped job. You can clone a job to start over or chain the job to continue from where it left off. Labels for any already labeled objects are written to the output file location. For more information, see [Labeling job output data](#).

## Label Images

Use Ground Truth to label images. Select one of the following built in task types to learn more about that task type. Each page includes instructions to help you create a labeling job using that task type.

### Tip

To learn more about supported file types and input data quotas, see [Input data](#).

## Topics

- [Classify image objects using a bounding box](#)
- [Identify image contents using semantic segmentation](#)
- [Auto-Segmentation Tool](#)
- [Create an image classification job \(Single Label\)](#)
- [Create an image classification job \(Multi-label\)](#)
- [Image Label Verification](#)

## Classify image objects using a bounding box

The images used to train a machine learning model often contain more than one object. To classify and localize one or more objects within images, use the Amazon SageMaker Ground Truth bounding box labeling job task type. In this context, localization means the pixel-location of the bounding box. You create a bounding box labeling job using the Ground Truth section of the Amazon SageMaker AI console or the [CreateLabelingJob](#) operation.

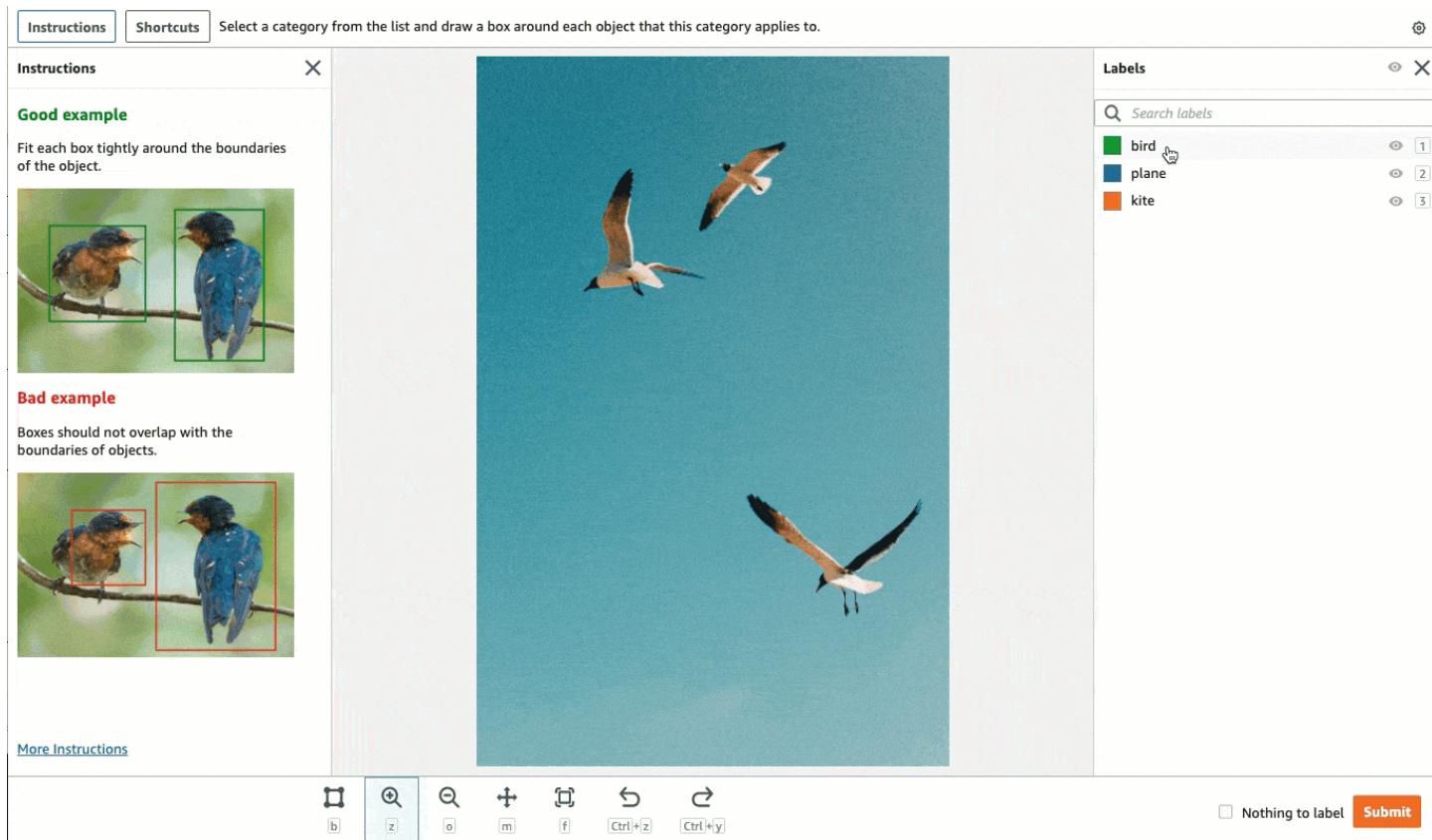
### Important

For this task type, if you create your own manifest file, use "source-ref" to identify the location of each image file in Amazon S3 that you want labeled. For more information, see [Input data](#).

### Creating a Bounding Box Labeling Job (Console)

You can follow the instructions [Create a Labeling Job \(Console\)](#) to learn how to create a bounding box labeling job in the SageMaker AI console. In Step 10, choose **Image** from the **Task category** drop down menu, and choose **Bounding box** as the task type.

Ground Truth provides a worker UI similar to the following for labeling tasks. When you create the labeling job with the console, you specify instructions to help workers complete the job and up to 50 labels that workers can choose from.



## Create a Bounding Box Labeling Job (API)

To create a bounding box labeling job, use the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

Follow the instructions on [Create a Labeling Job \(API\)](#) and do the following while you configure your request:

- Pre-annotation Lambda functions for this task type end with PRE-BoundingBox. To find the pre-annotation Lambda ARN for your Region, see [PreHumanTaskLambdaArn](#).
- Annotation-consolidation Lambda functions for this task type end with ACS-BoundingBox. To find the annotation-consolidation Lambda ARN for your Region, see [AnnotationConsolidationLambdaArn](#).

The following is an example of an [AWS Python SDK \(Boto3\) request](#) to create a labeling job in the US East (N. Virginia) Region. All parameters in red should be replaced with your specifications and resources.

```
response = client.create_labeling_job(
    LabelingJobName='example-bounding-box-labeling-job',
    LabelAttributeName='label',
    InputConfig={
        'DataSource': {
            'S3DataSource': {
                'ManifestS3Uri': 's3://bucket/path/manifest-with-input-data.json'
            }
        },
        'DataAttributes': {
            'ContentClassifiers': [
                'FreeOfPersonallyIdentifiableInformation' | 'FreeOfAdultContent',
            ]
        }
    },
    OutputConfig={
        'S3OutputPath': 's3://bucket/path/file-to-store-output-data',
        'KmsKeyId': 'string'
    },
    RoleArn='arn:aws:iam::*:role/*',
    LabelCategoryConfigS3Uri='s3://bucket/path/label-categories.json',
    StoppingConditions={
        'MaxHumanLabeledObjectCount': 123,
        'MaxPercentageOfInputDatasetLabeled': 123
    },
    HumanTaskConfig={
        'WorkteamArn': 'arn:aws:sagemaker:region::workteam/private-crowd/*',
        'UiConfig': {
            'UiTemplateS3Uri': 's3://bucket/path/worker-task-template.html'
        },
        'PreHumanTaskLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:PRE-
        BoundingBox',
        'TaskKeywords': [
            'Bounding Box',
        ],
        'TaskTitle': 'Bounding Box task',
        'TaskDescription': 'Draw bounding boxes around objects in an image',
        'NumberOfHumanWorkersPerDataObject': 123,
        'TaskTimeLimitInSeconds': 123,
        'TaskAvailabilityLifetimeInSeconds': 123,
        'MaxConcurrentTaskCount': 123,
        'AnnotationConsolidationConfig': {
```

```
'AnnotationConsolidationLambdaArn': 'arn:aws:lambda:us-  
east-1:432418664414:function:ACS-BoundingBox'  
    }  
},  
Tags=[  
{  
    'Key': 'string',  
    'Value': 'string'  
},  
]  
)
```

## Provide a Template for Bounding Box Labeling Jobs

If you create a labeling job using the API, you must supply a worker task template in `UiTemplateS3Uri`. Copy and modify the following template. Only modify the [short-instructions](#), [full-instructions](#), and header. Upload this template to S3, and provide the S3 URI for this file in `UiTemplateS3Uri`.

```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>  
<crowd-form>  
  <crowd-bounding-box  
    name="boundingBox"  
    src="{{ task.input.taskObject | grant_read_access }}"  
    header="please draw box"  
    labels="{{ task.input.labels | to_json | escape }}"  
  >  
  
  <full-instructions header="Bounding box instructions">  
    <ol><li><strong>Inspect</strong> the image</li><li><strong>Determine</strong> if the specified label is/are visible in the picture.</li>  
      <li><strong>Outline</strong> each instance of the specified label in the image using the provided "Box" tool.</li></ol>  
    <ul><li>Boxes should fit tight around each object</li>  
      <li>Do not include parts of the object are overlapping or that cannot be seen, even though you think you can interpolate the whole shape.</li>  
      <li>Avoid including shadows.</li>  
      <li>If the target is off screen, draw the box up to the edge of the image.</li>  
    </ul>  
  </full-instructions>  
  
  <short-instructions>  
    <h3><span style="color: #008000;">Good example</span></h3>
```

```
<p>Enter description of a correct bounding box label and add images</p>
<h3><span style="color: #0000ff; font-weight: bold;">Bad example</h3>
<p>Enter description of an incorrect bounding box label and add images</p>
</short-instructions>

</crowd-bounding-box>
</crowd-form>
```

## Bounding Box Output Data

Once you have created a bounding box labeling job, your output data will be located in the Amazon S3 bucket specified in the `S3OutputPath` parameter when using the API or in the **Output dataset location** field of the **Job overview** section of the console.

For example, the output manifest file of a successfully completed single-class bounding box task will contain the following:

```
[  
 {  
   "boundingBox": {  
     "boundingBoxes": [  
       {  
         "height": 2832,  
         "label": "bird",  
         "left": 681,  
         "top": 599,  
         "width": 1364  
       }  
     ],  
     "inputImageProperties": {  
       "height": 3726,  
       "width": 2662  
     }  
   }  
 }]
```

The `boundingBoxes` parameter identifies the location of the bounding box drawn around an object identified as a "bird" relative to the top-left corner of the image which is taken to be the (0,0) pixel-coordinate. In the previous example, `left` and `top` identify the location of the pixel in the top-left corner of the bounding box relative to the top-left corner of the

image. The dimensions of the bounding box are identified with **height** and **width**. The **inputImageProperties** parameter gives the pixel-dimensions of the original input image.

When you use the bounding box task type, you can create single- and multi-class bounding box labeling jobs. The output manifest file of a successfully completed multi-class bounding box will contain the following:

```
[  
  {  
    "boundingBox": {  
      "boundingBoxes": [  
        {  
          "height": 938,  
          "label": "squirrel",  
          "left": 316,  
          "top": 218,  
          "width": 785  
        },  
        {  
          "height": 825,  
          "label": "rabbit",  
          "left": 1930,  
          "top": 2265,  
          "width": 540  
        },  
        {  
          "height": 1174,  
          "label": "bird",  
          "left": 748,  
          "top": 2113,  
          "width": 927  
        },  
        {  
          "height": 893,  
          "label": "bird",  
          "left": 1333,  
          "top": 847,  
          "width": 736  
        }  
      ],  
      "inputImageProperties": {  
        "height": 3726,  
        "width": 2662  
      }  
    }  
]
```

```
    }  
}  
}  
]  
]
```

To learn more about the output manifest file that results from a bounding box labeling job, see [Bounding box job output](#).

To learn more about the output manifest file generated by Ground Truth and the file structure the Ground Truth uses to store your output data, see [Labeling job output data](#).

## Identify image contents using semantic segmentation

To identify the contents of an image at the pixel level, use an Amazon SageMaker Ground Truth semantic segmentation labeling task. When assigned a semantic segmentation labeling job, workers classify pixels in the image into a set of predefined labels or classes. Ground Truth supports single and multi-class semantic segmentation labeling jobs. You create a semantic segmentation labeling job using the Ground Truth section of the Amazon SageMaker AI console or the [CreateLabelingJob](#) operation.

Images that contain large numbers of objects that need to be segmented require more time. To help workers (from a private or vendor workforce) label these objects in less time and with greater accuracy, Ground Truth provides an AI-assisted auto-segmentation tool. For information, see [Auto-Segmentation Tool](#).

### Important

For this task type, if you create your own manifest file, use "source-ref" to identify the location of each image file in Amazon S3 that you want labeled. For more information, see [Input data](#).

## Creating a Semantic Segmentation Labeling Job (Console)

You can follow the instructions [Create a Labeling Job \(Console\)](#) to learn how to create a semantic segmentation labeling job in the SageMaker AI console. In Step 10, choose **Image** from the **Task category** drop down menu, and choose **Semantic segmentation** as the task type.

Ground Truth provides a worker UI similar to the following for labeling tasks. When you create the labeling job with the console, you specify instructions to help workers complete the job and labels that workers can choose from.

**Instructions** X

[View full instructions](#)  
[View tool guide](#)  
[How to use the Auto-segment tool](#)

**Good example**

All pixels in the image that are part of an animal have been colored with the appropriate label color.

**Bad example**

Some animals in the image have not been colored in completely.

The color for a given animal extends beyond the boundaries of the animal.

For each animal in the photo, select the appropriate label and fill in the animal with the appropriate color using the tools provided.



**Labels** X

Label	Count
squirrel	1
rabbit	2
bird	3

Nothing to label Submit

Auto-segment  Polygon  Brush  Eraser  Dimmer  Undo  Redo  Zoom in  Zoom out  Move  Fit image

## Create a Semantic Segmentation Labeling Job (API)

To create a semantic segmentation labeling job, use the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

Follow the instructions on [Create a Labeling Job \(API\)](#) and do the following while you configure your request:

- Pre-annotation Lambda functions for this task type end with PRE-SemanticSegmentation. To find the pre-annotation Lambda ARN for your Region, see [PreHumanTaskLambdaArn](#) .
- Annotation-consolidation Lambda functions for this task type end with ACS-SemanticSegmentation. To find the annotation-consolidation Lambda ARN for your Region, see [AnnotationConsolidationLambdaArn](#).

The following is an example of an [AWS Python SDK \(Boto3\) request](#) to create a labeling job in the US East (N. Virginia) Region. All parameters in red should be replaced with your specifications and resources.

```
response = client.create_labeling_job(
    LabelingJobName='example-semantic-segmentation-labeling-job',
    LabelAttributeName='label',
    InputConfig={
        'DataSource': {
            'S3DataSource': {
                'ManifestS3Uri': 's3://bucket/path/manifest-with-input-data.json'
            }
        },
        'DataAttributes': {
            'ContentClassifiers': [
                'FreeOfPersonallyIdentifiableInformation' | 'FreeOfAdultContent',
            ]
        }
    },
    OutputConfig={
        'S3OutputPath': 's3://bucket/path/file-to-store-output-data',
        'KmsKeyId': 'string'
    },
    RoleArn='arn:aws:iam::*:role/*',
    LabelCategoryConfigS3Uri='s3://bucket/path/label-categories.json',
    StoppingConditions={
        'MaxHumanLabeledObjectCount': 123,
        'MaxPercentageOfInputDatasetLabeled': 123
    },
    HumanTaskConfig={
        'WorkteamArn': 'arn:aws:sagemaker:region:*:workteam/private-crowd/*',
        'UiConfig': {
            'UiTemplateS3Uri': 's3://bucket/path/worker-task-template.html'
        },
        'PreHumanTaskLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:PRE-SemanticSegmentation',
        'TaskKeywords': [
            'Semantic Segmentation',
        ],
        'TaskTitle': 'Semantic segmentation task',
        'TaskDescription': 'For each category provided, segment out each relevant object using the color associated with that category',
        'NumberOfHumanWorkersPerDataObject': 123,
    }
)
```

```
'TaskTimeLimitInSeconds': 123,
'TaskAvailabilityLifetimeInSeconds': 123,
'MaxConcurrentTaskCount': 123,
'AnnotationConsolidationConfig': {
    'AnnotationConsolidationLambdaArn': 'arn:aws:lambda:us-
east-1:432418664414:function:ACS-SemanticSegmentation'
},
Tags=[  
    {  
        'Key': 'string',  
        'Value': 'string'  
    },
]  
)
```

## Provide a Template for Semantic Segmentation Labeling Jobs

If you create a labeling job using the API, you must supply a worker task template in `UiTemplateS3Uri`. Copy and modify the following template. Only modify the [short-instructions](#), [full-instructions](#), and header.

Upload this template to S3, and provide the S3 URI for this file in `UiTemplateS3Uri`.

```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>
<crowd-form>
<crowd-semantic-segmentation
    name="crowd-semantic-segmentation"
    src="{{ task.input.taskObject | grant_read_access }}"
    header="Please segment out all pedestrians."
    labels="{{ task.input.labels | to_json | escape }}"
>
<full-instructions header="Segmentation instructions">
    <ol><li><strong>Read</strong> the task carefully and inspect the image.</li>
    <li><strong>Read</strong> the options and review the examples provided to
understand more about the labels.</li>
    <li><strong>Choose</strong> the appropriate label that best suits an object and
paint that object using the tools provided.</li><ol>
</full-instructions>
<short-instructions>
    <h2><span style="color: rgb(0, 138, 0);">Good example</span></h2>
    <p>Enter description to explain a correctly done segmentation</p>
    <p><br><h2><span style="color: rgb(230, 0, 0);">Bad example</span></h2>
    <p>Enter description of an incorrectly done segmentation</p>
```

```
</short-instructions>
</crowd-semantic-segmentation>
</crowd-form>
```

## Semantic Segmentation Output Data

Once you have created a semantic segmentation labeling job, your output data will be located in the Amazon S3 bucket specified in the S3OutputPath parameter when using the API or in the **Output dataset location** field of the **Job overview** section of the console.

To learn more about the output manifest file generated by Ground Truth and the file structure the Ground Truth uses to store your output data, see [Labeling job output data](#).

To see an example of an output manifest file for a semantic segmentation labeling job, see [3D point cloud semantic segmentation output](#).

## Auto-Segmentation Tool

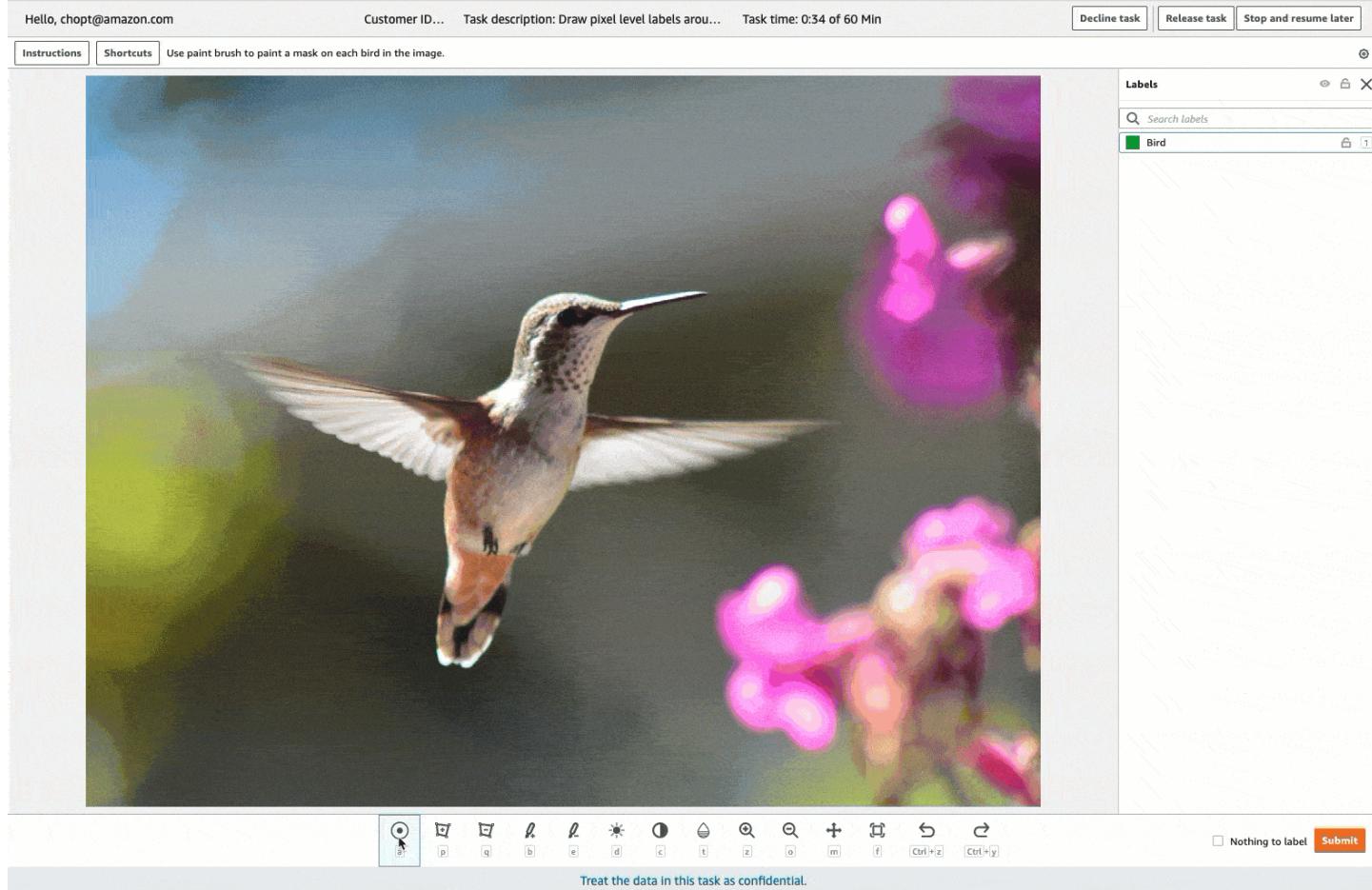
Image segmentation is the process of dividing an image into multiple segments, or sets of labeled pixels. In Amazon SageMaker Ground Truth, the process of identifying all pixels that fall under a given label involves applying a colored filler, or "mask", over those pixels. Some labeling job tasks contain images with a large numbers of objects that need to be segmented. To help workers label these objects in less time and with greater accuracy, Ground Truth provides an auto-segmentation tool for segmentation tasks assigned to private and vendor workforces. This tool uses a machine learning model to automatically segment individual objects in the image with minimal worker input. Workers can refine the mask generated by the auto-segmentation tool using other tools found in the worker console. This helps workers complete image segmentation tasks faster and more accurately, resulting in lower cost and higher label quality. The following page gives information about the tool and its availability.

### Note

The auto-segmentation tool is available for segmentation tasks that are sent to a private workforce or vendor workforce. It isn't available for tasks sent to the public workforce (Amazon Mechanical Turk).

## Tool Preview

When workers are assigned a labeling job that provides the auto-segmentation tool, they are provided with detailed instructions on how to use the tool. For example, a worker might see the following in the worker console:



Workers can use **View full instructions** to learn how to use the tool. Workers will need to place a point on four extreme-points (top-most, bottom-most, left-most, and right-most points) of the object of interest, and the tool will automatically generate a mask for the object. Workers can further-refine the mask using the other tools provided, or by using the auto-segment tool on smaller portions of the object that were missed.

## Tool Availability

The auto-segmentation tool automatically appears in your workers' consoles if you create a semantic segmentation labeling job using the Amazon SageMaker AI console. While creating a semantic segmentation job in the SageMaker AI console, you will be able to preview the tool while

creating worker instructions. To learn how to create a semantic segmentation labeling job in the SageMaker AI console, see [Getting started: Create a bounding box labeling job with Ground Truth](#).

If you are creating a custom instance segmentation labeling job in the SageMaker AI console or creating an instance- or semantic-segmentation labeling job using the Ground Truth API, you need to create a custom task template to design your worker console and instructions. To include the auto-segmentation tool in your worker console, ensure that the following conditions are met in your custom task template:

- For semantic segmentation labeling jobs created using the API, the <crowd-semantic-segmentation> is present in the task template. For custom instance segmentation labeling jobs, the <crowd-instance-segmentation> tag is present in the task template.
- The task is assigned to a private workforce or vendor workforce.
- The images to be labeled are Amazon Simple Storage Service Amazon S3) objects that have been pre-signed for the Worker so that they can access it. This is true if the task template includes the grant\_read\_access filter. For information about the grant\_read\_access filter, see [Adding automation with Liquid](#).

The following is an example of a custom task template for a custom instance segmentation labeling job, which includes the <crowd-instance-segmentation/> tag and the grant\_read\_access Liquid filter.

```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>
<crowd-form>
  <crowd-instance-segmentation
    name="crowd-instance-segmentation"
    src="{{ task.input.taskObject | grant_read_access }}"
    labels="['Car', 'Road']"
  <full-instructions header="Segmentation instructions">
    Segment each instance of each class of objects in the image.
  </full-instructions>

  <short-instructions>
    <p>Segment each instance of each class of objects in the image.</p>

    <h3 style="color: green">GOOD EXAMPLES</h3>
    
    <p>Good because A, B, C.</p>

    <h3 style="color: red">BAD EXAMPLES</h3>
```

```

<p>Bad because X, Y, Z.</p>
</short-instructions>
</crowd-instance-segmentation>
</crowd-form>
```

## Create an image classification job (Single Label)

Use an Amazon SageMaker Ground Truth image classification labeling task when you need workers to classify images using predefined labels that you specify. Workers are shown images and are asked to choose one label for each image. You can create an image classification labeling job using the Ground Truth section of the Amazon SageMaker AI console or the [CreateLabelingJob](#) operation.

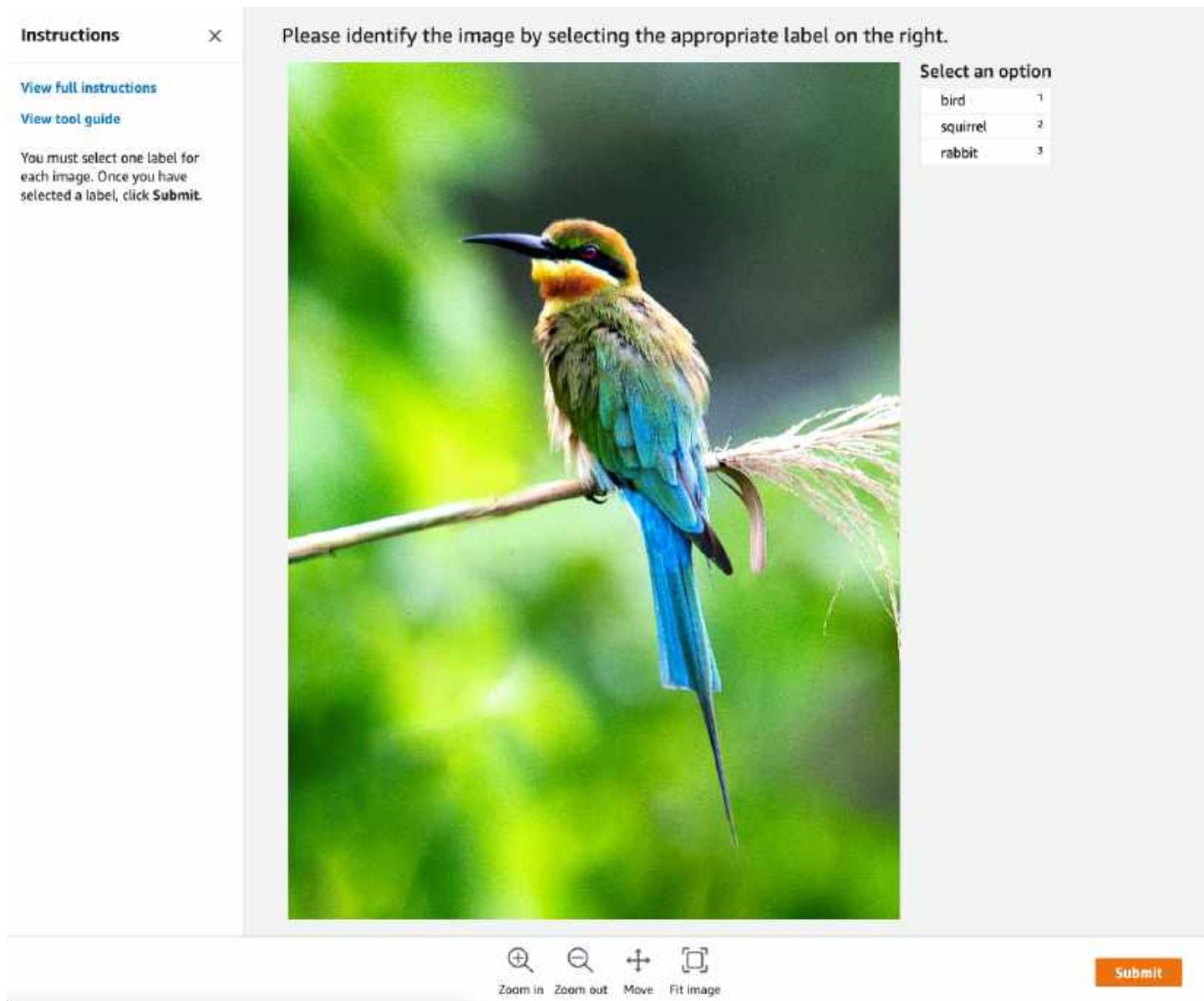
### Important

For this task type, if you create your own manifest file, use "source-ref" to identify the location of each image file in Amazon S3 that you want labeled. For more information, see [Input data](#).

## Create an Image Classification Labeling Job (Console)

You can follow the instructions [Create a Labeling Job \(Console\)](#) to learn how to create a image classification labeling job in the SageMaker AI console. In Step 10, choose **Image** from the **Task category** drop down menu, and choose **Image Classification (Single Label)** as the task type.

Ground Truth provides a worker UI similar to the following for labeling tasks. When you create the labeling job with the console, you specify instructions to help workers complete the job and labels that workers can choose from.



## Create an Image Classification Labeling Job (API)

To create an image classification labeling job, use the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

Follow the instructions on [Create a Labeling Job \(API\)](#) and do the following while you configure your request:

- Pre-annotation Lambda functions for this task type end with `PRE-ImageMultiClass`. To find the pre-annotation Lambda ARN for your Region, see [PreHumanTaskLambdaArn](#) .

- Annotation-consolidation Lambda functions for this task type end with ACS-ImageMultiClass. To find the annotation-consolidation Lambda ARN for your Region, see [AnnotationConsolidationLambdaArn](#).

The following is an example of an [AWS Python SDK \(Boto3\) request](#) to create a labeling job in the US East (N. Virginia) Region. All parameters in red should be replaced with your specifications and resources.

```
response = client.create_labeling_job(
    LabelingJobName='example-image-classification-labeling-job',
    LabelAttributeName='label',
    InputConfig={
        'DataSource': {
            'S3DataSource': {
                'ManifestS3Uri': 's3://bucket/path/manifest-with-input-data.json'
            }
        },
        'DataAttributes': {
            'ContentClassifiers': [
                'FreeOfPersonallyIdentifiableInformation' | 'FreeOfAdultContent',
            ]
        }
    },
    OutputConfig={
        'S3OutputPath': 's3://bucket/path/file-to-store-output-data',
        'KmsKeyId': 'string'
    },
    RoleArn='arn:aws:iam::*:role/*',
    LabelCategoryConfigS3Uri='s3://bucket/path/label-categories.json',
    StoppingConditions={
        'MaxHumanLabeledObjectCount': 123,
        'MaxPercentageOfInputDatasetLabeled': 123
    },
    HumanTaskConfig={
        'WorkteamArn': 'arn:aws:sagemaker:region:*:workteam/private-crowd/*',
        'UiConfig': {
            'UiTemplateS3Uri': 's3://bucket/path/worker-task-template.html'
        },
        'PreHumanTaskLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:PRE-ImageMultiClass',
        'TaskKeywords': [
            'Image classification',
        ]
    }
)
```

```
],
    'TaskTitle': Image classification task,
    'TaskDescription': 'Carefully inspect the image and classify it by selecting one label from the categories provided.',
    'NumberOfHumanWorkersPerDataObject': 123,
    'TaskTimeLimitInSeconds': 123,
    'TaskAvailabilityLifetimeInSeconds': 123,
    'MaxConcurrentTaskCount': 123,
    'AnnotationConsolidationConfig': {
        'AnnotationConsolidationLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:ACS-ImageMultiClass'
    },
    Tags:[
        {
            'Key': 'string',
            'Value': 'string'
        },
    ]
}
```

## Provide a Template for Image Classification Labeling Jobs

If you create a labeling job using the API, you must supply a worker task template in `UiTemplateS3Uri`. Copy and modify the following template. Only modify the [short-instructions](#), [full-instructions](#), and header.

Upload this template to S3, and provide the S3 URI for this file in `UiTemplateS3Uri`.

```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>
<crowd-form>
    <crowd-image-classifier
        name="crowd-image-classifier"
        src="{{ task.input.taskObject | grant_read_access }}"
        header="please classify"
        categories="{{ task.input.labels | to_json | escape }}"
    >
        <full-instructions header="Image classification instructions">
            <ol><li><strong>Read</strong> the task carefully and inspect the image.</li>
                <li><strong>Read</strong> the options and review the examples provided to understand more about the labels.</li>
                    <li><strong>Choose</strong> the appropriate label that best suits the image.</li>
                </ol>
        </full-instructions>
    </crowd-image-classifier>
</crowd-form>
```

```
<short-instructions>
  <h3><span style="color: #008C00;">Good example</span></h3>
  <p>Enter description to explain the correct label to the workers</p>
  <h3><span style="color: #C00000;">Bad example</span></h3><p>Enter
description of an incorrect label</p>
</short-instructions>
</crowd-image-classifier>
</crowd-form>
```

## Image Classification Output Data

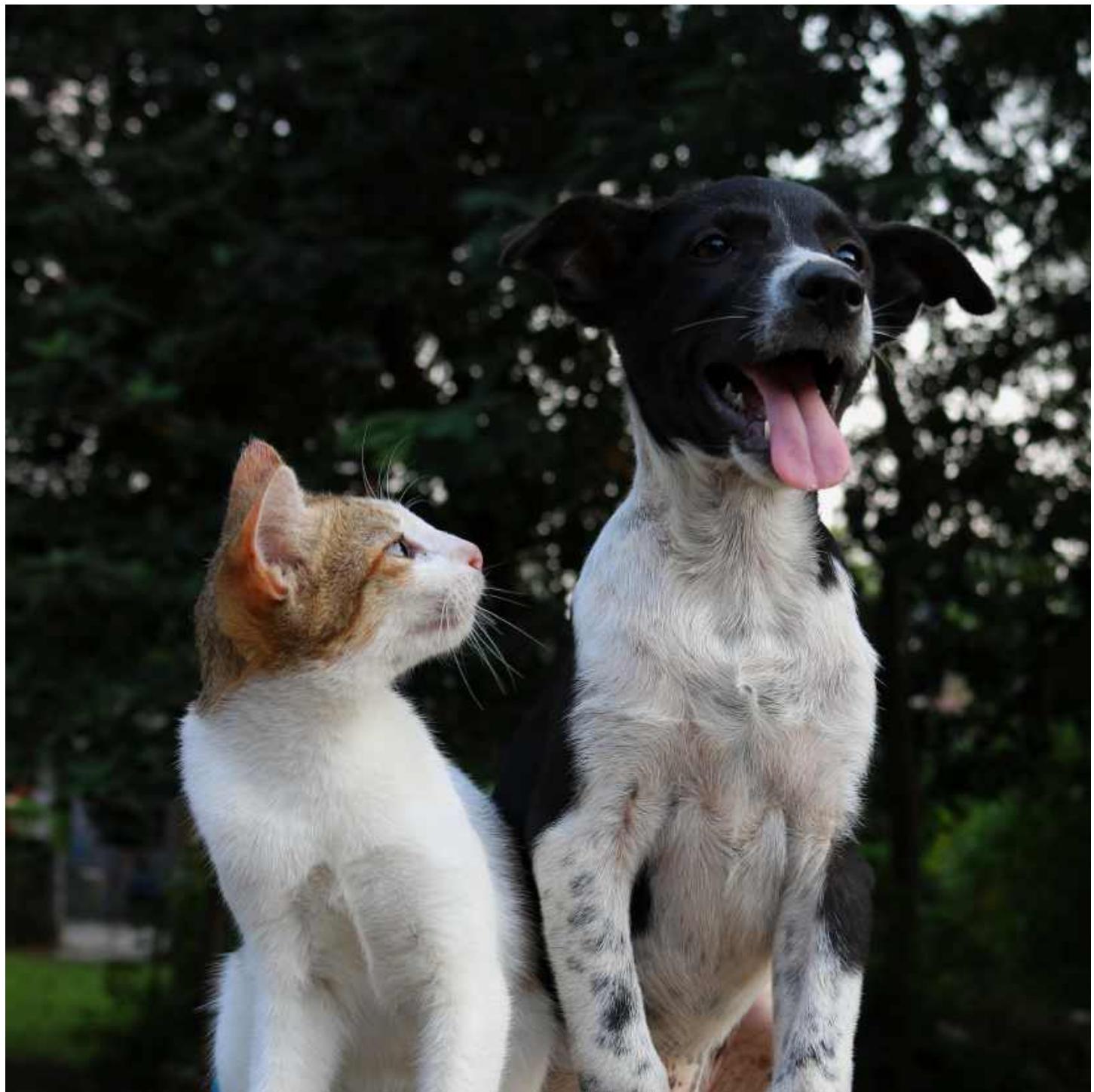
Once you have created an image classification labeling job, your output data will be located in the Amazon S3 bucket specified in the `S3OutputPath` parameter when using the API or in the **Output dataset location** field of the **Job overview** section of the console.

To learn more about the output manifest file generated by Ground Truth and the file structure the Ground Truth uses to store your output data, see [Labeling job output data](#).

To see an example of an output manifest file from an image classification labeling job, see [Classification job output](#).

## Create an image classification job (Multi-label)

Use an Amazon SageMaker Ground Truth multi-label image classification labeling task when you need workers to classify multiple objects in an image. For example, the following image features a dog and a cat. You can use multi-label image classification to associate the labels "dog" and "cat" with this image. The following page gives information about creating an image classification job.



When working on a multi-label image classification task, workers should choose all applicable labels, but must choose at least one. When creating a job using this task type, you can provide up to 50 label-categories.

When creating a labeling job in the console, Ground Truth doesn't provide a "none" category for when none of the labels applies to an image. To provide this option to workers, include a label similar to "none" or "other" when you create a multi-label image classification job.

To restrict workers to choosing a single label for each image, use the [Create an image classification job \(Single Label\)](#) task type.

 **Important**

For this task type, if you create your own manifest file, use "source-ref" to identify the location of each image file in Amazon S3 that you want labeled. For more information, see [Input data](#).

## Create a Multi-Label Image Classification Labeling Job (Console)

You can follow the instructions [Create a Labeling Job \(Console\)](#) to learn how to create a multi-label image classification labeling job in the SageMaker AI console. In Step 10, choose **Image** from the **Task category** drop down menu, and choose **Image Classification (Multi-label)** as the task type.

Ground Truth provides a worker UI similar to the following for labeling tasks. When you create a labeling job in the console, you specify instructions to help workers complete the job and labels that workers can choose from.

Instructions X

[View full instructions](#)

[View tool guide](#)

You must select at least one label for each image.

If multiple labels apply to the image, select multiple labels.

Please read each label and select all of those that apply to this image.



Select an option

pedestrian	1
car	2
ambulance	3
crosswalk	4
trees	5

+ - x

Zoom in Zoom out Move Fit image

Submit

## Create a Multi-Label Image Classification Labeling Job (API)

To create a multi-label image classification labeling job, use the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

Follow the instructions on [Create a Labeling Job \(API\)](#) and do the following while you configure your request:

- Pre-annotation Lambda functions for this task type end with `PRE-ImageMultiClassMultiLabel`. To find the pre-annotation Lambda ARN for your Region, see [PreHumanTaskLambdaArn](#).
- Annotation-consolidation Lambda functions for this task type end with `ACS-ImageMultiClassMultiLabel`. To find the annotation-consolidation Lambda ARN for your Region, see [AnnotationConsolidationLambdaArn](#).

The following is an example of an [AWS Python SDK \(Boto3\) request](#) to create a labeling job in the US East (N. Virginia) Region. All parameters in red should be replaced with your specifications and resources.

```
response = client.create_labeling_job(
    LabelingJobName='example-multi-label-image-classification-labeling-job',
    LabelAttributeName='label',
    InputConfig={
        'DataSource': {
            'S3DataSource': {
                'ManifestS3Uri': 's3://bucket/path/manifest-with-input-data.json'
            }
        },
        'DataAttributes': {
            'ContentClassifiers': [
                'FreeOfPersonallyIdentifiableInformation' | 'FreeOfAdultContent',
            ]
        }
    },
    OutputConfig={
        'S3OutputPath': 's3://bucket/path/file-to-store-output-data',
        'KmsKeyId': 'string'
    },
    RoleArn='arn:aws:iam::*:role/*',
    LabelCategoryConfigS3Uri='s3://bucket/path/label-categories.json',
    StoppingConditions={
        'MaxHumanLabeledObjectCount': 123,
        'MaxPercentageOfInputDatasetLabeled': 123
    },
    HumanTaskConfig={
        'WorkteamArn': 'arn:aws:sagemaker:region::workteam/private-crowd/*',
        'UiConfig': {
            'UiTemplateS3Uri': 's3://bucket/path/worker-task-template.html'
        },
        'PreHumanTaskLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:PRE-ImageMultiClassMultiLabel',
        'TaskKeywords': [
            'Image Classification',
        ],
        'TaskTitle': 'Multi-label image classification task',
        'TaskDescription': 'Select all labels that apply to the images shown',
        'NumberOfHumanWorkersPerDataObject': 123,
        'TaskTimeLimitInSeconds': 123,
```

```
'TaskAvailabilityLifetimeInSeconds': 123,
'MaxConcurrentTaskCount': 123,
'AnnotationConsolidationConfig': {
    'AnnotationConsolidationLambdaArn': 'arn:aws:lambda:us-
east-1:432418664414:function:ACS-ImageMultiClassMultiLabel'
},
Tags:[
{
    'Key': 'string',
    'Value': 'string'
},
]
)
```

## Provide a Template for Multi-label Image Classification

If you create a labeling job using the API, you must supply a worker task template in `UiTemplateS3Uri`. Copy and modify the following template. Only modify the [short-instructions](#), [full-instructions](#), and header.

Upload this template to S3, and provide the S3 URI for this file in `UiTemplateS3Uri`.

```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>
<crowd-form>
<crowd-image-classifier-multi-select
    name="crowd-image-classifier-multi-select"
    src="{{ task.input.taskObject | grant_read_access }}"
    header="Please identify all classes in image"
    categories="{{ task.input.labels | to_json | escape }}"
>
<full-instructions header="Multi Label Image classification instructions">
    <ol><li><strong>Read</strong> the task carefully and inspect the image.</li>
        <li><strong>Read</strong> the options and review the examples provided to
        understand more about the labels.</li>
        <li><strong>Choose</strong> the appropriate labels that best suit the image.</li>
    </ol>
</full-instructions>
<short-instructions>
    <h3><span style="color: #008000;">Good example</span></h3>
    <p>Enter description to explain the correct label to the workers</p>
    <h3><span style="color: #FF0000;">Bad example</span></h3>
    <p>Enter description of an incorrect label</p>
</short-instructions>
```

```
</crowd-image-classifier-multi-select>  
</crowd-form>
```

## Multi-label Image Classification Output Data

Once you have created a multi-label image classification labeling job, your output data will be located in the Amazon S3 bucket specified in the `S3OutputPath` parameter when using the API or in the **Output dataset location** field of the **Job overview** section of the console.

To learn more about the output manifest file generated by Ground Truth and the file structure the Ground Truth uses to store your output data, see [Labeling job output data](#).

To see an example of output manifest files for multi-label image classification labeling job, see [Multi-label classification job output](#).

## Image Label Verification

Building a highly accurate training dataset for your machine learning (ML) algorithm is an iterative process. Typically, you review and continuously adjust your labels until you are satisfied that they accurately represent the ground truth, or what is directly observable in the real world. You can use an Amazon SageMaker Ground Truth image label verification task to direct workers to review a dataset's labels and improve label accuracy. Workers can indicate if the existing labels are correct or rate label quality. They can also add comments to explain their reasoning. Amazon SageMaker Ground Truth supports label verification for [Classify image objects using a bounding box](#) and [Identify image contents using semantic segmentation](#) labels. You create an image label verification labeling job using the Ground Truth section of the Amazon SageMaker AI console or the [CreateLabelingJob](#) operation.

Ground Truth provides a worker console similar to the following for labeling tasks. When you create the labeling job with the console, you can modify the images and content that are shown. To learn how to create a labeling job using the Ground Truth console, see [Create a Labeling Job \(Console\)](#).

**Instructions**

[View full instructions](#)

[View tool guide](#)

[▼ Existing labels](#)

- █ bird
- █ rabbit
- █ squirrel

**Instructions**

Please review the labels selected and corresponding box(es) draw for each animal in the image. If the incorrect animal has been selected, or the box has been incorrectly drawn choose reject. Otherwise, choose accept.

**About existing labels**

Select the appropriate label to identify the animal and draw a box around the animal.

Review the existing labels on the objects and choose the appropriate option.

Select an option

accept	1
reject	2

Add a comment:

█ bird    █ rabbit    █ squirrel



Dimmer   Zoom in   Zoom out   Move   Fit image

**Submit**

You can create a label verification labeling job using the SageMaker AI console or API. To learn how to create a labeling job using the Ground Truth API operation `CreateLabelingJob`, see [Create a Labeling Job \(API\)](#).

## Text labeling with Ground Truth

Use Ground Truth to label text. Ground Truth supports labeling text for named entity recognition, single label text classification, and multi-label text classification. The following topics give information about these built-in task types, as well as instructions to help you create a labeling job using that task type.

### i Tip

To learn more about supported file types and input data quotas, see [Input data](#).

## Topics

- [Extract text information using named entity recognition](#)
- [Categorize text with text classification \(Single Label\)](#)
- [Categorize text with text classification \(Multi-label\)](#)

## Extract text information using named entity recognition

To extract information from unstructured text and classify it into predefined categories, use an Amazon SageMaker Ground Truth named entity recognition (NER) labeling task. Traditionally, NER involves sifting through text data to locate noun phrases, called *named entities*, and categorizing each with a label, such as "person," "organization," or "brand." You can broaden this task to label longer spans of text and categorize those sequences with predefined labels that you specify. You can create a named entity recognition labeling job using the Ground Truth section of the Amazon SageMaker AI console or the [CreateLabelingJob](#) operation.

When tasked with a named entity recognition labeling job, workers apply your labels to specific words or phrases within a larger text block. They choose a label, then apply it by using the cursor to highlight the part of the text to which the label applies. The Ground Truth named entity recognition tool supports overlapping annotations, in-context label selection, and multi-label selection for a single highlight. Also, workers can use their keyboards to quickly select labels.

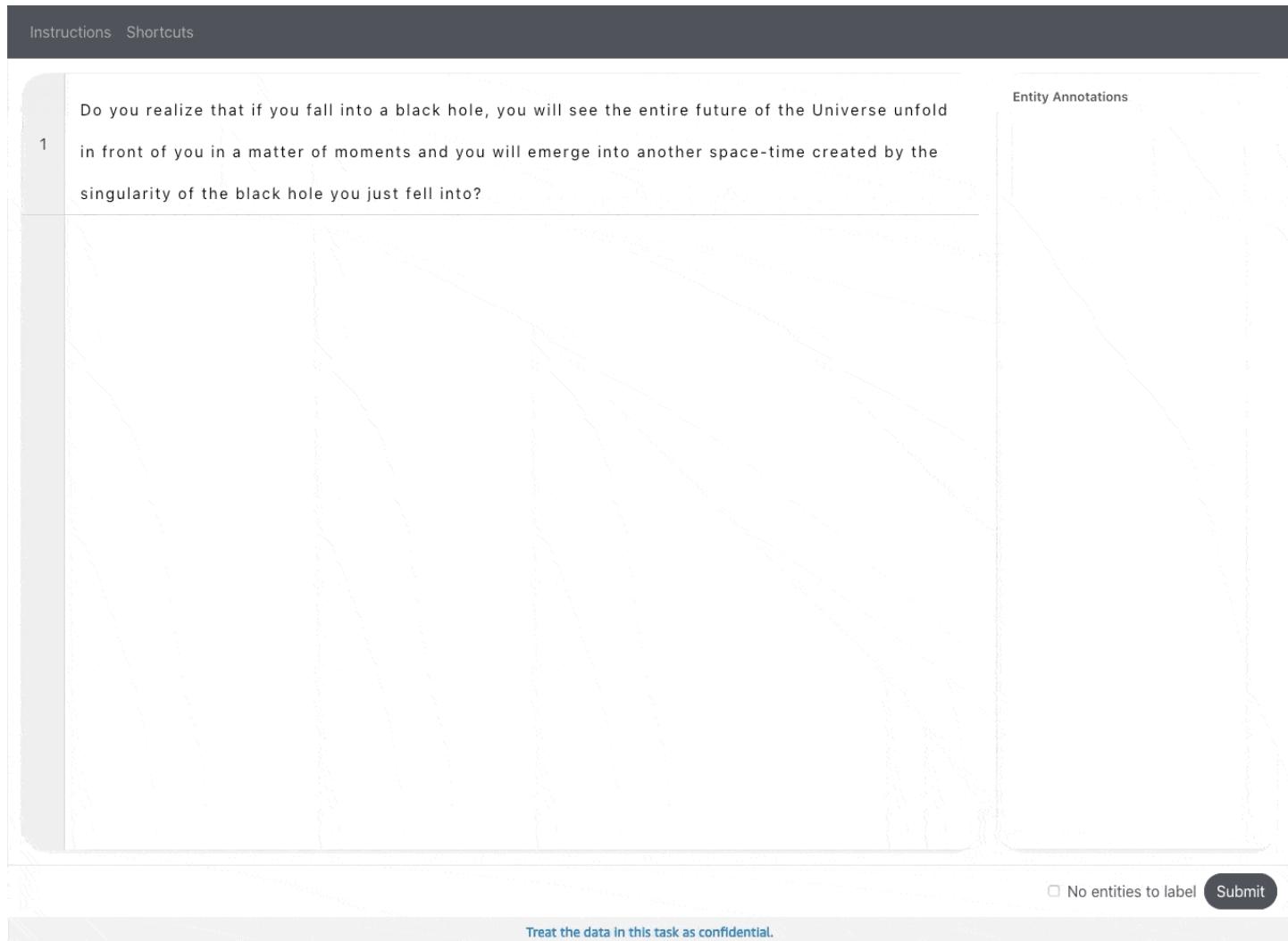
### Important

If you manually create an input manifest file, use "source" to identify the text that you want labeled. For more information, see [Input data](#).

## Create a Named Entity Recognition Labeling Job (Console)

You can follow the instructions [Create a Labeling Job \(Console\)](#) to learn how to create a named entity recognition labeling job in the SageMaker AI console. In Step 10, choose **Text** from the **Task category** drop down menu, and choose **Named entity recognition** as the task type.

Ground Truth provides a worker UI similar to the following for labeling tasks. When you create the labeling job with the console, you specify instructions to help workers complete the job and labels that workers can choose from.



## Create a Named Entity Recognition Labeling Job (API)

To create a named entity recognition labeling job, using the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

Follow the instructions on [Create a Labeling Job \(API\)](#) and do the following while you configure your request:

- Pre-annotation Lambda functions for this task type end with PRE-`NamedEntityRecognition`. To find the pre-annotation Lambda ARN for your Region, see [PreHumanTaskLambdaArn](#).
  - Annotation-consolidation Lambda functions for this task type end with ACS-`NamedEntityRecognition`. To find the annotation-consolidation Lambda ARN for your Region, see [AnnotationConsolidationLambdaArn](#).

- You must provide the following ARN for [HumanTaskUiArn](#):

```
arn:aws:sagemaker:aws-region:394669845002:human-task-ui/NamedEntityRecognition
```

Replace *aws-region* with the AWS Region you use to create the labeling job. For example, use `us-west-1` if you create a labeling job in US West (N. California).

- Provide worker instructions in the label category configuration file using the `instructions` parameter. You can use a string, or HTML markup language in the `shortInstruction` and `fullInstruction` fields. For more details, see [Provide Worker Instructions in a Label Category Configuration File](#).

```
"instructions": {"shortInstruction": "<h1>Add header</h1><p>Add Instructions</p>",  
 "fullInstruction": "<p>Add additional instructions.</p>"}
```

The following is an example of an [AWS Python SDK \(Boto3\) request](#) to create a labeling job in the US East (N. Virginia) Region. All parameters in red should be replaced with your specifications and resources.

```
response = client.create_labeling_job(  
    LabelingJobName='example-ner-labeling-job',  
    LabelAttributeName='label',  
    InputConfig={  
        'DataSource': {  
            'S3DataSource': {  
                'ManifestS3Uri': 's3://bucket/path/manifest-with-input-data.json'  
            }  
        },  
        'DataAttributes': {  
            'ContentClassifiers': [  
                'FreeOfPersonallyIdentifiableInformation' | 'FreeOfAdultContent',  
            ]  
        }  
    },  
    OutputConfig={  
        'S3OutputPath': 's3://bucket/path/file-to-store-output-data',  
        'KmsKeyId': 'string'  
    },  
    RoleArn='arn:aws:iam::*:role/*',  
    LabelCategoryConfigS3Uri='s3://bucket/path/label-categories.json',  
    StoppingConditions={
```

```
'MaxHumanLabeledObjectCount': 123,
'MaxPercentageOfInputDatasetLabeled': 123
},
HumanTaskConfig={
    'WorkteamArn': 'arn:aws:sagemaker:region:*:workteam/private-crowd/*',
    'UiConfig': {
        'HumanTaskUiArn': 'arn:aws:sagemaker:us-east-1:394669845002:human-task-ui/NamedEntityRecognition'
    },
    'PreHumanTaskLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:PRE-NamedEntityRecognition',
    'TaskKeywords': [
        'Named entity Recognition',
    ],
    'TaskTitle': 'Named entity Recognition task',
    'TaskDescription': 'Apply the labels provided to specific words or phrases within the larger text block.',
    'NumberOfHumanWorkersPerDataObject': 1,
    'TaskTimeLimitInSeconds': 28800,
    'TaskAvailabilityLifetimeInSeconds': 864000,
    'MaxConcurrentTaskCount': 1000,
    'AnnotationConsolidationConfig': {
        'AnnotationConsolidationLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:ACS-NamedEntityRecognition'
    },
    Tags:[
        {
            'Key': 'string',
            'Value': 'string'
        },
    ],
}
)
```

## Provide Worker Instructions in a Label Category Configuration File

You must provide worker instructions in the label category configuration file you identify with the `LabelCategoryConfigS3Uri` parameter in `CreateLabelingJob`. You can use these instructions to provide details about the task you want workers to perform and help them use the tool efficiently.

You provide short and long instructions using `shortInstruction` and `fullInstruction` in the `instructions` parameter, respectively. To learn more about these instruction types, see [Create instruction pages](#).

The following is an example of a label category configuration file with instructions that can be used for a named entity recognition labeling job.

```
{  
    "document-version": "2018-11-28",  
    "labels": [  
        {  
            "label": "label1",  
            "shortDisplayName": "L1"  
        },  
        {  
            "label": "label2",  
            "shortDisplayName": "L2"  
        },  
        {  
            "label": "label3",  
            "shortDisplayName": "L3"  
        },  
        {  
            "label": "label4",  
            "shortDisplayName": "L4"  
        },  
        {  
            "label": "label5",  
            "shortDisplayName": "L5"  
        }  
],  
    "instructions": {  
        "shortInstruction": "<p>Enter description of the labels that workers have  
                           to choose from</p><br><p>Add examples to help workers  
understand the label</p>",  

```

```
<li>You can select all of a previously highlighted text, but  
not a portion of it.</li>  
</ol>"  
}  
}
```

## Named Entity Recognition Output Data

Once you have created a named entity recognition labeling job, your output data will be located in the Amazon S3 bucket specified in the `S3OutputPath` parameter when using the API or in the **Output dataset location** field of the **Job overview** section of the console.

To learn more about the output manifest file generated by Ground Truth and the file structure the Ground Truth uses to store your output data, see [Labeling job output data](#).

## Categorize text with text classification (Single Label)

To categorize articles and text into predefined categories, use text classification. For example, you can use text classification to identify the sentiment conveyed in a review or the emotion underlying a section of text. Use Amazon SageMaker Ground Truth text classification to have workers sort text into categories that you define. You create a text classification labeling job using the Ground Truth section of the Amazon SageMaker AI console or the [CreateLabelingJob](#) operation.

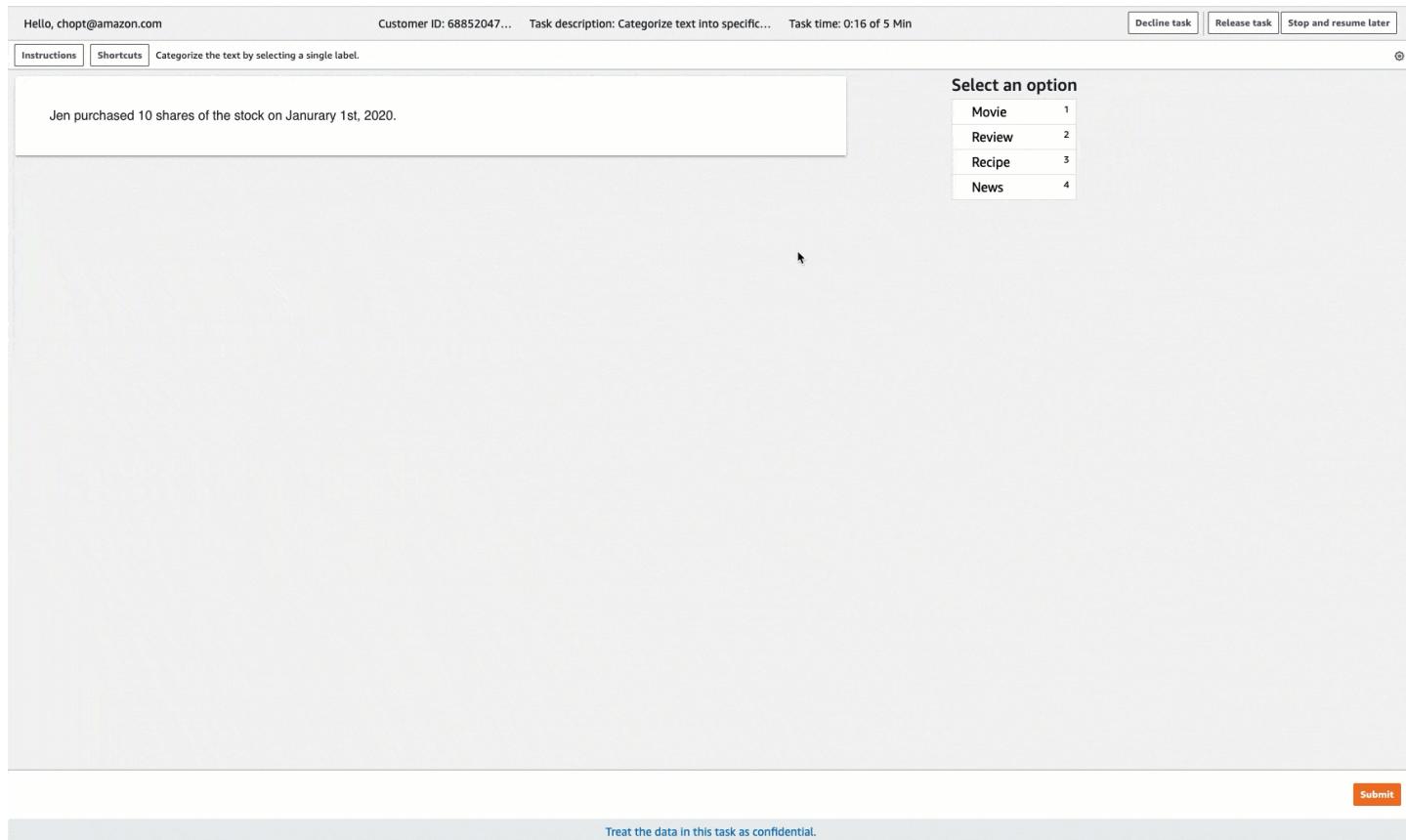
### Important

If you manually create an input manifest file, use "source" to identify the text that you want labeled. For more information, see [Input data](#).

## Create a Text Classification Labeling Job (Console)

You can follow the instructions [Create a Labeling Job \(Console\)](#) to learn how to create a text classification labeling job in the SageMaker AI console. In Step 10, choose **Text** from the **Task category** drop down menu, and choose **Text Classification (Single Label)** as the task type.

Ground Truth provides a worker UI similar to the following for labeling tasks. When you create the labeling job with the console, you specify instructions to help workers complete the job and labels that workers can choose from.



## Create a Text Classification Labeling Job (API)

To create a text classification labeling job, use the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

Follow the instructions on [Create a Labeling Job \(API\)](#) and do the following while you configure your request:

- Pre-annotation Lambda functions for this task type end with PRE-TextMultiClass. To find the pre-annotation Lambda ARN for your Region, see [PreHumanTaskLambdaArn](#).
- Annotation-consolidation Lambda functions for this task type end with ACS-TextMultiClass. To find the annotation-consolidation Lambda ARN for your Region, see [AnnotationConsolidationLambdaArn](#).

The following is an example of an [AWS Python SDK \(Boto3\) request](#) to create a labeling job in the US East (N. Virginia) Region. All parameters in red should be replaced with your specifications and resources.

```
response = client.create_labeling_job(
    LabelingJobName='example-text-classification-labeling-job',
    LabelAttributeName='label',
    InputConfig={
        'DataSource': {
            'S3DataSource': {
                'ManifestS3Uri': 's3://bucket/path/manifest-with-input-data.json'
            }
        },
        'DataAttributes': {
            'ContentClassifiers': [
                'FreeOfPersonallyIdentifiableInformation' | 'FreeOfAdultContent',
            ]
        }
    },
    OutputConfig={
        'S3OutputPath': 's3://bucket/path/file-to-store-output-data',
        'KmsKeyId': 'string'
    },
    RoleArn='arn:aws:iam::*:role/*',
    LabelCategoryConfigS3Uri='s3://bucket/path/label-categories.json',
    StoppingConditions={
        'MaxHumanLabeledObjectCount': 123,
        'MaxPercentageOfInputDatasetLabeled': 123
    },
    HumanTaskConfig={
        'WorkteamArn': 'arn:aws:sagemaker:region::workteam/private-crowd/*',
        'UiConfig': {
            'UiTemplateS3Uri': 's3://bucket/path/worker-task-template.html'
        },
        'PreHumanTaskLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:PRE-
TextMultiClass',
        'TaskKeywords': [
            'Text classification',
        ],
        'TaskTitle': 'Text classification task',
        'TaskDescription': 'Carefully read and classify this text using the categories
provided.',
        'NumberOfHumanWorkersPerDataObject': 123,
        'TaskTimeLimitInSeconds': 123,
        'TaskAvailabilityLifetimeInSeconds': 123,
        'MaxConcurrentTaskCount': 123,
        'AnnotationConsolidationConfig': {
    
```

```
'AnnotationConsolidationLambdaArn': 'arn:aws:lambda:us-  
east-1:432418664414:function:ACS-TextMultiClass'  
    },  
    Tags=[  
        {  
            'Key': 'string',  
            'Value': 'string'  
        },  
    ]  
)
```

## Provide a Template for Text Classification Labeling Jobs

If you create a labeling job using the API, you must supply a worker task template in `UiTemplateS3Uri`. Copy and modify the following template. Only modify the [short-instructions](#), [full-instructions](#), and header.

Upload this template to S3, and provide the S3 URI for this file in `UiTemplateS3Uri`.

```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>  
<crowd-form>  
    <crowd-classifier  
        name="crowd-classifier"  
        categories="{{ task.input.labels | to_json | escape }}"  
        header="classify text"  
    >  
        <classification-target style="white-space: pre-wrap">  
            {{ task.input.taskObject }}  
        </classification-target>  
        <full-instructions header="Classifier instructions">  
            <ol><li><strong>Read</strong> the text carefully.</li>  
            <li><strong>Read</strong> the examples to understand more about the options.</li>  
            <li><strong>Choose</strong> the appropriate labels that best suit the text.</li>  
        </ol>  
        </full-instructions>  
        <short-instructions>  
            <p>Enter description of the labels that workers have to choose from</p>  
            <p><br></p><p><br></p><p>Add examples to help workers understand the label</p>  
            <p><br></p><p><br></p><p><br></p><p><br></p><p><br></p><p><br></p><p><br></p>  
        </short-instructions>  
    </crowd-classifier>  
</crowd-form>
```

## Text Classification Output Data

Once you have created a text classification labeling job, your output data will be located in the Amazon S3 bucket specified in the S3OutputPath parameter when using the API or in the **Output dataset location** field of the **Job overview** section of the console.

To learn more about the output manifest file generated by Ground Truth and the file structure the Ground Truth uses to store your output data, see [Labeling job output data](#).

To see an example of an output manifest files from a text classification labeling job, see [Classification job output](#).

## Categorize text with text classification (Multi-label)

To categorize articles and text into multiple predefined categories, use the multi-label text classification task type. For example, you can use this task type to identify more than one emotion conveyed in text. The following sections give information about how to create a multi-label text classification task from the console and API.

When working on a multi-label text classification task, workers should choose all applicable labels, but must choose at least one. When creating a job using this task type, you can provide up to 50 label categories.

Amazon SageMaker Ground Truth doesn't provide a "none" category for when none of the labels applies. To provide this option to workers, include a label similar to "none" or "other" when you create a multi-label text classification job.

To restrict workers to choosing a single label for each document or text selection, use the [Categorize text with text classification \(Single Label\)](#) task type.

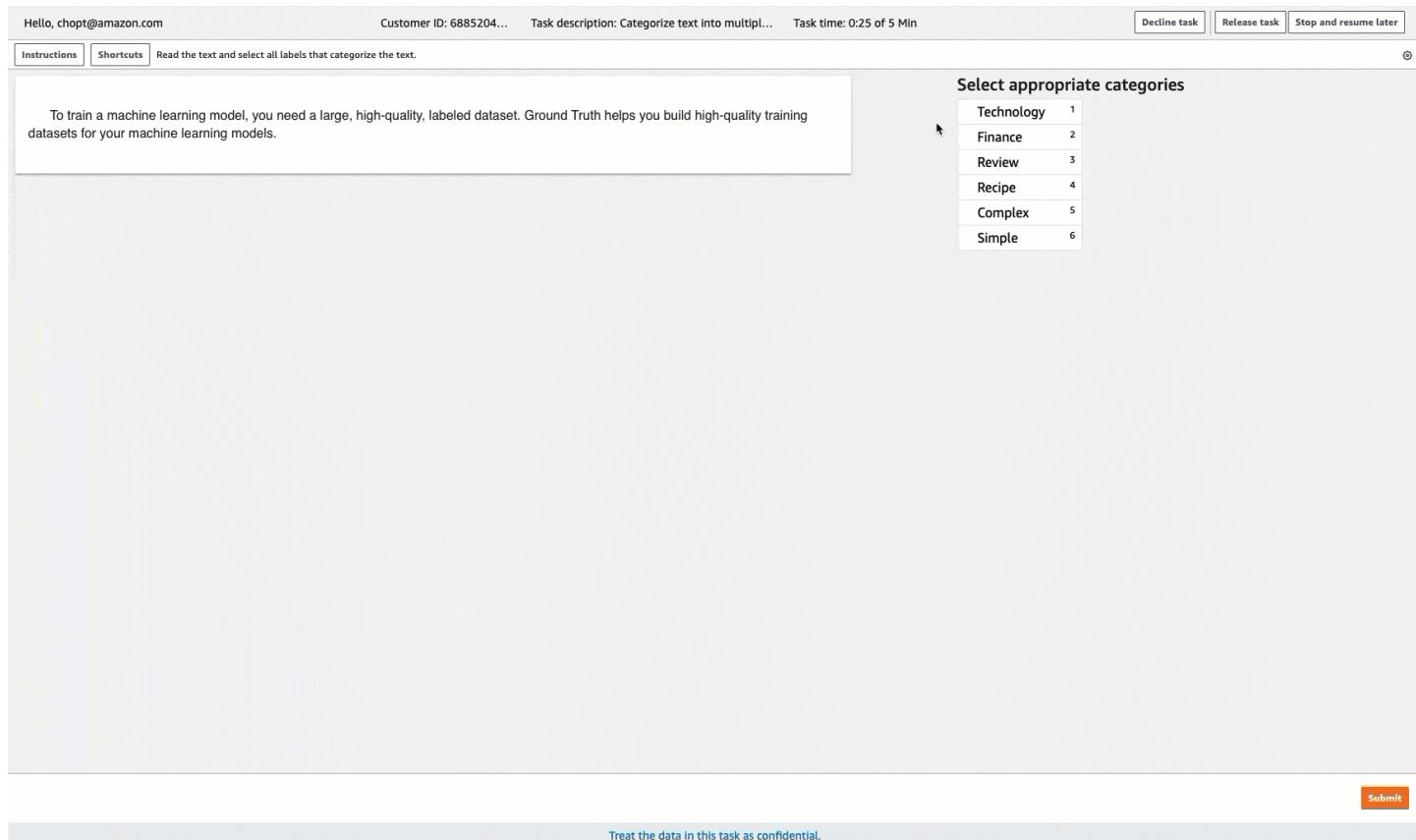
### Important

If you manually create an input manifest file, use "source" to identify the text that you want labeled. For more information, see [Input data](#).

## Create a Multi-Label Text Classification Labeling Job (Console)

You can follow the instructions [Create a Labeling Job \(Console\)](#) to learn how to create a multi-label text classification labeling job in the Amazon SageMaker AI console. In Step 10, choose **Text** from the **Task category** drop down menu, and choose **Text Classification (Multi-label)** as the task type.

Ground Truth provides a worker UI similar to the following for labeling tasks. When you create the labeling job with the console, you specify instructions to help workers complete the job and labels that workers can choose from.



## Create a Multi-Label Text Classification Labeling Job (API)

To create a multi-label text classification labeling job, use the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

Follow the instructions on [Create a Labeling Job \(API\)](#) and do the following while you configure your request:

- Pre-annotation Lambda functions for this task type end with `PRE-TextMultiClassMultiLabel`. To find the pre-annotation Lambda ARN for your Region, see [PreHumanTaskLambdaArn](#).
- Annotation-consolidation Lambda functions for this task type end with `ACS-TextMultiClassMultiLabel`. To find the annotation-consolidation Lambda ARN for your Region, see [AnnotationConsolidationLambdaArn](#).

The following is an example of an [AWS Python SDK \(Boto3\) request](#) to create a labeling job in the US East (N. Virginia) Region. All parameters in red should be replaced with your specifications and resources.

```
response = client.create_labeling_job(
    LabelingJobName='example-multi-label-text-classification-labeling-job',
    LabelAttributeName='label',
    InputConfig={
        'DataSource': {
            'S3DataSource': {
                'ManifestS3Uri': 's3://bucket/path/manifest-with-input-data.json'
            }
        },
        'DataAttributes': {
            'ContentClassifiers': [
                'FreeOfPersonallyIdentifiableInformation' | 'FreeOfAdultContent',
            ]
        }
    },
    OutputConfig={
        'S3OutputPath': 's3://bucket/path/file-to-store-output-data',
        'KmsKeyId': 'string'
    },
    RoleArn='arn:aws:iam::*:role/*',
    LabelCategoryConfigS3Uri='s3://bucket/path/label-categories.json',
    StoppingConditions={
        'MaxHumanLabeledObjectCount': 123,
        'MaxPercentageOfInputDatasetLabeled': 123
    },
    HumanTaskConfig={
        'WorkteamArn': 'arn:aws:sagemaker:region:*:workteam/private-crowd/*',
        'UiConfig': {
            'UiTemplateS3Uri': 's3://bucket/path/custom-worker-task-template.html'
        },
        'PreHumanTaskLambdaArn': 'arn:aws:lambda:function:PRE-  
TextMultiClassMultiLabel',
        'TaskKeywords': [
            'Text Classification',
        ],
        'TaskTitle': 'Multi-label text classification task',
        'TaskDescription': 'Select all labels that apply to the text shown',
        'NumberOfHumanWorkersPerDataObject': 123,
        'TaskTimeLimitInSeconds': 123,
```

```
'TaskAvailabilityLifetimeInSeconds': 123,
'MaxConcurrentTaskCount': 123,
'AnnotationConsolidationConfig': {
    'AnnotationConsolidationLambdaArn': 'arn:aws:lambda:us-
east-1:432418664414:function:ACS-TextMultiClassMultiLabel'
},
Tags:[
{
    'Key': 'string',
    'Value': 'string'
},
]
)
```

## Create a Template for Multi-label Text Classification

If you create a labeling job using the API, you must supply a worker task template in `UiTemplateS3Uri`. Copy and modify the following template. Only modify the [short-instructions](#), [full-instructions](#), and header.

Upload this template to S3, and provide the S3 URI for this file in `UiTemplateS3Uri`.

```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>
<crowd-form>
    <crowd-classifier-multi-select
        name="crowd-classifier-multi-select"
        categories="{{ task.input.labels | to_json | escape }}"
        header="Please identify all classes in the below text"
    >
        <classification-target style="white-space: pre-wrap">
            {{ task.input.taskObject }}
        </classification-target>
        <full-instructions header="Classifier instructions">
            <ol><li><strong>Read</strong> the text carefully.</li>
            <li><strong>Read</strong> the examples to understand more about the options.</li>
            <li><strong>Choose</strong> the appropriate labels that best suit the text.</li>
        </ol>
        </full-instructions>
        <short-instructions>
            <p>Enter description of the labels that workers have to choose from</p>
            <p><br></p>
            <p><br></p><p>Add examples to help workers understand the label</p>
            <p><br></p><p><br></p><p><br></p><p><br></p><p><br></p><p><br></p>
        </short-instructions>
    </crowd-classifier-multi-select>
</crowd-form>
```

```
</short-instructions>
</crowd-classifier-multi-select>
</crowd-form>
```

To learn how to create a custom template, see [Custom labeling workflows](#).

## Multi-label Text Classification Output Data

Once you have created a multi-label text classification labeling job, your output data will be located in the Amazon S3 bucket specified in the S3OutputPath parameter when using the API or in the **Output dataset location** field of the **Job overview** section of the console.

To learn more about the output manifest file generated by Ground Truth and the file structure the Ground Truth uses to store your output data, see [Labeling job output data](#).

To see an example of output manifest files for multi-label text classification labeling job, see [Multi-label classification job output](#).

## Videos and video frame labeling

You can use Ground Truth to classify videos and annotate video frames (still images extracted from videos) using one of the three built-in video task types. These task types streamline the process of creating video and video frame labeling jobs using the Amazon SageMaker AI console, API, and language-specific SDKs.

- Video clip classification – Enable workers to classify videos into categories you specify. For example, you can use this task type to have workers categorize videos into topics like sports, comedy, music, and education. To learn more, see [Classify videos](#).
- Video frame labeling jobs – Enable workers to annotate video frames extracted from a video using bounding boxes, polylines, polygons or keypoint annotation tools. Ground Truth offers two built-in task types to label video frames:
  - *Video frame object detection*: Enable workers to identify and locate objects in video frames.
  - *Video frame object tracking*: Enable workers to track the movement of objects across video frames.
  - *Video frame adjustment jobs*: Have workers adjust labels, label category attributes, and frame attributes from a previous video frame object detection or object tracking labeling job.
  - *Video frame verification jobs*: Have workers verify labels, label category attributes, and frame attributes from a previous video frame object detection or object tracking labeling job.

If you have video files, you can use the Ground Truth automatic frame extraction tool to extract video frames from your videos. To learn more, see [Video Frame Input Data](#).

### Tip

To learn more about supported file types and input data quotas, see [Input data](#).

## Topics

- [Classify videos](#)
- [Video frames](#)
- [Worker Instructions](#)

## Classify videos

Use an Amazon SageMaker Ground Truth video classification labeling task when you need workers to classify videos using predefined labels that you specify. Workers are shown videos and are asked to choose one label for each video. You create a video classification labeling job using the Ground Truth section of the Amazon SageMaker AI console or the [CreateLabelingJob](#) operation.

Your video files must be encoded in a format that is supported by the browser used by the work team that labels your data. It is recommended that you verify that all video file formats in your input manifest file display correctly using the worker UI preview. You can communicate supported browsers to your workers using worker instructions. To see supported file formats, see [Supported data formats](#).

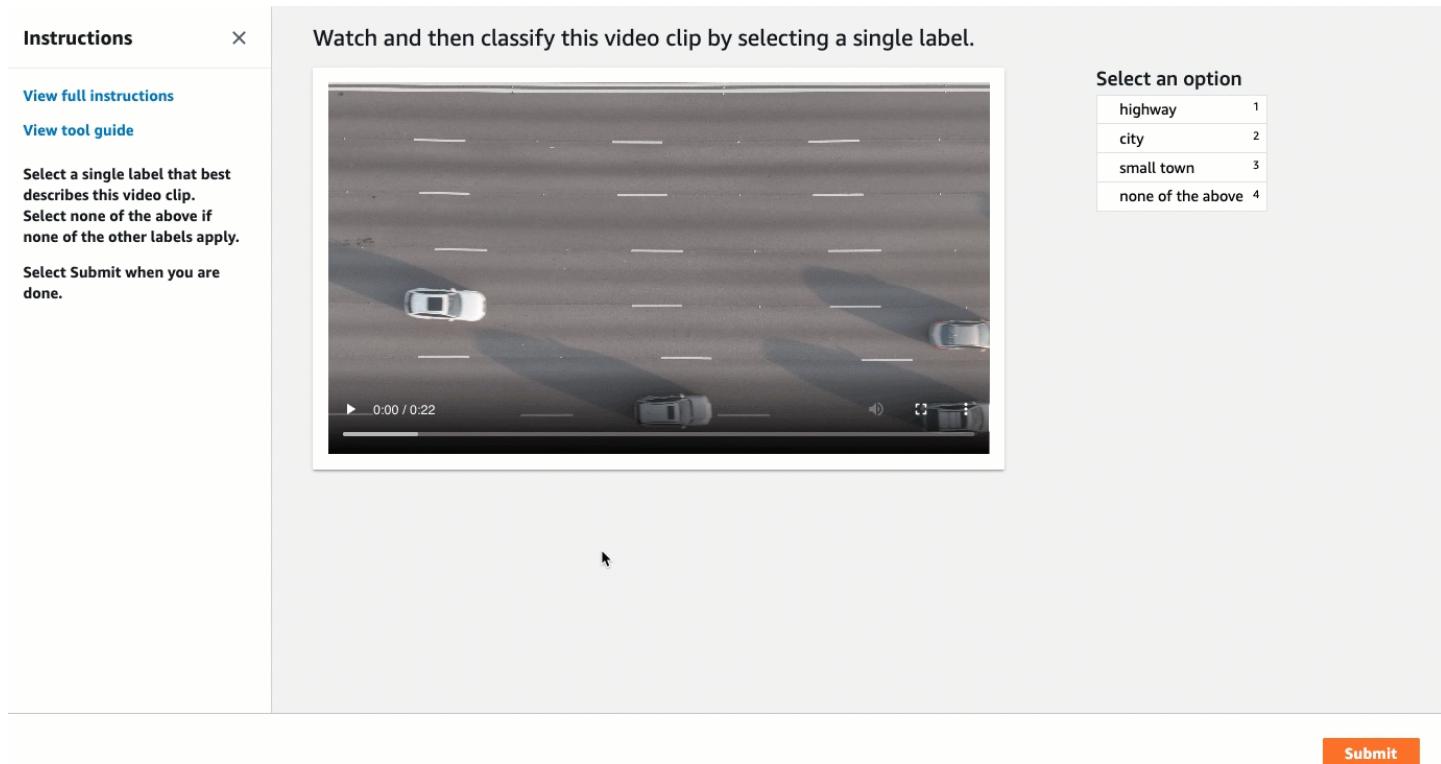
### Important

For this task type, if you create your own manifest file, use "source-ref" to identify the location of each video file in Amazon S3 that you want labeled. For more information, see [Input data](#).

## Create a Video Classification Labeling Job (Console)

You can follow the instructions in [Create a Labeling Job \(Console\)](#) to learn how to create a video classification labeling job in the SageMaker AI console. In step 10, choose **Video** from the **Task category** dropdown list, and choose **Video Classification** as the task type.

Ground Truth provides a worker UI similar to the following for labeling tasks. When you create a labeling job in the console, you specify instructions to help workers complete the job and labels from which workers can choose.



## Create a Video Classification Labeling Job (API)

This section covers details you need to know when you create a labeling job using the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

Follow the instructions on [Create a Labeling Job \(API\)](#) and do the following while you configure your request:

- Use a pre-annotation Lambda function that ends with PRE-VideoClassification. To find the pre-annotation Lambda ARN for your Region, see [PreHumanTaskLambdaArn](#).

- Use an annotation-consolidation Lambda function that ends with ACS-VideoClassification. To find the annotation-consolidation Lambda ARN for your Region, see [AnnotationConsolidationLambdaArn](#).

The following is an example of an [AWS Python SDK \(Boto3\) request](#) to create a labeling job in the US East (N. Virginia) Region.

```
response = client.create_labeling_job(
    LabelingJobName='example-video-classification-labeling-job',
    LabelAttributeName='label',
    InputConfig={
        'DataSource': {
            'S3DataSource': {
                'ManifestS3Uri': 's3://bucket/path/manifest-with-input-data.json'
            }
        },
        'DataAttributes': {
            'ContentClassifiers': [
                'FreeOfPersonallyIdentifiableInformation' | 'FreeOfAdultContent',
            ]
        }
    },
    OutputConfig={
        'S3OutputPath': 's3://bucket/path/file-to-store-output-data',
        'KmsKeyId': 'string'
    },
    RoleArn='arn:aws:iam::*:role/*',
    LabelCategoryConfigS3Uri='s3://bucket/path/label-categories.json',
    StoppingConditions={
        'MaxHumanLabeledObjectCount': 123,
        'MaxPercentageOfInputDatasetLabeled': 123
    },
    HumanTaskConfig={
        'WorkteamArn': 'arn:aws:sagemaker:region::workteam/private-crowd/*',
        'UiConfig': {
            'UiTemplateS3Uri': 's3://bucket/path/worker-task-template.html'
        },
        'PreHumanTaskLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:PRE-VideoClassification',
        'TaskKeywords': [
            'Video Classification',
        ],
    },
)
```

```
'TaskTitle': 'Video classification task',
'TaskDescription': 'Select a label to classify this video',
'NumberOfHumanWorkersPerDataObject': 123,
'TaskTimeLimitInSeconds': 123,
'TaskAvailabilityLifetimeInSeconds': 123,
'MaxConcurrentTaskCount': 123,
'AnnotationConsolidationConfig': {
    'AnnotationConsolidationLambdaArn': 'arn:aws:lambda:us-
east-1:432418664414:function:ACS-VideoClassification'
},
Tags:[
{
    'Key': 'string',
    'Value': 'string'
},
]
)
```

## Provide a Template for Video Classification

If you create a labeling job using the API, you must supply a worker task template in `UiTemplateS3Uri`. Copy and modify the following template by modifying the `short-instructions`, `full-instructions`, and `header`. Upload this template to Amazon S3, and provide the Amazon S3 URI to this file in `UiTemplateS3Uri`.

```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>

<crowd-form>
    <crowd-classifier
        name="crowd-classifier"
        categories="{{ task.input.labels | to_json | escape }}"
        header="Please classify video"
    >
        <classification-target>
            <video width="100%" controls/>
            <source src="{{ task.input.taskObject | grant_read_access }}"
type="video/mp4"/>
            <source src="{{ task.input.taskObject | grant_read_access }}"
type="video/webm"/>
            <source src="{{ task.input.taskObject | grant_read_access }}"
type="video/ogg"/>
            Your browser does not support the video tag.
        </video>
```

```
</classification-target>
<full-instructions header="Video classification instructions">
    <ol><li><strong>Read</strong> the task carefully and inspect the
video.</li>
        <li><strong>Read</strong> the options and review the examples
provided to understand more about the labels.</li>
        <li><strong>Choose</strong> the appropriate label that best
suits the video.</li></ol>
    </full-instructions>
    <short-instructions>
        <h3><span style="color: rgb(0, 138, 0);">Good example</span></h3>
        <p>Enter description to explain the correct label to the
workers</p>
        <p></p>
        <h3><span style="color: rgb(230, 0, 0);">Bad example</span></
h3>
        <p>Enter description of an incorrect label</p>
        <p></p>
    </short-instructions>
    </crowd-classifier>
</crowd-form>
```

## Video Classification Output Data

Once you have created a video classification labeling job, your output data is located in the Amazon S3 bucket specified in the `S3OutputPath` parameter when using the API or in the **Output dataset location** field of the **Job overview** section of the console.

To learn more about the output manifest file generated by Ground Truth and the file structure the Ground Truth uses to store your output data, see [Labeling job output data](#).

To see an example of output manifest files for video classification labeling jobs, see [Classification job output](#).

## Video frames

You can use Ground Truth built-in video frame task types to have workers annotate video frames using bounding boxes, polylines, polygons or keypoints. A *video frame* is a sequence of images that have been extracted from a video.

If you do not have video frames, you can provide video files (MP4 files) and use the Ground Truth automated frame extraction tool to extract video frames. To learn more, see [Provide Video Files](#).

You can use the following built-in video task types to create video frame labeling jobs using the Amazon SageMaker AI console, API, and language-specific SDKs.

- **Video frame object detection** – Use this task type when you want workers to identify and locate objects in sequences of video frames. You provide a list of categories, and workers can select one category at a time and annotate objects which the category applies to in all frames. For example, you can use this task to ask workers to identify and localize various objects in a scene, such as cars, bikes, and pedestrians.
- **Video frame object tracking** – Use this task type when you want workers to track the movement of instances of objects across sequences of video frames. When a worker adds an annotation to a single frame, that annotation is associated with a unique instance ID. The worker adds annotations associated with the same ID in all other frames to identify the same object or person. For example, a worker can track the movement of a vehicle across a sequences of video frames by drawing bounding boxes associated with the same ID around the vehicle in each frame that it appears.

Use the following topics to learn more about these built-in task types and to how to create a labeling job using each task type. See [Task types](#) to learn more about the annotations tools (bounding boxes, polylines, polygons and keypoints) available for these task types.

Before you create a labeling job, we recommend that you review [Video frame labeling job reference](#).

### Topics

- [Identify objects using video frame object detection](#)
- [Track objects in video frames using video frame object tracking](#)
- [Video frame labeling job reference](#)

## Identify objects using video frame object detection

You can use the video frame object detection task type to have workers identify and locate objects in a sequence of video frames (images extracted from a video) using bounding boxes, polylines, polygons or keypoint *annotation tools*. The tool you choose defines the video frame task type you create. For example, you can use a bounding box video frame object detection task type workers to identify and localize various objects in a series of video frames, such as cars, bikes, and pedestrians. You can create a video frame object detection labeling job using the Amazon SageMaker AI Ground Truth console, the SageMaker API, and language-specific AWS SDKs. To learn more, see [Create a Video Frame Object Detection Labeling Job](#) and select your preferred method. See [Task types](#) to learn more about the annotations tools you can choose from when you create a labeling job.

Ground Truth provides a worker UI and tools to complete your labeling job tasks: [Preview the Worker UI](#).

You can create a job to adjust annotations created in a video object detection labeling job using the video object detection adjustment task type. To learn more, see [Create Video Frame Object Detection Adjustment or Verification Labeling Job](#).

### Preview the Worker UI

Ground Truth provides workers with a web user interface (UI) to complete your video frame object detection annotation tasks. You can preview and interact with the worker UI when you create a labeling job in the console. If you are a new user, we recommend that you create a labeling job through the console using a small input dataset to preview the worker UI and ensure your video frames, labels, and label attributes appear as expected.

The UI provides workers with the following assistive labeling tools to complete your object detection tasks:

- For all tasks, workers can use the **Copy to next** and **Copy to all** features to copy an annotation to the next frame or to all subsequent frames respectively.
- For tasks that include the bounding box tools, workers can use a **Predict next** feature to draw a bounding box in a single frame, and then have Ground Truth predict the location of boxes with the same label in all other frames. Workers can then make adjustments to correct predicted box locations.

## Create a Video Frame Object Detection Labeling Job

You can create a video frame object detection labeling job using the SageMaker AI console or the [CreateLabelingJob](#) API operation.

This section assumes that you have reviewed the [Video frame labeling job reference](#) and have chosen the type of input data and the input dataset connection you are using.

### Create a Labeling Job (Console)

You can follow the instructions in [Create a Labeling Job \(Console\)](#) to learn how to create a video frame object tracking job in the SageMaker AI console. In step 10, choose **Video - Object detection** from the **Task category** dropdown list. Select the task type you want by selecting one of the cards in **Task selection**.

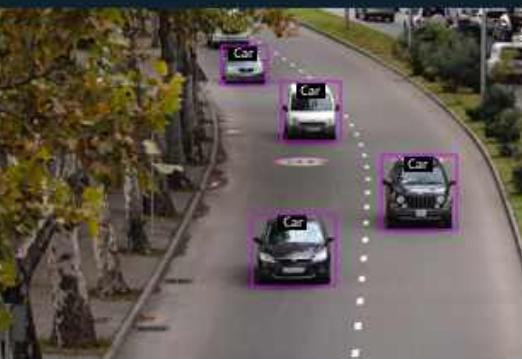
**Task type** [Info](#)

**Task category**  
Select the type of data being labeled to view available task templates for it or select 'Custom' to create your own.

**Video - Object detection** ▾

**Task selection**  
Select the task that a human worker will perform to label objects in your dataset.

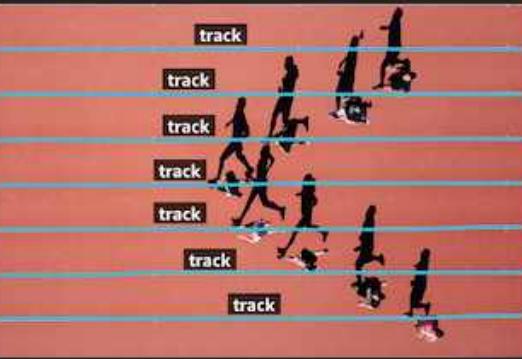
**Bounding box**  
Get workers to draw bounding boxes around specified objects in your video. [Info](#)



**Polygon**  
Get workers to draw polygons around specified objects in your video. [Info](#)



**Polyline**  
Get workers to draw polyline around specified objects in your video. [Info](#)



**Key point**  
Get workers to draw key points around specified objects in your video. [Info](#)



## Create a Labeling Job (API)

You create an object detection labeling job using the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

[Create a Labeling Job \(API\)](#) provides an overview of the CreateLabelingJob operation. Follow these instructions and do the following while you configure your request:

- You must enter an ARN for HumanTaskUiArn. Use  
`arn:aws:sagemaker:<region>:394669845002:human-task-ui/VideoObjectDetection`. Replace `<region>` with the AWS Region in which you are creating the labeling job.

Do not include an entry for the UiTemplateS3Uri parameter.

- Your [LabelAttributeName](#) must end in `-ref`. For example, `video-od-labels-ref`.
- Your input manifest file must be a video frame sequence manifest file. You can create this manifest file using the SageMaker AI console, or create it manually and upload it to Amazon S3. For more information, see [Input Data Setup](#).
- You can only use private or vendor work teams to create video frame object detection labeling jobs.
- You specify your labels, label category and frame attributes, the task type, and worker instructions in a label category configuration file. Specify the task type (bounding boxes, polylines, polygons or keypoint) using annotationType in your label category configuration file. For more information, see [Labeling category configuration file with label category and frame attributes reference](#) to learn how to create this file.
- You need to provide pre-defined ARNs for the pre-annotation and post-annotation (ACS) Lambda functions. These ARNs are specific to the AWS Region you use to create your labeling job.
  - To find the pre-annotation Lambda ARN, refer to [PreHumanTaskLambdaArn](#). Use the Region in which you are creating your labeling job to find the correct ARN that ends with PRE-VideoObjectDetection.
  - To find the post-annotation Lambda ARN, refer to [AnnotationConsolidationLambdaArn](#). Use the Region in which you are creating your labeling job to find the correct ARN that ends with ACS-VideoObjectDetection.
- The number of workers specified in `NumberOfHumanWorkersPerDataObject` must be 1.
- Automated data labeling is not supported for video frame labeling jobs. Do not specify values for parameters in [LabelingJobAlgorithmsConfig](#).
- Video frame object tracking labeling jobs can take multiple hours to complete. You can specify a longer time limit for these labeling jobs in `TaskTimeLimitInSeconds` (up to 7 days, or 604,800 seconds).

The following is an example of an [AWS Python SDK \(Boto3\) request](#) to create a labeling job in the US East (N. Virginia) Region.

```
response = client.create_labeling_job(
    LabelingJobName='example-video-od-labeling-job',
    LabelAttributeName='label',
    InputConfig={
        'DataSource': {
            'S3DataSource': {
                'ManifestS3Uri': 's3://amzn-s3-demo-bucket/path/video-frame-sequence-
input-manifest.json'
            }
        },
        'DataAttributes': {
            'ContentClassifiers': [
                'FreeOfPersonallyIdentifiableInformation' | 'FreeOfAdultContent',
            ]
        }
    },
    OutputConfig={
        'S3OutputPath': 's3://amzn-s3-demo-bucket/prefix/file-to-store-output-data',
        'KmsKeyId': 'string'
    },
    RoleArn='arn:aws:iam::*:role/*',
    LabelCategoryConfigS3Uri='s3://bucket/prefix/label-categories.json',
    StoppingConditions={
        'MaxHumanLabeledObjectCount': 123,
        'MaxPercentageOfInputDatasetLabeled': 123
    },
    HumanTaskConfig={
        'WorkteamArn': 'arn:aws:sagemaker:us-east-1:*:workteam/private-crowd/*',
        'UiConfig': {
            'HumanTaskUiArn': 'arn:aws:sagemaker:us-east-1:394669845002:human-task-ui/
VideoObjectDetection'
        },
        'PreHumanTaskLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:PRE-
VideoObjectDetection',
        'TaskKeywords': [
            'Video Frame Object Detection',
        ],
        'TaskTitle': 'Video frame object detection task',
        'TaskDescription': 'Classify and identify the location of objects and people in
video frames',
    }
)
```

```
'NumberOfHumanWorkersPerDataObject': 123,
'TaskTimeLimitInSeconds': 123,
'TaskAvailabilityLifetimeInSeconds': 123,
'MaxConcurrentTaskCount': 123,
'AnnotationConsolidationConfig': {
    'AnnotationConsolidationLambdaArn': 'arn:aws:lambda:us-
east-1:432418664414:function:ACS-VideoObjectDetection'
},
Tags:[
{
    'Key': 'string',
    'Value': 'string'
},
]
)
```

## Create Video Frame Object Detection Adjustment or Verification Labeling Job

You can create an adjustment and verification labeling job using the Ground Truth console or `CreateLabelingJob` API. To learn more about adjustment and verification labeling jobs, and to learn how create one, see [Label verification and adjustment](#).

### Output Data Format

When you create a video frame object detection labeling job, tasks are sent to workers. When these workers complete their tasks, labels are written to the Amazon S3 output location you specified when you created the labeling job. To learn about the video frame object detection output data format, see [Video frame object detection output](#). If you are a new user of Ground Truth, see [Labeling job output data](#) to learn more about the Ground Truth output data format.

### Track objects in video frames using video frame object tracking

You can use the video frame object tracking task type to have workers track the movement of objects in a sequence of video frames (images extracted from a video) using bounding boxes, polylines, polygons or keypoint *annotation tools*. The tool you choose defines the video frame task type you create. For example, you can use a bounding box video frame object tracking task type to ask workers to track the movement of objects, such as cars, bikes, and pedestrians by drawing boxes around them.

You provide a list of categories, and each annotation that a worker adds to a video frame is identified as an *instance* of that category using an instance ID. For example, if you provide the label category car, the first car that a worker annotates will have the instance ID car:1. The second car

the worker annotates will have the instance ID car:2. To track an object's movement, the worker adds annotations associated with the same instance ID around to object in all frames.

You can create a video frame object tracking labeling job using the Amazon SageMaker AI Ground Truth console, the SageMaker API, and language-specific AWS SDKs. To learn more, see [Create a Video Frame Object Detection Labeling Job](#) and select your preferred method. See [Task types](#) to learn more about the annotations tools you can choose from when you create a labeling job.

Ground Truth provides a worker UI and tools to complete your labeling job tasks: [Preview the Worker UI](#).

You can create a job to adjust annotations created in a video object detection labeling job using the video object detection adjustment task type. To learn more, see [Create Video Frame Object Detection Adjustment or Verification Labeling Job](#).

## Preview the Worker UI

Ground Truth provides workers with a web user interface (UI) to complete your video frame object tracking annotation tasks. You can preview and interact with the worker UI when you create a labeling job in the console. If you are a new user, we recommend that you create a labeling job through the console using a small input dataset to preview the worker UI and ensure your video frames, labels, and label attributes appear as expected.

The UI provides workers with the following assistive labeling tools to complete your object tracking tasks:

- For all tasks, workers can use the **Copy to next** and **Copy to all** features to copy an annotation with the same unique ID to the next frame or to all subsequent frames respectively.
- For tasks that include the bounding box tools, workers can use a **Predict next** feature to draw a bounding box in a single frame, and then have Ground Truth predict the location of boxes with the same unique ID in all other frames. Workers can then make adjustments to correct predicted box locations.

## Create a Video Frame Object Tracking Labeling Job

You can create a video frame object tracking labeling job using the SageMaker AI console or the [CreateLabelingJob](#) API operation.

This section assumes that you have reviewed the [Video frame labeling job reference](#) and have chosen the type of input data and the input dataset connection you are using.

## Create a Labeling Job (Console)

You can follow the instructions in [Create a Labeling Job \(Console\)](#) to learn how to create a video frame object tracking job in the SageMaker AI console. In step 10, choose **Video - Object tracking** from the **Task category** dropdown list. Select the task type you want by selecting one of the cards in **Task selection**.

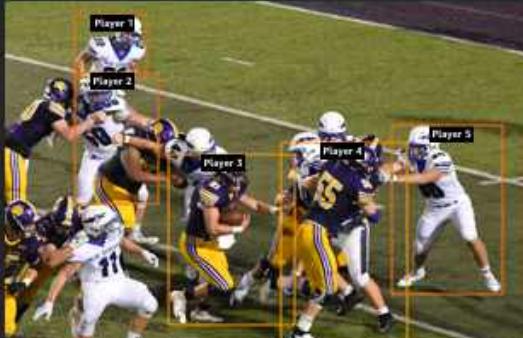
**Task type** [Info](#)

**Task category**  
Select the type of data being labeled to view available task templates for it or select 'Custom' to create your own.

**Video - Object tracking**

**Task selection**  
Select the task that a human worker will perform to label objects in your dataset.

**Bounding box**  
Get workers to track specific instances of objects in your video across multiple frames in your bounding boxes. [Info](#)



**Polygon**  
Get workers to track specific instances of objects in your video across multiple frames in your polygons. [Info](#)



**Polyline**  
Get workers to track specific instances of objects in your video across multiple frames in your polylines. [Info](#)



**Key point**  
Get workers to draw key points around specified objects in your video. [Info](#)



## Create a Labeling Job (API)

You create an object tracking labeling job using the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

[Create a Labeling Job \(API\)](#) provides an overview of the `CreateLabelingJob` operation. Follow these instructions and do the following while you configure your request:

- You must enter an ARN for `HumanTaskUiArn`. Use

`arn:aws:sagemaker:<region>:394669845002:human-task-ui/`

`VideoObjectTracking`. Replace `<region>` with the AWS Region in which you are creating the labeling job.

Do not include an entry for the `UiTemplateS3Uri` parameter.

- Your [LabelAttributeName](#) must end in `-ref`. For example, `ot-labels-ref`.
- Your input manifest file must be a video frame sequence manifest file. You can create this manifest file using the SageMaker AI console, or create it manually and upload it to Amazon S3. For more information, see [Input Data Setup](#). If you create a streaming labeling job, the input manifest file is optional.
- You can only use private or vendor work teams to create video frame object detection labeling jobs.
- You specify your labels, label category and frame attributes, the task type, and worker instructions in a label category configuration file. Specify the task type (bounding boxes, polylines, polygons or keypoint) using `annotationType` in your label category configuration file. For more information, see [Labeling category configuration file with label category and frame attributes reference](#) to learn how to create this file.
- You need to provide pre-defined ARNs for the pre-annotation and post-annotation (ACS) Lambda functions. These ARNs are specific to the AWS Region you use to create your labeling job.
  - To find the pre-annotation Lambda ARN, refer to [PreHumanTaskLambdaArn](#). Use the Region in which you are creating your labeling job to find the correct ARN that ends with `PRE-VideoObjectTracking`.
  - To find the post-annotation Lambda ARN, refer to [AnnotationConsolidationLambdaArn](#). Use the Region in which you are creating your labeling job to find the correct ARN that ends with `ACS-VideoObjectTracking`.

- The number of workers specified in `NumberOfHumanWorkersPerDataObject` must be 1.
- Automated data labeling is not supported for video frame labeling jobs. Do not specify values for parameters in `LabelingJobAlgorithmsConfig`.
- Video frame object tracking labeling jobs can take multiple hours to complete. You can specify a longer time limit for these labeling jobs in `TaskTimeLimitInSeconds` (up to 7 days, or 604,800 seconds).

The following is an example of an [AWS Python SDK \(Boto3\) request](#) to create a labeling job in the US East (N. Virginia) Region.

```
response = client.create_labeling_job(  
    LabelingJobName='example-video-ot-labeling-job',  
    LabelAttributeName='label',  
    InputConfig={  
        'DataSource': {  
            'S3DataSource': {  
                'ManifestS3Uri': 's3://amzn-s3-demo-bucket/path/video-frame-sequence-  
input-manifest.json'  
            }  
        },  
        'DataAttributes': {  
            'ContentClassifiers': [  
                'FreeOfPersonallyIdentifiableInformation' | 'FreeOfAdultContent',  
            ]  
        }  
    },  
    OutputConfig={  
        'S3OutputPath': 's3://amzn-s3-demo-bucket/prefix/file-to-store-output-data',  
        'KmsKeyId': 'string'  
    },  
    RoleArn='arn:aws:iam::*:role/*',  
    LabelCategoryConfigS3Uri='s3://bucket/prefix/label-categories.json',  
    StoppingConditions={  
        'MaxHumanLabeledObjectCount': 123,  
        'MaxPercentageOfInputDatasetLabeled': 123  
    },  
    HumanTaskConfig={  
        'WorkteamArn': 'arn:aws:sagemaker:us-east-1:*:workteam/private-crowd/*',  
        'UiConfig': {  
            'HumanTaskUiArn': 'arn:aws:sagemaker:us-east-1:394669845002:human-task-ui/  
VideoObjectTracking'
```

```
    },
    'PreHumanTaskLambdaArn': 'arn:aws:lambda:us-east-1:432418664414:function:PRE-
VideoObjectTracking',
    'TaskKeywords': [
        'Video Frame Object Tracking',
    ],
    'TaskTitle': 'Video frame object tracking task',
    'TaskDescription': 'Tracking the location of objects and people across video
frames',
    'NumberOfHumanWorkersPerDataObject': 123,
    'TaskTimeLimitInSeconds': 123,
    'TaskAvailabilityLifetimeInSeconds': 123,
    'MaxConcurrentTaskCount': 123,
    'AnnotationConsolidationConfig': {
        'AnnotationConsolidationLambdaArn': 'arn:aws:lambda:us-
east-1:432418664414:function:ACS-VideoObjectTracking'
    },
    Tags:[
        {
            'Key': 'string',
            'Value': 'string'
        },
    ],
}
```

## Create a Video Frame Object Tracking Adjustment or Verification Labeling Job

You can create an adjustment and verification labeling job using the Ground Truth console or `CreateLabelingJob` API. To learn more about adjustment and verification labeling jobs, and to learn how create one, see [Label verification and adjustment](#).

### Output Data Format

When you create a video frame object tracking labeling job, tasks are sent to workers. When these workers complete their tasks, labels are written to the Amazon S3 output location you specified when you created the labeling job. To learn about the video frame object tracking output data format, see [Video frame object tracking output](#). If you are a new user of Ground Truth, see [Labeling job output data](#) to learn more about the Ground Truth output data format.

### Video frame labeling job reference

Use this page to learn about the object detection and object tracking video frame labeling jobs. The information on this page applies to both of these built-in task types.

The video frame labeling job is unique because of the following:

- You can either provide data objects that are ready to be annotated (video frames), or you can provide video files and have Ground Truth automatically extract video frames.
- Workers have the ability to save work as they go.
- You cannot use the Amazon Mechanical Turk workforce to complete your labeling tasks.
- Ground Truth provides a worker UI, as well as assistive and basic labeling tools, to help workers complete your tasks. You do not need to provide a worker task template.

Use the following topics to learn more about video frame labeling jobs.

## Topics

- [Input data](#)
- [Job completion times](#)
- [Task types](#)
- [Workforces](#)
- [Worker user interface \(UI\)](#)
- [Video frame job permission requirements](#)

## Input data

The video frame labeling job uses *sequences* of video frames. A single sequence is a series of images that have been extracted from a single video. You can either provide your own sequences of video frames, or have Ground Truth automatically extract video frame sequences from your video files. To learn more, see [Provide Video Files](#).

Ground Truth uses sequence files to identify all images in a single sequence. All of the sequences that you want to include in a single labeling job are identified in an input manifest file. Each sequence is used to create a single worker task. You can automatically create sequence files and an input manifest file using Ground Truth automatic data setup. To learn more, see [Set up Automated Video Frame Input Data](#).

To learn how to manually create sequence files and an input manifest file, see [Create a Video Frame Input Manifest File](#).

## Job completion times

Video and video frame labeling jobs can take workers hours to complete. You can set the total amount of time that workers can work on each task when you create a labeling job. The maximum time you can set for workers to work on tasks is 7 days. The default value is 3 days.

We strongly recommend that you create tasks that workers can complete within 12 hours. Workers must keep the worker UI open while working on a task. They can save work as they go and Ground Truth saves their work every 15 minutes.

When using the SageMaker AI `CreateLabelingJob` API operation, set the total time a task is available to workers in the `TaskTimeLimitInSeconds` parameter of `HumanTaskConfig`.

When you create a labeling job in the console, you can specify this time limit when you select your workforce type and your work team.

## Task types

When you create a video object tracking or video object detection labeling job, you specify the type of annotation that you want workers to create while working on your labeling task. The annotation type determines the type of output data Ground Truth returns and defines the *task type* for your labeling job.

If you are creating a labeling job using the API operation [CreateLabelingJob](#), you specify the task type using the label category configuration file parameter `annotationType`. To learn more, see [Labeling category configuration file with label category and frame attributes reference](#).

The following task types are available for both video object tracking or video object detection labeling jobs:

- **Bounding box** – Workers are provided with tools to create bounding box annotations. A bounding box is a box that a worker draws around an objects to identify the pixel-location and label of that object in the frame.
- **Polyline** – Workers are provided with tools to create polyline annotations. A polyline is defined by the series of ordered x, y coordinates. Each point added to the polyline is connected to the previous point by a line. The polyline does not have to be closed (the start point and end point do not have to be the same) and there are no restrictions on the angles formed between lines.
- **Polygon** – Workers are provided with tools to create polygon annotations. A polygon is a closed shape defined by a series of ordered x, y coordinates. Each point added to the polygon

is connected to the previous point by a line and there are no restrictions on the angles formed between lines. Two lines (sides) of the polygon cannot cross. The start and end point of a polygon must be the same.

- **Keypoint** – Workers are provided with tools to create keypoint annotations. A keypoint is a single point associated with an x, y coordinate in the video frame.

## Workforces

When you create a video frame labeling job, you need to specify a work team to complete your annotation tasks. You can choose a work team from a private workforce of your own workers, or from a vendor workforce that you select in the AWS Marketplace. You cannot use the Amazon Mechanical Turk workforce for video frame labeling jobs.

To learn more about vendor workforces, see [Subscribe to vendor workforces](#).

To learn how to create and manage a private workforce, see [Private workforce](#).

## Worker user interface (UI)

Ground Truth provides a worker user interface (UI), tools, and assistive labeling features to help workers complete your video labeling tasks. You can preview the worker UI when you create a labeling job in the console.

When you create a labeling job using the API operation `CreateLabelingJob`, you must provide an ARN provided by Ground Truth in the parameter [HumanTaskUiArn](#) to specify the worker UI for your task type. You can use `HumanTaskUiArn` with the SageMaker AI [RenderUiTemplate](#) API operation to preview the worker UI.

You provide worker instructions, labels, and optionally, attributes that workers can use to provide more information about labels and video frames. These attributes are referred to as *label category attributes* and *frame attributes* respectively. They are all displayed in the worker UI.

## Label category and frame attributes

When you create a video object tracking or video object detection labeling job, you can add one or more *label category attributes* and *frame attributes*:

- **Label category attribute** – A list of options (strings), a free form text box, or a numeric field associated with one or more labels. It is used by workers to provide metadata about a label.

- **Frame attribute** – A list of options (strings), a free form text box, or a numeric field that appears on each video frame a worker is sent to annotate. It is used by workers to provide metadata about video frames.

Additionally, you can use label and frame attributes to have workers verify labels in a video frame label verification job.

Use the following sections to learn more about these attributes. To learn how to add label category and frame attributes to a labeling job, use the **Create Labeling Job** sections on the [task type page](#) of your choice.

### Label category attributes

Add label category attributes to labels to give workers the ability to provide more information about the annotations they create. A label category attribute is added to an individual label, or to all labels. When a label category attribute is applied to all labels it is referred to as a *global label category attribute*.

For example, if you add the label category *car*, you might also want to capture additional data about your labeled cars, such as if they are occluded or the size of the car. You can capture this metadata using label category attributes. In this example, if you added the attribute *occluded* to the car label category, you can assign *partial*, *completely*, *no* to the *occluded* attribute and enable workers to select one of these options.

When you create a label verification job, you add labels category attributes to each label you want workers to verify.

### Frame level attributes

Add frame attributes to give workers the ability to provide more information about individual video frames. Each frame attribute you add appears on all frames.

For example, you can add a number-frame attribute to have workers identify the number of objects they see in a particular frame.

In another example, you may want to provide a free-form text box to give workers the ability to provide an answer to a question.

When you create a label verification job, you can add one or more frame attributes to ask workers to provide feedback on all labels in a video frame.

## Worker instructions

You can provide worker instructions to help your workers complete your video frame labeling tasks. You might want to cover the following topics when writing your instructions:

- Best practices and things to avoid when annotating objects.
- The label category attributes provided (for object detection and object tracking tasks) and how to use them.
- How to save time while labeling by using keyboard shortcuts.

You can add your worker instructions using the SageMaker AI console while creating a labeling job. If you create a labeling job using the API operation `CreateLabelingJob`, you specify worker instructions in your label category configuration file.

In addition to your instructions, Ground Truth provides a link to help workers navigate and use the worker portal. View these instructions by selecting the task type on [Worker Instructions](#).

## Declining tasks

Workers are able to decline tasks.

Workers decline a task if the instructions are not clear, input data is not displaying correctly, or if they encounter some other issue with the task. If the number of workers per dataset object ([NumberOfHumanWorkersPerDataObject](#)) decline the task, the data object is marked as expired and will not be sent to additional workers.

## Video frame job permission requirements

When you create a video frame labeling job, in addition to the permission requirements found in [Assign IAM Permissions to Use Ground Truth](#), you must add a CORS policy to your S3 bucket that contains your input manifest file.

## CORS permission policy for your S3 bucket

When you create a video frame labeling job, you specify buckets in S3 where your input data and manifest file are located and where your output data will be stored. These buckets may be the same. You must attach the following Cross-origin resource sharing (CORS) policy to your input and output buckets. If you use the Amazon S3 console to add the policy to your bucket, you must use the JSON format.

## JSON

```
[  
  {  
    "AllowedHeaders": [  
      "*"  
    ],  
    "AllowedMethods": [  
      "GET",  
      "HEAD",  
      "PUT"  
    ],  
    "AllowedOrigins": [  
      "*"  
    ],  
    "ExposeHeaders": [  
      "Access-Control-Allow-Origin"  
    ],  
    "MaxAgeSeconds": 3000  
  }  
]
```

## XML

```
<?xml version="1.0" encoding="UTF-8"?>  
<CORSConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">  
<CORSRule>  
  <AllowedOrigin>*</AllowedOrigin>  
  <AllowedMethod>GET</AllowedMethod>  
  <AllowedMethod>HEAD</AllowedMethod>  
  <AllowedMethod>PUT</AllowedMethod>  
  <MaxAgeSeconds>3000</MaxAgeSeconds>  
  <ExposeHeader>Access-Control-Allow-Origin</ExposeHeader>  
  <AllowedHeader>*</AllowedHeader>  
</CORSRule>  
</CORSConfiguration>
```

To learn how to add a CORS policy to an S3 bucket, see [How do I add cross-domain resource sharing with CORS?](#) in the Amazon Simple Storage Service User Guide.

## Worker Instructions

This topic provides an overview of the Ground Truth worker portal and the tools available to complete your video frame labeling task. First, select the type of task you are working on from **Topics**.

### **Important**

It is recommended that you complete your task using a Google Chrome or Firefox web browser.

For adjustment jobs, select the original labeling job task type that produced the labels you are adjusting. Review and adjust the labels in your task as needed.

### Topics

- [Navigate the UI](#)
- [Bulk Edit Label and Frame Attributes](#)
- [Tool Guide](#)
- [Icons Guide](#)
- [Shortcuts](#)
- [Understand Release, Stop and Resume, and Decline Task Options](#)
- [Saving Your Work and Submitting](#)
- [Video Frame Object Tracking Tasks](#)
- [Video Frame Object Detection Tasks](#)

### **Navigate the UI**

You can navigate between video frames using the navigation bar in the bottom-left corner of your UI.

Use the play button to automatically move through the entire sequence of frames.

Use the next frame and previous frame buttons to move forward or back one frame at a time. You can also input a frame number to navigate to that frame.

You can zoom in to and out of all video frames. Once you have zoomed into a video frame, you can move around in that frame using the move icon. When you set a new view in a single video frame by zooming and moving within that frame, all video frames are set to the same view. You can reset all video frames to their original view using the fit screen icon. For additional view options, see [Icons Guide](#).

When you are in the worker UI, you see the following menus:

- **Instructions** – Review these instructions before starting your task. Additionally, select **More instructions** and review these instructions.
- **Shortcuts** – Use this menu to view keyboard shortcuts that you can use to navigate video frames and use the tools provided.
- **Help** – Use this option to refer back to this documentation.

## Bulk Edit Label and Frame Attributes

You can bulk edit label attributes and frame attributes (attributes).

When you bulk edit an attribute, you specify one or more ranges of frames that you want to apply the edit to. The attribute you select is edited in all frames in that range, including the start and end frames you specify. When you bulk edit label attributes, the range you specify *must* contain the label that the label attribute is attached to. If you specify frames that do not contain this label, you will receive an error.

To bulk edit an attribute you *must* specify the desired value for the attribute first. For example, if you want to change an attribute from *Yes* to *No*, you must select *No*, and then perform the bulk edit.

You can also specify a new value for an attribute that has not been filled in and then use the bulk edit feature to fill in that value in multiple frames. To do this, select the desired value for the attribute and complete the following procedure.

### To bulk edit a label or attribute:

1. Use your mouse to right click the attribute you want to bulk edit.
2. Specify the range of frames you want to apply the bulk edit to using a dash (-) in the text box. For example, if you want to apply the edit to frames one through ten, enter 1-10. If you want to apply the edit to frames two to five, eight to ten and twenty enter 2-5,8-10,20.
3. Select **Confirm**.

If you get an error message, verify that you entered a valid range and that the label associated with the label attribute you are editing (if applicable) exists in all frames specified.

You can quickly add a label to all previous or subsequent frames using the **Duplicate to previous frames** and **Duplicate to next frames** options in the **Label** menu at the top of your screen.

## Tool Guide

Your task will include one or more tools. The tool provided dictates the type of annotations you will create to identify and track objects. Use the following table to learn more about each tool provided.

Tool	Icon	Action	Description
Bounding box		Add a bounding box annotation.	Choose this icon to add a bounding box. Each bounding box you add is associated with the category you choose from the Label category drop down menu. Select the bounding box or its associated label to adjust it.
Predict next		Predict bounding boxes in the next frame.	Select a bounding box, and then choose this icon to predict the location of that box in the next frame. You can select the icon multiple times in a row to automatically detect the location of box in multiple frames. For example, select

Tool	Icon	Action	Description
			this icon 5 times to predict the location of a bounding box in the next 5 frames.
Keypoint		Add a keypoint annotation.	<p>Choose this icon to add a keypoint. Click on an object in the image to place the keypoint at that location.</p> <p>Each keypoint you add is associated with the category you choose from the Label category drop down menu. Select a keypoint or its associated label to adjust it.</p>

Tool	Icon	Action	Description
Polyline		<p>Add a polyline annotation.</p> <p>Each point added to the polyline is connected to the previous point by a line. The polyline does not have to be closed (the start point and end point do not have to be the same) and there are no restrictions on the angles formed between lines.</p> <p>Each polyline you add is associated with the category you choose from the Label category drop down menu. Select</p>	<p>Choose this icon to add a polyline. To add a polyline, continuously click around the object of interest to add new points. To stop drawing a polyline, select the last point that you placed a second time (this point will be green), or press <b>Enter</b> on your keyboard.</p>

Tool	Icon	Action	Description
			the polyline or its associated label to adjust it.

Tool	Icon	Action	Description
Polygon		<p>Add a polygon annotation.</p> <p>Choose this icon to add a polygon. To add a polygon, continuously click around the object of interest to add new points. To stop drawing the polygon, select the start point (this point will be green).</p> <p>A polygon is a closed shape defined by a series of points that you place. Each point added to the polygon is connected to the previous point by a line and there are no restrictions on the angles formed between lines. The start and end point must be the same.</p> <p>Each polygon you add is associated with the category you choose from the Label category drop down menu. Select the polygon or its associated label to adjust it.</p>	

Tool	Icon	Action	Description
Copy to Next		Copy annotations to the next frame.	If one or more annotations are selected in the current frame, those annotations are copied to the next frame. If no annotations are selected, all annotations in the current frame will be copied to the next frame.
Copy to All		Copy annotations to all subsequent frames.	If one or more annotations are selected in the current frame, those annotations are copied to all subsequent frames. If no annotations are selected, all annotations in the current frame will be copied to all subsequent frames.

## Icons Guide

Use this table to learn about the icons you see in your UI. You can automatically select some of these icons using the keyboard shortcuts found in the **Shortcuts** menu.

Icon	Action	Description
	brightness	Choose this icon to adjust the brightness of all video frames.
	contrast	Choose this icon to adjust the contrast of all video frames.
	zoom in	Choose this icon to zoom into all of the video frames.
	zoom out	Choose this icon to zoom out of all of the video frames.
	move screen	After you've zoomed into a video frame, choose this icon to move around in that video frame. You can move around the video frame using your mouse by clicking and dragging the frame in the direction you want it to move. This will change the view in all view frames.
	fit screen	Reset all video frames to their original position.
	undo	Undo an action. You can use this icon to remove a bounding box that you just added, or to undo an adjustment you made to a bounding box.
	redo	Redo an action that was undone using the undo icon.
	delete label	Delete a label. This will delete the bounding box associated with the label in a single frame.
	show or hide label	Select this icon to show a label that has been hidden. If this icon has a slash through it, select it to hide the label.

Icon	Action	Description
	edit label	Select this icon to open the <b>Edit instance</b> menu. Use this menu to edit a label category, ID, and to add or edit label attributes.

## Shortcuts

The keyboard shortcuts listed in the **Shortcuts** menu can help you quickly select icons, undo and redo annotations, and use tools to add and edit annotations. For example, once you add a bounding box, you can use **P** to quickly predict the location of that box in subsequent frames.

Before you start your task, it is recommended that you review the **Shortcuts** menu and become acquainted with these commands.

## Understand Release, Stop and Resume, and Decline Task Options

When you open the labeling task, three buttons on the top right allow you to decline the task (**Decline task**), release it (**Release task**), and stop and resume it at a later time (**Stop and resume later**). The following list describes what happens when you select one of these options:

- **Decline task:** You should only decline a task if something is wrong with the task, such as unclear video frame images or an issue with the UI. If you decline a task, you will not be able to return to the task.
- **Release Task:** Use this option to release a task and allow others to work on it. When you release a task, you lose all work done on that task and other workers on your team can pick it up. If enough workers pick up the task, you may not be able to return to it. When you select this button and then select **Confirm**, you are returned to the worker portal. If the task is still available, its status will be **Available**. If other workers pick it up, it will disappear from your portal.
- **Stop and resume later:** You can use the **Stop and resume later** button to stop working and return to the task at a later time. You should use the **Save** button to save your work before you select **Stop and resume later**. When you select this button and then select **Confirm**, you are returned to the worker portal, and the task status is **Stopped**. You can select the same task to resume work on it.

Be aware that the person that creates your labeling tasks specifies a time limit in which all tasks must be completed by. If you do not return to and complete this task within that time

limit, it will expire and your work will not be submitted. Contact your administrator for more information.

## Saving Your Work and Submitting

You should periodically save your work using the **Save** button. Ground Truth will automatically save your work ever 15 minutes.

When you open a task, you must complete your work on it before pressing **Submit**.

## Video Frame Object Tracking Tasks

Video frame object tracking tasks require you to track the movement of objects across video frames. A video frame is a still image from a video scene. You can use the worker UI to navigate between video frames and use the tools provided to identify unique objects and track their movement from one from to the next. Use the following topics to learn how to navigate your worker UI, use the tools provided, and complete your task.

It is recommended that you complete your task using a Google Chrome or Firefox web browser.

### **Important**

If you see annotations have already been added to one or more video frames when you open your task, adjust those annotations and add additional annotations as needed.

## Topics

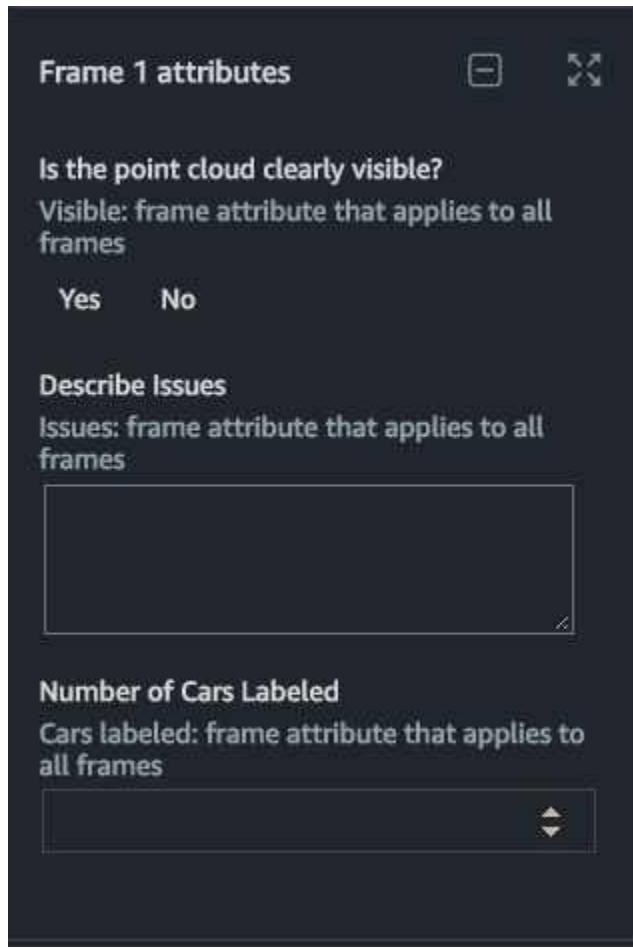
- [Your Task](#)

## Your Task

When you work on a video frame object tracking task, you need to select a category from the **Label category** menu on the right side of your worker portal to start annotating. After you've chosen a category, use the tools provided to annotate the objects that the category applies to. This annotation will be associated with a unique label ID that should only be used for that object. Use this same label ID to create additional annotations for the same object in all of the video frames that it appears in. Refer to [Tool Guide](#) to learn more about the tools provided.

After you've added a label, you may see a downward pointing arrow next to the label in the **Labels** menu. Select this arrow and then select one option for each label attribute you see to provide more information about that label.

You may see frame attributes under the **Labels** menu. These attributes will appear on each frame in your task. Use these attribute prompts to enter additional information about each frame.



After you've added a label, you can quickly add and edit a label category attribute value by using the downward pointing arrow next to the label in the **Labels** menu. If you select the pencil icon next to the label in the **Labels** menu, the **Edit instance** menu will appear. You can edit the label ID, label category, and label category attributes using this menu.

To edit an annotation, select the label of the annotation that you want to edit in the **Labels** menu or select the annotation in the frame. When you edit or delete an annotation, the action will only modify the annotation in a single frame.

If you are working on a task that includes a bounding box tool, use the predict next icon to predict the location of all bounding boxes that you have drawn in a frame in the next frame. If you select a

single box and then select the predict next icon, only that box will be predicted in the next frame. If you have not added any boxes to the current frame, you will receive an error. You must add at least one box to the frame before using this feature.

After you've used the predict next icon, review the location of each box in the next frame and make adjustments to the box location and size if necessary.

For all other tools, you can use the **Copy to next** and **Copy to all** tools to copy your annotations to the next or all frames respectively.

## Video Frame Object Detection Tasks

Video frame object detection tasks required you to classify and identify the location of objects in video frames using annotations. A video frame is a still image from a video scene. You can use the worker UI to navigate between video frames and create annotations to identify objects of interest. Use the following topics to learn how to navigate your worker UI, use the tools provided, and complete your task.

It is recommended that you complete your task using a Google Chrome web browser.

### **Important**

If you see annotations have already been added to one or more video frames when you open your task, adjust those annotations and add additional annotations as needed.

## Topics

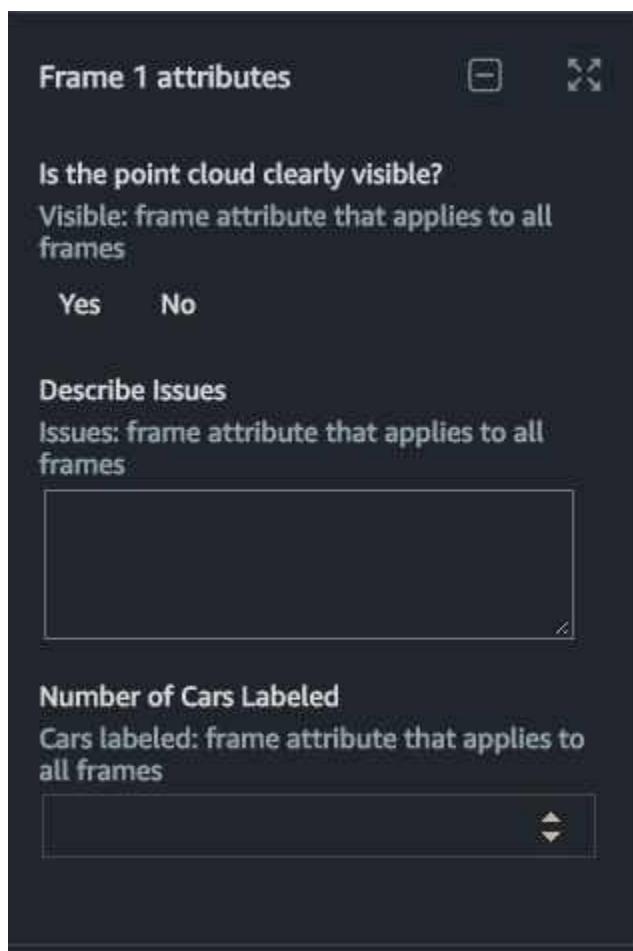
- [Your Task](#)

## Your Task

When you work on a video frame object detection task, you need to select a category from the **Label category** menu on the right side of your worker portal to start annotating. After you've chosen a category, draw annotations around objects that this category applies to. To learn more about the tools you see in your worker UI, refer to the [Tool Guide](#).

After you've added a label, you may see a downward pointing arrow next to the label in the **Labels** menu. Select this arrow and then select one option for each label attribute you see to provide more information about that label.

You may see frame attributes under the **Labels** menu. These attributes will appear on each frame in your task. Use these attribute prompts to enter additional information about each frame.



To edit an annotation, select the label of the annotation that you want to edit in the **Labels** menu or select the annotation in the frame. When you edit or delete an annotation, the action will only modify the annotation in a single frame.

If you are working on a task that includes a bounding box tool, use the predict next icon to predict the location of all bounding boxes that you have drawn in a frame in the next frame. If you select a single box and then select the predict next icon, only that box will be predicted in the next frame. If you have not added any boxes to the current frame, you will receive an error. You must add at least one box to the frame before using this feature.

**Note**

The predict next feature will not overwrite manually created annotations. It will only add annotations. If you use predict next and as a result have more than one bounding box

around a single object, delete all but one box. Each object should only be identified with a single box.

After you've used the predict next icon, review the location of each box in the next frame and make adjustments to the box location and size if necessary.

For all other tools, you can use the **Copy to next** and **Copy to all** tools to copy your annotations to the next or all frames respectively.

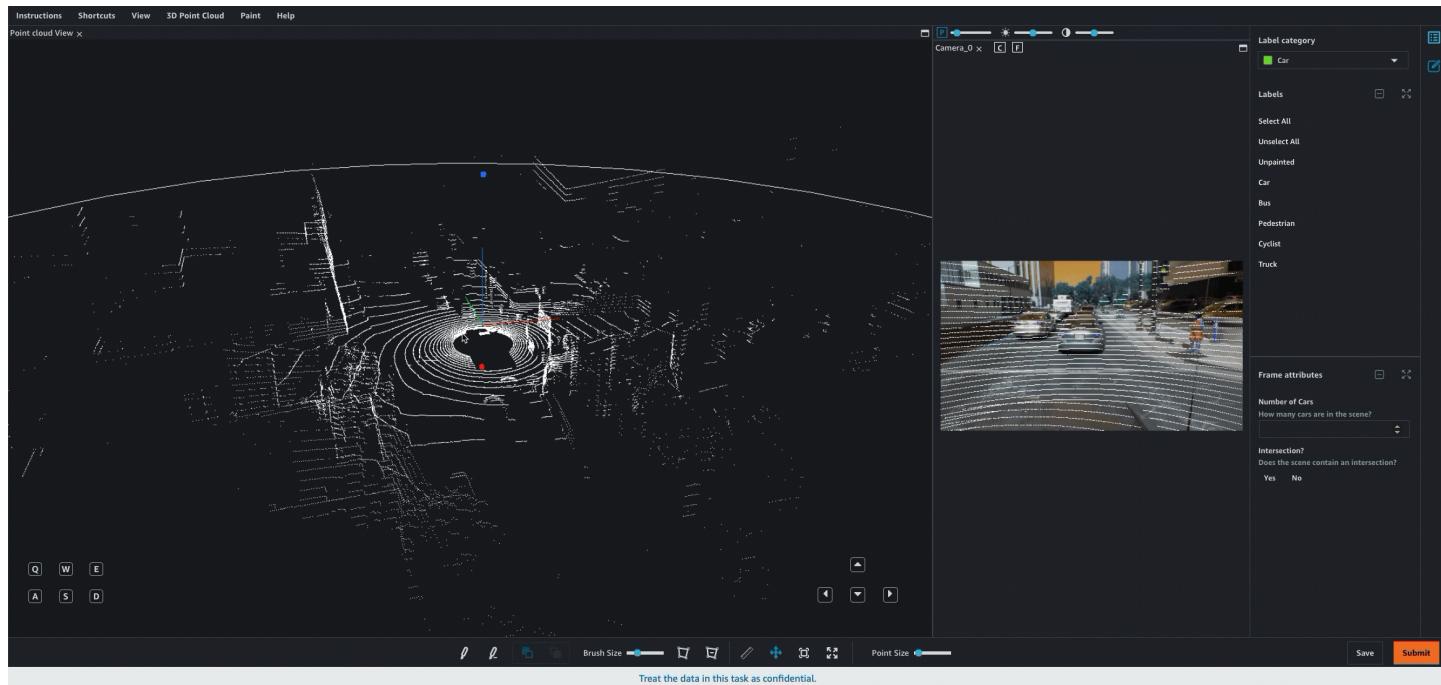
## Use Ground Truth to Label 3D Point Clouds

Create a 3D point cloud labeling job to have workers label objects in 3D point clouds generated from 3D sensors like Light Detection and Ranging (LiDAR) sensors and depth cameras, or generated from 3D reconstruction by stitching images captured by an agent like a drone.

### 3D Point Clouds

Point clouds are made up of three-dimensional (3D) visual data that consists of points. Each point is described using three coordinates, typically x, y, and z. To add color or variations in point intensity to the point cloud, points may be described with additional attributes, such as i for intensity or values for the red (r), green (g), and blue (b) 8-bit color channels. When you create a Ground Truth 3D point cloud labeling job, you can provide point cloud and, optionally, sensor fusion data.

The following image shows a single, 3D point cloud scene rendered by Ground Truth and displayed in the semantic segmentation worker UI.



## LiDAR

A Light Detection and Ranging (LiDAR) sensor is a common type of sensor used to collect measurements that are used to generate point cloud data. LiDAR is a remote sensing method that uses light in the form of a pulsed laser to measure the distances of objects from the sensor. You can provide 3D point cloud data generated from a LiDAR sensor for a Ground Truth 3D point cloud labeling job using the raw data formats described in [Accepted Raw 3D Data Formats](#).

## Sensor Fusion

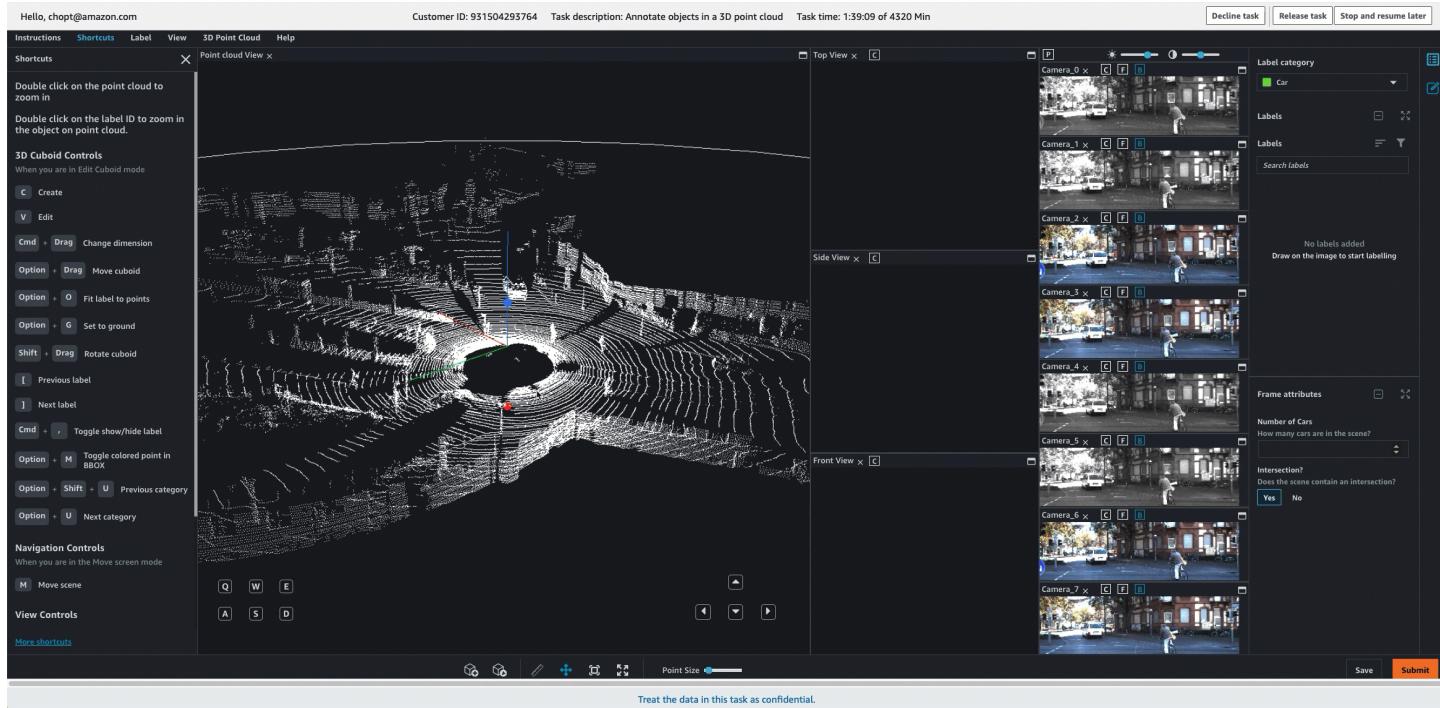
Ground Truth 3D point cloud labeling jobs include a sensor fusion feature that supports video camera sensor fusion for all task types. Some sensors come with multiple LiDAR devices and video cameras that capture images and associate them with a LiDAR frame. To help annotators visually complete your tasks with high confidence, you can use the Ground Truth sensor fusion feature to project annotations (labels) from a 3D point cloud to 2D camera images and vice versa using 3D scanner (such as LiDAR) extrinsic matrix and camera extrinsic and intrinsic matrices. To learn more, see [Sensor Fusion](#).

## Label 3D Point Clouds

Ground Truth provides a user interface (UI) and tools that workers use to label or *annotate* 3D point clouds. When you use the object detection or semantic segmentation task types, workers can

annotate a single point cloud frame. When you use object tracking, workers annotate a sequence of frames. You can use object tracking to track object movement across all frames in a sequence.

The following demonstrates how a worker would use the Ground Truth worker portal and tools to annotate a 3D point cloud for an object detection task. For similar visual examples of other task types, see [3D Point Cloud Task types](#).



## Assistive Labeling Tools for Point Cloud Annotation

Ground Truth offers assistive labeling tools to help workers complete your point cloud annotation tasks faster and with more accuracy. For details about assistive labeling tools that are included in the worker UI for each task type, [select a task type](#) and refer to the [View the Worker Task Interface](#) section of that page.

## Next Steps

You can create six types of tasks when you use Ground Truth 3D point cloud labeling jobs. Use the topics in [3D Point Cloud Task types](#) to learn more about these *task types* and to learn how to create a labeling job using the task type of your choice.

The 3D point cloud labeling job is different from other Ground Truth labeling modalities. Before creating a labeling job, we recommend that you read [3D point cloud labeling jobs overview](#). Additionally, review input data quotas in [3D Point Cloud and Video Frame Labeling Job Quotas](#).

For an end-to-end demo using the SageMaker API and AWS Python SDK (boto 3) to create a 3D point cloud labeling job, see [create-3D-pointcloud-labeling-job.ipynb](#) in the [SageMaker AI Examples notebook tab](#).

### **Important**

If you use a notebook instance created before June 5th, 2020 to run this notebook, you must stop and restart that notebook instance for the notebook to work.

## Topics

- [3D Point Cloud Task types](#)
- [3D point cloud labeling jobs overview](#)
- [Worker instructions](#)

## 3D Point Cloud Task types

You can use Ground Truth 3D point cloud labeling modality for a variety of use cases. The following list briefly describes each 3D point cloud task type. For additional details and instructions on how to create a labeling job using a specific task type, select the task type name to see its task type page.

- [3D point cloud object detection](#) – Use this task type when you want workers to locate and classify objects in a 3D point cloud by adding and fitting 3D cuboids around objects.
- [3D point cloud object tracking](#) – Use this task type when you want workers to add and fit 3D cuboids around objects to track their movement across a sequence of 3D point cloud frames. For example, you can use this task type to ask workers to track the movement of vehicles across multiple point cloud frames.
- [3D point cloud semantic segmentation](#) – Use this task type when you want workers to create a point-level semantic segmentation mask by painting objects in a 3D point cloud using different colors where each color is assigned to one of the classes you specify.
- 3D point cloud adjustment task types – Each of the task types above has an associated *adjustment* task type that you can use to audit and adjust annotations generated from a 3D point cloud labeling job. Refer to the task type page of the associated type to learn how to create an adjustment labeling job for that task.

## Classify objects in a 3D point cloud with object detection

Use this task type when you want workers to classify objects in a 3D point cloud by drawing 3D cuboids around objects. For example, you can use this task type to ask workers to identify different types of objects in a point cloud, such as cars, bikes, and pedestrians. The following page gives important information about the labeling job, as well as steps to create one.

For this task type, the *data object* that workers label is a single point cloud frame. Ground Truth renders a 3D point cloud using point cloud data you provide. You can also provide camera data to give workers more visual information about scenes in the frame, and to help workers draw 3D cuboids around objects.

Ground Truth provides workers with tools to annotate objects with 9 degrees of freedom ( $x,y,z,rx,ry,rz,l,w,h$ ) in three dimensions in both 3D scene and projected side views (top, side, and back). If you provide sensor fusion information (like camera data), when a worker adds a cuboid to identify an object in the 3D point cloud, the cuboid shows up and can be modified in the 2D images. After a cuboid has been added, all edits made to that cuboid in the 2D or 3D scene are projected into the other view.

You can create a job to adjust annotations created in a 3D point cloud object detection labeling job using the 3D point cloud object detection adjustment task type.

If you are a new user of the Ground Truth 3D point cloud labeling modality, we recommend you review [3D point cloud labeling jobs overview](#). This labeling modality is different from other Ground Truth task types, and this page provides an overview of important details you should be aware of when creating a 3D point cloud labeling job.

### Topics

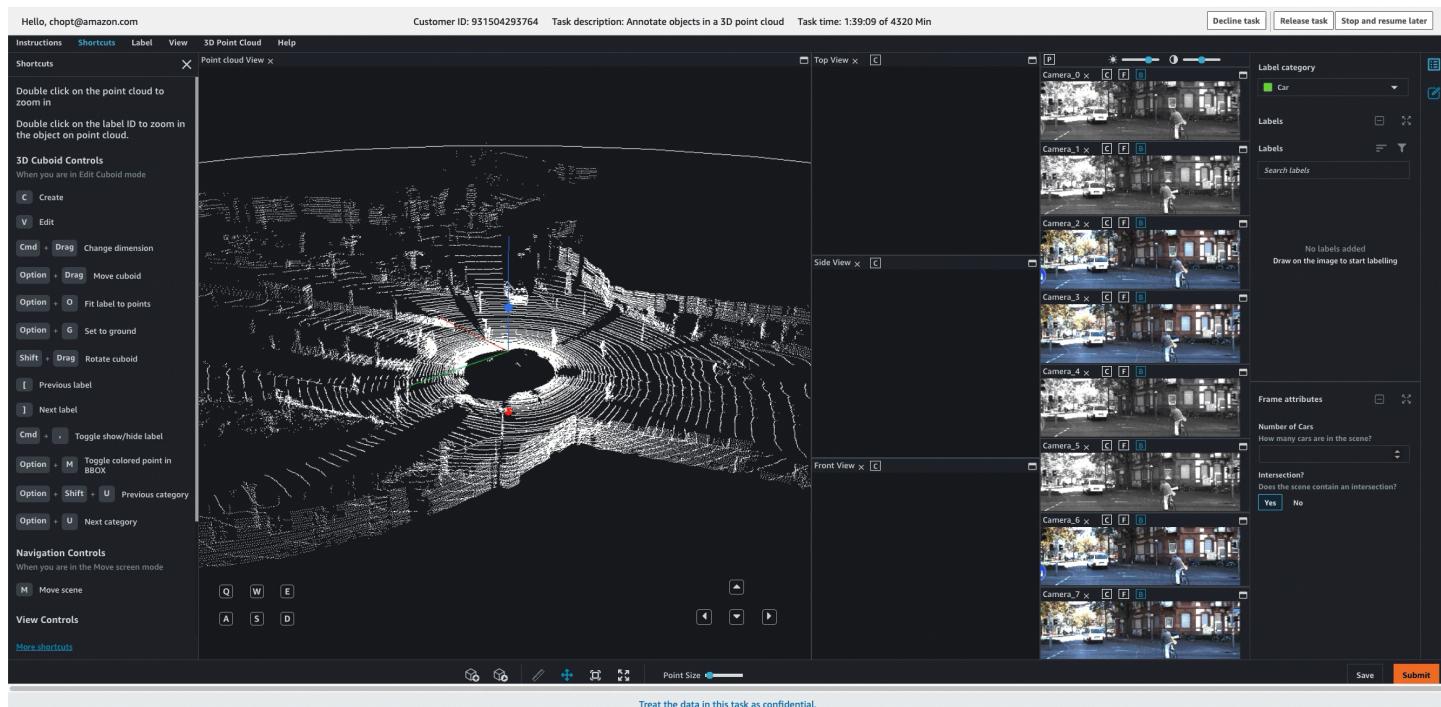
- [View the Worker Task Interface](#)
- [Create a 3D Point Cloud Object Detection Labeling Job](#)
- [Create a 3D Point Cloud Object Detection Adjustment or Verification Labeling Job](#)
- [Output Data Format](#)

### View the Worker Task Interface

Ground Truth provides workers with a web portal and tools to complete your 3D point cloud object detection annotation tasks. When you create the labeling job, you provide the Amazon Resource Name (ARN) for a pre-built Ground Truth worker UI in the `HumanTaskUiArn` parameter. When you

create a labeling job using this task type in the console, this worker UI is automatically used. You can preview and interact with the worker UI when you create a labeling job in the console. If you are a new user, it is recommended that you create a labeling job using the console to ensure your label attributes, point cloud frames, and if applicable, images, appear as expected.

The following is a GIF of the 3D point cloud object detection worker task interface. If you provide camera data for sensor fusion in the world coordinate system, images are matched up with scenes in the point cloud frame. These images appear in the worker portal as shown in the following GIF.

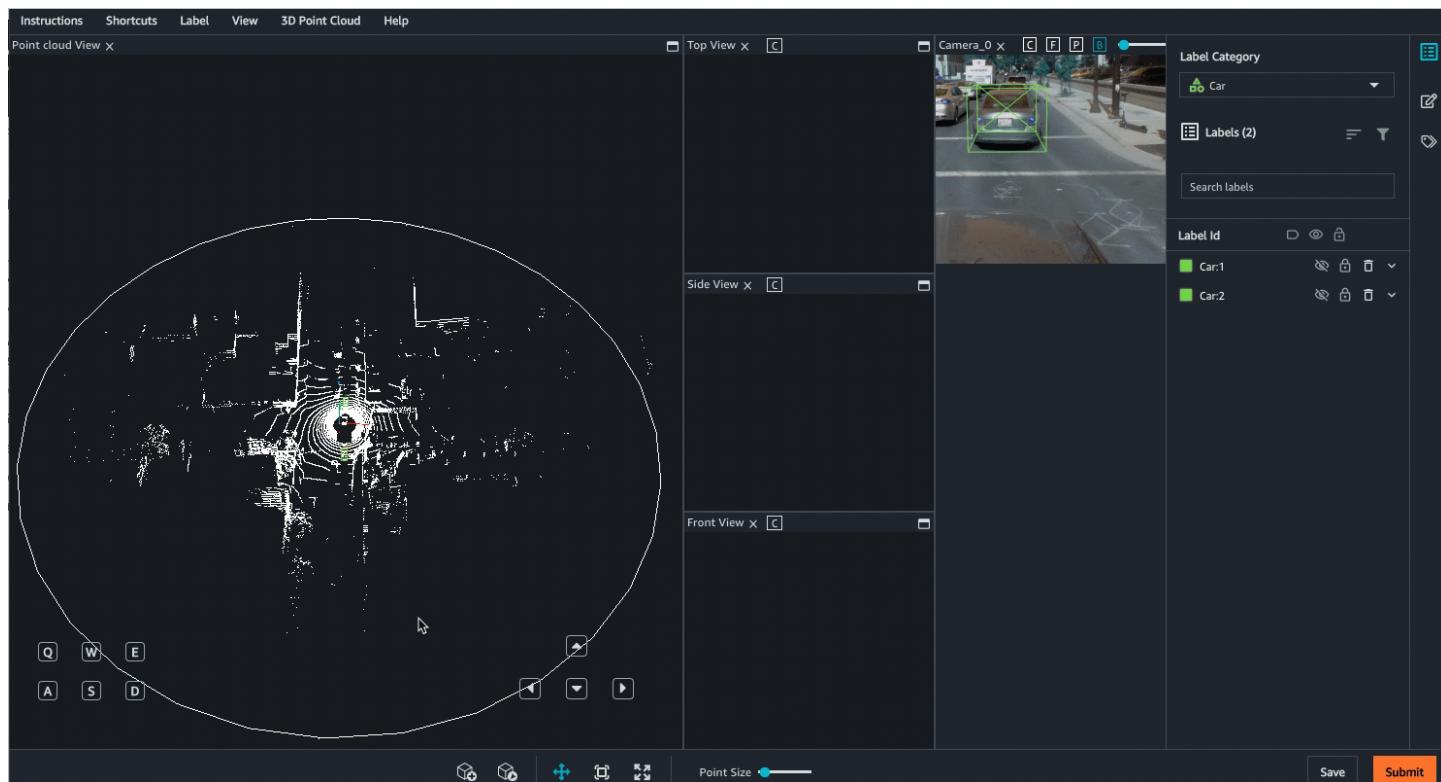


Worker can navigate in the 3D scene using their keyboard and mouse. They can:

- Double click on specific objects in the point cloud to zoom into them.
- Use a mouse-scroller or trackpad to zoom in and out of the point cloud.
- Use both keyboard arrow keys and Q, E, A, and D keys to move Up, Down, Left, Right. Use keyboard keys W and S to zoom in and out.

Once a worker places a cuboid in the 3D scene, a side-view will appear with the three projected side views: top, side, and back. These side-views show points in and around the placed cuboid and help workers refine cuboid boundaries in that area. Workers can zoom in and out of each of those side-views using their mouse.

The following video demonstrates movements around the 3D point cloud and in the side-view.



Additional view options and features are available in the **View** menu in the worker UI. See the [worker instruction page](#) for a comprehensive overview of the Worker UI.

## Assistive Labeling Tools

Ground Truth helps workers annotate 3D point clouds faster and more accurately using machine learning and computer vision powered assistive labeling tools for 3D point cloud object tracking tasks. The following assistive labeling tools are available for this task type:

- **Snapping** – Workers can add a cuboid around an object and use a keyboard shortcut or menu option to have Ground Truth's autofit tool snap the cuboid tightly around the object.
- **Set to ground** – After a worker adds a cuboid to the 3D scene, the worker can automatically snap the cuboid to the ground. For example, the worker can use this feature to snap a cuboid to the road or sidewalk in the scene.
- **Multi-view labeling** – After a worker adds a 3D cuboid to the 3D scene, a side panel displays front, side, and top perspectives to help the worker adjust the cuboid tightly around the object. In all of these views, the cuboid includes an arrow that indicates the orientation, or heading of the object. When the worker adjusts the cuboid, the adjustment will appear in real time on all of the views (that is, 3D, top, side, and front).

- **Sensor fusion** – If you provide data for sensor fusion, workers can adjust annotations in the 3D scenes and in 2D images, and the annotations will be projected into the other view in real time. Additionally, workers will have the option to view the direction the camera is facing and the camera frustum.
- **View options** – Enables workers to easily hide or view cuboids, label text, a ground mesh, and additional point attributes like color or intensity. Workers can also choose between perspective and orthogonal projections.

## Create a 3D Point Cloud Object Detection Labeling Job

You can create a 3D point cloud labeling job using the SageMaker AI console or API operation, [CreateLabelingJob](#). To create a labeling job for this task type you need the following:

- A single-frame input manifest file. To learn how to create this type of manifest file, see [Create a Point Cloud Frame Input Manifest File](#). If you are a new user of Ground Truth 3D point cloud labeling modalities, you may also want to review [Accepted Raw 3D Data Formats](#).
- A work team from a private or vendor workforce. You cannot use Amazon Mechanical Turk for video frame labeling jobs. To learn how to create workforces and work teams, see [Workforces](#).

Additionally, make sure that you have reviewed and satisfied the [Assign IAM Permissions to Use Ground Truth](#).

Use one of the following sections to learn how to create a labeling job using the console or an API.

### Create a Labeling Job (Console)

You can follow the instructions [Create a Labeling Job \(Console\)](#) in order to learn how to create a 3D point cloud object detection labeling job in the SageMaker AI console. While you are creating your labeling job, be aware of the following:

- Your input manifest file must be a single-frame manifest file. For more information, see [Create a Point Cloud Frame Input Manifest File](#).
- Optionally, you can provide label category and frame attributes. Workers can assign one or more of these attributes to annotations to provide more information about that object. For example, you might want to use the attribute *occluded* to have workers identify when an object is partially obstructed.
- Automated data labeling and annotation consolidation are not supported for 3D point cloud labeling tasks.

- 3D point cloud object detection labeling jobs can take multiple hours to complete. You can specify a longer time limit for these labeling jobs when you select your work team (up to 7 days, or 604800 seconds).

## Create a Labeling Job (API)

This section covers details you need to know when you create a labeling job using the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

[Create a Labeling Job \(API\)](#), provides an overview of the `CreateLabelingJob` operation. Follow these instructions and do the following while you configure your request:

- You must enter an ARN for `HumanTaskUiArn`. Use `arn:aws:sagemaker:<region>:394669845002:human-task-ui/PointCloudObjectDetection`. Replace `<region>` with the AWS Region you are creating the labeling job in.

There should not be an entry for the `UiTemplateS3Uri` parameter.

- Your input manifest file must be a single-frame manifest file. For more information, see [Create a Point Cloud Frame Input Manifest File](#).
- You specify your labels, label category and frame attributes, and worker instructions in a label category configuration file. To learn how to create this file, see [Labeling category configuration file with label category and frame attributes reference](#).
- You need to provide pre-defined ARNs for the pre-annotation and post-annotation (ACS) Lambda functions. These ARNs are specific to the AWS Region you use to create your labeling job.
  - To find the pre-annotation Lambda ARN, refer to [PreHumanTaskLambdaArn](#). Use the Region you are creating your labeling job in to find the correct ARN. For example, if you are creating your labeling job in us-east-1, the ARN will be `arn:aws:lambda:us-east-1:432418664414:function:PRE-3DPointCloudObjectDetection`.
  - To find the post-annotation Lambda ARN, refer to [AnnotationConsolidationLambdaArn](#). Use the Region you are creating your labeling job in to find the correct ARN. For example, if you are creating your labeling job in us-east-1, the ARN will be `arn:aws:lambda:us-east-1:432418664414:function:ACS-3DPointCloudObjectDetection`.

- The number of workers specified in `NumberOfHumanWorkersPerDataObject` must be 1.
- Automated data labeling is not supported for 3D point cloud labeling jobs. You should not specify values for parameters in [LabelingJobAlgorithmsConfig](#).
- 3D point cloud object detection labeling jobs can take multiple hours to complete. You can specify a longer time limit for these labeling jobs in `TaskTimeLimitInSeconds` (up to 7 days, or 604,800 seconds).

## Create a 3D Point Cloud Object Detection Adjustment or Verification Labeling Job

You can create an adjustment or verification labeling job using the Ground Truth console or `CreateLabelingJob` API. To learn more about adjustment and verification labeling jobs, and to learn how to create one, see [Label verification and adjustment](#).

When you create an adjustment labeling job, your input data to the labeling job can include labels, and yaw, pitch, and roll measurements from a previous labeling job or external source. In the adjustment job, pitch, and roll will be visualized in the worker UI, but cannot be modified. Yaw is adjustable.

Ground Truth uses Tait-Bryan angles with the following intrinsic rotations to visualize yaw, pitch and roll in the worker UI. First, rotation is applied to the vehicle according to the z-axis (yaw). Next, the rotated vehicle is rotated according to the intrinsic y'-axis (pitch). Finally, the vehicle is rotated according to the intrinsic x''-axis (roll).

## Output Data Format

When you create a 3D point cloud object detection labeling job, tasks are sent to workers. When these workers complete their tasks, labels are written to the Amazon S3 bucket you specified when you created the labeling job. The output data format determines what you see in your Amazon S3 bucket when your labeling job status ([LabelingJobStatus](#)) is Completed.

If you are a new user of Ground Truth, see [Labeling job output data](#) to learn more about the Ground Truth output data format. To learn about the 3D point cloud object detection output data format, see [3D point cloud object detection output](#).

## Understand the 3D point cloud object tracking task type

Use this task type when you want workers to add and fit 3D cuboids around objects to track their movement across 3D point cloud frames. For example, you can use this task type to ask workers to track the movement of vehicles across multiple point cloud frames.

For this task type, the data object that workers label is a sequence of point cloud frames. A *sequence* is defined as a temporal series of point cloud frames. Ground Truth renders a series of 3D point cloud visualizations using a sequence you provide and workers can switch between these 3D point cloud frames in the worker task interface.

Ground Truth provides workers with tools to annotate objects with 9 degrees of freedom ( $x,y,z,rx,ry,rz,l,w,h$ ) in three dimensions in both 3D scene and projected side views (top, side, and back). When a worker draws a cuboid around an object, that cuboid is given a unique ID, for example `Car:1` for one car in the sequence and `Car:2` for another. Workers use that ID to label the same object in multiple frames.

You can also provide camera data to give workers more visual information about scenes in the frame, and to help workers draw 3D cuboids around objects. When a worker adds a 3D cuboid to identify an object in either the 2D image or the 3D point cloud, and the cuboid shows up in the other view.

You can adjust annotations created in a 3D point cloud object detection labeling job using the 3D point cloud object tracking adjustment task type.

If you are a new user of the Ground Truth 3D point cloud labeling modality, we recommend you review [3D point cloud labeling jobs overview](#). This labeling modality is different from other Ground Truth task types, and this page provides an overview of important details you should be aware of when creating a 3D point cloud labeling job.

The following topics explain how to create a 3D point cloud object tracking job, show what the worker task interface looks like (what workers see when they work on this task), and provide an overview of the output data you get when workers complete their tasks. The final topic provides useful information for creating object tracking adjustment or verification labeling jobs.

## Topics

- [Create a 3D point cloud object tracking labeling job](#)
- [View the worker task interface for a 3D point cloud object tracking task](#)
- [Output data for a 3D point cloud object tracking labeling job](#)
- [Information for creating a 3D point cloud object tracking adjustment or verification labeling job](#)

## Create a 3D point cloud object tracking labeling job

You can create a 3D point cloud labeling job using the SageMaker AI console or API operation, [CreateLabelingJob](#). To create a labeling job for this task type you need the following:

- A sequence input manifest file. To learn how to create this type of manifest file, see [Create a Point Cloud Sequence Input Manifest](#). If you are a new user of Ground Truth 3D point cloud labeling modalities, we recommend that you review [Accepted Raw 3D Data Formats](#).
- A work team from a private or vendor workforce. You cannot use Amazon Mechanical Turk for 3D point cloud labeling jobs. To learn how to create workforces and work teams, see [Workforces](#).

Additionally, make sure that you have reviewed and satisfied the [Assign IAM Permissions to Use Ground Truth](#).

To learn how to create a labeling job using the console or an API, see the following sections.

### Create a labeling job (console)

You can follow the instructions [Create a Labeling Job \(Console\)](#) in order to learn how to create a 3D point cloud object tracking labeling job in the SageMaker AI console. While you are creating your labeling job, be aware of the following:

- Your input manifest file must be a sequence manifest file. For more information, see [Create a Point Cloud Sequence Input Manifest](#).
- Optionally, you can provide label category attributes. Workers can assign one or more of these attributes to annotations to provide more information about that object. For example, you might want to use the attribute *occluded* to have workers identify when an object is partially obstructed.
- Automated data labeling and annotation consolidation are not supported for 3D point cloud labeling tasks.
- 3D point cloud object tracking labeling jobs can take multiple hours to complete. You can specify a longer time limit for these labeling jobs when you select your work team (up to 7 days, or 604800 seconds).

### Create a labeling job (API)

This section covers details you need to know when you create a labeling job using the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see

a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

[Create a Labeling Job \(API\)](#) provides an overview of the CreateLabelingJob operation. Follow these instructions and do the following while you configure your request:

- You must enter an ARN for HumanTaskUiArn. Use  
`arn:aws:sagemaker:<region>:394669845002:human-task-ui/PointCloudObjectTracking`. Replace `<region>` with the AWS Region you are creating the labeling job in.

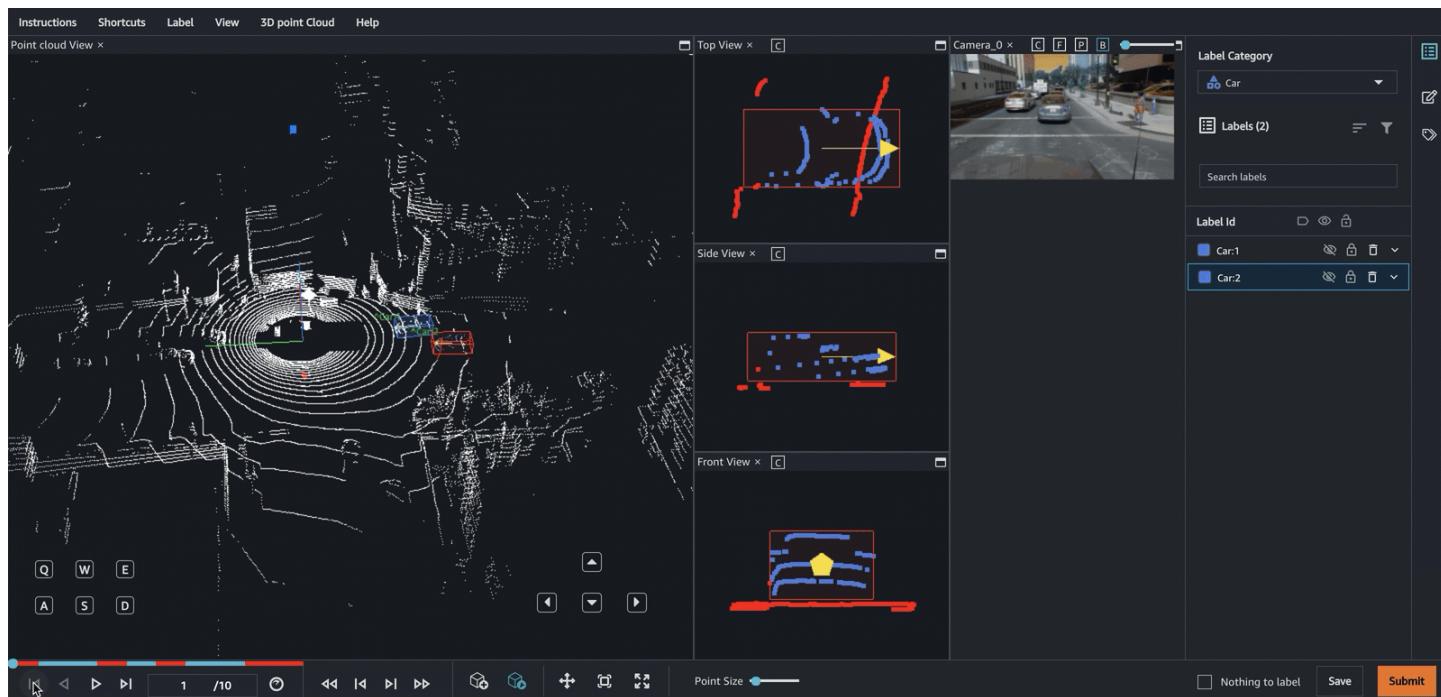
There should not be an entry for the UiTemplateS3Uri parameter.

- Your [LabelAttributeName](#) must end in `-ref`. For example, `ot-labels-ref`.
- Your input manifest file must be a point cloud frame sequence manifest file. For more information, see [Create a Point Cloud Sequence Input Manifest](#).
- You specify your labels, label category and frame attributes, and worker instructions in a label category configuration file. For more information, see [Labeling category configuration file with label category and frame attributes reference](#) to learn how to create this file.
- You need to provide pre-defined ARNs for the pre-annotation and post-annotation (ACS) Lambda functions. These ARNs are specific to the AWS Region you use to create your labeling job.
  - To find the pre-annotation Lambda ARN, refer to [PreHumanTaskLambdaArn](#). Use the Region you are creating your labeling job in to find the correct ARN that ends with `PRE-3DPointCloudObjectTracking`.
  - To find the post-annotation Lambda ARN, refer to [AnnotationConsolidationLambdaArn](#). Use the Region you are creating your labeling job in to find the correct ARN that ends with `ACS-3DPointCloudObjectTracking`.
- The number of workers specified in `NumberOfHumanWorkersPerDataObject` should be 1.
- Automated data labeling is not supported for 3D point cloud labeling jobs. You should not specify values for parameters in [LabelingJobAlgorithmsConfig](#).
- 3D point cloud object tracking labeling jobs can take multiple hours to complete. You can specify a longer time limit for these labeling jobs in `TaskTimeLimitInSeconds` (up to 7 days, or 604,800 seconds).

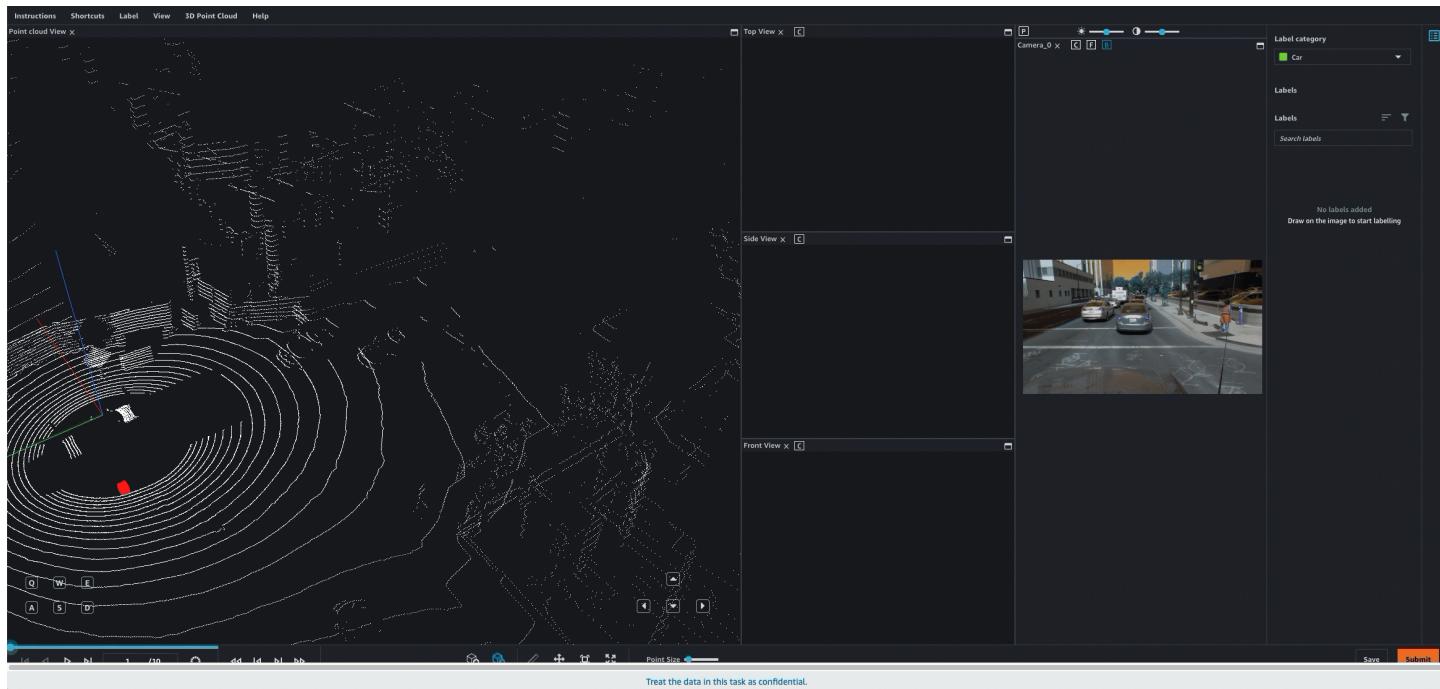
## View the worker task interface for a 3D point cloud object tracking task

Ground Truth provides workers with a web portal and tools to complete your 3D point cloud object tracking annotation tasks. When you create the labeling job, you provide the Amazon Resource Name (ARN) for a pre-built Ground Truth UI in the HumanTaskUiArn parameter. When you create a labeling job using this task type in the console, this UI is automatically used. You can preview and interact with the worker UI when you create a labeling job in the console. If you are a new user, it is recommended that you create a labeling job using the console to ensure your label attributes, point cloud frames, and if applicable, images, appear as expected.

The following is a GIF of the 3D point cloud object tracking worker task interface and demonstrates how the worker can navigate the point cloud frames in the sequence. The annotating tools are a part of the worker task interface. They are not available for the preview interface.



Once workers add a single cuboid, that cuboid is replicated in all frames of the sequence with the same ID. Once workers adjust the cuboid in another frame, Ground Truth will interpolate the movement of that object and adjust all cuboids between the manually adjusted frames. The following GIF demonstrates this interpolation feature. In the navigation bar on the bottom-left, red-areas indicate manually adjusted frames.



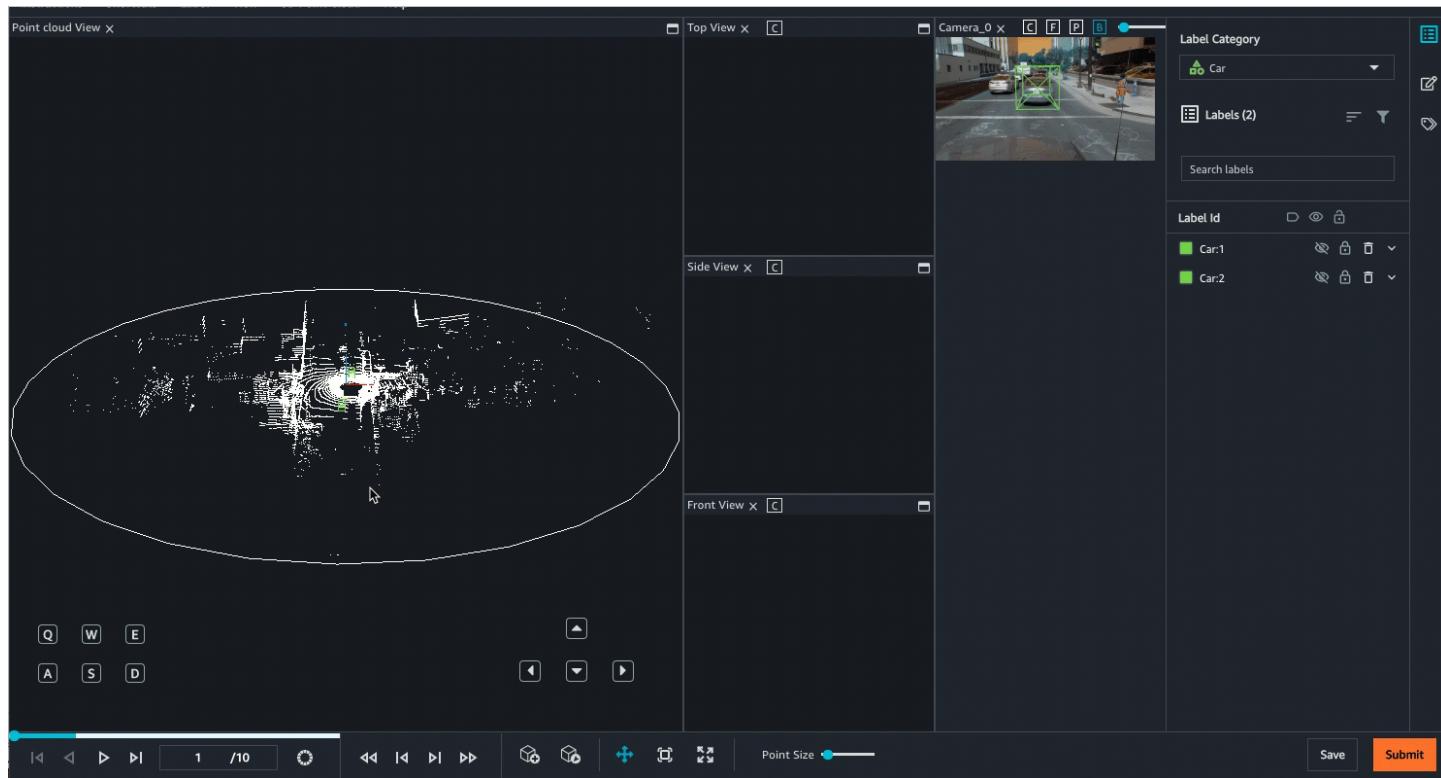
If you provide camera data for sensor fusion, images are matched up with scenes in point cloud frames. These images appear in the worker portal as shown in the following GIF.

Worker can navigate in the 3D scene using their keyboard and mouse. They can:

- Double click on specific objects in the point cloud to zoom into them.
- Use a mouse-scroller or trackpad to zoom in and out of the point cloud.
- Use both keyboard arrow keys and Q, E, A, and D keys to move Up, Down, Left, Right. Use keyboard keys W and S to zoom in and out.

Once a worker places a cuboids in the 3D scene, a side-view will appear with the three projected side views: top, side, and back. These side-views show points in and around the placed cuboid and help workers refine cuboid boundaries in that area. Workers can zoom in and out of each of those side-views using their mouse.

The following video demonstrates movements around the 3D point cloud and in the side-view.



Additional view options and features are available. See the [worker instruction page](#) for a comprehensive overview of the Worker UI.

## Worker tools

Workers can navigate through the 3D point cloud by zooming in and out, and moving in all directions around the cloud using the mouse and keyboard shortcuts. If workers click on a point in the point cloud, the UI will automatically zoom into that area. Workers can use various tools to draw 3D cuboid around objects. For more information, see [Assistive Labeling Tools](#).

After workers have placed a 3D cuboid in the point cloud, they can adjust these cuboids to fit tightly around cars using a variety of views: directly in the 3D cuboid, in a side-view featuring three zoomed-in perspectives of the point cloud around the box, and if you include images for sensor fusion, directly in the 2D image.

View options that enable workers to easily hide or view label text, a ground mesh, and additional point attributes. Workers can also choose between perspective and orthogonal projections.

## Assistive Labeling Tools

Ground Truth helps workers annotate 3D point clouds faster and more accurately using UX, machine learning and computer vision powered assistive labeling tools for 3D point cloud object tracking tasks. The following assistive labeling tools are available for this task type:

- **Label autofill** – When a worker adds a cuboid to a frame, a cuboid with the same dimensions and orientation is automatically added to all frames in the sequence.
- **Label interpolation** – After a worker has labeled a single object in two frames, Ground Truth uses those annotations to interpolate the movement of that object between those two frames. Label interpolation can be turned on and off.
- **Bulk label and attribute management** – Workers can add, delete, and rename annotations, label category attributes, and frame attributes in bulk.
  - Workers can manually delete annotations for a given object before or after a frame. For example, a worker can delete all labels for an object after frame 10 if that object is no longer located in the scene after that frame.
  - If a worker accidentally bulk deletes all annotations for an object, they can add them back. For example, if a worker deletes all annotations for an object before frame 100, they can bulk add them to those frames.
  - Workers can rename a label in one frame and all 3D cuboids assigned that label are updated with the new name across all frames.
  - Workers can use bulk editing to add or edit label category attributes and frame attributes in multiple frames.
- **Snapping** – Workers can add a cuboid around an object and use a keyboard shortcut or menu option to have Ground Truth's autofit tool snap the cuboid tightly around the object's boundaries.
- **Fit to ground** – After a worker adds a cuboid to the 3D scene, the worker can automatically snap the cuboid to the ground. For example, the worker can use this feature to snap a cuboid to the road or sidewalk in the scene.
- **Multi-view labeling** – After a worker adds a 3D cuboid to the 3D scene, a side -panel displays front and two side perspectives to help the worker adjust the cuboid tightly around the object. Workers can annotation the 3D point cloud, the side panel and the adjustments appear in the other views in real time.
- **Sensor fusion** – If you provide data for sensor fusion, workers can adjust annotations in the 3D scenes and in 2D images, and the annotations will be projected into the other view in real time.
- **Auto-merge cuboids** – Workers can automatically merge two cuboids across all frames if they determine that cuboids with different labels actually represent a single object.

- **View options** – Enables workers to easily hide or view label text, a ground mesh, and additional point attributes like color or intensity. Workers can also choose between perspective and orthogonal projections.

## Output data for a 3D point cloud object tracking labeling job

When you create a 3D point cloud object tracking labeling job, tasks are sent to workers. When these workers complete their tasks, their annotations are written to the Amazon S3 bucket you specified when you created the labeling job. The output data format determines what you see in your Amazon S3 bucket when your labeling job status ([LabelingJobStatus](#)) is Completed.

If you are a new user of Ground Truth, see [Labeling job output data](#) to learn more about the Ground Truth output data format. To learn about the 3D point cloud object tracking output data format, see [3D point cloud object tracking output](#).

## Information for creating a 3D point cloud object tracking adjustment or verification labeling job

You can create an adjustment and verification labeling job using the Ground Truth console or CreateLabelingJob API. To learn more about adjustment and verification labeling jobs, and to learn how to create one, see [Label verification and adjustment](#).

When you create an adjustment labeling job, your input data to the labeling job can include labels, and yaw, pitch, and roll measurements from a previous labeling job or external source. In the adjustment job, pitch, and roll will be visualized in the worker UI, but cannot be modified. Yaw is adjustable.

Ground Truth uses Tait-Bryan angles with the following intrinsic rotations to visualize yaw, pitch and roll in the worker UI. First, rotation is applied to the vehicle according to the z-axis (yaw). Next, the rotated vehicle is rotated according to the intrinsic y'-axis (pitch). Finally, the vehicle is rotated according to the intrinsic x''-axis (roll).

## Understand the 3D point cloud semantic segmentation task type

Semantic segmentation involves classifying individual points of a 3D point cloud into pre-specified categories. Use this task type when you want workers to create a point-level semantic segmentation mask for 3D point clouds. For example, if you specify the classes car, pedestrian, and bike, workers select one class at a time, and color all of the points that this class applies to the same color in the point cloud.

For this task type, the data object that workers label is a single point cloud frame. Ground Truth generates a 3D point cloud visualization using point cloud data you provide. You can also provide camera data to give workers more visual information about scenes in the frame, and to help workers paint objects. When a worker paints an object in either the 2D image or the 3D point cloud, the paint shows up in the other view.

You can also adjust or verify annotations created in a 3D point cloud object detection labeling job using the 3D point cloud semantic segmentation adjustment or labeling task type. To learn more about adjustment and verification labeling jobs, and to learn how to create one, see [Label verification and adjustment](#).

If you are a new user of the Ground Truth 3D point cloud labeling modality, we recommend you review [3D point cloud labeling jobs overview](#). This labeling modality is different from other Ground Truth task types, and this topic provides an overview of important details you should be aware of when creating a 3D point cloud labeling job.

The following topics explain how to create a 3D point cloud semantic segmentation job, show what the worker task interface looks like (what workers see when they work on this task), and provide an overview of the output data you get when workers complete their tasks.

## Topics

- [Create a 3D point cloud semantic segmentation labeling job](#)
- [View the worker task interface for a 3D point cloud semantic segmentation job](#)
- [Output data for a 3D point cloud semantic segmentation job](#)

## Create a 3D point cloud semantic segmentation labeling job

You can create a 3D point cloud labeling job using the SageMaker AI console or API operation, [CreateLabelingJob](#). To create a labeling job for this task type you need the following:

- A single-frame input manifest file. To learn how to create this type of manifest file, see [Create a Point Cloud Frame Input Manifest File](#). If you are a new user of Ground Truth 3D point cloud labeling modalities, we recommend that you review [Accepted Raw 3D Data Formats](#).
- A work team from a private or vendor workforce. You cannot use Amazon Mechanical Turk workers for 3D point cloud labeling jobs. To learn how to create workforces and work teams, see [Workforces](#).
- A label category configuration file. For more information, see [Labeling category configuration file with label category and frame attributes reference](#).

Additionally, make sure that you have reviewed and satisfied the [Assign IAM Permissions to Use Ground Truth](#).

Use one of the following sections to learn how to create a labeling job using the console or an API.

## Create a labeling job (console)

You can follow the instructions [Create a Labeling Job \(Console\)](#) in order to learn how to create a 3D point cloud semantic segmentation labeling job in the SageMaker AI console. While you are creating your labeling job, be aware of the following:

- Your input manifest file must be a single-frame manifest file. For more information, see [Create a Point Cloud Frame Input Manifest File](#).
- Automated data labeling and annotation consolidation are not supported for 3D point cloud labeling tasks.
- 3D point cloud semantic segmentation labeling jobs can take multiple hours to complete. You can specify a longer time limit for these labeling jobs when you select your work team (up to 7 days, or 604800 seconds).

## Create a labeling job (API)

This section covers details you need to know when you create a labeling job using the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

The page, [Create a Labeling Job \(API\)](#), provides an overview of the `CreateLabelingJob` operation. Follow these instructions and do the following while you configure your request:

- You must enter an ARN for `HumanTaskUiArn`. Use  
`arn:aws:sagemaker:<region>:394669845002:human-task-ui/PointCloudSemanticSegmentation`. Replace `<region>` with the AWS Region you are creating the labeling job in.

There should not be an entry for the `UiTemplateS3Uri` parameter.

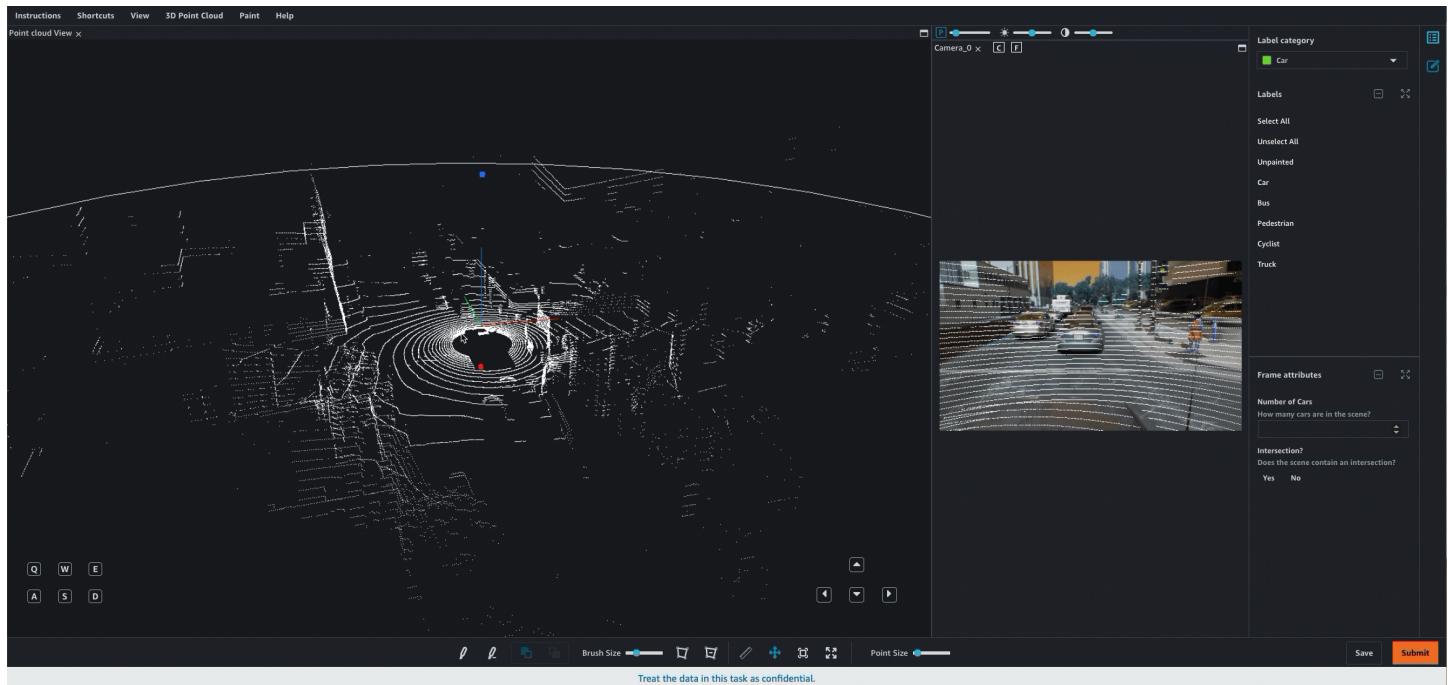
- Your `LabelAttributeName` must end in `-ref`. For example, `ss-labels-ref`.
- Your input manifest file must be a single-frame manifest file. For more information, see [Create a Point Cloud Frame Input Manifest File](#).

- You specify your labels and worker instructions in a label category configuration file. See [Labeling category configuration file with label category and frame attributes reference](#) to learn how to create this file.
- You need to provide a pre-defined ARNs for the pre-annotation and post-annotation (ACS) Lambda functions. These ARNs are specific to the AWS Region you use to create your labeling job.
  - To find the pre-annotation Lambda ARN, refer to [PreHumanTaskLambdaArn](#). Use the Region you are creating your labeling job in to find the correct ARN. For example, if you are creating your labeling job in us-east-1, the ARN will be `arn:aws:lambda:us-east-1:432418664414:function:PRE-3DPointCloudSemanticSegmentation`.
  - To find the post-annotation Lambda ARN, refer to [AnnotationConsolidationLambdaArn](#). Use the Region you are creating your labeling job in to find the correct ARN. For example, if you are creating your labeling job in us-east-1, the ARN will be `arn:aws:lambda:us-east-1:432418664414:function:ACS-3DPointCloudSemanticSegmentation`.
- The number of workers specified in `NumberOfHumanWorkersPerDataObject` should be 1.
- Automated data labeling is not supported for 3D point cloud labeling jobs. You should not specify values for parameters in [LabelingJobAlgorithmsConfig](#).
- 3D point cloud semantic segmentation labeling jobs can take multiple hours to complete. You can specify a longer time limit for these labeling jobs in `TaskTimeLimitInSeconds` (up to 7 days, or 604800 seconds).

## View the worker task interface for a 3D point cloud semantic segmentation job

Ground Truth provides workers with a web portal and tools to complete your 3D point cloud semantic segmentation annotation tasks. When you create the labeling job, you provide the Amazon Resource Name (ARN) for a pre-built Ground Truth UI in the `HumanTaskUiArn` parameter. When you create a labeling job using this task type in the console, this UI is automatically used. You can preview and interact with the worker UI when you create a labeling job in the console. If you are a new user, it is recommended that you create a labeling job using the console to ensure your label attributes, point cloud frames, and if applicable, images, appear as expected.

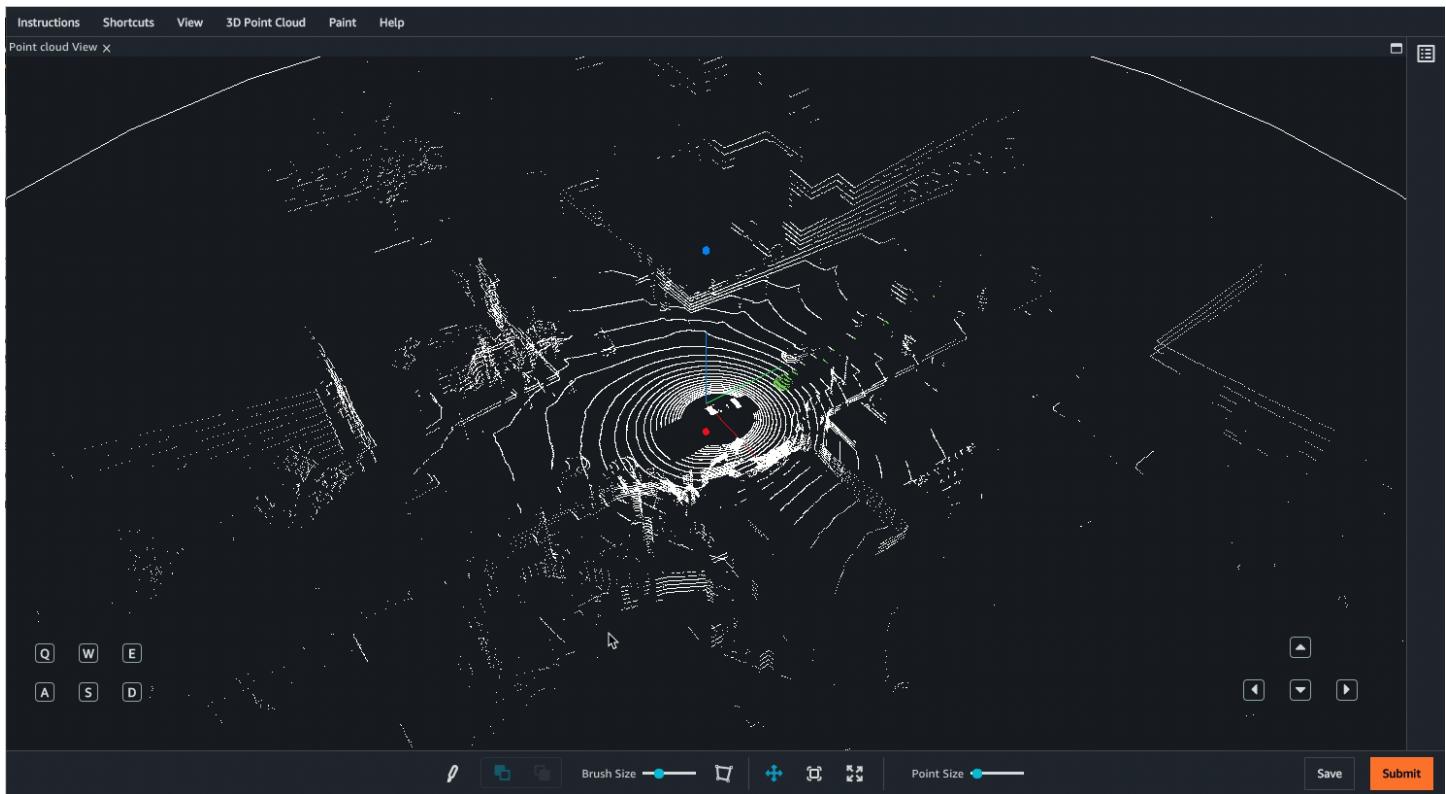
The following is a GIF of the 3D point cloud semantic segmentation worker task interface. If you provide camera data for sensor fusion, images are matched with scenes in the point cloud frame. Workers can paint objects in either the 3D point cloud or the 2D image, and the paint appears in the corresponding location in the other medium. These images appear in the worker portal as shown in the following GIF.



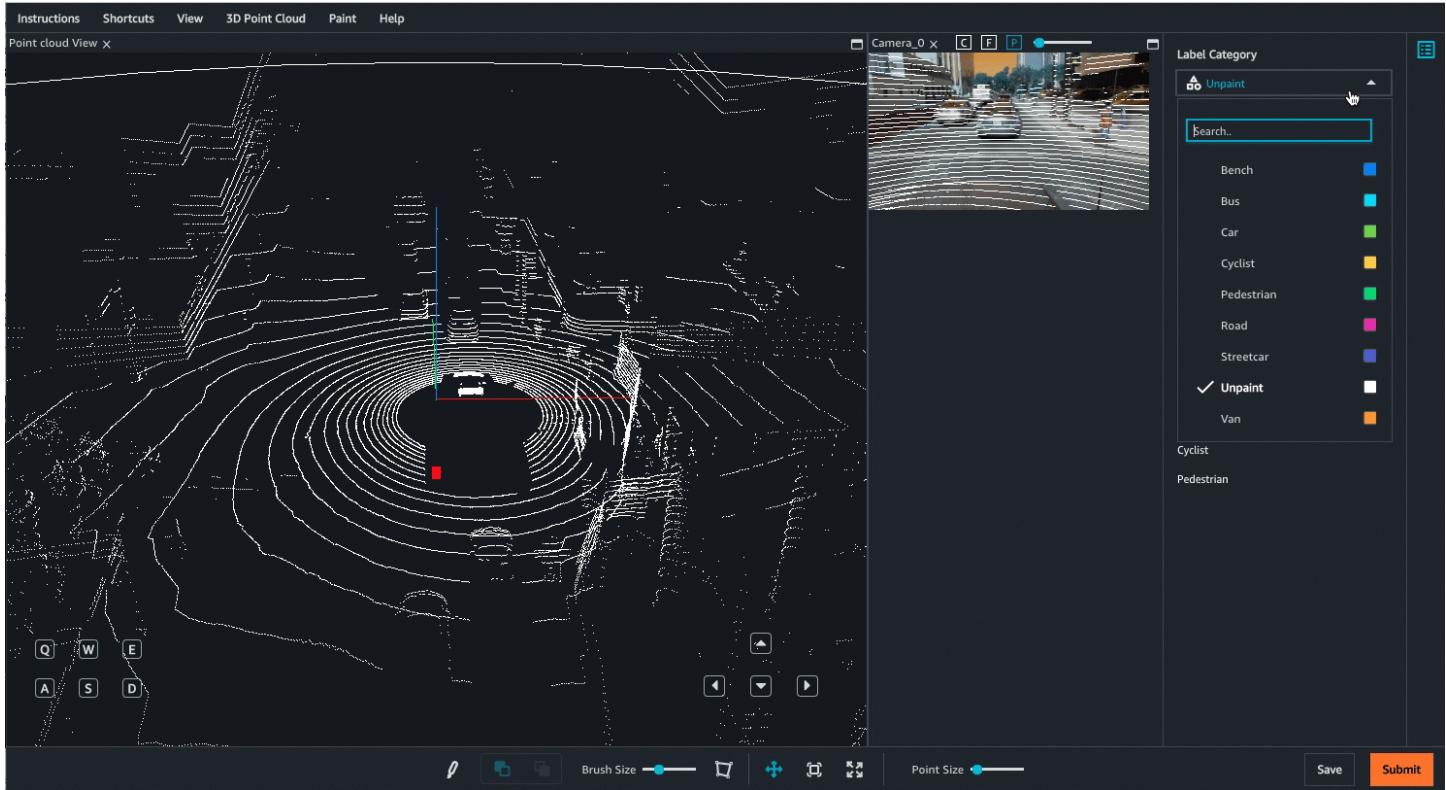
Worker can navigate in the 3D scene using their keyboard and mouse. They can:

- Double click on specific objects in the point cloud to zoom into them.
- Use a mouse-scroller or trackpad to zoom in and out of the point cloud.
- Use both keyboard arrow keys and Q, E, A, and D keys to move Up, Down, Left, Right. Use keyboard keys W and S to zoom in and out.

The following video demonstrates movements around the 3D point cloud. Workers can hide and re-expand all side views and menus. In this GIF, the side-views and menus have been collapsed.



The following GIF demonstrates how a worker can label multiple objects quickly, refine painted objects using the Unpaint option and then view only points that have been painted.



Additional view options and features are available. See the [worker instruction page](#) for a comprehensive overview of the Worker UI.

## Worker Tools

Workers can navigate through the 3D point cloud by zooming in and out, and moving in all directions around the cloud using the mouse and keyboard shortcuts. When you create a semantic segmentation job, workers have the following tools available to them:

- A paint brush to paint and unpaint objects. Workers paint objects by selecting a label category and then painting in the 3D point cloud. Workers unpaint objects by selecting the Unpaint option from the label category menu and using the paint brush to erase paint.
- A polygon tool that workers can use to select and paint an area in the point cloud.
- A background paint tool, which enables workers to paint behind objects they have already annotated without altering the original annotations. For example, workers might use this tool to paint the road after painting all of the cars on the road.
- View options that enable workers to easily hide or view label text, a ground mesh, and additional point attributes like color or intensity. Workers can also choose between perspective and orthogonal projections.

## Output data for a 3D point cloud semantic segmentation job

When you create a 3D point cloud semantic segmentation labeling job, tasks are sent to workers. When these workers complete their tasks, their annotations are written to the Amazon S3 bucket you specified when you created the labeling job. The output data format determines what you see in your Amazon S3 bucket when your labeling job status ([LabelingJobStatus](#)) is Completed.

If you are a new user of Ground Truth, see [Labeling job output data](#) to learn more about the Ground Truth output data format. To learn about the 3D point cloud object detection output data format, see [3D point cloud semantic segmentation output](#).

## Understand the 3D-2D point cloud object tracking task type

Use this task type when you want workers to link 3D point cloud annotations with 2D images annotations and also link 2D image annotations among various cameras. Currently, Ground Truth supports cuboids for annotation in a 3D point cloud and bounding boxes for annotation in 2D videos. For example, you can use this task type to ask workers to link the movement of a vehicle in 3D point cloud with its 2D video. Using 3D-2D linking, you can easily correlate point cloud data (like the distance of a cuboid) to video data (bounding box) for up to 8 cameras.

Ground Truth provides workers with tools to annotate cuboids in a 3D point cloud and bounding boxes in up to 8 cameras using the same annotation UI. Workers can also link various bounding boxes for the same object across different cameras. For example, a bounding box in camera1 can be linked to a bounding box in camera2. This lets you correlate an object across multiple cameras using a unique ID.

### Note

Currently, SageMaker AI does not support creating a 3D-2D linking job using the console.

To create a 3D-2D linking job using the SageMaker API, see [Create a labeling job \(API\)](#).

The following topics explain how to create a 3D-2D point cloud object tracking labeling job, show what the worker task interface looks like (what workers see when they work on this task), and provide an overview of the output data you get when workers complete their tasks.

## Topics

- [Create a 3D-2D point cloud object tracking labeling job](#)
- [View the worker task interface for a 3D-2D object tracking labeling job](#)
- [Output data for a 3D-2D object tracking labeling job](#)

## Create a 3D-2D point cloud object tracking labeling job

You can create a 3D-2D point cloud labeling job using the SageMaker API operation, [CreateLabelingJob](#). To create a labeling job for this task type you need the following:

- A work team from a private or vendor workforce. You cannot use Amazon Mechanical Turk for 3D point cloud labeling jobs. To learn how to create workforces and work teams, see [Workforces](#).
- Add a CORS policy to an S3 bucket that contains input data in the Amazon S3 console. To set the required CORS headers on the S3 bucket that contains your input images in the S3 console, follow the directions detailed in [CORS Permission Requirement](#).
- Additionally, make sure that you have reviewed and satisfied the [Assign IAM Permissions to Use Ground Truth](#).

To learn how to create a labeling job using the API, see the following sections.

## Create a labeling job (API)

This section covers details you need to know when you create a 3D-2D object tracking labeling job using the SageMaker API operation `CreateLabelingJob`. This API defines this operation for all AWS SDKs. To see a list of language-specific SDKs supported for this operation, review the **See Also** section of [CreateLabelingJob](#).

[Create a Labeling Job \(API\)](#) provides an overview of the `CreateLabelingJob` operation. Follow these instructions and do the following while you configure your request:

- You must enter an ARN for `HumanTaskUiArn`. Use  
`arn:aws:sagemaker:<region>:394669845002:human-task-ui/PointCloudObjectTracking`. Replace `<region>` with the AWS Region you are creating the labeling job in.

There should not be an entry for the `UiTemplateS3Uri` parameter.

- Your `LabelAttributeName` must end in `-ref`. For example, `ot-labels-ref`.
- Your input manifest file must be a point cloud frame sequence manifest file. For more information, see [Create a Point Cloud Sequence Input Manifest](#). You also need to provide a label category configuration file as mentioned above.
- You need to provide pre-defined ARNs for the pre-annotation and post-annotation (ACS) Lambda functions. These ARNs are specific to the AWS Region you use to create your labeling job.
  - To find the pre-annotation Lambda ARN, refer to [PreHumanTaskLambdaArn](#). Use the Region you are creating your labeling job in to find the correct ARN that ends with `PRE-3DPointCloudObjectTracking`.
  - To find the post-annotation Lambda ARN, refer to [AnnotationConsolidationLambdaArn](#). Use the Region you are creating your labeling job in to find the correct ARN that ends with `ACS-3DPointCloudObjectTracking`.
- The number of workers specified in `NumberOfHumanWorkersPerDataObject` should be 1.
- Automated data labeling is not supported for 3D point cloud labeling jobs. You should not specify values for parameters in [LabelingJobAlgorithmsConfig](#).
- 3D-2D object tracking labeling jobs can take multiple hours to complete. You can specify a longer time limit for these labeling jobs in `TaskTimeLimitInSeconds` (up to 7 days, or 604,800 seconds).

### Note

After you have successfully created a 3D-2D object tracking job, it shows up on the console under labeling jobs. The task type for the job is displayed as **Point Cloud Object Tracking**.

## Input data format

You can create a 3D-2D object tracking job using the SageMaker API operation, [CreateLabelingJob](#). To create a labeling job for this task type you need the following:

- A sequence input manifest file. To learn how to create this type of manifest file, see [Create a Point Cloud Sequence Input Manifest](#). If you are a new user of Ground Truth 3D point cloud labeling modalities, we recommend that you review [Accepted Raw 3D Data Formats](#).
- You specify your labels, label category and frame attributes, and worker instructions in a label category configuration file. For more information, see [Create a Labeling Category Configuration File with Label Category and Frame Attributes](#) to learn how to create this file. The following is an example showing a label category configuration file for creating a 3D-2D object tracking job.

```
{  
    "document-version": "2020-03-01",  
    "categoryGlobalAttributes": [  
        {  
            "name": "Occlusion",  
            "description": "global attribute that applies to all label categories",  
            "type": "string",  
            "enum": [  
                "Partial",  
                "Full"  
            ]  
        }  
    ],  
    "labels": [  
        {  
            "label": "Car",  
            "attributes": [  
                {  
                    "name": "Type",  
                    "type": "string",  
                    "enum": [  
                        "SUV",  
                        "Hatchback",  
                        "Sedan",  
                        "StationWagon",  
                        "Crossover",  
                        "Minivan",  
                        "Pickup",  
                        "Bicycle",  
                        "Motorcycle",  
                        "Bus",  
                        "Truck",  
                        "Person",  
                        "Animal",  
                        "Object"  
                    ]  
                }  
            ]  
        }  
    ]  
}
```

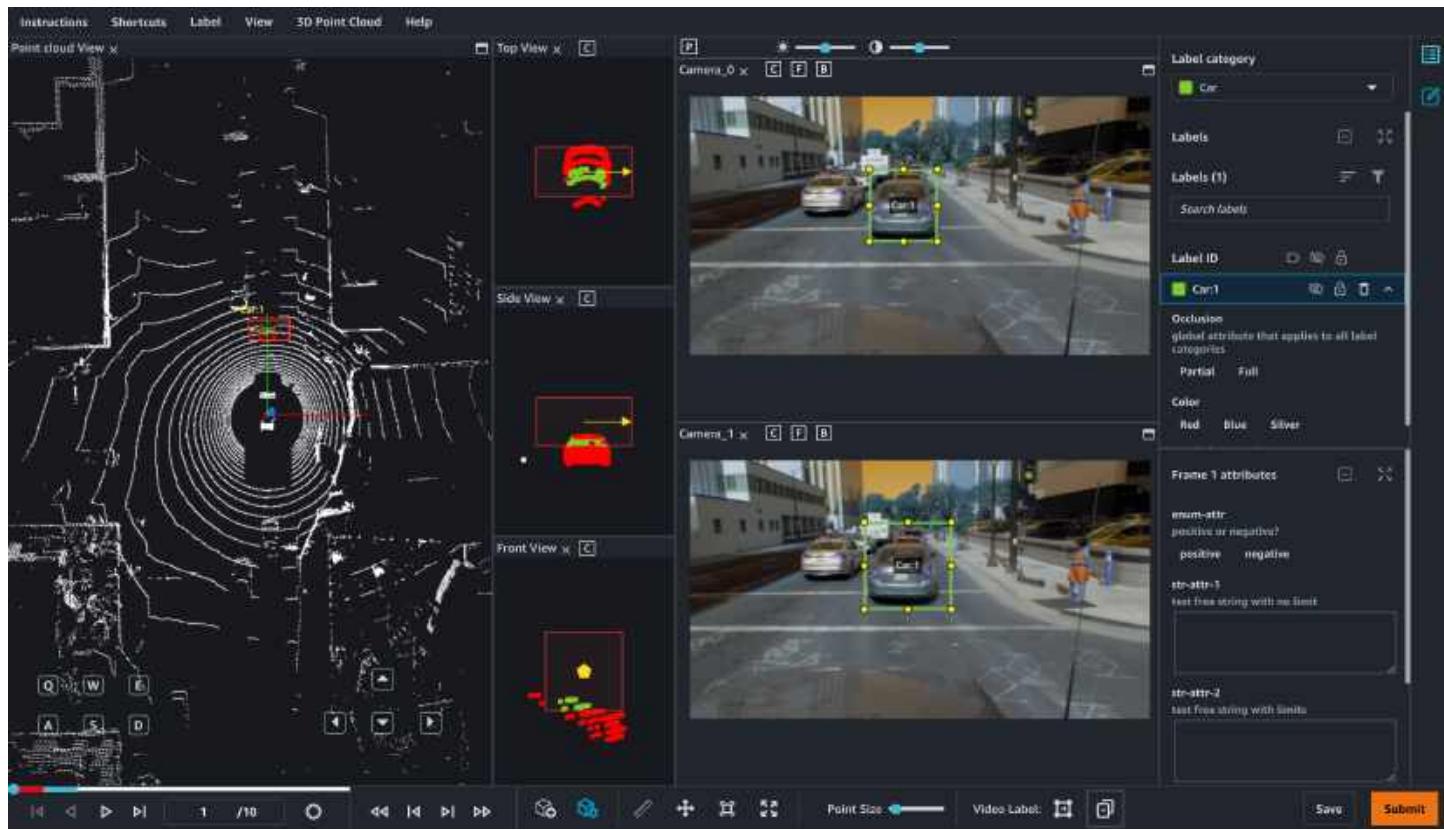
```
        "Sedan"
    ]
}
],
{
    "label": "Bus",
    "attributes": [
        {
            "name": "Size",
            "type": "string",
            "enum": [
                "Large",
                "Medium",
                "Small"
            ]
        }
    ],
    "instructions": {
        "shortIntroduction": "Draw a tight cuboid around objects after you select a category.",
        "fullIntroduction": "<p>Use this area to add more detailed worker instructions.</p>"
    },
    "annotationType": [
        {
            "type": "BoundingBox"
        },
        {
            "type": "Cuboid"
        }
    ]
}
```

 **Note**

You need to provide BoundingBox and Cuboid as annotationType in the label category configuration file to create a 3D-2D object tracking job.

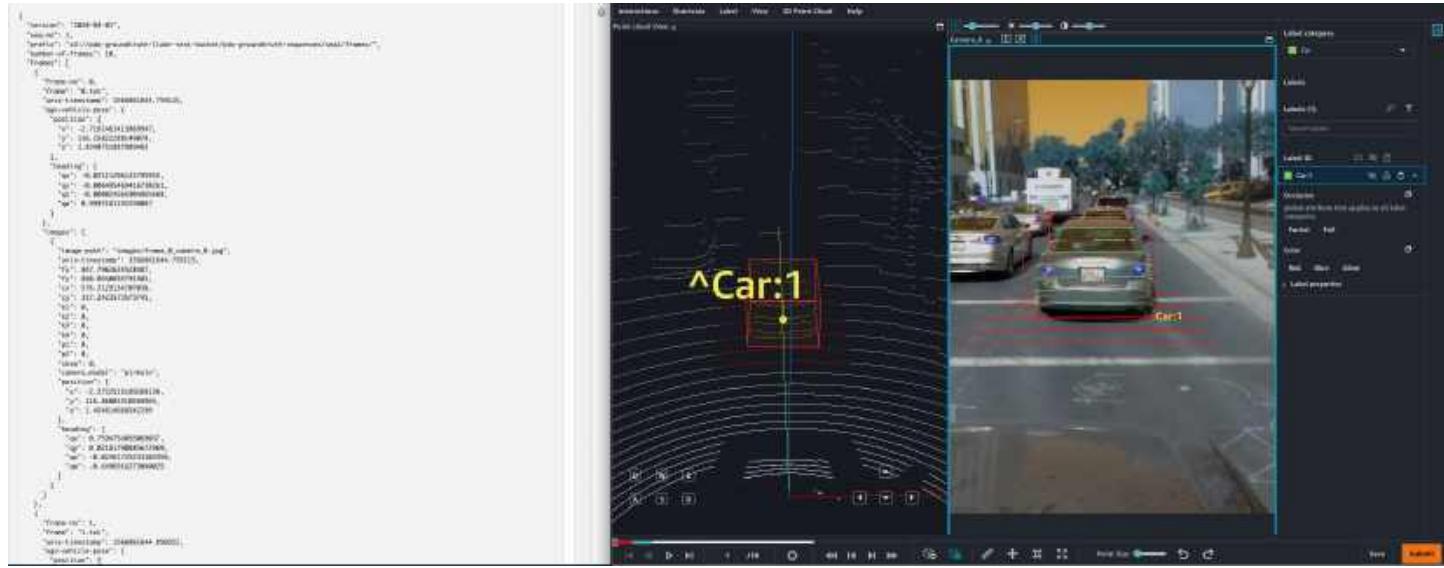
## View the worker task interface for a 3D-2D object tracking labeling job

Ground Truth provides workers with a web portal and tools to complete your 3D-2D object tracking annotation tasks. When you create the labeling job, you provide the Amazon Resource Name (ARN) for a pre-built Ground Truth UI in the HumanTaskUiArn parameter. To use the UI when you create a labeling job for this task type using the API, you need to provide the HumanTaskUiArn. You can preview and interact with the worker UI when you create a labeling job through the API. The annotating tools are a part of the worker task interface. They are not available for the preview interface. The following image demonstrates the worker task interface used for the 3D-2D point cloud object tracking annotation task.



When interpolation is enabled by default. After a worker adds a single cuboid, that cuboid is replicated in all frames of the sequence with the same ID. If the worker adjusts the cuboid in another frame, Ground Truth interpolates the movement of that object and adjust all cuboids between the manually adjusted frames. Additionally, using the camera view section, a cuboid can be shown with a projection (using to B button for "toggle labels" in the camera view) that provides the worker with a reference from the camera images. The accuracy of the cuboid to image projection is based on accuracy of calibrations captured in the extrinsic and intrinsinc data.

If you provide camera data for sensor fusion, images are matched up with scenes in point cloud frames. Note that the camera data should be time synchronized with the point cloud data to ensure an accurate depiction of point cloud to imagery over each frame in the sequence as shown in the following image.



The manifest file holds the extrinsic and intrinsic data and the pose to allow the cuboid projection on the camera image to be shown by using the **P** button.

Worker can navigate in the 3D scene using their keyboard and mouse. They can:

- Double click on specific objects in the point cloud to zoom into them.
- Use a mouse-scroller or trackpad to zoom in and out of the point cloud.
- Use both keyboard arrow keys and Q, E, A, and D keys to move Up, Down, Left, Right. Use keyboard keys W and S to zoom in and out.

Once a worker places a cuboids in the 3D scene, a side-view appears with the three projected side views: top, side, and front. These side-views show points in and around the placed cuboid and help workers refine cuboid boundaries in that area. Workers can zoom in and out of each of those side-views using their mouse.

The worker should first select the cuboid to draw a corresponding bounding box on any of the camera views. This links the cuboid and the bounding box with a common name and unique ID.

The worker can also first draw a bounding box, select it and draw the corresponding cuboid to link them.

Additional view options and features are available. See the [worker instruction page](#) for a comprehensive overview of the Worker UI.

## Worker tools

Workers can navigate through the 3D point cloud by zooming in and out, and moving in all directions around the cloud using the mouse and keyboard shortcuts. If workers click on a point in the point cloud, the UI automatically zooms into that area. Workers can use various tools to draw 3D cuboid around objects. For more information, see **Assistive Labeling Tools** in the following discussion.

After workers have placed a 3D cuboid in the point cloud, they can adjust these cuboids to fit tightly around cars using a variety of views: directly in the 3D point cloud, in a side-view featuring three zoomed-in perspectives of the point cloud around the box, and if you include images for sensor fusion, directly in the 2D image.

Additional view options enable workers to easily hide or view label text, a ground mesh, and additional point attributes. Workers can also choose between perspective and orthogonal projections.

## Assistive Labeling Tools

Ground Truth helps workers annotate 3D point clouds faster and more accurately using UX, machine learning and computer vision powered assistive labeling tools for 3D point cloud object tracking tasks. The following assistive labeling tools are available for this task type:

- **Label autofill** – When a worker adds a cuboid to a frame, a cuboid with the same dimensions, orientation and xyz position is automatically added to all frames in the sequence.
- **Label interpolation** – After a worker has labeled a single object in two frames, Ground Truth uses those annotations to interpolate the movement of that object between all the frames. Label interpolation can be turned on and off. It is on by default. For example, if a worker working with 5 frames adds a cuboid in frame 2, it is copied to all the 5 frames. If the worker then makes adjustments in frame 4, frame 2 and 4 now act as two points, through which a line is fit. The cuboid is then interpolated in frames 1,3 and 5.
- **Bulk label and attribute management** – Workers can add, delete, and rename annotations, label category attributes, and frame attributes in bulk.
  - Workers can manually delete annotations for a given object before and after a frame, or in all frames. For example, a worker can delete all labels for an object after frame 10 if that object is no longer located in the scene after that frame.

- If a worker accidentally bulk deletes all annotations for an object, they can add them back. For example, if a worker deletes all annotations for an object before frame 100, they can bulk add them to those frames.
- Workers can rename a label in one frame and all 3D cuboids assigned that label are updated with the new name across all frames.
- Workers can use bulk editing to add or edit label category attributes and frame attributes in multiple frames.
- **Snapping** – Workers can add a cuboid around an object and use a keyboard shortcut or menu option to have Ground Truth's autofit tool snap the cuboid tightly around the object's boundaries.
- **Fit to ground** – After a worker adds a cuboid to the 3D scene, the worker can automatically snap the cuboid to the ground. For example, the worker can use this feature to snap a cuboid to the road or sidewalk in the scene.
- **Multi-view labeling** – After a worker adds a 3D cuboid to the 3D scene, a side-panel displays front and two side perspectives to help the worker adjust the cuboid tightly around the object. Workers can annotation the 3D point cloud, the side panel and the adjustments appear in the other views in real time.
- **Sensor fusion** – If you provide data for sensor fusion, workers can adjust annotations in the 3D scenes and in 2D images, and the annotations are projected into the other view in real time. To learn more about the data for sensor fusion, see [Understand Coordinate Systems and Sensor Fusion](#).
- **Auto-merge cuboids** – Workers can automatically merge two cuboids across all frames if they determine that cuboids with different labels actually represent a single object.
- **View options** – Enables workers to easily hide or view label text, a ground mesh, and additional point attributes like color or intensity. Workers can also choose between perspective and orthogonal projections.

## Output data for a 3D-2D object tracking labeling job

When you create a 3D-2D object tracking labeling job, tasks are sent to workers. When these workers complete their tasks, their annotations are written to the Amazon S3 bucket you specified when you created the labeling job. The output data format determines what you see in your Amazon S3 bucket when your labeling job status ([LabelingJobStatus](#)) is Completed.

If you are a new user of Ground Truth, see [Labeling job output data](#) to learn more about the Ground Truth output data format. To learn about the 3D-2D point cloud object tracking output data format, see [3D-2D object tracking point cloud object tracking output](#).

## 3D point cloud labeling jobs overview

This topic provides an overview of the unique features of a Ground Truth 3D point cloud labeling job. You can use the 3D point cloud labeling jobs to have workers label objects in a 3D point cloud generated from 3D sensors like LiDAR and depth cameras or generated from 3D reconstruction by stitching images captured by an agent like a drone.

### Job pre-processing time

When you create a 3D point cloud labeling job, you need to provide an [input manifest file](#). The input manifest file can be:

- A *frame input manifest file* that has a single point cloud frame on each line.
- A *sequence input manifest file* that has a single sequence on each line. A sequence is defined as a temporal series of point cloud frames.

For both types of manifest files, *job pre-processing time* (that is, the time before Ground Truth starts sending tasks to your workers) depends on the total number and size of point cloud frames you provide in your input manifest file. For frame input manifest files, this is the number of lines in your manifest file. For sequence manifest files, this is the number of frames in each sequence multiplied by the total number of sequences, or lines, in your manifest file.

Additionally, the number of points per point cloud and the number of fused sensor data objects (like images) factor into job pre-processing times. On average, Ground Truth can pre-process 200 point cloud frames in approximately 5 minutes. If you create a 3D point cloud labeling job with a large number of point cloud frames, you might experience longer job pre-processing times.

For example, if you create a sequence input manifest file with 4 point cloud sequences, and each sequence contains 200 point clouds, Ground Truth pre-processes 800 point clouds and so your job pre-processing time might be around 20 minutes. During this time, your labeling job status is `InProgress`.

While your 3D point cloud labeling job is pre-processing, you receive CloudWatch messages notifying you of the status of your job. To identify these messages, search for `3D_POINT_CLOUD_PROCESSING_STATUS` in your labeling job logs.

For **frame input manifest files**, your CloudWatch logs will have a message similar to the following:

```
{  
    "labeling-job-name": "example-point-cloud-labeling-job",  
    "event-name": "3D_POINT_CLOUD_PROCESSING_STATUS",  
    "event-log-message": "datasetObjectId from: 0 to 10, status: IN_PROGRESS"  
}
```

The event log message, datasetObjectId from: 0 to 10, status: IN\_PROGRESS identifies the number of frames from your input manifest that have been processed. You receive a new message every time a frame has been processed. For example, after a single frame has processed, you receive another message that says datasetObjectId from: 1 to 10, status: IN\_PROGRESS.

For **sequence input manifest files**, your CloudWatch logs will have a message similar to the following:

```
{  
    "labeling-job-name": "example-point-cloud-labeling-job",  
    "event-name": "3D_POINT_CLOUD_PROCESSING_STATUS",  
    "event-log-message": "datasetObjectId: 0, status: IN_PROGRESS"  
}
```

The event log message, datasetObjectId from: 0, status: IN\_PROGRESS identifies the number of sequences from your input manifest that have been processed. You receive a new message every time a sequence has been processed. For example, after a single sequence has processed, you receive a message that says datasetObjectId from: 1, status: IN\_PROGRESS as the next sequence begins processing.

## Job completion times

3D point cloud labeling jobs can take workers hours to complete. You can set the total amount of time that workers can work on each task when you create a labeling job. The maximum time you can set for workers to work on tasks is 7 days. The default value is 3 days.

It is strongly recommended that you create tasks that workers can complete within 12 hours. Workers must keep the worker UI open while working on a task. They can save work as they go and Ground Truth will save their work every 15 minutes.

When using the SageMaker AI CreateLabelingJob API operation, set the total time a task is available to workers in the TaskTimeLimitInSeconds parameter of HumanTaskConfig.

When you create a labeling job in the console, you can specify this time limit when you select your workforce type and your work team.

## Workforces

When you create a 3D point cloud labeling job, you need to specify a work team that will complete your point cloud annotation tasks. You can choose a work team from a private workforce of your own workers, or from a vendor workforce that you select in the AWS Marketplace. You cannot use the Amazon Mechanical Turk workforce for 3D point cloud labeling jobs.

To learn more about vendor workforce, see [Subscribe to vendor workforces](#).

To learn how to create and manage a private workforce, see [Private workforce](#).

## Worker user interface (UI)

Ground Truth provides a worker user interface (UI), tools, and assistive labeling features to help workers complete your 3D point cloud labeling tasks.

You can preview the worker UI when you create a labeling job in the console.

When you create a labeling job using the API operation `CreateLabelingJob`, you must provide an ARN provided by Ground Truth in the parameter [HumanTaskUiArn](#) to specify the worker UI for your task type. You can use `HumanTaskUiArn` with the SageMaker AI [RenderUiTemplate](#) API operation to preview the worker UI.

You provide worker instructions, labels, and optionally, label category attributes that are displayed in the worker UI.

## Label category attributes

When you create a 3D point cloud object tracking or object detection labeling job, you can add one or more *label category attributes*. You can add *frame attributes* to all 3D point cloud task types:

- **Label category attribute** – A list of options (strings), a free form text box, or a numeric field associated with one or more labels. It is used by workers to provide metadata about a label.
- **Frame attribute** – A list of options (strings), a free form text box, or a numeric field that appears on each point cloud frame a worker is sent to annotate. It is used by workers to provide metadata about frames.

Additionally, you can use label and frame attributes to have workers verify labels in a 3D point cloud label verification job.

Use the following sections to learn more about these attributes. To learn how to add label category and frame attributes to a labeling job, use the **Create Labeling Job** section on the [task type page](#) of your choice.

## Label category attributes

Add label category attributes to labels to give workers the ability to provide more information about the annotations they create. A label category attribute is added to an individual label, or to all labels. When a label category attribute is applied to all labels it is referred to as a *global label category attribute*.

For example, if you add the label category *car*, you might also want to capture additional data about your labeled cars, such as if they are occluded or the size of the car. You can capture this metadata using label category attributes. In this example, if you added the attribute *occluded* to the car label category, you can assign *partial*, *completely*, *no* to the *occluded* attribute and enable workers to select one of these options.

When you create a label verification job, you add labels category attributes to each label you want workers to verify.

## Frame attributes

Add frame attributes to give workers the ability to provide more information about individual point cloud frames. You can specify up to 10 frame attributes, and these attributes will appear on all frames.

For example, you can add a frame attribute that allows workers to enter a number. You may want to use this attribute to have workers identify the number of objects they see in a particular frame.

In another example, you may want to provide a free-form text box to give workers the ability to provide a free form answer to a question.

When you create a label verification job, you can add one or more frame attributes to ask workers to provide feedback on all labels in a point cloud frame.

## Worker instructions

You can provide worker instructions to help your workers complete your point cloud labeling tasks. You might want to use these instructions to do the following:

- Best practices and things to avoid when annotating objects.
- Explanation of the label category attributes provided (for object detection and object tracking tasks), and how to use them.
- Advice on how to save time while labeling by using keyboard shortcuts.

You can add your worker instructions using the SageMaker AI console while creating a labeling job. If you create a labeling job using the API operation `CreateLabelingJob`, you specify worker instructions in your label category configuration file.

In addition to your instructions, Ground Truth provides a link to help workers navigate and use the worker portal. View these instructions by selecting the task type on [Worker instructions](#).

## Declining tasks

Workers are able to decline tasks.

Workers decline a task if the instructions are not clear, input data is not displaying correctly, or if they encounter some other issue with the task. If the number of workers per dataset object ([NumberOfHumanWorkersPerDataObject](#)) decline the task, the data object is marked as expired and will not be sent to additional workers.

## 3D point cloud labeling job permission requirements

When you create a 3D point cloud labeling job, in addition to the permission requirements found in [Assign IAM Permissions to Use Ground Truth](#), you must add a CORS policy to your S3 bucket that contains your input manifest file.

### Add a CORS permission policy to S3 bucket

When you create a 3D point cloud labeling job, you specify buckets in S3 where your input data and manifest file are located and where your output data will be stored. These buckets may be the same. You must attach the following Cross-origin resource sharing (CORS) policy to your input and output buckets. If you use the Amazon S3 console to add the policy to your bucket, you must use the JSON format.

### JSON

```
[  
  {  
    "AllowedHeaders": [  
      "*"  
    ]  
  }  
]
```

```
        ],
        "AllowedMethods": [
            "GET",
            "HEAD",
            "PUT"
        ],
        "AllowedOrigins": [
            "*"
        ],
        "ExposeHeaders": [
            "Access-Control-Allow-Origin"
        ],
        "MaxAgeSeconds": 3000
    }
]
```

## XML

```
<?xml version="1.0" encoding="UTF-8"?>
<CORSConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<CORSRule>
    <AllowedOrigin>*</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>HEAD</AllowedMethod>
    <AllowedMethod>PUT</AllowedMethod>
    <MaxAgeSeconds>3000</MaxAgeSeconds>
    <ExposeHeader>Access-Control-Allow-Origin</ExposeHeader>
    <AllowedHeader>*</AllowedHeader>
</CORSRule>
</CORSConfiguration>
```

To learn how to add a CORS policy to an S3 bucket, see [How do I add cross-domain resource sharing with CORS?](#) in the Amazon Simple Storage Service User Guide.

## Worker instructions

This topic provides an overview of the Ground Truth worker portal and the tools available to complete your 3D Point Cloud labeling task. First, select the type of task you are working on from **Topics**.

For adjustment jobs, select the original labeling job task type that produced the labels you are adjusting. Review and adjust the labels in your task as needed.

## Important

It is recommended that you complete your task using a Google Chrome or Firefox web browser.

## Topics

- [3D point cloud semantic segmentation](#)
- [3D point cloud object detection](#)
- [3D point cloud object tracking](#)

## **3D point cloud semantic segmentation**

Use this page to become familiarize with the user interface and tools available to complete your 3D point cloud semantic segmentation task.

## Topics

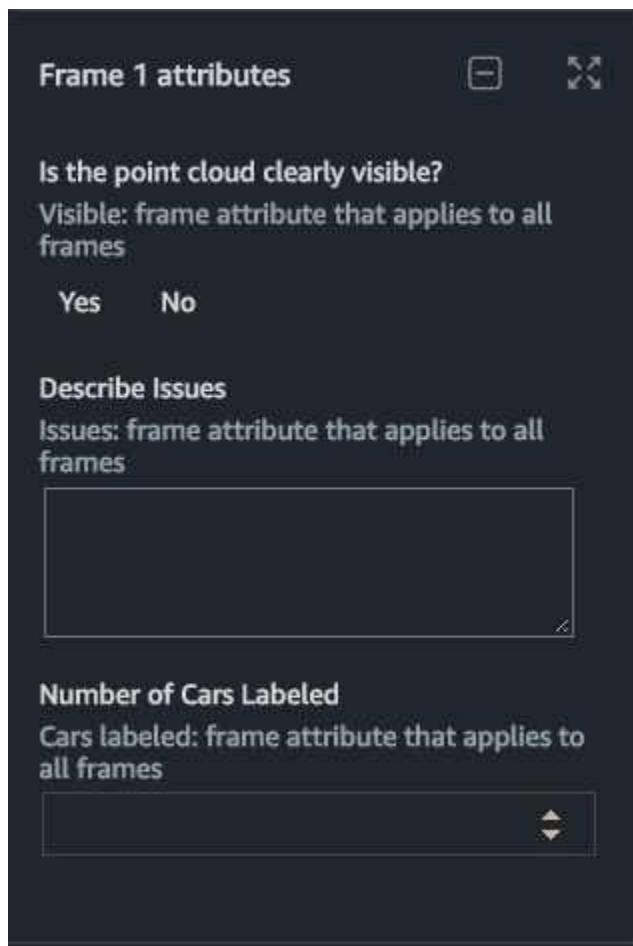
- [Your Task](#)
- [Navigate the UI](#)
- [Icon Guide](#)
- [Shortcuts](#)
- [Release, Stop and Resume, and Decline Tasks](#)
- [Saving Your Work and Submitting](#)

## Your Task

When you work on a 3D point cloud semantic segmentation task, you need to select a category from the **Annotations** menu on the right side of your worker portal using the drop down menu **Label Categories**. After you've selected a category, use the paint brush and polygon tools to paint each object in the 3D point cloud that this category applies to. For example, if you select the category **Car**, you would use these tools to paint all of the cars in the point cloud. The following video demonstrates how to use the paint brush tool to paint an object.

If you see one or more images in your worker portal, you can paint in the images or paint in the 3D point cloud and the paint will show up in the other medium.

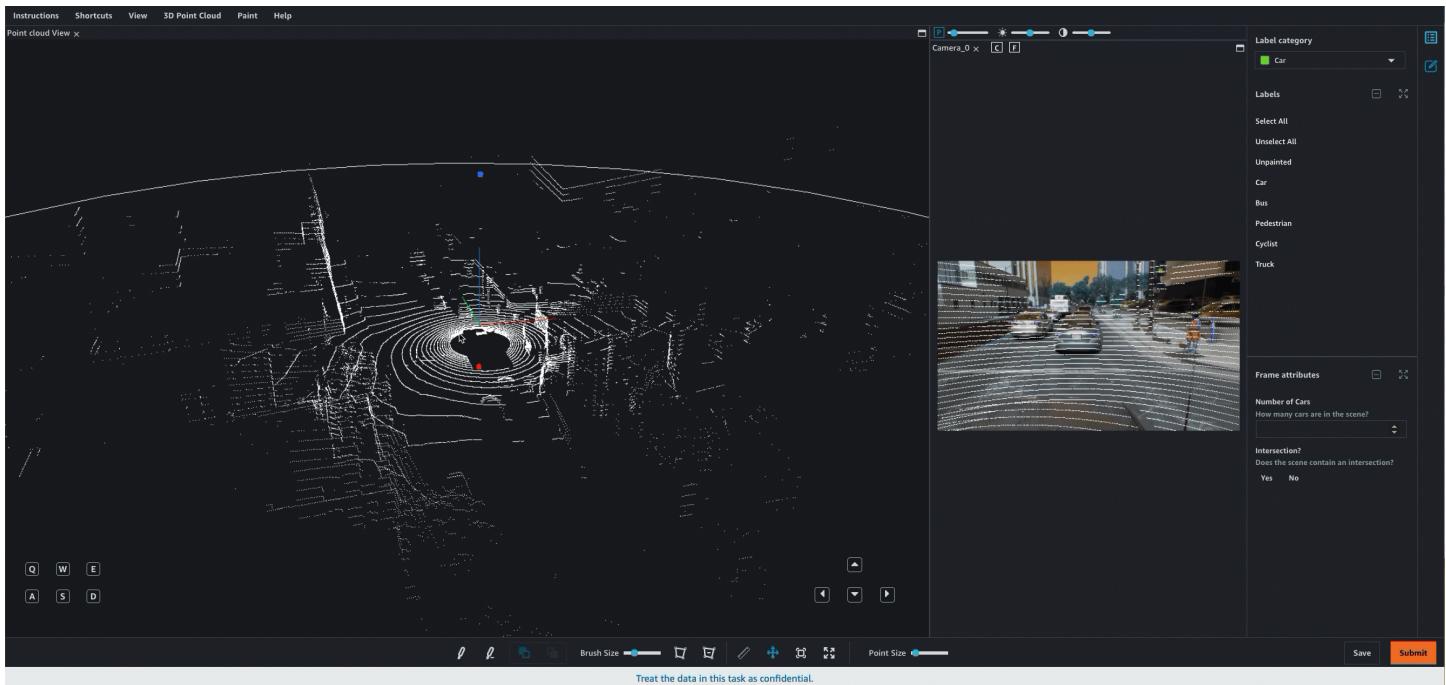
You may see frame attributes under the **Labels** menu. Use these attribute prompts to enter additional information about the point cloud.



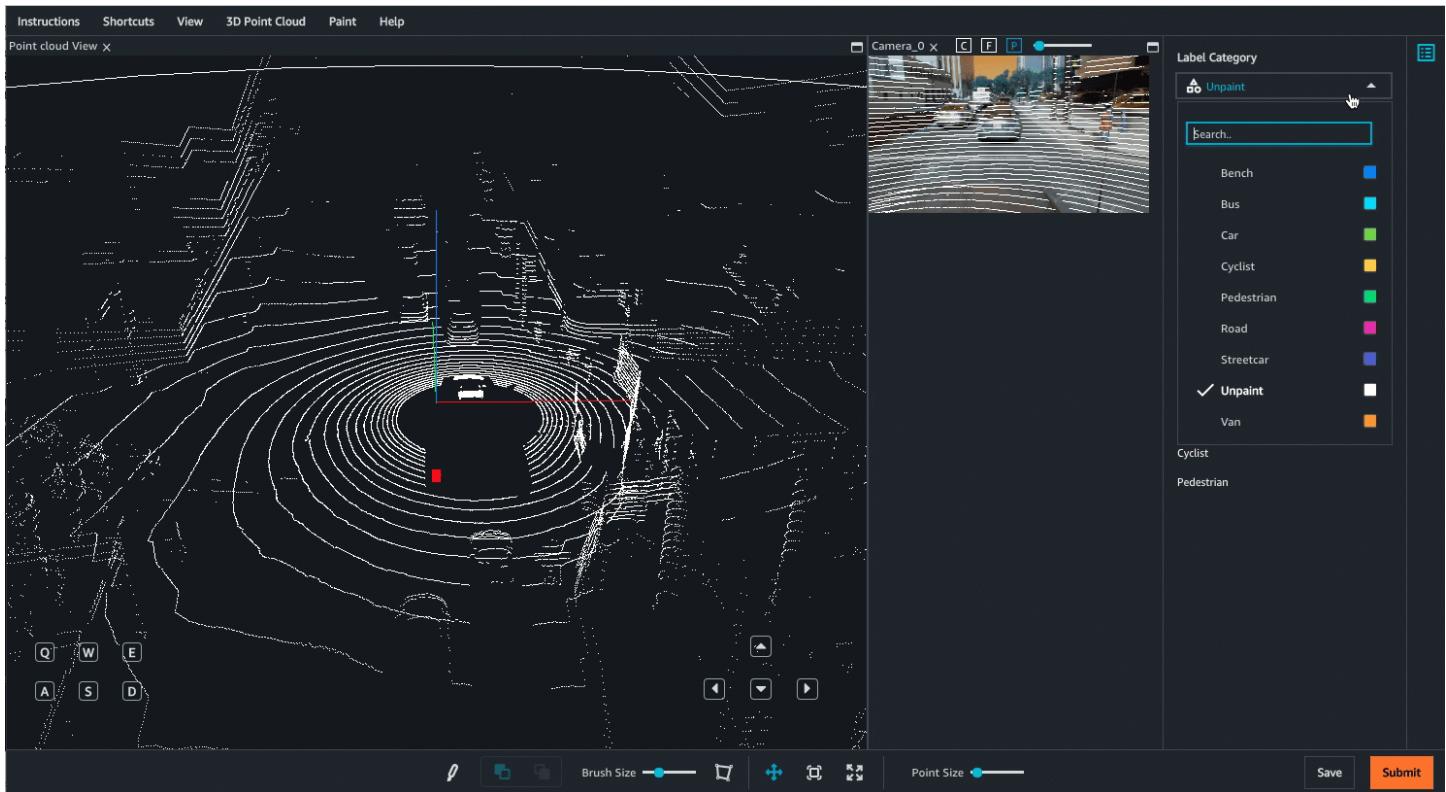
**⚠️ Important**

If you see that objects have already been painted when you open the task, adjust those annotations.

The following video includes an image that can be annotated. You may not see an image in your task.



After you've painted one or more objects using a label category, you can select that category from the Label Category menu on the right to only view points painted for that category.

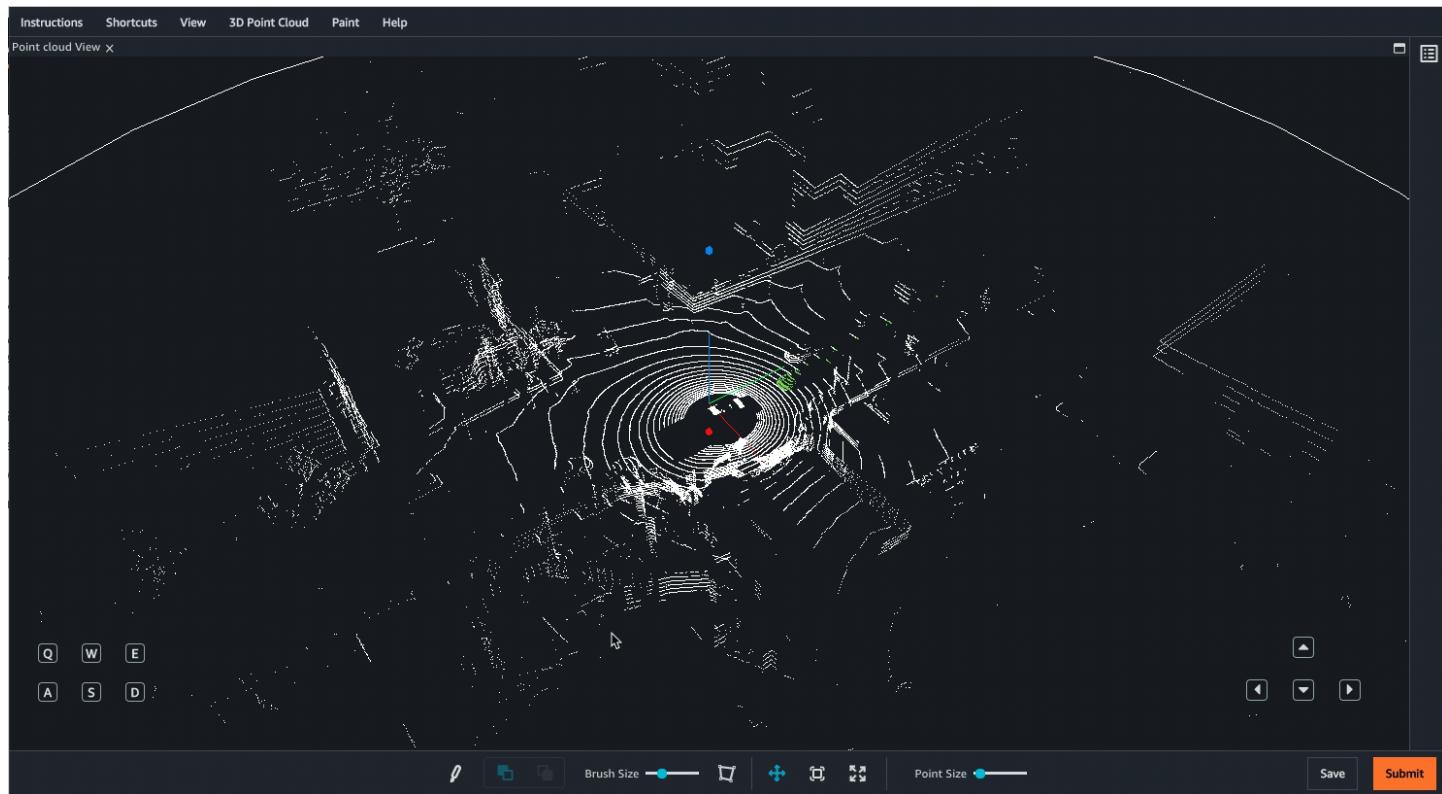


## Navigate the UI

You can navigate in the 3D scene using their keyboard and mouse. You can:

- Double click on specific objects in the point cloud to zoom into them.
- Use a mouse-scroller or trackpad to zoom in and out of the point cloud.
- Use both keyboard arrow keys and Q, E, A, and D keys to move Up, Down, Left, Right. Use keyboard keys W and S to zoom in and out.

The following video demonstrates movements around the 3D point cloud and in the side-view. You can hide and re-expand all side views using the full screen icon. In this GIF, the side-views and menus have been collapsed.



When you are in the worker UI, you see the following menus:

- **Instructions** – Review these instructions before starting your task.
- **Shortcuts** – Use this menu to view keyboard shortcuts that you can use to navigate the point cloud and use the annotation tools provided.
- **View** – Use this menu to toggle different view options on and off. For example, you can use this menu to add a ground mesh to the point cloud, and to choose the projection of the point cloud.

- **3D Point Cloud** – Use this menu to add additional attributes to the points in the point cloud, such as color, and pixel intensity. Note that some or all of these options may not be available.
- **Paint** – Use this menu to modify the functionality of the paint brush.

When you open a task, the move scene icon is on, and you can move around the point cloud using your mouse and the navigation buttons in the point cloud area of the screen. To return to the original view you see when you first opened the task, choose the reset scene icon.

After you select the paint icon, you can add paint to the point cloud and images (if included). You must select the move scene icon again to move to another area in the 3D point cloud or image.

To collapse all panels on the right and make the 3D point cloud full screen, select the full screen icon.

For the camera images and side-panels, you have the following view options:

- **C** – View the camera angle on point cloud view.
- **F** – View the frustum, or field of view, of the camera used to capture that image on point cloud view.
- **P** – View the point cloud overlaid on the image.

## Icon Guide

Use this table to learn about the icons available in your worker task portal.

Icon	Name	Description
	brush	Choose this icon to turn on the brush tool. To use with this tool, choose and move over the objects that you want to paint with your mouse. After you choose it, everything you paint be associated with the category you chose.
	polygon	Choose this icon to use the polygon paint tool. Use this tool to draw polygons around objects that you want to paint. After you choose it, everything you draw a polygon around will be associated with the category you have chosen.

Icon	Name	Description
	reset scene	Choose this icon to reset the view of the point cloud, side panels, and if applicable, all images to their original position when the task was first opened.
	move scene	Choose this icon to move the scene. By default, this icon will be selected when you first start a task.
	full screen	Choose this icon to make the 3D point cloud visualization full screen, and to collapse all side panels.

Icon	Name	Description
	ruler	<p>Use this icon to measure distances, in meters, in the point cloud. You may want to use this tool if your instructions ask you to annotate all objects in a given distance from the center of the cuboid or the object used to capture data.</p> <p>When you select this icon, you can place the starting point (first marker) anywhere in the point cloud by selecting it with your mouse. The tool will automatically use interpolation to place a marker on the closest point within threshold distance to the location you select, otherwise the marker will be placed on ground. If you place a starting point by mistake, you can use the Escape key to revert marker placement.</p> <p>After you place the first marker, you see a dotted line and a dynamic label that indicates the distance you have moved away from the first marker. Click somewhere else on the point cloud to place a second marker. When you place the second marker, the dotted line becomes solid, and the distance is set.</p> <p>After you set a distance, you can edit it by selecting either marker. You can delete a ruler by selecting anywhere on the ruler and using the Delete key on your keyboard.</p>

## Shortcuts

The shortcuts listed in the **Shortcuts** menu can help you navigate the 3D point cloud and use the paint tool.

Before you start your task, it is recommended that you review the **Shortcuts** menu and become acquainted with these commands.

## Release, Stop and Resume, and Decline Tasks

When you open the labeling task, three buttons on the top right allow you to decline the task (**Decline task**), release it (**Release task**), and stop and resume it at a later time (**Stop and resume later**). The following list describes what happens when you select one of these options:

- **Decline task:** You should only decline a task if something is wrong with the task, such as an issue with the 3D point cloud, images or the UI. If you decline a task, you will not be able to return to the task.
- **Release Task:** If you release a task, you lose all work done on that task. When the task is released, other workers on your team can pick it up. If enough workers pick up the task, you may not be able to return to it. When you select this button and then select **Confirm**, you are returned to the worker portal. If the task is still available, its status will be **Available**. If other workers pick it up, it will disappear from your portal.
- **Stop and resume later:** You can use the **Stop and resume later** button to stop working and return to the task at a later time. You should use the **Save** button to save your work before you select **Stop and resume later**. When you select this button and then select **Confirm**, you are returned to the worker portal, and the task status is **Stopped**. You can select the same task to resume work on it.

Be aware that the person that creates your labeling tasks specifies a time limit in which all tasks must be completed by. If you do not return to and complete this task within that time limit, it will expire and your work will not be submitted. Contact your administrator for more information.

## Saving Your Work and Submitting

You should periodically save your work. Ground Truth will automatically save your work every 15 minutes.

When you open a task, you must complete your work on it before pressing **Submit**.

## 3D point cloud object detection

Use this page to familiarize yourself with the user interface and tools available to complete your 3D point cloud object detection task.

## Topics

- [Your Task](#)

- [Navigate the UI](#)
- [Icon Guide](#)
- [Shortcuts](#)
- [Release, Stop and Resume, and Decline Tasks](#)
- [Saving Your Work and Submitting](#)

## Your Task

When you work on a 3D point cloud object detection task, you need to select a category from the **Annotations** menu on the right side of your worker portal using the **Label Categories** menu. After you've chosen a category, use the add cuboid and fit cuboid tools to fit a cuboid around objects in the 3D point cloud that this category applies to. After you place a cuboid, you can modify its dimensions, location, and orientation directly in the point cloud, and the three panels shown on the right.

If you see one or more images in your worker portal, you can also modify cuboids in the images or in the 3D point cloud and the edits will show up in the other medium.

If you see cuboids have already been added to the 3D point cloud when you open your task, adjust those cuboids and add additional cuboids as needed.

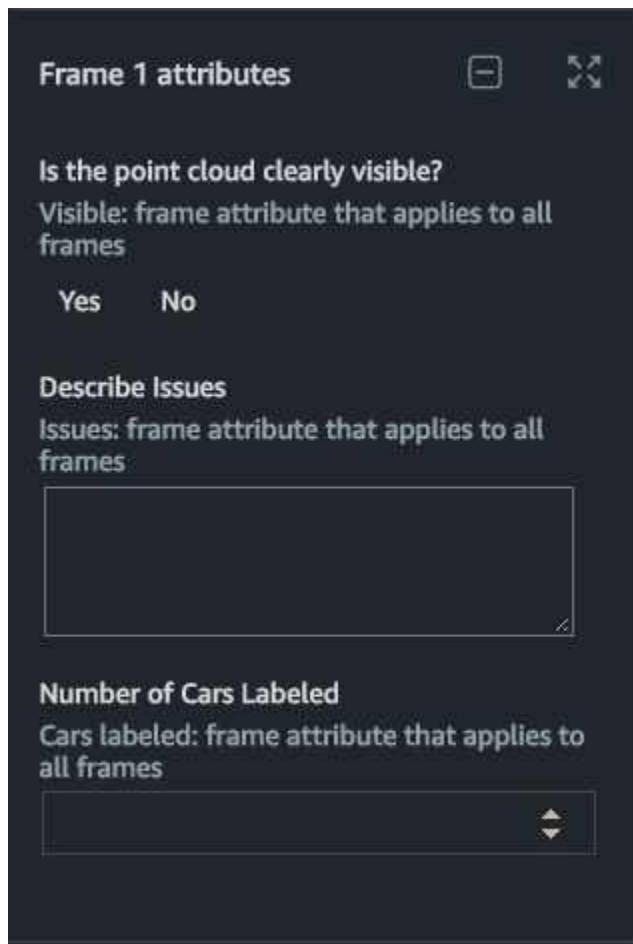
To edit a cuboid, including moving, re-orienting, and changing cuboid dimensions, you must use shortcut keys. You can see a full list of shortcut keys in the **Shortcuts** menu in your UI. The following are important key-combinations that you should become familiar with before starting your labeling task.

Mac Command	Windows Command	Action
Cmd + Drag	Ctrl + Drag	Modify the dimensions of the cuboid.
Option + Drag	Alt + Drag	Move the cuboid.
Shift + Drag	Shift + Drag	Rotate the cuboid.
Option + O	Alt + O	Fit the cuboid tightly around the points it has been drawn around. Before using the

Mac Command	Windows Command	Action
		option, make sure the cuboid fully-surrounds the object of interest.
Option + G	Alt + G	Set the cuboid to the ground.

Individual labels may have one or more label attributes. If a label has a label attribute associated with it, it will appear when you select the downward pointing arrow next to the label from the **Label Id** menu. Fill in required values for all label attributes.

You may see frame attributes under the **Labels** menu. Use these attribute prompts to enter additional information about each frame.



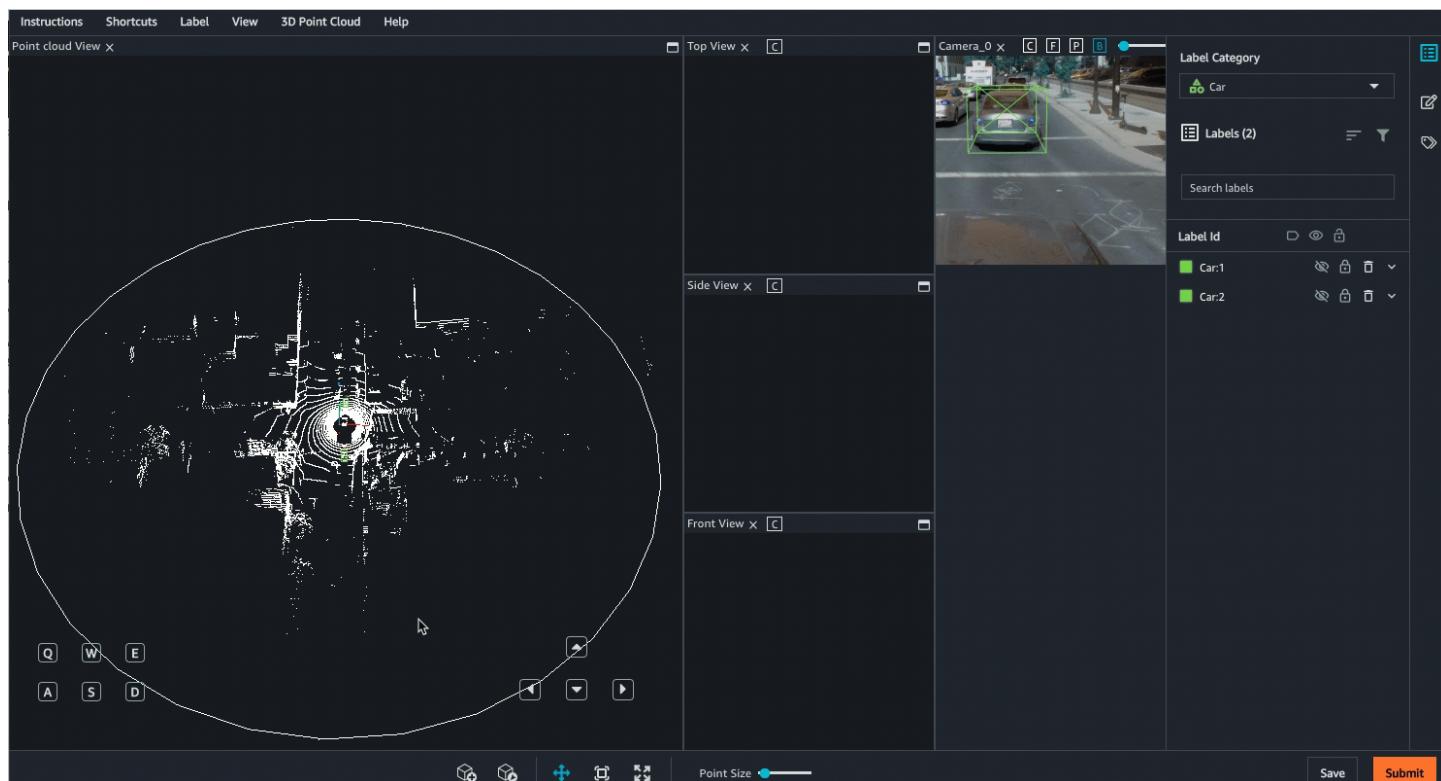
## Navigate the UI

You can navigate in the 3D scene using your keyboard and mouse. You can:

- Double click on specific objects in the point cloud to zoom into them.
- You can use the [ and ] keys on your keyboard to zoom into and move from one label to the next. If no label is selected, when you select [ or ], the UI will zoom into the first label in the **Label Id** list.
- Use a mouse-scroller or trackpad to zoom in and out of the point cloud.
- Use both keyboard arrow keys and Q, E, A, and D keys to move Up, Down, Left, Right. Use keyboard keys W and S to zoom in and out.

Once you place a cuboids in the 3D scene, a side-view will appear with three projected views: top, side, and back. These side-views show points in and around the placed cuboid and help workers refine cuboid boundaries in that area. Workers can zoom in and out of each of those side-views using their mouse.

The following video demonstrates movements around the 3D point cloud and in the side-view.



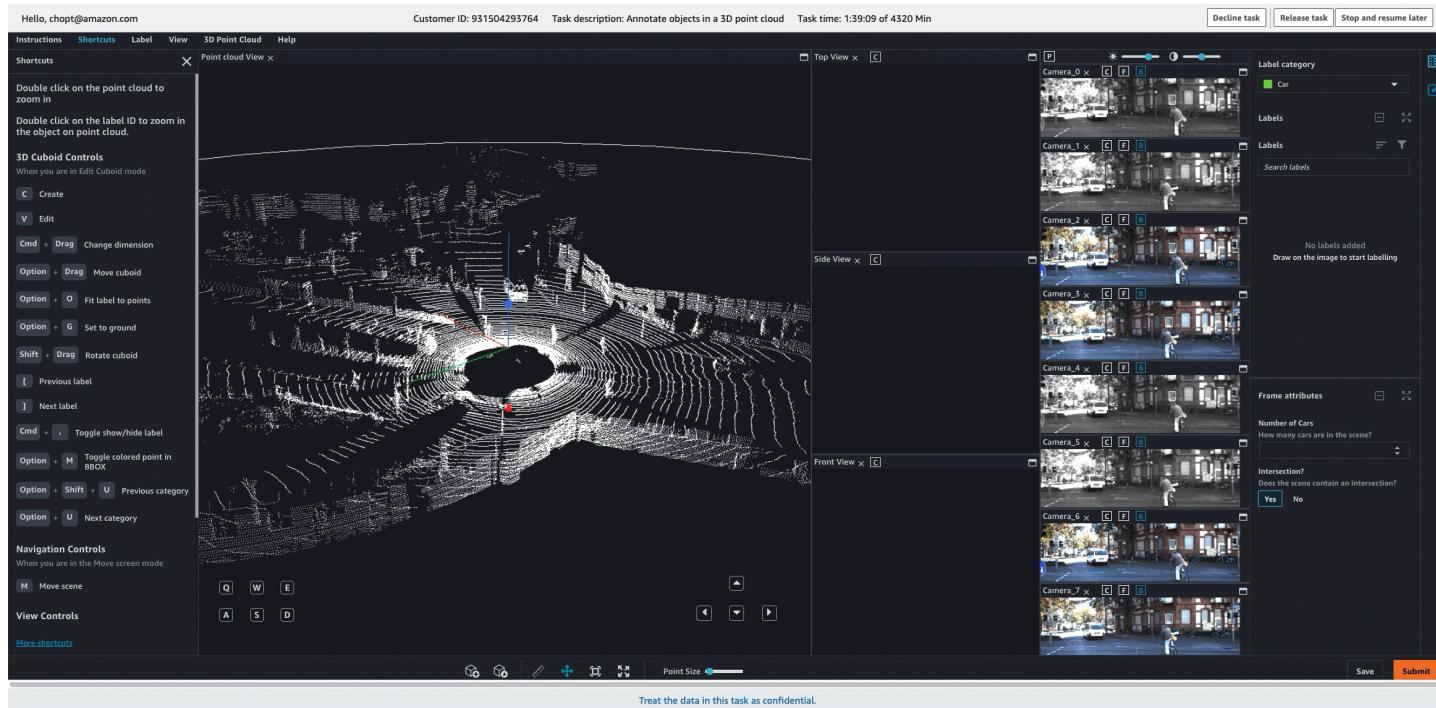
When you are in the worker UI, you see the following menus:

- **Instructions** – Review these instructions before starting your task.
- **Shortcuts** – Use this menu to view keyboard shortcuts that you can use to navigate the point cloud and use the annotation tools provided.

- **Label** – Use this menu to modify a cuboid. First, select a cuboid, and then choose an option from this menu. This menu includes assistive labeling tools like setting a cuboid to the ground and automatically fitting the cuboid to the object's boundaries.
- **View** – Use this menu to toggle different view options on and off. For example, you can use this menu to add a ground mesh to the point cloud, and to choose the projection of the point cloud.
- **3D Point Cloud** – Use this menu to add additional attributes to the points in the point cloud, such as color, and pixel intensity. Note that these options may not be available.

When you open a task, the move scene icon is on, and you can move around the point cloud using your mouse and the navigation buttons in the point cloud area of the screen. To return to the original view you see when you first opened the task, choose the reset scene icon. Resetting the view will not modify your annotations.

After you select the add cuboid icon, you can add cuboids to the 3D point cloud visualization. Once you've added a cuboid, you can adjust it in the three views (top, side, and front) and in the images (if included).



You must choose the move scene icon again to move to another area in the 3D point cloud or image.

To collapse all panels on the right and make the 3D point cloud full-screen, choose the full screen icon.

If camera images are included, you may have the following view options:

- **C** – View the camera angle on point cloud view.
- **F** – View the frustum, or field of view, of the camera used to capture that image on point cloud view.
- **P** – View the point cloud overlaid on the image.
- **B** – View cuboids in the image.

The following video demonstrates how to use these view options. The **F** option is used to view the field of view of the camera (the gray area), the **C** options shows the direction the camera is facing and angle of the camera (blue lines), and the **B** option is used to view the cuboid.



## Icon Guide

Use this table to learn about the icons you see in your worker task portal.

Icon	Name	Description
	add cuboid	Choose this icon to add a cuboid. Each cuboid you add is associated with the category you chose.
	edit cuboid	Choose this icon to edit a cuboid. After you have added a cuboid, you can edit its dimensions, location, and

Icon	Name	Description
		orientation. After a cuboid is added, it automatically switches to edit cuboid mode.
	ruler	<p>Use this icon to measure distances, in meters, in the point cloud. You may want to use this tool if your instructions ask you to annotate all objects in a given distance from the center of the cuboid or the object used to capture data.</p> <p>When you select this icon, you can place the starting point (first marker) anywhere in the point cloud by selecting it with your mouse. The tool will automatically use interpolation to place a marker on the closest point within threshold distance to the location you select, otherwise the marker will be placed on ground. If you place a starting point by mistake, you can use the Escape key to revert marker placement.</p> <p>After you place the first marker, you see a dotted line and a dynamic label that indicates the distance you have moved away from the first marker. Click somewhere else on the point cloud to place a second marker. When you place the second marker, the dotted line becomes solid, and the distance is set.</p> <p>After you set a distance, you can edit it by selecting either marker. You can delete a ruler by selecting anywhere on the ruler and using the Delete key on your keyboard.</p>
	reset scene	Choose this icon to reset the view of the point cloud, side panels, and if applicable, all images to their original position when the task was first opened.
	move scene	Choose this icon to move the scene. By default, this icon is chosen when you first start a task.

Icon	Name	Description
	full screen	Choose this icon to make the 3D point cloud visualization full screen, and to collapse all side panels.
	show labels	Show labels in the 3D point cloud visualization, and if applicable, in images.
	hide labels	Hide labels in the 3D point cloud visualization, and if applicable, in images.
	delete labels	Delete a label.

## Shortcuts

The shortcuts listed in the **Shortcuts** menu can help you navigate the 3D point cloud and use tools to add and edit cuboids.

Before you start your task, it is recommended that you review the **Shortcuts** menu and become acquainted with these commands. You need to use some of the 3D cuboid controls to edit your cuboid.

## Release, Stop and Resume, and Decline Tasks

When you open the labeling task, three buttons on the top right allow you to decline the task (**Decline task**), release it (**Release task**), and stop and resume it at a later time (**Stop and resume later**). The following list describes what happens when you select one of these options:

- **Decline task:** You should only decline a task if something is wrong with the task, such as an issue with the 3D point cloud, images or the UI. If you decline a task, you will not be able to return to the task.
- **Release Task:** If you release a task, you lose all work done on that task. When the task is released, other workers on your team can pick it up. If enough workers pick up the task, you may not be able to return to it. When you select this button and then select **Confirm**, you are returned to the worker portal. If the task is still available, its status will be **Available**. If other workers pick it up, it will disappear from your portal.

- **Stop and resume later:** You can use the **Stop and resume later** button to stop working and return to the task at a later time. You should use the **Save** button to save your work before you select **Stop and resume later**. When you select this button and then select **Confirm**, you are returned to the worker portal, and the task status is **Stopped**. You can select the same task to resume work on it.

Be aware that the person that creates your labeling tasks specifies a time limit in which all tasks must be completed by. If you do not return to and complete this task within that time limit, it will expire and your work will not be submitted. Contact your administrator for more information.

## Saving Your Work and Submitting

You should periodically save your work. Ground Truth will automatically save your work ever 15 minutes.

When you open a task, you must complete your work on it before pressing **Submit**.

## 3D point cloud object tracking

Use this page to become familiarize with the user interface and tools available to complete your 3D point cloud object detection task.

### Topics

- [Your Task](#)
- [Navigate the UI](#)
- [Bulk Edit Label Category and Frame Attributes](#)
- [Icon Guide](#)
- [Shortcuts](#)
- [Release, Stop and Resume, and Decline Tasks](#)
- [Saving Your Work and Submitting](#)

## Your Task

When you work on a 3D point cloud object tracking task, you need to select a category from the **Annotations** menu on the right side of your worker portal using the **Label Categories** menu. After you've selected a category, use the add cuboid and fit cuboid tools to fit a cuboid around objects

in the 3D point cloud that this category applies to. After you place a cuboid, you can modify its location, dimensions, and orientation directly in the point cloud, and the three panels shown on the right. If you see one or more images in your worker portal, you can also modify cuboids in the images or in the 3D point cloud and the edits will show up in the other medium.

 **Important**

If you see cuboids have already been added to the 3D point cloud frames when you open your task, adjust those cuboids and add additional cuboids as needed.

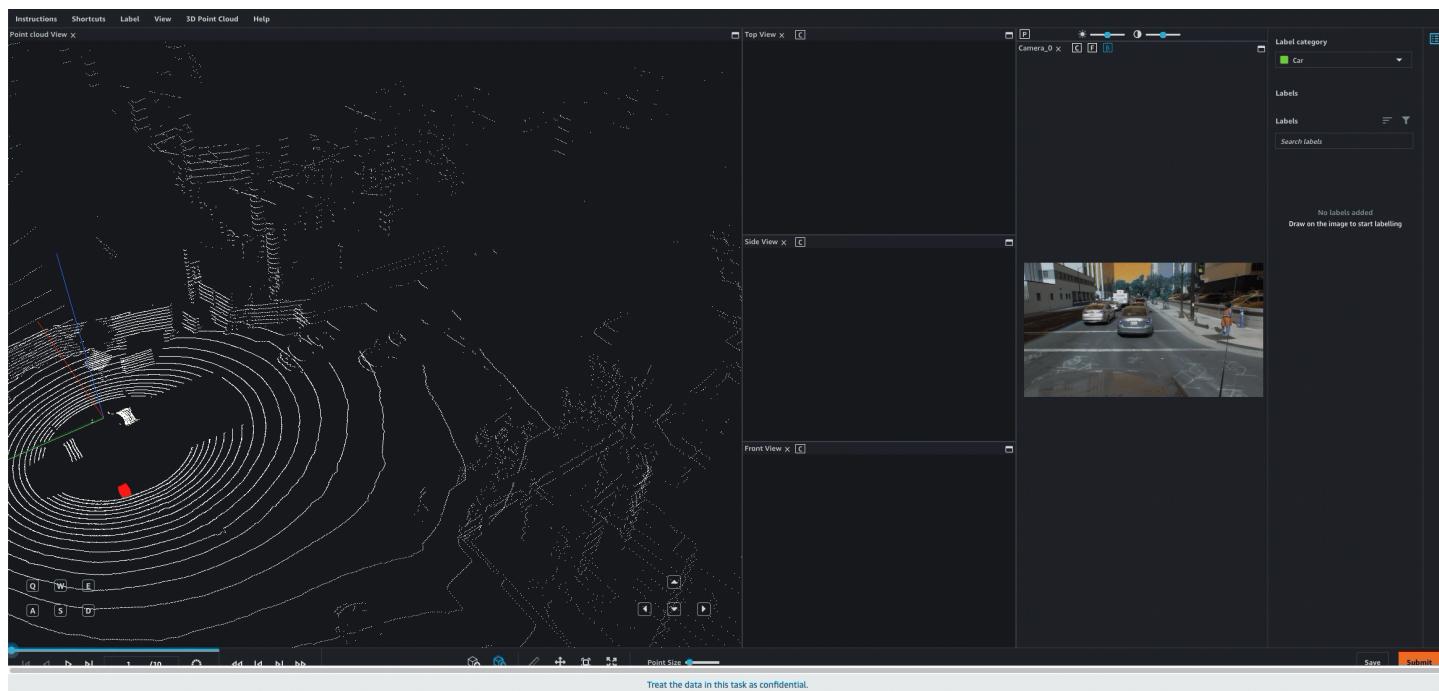
To edit a cuboid, including moving, re-orienting, and changing cuboid dimensions, you must use shortcut keys. You can see a full list of shortcut keys in the **Shortcuts** menu in your UI. The following are important key-combinations that you should become familiar with before starting your labeling task.

Mac Command	Windows Command	Action
Cmd + Drag	Ctrl + Drag	Modify the dimensions of the cuboid.
Option + Drag	Alt + Drag	Move the cuboid.
Shift + Drag	Shift + Drag	Rotate the cuboid.
Option + O	Alt + O	Fit the cuboid tightly around the points it has been drawn around. Before using the option, make sure the cuboid fully-surrounds the object of interest.
Option + G	Alt + G	Set the cuboid to the ground.

When you open your task, two frames will be loaded. If your task includes more than two frames, you need to use the navigation bar in the lower-left corner, or the load frames icon to load additional frames. You should annotate and adjust labels in all frames before submitting.

After you fit a cuboid tightly around the boundaries of an object, navigate to another frame using the navigation bar in the lower-left corner of the UI. If that same object has moved to a new location, add another cuboid and fit it tightly around the boundaries of the object. Each time you manually add a cuboid, you see the frame sequence bar in the lower-left corner of the screen turn red where that frame is located temporally in the sequence.

Your UI automatically infers the location of that object in all other frames after you've placed a cuboid. This is called *interpolation*. You can see the movement of that object, and the inferred and manually created cuboids using the arrows. Adjust inferred cuboids as needed. The following video demonstrates how to navigate between frames. The following video shows how, if you add a cuboid in one frame, and then adjust it in another, your UI will automatically infer the location of the cuboid in all of the frames in-between.



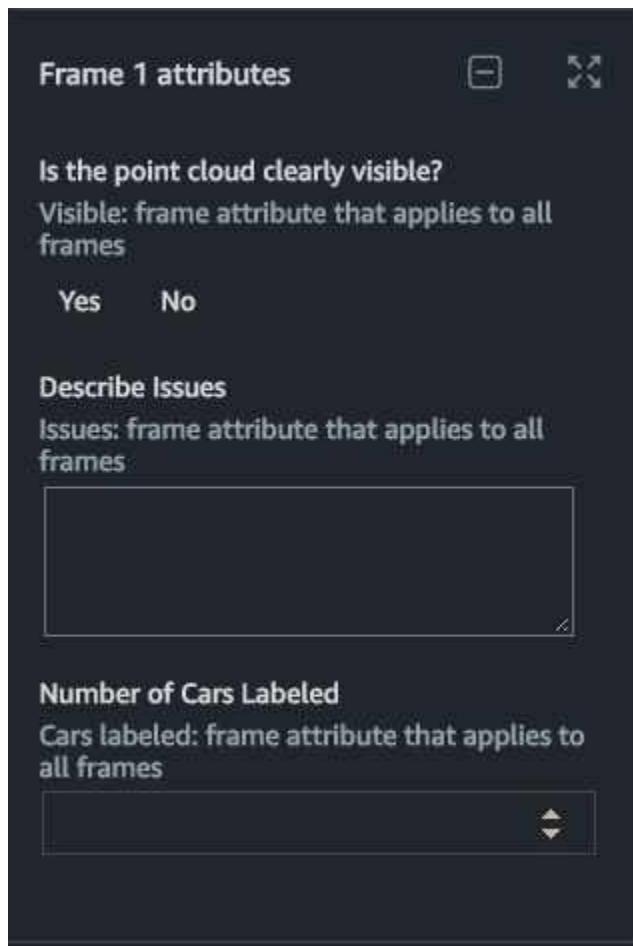
### Tip

You can turn off the automatic cuboid interpolation across frames using the 3D Point Cloud menu item. Select **3D Point Cloud** from the top-menu, and then select **Interpolate Cuboids Across Frames**. This will uncheck this option and stop cuboid interpolation. You can reselect this item to turn cuboid interpolation back on.

Turning cuboid interpolation off will not impact cuboids that have already been interpolated across frames.

Individual labels may have one or more label attributes. If a label has a label attribute associated with it, it will appear when you select the downward pointing arrow next to the label from the **Label Id** menu. Fill in required values for all label attributes.

You may see frame attributes under the **Label Id** menu. These attributes will appear on each frame in your task. Use these attribute prompts to enter additional information about each frame.



## Navigate the UI

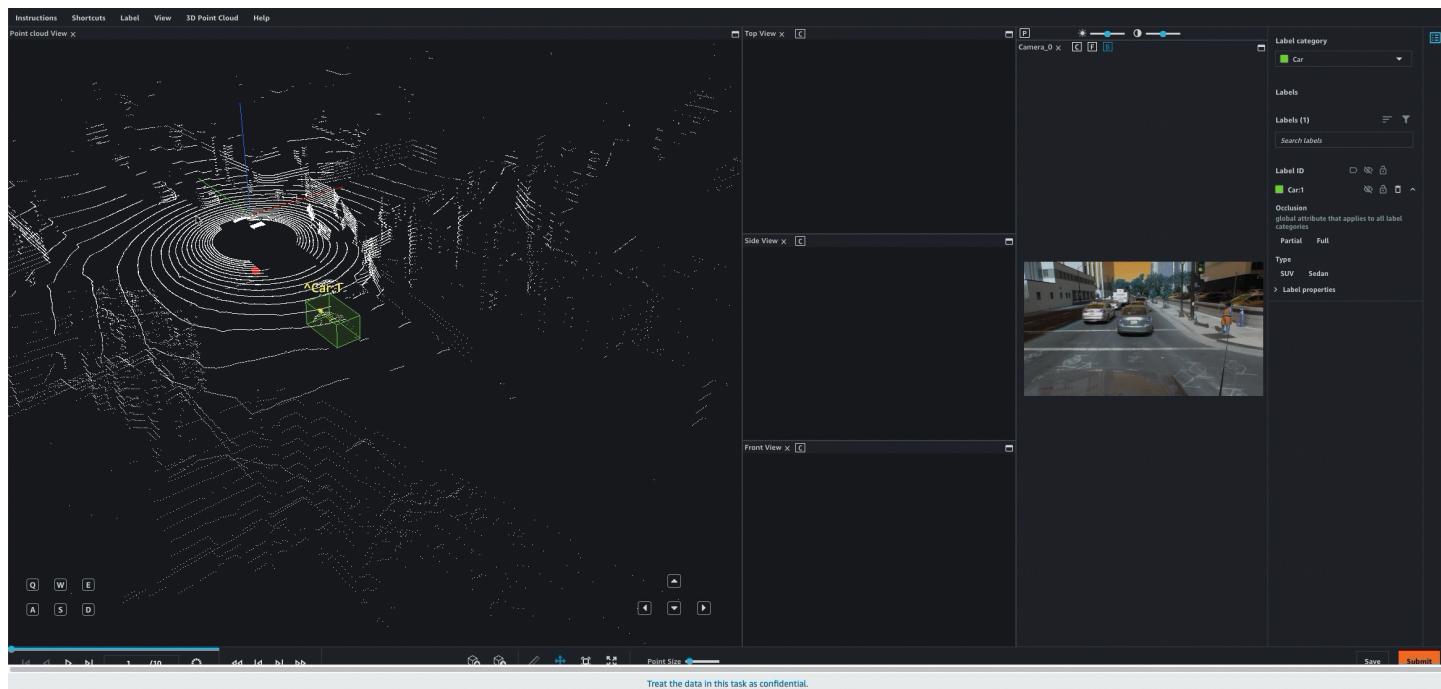
You can navigate in the 3D scene using your keyboard and mouse. You can:

- Double click on specific objects in the point cloud to zoom into them.
- You can use the [ and ] keys on your keyboard to zoom into and move from one label to the next. If no label is selected, when you select [ or ], the UI will zoom into the first label in the **Label Id** list.
- Use a mouse-scroller or trackpad to zoom in and out of the point cloud.

- Use both keyboard arrow keys and Q, E, A, and D keys to move Up, Down, Left, Right. Use keyboard keys W and S to zoom in and out.

Once you place a cuboids in the 3D scene, a side-view will appear with three projected views: top, side, and back. These side-views show points in and around the placed cuboid and help workers refine cuboid boundaries in that area. Workers can zoom in and out of each of those side-views using their mouse.

The following video demonstrates movements around the 3D point cloud and in the side-view.



When you are in the worker UI, you see the following menus:

- **Instructions** – Review these instructions before starting your task.
- **Shortcuts** – Use this menu to view keyboard shortcuts that you can use to navigate the point cloud and use the annotation tools provided.
- **Label** – Use this menu to modify a cuboid. First, select a cuboid, and then choose an option from this menu. This menu includes assistive labeling tools like setting a cuboid to the ground and automatically fitting the cuboid to the object's boundaries.
- **View** – Use this menu to toggle different view options on and off. For example, you can use this menu to add a ground mesh to the point cloud, and to choose the projection of the point cloud.
- **3D Point Cloud** – Use this menu to add additional attributes to the points in the point cloud, such as color, and pixel intensity. Note that these options may not be available.

When you open a task, the move scene icon is on, and you can move around the point cloud using your mouse and the navigation buttons in the point cloud area of the screen. To return to the original view you see when you first opened the task, choose the reset scene icon.

After you select the add cuboid icon, you can add cuboids to the point cloud and images (if included). You must select the move scene icon again to move to another area in the 3D point cloud or image.

To collapse all panels on the right and make the 3D point cloud full-screen, choose the full screen icon.

If camera images are included, you may have the following view options:

- **C** – View the camera angle on point cloud view.
- **F** – View the frustum, or field of view, of the camera used to capture that image on point cloud view.
- **P** – View the point cloud overlaid on the image.
- **B** – View cuboids in the image.

The following video demonstrates how to use these view options. The **F** option is used to view the field of view of the camera (the gray area), the **C** options shows the direction the camera is facing and angle of the camera (blue lines), and the **B** option is used to view the cuboid.



## Delete Cuboids

You can select a cuboid or label ID and:

- Delete an individual cuboid in the current frame you are viewing.
- Delete all cuboids with that label ID before or after the frame you are viewing.
- Delete all cuboids with that label ID in all frames.

A common use-case for cuboid deletion is if the object leaves the scene.

You can use one or more of these options to delete both manually placed and interpolated cuboids with the same label ID.

- To delete all cuboids before or after the frame you are currently on, select the cuboid, select the **Label** menu item at the top of the UI and then select one of **Delete in previous frames** or **Delete in next frames**. Use the Shortcuts menu to see the shortcut keys you can use for these options.
- To delete a label in all frames, select **Delete in all frames** from the **Labels** menu, or use the shortcut **Shift + Delete** on your keyboard.
- To delete an individual cuboid from a single frame, select the cuboid and either select the trashcan icon



)

next to that label ID in the **Label ID** sidebar on the right or use the Delete key on your keyboard to delete that cuboid.

If you have manually placed more than one cuboid with the same label in different frames, when you delete one of the manually placed cuboids, all interpolated cuboids adjust. This adjustment happens because the UI uses manually placed cuboids as anchor points when calculating the location of interpolated cuboid. When you remove one of these anchor points, the UI must recalculate the position of interpolated cuboids.

If you delete a cuboid from a frame, but later decide that you want to get it back, you can use the **Duplicate to previous frames** or **Duplicate to next frames** options in the **Label** menu to copy the cuboid into all the previous or all of the following frames, respectively.

## Bulk Edit Label Category and Frame Attributes

You can bulk edit label attributes and frame attributes.

When you bulk edit an attribute, you specify one or more ranges of frames that you want to apply the edit to. The attribute you select is edited in all frames in that range, including the start and end frames you specify. When you bulk edit label attributes, the range you specify *must* contain the label that the label attribute is attached to. If you specify frames that do not contain this label, you will receive an error.

To bulk edit an attribute you *must* specify the desired value for the attribute first. For example, if you want to change an attribute from *Yes* to *No*, you must select *No*, and then perform the bulk edit.

You can also specify a new value for an attribute that has not been filled in and then use the bulk edit feature to fill in that value in multiple frames. To do this, select the desired value for the attribute and complete the following procedure.

### To bulk edit a label or attribute:

1. Use your mouse to right click the attribute you want to bulk edit.
2. Specify the range of frames you want to apply the bulk edit to using a dash (-) in the text box. For example, if you want to apply the edit to frames one through ten, enter 1-10. If you want to apply the edit to frames two to five, eight to ten and twenty enter 2-5, 8-10, 20.
3. Select **Confirm**.

If you get an error message, verify that you entered a valid range and that the label associated with the label attribute you are editing (if applicable) exists in all frames specified.

You can quickly add a label to all previous or subsequent frames using the **Duplicate to previous frames** and **Duplicate to next frames** options in the **Label** menu at the top of your screen.

## Icon Guide

Use this table to learn about the icons you see in your worker task portal.

Icon	Name	Description
	add cuboid	Choose this icon to add a cuboid. Each cuboid you add is associated with the category you chose.
	edit cuboid	Choose this icon to edit a cuboid. After you add a cuboid, you can edit its dimensions, location, and orientation. After a cuboid is added, it automatically switches to edit cuboid mode.
	ruler	<p>Use this icon to measure distances, in meters, in the point cloud. You may want to use this tool if your instructions ask you to annotate all objects in a given distance from the center of the cuboid or the object used to capture data.</p> <p>When you select this icon, you can place the starting point (first marker) anywhere in the point cloud by selecting it with your mouse. The tool will automatically use interpolation to place a marker on the closest point within threshold distance to the location you select, otherwise the marker will be placed on ground. If you place a starting point by mistake, you can use the Escape key to revert marker placement.</p> <p>After you place the first marker, you see a dotted line and a dynamic label that indicates the distance you have moved away from the first marker. Click somewhere else on the point cloud to place a second marker. When you place the second marker, the dotted line becomes solid, and the distance is set.</p> <p>After you set a distance, you can edit it by selecting either marker. You can delete a ruler by selecting anywhere on the ruler and using the Delete key on your keyboard.</p>

Icon	Name	Description
	reset scene	Choose this icon to reset the view of the point cloud, side panels, and if applicable, all images to their original position when the task was first opened.
	move scene	Choose this icon to move the scene. By default, this icon is chosen when you first start a task.
	full screen	Choose this icon to make the 3D point cloud visualization full screen and to collapse all side panels.
	load frames	Choose this icon to load additional frames.
	hide labels	Hide labels in the 3D point cloud visualization, and if applicable, in images.
	show labels	Show labels in the 3D point cloud visualization, and if applicable, in images.
	delete labels	Delete a label. This option can only be used to delete labels you have manually created or adjusted.

## Shortcuts

The shortcuts listed in the **Shortcuts** menu can help you navigate the 3D point cloud and use tools to add and edit cuboids.

Before you start your task, it is recommended that you review the **Shortcuts** menu and become acquainted with these commands. You need to use some of the 3D cuboid controls to edit your cuboid.

## Release, Stop and Resume, and Decline Tasks

When you open the labeling task, three buttons on the top right allow you to decline the task (**Decline task**), release it (**Release task**), and stop and resume it at a later time (**Stop and resume later**). The following list describes what happens when you select one of these options:

- **Decline task:** You should only decline a task if something is wrong with the task, such as an issue with the 3D point clouds, images or the UI. If you decline a task, you will not be able to return to the task.
- **Release Task:** Use this option to release a task and allow others to work on it. When you release a task, you lose all work done on that task and other workers on your team can pick it up. If enough workers pick up the task, you may not be able to return to it. When you select this button and then select **Confirm**, you are returned to the worker portal. If the task is still available, its status will be **Available**. If other workers pick it up, it will disappear from your portal.
- **Stop and resume later:** You can use the **Stop and resume later** button to stop working and return to the task at a later time. You should use the **Save** button to save your work before you select **Stop and resume later**. When you select this button and then select **Confirm**, you are returned to the worker portal, and the task status is **Stopped**. You can select the same task to resume work on it.

Be aware that the person that creates your labeling tasks specifies a time limit in which all tasks must be completed by. If you do not return to and complete this task within that time limit, it will expire and your work will not be submitted. Contact your administrator for more information.

## Saving Your Work and Submitting

You should periodically save your work. Ground Truth will automatically save your work every 15 minutes.

When you open a task, you must complete your work on it before pressing **Submit**.

## Label verification and adjustment

When the labels on a dataset need to be validated, Amazon SageMaker Ground Truth provides functionality to have workers verify that labels are correct or to adjust previous labels. These types of jobs fall into two distinct categories:

- *Label verification* — Workers indicate if the existing labels are correct, or rate their quality, and can add comments to explain their reasoning. Workers will not be able to modify or adjust labels.

If you create a 3D point cloud or video frame label adjustment or verification job, you can choose to make label category attributes (not supported for 3D point cloud semantic segmentation) and frame attributes editable by workers.

- *Label adjustment* — Workers adjust prior annotations and, if applicable, label category and frame attributes to correct them.

The following Ground Truth [built-in task types](#) support adjustment and verification labeling jobs:

- Bounding box
- Semantic segmentation
- 3D point cloud object detection, 3D point cloud object tracking, and 3D point cloud semantic segmentation
- All video frame object detection and video frame object tracking task types — bounding box, polyline, polygon and keypoint

 **Tip**

For 3D point cloud and video frame labeling verification jobs, it is recommended that you add new label category attributes or frame attributes to the labeling job. Workers can use these attribute to verify individual labels or the entire frame. To learn more about label category and frame attributes, see [Worker user interface \(UI\)](#) for 3D point cloud and [Worker user interface \(UI\)](#) for video frame.

You can start a label verification and adjustment jobs using the SageMaker AI console or the API.

## Cautions and considerations

To get expected behavior when creating a label verification or adjustment job, carefully verify your input data.

- If you are using image data, verify that your manifest file contains hexadecimal RGB color information.

- To save money on processing costs, filter your data to ensure you are not including unwanted objects in your labeling job input manifest.
- Add required Amazon S3 permissions to ensure your input data is processed correctly.

When you create an adjustment or verification labeling job using the Ground Truth API, you *must* use a different LabelAttributeName than the original labeling job.

## Color information requirements for semantic segmentation jobs

To properly reproduce color information in verification or adjustment tasks, the tool requires hexadecimal RGB color information in the manifest (for example, #FFFFFF for white). When you set up a Semantic Segmentation verification or adjustment job, the tool examines the manifest to determine if this information is present. If it can't find it, Amazon SageMaker Ground Truth displays an error message and the ends job setup.

In prior iterations of the Semantic Segmentation tool, category color information wasn't output in hexadecimal RGB format to the output manifest. That feature was introduced to the output manifest at the same time the verification and adjustment workflows were introduced. Therefore, older output manifests aren't compatible with this new workflow.

## Filter your data before starting the job

Amazon SageMaker Ground Truth processes all objects in your input manifest. If you have a partially labeled data set, you might want to create a custom manifest using an [Amazon S3 Select query](#) on your input manifest. Unlabeled objects individually fail, but they don't cause the job to fail, and they might incur processing costs. Filtering out objects you don't want verified reduces your costs.

If you create a verification job using the console, you can use the filtering tools provided there. If you create jobs using the API, make filtering your data part of your workflow where needed.

## Topics

- [Requirements to create verification and adjustment labeling jobs](#)
- [Create a label verification job \(console\)](#)
- [Create a label adjustment job \(console\)](#)
- [Start a label verification or adjustment job \(API\)](#)
- [Label verification and adjustment data in the output manifest](#)

## Requirements to create verification and adjustment labeling jobs

To create a label verification or adjustment job, you must satisfy the following criteria.

- For non streaming labeling jobs: The input manifest file you use must contain the label attribute name (`LabelAttributeName`) of the labels that you want adjusted. When you chain a successfully completed labeling job, the output manifest file is used as the input manifest file for the new, chained job. To learn more about the format of the output manifest file Ground Truth produces for each task type, see [Labeling job output data](#).

For streaming labeling jobs: The Amazon SNS message you sent to the Amazon SNS input topic of the adjustment or verification labeling job must contain the label attribute name of the labels you want adjusted or verified. To see an example of how you can create an adjustment or verification labeling job with streaming labeling jobs, see this [Jupyter Notebook example](#) in GitHub.

- The task type of the verification or adjustment labeling job must be the same as the task type of the original job unless you are using the [Image Label Verification](#) task type to verify bounding box or semantic segmentation image labels. See the next bullet point for more details about the video frame task type requirements.
- For video frame annotation verification and adjustment jobs, you must use the same annotation task type used to create the annotations from the previous labeling job. For example, if you create a video frame object detection job to have workers draw bounding boxes around objects, and then you create a video object detection adjustment job, you must specify *bounding boxes* as the annotation task type. To learn more video frame annotation task types, see [Task types](#).
- The task type you select for the adjustment or verification labeling job must support an audit workflow. The following Ground Truth [built-in task types](#) support adjustment and verification labeling jobs: bounding box, semantic segmentation, 3D point cloud object detection, 3D point cloud object tracking, and 3D point cloud semantic segmentation, and all video frame object detection and video frame object tracking task types — bounding box, polyline, polygon and keypoint.

## Create a label verification job (console)

Use one of the following sections to create a label verification job for your task type. Bounding box and semantic segmentation labeling jobs are created by choosing the **Label verification** task type in the console. To create a verification job for 3D point cloud and video frame task types, you must choose the same task type as the original labeling job and choose to display existing labels.

## Create an image label verification job (console)

Use the following procedure to create a bounding box or semantic segmentation verification job using the console. This procedure assumes that you have already created a bounding box or semantic segmentation labeling job and its status is Complete. This is the labeling job that produces the labels you want verified.

### To create an image label verification job:

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/> and choose **Labeling jobs**.
2. Start a new labeling job by chaining a prior job or start from scratch, specifying an input manifest that contains labeled data objects.
3. In the **Task type** pane, select **Label verification**.
4. Choose **Next**.
5. In the **Workers** section, choose the type of workforce you would like to use. For more details about your workforce options see [Workforces](#).
6. (Optional) After you've selected your workforce, specify the **Task timeout** and **Task expiration time**.
7. In the **Existing-labels display options** pane, the system shows the available label attribute names in your manifest. Choose the label attribute name that identifies the labels that you want workers to verify. Ground Truth tries to detect and populate these values by analyzing the manifest, but you might need to set the correct value.
8. Use the instructions areas of the tool designer to provide context about what the previous labelers were asked to do and what the current verifiers need to check.

You can add new labels that workers choose from to verify labels. For example, you can ask workers to verify the image quality, and provide the labels *Clear* and *Blurry*. Workers will also have the option to add a comment to explain their selection.

9. Choose **See preview** to check that the tool is displaying the prior labels correctly and presents the label verification task clearly.
10. Select **Create**. This will create and start your labeling job.

## Create a point cloud or video frame label verification job (console)

Use the following procedure to create a 3D point cloud or video frame verification job using the console. This procedure assumes that you have already created a labeling job using the task type that produces the types of labels you want to be verified and its status is Complete.

### To create an image label verification job:

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/> and choose **Labeling jobs**.
2. Start a new labeling job by chaining a prior job or start from scratch, specifying an input manifest that contains labeled data objects.
3. In the **Task type** pane, select the same task type as the labeling job that you chained. For example, if the original labeling job was a video frame object detection keypoint labeling job, select that task type.
4. Choose **Next**.
5. In the **Workers** section, choose the type of workforce you would like to use. For more details about your workforce options see [Workforces](#).
6. (Optional) After you've selected your workforce, specify the **Task timeout** and **Task expiration time**.
7. Toggle on the switch next to **Display existing labels**.
8. Select **Verification**.
9. For **Label attribute name**, choose the name from your manifest that corresponds to the labels that you want to display for verification. You will only see label attribute names for labels that match the task type you selected on the previous screen. Ground Truth tries to detect and populate these values by analyzing the manifest, but you might need to set the correct value.
10. Use the instructions areas of the tool designer to provide context about what the previous labelers were asked to do and what the current verifiers need to check.

You cannot modify or add new labels. You can remove, modify and add new label category attributes or frame attributes. It is recommended that you add new label category attributes or frame attributes to the labeling job. Workers can use these attribute to verify individual labels or the entire frame.

By default, preexisting label category attributes and frame attributes will not be editable by workers. If you want to make a label category or frame attribute editable, select the **Allow workers to edit this attribute** check box for that attribute.

To learn more about label category and frame attributes, see [Worker user interface \(UI\)](#) for 3D point cloud and [Worker user interface \(UI\)](#) for video frame.

11. Choose **See preview** to check that the tool is displaying the prior labels correctly and presents the label verification task clearly.
12. Select **Create**. This will create and start your labeling job.

## Create a label adjustment job (console)

Use one of the following sections to create a label verification job for your task type.

### Topics

- [Create an image label adjustment job \(console\)](#)
- [Create a point cloud or video frame label adjustment job \(console\)](#)

## Create an image label adjustment job (console)

Use the following procedure to create a bounding box or semantic segmentation adjustment labeling job using the console. This procedure assumes that you have already created a bounding box or semantic segmentation labeling job and its status is Complete. This the labeling job that produces the labels you want adjusted.

### To create an image label adjustment job (console)

1. Open the SageMaker AI console at <https://console.aws.amazon.com/sagemaker/> and choose **Labeling jobs**.
2. Start a new labeling job by [chaining](#) a prior job or start from scratch, specifying an input manifest that contains labeled data objects.
3. Choose the same task type as the original labeling job.
4. Choose **Next**.
5. In the **Workers** section, choose the type of workforce you would like to use. For more details about your workforce options see [Workforces](#).

6. (Optional) After you've selected your workforce, specify the **Task timeout** and **Task expiration time**.
7. Expand **Existing-labels display options** by selecting the arrow next to the title.
8. Check the box next to **I want to display existing labels from the dataset for this job**.
9. For **Label attribute name**, choose the name from your manifest that corresponds to the labels that you want to display for adjustment. You will only see label attribute names for labels that match the task type you selected on the previous screen. Ground Truth tries to detect and populate these values by analyzing the manifest, but you might need to set the correct value.
10. Use the instructions areas of the tool designer to provide context about what the previous labelers were tasked with doing and what the current verifiers need to check and adjust.
11. Choose **See preview** to check that the tool shows the prior labels correctly and presents the task clearly.
12. Select **Create**. This will create and start your labeling job.

### Create a point cloud or video frame label adjustment job (console)

Use the following procedure to create a 3D point cloud or video frame adjustment job using the console. This procedure assumes that you have already created a labeling job using the task type that produces the types of labels you want to be verified and its status is Complete.

### To create a 3D point cloud or video frame label adjustment job (console)

1. Open the SageMaker AI console: <https://console.aws.amazon.com/sagemaker/> and choose **Labeling jobs**.
2. Start a new labeling job by chaining a prior job or start from scratch, specifying an input manifest that contains labeled data objects.
3. Choose the same task type as the original labeling job.
4. Toggle on the switch next to **Display existing labels**.
5. Select **Adjustment**.
6. For **Label attribute name**, choose the name from your manifest that corresponds to the labels that you want to display for adjustment. You will only see label attribute names for labels that match the task type you selected on the previous screen. Ground Truth tries to detect and populate these values by analyzing the manifest, but you might need to set the correct value.
7. Use the instructions areas of the tool designer to provide context about what the previous labelers were asked to do and what the current adjusters need to check.

You cannot remove or modify existing labels but you can add new labels. You can remove, modify and add new label category attributes or frame attributes.

By default, preexisting label category attributes and frame attributes will be editable by workers. If you want to make a label category or frame attribute uneditable, deselect the **Allow workers to edit this attribute** check box for that attribute.

To learn more about label category and frame attributes, see [Worker user interface \(UI\)](#) for 3D point cloud and [Worker user interface \(UI\)](#) for video frame.

8. Choose **See preview** to check that the tool shows the prior labels correctly and presents the task clearly.
9. Select **Create**. This will create and start your labeling job.

## Start a label verification or adjustment job (API)

Start a label verification or adjustment job by chaining a successfully completed job or starting a new job from scratch using the [CreateLabelingJob](#) operation. The procedure is almost the same as setting up a new labeling job with CreateLabelingJob, with a few modifications. Use the following sections to learn what modifications are required to chain a labeling job to create an adjustment or verification labeling job.

When you create an adjustment or verification labeling job using the Ground Truth API, you *must* use a different LabelAttributeName than the original labeling job. The original labeling job is the job used to create the labels you want adjusted or verified.

### Important

The label category configuration file you identify for an adjustment or verification job in [LabelCategoryConfigS3Uri](#) of CreateLabelingJob must contain the same labels used in the original labeling job. You can add new labels. For 3D point cloud and video frame jobs, you can add new label category and frame attributes to the label category configuration file.

## Bounding Box and Semantic Segmentation

To create a bounding box or semantic segmentation label verification or adjustment job, use the following guidelines to specify API attributes for the CreateLabelingJob operation.

- Use the [LabelAttributeName](#) parameter to specify the output label name that you want to use for verified or adjusted labels. You must use a different LabelAttributeName than the one used for the original labeling job.
- If you are chaining the job, the labels from the previous labeling job to be adjusted or verified will be specified in the custom UI template. To learn how to create a custom template, see [Create Custom Worker Task Templates](#).

Identify the location of the UI template in the [UiTemplateS3Uri](#) parameter. SageMaker AI provides widgets that you can use in your custom template to display old labels. Use the initial-value attribute in one of the following crowd elements to extract the labels that need verification or adjustment and include them in your task template:

- [crowd-semantic-segmentation](#)—Use this crowd element in your custom UI task template to specify semantic segmentation labels that need to be verified or adjusted.
- [crowd-bounding-box](#)—Use this crowd element in your custom UI task template to specify bounding box labels that need to be verified or adjusted.
- The [LabelCategoryConfigS3Uri](#) parameter must contain the same label categories as the previous labeling job.
- Use the bounding box or semantic segmentation adjustment or verification lambda ARNs for [PreHumanTaskLambdaArn](#) and [AnnotationConsolidationLambdaArn](#):
  - For bounding box, the adjustment labeling job lambda function ARNs end with AdjustmentBoundingBox and the verification lambda function ARNs end with VerificationBoundingBox.
  - For semantic segmentation, the adjustment labeling job lambda function ARNs end with AdjustmentSemanticSegmentation and the verification lambda function ARNs end with VerificationSemanticSegmentation.

## 3D point cloud and video frame

- Use the [LabelAttributeName](#) parameter to specify the output label name that you want to use for verified or adjusted labels. You must use a different LabelAttributeName than the one used for the original labeling job.
- You must use the human task UI Amazon Resource Name (ARN) (HumanTaskUiArn) used for the original labeling job. To see supported ARNs, see [HumanTaskUiArn](#).

- In the label category configuration file, you must specify the label attribute name ([LabelAttributeName](#)) of the previous labeling job that you use to create the adjustment or verification labeling job in the auditLabelAttributeName parameter.
- You specify whether your labeling job is a *verification* or *adjustment* labeling job using the editsAllowed parameter in your label category configuration file identified by the [LabelCategoryConfigS3Uri](#) parameter.
  - For *verification* labeling jobs, you must use the editsAllowed parameter to specify that all labels cannot be modified. editsAllowed must be set to "none" in each entry in labels. Optionally, you can specify whether or not label categories attributes and frame attributes can be adjusted by workers.
  - Optionally, for *adjustment* labeling jobs, you can use the editsAllowed parameter to specify labels, label category attributes, and frame attributes that can or cannot be modified by workers. If you do not use this parameter, all labels, label category attributes, and frame attributes will be adjustable.

To learn more about the editsAllowed parameter and configuring your label category configuration file, see [Label category configuration file schema](#).

- Use the 3D point cloud or video frame adjustment lambda ARNs for [PreHumanTaskLambdaArn](#) and [AnnotationConsolidationLambdaArn](#) for both adjustment and verification labeling jobs:
  - For 3D point clouds, the adjustment and verification labeling job lambda function ARNs end with Adjustment3DPointCloudSemanticSegmentation, Adjustment3DPointCloudObjectTracking, and Adjustment3DPointCloudObjectDetection for 3D point cloud semantic segmentation, object detection, and object tracking respectively.
  - For video frames, the adjustment and verification labeling job lambda function ARNs end with AdjustmentVideoObjectDetection and AdjustmentVideoObjectTracking for video frame object detection and object tracking respectively.

Ground Truth stores the output data from a label verification or adjustment job in the S3 bucket that you specified in the [S3 outputPath](#) parameter of the [CreateLabelingJob](#) operation. For more information about the output data from a label verification or adjustment labeling job, see [Label verification and adjustment data in the output manifest](#).

## Label verification and adjustment data in the output manifest

Amazon SageMaker Ground Truth writes label verification data to the output manifest within the metadata for the label. It adds two properties to the metadata:

- A type property, with a value of "groundtruth/label-verification".
- A worker-feedback property, with an array of comment values. This property is added when the worker enters comments. If there are no comments, the field doesn't appear.

The following example output manifest shows how label verification data appears:

```
{  
  "source-ref": "S3 bucket location",  
  "verify-bounding-box": "1",  
  "verify-bounding-box-metadata":  
  {  
    "class-name": "bad",  
    "confidence": 0.93,  
    "type": "groundtruth/label-verification",  
    "job-name": "verify-bounding-boxes",  
    "human-annotated": "yes",  
    "creation-date": "2018-10-18T22:18:13.527256",  
    "worker-feedback": [  
      {"comment": "The bounding box on the bird is too wide on the right side."},  
      {"comment": "The bird on the upper right is not labeled."}  
    ]  
  }  
}
```

The worker output of adjustment tasks resembles the worker output of the original task, except that it contains the adjusted values and an adjustment-status property with the value of adjusted or unadjusted to indicate whether an adjustment was made.

For more examples of the output of different tasks, see [Labeling job output data](#).

## Custom labeling workflows

These topics help you set up a Ground Truth labeling job that uses a custom labeling template. A custom labeling template allows you to create a custom worker portal UI that workers will use to label data. Template can be created using HTML, CSS, JavaScript, [Liquid template language](#), and [Crowd HTML Elements](#).

## Overview

If this is your first time creating a custom labeling workflow in Ground Truth, the following list is a high-level summary of the steps required.

1. *Set up your workforce* – To create a custom labeling workflow you need a workforce. This topic teaches you about configuring a workforce.
2. *Creating a custom template* – To create a custom template you must map the data from your input manifest file correctly to the variables in your template.
3. *Using optional processing Lambda functions* – To control how data from your input manifest is added to your worker template, and how worker annotations are logged in your job's output file.

This topic also has three end-to-end demos to help you better understand how to use custom labeling templates.

 **Note**

The examples in the links below all include pre-annotation and post-annotation Lambda functions. These Lambda functions are optional.

- [Demo template: Annotation of images with crowd-bounding-box](#)
- [Demo Template: Labeling Intents with crowd-classifier](#)
- [Build a custom data labeling workflow with Amazon SageMaker Ground Truth](#)

### Topics

- [Set up your workforce](#)
- [Creating a custom worker task template](#)
- [Adding automation with Liquid](#)
- [Processing data in a custom labeling workflow with AWS Lambda](#)
- [Demo template: Annotation of images with crowd-bounding-box](#)
- [Demo Template: Labeling Intents with crowd-classifier](#)
- [Create a custom workflow using the API](#)

## Set up your workforce

In this step you use the console to establish which worker type to use and make the necessary sub-selections for the worker type. It assumes you have already completed the steps up to this point in the [Getting started: Create a bounding box labeling job with Ground Truth](#) section and have chosen the **Custom labeling task** as the **Task type**.

### To configure your workforce.

1. First choose an option from the **Worker types**. There are three types currently available:
  - **Public** uses an on-demand workforce of independent contractors, powered by Amazon Mechanical Turk. They are paid on a per-task basis.
  - **Private** uses your employees or contractors for handling data that needs to stay within your organization.
  - **Vendor** uses third party vendors that specialize in providing data labeling services, available via the AWS Marketplace.
2. If you choose the **Public** option, you are asked to set the **number of workers per dataset object**. Having more than one worker perform the same task on the same object can help increase the accuracy of your results. The default is three. You can raise or lower that depending on the accuracy you need.

You are also asked to set a **price per task** by using a drop-down menu. The menu recommends price points based on how long it will take to complete the task.

The recommended method to determine this is to first run a short test of your task with a **private** workforce. The test provides a realistic estimate of how long the task takes to complete. You can then select the range your estimate falls within on the **Price per task** menu. If your average time is more than 5 minutes, consider breaking your task into smaller units.

### Next

#### [Creating a custom worker task template](#)

## Creating a custom worker task template

To create a custom labeling job, you need to update the worker task template, map the input data from your manifest file to the variables used in the template, and map the output data to Amazon

S3. To learn more about advanced features that use Liquid automation, see [Adding automation with Liquid](#).

The following sections describe each of the required steps.

## Worker task template

A *worker task template* is a file used by Ground Truth to customize the worker user interface (UI). You can create a worker task template using HTML, CSS, JavaScript, [Liquid template language](#), and [Crowd HTML Elements](#). Liquid is used to automate the template. Crowd HTML Elements are used to include common annotation tools and provide the logic to submit to Ground Truth.

Use the following topics to learn how you can create a worker task template. You can see a repository of example Ground Truth worker task templates on [GitHub](#).

### Using the base worker task template in the SageMaker AI console

You can use a template editor in the Ground Truth console to start creating a template. This editor includes a number of pre-designed base templates. It supports autofill for HTML and Crowd HTML Element code.

#### To access the Ground Truth custom template editor:

1. Following the instructions in [Create a Labeling Job \(Console\)](#).
2. Then select **Custom** for the labeling job **Task type**.
3. Choose **Next**, and then you can access the template editor and base templates in the **Custom labeling task setup** section.
4. (Optional) Select a base template from the drop-down menu under **Templates**. If you prefer to create a template from scratch, choose **Custom** from the drop down-menu for a minimal template skeleton.

Use the following section to learn how to visualize a template developed in the console locally.

### Visualizing your worker task templates locally

You must use the console to test how your template processes incoming data. To test the look and feel of your template's HTML and custom elements you can use your browser.

**Note**

Variables will not be parsed. You may need to replace them with sample content while viewing your content locally.

The following example code snippet loads the necessary code to render the custom HTML elements. Use this if you want to develop your template's look and feel in your preferred editor rather than in the console.

**Example**

```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>
```

**Creating a simple HTML task sample**

Now that you have the base worker task template, you can use this topic to create a simple HTML-based task template.

The following is an example entry from an input manifest file.

```
{  
  "source": "This train is really late.",  
  "labels": [ "angry" , "sad", "happy" , "inconclusive" ],  
  "header": "What emotion is the speaker feeling?"  
}
```

In the HTML task template we need to map the variables from input manifest file to the template. The variable from the example input manifest would be mapped using the following syntax **task.input.source**, **task.input.labels**, and **task.input.header**.

The following is a simple example HTML worker task template for tweet-analysis. All tasks begin and end with the `<crowd-form> </crowd-form>` elements. Like standard HTML `<form>` elements, all of your form code should go between them. Ground Truth generates the workers' tasks directly from the context specified in the template, unless you implement a pre-annotation Lambda. The `taskInput` object returned by Ground Truth or [Pre-annotation Lambda](#) is the `task.input` object in your templates.

For a simple tweet-analysis task, use the `<crowd-classifier>` element. It requires the following attributes:

- *name* - The name of your output variable. Worker annotations are saved to this variable name in your output manifest.
- *categories* - a JSON formatted array of the possible answers.
- *header* - a title for the annotation tool

The <crowd-classifier> element requires at least the three following child elements.

- <*classification-target*> - The text the worker will classify based on the options specified in the categories attribute above.
- <*full-instructions*> - Instructions that are available from the "View full instructions" link in the tool. This can be left blank, but it is recommended that you give good instructions to get better results.
- <*short-instructions*> - A more brief description of the task that appears in the tool's sidebar. This can be left blank, but it is recommended that you give good instructions to get better results.

A simple version of this tool would look like the following. The variable

`{{ task.input.source }}` is what specifies the source data from your input manifest file. The `{{ task.input.labels | to_json }}` is an example of a variable filter to turn the array into a JSON representation. The categories attribute must be JSON.

### Example of using **crowd-classifier** with the sample input manifest json

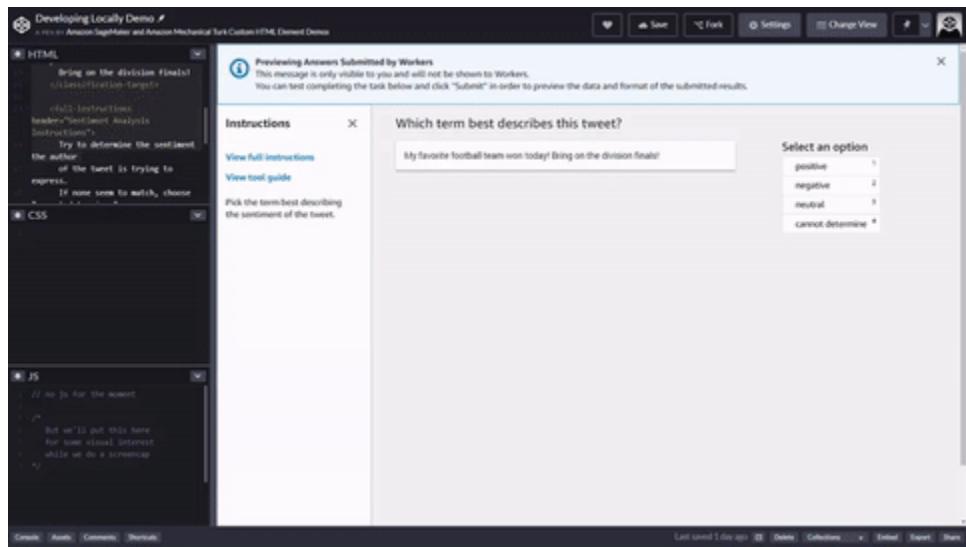
```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>
<crowd-form>
  <crowd-classifier
    name="tweetFeeling"
    categories="='{{ task.input.labels | to_json }}'"
    header="{{ task.input.header }}"
  >
    <classification-target>
      {{ task.input.source }}
    </classification-target>

    <full-instructions header="Sentiment Analysis Instructions">
      Try to determine the sentiment the author
      of the tweet is trying to express.
      If none seem to match, choose "cannot determine."
    </full-instructions>
```

```
<short-instructions>
  Pick the term that best describes the sentiment of the tweet.
</short-instructions>

</crowd-classifier>
</crowd-form>
```

You can copy and paste the code into the editor in the Ground Truth labeling job creation workflow to preview the tool, or try out a [demo of this code on CodePen](#).



## Input data, external assets and your task template

Following sections describe the use of external assets, input data format requirements, and when to consider using pre-annotation Lambda functions.

### Input data format requirements

When you create an input manifest file to use in your custom Ground Truth labeling job, you must store the data in Amazon S3. The input manifest files must also be saved in the same AWS Region in which your custom Ground Truth labeling job is to be run. Furthermore, it can be stored in any Amazon S3 bucket that is accessible to the IAM service role that you use to run your custom labeling job in Ground Truth.

Input manifest files must use the newline-delimited JSON or JSON lines format. Each line is delimited by a standard line break, `\n` or `\r\n`. Each line must also be a valid JSON object.

Furthermore, each JSON object in the manifest file must contain one of the following keys: `source-ref` or `source`. The value of the keys are interpreted as follows:

- **source-ref** – The source of the object is the Amazon S3 object specified in the value. Use this value when the object is a binary object, such as an image.
- **source** – The source of the object is the value. Use this value when the object is a text value.

To learn more about formatting your input manifest files, see [Input manifest files](#).

## Pre-annotation Lambda function

You can optionally specify a *pre-annotation Lambda* function to manage how data from your input manifest file is handled prior to labeling. If you have specified the `isHumanAnnotationRequired` key-value pair you must use a pre-annotation Lambda function. When Ground Truth sends the pre-annotation Lambda function a JSON formatted request it uses the following schemas.

### Example data object identified with the `source-ref` key-value pair

```
{  
  "version": "2018-10-16",  
  "labelingJobArn": arn:aws:lambda:us-west-2:555555555555:function:my-function  
  "dataObject" : {  
    "source-ref": s3://input-data-bucket/data-object-file-name  
  }  
}
```

### Example data object identified with the `source` key-value pair

```
{  
  "version": "2018-10-16",  
  "labelingJobArn" : arn:aws:lambda:us-west-2:555555555555:function:my-function  
  "dataObject" : {  
    "source": Sue purchased 10 shares of the stock on April 10th, 2020  
  }  
}
```

The following is the expected response from the Lambda function when `isHumanAnnotationRequired` is used.

```
{  
  "taskInput": {  
    "source": "This train is really late.",  
    "labels": [ "angry" , "sad" , "happy" , "inconclusive" ],  
  }
```

```
"header": "What emotion is the speaker feeling?"  
},  
"isHumanAnnotationRequired": False  
}
```

## Using External Assets

Amazon SageMaker Ground Truth custom templates allow external scripts and style sheets to be embedded. For example, the following code block demonstrates how you would add a style sheet located at <https://www.example.com/my-enhancement-styles.css> to your template.

### Example

```
<script src="https://www.example.com/my-enhancement-script.js"></script>  
<link rel="stylesheet" type="text/css" href="https://www.example.com/my-enhancement-  
styles.css">
```

If you encounter errors, ensure that your originating server is sending the correct MIME type and encoding headers with the assets.

For example, the MIME and encoding types for remote scripts are: application/javascript;CHARSET=UTF-8.

The MIME and encoding type for remote stylesheets are: text/css;CHARSET=UTF-8.

## Output data and your task template

The following sections describe the output data from a custom labeling job, and when to consider using a post-annotation Lambda function.

### Output data

When your custom labeling job is finished, the data is saved in the Amazon S3 bucket specified when the labeling job was created. The data is saved in an output.manifest file.

#### Note

*LabelAttributeName* is a placeholder variable. In your output it is either the name of your labeling job, or the label attribute name you specify when you create the labeling job.

- **source** or **source-ref** – Either the string or an S3 URI workers were asked to label.
- **labelAttributeName** – A dictionary containing consolidated label content from the [post-annotation Lambda function](#). If a post-annotation Lambda function is not specified, this dictionary will be empty.
- **labelAttributeName-metadata** – Metadata from your custom labeling job added by Ground Truth.
- **worker-response-ref** – The S3 URI of the bucket where the data is saved. If a post-annotation Lambda function is specified this key-value pair will not present.

In this example the JSON object is formatted for readability, in the actual output file the JSON object is on a single line.

```
{  
    "source" : "This train is really late.",  
    "labelAttributeName" : {},  
    "labelAttributeName-metadata": { # These key values pairs are added by Ground Truth  
        "job_name": "test-labeling-job",  
        "type": "groundTruth/custom",  
        "human-annotated": "yes",  
        "creation_date": "2021-03-08T23:06:49.111000",  
        "worker-response-ref": "s3://amzn-s3-demo-bucket/test-labeling-job/annotations/  
        worker-response/iteration-1/0/2021-03-08_23:06:49.json"  
    }  
}
```

## Using a post annotation Lambda to consolidate the results from your workers

By default Ground Truth saves worker responses unprocessed in Amazon S3. To have more fine-grained control over how responses are handled, you can specify a *post-annotation Lambda function*. For example, a post-annotation Lambda function could be used to consolidate annotation if multiple workers have labeled the same data object. To learn more about creating post-annotation Lambda functions, see [Post-annotation Lambda](#).

If you want to use a post-annotation Lambda function, it must be specified as part of the [AnnotationConsolidationConfig](#) in a CreateLabelingJob request.

To learn more about how annotation consolidation works, see [Annotation consolidation](#).

## Adding automation with Liquid

Our custom template system uses [Liquid](#) for automation. It is an open source inline markup language. In Liquid, the text between single curly braces and percent symbols is an instruction or *tag* that performs an operation like control flow or iteration. Text between double curly braces is a variable or *object* that outputs its value.

The most common use of Liquid will be to parse the data coming from your input manifest file, and pull out the relevant variables to create the task. Ground Truth automatically generates the tasks unless a pre-annotation Lambda is specified. The `taskInput` object returned by Ground Truth or your [Pre-annotation Lambda](#) is the `task.input` object in your templates.

The properties in your input manifest are passed into your template as the `event.dataObject`.

### Example manifest data object

```
{  
  "source": "This is a sample text for classification",  
  "labels": [ "angry" , "sad" , "happy" , "inconclusive" ],  
  "header": "What emotion is the speaker feeling?"  
}
```

### Example sample HTML using variables

```
<crowd-classifier  
  name='tweetFeeling'  
  categories='{{ task.input.labels | to_json }}'  
  header='{{ task.input.header }}'>  
<classification-target>  
  {{ task.input.source }}  
</classification-target>
```

Note the addition of `| to_json` to the `labels` property above. That is a filter that turns the input manifest array into a JSON representation of the array. Variable filters are explained in the next section.

The following list includes two types of Liquid tags that you may find useful to automate template input data processing. If you select one of the following tag-types, you will be redirected to the Liquid documentation.

- [Control flow](#): Includes programming logic operators like `if/else`, `unless`, and `case/when`.

- [Iteration](#): Enables you to run blocks of code repeatedly using statements like for loops.

For an example of an HTML template that uses Liquid elements to create a for loop, see [translation-review-and-correction.liquid.html](#) in GitHub.

For more information and documentation, visit the [Liquid homepage](#).

## Variable filters

In addition to the standard [Liquid filters](#) and actions, Ground Truth offers a few additional filters. Filters are applied by placing a pipe (|) character after the variable name, then specifying a filter name. Filters can be chained in the form of:

### Example

```
{{ <content> | <filter> | <filter> }}
```

## Autoescape and explicit escape

By default, inputs will be HTML escaped to prevent confusion between your variable text and HTML. You can explicitly add the escape filter to make it more obvious to someone reading the source of your template that the escaping is being done.

### escape\_once

escape\_once ensures that if you've already escaped your code, it doesn't get re-escaped on top of that. For example, so that & doesn't become &&;

### skip\_autoescape

skip\_autoescape is useful when your content is meant to be used as HTML. For example, you might have a few paragraphs of text and some images in the full instructions for a bounding box.

#### Use skip\_autoescape sparingly

The best practice in templates is to avoid passing in functional code or markup with skip\_autoescape unless you are absolutely sure you have strict control over what's being passed. If you're passing user input, you could be opening your workers up to a Cross Site Scripting attack.

## to\_json

to\_json will encode what you feed it to JSON (JavaScript Object Notation). If you feed it an object, it will serialize it.

## grant\_read\_access

grant\_read\_access takes an S3 URI and encodes it into an HTTPS URL with a short-lived access token for that resource. This makes it possible to display to workers the photo, audio, or video objects stored in S3 buckets that are not otherwise publicly accessible.

## s3\_presign

The s3\_presign filter works the same way as the grant\_read\_access filter. s3\_presign takes an Amazon S3 URI and encodes it into an HTTPS URL with a short-lived access token for that resource. This makes it possible to display photo, audio, or video objects stored in S3 buckets that are not otherwise publicly accessible to workers.

## Example of the variable filters

### Input

```
auto-escape: {{ "Have you read 'James & the Giant Peach'?" }}
explicit escape: {{ "Have you read 'James & the Giant Peach'?" | escape }}
explicit escape_once: {{ "Have you read 'James & the Giant Peach'?" | escape_once }}
skip_autoescape: {{ "Have you read 'James & the Giant Peach'?" | skip_autoescape }}
to_json: {{ js0bject | to_json }}
grant_read_access: {{ "s3://amzn-s3-demo-bucket/myphoto.png" | grant_read_access }}
s3_presign: {{ "s3://amzn-s3-demo-bucket/myphoto.png" | s3_presign }}
```

### Example

### Output

```
auto-escape: Have you read 'James & the Giant Peach'?
explicit escape: Have you read 'James & the Giant Peach'?
explicit escape_once: Have you read 'James & the Giant Peach'?
skip_autoescape: Have you read 'James & the Giant Peach'?
to_json: { "point_number": 8, "coords": [ 59, 76 ] }
grant_read_access: https://s3.amazonaws.com/amzn-s3-demo-bucket/myphoto.png?<access
token and other params>
```

```
s3_presign: https://s3.amazonaws.com/amzn-s3-demo-bucket/myphoto.png?<access token and  
other params>
```

## Example of an automated classification template.

To automate the simple text classification sample, replace the tweet text with a variable.

The text classification template is below with automation added. The changes/additions are highlighted in bold.

```
<script src="https://assets.crowd.aws/crowd-html-elements.js"></script>
<crowd-form>
  <crowd-classifier
    name="tweetFeeling"
    categories="['positive', 'negative', 'neutral', 'cannot determine']"
    header="Which term best describes this tweet?"
  >
    <classification-target>
      {{ task.input.source }}
    </classification-target>

    <full-instructions header="Analyzing a sentiment">
      Try to determine the feeling the author
      of the tweet is trying to express.
      If none seem to match, choose "other."
    </full-instructions>

    <short-instructions>
      Pick the term best describing the sentiment
      of the tweet.
    </short-instructions>

  </crowd-classifier>
</crowd-form>
```

The tweet text in the prior sample is now replaced with an object. The `entry.taskInput` object uses `source` (or another name you specify in your pre-annotation Lambda) as the property name for the text, and it is inserted directly in the HTML by virtue of being between double curly braces.

## Processing data in a custom labeling workflow with AWS Lambda

In this topic, you can learn how to deploy optional [AWS Lambda](#) functions when creating a custom labeling workflow. You can specify two types of Lambda functions to use with your custom labeling workflow.

- *Pre-annotation Lambda*: This function pre-processes each data object sent to your labeling job prior to sending it to workers.
- *Post-annotation Lambda*: This function processes the results once workers submit a task. If you specify multiple workers per data object, this function may include logic to consolidate annotations.

If you are a new user of Lambda and Ground Truth, we recommend that you use the pages in this section as follows:

1. First, review [Using pre-annotation and post-annotation Lambda functions](#).
2. Then, use the page [Add required permissions to use AWS Lambda with Ground Truth](#) to learn about security and permission requirements to use your pre-annotation and post-annotation Lambda functions in a Ground Truth custom labeling job.
3. Next, you need to visit the Lambda console or use Lambda's APIs to create your functions. Use the section [Create Lambda functions using Ground Truth templates](#) to learn how to create Lambda functions.
4. To learn how to test your Lambda functions, see [Test pre-annotation and post-annotation Lambda functions](#).
5. After you create pre-processing and post-processing Lambda functions, select them from the **Lambda functions** section that comes after the code editor for your custom HTML in the Ground Truth console. To learn how to use these functions in a CreateLabelingJob API request, see [Create a Labeling Job \(API\)](#).

For a custom labeling workflow tutorial that includes example pre-annotation and post-annotation Lambda functions, see [Demo template: Annotation of images with crowd-bounding-box](#).

### Topics

- [Using pre-annotation and post-annotation Lambda functions](#)
- [Add required permissions to use AWS Lambda with Ground Truth](#)

- [Create Lambda functions using Ground Truth templates](#)
- [Test pre-annotation and post-annotation Lambda functions](#)

## Using pre-annotation and post-annotation Lambda functions

Use these topics to learn about the syntax of the requests sent to pre-annotation and post-annotation Lambda functions, and the required response syntax that Ground Truth uses in custom labeling workflows.

### Topics

- [Pre-annotation Lambda](#)
- [Post-annotation Lambda](#)

### Pre-annotation Lambda

Before a labeling task is sent to the worker, a optional pre-annotation Lambda function can be invoked.

Ground Truth sends your Lambda function a JSON formatted request to provide details about the labeling job and the data object.

The following are 2 example JSON formatted requests.

Data object identified with "source-ref"

```
{  
    "version": "2018-10-16",  
    "labelingJobArn": <labelingJobArn>  
    "dataObject" : {  
        "source-ref": <s3Uri>  
    }  
}
```

Data object identified with "source"

```
{  
    "version": "2018-10-16",  
    "labelingJobArn": <labelingJobArn>  
    "dataObject" : {  
        "source": <string>  
    }  
}
```

```
}
```

The following list contains the pre-annotation request schemas. Each parameter is described below.

- **version (string)**: This is a version number used internally by Ground Truth.
- **labelingJobArn (string)**: This is the Amazon Resource Name, or ARN, of your labeling job. This ARN can be used to reference the labeling job when using Ground Truth API operations such as `DescribeLabelingJob`.
- The **dataObject (JSON object)**: The key contains a single JSON line, either from your input manifest file or sent from Amazon SNS. The JSON line objects in your manifest can be up to 100 kilobytes in size and contain a variety of data. For a very basic image annotation job, the `dataObject` JSON may just contain a `source-ref` key, identifying the image to be annotated. If the data object (for example, a line of text) is included directly in the input manifest file, the data object is identified with `source`. If you create a verification or adjustment job, this line may contain label data and metadata from the previous labeling job.

The following tabbed examples show examples of a pre-annotation request. Each parameter in these example requests is explained below the tabbed table.

#### Data object identified with "source-ref"

```
{
    "version": "2018-10-16",
    "labelingJobArn": "arn:aws:sagemaker:us-west-2:111122223333:labeling-job/
<labeling_job_name>"
    "dataObject" : {
        "source-ref": "s3://input-data-bucket/data-object-file-name"
    }
}
```

#### Data object identified with "source"

```
{
    "version": "2018-10-16",
    "labelingJobArn": "arn:aws:sagemaker:<aws_region>:111122223333:labeling-job/
<labeling_job_name>"
```