

Cloud Guru - 8 - Implementation and Operations

2 types of usage

			
		Offline Usage	Online Usage
What	Make inferences on datasets in batch and return results as a set.	Make inferences on demand as the model is called and return results immediately.	
Why	Entire dataset is needed for inferences; Pre-process data before using as an input for another model.		Need instance response when endpoint is called via an app or service.
When	<ul style="list-style-type: none">- Predictive models with large historic dataset inputs- Feature engineering for a follow-on model		<ul style="list-style-type: none">- Real-time fraud detection- Autonomous machines

Type of deployments:

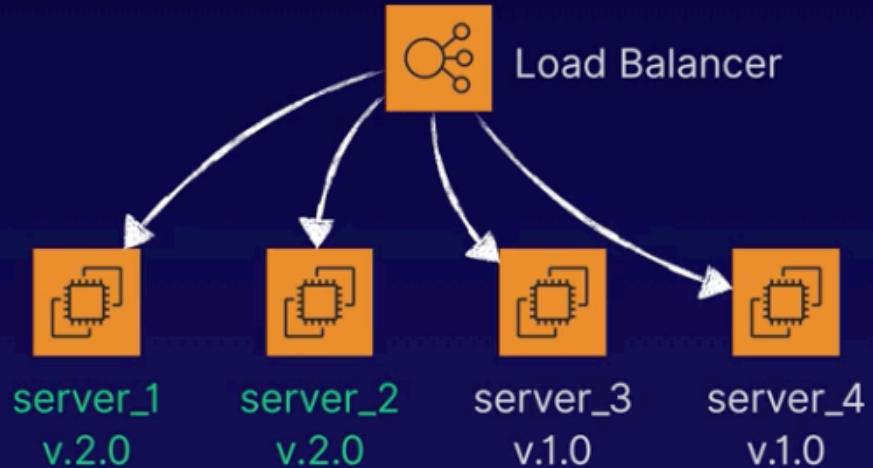
		“Big Bang”	Phased Rollout	Parallel Adoption
Time				
Risk				
Cost	function(Risk, Time)	function(Risk, Time)	function(Risk, Time)	

* Sometimes, risk is amplified in a Parallel Adoption due to concurrency such as data synchronization issues, multiple processes, temporary integrations, etc.



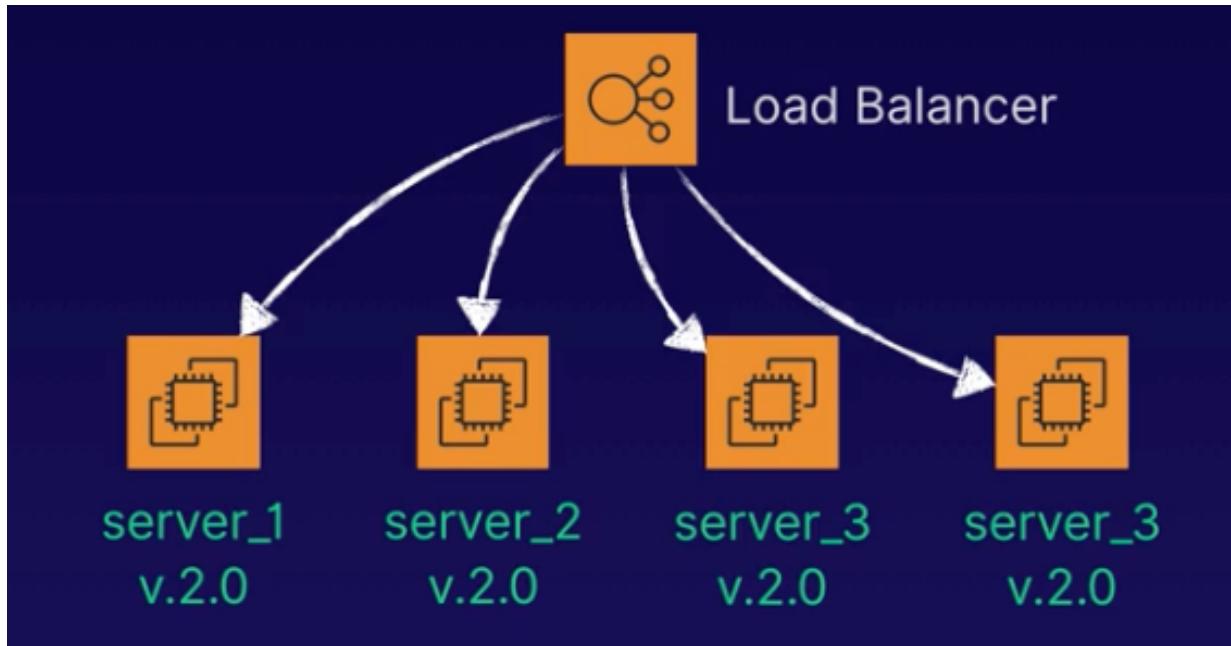
Rolling Deployment

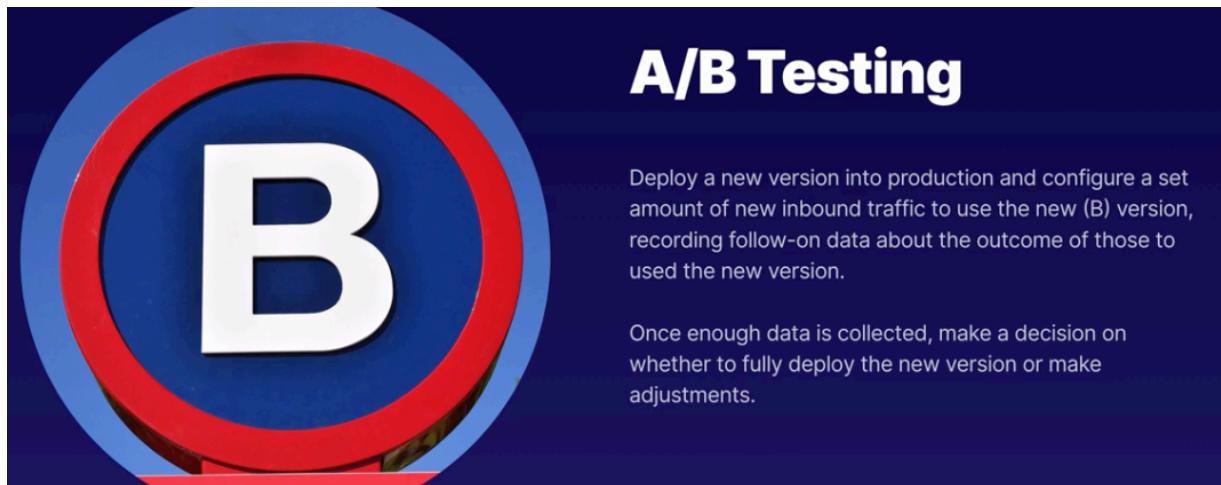
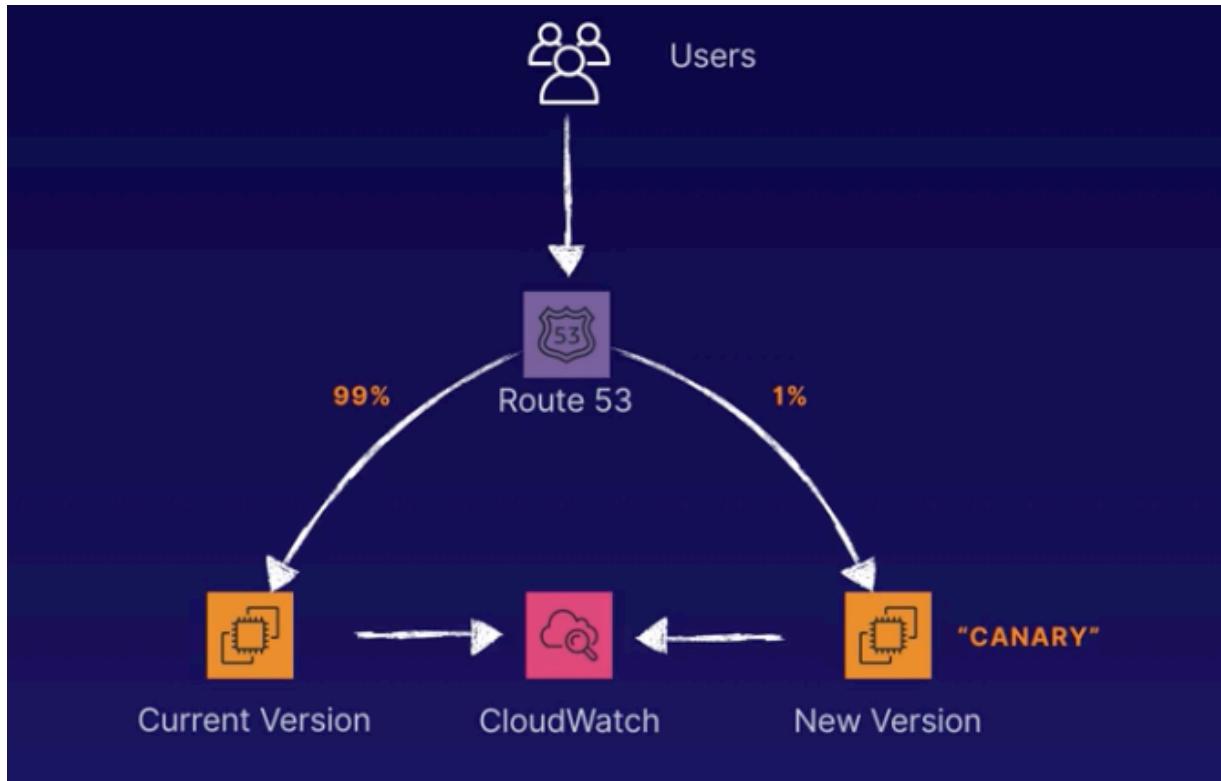
Rather than upgrade all resources at once, the upgrade is done one by one to minimize downtime.

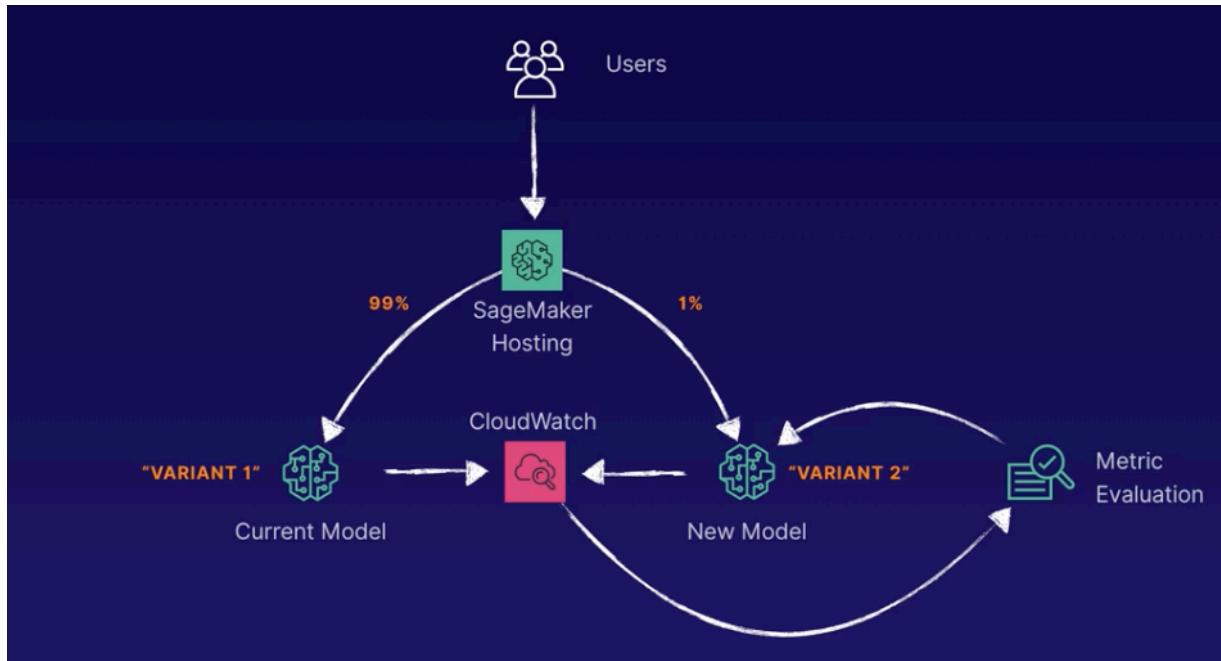


Multiple versions in production at the same time.

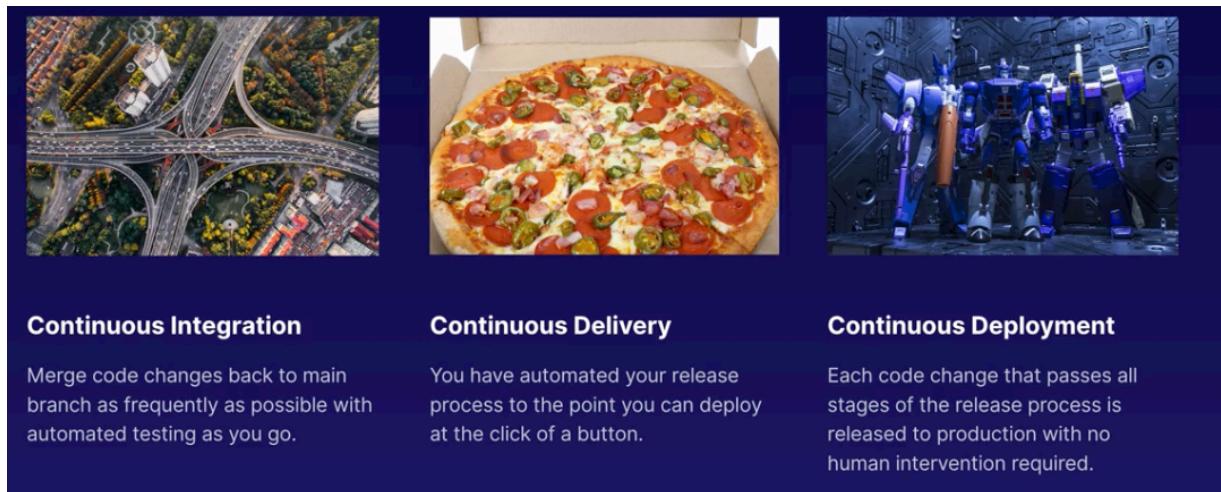
Eventually:



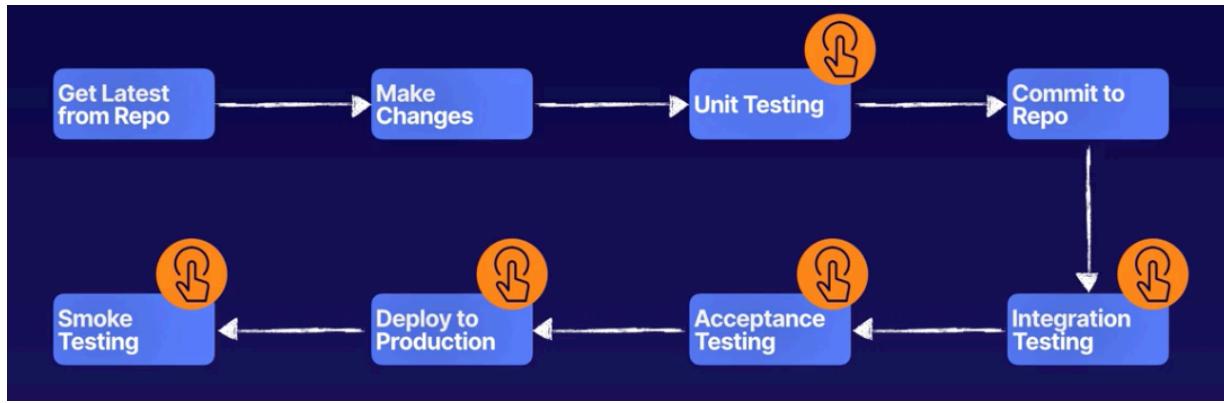




CI/CD pipeline

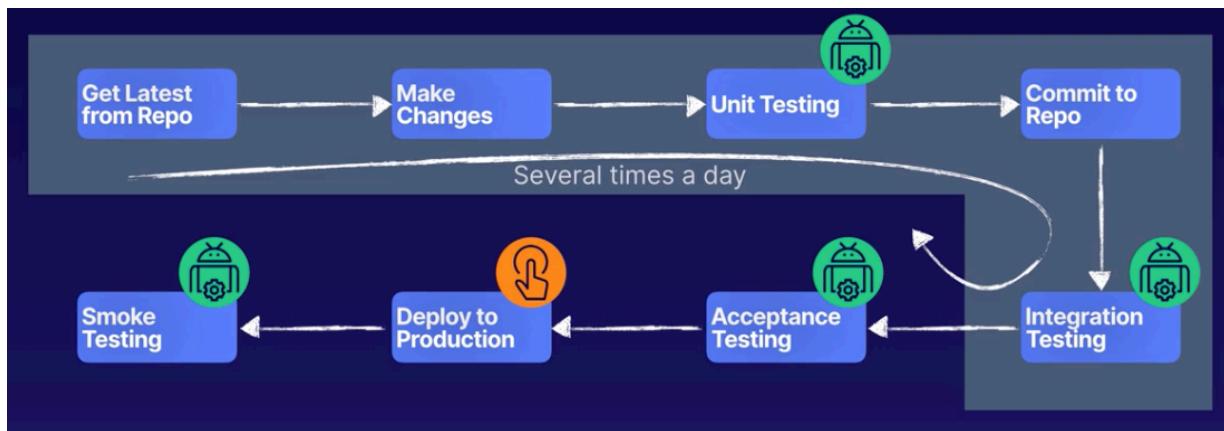


Typical Development Lifecycle:
Involves a human for different gates



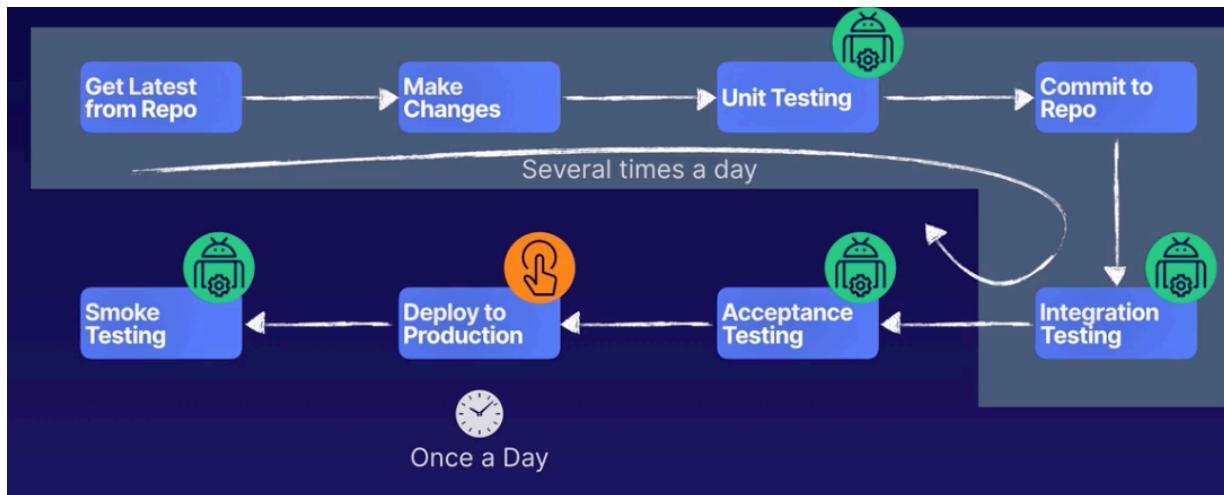
Continuous integration:

With each check-in, integration tests to make sure nothing breaks

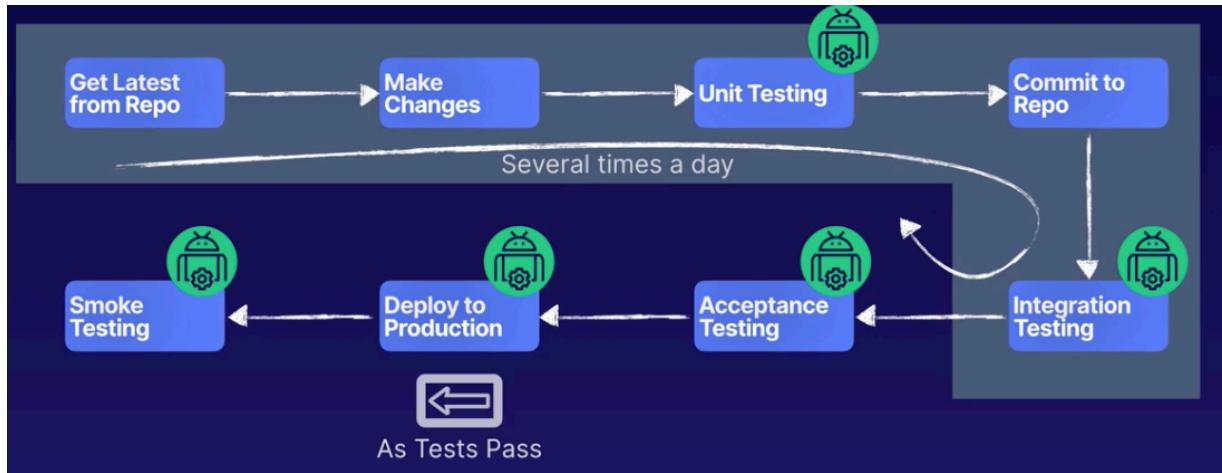


Continuous delivery:

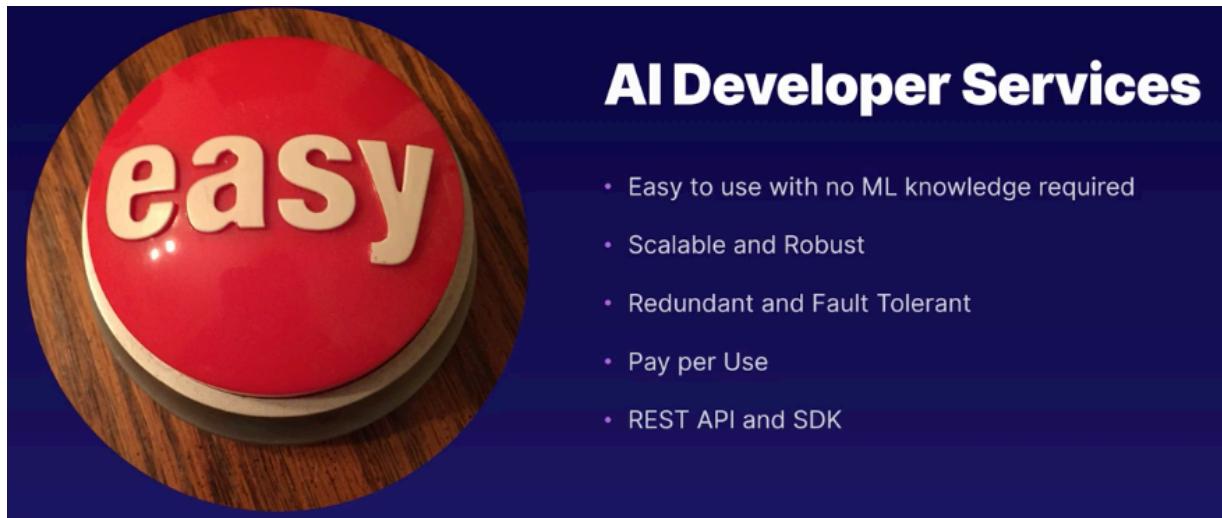
Same process, but can release once per day



Continuous Deployment:



AI Developer services



	Amazon Comprehend		Amazon Forecast
What	Natural Language Processing (NLP) service that finds insight and relationships within text.	Combines time-series data with other variables to deliver highly accurate forecasts.	
When	Sentiment analysis of social media posts	Forecast seasonal demand for a specific color of shirt.	

	Amazon Lex		Amazon Personalize
What	Build conversational interfaces that can understand the intent and context of natural speech.	Recommendation engine as a service based on demographic and behavioral data.	
When	Create a customer service chatbot to automatically handle routine requests.	Provide potential upsell products at checkout during a web transaction.	

	Amazon Polly		Amazon Rekognition
What	Text-to-Speech service supporting multiple languages, accents and voices.	Image and video analysis to parse and recognize objects, people, activities and facial expressions.	
When	Provide dynamically generated personalized voice response for inbound callers.	Provide an additional form of employee authentication through facial recognition as they scan an access badge.	



Amazon
Textract



Amazon
Transcribe

What

Extract text, context and metadata from scanned documents

Speech-to-Text as a service

When

Automatically digitize and process physical paper forms

Automatically create transcripts of recorded presentations.



Amazon
Translate

What

Translate text to and from many different languages

When

Dynamically create localized web content for users based on their geography.

Console experience

✓ç

Screenshot of the AWS Amazon Comprehend service showing the Insights interface. The sidebar lists various analysis options like Real-time analysis, Job management, Customization, and Comprehend Medical. The main pane displays entities identified in an analyzed text sample about Amazon.com.

Analyzed text:

Amazon.com, Inc. is located in Seattle, WA and was founded July 5th, 1994 by Jeff Bezos, allowing customers to buy everything from books to blenders. Seattle is north of Portland and south of Vancouver, BC. Other notable Seattle-based companies are Starbucks and Boeing.

Results:

Entity	Category	Confidence
Amazon.com, Inc	Organization	0.90
Seattle, WA	Location	0.89
July 5th, 1994	Date	0.99+

Screenshot of the AWS Amazon Rekognition service showing the Object and scene detection demo. The sidebar lists various detection options like Metrics, Demos, and Video Demos. The main pane shows a street scene with detected objects and a confidence score table.

Object and scene detection:

Rekognition automatically labels objects, concepts and scenes in your images, and provides a confidence score.

Done with the demo? [Learn more](#)

Results:

Transportation	98.8 %
Automobile	98.8 %
Car	98.8 %
Vehicle	98.8 %
Human	98.3 %
Person	98.3 %

Choose a sample image: 

Use your own image: Image must be .jpg or .png format and no larger than 5MB. Your image isn't stored.

Upload or drag and drop **Go**

Next lesson

Amazon SageMaker Deployments

Inferences - deployment

The screenshot shows the Amazon SageMaker console interface. On the left is a navigation sidebar with the following menu items:

- Dashboard
- Search^{Beta}
- Ground Truth
 - Labeling jobs
 - Labeling datasets
 - Labeling workforces
- Notebook
 - Notebook instances
 - Lifecycle configurations
 - Git repositories
- Training
 - Algorithms
 - Training jobs
 - Hyperparameter tuning jobs
- Inference
 - Compilation jobs
 - Model packages
 - Models
 - Endpoint configurations
 - Endpoints
 - Batch transform jobs
- AWS Marketplace

The main content area contains four sections:

- Ground Truth**: Set up and manage labeling jobs for training datasets using active learning and human labeling.
- Notebook**: Access a managed Jupyter Notebook environment.
- Training**: Train and tune models.
- Inference**: Package and deploy your machine learning models at scale.

2 types of deployment: Offline vs Online

The diagram illustrates two types of usage:

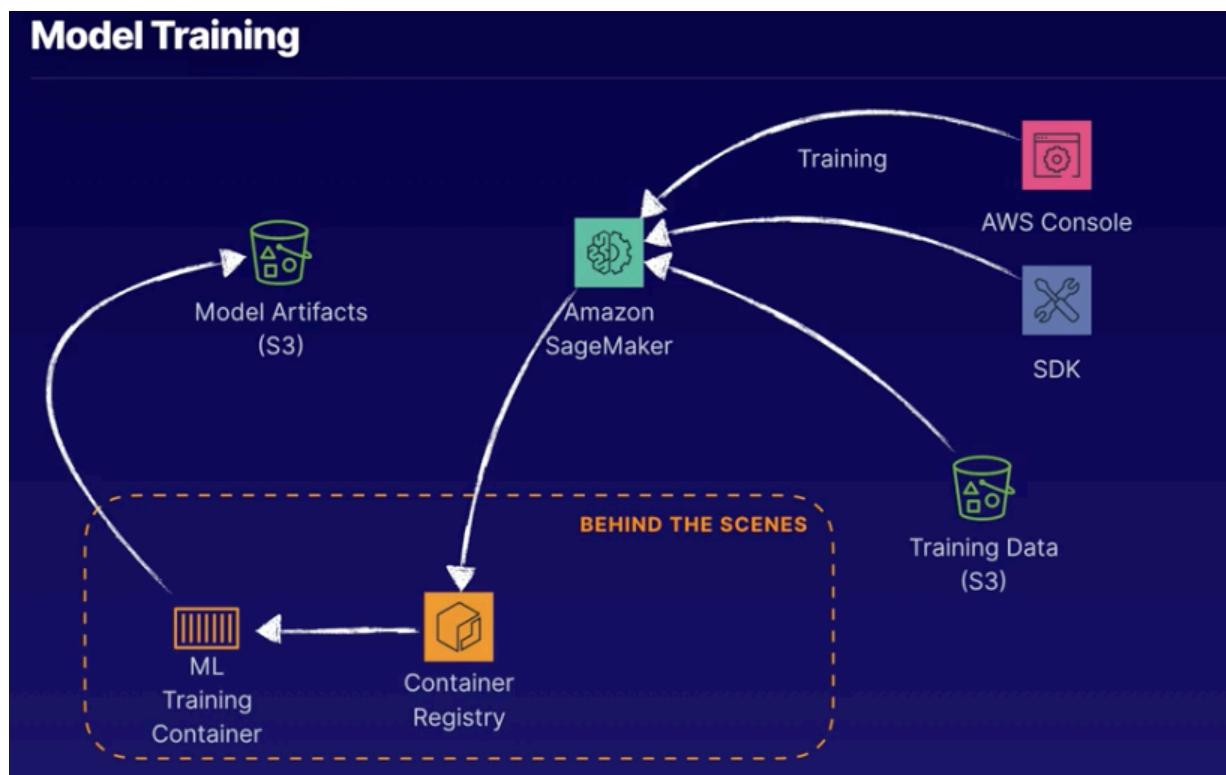
- Offline Usage** (represented by a shield icon): Asynchronous or Batch. Generate predictions for a whole set of data all at once. Method: SageMaker Batch Transform. Input Format: Varies depending on Algorithm. Output Format: Varies depending on Algorithm.
- Online Usage** (represented by a radio tower icon): Synchronous or Real-Time. Generate low-latency predictions. Method: SageMaker Hosting Services. Input Format: Varies depending on Algorithm. Output Format: JSON string.

Usage	Asynchronous or Batch	Synchronous or Real-Time
When	Generate predictions for a whole set of data all at once.	Generate low-latency predictions.
Method	SageMaker Batch Transform	SageMaker Hosting Services
Input Format	Varies depending on Algorithm	Varies depending on Algorithm
Output Format	Varies depending on Algorithm	JSON string

Steps:

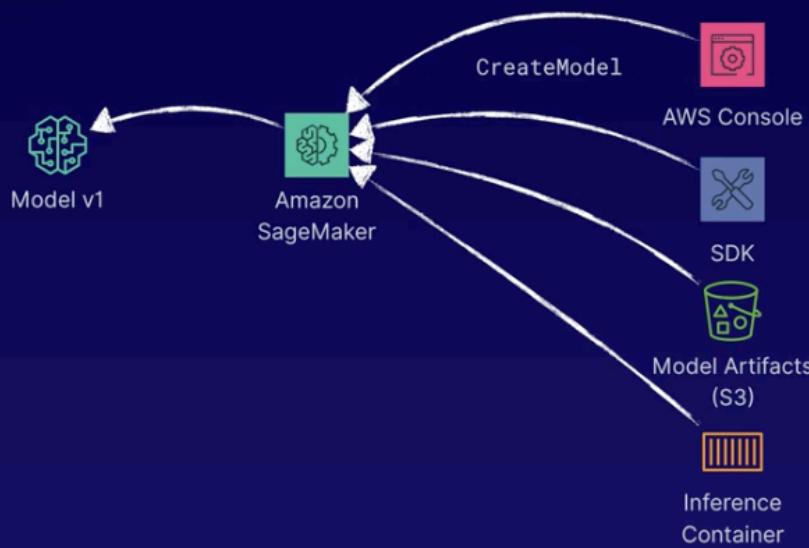
1	2	3
Create a Model	Create an Endpoint Configuration	Create an Endpoint
This is the inference engine that will provide predictions for your endpoint.	Defines the model to use, inference instance type, instance count, variant name and weight. Also called a Production Variant.	Publishes the model via the endpoint configuration to be called by the SageMaker API <code>InvokeEndpoint()</code> method.

Model Training



Model Creation (once training is finished)

Model Creation



Console

Container input options

- Provide model artifacts and inference image location
Use this for models trained using built-in algorithms, BYO algorithms, or models trained outside Amazon SageMaker.
- Use a model package resource
Use this for model packages that contain inference images and artifacts from AWS Marketplace subscribed algorithms.
- Use a model package subscription from AWS Marketplace
Use this for model packages published by vendors from AWS Marketplace.

Provide model artifacts and inference image

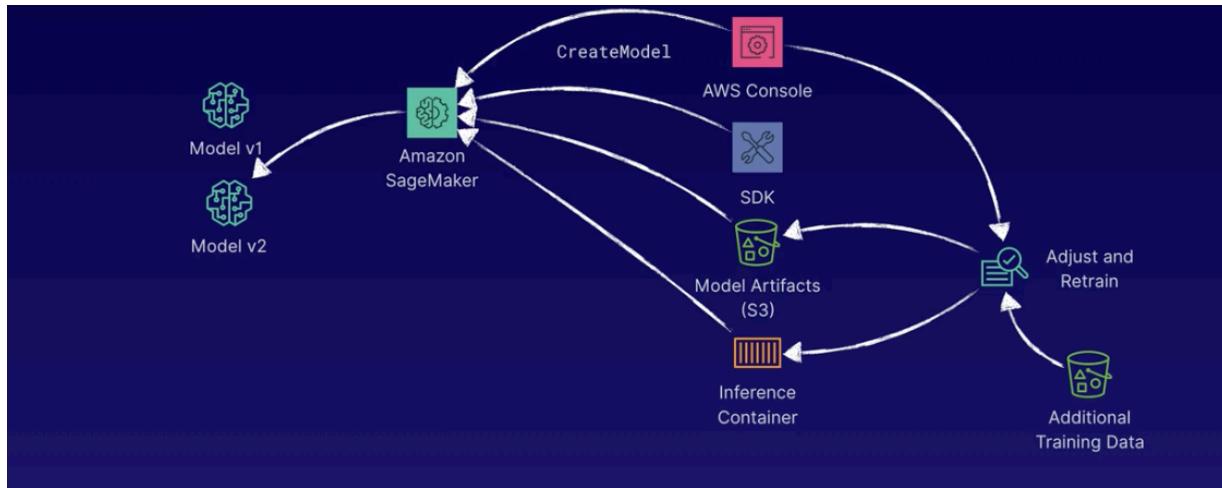
Location of inference code image
Type the registry path where the inference code image is stored in Amazon ECR.
`aws_account_id.dkr.ecr.region.amazonaws.com/repository[:tag] or [@digest]`

Location of model artifacts - optional
Type the URL where model artifacts are stored in S3.
`s3://bucket/path-to-your-data/`
The path must point to a single gzip compressed tar archive (.tar.gz suffix).

Container host name - optional
Type the DNS host name for the container.
`my-inference-container`
Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

- Specify model artifacts generated via the training process
- Use a model package from AWS Marketplace

After model has been on prod for a while, we can RE-train it, and create a new version of that model

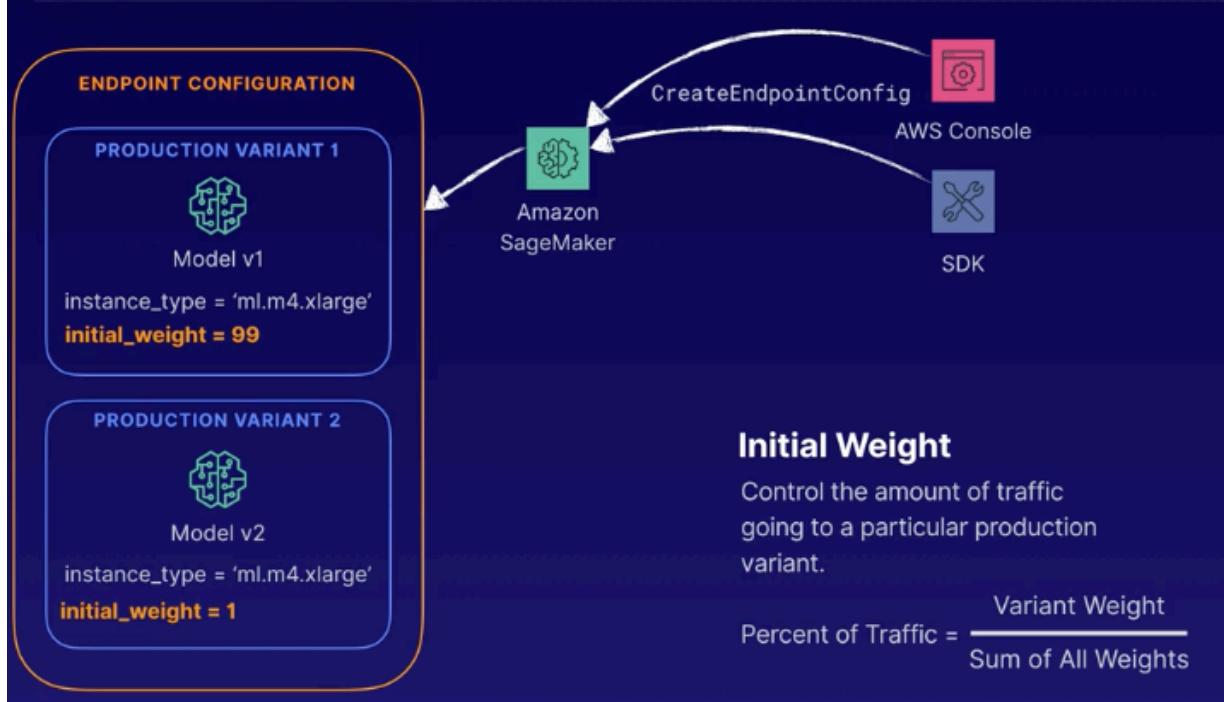


Amazon SageMaker > Models

Models				
		Create endpoint	Create endpoint configuration	Actions ▾
Create model				
<input type="text" value="Search models"/> < 1 > ⚙				
Name	ARN	Creation time		
linear-learner-v2	arn:aws:sagemaker:us-east-1: XXXXXXXXXX :model/linear-learner-v2	Feb 21, 2019 20:09 UTC		
linear-learner-v1	arn:aws:sagemaker:us-east-1: XXXXXXXXXX :model/linear-learner-v1	Feb 21, 2019 20:09 UTC		

Endpoint configuration - to enable one or multiple production variant(s) of a model
 We can define the weight to control what version gets how much traffic

Endpoint Configuration

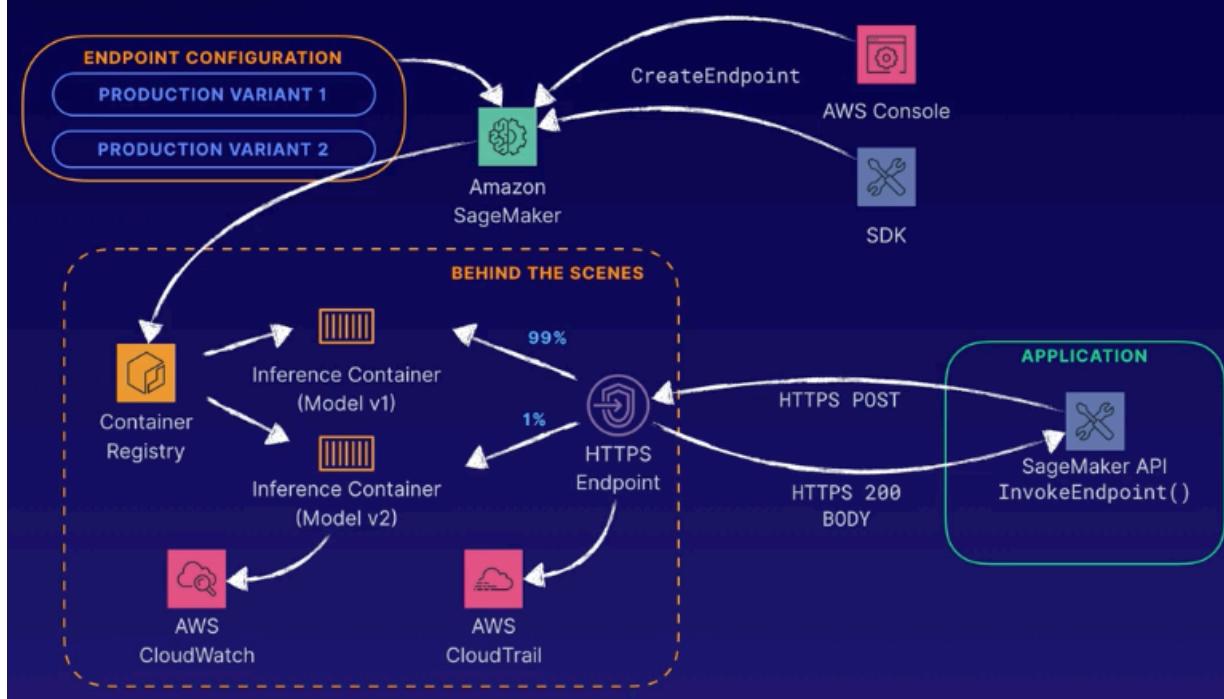


Notes for Create endpoint:

- **end point** traffic is logged into **CloudDtail**
- **Container** logs is logged into **CloudWatch**

InvokeEndpoint() allows us to get the inferences

Create Endpoint

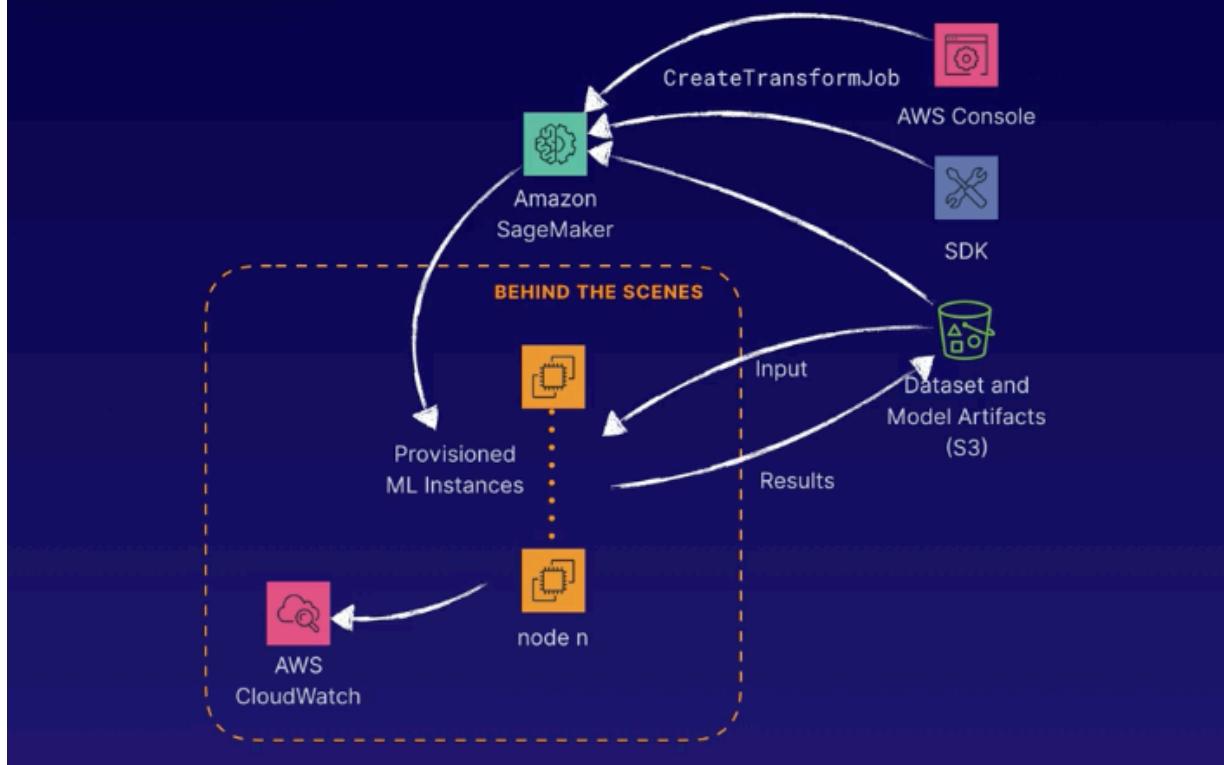


Batch Transform starts in a similar manner

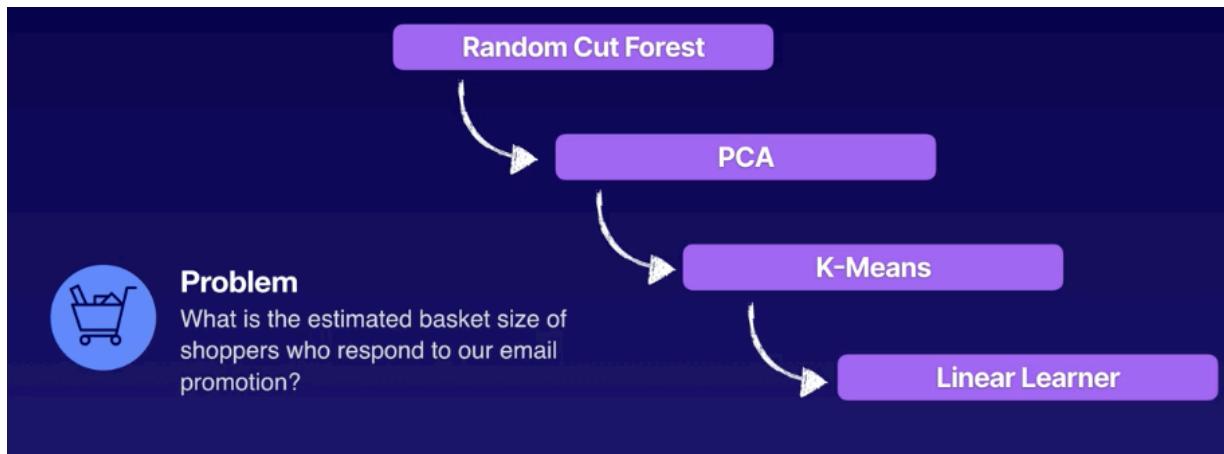
SM provisions batch resources. Once inference jobs are complete, results get dropped into S3

This is common to evaluate the whole dataset, like in Forecasting

Batch Transform



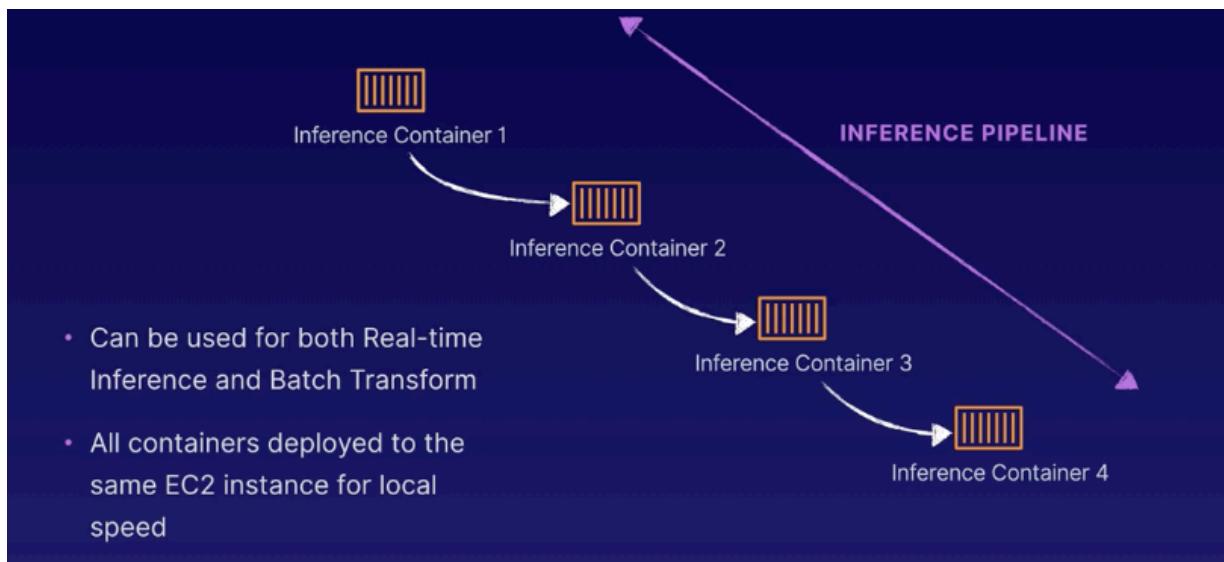
Inference pipelines





Inference Pipelines

A SageMaker model composed of a sequence of two to five containers which can process data as a flow. These can be built-in algorithms or your own custom algorithms in Docker containers.



SageMaker Neo

Neo is a compiler and runtime component that allows us to optimize models for specific architectures => raspberry pie, IoT device,...



SageMaker Neo

Enables a simplified way to optimize machine learning models for a variety of computing architectures such as ARM, Intel and nVidia processors.

Consists of a compiler to convert the machine learning model into an optimized binary and a runtime to execute the model on the target architecture.

If we used a model with MXnet, we can use Neo to optimize this model to run on an IoT device or Nvidia cud platform without changing anything about the model



Elastic Inference = Turbo boos

Elastic Inference

Speeds up throughput and decreases latency of real-time inferences deployed on SageMaker Hosted Services using only CPU-based instances but much more cost-effective than a full GPU instance.

Must be configured when you create a deployable model and EI is not available for all algorithms yet.

SageMaker supports **Automatic Scaling**

Automatic Scaling

Dynamically add and remove instances to a production variant based on changes in workload.

You define and apply a scaling policy that uses a CloudWatch metric and target value such as `InvocationsPerInstance`.

We define a Target Metric we want SM to watch

Configure variant automatic scaling

Variant name: variant-name-1

Instance type: ml.m4.xlarge

Current instance count: 1

Elastic Inference: -

Current weight: 1

Minimum instance count: 1

Maximum instance count: 1

IAM role: AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint

Built-in scaling policy: SageMakerEndpointInvocationScalingPolicy

Target metric: SageMakerVariantInvocationsPerInstance

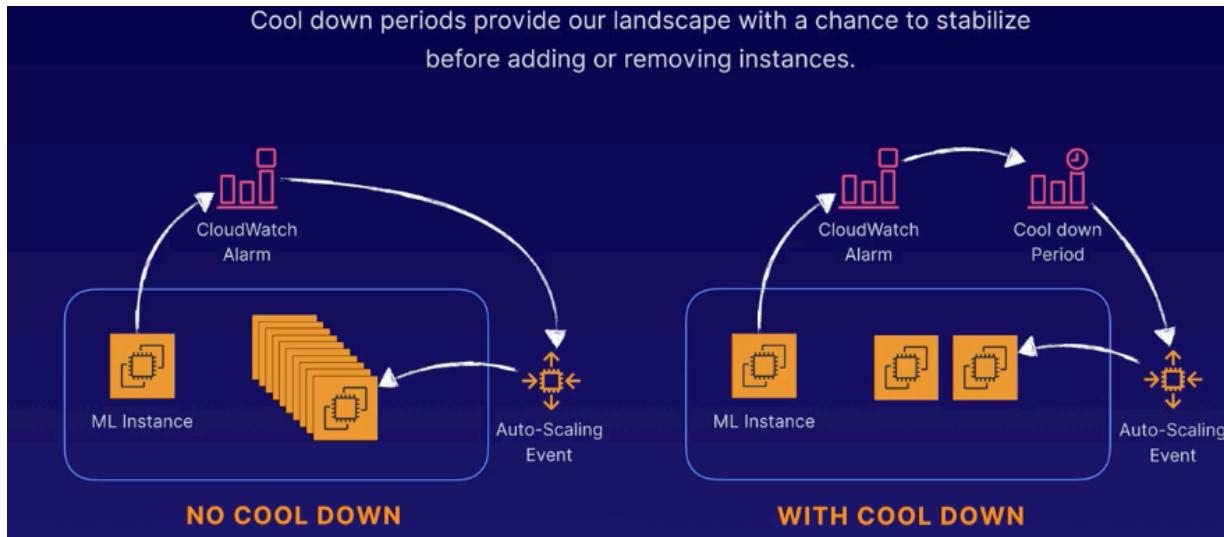
Target value: 1

Scale in cool down (seconds) - optional: 300

Scale out cool down (seconds) - optional: 300

Disable scale in

Cool Down period - to smooth scale in and scale up (buffer for newly introduced instances before removing them or adding more)



SM recommends we set up multiple instances for high availability

High Availability with SageMaker

A CLOUD



Other ML Deployment options

Other Deployment Options



Amazon Elastic Container Service



Amazon EC2

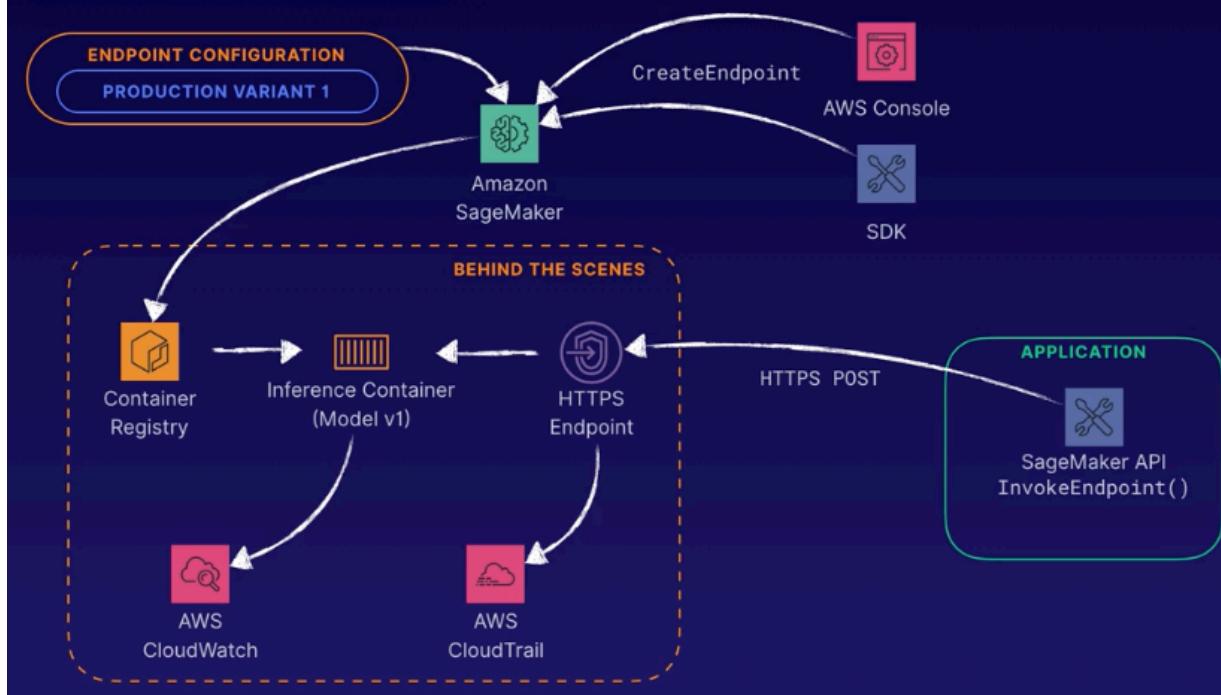


Amazon Elastic Map Reduce



On-Premises

SageMaker Hosting Services



To deploy to other places, it starts the same way:

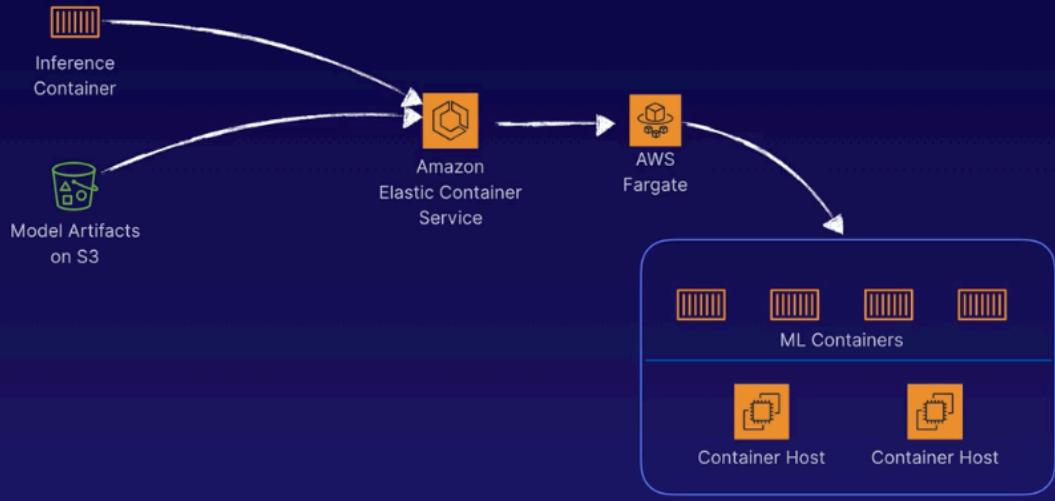
Creating a Model



ECS deployment:

Deploying with ECS

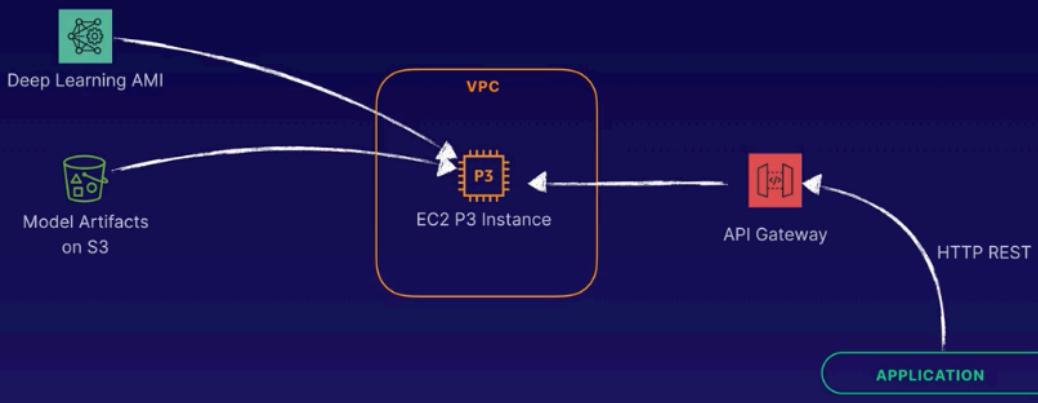
A CLOUD GURU



EC2 deployment:

Deploying with EC2

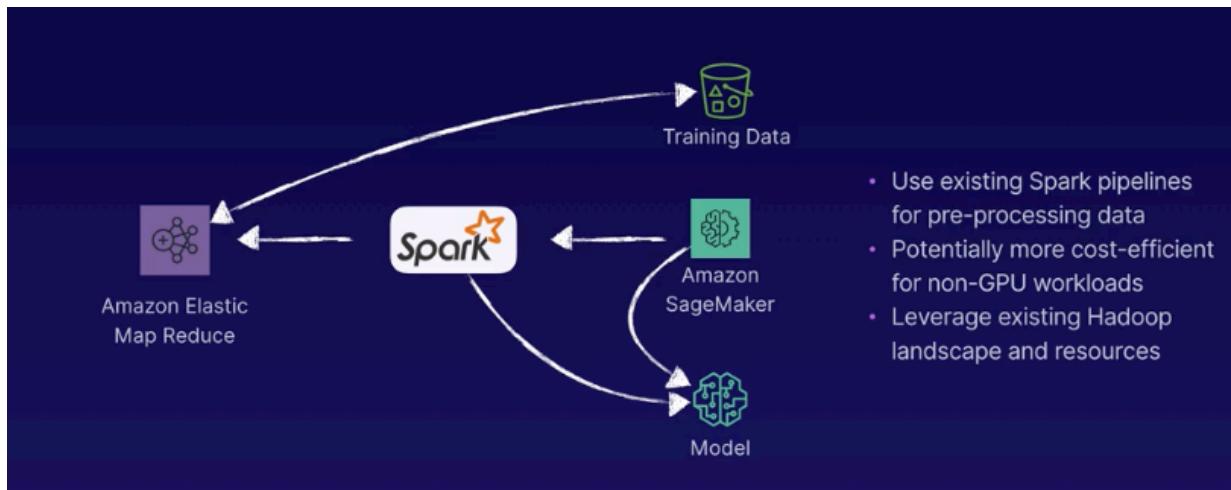
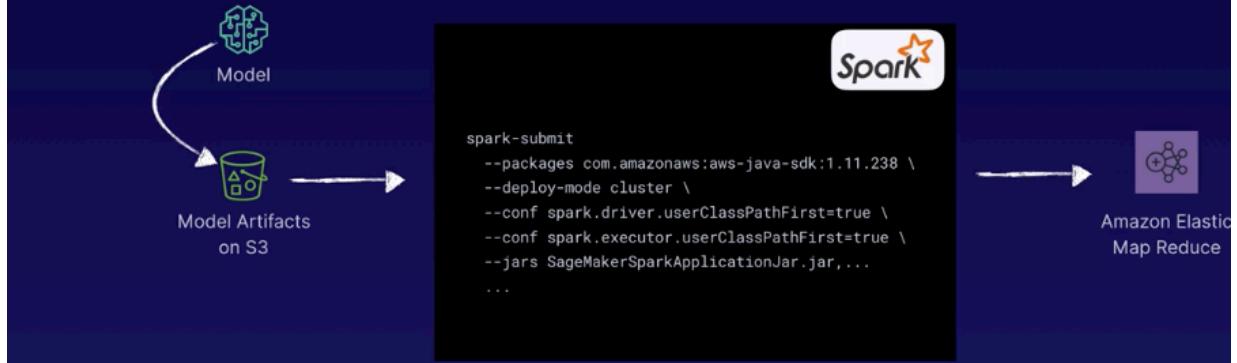
A CLOUD GURU



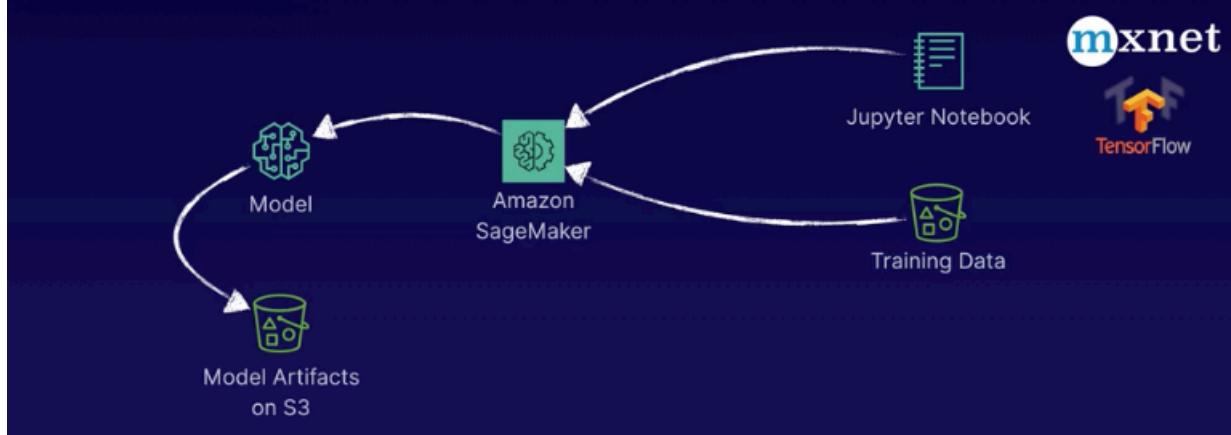
Deploying to EMR

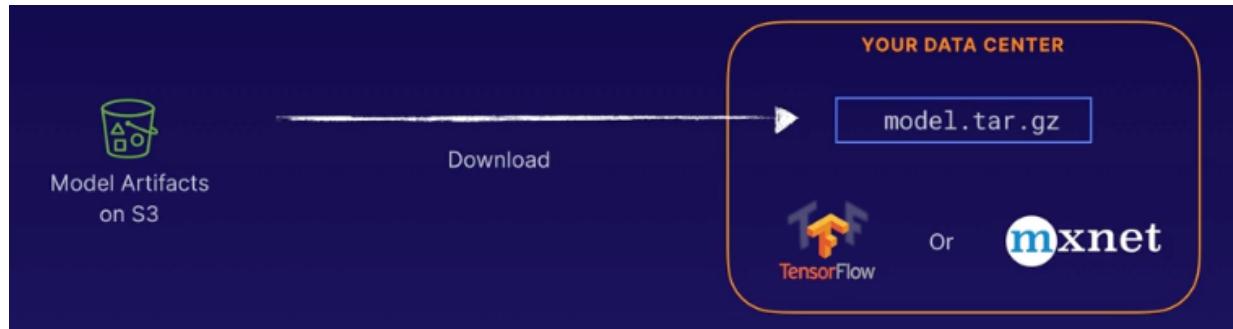
Deploying with EMR (Spark)

A CLOUD



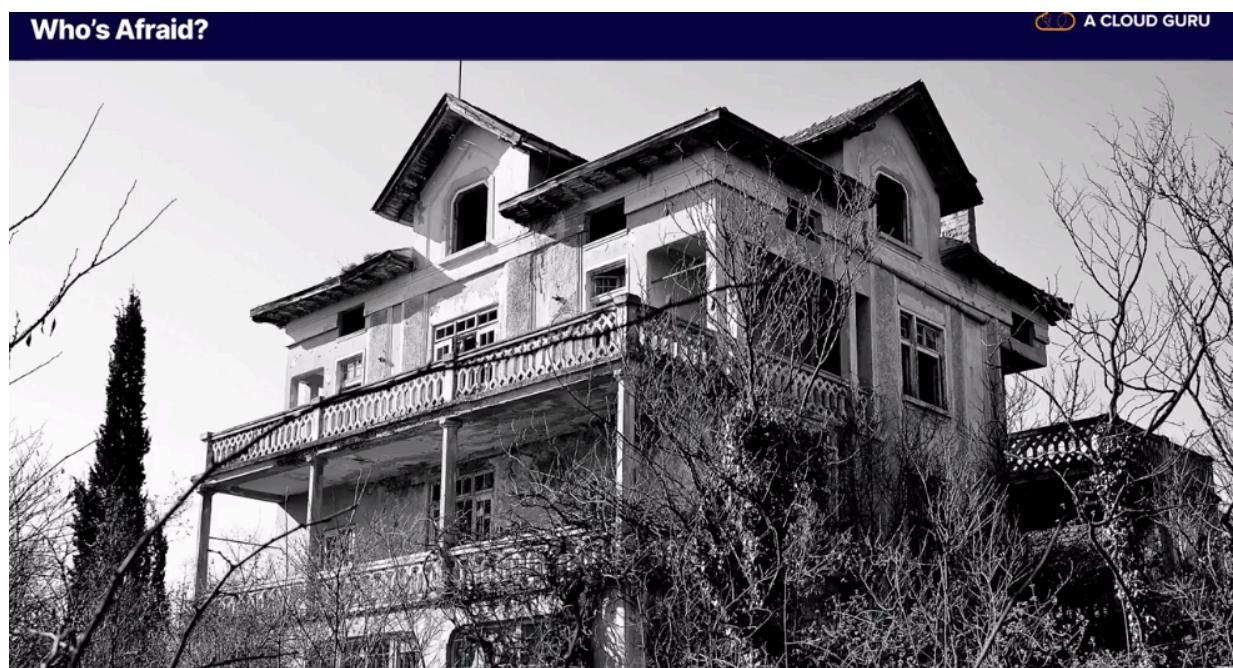
Deploying Locally

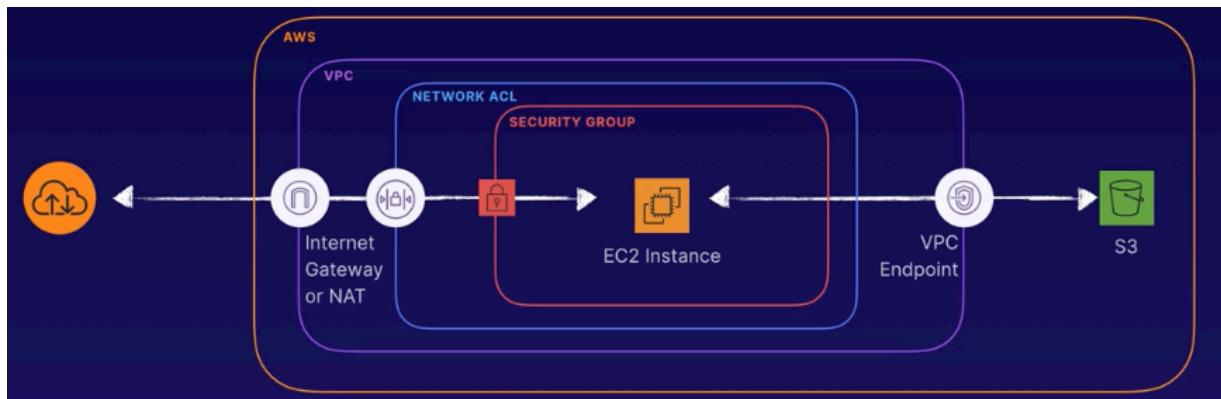
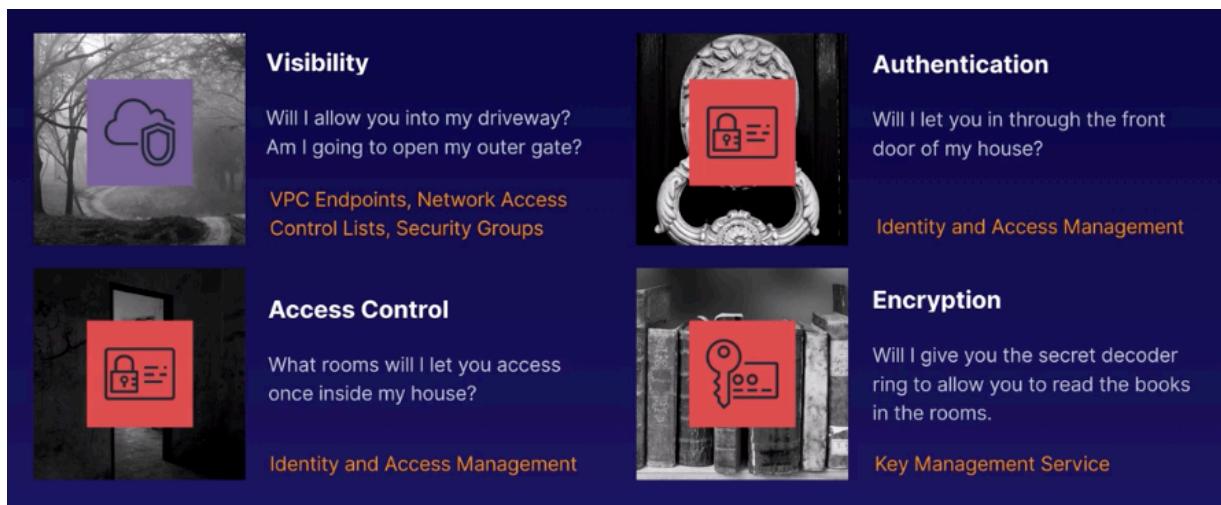
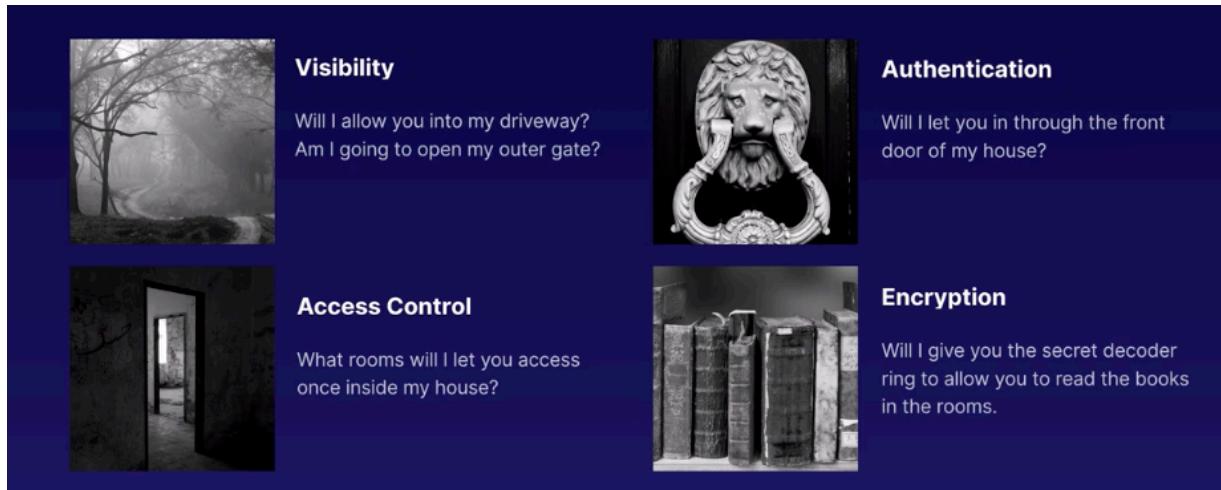




Security

Using scary house as an analogy to security





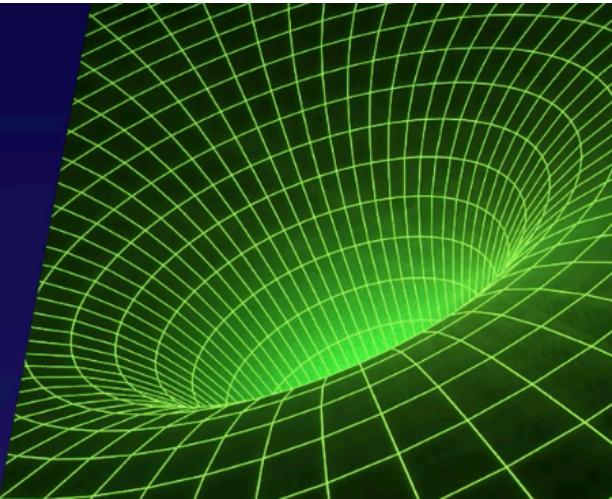
VPC Endpoint

A way to access other AWS services using the AWS network without having to send traffic over the public internet.

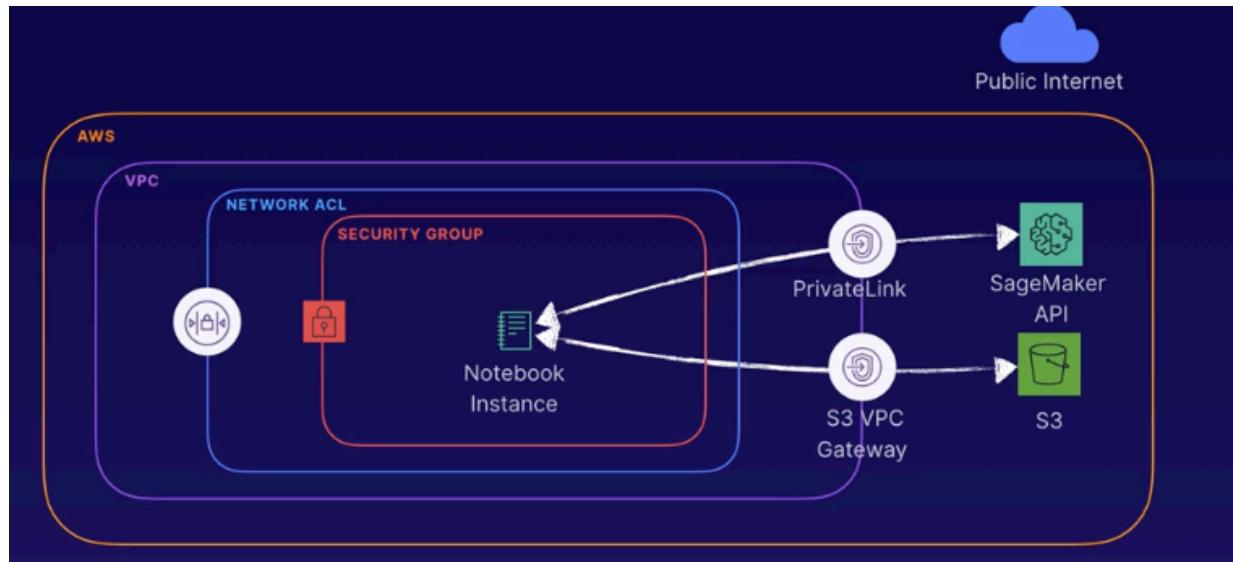
VPC Endpoints increase security and allow AWS services to communicate privately and reliably.

There are two types of VPC Endpoints: Interface Endpoints and Gateway Endpoints. Gateway Endpoints are only available for S3 and DynamoDB and differ because they use a route table entry versus a private DNS entry that the Interface Endpoints use. In the end, they both fulfill the same function.

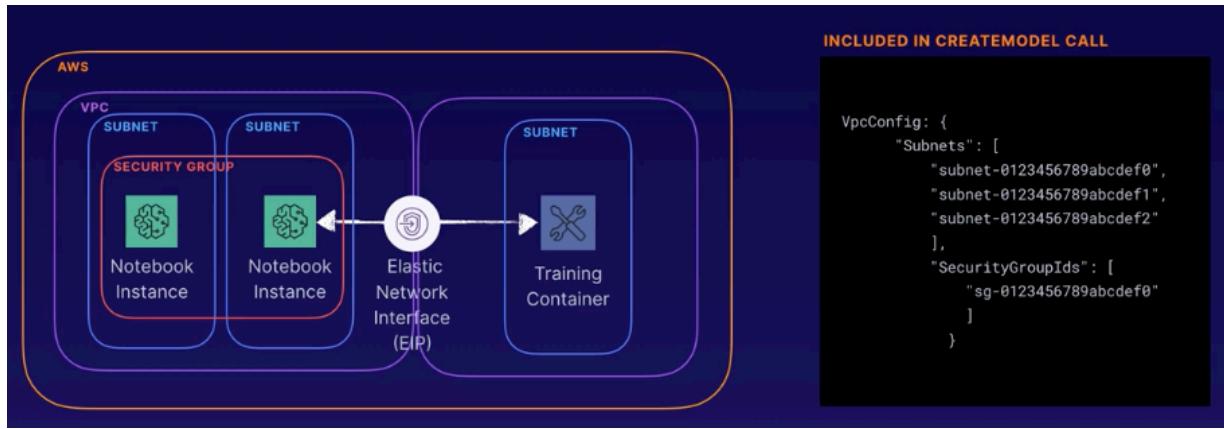
(You might also hear PrivateLink when referring to endpoints...this is the AWS service that powers Interface Endpoints.)



Securing ML resources, not going over Public internet



When we create a model for the first time, we can tell SM which subsets and security groups we want the training job to use
=> use EIP - Elastic Network Interface



Notebook instances

1	2	3
Internet-enabled by Default By default, SageMaker notebook instances are internet-enabled. This allows for download and access to popular packages and libraries.	Can Disable Internet Access Internet access may represent a security concern to some, so you can disable direct Internet access. However, for training and hosting models, you will need a NAT gateway, routes and security groups that permit Internet access.	Designed for Single User Notebook instances provide the user with root access for installing packages. Best practice is one user per notebook instance. We can restrict access to users or roles via IAM policy.

2 types of IAM policies:

- Identity-based
- Resource-based

	Identity-based Policy	Resource-based Policy
What	A permissions policy attached to IAM identities.	A permissions policy attached to a resource.
Why	Allow or deny access to a role or user.	Allow or deny access at the resource.
When	Only allow user "mary" to access the notebook instance "Mary_Notebook".	Restrict R/W access to an S3 bucket by specific AWS accounts.

Example of IAM permission policy that we can attach to an IAM user

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreate-Describe-Delete-Models",
      "Effect": "Allow",
      "Action": [
        "sagemaker>CreateModel",
        "sagemaker>DescribeModel",
        "sagemaker>DeleteModel"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AdditionalIamPermission",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::account-id:role/role-name"
    }
  ]
}
```

PassRole action => user of this policy has the ability to hand off their role to SM service

2 types of encryption:

- at rest
- in transit

	What	Generic Example	AWS Example
Encryption At Rest	Store data in an encrypted format	Use GPG to encrypt a file.	Configure S3 bucket to use S3-KMS encryption.
Encryption in Transit	Encrypt the data stream carrying the data	Using the TLS to encrypt an HTTP connection (HTTPS).	Use Certificate Manager with CloudFront to provide TLS support for a custom domain.

SM offers options to encrypt for notebooks and training jobs

The image contains two side-by-side screenshots of the Amazon SageMaker console interface, demonstrating encryption options for different types of resources.

Left Screenshot: Create notebook instance

- Notebook Instance settings:**
 - Notebook instance name:** A text input field.
 - Notebook instance type:** A dropdown menu set to "ml.t2.medium".
 - Elastic inference:** A dropdown menu set to "none".
 - IAM role:** A dropdown menu set to "AmazonSageMaker-ExecutionRole-20181221T161274".
 - VPC:** A dropdown menu set to "No VPC".
 - Lifecycle configuration - optional:** A dropdown menu set to "No configuration".
 - Encryption key - optional:** A dropdown menu set to "No Custom Encryption". This field is highlighted with an orange box.
 - Volume size in GB - optional:** An input field containing the value "5".

Right Screenshot: Create training job

- Job settings:**
 - Job name:** A text input field.
 - IAM role:** A dropdown menu set to "AmazonSageMaker-ExecutionRole-20181221T161274".
 - Algorithm options:** A dropdown menu set to "AmazonSageMaker built-in algorithms".
 - Algorithm source:**
 - Amazon SageMaker built-in algorithms
 - Your own algorithm resource
 - Your own algorithm container in ECR
 - An algorithm subscription from AWS Marketplace
 - Choose an algorithm:** A dropdown menu set to "Choose an algorithm or custom training image...".
- Resource configuration:**
 - Instance type:** A dropdown menu set to "ml.m4.xlarge".
 - Instance count:** An input field containing "1".
 - Additional storage volume per instance (GB):** An input field containing "1".
 - Encryption key - optional:** A dropdown menu set to "No Custom Encryption". This field is highlighted with an orange box.
- Stopping condition:**
 - Maximum runtime:** An input field containing "24".
 - Unit:** A dropdown menu set to "Hours".

Same with end points or batch transform

Create and configure endpoint

To deploy models to Amazon SageMaker, first create an endpoint. Provide an endpoint configuration to specify which models to deploy and the hardware requirements for each. See [Deploying a Model on Amazon SageMaker Hosting Services](#) Learn more about the API

Endpoint

Endpoint name
Your application uses this name to access this endpoint.
Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Attach endpoint configuration

Use an existing endpoint configuration
Use an existing endpoint configuration or clone an endpoint configuration.
 Create a new endpoint configuration
Add models and configure the instance and initial weight for each model.

New endpoint configuration

To deploy models to Amazon SageMaker, first create an endpoint configuration. In the configuration, specify which models to deploy, and the relative traffic weighting and hardware requirements for each.

Endpoint configuration name
Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Encryption key - optional
Encrypt your data. Choose an existing KMS key or enter a key's ARN.
No Custom Encryption

Create batch transform job

A transform job uses a model to transform data and stores the results at a specified location. [Learn more](#)

Batch transform job configuration

Job name
Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in the same AWS Region.

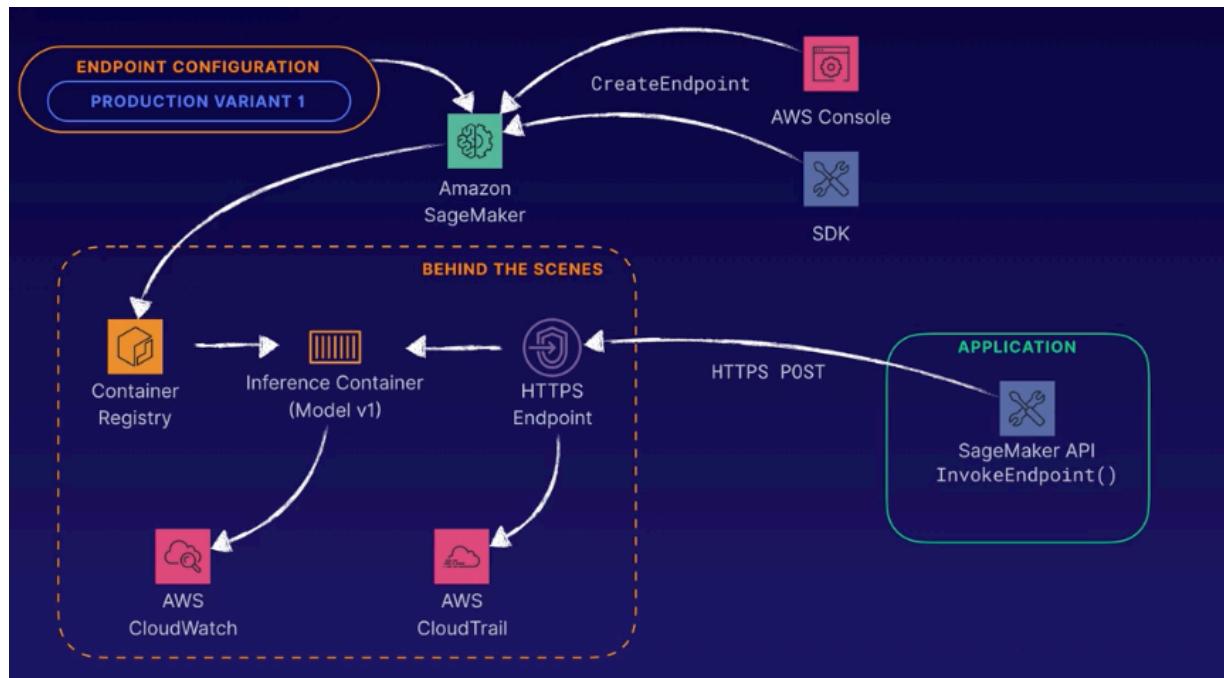
Model name
 Q Find model

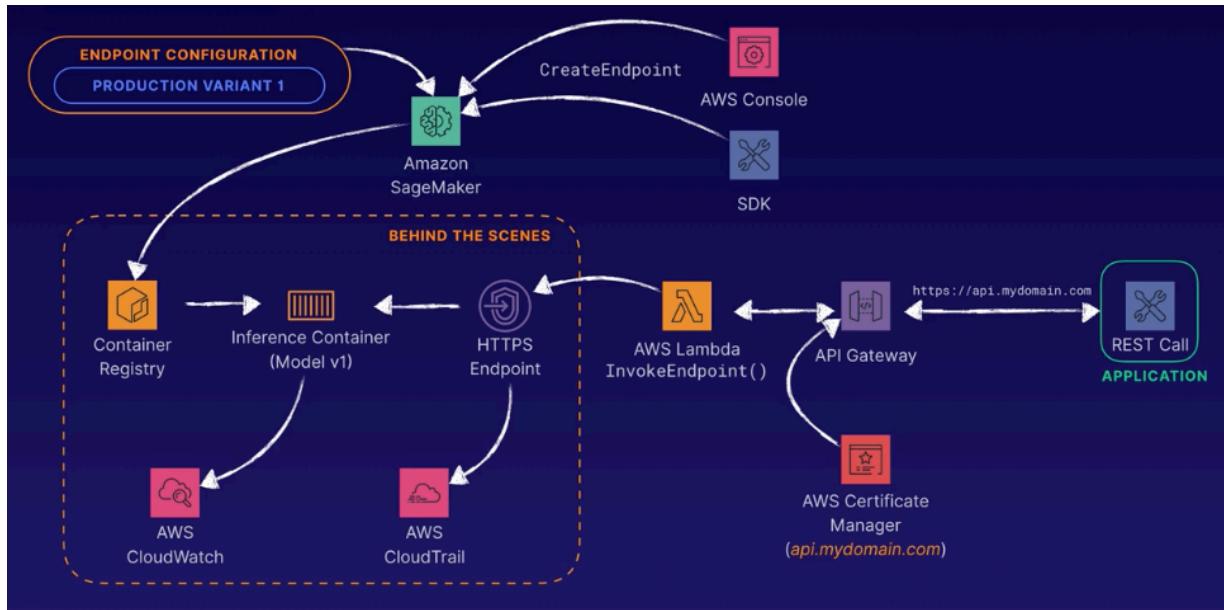
Instance type
Instance count
Encryption key - optional
Encrypt your data. Choose an existing KMS key or enter a key's ARN.
No Custom Encryption

Additional configuration

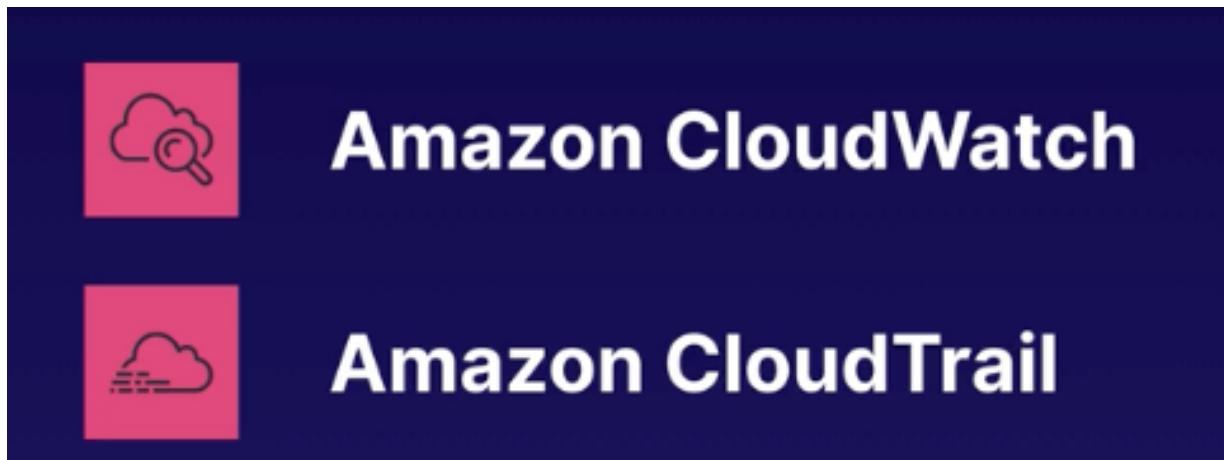
All these examples are encryption AT REST

For encryption in TRANSIT: using https





Monitor and Evaluate



1	Wide Variety of Metrics Endpoint Invocation, Endpoint Instance, Training, Transform and Ground Truth metrics available in addition to algorithm-specific metrics.	3	Metrics Available for 15 Months Statistics are kept 15 months to provide plenty of historical information on model performance and trends.
2	Near Real-Time Most SageMaker-specific metrics are available at a 1-minute frequency for quick reactions.	4	2 Week Limit on Console Metrics are collected and sent every five minutes for training and prediction jobs.

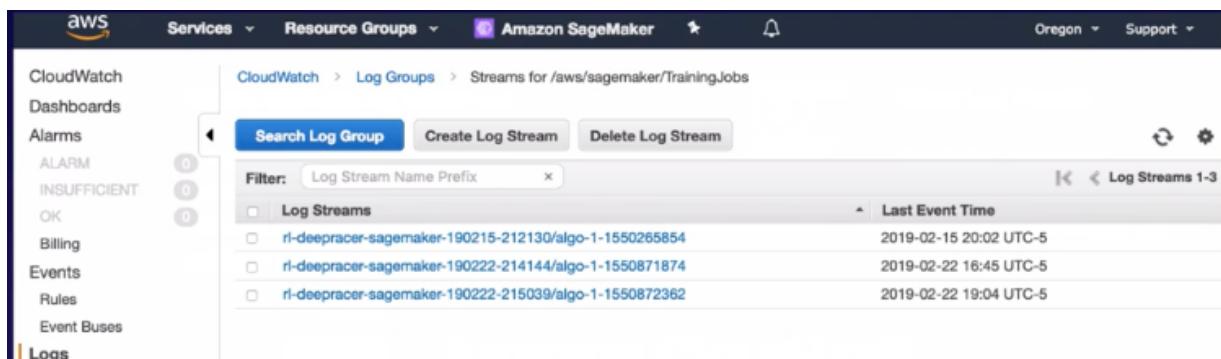
 Amazon CloudWatch

Any stdout or stderr messages from notebook instances, algorithm containers, or model containers is sent to Amazon CloudWatch Logs.

Log Group Name	Log Stream Name
/aws/sagemaker/TrainingJobs	[training-job-name]/algo-[instance-number-in-cluster]-[epoch_timestamp]
/aws/sagemaker/Endpoints/[EndpointName]	[production-variant-name]/[instance-id]
/aws/sagemaker/NotebookInstances	[notebook-instance-name]/[LifecycleConfigHook]
/aws/sagemaker/TransformJobs	[transform-job-name]/[instance-id]-[epoch_timestamp] [transform-job-name]/[instance-id]-[epoch_timestamp]/data-log

 Amazon CloudWatch

Example with deepracer

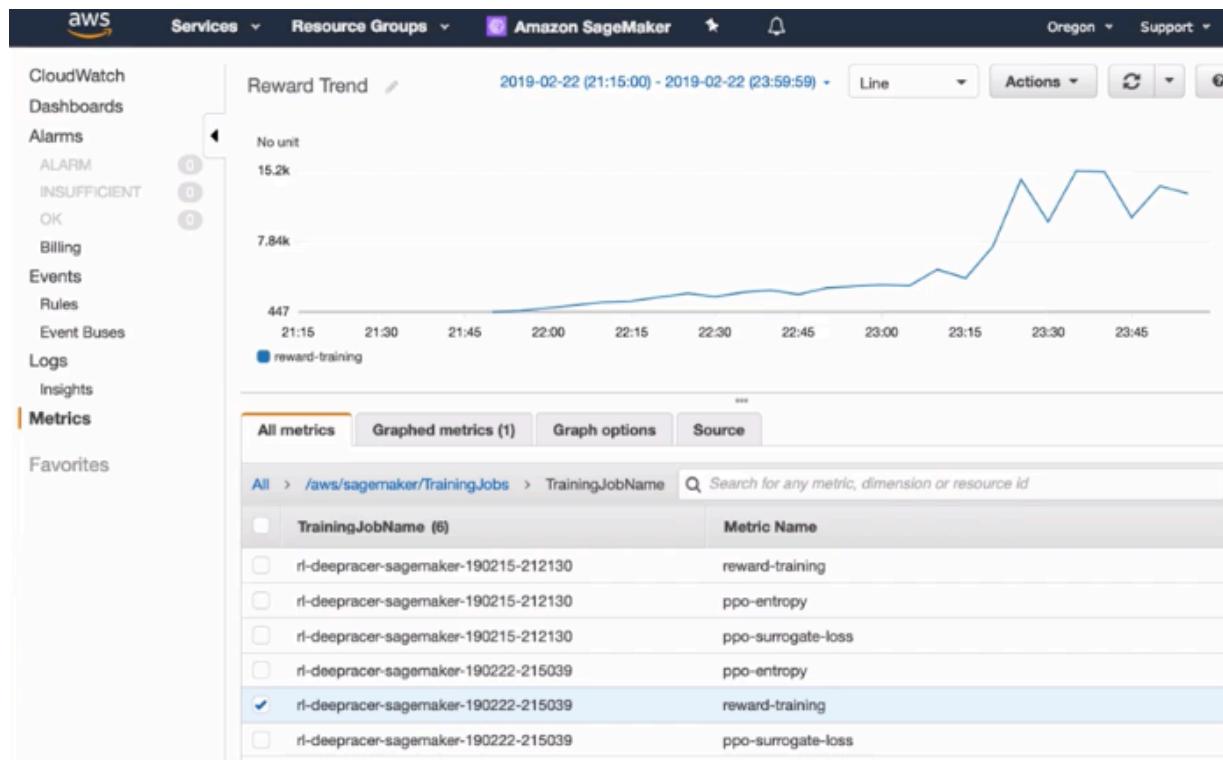


The screenshot shows the AWS CloudWatch Log Groups interface. The left sidebar lists services like CloudWatch, Dashboards, Alarms, Events, Rules, Event Buses, and Loas. The main area shows the path: CloudWatch > Log Groups > Streams for /aws/sagemaker/TrainingJobs. There are three log streams listed:

Last Event Time
2019-02-15 20:02 UTC-5
2019-02-22 16:45 UTC-5
2019-02-22 19:04 UTC-5

Amazon SageMaker																																																			
CloudWatch	CloudWatch > Log Groups > /aws/sagemaker/TrainingJobs > rl-deepracer-sagemaker-190222-215039/algo-1-1550872362	Oregon Support																																																	
Dashboards		Expand all Row Text Filter Copy Open in new tab Help																																																	
Alarms																																																			
ALARM																																																			
INSUFFICIENT																																																			
OK																																																			
Billing																																																			
Events																																																			
Rules																																																			
Event Buses																																																			
Logs																																																			
Insights																																																			
Metrics																																																			
Favorites																																																			
<div style="border: 1px solid #ccc; padding: 5px;"> <p>Filter events</p> <p>Time (UTC +00:00) Message</p> <p>all 2019-02-22 (00:04:01)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Time</th> <th>Message</th> </tr> </thead> <tbody> <tr><td>2019-02-22 21:54:06</td><td>Training> Name=main_level/agent, Worker=0, Episode=4, Total reward=57.86, Steps=85, Training iteration=0</td></tr> <tr><td>2019-02-22 21:54:08</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=5, Total reward=36.66, Steps=98, Training iteration=0</td></tr> <tr><td>2019-02-22 21:54:12</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=6, Total reward=139.94, Steps=128, Training iteration=0</td></tr> <tr><td>2019-02-22 21:54:16</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=7, Total reward=68.37, Steps=152, Training iteration=0</td></tr> <tr><td>2019-02-22 21:54:26</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=8, Total reward=908.4, Steps=234, Training iteration=0</td></tr> <tr><td>2019-02-22 21:54:34</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=9, Total reward=626.1, Steps=303, Training iteration=0</td></tr> <tr><td>2019-02-22 21:54:47</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=10, Total reward=1474.12, Steps=414, Training iteration=0</td></tr> <tr><td>2019-02-22 21:54:55</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=11, Total reward=560.35, Steps=480, Training iteration=0</td></tr> <tr><td>2019-02-22 21:55:03</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=12, Total reward=525.85, Steps=542, Training iteration=0</td></tr> <tr><td>2019-02-22 21:55:07</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=13, Total reward=154.4, Steps=574, Training iteration=0</td></tr> <tr><td colspan="2">Training> Name=main_level/agent, Worker=0, Episode=13, Total reward=154.4, Steps=574, Training iteration=0</td></tr> <tr><td>2019-02-22 21:55:13</td><td>Training> Name=main_level/agent, Worker=0, Episode=14, Total reward=345.03, Steps=622, Training iteration=0</td></tr> <tr><td>2019-02-22 21:55:16</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=15, Total reward=95.21, Steps=646, Training iteration=0</td></tr> <tr><td>2019-02-22 21:55:24</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=16, Total reward=544.77, Steps=711, Training iteration=0</td></tr> <tr><td>2019-02-22 21:55:28</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=17, Total reward=132.7, Steps=740, Training iteration=0</td></tr> <tr><td>2019-02-22 21:55:40</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=18, Total reward=1412.0, Steps=844, Training iteration=0</td></tr> <tr><td>2019-02-22 21:55:50</td><td>Trainings> Name=main_level/agent, Worker=0, Episode=19, Total reward=724.22, Steps=919, Training iteration=0</td></tr> <tr><td>2019-02-22 21:55:53</td><td>Training> Name=main_level/agent, Worker=0, Episode=20, Total reward=100.1, Steps=942, Training iteration=0</td></tr> <tr><td>2019-02-22 21:55:57</td><td>Policy training> Surrogate loss=-0.008326728054283638, KL divergence=0.014008121564984322, Entropy=1.5954</td></tr> <tr><td>2019-02-22 21:55:58</td><td>Policy training> Surrogate loss=-0.01831730082631111, KL divergence=0.008071099407970905, Entropy=1.6021</td></tr> <tr><td>2019-02-22 21:56:00</td><td>Policy training> Surrogate loss=-0.01681138575077057, KL divergence=0.012024997733533382, Entropy=1.6001</td></tr> <tr><td>2019-02-22 21:56:02</td><td>Policy training> Surrogate loss=-0.0250502061088814, KL divergence=0.010500132106244564, Entropy=1.60057</td></tr> <tr><td>2019-02-22 21:56:04</td><td>Policy training> Surrogate loss=-0.03210756555199523, KL divergence=0.010665261186659336, Entropy=1.5998</td></tr> <tr><td>2019-02-22 21:56:05</td><td>Policy training> Surrogate loss=-0.029866738244891167, KL divergence=0.012668825685977936, Entropy=1.597</td></tr> </tbody> </table> </div>	Time	Message	2019-02-22 21:54:06	Training> Name=main_level/agent, Worker=0, Episode=4, Total reward=57.86, Steps=85, Training iteration=0	2019-02-22 21:54:08	Trainings> Name=main_level/agent, Worker=0, Episode=5, Total reward=36.66, Steps=98, Training iteration=0	2019-02-22 21:54:12	Trainings> Name=main_level/agent, Worker=0, Episode=6, Total reward=139.94, Steps=128, Training iteration=0	2019-02-22 21:54:16	Trainings> Name=main_level/agent, Worker=0, Episode=7, Total reward=68.37, Steps=152, Training iteration=0	2019-02-22 21:54:26	Trainings> Name=main_level/agent, Worker=0, Episode=8, Total reward=908.4, Steps=234, Training iteration=0	2019-02-22 21:54:34	Trainings> Name=main_level/agent, Worker=0, Episode=9, Total reward=626.1, Steps=303, Training iteration=0	2019-02-22 21:54:47	Trainings> Name=main_level/agent, Worker=0, Episode=10, Total reward=1474.12, Steps=414, Training iteration=0	2019-02-22 21:54:55	Trainings> Name=main_level/agent, Worker=0, Episode=11, Total reward=560.35, Steps=480, Training iteration=0	2019-02-22 21:55:03	Trainings> Name=main_level/agent, Worker=0, Episode=12, Total reward=525.85, Steps=542, Training iteration=0	2019-02-22 21:55:07	Trainings> Name=main_level/agent, Worker=0, Episode=13, Total reward=154.4, Steps=574, Training iteration=0	Training> Name=main_level/agent, Worker=0, Episode=13, Total reward=154.4, Steps=574, Training iteration=0		2019-02-22 21:55:13	Training> Name=main_level/agent, Worker=0, Episode=14, Total reward=345.03, Steps=622, Training iteration=0	2019-02-22 21:55:16	Trainings> Name=main_level/agent, Worker=0, Episode=15, Total reward=95.21, Steps=646, Training iteration=0	2019-02-22 21:55:24	Trainings> Name=main_level/agent, Worker=0, Episode=16, Total reward=544.77, Steps=711, Training iteration=0	2019-02-22 21:55:28	Trainings> Name=main_level/agent, Worker=0, Episode=17, Total reward=132.7, Steps=740, Training iteration=0	2019-02-22 21:55:40	Trainings> Name=main_level/agent, Worker=0, Episode=18, Total reward=1412.0, Steps=844, Training iteration=0	2019-02-22 21:55:50	Trainings> Name=main_level/agent, Worker=0, Episode=19, Total reward=724.22, Steps=919, Training iteration=0	2019-02-22 21:55:53	Training> Name=main_level/agent, Worker=0, Episode=20, Total reward=100.1, Steps=942, Training iteration=0	2019-02-22 21:55:57	Policy training> Surrogate loss=-0.008326728054283638, KL divergence=0.014008121564984322, Entropy=1.5954	2019-02-22 21:55:58	Policy training> Surrogate loss=-0.01831730082631111, KL divergence=0.008071099407970905, Entropy=1.6021	2019-02-22 21:56:00	Policy training> Surrogate loss=-0.01681138575077057, KL divergence=0.012024997733533382, Entropy=1.6001	2019-02-22 21:56:02	Policy training> Surrogate loss=-0.0250502061088814, KL divergence=0.010500132106244564, Entropy=1.60057	2019-02-22 21:56:04	Policy training> Surrogate loss=-0.03210756555199523, KL divergence=0.010665261186659336, Entropy=1.5998	2019-02-22 21:56:05	Policy training> Surrogate loss=-0.029866738244891167, KL divergence=0.012668825685977936, Entropy=1.597	
Time	Message																																																		
2019-02-22 21:54:06	Training> Name=main_level/agent, Worker=0, Episode=4, Total reward=57.86, Steps=85, Training iteration=0																																																		
2019-02-22 21:54:08	Trainings> Name=main_level/agent, Worker=0, Episode=5, Total reward=36.66, Steps=98, Training iteration=0																																																		
2019-02-22 21:54:12	Trainings> Name=main_level/agent, Worker=0, Episode=6, Total reward=139.94, Steps=128, Training iteration=0																																																		
2019-02-22 21:54:16	Trainings> Name=main_level/agent, Worker=0, Episode=7, Total reward=68.37, Steps=152, Training iteration=0																																																		
2019-02-22 21:54:26	Trainings> Name=main_level/agent, Worker=0, Episode=8, Total reward=908.4, Steps=234, Training iteration=0																																																		
2019-02-22 21:54:34	Trainings> Name=main_level/agent, Worker=0, Episode=9, Total reward=626.1, Steps=303, Training iteration=0																																																		
2019-02-22 21:54:47	Trainings> Name=main_level/agent, Worker=0, Episode=10, Total reward=1474.12, Steps=414, Training iteration=0																																																		
2019-02-22 21:54:55	Trainings> Name=main_level/agent, Worker=0, Episode=11, Total reward=560.35, Steps=480, Training iteration=0																																																		
2019-02-22 21:55:03	Trainings> Name=main_level/agent, Worker=0, Episode=12, Total reward=525.85, Steps=542, Training iteration=0																																																		
2019-02-22 21:55:07	Trainings> Name=main_level/agent, Worker=0, Episode=13, Total reward=154.4, Steps=574, Training iteration=0																																																		
Training> Name=main_level/agent, Worker=0, Episode=13, Total reward=154.4, Steps=574, Training iteration=0																																																			
2019-02-22 21:55:13	Training> Name=main_level/agent, Worker=0, Episode=14, Total reward=345.03, Steps=622, Training iteration=0																																																		
2019-02-22 21:55:16	Trainings> Name=main_level/agent, Worker=0, Episode=15, Total reward=95.21, Steps=646, Training iteration=0																																																		
2019-02-22 21:55:24	Trainings> Name=main_level/agent, Worker=0, Episode=16, Total reward=544.77, Steps=711, Training iteration=0																																																		
2019-02-22 21:55:28	Trainings> Name=main_level/agent, Worker=0, Episode=17, Total reward=132.7, Steps=740, Training iteration=0																																																		
2019-02-22 21:55:40	Trainings> Name=main_level/agent, Worker=0, Episode=18, Total reward=1412.0, Steps=844, Training iteration=0																																																		
2019-02-22 21:55:50	Trainings> Name=main_level/agent, Worker=0, Episode=19, Total reward=724.22, Steps=919, Training iteration=0																																																		
2019-02-22 21:55:53	Training> Name=main_level/agent, Worker=0, Episode=20, Total reward=100.1, Steps=942, Training iteration=0																																																		
2019-02-22 21:55:57	Policy training> Surrogate loss=-0.008326728054283638, KL divergence=0.014008121564984322, Entropy=1.5954																																																		
2019-02-22 21:55:58	Policy training> Surrogate loss=-0.01831730082631111, KL divergence=0.008071099407970905, Entropy=1.6021																																																		
2019-02-22 21:56:00	Policy training> Surrogate loss=-0.01681138575077057, KL divergence=0.012024997733533382, Entropy=1.6001																																																		
2019-02-22 21:56:02	Policy training> Surrogate loss=-0.0250502061088814, KL divergence=0.010500132106244564, Entropy=1.60057																																																		
2019-02-22 21:56:04	Policy training> Surrogate loss=-0.03210756555199523, KL divergence=0.010665261186659336, Entropy=1.5998																																																		
2019-02-22 21:56:05	Policy training> Surrogate loss=-0.029866738244891167, KL divergence=0.012668825685977936, Entropy=1.597																																																		

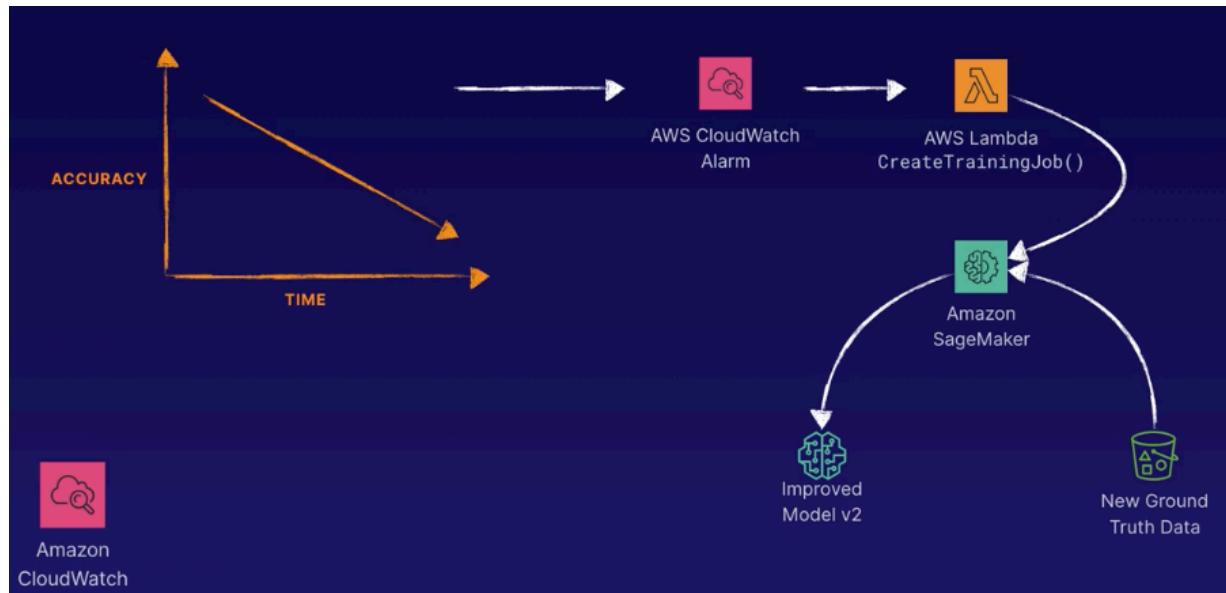
Can also view some metrics in graph format - reward function below



Re-Training

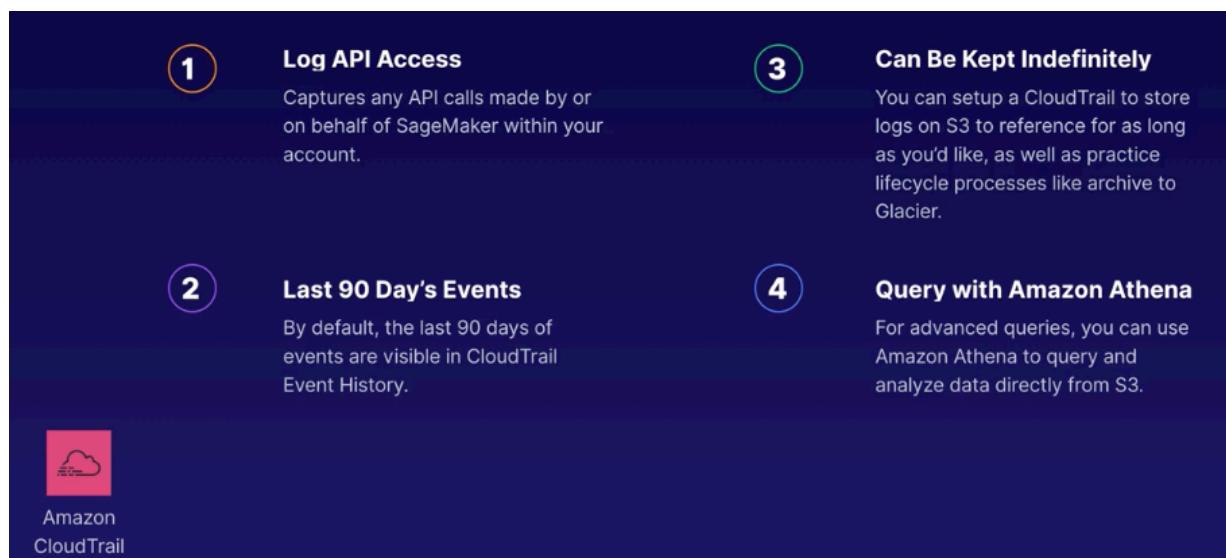
Over time, our model may be less accurate

We can use Cloudwatch alarms to trigger lambda which can initiate another training job



Amazon CloudTrail

Captures API access in/out of the account



Example of Cludtrail entry - stopping a notebook instance

Sales Services Resource Groups Amazon SageMaker Oregon Support

CloudTrail
Dashboard Event history Trails

Event history

Your event history contains the activities taken by people, groups, or AWS services in supported services in your AWS account. By default, the view filters out read-only events. You can change or remove that filter, or apply other filters.

You can view the last 90 days of events. Choose an event to view more information about it. To view a complete log of your CloudTrail events, create a trail and then go to your Amazon S3 bucket or CloudWatch Logs. [Learn more](#)

Can't find what you're looking for? Run advanced queries in Amazon Athena

Filter: Read only false Time range: 2019-02-22 12:00 AM — 2019-02-23 12:00 AM

Event time	User name	Event name	Resource type	Resource name
2019-02-22, 07:27:37 PM	scott	ListTagsForResources		
2019-02-22, 07:27:28 PM	scott	StopNotebookInstance		

AWS access key Hey, I've been to DEFCON! Event time 2019-02-22, 07:27:28 PM
AWS region us-west-2 Read only false
Error code Request ID 246d1dc9-f32c-46b9-8540-abb3d6f29f5b
Event ID 1a3c3e07-63e6-4fea-ab77-40b71d696b0f Source IP address Hey, I've been to DEFCON!
Event name StopNotebookInstance User name scott
Event source sagemaker.amazonaws.com

Resources Referenced (0)

View event

2019-02-22, 07:12:12 PM	SageMaker	DeleteNetworkInterface	EC2 NetworkInterface	eni-0aaa393aae14f7bd3
2019-02-22, 07:10:33 PM	SageMaker	DeleteNetworkInterface	EC2 NetworkInterface	eni-094db192c00e25486

View Event X

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "DEFCONE5",
    "arn": "arn:aws:iam::DEFCON:user/scott",
    "accountId": "DEFCON",
    "accessKeyId": "DEFCONE5",
    "userName": "scott",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "true",
        "creationDate": "2019-02-22T17:34:08Z"
      }
    },
    "invokedBy": "signin.amazonaws.com"
  },
  "eventTime": "2019-02-23T00:27:28Z",
  "eventSource": "sagemaker.amazonaws.com",
  "eventName": "StopNotebookInstance"
}
```

Exam tips

Concepts

- Understand the difference between offline and online usage for models and when you might use each.
- Have a high-level understanding of the types of deployments and respective pros and cons.
- Understand A/B Testing and how it might be used to introduce a newly updated model.
- Understand the concepts behind continuous integration, continuous delivery and continuous deployment.

AI Developer Services

- The AI developer services provided by AWS are scalable, fault-tolerant and ready to use.
- Know each service, its purpose and in what use-cases they might be used.
- Experiment with each service via the Console or CLI to get a feel for what it does and how it works.

Amazon SageMaker Deployments

- Know the three main steps in creating a deployment using SageMaker Hosting Services.
- Understand ProductionVariants and how they can be used to introduce new evolutions of your models.
- Know how to calculate the percentage of traffic given weights for each production variant.
- Understand the purpose and limitations of Interface Pipelines, SageMaker Neo, Elastic Inference, Auto-Scaling and Cool Down
- Recall best practice suggestions for high availability with SageMaker.

Other ML Deployment Options

- Know the four other options for ML deployment aside from SageMaker Hosting Services and when you might chose them instead of SageMaker.

Security

- Be familiar with using VPCs, NACLs, SGs, IAM and KMS to secure Amazon ML and SageMaker resources.
- Understand a VPC Endpoint and why it increases security.
- Know that Notebook Instances are Internet-enabled by default but this can be disabled given certain conditions and introducing limitations.

Monitor and Evaluate

- Understand the difference between CloudWatch and CloudTrail.
- Know that CloudWatch has a limited storage while CloudTrail can have unlimited storage if you log to an S3 bucket.
- Understand how you might use metrics to trigger events like re-training.