AWS Solutions Training for Partners: Machine Learning (ML) on AWS for ML Practitioners – Technical

Course Agenda

aws training and certification

- Module 1: Introduction to Machine Learning
- Module 2: Artificial Intelligence Services on AWS
- Module 3: Machine Learning Process
- Module 4: Data Collection, Integration, Preparation and Visualization, and Analysis
- Module 5: Deep Learning Amazon Machine Images
- Module 6: Amazon SageMaker
- Module 7: Amazon SageMaker Notebooks
- Module 8: Amazon SageMaker Built-In Algorithms
- Module 9: Amazon SageMaker – Debugging and Monitoring
- Module 10: Introduction to MLOps
- Module 11: Next Steps and Additional Learning

Module 0:

Course Introduction

aws training and certification

- Identify key use cases for machine learning.
- Framing business problem for ML solution
- Choose the AWS AI/ML services that are appropriate for the solution
- Explain the end-to-end process of implementing ML features
- Walk through the feature design process, requirements, and data dependencies for implementing ML on AWS.
- Describe the functions and capabilities of AWS Machine Learning managed services.
- Create and deploy a customer machine learning model at scale using Amazon SageMaker
- Explain MLOps and AWS services to orchestrate ML pipeline
- Find additional resources to continue their training and development in ML

Module 0:

Course Introduction

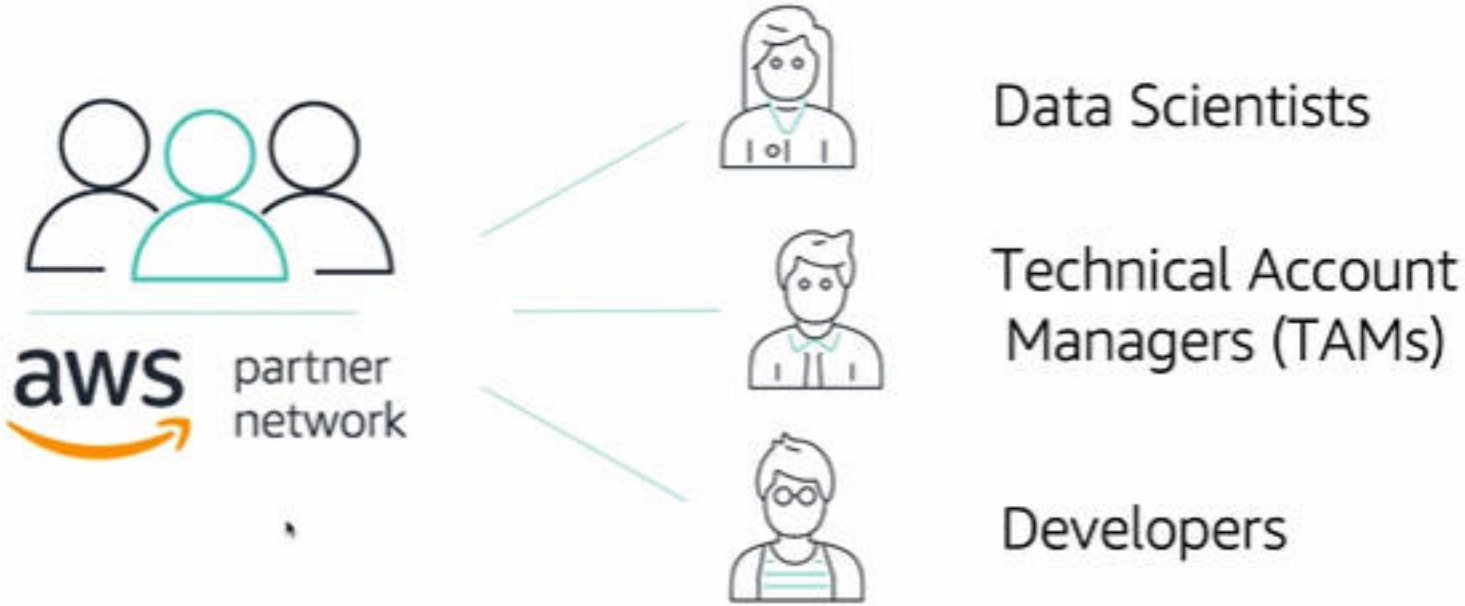
aws training and certification

- Identify key use cases for machine learning.
- Framing business problem for ML solution
- Choose the AWS AI/ML services that are appropriate for the solution
- Explain the end-to-end process of implementing ML features
- Walk through the feature design process, requirements, and data dependencies for implementing ML on AWS.
- Describe the functions and capabilities of AWS Machine Learning managed services.
- Create and deploy a customer machine learning model at scale using Amazon SageMaker
- Explain MLOps and AWS services to orchestrate ML pipeline
- Find additional resources to continue their training and development in ML

Who this course is for

aws training and certification

This course is for APN Consulting Partners and Technology Partners.



aws training and certification

Module 1:

Introduction to Machine Learning

- Identify machine learning, artificial intelligence, and deep learning (DL)
- Explain how traditional programming differs from solving problems with machine learning
- List and explain the various types of machine learning
- Identify key machine learning terms
- Key Challenges and limitations of AI
- Insurance Fraud use case

-46:08 2x

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Module 1:

Introduction to Machine Learning

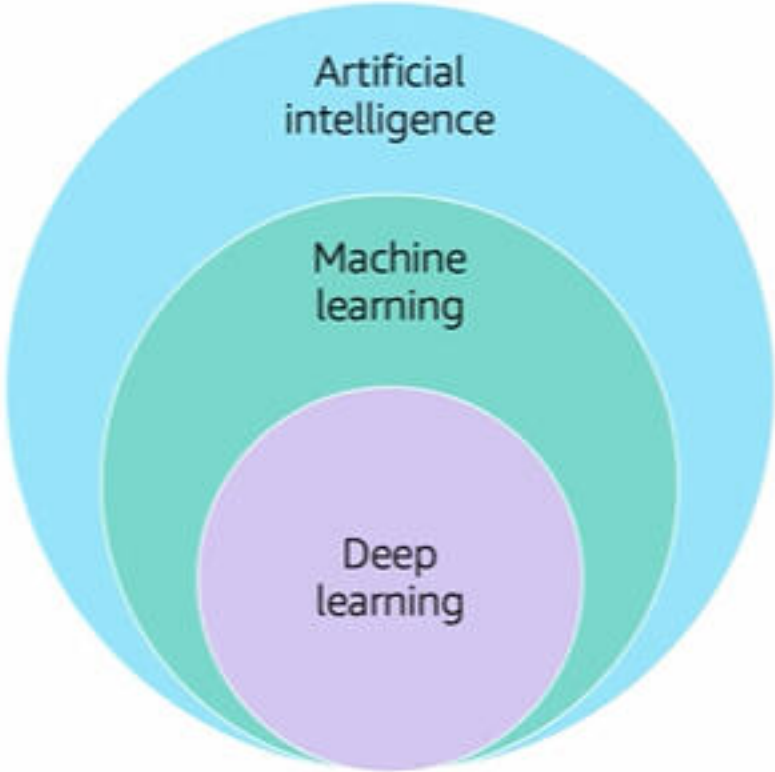
aws training and certification

- Identify machine learning, artificial intelligence, and deep learning (DL)
- Explain how traditional programming differs from solving problems with machine learning
- List and explain the various types of machine learning
- Identify key machine learning terms
- Key Challenges and limitations of AI
- Insurance Fraud use case

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

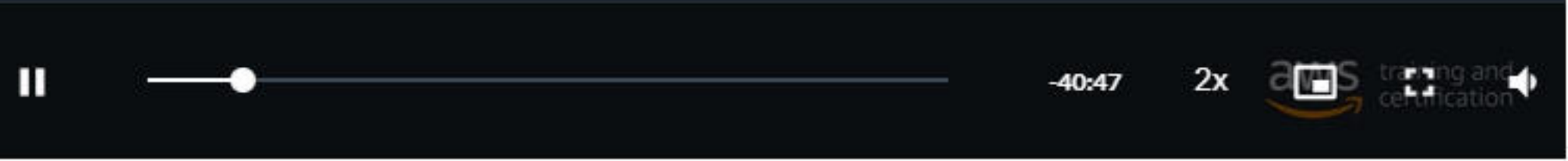
Let's define AI, ML, and DL

aws training and certification



© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Traditional Programming vs. ML



Traditional programming: You write the business rules

aws training and certification

Input

Age	Gender	Purchase Date	Items
30	M	3/1/2017	Toy
40	M	1/3/2017	Books
....	.....	....	....

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Traditional programming: You write the business rules

aws training and certification

Human Programmer

Input

Age	Gender	Purchase Date	Items
30	M	3/1/2017	Toy
40	M	1/3/2017	Books
....	.....	....	....

Rule 1: 15 <age< 30
Rule 2: Bought Toy=Y, Last Purchase<30 days
Rule 3: Gender = 'M', Bought Toy ='Y'
Rule 4: .....
Rule 5: .....

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Traditional programming: You write the business rules

aws training and certification

Human Programmer

Input

Age	Gender	Purchase Date	Items
30	M	3/1/2017	Toy
40	M	1/3/2017	Books
....	.....	....	....

Rule 1: 15 <age< 30
Rule 2: Bought Toy=Y, Last Purchase<30 days
Rule 3: Gender = 'M', Bought Toy ='Y'
Rule 4: .....
Rule 5: .....

Output

Age	Gender	Purchase Date	Items
30	M	3/1/2017	Toy
....	.....	....	....

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

In ML: The data writes the business rules

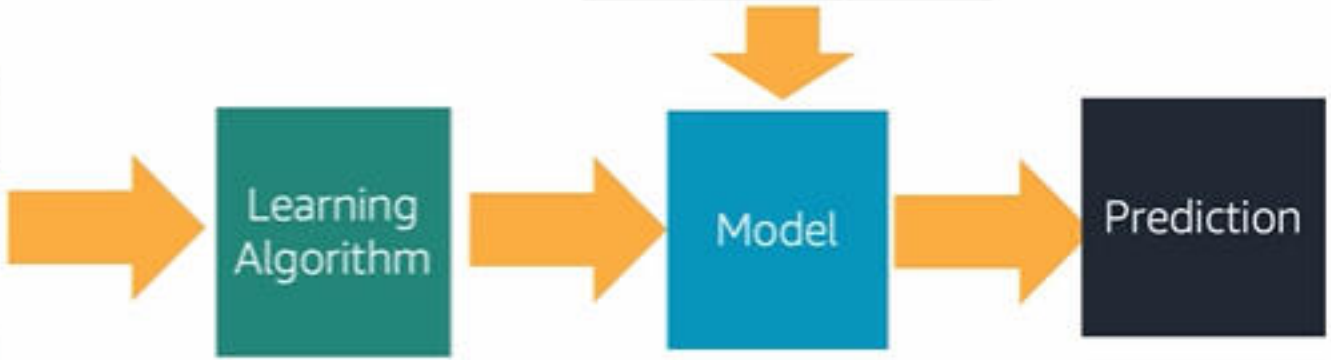
aws training and certification

Input - New Unseen Data

Age	Gender	Items
35	F	
39	M	Toy

Age	Gender	Purchase Date	Items
30	M	3/20/2017	Toy *
40	M	1/3/2017	Books
....	.....	.....	.....

Historical Purchase Data
(Training Data - Input)



Output

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

When to use machine learning

aws training and certification

- Use ML when you **can't code it**
  - Complex tasks where deterministic solutions don't suffice
    - E.g., recognizing speech/images

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

When to use machine learning

aws training and certification

- Use ML when you **can't code it**
  - Complex tasks where deterministic solutions don't suffice
    - E.g., recognizing speech/images
- Use ML when you **can't scale it**
  - Replace repetitive tasks needing human-like expertise
    - E.g., recommendations, spam, fraud detection, machine translation
- Use ML when you have to **adapt/personalize**
  - E.g., recommendation and personalization

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

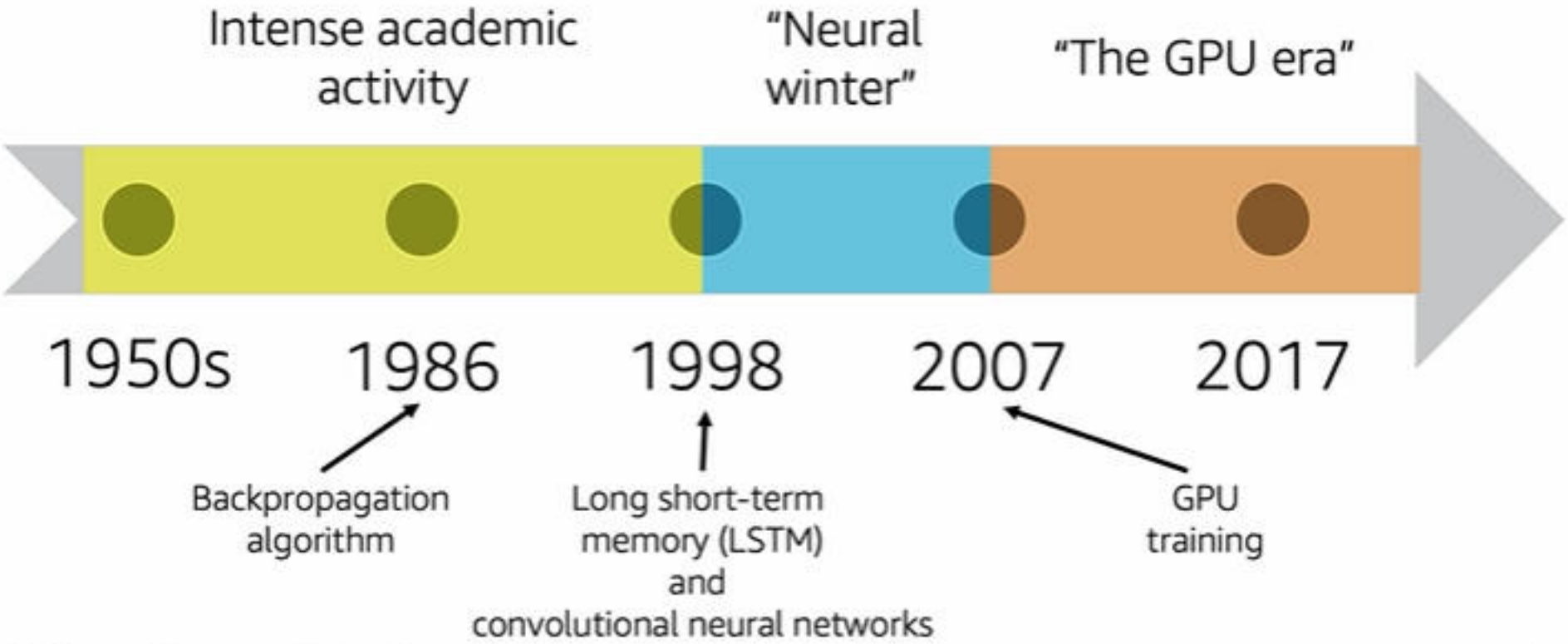
When to use machine learning

aws training and certification

- Use ML when you **can't code it**
  - Complex tasks where deterministic solutions don't suffice
    - E.g., recognizing speech/images
- Use ML when you **can't scale it**
  - Replace repetitive tasks needing human-like expertise
    - E.g., recommendations, spam, fraud detection, machine translation
- Use ML when you have to **adapt/personalize**
  - E.g., recommendation and personalization
- Use ML when you **can't track it**
  - E.g., automated driving

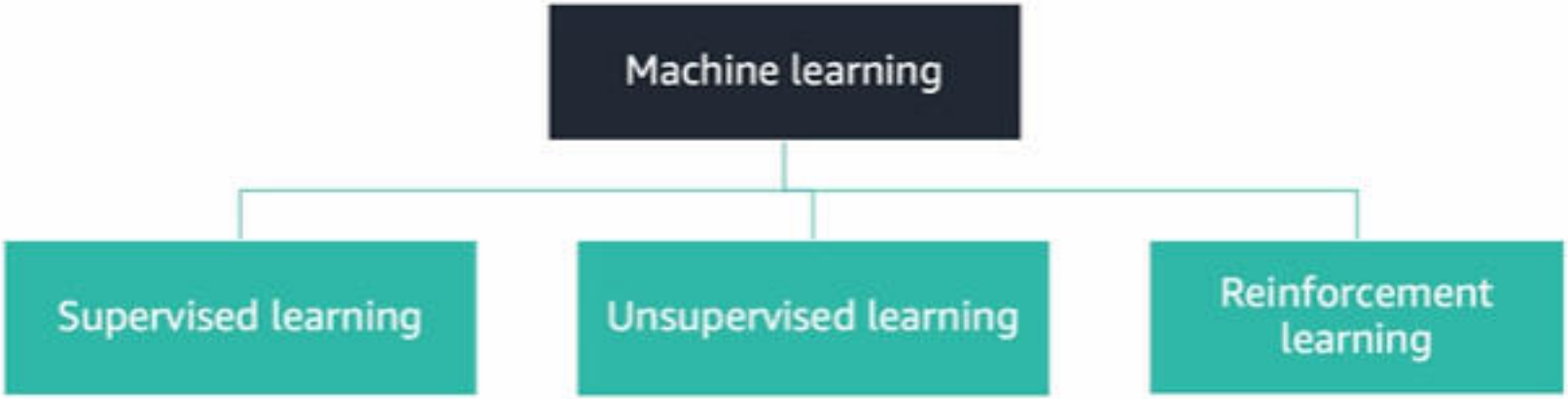
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML timeline



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Machine learning types

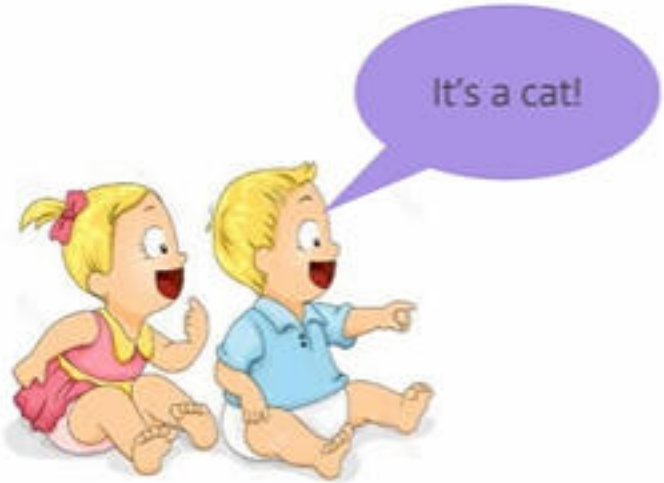


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Supervised learning: How humans learn

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Supervised learning: How humans learn

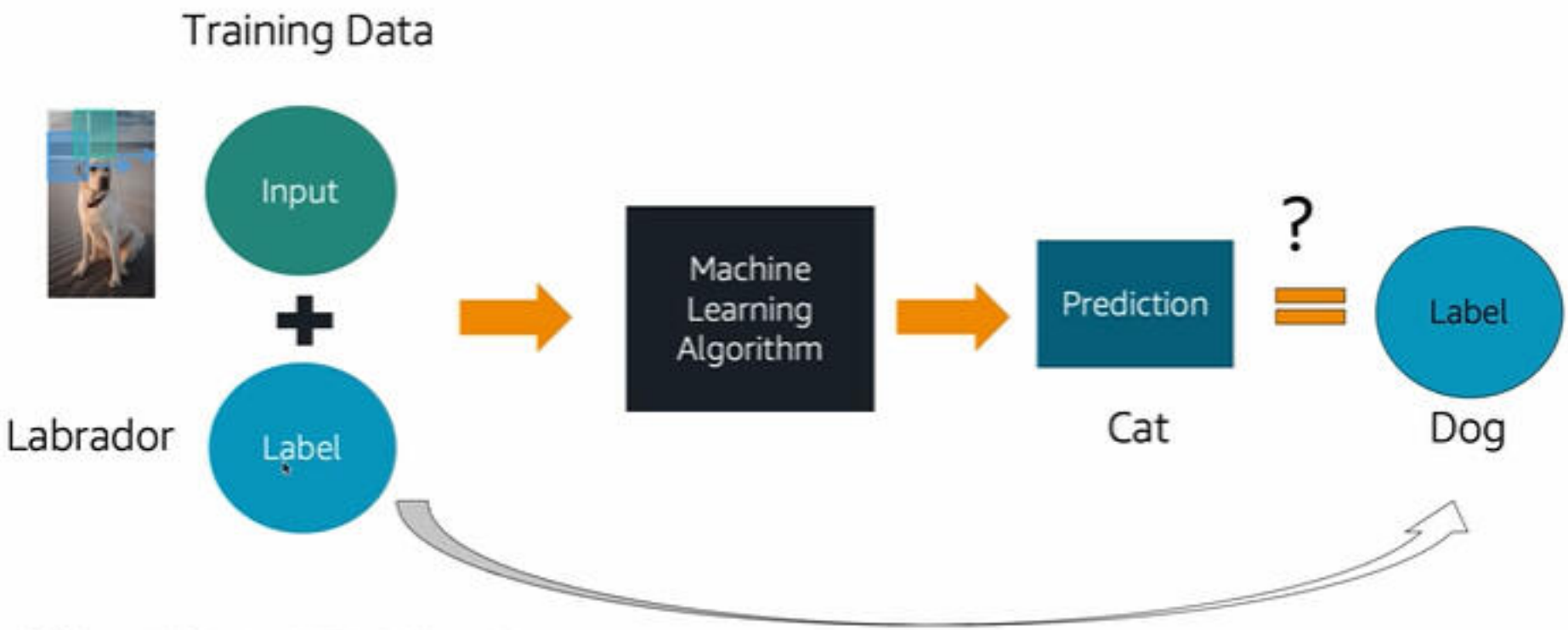


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Supervised learning: How machines learn

aws training and certification

Human intervention and validation required; e.g., photo classification and tagging

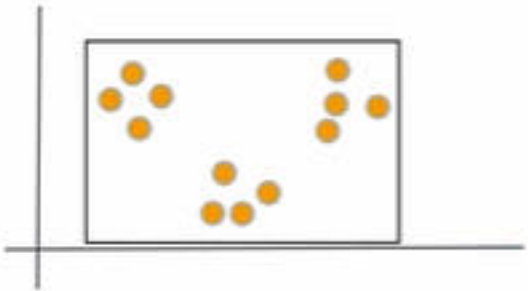


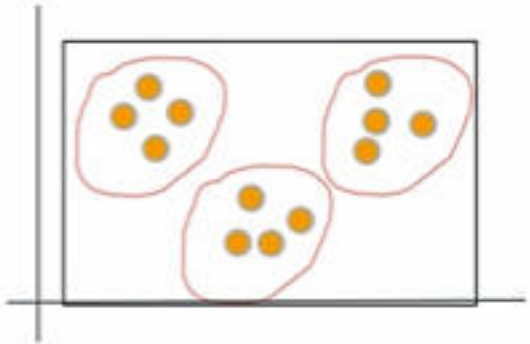
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Unsupervised learning

aws training and certification

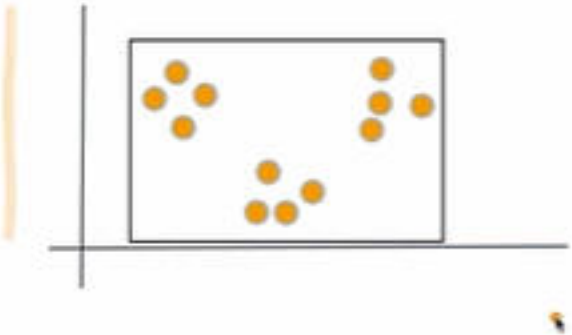
Input

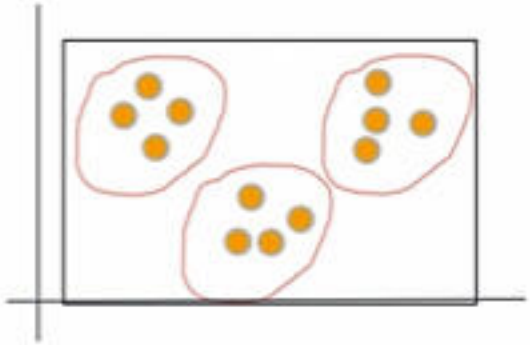




© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Unsupervised learning

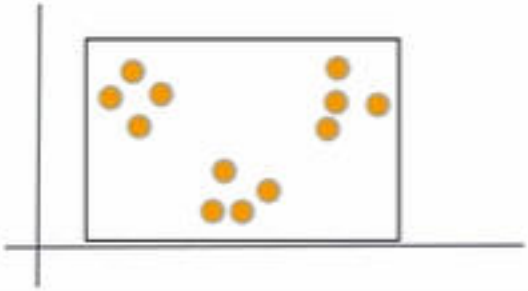


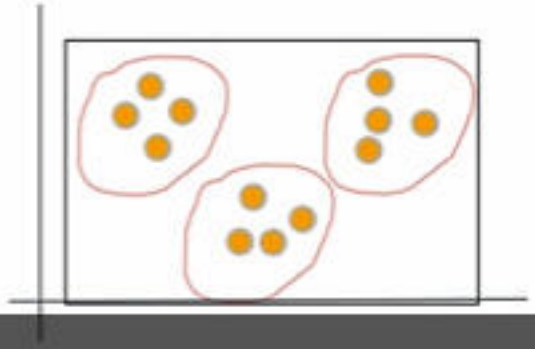


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Unsupervised learning

aws training and certification

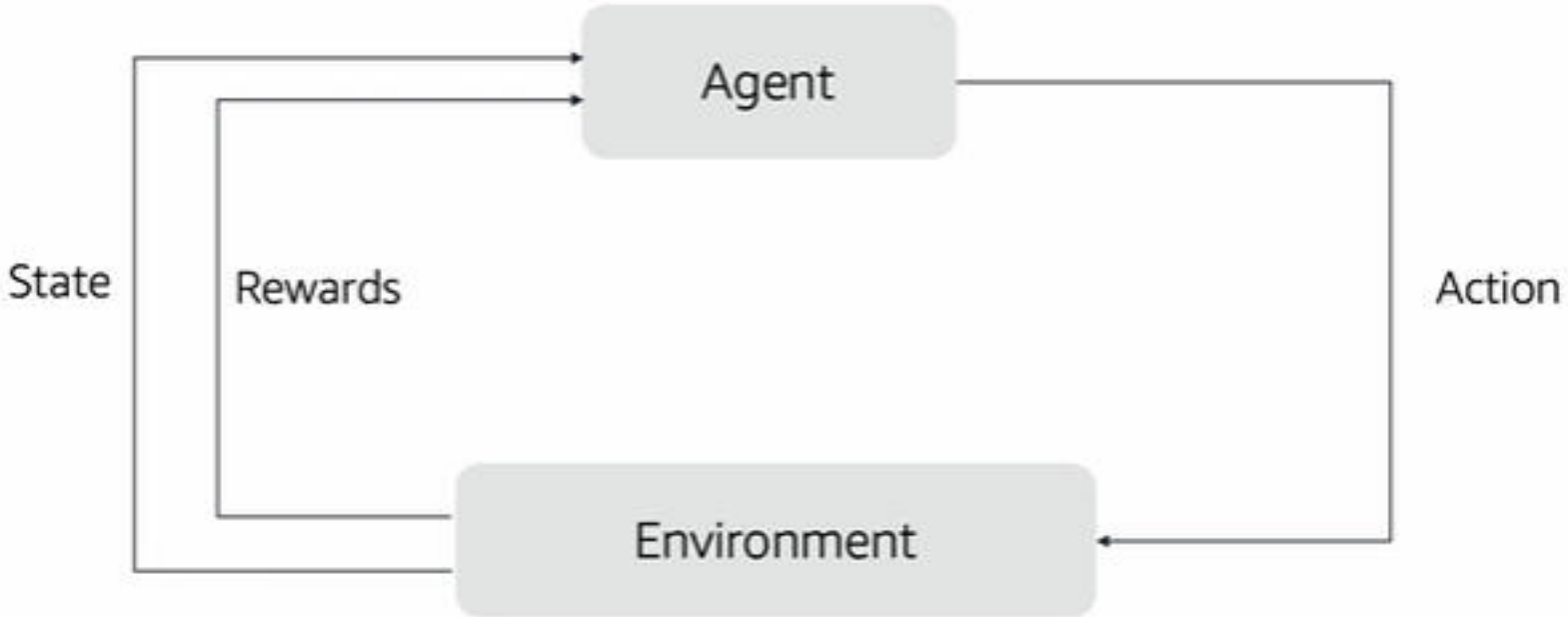




-26:40 2x

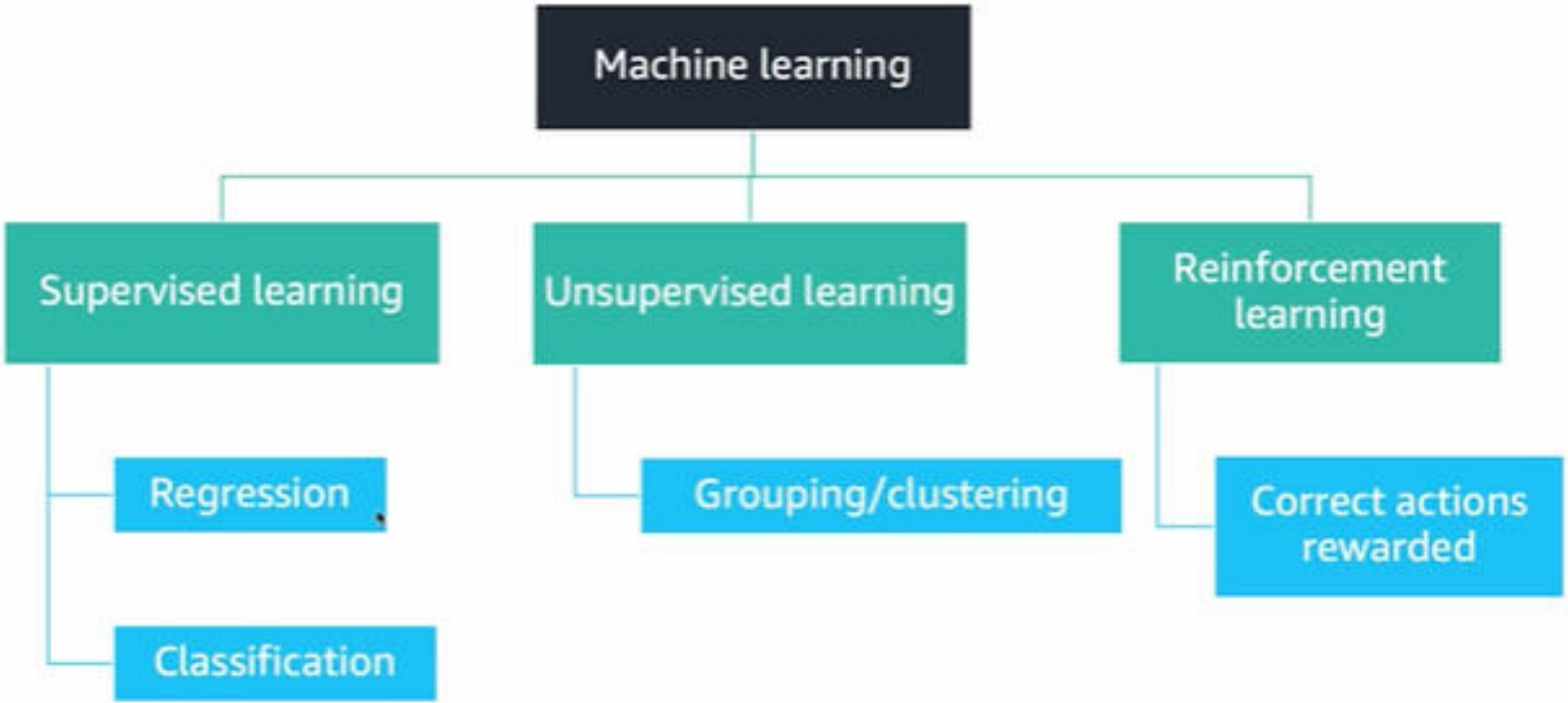
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Reinforcement learning



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Machine learning types



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Terminology and concepts

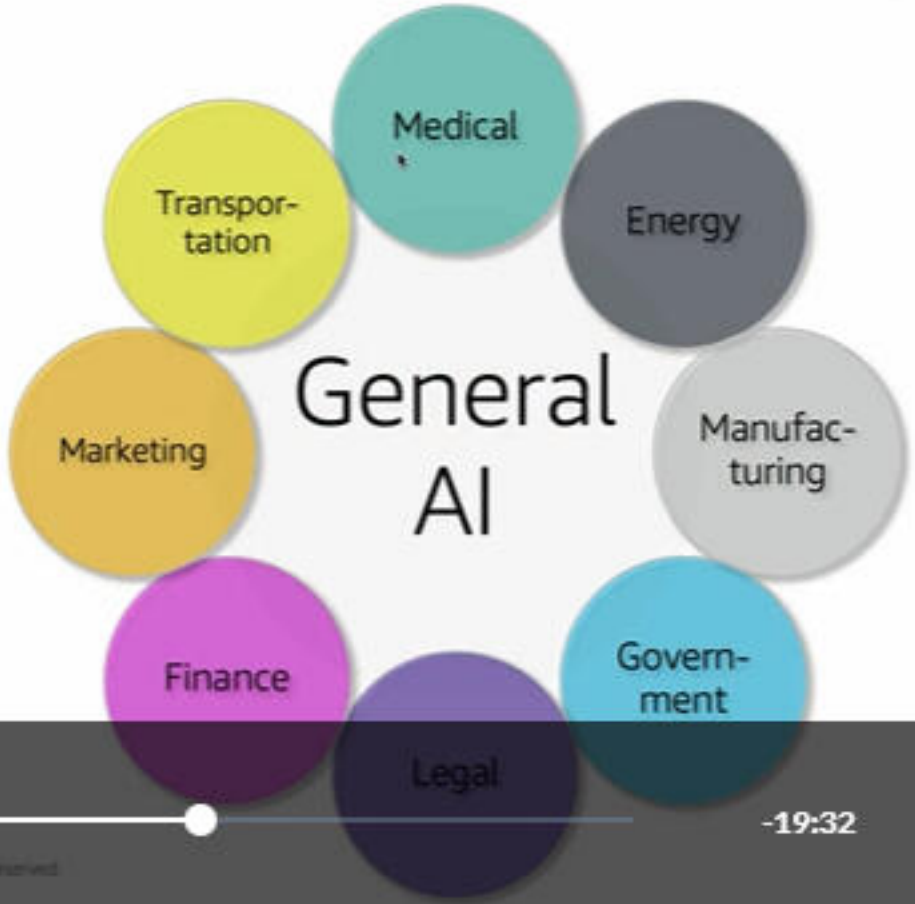
aws training and certification

	Statistical Definition	Everyday Definition
Label/Target	Dependent variable	What you are trying to predict
Feature	Independent variable	Data that helps you make predictions
Feature Engineering	Data transformation	Process of reshaping data to get more value out of it
Feature Selection	Variable/subset selection	Process of using the most valuable data

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Common use cases

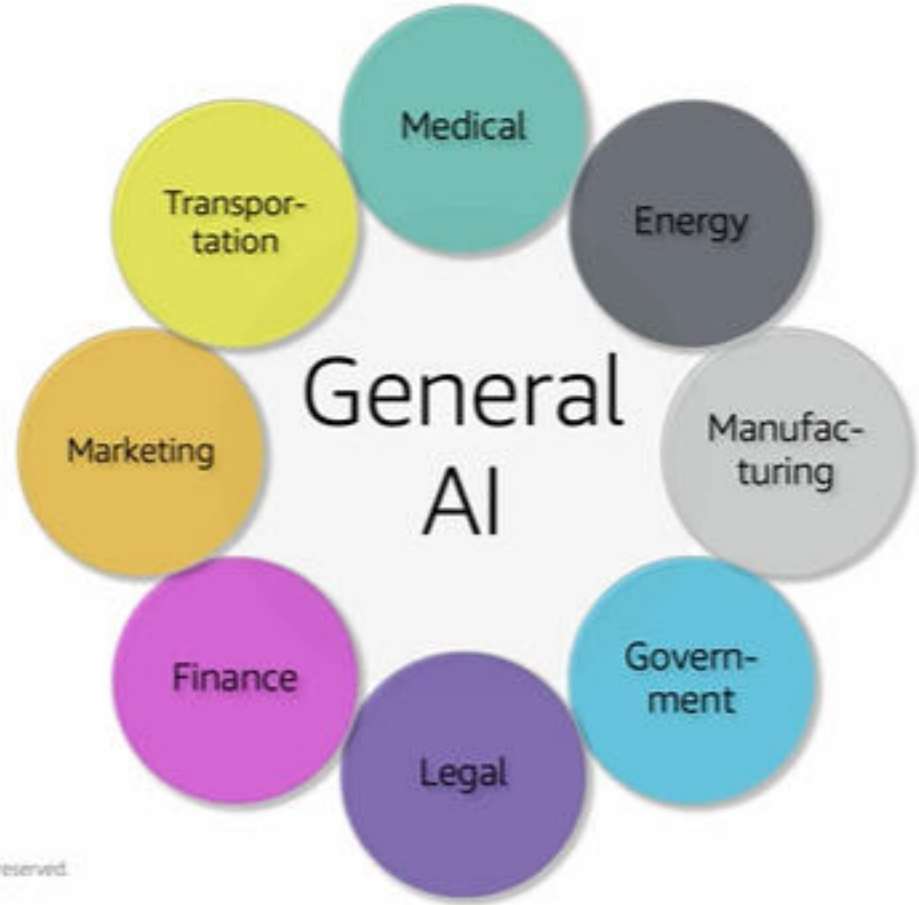
aws training and certification



-19:32 2x

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

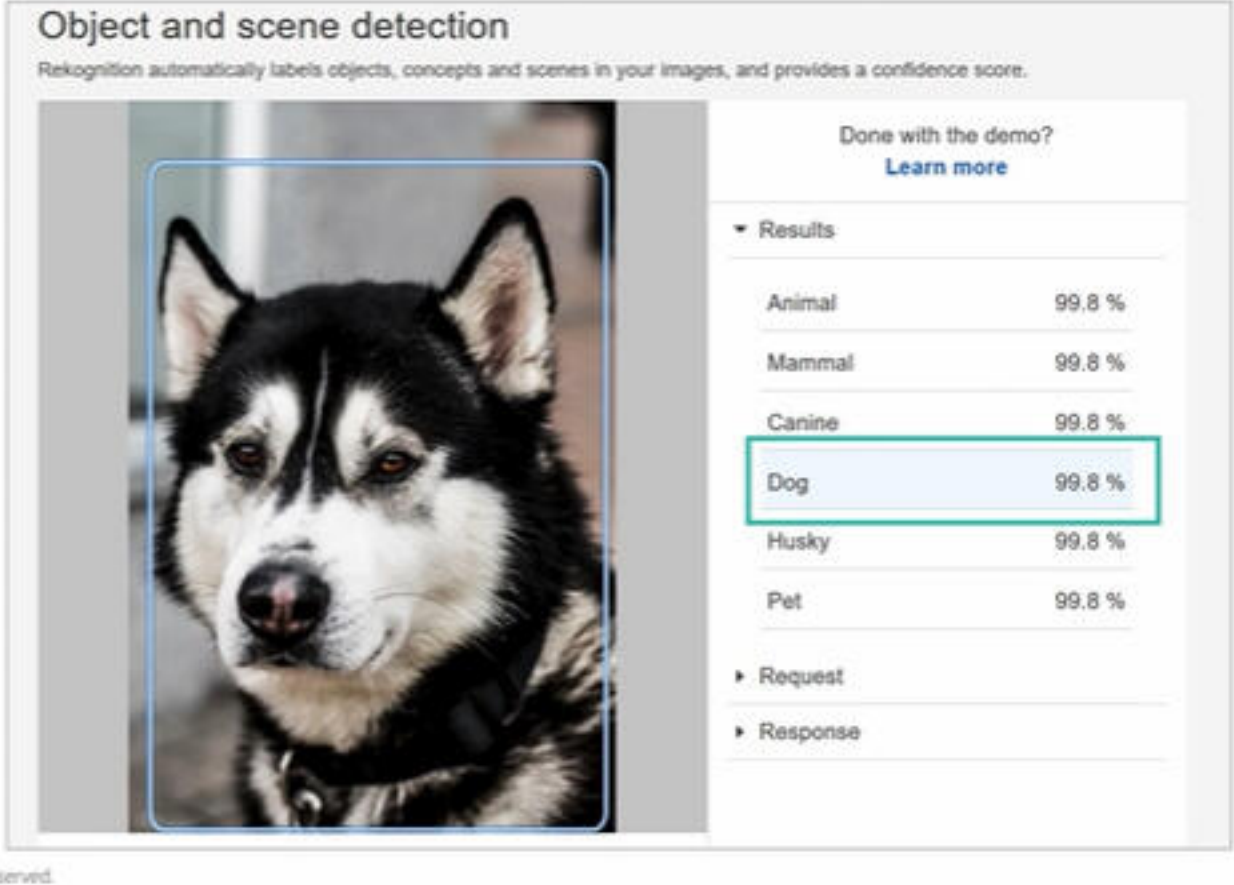
Common use cases



© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Image recognition

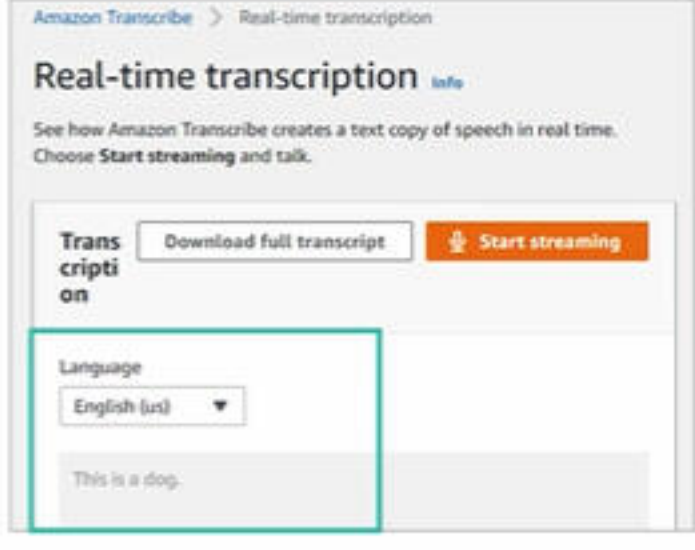
Amazon Rekognition

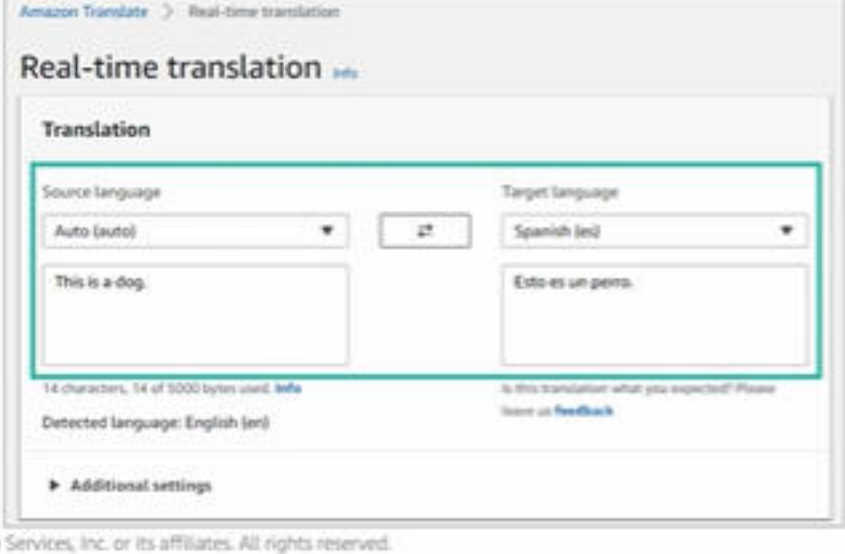


© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Translation and transcription

This is a dog.





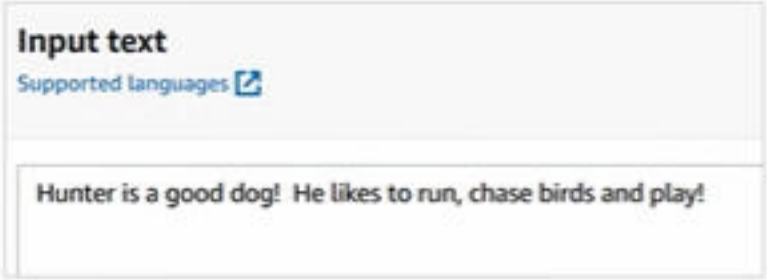
Amazon Translate

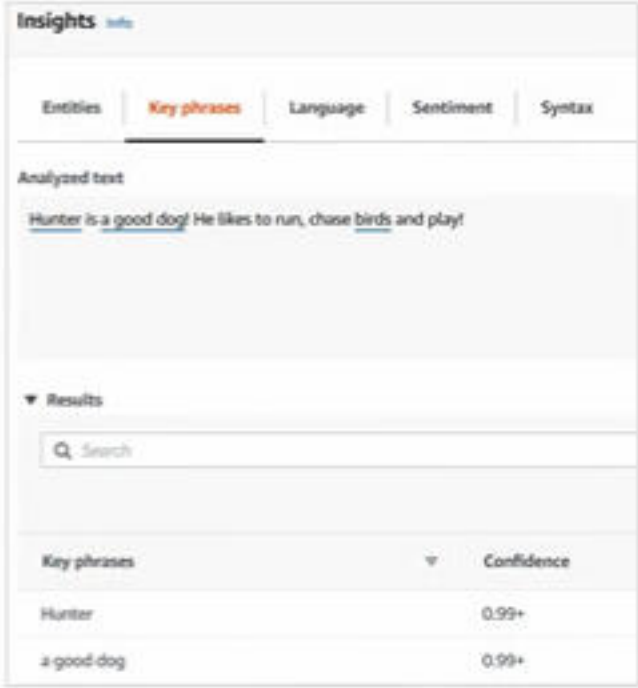
Amazon Transcribe

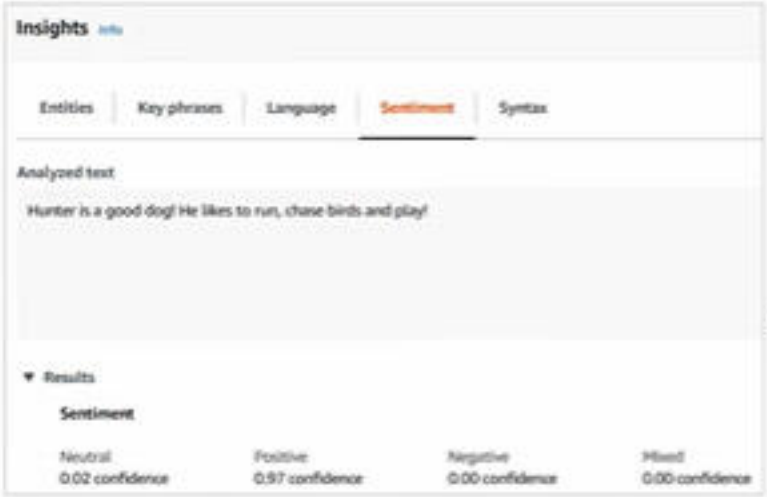
© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Text analysis

Amazon Comprehend







© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Key Challenges and Limitations of AI

aws training and certification

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Some AI limitations

aws training and certification

- **Data**: It fuels ML, so garbage-in / garbage-out still applies
- **Bias**: Poor predictions and decisions can be traced back to biased data
- **Explainability**: How and why the model makes certain predictions
- **Narrow AI**: Lack of generalized AI, where the current AI can only perform specific tasks within a narrow domain
- **Algorithm transparency**: Ability to see internal implementation used
- **Transfer learning**: Ability of ML models to take learning from one arena and apply to another
- **Emotional intelligence**: Basic human trait is still a challenge in ML

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Key challenges in ML: Biases and responsible AI

aws training and certification

- Biases in ML that we need to be aware of and deal with, such as
  - Sample bias
  - Prejudicial bias
  - Exclusion bias
  - Measurement bias
  - Algorithmic bias

https://developer.ibm.com/technologies/machine-learning/articles/machine-learning-and-bias

- Responsible AI: harnessing the power of AI in an ethical, fair, and responsible manner with full transparency, accountability, and freedom from bias.

https://www.pwc.co.uk/issues/data-analytics/artificial-intelligence/what-is-responsible-ai.html

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Key challenges in ML: Transparency

aws training and certification

- **Data lineage** tracks:
  - The origin of the data
  - What happens to it
  - Where it moves over time during data analytics processes
- **Reproducibility/Auditability**: Any results should be documented by making all data and code available in such a way that computations can be run again with identical results.
- Data lineage, reproducibility, and auditability are important to investigate potential bias in data and to ensure transparency.

Workbook: AnyCompany use case

aws training and certification

- Focused on many types of insurance
- Key challenge today is fraud
- Costing them millions of dollars per year
- Lots of historical data
- Wants to improve new claims fraud prediction
- Resolution needed by EOY

Knowledge check 1

aws training and certification

Which option uses deeply multilayered neural networks that perform tasks like speech and image recognition?

A. Artificial intelligence

B. Machine learning

C. Deep learning

D. Reinforcement learning

Knowledge check 2

aws training and certification

Which ML algorithm would you use for forecasting the demand for a product?

A. Clustering

B. Classification

C. Regression

D. Correct Action Rewarded

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Knowledge check 2

aws training and certification

Which ML algorithm would you use for forecasting the demand for a product?

A. Clustering

B. Classification

C. Regression

D. Correct Action Rewarded

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws training and certification

Module 2:

AI Services on AWS

- Overview of AWS ML/AI Portfolio
- Identify and describe key AWS AI services and Identify their use cases

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Our mission at AWS

Put machine learning in the hands of every developer

Our mission at AWS

Put machine learning in the hands of every developer

Why AWS for ML?

aws training and certification

Broad and deep set of ML and AI services

200+ new features and services launched within the last year

Solutions for everyone from ML scientists to application developers

Support for all three major frameworks

Machine learning with Amazon SageMaker

Single integrated development environment (IDE) for the entire ML workflow

At least 54% lower total cost of ownership (TCO)

In the last 2 years, 10,000 customers have adopted Amazon SageMaker

Comprehensive cloud offerings

Highly secure, reliable, fully featured data store

Strong set of compute, storage, security, database, and analytics capabilities to build on

85% of TensorFlow in the cloud runs in the AWS Cloud

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon ML stack – AI services

aws training and certification

AI services

Vision | Speech | Text | Search | Chatbots | Personalize | Forecast | Fraud | Development | Contact Centers

ML services

Amazon SageMaker | Ground Truth | AWS Marketplace for ML | SageMaker Studio IDE | Neo | Augmented AI

ML frameworks and infrastructure

TensorFlow | MXNet | Gluon | Keras | PyTorch | DeepGraphLibrary | Deep Learning AMIs and Containers | GPUs and CPUs | Elastic Inference | Inferencia | FPGA

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon ML stack – AI services

aws training and certification

AI services

Vision | Speech | Text | Search | Chatbots | Personalize | Forecast | Fraud | Development | Contact Centers

ML services

Amazon SageMaker | Ground Truth | AWS Marketplace for ML | SageMaker Studio IDE | Neo | Augmented AI

ML frameworks and infrastructure

TensorFlow | MXNet | Gluon | Keras | PyTorch | DeepGraphLibrary | Deep Learning AMIs and Containers | GPUs and CPUs | Elastic Inference | Inferentia | FPGA

-26:37 2x

© 2020 Amazon Web Services, Inc or its affiliates. All rights reserved.

Amazon Rekognition

aws training and certification

Amazon Rekognition

Function

Automate image and video analysis with machine learning

Use cases

- Media analysis
- Identity verification
- Content moderation

Key Features	
Labels	Custom labels
Content moderation	Text detection
Facial detection	Face search and verification
Celebrity recognition	Pathing

https://aws.amazon.com/rekognition

Highlighted customer

C-SPAN

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Rekognition

aws training and certification

Amazon Rekognition

Function

Automate image and video analysis with machine learning

Use cases

- Media analysis
- Identity verification
- Content moderation

Key Features	
Labels	Custom labels
Content moderation	Text detection
Facial detection	Face search and verification
Celebrity recognition	Pathing

https://aws.amazon.com/rekognition

Highlighted customer

C-SPAN

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Textract

aws training and certification

Amazon Textract

Function

Extract any text and data from any document using machine learning and without manual effort

Use cases

- Create smart search indexes
- Build automated document processing workflows
- Maintain compliance in document archives

Benefits
Extract structured and unstructured data
Go beyond simple optical character recognition (OCR)
Security and compliance
Implement human reviews

https://aws.amazon.com/textract/

Highlighted customer

Filevine

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Comprehend

aws training and certification

Amazon Comprehend

Function

Discover insights and relationships in text

Use cases

- Call center analytics
- Index and search product reviews
- Personalize content on a website

Benefits
Get answers from text
Organize documents by topics
Train models on your own data
Support general and industry-specific text

https://aws.amazon.com/comprehend

Highlighted customer

FINRA

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Personalize

aws training and certification

Amazon Personalize

Function

Create real-time, personalized user experiences fast, at scale

Use cases

- Retail – Help customers discover products
- Media and entertainment – Recommend new content, based on preference

Benefits
Deliver recommendations in real time
Implement personalized recommendations, in days
Personalize touchpoints along the customer journey

https://aws.amazon.com/personalize

Highlighted customers

Retail – Subway

Media and Entertainment – Coursera

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

CORD-19 Search

Query the COVID-19 Open Research Dataset (CORD-19), with natural language questions and answers from Amazon Kendra. CORD-19 Search includes data from CORD-19 which has been processed using Amazon Comprehend Medical. Data is updated regularly.

Type a question, or click an example below

Example questions

What are COVID-19 symptoms?

How infectious is COVID-19?

What is the mean value of R0 for COVID-19?

When is the salivary viral load highest for COVID-19?

What are prevalent antibodies in saliva?

Are IL-6 inhibitors key to COVID-19?

How does the convalescent plasma therapy differ from a vaccination?

Is convalescent plasma therapy a precursor to vaccine?

What is the origin of COVID-19?

-13:02 2x

Amazon Kendra

aws training and certification

Amazon Kendra

Function

Enterprise search service powered by machine learning

Use cases

- Improve access to internal knowledge
- Enhance sales and customer support services
- Help customers find information efficiently

Benefits
Ask natural-language questions, get immediate answers
Bring data together with a few clicks
Constantly improve search results

https://aws.amazon.com/kendra

Highlighted customers

CORD-19 Search

COVID-19 Open Research Dataset (CORD-19)

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Forecast

aws training and certification

Amazon Forecast

Function

Time-series forecasting service

Use cases

- Product demand planning
- Financial planning
- Resource planning

Benefits
Reduce forecasting time from months to hours
Create any time series forecast
Secure business data

https://aws.amazon.com/forecast

Highlighted customer

Puget Sound Energy

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Forecast

aws training and certification

Amazon Forecast

Function

Time-series forecasting service

Use cases

- Product demand planning
- Financial planning
- Resource planning

Benefits
Reduce forecasting time from months to hours
Create any time series forecast
Secure business data

https://aws.amazon.com/forecast

Highlighted customer

Puget Sound Energy

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Fraud Detector

aws training and certification

Amazon
Fraud Detector

Function

Detect more online fraud faster

Use cases

Detect common types of fraud:

- New account
- Online payment
- Guest checkout
- Online service and loyalty program abuse

Benefits
Prevent and detect online fraud
Fraud detection in minutes
Customized for your unique business needs

https://aws.amazon.com/fraud-detector

Highlighted customers

Customers

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

```
        'n_faces',
        'label',
        'label_conf']

results_df = pd.DataFrame(index=image_keys, columns=cols)
results_df['bucket_name'] = BUCKET_NAME
results_df['image_key'] = image_keys
```

In [8]:
```
results_df.head()
```

Out[8]:

ominant_language_code	dominant_language_score	sentiment	sentiment_positive_score	sentiment_negative_score	sentiment_neutral_score	sentiment_mixed_sco
NaN	NaN	NaN	NaN	NaN	NaN	Na
NaN	NaN	NaN	NaN	NaN	NaN	Na
NaN	NaN	NaN	NaN	NaN	NaN	Na
NaN	NaN	NaN	NaN	NaN	NaN	Na
NaN	NaN	NaN	NaN	NaN	NaN	Na

Let's look at a sample of images

The following code snippet, displays 10 random. You can run it couple of times to get a sense of images in the archive.

In [9]:
```
def display_images(image_keys, n_row=5, n_col=2):
    '''Display sample of images stored locally
        Arguments:
            image_keys (list): List of image keys
            n_row (int): Number of rows in the image display
```

-7:02 2x

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
rekognition.detect_text()
```

In [11]:
```
rekognition = boto3.client('rekognition')
```

In [14]:
```
def extract_text_from_image(bucket_name, image_key):
    '''Extracts text from an image by calling Amazon Rekognition detect_text API
       Arguments:
           bucketName (str): Name of the S3 bucket containing images
           image_key (str): object key of the image
       Output:
           text (str): Space joined text lines found on the image
    '''
    ### Your code (less than 5 lines of code)
    ### https://docs.aws.amazon.com/rekognition/latest/dg/text-detecting-text-procedure.html
    response=rekognition.detect_text(Image={'S3Object':{'Bucket':bucket_name,'Name':image_key}})
    textDetections=response['TextDetections']
    text = ''
    for found in textDetections:
            if (found['Type']=='WORD'):
                text = text + found['DetectedText'] + ' '

    return(text.strip())
```

Sample test for the "extract_text_from_image" function (extracted text is in the title)

In [15]:
```
# Sample test for the "extract_text_from_image" function. Extracted text is in the title
sample_image = '182.jpg'
text = extract_text_from_image(BUCKET_NAME, sample_image)
print('Automatically extracted text: {}|'.format(text))
plot_image(sample_image, image_title=text)
```

```
Automatically extracted text: PUSH BUTTON WAIT FOR WALK SIGNAL|
```

182.jpg: PUSH BUTTON WAIT FOR WALK SIGNAL

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
rekognition.detect_text()
```

Task 1 Step 3: Find the image containing the word 'Hume' and submit your answer (YOUR INPUT NEEDED)

Based on the 'text_in_image' column of the results_df, which image has the word "Hume" (case insensitive) in it?

In [18]:
```
for index, row in results_df.iterrows():
    if 'hume' in row['text_in_image'].lower():
        print(index)
```

```
101.jpg
```

Do you see any images containing the word "Hume"? If yes, provide the 'task1_answer' variable below with the image name ('xxx.jpg'). Let's also plot to make sure this is the correct answer!

In [19]:
```
task1_answer = '101.jpg'### YOUR ANSWER IN THE FORM of 'xxx.jpg'
plot_image(task1_answer, image_title='')
```

101.jpg

-4:40 2x

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
rekognition.detect_text()
```

Do you see any images containing the word "Hume"? If yes, provide the 'task1_answer' variable below with the image name ('xxx.jpg'). Let's also plot to make sure this is the correct answer!

In [19]:
```
task1_answer = '101.jpg'### YOUR ANSWER IN THE FORM of 'xxx.jpg'
plot_image(task1_answer, image_title='')
```

101.jpg

##

SUBMISSION (DON'T CHANGE THE CODE

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
rekognition.detect_text()
```

SUBMISSION (DON'T CHANGE THE CODE)

Run the following cell which turns one row of the "results_df" corresponding to your answer into json and uploads it to S3. DON'T CHANGE THIS LINE. Just run it as it is once you have your answer in 'task1_answer'. This is your submission for Task #1. You'll receive credit shortly if you answer is correct.

##

In [20]:
```
# DON'T CHANGE THIS LINE OR YOUR SUBMISSION FAILS
submit_your_answer(results_df, 'task1_res.json', task1_answer)
```

Task 2: Detect Dominant Language in Text

Your manager is happy with your progress but tells you that some of the images contain languages other than English. It is important to know which ones, as it impacts image selection. So, she wonders if it is possible to detect the dominant language automatically.

Your task is first to find dominant language in the text of all images. Out of all images containing German text (not all images), which one has the highest dominant language confidence score. To do this, you are going to leverage detect_dominant_language API of Amazon Comprehend AI service. More about the 'detect_dominant_language' API:

https://docs.aws.amazon.com/comprehend/latest/dg/get-started-api-dominant-language.html

Boto3 Amazon Comprehend API 'detect_dominant_language' reference:

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/comprehend.html#Comprehend.Client.detect_dominant_language

Step #1 (YOUR INPUT NEEDED):

Complete "extract_dominant_language" function: This function uses Amazon Comprehend detect_dominant_language API to detect the dominant language in the text. Once you receive raw response from Amazon Comprehend, analyze it to see where you can find dominant language code and dominant language score. Return both the dominant language code and the dominant language score (detection confidence). The portion of the function you write shouldn't be more than a few lines of code.

Step #2:

Let's run the "extract_dominant_language" function you just completed on the text of all the images. It is going to extract dominant language and confidence score from 'text_in_image' column of 'results_df' and fill the 'dominant_language_code' and 'dominant_language_score' columns. You don't need to do anything (no coding) other than running the code block under Task 2 Step 2 subsection. Please makes sure the "extract_dominant_language" function in Step #1 works as expected before running this step. After it is done, take a look at "results_df" dataframe to

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
rekognition.detect_text()
```

##

In [20]:
```
# DON'T CHANGE THIS LINE OR YOUR SUBMISSION FAILS
submit_your_answer(results_df, 'task1_res.json', task1_answer)
```

Task 2: Detect Dominant Language in Text

Your manager is happy with your progress but tells you that some of the images contain languages other than English. It is important to know which ones, as it impacts image selection. So, she wonders if it is possible to detect the dominant language automatically.

Your task is first to find dominant language in the text of all images. Out of all images containing German text (not all images), which one has the highest dominant language confidence score. To do this, you are going to leverage detect_dominant_language API of Amazon Comprehend AI service. More about the 'detect_dominant_language' API:

https://docs.aws.amazon.com/comprehend/latest/dg/get-started-api-dominant-language.html

Boto3 Amazon Comprehend API 'detect_dominant_language' reference:

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/comprehend.html#Comprehend.Client.detect_dominant_language

Step #1 (YOUR INPUT NEEDED):

Complete "extract_dominant_language" function: This function uses Amazon Comprehend detect_dominant_language API to detect the dominant language in the text. Once you receive raw response from Amazon Comprehend, analyze it to see where you can find dominant language code and dominant language score. Return both the dominant language code and the dominant language score (detection confidence). The portion of the function you write shouldn't be more than a few lines of code.

Step #2:

Let's run the "extract_dominant_language" function you just completed on the text of all the images. It is going to extract dominant language and confidence score from 'text_in_image' column of 'results_df' and fill the 'dominant_language_code' and 'dominant_language_score' columns. You don't need to do anything (no coding) other than running the code block under Task 2 Step 2 subsection. Please makes sure the "extract_dominant_language" function in Step #1 works as expected before running this step. After it is done, take a look at "results_df" dataframe to make sure the 'dominant_language_code' and 'dominant_language_score' are populated. Other columns should be empty at this point.

Step #3 (YOUR INPUT NEEDED):

Out of all images with German text (language code: 'de'), which one has the highest 'dominant_language_score'? Code is provided to filter

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

rekognition.detect_text()

Step #1 (YOUR INPUT NEEDED):

Complete "extract_dominant_language" function: This function uses Amazon Comprehend detect_dominant_language API to detect the dominant language in the text. Once you receive raw response from Amazon Comprehend, analyze it to see where you can find dominant language code and dominant language score. Return both the dominant language code and the dominant language score (detection confidence). The portion of the function you write shouldn't be more than a few lines of code.

Step #2:

Let's run the "extract_dominant_language" function you just completed on the text of all the images. It is going to extract dominant language and confidence score from 'text_in_image' column of 'results_df' and fill the 'dominant_language_code' and 'dominant_language_score' columns. You don't need to do anything (no coding) other than running the code block under Task 2 Step 2 subsection. Please makes sure the "extract_dominant_language" function in Step #1 works as expected before running this step. After it is done, take a look at "results_df" dataframe to make sure the 'dominant_language_code' and 'dominant_language_score' are populated. Other columns should be empty at this point.

Step #3 (YOUR INPUT NEEDED):

Out of all images with German text (language code: 'de'), which one has the highest 'dominant_language_score'? Code is provided to filter "results_df" to rows with German text. You can then either use the 'index' or the 'image_key' column to extract the image file name. Once you find that image, populate "task2_answer" with the image file name. Your response should be a string in the form of 'xxx.jpg'.

Run 'upload_answer_to_s3' line to upload you answer as json to S3. You are done! You'll receive credit in few minutes if your answer is correct.

Task 2 Step 1- Detect Dominant Language (YOUR INPUT NEEDED)

Complete "extract_dominant_language" function: This function uses Amazon Comprehend detect_dominant_language API to detect the dominant language in the text. Once you receive raw response from Amazon Comprehend, analyze it to see where you can find dominant language code and dominant language score. Return both the dominant language code and the dominant language score (detection confidence). The portion of the function you write shouldn't be more than a few lines of code.

In [21]:
```
comprehend = s3 = boto3.client('comprehend')
```

In [22]:
```
def extract_dominant_language(text):
    '''Extracts dominant language by calling Amazon Comprehend 'detect_dominant_language' API
        Arguments:
            text (str): Text to be analyzed
        Output:
            dominant_language_code (str): Detected dominant language code
```

File Edit View Insert Cell Kernel Widgets Help

rekognition.detect_text()

comprehend.detect_dominant_language()

Task 2 Step 1- Detect Dominant Language (YOUR INPUT NEEDED)

Complete "extract_dominant_language" function: This function uses Amazon Comprehend detect_dominan language in the text. Once you receive raw response from Amazon Comprehend, analyze it to see where you can find dominant language code and dominant language score. Return both the dominant language code and the dominant language score (detection confidence). The portion of the function you write shouldn't be more than a few lines of code.

In [21]:
```
comprehend = s3 = boto3.client('comprehend')
```

In [22]:
```
def extract_dominant_language(text):
    '''Extracts dominant language by calling Amazon Comprehend 'detect_dominant_language' API
       Arguments:
           text (str): Text to be analyzed
       Output:
           dominant_language_code (str): Detected dominant language code
           dominant_language_score (float): Score that indicates the confidence level on detected dominant langu
    '''
    ### Your code (less than 5 lines of code)
    # https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/comprehend.html#Comprehend.Clie
    response = comprehend.detect_dominant_language(Text=text)
    print(response)
    dominant_language_code = ''
    dominant_language_score = 0
    for lang in response['Languages']:
        if (lang['Score'] > dominant_language_score):
            dominant_language_code = lang['LanguageCode']
            dominant_language_score = lang['Score']

    return dominant_language_code, dominant_language_score
```

In [23]:
```
# Sample test for the "extract_dominant_language" function
extract_dominant_language('I am learning AWS AI services. Allez les bleus!')
```

```
{'Languages': [{'LanguageCode': 'fr', 'Score': 0.5306553244590759}, {'LanguageCode': 'en', 'Score': 0.394967883
8253021}], 'ResponseMetadata': {'RequestId': '901b34e2-fe67-4199-98d3-70528a895680', 'HTTPStatusCode': 200, 'HT
TPHeaders': {'x-amzn-requestid': '901b34e2-fe67-4199-98d3-70528a895680', 'content-type': 'application/x-amz-jso
n-1.1', 'content-length': '113', 'date': 'Sun, 20 Dec 2020 17:37:12 GMT'}, 'RetryAttempts': 0}}
```

Out[23]:
```
('fr', 0.5306553244590759)
```

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
rekognition.detect_text()
```

```
comprehend.detect_dominant_language()
```

Complete "extract_dominant_language" function: This function uses Amazon Comprehend detect_dominant_language API to detect the dominant language in the text. Once you receive raw response from Amazon Comprehend, analyze it to see where yo
dominant language score. Return both the dominant language code and the dominant language score (dete
function you write shouldn't be more than a few lines of code.

In [21]:
```
comprehend = s3 = boto3.client('comprehend')
```

In [22]:
```
def extract_dominant_language(text):
    '''Extracts dominant language by calling Amazon Comprehend 'detect_dominant_language' API
       Arguments:
           text (str): Text to be analyzed
       Output:
           dominant_language_code (str): Detected dominant language code
           dominant_language_score (float): Score that indicates the confidence level on detected dominant langu
    '''
    ### Your code (less than 5 lines of code)
    # https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/comprehend.html#Comprehend.Clie
    response = comprehend.detect_dominant_language(Text=text)
    print(response)
    dominant_language_code = ''
    dominant_language_score = 0
    for lang in response['Languages']:
        if (lang['Score'] > dominant_language_score):
            dominant_language_code = lang['LanguageCode']
            dominant_language_score = lang['Score']

    return dominant_language_code, dominant_language_score
```

In [23]:
```
# Sample test for the "extract_dominant_language" function
extract_dominant_language('I am learning AWS AI services. Allez les bleus!')
```

```
{'Languages': [{'LanguageCode': 'fr', 'Score': 0.5306553244590759}, {'LanguageCode': 'en', 'Score': 0.394967883
8253021}], 'ResponseMetadata': {'RequestId': '901b34e2-fe67-4199-98d3-70528a895680', 'HTTPStatusCode': 200, 'HT
TPHeaders': {'x-amzn-requestid': '901b34e2-fe67-4199-98d3-70528a895680', 'content-type': 'application/x-amz-jso
n-1.1', 'content-length': '113', 'date': 'Sun, 20 Dec 2020 17:37:12 GMT'}, 'RetryAttempts': 0}}
```

Out[23]:
```
('fr', 0.5306553244590759)
```

Task 2 Step 2: Populate 'dominant_language_code' and 'dominant_language_score' columns of the dataframe

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

rekognition.detect_text()

comprehend.detect_dominant_language()

```
        Output:
            dominant_language_code (str): Detected dominant language code
            dominant_language_score (float): Score that indicates the confidence
    '''
    ### Your code (less than 5 lines of code)
    # https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/comprehend.html#Comprehend.Client.detect_dominant_language
    response = comprehend.detect_dominant_language(Text=text)
    print(response)
    dominant_language_code = ''
    dominant_language_score = 0
    for lang in response['Languages']:
        if (lang['Score'] > dominant_language_score):
            dominant_language_code = lang['LanguageCode']
            dominant_language_score = lang['Score']

    return dominant_language_code, dominant_language_score
```

In [23]:
```
# Sample test for the "extract_dominant_language" function
extract_dominant_language('I am learning AWS AI services. Allez les bleus!')
```

```
{'Languages': [{'LanguageCode': 'fr', 'Score': 0.53065532445907 59}, {'LanguageCode': 'en', 'Score': 0.394967883
8253021}], 'ResponseMetadata': {'RequestId': '901b34e2-fe67-4199-98d3-70528a895680', 'HTTPStatusCode': 200, 'HT
TPHeaders': {'x-amzn-requestid': '901b34e2-fe67-4199-98d3-70528a895680', 'content-type': 'application/x-amz-jso
n-1.1', 'content-length': '113', 'date': 'Sun, 20 Dec 2020 17:37:12 GMT'}, 'RetryAttempts': 0}}
```

Out[23]:
```
('fr', 0.53065532445907 59)
```

Task 2 Step 2: Populate 'dominant_language_code' and 'dominant_language_score' columns of the dataframe

Run the following cell once you completed "extract_dominant_language". It is going to extract dominant language and confidence score from 'text_in_image' column of 'results_df' and fill the 'dominant_language_code' and 'dominant_language_score' columns. You don't need to do anything (no coding) other than running the code block under Task 2 Step 2 subsection. Please makes sure the "extract_dominant_language" function in Step #1 works as expected before running this step. After it is done, take a look at "results_df" dataframe to make sure the 'dominant_language_code' and 'dominant_language_score' are populated. Other columns should be empty at this point. Note that in some cases it is not possible to extract the dominant language (maybe not enough text). Don't worry if you see those cases!

In [24]:
```
for i, (key, row) in enumerate(results_df.iterrows()):
    print('processing image: {}'.format(key))

    # Extract text from results_df
```

-3:53 2x

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
rekognition.detect_text()
```

```
comprehend.detect_dominant_language()
```

```
    dominant_language_code = ''
    dominant_language_score = 0
    for lang in response['Languages']:
        if (lang['Score'] > dominant_language_score):
            dominant_language_code = lang['LanguageCode']
            dominant_language_score = lang['Score']

    return dominant_language_code, dominant_language_score
```

In [23]:
```
# Sample test for the "extract_dominant_language" function
extract_dominant_language('I am learning AWS AI services. Allez les bleus!')
```

```
{'Languages': [{'LanguageCode': 'fr', 'Score': 0.5306553244590759}, {'LanguageCode': 'en', 'Score': 0.3949678838253021}], 'ResponseMetadata': {'RequestId': '901b34e2-fe67-4199-98d3-70528a895680', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': '901b34e2-fe67-4199-98d3-70528a895680', 'content-type': 'application/x-amz-json-1.1', 'content-length': '113', 'date': 'Sun, 20 Dec 2020 17:37:12 GMT'}, 'RetryAttempts': 0}}
```

Out[23]:
```
('fr', 0.5306553244590759)
```

Task 2 Step 2: Populate 'dominant_language_code' and 'dominant_language_score' columns of the dataframe

Run the following cell once you completed "extract_dominant_language". It is going to extract dominant language and confidence score from 'text_in_image' column of 'results_df' and fill the 'dominant_language_code' and 'dominant_language_score' columns. You don't need to do anything (no coding) other than running the code block under Task 2 Step 2 subsection. Please makes sure the "extract_dominant_language" function in Step #1 works as expected before running this step. After it is done, take a look at "results_df" dataframe to make sure the 'dominant_language_code' and 'dominant_language_score' are populated. Other columns should be empty at this point. Note that in some cases it is not possible to extract the dominant language (maybe not enough text). Don't worry if you see those cases!

In [24]:
```
for i, (key, row) in enumerate(results_df.iterrows()):
    print('processing image: {}'.format(key))

    # Extract text from results_df
    text_in_image = row['text_in_image']
    results_df.loc[key, 'text_in_image'] = text_in_image
    try:
        dominant_language_code, dominant_language_score = extract_dominant_language(text_in_image)
        results_df.loc[key, 'dominant_language_code'] = dominant_language_code
        results_df.loc[key, 'dominant_language_score'] = round(dominant_language_score, 2)
    except:
        print('dominant language extraction failed')
```

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
TPHeaders': {'x-amzn-requestid': '901b34e2-fe67-4199-98d3-70528a895680', 'content
n-1.1', 'content-length': '113', 'date': 'Sun, 20 Dec 2020 17:37:12 GMT'}, 'Retr
```

rekognition.detect_text()

comprehend.detect_dominant_language()

Out[23]: ('fr', 0.5306553244590759)

Task 2 Step 2: Populate 'dominant_language_code' and 'dominant_language_score' columns of the dataframe

Run the following cell once you completed "extract_dominant_language". It is going to extract dominant language and confidence score from 'text_in_image' column of 'results_df' and fill the 'dominant_language_code' and 'dominant_language_score' columns. You don't need to do anything (no coding) other than running the code block under Task 2 Step 2 subsection. Please makes sure the "extract_dominant_language" function in Step #1 works as expected before running this step. After it is done, take a look at "results_df" dataframe to make sure the 'dominant_language_code' and 'dominant_language_score' are populated. Other columns should be empty at this point. Note that in some cases it is not possible to extract the dominant language (maybe not enough text). Don't worry if you see those cases!

In [24]:
```
for i, (key, row) in enumerate(results_df.iterrows()):
    print('processing image: {}'.format(key))

    # Extract text from results_df
    text_in_image = row['text_in_image']
    results_df.loc[key, 'text_in_image'] = text_in_image
    try:
        dominant_language_code, dominant_language_score = extract_dominant_language(text_in_image)
        results_df.loc[key, 'dominant_language_code'] = dominant_language_code
        results_df.loc[key, 'dominant_language_score'] = round(dominant_language_score, 2)
    except:
        print('dominant language extraction  failed')
        pass
```

```
28-e37b-4f6c-9590-e93ce78953ba', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': 'b4fb2e28-e37b-4f6
c-9590-e93ce78953ba', 'content-type': 'application/x-amz-json-1.1', 'content-length': '64', 'date': 'Sun, 20 D
ec 2020 17:37:12 GMT'}, 'RetryAttempts': 0}}
processing image: 102.jpg
{'Languages': [{'LanguageCode': 'en', 'Score': 0.9984759092330933}], 'ResponseMetadata': {'RequestId': 'ae94f9
c2-da04-4ba3-bf54-92fd51ef208c', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': 'ae94f9c2-da04-4ba
3-bf54-92fd51ef208c', 'content-type': 'application/x-amz-json-1.1', 'content-length': '64', 'date': 'Sun, 20 D
ec 2020 17:37:12 GMT'}, 'RetryAttempts': 0}}
processing image: 103.jpg
{'Languages': [{'LanguageCode': 'en', 'Score': 0.9813656210899353}], 'ResponseMetadata': {'RequestId': 'b60de7
83-71d9-4895-af3b-6731d62ff74b', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': 'b60de783-71d9-489
5-af3b-6731d62ff74b', 'content-type': 'application/x-amz-json-1.1', 'content-length': '64', 'date': 'Sun, 20 D
ec 2020 17:37:12 GMT'}, 'RetryAttempts': 0}}
```

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
rekognition.detect_text()
```

```
comprehend.detect_dominant_language()
```

```
comprehend.detect_sentiment()
```

111.jpg	sagemaker-us-east-1-649088000912-ai	111.jpg	MAY PLEASE THEY BELIEVE NEVER THESE BE DAYS DE...	en	0.98	NEGAT	
163.jpg	sagemaker-us-east-1-649088000912-ai	163.jpg	OUT OF THE MOUNTAIN OF DESPAIR, A STONE OF HOPE	en	1	NEGATIVE	0.01
100.jpg	sagemaker-us-east-1-649088000912-ai	100.jpg	AUX HEROS DE LARMEE DORIENT ET DES TERRES LOIN...	fr	0.88	NEGATIVE	
202.jpg	sagemaker-us-east-1-649088000912-ai	202.jpg	FIRE EXTINGUISHER DO NOT BLOCK m-71 mzu-oqz-xm	en	0.95	NEGATIVE	0.01
...	...	...	...	...	...	...	...
138.jpg	sagemaker-us-east-1-649088000912-ai	138.jpg		NaN	NaN	NaN	NaN
139.jpg	sagemaker-us-east-1-649088000912-ai	139.jpg		NaN	NaN	NaN	NaN
143.jpg	sagemaker-us-east-1-649088000912-ai	143.jpg		NaN	NaN	NaN	NaN
159.jpg	sagemaker-us-east-1-649088000912-ai	159.jpg	ETRO	nl	0.25	NaN	NaN
189.jpg	sagemaker-us-east-1-649088000912-ai	189.jpg	Kiwitingsparafie & Stuurwiel SvT Tel: 020-421 9...	nl	0.31	NaN	NaN

94 rows × 13 columns

Did you find the most negative image? If yes, provide the 'task3_answer' variable below with the image name ('xxx.jpg'). Let's also plot to make sure

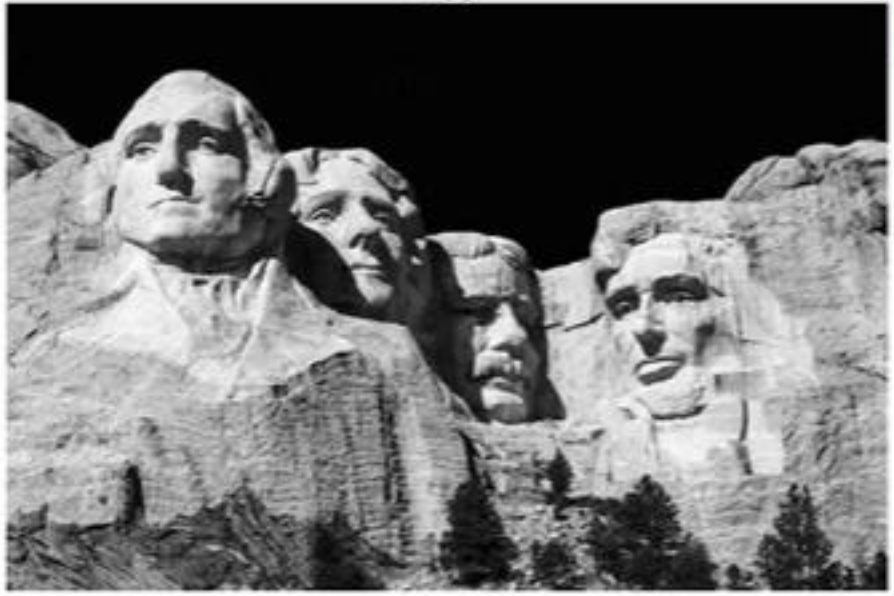
-1:50 2x

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
rekognition.detect_text()
```

143.jpg



```
comprehend.detect_dominant_language()
```

```
comprehend.detect_sentiment()
```

```
rekognition.detect_faces()
```

##

SUBMISSION (DON'T CHANGE THE CODE)

Run the following cell which turns one row of the "results_df" corresponding to your answer into json and uploads it to S3. DON'T CHANGE THIS LINE. Just run it as it is once you have your answer in 'task4_answer'. This is your submission for Task #4. You'll receive credit shortly if you answer is correct.

##

In [42]:
```
# DON'T CHANGE THIS LINE OR YOUR SUBMISSION FAILS
submit_your_answer(results_df, 'task4_res.json', task4_answer)
```

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

```
rekognition.detect_text()
```

```
comprehend.detect_dominant_language()
```

```
comprehend.detect_sentiment()
```

```
rekognition.detect_faces()
```

```
##########################################################################
```

In [42]:
```
# DON'T CHANGE THIS LINE OR YOUR SUBMISSION FAILS
submit_your_answer(results_df, 'task4_res.json', task4_answer)
```

Task 5: Label the images using Amazon Rekognition 'detect_labels' API!

Now it is your turn to surprise your manager. You think she will be impressed if you can extract label for of all the images aut

Label all the images using Amazon Rekognition "detect_labels" API. This API analyses an image and returns labels along with their c confidence score. More about the 'detect_labels' API:

https://docs.aws.amazon.com/rekognition/latest/dg/labels-detect-labels-image.html

Amazon Rekognition API 'detect_labels' reference:

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/rekognition.html#Rekognition.Client.detect_labels

Step #1 (YOUR INPUT IS NEEDED):

Complete 'extract_image_labels' function: This function uses Amazon Rekognition 'detect_labels' API to extract labels from an image. Once you have response from 'detect_labels' API, extract top label and associate confidence score. The function should return these two values. The portion of the function you write shouldn't be more than a few lines of code. Examine the raw response from 'detect_labels' to understand how to extract top label and the associated confidence score.

Step #2:

Let's run the "extract_image_labels" function on the text of all images. It is going to extract label and confidence score on all images and fill the 'label' and 'label_conf' columns. You don't need to do anything (no coding) other than running the code block under Task 5 Step 2 subsection. Please makes sure the "extract_image_labels" function in Step #1 works as expected before running this step. After it is done, take a look at "results_df" dataframe to make sure the 'label' and 'label_conf' columns are populated. All columns should be populated at this point.

Step #3 (YOUR INPUT IS NEEDED):

Based on the 'label' column of the results_df, which image is labeled as 'bird' or 'animal'? To help, code is provided to sort the results_df based on 'label'. You can then either use the 'index' or the 'image_key' column to extract the image file name. Once you find that image, populate "task5_answer" with the image file name. Your response should be a string in the form of 'xxx.jpg'.

File Edit View Insert Cell Kernel Widgets Help Trusted

Run 'upload_answer_to_s3' line to upload you answer as json to S3. You are done! You'll receive credit in few minutes if your answer is correct.

Task 5 Step 1- Find image labels (YOUR INPUT IS NEEDED):

Complete 'extract_image_labels' function: This function uses Amazon Rekognition 'detect_labels' API to extract labels from an image. Once you have response from 'detect_labels' API, extract top label and associated confidence score. The function should return these two values. The function that function you write shouldn't be more than a few lines of code. Examine the raw response from 'detect_labels' to understand and the associated confidence score.

In [43]:
```
def extract_image_labels(bucket_name, image_key):
    '''Extracts labels in the image using Amazon Rekognition
       Arguments:
           bucketName (str): Name of the S3 bucket containing images
           image_key (str): object key of the image
       Output:
           label (str): Label with highest score
           label_conf (float): Label confidence score
    '''
    ### Your code (less than 5 lines of code)
    #https://docs.aws.amazon.com/rekognition/latest/dg/labels-detect-labels-image.html
    # Call the API first
    response = rekognition.detect_labels(Image={'S3Object':{'Bucket':bucket_name,'Name':image_key}}, MaxLabels=1
    #print(response)
    label = ''
    label_conf = 0.1
    for foundLabel in response['Labels']:
        if (foundLabel['Confidence'] > label_conf):
            label = foundLabel['Name']
            label_conf = foundLabel['Confidence']

    return(label, label_conf)
```

In [44]:
```
## Sample test for your function
sample_image = '144.jpg'
label = extract_image_labels(BUCKET_NAME, sample_image)
print('Automatically labeled as: {}'.format(label))
plot_image(sample_image, image_title=results_df.loc[sample_image, 'text_in_image'])
```

```
Automatically labeled as: ('Road Sign', 99.79357147216797)
```

144.jpg: NARROW ROAD

rekognition.detect_text()

comprehend.detect_dominant_language()

comprehend.detect_sentiment()

rekognition.detect_faces()

rekognition.detect_labels()

-0:52 2x

File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code nbdiff

Run 'upload_answer_to_s3' line to upload you answer as json to S3. You are done! You'll receive credit in few minutes if your answer is correct.

Task 5 Step 1- Find image labels (YOUR INPUT IS NEEDED):

Complete 'extract_image_labels' function: This function uses Amazon Rekognition 'detect_labels' API to extract labels from an image. Once you have response from 'detect_labels' API, extract top label and associated confidence score. The function should return these two values. The entire of the function you write shouldn't be more than a few lines of code. Examine the raw response from 'detect_labels' to understand and the associated confidence score.

In [43]:
```
def extract_image_labels(bucket_name, image_key):
    '''Extracts labels in the image using Amazon Rekognition
       Arguments:
           bucketName (str): Name of the S3 bucket containing images
           image_key (str): object key of the image
       Output:
           label (str): Label with highest score
           label_conf (float): Label confidence score
    '''
    ### Your code (less than 5 lines of code)
    #https://docs.aws.amazon.com/rekognition/latest/dg/labels-detect-labels-image.html
    # Call the API first
    response = rekognition.detect_labels(Image={'S3Object':{'Bucket':bucket_name,'Name':image_key}}, MaxLabels=1
    #print(response)
    label = ''
    label_conf = 0.1
    for foundLabel in response['Labels']:
        if (foundLabel['Confidence'] > label_conf):
            label = foundLabel['Name']
            label_conf = foundLabel['Confidence']

    return(label, label_conf)
```

In [44]:
```
## Sample test for your function
sample_image = '144.jpg'
label = extract_image_labels(BUCKET_NAME, sample_image)
print('Automatically labeled as: {}'.format(label))
plot_image(sample_image, image_title=results_df.loc[sample_image, 'text_in_image'])
```

```
Automatically labeled as: ('Road Sign', 99.79357147216797)
```

144.jpg: NARROW ROAD

rekognition.detect_text()

comprehend.detect_dominant_language()

comprehend.detect_sentiment()

rekognition.detect_faces()

rekognition.detect_labels()

aws training and certification

Module 3:

ML Process

- Identify phases in the machine learning process
- Explain Machine Learning Pipeline
- Walk through the feature design process, requirements, and data dependencies for implementing ML on AWS
- Determine if a business problem can be solved with machine learning
- Identify your role, including how you can help customers
- Discuss Insurance Fraud Use Case

Module 3:

ML Process

- Identify phases in the machine learning process
- Explain Machine Learning Pipeline
- Walk through the feature design process, requirements, and data dependencies for implementing ML on AWS
- Determine if a business problem can be solved with machine learning
- Identify your role, including how you can help customers
- Discuss Insurance Fraud Use Case

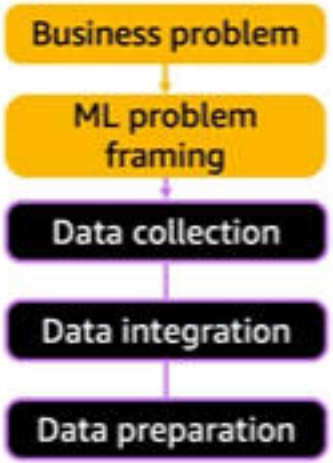
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Phases of Machine Learning & ML Pipeline

aws training and certification

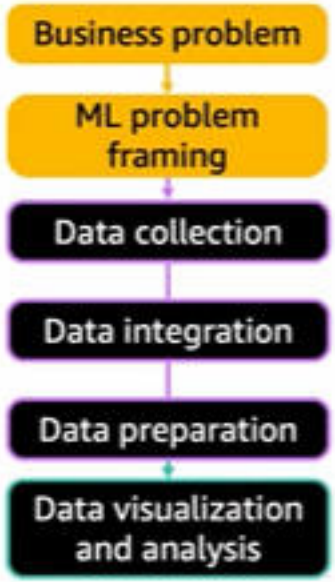
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML process overview



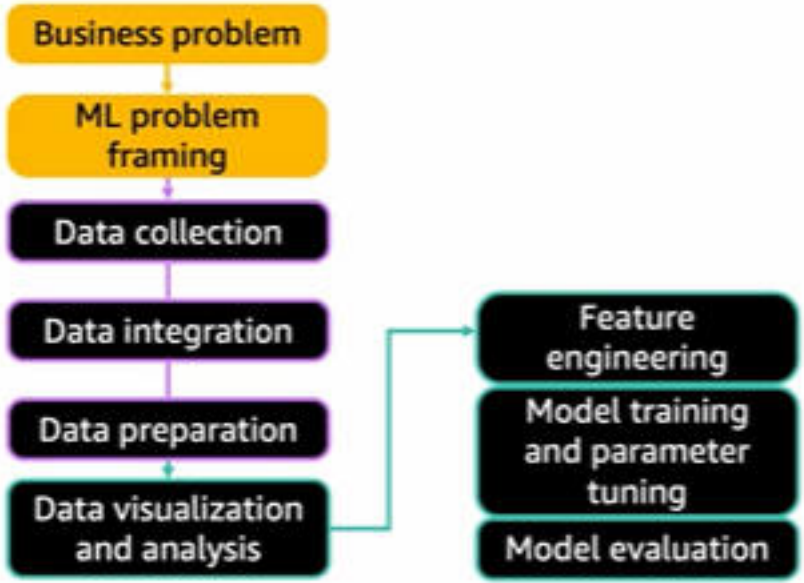
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML process overview



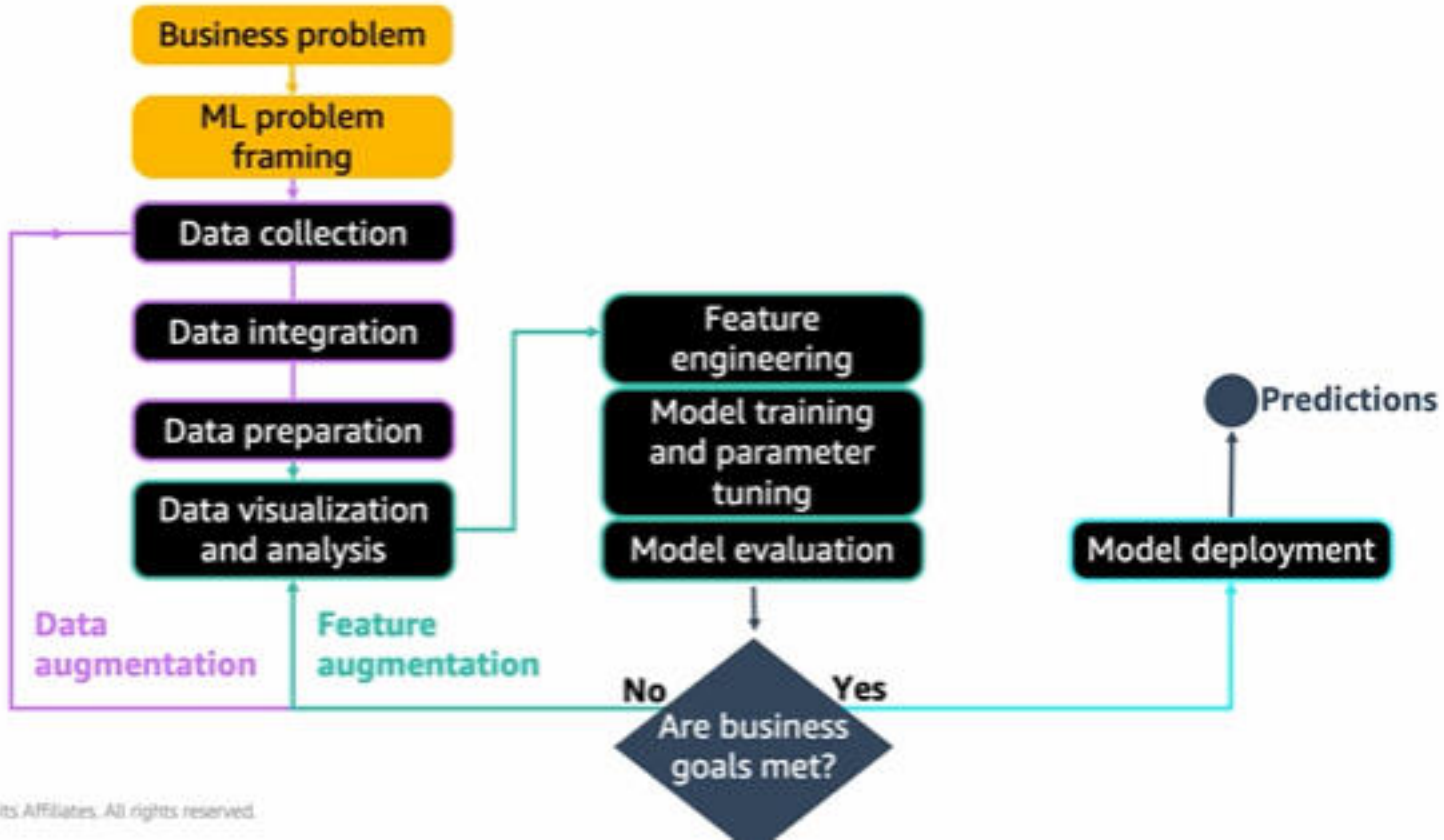
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML process overview



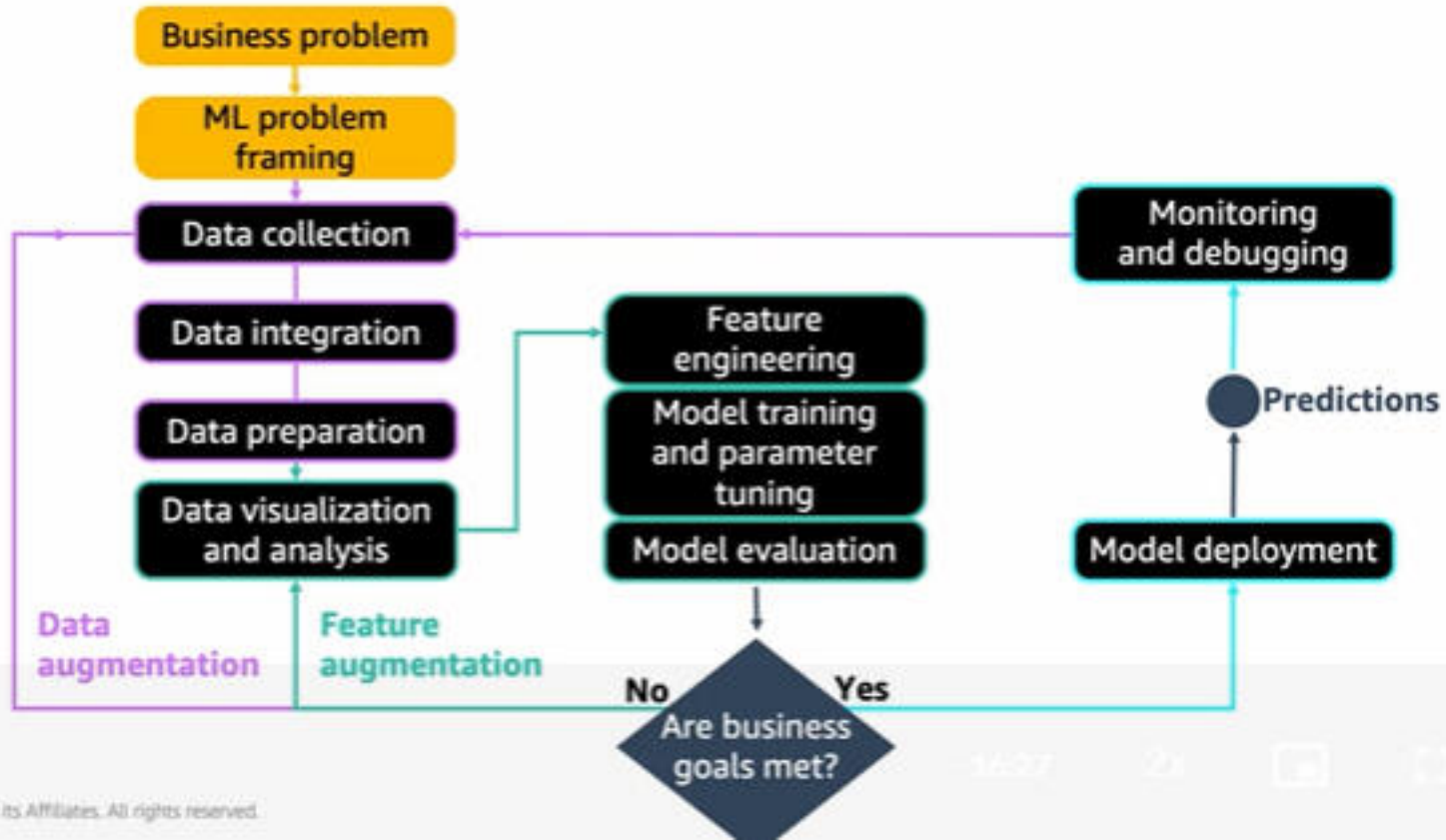
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML process overview



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML process overview

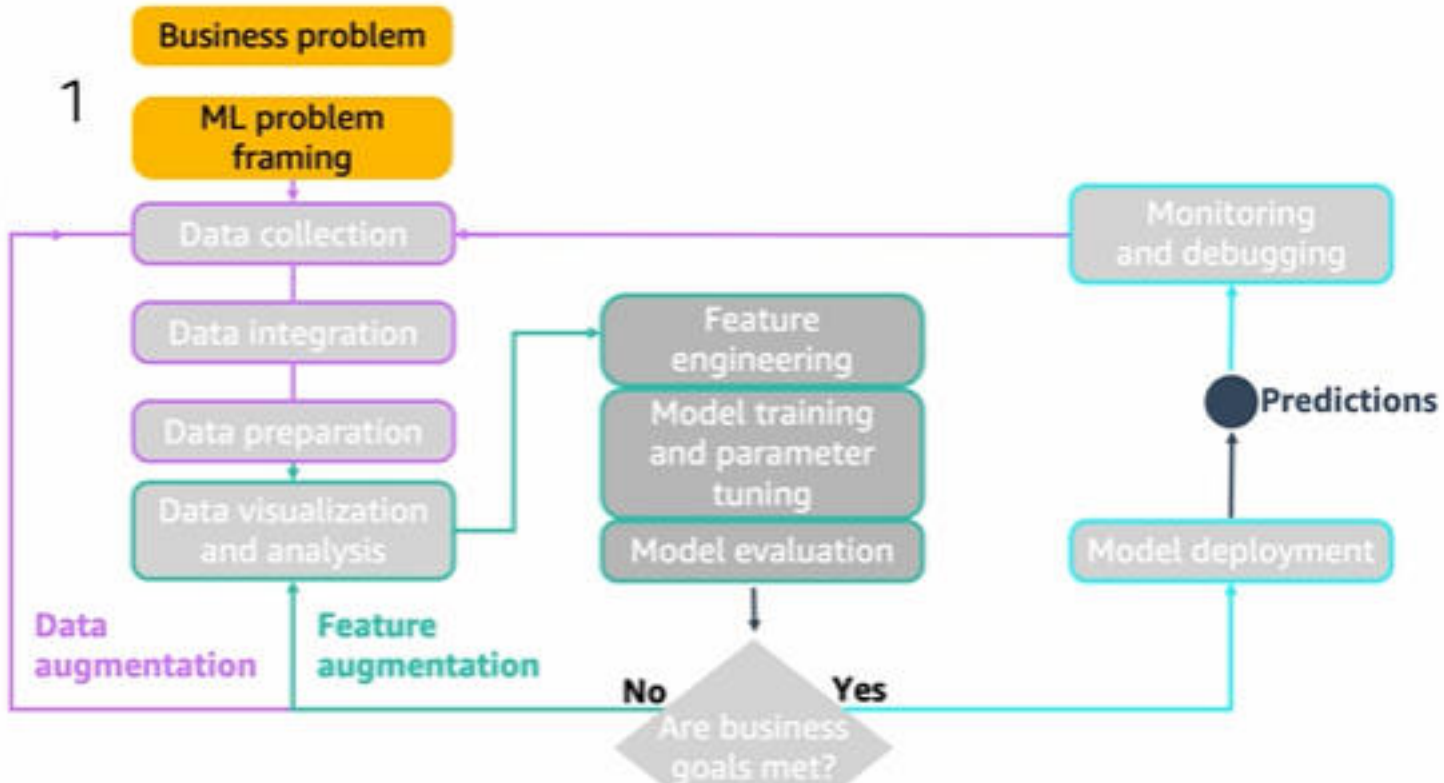


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

1. Business process: Framing a business problem

aws training and certification

- Can the business problem be framed as a ML problem?

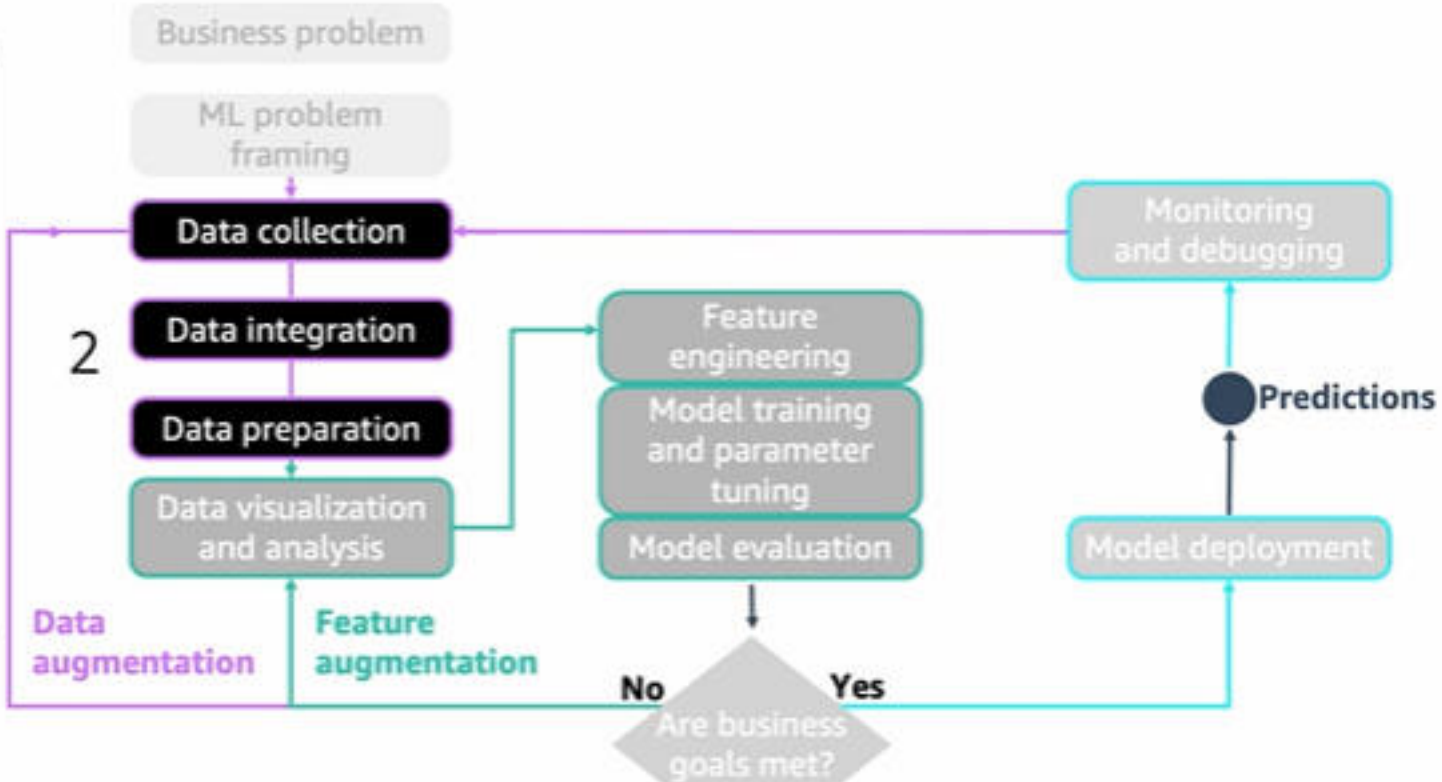


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

2. Data collection, integration, and preparation

aws training and certification

- Set up Data Pipeline to
  - Collect data
  - Store data
- Cleanse, analyze and prepare data
- Use different tools for these tasks

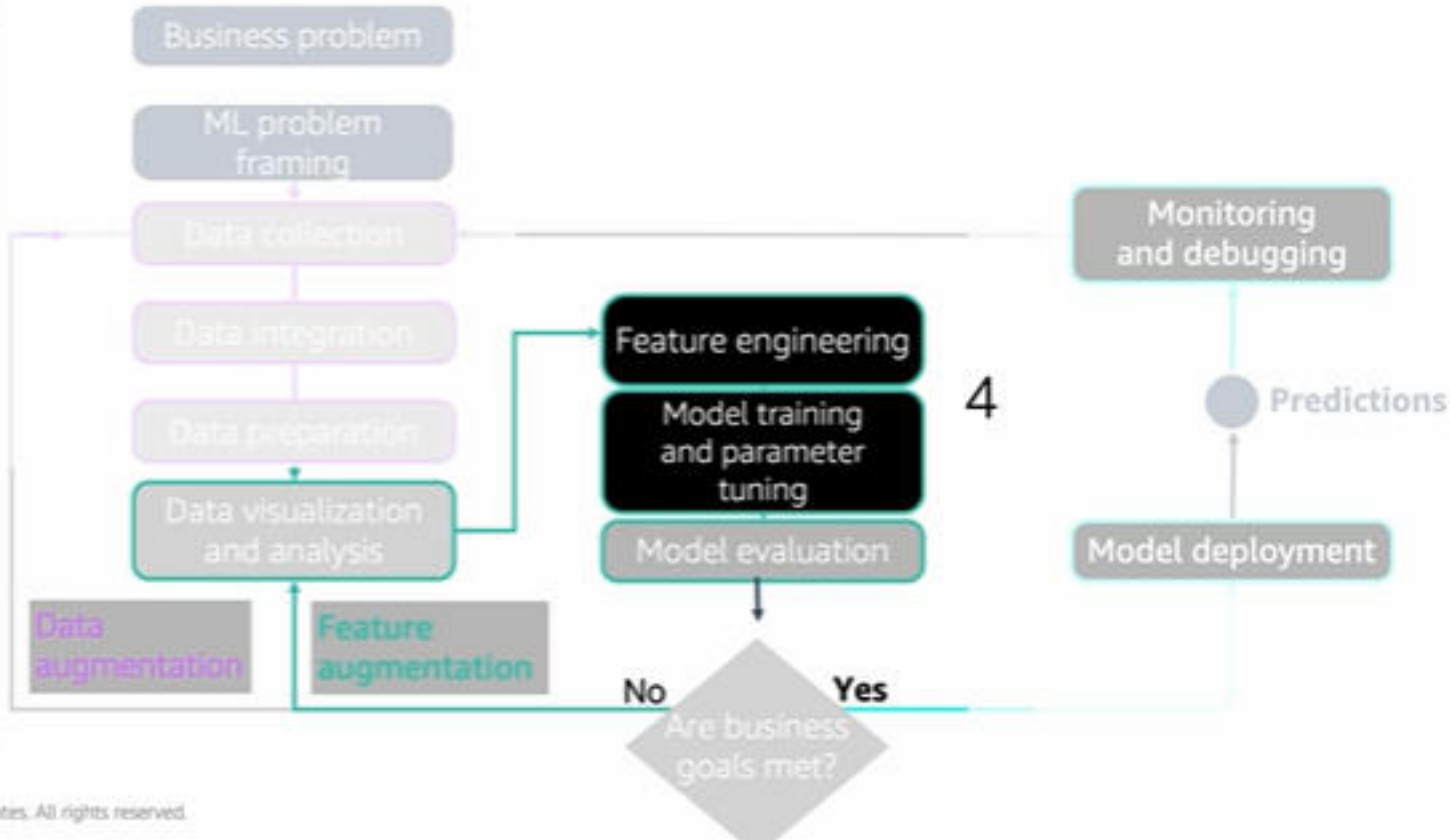


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

4. Feature engineering, model training, and parameter tuning

aws training and certification

- Perform feature engineering
- Set up and manage inference clusters
- Manage and scale model inference APIs
- Train, monitor and debug model predictions
- Model versioning and performance tracking

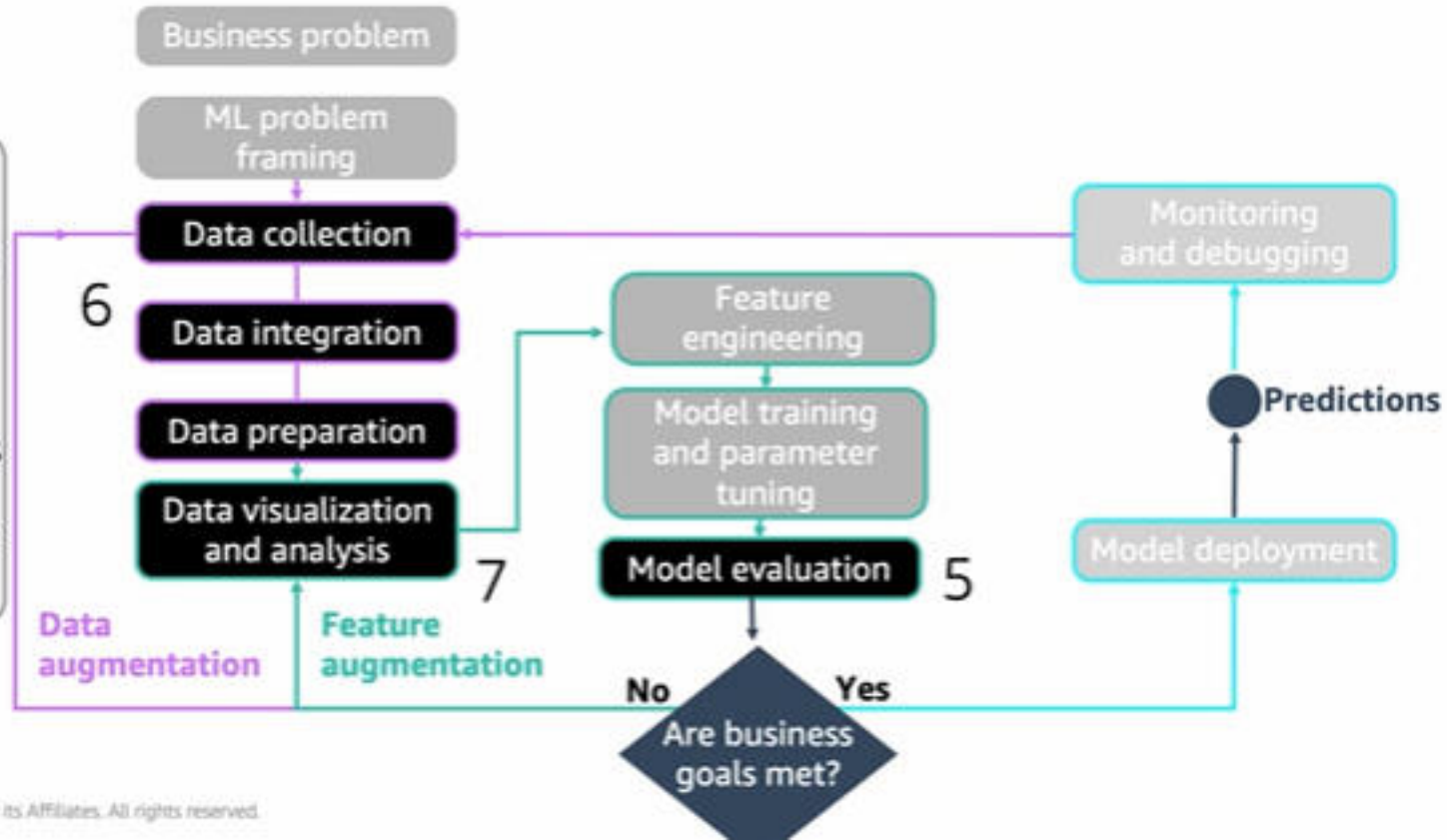


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

5, 6, 7 – Model evaluation, data and feature augmentations

aws training and certification

- Evaluate if model performance meets business goals
- Will feature augmentations or data augmentations lead to improve model performance

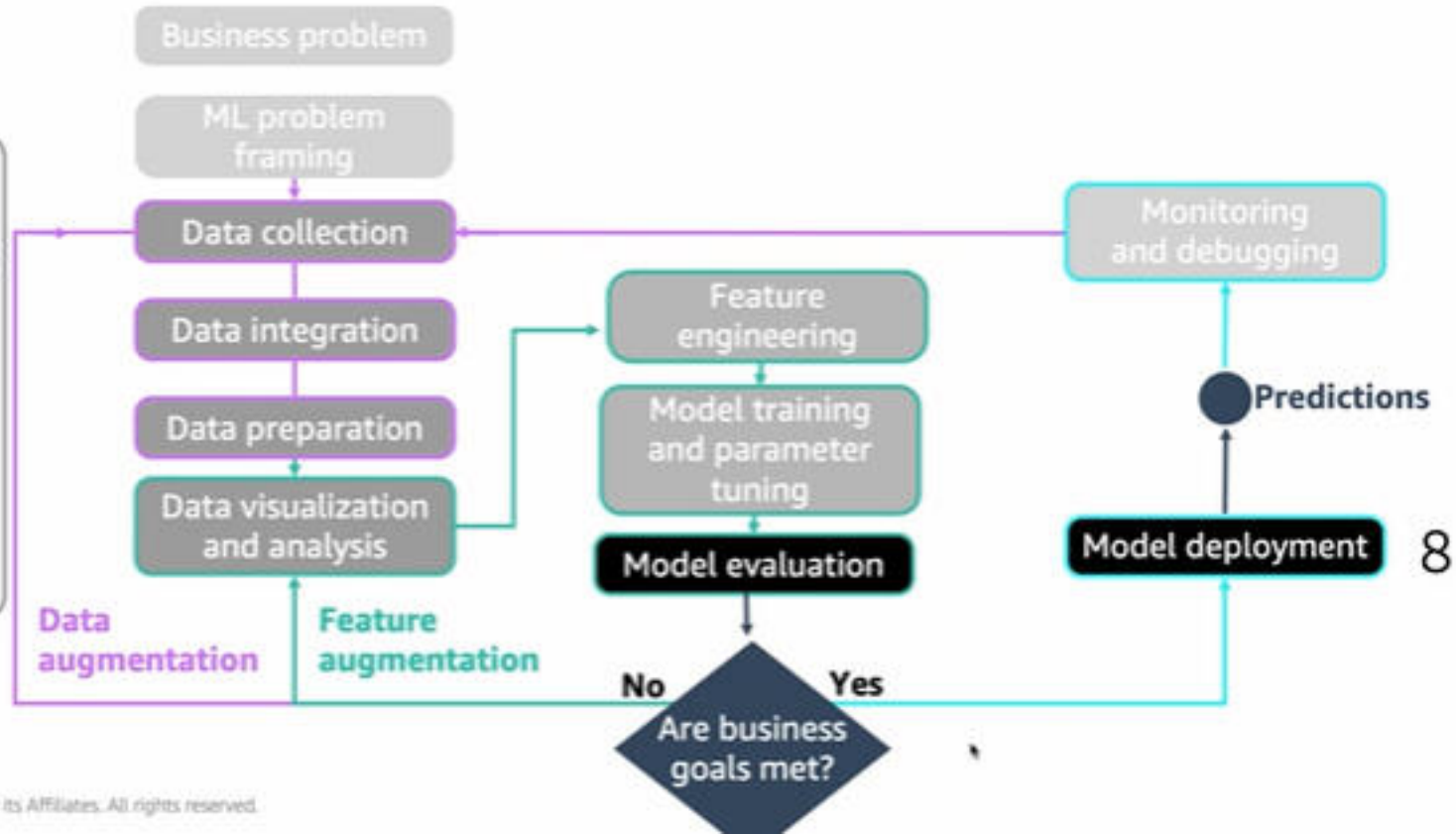


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

8. Model deployment

aws training and certification

- If the model performance meets business goals
- Work with DevOps to deploy the ML model

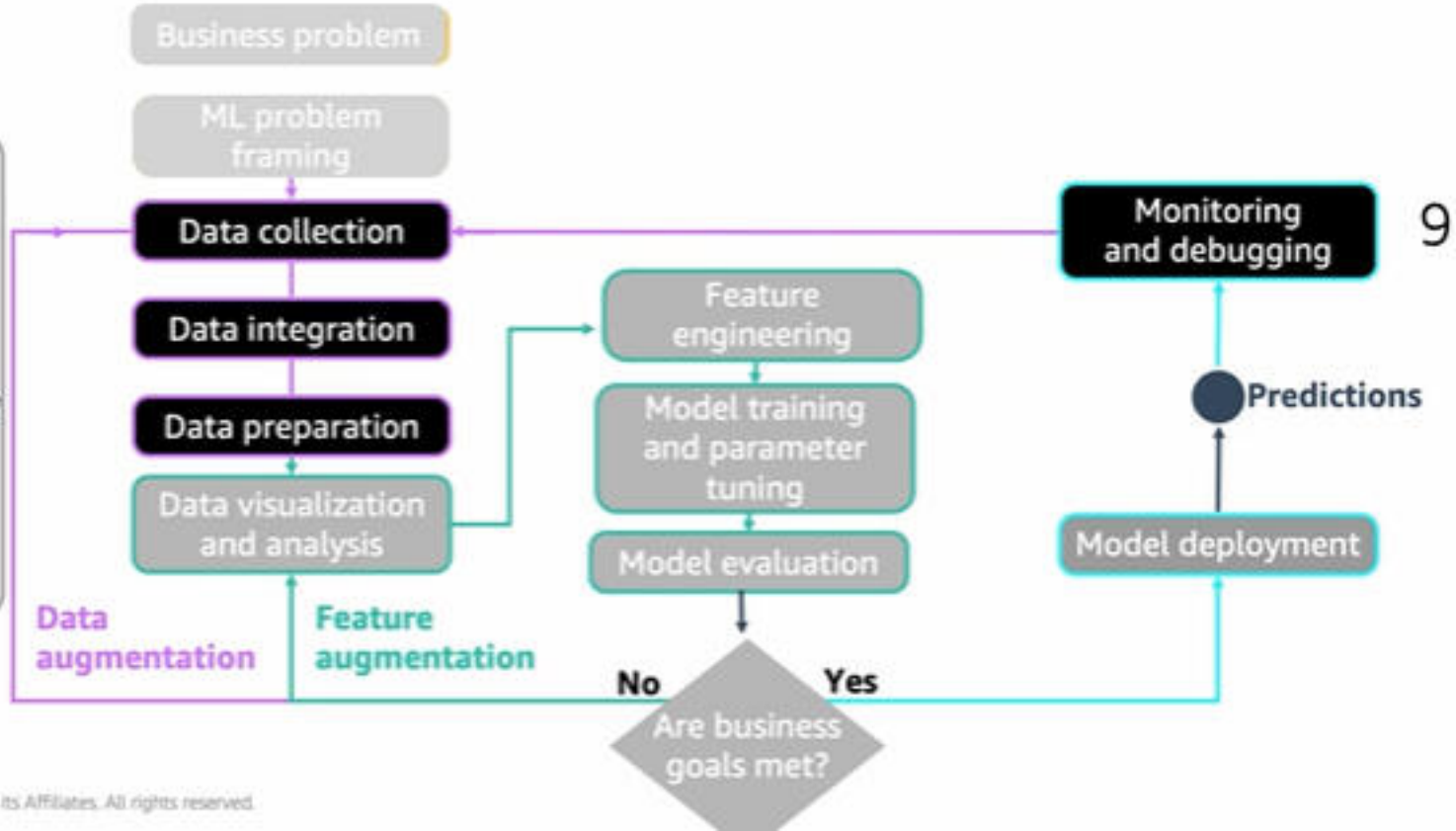


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

9. Model monitoring and debugging

aws training and certification

- Monitor for model performance and model drift
- To correct model drift, will need to go back to data collection and training phase

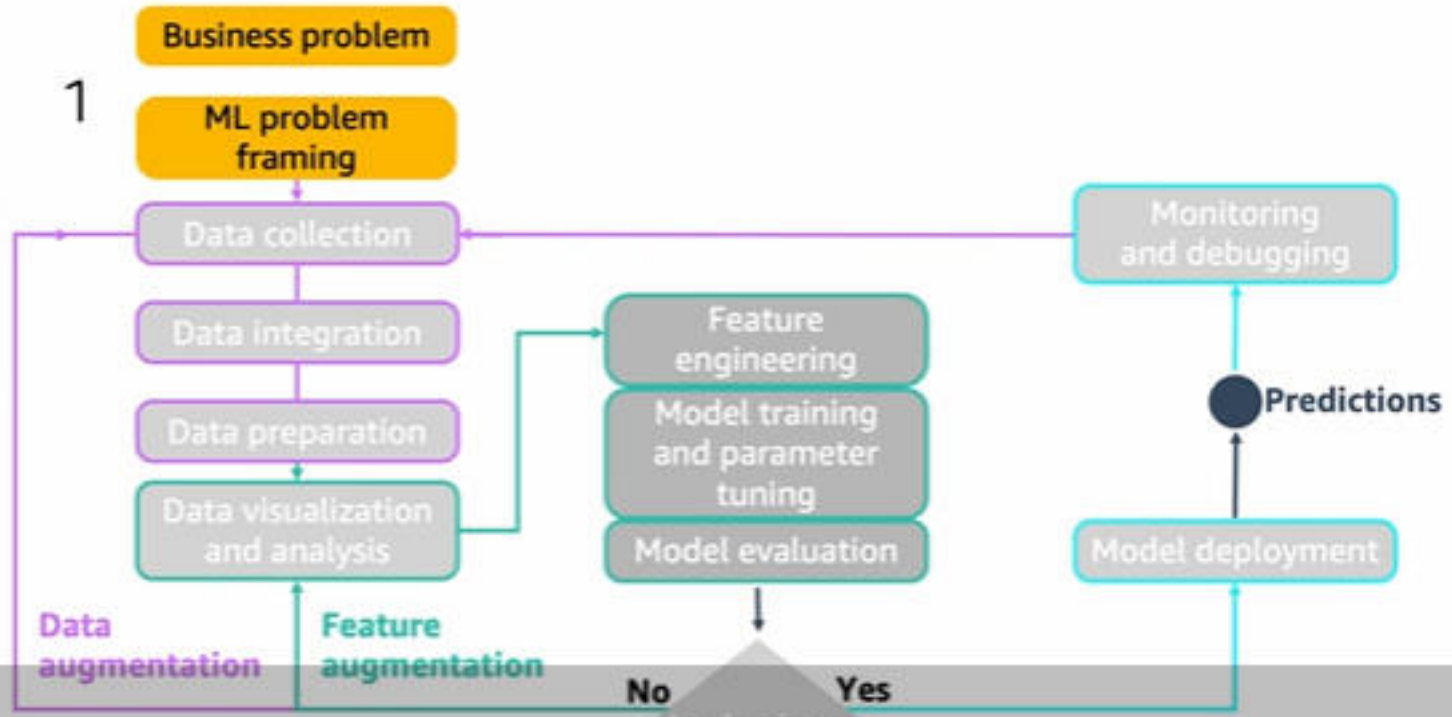


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

1. Business process: Framing a business problem

aws training and certification

- Can the business problem be framed as a ML problem?



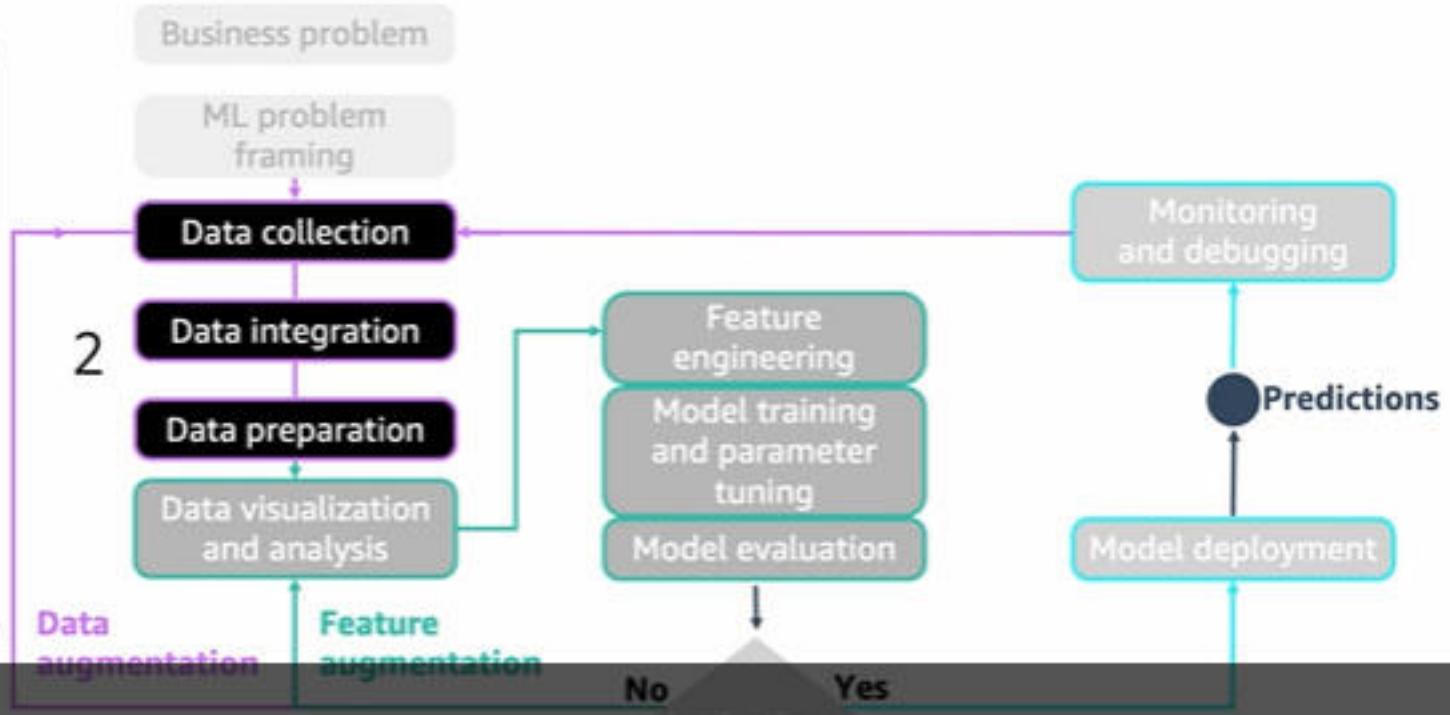
-13:07 2x

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

2. Data collection, integration, and preparation

aws training and certification

- Set up Data Pipeline to
  - Collect data
  - Store data
- Cleanse, analyze and prepare data
- Use different tools for these tasks



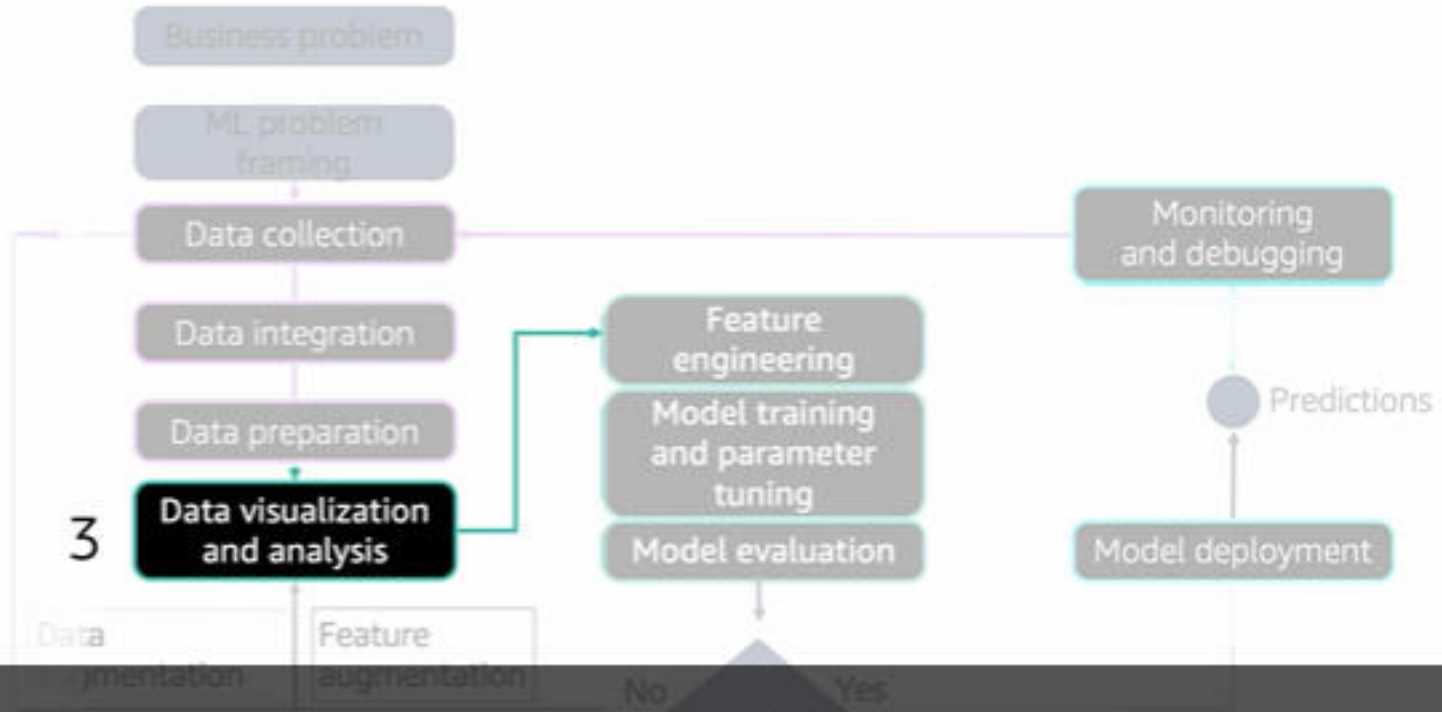
-12:31 2x

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

3. Data visualization

aws training and certification

- Set up and manage
  - Notebook environments
- Use statistical tools to visualize and analyze data



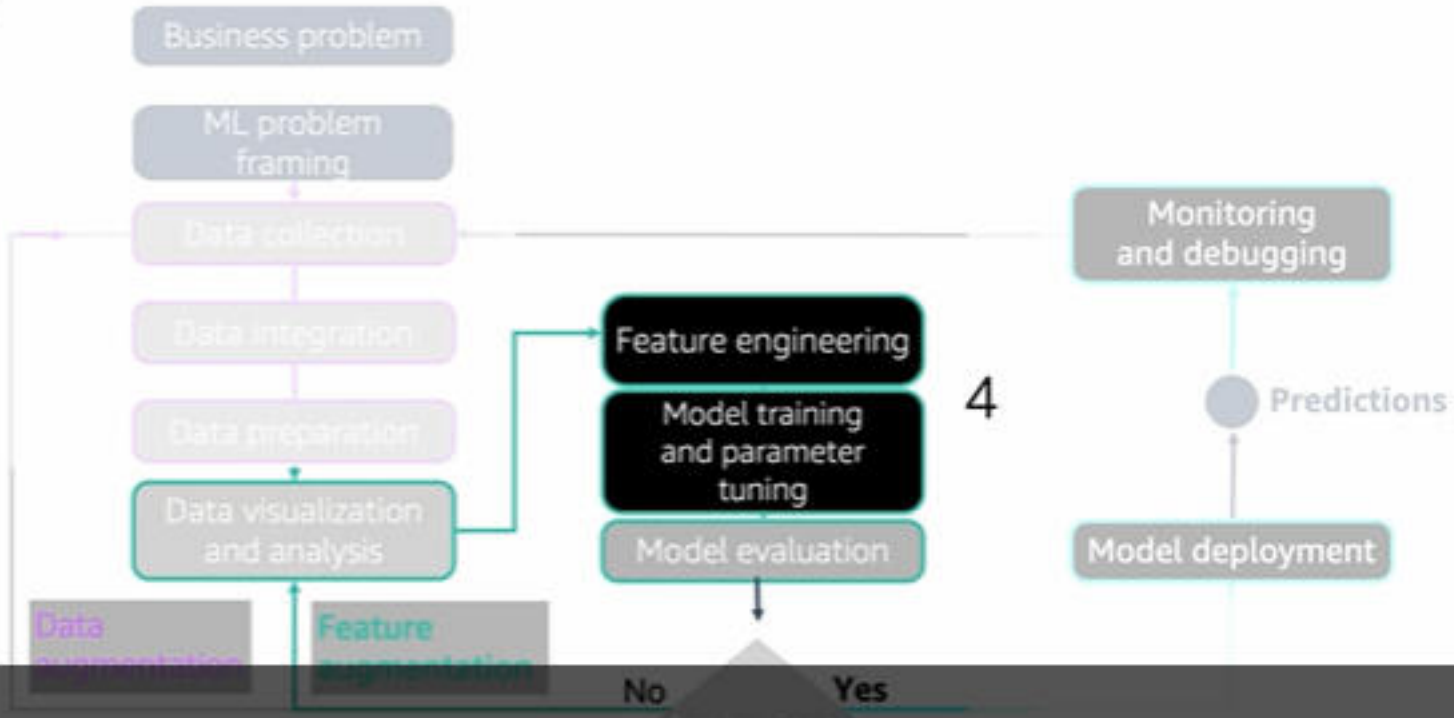
-11:51 2x

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

4. Feature engineering, model training, and parameter tuning

aws training and certification

- Perform feature engineering
- Set up and manage inference clusters
- Manage and scale model inference APIs
- Train, monitor and debug model predictions
- Model versioning and performance tracking



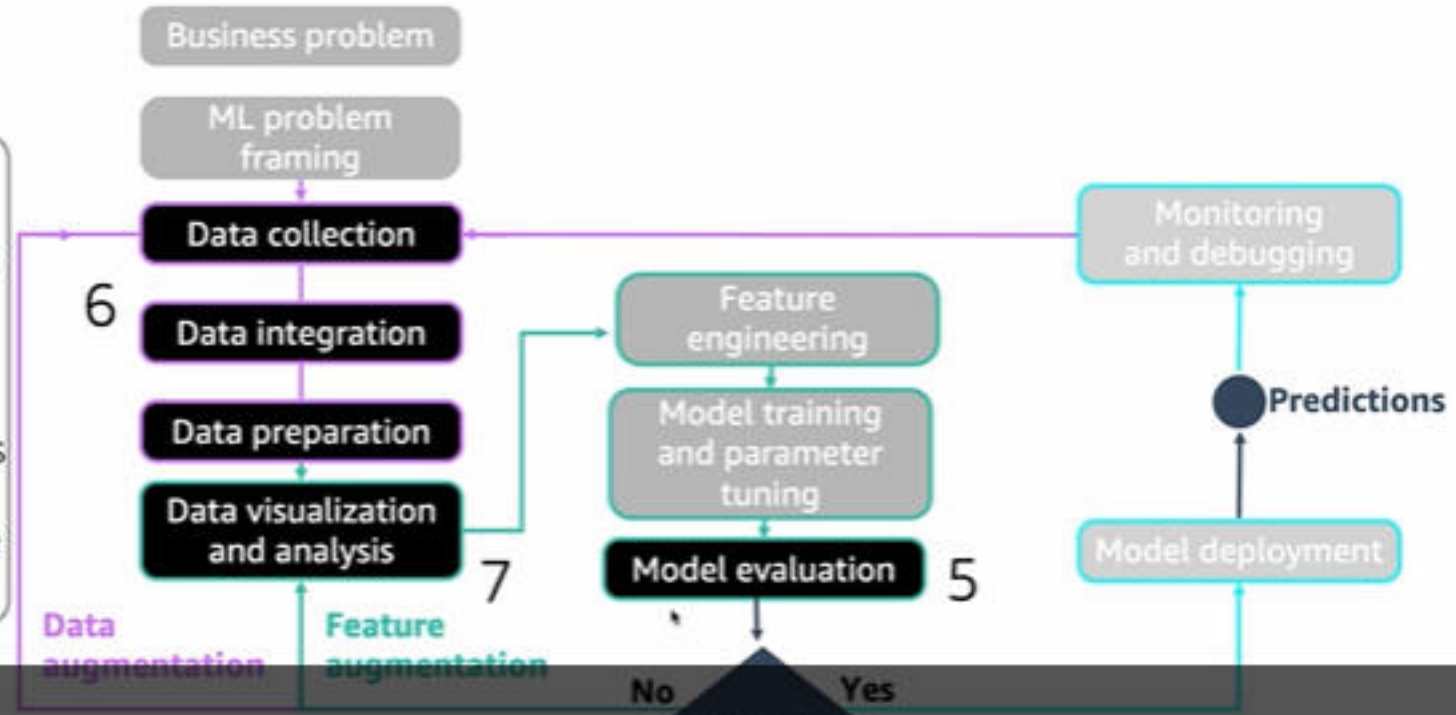
-10:55 2x

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

5, 6, 7 – Model evaluation, data and feature augmentations

aws training and certification

- Evaluate if model performance meets business goals
- Will feature augmentations or data augmentations lead to improve model performance



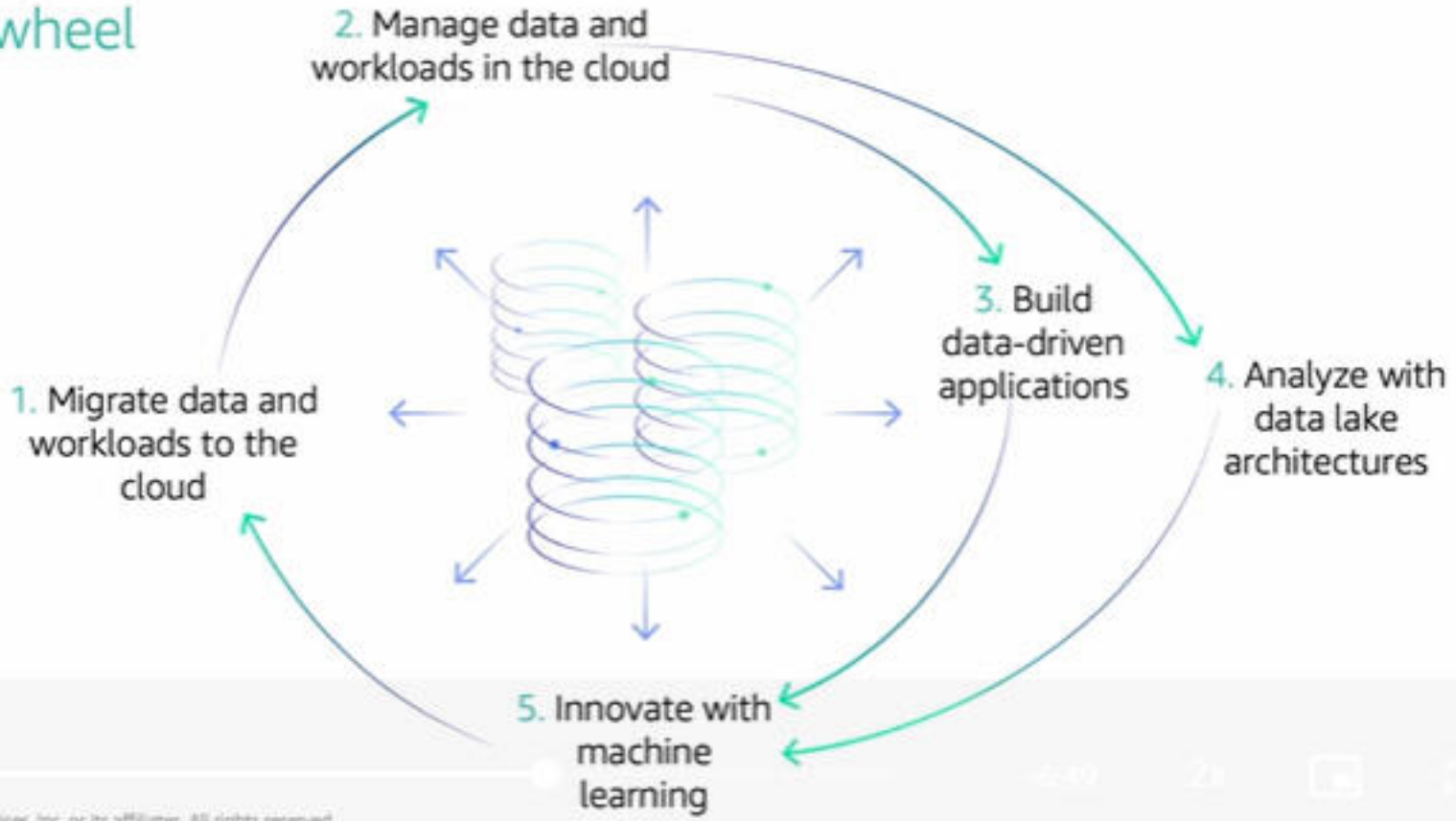
-9:54 2x

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML and the cloud journey

aws training and certification

Data Flywheel



© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

How you can help customers

aws training and certification

Customers

Just starting | Specific use case | Expert

Most business problems are not ML problems

Amazon AI/ ML

Pricing

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Workbook 3.1: Use case

aws training and certification

- Focused on many types of insurance
- Key challenge today is fraud
- Costing them millions of dollars per year
- Lots of historical data
- Wants to improve new claims fraud prediction
- Resolution needed by EOY

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Workbook 3.2

aws training and certification

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	policy_deductable	policy_annual_premium	umbrella_limit	insured_zip	police_report_available	total_claim_amount	injury_claim	property_claim	vehicle_claim	auto_make	auto_model	auto_year	fraud_reported
0	328	48	521585	2014-10-17	OH	250/500	1000	1406.91	0	466132	YES	71610	6510	13020	52080	Saab	92x	2004	Y
1	228	42	342868	2006-06-27	IN	250/500	2000	1197.22	5000000	468176	?	5070	780	780	3510	Mercedes	E400	2007	Y
2	134	29	687698	2000-09-06	OH	100/300	2000	1413.14	5000000	430632	NO	34650	7700	3850	23100	Dodge	RAM	2007	N
3	256	41	227811	1990-05-25	IL	250/500	2000	1415.74	6000000	608117	NO	63400	6340	6340	50720	Chevrolet	Tahoe	2014	Y
4	228	44	367455	2014-06-06	IL	500/1000	1000	1583.91	6000000	610706	NO	6500	1300	650	4550	Acura	RSX	2009	N

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

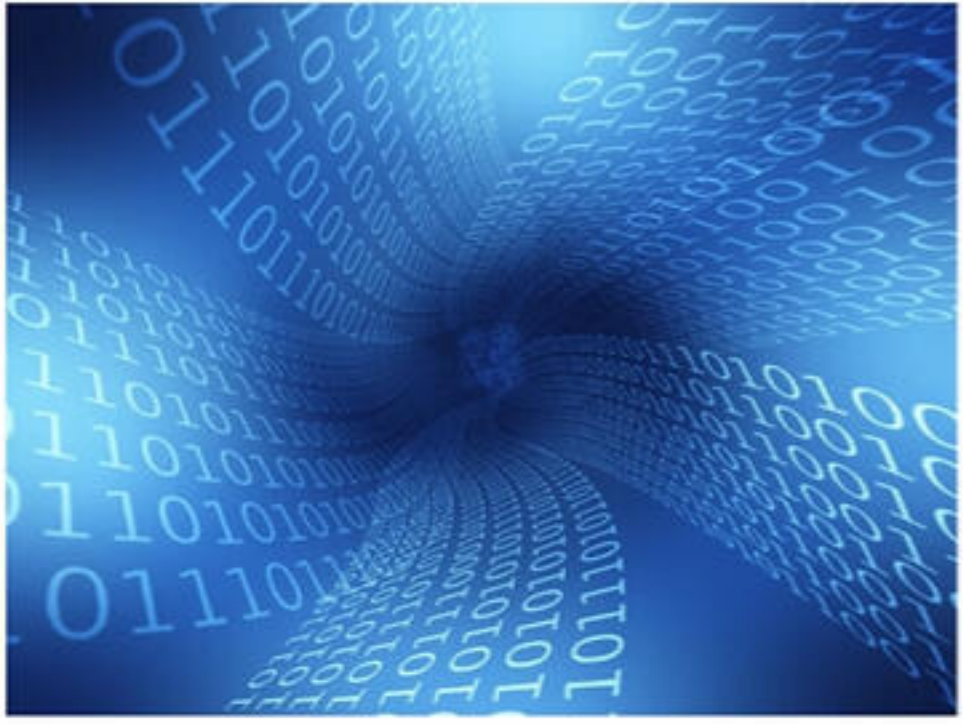
Module 4: Data Collection, Integration, Preparation, Visualization, and Analysis

- Explain why data is critical to machine learning problems
- Explain structured and unstructured data
- Explain the purpose of data phases in the ML process
- Identify AWS services that are used during data collection, integration, preparation, and visualization
- Explain Amazon SageMaker Data Wrangler
- Explain Amazon SageMaker Feature Store
- Explain the purpose of Amazon SageMaker Ground Truth and identify its use cases

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Data challenges in ML projects

aws training and certification



- ML project requires lots of data
- **ML requires clean data.**
- More than 50% of project time is spent gathering, cleansing, and visualizing data.
- 36% see dirty data as the #1 challenge.

Source Kaggle Machine Learning and Data Science Survey 2017:
https://www.kaggle.com/kaggle/kaggle-survey-2017

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Structured versus unstructured data

aws training and certification

Name	Color	Vegetable
Apple	Red	No
Broccoli	Green	Yes
Spinach	Green	Yes
Grape	Purple	No
Squash	Yellow	Yes

Structured
(relational databases, .csv files)

Unstructured
(documents,
audio, or video files)

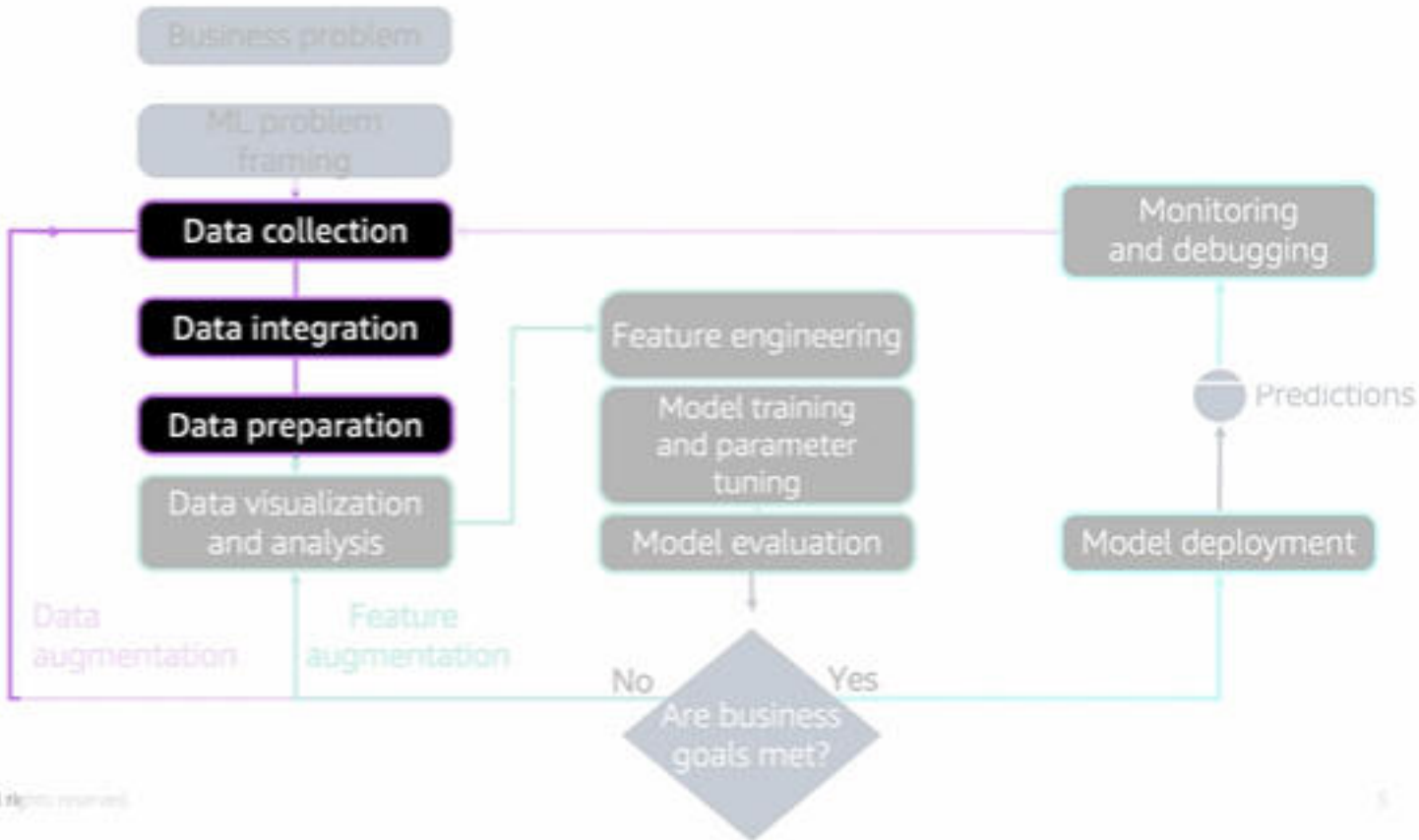
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML process: Collection, Integration

aws training and certification

Build the data platform:

- Amazon Simple Storage Service (Amazon S3)
- Amazon Athena
- Amazon EMR
- AWS Glue
- Amazon QuickSight
- AWS Lake Formation
- Amazon Redshift



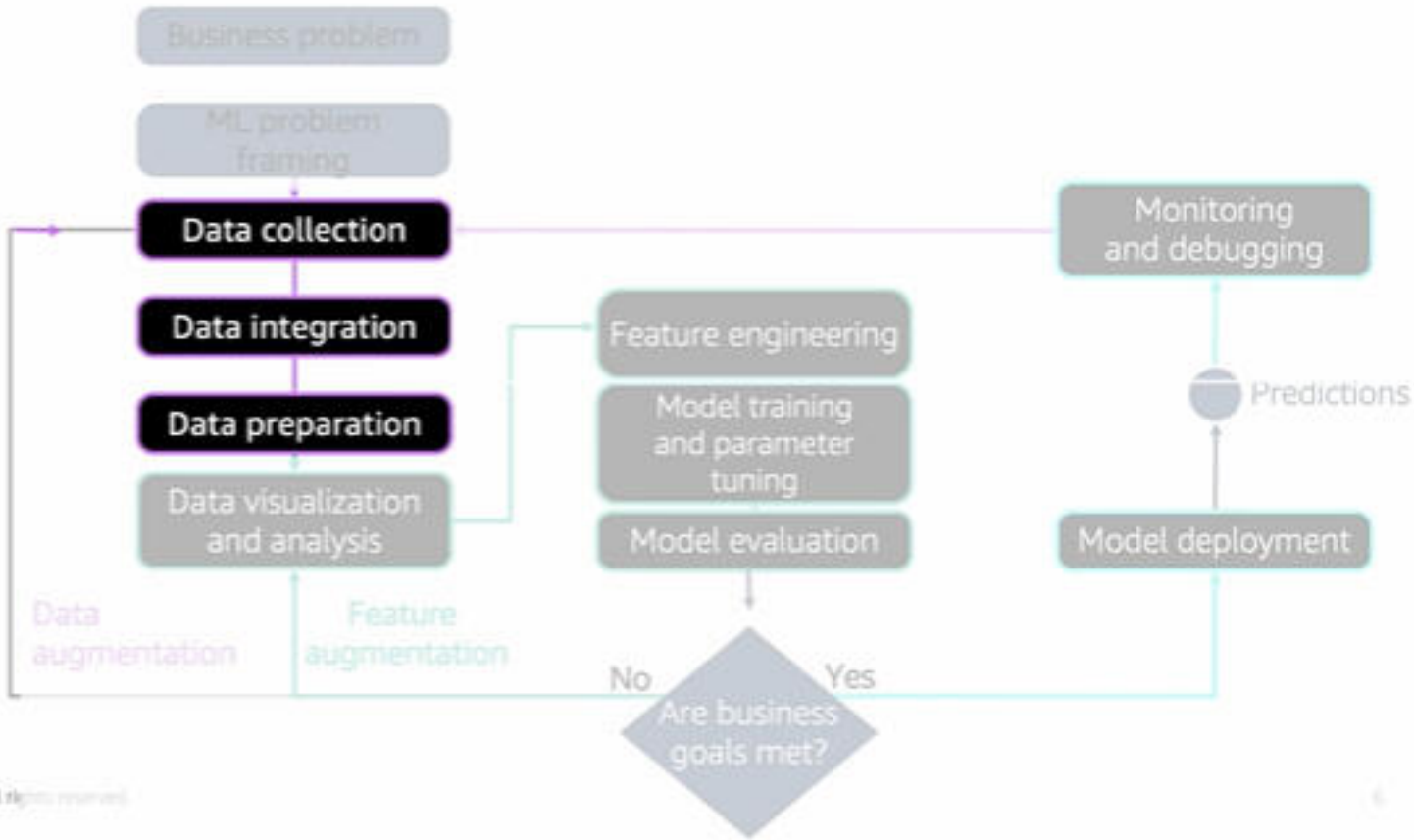
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML process: Preparation

aws training and certification

Preparation steps:

- Data exploration and profiling
- Data formatting
- Data conversion
- Encoding
- Data cleaning
- Normalization
- Resampling (oversampling/ undersampling)



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker Data Wrangler

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws training and certification

SageMaker Data Wrangler

The fastest and easiest way to prepare data for ML

aws training and certification

With a **single click**

- Ingest data from various data sources
- Deploy data preparation workflows into production

Leverage **built-in feature** to

- Transform data with data transformations
- Data visually using visualization templates

Ability to

- Bring your custom transformation
- Estimate ML model accuracy

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

SageMaker Data Wrangler

The fastest and easiest way to prepare data for ML

aws training and certification

With a **single click**

- Ingest data from various data sources
- Deploy data preparation workflows into production

Leverage **built-in feature** to

- Transform data with data transformations
- Data visually using visualization templates

Ability to

- Bring your custom transformation
- Estimate ML model accuracy

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

SageMaker Data Wrangler

The fastest and easiest way to prepare data for ML

aws training and certification

With a **single click**

- Ingest data from various data sources
- Deploy data preparation workflows into production

Leverage **built-in feature** to

- Transform data with data transformations
- Data visually using visualization templates

Ability to

- Bring your custom transformation
- Estimate ML model accuracy

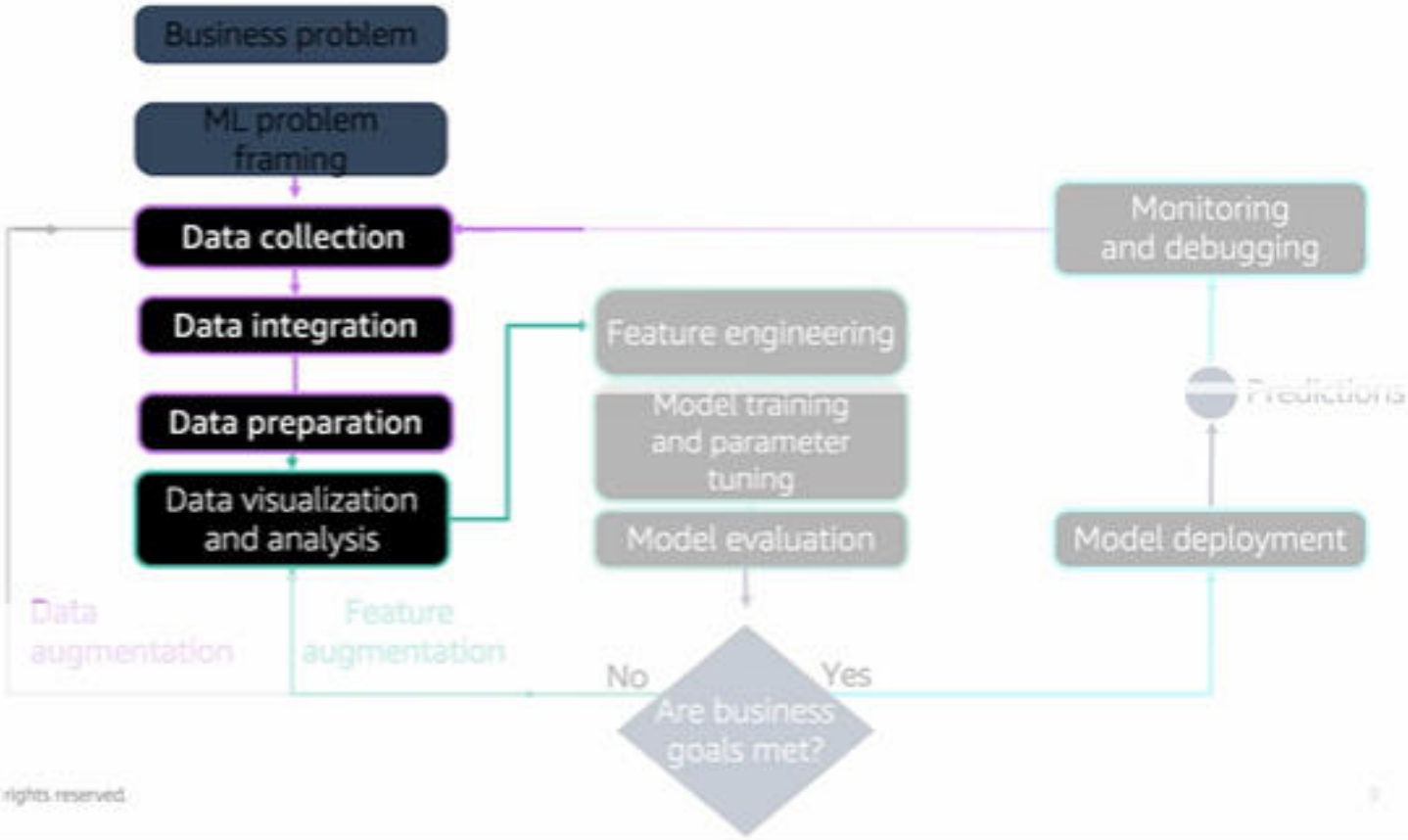
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Data collection, integration, preparation and visualization with Data Wrangler

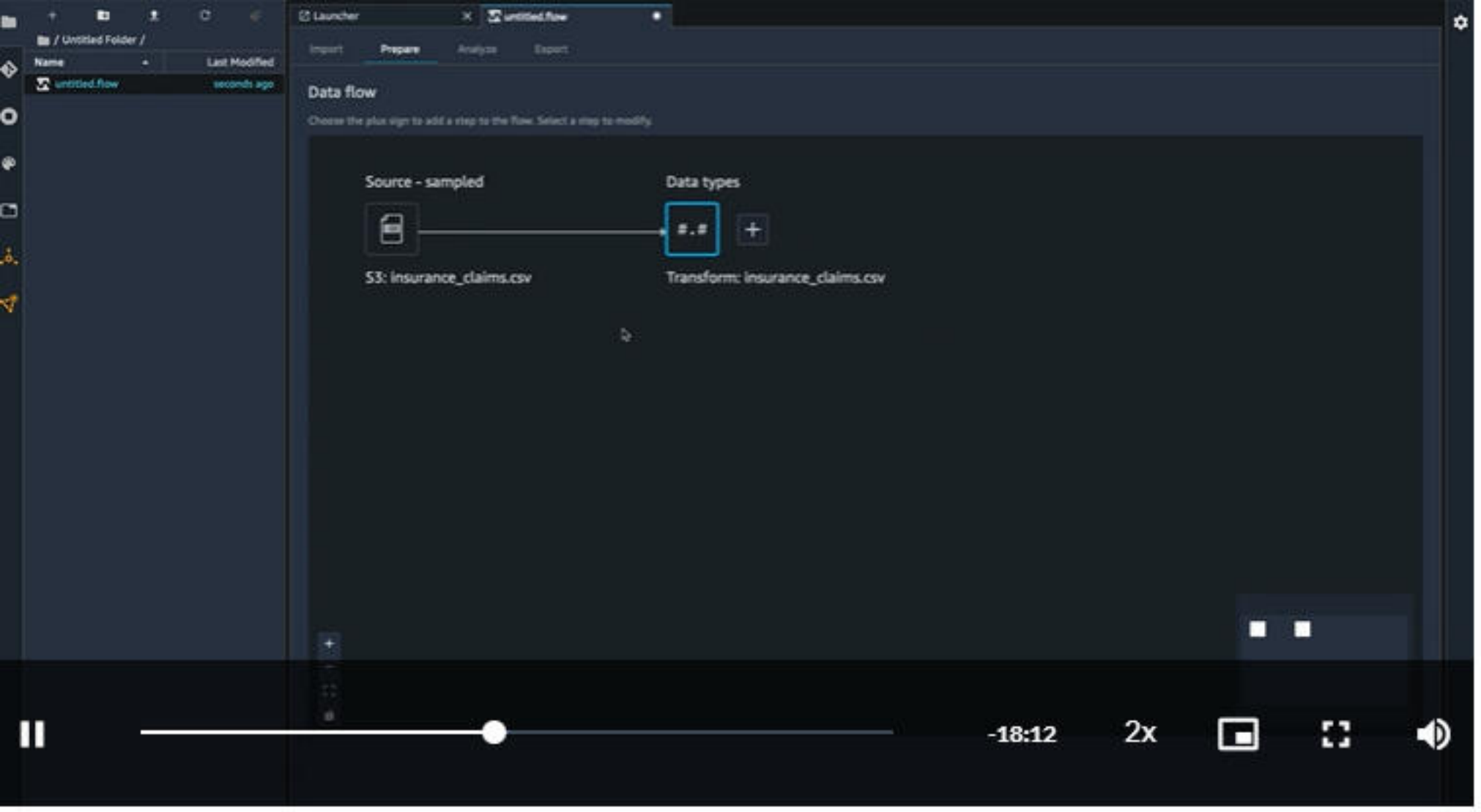
aws training and certification

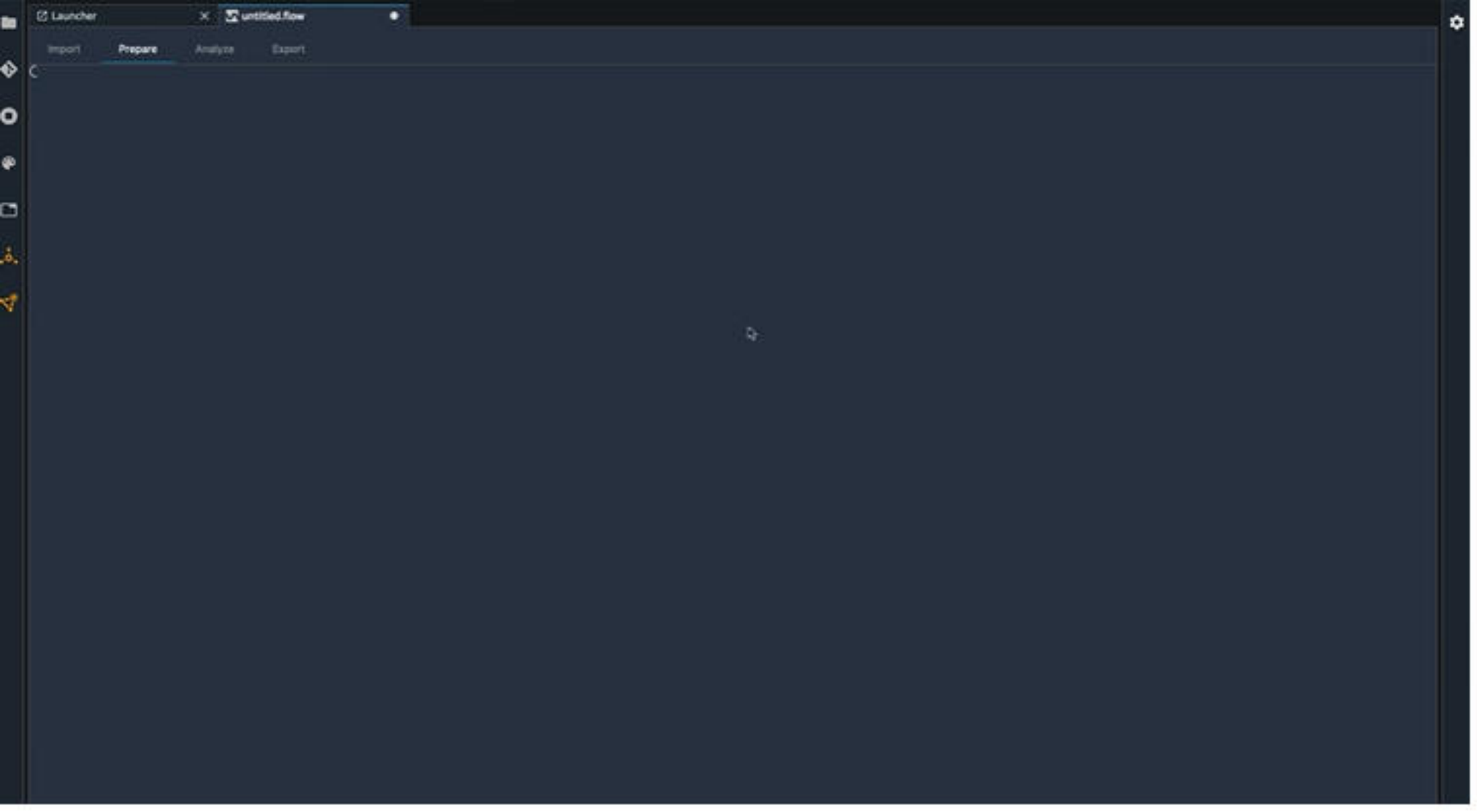
Data Wrangler's

- Data Selection tool to import data in various formats from multiple sources
- Leverage 300+ data transformations to transform imported data
- Leverage pre-configured visualization template to visualize your data



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.


/ Untitled Folder /
Name
Last Modified
untitled.flow
seconds ago
Launcher
untitled.flow
Import
Prepare
Analyze
Export
Data flow
Choose the plus sign to add a step to the flow. Select a step to modify.
Source - sampled
Data types
S3: insurance_claims.csv
Transform: insurance_claims.csv
-18:12
2x


Launcher
untitled.flow
Import
Prepare
Analyze
Export

Launcher | untitled.flow

Import | Prepare | Analyze | Export

Data flow / Transform: insurance_claims.csv

Transform: insurance_claims.csv

months_as_custome...	age (long)	policy_number (long)	policy_bind_date (da...	policy_state (string)	policy_csl (string)	policy_deductable (l...	policy_
328	48	521585	2014-10-17	OH	250/500	1000	1406.9
228	42	342868	2006-06-27	IN	250/500	2000	1197.
134	29	687698	2000-09-06	OH	100/300	2000	1413.
256	41	227811	1990-05-25	IL	250/500	2000	1415.
228	44	367455	2014-06-06	IL	500/1000	1000	1585.
256	39	104594	2006-10-12	OH	250/500	1000	1351.
137	34	413978	2000-06-04	IN	250/500	1000	1333.
165	37	429027	1990-02-03	IL	100/300	1000	1137.
27	33	485665	1997-02-05	IL	100/300	500	1442.
212	42	636550	2011-07-25	IL	100/300	500	1315.
235	42	543610	2002-05-26	OH	100/300	500	1253.
447	61	214618	1999-05-29	OH	100/300	2000	1157.
60	23	842643	1997-11-20	OH	500/1000	500	1215.
121	34	626808	2012-10-26	OH	100/300	1000	956.6
180	38	644081	1998-12-28	OH	250/500	2000	1301.
473	58	892874	1992-10-19	IN	100/300	2000	1131.
70	26	558938	2005-06-08	OH	500/1000	1000	1199.
140	31	275265	2004-11-15	IN	500/1000	500	708.8
160	37	921202	2014-12-28	OH	500/1000	500	1374.
196	39	143972	1992-08-02	IN	500/1000	2000	1475.
460	62	185430	2002-06-25	IN	250/500	1000	1187.
217	41	431876	2005-11-27	IL	500/1000	2000	875.1
370	55	285496	1994-05-27	IL	100/300	2000	972.1
413	55	115399	1991-02-08	IN	100/300	2000	1268.
237	40	736882	1996-02-02	IN	100/300	1000	883.3
8	35	699044	2015-12-05	OH	100/300	2000	1266.
257	43	863236	1990-09-20	IN	100/300	2000	1322.
202	34	608515	2002-07-18	IN	100/300	500	848.0
224	40	914088	1990-02-08	OH	100/300	2000	1291.

Back to data flow

TRANSFORM

Add | Previous steps

- Custom Transform
- Custom formula
- Encode categorical
- Featurize date/time
- Featurize text
- Format string
- Handle missing
- Handle outliers
- Manage columns
- Manage rows
- Manage vectors
- Parse column as type
- Process numeric
- Search and edit
- Validate string

Launcher | untitled.flow

Import | Prepare | Analyze | Export

Data flow / Transform: insurance_claims.csv

Transform: insurance_claims.csv

months_as_custome...	age (long)	policy_number (long)	policy_bind_date (da...	policy_state (string)	policy_csl (string)	policy_deductable (l...	policy_
328	48	521585	2014-10-17	OH	250/500	1000	1406.9
228	42	342868	2006-06-27	IN	250/500	2000	1197.
134	29	687698	2000-09-06	OH	100/300	2000	1413.
256	41	227811	1990-05-25	IL	250/500	2000	1415.
228	44	367455	2014-06-06	IL	500/1000	1000	1585.
256	39	104594	2006-10-12	OH	250/500	1000	1351.
137	34	413978	2000-06-04	IN	250/500	1000	1333.
165	37	429027	1990-02-03	IL	100/300	1000	1137.
27	33	485665	1997-02-05	IL	100/300	500	1442.
212	42	636550	2011-07-25	IL	100/300	500	1315.
235	42	543610	2002-05-26	OH	100/300	500	1253.
447	61	214618	1999-05-29	OH	100/300	2000	1157.
60	23	842643	1997-11-20	OH	500/1000	500	1215.
121	34	626808	2012-10-26	OH	100/300	1000	956.6
180	38	644081	1998-12-28	OH	250/500	2000	1301.
473	58	892874	1992-10-19	IN	100/300	2000	1131.
70	26	558938	2005-06-08	OH	500/1000	1000	1199.
140	31	275265	2004-11-15	IN	500/1000	500	708.6
160	37	921202	2014-12-28	OH	500/1000	500	1374.
196	39	143972	1992-08-02	IN	500/1000	2000	1475.
460	62	185430	2002-06-25	IN	250/500	1000	1187.
217	41	431876	2005-11-27	IL	500/1000	2000	875.1
370	55	285496	1994-05-27	IL	100/300	2000	972.1
413	55	115399	1991-02-08	IN	100/300	2000	1268.
237	40	736882	1996-02-02	IN	100/300	1000	883.3
8	35	699044	2015-12-05	OH	100/300	2000	1266.
257	43	863236	1990-09-20	IN	100/300	2000	1322.
202	34	608515	2002-07-18	IN	100/300	500	848.0
224	40	914088	1990-02-08	OH	100/300	2000	1291.

Back to data flow

TRANSFORM

Add | Previous steps

- Custom Transform
- Custom formula
- Encode categorical
- Featurize date/time
- Featurize text
- Format string
- Handle missing
- Handle outliers
- Manage columns
- Manage rows
- Manage vectors
- Parse column as type
- Process numeric
- Search and edit
- Validate string

Launcher | untitled.flow

Import | Prepare | Analyze | Export

Imported datasets / Transform: insurance_claims.csv /

Create Analysis

Create an analysis of your data. Learn more

Histogram:

No Preview available

Use Configure for built-in analyses

Use Code to create a custom analysis

Data table

fraud_reported	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	policy_deductable	policy_an
Y	328	48	521585	2014-10-17	OH	250/500	1000	1406.91
Y	228	42	342868	2006-06-27	IN	250/500	2000	1197.22
N	134	29	687698	2000-09-06	OH	100/300	2000	1413.14
Y	256	41	227811	1990-05-25	IL	250/500	2000	1415.74
N	228	44	367455	2014-06-06	IL	500/1000	1000	1583.91
Y	256	39	104594	2006-10-12	OH	250/500	1000	1351.1
N	137	34	413978	2000-06-04	IN	250/500	1000	1333.35

Back to all analyses

Configure | Code

Chart

Histogram

Bias Report

Histogram

Quick Model

Scatter Plot

Table Summary

Target Leakage

Color by

Select...

Optional

Facet by

Select...

Optional

Cancel | Preview | Create

Amazon SageMaker Feature Store

aws training and certification

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Challenges in Scaling ML Productivity

aws training and certification

- Long and tedious feature engineering
- Redundant feature pipelines
- No sharing or discovery mechanism
- Troubleshooting overhead due to training inference skew

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

SageMaker Feature Store benefits

aws training and certification

- Ingest features from many sources including Data Wrangler
- Search and discover features easily
- Ensure feature consistency for both training and for inferences
- Standardize with a single source of feature definition
- Integrate with SageMaker Pipeline

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Launcher | untitled.flow | untitled1.ipynb | untitled.ipynb

Markdown | 2 vCPU + 4 GiB | Python 3 (Data Science) | Kernel: CPU: 19.90% MEM: 5.60% | Share

```
    },
    {
        "name": "auto_make",
        "type": "string"
    },
    {
        "name": "auto_model",
        "type": "string"
    },
    {
        "name": "auto_year",
        "type": "long"
    },
    {
        "name": "fraud_reported",
        "type": "string"
    }
]
```

Select Record identifier and Event time feature name. These are required parameters for feature group creation. * **Record identifier name** is the name of the feature whose value uniquely identifies a Record defined in the feature group's feature definitions. * **Event time feature name** is the name of the EventTime of a Record in FeatureGroup. A EventTime is point in time when a new event occurs that corresponds to the creation or update of a Record in FeatureGroup. All Records in the FeatureGroup must have a corresponding EventTime.

[14]:
```
record_identifier_name = "property_claim"
if record_identifier_name is None:
    raise RuntimeError("Select a column name as the feature group identifier.")

event_time_feature_name = "incident_date"
if event_time_feature_name is None:
    raise RuntimeError("Select a column name as the event time feature name.")

# Below you map the schema detected from Data Wrangler to Feature Group types.
feature_definitions = [
    {
        "FeatureName": schema['name'],
        "FeatureType": datawrangler_FG_type_mapping.get(
            schema['type'],
            default_FG_type
        )
    } for schema in column_schema
]
print(feature_definitions)
```

```
[{'FeatureName': 'months_as_customer', 'FeatureType': 'Integral'}, {'FeatureName': 'age', 'FeatureType': 'Integral'}, {'FeatureName': 'policy_number', 'FeatureType': 'Integral'}, {'Featur
eName': 'policy_bind_date', 'FeatureType': 'String'}, {'FeatureName': 'policy_state', 'FeatureType': 'String'}, {'FeatureName': 'policy_csl', 'FeatureType': 'String'}, {'FeatureName': 'po
licy_deductable', 'FeatureType': 'Integral'}, {'FeatureName': 'policy_annual_premium', 'FeatureType': 'Fractional'}, {'FeatureName': 'umbrella_limit', 'FeatureType': 'Integral'}, {'Featur
eName': 'insured_zip', 'FeatureType': 'Integral'}, {'FeatureName': 'insured_sex', 'FeatureType': 'String'}, {'FeatureName': 'insured_education_level', 'FeatureType': 'String'}, {'FeatureN
ame': 'insured_occupation', 'FeatureType': 'String'}, {'FeatureName': 'insured_hobbies', 'FeatureType': 'String'}, {'FeatureName': 'insured_relationship', 'FeatureType': 'String'}, {'Feat
```

Amazon SageMaker Ground Truth

aws training and certification

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker Ground Truth Overview

aws training and certification

Quickly label training data

Easily integrate human labelers

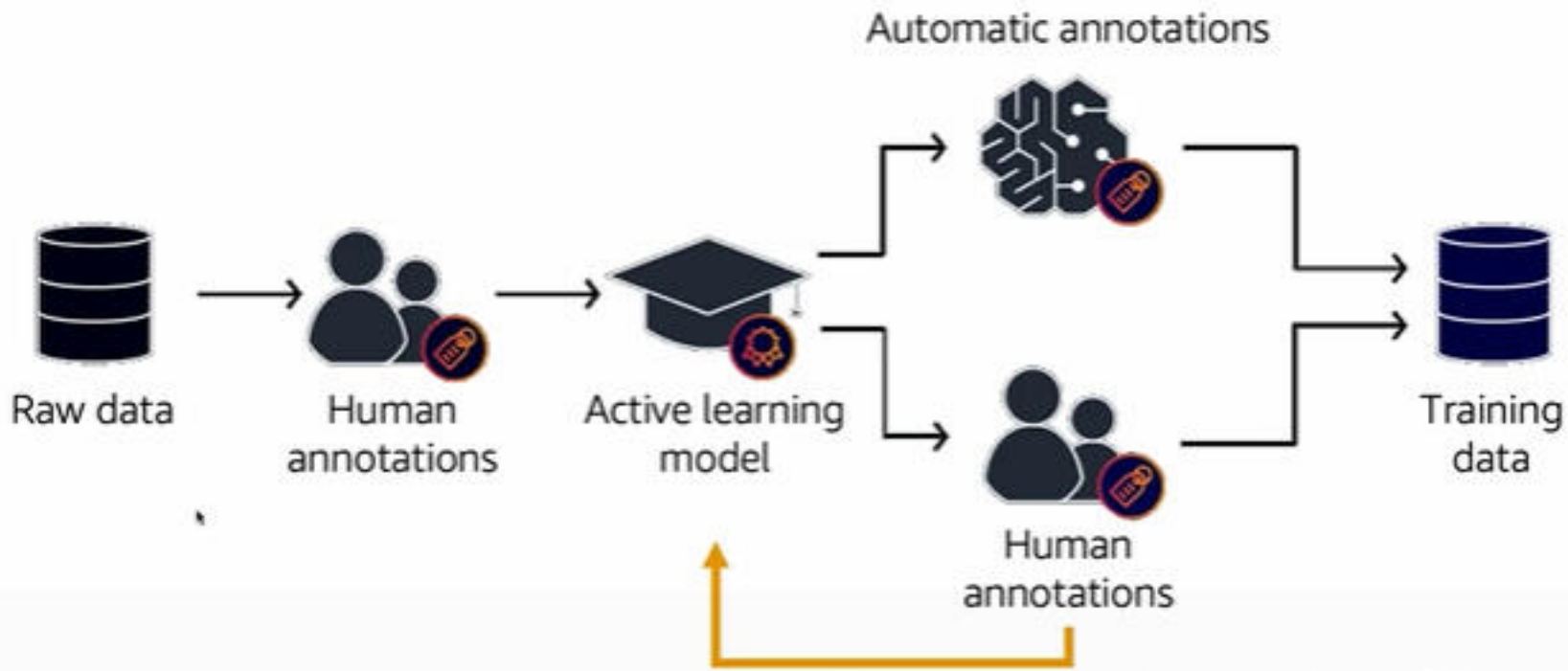
Get accurate results

KEY FEATURES

- Automatic labeling by way of machine learning
- Ready-made and custom workflows for image bounding box, segmentation, and text
- Private and public human workforce
- Label management

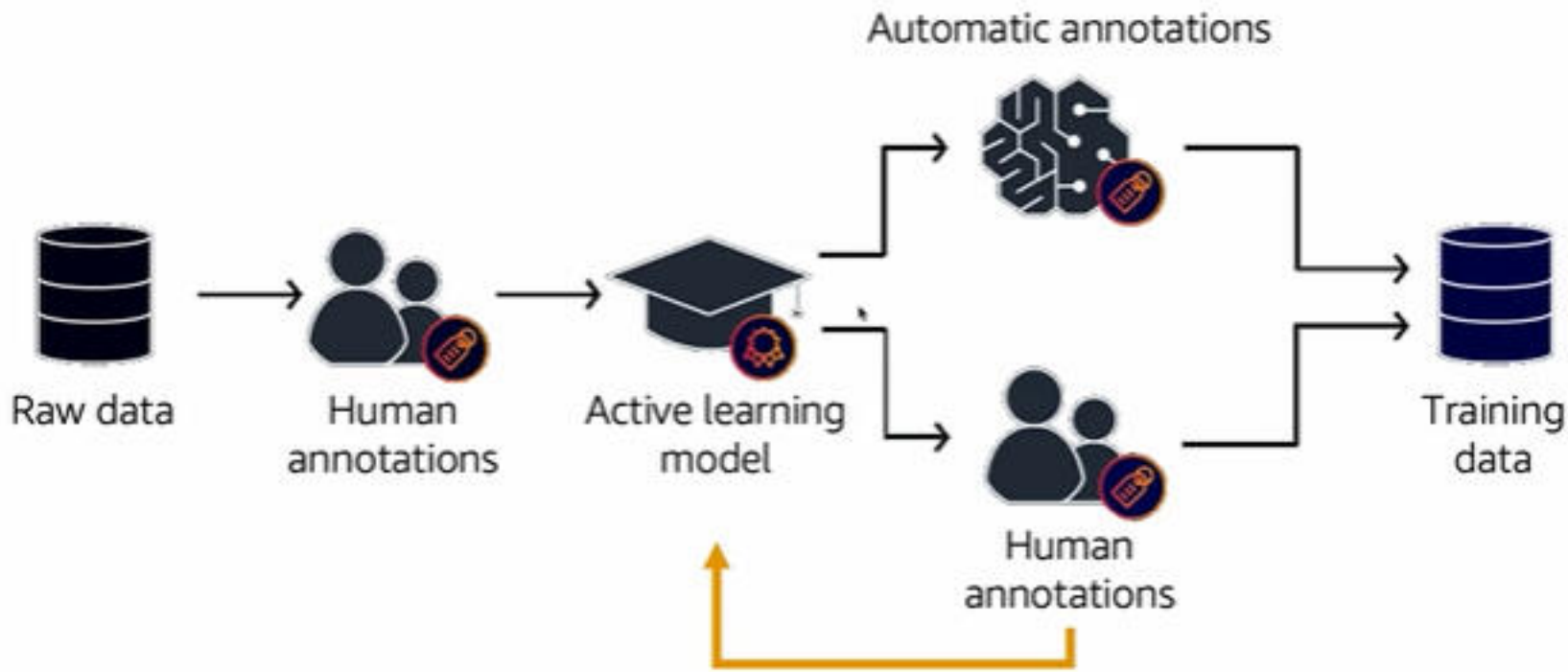
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

SageMaker Ground Truth: How it works



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

SageMaker Ground Truth: How it works



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker

Amazon SageMaker Studio

- Dashboard
- Search
- Images

▼ Ground Truth
- Labeling jobs
- Labeling datasets
- Labeling workforces

▼ Notebook
- Notebook instances
- Lifecycle configurations
- Git repositories

▼ Processing
- Processing jobs

▼ Training
- Algorithms
- Training jobs
- Hyperparameter tuning jobs

▼ Inference
- Compilation jobs
- Model packages
- Models
- Endpoint configurations
- Endpoints
- Batch transform jobs

▼ Edge Inference
- Get started
- Edge packaging jobs
- Edge device fleets
- Edge devices

MACHINE LEARNING

Amazon SageMaker

Build, train, and deploy machine learning models at scale

The quickest and easiest way to get ML models from idea to production.

Get started

Explore SageMaker Studio, a machine learning Integrated Development Environment (IDE) for building, training, and debugging models, tracking experiments, deploying models, and monitoring their performance. This is available in the following AWS Regions: US East (Ohio), US East (N. Virginia), US West (Oregon), and Europe (Ireland).

SageMaker Studio

Pricing (US)

With Amazon SageMaker, you pay only for what you use. Authoring, training and hosting is billed by the second, with no minimum fees and no upfront commitments.

Learn more

Related services

- AWS Glue
- Amazon EC2
- Amazon Elastic Block Store (EBS)

More resources

- Developer resources
- AWS Developer Forum

How it works

Label

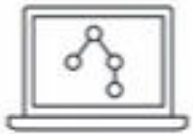
Set up and manage labeling jobs for highly accurate training datasets within Amazon SageMaker, using active learning and human labeling

Build

Connect to other AWS services and transform data in Amazon SageMaker notebooks

Train

Use Amazon SageMaker's algorithms and frameworks, or bring your own, for distributed training



Amazon SageMaker

Amazon SageMaker Studio

- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
  - Get started
  - Edge packaging jobs
  - Edge device fleets
  - Edge devices

Amazon SageMaker > Labeling jobs

Labeling jobs

Info

Actions

Create labeling job

Search labeling jobs

Name	Status	Task type	Labeled objects/total	Creation time

There are currently no labeling jobs created.

Create labeling job

Amazon SageMaker > Labeling jobs > Create labeling job

Step 1
Specify job details

Step 2
Select workers and configure tool

Specify job details

Job overview

Job name

Name-of-labeling-job

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

I want to specify a label attribute name different from the labeling job name.

Label attribute name is the key where your labels are stored in the augmented manifest. Ground Truth uses the labeling job name as the default label attribute name.

Input data setup Info

Use the automated setup to have Ground Truth automatically identify your dataset in S3. Use the manual setup if you have an input manifest file.

Automated data setup
Provide the S3 location of the dataset you want labeled and let Ground Truth automatically connect to and use the dataset for your job.

Manual data setup
Provide the S3 location of a file (an input manifest file) that identifies the data objects you want labeled.

Data setup

S3 location for input datasets Info

This is the location in S3 where your dataset objects are stored. Ground Truth will use all data objects in this location for your labeling job.

s3://input-dataset-bucket/optional-prefix

Browse S3

S3 location for output datasets Info

This is the location in S3 where your labeling job output data is stored.

Same location as input dataset

Specify a new location

Data type

Choose an option

IAM Role Info

Provide the ID or ARN for your own AWS KMS encryption key for Amazon SageMaker to access your S3 bucket. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

Choose an option

Use this button to process and complete your input data setup.

Complete data setup

▸ Additional configuration - optional

Choose an option

Use this button to process and complete your input data setup.

Complete data setup

▸ Additional configuration - optional

Dataset object selection, encryption

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.

Image

Task selection

Select the task that a human worker will perform to label objects in your dataset.

Image Classification (Single Label)

Get workers to categorize images into individual classes. Info

Image Classification (Multi-label)

Get workers to categorize images into one or more classes. Info

Bounding box

Get workers to draw bounding boxes around specified objects in your images. Info

Semantic segmentation

Get workers to draw pixel level labels around specific objects and segments in your images. Info

Choose an option

Use this button to process and complete your input data setup.

Complete data setup

▸ Additional configuration - optional

Dataset object selection, encryption

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.

Image

- Image
- Text
- Video
  - Video - All
  - Video - Classification
  - Video - Object detection
  - Video - Object tracking
- Point cloud
- Custom

Bounding box

Get workers to draw bounding boxes around specified objects in your images. Info

Semantic segmentation

Get workers to draw pixel level labels around specific objects and segments in your images. Info

Use this button to process and complete your input data setup.

Complete data setup

▶ Additional configuration - optional

Dataset object selections, encryption

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.

Image

Task selection

Select the task that a human worker will perform to label objects in your dataset.

Image Classification (Single Label)

Get workers to categorize images into individual classes. Info

Image Classification (Multi-label)

Get workers to categorize images into one or more classes. Info

Bounding box

Get workers to draw bounding boxes around specified objects in your images. Info

Semantic segmentation

Get workers to draw pixel-level labels around specific objects and segments in your images. Info

Task selection

Select the task that a human worker will perform to label objects in your dataset.

Image Classification (Single Label)

Get workers to categorize images into individual classes. Info

Image Classification (Multi-label)

Get workers to categorize images into one or more classes. Info

Bounding box

Get workers to draw bounding boxes around specified objects in your images. Info

Semantic segmentation

Get workers to draw pixel level labels around specific objects and segments in your images. Info

Label verification

Get workers to verify existing labels in your dataset. Info

▸ Additional configuration - *optional*

Dataset object selection, encryption

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.

Text

Task selection

Select the task that a human worker will perform to label objects in your dataset.

Text Classification (Single Label)

Get workers to categorize text into individual classes. Info

Positive

Negative

'The movie tells a lovely and wise story with honesty and has been acted out with unassuming grace.'

Text Classification (Multi-label)

Get workers to categorize text into one or more classes. Info

Positive

Inspiring

Jargon

'Every day is a fresh start. Always start your day with a cup of positivites.'

Named entity recognition

Get workers to apply labels to words or phrases within a larger text. Info

▸ Tags - *optional*

▸ Additional configuration - *optional*

Dataset object selection, encryption

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.

Text

Task selection

Select the task that a human worker will perform to label objects in your dataset.

Text Classification (Single Label)

Get workers to categorize text into individual classes. Info

Positive

Negative

'The movie tells a lovely and wise story with honesty and has been acted out with unassuming grace.'

Text Classification (Multi-label)

Get workers to categorize text into one or more classes. Info

Positive

Inspiring

Jargon

'Every day is a fresh start. Always start your day with a cup of positivites.'

Named entity recognition

Get workers to apply labels to words or phrases within a larger text. Info

▸ Tags - *optional*

▸ Additional configuration - *optional*

Dataset object selection, encryption

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.

Point cloud

Task selection

Select the task that a human worker will perform to label objects in your dataset.

Point cloud object detection

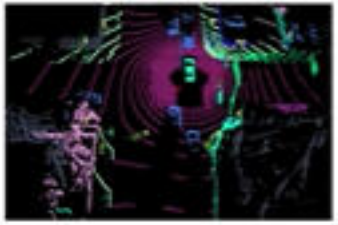
Have workers identify objects in LiDAR point cloud frames by drawing cuboids around objects. Info

Point cloud object tracking

Have workers track the movement of objects in LiDAR point cloud frames over time. Info

Point cloud semantic segmentation

Have workers identify objects in a LiDAR point cloud frame by painting all pixels associated with a label. Info



▸ Tags - *optional*

▸ Additional configuration - *optional*

Dataset object selection, encryption

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.

Point cloud

Task selection

Select the task that a human worker will perform to label objects in your dataset.

Point cloud object detection

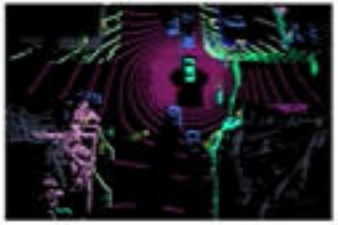
Have workers identify objects in LiDAR point cloud frames by drawing cuboids around objects. Info

Point cloud object tracking

Have workers track the movement of objects in LiDAR point cloud frames over time. Info

Point cloud semantic segmentation

Have workers identify objects in a LiDAR point cloud frame by painting all pixels associated with a label. Info



▸ Tags - *optional*

How you can help customers

- Understand the data structure and its purpose
- Understand where data comes from and how it is used today
- Devise the appropriate data processing solution

Clean, prepare, and visualize data first, before using it for ML!

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Workbook 4.1

aws training and certification

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	policy_deductable	policy_annual_premium	umbrella_limit	insured_zip	police_report_available	total_claim_amount	injury_claim	property_claim	vehicle_claim	auto_make	auto_model	auto_year	fraud_reported
0	328	48	521585	2014-10-17	OH	250/500	1000	1406.91	0	466132	YES	71610	6510	13020	52080	Saab	92x	2004	Y
1	228	42	342868	2006-06-27	IN	250/500	2000	1197.22	5000000	468176	?	5070	780	780	3510	Mercedes	E400	2007	Y
2	134	29	687698	2000-09-06	OH	100/300	2000	1413.14	5000000	430632	NO	34650	7700	3850	23100	Dodge	RAM	2007	N
3	256	41	227811	1990-05-25	IL	250/500	2000	1415.74	6000000	608117	NO	63400	6340	6340	50720	Chevrolet	Tahoe	2014	Y
4	228	44	367455	2014-06-06	IL	500/1000	1000	1583.91	6000000	610706	NO	6500	1300	650	4550	Acura	RSX	2009	N

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws training and certification

Module 5: Deep Learning Amazon Machine Images

- Identify what deep learning AMIs are
- Identify the capabilities of AWS Deep Learning AMIs
- Identify use cases for AWS Deep Learning AMIs
- Explain when, how, and why to use AWS Deep Learning AMIs

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon ML stack

aws training and certification

AI services

Vision | Speech | Text | Search | Chatbots | Personalize | Forecast | Fraud | Development | Contact Centers

ML services

Amazon SageMaker | Ground Truth | AWS Marketplace for ML | SageMaker Studio IDE | Neo | Augmented AI

ML frameworks and infrastructure

TensorFlow | PyTorch | Gluon | Keras | DeepGraphLibrary | Deep Learning AMIs and Containers | GPUs and CPUs | Elastic Inference | Inferencia | FPGA

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Deep Learning AMIs: Two choices

aws training and certification

- Pre-configured environments to quickly build deep learning applications for ML practitioners and researchers
- Quickly launch Amazon EC2 instances pre-installed with popular deep learning frameworks and interfaces

Conda AMI

- Pre-installed pip package for deep learning framework
- Available in Ubuntu, Amazon Linux, and Windows 2016 versions

Base AMI

- Clean slate to set up custom private deep learning engine repository with no frameworks installed; only NVIDIA CUDA and other dependencies
- Available in Ubuntu and Amazon Linux versions

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Supported deep learning frameworks

aws training and certification

TensorFlow

PyTorch

mxnet

Chainer

theano

K

The Microsoft Cognitive Toolkit

GLUON

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML Infrastructure: AWS Deep Learning AMIs

aws training and certification

Provides access to the Amazon ML infrastructure

ML infrastructure

TensorFlow
MXNet
PyTorch
GLUON
Chainer
Keras

P3 instance

C5 instance

EC2 Inf1

Amazon Elastic Inference

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML Infrastructure: AWS Deep Learning AMIs

aws training and certification

Provides access to the Amazon ML infrastructure

ML infrastructure

TensorFlow
MXNet
PyTorch
GLUON
Chainer
Keras

P3 instance

C5 instance

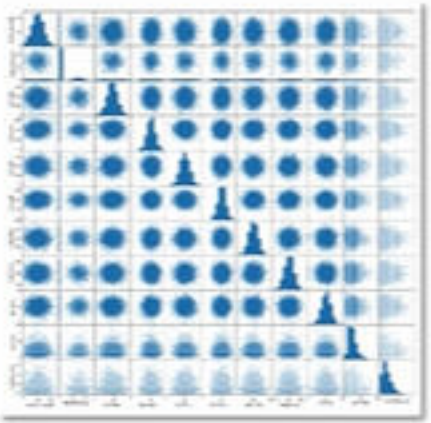
EC2 Inf1

Amazon Elastic Inference

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Use cases: DLAMI

Learning and app development



Machine learning and data analytics

Research

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Cost optimization

- AMIs are free. Customers pay only for AWS infrastructure.
- Benefits are gained from using powerful compute instances.
- P3 instances speed up training and reduce costs.
- Anaconda Data Science Platform simplifies package management and deployment (everything is ready for you to get started).
- Customers can containerize their code to realize economies of scale.

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

DLAMI customers

- Machine learning experts
- Build custom algorithms
- Need more flexibility
- Building internal data science teams

-0:49 2x

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Module 6: Amazon Sagemaker Concepts

aws training and certification

- Review a typical machine learning workflow using the AWS Deep Learning AMIs
- Identify what Amazon SageMaker is and explain how it works
- Identify the key components and features of Amazon SageMaker
- Discuss Amazon SageMaker Studio
- Identify use cases for Amazon SageMaker
- Identify the advantages of using Amazon SageMaker
- Summary of latest SageMaker services

aws training and certification

Module 6: Amazon Sagemaker Concepts

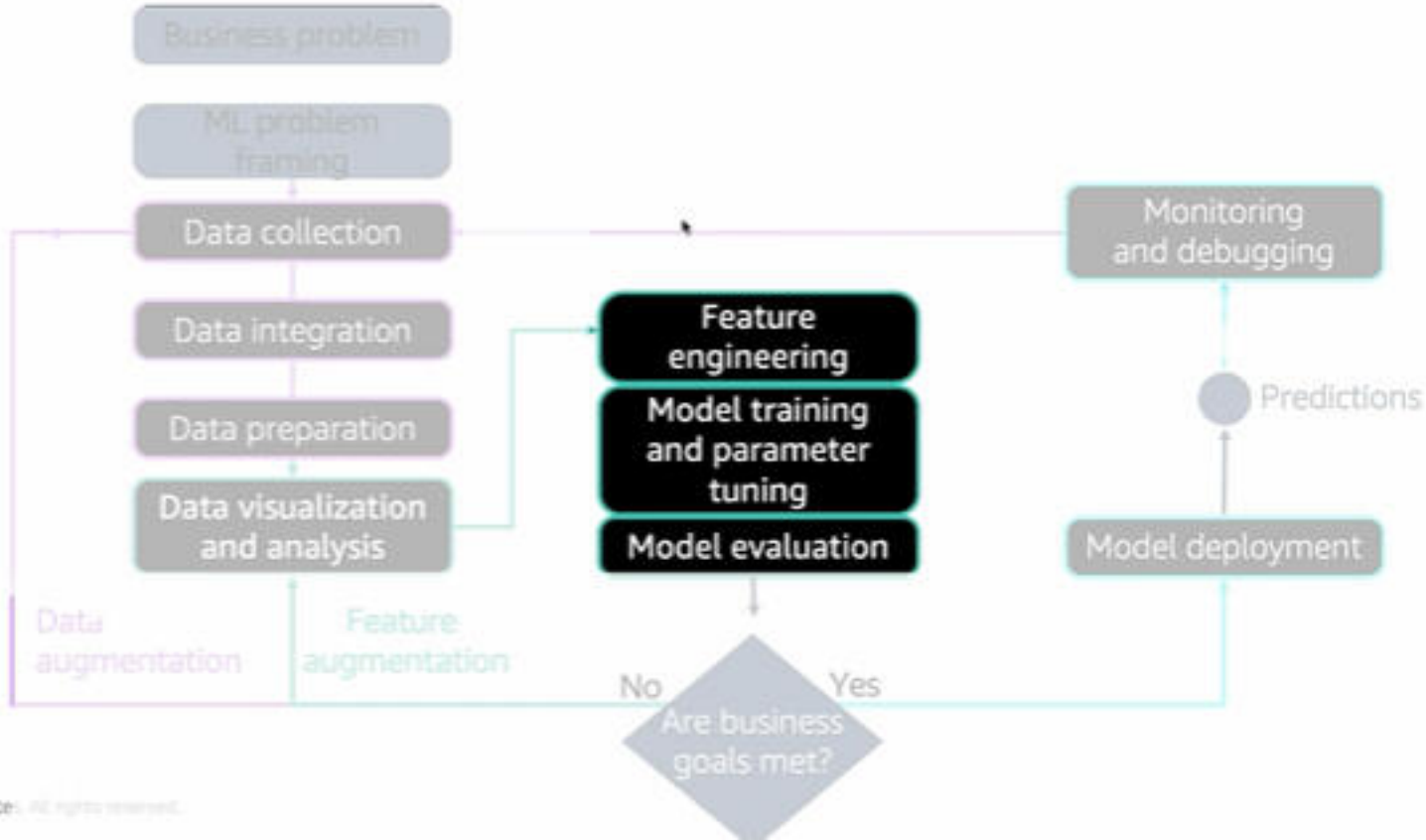
- Review a typical machine learning workflow using the AWS Deep Learning AMIs
- Identify what Amazon SageMaker is and explain how it works
- Identify the key components and features of Amazon SageMaker
- Discuss Amazon SageMaker Studio
- Identify use cases for Amazon SageMaker
- Identify the advantages of using Amazon SageMaker
- Summary of latest SageMaker services

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML process: Model training

aws training and certification

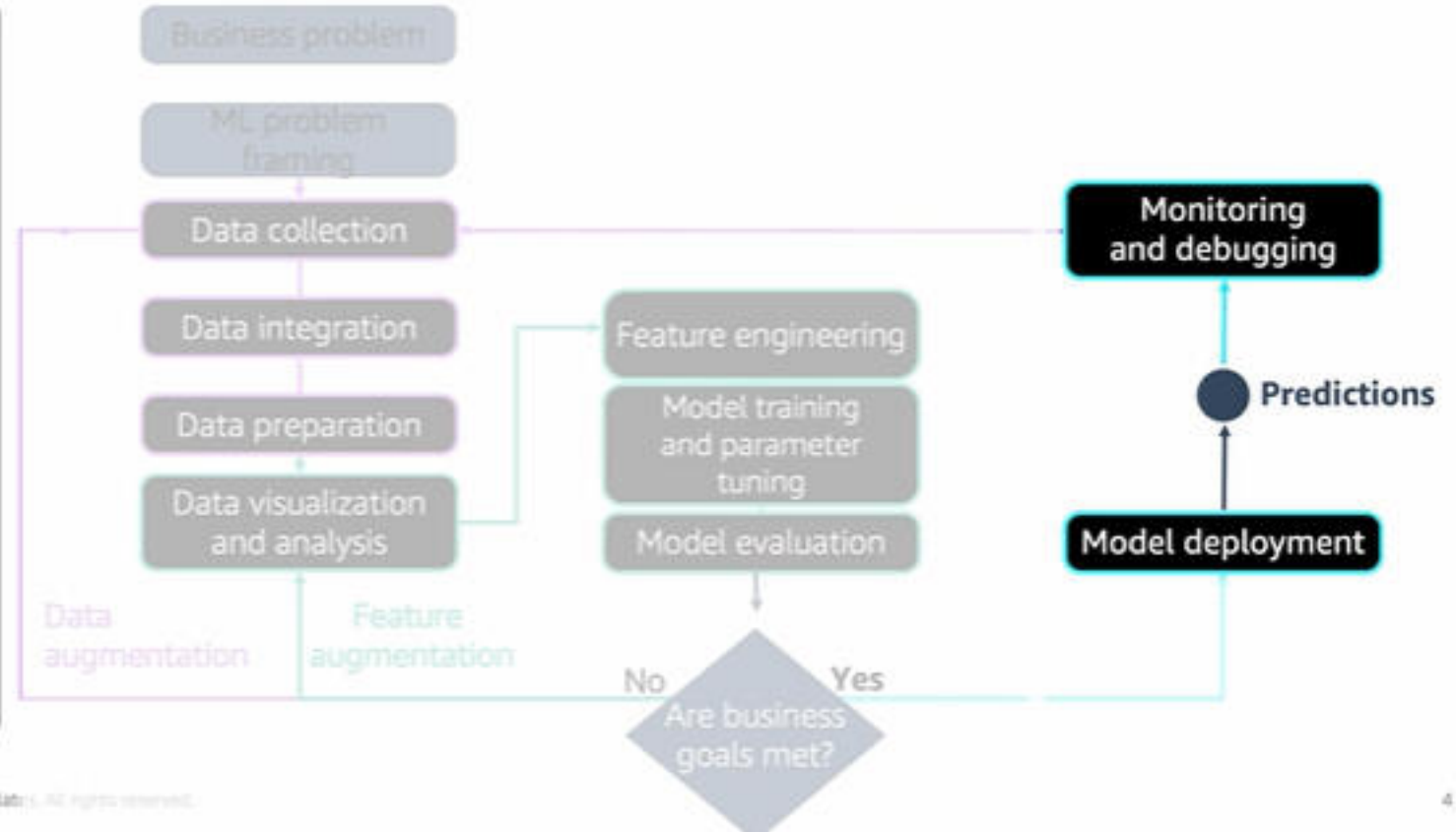
- Set up and manage
  - Notebook environments
  - Training clusters
- Write data connectors
- Scale ML algorithms to large datasets
- Distribute ML training algorithm to multiple machines
- Secure model artifacts



© 2020 Amazon Web Services, Inc. or its Affiliate. All rights reserved.

ML process: DevOps

- Set up and manage inference clusters
- Manage and scale model inference APIs
- Monitor and debug model predictions
- Model versioning and performance tracking
- Automate new model version promotion to production (A/B testing)



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Challenges building and deploying an ML model

aws training and certification

- ML development can be complex, cumbersome, time consuming, error prone, expensive, and a very iterative process.
- Different personas often use different tools in different phases of ML development, which can make collaboration and sharing work a challenge, affecting productivity.
- ML development lacked a robust integrated development environment (IDE) for the entire ML workflow until Amazon introduced Amazon SageMaker.

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker Overview

aws training and certification

Prepare	Build	Train & tune	Deploy & manage
SageMaker Ground Truth	SageMaker Studio Notebook	One-click Training	One-click Deployment
SageMaker Data Wrangler	Built-in and Bring-your-own Algorithms	SageMaker Experiments	Kubernetes & Kubeflow integration
SageMaker Processing	Local Mode	SageMaker Automatic Model Tuning	Multi-Model Endpoints
SageMaker Feature Store	SageMaker Autopilot	SageMaker Distributed Training	SageMaker Model Monitor
SageMaker Clarify	SageMaker JumpStart	SageMaker Debugger	SageMaker Edge Manager
		Managed Spot Training	SageMaker Pipeline

Amazon SageMaker Overview

aws training and certification

Prepare	Build	Train & tune	Deploy & manage
SageMaker Ground Truth	SageMaker Studio Notebook	One-click Training	One-click Deployment
SageMaker Data Wrangler	Built-in and Bring-your-own Algorithms	SageMaker Experiments	Kubernetes & Kubeflow integration
SageMaker Processing	Local Mode	SageMaker Automatic Model Tuning	Multi-Model Endpoints
SageMaker Feature Store	SageMaker Autopilot	SageMaker Distributed Training	SageMaker Model Monitor
SageMaker Clarify	SageMaker JumpStart	SageMaker Debugger	SageMaker Edge Manager
		Managed Spot Training	SageMaker Pipeline

Amazon SageMaker Overview

aws training and certification

Prepare	Build	Train & tune	Deploy & manage
SageMaker Ground Truth	SageMaker Studio Notebook	One-click Training	One-click Deployment
SageMaker Data Wrangler	Built-in and Bring-your-own Algorithms	SageMaker Experiments	Kubernetes & Kubeflow integration
SageMaker Processing	Local Mode	SageMaker Automatic Model Tuning	Multi-Model Endpoints
SageMaker Feature Store	SageMaker Autopilot	SageMaker Distributed Training	SageMaker Model Monitor
SageMaker Clarify	SageMaker JumpStart	SageMaker Debugger	SageMaker Edge Manager
		Managed Spot Training	SageMaker Pipeline

Amazon SageMaker Overview

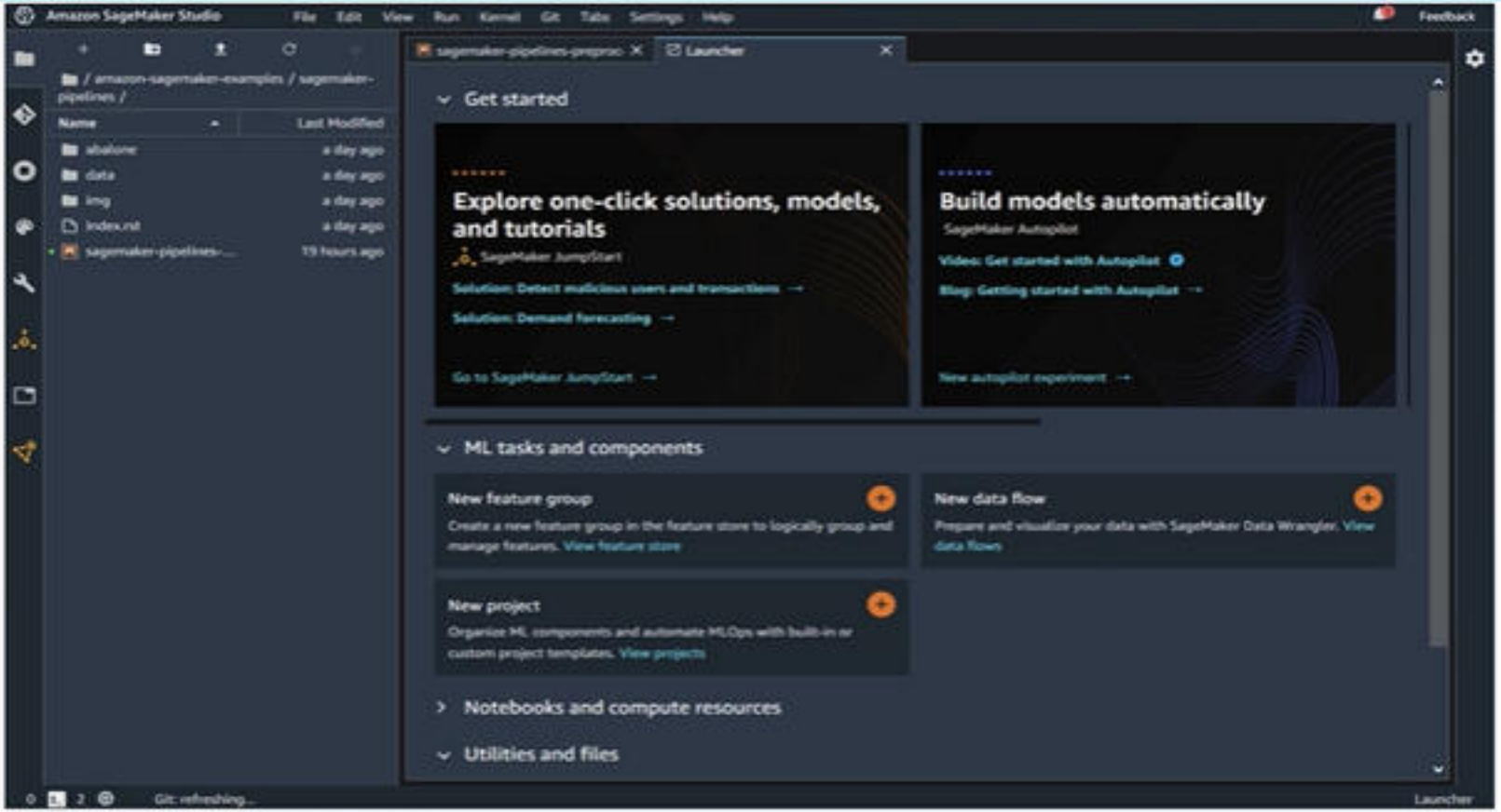
aws training and certification

Prepare	Build	Train & tune	Deploy & manage
SageMaker Ground Truth	SageMaker Studio Notebook	One-click Training	One-click Deployment
SageMaker Data Wrangler	Built-in and Bring-your-own Algorithms	SageMaker Experiments	Kubernetes & Kubeflow integration
SageMaker Processing	Local Mode	SageMaker Automatic Model Tuning	Multi-Model Endpoints
SageMaker Feature Store	SageMaker Autopilot	SageMaker Distributed Training	SageMaker Model Monitor
SageMaker Clarify	SageMaker JumpStart	SageMaker Debugger	SageMaker Edge Manager
		Managed Spot Training	SageMaker Pipeline

Amazon SageMaker Studio IDE

Fully integrated development environment (IDE) for Machine Learning

aws training and certification



Amazon SageMaker

Amazon SageMaker Studio

- Dashboard
- Search
- Images
- Ground Truth
- Notebook
- Processing
- Training
- Inference
- Edge Inference
- Augmented AI
- AWS Marketplace

Amazon SageMaker > Dashboard

Overview

Hide

Ground Truth

Set up and manage labeling jobs for highly accurate training datasets using active learning and human labeling.

Labeling jobs

Notebook

Availability of AWS and SageMaker SDKs and sample notebooks to create training Jobs and deploy models.

Notebook instances

Training

Train and tune models at any scale. Leverage high performance AWS algorithms or bring your own.

Training jobs

Hyperparameter tuning jobs

Inference

Create models from training jobs or import external models for hosting to run inferences on new data.

Models

Endpoints

Batch transform jobs

Processing Run

Pre- or post-processing and model evaluation workloads with a fully managed experience.

Processing jobs

Recent activity

Recent activity within the Last 7 days

Ground Truth	Notebook	Training	Inference	Processing
Labeling jobs	**Notebook instances**	**Training jobs**	**Models**	**Processing jobs**
No recent activity.	2 Created	8 Completed	2 Created	1 Completed
		8 Created	**Endpoints**	1 Created
		Hyperparameter tuning jobs	1 In Service	
		No recent activity.	1 Created	

Amazon SageMaker

Amazon SageMaker Studio

- Dashboard
- Search
- Images
- Ground Truth
- Notebook
- Processing
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
- Edge Inference
- Augmented AI
- AWS Marketplace

Amazon SageMaker > Hyperparameter tuning jobs

Hyperparameter tuning jobs

Add/Edit tags | Create hyperparameter tuning job

Search hyperparameter tuning jobs

1

Name	Status	Training completed/total	Creation time	Duration
tuning-job-1-1a853e2106da4cbcbe	Completed	18 / 20 2 Failed	Jun 25, 2020 08:01 UTC	8 minutes
tuning-job-1-88bc9b41655f64b5e8f	Completed	17 / 20 3 Failed	Jun 24, 2020 23:18 UTC	8 minutes
tuning-job-1-a27828026e9e418390	Completed	250 / 251 1 Failed	Jun 24, 2020 22:12 UTC	2 hours
xgboost-200513-1407	Completed	20 / 20	May 13, 2020 14:07 UTC	26 minutes
ufo-linear-learner-hptuning	Completed	50 / 50	Jan 23, 2020 17:06 UTC	39 minutes

Amazon SageMaker

Amazon SageMaker Studio

- Dashboard
- Search
- Images
- Ground Truth
- Notebook
- Processing
- Training
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
- Augmented AI
- AWS Marketplace

Amazon SageMaker > Models

Models

Create endpoint | Create endpoint configuration | Actions | Create model

Search models

1

Name	ARN	Creation time
insurance-fraud-predictor-2020-12-20-13-45-24-589	arn:aws:sagemaker:us-east-1:649088000912:model/insurance-fraud-predictor-2020-12-20-13-45-24-589	Dec 20, 2020 13:45 UTC
linear-learner-2020-12-20-11-42-46-301	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-12-20-11-42-46-301	Dec 20, 2020 11:42 UTC
demo-xgboost-customer-churn-2020-12-01-12-59-24-774	arn:aws:sagemaker:us-east-1:649088000912:model/demo-xgboost-customer-churn-2020-12-01-12-59-24-774	Dec 01, 2020 13:53 UTC
linear-learner-2020-11-02-10-11-07-597	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-11-02-10-11-07-597	Nov 02, 2020 10:16 UTC
xgboost-2020-10-06-10-06-01-773	arn:aws:sagemaker:us-east-1:649088000912:model/xgboost-2020-10-06-10-06-01-773	Oct 06, 2020 10:10 UTC
linear-learner-2020-09-15-10-02-48-531	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-09-15-10-02-48-531	Sep 15, 2020 10:08 UTC
linear-learner-2020-09-04-11-08-33-236	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-09-04-11-08-33-236	Sep 04, 2020 11:14 UTC
linear-learner-2020-09-02-10-51-42-896	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-09-02-10-51-42-896	Sep 02, 2020 10:57 UTC
linear-learner-2020-09-01-14-33-25-510	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-09-01-14-33-25-510	Sep 01, 2020 14:39 UTC
automl-ifp-model-25-07-33-08	arn:aws:sagemaker:us-east-1:649088000912:model/automl-ifp-model-25-07-33-08	Jun 25, 2020 08:10 UTC

Amazon SageMaker

Amazon SageMaker Studio

- Dashboard
- Search
- Images
- Ground Truth
- Notebook
- Processing
- Training
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
- Augmented AI
- AWS Marketplace

Amazon SageMaker > Models

Models

Create endpoint | Create endpoint configuration | Actions | Create model

Search models

1

Name	ARN	Creation time
insurance-fraud-predictor-2020-12-20-13-45-24-589	arn:aws:sagemaker:us-east-1:649088000912:model/insurance-fraud-predictor-2020-12-20-13-45-24-589	Dec 20, 2020 13:45 UTC
linear-learner-2020-12-20-11-42-46-301	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-12-20-11-42-46-301	Dec 20, 2020 11:42 UTC
demo-xgboost-customer-churn-2020-12-01-12-59-24-774	arn:aws:sagemaker:us-east-1:649088000912:model/demo-xgboost-customer-churn-2020-12-01-12-59-24-774	Dec 01, 2020 13:53 UTC
linear-learner-2020-11-02-10-11-07-597	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-11-02-10-11-07-597	Nov 02, 2020 10:16 UTC
xgboost-2020-10-06-10-06-01-773	arn:aws:sagemaker:us-east-1:649088000912:model/xgboost-2020-10-06-10-06-01-773	Oct 06, 2020 10:10 UTC
linear-learner-2020-09-15-10-02-48-531	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-09-15-10-02-48-531	Sep 15, 2020 10:08 UTC
linear-learner-2020-09-04-11-08-33-236	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-09-04-11-08-33-236	Sep 04, 2020 11:14 UTC
linear-learner-2020-09-02-10-51-42-896	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-09-02-10-51-42-896	Sep 02, 2020 10:57 UTC
linear-learner-2020-09-01-14-33-25-510	arn:aws:sagemaker:us-east-1:649088000912:model/linear-learner-2020-09-01-14-33-25-510	Sep 01, 2020 14:39 UTC
automl-ifp-model-25-07-33-08	arn:aws:sagemaker:us-east-1:649088000912:model/automl-ifp-model-25-07-33-08	Jun 25, 2020 08:10 UTC

Amazon SageMaker

Amazon SageMaker Studio

- Dashboard
- Search
- Images
- Ground Truth
- Notebook
- Processing
- Training
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
- Augmented AI
- AWS Marketplace

Amazon SageMaker > Endpoint configuration

Endpoint configuration

Apply to endpoint | Clone | Actions | Create endpoint configuration

Search endpoint configuration

1 ...

Name	ARN	Creation time
insurance-fraud-predictor-2020-12-20-13-45-25-032	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/insurance-fraud-predictor-2020-12-20-13-45-25-032	Dec 20, 2020 13:45 UTC
linear-learner-2020-12-20-11-42-46-301	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-12-20-11-42-46-301	Dec 20, 2020 11:42 UTC
demo-xgboost-customer-churn-2020-12-01-13-33-51	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/demo-xgboost-customer-churn-2020-12-01-13-33-51	Dec 01, 2020 13:33 UTC
linear-learner-2020-11-02-10-11-07-597	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-11-02-10-11-07-597	Nov 02, 2020 10:16 UTC
xgboost-2020-10-06-10-06-01-773	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/xgboost-2020-10-06-10-06-01-773	Oct 06, 2020 10:10 UTC
linear-learner-2020-09-15-10-02-48-531	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-09-15-10-02-48-531	Sep 15, 2020 10:08 UTC
linear-learner-2020-09-04-11-08-33-236	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-09-04-11-08-33-236	Sep 04, 2020 11:14 UTC
linear-learner-2020-09-02-10-51-42-896	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-09-02-10-51-42-896	Sep 02, 2020 10:57 UTC
linear-learner-2020-09-01-14-33-25-510	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-09-01-14-33-25-510	Sep 01, 2020 14:39 UTC
xgboost-2020-06-24-20-20-33-681	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/xgboost-2020-06-24-20-20-33-681	Jun 24, 2020 20:24 UTC

Amazon SageMaker

Amazon SageMaker Studio

- Dashboard
- Search
- Images
- Ground Truth
- Notebook
- Processing
- Training
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
- Augmented AI
- AWS Marketplace

Amazon SageMaker > Endpoint configuration

Endpoint configuration

Apply to endpoint | Clone | Actions | Create endpoint configuration

Search endpoint configuration

1 ...

Name	ARN	Creation time
insurance-fraud-predictor-2020-12-20-13-45-25-032	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/insurance-fraud-predictor-2020-12-20-13-45-25-032	Dec 20, 2020 13:45 UTC
linear-learner-2020-12-20-11-42-46-301	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-12-20-11-42-46-301	Dec 20, 2020 11:42 UTC
demo-xgboost-customer-churn-2020-12-01-13-33-51	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/demo-xgboost-customer-churn-2020-12-01-13-33-51	Dec 01, 2020 13:33 UTC
linear-learner-2020-11-02-10-11-07-597	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-11-02-10-11-07-597	Nov 02, 2020 10:16 UTC
xgboost-2020-10-06-10-06-01-773	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/xgboost-2020-10-06-10-06-01-773	Oct 06, 2020 10:10 UTC
linear-learner-2020-09-15-10-02-48-531	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-09-15-10-02-48-531	Sep 15, 2020 10:08 UTC
linear-learner-2020-09-04-11-08-33-236	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-09-04-11-08-33-236	Sep 04, 2020 11:14 UTC
linear-learner-2020-09-02-10-51-42-896	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-09-02-10-51-42-896	Sep 02, 2020 10:57 UTC
linear-learner-2020-09-01-14-33-25-510	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/linear-learner-2020-09-01-14-33-25-510	Sep 01, 2020 14:39 UTC
xgboost-2020-06-24-20-20-33-681	arn:aws:sagemaker:us-east-1:649088000912:endpoint-config/xgboost-2020-06-24-20-20-33-681	Jun 24, 2020 20:24 UTC

Amazon SageMaker

Amazon SageMaker Studio

- Dashboard
- Search
- Images
- Ground Truth
- Notebook
- Processing
- Training
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
- Augmented AI
- AWS Marketplace

Amazon SageMaker > Endpoints

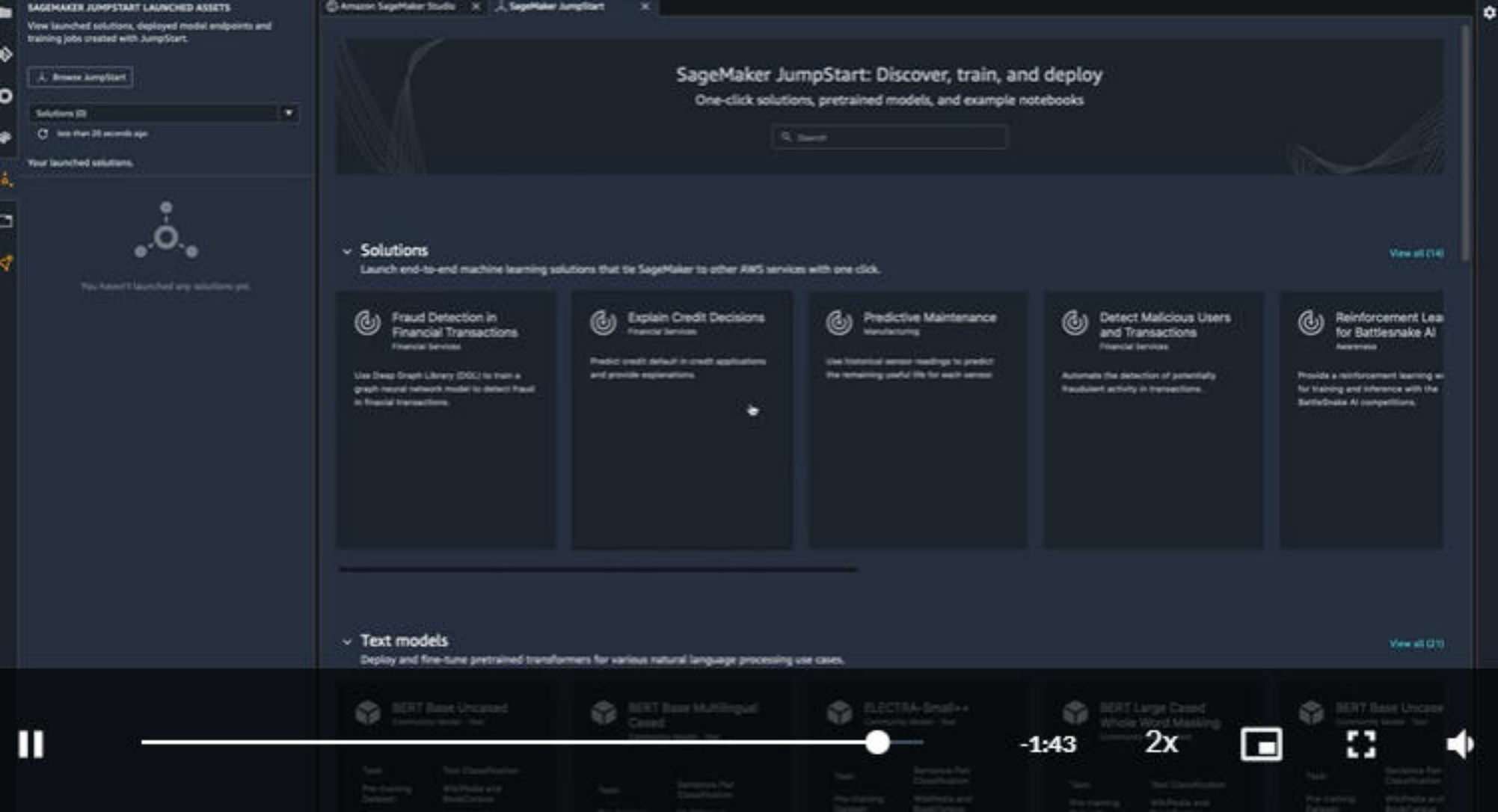
Endpoints

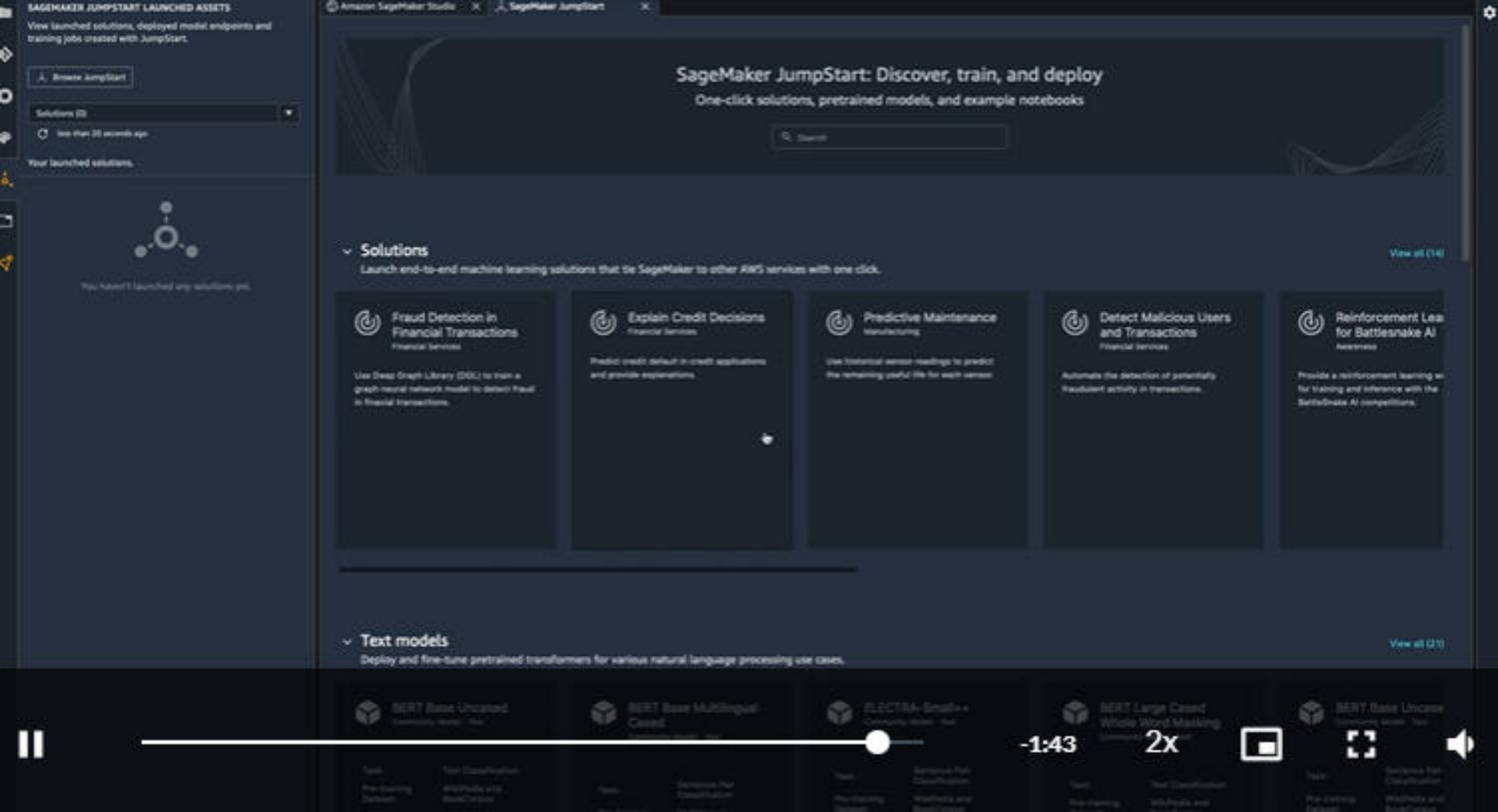
Update endpoint | Actions | Create endpoint

Search endpoints

1

Name	ARN	Creation time	Status	Last updated
insurance-fraud-predictor-2020-12-20-13-45-25-103	arn:aws:sagemaker:us-east-1:649088000912:endpoint/insurance-fraud-predictor-2020-12-20-13-45-25-103	Dec 20, 2020 13:45 UTC	InService	Dec 20, 2020 13:53 UTC


SAGEMAKER JUMPSTART LAUNCHED ASSETS
View launched solutions, deployed model endpoints and training jobs created with JumpStart.
Browse JumpStart
Your launched solutions.
SageMaker JumpStart: Discover, train, and deploy
One-click solutions, pretrained models, and example notebooks
Solutions
Launch end-to-end machine learning solutions that tie SageMaker to other AWS services with one click.
Fraud Detection in Financial Transactions
Explain Credit Decisions
Predictive Maintenance
Detect Malicious Users and Transactions
Text models
Deploy and fine-tune pretrained transformers for various natural language processing use cases.
-1:43
2x


SAGEMAKER JUMPSTART LAUNCHED ASSETS
View launched solutions, deployed model endpoints and training jobs created with JumpStart.
Browse JumpStart
Solutions (0)
less than 20 seconds ago
Your launched solutions.
You haven't launched any solutions yet.
Amazon SageMaker Studio
SageMaker JumpStart
SageMaker JumpStart: Discover, train, and deploy
One-click solutions, pretrained models, and example notebooks
Solutions
Launch end-to-end machine learning solutions that tie SageMaker to other AWS services with one click.
View all (14)
Fraud Detection in Financial Transactions
Financial Services
Use Deep Graph Library (DGL) to train a graph neural network model to detect fraud in financial transactions.
Explain Credit Decisions
Financial Services
Predict credit default in credit applications and provide explanations.
Predictive Maintenance
Manufacturing
Use historical sensor readings to predict the remaining useful life for each sensor.
Detect Malicious Users and Transactions
Financial Services
Automate the detection of potentially fraudulent activity in transactions.
Reinforcement Lea for Battlesnake AI
Gameness
Provide a reinforcement learning w for training and inference with the BattleSnake AI competitions.
Text models
Deploy and fine-tune pretrained transformers for various natural language processing use cases.
View all (21)
BERT Base Uncased
BERT Base Multilingual Cased
ELECTRA-Small++
BERT Large Cased Whole Word Masking
BERT Base Uncase
-1:43
2x

Module 7: Amazon SageMaker Notebooks

aws training and certification

- Introduction to Jupyter Notebook and how Amazon SageMaker uses it
- Demo SageMaker notebooks - Create a notebook instance and explain how this is accomplished using Amazon SageMaker

Amazon SageMaker notebooks

aws training and certification

Single Sign-On (SSO) - Access your notebooks in seconds

Access controls - Administrators manage access and permissions

Fully managed - Use your notebooks without manually spinning up or configuring compute resources

Collaboration - Share notebooks with a single click

Scalability - Scale up or down compute resources as needed

Amazon SageMaker notebooks

aws training and certification

Single Sign-On (SSO) - Access your notebooks in seconds

Access controls - Administrators manage access and permissions

Fully managed - Use your notebooks without manually spinning up or configuring compute resources

Collaboration - Share notebooks with a single click

Scalability - Scale up or down compute resources as needed

Notebook instances

- Recommendations/personalization
- Fraud detection
- Forecasting
- Image classification
- Churn prediction
- Marketing email/campaign targeting
- Log processing and anomaly detection
- Speech to text
- More…

Amazon SageMaker notebook demo – Outcomes

aws training and certification

- Identify Amazon Elastic Compute Cloud (Amazon EC2) instance types suitable for ML
- Explain the purpose of an AWS Identity and Access Management (IAM) role with Amazon SageMaker instances
- Explain what Amazon Elastic Inference is and how it applies to the ML process in Amazon SageMaker
- Explain the purpose of lifecycle configuration in Amazon SageMaker
- Explain how to configure and use a Git repository with an Amazon SageMaker instance

Amazon SageMaker notebook demo – Outcomes

aws training and certification

- Identify where to find Amazon SageMaker Examples in the Amazon SageMaker console
- Identify which ML frameworks are available to Amazon SageMaker instances
- Explain how to access an EC2 instance through the terminal in the Amazon SageMaker console or using SSH
- Explain how to start and shut down instances using the console

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name

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

Notebook instance type

ml.t2.medium

Elastic Inference Learn more

none

▸ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook

  Lifecycle configurations always have root access

Encryption key - *optional*

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

▸ **Network** - *optional*

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name

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

Notebook instance type

ml.t2.medium

Elastic Inference Learn more

none

▶ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook

  Lifecycle configurations always have root access

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name

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

Notebook instance type

ml.t2.medium

Elastic Inference Learn more

none

▶ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name

mltd-2

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

Notebook instance type

ml.t3.medium

Elastic Inference Learn more

none

▸ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the **AmazonSageMakerFullAccess** IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name

mltd-2

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

Notebook instance type

ml.t3.medium

Elastic Inference Learn more

none

▸ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the **AmazonSageMakerFullAccess** IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name

mltd-2

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

Notebook instance type

ml.t3.medium

Elastic Inference Learn more

none

▶ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Encryption key - *optional*

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name

mltd-2

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

Notebook instance type

ml.t3.medium

Elastic Inference Learn more

none

▶ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Encryption key - *optional*

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

Why GitHub? Team Enterprise Explore Marketplace Pricing Search Sign in Sign up

aws-samples / amazon-sagemaker-notebook-instance-lifecycle-config-samples

Watch 12 Star 164 Fork 98

Code Issues 10 Pull requests 5 Actions Projects Security Insights

master 2 branches 0 tags Go to file Code

jaipreet-s Merge pull request #54 from tuliocasagrande/master 27 days ago 101 commits

.github	Add new lifecycle entries to README.MD; Updated PULL_REQUEST...	16 months ago
scripts	Merge pull request #54 from tuliocasagrande/master	27 days ago
.DS_Store	Added export-to-pdf-fix	9 months ago
CODE_OF_CONDUCT.md	Creating initial file from template	2 years ago
CONTRIBUTING.md	Creating initial file from template	2 years ago
LICENSE	Creating initial file from template	2 years ago
README.md	Merge pull request #54 from tuliocasagrande/master	27 days ago

README.md

SageMaker Notebook Instance Lifecycle Config Samples

Overview

Lifecycle Configurations provide a mechanism to customize Notebook Instances via shell scripts that are executed during the lifecycle of a Notebook Instance.

About

A collection of sample scripts to customize Amazon SageMaker Notebook Instances using Lifecycle Configurations

Readme

View license

Releases

No releases published

Packages

No packages published

Contributors 20

-22:48 2x

aws-samples / amazon-sagemaker-notebook-instance-lifecycle-config-samples

Watch 12 | Star 164 | Fork 98

Code | Issues 10 | Pull requests 5 | Actions | Projects | Security | Insights

master | amazon-sagemaker-notebook-instance-lifecycle-config-samples / scripts / | Go to file

jaipreet-s Merge pull request #54 from tuliocasagrande/master — 60f3560 27 days ago | History

..

add-pypi-repository	Added a script to configure private pypi repositories	9 months ago
auto-stop-idle	Merge pull request #42 from AurelienMassiot/master	6 months ago
connect-emr-cluster	Add scripts for conda persistence, EFS, EMR, and conda packages	2 years ago
disable-uninstall-ssm-agent	Script to disable and uninstall SSM agent	14 months ago
execute-notebook-on-startup	Script to execute notebook file on startup using nbconvert	16 months ago
export-to-pdf-enable	Added a note in each script that internet access is required	9 months ago
install-conda-package-all-environments	Fix heredoc quoting for variable interpolation	9 months ago
install-conda-package-single-environment	Added single quotes around EOF to correct error	12 months ago
install-lab-extension	Fix heredoc quoting for variable interpolation	9 months ago
install-nb-extension	Fix heredoc quoting for variable interpolation	9 months ago
install-pip-package-all-environments	Fix heredoc quoting for variable interpolation	9 months ago
install-pip-package-single-environment	Fix heredoc quoting for variable interpolation	9 months ago
install-server-extension	Fix heredoc quoting for variable interpolation	9 months ago
mount-efs-file-system	Add scripts for conda persistence, EFS, EMR, and conda packages	2 years ago
mount-fsx-lustre-file-system	Updated documentation	9 months ago
notebook-history-s3	corrected on-start to pull correct file.	5 months ago

Code | Issues 10 | Pull requests 5 | Actions | Projects | Security | Insights

master · amazon-sagemaker-notebook-instance-lifecycle-config-samples / scripts / Go to file

jalpreet-s Merge pull request #54 from tuliocasagrande/master 6873568 27 days ago History

..

add-pypi-repository	Added a script to configure private pypi repositories	9 months ago
auto-stop-idle	Merge pull request #42 from AurelienMassiot/master	6 months ago
connect-emr-cluster	Add scripts for conda persistence, EFS, EMR, and conda packages	2 years ago
disable-uninstall-ssm-agent	Script to disable and uninstall SSM agent	14 months ago
execute-notebook-on-startup	Script to execute notebook file on startup using nbconvert	16 months ago
export-to-pdf-enable	Added a note in each script that internet access is required	9 months ago
install-conda-package-all-environments	Fix heredoc quoting for variable interpolation	9 months ago
install-conda-package-single-environment	Added single quotes around EOF to correct error	12 months ago
install-lab-extension	Fix heredoc quoting for variable interpolation	9 months ago
install-nb-extension	Fix heredoc quoting for variable interpolation	9 months ago
install-pip-package-all-environments	Fix heredoc quoting for variable interpolation	9 months ago
install-pip-package-single-environment	Fix heredoc quoting for variable interpolation	9 months ago
install-server-extension	Fix heredoc quoting for variable interpolation	9 months ago
mount-efs-file-system	Add scripts for conda persistence, EFS, EMR, and conda packages	2 years ago
mount-fsx-lustre-file-system	Updated documentation	9 months ago
notebook-history-s3	corrected on-start to pull correct file.	5 months ago
persistent-conda-ebs	updated script comments to remove pointer to docs and completed sente	5 months ago

Notebook instance settings

Notebook instance name

mltd-2

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

Notebook instance type

ml.t3.medium

Elastic Inference Learn more

none

▼ Additional configuration

Lifecycle configuration - *optional*

Customize your notebook environment with default scripts and plugins.

auto-stop-idle

Volume size in GB - *optional*

Enter the volume size of the notebook instance in GB. The volume size must be from 5 GB to 16384 GB (16 TB).

5

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook

  Lifecycle configurations always have root access

Encryption key - *optional*

Notebook instance settings

Notebook instance name

mltd-2

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

Notebook instance type

ml.t3.medium

Elastic Inference Learn more

none

▼ Additional configuration

Lifecycle configuration - *optional*

Customize your notebook environment with default scripts and plugins.

auto-stop-idle

Volume size in GB - *optional*

Enter the volume size of the notebook instance in GB. The volume size must be from 5 GB to 16384 GB (16 TB).

5

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook

  Lifecycle configurations always have root access

Encryption key - *optional*

Notebook instance settings

Notebook instance name

mltd-2

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

Notebook instance type

ml.t3.medium

Elastic Inference Learn more

none

▼ Additional configuration

Lifecycle configuration - *optional*

Customize your notebook environment with default scripts and plugins.

auto-stop-idle

Volume size in GB - *optional*

Enter the volume size of the notebook instance in GB. The volume size must be from 5 GB to 16384 GB (16 TB).

5

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook

  Lifecycle configurations always have root access

Encryption key - *optional*

Notebook instance settings

Notebook instance name

mltd-2

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

Notebook instance type

ml.t3.medium

Elastic Inference Learn more

none

▼ Additional configuration

Lifecycle configuration - *optional*

Customize your notebook environment with default scripts and plugins.

auto-stop-idle

Volume size in GB - *optional*

Enter the volume size of the notebook instance in GB. The volume size must be from 5 GB to 16384 GB (16 TB).

5

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook

  Lifecycle configurations always have root access

Encryption key - *optional*

Create an IAM role

Passing an IAM role gives Amazon SageMaker permission to perform actions in other AWS services on your behalf. Creating a role here will grant permissions described by the **AmazonSageMakerFullAccess** IAM policy to the role you create.

The IAM role you create will provide access to:

- S3 buckets you specify - *optional*
  - (•) Any S3 bucket
    Allow users that have access to your notebook instance access to any bucket and its contents in your account.
  - ( ) Specific S3 buckets
    Example: bucket-name-1, buck
    Comma delimited. ARNs, "*" and "/" are not supported.
  - ( ) None
- Any S3 bucket with "sagemaker" in the name
- Any S3 object with "sagemaker" in the name
- Any S3 object with the tag "sagemaker" and value "true" — See Object tagging
- S3 bucket with a Bucket Policy allowing access to SageMaker — See S3 bucket policies

Cancel | Create role

Elastic Inference Learn more

none

▸ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the **AmazonSageMakerFullAccess** IAM policy attached.

AmazonSageMaker-ExecutionRole-20201220T123075

Success! You created an IAM role.
AmazonSageMaker-ExecutionRole-20201220T123075

Root access - *optional*

- Enable - Give users root access to the notebook
- Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Encryption key - *optional*

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

▸ **Network** - *optional*

▸ **Git repositories** - *optional*

▸ **Tags** - *optional*

▸ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the **AmazonSageMakerFullAccess** IAM policy attached.

AmazonSageMaker-ExecutionRole-20201220T123075

Success! You created an IAM role.
AmazonSageMaker-ExecutionRole-20201220T123075

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Encryption key - *optional*

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

▸ **Network** - *optional*

▸ **Git repositories** - *optional*

▸ **Tags** - *optional*

Cancel | Create notebook instance

▸ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the **AmazonSageMakerFullAccess** IAM policy attached.

AmazonSageMaker-ExecutionRole-20201220T123075

Success! You created an IAM role.
AmazonSageMaker-ExecutionRole-20201220T123075

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Encryption key - *optional*

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

▸ **Network** - *optional*

▸ **Git repositories** - *optional*

▸ **Tags** - *optional*

Cancel | Create notebook instance

aws Services ▼ Search for services, features, marketplace products, and docs [Option+S] admin/corbuno-isengard @ corbuno ▼ N. Virginia ▼ Support ▼

Amazon SageMaker ×

Amazon SageMaker Studio

- Dashboard
- Search
- Images

▼ Ground Truth
- Labeling jobs
- Labeling datasets
- Labeling workforces

▼ Notebook
- Notebook instances
- Lifecycle configurations
- Git repositories

▼ Processing
- Processing jobs

▼ Training
- Algorithms
- Training jobs
- Hyperparameter tuning jobs

▼ Inference
- Compilation jobs
- Model packages
- Models
- Endpoint configurations

Amazon SageMaker > Git repositories > Add repository

Add repository

Repository settings

What type of repository do you want to add?

AWS CodeCommit
Highly available Git source control from AWS. Includes encryption, IAM integration, and more.

GitHub/Other Git-based repo
Connect GitHub or other Git-based repository. Store credentials with AWS Secrets Manager.

Choose an option

- (●) Use existing repository
- ( ) Create new repository

Repository

Search repositories ▼

Amazon SageMaker repository name

Provide a name for the repository to use in AWS CodeCommit and Amazon SageMaker

The repository name must be unique in your account and in the AWS Region and can have up to 63 characters. Valid characters: a-z, A-Z, 0-9, and - (hyphen)

Cancel | Add repository

▸ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the **AmazonSageMakerFullAccess** IAM policy attached.

AmazonSageMaker-ExecutionRole-20201220T123075 ▼

Success! You created an IAM role.
AmazonSageMaker-ExecutionRole-20201220T123075

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Encryption key - *optional*

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption ▼

▸ **Network** - *optional*

▸ **Git repositories** - *optional*

▸ **Tags** - *optional*

Cancel | Create notebook instance

mltd-1

Delete | Stop | Open Jupyter | Open JupyterLab

Notebook instance settings

Edit

Name mltd-1	Status InService	Notebook instance type ml.t3.medium
ARN arn:aws:sagemaker:us-east-1:649088000912:notebook-instance/mltd-1	Creation time Dec 20, 2020 10:42 UTC	Elastic Inference -
Lifecycle configuration arn:aws:sagemaker:us-east-1:649088000912:notebook-instance-lifecycle-config/auto-stop-idle View logs	Last updated Dec 20, 2020 10:44 UTC	Volume Size 5GB EBS

Git repositories

Name	Repository URL	Type
There are currently no resources.		

Permissions and encryption

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

Select items to perform actions on them.

Upload | New | 0 | / | Name | Last Modified | File size

The notebook list is empty.

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

Select items to perform actions on them.

Upload | New

0 /

Name | Last Modified | File size

The notebook list is empty.

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

23 Conda environments

Action	Name	Default?	Directory
	root		/home/ec2-user/anaconda3
	anaconda3		/home/ec2-user/anaconda3
	JupyterSystemEnv	✔	/home/ec2-user/anaconda3/envs/JupyterSystemEnv
	R		/home/ec2-user/anaconda3/envs/R
	amazonei_mxnet_p27		/home/ec2-user/anaconda3/envs/amazonei_mxnet_p27
	amazonei_mxnet_p36		/home/ec2-user/anaconda3/envs/amazonei_mxnet_p36

Available packages

Search...

Name	Version	Channel

255 installed packages in environment "JupyterSystemEnv"

Name	Version	Build	Available
_libgcc_mutex	0.1	conda_forge	
_openmp_mutex	4.5	1_gnu	
alsa-lib	1.2.3	h516909a_0	
atk	2.36.0	3	
atk-1.0	2.36.0	h63f31ab_3	
attrs	20.2.0	pyh9f0ad1d_0	

jupyter

Logout

```
sh-4.2$ whoami
ec2-user
sh-4.2$
```

jupyter

Logout

```
2362417    4 drwxrwxr-x 26 ec2-user ec2-user    4096 Oct 30 20:41 anaconda3
6553606    4 drwxr-xr-x  2 ec2-user root        4096 Dec 20 10:43 .aws
 274142    4 -rw-------  1 ec2-user ec2-user       5 Dec  7 21:53 .bash_history
 269781    4 -rw-r--r--  1 ec2-user ec2-user      18 Aug 30  2017 .bash_logout
 269783    4 -rw-r--r--  1 ec2-user ec2-user     193 Aug 30  2017 .bash_profile
 262167    4 -rw-rw-r--  1 ec2-user ec2-user     658 Oct 30 20:36 .bashrc
2107349    4 drwxr-xr-x  4 ec2-user ec2-user    4096 Nov  2 19:34 .cache
2362473    4 drwxrwxr-x  2 ec2-user ec2-user    4096 Oct 30 20:36 .conda
 262191    4 -rw-rw-r--  1 ec2-user ec2-user     148 Nov  2 19:23 .condarc
 665403    4 drwxrwxr-x  5 ec2-user ec2-user    4096 Nov  2 19:42 .config
 270194    0 -rw-r--r--  1 ec2-user ec2-user       0 Nov  2 19:42 .cw-agent-baked
 262165    4 -rw-rw-r--  1 ec2-user ec2-user    1201 Oct 30 20:36 .dlamirc
2102240    4 drwxrwxr-x  8 ec2-user ec2-user    4096 Oct 30 22:32 .dl_binaries
2102257    4 drwxrwxr-x 10 ec2-user ec2-user    4096 Oct 30 20:30 examples
 270165    4 -rw-r--r--  1 ec2-user ec2-user     481 Nov  2 19:42 .gitconfig
 400209    4 drwxr-xr-x  3 ec2-user ec2-user    4096 Dec 20 10:43 .jupyter
 666749    4 drwxrwxr-x  2 ec2-user ec2-user    4096 Oct 30 22:02 .keras
 262168   28 -rw-rw-r--  1 ec2-user ec2-user   25646 Aug 19 00:03 LICENSE
1617497    4 drwx------  3 ec2-user ec2-user    4096 Nov  2 19:28 .local
2107407    4 drwxr-xr-x  4 ec2-user ec2-user    4096 Nov  2 19:37 .npm
 269767 2776 -rw-rw-r--  1 ec2-user ec2-user 2839191 Oct 30 17:57 Nvidia_Cloud_EULA.pdf
 262185    4 -rw-rw-r--  1 ec2-user ec2-user      47 Oct 30 20:36 .profile
 262419    4 -rw-rw-r--  1 ec2-user ec2-user    3741 Oct 30 23:12 README
 274143    4 -rw-------  1 ec2-user ec2-user    1024 Dec 20 10:43 .rnd
      2    4 drwxr-xr-x  4 ec2-user ec2-user    4096 Dec 20 10:43 SageMaker
 274144    0 lrwxrwxrwx  1 root     root          42 Dec 20 10:43 sample-notebooks -> /home/ec2-user/sample-notebooks-160846
1014
6553601    4 drwxr-xr-x 37 root     root        4096 Dec 20 10:43 sample-notebooks-1608461014
 262238    0 lrwxrwxrwx  1 root     root          36 Dec 20 10:43 .sparkmagic -> /home/ec2-user/SageMaker/.sparkmagic
 395470    4 drwxrwxr-x  4 ec2-user ec2-user    4096 Oct 30 18:19 src
 394556    4 drwx------  2 ec2-user ec2-user    4096 Nov  2 19:20 .ssh
5798648    4 drwxrwx---  4 ec2-user ec2-user    4096 Oct 30 22:57 .theano
 263977    4 drwxrwxr-x  3 ec2-user ec2-user    4096 Oct 30 17:57 tools
2102242    4 drwxrwxr-x  5 ec2-user ec2-user    4096 Oct 30 20:30 tutorials
2110416    4 drwxr-xr-x  3 ec2-user ec2-user    4096 Nov  2 19:35 .yarn
```

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

A collection of Amazon SageMaker sample notebooks.

Introduction to Amazon Algorithms

DeepAR-Electricity.ipynb	Preview	Use
Image-classification-fulltraining-elastic-inference.ipynb	Preview	Use
Image-classification-fulltraining-highlevel.ipynb	Preview	Use
Image-classification-fulltraining.ipynb	Preview	Use
Image-classification-incremental-training-highlevel.ipynb	Preview	Use
Image-classification-lst-format-highlevel.ipynb	Preview	Use
Image-classification-lst-format.ipynb	Preview	Use
Image-classification-multilabel-lst.ipynb	Preview	Use
Image-classification-transfer-learning-highlevel.ipynb	Preview	Use
Image-classification-transfer-learning.ipynb	Preview	Use
LDA-Introduction.ipynb	Preview	Use
Linear_Learner_Regression_csv_format.ipynb	Preview	Use
SageMaker-Seq2Seq-Translation-English-German.ipynb	Preview	Use
blazingtext_hosting_pretrained_fasttext.ipynb	Preview	Use
blazingtext_text_classification_dbpedia.ipynb	Preview	Use
blazingtext_word2vec_subwords_text8.ipynb	Preview	Use
blazingtext_word2vec_text8.ipynb	Preview	Use

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

A collection of Amazon SageMaker sample notebooks.

Introduction to Amazon Algorithms

DeepAR-Electricity.ipynb	Preview	Use
Image-classification-fulltraining-elastic-inference.ipynb	Preview	Use
Image-classification-fulltraining-highlevel.ipynb	Preview	Use
Image-classification-fulltraining.ipynb	Preview	Use
Image-classification-incremental-training-highlevel.ipynb	Preview	Use
Image-classification-lst-format-highlevel.ipynb	Preview	Use
Image-classification-lst-format.ipynb	Preview	Use
Image-classification-multilabel-lst.ipynb	Preview	Use
Image-classification-transfer-learning-highlevel.ipynb	Preview	Use
Image-classification-transfer-learning.ipynb	Preview	Use
LDA-Introduction.ipynb	Preview	Use
Linear_Learner_Regression_csv_format.ipynb	Preview	Use
SageMaker-Seq2Seq-Translation-English-German.ipynb	Preview	Use
blazingtext_hosting_pretrained_fasttext.ipynb	Preview	Use
blazingtext_text_classification_dbpedia.ipynb	Preview	Use
blazingtext_word2vec_subwords_text8.ipynb	Preview	Use
blazingtext_word2vec_text8.ipynb	Preview	Use

jupyter

Open JupyterLab Quit Logout

image-classification-transfer-learning
LDA-Introduction.ipynb
Linear_Learner_Regression_csv_for
SageMaker-Seq2Seq-Translation-En
blazingtext_hosting_pretrained_fastt
blazingtext_text_classification_dbpe
blazingtext_word2vec_subwords_te
blazingtext_word2vec_text8.ipynb
deepar_synthetic.ipynb
factorization_machines_mnist.ipynb
ipinsights-tutorial.ipynb
k_nearest_neighbors_covtype.ipynb
linear_learner_mnist.ipynb
linear_learner_mnist_with_file_system_data_source.ipynb
managed_spot_training_object_detection.ipynb
ntm_synthetic.ipynb
object2vec_movie_recommendation.ipynb
object2vec_sentence_similarity.ipynb
object_detection_birds.ipynb
object_d
object_detection_incremental_training.ipynb

Preview Use

Create a copy in your home directory

Save copy as

linear_learner_mnist.ipynb

Saving the following directory and support files:

Files / linear_learner_mnist /

- linear_learner_mnist_with_file_system_data_source.ipynb

Cancel Create copy

-17:28 2x

jupyter

Open JupyterLab | Quit | Logout

image-classification-transfer-learning

LDA-Introduction.ipynb

Linear_Learner_Regression_csv_for

SageMaker-Seq2Seq-Translation-En

blazingtext_hosting_pretrained_fastt

blazingtext_text_classification_dbpe

blazingtext_word2vec_subwords_te

blazingtext_word2vec_text8.ipynb

deepar_synthetic.ipynb

factorization_machines_mnist.ipynb

ipinsights-tutorial.ipynb

k_nearest_neighbors_covtype.ipynb

linear_learner_mnist.ipynb

linear_learner_mnist_with_file_system_data_source.ipynb

managed_spot_training_object_detection.ipynb

ntm_synthetic.ipynb

object2vec_movie_recommendation.ipynb

object2vec_sentence_similarity.ipynb

object_detection_birds.ipynb

object_detection_image_json_format.ipynb

object_detection_incremental_training.ipynb

Preview | Use

Create a copy in your home directory

Save copy as

linear_learner_mnist_mltd.ipynb

Saving the following directory and support files:

Files / linear_learner_mnist /

- linear_learner_mnist_with_file_system_data_source.ipynb

Cancel | Create copy

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Kernel starting, please wait... Not Trusted conda_python3

Run Markdown nbdiff

Add tag

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Add tag

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (`x` , `y`). `x` is a high dimensional vector and `y` is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of `y`). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector `x` to an approximation of the label `y`.

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted conda_python3

Run Markdown nbdiff

Add tag

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Add tag

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (`x` , `y`). `x` is a high dimensional vector and `y` is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of `y`). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector `x` to an approximation of the label `y` .

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted conda_python3

Run Markdown nbdiff

Add tag

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is ∈ 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Add tag

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (`x` , `y`). `x` is a high dimensional vector and `y` is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of `y`). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector `x` to an approximation of the label `y` .

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted conda_python3

Run Markdown nbdiff

- The IAM role arn used to give training and hosting access to your data. See the documentation for how to create these. Note, if more than one role is required for notebook instances, training, and/or hosting, please replace the boto regexp with a the appropriate full IAM role arn string(s).

In [ ]: parameters Add tag

```
import re
import boto3
import sagemaker
from sagemaker import get_execution_role

sess = sagemaker.Session()

region = boto3.Session().region_name

# S3 bucket where the original mnist data is downloaded and stored.
downloaded_data_bucket = f"jumpstart-cache-prod-{region}"
downloaded_data_prefix = "1p-notebooks-datasets/mnist"

# S3 bucket for saving code and model artifacts.
# Feel free to specify a different bucket and prefix
bucket = sess.default_bucket()
prefix = "sagemaker/DEMO-linear-mnist"

# Define IAM role
role = get_execution_role()
```

Add tag

Data ingestion

Next, we read the MNIST dataset [1] from an existing repository into memory, for preprocessing prior to training. It was downloaded from this link and stored on the `downloaded_data_bucket`. Processing could be done *in situ* by Amazon Athena, Apache Spark in Amazon EMR, Amazon Redshift, etc., assuming the dataset is present in the appropriate location. Then, the next step would be to transfer the data to S3 for use in training. For small datasets, such as this one, reading into memory isn't onerous, though it would be for larger datasets.

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted conda_python3

Run Code nbdiff

In [1]: parameters Add tag

```
import re
import boto3
import sagemaker
from sagemaker import get_execution_role

sess = sagemaker.Session()

region = boto3.Session().region_name

# S3 bucket where the original mnist data is downloaded and stored.
downloaded_data_bucket = f"jumpstart-cache-prod-{region}"
downloaded_data_prefix = "1p-notebooks-datasets/mnist"

# S3 bucket for saving code and model artifacts.
# Feel free to specify a different bucket and prefix
bucket = sess.default_bucket()
prefix = "sagemaker/DEMO-linear-mnist-mltd1"

# Define IAM role
role = get_execution_role()
```

Add tag

Data ingestion

Next, we read the MNIST dataset [1] from an existing repository into memory, for preprocessing prior to training. It was downloaded from this link and stored on the `downloaded_data_bucket`. Processing could be done *in situ* by Amazon Athena, Apache Spark in Amazon EMR, Amazon Redshift, etc., assuming the dataset is present in the appropriate location. Then, the next step would be to transfer the data to S3 for use in training. For small datasets, such as this one, reading into memory isn't onerous, though it would be for larger datasets.

> [1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

Data conversion

Since algorithms have particular input and output requirements, converting the dataset is also part of the process that a data scientist goes through prior to initiating training. In this particular case, the Amazon SageMaker implementation of Linear Learner takes recordIO-wrapped protobuf, where the data we have today is a pickle-ized numpy array on disk.

Most of the conversion effort is handled by the Amazon SageMaker Python SDK, imported as `sagemaker` below.

In [4]:

```
import io
import numpy as np
import sagemaker.amazon.common as smac

vectors = np.array([t.tolist() for t in train_set[0]]).astype("float32")
labels = np.where(np.array([t.tolist() for t in train_set[1]]) == 0, 1, 0).astype("float32")

buf = io.BytesIO()
smac.write_numpy_to_dense_tensor(buf, vectors, labels)
buf.seek(0)
```

Out[4]: 0

Upload training data

Now that we've created our recordIO-wrapped protobuf, we'll need to upload it to S3, so that Amazon SageMaker training can use it.

In [5]:

```
import boto3
import os
```

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

Upload training data

Now that we've created our recordIO-wrapped protobuf, we'll need to upload it to S3, so that Amazon SageMaker training can use it.

In [5]:

```
import boto3
import os

key = "recordio-pb-data"
boto3.resource("s3").Bucket(bucket).Object(os.path.join(prefix, "train", key)).upload_fileobj(buf)
s3_train_data = f"s3://{bucket}/{prefix}/train/{key}"
print(f"uploaded training data location: {s3_train_data}")
```

```
uploaded training data location: s3://sagemaker-us-east-1-649088000912/sagemaker/DEMO-linear-mnist-mltd1/train/reco
rdio-pb-data
```

Let's also setup an output S3 location for the model artifact that will be output as the result of training with the algorithm.

In [6]:

```
output_location = f"s3://{bucket}/{prefix}/output"
print(f"training artifacts will be uploaded to: {output_location}")
```

```
training artifacts will be uploaded to: s3://sagemaker-us-east-1-649088000912/sagemaker/DEMO-linear-mnist-mltd1/out
put
```

Training the linear model

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

Upload training data

Now that we've created our recordIO-wrapped protobuf, we'll need to upload it to S3, so that Amazon SageMaker training can use it.

In [5]:

```
import boto3
import os

key = "recordio-pb-data"
boto3.resource("s3").Bucket(bucket).Object(os.path.join(prefix, "train", key)).upload_fileobj(buf)
s3_train_data = f"s3://{bucket}/{prefix}/train/{key}"
print(f"uploaded training data location: {s3_train_data}")
```

```
uploaded training data location: s3://sagemaker-us-east-1-649088000912/sagemaker/DEMO-linear-mnist-mltd1/train/reco
rdio-pb-data
```

Let's also setup an output S3 location for the model artifact that will be output as the result of training with the algorithm.

In [6]:

```
output_location = f"s3://{bucket}/{prefix}/output"
print(f"training artifacts will be uploaded to: {output_location}")
```

```
training artifacts will be uploaded to: s3://sagemaker-us-east-1-649088000912/sagemaker/DEMO-linear-mnist-mltd1/out
put
```

Training the linear model

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

Add tag

Training the linear model

Once we have the data preprocessed and available in the correct format for training, the next step is to actually train the model using the data. Since this data is relatively small, it isn't meant to show off the performance of the Linear Learner training algorithm, although we have tested it on multi-terabyte datasets.

Again, we'll use the Amazon SageMaker Python SDK to kick off training, and monitor status until it is completed. In this example that takes between 7 and 11 minutes. Despite the dataset being small, provisioning hardware and loading the algorithm container take time upfront.

First, let's specify our containers. Since we want this notebook to run in all 4 of Amazon SageMaker's regions, we'll create a small lookup. More details on algorithm containers can be found in AWS documentation.

In [7]:

Add tag

```
from sagemaker.amazon.amazon_estimator import get_image_uri

container = get_image_uri(boto3.Session().region_name, "linear-learner")
```

```
The method get_image_uri has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
Defaulting to the only supported framework/algorithm version: 1. Ignoring framework/algorithm version: 1.
```

Add tag

Next we'll kick off the base estimator, making sure to pass in the necessary hyperparameters. Notice:

- `feature_dim` is set to 784, which is the number of pixels in each 28 x 28 image.
- `predictor_type` is set to 'binary_classifier' since we are trying to predict whether the image is or is not a 0.
- `mini_batch_size` is set to 200. This value can be tuned for relatively minor improvements in fit and speed, but selecting a reasonable value relative to the dataset is appropriate in most cases.

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

Once we have the data preprocessed and available in the correct format for training, the next step is to actually train the model using the data. Since this data is relatively small, it isn't meant to show off the performance of the Linear Learner training algorithm, although we have tested it on multi-terabyte datasets.

Again, we'll use the Amazon SageMaker Python SDK to kick off training, and monitor status until it is completed. In this example that takes between 7 and 11 minutes. Despite the dataset being small, provisioning hardware and loading the algorithm container take time upfront.

First, let's specify our containers. Since we want this notebook to run in all 4 of Amazon SageMaker's regions, we'll create a small lookup. More details on algorithm containers can be found in AWS documentation.

In [7]:

Add tag

```
from sagemaker.amazon.amazon_estimator import get_image_uri

container = get_image_uri(boto3.Session().region_name, "linear-learner")
```

```
The method get_image_uri has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
Defaulting to the only supported framework/algorithm version: 1. Ignoring framework/algorithm version: 1.
```

Add tag

Next we'll kick off the base estimator, making sure to pass in the necessary hyperparameters. Notice:

- `feature_dim` is set to 784, which is the number of pixels in each 28 x 28 image.
- `predictor_type` is set to 'binary_classifier' since we are trying to predict whether the image is or is not a 0.
- `mini_batch_size` is set to 200. This value can be tuned for relatively minor improvements in fit and speed, but selecting a reasonable value relative to the dataset is appropriate in most cases.

In [*]:

Add tag

```
import boto3
```

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

In [*]:

```
import boto3
import sagemaker

sess = sagemaker.Session()

linear = sagemaker.estimator.Estimator(
    container,
    role,
    instance_count=1,
    instance_type="ml.c4.xlarge",
    output_path=output_location,
    sagemaker_session=sess,
)
linear.set_hyperparameters(feature_dim=784, predictor_type="binary_classifier", mini_batch_size=200)

linear.fit({"train": s3_train_data})
```

```
2020-12-20 11:37:02 Starting - Starting the training job...
2020-12-20 11:37:26 Starting - Launching requested ML instancesProfilerReport-1608464222: InProgress
......
2020-12-20 11:38:26 Starting - Preparing the instances for training.......
```

Set up hosting for the model

Now that we've trained our model, we can deploy it behind an Amazon SageMaker real-time hosted endpoint. This will allow out to make predictions (or inference) from the model dyanamically.

Note, Amazon SageMaker allows you the flexibility of importing models trained elsewhere, as well as the choice of not importing models if the target of model creation is AWS Lambda, AWS Greengrass, Amazon Redshift, Amazon Athena, or other deployment target.

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

In [*]:

Add tag

```
import boto3
import sagemaker

sess = sagemaker.Session()

linear = sagemaker.estimator.Estimator(
    container,
    role,
    instance_count=1,
    instance_type="ml.c4.xlarge",
    output_path=output_location,
    sagemaker_session=sess,
)
linear.set_hyperparameters(feature_dim=784, predictor_type="binary_classifier", mini_batch_size=200)

linear.fit({"train": s3_train_data})
```

```
2020-12-20 11:37:02 Starting - Starting the training job...
2020-12-20 11:37:26 Starting - Launching requested ML instancesProfilerReport-1608464222: InProgress
......
2020-12-20 11:38:26 Starting - Preparing the instances for training.........
```

Add tag

Set up hosting for the model

Now that we've trained our model, we can deploy it behind an Amazon SageMaker real-time hosted endpoint. This will allow out to make predictions (or inference) from the model dyanamically.

Note, Amazon SageMaker allows you the flexibility of importing models trained elsewhere, as well as the choice of not importing models if the target of model creation is AWS Lambda, AWS Greengrass, Amazon Redshift, Amazon Athena, or other deployment target.

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

In [*]:

Add tag

```
import boto3
import sagemaker

sess = sagemaker.Session()

linear = sagemaker.estimator.Estimator(
    container,
    role,
    instance_count=1,
    instance_type="ml.c4.xlarge",
    output_path=output_location,
    sagemaker_session=sess,
)
linear.set_hyperparameters(feature_dim=784, predictor_type="binary_classifier", mini_batch_size=200)

linear.fit({"train": s3_train_data})
```

```
2020-12-20 11:37:02 Starting - Starting the training job...
2020-12-20 11:37:26 Starting - Launching requested ML instancesProfilerReport-1608464222: InProgress
......
2020-12-20 11:38:26 Starting - Preparing the instances for training.........
2020-12-20 11:40:03 Downloading - Downloading input data
2020-12-20 11:40:03 Training - Downloading the training image..
```

Add tag

Set up hosting for the model

Now that we've trained our model, we can deploy it behind an Amazon SageMaker real-time hosted endpoint. This will allow out to make predictions (or inference) from the model dyanamically.

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

In [*]:

```
import boto3
import sagemaker

sess = sagemaker.Session()

linear = sagemaker.estimator.Estimator(
    container,
    role,
    instance_count=1,
    instance_type="ml.c4.xlarge",
    output_path=output_location,
    sagemaker_session=sess,
)
linear.set_hyperparameters(feature_dim=784, predictor_type="binary_classifier", mini_batch_size=200)

linear.fit({"train": s3_train_data})
```

```
2020-12-20 11:37:02 Starting - Starting the training job...
2020-12-20 11:37:26 Starting - Launching requested ML instancesProfilerReport-1608464222: InProgress
......
2020-12-20 11:38:26 Starting - Preparing the instances for training.........
2020-12-20 11:40:03 Downloading - Downloading input data
2020-12-20 11:40:03 Training - Downloading the training image...
2020-12-20 11:40:28 Training - Training image download completed. Training in progress.
```

Set up hosting for the model

Now that we've trained our model, we can deploy it behind an Amazon SageMaker real-time hosted endpoint. This will allow out to make predictions (or inference) from the model dyanamically.

jupyter linear_learner_mnist_mltd (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Raw NBConvert nbdiff

```
lr": 0.0001}
[12/20/2020 11:42:25 INFO 140186382522176] Saved checkpoint to "/tmp/tmpt_qYOk/mx-mod-0000.params"
[12/20/2020 11:42:25 INFO 140186382522176] Test data is not provided.
#metrics {"Metrics": {"totaltime": {"count": 1, "max": 113986.12594604492, "sum": 113986.12594604492, "min": 11398
6.12594604492}, "finalize.time": {"count": 1, "max": 1398.9779949188232, "sum": 1398.9779949188232, "min": 1398.97
79949188232}, "initialize.time": {"count": 1, "max": 566.2128925323486, "sum": 566.2128925323486, "min": 566.21289
25323486}, "check_early_stopping.time": {"count": 15, "max": 1.2140274047851562, "sum": 14.268159866333008, "min":
0.8690357208251953}, "setuptime": {"count": 1, "max": 30.244827270507812, "sum": 30.244827270507812, "min": 30.244
827270507812}, "update.time": {"count": 15, "max": 7920.625925064087, "sum": 111644.9511051178, "min": 7310.935020
446777}, "epochs": {"count": 1, "max": 15, "sum": 15.0, "min": 15}}, "EndTime": 1608464545.324948, "Dimensions": {
"Host": "algo-1", "Operation": "training", "Algorithm": "Linear Learner"}, "StartTime": 1608464431.648941}


Training seconds: 166
Billable seconds: 166
```

Add tag

Set up hosting for the model

Now that we've trained our model, we can deploy it behind an Amazon SageMaker real-time hosted endpoint. This will allow out to make predictions (or inference) from the model dyanamically.

Note, Amazon SageMaker allows you the flexibility of importing models trained elsewhere, as well as the choice of not importing models if the target of model creation is AWS Lambda, AWS Greengrass, Amazon Redshift, Amazon Athena, or other deployment target.

In [9]:

Add tag

```
linear_predictor = linear.deploy(initial_instance_count=1, instance_type="ml.m4.xlarge")
```

```
---------------!
```

Add tag

jupyter linear_learner_mnist_mltd (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Raw NBConvert nbdiff

OK, a single prediction works. We see that for one record our endpoint returned some JSON which contains `predictions`, including the `score` and `predicted_label`. In this case, `score` will be a continuous value between [0, 1] representing the probability we think the digit is a 0 or not. `predicted_label` will take a value of either `0` or `1` where (somewhat counterintuitively) `1` denotes that we predict the image is a 0, while `0` denotes that we are predicting the image is not of a 0.

Let's do a whole batch of images and evaluate our predictive accuracy.

In [12]:

```
import numpy as np

predictions = []
for array in np.array_split(test_set[0], 100):
    result = linear_predictor.predict(array)
    predictions += [r["predicted_label"] for r in result["predictions"]]

predictions = np.array(predictions)
```

In [13]:

```
import pandas as pd

pd.crosstab(
    np.where(test_set[1] == 0, 1, 0), predictions, rownames=["actuals"], colnames=["predictions"]
)
```

Out[13]:

predictions	0	1
actuals		
0	8973	47
1	37	943

Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints
Batch transform jobs
Edge Inference
Get started
Edge packaging jobs
Edge device fleets
Edge devices
Augmented AI

Training jobs

Actions | Create training job

Search training jobs

1 ...

Name	Creation time	Duration	Status
linear-learner-2020-12-20-11-37-02-293	Dec 20, 2020 11:37 UTC	6 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-10-06-24-228	Dec 20, 2020 10:06 UTC	4 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-10-06-23-446	Dec 20, 2020 10:06 UTC	4 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-10-06-12-389	Dec 20, 2020 10:06 UTC	3 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-10-06-11-819	Dec 20, 2020 10:06 UTC	3 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-10-06-11-504	Dec 20, 2020 10:06 UTC	4 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-09-57-14-462	Dec 20, 2020 09:57 UTC	4 minutes	Completed
demo-xgboost-customer-churn-2020-12-01-13-24-05-381	Dec 01, 2020 13:24 UTC	4 minutes	Completed
demo-xgboost-customer-churn-2020-12-01-13-10-24-030	Dec 01, 2020 13:10 UTC	5 minutes	Completed
demo-xgboost-customer-churn-2020-12-01-13-10-23-373	Dec 01, 2020 13:10 UTC	4 minutes	Completed

jupyter

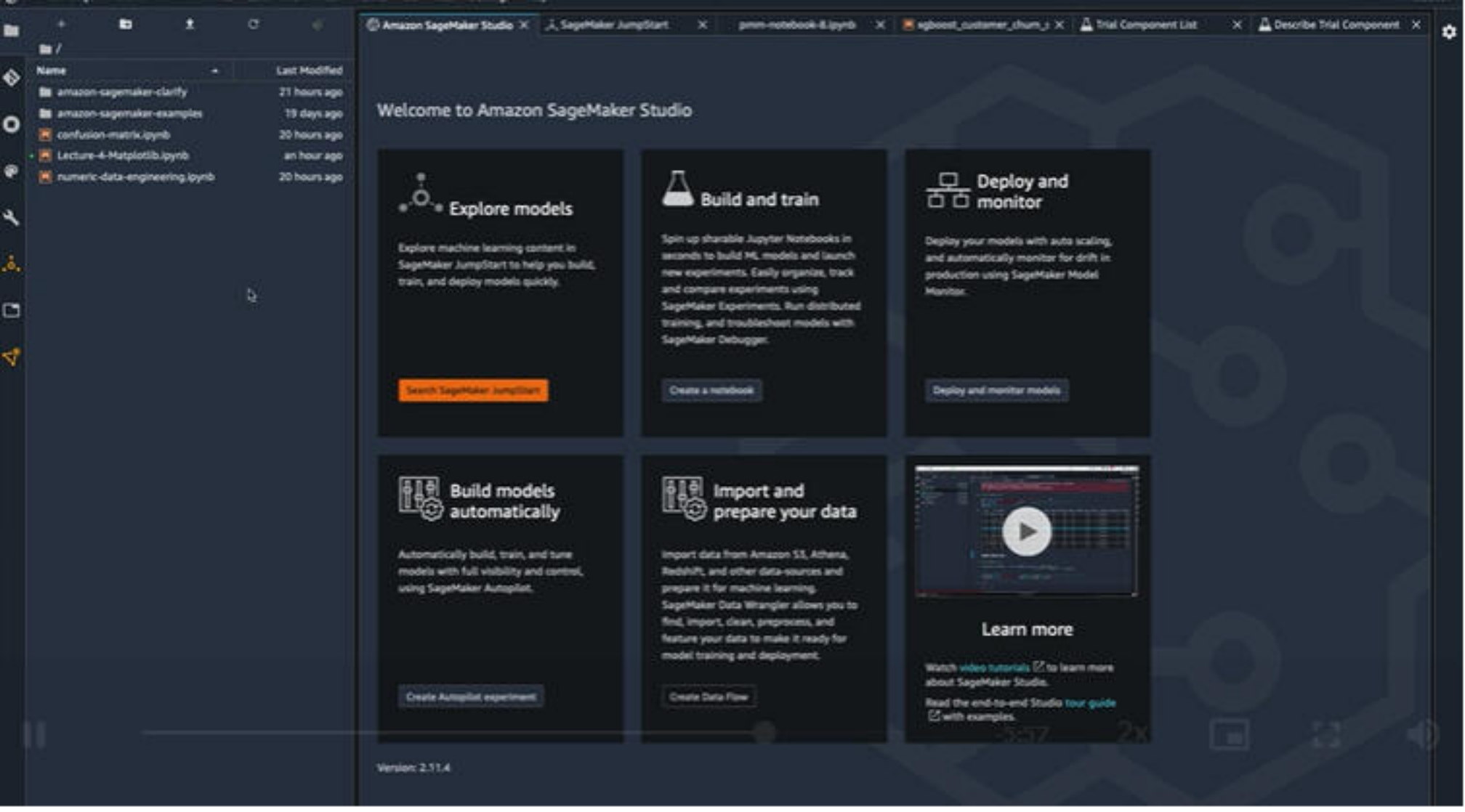
Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

Select items to perform actions on them.

Upload | New

0 / linear_learner_mnist_2020-12-20		Name	Last Modified	File size
..			seconds ago	
linear_learner_mnist_mltd.ipynb	Running		a minute ago	302 kB
linear_learner_mnist_with_file_system_data_source.ipynb			2 days ago	6.46 kB
mnist.pkl.gz			20 minutes ago	16.2 MB


Amazon SageMaker Studio
SageMaker JumpStart
pmm-notebook-8.ipynb
xgboost_customer_churn_s
Trial Component List
Describe Trial Component
/
Name
Last Modified
amazon-sagemaker-clarify
21 hours ago
amazon-sagemaker-examples
19 days ago
confusion-matrix.ipynb
20 hours ago
Lecture-4-Matplotlib.ipynb
an hour ago
numeric-data-engineering.ipynb
20 hours ago
Welcome to Amazon SageMaker Studio
Explore models
Explore machine learning content in SageMaker JumpStart to help you build, train, and deploy models quickly.
Search SageMaker JumpStart
Build and train
Spin up sharable Jupyter Notebooks in seconds to build ML models and launch new experiments. Easily organize, track and compare experiments using SageMaker Experiments. Run distributed training, and troubleshoot models with SageMaker Debugger.
Create a notebook
Deploy and monitor
Deploy your models with auto scaling, and automatically monitor for drift in production using SageMaker Model Monitor.
Deploy and monitor models
Build models automatically
Automatically build, train, and tune models with full visibility and control, using SageMaker Autopilot.
Create Autopilot experiment
Import and prepare your data
Import data from Amazon S3, Athena, Redshift, and other data-sources and prepare it for machine learning. SageMaker Data Wrangler allows you to find, import, clean, preprocess, and feature your data to make it ready for model training and deployment.
Create Data Flow
Learn more
Watch video tutorials to learn more about SageMaker Studio.
Read the end-to-end Studio tour guide with examples.
Version: 2.11.4

You are not currently in a Git repository. To use Git, navigate to a local repository, initialize a repository here, or clone an existing repository.

Open the File Browser

Initialize a Repository

Clone a Repository

Amazon SageMaker Studio | SageMaker JumpStart | pmm-notebook-8.ipynb | xgboost_customer_churn_s | Trial Component List | Describe Trial Component

Welcome to Amazon SageMaker Studio

Explore models

Explore machine learning content in SageMaker JumpStart to help you build, train, and deploy models quickly.

Search SageMaker JumpStart

Build and train

Spin up shareable Jupyter Notebooks in seconds to build ML models and launch new experiments. Easily organize, track and compare experiments using SageMaker Experiments. Run distributed training, and troubleshoot models with SageMaker Debugger.

Create a notebook

Deploy and monitor

Deploy your models with auto scaling, and automatically monitor for drift in production using SageMaker Model Monitor.

Deploy and monitor models

Build models automatically

Automatically build, train, and tune models with full visibility and control, using SageMaker Autopilot.

Create Autopilot experiment

Import and prepare your data

Import data from Amazon S3, Athena, Redshift, and other data-sources and prepare it for machine learning. SageMaker Data Wrangler allows you to find, import, clean, preprocess, and feature your data to make it ready for model training and deployment.

Create Data Flow

Learn more

Watch video tutorials to learn more about SageMaker Studio.

Read the end-to-end Studio tour guide with examples.

Version: 2.11.4

You are not currently in a Git repository. To use Git, navigate to a local repository, initialize a repository here, or clone an existing repository.

Open the File Browser

Initialize a Repository

Clone a Repository

Amazon SageMaker Studio | SageMaker JumpStart | pmm-notebook-8.ipynb | xgboost_customer_churn_s | Trial Component List | Describe Trial Component

Welcome to Amazon SageMaker Studio

Explore models

Explore machine learning content in SageMaker JumpStart to help you build, train, and deploy models quickly.

Search SageMaker JumpStart

Build and train

Spin up sharable Jupyter Notebooks in seconds to build ML models and launch new experiments. Easily organize, track and compare experiments using SageMaker Experiments. Run distributed training, and troubleshoot models with SageMaker Debugger.

Create a notebook

Deploy and monitor

Deploy your models with auto scaling, and automatically monitor for drift in production using SageMaker Model Monitor.

Deploy and monitor models

Build models automatically

Automatically build, train, and tune models with full visibility and control, using SageMaker Autopilot.

Create Autopilot experiment

Import and prepare your data

Import data from Amazon S3, Athena, Redshift, and other data-sources and prepare it for machine learning. SageMaker Data Wrangler allows you to find, import, clean, preprocess, and feature your data to make it ready for model training and deployment.

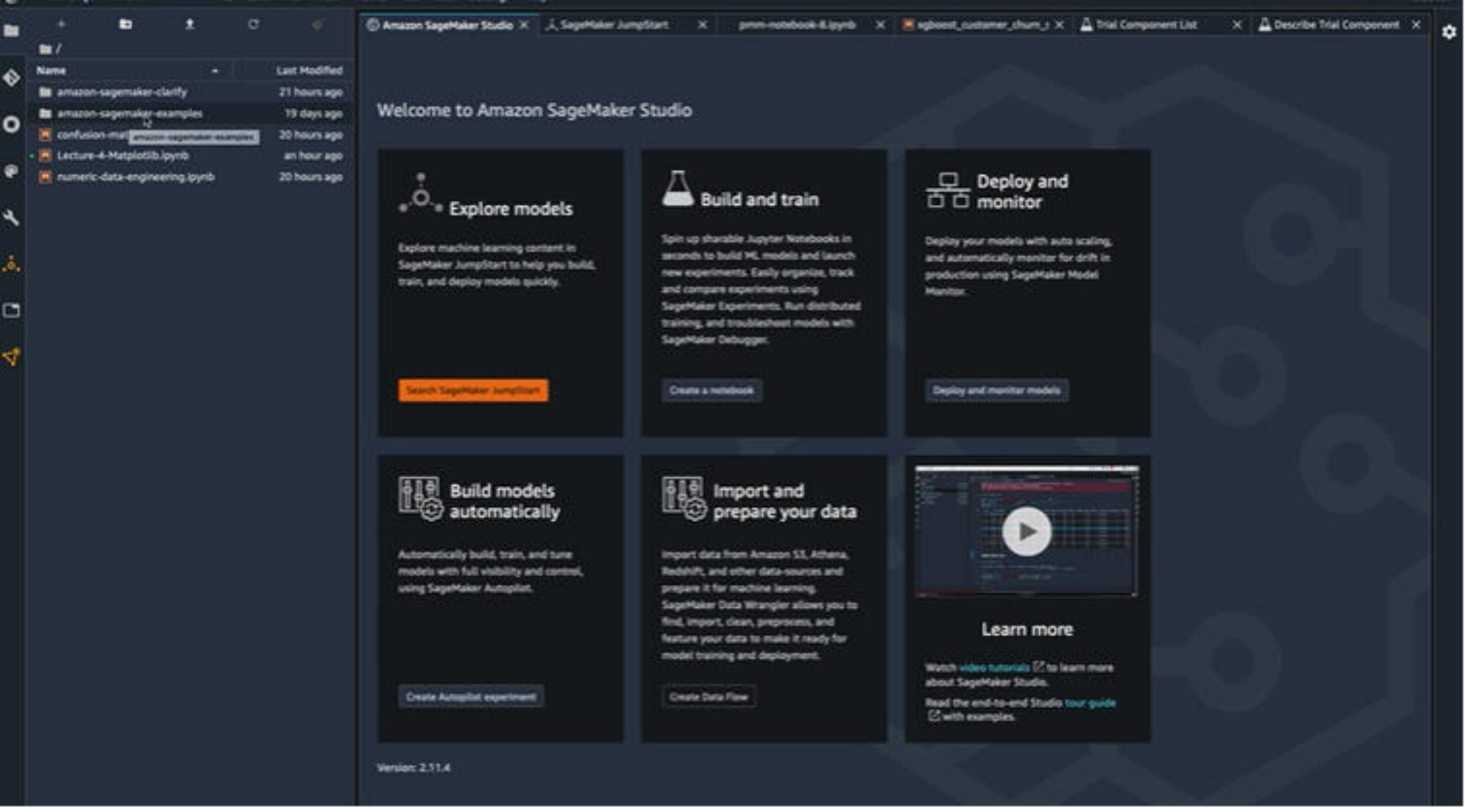
Create Data Flow

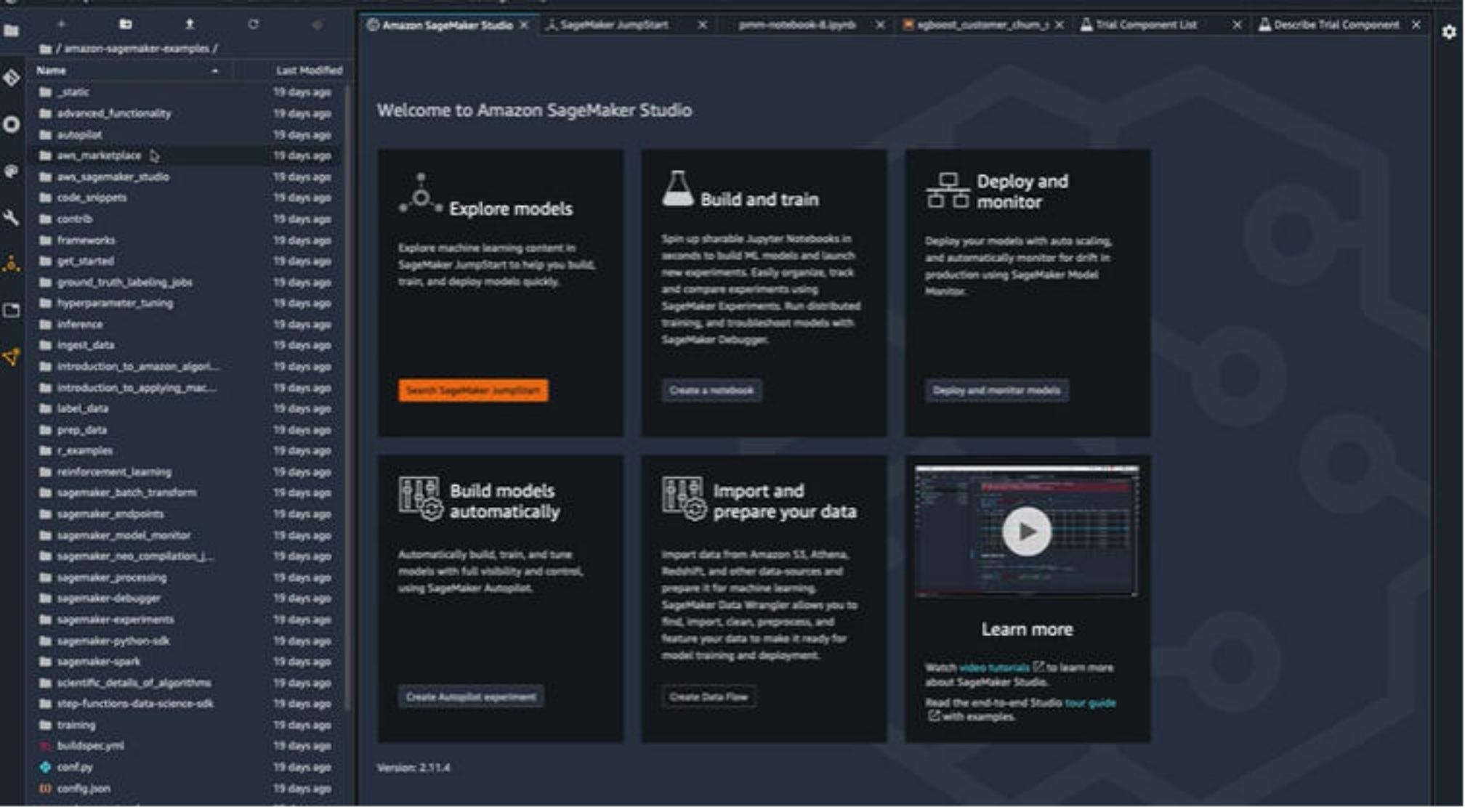
Learn more

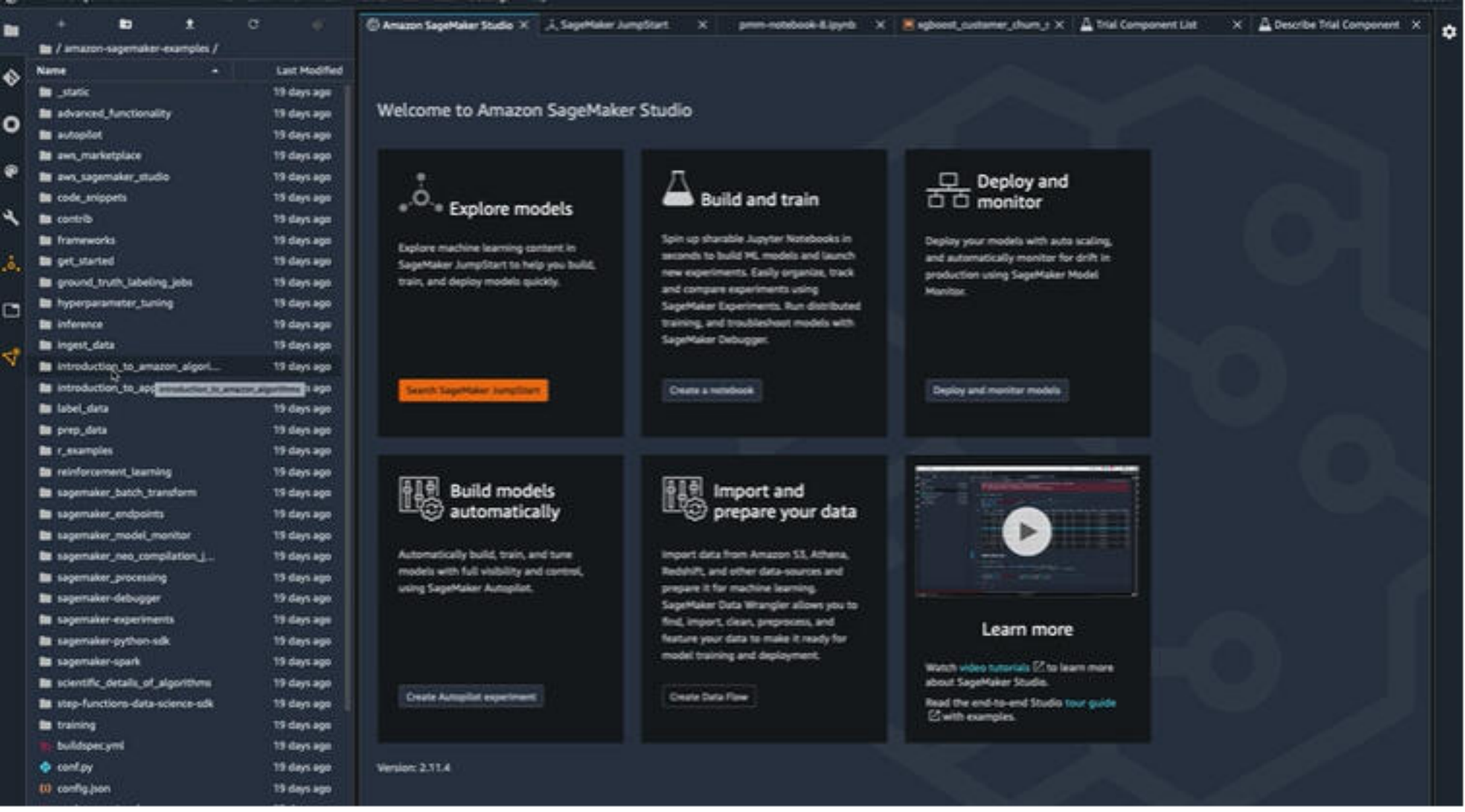
Watch video tutorials to learn more about SageMaker Studio.

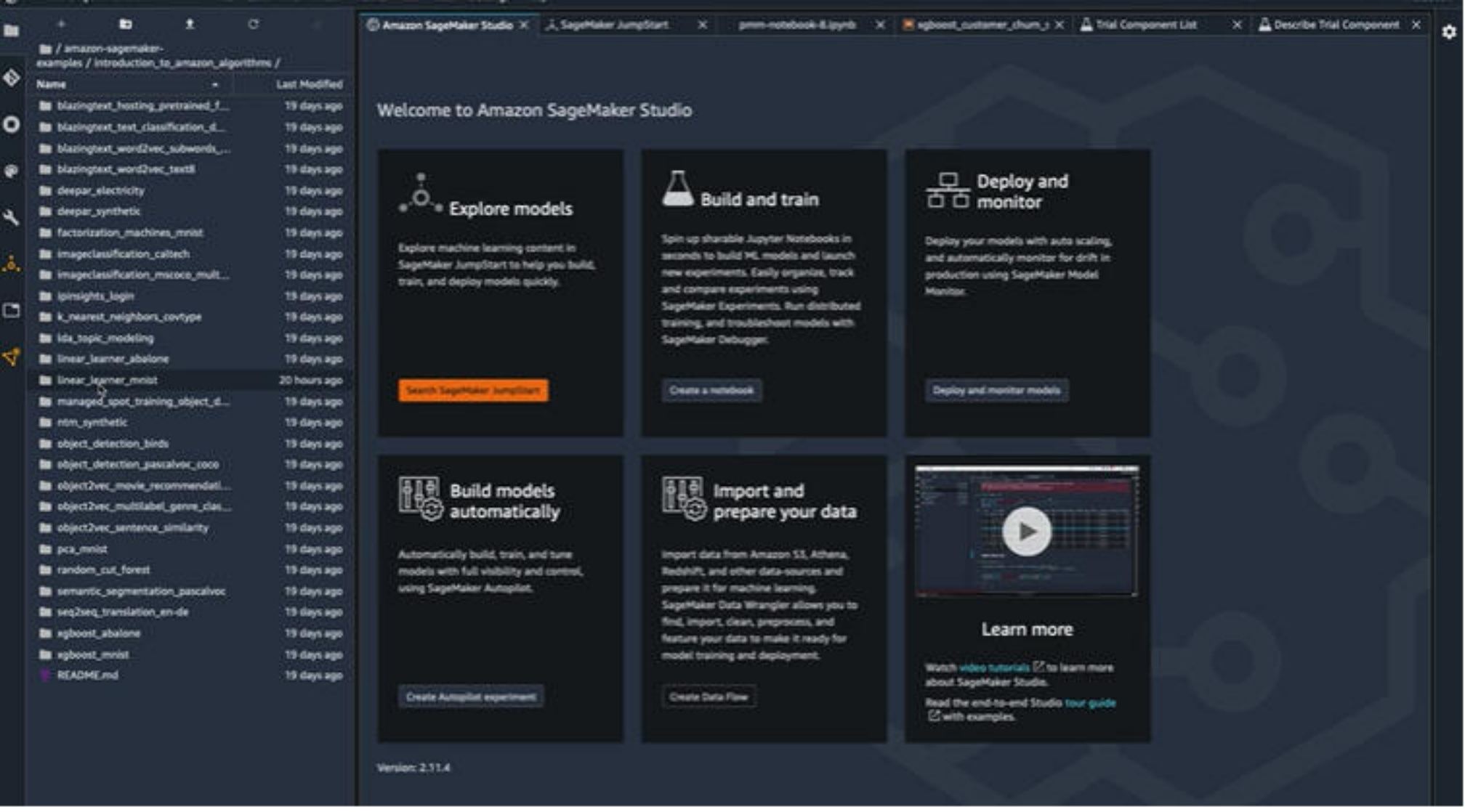
Read the end-to-end Studio tour guide with examples.

Version: 2.11.4


Amazon SageMaker Studio
SageMaker JumpStart
pmm-notebook-8.ipynb
xgboost_customer_churn_s
Trial Component List
Describe Trial Component
/
Name
Last Modified
amazon-sagemaker-clarify
21 hours ago
amazon-sagemaker-examples
19 days ago
confusion-mat
20 hours ago
Lecture-4-Matplotlib.ipynb
an hour ago
numeric-data-engineering.ipynb
20 hours ago
Welcome to Amazon SageMaker Studio
Explore models
Explore machine learning content in SageMaker JumpStart to help you build, train, and deploy models quickly.
Search SageMaker JumpStart
Build and train
Spin up sharable Jupyter Notebooks in seconds to build ML models and launch new experiments. Easily organize, track and compare experiments using SageMaker Experiments. Run distributed training, and troubleshoot models with SageMaker Debugger.
Create a notebook
Deploy and monitor
Deploy your models with auto scaling, and automatically monitor for drift in production using SageMaker Model Monitor.
Deploy and monitor models
Build models automatically
Automatically build, train, and tune models with full visibility and control, using SageMaker Autopilot.
Create Autopilot experiment
Import and prepare your data
Import data from Amazon S3, Athena, Redshift, and other data-sources and prepare it for machine learning. SageMaker Data Wrangler allows you to find, import, clean, preprocess, and feature your data to make it ready for model training and deployment.
Create Data Flow
Learn more
Watch video tutorials to learn more about SageMaker Studio.
Read the end-to-end Studio tour guide with examples.
Version: 2.11.4


Amazon SageMaker Studio
SageMaker JumpStart
pmm-notebook-8.ipynb
xgboost_customer_churn_s
Trial Component List
Describe Trial Component
/ amazon-sagemaker-examples /
Name
Last Modified
_static
advanced_functionality
autopilot
aws_marketplace
aws_sagemaker_studio
code_snippets
contrib
frameworks
get_started
ground_truth_labeling_jobs
hyperparameter_tuning
inference
ingest_data
introduction_to_amazon_algori...
introduction_to_applying_mac...
label_data
prep_data
r_examples
reinforcement_learning
sagemaker_batch_transform
sagemaker_endpoints
sagemaker_model_monitor
sagemaker_neo_compilation_j...
sagemaker_processing
sagemaker-debugger
sagemaker-experiments
sagemaker-python-sdk
sagemaker-spark
scientific_details_of_algorithms
step-functions-data-science-sdk
training
buildspec.yml
conf.py
config.json
19 days ago
Welcome to Amazon SageMaker Studio
Explore models
Explore machine learning content in SageMaker JumpStart to help you build, train, and deploy models quickly.
Search SageMaker JumpStart
Build and train
Spin up sharable Jupyter Notebooks in seconds to build ML models and launch new experiments. Easily organize, track and compare experiments using SageMaker Experiments. Run distributed training, and troubleshoot models with SageMaker Debugger.
Create a notebook
Deploy and monitor
Deploy your models with auto scaling, and automatically monitor for drift in production using SageMaker Model Monitor.
Deploy and monitor models
Build models automatically
Automatically build, train, and tune models with full visibility and control, using SageMaker Autopilot.
Create Autopilot experiment
Import and prepare your data
Import data from Amazon S3, Athena, Redshift, and other data-sources and prepare it for machine learning. SageMaker Data Wrangler allows you to find, import, clean, preprocess, and feature your data to make it ready for model training and deployment.
Create Data Flow
Learn more
Watch video tutorials to learn more about SageMaker Studio.
Read the end-to-end Studio tour guide with examples.
Version: 2.11.4


Amazon SageMaker Studio
SageMaker JumpStart
pmm-notebook-8.ipynb
xgboost_customer_churn_c
Trial Component List
Describe Trial Component
/ amazon-sagemaker-examples /
Name
Last Modified
_static
advanced_functionality
autopilot
aws_marketplace
aws_sagemaker_studio
code_snippets
contrib
frameworks
get_started
ground_truth_labeling_jobs
hyperparameter_tuning
inference
ingest_data
introduction_to_amazon_algori...
introduction_to_ap
label_data
prep_data
r_examples
reinforcement_learning
sagemaker_batch_transform
sagemaker_endpoints
sagemaker_model_monitor
sagemaker_neo_compilation_j...
sagemaker_processing
sagemaker-debugger
sagemaker-experiments
sagemaker-python-sdk
sagemaker-spark
scientific_details_of_algorithms
step-functions-data-science-sdk
training
buildspec.yml
conf.py
config.json
19 days ago
Welcome to Amazon SageMaker Studio
Explore models
Explore machine learning content in SageMaker JumpStart to help you build, train, and deploy models quickly.
Search SageMaker JumpStart
Build and train
Spin up sharable Jupyter Notebooks in seconds to build ML models and launch new experiments. Easily organize, track and compare experiments using SageMaker Experiments. Run distributed training, and troubleshoot models with SageMaker Debugger.
Create a notebook
Deploy and monitor
Deploy your models with auto scaling, and automatically monitor for drift in production using SageMaker Model Monitor.
Deploy and monitor models
Build models automatically
Automatically build, train, and tune models with full visibility and control, using SageMaker Autopilot.
Create Autopilot experiment
Import and prepare your data
Import data from Amazon S3, Athena, Redshift, and other data-sources and prepare it for machine learning. SageMaker Data Wrangler allows you to find, import, clean, preprocess, and feature your data to make it ready for model training and deployment.
Create Data Flow
Learn more
Watch video tutorials to learn more about SageMaker Studio.
Read the end-to-end Studio tour guide with examples.
Version: 2.11.4


Amazon SageMaker Studio
SageMaker JumpStart
pmm-notebook-8.ipynb
xgboost_customer_churn_s
Trial Component List
Describe Trial Component
/ amazon-sagemaker-examples / introduction_to_amazon_algorithms /
Name
Last Modified
blazingtext_hosting_pretrained_f...
blazingtext_text_classification_d...
blazingtext_word2vec_subwords_...
blazingtext_word2vec_textfl
deepar_electricity
deepar_synthetic
factorization_machines_mnist
imageclassification_caltech
imageclassification_mscoco_mult...
ipinsights_login
k_nearest_neighbors_covtype
lda_topic_modeling
linear_learner_abalone
linear_learner_mnist
managed_spot_training_object_d...
ntm_synthetic
object_detection_birds
object_detection_pascalvoc_coco
object2vec_movie_recommendati...
object2vec_multilabel_genre_clas...
object2vec_sentence_similarity
pca_mnist
random_cut_forest
semantic_segmentation_pascalvoc
seq2seq_translation_en-de
xgboost_abalone
xgboost_mnist
README.md
Welcome to Amazon SageMaker Studio
Explore models
Explore machine learning content in SageMaker JumpStart to help you build, train, and deploy models quickly.
Search SageMaker JumpStart
Build and train
Spin up sharable Jupyter Notebooks in seconds to build ML models and launch new experiments. Easily organize, track and compare experiments using SageMaker Experiments. Run distributed training, and troubleshoot models with SageMaker Debugger.
Create a notebook
Deploy and monitor
Deploy your models with auto scaling, and automatically monitor for drift in production using SageMaker Model Monitor.
Deploy and monitor models
Build models automatically
Automatically build, train, and tune models with full visibility and control, using SageMaker Autopilot.
Create Autopilot experiment
Import and prepare your data
Import data from Amazon S3, Athena, Redshift, and other data-sources and prepare it for machine learning. SageMaker Data Wrangler allows you to find, import, clean, preprocess, and feature your data to make it ready for model training and deployment.
Create Data Flow
Learn more
Watch video tutorials to learn more about SageMaker Studio.
Read the end-to-end Studio tour guide with examples.
Version: 2.11.4

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

A collection of Amazon SageMaker sample notebooks.

Introduction to Amazon Algorithms

DeepAR-Electricity.ipynb	Preview	Use
Image-classification-fulltraining-elastic-inference.ipynb	Preview	Use
Image-classification-fulltraining-highlevel.ipynb	Preview	Use
Image-classification-fulltraining.ipynb	Preview	Use
Image-classification-incremental-training-highlevel.ipynb	Preview	Use
Image-classification-lst-format-highlevel.ipynb	Preview	Use
Image-classification-lst-format.ipynb	Preview	Use
Image-classification-multilabel-lst.ipynb	Preview	Use
Image-classification-transfer-learning-highlevel.ipynb	Preview	Use
Image-classification-transfer-learning.ipynb	Preview	Use
LDA-Introduction.ipynb	Preview	Use
Linear_Learner_Regression_csv_format.ipynb	Preview	Use
SageMaker-Seq2Seq-Translation-English-German.ipynb	Preview	Use
blazingtext_hosting_pretrained_fasttext.ipynb	Preview	Use
blazingtext_text_classification_dbpedia.ipynb	Preview	Use
blazingtext_word2vec_subwords_text8.ipynb	Preview	Use
blazingtext_word2vec_text8.ipynb	Preview	Use

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

A collection of Amazon SageMaker sample notebooks.

Introduction to Amazon Algorithms

DeepAR-Electricity.ipynb	Preview	Use
Image-classification-fulltraining-elastic-inference.ipynb	Preview	Use
Image-classification-fulltraining-highlevel.ipynb	Preview	Use
Image-classification-fulltraining.ipynb	Preview	Use
Image-classification-incremental-training-highlevel.ipynb	Preview	Use
Image-classification-lst-format-highlevel.ipynb	Preview	Use
Image-classification-lst-format.ipynb	Preview	Use
Image-classification-multilabel-lst.ipynb	Preview	Use
Image-classification-transfer-learning-highlevel.ipynb	Preview	Use
Image-classification-transfer-learning.ipynb	Preview	Use
LDA-Introduction.ipynb	Preview	Use
Linear_Learner_Regression_csv_format.ipynb	Preview	Use
SageMaker-Seq2Seq-Translation-English-German.ipynb	Preview	Use
blazingtext_hosting_pretrained_fasttext.ipynb	Preview	Use
blazingtext_text_classification_dbpedia.ipynb	Preview	Use
blazingtext_word2vec_subwords_text8.ipynb	Preview	Use
blazingtext_word2vec_text8.ipynb	Preview	Use

jupyter

Open JupyterLab | Quit | Logout

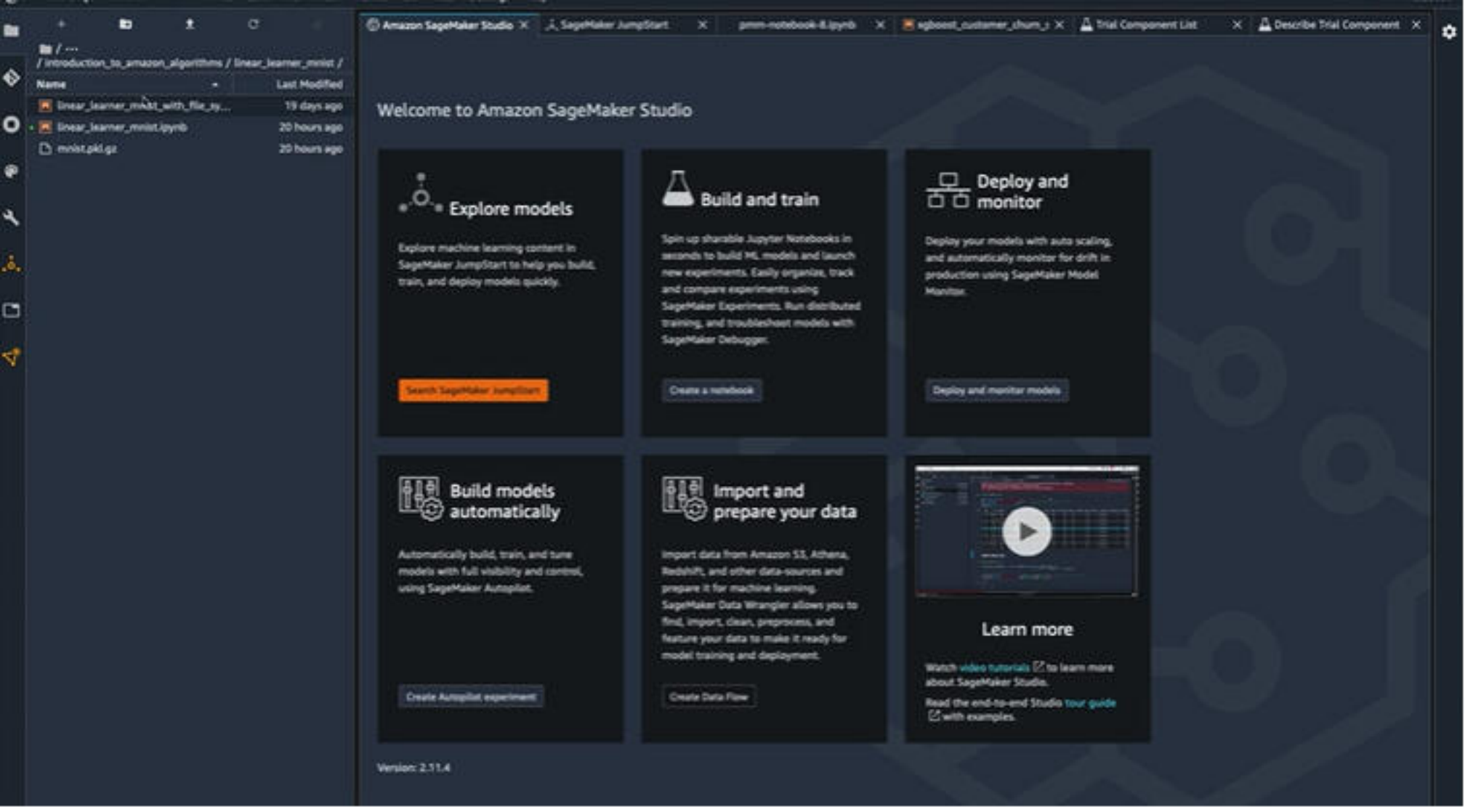
Files | Running | Clusters | SageMaker Examples | Conda

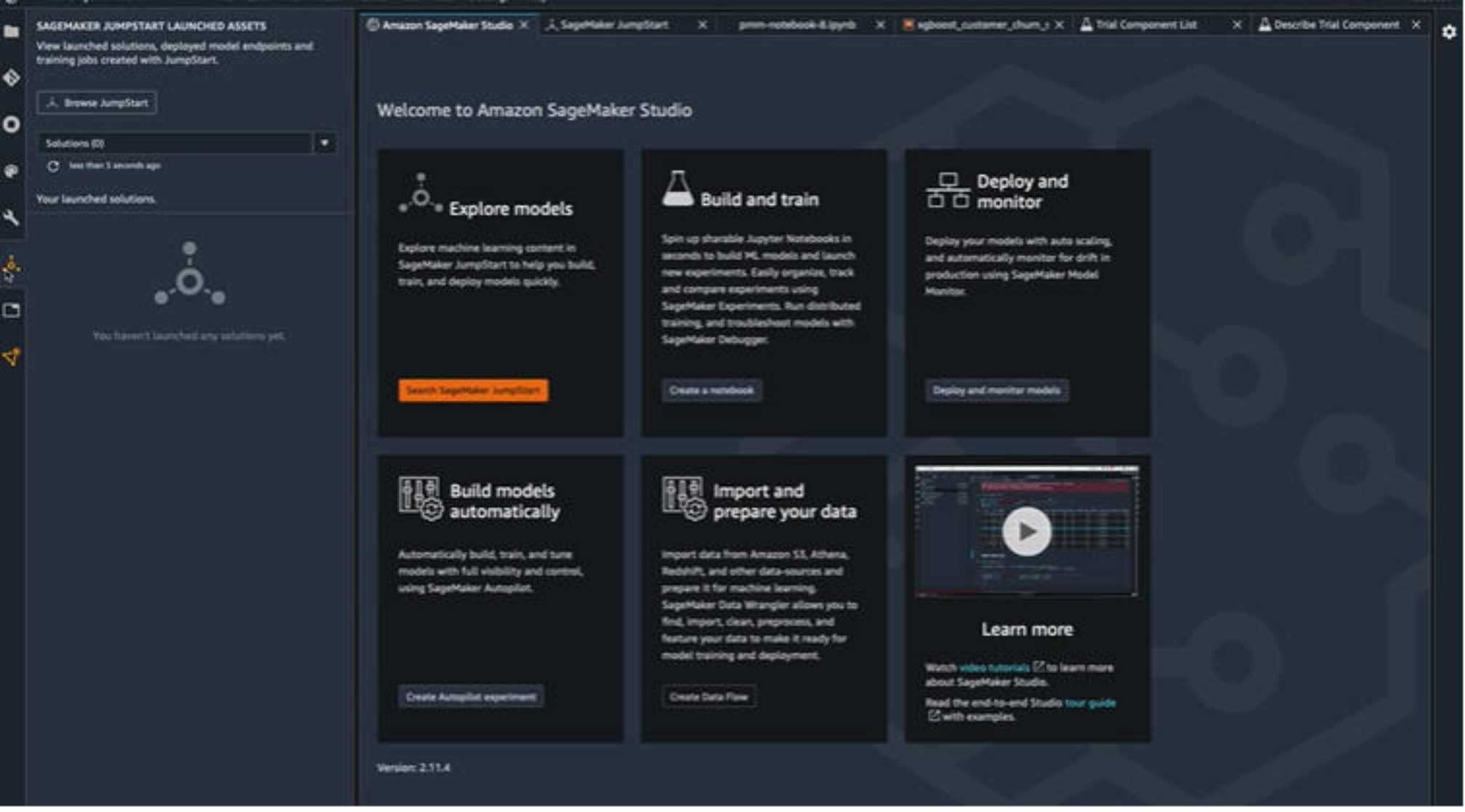
Select items to perform actions on them.

Upload | New

0 / linear_learner_mnist_2020-12-20

Name	Last Modified	File size
..	seconds ago	
linear_learner_mnist_mltd.ipynb	Running seconds ago	302 kB
linear_learner_mnist_with_file_system_data_source.ipynb	2 days ago	6.46 kB
mnist.pkl.gz	25 minutes ago	16.2 MB


Amazon SageMaker Studio
SageMaker JumpStart
pmm-notebook-8.ipynb
xgboost_customer_churn_s
Trial Component List
Describe Trial Component
/ introduction_to_amazon_algorithms / linear_learner_mnist /
Name
Last Modified
linear_learner_mnist_with_file_sy...
19 days ago
linear_learner_mnist.ipynb
20 hours ago
mnist.pkl.gz
20 hours ago
Welcome to Amazon SageMaker Studio
Explore models
Explore machine learning content in SageMaker JumpStart to help you build, train, and deploy models quickly.
Search SageMaker JumpStart
Build and train
Spin up sharable Jupyter Notebooks in seconds to build ML models and launch new experiments. Easily organize, track and compare experiments using SageMaker Experiments. Run distributed training, and troubleshoot models with SageMaker Debugger.
Create a notebook
Deploy and monitor
Deploy your models with auto scaling, and automatically monitor for drift in production using SageMaker Model Monitor.
Deploy and monitor models
Build models automatically
Automatically build, train, and tune models with full visibility and control, using SageMaker Autopilot.
Create Autopilot experiment
Import and prepare your data
Import data from Amazon S3, Athena, Redshift, and other data-sources and prepare it for machine learning. SageMaker Data Wrangler allows you to find, import, clean, preprocess, and feature your data to make it ready for model training and deployment.
Create Data Flow
Learn more
Watch video tutorials to learn more about SageMaker Studio.
Read the end-to-end Studio tour guide with examples.
Version: 2.11.4


SAGEMAKER JUMPSTART LAUNCHED ASSETS
View launched solutions, deployed model endpoints and training jobs created with JumpStart.
Browse JumpStart
Solutions (0)
less than 1 seconds ago
Your launched solutions
You haven't launched any solutions yet.
Amazon SageMaker Studio
SageMaker JumpStart
pmm-notebook-8.ipynb
xgboost_customer_churn_s
Trial Component List
Describe Trial Component
Welcome to Amazon SageMaker Studio
Explore models
Explore machine learning content in SageMaker JumpStart to help you build, train, and deploy models quickly.
Search SageMaker JumpStart
Build and train
Spin up sharable Jupyter Notebooks in seconds to build ML models and launch new experiments. Easily organize, track and compare experiments using SageMaker Experiments. Run distributed training, and troubleshoot models with SageMaker Debugger.
Create a notebook
Deploy and monitor
Deploy your models with auto scaling, and automatically monitor for drift in production using SageMaker Model Monitor.
Deploy and monitor models
Build models automatically
Automatically build, train, and tune models with full visibility and control, using SageMaker Autopilot.
Create Autopilot experiment
Import and prepare your data
Import data from Amazon S3, Athena, Redshift, and other data-sources and prepare it for machine learning. SageMaker Data Wrangler allows you to find, import, clean, preprocess, and feature your data to make it ready for model training and deployment.
Create Data Flow
Learn more
Watch video tutorials to learn more about SageMaker Studio.
Read the end-to-end Studio tour guide with examples.
Version: 2.11.4

SAGEMAKER JUMPSTART LAUNCHED ASSETS

View launched solutions, deployed model endpoints and training jobs created with JumpStart.

Browse JumpStart

Solutions (0)

less than 10 seconds ago

Your launched solutions.

You haven't launched any solutions yet.

Amazon SageMaker Studio | SageMaker JumpStart | pmm-notebook-8.ipynb | xgboost_customer_churn_s | Trial Component List | Describe Trial Component

SageMaker JumpStart: Discover, train, and deploy

One-click solutions, pretrained models, and example notebooks

Search

Solutions

Launch end-to-end machine learning solutions that tie SageMaker to other AWS services with one click.

View all (14)

Text models

Deploy and fine-tune pretrained transformers for various natural language processing use cases.

View all (21)

Vision models

Deploy and fine-tune pretrained models for image classification and object detection with one click.

View all (124)

SageMaker algorithms

Train and deploy SageMaker built-in Algorithms for various problem types with these example notebooks.

View all (14)

Example notebooks

Run example notebooks that use SageMaker features like spot instance training and experiments over a large variety of model types and use cases.

View all (5)

SAGEMAKER JUMPSTART LAUNCHED ASSETS

View launched solutions, deployed model endpoints and training jobs created with JumpStart.

Browse JumpStart

Solutions (0)

less than 20 seconds ago

Your launched solutions

You haven't launched any solutions yet.

Amazon SageMaker Studio | SageMaker JumpStart | pmm-notebook-8.ipynb | xgboost_customer_churn_s | Trial Component List | Describe Trial Component

Search

Solutions

Launch end-to-end machine learning solutions that tie SageMaker to other AWS services with one click.

View all (14)

Text models

Deploy and fine-tune pretrained transformers for various natural language processing use cases.

View all (21)

Vision models

Deploy and fine-tune pretrained models for image classification and object detection with one click.

View all (124)

SageMaker algorithms

Train and deploy SageMaker built-in Algorithms for various [illegible] types with these example notebooks.

View all (14)

Introduction to Linear Learner

Algorithm

This notebook demonstrates the use of Amazon SageMaker's implementation of the Linear Learner algorithm to train and host a binary classification model.

Regression With XGBoost

Algorithm

This notebook demonstrates the use of Amazon SageMaker's implementation of the XGBoost algorithm to train and host a regression model.

Intro to the IP Insights Algorithm

Algorithm

This notebook demonstrates the use of SageMaker's IP Insights algorithm to train a model for unsupervised anomaly detection of suspicious IP addresses.

Intro to SageMaker Random Cut Forests

Algorithm

This notebook demonstrates the use of Amazon SageMaker's Random Cut Forest algorithm to train and host a model for anomaly detection.

SAGEMAKER JUMPSTART LAUNCHED ASSETS

View launched solutions, deployed model endpoints and training jobs created with JumpStart.

Browse JumpStart

Solutions (0)

less than 20 seconds ago

Your launched solutions.

You haven't launched any solutions yet.

Amazon SageMaker Studio | SageMaker JumpStart | pmm-notebook-8.ipynb | xgboost_customer_churn_s | Trial Component List | Describe Trial Component

Solutions

View all (14)

Launch end-to-end machine learning solutions that tie SageMaker to other AWS services with one click.

Text models

View all (21)

Deploy and fine-tune pretrained transformers for various natural language processing use cases.

Vision models

View all (124)

Deploy and fine-tune pretrained models for image classification and object detection with one click.

SageMaker algorithms

View all (14)

Train and deploy SageMaker built-in Algorithms for various problem types with these example notebooks.

Introduction to Linear Learner

Algorithm

This notebook demonstrates the use of Amazon SageMaker's implementation of the Linear Learner algorithm to train and host a binary classification model.

Regression With XGBoost

Algorithm

This notebook demonstrates the use of Amazon SageMaker's implementation of the XGBoost algorithm to train and host a regression model.

Intro to the IP Insights Algorithm

Algorithm

This notebook demonstrates the use of SageMaker's IP Insights algorithm to train a model for unsupervised anomaly detection of suspicious IP addresses.

Intro to SageMaker Random Cut Forests

Algorithm

This notebook demonstrates the use of Amazon SageMaker's Random Cut Forest algorithm to train and host a model for anomaly detection.

SAGEMAKER JUMPSTART LAUNCHED ASSETS

View launched solutions, deployed model endpoints and training jobs created with JumpStart.

Browse JumpStart

Solutions (0)

less than 20 seconds ago

Your launched solutions.

You haven't launched any solutions yet.

Amazon SageMaker S | SageMaker JumpStart | pmm-notebook-8.ipy | pmm-notebook-8.ipy | xgboost_customer_ch | Trial Component List | Describe Trial Compo

Read-only mode

Import notebook

An Introduction to Linear Learner with MNIST

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (x , y). x is a high dimensional vector and y is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label y .

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

SAGEMAKER JUMPSTART LAUNCHED ASSETS

View launched solutions, deployed model endpoints and training jobs created with JumpStart.

Browse JumpStart

Solutions (0)

half a minute ago

Your launched solutions.

You haven't launched any solutions yet.

Amazon SageMaker S | SageMaker JumpStart | pmm-notebook-8.ipy | pmm-notebook-8.ipy | xgboost_customer_ch | Trial Component List | Describe Trial Compo

Read-only mode

Import notebook

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

- The S3 buckets and prefixes that you want to use for training and model data and where original data is located. These should be within the same region as the Notebook Instance, training, and hosting.
- The IAM role arn used to give training and hosting access to your data. See the documentation for how to create these. Note, if more than one role is required for notebook instances, training, and/or hosting, please replace the boto regexp with a the appropriate full IAM role arn string(s).

[ ]:
```
import re
import boto3
import sagemaker
from sagemaker import get_execution_role

sess = sagemaker.Session()

region = boto3.Session().region_name

# S3 bucket where the original mnist data is downloaded and stored.
downloaded_data_bucket = f"jumpstart-cache-prod-{region}"
downloaded_data_prefix = "1p-notebooks-datasets/mnist"

# S3 bucket for saving code and model artifacts.
# Feel free to specify a different bucket and prefix
bucket = sess.default_bucket()
prefix = "sagemaker/DEMO-linear-mnist"

# Define IAM role
role = get_execution_role()
```

Data ingestion

Next, we read the MNIST dataset [1] from an existing repository into memory, for preprocessing prior to training. It was downloaded from this link and stored on the `downloaded_data_bucket`. Processing could be done *in situ* by Amazon Athena, Apache Spark in Amazon EMR, Amazon Redshift, etc., assuming the dataset is present in the appropriate location. Then, the next step would be to transfer the data to S3 for use in training. For small datasets, such as this one, reading into memory isn't onerous, though it would be for larger datasets.

> [1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278-2324, November 1998.

[ ]:
```
%%time
import pickle, gzip, numpy, urllib.request, json

# Load the dataset
s3 = boto3.client("s3")
s3.download_file(downloaded_data_bucket, f"{downloaded_data_prefix}/mnist.pkl.gz", "mnist.pkl.gz")
```

SAGEMAKER JUMPSTART LAUNCHED ASSETS

View launched solutions, deployed model endpoints and training jobs created with JumpStart.

Browse JumpStart

Solutions (0)

half a minute ago

Your launched solutions.

You haven't launched any solutions yet.

Amazon SageMaker S | SageMaker JumpStart | pmm-notebook-8.ipy | pmm-notebook-8.ipy | xgboost_customer_ch | Trial Component List | Describe Trial Compo

Read-only mode

Import notebook

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (x , y). x is a high dimensional vector and y is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label y .

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

/ introduction_to_amazon_algorithms / linear_learner_mnist /

Name	Last Modified
linear_learner_mnist_with_file_sy...	19 days ago
linear_learner_mnist.ipynb	20 hours ago
mnist.pkl.gz	20 hours ago

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (x , y). x is a high dimensional vector and y is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label y .

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

Amazon SageMak | SageMaker JumpS | pmm-notebook-8 | pmm-notebook-8 | pmm-notebook-8 | xgboost_custome | Trial Component I | Describe Trial Con

Markdown | 2 vCPU + 4 GiB | Python 3 (Data Science) | Kernel: CPU: 0.00% MEM: 3.44% | Share

/

Name	Last Modified
amazon-sagemaker-clarify	21 hours ago
amazon-sagemaker-examples	19 days ago
confusion-matrix.ipynb	20 hours ago
Lecture-4-Matplotlib.ipynb	an hour ago
numeric-data-engineering.ipynb	20 hours ago
pmm-notebook-8.ipynb	seconds ago

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (x , y). x is a high dimensional vector and y is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label y .

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

Name | Last Modified

- amazon-sagemaker-clarify — 21 hours ago
- amazon-sagemaker-examples — 19 days ago
- confusion-matrix.ipynb — 20 hours ago
- Lecture-4-Matplotlib.ipynb — an hour ago
- numeric-data-engineering.ipynb — 20 hours ago
- pmm-notebook-8.ipynb — seconds ago

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

- The S3 buckets and prefixes that you want to use for training and model data and where original data is located. These should be within the same region as the Notebook Instance, training, and hosting.
- The IAM role arn used to give training and hosting access to your data. See the documentation for how to create these. Note, if more than one role is required for notebook instances, training, and/or hosting, please replace the boto regexp with a the appropriate full IAM role arn string(s).

[ ]:
```
import re
import boto3
import sagemaker
from sagemaker import get_execution_role

sess = sagemaker.Session()

region = boto3.Session().region_name

# S3 bucket where the original mnist data is downloaded and stored.
downloaded_data_bucket = f"jumpstart-cache-prod-{region}"
downloaded_data_prefix = "1p-notebooks-datasets/mnist"

# S3 bucket for saving code and model artifacts.
# Feel free to specify a different bucket and prefix
bucket = sess.default_bucket()
prefix = "sagemaker/DEMO-linear-mnist"

# Define IAM role
role = get_execution_role()
```

Data ingestion

Next, we read the MNIST dataset [1] from an existing repository into memory, for preprocessing prior to training. It was downloaded from this link and stored on the `downloaded_data_bucket`. Processing could be done in situ by Amazon Athena, Apache Spark in Amazon EMR, Amazon Redshift, etc., assuming the dataset is present in the appropriate location. Then, the next step would be to transfer the data to S3 for use in training. For small datasets, such as this one, reading into memory isn't onerous, though it would be for larger datasets.

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11): 2278-2324, November 1998.

-4:12 2x

Amazon SageMak | SageMaker Jumpt | pmm-notebook-8 | pmm-notebook-8 | pmm-notebook-8 | xgboost_custome | Trial Component I | Describe Trial Con

Markdown | 2 vCPU + 4 GiB | Python 3 (Data Science) | Kernel: CPU: 0.00% MEM: 3.44% | Share

/

Name	Last Modified
amazon-sagemaker-clarify	21 hours ago
amazon-sagemaker-examples	19 days ago
confusion-matrix.ipynb	20 hours ago
Lecture-4-Matplotlib.ipynb	an hour ago
numeric-data-engineering.ipynb	20 hours ago
pmm-notebook-8.ipynb	seconds ago

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (x , y). x is a high dimensional vector and y is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label y .

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit Is a 0

Select instance

Running notebook

pmm-notebook-8.ipynb

Current instance type

2 vCPU + 4 GiB **ml.t3.medium** Fast Launch

If you change your instance, existing settings for this notebook will be lost, and installed packages will not be carried over.

Instances 4 of 29

Fast launch only

Instance Type	Instance Category	vCPU	GPU	Memory	Fast Launch
ml.t3.medium	General purpose	2	0	4 GiB	✓
ml.g4dn.xlarge	Accelerated computing	4	1	16 GiB	✓
ml.m5.large	General purpose	2	0	8 GiB	✓
ml.c5.large	Compute optimized	2	0	4 GiB	✓

Cancel

Save and continue

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit Is a 0

Select instance

Running notebook

pmm-notebook-8.ipynb

Current instance type

2 vCPU + 4 GiB **ml.t3.medium** Fast Launch

If you change your instance, existing settings for this notebook will be lost, and installed packages will not be carried over.

Instances 4 of 29

Fast launch only

Instance Type	Instance Category	vCPU	GPU	Memory	Fast Launch
ml.t3.medium	General purpose	2	0	4 GiB	✓
ml.g4dn.xlarge	Accelerated computing	4	1	16 GiB	✓
ml.m5.large	General purpose	2	0	8 GiB	✓
ml.c5.large	Compute optimized	2	0	4 GiB	✓

Cancel

Save and continue

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit Is a 0

Select instance

Running notebook

pmm-notebook-8.ipynb

Current instance type

2 vCPU + 4 GiB **ml.t3.medium** Fast Launch

If you change your instance, existing settings for this notebook will be lost, and installed packages will not be carried over.

Instances 29 of 29

Fast launch only

Instance Type	Instance Category	vCPU	GPU	Memory	Fast Launch
ml.t3.large	General purpose	2	0	8 GiB	
ml.t3.xlarge	General purpose	4	0	16 GiB	
ml.t3.2xlarge	General purpose	8	0	32 GiB	
ml.m5.large	General purpose	2	0	8 GiB	✓
ml.m5.xlarge	General purpose	4	0	16 GiB	
ml.m5.2xlarge	General purpose	8	0	32 GiB	
ml.m5.4xlarge	General purpose	16	0	64 GiB	
ml.m5.8xlarge	General purpose	32	0	128 GiB	

Cancel

Save and continue

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit Is a 0

Select instance

Running notebook

pmm-notebook-8.ipynb

Current instance type

2 vCPU + 4 GiB **ml.t3.medium** Fast Launch

If you change your instance, existing settings for this notebook will be lost, and installed packages will not be carried over.

Instances 4 of 29

Fast launch only

Instance Type	Instance Category	vCPU	GPU	Memory	Fast Launch
ml.t3.medium	General purpose	2	0	4 GiB	✓
ml.g4dn.xlarge	Accelerated computing	4	1	16 GiB	✓
ml.m5.large	General purpose	2	0	8 GiB	✓
ml.c5.large	Compute optimized	2	0	4 GiB	✓

Cancel Save and continue

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

Amazon SageMak | SageMaker JumpS | pmm-notebook-8 | pmm-notebook-8 | pmm-notebook-8 | xgboost_custome | Trial Component I | Describe Trial Con

Markdown | 2 vCPU + 4 GiB | Python 3 (Data Science) | Kernel: CPU: 0.00% MEM: 3.44% | Share

/

Name	Last Modified
amazon-sagemaker-clarify	21 hours ago
amazon-sagemaker-examples	19 days ago
confusion-matrix.ipynb	20 hours ago
Lecture-4-Matplotlib.ipynb	an hour ago
numeric-data-engineering.ipynb	20 hours ago
pmm-notebook-8.ipynb	a minute ago

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (x , y). x is a high dimensional vector and y is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label y .

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

Amazon SageMak | SageMaker Jumpt | pmm-notebook-8 | pmm-notebook-8 | pmm-notebook-8 | xgboost_custome | Trial Component I | Describe Trial Con

Markdown | 2 vCPU + 4 GiB | Python 3 (Data Science) | Kernel: CPU: 0.00% MEM: 3.44% | Share

/

Name	Last Modified
amazon-sagemaker-clarify	21 hours ago
amazon-sagemaker-examples	19 days ago
confusion-matrix.ipynb	20 hours ago
Lecture-4-Matplotlib.ipynb	an hour ago
numeric-data-engineering.ipynb	20 hours ago
pmm-notebook-8.ipynb	a minute ago

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (x , y). x is a high dimensional vector and y is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label y .

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the linear model
5. Validate the model for use

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero through nine. We'll use the linear learner algorithm on a single image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we use here supports both regression and classification. For classification, we must provide a label for each training input, and the algorithm expects two different labels. In general, they should be 0 and 1 for binary classification, and 0, 1, ... num_classes - 1 for multi-class classification. For regression, the label must be a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y.) The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label, y.

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

Create shareable snapshot

Sharing notebook: pmm-notebook-8.ipynb

☐ Include Git repo information

No Git repo was found

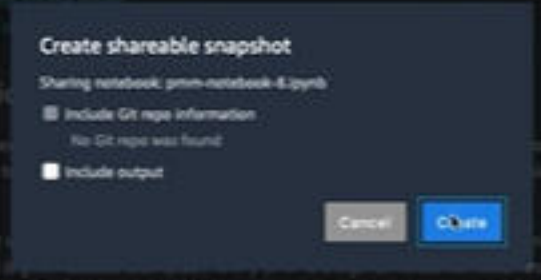
☐ Include output

Cancel | Create

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the



Introducti

Welcome to our e ... today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero t ... e image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that ... ning algorithms used for solving either classification or regression problems. As input, the model ... ed. y is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label, y .

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit Is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the

Create shareable snapshot

Sharing notebook: pmm-notebook-8.ipynb

☐ Include Git repo information

No Git repo was found

☐ Include output

Cancel | Create

Introducti

Welcome to our e ... day, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero t ... e image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that ... ing algorithms used for solving either classification or regression problems. As input, the model ... ful. y is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y.). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label y.

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

/

Name	Last Modified
amazon-sagemaker-clarify	21 hours ago
amazon-sagemaker-examples	19 days ago
confusion-matrix.ipynb	20 hours ago
Lecture-4-Matplotlib.ipynb	an hour ago
numeric-data-engineering.ipynb	20 hours ago
pmm-notebook-8.ipynb	seconds ago

Amazon SageMak | SageMaker Jumpt | pmm-notebook-8 | pmm-notebook-8 | pmm-notebook-8 | xgboost_custome | Trial Component I | Describe Trial Con

Markdown | 2 vCPU + 4 GiB | Python 3 (Data Science) | Kernel: CPU: 0.00% MEM: 3.44% | Share

An Introduction to Linear Learner with MNIST

Making a Binary Prediction of Whether a Handwritten Digit is a 0

1. Introduction
2. Prerequisites and Preprocessing
   A. Permissions and environment variables
   B. Data ingestion
   C. Data inspection
   D. Data conversion
3. Training the linear model
4. Set up hosting for the model
5. Validate the model for use

Introduction

Welcome to our example introducing Amazon SageMaker's Linear Learner Algorithm! Today, we're analyzing the MNIST dataset which consists of images of handwritten digits, from zero to nine. We'll use the individual pixel values from each 28 x 28 grayscale image to predict a yes or no label of whether the digit is a 0 or some other digit (1, 2, 3, ... 9).

The method that we'll use is a linear binary classifier. Linear models are supervised learning algorithms used for solving either classification or regression problems. As input, the model is given labeled examples (x , y). x is a high dimensional vector and y is a numeric label. Since we are doing binary classification, the algorithm expects the label to be either 0 or 1 (but Amazon SageMaker Linear Learner also supports regression on continuous values of y). The algorithm learns a linear function, or linear threshold function for classification, mapping the vector x to an approximation of the label y .

Amazon SageMaker's Linear Learner algorithm extends upon typical linear models by training many models in parallel, in a computationally efficient manner. Each model has a different set of hyperparameters, and then the algorithm finds the set that optimizes a specific criteria. This can provide substantially more accurate models than typical linear algorithms at the same, or lower, cost.

To get started, we need to set up the environment with a few prerequisite steps, for permissions, configurations, and so on.

Prequisites and Preprocessing

Permissions and environment variables

This notebook was created and tested on an ml.m4.xlarge notebook instance.

Let's start by specifying:

Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints
Batch transform jobs
Edge Inference
Get started
Edge packaging jobs
Edge device fleets
Edge devices
Augmented AI
Human review workflows

Endpoints

Update endpoint | Actions | Create endpoint

Search endpoints

1

Name	ARN	Creation time	Status	Last updated
There are currently no resources.				

Amazon SageMaker Studio
Dashboard
Search
Images
Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints

Endpoints

Update endpoint | Actions | Create endpoint

Search endpoints

1

Name	ARN	Creation time	Status	Last updated

There are currently no resources.

Amazon SageMaker Studio
Dashboard
Search
Images
Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints

Notebook instances

Actions | Create notebook instance

Search notebook instances

1

Name	Instance	Creation time	Status	Actions

Amazon SageMaker Studio
Dashboard
Search
Images
Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints

Notebook instances

Actions
Create notebook instance

Search notebook instances

1

Name	Instance	Creation time	Status	Actions

Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corbuno-liengard @ corbuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker ×

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints

Amazon SageMaker > Notebook instances

Notebook instances

Actions ▾ | Create notebook instance

Search notebook instances

1

Name	Instance	Creation time	Status	Actions
mltd-2	ml.t3.medium	Dec 20, 2020 11:31 UTC	InService	Open Jupyter | Open JupyterLab
mltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	InService	Open Jupyter | Open JupyterLab
LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
mltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
mltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
mltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
mltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start

Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corbuno-liengard @ corbuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints

Amazon SageMaker > Notebook instances

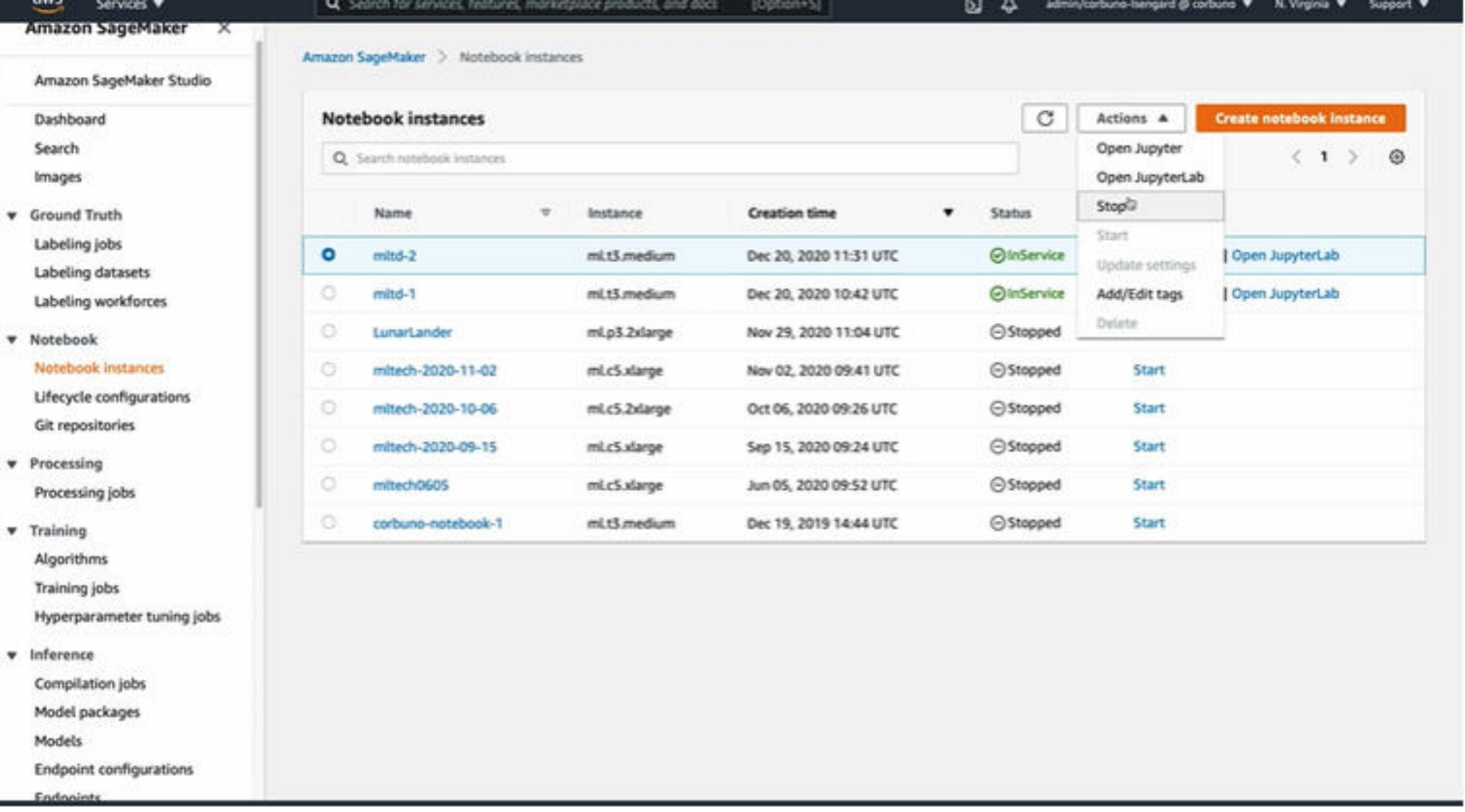
Notebook instances

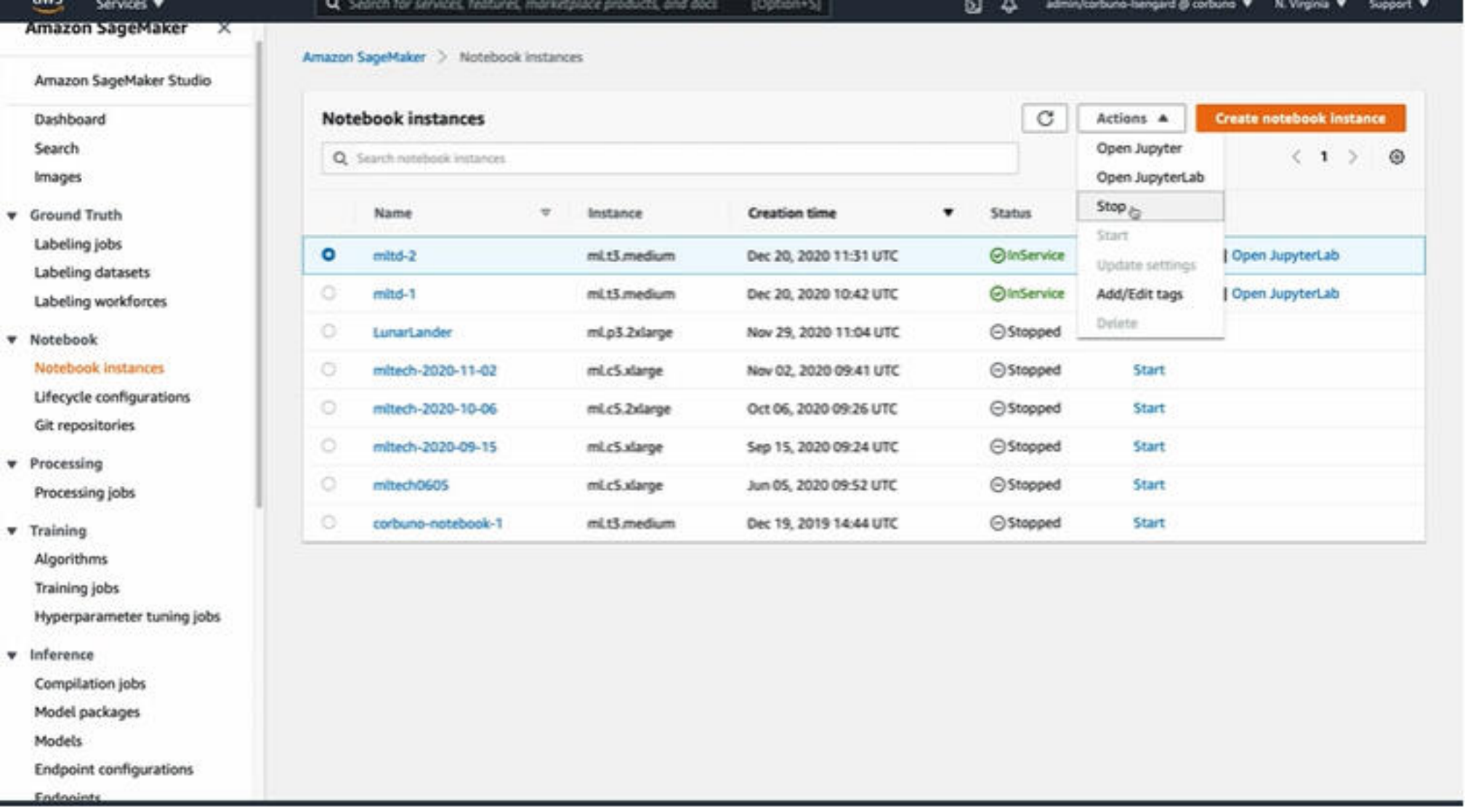
Actions ▾ | Create notebook instance

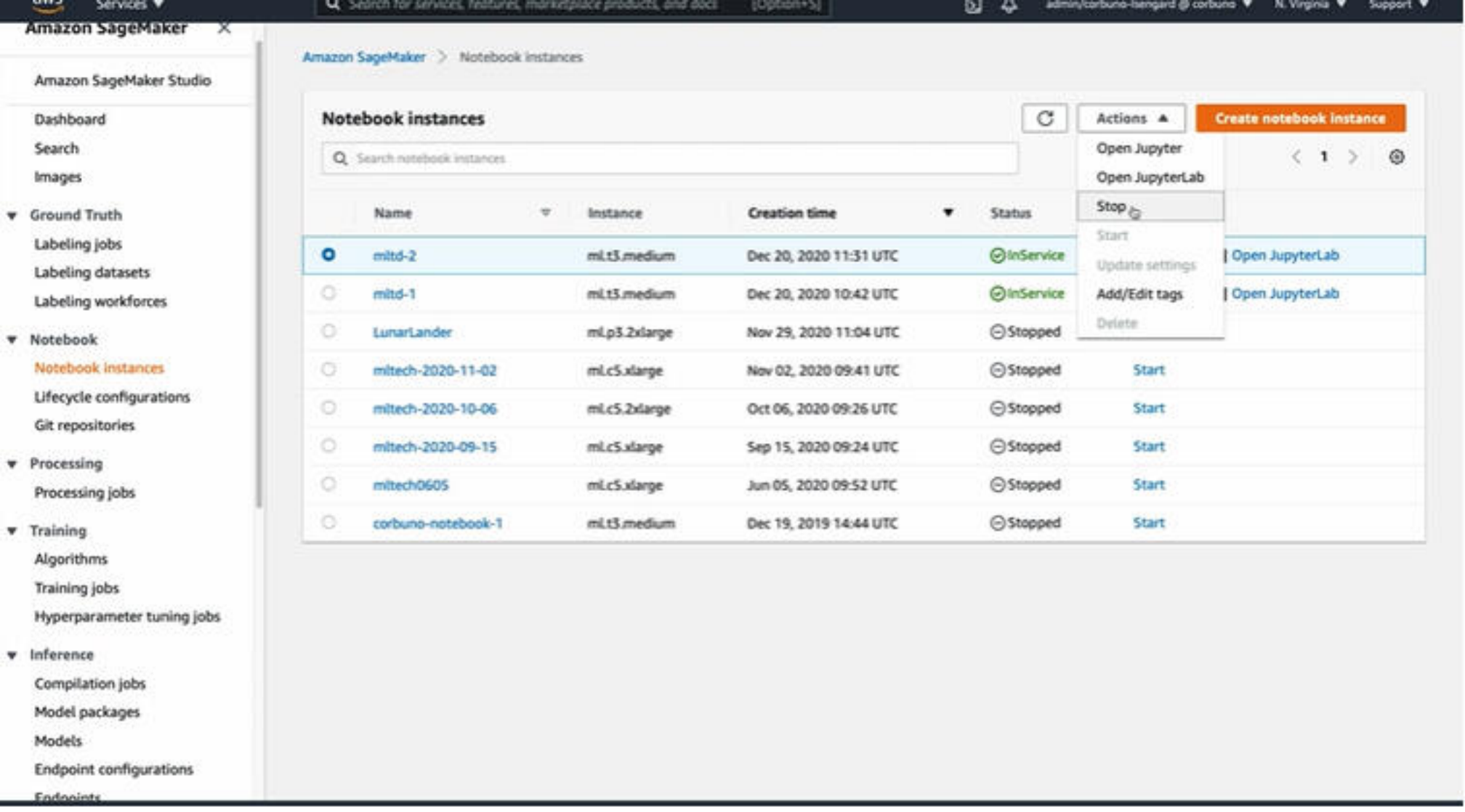
Search notebook instances

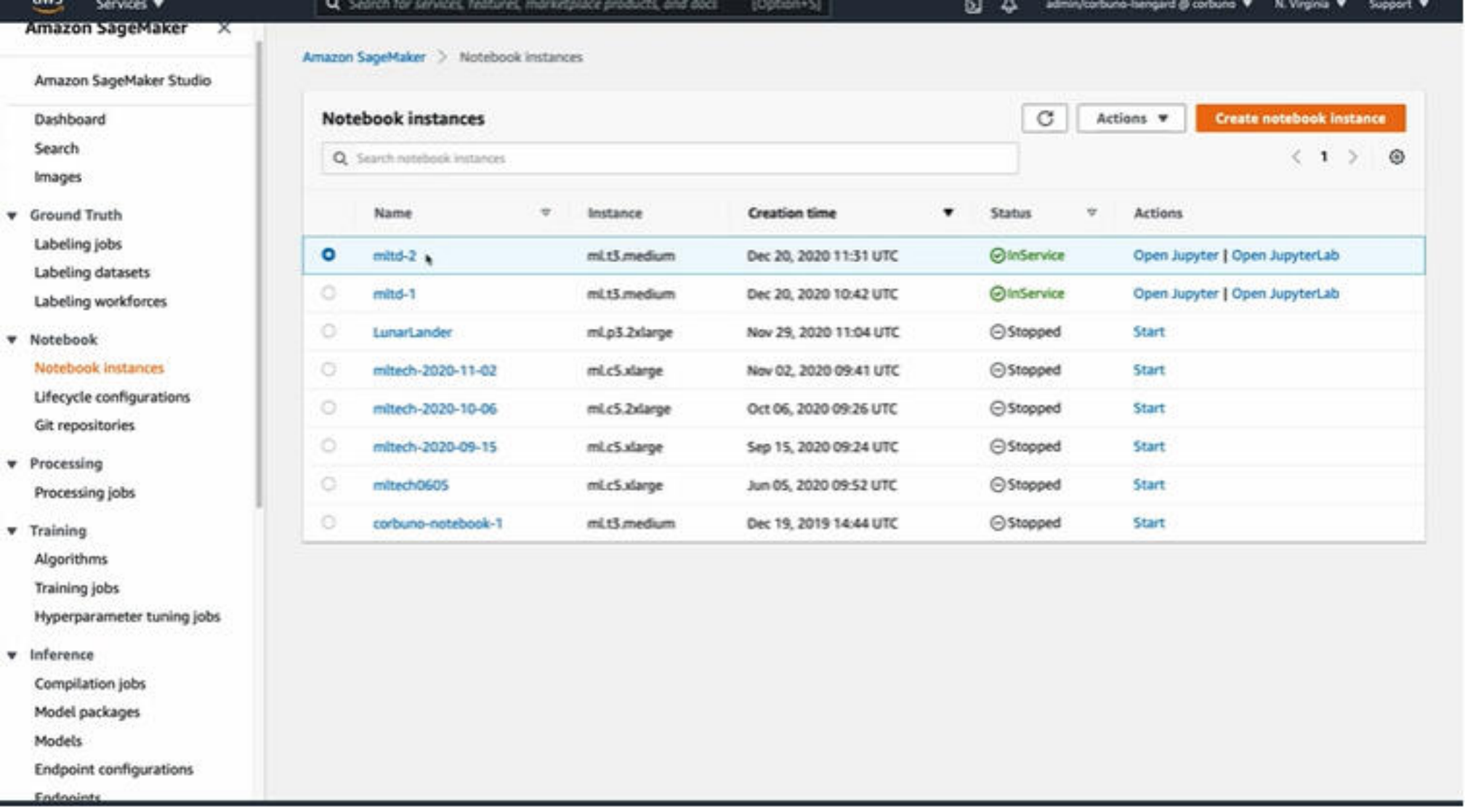
1

Name	Instance	Creation time	Status	Actions
mltd-2	ml.t3.medium	Dec 20, 2020 11:31 UTC	InService	Open Jupyter | Open JupyterLab
mltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	InService	Open Jupyter | Open JupyterLab
LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
mltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
mltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
mltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
mltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start


Services
Search for services, features, marketplace products, and docs [Option+S]
admin/corbuno-liengard @ corbuno
N. Virginia
Support
Amazon SageMaker
Amazon SageMaker Studio
Dashboard
Search
Images
Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints
Amazon SageMaker > Notebook instances
Notebook instances
Actions
Create notebook instance
Search notebook instances
Open Jupyter
Open JupyterLab
Stop
Start
Update settings
Add/Edit tags
Delete
Name
Instance
Creation time
Status
mltd-2 ml.t3.medium Dec 20, 2020 11:31 UTC InService Open JupyterLab
mltd-1 ml.t3.medium Dec 20, 2020 10:42 UTC InService Open JupyterLab
LunarLander ml.p3.2xlarge Nov 29, 2020 11:04 UTC Stopped
mltech-2020-11-02 ml.c5.xlarge Nov 02, 2020 09:41 UTC Stopped Start
mltech-2020-10-06 ml.c5.2xlarge Oct 06, 2020 09:26 UTC Stopped Start
mltech-2020-09-15 ml.c5.xlarge Sep 15, 2020 09:24 UTC Stopped Start
mltech0605 ml.c5.xlarge Jun 05, 2020 09:52 UTC Stopped Start
corbuno-notebook-1 ml.t3.medium Dec 19, 2019 14:44 UTC Stopped Start


Services
Search for services, features, marketplace products, and docs [Option+S]
admin/corbuno-liengard @ corbuno
N. Virginia
Support
Amazon SageMaker
Amazon SageMaker Studio
Dashboard
Search
Images
Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints
Amazon SageMaker > Notebook instances
Notebook instances
Actions
Create notebook instance
Search notebook instances
Open Jupyter
Open JupyterLab
Stop
Start
Update settings
Add/Edit tags
Delete
Name
Instance
Creation time
Status
mltd-2 ml.t3.medium Dec 20, 2020 11:31 UTC InService Open JupyterLab
mltd-1 ml.t3.medium Dec 20, 2020 10:42 UTC InService Open JupyterLab
LunarLander ml.p3.2xlarge Nov 29, 2020 11:04 UTC Stopped
mltech-2020-11-02 ml.c5.xlarge Nov 02, 2020 09:41 UTC Stopped Start
mltech-2020-10-06 ml.c5.2xlarge Oct 06, 2020 09:26 UTC Stopped Start
mltech-2020-09-15 ml.c5.xlarge Sep 15, 2020 09:24 UTC Stopped Start
mltech0605 ml.c5.xlarge Jun 05, 2020 09:52 UTC Stopped Start
corbuno-notebook-1 ml.t3.medium Dec 19, 2019 14:44 UTC Stopped Start


Services
Search for services, features, marketplace products, and docs [Option+S]
admin/corbuno-liengard @ corbuno
N. Virginia
Support
Amazon SageMaker
Amazon SageMaker Studio
Dashboard
Search
Images
Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints
Amazon SageMaker > Notebook instances
Notebook instances
Actions
Create notebook instance
Search notebook instances
Open Jupyter
Open JupyterLab
Stop
Start
Update settings
Add/Edit tags
Delete
Name
Instance
Creation time
Status
mltd-2 ml.t3.medium Dec 20, 2020 11:31 UTC InService Open JupyterLab
mltd-1 ml.t3.medium Dec 20, 2020 10:42 UTC InService Open JupyterLab
LunarLander ml.p3.2xlarge Nov 29, 2020 11:04 UTC Stopped
mltech-2020-11-02 ml.c5.xlarge Nov 02, 2020 09:41 UTC Stopped Start
mltech-2020-10-06 ml.c5.2xlarge Oct 06, 2020 09:26 UTC Stopped Start
mltech-2020-09-15 ml.c5.xlarge Sep 15, 2020 09:24 UTC Stopped Start
mltech0605 ml.c5.xlarge Jun 05, 2020 09:52 UTC Stopped Start
corbuno-notebook-1 ml.t3.medium Dec 19, 2019 14:44 UTC Stopped Start


Services
Search for services, features, marketplace products, and docs [Option+S]
admin/corbuno-liengard @ corbuno
N. Virginia
Support
Amazon SageMaker
Amazon SageMaker Studio
Dashboard
Search
Images
Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints
Amazon SageMaker > Notebook instances
Notebook instances
Actions
Create notebook instance
Search notebook instances
1
Name
Instance
Creation time
Status
Actions
mltd-2 ml.t3.medium Dec 20, 2020 11:31 UTC InService Open Jupyter | Open JupyterLab
mltd-1 ml.t3.medium Dec 20, 2020 10:42 UTC InService Open Jupyter | Open JupyterLab
LunarLander ml.p3.2xlarge Nov 29, 2020 11:04 UTC Stopped Start
mltech-2020-11-02 ml.c5.xlarge Nov 02, 2020 09:41 UTC Stopped Start
mltech-2020-10-06 ml.c5.2xlarge Oct 06, 2020 09:26 UTC Stopped Start
mltech-2020-09-15 ml.c5.xlarge Sep 15, 2020 09:24 UTC Stopped Start
mltech0605 ml.c5.xlarge Jun 05, 2020 09:52 UTC Stopped Start
corbuno-notebook-1 ml.t3.medium Dec 19, 2019 14:44 UTC Stopped Start

Notebook instances

Actions ▾ | Create notebook instance

Search notebook instances

1

	Name	Instance	Creation time	Status	Actions
◉	mltd-2	ml.t3.medium	Dec 20, 2020 11:31 UTC	InService	Open Jupyter | Open JupyterLab
○	mltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	InService	Open Jupyter | Open JupyterLab
○	LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
○	mltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
○	mltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
○	mltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
○	mltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
○	corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start

mltd-2

Delete | Stop | Open Jupyter | Open JupyterLab

Notebook instance settings

Edit

Name mltd-2	Status InService	Notebook instance type ml.t3.medium
ARN arn:aws:sagemaker:us-east-1:649088000912:notebook-instance/mltd-2	Creation time Dec 20, 2020 11:31 UTC	Elastic Inference -
Lifecycle configuration arn:aws:sagemaker:us-east-1:649088000912:notebook-instance-lifecycle-config/auto-stop-idle View logs	Last updated Dec 20, 2020 11:34 UTC	Volume Size 5GB EBS

Git repositories

Name	Repository URL	Type
There are currently no resources.		

Permissions and encryption

mltd-2

Delete | Stop | Open Jupyter | Open JupyterLab

Notebook instance settings

Edit

Name mltd-2	Status InService	Notebook instance type ml.t3.medium
ARN arn:aws:sagemaker:us-east-1:649088000912:notebook-instance/mltd-2	Creation time Dec 20, 2020 11:31 UTC	Elastic Inference -
Lifecycle configuration arn:aws:sagemaker:us-east-1:649088000912:notebook-instance-lifecycle-config/auto-stop-idle View logs	Last updated Dec 20, 2020 11:34 UTC	Volume Size 5GB EBS

Git repositories

Name	Repository URL	Type
There are currently no resources.		

Permissions and encryption

Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corbuno-liengard @ corbuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints

Amazon SageMaker > Notebook instances

Notebook instances

Actions ▾ | Create notebook instance

Search notebook instances

1

Name	Instance	Creation time	Status	Actions
miltd-2	ml.t3.medium	Dec 20, 2020 11:31 UTC	InService	Open Jupyter | Open JupyterLab
miltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	InService	Open Jupyter | Open JupyterLab
LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
miltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
miltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
miltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
miltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start

Amazon SageMaker

Amazon SageMaker Studio

Dashboard
Search
Images

Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces

Notebook
Notebook instances
Lifecycle configurations
Git repositories

Processing
Processing jobs

Training
Algorithms
Training jobs
Hyperparameter tuning jobs

Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints

Amazon SageMaker > Notebook instancesLunarLander

LunarLander

Delete | Start | Open Jupyter | Open JupyterLab

Notebook instance settings

Edit

Name LunarLander	Status Stopped	Notebook instance type ml.p3.2xlarge
ARN arn:aws:sagemaker:us-east-1:649088000912:notebook-instance/lunarlander	Creation time Nov 29, 2020 11:04 UTC	Elastic Inference -
Lifecycle configuration arn:aws:sagemaker:us-east-1:649088000912:notebook-instance-lifecycle-config/lifecycleconfig View logs	Last updated Nov 30, 2020 11:46 UTC	Volume Size 5GB EBS

Git repositories

Name	Repository URL	Type
There are currently no resources.		

Permissions and encryption

Amazon SageMaker Studio
Dashboard
Search
Images
Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints

Edit notebook instance

Notebook instance settings

Notebook instance name

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

Notebook instance type

Elastic Inference Learn more

▼ Additional configuration

Lifecycle configuration - *optional*

Customize your notebook environment with default scripts and plugins.

No configuration

Volume size in GB - *optional*

Enter the volume size of the notebook instance in GB. The volume size must be from 5 GB to 16384 GB (16 TB).

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the AmazonSageMakerFullAccess IAM policy attached.

Amazon SageMaker

Amazon SageMaker Studio

Dashboard
Search
Images

Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces

Notebook
Notebook instances
Lifecycle configurations
Git repositories

Processing
Processing jobs

Training
Algorithms
Training jobs
Hyperparameter tuning jobs

Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints

Amazon SageMaker > Notebook instancesLunarLander > Edit notebook instance

Edit notebook instance

Notebook instance settings

Notebook instance name

LunarLander

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

Notebook instance type

ml.p3.2xlarge

ml.c5d.9xlarge
ml.c5d.18xlarge
ml.c4.xlarge
ml.c4.2xlarge
ml.c4.4xlarge
ml.c4.8xlarge

Accelerated computing

ml.p2.xlarge
ml.p2.8xlarge
ml.p2.16xlarge
ml.p3.2xlarge
ml.p3.8xlarge
ml.p3.16xlarge

AmazonSageMakerFullAccess IAM policy attached.

LunarLander-SagemakerIAM-D1AH3G7NVVSJ

Amazon SageMaker Studio
Dashboard
Search
Images
Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints

Notebook instances

Actions | Create notebook instance

Search notebook instances

1

Name	Instance	Creation time	Status	Actions

Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corbuno-liengard @ corbuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker ×

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints

Amazon SageMaker > Notebook instances

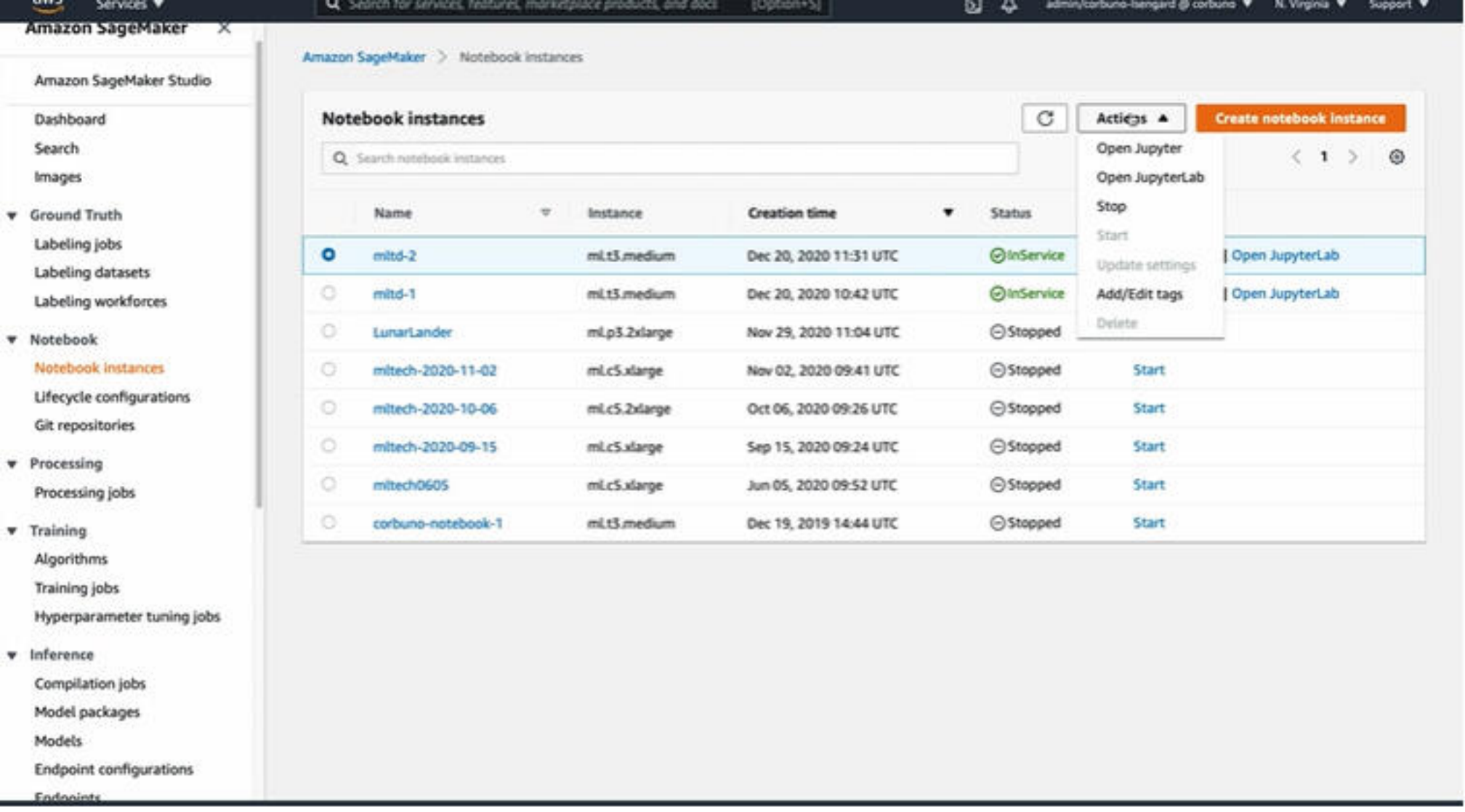
Notebook instances

Actions ▾ | Create notebook instance

Search notebook instances

1

Name	Instance	Creation time	Status	Actions
miltd-2	ml.t3.medium	Dec 20, 2020 11:31 UTC	InService	Open Jupyter | Open JupyterLab
miltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	InService	Open Jupyter | Open JupyterLab
LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
miltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
miltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
miltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
miltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start


Services
Search for services, features, marketplace products, and docs [Option+S]
admin/corbuno-liengard @ corbuno
N. Virginia
Support
Amazon SageMaker
Amazon SageMaker Studio
Dashboard
Search
Images
Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces
Notebook
Notebook instances
Lifecycle configurations
Git repositories
Processing
Processing jobs
Training
Algorithms
Training jobs
Hyperparameter tuning jobs
Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints
Amazon SageMaker > Notebook instances
Notebook instances
Actions
Create notebook instance
Search notebook instances
Open Jupyter
Open JupyterLab
Stop
Start
Update settings
Add/Edit tags
Delete
Name
Instance
Creation time
Status
mltd-2 ml.t3.medium Dec 20, 2020 11:31 UTC InService Open JupyterLab
mltd-1 ml.t3.medium Dec 20, 2020 10:42 UTC InService Open JupyterLab
LunarLander ml.p3.2xlarge Nov 29, 2020 11:04 UTC Stopped
mltech-2020-11-02 ml.c5.xlarge Nov 02, 2020 09:41 UTC Stopped Start
mltech-2020-10-06 ml.c5.2xlarge Oct 06, 2020 09:26 UTC Stopped Start
mltech-2020-09-15 ml.c5.xlarge Sep 15, 2020 09:24 UTC Stopped Start
mltech0605 ml.c5.xlarge Jun 05, 2020 09:52 UTC Stopped Start
corbuno-notebook-1 ml.t3.medium Dec 19, 2019 14:44 UTC Stopped Start

Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corbuno-liengard @ corbuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker ×

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints

Amazon SageMaker > Notebook instances

Notebook instances

Actions ▾ | Create notebook instance

Search notebook instances

1

Name	Instance	Creation time	Status	Actions
mltd-2	ml.t3.medium	Dec 20, 2020 11:31 UTC	Stopping	-
mltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	InService	Open Jupyter | Open JupyterLab
LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
mltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
mltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
mltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
mltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start

Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corbuno-liengard @ corbuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker ×

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints

Amazon SageMaker > Notebook instances

Notebook instances

Actions ▾ | Create notebook instance

Search notebook instances

1

	Name	Instance	Creation time	Status	Actions
○	mltd-2	ml.t3.medium	Dec 20, 2020 11:31 UTC	Stopping	-
●	mltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	InService	Open Jupyter | Open JupyterLab
○	LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
○	mltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
○	mltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
○	mltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
○	mltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
○	corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start

Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corbuno-liengard @ corbuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker ×

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints

Amazon SageMaker > Notebook instances

Notebook instances

Actions ▾ | Create notebook instance

Search notebook instances

1

	Name	Instance	Creation time	Status	Actions
○	mltd-2	ml.t3.medium	Dec 20, 2020 11:31 UTC	Stopping	-
●	mltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	Stopping	-
○	LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
○	mltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
○	mltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
○	mltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
○	mltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
○	corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start

Built-in algorithm mode – SageMaker Python API

```
# BUILD
algorithm = get_image_uri(...)
model = Estimator(algorithm,role,instance_type)
model.set_hyperparameters(...)

# TRAIN
model.fit(train_data)

# DEPLOY
endpoint = model.deploy(instance_type)
endpoint.predict(test_data)
```

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Built-in algorithm mode – SageMaker Python API

```
# BUILD
algorithm = get_image_uri(...)
model = Estimator(algorithm,role,instance_type)
model.set_hyperparameters(...)

# TRAIN
model.fit(train_data)

# DEPLOY
endpoint = model.deploy(instance_type)
endpoint.predict(test_data)
```

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Built-in algorithm mode – SageMaker Python API

```
# BUILD
algorithm = get_image_uri(...)
model = Estimator(algorithm,role,instance_type)
model.set_hyperparameters(...)

# TRAIN
model.fit(train_data)

# DEPLOY
endpoint = model.deploy(instance_type)
endpoint.predict(test_data)
```

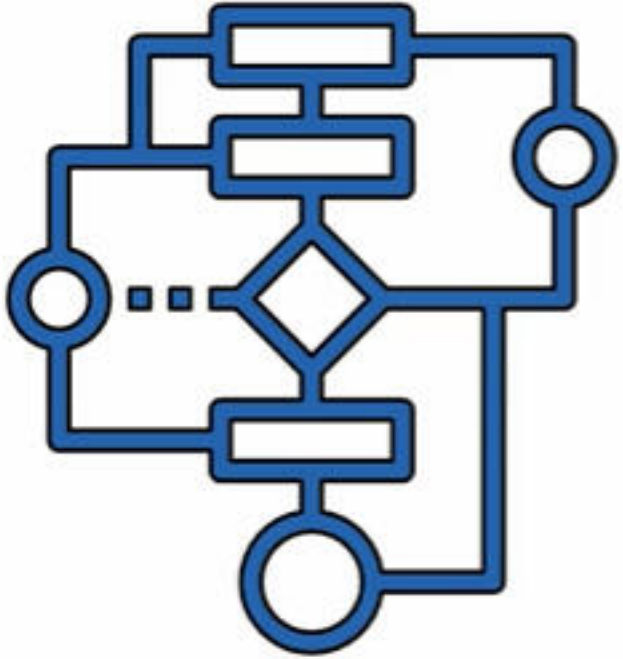
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Module 8: SageMaker Built-In Algorithms

- Identify and explain the purpose of the Amazon SageMaker built-in algorithms
- Locate Amazon SageMaker algorithm examples in the console
- Select the appropriate algorithm for a business problem and data
- Configure and initiate a training job
- Associate a training job with an appropriate Amazon Simple Storage Service (Amazon S3) bucket
- Use queries with a model

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

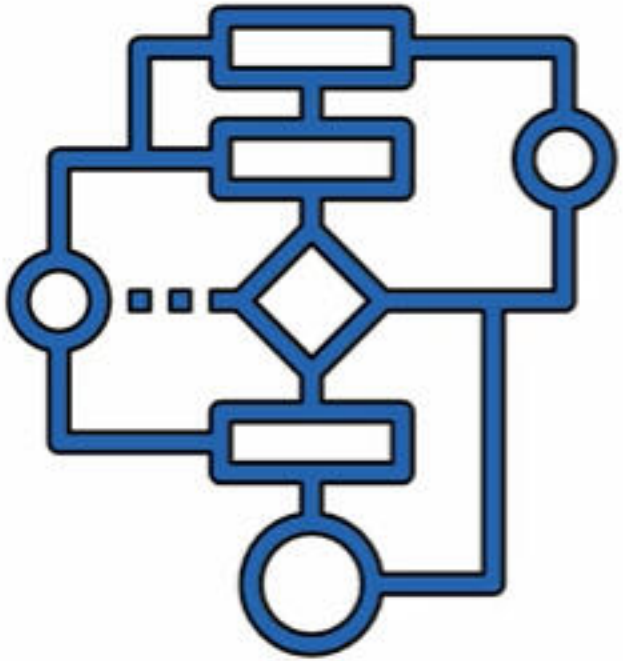
Algorithms



- *Algorithms* are standardized methods used to train models.
- A *model* is a function that maps inputs to a set of predicted outcomes using algorithms.
- The exact definition of a model is unknown.
- Existing data is used to build a function, using rules, and this is called *training*.
- Algorithms define model rules.

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Types of algorithms



- Linear regression
- Logistic regression
- Clustering
- Vectorization
- Image classification
- Encoding and decoding
- Language processing

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker Built-in Algorithms

aws training and certification

General purpose

Numerical regression or classification
- Linear Learner
- XGBoost
- K-Nearest Neighbors (K-NN)

Recommendation
Factorization Machines

Group entities based on data
K-Means

Detect anomalies in time series data
Random Cut Forest (RCF)

Specific use cases

Classify images or find objects in images
- Image Classification
- Object Detection
- Semantic Segmentation

Classify, encode, and transform text data
- Sequence to Sequence (seq2seq)
- Neural Topic Model (NTM)
- Latent Dirichlet Allocation (LDA)
- Blazing Text
- Object2Vec

Predict future trends based on past history (time series)
DeepAR Forecasting

Reduce dimensions in datasets with high numbers of attributes
Principle Component Analysis (PCA)

Find usage patterns in network access logs
IP Insights

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

- **Five** scenarios relating to Amazon SageMaker algorithms
- **Four** options
- **One** option addresses the scenario

-46:34 2x

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 1

- Real estate company
- Predict price of newly listed home
- Dataset has 100,000 records of selling prices of house in the area
- Dataset has 12 features

Question 1

Which Amazon SageMaker algorithm would be most appropriate to use?

DeepAR Forecasting	Linear Learner algorithm
Random Cut Forest algorithm	Factorization Machines

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 1

- Real estate company
- Predict price of newly listed home
- Dataset has 100,000 records of selling prices of house in the area
- Dataset has 12 features

Question 1

Which Amazon SageMaker algorithm would be most appropriate to use?

DeepAR Forecasting	Linear Learner algorithm
Random Cut Forest algorithm	Factorization Machines

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 1

- Real estate company
- Predict price of newly listed home
- Dataset has 100,000 records of selling prices of house in the area
- Dataset has 12 features

Question 1

Which Amazon SageMaker algorithm would be most appropriate to use?

DeepAR Forecasting	Linear Learner algorithm
Random Cut Forest algorithm	Factorization Machines

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 2

- Home insurance company
- Estimating rebuild cost has been manual
- Goal is to automate this process using historical data
- Hundreds of features and some are more important than others

Question 2

Which Amazon SageMaker algorithm should you use?

Random Cut Forest algorithm	Linear Learner algorithm
XGBoost algorithm	K-Nearest Neighbors (k-NN) algorithm

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 2

- Home insurance company
- Estimating rebuild cost has been manual
- Goal is to automate this process using historical data
- Many features and some are more important than others

Question 2

Which Amazon SageMaker algorithm should you use?

Random Cut Forest algorithm	Linear Learner algorithm
XGBoost algorithm	K-Nearest Neighbors (k-NN) algorithm

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 2

- Home insurance company
- Estimating rebuild cost has been manual
- Goal is to automate this process using historical data
- Many features and some are more important than others

Question 2

Which Amazon SageMaker algorithm should you use?

Random Cut Forest algorithm	Linear Learner algorithm
XGBoost algorithm	K-Nearest Neighbors (k-NN) algorithm

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 3

- National bank
- Risk scoring clients based on parameters
- Current algorithm based on hand-coded business rules and 50 parameters
- Need to classify based on similarity of parameters

Question 3

Which Amazon SageMaker algorithm should you use?

Factorization Machines	K-Nearest Neighbors (k-NN) algorithm
K-Means algorithm	Random Cut Forest algorithm

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 4

- Credit union
- Basing lending decisions on credit score
- They have data on members and want to group based on low, medium, and high risk
- Want to predict which group a potential borrower fits into

Question 4

Which Amazon SageMaker algorithm should you use?

Linear Learner algorithm	K-Nearest Neighbors (k-NN) algorithm
K-Means algorithm	XGBoost algorithm

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 4

- Credit union
- Basing lending decisions on credit score
- They have data on members and want to group based on low, medium, and high risk
- Want to predict which group a potential borrower fits into

Question 4

Which Amazon SageMaker algorithm should you use?

Linear Learner algorithm	K-Nearest Neighbors (k-NN) algorithm
K-Means algorithm	XGBoost algorithm

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 5

- Manufacturing company, making 10 valves
- Products built in random order, need to be classified for labeling and packaging
- Need to automate classification robotics
- Using AWS IoT

Question 5

Which Amazon SageMaker algorithm should you use?

Object2Vec	Object Detection algorithm
Image Classification algorithm	K-Means algorithm

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Algorithms

Game Show

Scenario 5

- Manufacturing company, making 10 valves
- Products built in random order, need to be classified for labeling and packaging
- Need to automate classification robotics
- Using AWS IoT

Question 5

Which Amazon SageMaker algorithm should you use?

Object2Vec	Object Detection algorithm
Image Classification algorithm	K-Means algorithm

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Workbook 8.1

aws training and certification

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	policy_deductable	policy_annual_premium	umbrella_limit	insured_zip	police_report_available	total_claim_amount	injury_claim	property_claim	vehicle_claim	auto_make	auto_model	auto_year	fraud_reported
0	328	48	521585	2014-10-17	OH	250/500	1000	1406.91	0	466132	YES	71610	6510	13020	52080	Saab	92x	2004	Y
1	228	42	342868	2006-06-27	IN	250/500	2000	1197.22	5000000	468176	?	5070	780	780	3510	Mercedes	E400	2007	Y
2	134	29	687698	2000-09-06	OH	100/300	2000	1413.14	5000000	430632	NO	34650	7700	3850	23100	Dodge	RAM	2007	N
3	256	41	227811	1990-05-25	IL	250/500	2000	1415.74	6000000	608117	NO	63400	6340	6340	50720	Chevrolet	Tahoe	2014	Y
4	228	44	367455	2014-06-06	IL	500/1000	1000	1583.91	6000000	610706	NO	6500	1300	650	4550	Acura	RSX	2009	N

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

How you can help customers

- Understand the data structure and the data's purpose
- Determine which algorithm to use based on an ML problem and data
- Use the *Amazon SageMaker Developer Guide*
- Experiment with algorithms to find the optimal algorithm to solve an ML problem

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

How you can help customers

- Understand the data structure and the data's purpose
- Determine which algorithm to use based on an ML problem and data
- Use the *Amazon SageMaker Developer Guide*
- Experiment with algorithms to find the optimal algorithm to solve an ML problem

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

How you can help customers

- Understand the data structure and the data's purpose
- Determine which algorithm to use based on an ML problem and data
- Use the *Amazon SageMaker Developer Guide*
- Experiment with algorithms to find the optimal algorithm to solve an ML problem

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

- Autopilot: Automated ML
- Label Data
- Prepare and Analyze Datasets
- Create, Store, and Share Features
- Training
  - Choose an algorithm
    - Use built-in algorithms
      - Common Information
      - BlazingText
      - DeepAR Forecasting
      - Factorization Machines
      - Image Classification Algorithm
      - IP Insights
      - K-Means Algorithm
        - How It Works
        - Hyperparameters
        - Model Tuning
        - Inference Formats
      - K-Nearest Neighbors (k-NN) Algorithm
      - Latent Dirichlet Allocation (LDA)
      - Linear Learner Algorithm
      - Neural Topic Model (NTM) Algorithm
      - Object2Vec
      - Object Detection Algorithm

Choose an Algorithm

PDF | Kindle | RSS

Machine leaning can help you accomplish empirical tasks that require some sort of inductive inference. This task involves induction as it uses data to train algorithms to make generalizable inferences. This means that the algorithms can make statistically reliable predictions or decisions, or complete other tasks when applied to new data that was not used to train them.

To help you select the best algorithm for your task, we classify these tasks on various levels of abstraction. At the highest level of abstraction, machine learning attempts to find patterns or relationships between features or less structured items, such as text in a data set. Pattern recognition techniques can be classified into distinct machine learning paradigms, each of which address specific problem types. There are currently three basic paradigms for machine learning used to address various problem types:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

The types of problems that each learning paradigm can address are identified by considering the inferences (or predictions, decisions, or other tasks) you want to make from the type of data that you have or could collect. Machine learning paradigms use algorithmic methods to address their various problem types. The algorithms provide recipes for solving these problems.

However, many algorithms, such as neural networks, can be deployed with different learning paradigms and on different types of problems. Multiple algorithms can also address a specific problem type. Some algorithms are more generally applicable and others are quite specific for certain kinds of objectives and data. So the mapping between machine learning algorithms and problem types is many-to-many. Also, there are various implementation options available for algorithms.

The following sections provide guidance concerning implementation options, machine learning paradigms, and algorithms appropriate for different problem types.

Topics

- Choose an algorithm implementation
- Problem types for the basic machine learning paradigms

On this page

Choose an algorithm implementation
Problem types for the basic machine learning paradigms

- Autopilot: Automated ML
- Label Data
- Prepare and Analyze Datasets
- Create, Store, and Share Features
- Training
  - Choose an algorithm
    - Use built-in algorithms
      - Common Information
      - BlazingText
      - DeepAR Forecasting
      - Factorization Machines
      - Image Classification Algorithm
      - IP Insights
      - K-Means Algorithm
        - How It Works
        - Hyperparameters
        - Model Tuning
        - Inference Formats
      - K-Nearest Neighbors (k-NN) Algorithm
      - Latent Dirichlet Allocation (LDA)
      - Linear Learner Algorithm
      - Neural Topic Model (NTM) Algorithm
      - Object2Vec
      - Object Detection Algorithm

them.

To help you select the best algorithm for your task, we classify these tasks on various levels of abstraction. At the highest level of abstraction, machine learning attempts to find patterns or relationships between features or less structured items, such as text in a data set. Pattern recognition techniques can be classified into distinct machine learning paradigms, each of which address specific problem types. There are currently three basic paradigms for machine learning used to address various problem types:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

The types of problems that each learning paradigm can address are identified by considering the inferences (or predictions, decisions, or other tasks) you want to make from the type of data that you have or could collect. Machine learning paradigms use algorithmic methods to address their various problem types. The algorithms provide recipes for solving these problems.

However, many algorithms, such as neural networks, can be deployed with different learning paradigms and on different types of problems. Multiple algorithms can also address a specific problem type. Some algorithms are more generally applicable and others are quite specific for certain kinds of objectives and data. So the mapping between machine learning algorithms and problem types is many-to-many. Also, there are various implementation options available for algorithms.

The following sections provide guidance concerning implementation options, machine learning paradigms, and algorithms appropriate for different problem types.

Topics

- Choose an algorithm implementation
- Problem types for the basic machine learning paradigms
- Use Amazon SageMaker Built-in Algorithms
- Use reinforcement learning with Amazon SageMaker

Choose an algorithm implementation

After choosing an algorithm, you must decide which implementation of it you want to use. Amazon SageMaker supports three implementation options that require increasing levels of effort

On this page

Choose an algorithm implementation
Problem types for the basic machine learning paradigms

- Autopilot: Automated ML
- Label Data
- Prepare and Analyze Datasets
- Create, Store, and Share Features
- Training
  - Choose an algorithm
    - Use built-in algorithms
      - Common Information
      - BlazingText
      - DeepAR Forecasting
      - Factorization Machines
      - Image Classification Algorithm
      - IP Insights
      - K-Means Algorithm
        - How It Works
        - Hyperparameters
        - Model Tuning
        - Inference Formats
      - K-Nearest Neighbors (k-NN) Algorithm
      - Latent Dirichlet Allocation (LDA)
      - Linear Learner Algorithm
      - Neural Topic Model (NTM) Algorithm
      - Object2Vec
      - Object Detection Algorithm

Use Amazon SageMaker Built-in Algorithms

PDF | Kindle | RSS

Amazon SageMaker provides a suite of built-in algorithms to help data scientists and machine learning practitioners get started on training and deploying machine learning models quickly. For someone that is new to SageMaker, choosing the right algorithm for your particular use case can be a challenging task. The following table provides a quick cheat sheet that shows how you can start with an example problem or use case and find an appropriate built-in algorithm offered by SageMaker that is valid for that problem type. Additional guidance organized by learning paradigms (supervised and unsupervised) and important data domains (text and images) is provided in the sections following the table.

Table: Mapping use cases to built-in algorithms

Example problems and use cases	Learning paradigm or domain	Problem types	Data input format	Built-in algorithms
Predict if an item belongs to a category: an email spam filter	Supervised Learning	Binary/multi-class classification	Tabular	Factorization Machines Algorithm, K-Nearest Neighbors (k-NN) Algorithm, Linear Learner Algorithm, XGBoost Algorithm
Predict a numeric/continuous value: estimate the value of a house		Regression	Tabular	Factorization Machines Algorithm, K-Nearest Neighbors (k-NN)

On this page

- Supervised learning algorithms
- Unsupervised Learning
- Textual Analysis
- Image Processing

- Autopilot: Automated ML
- Label Data
- Prepare and Analyze Datasets
- Create, Store, and Share Features
- Training
  - Choose an algorithm
    - Use built-in algorithms
      - Common Information
      - BlazingText
      - DeepAR Forecasting
      - Factorization Machines
      - Image Classification Algorithm
      - IP Insights
      - K-Means Algorithm
        - How It Works
        - Hyperparameters
        - Model Tuning
        - Inference Formats
      - K-Nearest Neighbors (k-NN) Algorithm
      - Latent Dirichlet Allocation (LDA)
      - Linear Learner Algorithm
      - Neural Topic Model (NTM) Algorithm
      - Object2Vec
      - Object Detection Algorithm

how you can start with an example problem or use case and find an appropriate built-in algorithm offered by SageMaker that is valid for that problem type. Additional guidance organized by learning paradigms (supervised and unsupervised) and important data domains (text and images) is provided in the sections following the table.

Table: Mapping use cases to built-in algorithms

Example problems and use cases	Learning paradigm or domain	Problem types	Data input format	Built-in algorithms
Predict if an item belongs to a category: an email spam filter	Supervised Learning	Binary/multi-class classification	Tabular	Factorization Machines Algorithm, K-Nearest Neighbors (k-NN) Algorithm, Linear Learner Algorithm, XGBoost Algorithm
Predict a numeric/continuous value: estimate the value of a house		Regression	Tabular	Factorization Machines Algorithm, K-Nearest Neighbors (k-NN) Algorithm, Linear Learner Algorithm

For important information about Docker registry paths, data formats, recommended Amazon EC2 instance types, and CloudWatch logs common to all of the built-in algorithms provided by SageMaker, see Common Information About Built-in Algorithms.

The following sections provide additional guidance for the Amazon SageMaker built-in algorithms grouped by the supervised

On this page
- Supervised learning algorithms
- Unsupervised Learning
- Textual Analysis
- Image Processing

- Autopilot: Automated ML
- Label Data
- Prepare and Analyze Datasets
- Create, Store, and Share Features
- Training
  - Choose an algorithm
    - Use built-in algorithms
      - Common Information
      - BlazingText
      - DeepAR Forecasting
      - Factorization Machines
      - Image Classification Algorithm
      - IP Insights
      - K-Means Algorithm
        - How It Works
        - Hyperparameters
        - Model Tuning
        - Inference Formats
      - K-Nearest Neighbors (k-NN) Algorithm
      - Latent Dirichlet Allocation (LDA)
      - Linear Learner Algorithm
      - Neural Topic Model (NTM) Algorithm
      - Object2Vec
      - Object Detection Algorithm

how you can start with an example problem or use case and find an appropriate built-in algorithm offered by SageMaker that is valid for that problem type. Additional guidance organized by learning paradigms (supervised and unsupervised) and important data domains (text and images) is provided in the sections following the table.

Table: Mapping use cases to built-in algorithms

Example problems and use cases	Learning paradigm or domain	Problem types	Data input format	Built-in algorithms
Predict if an item belongs to a category: an email spam filter	Supervised Learning	Binary/multi-class classification	Tabular	Factorization Machines Algorithm, K-Nearest Neighbors (k-NN) Algorithm, Linear Learner Algorithm, XGBoost Algorithm
Predict a numeric/continuous value: estimate the value of a house		Regression	Tabular	Factorization Machines Algorithm, K-Nearest Neighbors (k-NN) Algorithm, Linear Learner Algorithm,

For important information about Docker registry paths, data formats, recommenced Amazon EC2 instance types, and CloudWatch logs common to all of the built-in algorithms provided by SageMaker, see Common Information About Built-in Algorithms.

The following sections provide additional guidance for the Amazon SageMaker built-in algorithms grouped by the supervised

On this page

- Supervised learning algorithms
- Unsupervised Learning
- Textual Analysis
- Image Processing

- Autopilot: Automated ML
- Label Data
- Prepare and Analyze Datasets
- Create, Store, and Share Features
- Training
  - Choose an algorithm
    - Use built-in algorithms
      - Common Information
      - BlazingText
      - DeepAR Forecasting
      - Factorization Machines
      - Image Classification Algorithm
      - IP Insights
      - K-Means Algorithm
        - How It Works
        - Hyperparameters
        - Model Tuning
        - Inference Formats
      - K-Nearest Neighbors (k-NN) Algorithm
      - Latent Dirichlet Allocation (LDA)
      - Linear Learner Algorithm
      - Neural Topic Model (NTM) Algorithm
      - Object2Vec
      - Object Detection Algorithm

K-Means Algorithm

PDF | Kindle | RSS

K-means is an unsupervised learning algorithm. It attempts to find discrete groupings within data, where members of a group are as similar as possible to one another and as different as possible from members of other groups. You define the attributes that you want the algorithm to use to determine similarity.

Amazon SageMaker uses a modified version of the web-scale k-means clustering algorithm. Compared with the original version of the algorithm, the version used by Amazon SageMaker is more accurate. Like the original algorithm, it scales to massive datasets and delivers improvements in training time. To do this, the version used by Amazon SageMaker streams mini-batches (small, random subsets) of the training data. For more information about mini-batch k-means, see Web-scale k-means Clustering.

The k-means algorithm expects tabular data, where rows represent the observations that you want to cluster, and the columns represent attributes of the observations. The *n* attributes in each row represent a point in *n*-dimensional space. The Euclidean distance between these points represents the similarity of the corresponding observations. The algorithm groups observations with similar attribute values (the points corresponding to these observations are closer together). For more information about how k-means works in Amazon SageMaker, see How K-Means Clustering Works.

Topics

- Input/Output Interface for the K-Means Algorithm
- EC2 Instance Recommendation for the K-Means Algorithm
- K-Means Sample Notebooks
- How K-Means Clustering Works
- K-Means Hyperparameters
- Tune a K-Means Model
- K-Means Response Formats

On this page

- Input/Output Interface for the K-Means Algorithm
- EC2 Instance Recommendation for the K-Means Algorithm
- Sample Notebooks

-23:27 2x

Input/Output Interface for the K-Means Algorithm

- Autopilot: Automated ML
- Label Data
- Prepare and Analyze Datasets
- Create, Store, and Share Features
- Training
  - Choose an algorithm
    - Use built-in algorithms
      - Common Information
      - BlazingText
      - DeepAR Forecasting
      - Factorization Machines
      - Image Classification Algorithm
      - IP Insights
      - K-Means Algorithm
        - How It Works
        - Hyperparameters
        - Model Tuning
        - Inference Formats
      - K-Nearest Neighbors (k-NN) Algorithm
      - Latent Dirichlet Allocation (LDA)
      - Linear Learner Algorithm
      - Neural Topic Model (NTM) Algorithm
      - Object2Vec
      - Object Detection Algorithm

K-Means Hyperparameters

PDF | Kindle | RSS

In the `CreateTrainingJob` request, you specify the training algorithm that you want to use. You can also specify algorithm-specific hyperparameters as string-to-string maps. The following table lists the hyperparameters for the k-means training algorithm provided by Amazon SageMaker. For more information about how k-means clustering works, see How K-Means Clustering Works.

Parameter Name	Description
`feature_dim`	The number of features in the input data. **Required** Valid values: Positive integer
`k`	The number of required clusters. **Required** Valid values: Positive integer
`epochs`	The number of passes done over the training data. **Optional** Valid values: Positive integer Default value: 1
`eval_metrics`	A JSON list of metric types used to report a score for the model. Allowed values are `msd` for Means Square Error and `ssd` for Sum of Square Distance. If test data is provided, the score is reported for each of the metrics requested. **Optional** Valid values: Either `[\"msd\"]` or `[\"ssd\"]` or `[\"msd\",\"ssd\"]`. Default value: `[\"msd\"]`

How you can help customers

- Understand the data structure and the data's purpose
- Determine which algorithm to use based on an ML problem and data
- Use the *Amazon SageMaker Developer Guide*
- Experiment with algorithms to find the optimal algorithm to solve an ML problem

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

- Autopilot: Automated ML
- Label Data
- Prepare and Analyze Datasets
- Create, Store, and Share Features
- Training
  - Choose an algorithm
    - Use built-in algorithms
      - Common Information
      - BlazingText
      - DeepAR Forecasting
      - Factorization Machines
      - Image Classification Algorithm
      - IP Insights
      - K-Means Algorithm
        - How It Works
        - Hyperparameters
        - Model Tuning
        - Inference Formats
      - K-Nearest Neighbors (k-NN) Algorithm
      - Latent Dirichlet Allocation (LDA)
      - Linear Learner Algorithm
      - Neural Topic Model (NTM) Algorithm
      - Object2Vec
      - Object Detection Algorithm

Choose an Algorithm

PDF | Kindle | RSS

Machine leaning can help you accomplish empirical tasks that require some sort of inductive inference. This task involves induction as it uses data to train algorithms to make generalizable inferences. This means that the algorithms can make statistically reliable predictions or decisions, or complete other tasks when applied to new data that was not used to train them.

To help you select the best algorithm for your task, we classify these tasks on various levels of abstraction. At the highest level of abstraction, machine learning attempts to find patterns or relationships between features or less structured items, such as text in a data set. Pattern recognition techniques can be classified into distinct machine learning paradigms, each of which address specific problem types. There are currently three basic paradigms for machine learning used to address various problem types:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

The types of problems that each learning paradigm can address are identified by considering the inferences (or predictions, decisions, or other tasks) you want to make from the type of data that you have or could collect. Machine learning paradigms use algorithmic methods to address their various problem types. The algorithms provide recipes for solving these problems.

However, many algorithms, such as neural networks, can be deployed with different learning paradigms and on different types of problems. Multiple algorithms can also address a specific problem type. Some algorithms are more generally applicable and others are quite specific for certain kinds of objectives and data. So the mapping between machine learning algorithms and problem types is many-to-many. Also, there are various implementation options available for algorithms.

The following sections provide guidance concerning implementation options, machine learning paradigms, and algorithms appropriate for different problem types.

Topics

- Choose an algorithm implementation
- Problem types for the basic machine learning paradigms

On this page

Choose an algorithm implementation
Problem types for the basic machine learning paradigms

- Autopilot: Automated ML
- Label Data
- Prepare and Analyze Datasets
- Create, Store, and Share Features
- Training
  - Choose an algorithm
    - Use built-in algorithms
      - Common Information
      - BlazingText
      - DeepAR Forecasting
      - Factorization Machines
      - Image Classification Algorithm
      - IP Insights
      - K-Means Algorithm
        - How It Works
        - Hyperparameters
        - Model Tuning
        - Inference Formats
      - K-Nearest Neighbors (k-NN) Algorithm
      - Latent Dirichlet Allocation (LDA)
      - Linear Learner Algorithm
      - Neural Topic Model (NTM) Algorithm
      - Object2Vec
      - Object Detection Algorithm

Use Amazon SageMaker Built-in Algorithms

PDF | Kindle | RSS

Amazon SageMaker provides a suite of built-in algorithms to help data scientists and machine learning practitioners get started on training and deploying machine learning models quickly. For someone that is new to SageMaker, choosing the right algorithm for your particular use case can be a challenging task. The following table provides a quick cheat sheet that shows how you can start with an example problem or use case and find an appropriate built-in algorithm offered by SageMaker that is valid for that problem type. Additional guidance organized by learning paradigms (supervised and unsupervised) and important data domains (text and images) is provided in the sections following the table.

Table: Mapping use cases to built-in algorithms

Example problems and use cases	Learning paradigm or domain	Problem types	Data input format	Built-in algorithms
Predict if an item belongs to a category: an email spam filter	Supervised Learning	Binary/multi-class classification	Tabular	Factorization Machines Algorithm, K-Nearest Neighbors (k-NN) Algorithm, Linear Learner Algorithm, XGBoost Algorithm
Predict a numeric/continuous value: estimate the value of a house		Regression	Tabular	Factorization Machines Algorithm, K-Nearest Neighbors (k-NN)

On this page

- Supervised learning algorithms
- Unsupervised Learning
- Textual Analysis
- Image Processing

Autopilot: Automated ML
Label Data
Prepare and Analyze Datasets
Create, Store, and Share Features
Training
Choose an algorithm
Use built-in algorithms
Common Information
BlazingText
DeepAR Forecasting
Factorization Machines
Image Classification Algorithm
IP Insights
K-Means Algorithm
How It Works
Hyperparameters
Model Tuning
Inference Formats
K-Nearest Neighbors (k-NN) Algorithm
Latent Dirichlet Allocation (LDA)
Linear Learner Algorithm
Neural Topic Model (NTM) Algorithm
Object2Vec
Object Detection Algorithm

how you can start with an example problem or use case and find an appropriate built-in algorithm offered by SageMaker that is valid for that problem type. Additional guidance organized by learning paradigms (supervised and unsupervised) and important data domains (text and images) is provided in the sections following the table.

Table: Mapping use cases to built-in algorithms

Example problems and use cases	Learning paradigm or domain	Problem types	Data input format	Built-in algorithms
Predict if an item belongs to a category: an email spam filter	Supervised Learning	Binary/multi-class classification	Tabular	Factorization Machines Algorithm, K-Nearest Neighbors (k-NN) Algorithm, Linear Learner Algorithm, XGBoost Algorithm
Predict a numeric/continuous value: estimate the value of a house		Regression	Tabular	Factorization Machines Algorithm, K-Nearest Neighbors (k-NN) Algorithm, Linear Learner Algorithm

For important information about Docker registry paths, data formats, recommended Amazon EC2 instance types, and CloudWatch logs common to all of the built-in algorithms provided by SageMaker, see Common Information About Built-in Algorithms.

The following sections provide additional guidance for the Amazon SageMaker built-in algorithms grouped by the supervised

On this page

Supervised learning algorithms
Unsupervised Learning
Textual Analysis
Image Processing

Insurance Fraud Notebook Demo 2: Data preparation and model training in Amazon SageMaker

aws training and certification

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Demo 2 – Outcomes

- Use Amazon SageMaker for heavy lifting after data discovery
- Configure the Amazon S3 bucket
- Ensure that cleaning and preparation steps occur in Amazon SageMaker
- Obtain container image for linear-learner algorithm
- Create a training job
- Set hyperparameters
- Monitor training job status

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Demo 2 – Outcomes

aws training and certification

- Use Amazon SageMaker for heavy lifting after data discovery
- Configure the Amazon S3 bucket
- Ensure that cleaning and preparation steps occur in Amazon SageMaker
- Obtain container image for linear-learner algorithm
- Create a training job
- Set hyperparameters
- Monitor training job status

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Demo 2 – Outcomes

- Use Amazon SageMaker for heavy lifting after data discovery
- Configure the Amazon S3 bucket
- Ensure that cleaning and preparation steps occur in Amazon SageMaker
- Obtain container image for linear-learner algorithm
- Create a training job
- Set hyperparameters
- Monitor training job status

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker

Amazon SageMaker Studio

Dashboard
Search
Images

Ground Truth
Labeling jobs
Labeling datasets
Labeling workforces

Notebook
Notebook instances
Lifecycle configurations
Git repositories

Processing
Processing jobs

Training
Algorithms
Training jobs
Hyperparameter tuning jobs

Inference
Compilation jobs
Model packages
Models
Endpoint configurations
Endpoints
Batch transform jobs

Edge Inference
Get started
Edge packaging jobs
Edge device fleets

MACHINE LEARNING

Amazon SageMaker

Build, train, and deploy machine learning models at scale

The quickest and easiest way to get ML models from idea to production.

Get started

Explore SageMaker Studio, a machine learning Integrated Development Environment (IDE) for building, training, and debugging models, tracking experiments, deploying models, and monitoring their performance. This is available in the following AWS Regions: US East (Ohio), US East (N. Virginia), US West (Oregon), and Europe (Ireland).

SageMaker Studio

Pricing (US)

With Amazon SageMaker, you pay only for what you use. Authoring, training and hosting is billed by the second, with no minimum fees and no upfront commitments.

Learn more

Related services

AWS Glue

Amazon EC2

Amazon Elastic Block Store (EBS)

More resources

Developer resources

AWS Developer Forum

How it works

Label

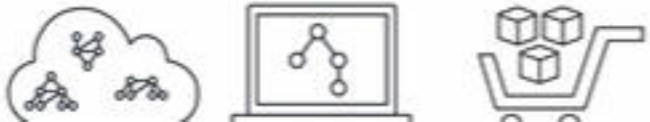
Set up and manage labeling jobs for highly accurate training datasets within Amazon SageMaker, using active learning and human labeling

Build

Connect to other AWS services and transform data in Amazon SageMaker notebooks

Train

Use Amazon SageMaker's algorithms and frameworks, or bring your own, for distributed training



JupyterLab | Amazon SageMaker | Home | Insurance_Fraud_Prediction_Scikit-Learn | Insurance_Fraud_Prediction_SageMaker-Linear_Learner -...

aws Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corkuno-bergard @ corkuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
  - Get started
  - Edge packaging jobs
  - Edge device fleets

Amazon SageMaker > Notebook instances

Notebook instances

Actions | Create notebook instance

Search notebook instances

Name	Instance	Creation time	Status	Actions

JupyterLab | Amazon SageMaker | Home | Insurance_Fraud_Prediction_Solo1-Learn | Insurance_Fraud_Prediction_SageMaker-Linear_Learner -...

aws Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corbuno-bergard @ corbuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
  - Get started
  - Edge packaging jobs
  - Edge device fleets

Amazon SageMaker > Notebook instances

Notebook instances

Actions | Create notebook instance

Search notebook instances

1

Name	Instance	Creation time	Status	Actions
mltd-2	ml.t3.medium	Dec 20, 2020 11:51 UTC	InService	Open Jupyter | Open JupyterLab
mltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	Stopped	Start
LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
mltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
mltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
mltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
mltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start

JupyterLab | Amazon SageMaker | Home | Insurance_Fraud_Prediction_Scikit-Learn | Insurance_Fraud_Prediction_SageMaker-Linear_Learner -...

aws Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corbuno-lengard @ corbuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
  - Get started
  - Edge packaging jobs
  - Edge device fleets

Amazon SageMaker > Notebook instances

Notebook instances

Actions | Create notebook instance

Search notebook instances

1

Name	Instance	Creation time	Status	Actions
mltd-2	ml.t3.medium	Dec 20, 2020 11:51 UTC	InService	Open Jupyter | Open JupyterLab
mltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	Stopped	Start
LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
mltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
mltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
mltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
mltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start

JupyterLab | Amazon SageMaker | Home | Insurance_Fraud_Prediction_Scikit-Learn | Insurance_Fraud_Prediction_SageMaker-Linear_Learner -...

aws Services ▾ | Search for services, features, marketplace products, and docs [Option+S] | admin/corbuno-lengard @ corbuno ▾ | N. Virginia ▾ | Support ▾

Amazon SageMaker

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
  - Get started
  - Edge packaging jobs
  - Edge device fleets

Amazon SageMaker > Notebook instances

Notebook instances

Actions ▾ | Create notebook instance

Search notebook instances

Name	Instance	Creation time	Status	Actions
mltd-2	ml.t3.medium	Dec 20, 2020 11:51 UTC	InService	Open Jupyter | Open JupyterLab
mltd-1	ml.t3.medium	Dec 20, 2020 10:42 UTC	Stopped	Start
LunarLander	ml.p3.2xlarge	Nov 29, 2020 11:04 UTC	Stopped	Start
mltech-2020-11-02	ml.c5.xlarge	Nov 02, 2020 09:41 UTC	Stopped	Start
mltech-2020-10-06	ml.c5.2xlarge	Oct 06, 2020 09:26 UTC	Stopped	Start
mltech-2020-09-15	ml.c5.xlarge	Sep 15, 2020 09:24 UTC	Stopped	Start
mltech0605	ml.c5.xlarge	Jun 05, 2020 09:52 UTC	Stopped	Start
corbuno-notebook-1	ml.t3.medium	Dec 19, 2019 14:44 UTC	Stopped	Start

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2.ipynb	Running 14 minutes ago	32.9 kB
Insurance_Fraud_Prediction_Scikit-Learn.ipynb	Running 14 minutes ago	24 kB
insurance_claims.csv	19 minutes ago	267 kB

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2.ipynb	Running 14 minutes ago	32.9 kB
Insurance_Fraud_Prediction_Scikit-Learn.ipynb	Running 14 minutes ago	24 kB
insurance_claims.csv	19 minutes ago	267 kB

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2.ipynb	Running 14 minutes ago	32.9 kB
Insurance_Fraud_Prediction_Scikit-Learn.ipynb	Running 14 minutes ago	24 kB
insurance_claims.csv	19 minutes ago	267 kB

jupyter

Open JupyterLab | Quit | Logout

Files | Running | Clusters | SageMaker Examples | Conda

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2.ipynb	Running 14 minutes ago	32.9 kB
Insurance_Fraud_Prediction_Scikit-Learn.ipynb	Running 14 minutes ago	24 kB
insurance_claims.csv	19 minutes ago	267 kB

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 14 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

Slide Type Slide

Myth: Fraud is a victimless crime.

Reality : Insurance fraud is one of the biggest crimes in the U.S.

- Every year claims and underwriting fraud cost **$80 billion**
- Fraudulent claims account for **5-10%** of all claims

Exhibit 2

Estimated Annual Losses in P&C Lines due to Fraudulent Claims in 2014 ($ Million)[6]

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 14 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

Slide Type Slide

We'll demonstrate an attempt to solve this problem by using an auto insurance dataset containing 1000 observations on 39 features, and using this data to train a classifier to be able to predict whether a particular claim is fraudulent or not.

The dataset is made available by APN Partner - Databricks, a company that provides an unified analytics platform, nuifying data science and engineering across the Machine Learning lifecycle.

In [*]:

Slide Type Sub-Slide

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv("insurance_claims.csv")
```

Slide Type Slide

Data exploration

Display number of observations (rows) and features (columns)

In [ ]:

Slide Type Sub-Slide

```
data.shape
```

Slide Type Slide

Quick look at some sample observations reveal the dataset is not the cleanest:

- Some features have wide variations in values
- Some features have missing values

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 15 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

```
data = pd.read_csv('insurance_claims.csv')
```

Slide Type Slide

Data exploration

Display number of observations (rows) and features (columns)

In [4]:

Slide Type Sub-Slide

```
data.shape
```

Out[4]: (1000, 40)

Slide Type Slide

Quick look at some sample observations reveal the dataset is not the cleanest:

- Some features have wide variations in values
- Some features have missing values
- There is a mix of numberic and categorical features

In [5]:

Slide Type Sub-Slide

```
data.head(10)
```

Out[5]:

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	policy_deductable	policy_annual_premium	umbrella_limit	insured_zip
0	328	48	521585	2014-10-17	OH	250/500	1000	1406.91	0	466132
1	228	42	342868	2006-06-27	IN	250/500	2000	1197.22	5000000	468176
2	134	29	687698	2000-09-06	OH	100/300	2000	1413.14	5000000	430632

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 15 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

- There is a mix of numbenc and categoncal features

In [5]:

Slide Type Sub-Slide

```
data.head(10)
```

Out[5]:

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	policy_deductable	policy_annual_premium	umbrella_limit	insured_zip
0	328	48	521585	2014-10-17	OH	250/500	1000	1406.91	0	466132
1	228	42	342868	2006-06-27	IN	250/500	2000	1197.22	5000000	468176
2	134	29	687698	2000-09-06	OH	100/300	2000	1413.14	5000000	430632
3	256	41	227811	1990-05-25	IL	250/500	2000	1415.74	6000000	608117
4	228	44	367455	2014-06-06	IL	500/1000	1000	1583.91	6000000	610706
5	256	39	104594	2006-10-12	OH	250/500	1000	1351.10	0	478456
6	137	34	413978	2000-06-04	IN	250/500	1000	1333.35	0	441716
7	165	37	429027	1990-02-03	IL	100/300	1000	1137.03	0	603195
8	27	33	485665	1997-02-05	IL	100/300	500	1442.99	0	601734
9	212	42	636550	2011-07-25	IL	100/300	500	1315.68	0	600983

10 rows × 40 columns

In [6]:

Slide Type Slide

```
data.dtypes
```

Out[6]:

```
months_as_customer            int64
age                           int64
policy_number                 int64
```

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 15 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

- There is a mix of numbenc and categoncal features

In [5]:

Slide Type Sub-Slide

```
data.head(10)
```

Out[5]:

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	policy_deductable	policy_annual_premium	umbrella_limit	insured_zip
0	328	48	521585	2014-10-17	OH	250/500	1000	1406.91	0	466132
1	228	42	342868	2006-06-27	IN	250/500	2000	1197.22	5000000	468176
2	134	29	687698	2000-09-06	OH	100/300	2000	1413.14	5000000	430632
3	256	41	227811	1990-05-25	IL	250/500	2000	1415.74	6000000	608117
4	228	44	367455	2014-06-06	IL	500/1000	1000	1583.91	6000000	610706
5	256	39	104594	2006-10-12	OH	250/500	1000	1351.10	0	478456
6	137	34	413978	2000-06-04	IN	250/500	1000	1333.35	0	441716
7	165	37	429027	1990-02-03	IL	100/300	1000	1137.03	0	603195
8	27	33	485665	1997-02-05	IL	100/300	500	1442.99	0	601734
9	212	42	636550	2011-07-25	IL	100/300	500	1315.68	0	600983

10 rows × 40 columns

In [6]:

Slide Type Slide

```
data.dtypes
```

Out[6]:

```
months_as_customer              int64
age                             int64
policy_number                   int64
```

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 16 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

- incident_location

Among numeric columns, `insured_zip` is in reality a categorical value, and cannot be encoded meaningfully.

In addition, `policy_number` serves as an idnetifier, and likely have no effect on a claim being fraudulent or not.

These columns are therefore removed, for the sake of simplicity.

In [8]:

Slide Type Sub-Slide

```
colsToDelete = ["policy_number", "policy_bind_date", "insured_zip", "incident_location", "incident_date", "_c39"]

for col in colsToDelete:
    del data[col]
```

Slide Type Sub-Slide

Remaining columns will be encoded later.

In [9]:

Slide Type Sub-Slide

```
filteredStringColList = [i for i in stringCols if i not in colsToDelete]
print(filteredStringColList)
```

```
['policy_state', 'policy_csl', 'insured_sex', 'insured_education_level', 'insured_occupation', 'insured_hobbies', '
insured_relationship', 'incident_type', 'collision_type', 'incident_severity', 'authorities_contacted', 'incident_s
tate', 'incident_city', 'property_damage', 'police_report_available', 'auto_make', 'auto_model', 'fraud_reported']
```

Slide Type Slide

The target variable, to be predicted is `fraud-reported`. Plotting the count of observations belonging the two classes `Y` and `N` also reveals a large imbalance.

In [10]:

Slide Type Sub-Slide

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 17 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

In [12]:

Slide Type Slide

```
data['incident_state_count'] = data['incident_state']
numCol = data.shape[1] -1
for i in range(len(data['incident_state_count'])):
    data.iloc[i, numCol] = istate[data.iloc[i, numCol]]
```

In [*]:

Slide Type -

```
!pip install chart_studio
```

```
Collecting chart_studio
  Downloading chart_studio-1.1.0-py3-none-any.whl (64 kB)
     |[redacted]| 64 kB 2.6 MB/s  eta 0:00:01
Requirement already satisfied: six in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from chart
_studio) (1.14.0)
Requirement already satisfied: plotly in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from ch
art_studio) (4.12.0)
Requirement already satisfied: retrying>=1.3.3 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages
(from chart_studio) (1.3.3)
Requirement already satisfied: requests in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from
chart_studio) (2.22.0)
Requirement already satisfied: idna<2.9,>=2.5 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages
(from requests->chart_studio) (2.8)
Requirement already satisfied: certifi>=2017.4.17 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packa
ges (from requests->chart_studio) (2020.6.20)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-pa
ckages (from requests->chart_studio) (3.0.4)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /home/ec2-user/anaconda3/envs/python3/lib
/python3.6/site-packages (from requests->chart_studio) (1.25.10)
```

In [14]:

Slide Type Slide

```
import plotly.graph_objs as go
```

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 17 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

In [12]:

Slide Type Slide

```
data['incident_state_count'] = data['incident_state']
numCol = data.shape[1] -1
for i in range(len(data['incident_state_count'])):
    data.iloc[i, numCol] = istate[data.iloc[i, numCol]]
```

In [*]:

Slide Type -

```
!pip install chart_studio
```

```
Collecting chart_studio
  Downloading chart_studio-1.1.0-py3-none-any.whl (64 kB)
     |[redacted]| 64 kB 2.6 MB/s  eta 0:00:01
Requirement already satisfied: six in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from chart
_studio) (1.14.0)
Requirement already satisfied: plotly in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from ch
art_studio) (4.12.0)
Requirement already satisfied: retrying>=1.3.3 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages
(from chart_studio) (1.3.3)
Requirement already satisfied: requests in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from
chart_studio) (2.22.0)
Requirement already satisfied: idna<2.9,>=2.5 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages
(from requests->chart_studio) (2.8)
Requirement already satisfied: certifi>=2017.4.17 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packa
ges (from requests->chart_studio) (2020.6.20)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-pa
ckages (from requests->chart_studio) (3.0.4)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /home/ec2-user/anaconda3/envs/python3/lib
/python3.6/site-packages (from requests->chart_studio) (1.25.10)
```

In [14]:

Slide Type Slide

```
import plotly.graph_objs as go
```

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 17 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

- Run Cells
- Run Cells and Select Below
- Run Cells and Insert Below
- Run All
- Run All Above
- Run All Below
- Cell Type
- Current Outputs
- All Output

EDIT CHART

In [ ]: Slide Type -

```
del data['incident_state_count']
```

Slide Type Slide

Plotting total claim amount report by year shows no pattern in claims, meaning date of claim of likely have not much influence.

In [ ]: Slide Type Sub-Slide

```
import plotly.express as px
fig = px.bar(data, x='auto_year', y='total_claim_amount')
fig.show()
```

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 18 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

```
(1000, 161)
(1000,)
```

Slide Type Slide

LogisticRegression - Random Cross-validation

The problem we have at hand now is a binary classification, where we have to predict observations into one of two classes - `1` or `0` (corresponding to whether a particular claim is identified as frauduent or not)

This can be achieved using different algorithms, such as LogisticRegression, RandomForest, DecisionTree, SVM, etc., with various hyper-paramaeters, such as penalty function used, number of estimators, maximum depth of tree etc.

Before doing a full Grid search, our first attempt is to run a Logistic Regression model through grid search and random cross validation, using Scikit-learn libraries, that are computationally inexpensive.

Grid search is performed using a LogisticRegresion learner with `lblinear` solver, with hyperparameter space defined over penalty (L1 and L2 norms) and inverse regularization strenth (0 to 10, with $1/10^6$ fold increments)

In [21]:

Slide Type Slide

```
from sklearn import linear_model, datasets
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
warnings.simplefilter(action='ignore', category=FutureWarning)
```

Slide Type Slide

```
%%time
from scipy.stats import uniform
from sklearn.model_selection import RandomizedSearchCV
```

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 18 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

```
(1000, 161)
(1000,)
```

Slide Type Slide

LogisticRegression - Random Cross-validation

The problem we have at hand now is a binary classification, where we have to predict observations into one of two classes - `1` or `0` (corresponding to whether a particular claim is identified as fraudulent or not)

This can be achieved using different algorithms, such as LogisticRegression, RandomForest, DecisionTree, SVM, etc., with various hyper-paramaeters, such as penalty function used, number of estimators, maximum depth of tree etc.

Before doing a full Grid search, our first attempt is to run a Logistic Regression model through grid search and random cross validation, using Scikit-learn libraries, that are computationally inexpensive.

Grid search is performed using a LogisticRegresion learner with `lblinear` solver, with hyperparameter space defined over penalty (L1 and L2 norms) and inverse regularization strenth (0 to 10, with $1/10^6$ fold increments)

In [21]:

Slide Type Slide

```
from sklearn import linear_model, datasets
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
warnings.simplefilter(action='ignore', category=FutureWarning)
```

Slide Type Slide

```
%%time
from scipy.stats import uniform
from sklearn.model_selection import RandomizedSearchCV
```

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 19 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

```
C = np.random.uniform(0, 10, 100000)
hyperparameters = dict(C=C, penalty=norms)
model = perform_randomized_search(X, Y, logistic, hyperparameters, 10)
```

Slide Type Slide

Classification report - Logistic Regression

As a result of this grid search, we obtain a model that yields an accuracy of close to 88%.

However accuracy in itself is not of much value, specially when dealing with classification with imbalanced classes. In this particular case, as we saw earlier, there is a large imbalance where much more claims are identified to be not fraudulent in training data than the ones that are reported to be fraudulent.

In reality, insurance industry would prefer to predict as many true negatives (under represented class) to minimize the loss.

So we create a classification report to see how our Logistic Regression model performed in predicting both classes.

In [22]:

Slide Type Slide

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3, random_state=30)

logReg = linear_model.LogisticRegression(C=2.2014271313563083, penalty='l1', solver='liblinear')
logReg.fit(X_train, y_train)
y1_pred = logReg.predict(X_test)

print(classification_report(y_test, y1_pred))
```

```
              precision    recall  f1-score   support

           0       0.88      0.91      0.89       233
```

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 20 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

Classification report - DecisionTreeClassifier

We therefore run a classification report using the identified classifier from GridSearch, so as to be able to compare it with the one obtained for Logistic Regression earlier.

In [24]:

Slide Type Slide

```
from sklearn.tree import DecisionTreeClassifier

decT = DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=5,
            max_features=None, max_leaf_nodes=5, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=10,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            presort=False, random_state=None, splitter='random')

decT.fit(X_train, y_train)
y2_pred = decT.predict(X_test)
print(classification_report(y_test, y2_pred))
```

```
              precision    recall  f1-score   support

           0       0.94      0.83      0.88       233
           1       0.59      0.82      0.68        67

    accuracy                           0.83       300
   macro avg       0.76      0.83      0.78       300
weighted avg       0.86      0.83      0.84       300
```

In [25]:

Slide Type Slide

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y2_pred, labels=[0, 1])
tn, fp, fn, tp = cm.ravel()
```

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 21 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

Classification report - DecisionTreeClassifier

We therefore run a classification report using the identified classifier from GridSearch, so as to be able to compare it with the one obtained for Logistic Regression earlier.

In [24]:

Slide Type Slide

```
from sklearn.tree import DecisionTreeClassifier

decT = DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=5,
            max_features=None, max_leaf_nodes=5, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=10,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            presort=False, random_state=None, splitter='random')

decT.fit(X_train, y_train)
y2_pred = decT.predict(X_test)
print(classification_report(y_test, y2_pred))
```

```
              precision    recall  f1-score   support

           0       0.94      0.83      0.88       233
           1       0.59      0.82      0.68        67

    accuracy                           0.83       300
   macro avg       0.76      0.83      0.78       300
weighted avg       0.86      0.83      0.84       300
```

In [25]:

Slide Type Slide

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y2_pred, labels=[0, 1])
tn, fp, fn, tp = cm.ravel()
print("----------------------")
```

jupyter Insurance_Fraud_Prediction_Scikit-Learn Last Checkpoint: 21 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

Classification report - DecisionTreeClassifier

We therefore run a classification report using the identified classifier from GridSearch, so as to be able to compare it with the one obtained for Logistic Regression earlier.

In [24]:

Slide Type Slide

```
from sklearn.tree import DecisionTreeClassifier

decT = DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=5,
            max_features=None, max_leaf_nodes=5, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=10,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            presort=False, random_state=None, splitter='random')

decT.fit(X_train, y_train)
y2_pred = decT.predict(X_test)
print(classification_report(y_test, y2_pred))
```

```
              precision    recall  f1-score   support

           0       0.94      0.83      0.88       233
           1       0.59      0.82      0.68        67

    accuracy                           0.83       300
   macro avg       0.76      0.83      0.78       300
weighted avg       0.86      0.83      0.84       300
```

In [25]:

Slide Type Slide

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y2_pred, labels=[0, 1])
tn, fp, fn, tp = cm.ravel()
print("----------------------")
```

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 22 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Markdown nbdiff

Slide Type Sub-Slide

Configuring an S3 bucket to store training and validation data, and capture model artifacts

Before we build, train and deploy our model we need a place to store the training and validation data and capture the model artifacts. In this case, since we are using Python, notice below that we're using the `boto3` interface to communicate with SageMaker API, and S3 to store the data and model Artifacts. Hence there are no other ML specific libraries imported in this notebook.

The code below connects to an S3 bucket, creates a root folder in that bucket and a subfolder entitled data. This is where the training and validation files will be stored.

In [ ]:

Slide Type Sub-Slide

```
import sagemaker
import boto3

#Create a SageMaker session, you'd use this to use an S3 bucket that SageMaker automatically associates for your use
session = sagemaker.Session()

#Get hold of the region you are operating, you'd use this to get ARN for algorithm container for that region
region = boto3.Session().region_name

#Get the session default bucket
s3_bucket = session.default_bucket()

#Designate location in S3 bucket where data will be staged
s3_root = 'insurance_fraud_prediction-mltb'
s3_data_prefix = s3_root + '/data'
print("Data will be staged at s3://{}/{}".format(s3_bucket, s3_data_prefix))
```

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 23 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

```python
import pandas as pd
from sklearn.model_selection import train_test_split

#Load raw data from source
#data = pd.read_csv("https://raw.githubusercontent.com/jodb/sparkTestData/master/insurance_claims.csv").drop(columns
data = pd.read_csv("insurance_claims.csv").drop(columns="_c39")

#Remove unnecessary columns
colsToDelete = ["policy_number", "policy_bind_date", "insured_zip", "incident_location", "incident_date"]
for col in colsToDelete:
    del data[col]

#Switch target column beginning of the data frame, as is required by SageMaker Linear Learner algorithm
cols = list(data.columns)
cols = [cols[-1]]+cols[:-1]
data = data[cols]

#Change target columns to have '0' and '1' values only
data['fraud_reported'] = data['fraud_reported'].apply(lambda x: 1 if x=='Y' else 0)

#Encode all other categorical column using one-hot encoding
data = pd.get_dummies(data)

#Split transformed dataset into training and validation sets
train_data, val_data = train_test_split(data, test_size=0.2, random_state=30)

print("Training Data : {} Features, {} Records".format(train_data.shape[1], train_data.shape[0]))
print("Validation Data : {} Features, {} Records".format(val_data.shape[1], val_data.shape[0]))
```

```
Training Data : 162 Features, 800 Records
Validation Data : 162 Features, 200 Records
```

Slide Type Slide

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 24 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

In [1]:

Slide Type Sub-Slide

```
import sagemaker
import boto3

#Create a SageMaker session, you'd use this to use an S3 bucket that SageMaker automatically associates for your use
session = sagemaker.Session()

#Get hold of the region you are operating, you'd use this to get ARN for algorithm container for that region
region = boto3.Session().region_name

#Get the session default bucket
s3_bucket = session.default_bucket()

#Designate location in S3 bucket where data will be staged
s3_root = 'insurance_fraud_prediction-mltb'
s3_data_prefix = s3_root + '/data'
print("Data will be staged at s3://{}/{}".format(s3_bucket, s3_data_prefix))
```

```
Data will be staged at s3://sagemaker-us-east-1-649088800912/insurance_fraud_prediction-mltb/data
```

Slide Type Slide

Staging, cleaning and preparing data ¶

Since we already conducted data exploration using Scikit-Learn locally on our notebook, we now need to stage the data to be used in SageMaker managed training.

When using SageMaker Managed training, note in the diagram below, that unlike our previous experiment, training would not run on the Jupyter Notebook in itself. Instead, you would be supplying cleaned up data, as stage on an S3 bucket, pointing to a specific algorithm container, and SageMaker will manage procuring the training instance, running the training (in distributed mode, if chosen), de-provisioning of the traingn infrastructure, and persisiting the training output (in form of trained model file(s)) on S3 bucket.

-11:03 2x

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 24 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

#insured_zip, incident_location, and incident_date were extraneous fields and did not help us to solve the ML problem. So these will be removed.

- Arrange columns in order, as required
- Encode columns to numeric values - Here we are taking the Yes (Y) and No (N) values for the field fraud_reported and converting these to 1 (Yes) and 0 (No). This will allow us to do math on these values later.
- Split into training and validation sets - Validation data set is not used during the actual training, and is used to test the performance of the model. In the cell below we randomly select 80% of data as training_data set and 20% as val_data set.

Once the cell is processed you will see the results of building these two data sets with the number of features and records per set.

In [2]:

Slide Type Sub-Slide

```
import pandas as pd
from sklearn.model_selection import train_test_split

#Load raw data from source
#data = pd.read_csv("https://raw.githubusercontent.com/jodb/sparkTestData/master/insurance_claims.csv").drop(columns
data = pd.read_csv("insurance_claims.csv").drop(columns="_c39")

#Remove unnecessary columns
colsToDelete = ["policy_number", "policy_bind_date", "insured_zip", "incident_location", "incident_date"]
for col in colsToDelete:
    del data[col]

#Switch target column beginning of the data frame, as is required by SageMaker Linear Learner algorithm
cols = list(data.columns)
cols = [cols[-1]]+cols[:-1]
data = data[cols]

#Change target columns to have '0' and '1' values only
data['fraud_reported'] = data['fraud_reported'].apply(lambda x: 1 if x=='Y' else 0)

#Encode all other categorical column using one-hot encoding
data = pd.get_dummies(data)
```

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 24 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

```
#Load raw data from source
#data = pd.read_csv("https://raw.githubusercontent.com/jodb/sparkTestData/master/insurance_claims.csv").drop(columns
data = pd.read_csv("insurance_claims.csv").drop(columns="_c39")

#Remove unnecessary columns
colsToDelete = ["policy_number", "policy_bind_date", "insured_zip", "incident_location", "incident_date"]
for col in colsToDelete:
    del data[col]

#Switch target column beginning of the data frame, as is required by SageMaker Linear Learner algorithm
cols = list(data.columns)
cols = [cols[-1]]+cols[:-1]
data = data[cols]

#Change target columns to have '0' and '1' values only
data['fraud_reported'] = data['fraud_reported'].apply(lambda x: 1 if x=='Y' else 0)

#Encode all other categorical column using one-hot encoding
data = pd.get_dummies(data)

#Split transformed dataset into training and validation sets
train_data, val_data = train_test_split(data, test_size=0.2, random_state=30)

print("Training Data : {} Features, {} Records".format(train_data.shape[1], train_data.shape[0]))
print("Validation Data : {} Features, {} Records".format(val_data.shape[1], val_data.shape[0]))
```

```
Training Data : 162 Features, 800 Records
Validation Data : 162 Features, 200 Records
```

Slide Type Slide

Pushing the traning and validation dataset to S3

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 24 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

- The training data set is saved as a csv file (train_data.csv) and placed in the data subfolder for the notebook.
- The validation data set is saved as a csv file (val_data.csv) and placed in the data subfolder for the notebook.
- A new subfoder is created for the model artififacts from the notebook called output. This is where the models created from the notebook are stored.

In [3]:

Slide Type Sub-Slide

```
import io

s3_resource = boto3.resource('s3')

#Upload training data file to S3
csv_buffer = io.StringIO()
train_data.to_csv(csv_buffer, header=False, index=False)
s3_train_data_key = '{}/train_data.csv'.format(s3_data_prefix)
s3_resource.Object(s3_bucket, s3_train_data_key).put(Body=csv_buffer.getvalue())
s3_train_data = 's3://{}/{}'.format(s3_bucket, s3_train_data_key)
print("Training data uploaded to: {}".format(s3_train_data))

#Upload validation data file to S3
csv_buffer = io.StringIO()
val_data.to_csv(csv_buffer, header=False, index=False)
s3_val_data_key = '{}/val_data.csv'.format(s3_data_prefix)
s3_resource.Object(s3_bucket, s3_val_data_key).put(Body=csv_buffer.getvalue())
s3_val_data = 's3://{}/{}'.format(s3_bucket, s3_val_data_key)
print("Validation data uploaded to: {}".format(s3_val_data))

output_location = 's3://{}/{}/output'.format(s3_bucket, s3_root)
print('Model artifacts will be uploaded to: {}'.format(output_location))
```

```
Training data uploaded to: s3://sagemaker-us-east-1-649088000912/insurance_fraud_prediction-mltb/data/train_data.cs
v
Validation data uploaded to: s3://sagemaker-us-east-1-649088000912/insurance_fraud_prediction-mltb/data/val_data.cs
```

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 24 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

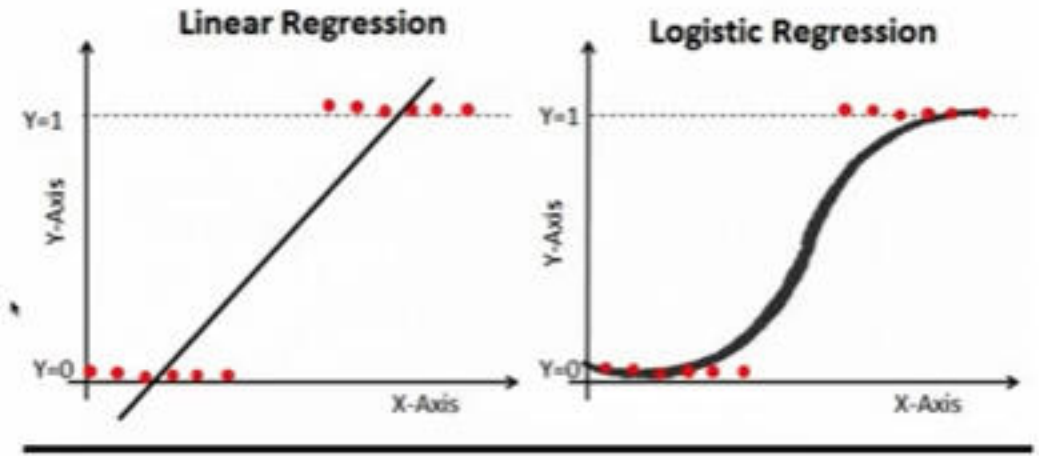
```
Model artifacts will be uploaded to: s3://sagemaker-us-east-1-649088800912/insurance_fraud_prediction-mltb/output
```

Slide Type Slide

Model Selection

Since our initial experiment showed promising results on this dataset using logistic regression within Scikit-Learn, we can implement the same using SageMaker using Linear Learner model with Binary classifier predictor type. **Urbanek Developer Note: Should the last part of this sentence be, "using SageMaker's built-in algorithm Linear Learner with the Binary Classifier predictory_type hyperparameter?**

SageMaker's linear-learner algorithm performs logistic regression assuming a linear separation boundary between two classes.



Urbanek Developer Note: Something feels missing here. Have we explained this thoroughly enough for the target audience?

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 24 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

Obtaining the Docker container image for the linear-learner algorithm and generating unique names for model artifacts

First step therefore is to get hold of the container image for `linear-learner` in the region we are operating.

As a first step, we now need to obtain the container image for the `linear-learner` algorithm in the region we are opperating in.

Each training job also needs a unique name, which we create by appening current timestamp with a name of our choice.

In [4]:

Slide Type Sub-Slide

```
import time
from datetime import datetime
from sagemaker import get_execution_role
from sagemaker.amazon.amazon_estimator import get_image_uri

# Obtain container image for Linear Learner Algorithm
container = get_image_uri(region, 'linear-learner')
print("Algorithm container: {}".format(container))

# Obtain SageMaker execution role that will be used to invoke Sagemaker commands
role = get_execution_role()

# SageMaker client using Python boto3 API
sagemaker = boto3.client('sagemaker')

#Generate unique names for artefacts, with timestamp, for easy tracking
pipeline_name = 'insurance-fraud-predictor'
runtime = datetime.now().strftime("%Y-%m-%d-%H-%M-%S-%f")[:-3]
run_name = "{}-{}".format(pipeline_name, runtime)
```

```
The method get_image_uri has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
Defaulting to the only supported framework/algorithm version: 1. Ignoring framework/algorithm version: 1.
```

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 25 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

```
import time
from datetime import datetime
from sagemaker import get_execution_role
from sagemaker.amazon.amazon_estimator import get_image_uri

# Obtain container image for Linear Learner Algorithm
container = get_image_uri(region, 'linear-learner')
print("Algorithm container: {}".format(container))

# Obtain SageMaker execution role that will be used to invoke Sagemaker commands
role = get_execution_role()

# SageMaker client using Python boto3 API
sagemaker = boto3.client('sagemaker')

#Generate unique names for artefacts, with timestamp, for easy tracking
pipeline_name = 'insurance-fraud-predictor'
runtime = datetime.now().strftime("%Y-%m-%d-%H-%M-%S-%f")[:-3]
run_name = "{}-{}".format(pipeline_name, runtime)
```

```
The method get_image_uri has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
Defaulting to the only supported framework/algorithm version: 1. Ignoring framework/algorithm version: 1.
```

```
Algorithm container: 382416733822.dkr.ecr.us-east-1.amazonaws.com/linear-learner:1
```

Slide Type Slide

Model training

With the image URI readily available, not a single line of Model definition or training code needs to be written. Instead we could follow one of two simple approaches:

1. Create an instance of SageMaker Estimator, which provides a familar programming interface common to most other Machine Language frameworks

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 40 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

```
            true_negative = true_negative + 1
        else:
            false_positive = false_positive + 1
    else:
        if sample_predicted == 1:
            true_positive = true_positive + 1
        else:
            false_negative = false_negative + 1
```

```
Test-12: Target - 0, Predicted - 0
Test-13: Target - 0, Predicted - 0
Test-14: Target - 0, Predicted - 1
Test-15: Target - 0, Predicted - 0
Test-16: Target - 1, Predicted - 1
Test-17: Target - 1, Predicted - 1
Test-18: Target - 0, Predicted - 0
Test-19: Target - 0, Predicted - 0
Test-20: Target - 1, Predicted - 1
Test-21: Target - 0, Predicted - 1
Test-22: Target - 1, Predicted - 1
Test-23: Target - 1, Predicted - 0
Test-24: Target - 0, Predicted - 1
Test-25: Target - 0, Predicted - 0
Test-26: Target - 0, Predicted - 0
Test-27: Target - 0, Predicted - 0
Test-28: Target - 1, Predicted - 0
Test-29: Target - 0, Predicted - 0
Test-30: Target - 0, Predicted - 0
Test-31: Target - 1, Predicted - 1
```

In [12]: Slide Type -

```
print("Test Results:")
print("True Negative = {}".format(true_negative))
print("False Positive = {}".format(false_positive))
print("True Positive = {}".format(true_positive))
```

jupyter Insurance_Fraud_Prediction_SageMaker-Linear_Learner - v2 Last Checkpoint: 40 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

```
print("Test Results:")
print("True Negative = {}".format(true_negative))
print("False Positive = {}".format(false_positive))
print("True Positive = {}".format(true_positive))
print("False Negative = {}".format(false_negative))
```

```
Test Results:
True Negative = 122
False Positive = 32
True Positive = 37
False Negative = 9
```

Slide Type Slide

With the counts of prediction outcome computed, we can use the simple formulas to determine the Precision, Recall and F1 score for the model.

		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative	False Positive
	Positive	False Negative	True Positive

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Amazon SageMaker

- Amazon SageMaker Studio
- Dashboard
- Search
- Images
- Ground Truth
  - Labeling jobs
  - Labeling datasets
  - Labeling workforces
- Notebook
  - Notebook instances
  - Lifecycle configurations
  - Git repositories
- Processing
  - Processing jobs
- Training
  - Algorithms
  - Training jobs
  - Hyperparameter tuning jobs
- Inference
  - Compilation jobs
  - Model packages
  - Models
  - Endpoint configurations
  - Endpoints
  - Batch transform jobs
- Edge Inference
  - Get started
  - Edge packaging jobs
  - Edge device fleets

Amazon SageMaker > Training jobs

Training jobs

Search training jobs

Actions | Create training job

Name	Creation time	Duration	Status
insurance-fraud-predictor-2020-12-20-15-41-22-754	Dec 20, 2020 15:41 UTC	4 minutes	Completed
linear-learner-2020-12-20-11-37-02-295	Dec 20, 2020 11:37 UTC	6 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-10-06-24-228	Dec 20, 2020 10:06 UTC	4 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-10-06-23-446	Dec 20, 2020 10:06 UTC	4 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-10-06-12-389	Dec 20, 2020 10:06 UTC	5 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-10-06-11-819	Dec 20, 2020 10:06 UTC	5 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-10-06-11-504	Dec 20, 2020 10:06 UTC	4 minutes	Completed
demo-xgboost-customer-churn-2020-12-20-09-57-14-462	Dec 20, 2020 09:57 UTC	4 minutes	Completed
demo-xgboost-customer-churn-2020-12-01-15-24-05-581	Dec 01, 2020 15:24 UTC	4 minutes	Completed
demo-xgboost-customer-churn-2020-12-01-15-10-24-030	Dec 01, 2020 15:10 UTC	5 minutes	Completed

```
# BUILD
model1 = linear_model.LogisticRegression(hyperparams)
model2 = DecisionTreeClassifier(hyperparams)

# TRAIN
model1.fit(train_data)
model2.fit(train_data)

# TEST
model1.predict(test_data)
model2.predict(test_data)
```

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Built-in algorithm mode – Python SDK

```
client = boto3.client('sagemaker')
runtime = boto3.client('sagemaker-runtime')

# BUILD
algorithm = get_image_uri(...)
hyperparams = …

# TRAIN
client.create_training_job(
     algorithm, hyperparams, role, train_data, instance_type)

# DEPLOY
client.create_model(model_name, model_url, container)
client.create_endpoint_config(config_name, instance_type)
client.create_endpoint(config_name)
predictions = runtime.invoke_endpoint(test_data)
```

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws training and certification

Many ways to train and deploy a model in SageMaker…

- Jupyter Notebook
  - Any platform: ScikitLearn, …
  - `from sklearn import linear_model`
- Built-in Algorithms
  - SageMaker API: `import sagemaker`
  - AWS SDK: `import boto3`
- Script mode
  managed Docker container
  - 6 supported platforms: TensorFlow, Mxnet, PyTorch, Chainer, Spark, ScikitLearn.
  - `from sagemaker.tensorflow import TensorFlow`
- Docker container
  unmanaged Docker container
- Marketplace
  - Model or Algorithm

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Module 9: Training, Debugging and Monitoring

aws training and certification

- Explain how SageMaker AutoPilot, Experiments Debugger improve ML productivity
- Explain deployment option hosting service, or batch transform
- Identify what to do once a model is trained, specifically the steps required to use inference
- Demo Use case - How to Manage endpoints through the Amazon SageMaker console
- Explain Model Performance with Confusion Matrix
- Identify how to use a production variant for traffic weighting
- Explain how to monitor and log Amazon SageMaker events with Amazon CloudWatch

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker Experiments

aws training and certification

Organize, track, and compare training experiments:

Tracking at scale

Track parameters and metrics across experiments and users

Custom organization

Organize experiments by teams, goals, and hypotheses

Amazon SageMaker Studio File Edit View Run Kernel Git Tabs Settings Help Feedback

Amazon SageMaker Studio | xgboost_customer_churn_stu

Welcome to Amazon SageMaker Studio

Explore models

Explore machine learning content in SageMaker JumpStart to help you build, train, and deploy models quickly.

Search SageMaker JumpStart

Build and train

Spin up sharable Jupyter Notebooks in seconds to build ML models and launch new experiments. Easily organize, track and compare experiments using SageMaker Experiments. Run distributed training, and troubleshoot models with SageMaker Debugger.

Create a notebook

Deploy and monitor

Deploy your models with auto scaling, and automatically monitor for drift in production using SageMaker Model Monitor.

Deploy and monitor models

Build models automatically

Automatically build, train, and tune models with full visibility and control, using SageMaker Autopilot.

Create Autopilot experiment

Import and prepare your data

Import data from Amazon S3, Athena, Redshift, and other data-sources and prepare it for machine learning. SageMaker Data Wrangler allows you to find, import, clean, preprocess, and feature your data to make it ready for model training and deployment.

Create Data Flow

Learn more

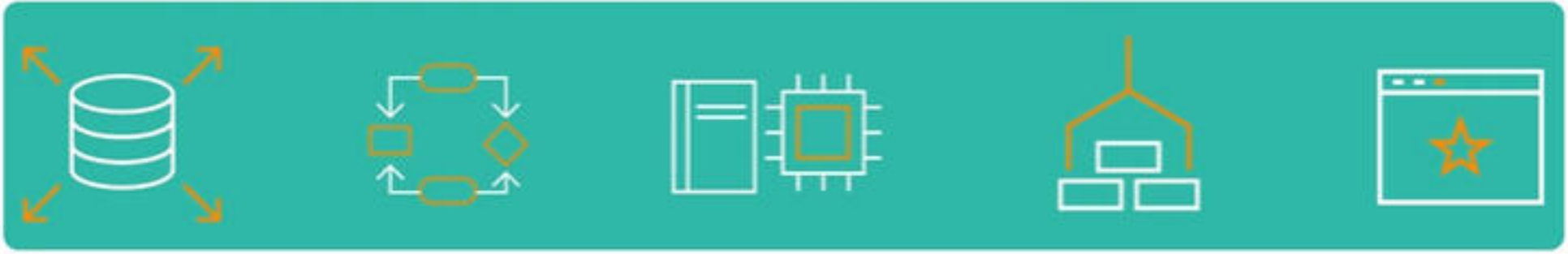
Watch video tutorials to learn more about SageMaker Studio.

Read the end-to-end Studio tour guide with examples.

Amazon SageMaker Debugger

aws training and certification

- Analysis and debugging, explainability, and alert generation



Relevant data capture

Data is automatically captured for analysis

Data analysis and debugging

Analyze and debug data with no code changes

Automatic error detection

Errors are automatically detected based on rules

Improved productivity with alerts

Take corrective action based on alerts

Visual analysis and debugging

Visually analyze and debug from Amazon SageMaker Studio

Hyperparameter model tuning

aws training and certification

Hyperparameters are properties that **control the behavior** of a training algorithm. They are configured before training a model.

Choosing the right hyperparameters has a **significant impact** on model performance.

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Deployment Options

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws training and certification

Deployment options

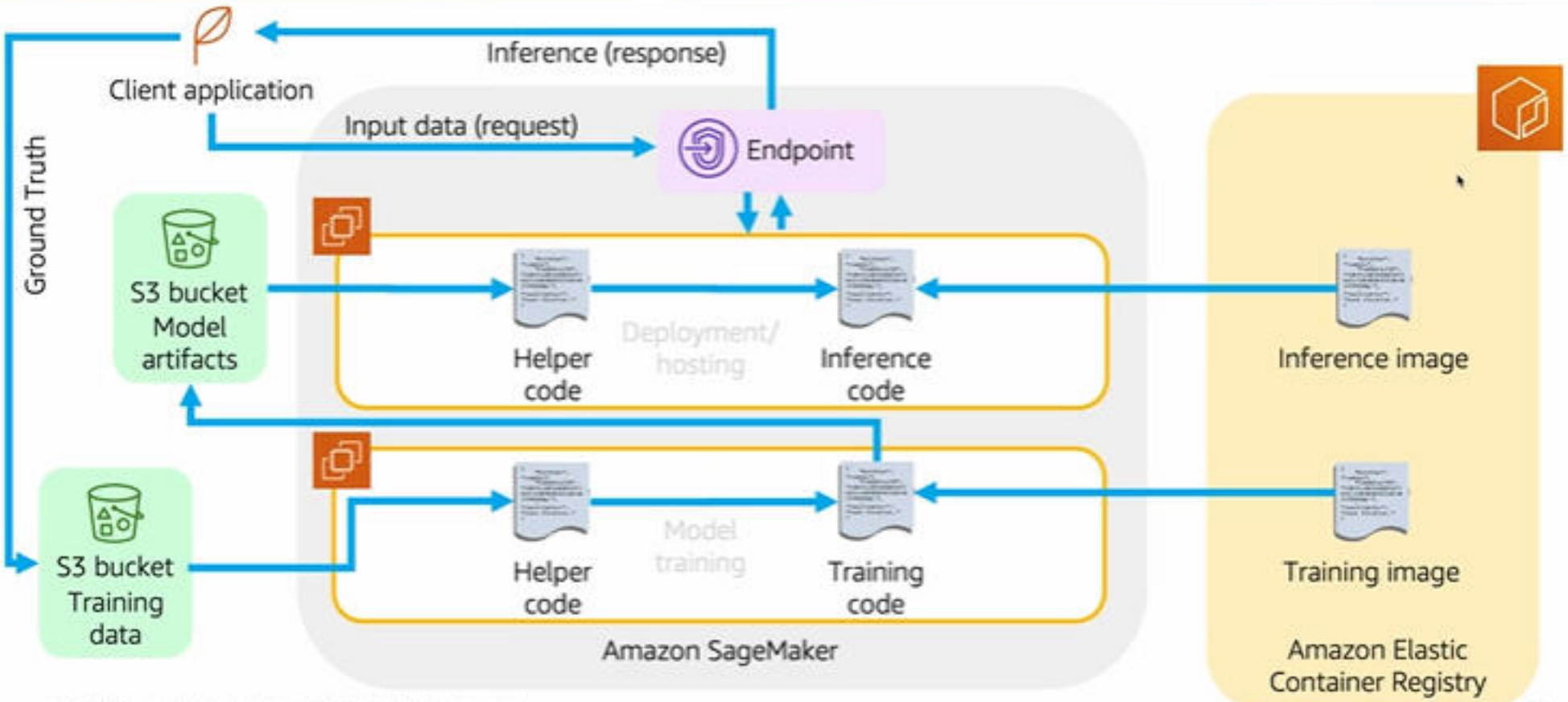
aws training and certification

Amazon SageMaker Hosting Services	Amazon SageMaker Batch Transform
Persistent endpoint	Predictions for an entire dataset
One prediction at a time	One after another

https://docs.aws.amazon.com/sagemaker/latest/dg/how-it-works-deployment.html

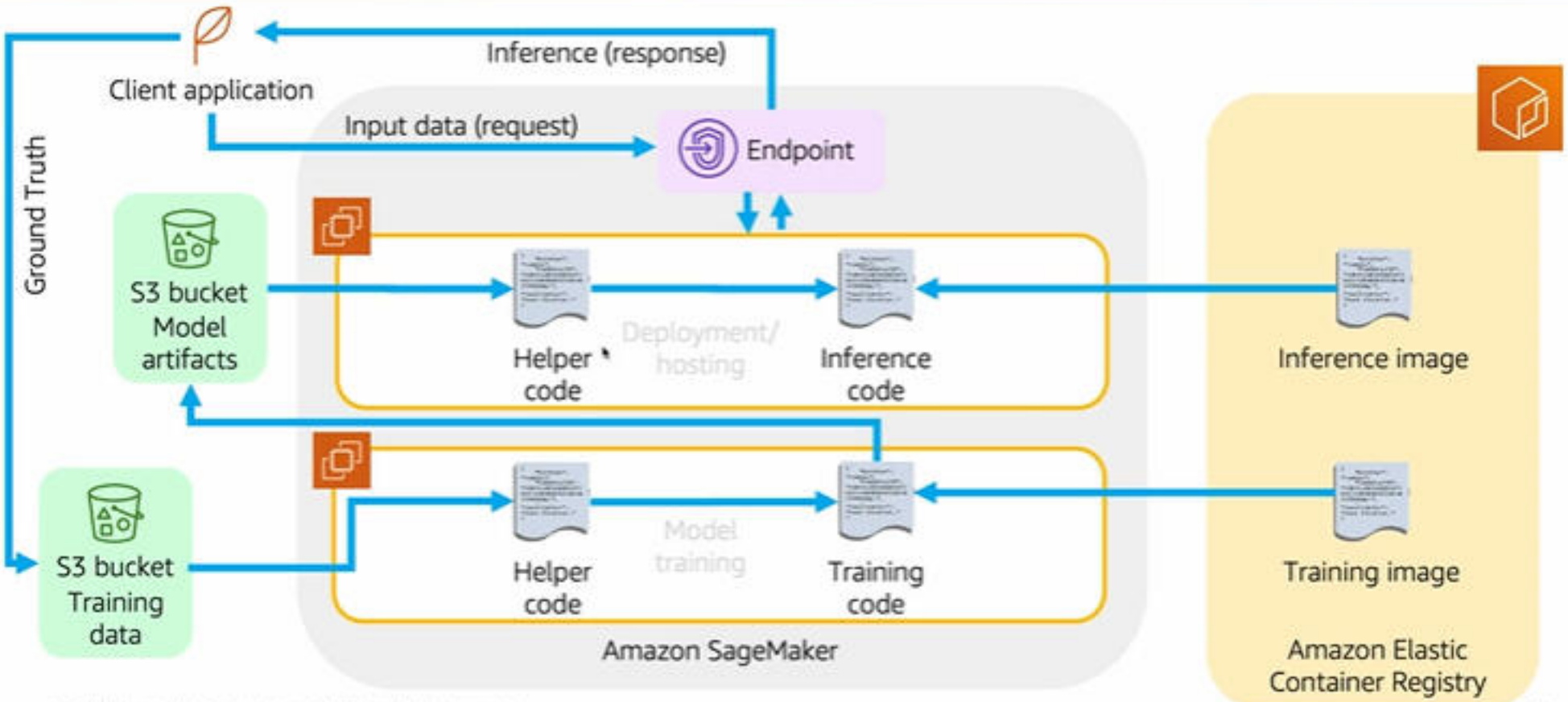
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker hosting services



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker hosting services



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker Model Monitor

aws training and certification

Continuous monitoring of models in production

Automatic data collection

Data is automatically collected from your endpoints

Continuous monitoring

Define a monitoring schedule, and detect changes in quality against a predefined baseline

Flexibility with rules

Use built-in rules to detect data drift, or write your own rules for custom analysis

Visual data analysis

See monitoring results, data statistics, and violation reports in Amazon SageMaker Studio

Integration with Amazon CloudWatch

Automate corrective actions based on CloudWatch alerts

Amazon SageMaker Model Monitor

aws training and certification

Continuous monitoring of models in production

Automatic data collection

Data is automatically collected from your endpoints

Continuous monitoring

Define a monitoring schedule, and detect changes in quality against a predefined baseline

Flexibility with rules

Use built-in rules to detect data drift, or write your own rules for custom analysis

Visual data analysis

See monitoring results, data statistics, and violation reports in Amazon SageMaker Studio

Integration with Amazon CloudWatch

Automate corrective actions based on CloudWatch alerts

Amazon SageMaker Model Monitor

aws training and certification

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Confusion matrix

aws training and certification

Actual	Predicted Negative	Predicted Positive
Negative	True Negative	False Positive
Positive	False Negative	True Positive

Validating model with confusion matrix

aws training and certification

Actual	Predicted Negative	Predicted Positive
Negative	True Negative	False Positive
Positive	False Negative	True Positive

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

$$F1 = 2 \times \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Precision: Spam filtering with 1000 emails

aws training and certification

Precision = 650/[650 + 25] = 0.96

Recall = 650/[650 + 75] = 0.89

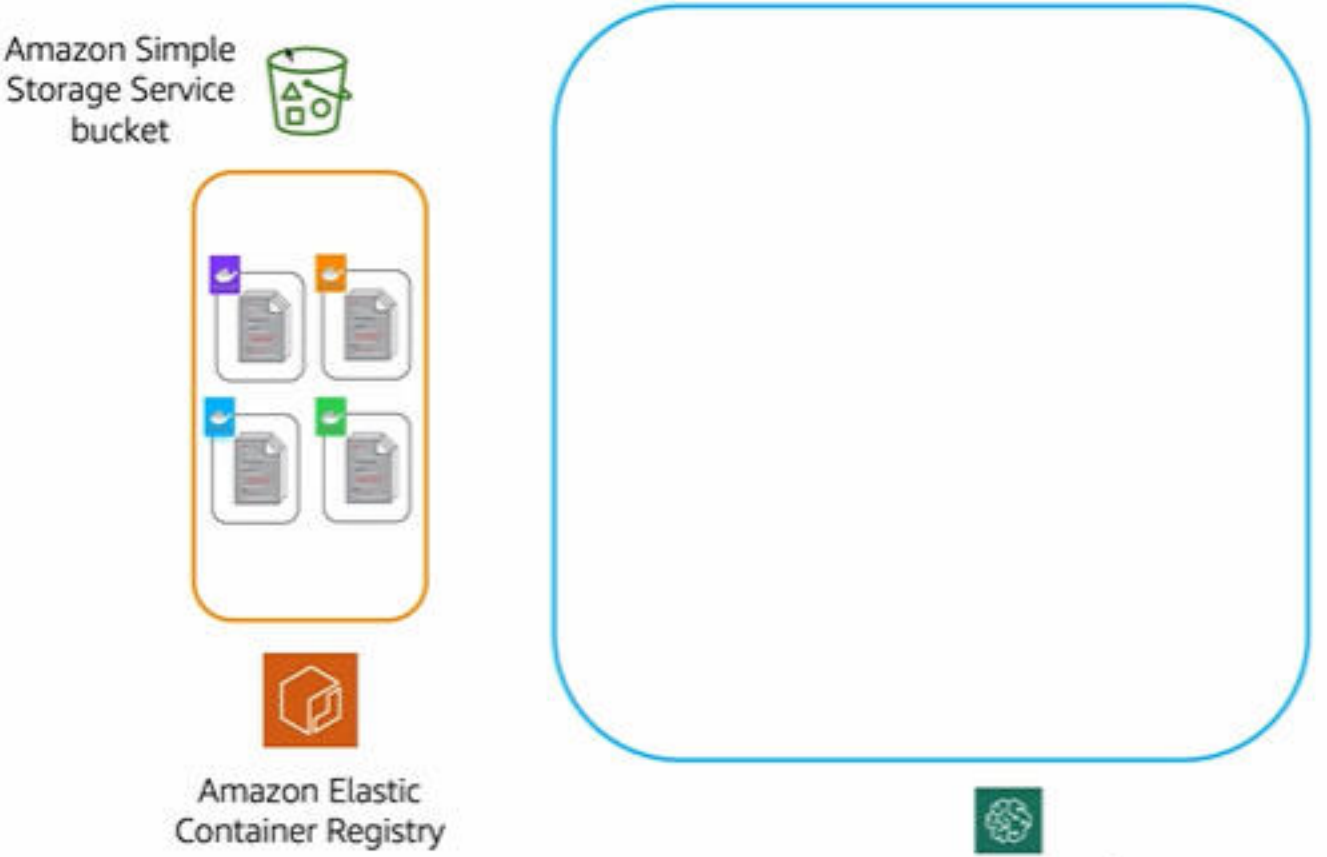
F1 Score = 2 *[0.96 * 0.89]/[0.96+0.89] = 0.94

Production variants (traffic weighting)

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

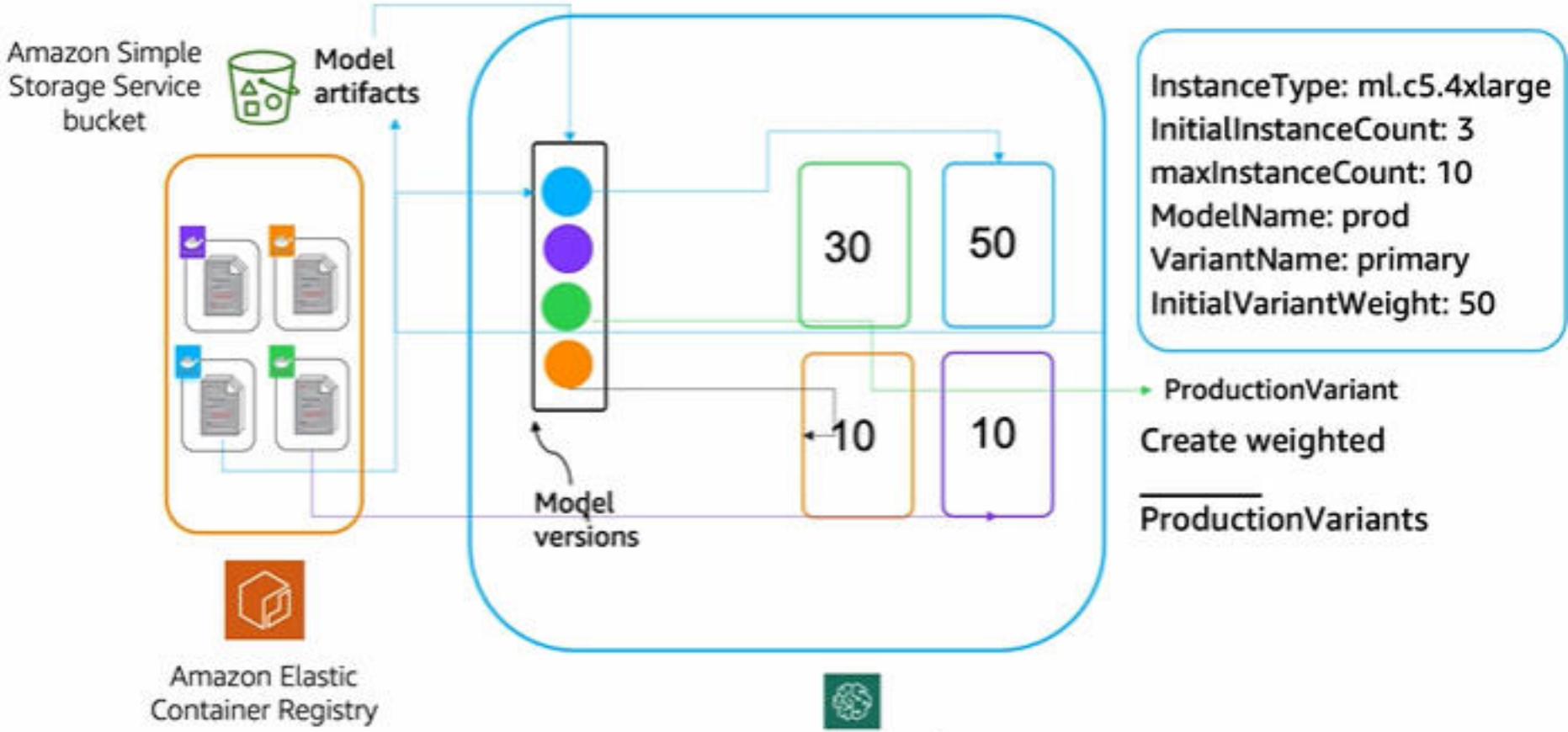
aws training and certification

Model deployment to Amazon SageMaker



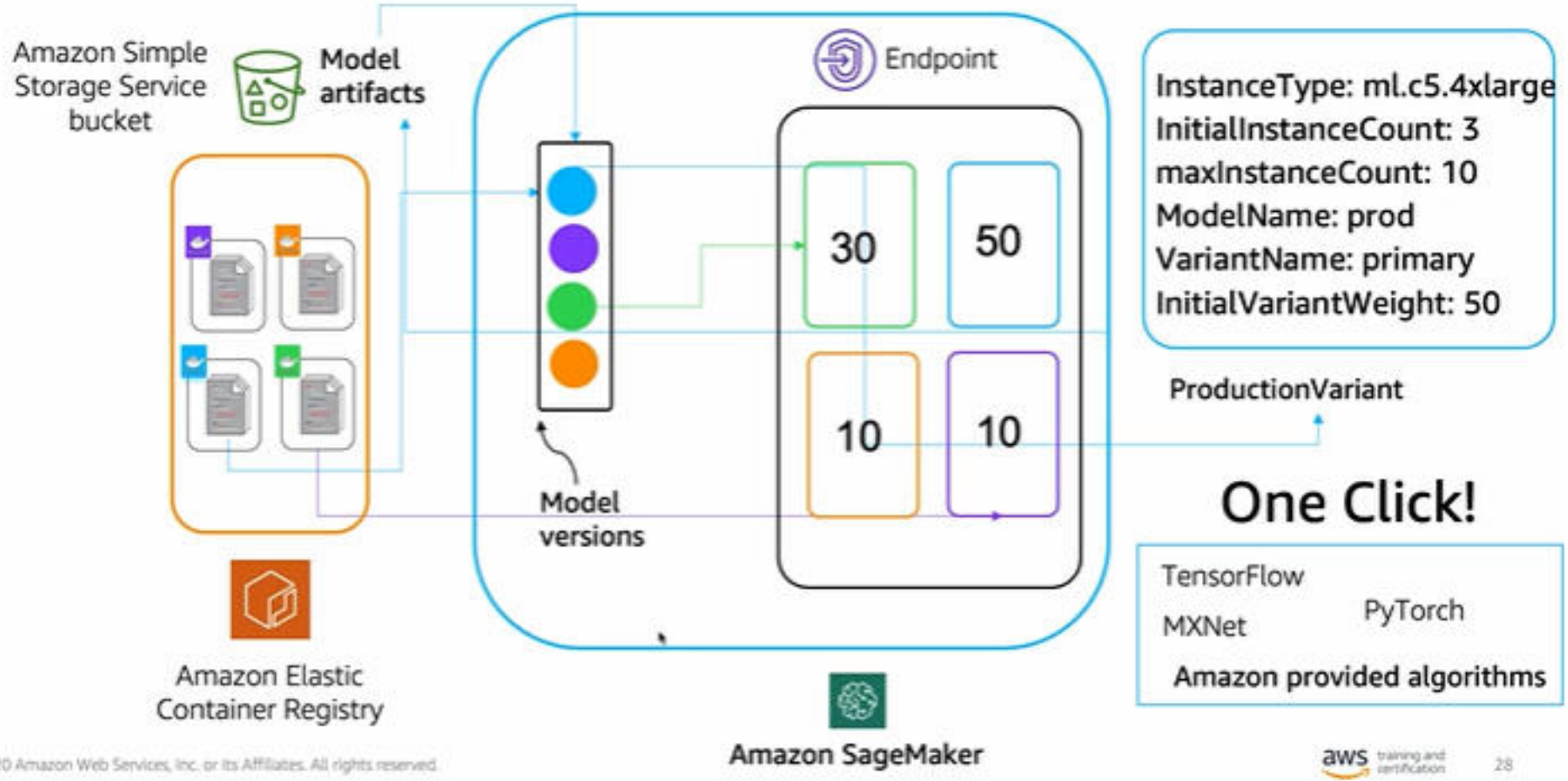
© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Model deployment to Amazon SageMaker



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Model deployment to Amazon SageMaker



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Production variant defined

aws training and certification

```
# Get the current endpoint configuration
endpoint = sage_client.describe_endpoint(EndpointName=xgb_predictor.endpoint)
endpoint_config = sage_client.describe_endpoint_config(
    EndpointConfigName=endpoint['EndpointConfigName'])

# Change the current deployment weight to 0.5 (we'll move 50% of the traffic to
current_model_config = endpoint_config['ProductionVariants'][0]
current_model_config['InitialVariantWeight'] = 0.5
current_model_config['VariantName'] = 'XGBoost'
Variant = 'TunedXGBoost'

tuned_model_config = {'ModelName': model_name,
                      'InitialInstanceCount': 1,
                      'InstanceType': 'ml.m4.xlarge',
                      'VariantName': Variant,
                      'InitialVariantWeight': 0.5}
```

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Production variant defined

aws training and certification

```
# Create the new endpoint configuration
sage_client.create_endpoint_config(
    EndpointConfigName='AB-Config',
    ProductionVariants=[current_model_config,
                        tuned_model_config])

# Update the endpoint
sage_client.update_endpoint(
    EndpointName=endpoint['EndpointConfigName'],
    EndpointConfigName='AB-Config'
)
result = sess.wait_for_endpoint(endpoint['EndpointConfigName'])
```

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Model deployment features of Amazon SageMaker

- ✓ Auto Scaling Inference APIs
- ✓ A/B testing
- ✓ Low latency and high throughput
- ✓ Bring Your Own Model
- ✓ Python SDK

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Module 11: Next Steps and Additional Learning

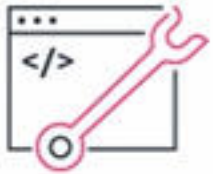
- Identify and describe the AWS Machine Learning Competency Program
- Explain how the AWS Machine Learning Competency Program impacts independent software vendors (ISVs) and consulting partners
- Locate resources for additional learning on ML topics

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML learning paths

aws training and certification

Pick Your Learning Path



Developer

Business Decision Maker

Data Scientist

Data Platform Engineer

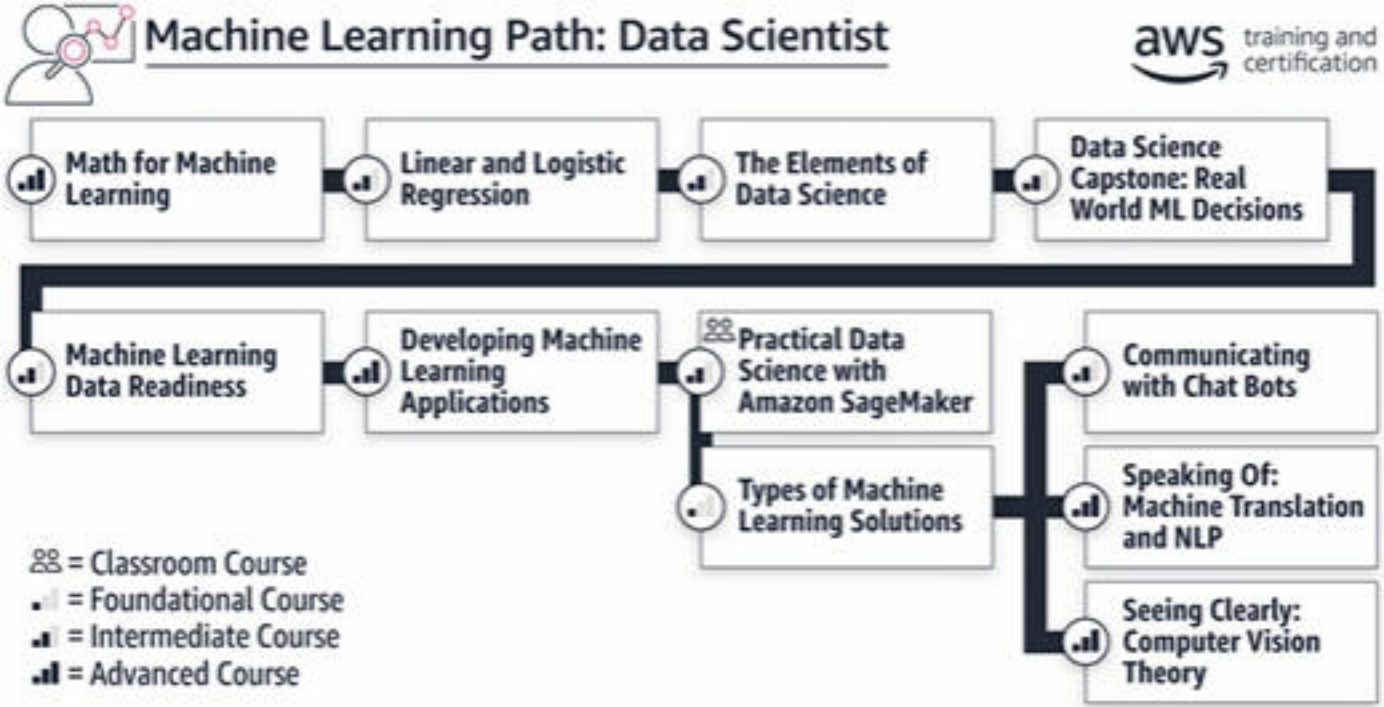
Exam Preparation

Browse all Machine Learning classes

https://pages.awscloud.com/AWS-Partner-Learning-Path-Tool.html

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Learning path: ML for data scientists



https://aws.amazon.com/training/learning-paths/machine-learning

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Machine Learning Certification

https://aws.amazon.com/certification/certified-machine-learning-specialty/

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

AWS lofts

aws training and certification

aws loft NEW YORK CITY

aws loft SAN FRANCISCO

aws pop-up loft JAKARTA

aws | FLOOR28 TEL-AVIV

aws loft TOKYO

AWS Pop-up Loft Munich

https://aws.amazon.com/start-ups/loft

https://aws.amazon.com/start-ups/loft/faq

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Machine Learning Certification

aws training and certification

AWS Certified Machine Learning – Specialty

Validate your ability to build, train, tune, and deploy machine learning models using the AWS Cloud.

Schedule an exam

https://aws.amazon.com/certification/certified-machine-learning-specialty/

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

AWS Machine Learning Competency

aws training and certification

amazon web services | Partner Network

AWS Machine Learning Competency
Technology Partner Validation Checklist

https://aws.amazon.com/blogs/apn/new-look-aws-competency-validation-checklists-for-apn-technology-partners/

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

AWS Competency benefits

aws training and certification

AWS Competency Partner Benefits

In addition to the benefits you receive as an APN member, partners that qualify for AWS Competencies will receive a number of valuable benefits, which may include:

Special Access Benefits | Go-To-Market Benefits | Selective Eligibility | Event Specific Benefits

- Increased discount through the AWS Solution Provider Program
- Shared customer opportunities
- Preferred access to Market Development Funds
- Direct and prioritized access to AWS Partner Network reference team
- AWS Competency spotlight guest series on the APN Blog
- APN Partner spotlight opportunities
- AWS Sales enablement
- Preferred Partner recommendations

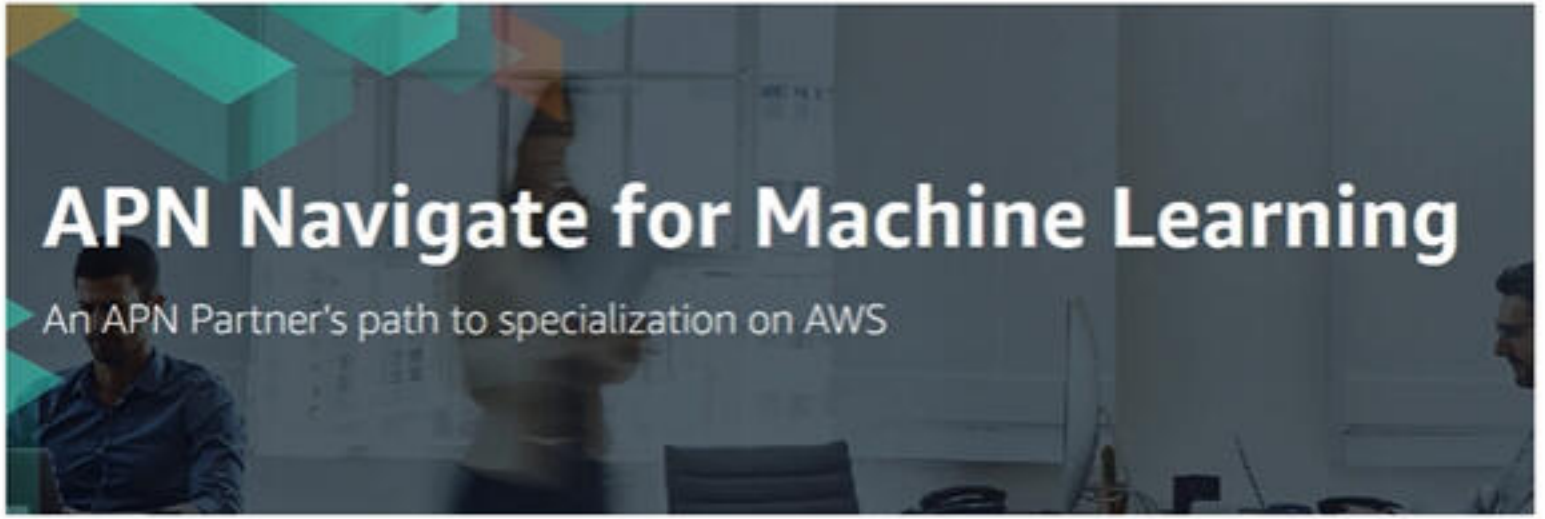
Special Access Benefits

https://aws.amazon.com/partners/competencies/#AWS_Competency_Partner_Benefits

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

APN Navigate for Machine Learning

aws training and certification



https://aws.amazon.com/partners/navigate/machine-learning/

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Resources and references

- Located in your participant guide
- Websites and references
- Please review
- Subject to change

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Book - *Learn Amazon SageMaker*

aws training and certification

A guide to building, training, and deploying machine learning models for developers and data scientists

September 2020

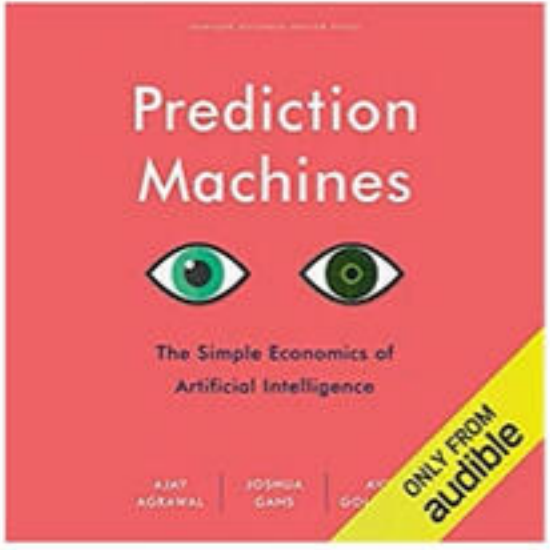
Julien Simon
Principal Advocate, ML/AI

1. Getting Started with Amazon SageMaker
2. Handling Data Preparation Techniques
3. AutoML with Amazon SageMaker AutoPilot
4. Training Machine Learning Models
5. Training Computer Vision Models
6. Training Natural Language Processing Models
7. Extending Machine Learning Services Using Built-In Frameworks
8. Using Your Algorithms and Code
9. Scaling Your Training Jobs
10. Advanced Training Techniques
11. Deploying Machine Learning Models
12. Automating Machine Learning Workflows
13. Optimizing Prediction Cost and Performance

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

https://www.amazon.XX/Learn-Amazon-SageMaker-developers-scientists/dp/180020891X

Further learning



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

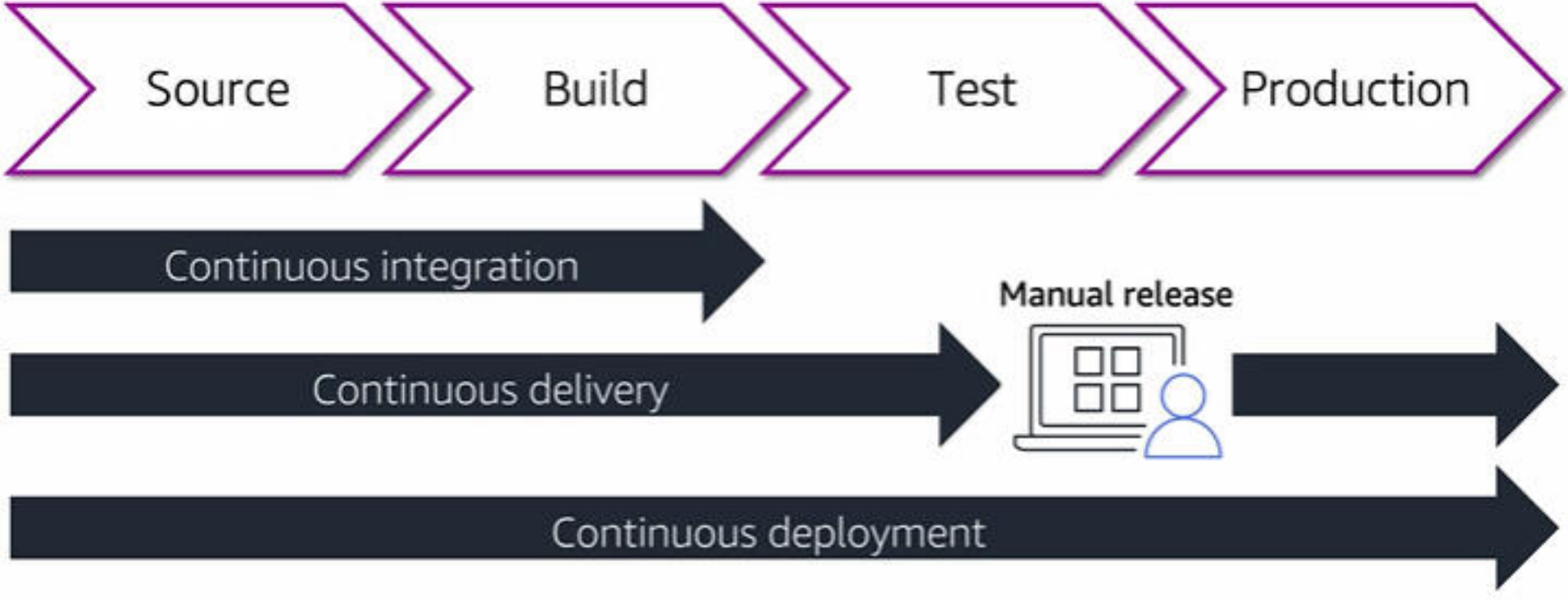
Thank You

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved. This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited. Corrections, feedback, or other questions? Contact us at https://support.aws.amazon.com/#/contacts/aws-training. All trademarks are the property of their owners.

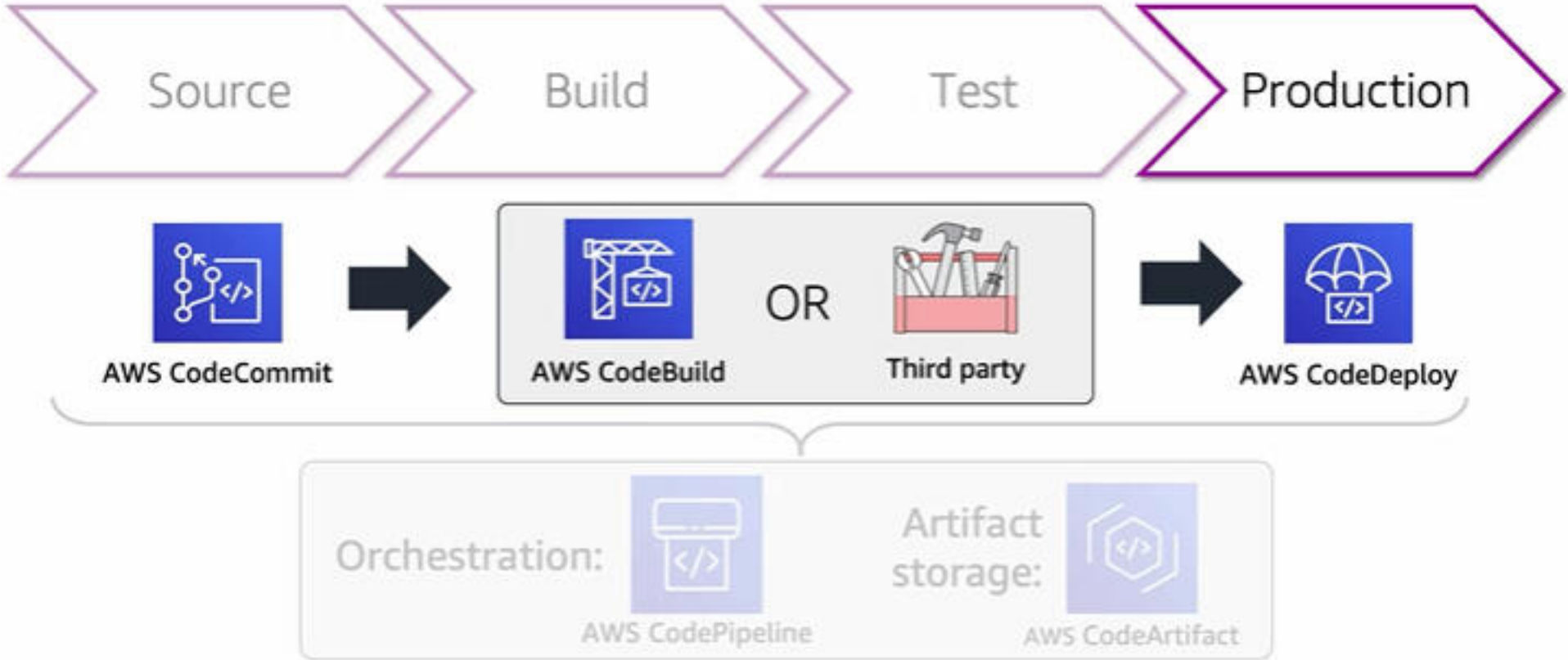
aws training and certification

DevOps and Release Process

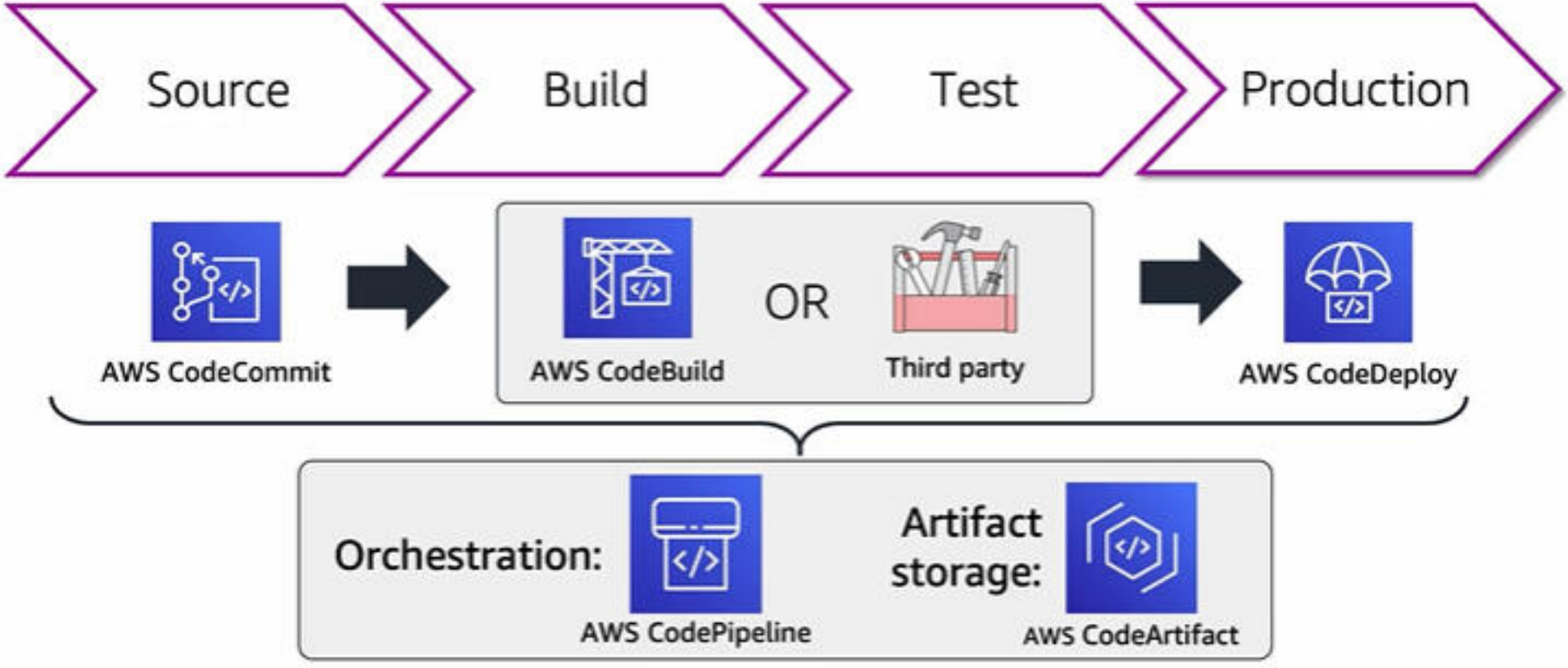
Automated CI/CD release process



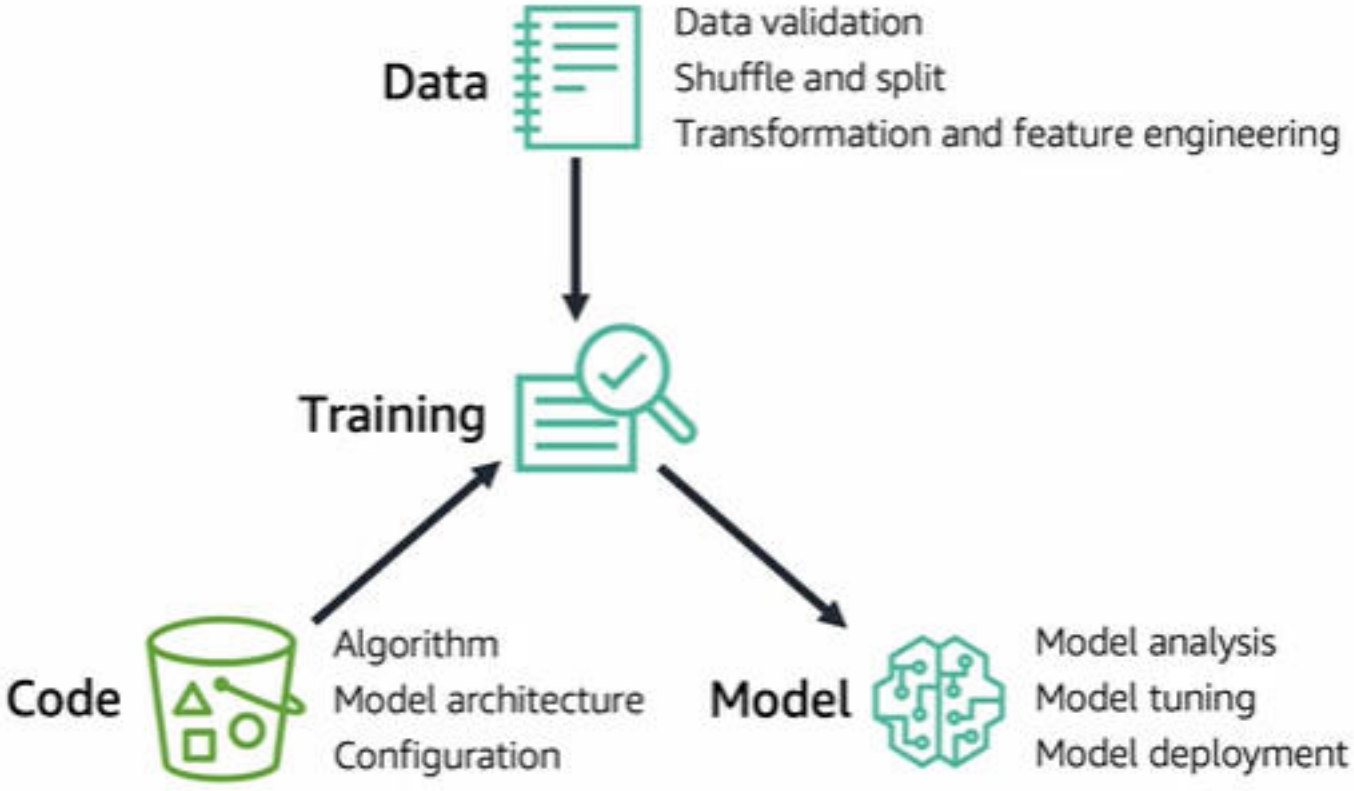
Software release steps



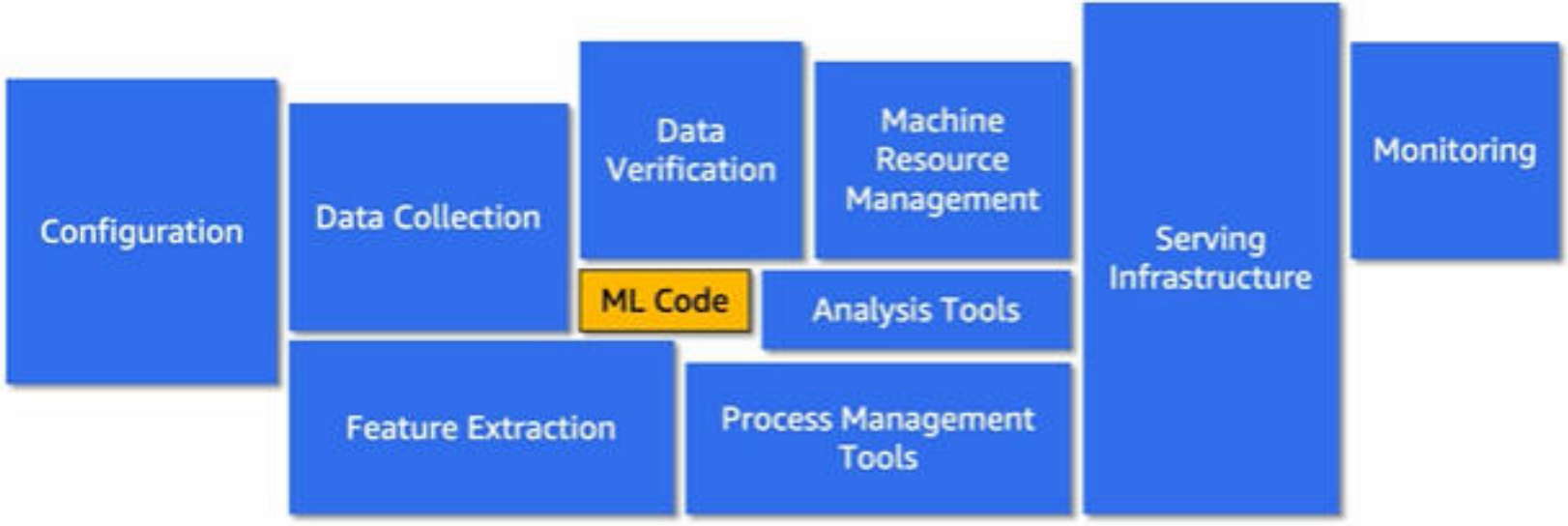
Software release steps



ML code and data are independent



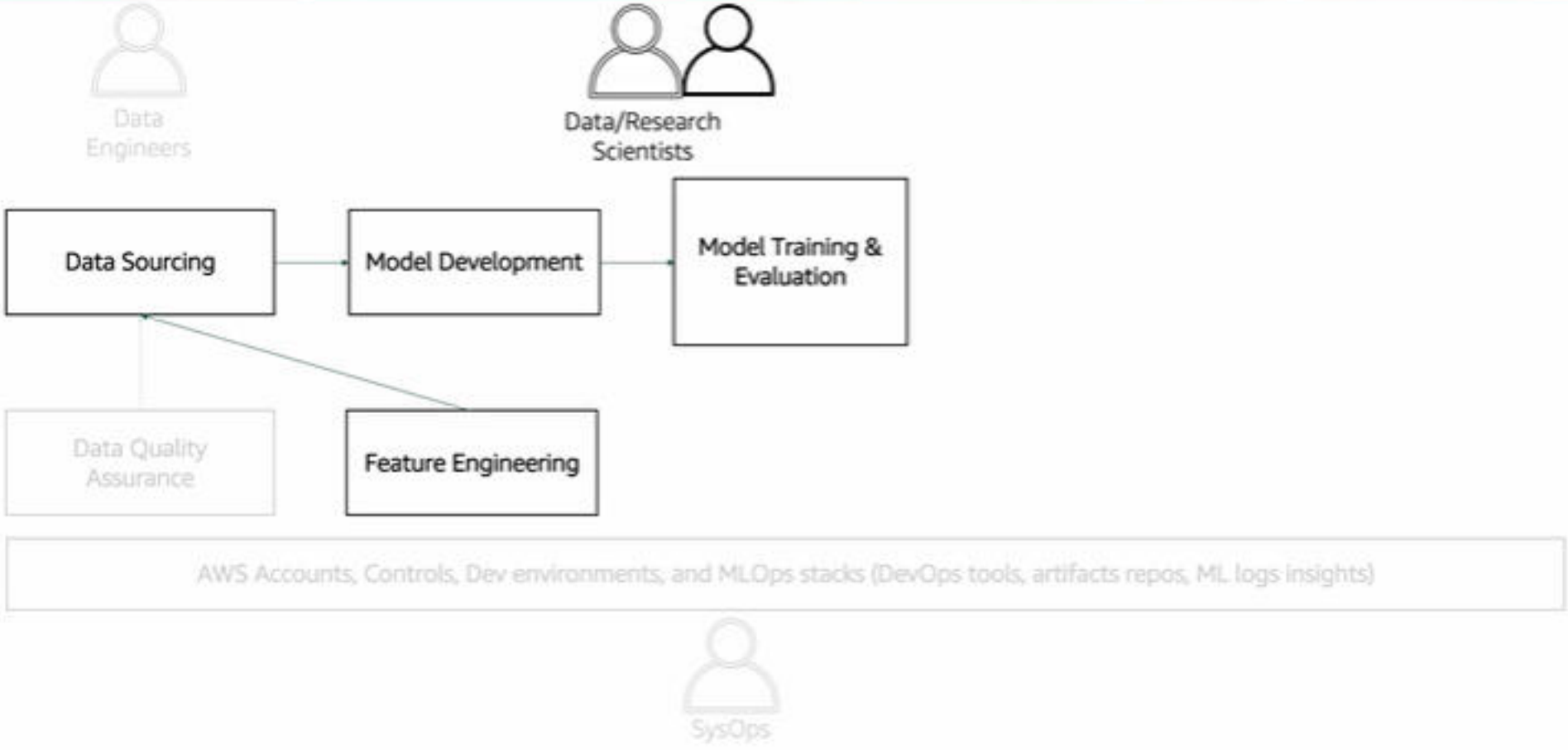
ML code is one small part of the solution



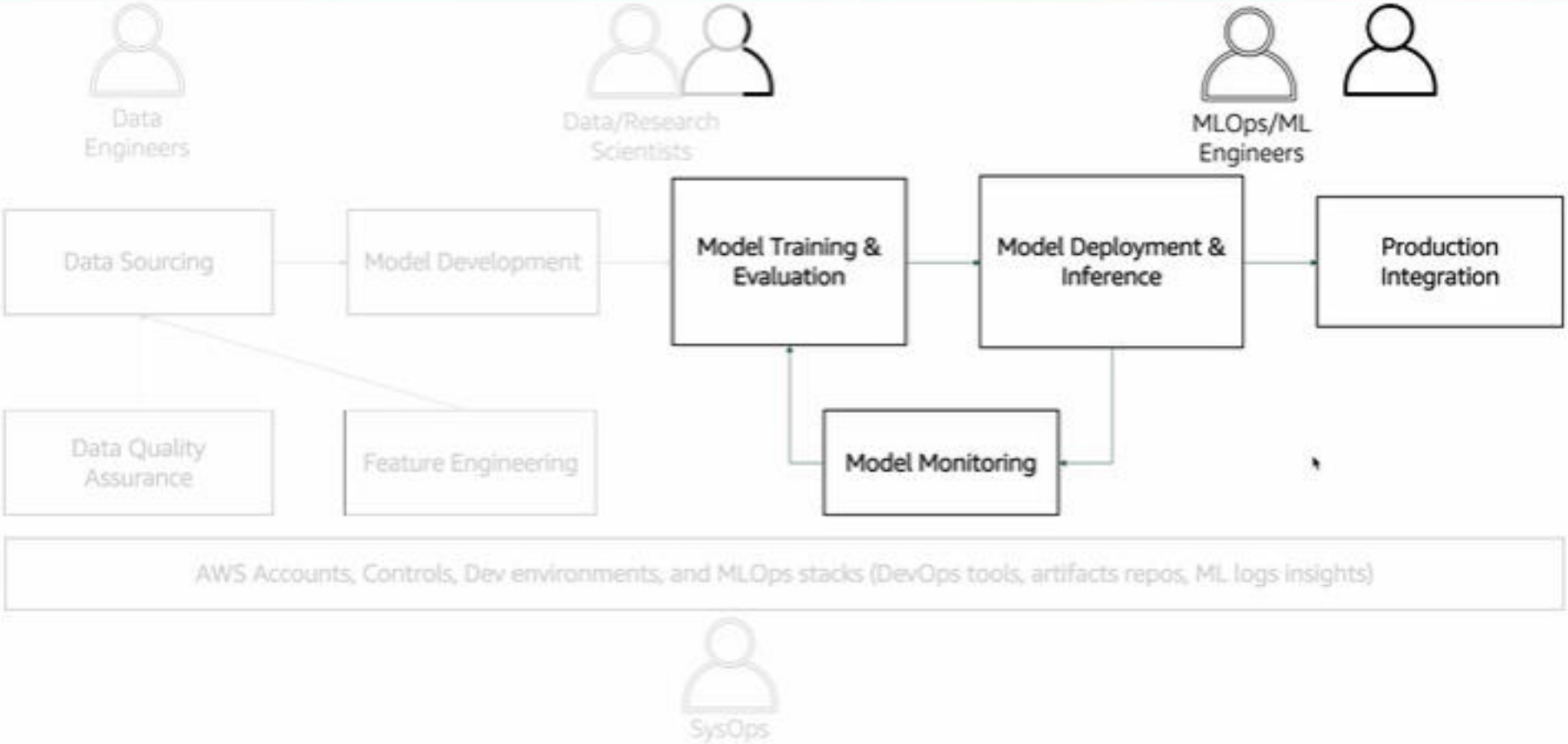
"Hidden Technical Debt in Machine Learning Systems" — Sculley et al.

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Different teams might own part of process



Different teams might own part of process



ML has additional requirements

aws training and certification

- Consistency: Minimal variance between environments (i.e. using containers)
- Flexibility: Can accommodate most frameworks
- Reproducibility: Can recreate past experiments/training
- Reusability: Components are reusable across projects
- Scalability: Able to scale resources to efficiently meet demand
- Auditability: Logs, versions, and dependencies of artifacts are available
- Explainability: Decision transparency

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

MLOps = DevOps for ML

aws training and certification

	DevOps	DevOps for ML
Code versioning	☑	☑
Compute environment	☑	☑
Continuous integration/delivery	☑	☑
Monitoring in production	☑	☑
Data provenance		☑
Datasets		☑
Models		☑
Hyperparameters		☑
Metrics		☑
Workflows		☑

https://medium.com/analytics-vidhya/mlops-the-epoch-of-productionizing-ml-models-4eec06d93623

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

MLOps = DevOps for ML

	DevOps	DevOps for ML
Code versioning	☑	☑
Compute environment	☑	☑
Continuous integration/delivery	☑	☑
Monitoring in production	☑	☑
Data provenance		☑
Datasets		☑
Models		☑
Hyperparameters		☑
Metrics		☑
Workflows		☑

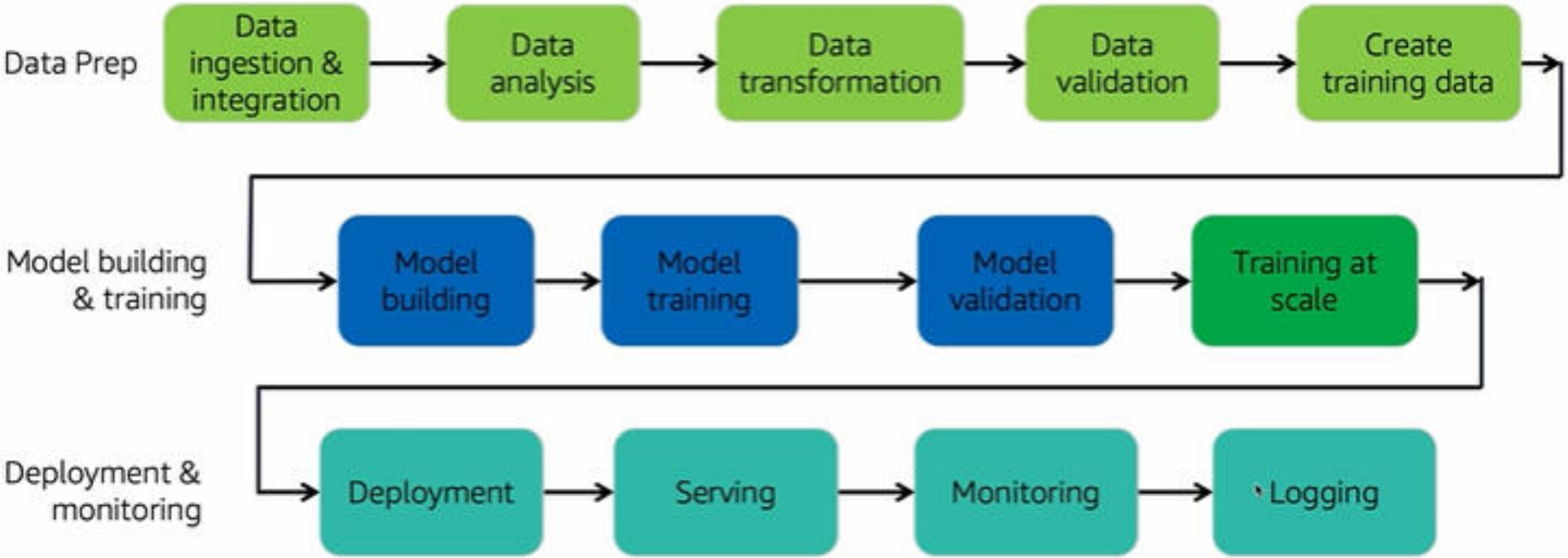
https://medium.com/analytics-vidhya/mlops-the-epoch-of-productionizing-ml-models-4eec06d93623

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML pipeline is more complex

aws training and certification

Phases:



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML pipeline orchestration options

aws training and certification

- Amazon SageMaker Pipeline
- AWS Step Functions and AWS CodePipeline
- Kubeflow: ML toolkit for Kubernetes
- Apache Airflow: Platform to author, schedule, and monitor workflows
- MLflow: Platform to manage the ML lifecycle

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

ML pipeline orchestration options

aws training and certification

- Amazon SageMaker Pipeline
- AWS Step Functions and AWS CodePipeline
- Kubeflow: ML toolkit for Kubernetes
- Apache Airflow: Platform to author, schedule, and monitor workflows
- MLflow: Platform to manage the ML lifecycle

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Automate ML workflows with Amazon SageMaker Pipeline

aws training and certification

Amazon SageMaker Pipelines

aws training and certification

- Orchestrating workflows across each step of the ML process can take months of coding.
- SageMaker Pipeline - purpose-built CI/CD service for machine learning
- With SageMaker Pipelines you
  - Can use Python interface for creating pipelines to automate different steps of the ML workflow, including data loading, data transformation, training and tuning, and deployment.
  - Can build dozens of ML models a week, manage massive volumes of data, thousands of training experiments, and hundreds of different model versions.
  - Can share and re-use workflows to recreate or optimize models, helping you scale ML throughout your organization
  - Can manage dependencies, build correct sequences and automate steps without coding

Amazon SageMaker Pipelines

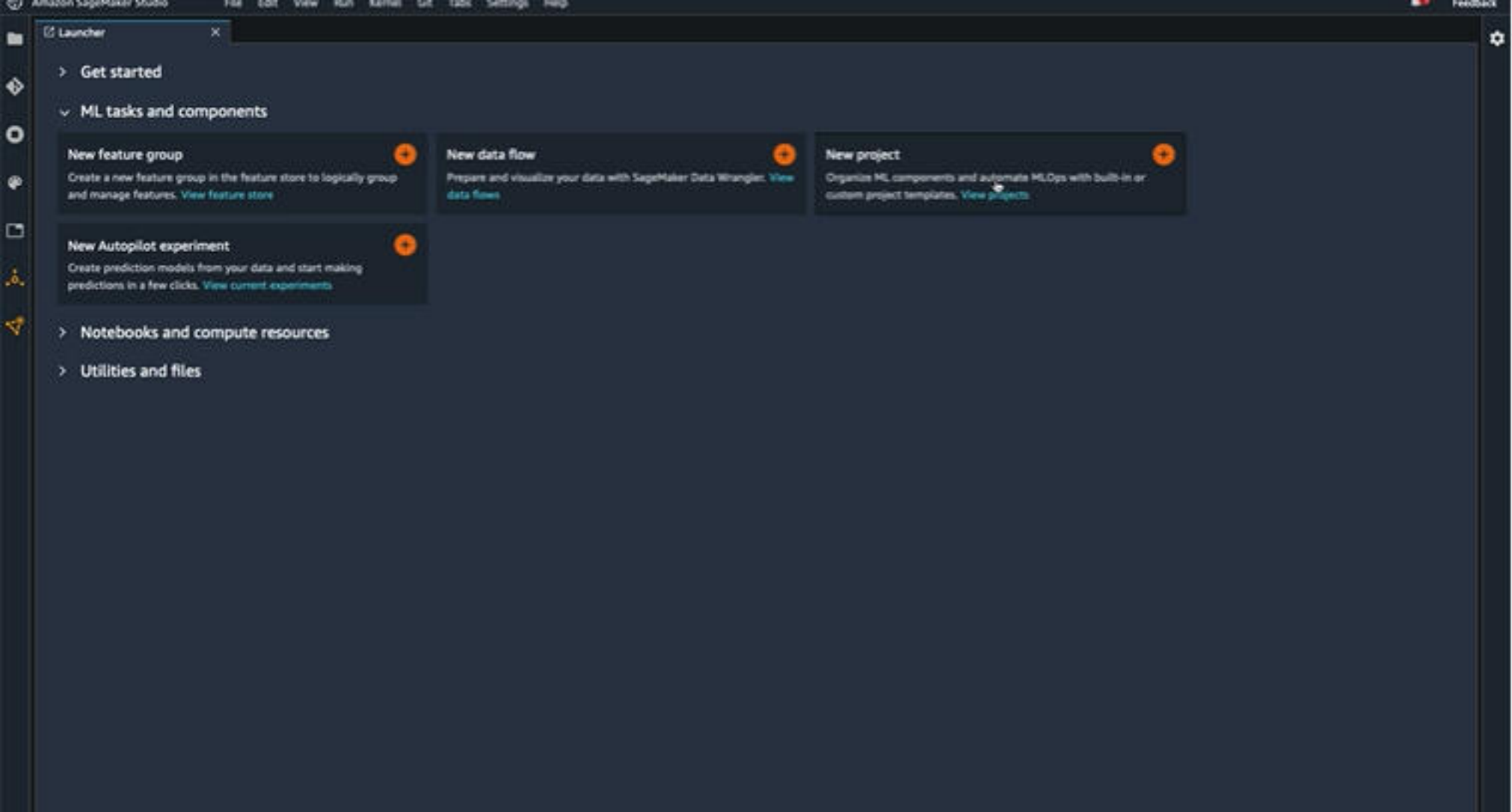
aws training and certification

- Orchestrating workflows across each step of the ML process can take months of coding.
- SageMaker Pipeline - purpose-built CI/CD service for machine learning
- With SageMaker Pipelines you
  - Can use Python interface for creating pipelines to automate different steps of the ML workflow, including data loading, data transformation, training and tuning, and deployment.
  - Can build dozens of ML models a week, manage massive volumes of data, thousands of training experiments, and hundreds of different model versions.
  - Can share and re-use workflows to recreate or optimize models, helping you scale ML throughout your organization
  - Can manage dependencies, build correct sequences and automate steps without coding

Amazon SageMaker Pipeline benefits

aws training and certification

- Compose and manage ML workflows
- Track model lineage for governance and audits
- Replay and re-run workflows
- Visually compare, select, and deploy models
- Access a central registry of trained models
- Fully managed ML Ops with built-in CI/CD support


Amazon SageMaker Studio
File Edit View Run Kernel Git Tabs Settings Help
Feedback
Launcher
Get started
ML tasks and components
New feature group
Create a new feature group in the feature store to logically group and manage features. View feature store
New data flow
Prepare and visualize your data with SageMaker Data Wrangler. View data flows
New project
Organize ML components and automate MLOps with built-in or custom project templates. View projects
New Autopilot experiment
Create prediction models from your data and start making predictions in a few clicks. View current experiments
Notebooks and compute resources
Utilities and files

Amazon SageMaker Studio File Edit View Run Kernel Git Tabs Settings Help

Launcher | Create project | mltb | sagemaker-pipelines-project.

/ mltb-p-ghm304ejpcti / sagemaker-mltb-p-ghm304ejpcti-modelbuild /

Name	Last Modified
img	35 minutes ago
tests	37 minutes ago
pipelines	37 minutes ago
sagemaker-pipelines...	54 minutes ago
tox.ini	37 minutes ago
setup.py	37 minutes ago
setup.cfg	37 minutes ago
codebuild-buildspec....	37 minutes ago
README.md	37 minutes ago
LICENSE	37 minutes ago
CONTRIBUTING.md	37 minutes ago

Markdown Unknown No Kernel Kernel: CPU: --% MEM: --% Share

Orchestrating Jobs, Model Registration, and Continuous Deployment with Amazon SageMaker

Amazon SageMaker offers Machine Learning application developers and Machine Learning operations engineers the ability to orchestrate SageMaker jobs and author reproducible Machine Learning pipelines, deploy custom-build models for inference in real-time with low latency or offline inferences with Batch Transform, and track lineage of artifacts. You can institute sound operational practices in deploying and monitoring production workflows, deployment of model artifacts, and track artifact lineage through a simple interface, adhering to safety and best-practice paradigmsfor Machine Learning application development.

The SageMaker Pipelines service supports a SageMaker Machine Learning Pipeline Domain Specific Language (DSL), which is a declarative Json specification. This DSL defines a Directed Acyclic Graph (DAG) of pipeline parameters and SageMaker job steps. The SageMaker Python Software Developer Kit (SDK) streamlines the generation of the pipeline DSL using constructs that are already familiar to engineers and scientists alike.

The SageMaker Model Registry is where trained models are stored, versioned, and managed. Data Scientists and Machine Learning Engineers can compare model versions, approve models for deployment, and deploy models from different AWS accounts, all from a single Model Registry. SageMaker enables customers to follow the best practices with ML Ops and getting started right. Customers are able to standup a full ML Ops end-to-end system with a single API call.

SageMaker Pipelines

Amazon SageMaker Pipelines support the following activites:

- Pipelines - A Directed Acyclic Graph of steps and conditions to orchestrate SageMaker jobs and resource creation.
- Processing Job steps - A simplified, managed experience on SageMaker to run data processing workloads, such as feature engineering, data validation, model evaluation, and model interpretation.
- Training Job steps - An iterative process that teaches a model to make predictions by presenting examples from a training dataset.
- Conditional step execution - Provides conditional execution of branches in a pipeline.
- Registering Models - Creates a model package resource in the Model Registry that can be used to create deployable models in Amazon SageMaker.
- Creating Model steps - Create a model for use in transform steps or later publication as an endpoint.
- Parametrized Pipeline executions - Allows pipeline executions to vary by supplied parameters.
- Transform Job steps - A batch transform to preprocess datasets to remove noise or bias that interferes with training or inference from your dataset, get inferences from large datasets, and run inference when you don't need a persistent endpoint.

Layout of the SageMaker ModelBuild Project Template

The template provides a starting point for bringing your SageMaker Pipeline development to production.

|— codebuild-buildspec.yml

-23:38 2x

A stubbed testing module for testing your pipeline as you develop:

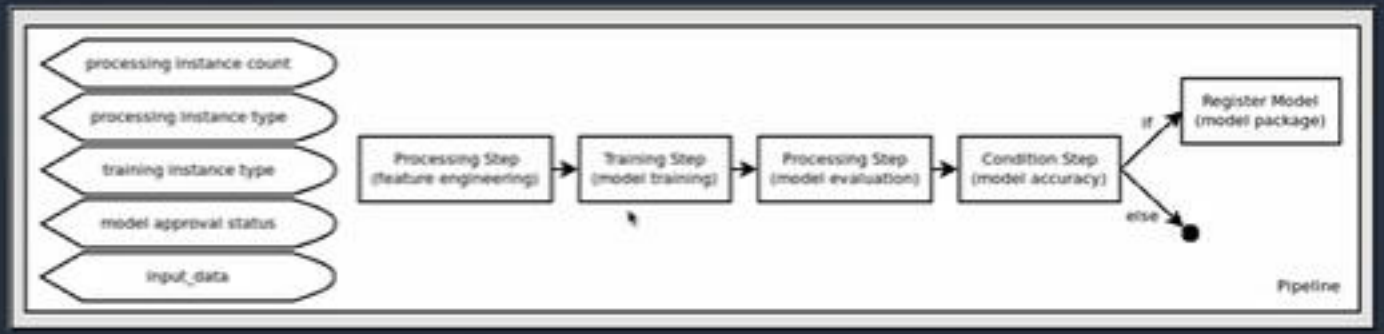
```
|-- tests
|   `-- test_pipelines.py
```

The `tox` testing framework configuration:

```
`-- tox.ini
```

A SageMaker Pipeline

The pipeline that we create follows a typical Machine Learning Application pattern of pre-processing, training, evaluation, and conditional model registration and publication, if the quality of the model is sufficient.



Getting some constants

We get some constants from the local execution environment.

```
[ ]: import boto3
     import sagemaker

     region = boto3.Session().region_name
```

Amazon SageMaker Studio File Edit View Run Kernel Git Tabs Settings Help

Launcher | Create project | mltb | sagemaker-pipelines-project

/ mltb-p-ghm304ejpcti / sagemaker-mltb-p-ghm304ejpcti-modelbuild /

Name	Last Modified
img	35 minutes ago
tests	37 minutes ago
pipelines	37 minutes ago
sagemaker-pipelines_...	54 minutes ago
tox.ini	37 minutes ago
setup.py	37 minutes ago
setup.cfg	37 minutes ago
codebuild-buildspec....	37 minutes ago
README.md	37 minutes ago
LICENSE	37 minutes ago
CONTRIBUTING.md	37 minutes ago

Markdown Unknown No Kernel Kernel: CPU: --% MEM: --% Share

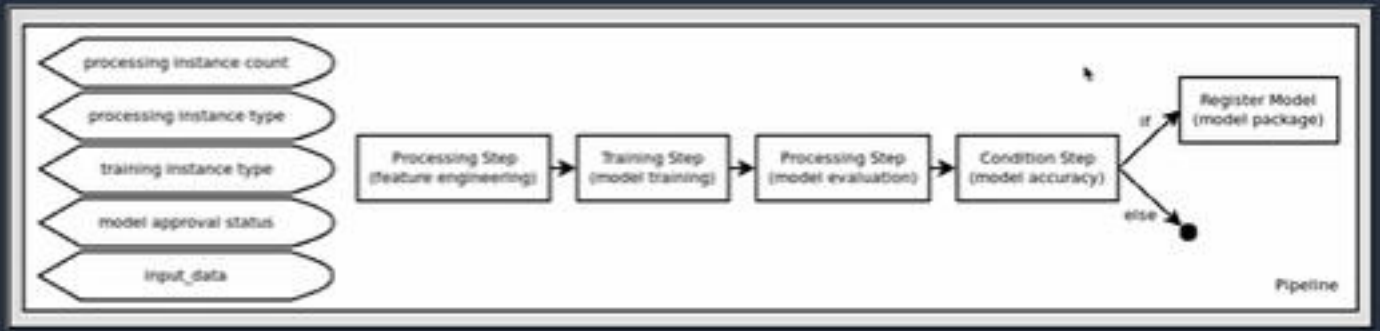
```
|   `-- test_pipelines.py
```

The `tox` testing framework configuration:

```
`-- tox.ini
```

A SageMaker Pipeline

The pipeline that we create follows a typical Machine Learning Application pattern of pre-processing, training, evaluation, and conditional model registration and publication, if the quality of the model is sufficient.



Getting some constants

We get some constants from the local execution environment.

[ ]:
```
import boto3
import sagemaker


region = boto3.Session().region_name
role = sagemaker.get_execution_role()
default_bucket = sagemaker.session.Session().default_bucket()

# Change these to reflect your project/business name or if you want to separate ModelPackageGroup/Pipeline from the rest of your team
model_package_group_name = f"AbaloneModelPackageGroup-Example"
pipeline_name = f"AbalonePipeline-Example"
```

Amazon SageMaker Studio File Edit View Run Kernel Git Tabs Settings Help

Feedback

Components and registries

Select the component or registry to view.

Projects

Search

Create project

Name	Status
mltb	Created

End of the list

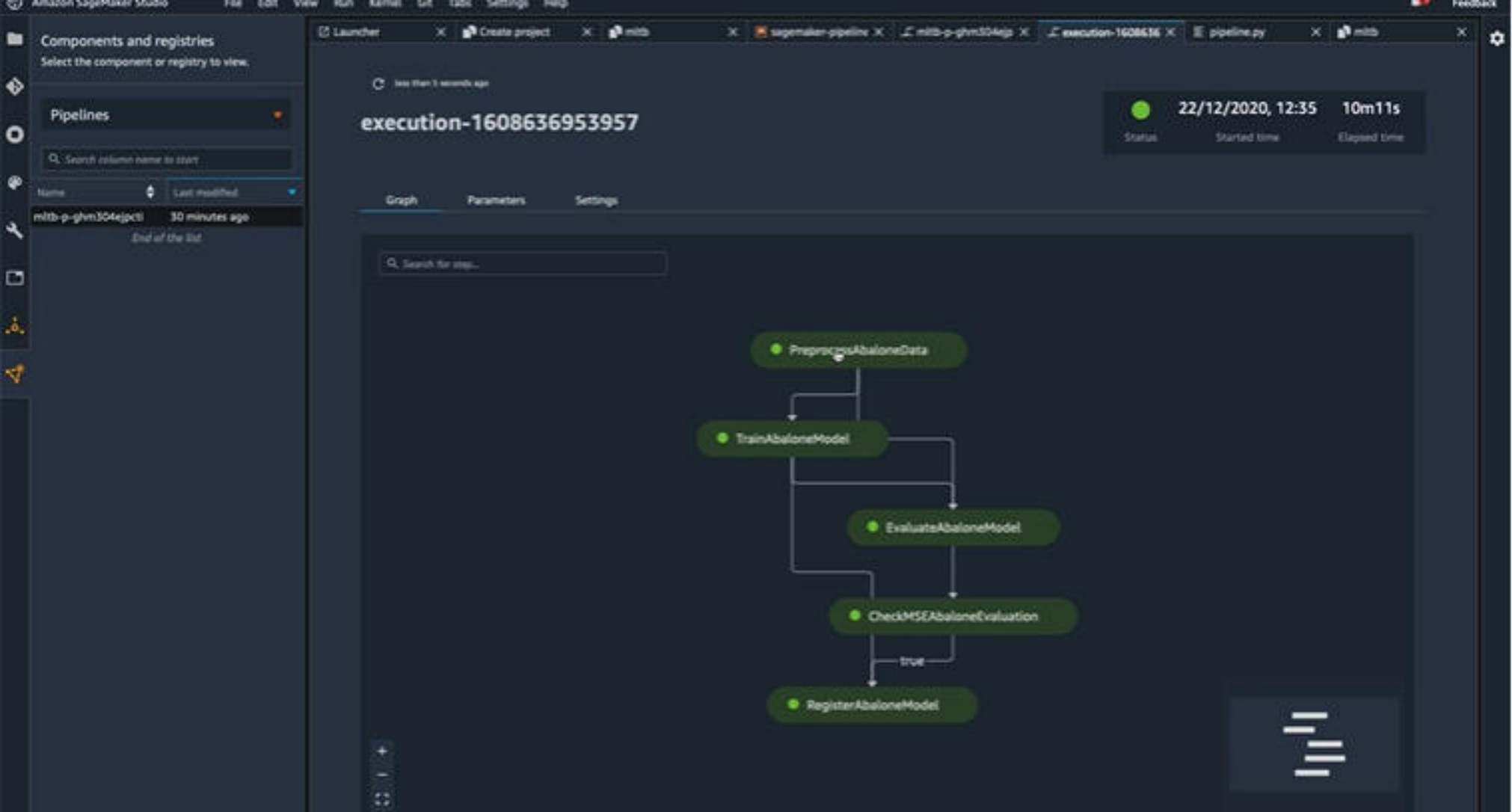
Launcher | Create project | mltb | sagemaker-pipelines-project | pipeline.py

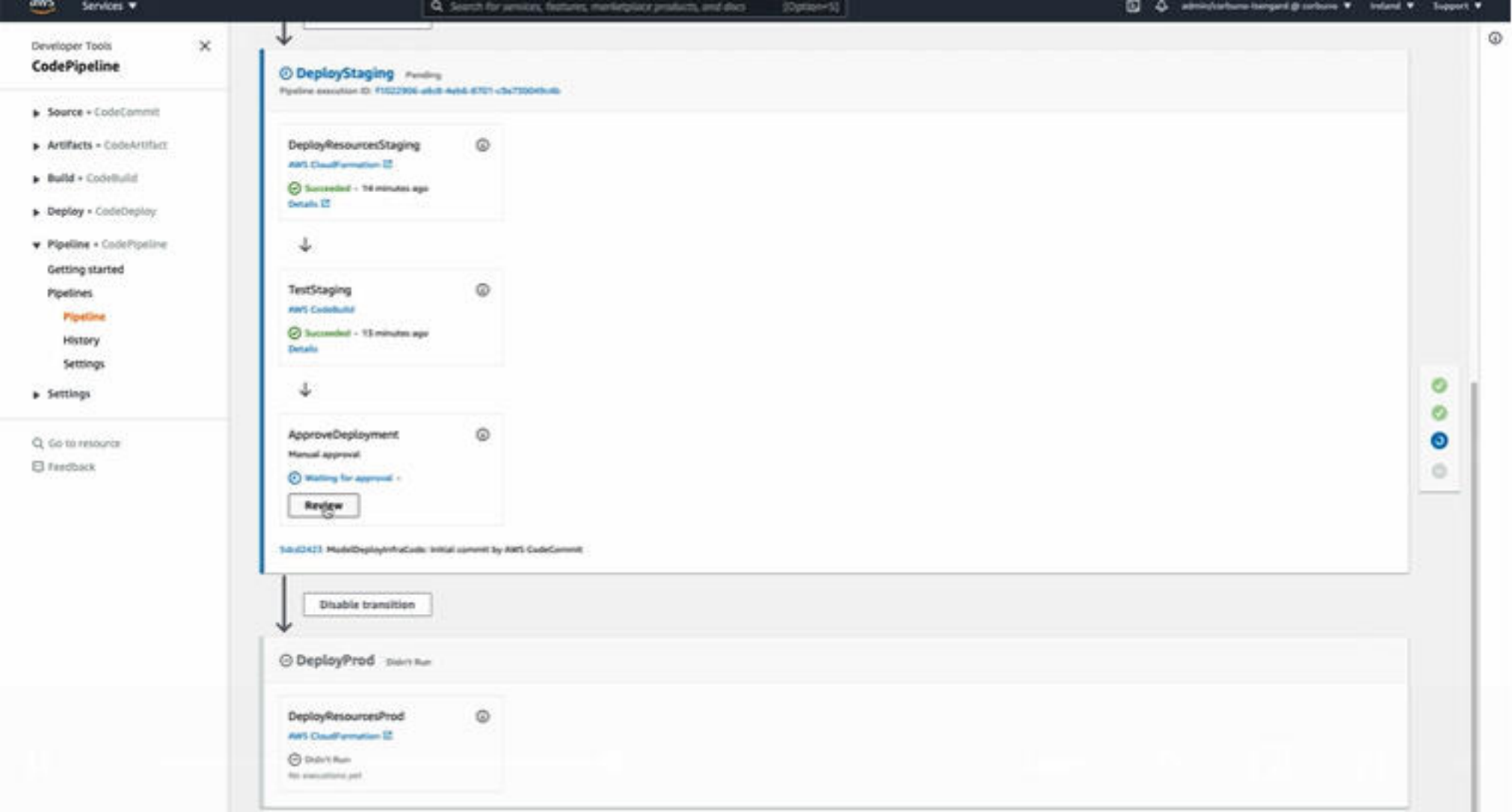
```
        name="ModelApprovalStatus", default_value="PendingManualApproval"
    )
    input_data = ParameterString(
        name="InputDataUrl",
        default_value=f"s3://sagemaker-servicecatalog-seedcode-{region}/dataset/abalone-dataset.csv",
    )

    # processing step for feature engineering
    sklearn_processor = SKLearnProcessor(
        framework_version="0.23-1",
        instance_type=processing_instance_type,
        instance_count=processing_instance_count,
        base_job_name=f"{base_job_prefix}/sklearn-abalone-preprocess",
        sagemaker_session=sagemaker_session,
        role=role,
    )
    step_process = ProcessingStep(
        name="PreprocessAbaloneData",
        processor=sklearn_processor,
        outputs=[
            ProcessingOutput(output_name="train", source="/opt/ml/processing/train"),
            ProcessingOutput(output_name="validation", source="/opt/ml/processing/validation"),
            ProcessingOutput(output_name="test", source="/opt/ml/processing/test"),
        ],
        code=os.path.join(BASE_DIR, "preprocess.py"),
        job_arguments=["--input-data", input_data],
    )

    # training step for generating model artifacts
    model_path = f"s3://{sagemaker_session.default_bucket()}/{base_job_prefix}/AbaloneTrain"
    image_uri = sagemaker.image_uris.retrieve(
        framework="xgboost",
        region=region,
        version="1.0-1",
        py_version="py3",
        instance_type=training_instance_type,
    )
    xgb_train = Estimator(
        image_uri=image_uri,
        instance_type=training_instance_type,
        instance_count=1,
        output_path=model_path,
        base_job_name=f"{base_job_prefix}/abalone-train",
        sagemaker_session=sagemaker_session,
        role=role,
    )
    xgb_train.set_hyperparameters(
        objective="reg:linear",
        num_round=50,
        max_depth=5,
```

-22:04 2x


Components and registries
Select the component or registry to view.
Pipelines
mltb-p-ghm304ejpcti
30 minutes ago
End of the list
execution-1608636953957
22/12/2020, 12:35
10m11s
Graph
Parameters
Settings
PreprocessAbaloneData
TrainAbaloneModel
EvaluateAbaloneModel
CheckMSEAbaloneEvaluation
true
RegisterAbaloneModel


Services
Developer Tools
CodePipeline
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Settings
Go to resource
Feedback
DeployStaging
Pending
DeployResourcesStaging
AWS CloudFormation
Succeeded - 14 minutes ago
Details
TestStaging
AWS CodeBuild
Succeeded - 15 minutes ago
Details
ApproveDeployment
Manual approval
Waiting for approval -
Review
Disable transition
DeployProd
Didn't Run
DeployResourcesProd
AWS CloudFormation
Didn't Run

Amazon SageMaker Studio File Edit View Run Kernel Git Tabs Settings Help

Feedback

/ mltb-p-ghm304ejpcti /

Name	Last Modified
sagemaker-mltb-p-g...	39 minutes ago
sagemaker-mltb-p-g...	41 minutes ago

Launcher | Create pn | mltb | sagemake | mltb-p-gl | execution | mltb | mltb-p-gl | Version 1 | mltb | pipeline.j | mltb

2 minutes ago

mltb

ML Tech Digital demo project

Repositories | Pipelines | Experiments | Model groups | Endpoints | Settings

Endpoints

Search column name to start

Name	Status	Model monitoring schedule	Model monitoring enabled	Last modified
mltb-staging	✓ InService		– Disabled	15 minutes ago

End of the list

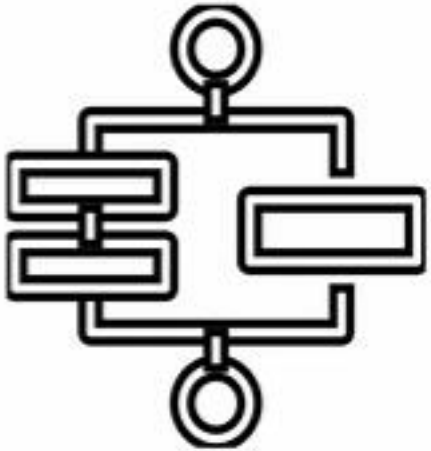
Automate ML workflows with AWS Step Function and AWS CodePipeline

aws training and certification

AWS Step Functions

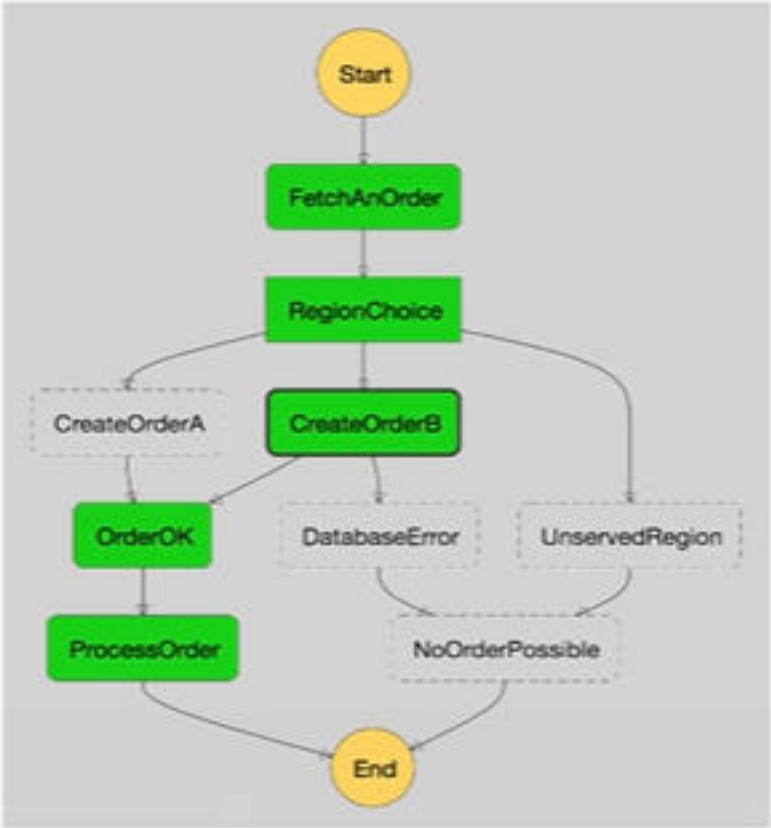
aws training and certification

- Resilient serverless workflows orchestration service
- Build visual workflows: Less code to write and maintain
- Can be used to:
  - Author and visualize your ML pipeline using Python and your Jupyter notebook
  - Leverage AWS CodePipeline for CI/CD



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Easy to create workflow with Step Functions



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

AWS Step Functions: Visual workflows

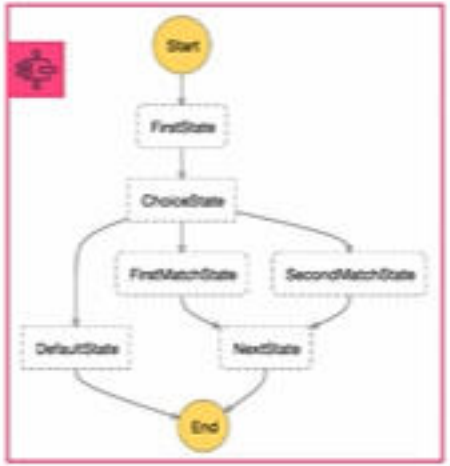
aws training and certification

Define in JSON

Code

```
{
  "Comment": "An AWL example using a choice state.",
  "StartAt": "FirstState",
  "States": {
    "FirstState": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:REGION:ACCOUNT_ID:function:FUNCTION_NAME",
      "Next": "ChoiceState"
    },
    "ChoiceState": {
      "Type" : "Choice",
      "Choices": [
        {
```

Visualize in the Console



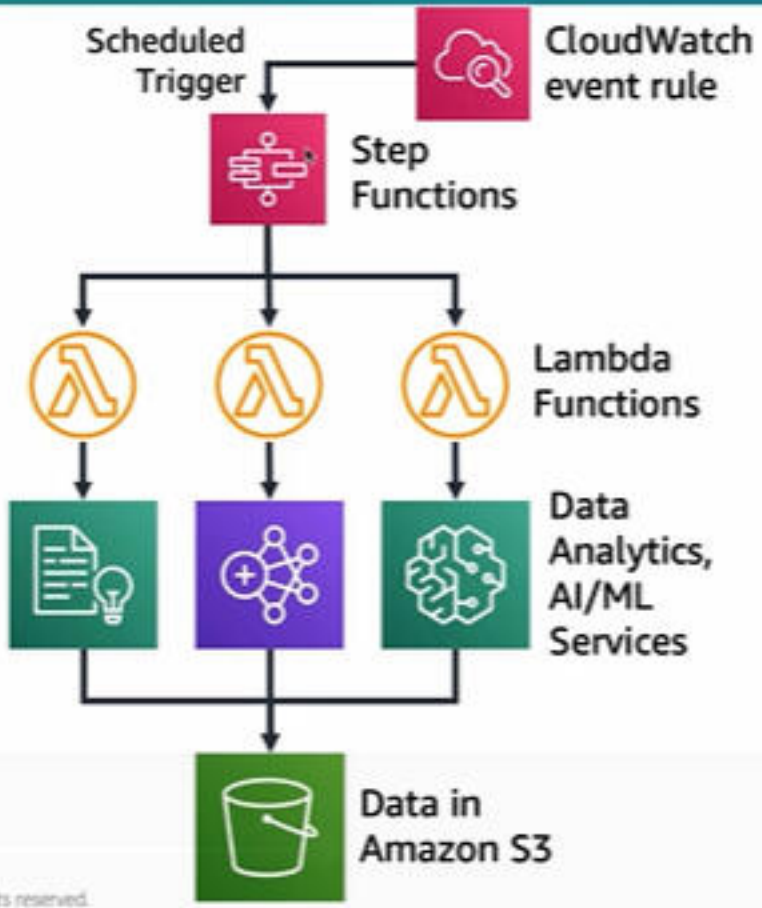
Monitor Run-time

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

AWS Step Functions: Build flexible and repeatable ML workflows

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

AWS Step Functions: Build flexible and repeatable ML workflows



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Enable auditability, reproducibility, traceability, and verifiability

aws training and certification

Code Repo

Data Repo

Model Registry

ML Logs Insights

Enable auditability, reproducibility, traceability, and verifiability

aws training and certification

Code Repo

Versioned Code: Amazon SageMaker

Experiments
- Python notebooks
- Python code

ML Training
- Python code
- Docker files
- BuildSpec Files
- CF Templates/CDK

ML Deployment
- Python code
- Docker files
- BuildSpec files
- CF templates/CDK

Data Repo

Versioned Data: Amazon S3

Data Files
- Raw data
- Processed data
- Labelled data
- Training data
- Validation data
- Inference data (batch)
- Inference feedback

Model Registry

Versioned Containers: Amazon ECR

Docker Images
- Data processing
- Model training
- Model deployment

Uniquely IDed Models: Amazon S3

Model Binaries
- model.tar.gz

Versioned Manifests: Amazon SageMaker

Release
- Container IDs
- Model name
- Model binary path
- Instance type
- Instance count
- Monitoring config
- Data feedback

Build
- Container IDs
- Input data
- Data processing
- Compute
- Training data
- Artefact registry
- Run-time outputs

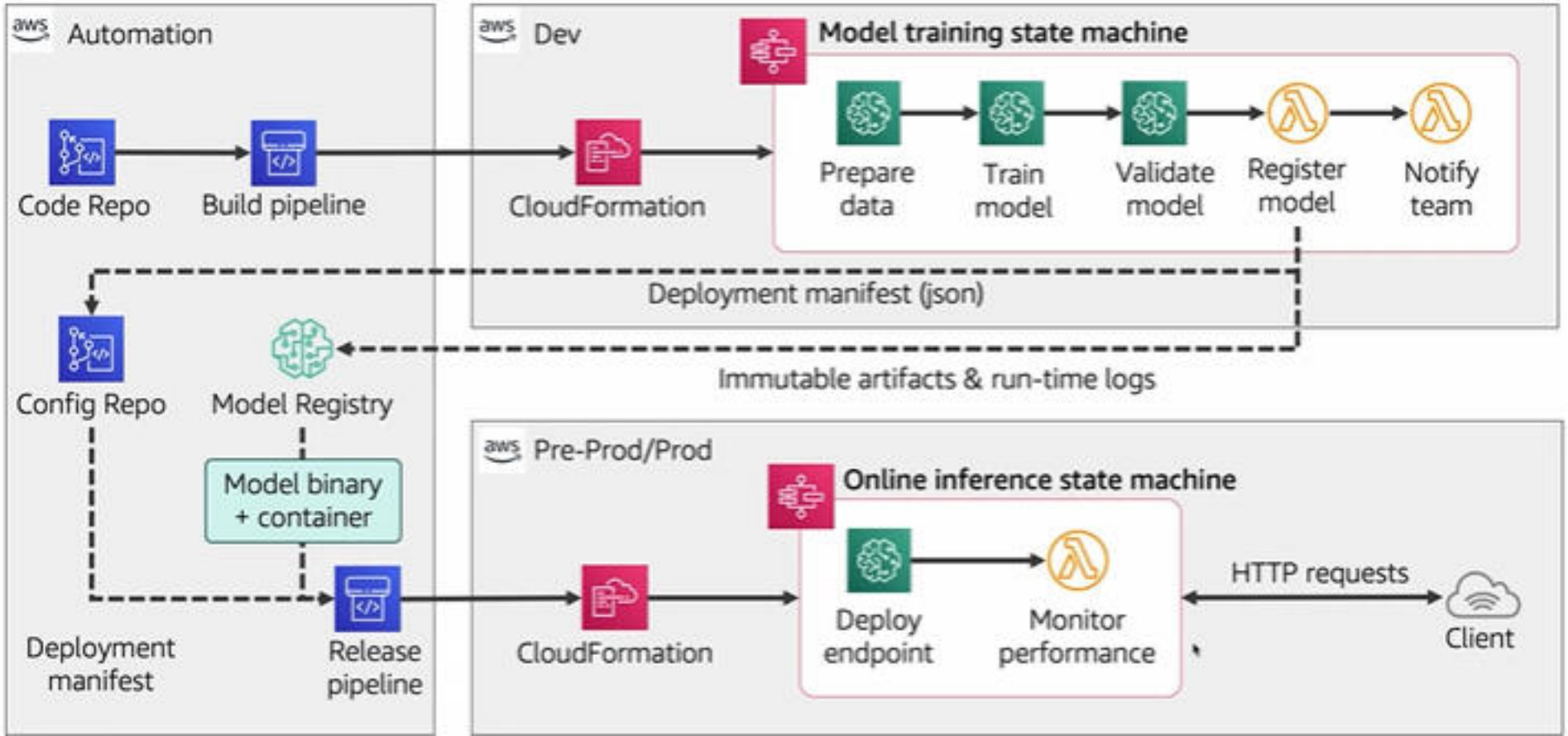
ML Logs Insights

ML Process Logs: Amazon CloudWatch

Logs Store
- Step Functions logs
- Data processing
- Training
- Validation
- Model fingerprint
- Deployment
- Monitoring

Logs Dashboard
- Project dashboard

MLOps pipeline orchestration for automated deployments



Automate ML workflows with Kubeflow

-7:48 2x aws training and certification

Kubeflow: ML Toolkit for Kubernetes

aws training and certification

Kubeflow

Notebook | Pipeline | Training | Serving

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Kubeflow Pipelines

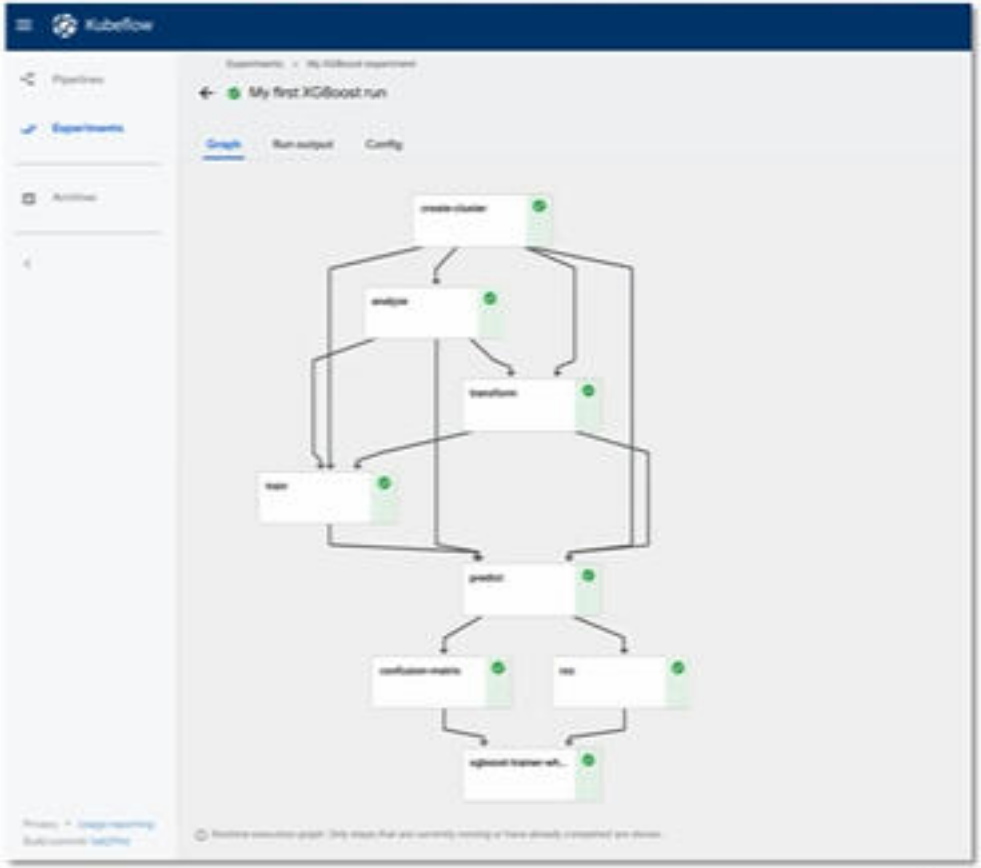
aws training and certification

You can use Kubeflow Pipelines platform to compose, deploy, and manage end-to-end Kubernetes ML workflows

Leverage Kubeflow Pipelines SDK

- `kfp.compiler, kfp.components, kfp.client`

Uses Argo under the hood to orchestrate resources

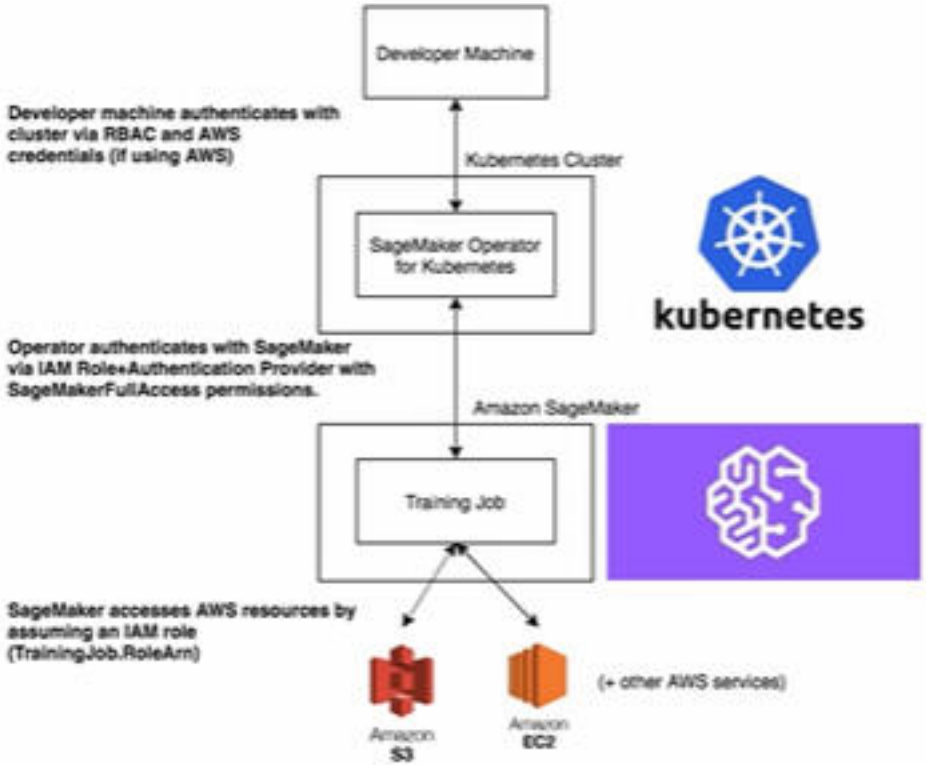


© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Amazon SageMaker Operators for Kubernetes

aws training and certification

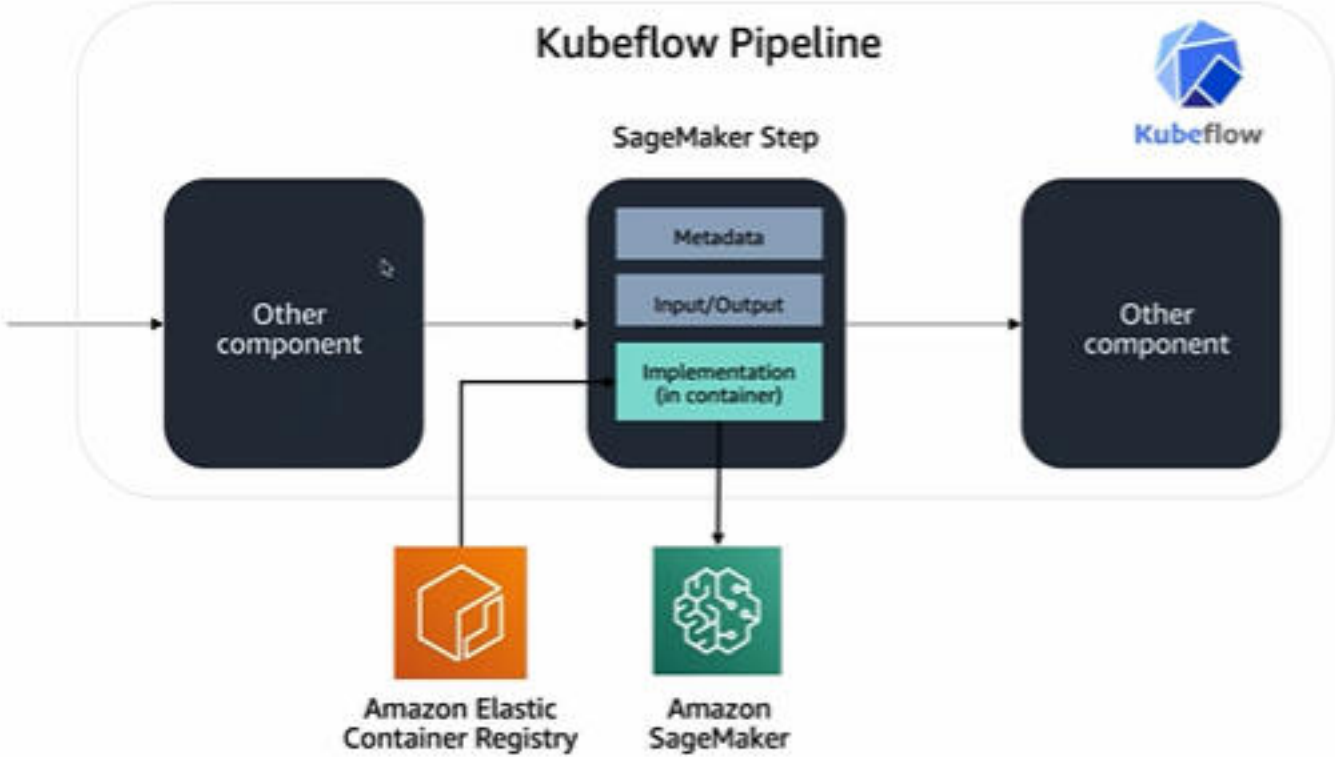
Authentication Layers in the SageMaker Operator for Kubernetes



© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Running SageMaker – pipeline components

aws training and certification



Supported components

- Training
- Model generation
- Hyperparameter tuning
- Model deployment
- Batch transform

https://github.com/kubeflow/pipelines/tree/master/components/aws/sagemaker

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Governance at Scale for ML on AWS

Enable self-service, governed ML development environments

aws training and certification

Balancing the needs of ML builders and central cloud IT

aws training and certification

ML Builders
Stay agile

Innovate with the speed and agility of AWS
Self-service access
Experiment fast
Respond quickly to change

Cloud IT
Establish governance

Govern at scale with central controls
Security
Compliance
Operations
Spend management

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Enable MLOps/development teams to innovate at speed while ensuring governance

aws training and certification

MLOps team

Discover products

Launch governed resources

AWS Service Catalog Administrator

Author templates

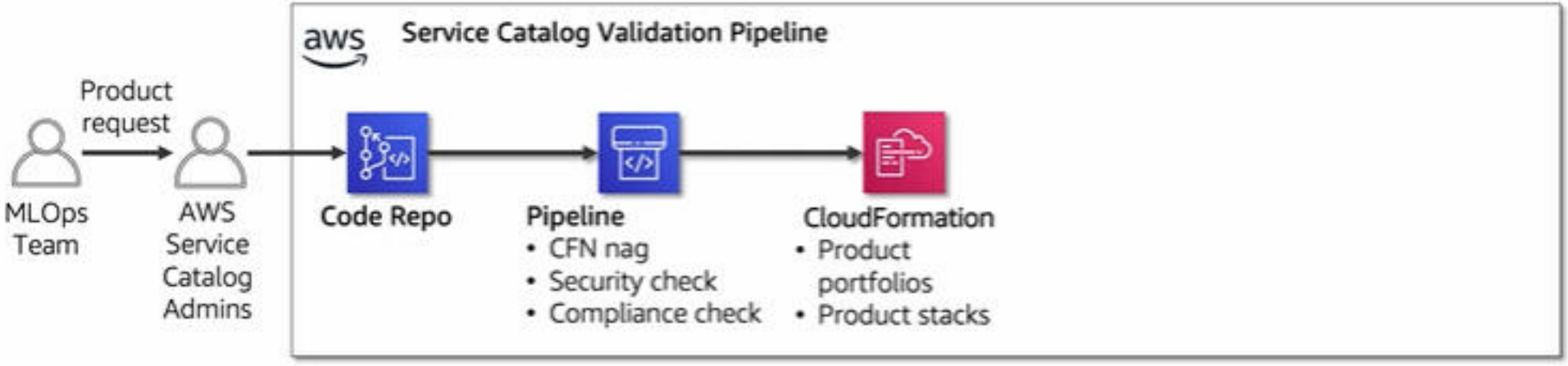
Create portfolios/products

Constraints - Security controls - Tags

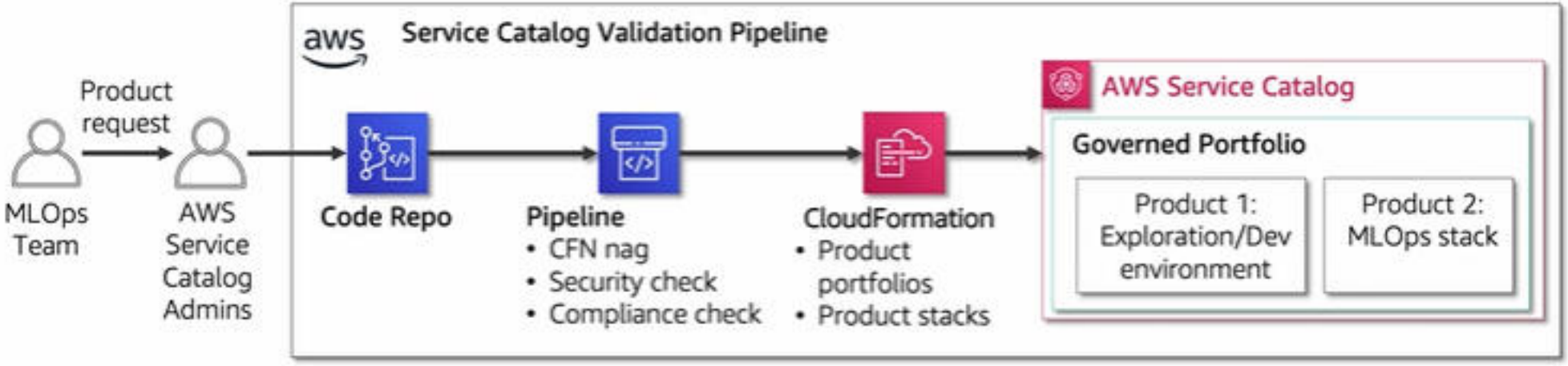
Ensure governance

© 2020 Amazon Web Services, Inc. or its Affiliates. All rights reserved.

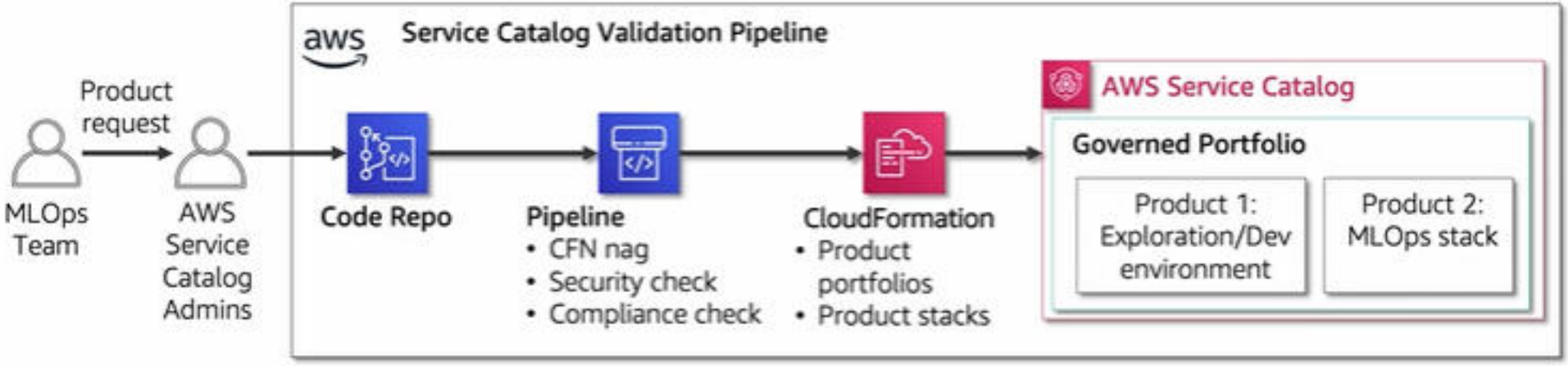
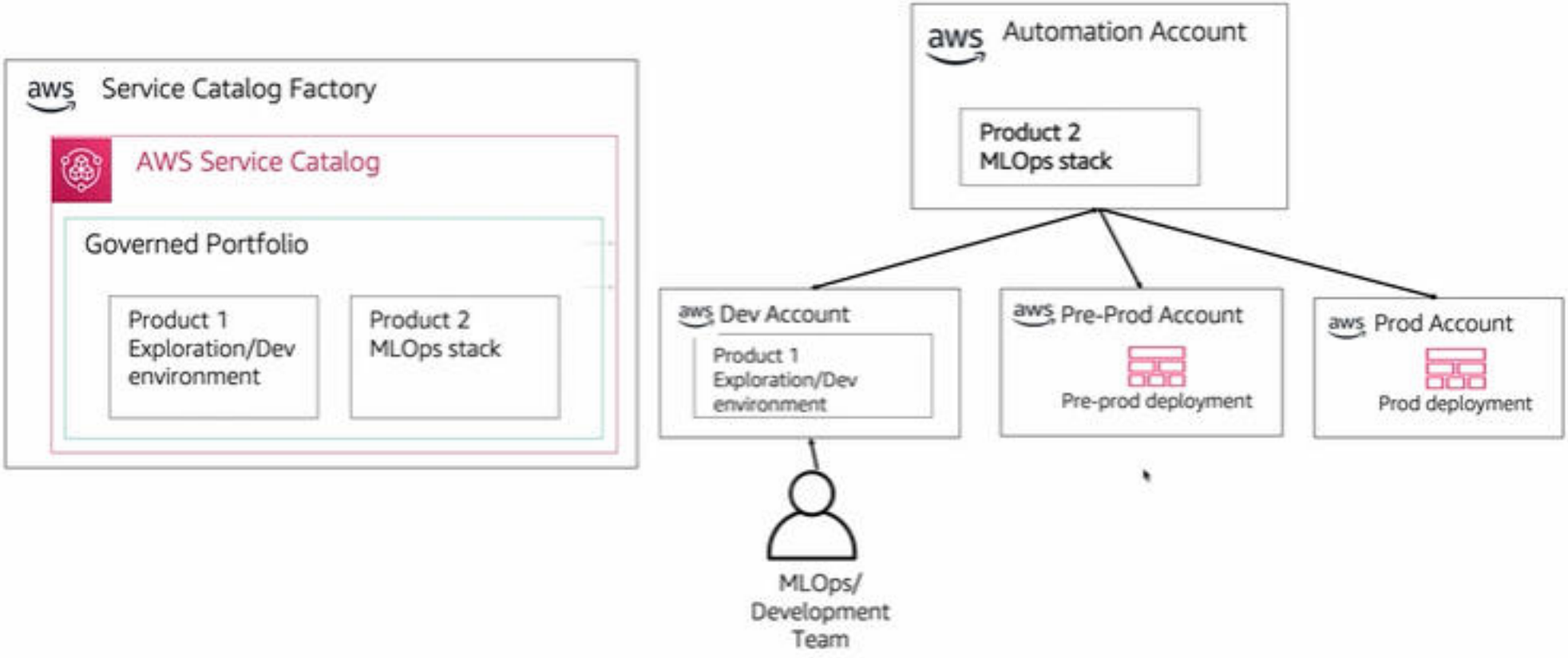
Create governed environments with AWS Service Catalog



Create governed environments with AWS Service Catalog



Create governed environments with AWS Service Catalog



Launch governed ML environments in a few clicks