Build, Train, and Deploy a Machine Learning Model

with Amazon SageMaker

In this tutorial, you will learn how to use Amazon SageMaker to build, train, and deploy a machine learning (ML) model. We will use the popular XGBoost ML algorithm for this exercise. Amazon SageMaker is a modular, fully managed machine learning service that enables developers and data scientists to build, train, and deploy ML models at scale.

Taking ML models from conceptualization to production is typically complex and time-consuming. You have to manage large amounts of data to train the model, choose the best algorithm for training it, manage the compute capacity while training it, and then deploy the model into a production environment. Amazon SageMaker reduces this complexity by making it much easier to build and deploy ML models. After you choose the right algorithms and frameworks from the wide range of choices available, it manages all of the underlying infrastructure to train your model at petabyte scale, and deploy it to production.

In this tutorial, you will assume the role of a machine learning developer working at a bank. You have been asked to develop a machine learning model to predict whether a customer will enroll for a certificate of deposit (CD). The model will be trained on the marketing dataset that contains information on customer demographics, responses to marketing events, and external factors.

The data has been labeled for your convenience and a column in the dataset identifies whether the customer is enrolled for a product offered by the bank. A version of this dataset is publicly available from the ML repository curated by the University of California, Irvine. This tutorial implements a supervised machine learning model since the data is labeled. (Unsupervised learning occurs when the datasets are not labeled.)

In this tutorial, you will:

1. Create a notebook instance

3. Train the model to learn from the data
4. Deploy the model
5. Evaluate your ML model's performance

The resources created and used in this tutorial are AWS free tier eligible. Remember to complete Step 7 and terminate your resources. If your account has been active with these resources for longer than two months, your account will charged less than $0.50.

This tutorial requires an AWS account

Create a Free Account

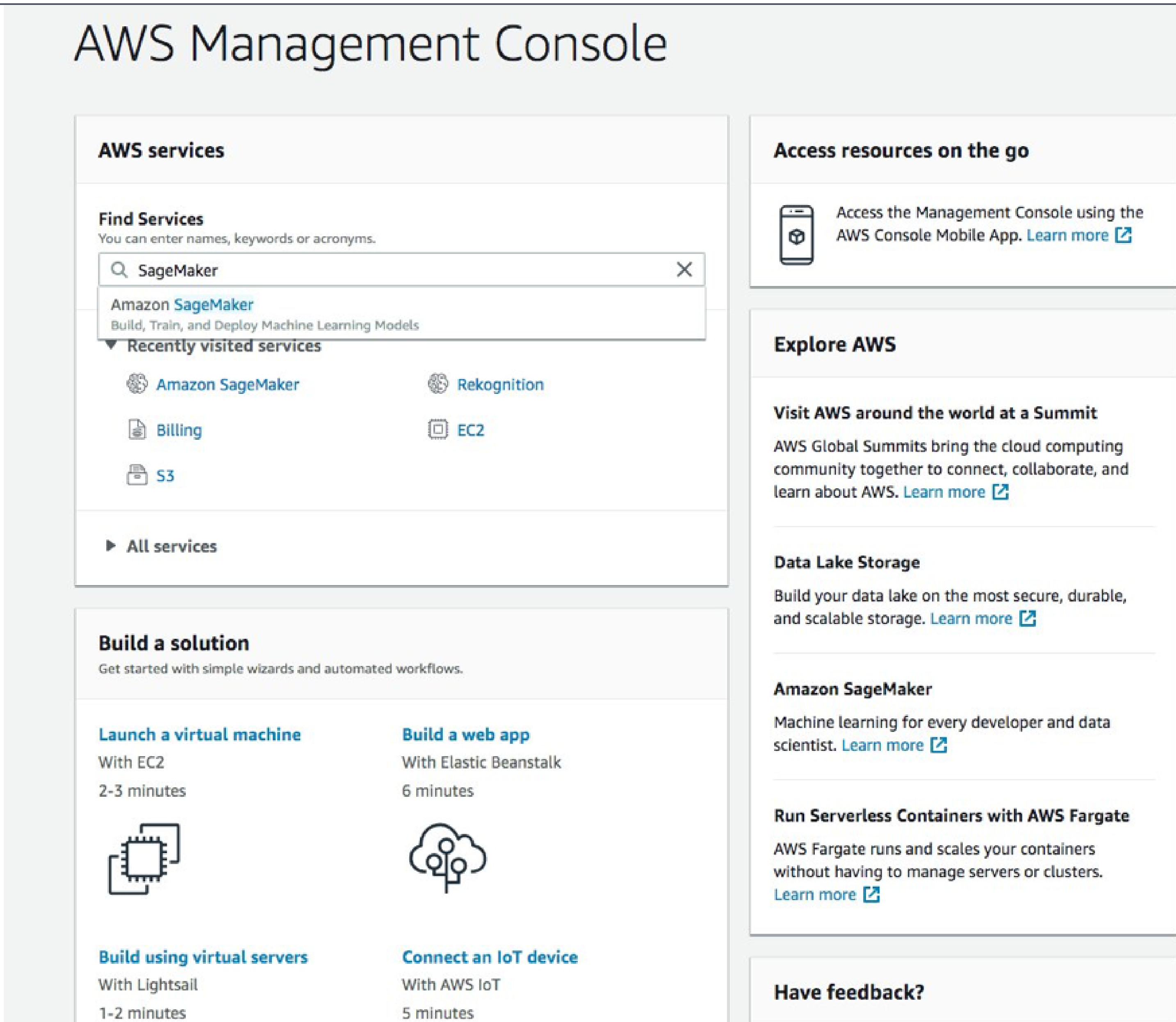
The resources you create in this tutorial are Free Tier eligible.

More about the Free Tier >>

Step 1. Enter the Amazon SageMaker console

Navigate to the Amazon SageMaker console.

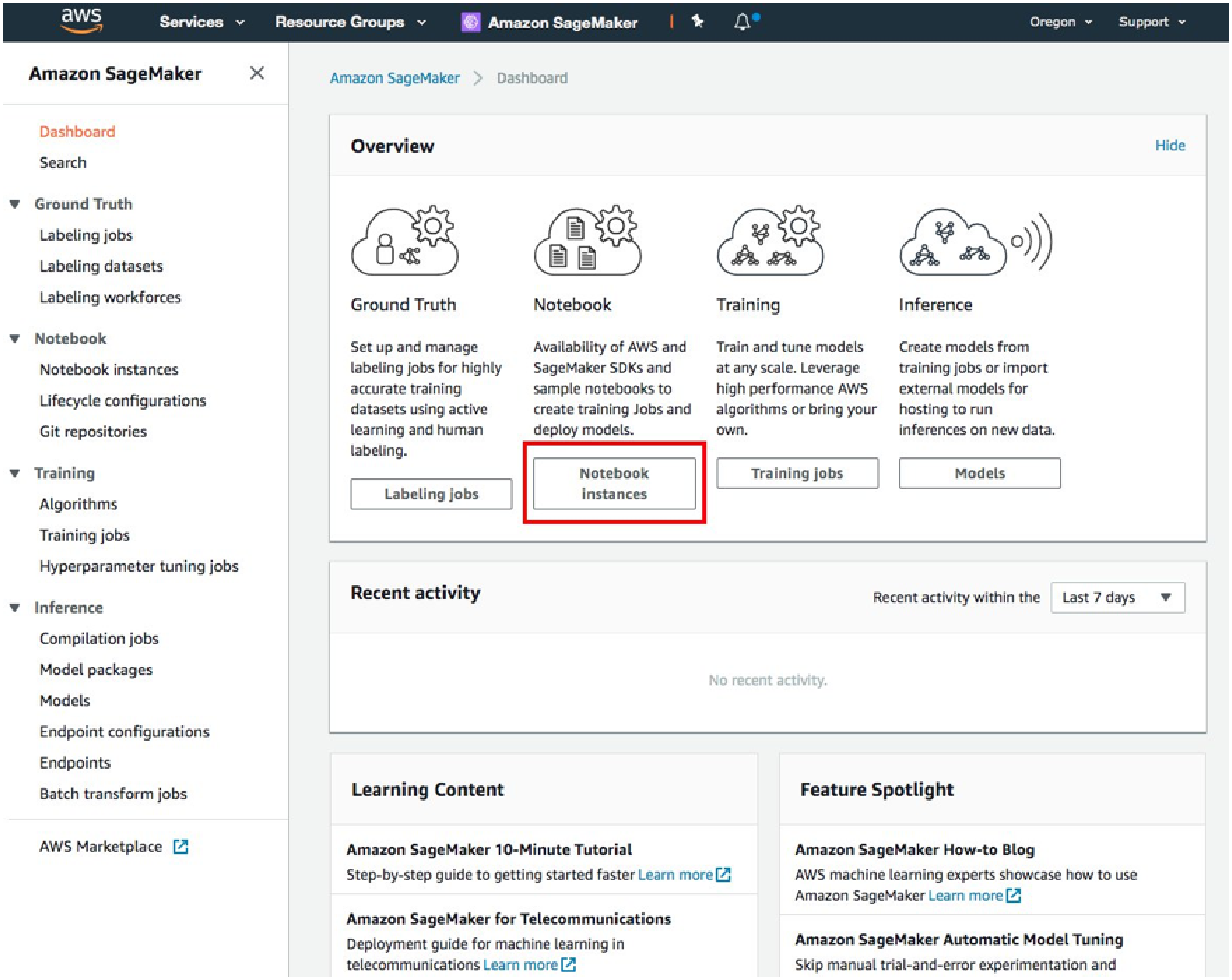
When you click here, the AWS Management Console will open in a new window, so that you can keep this step-by-step guide open. Begin typing *SageMaker* in the search bar and select **Amazon SageMaker** to open the service console.



(click to enlarge)

Step 2. Create an Amazon SageMaker notebook instance

In this step, you will create an Amazon SageMaker notebook instance.

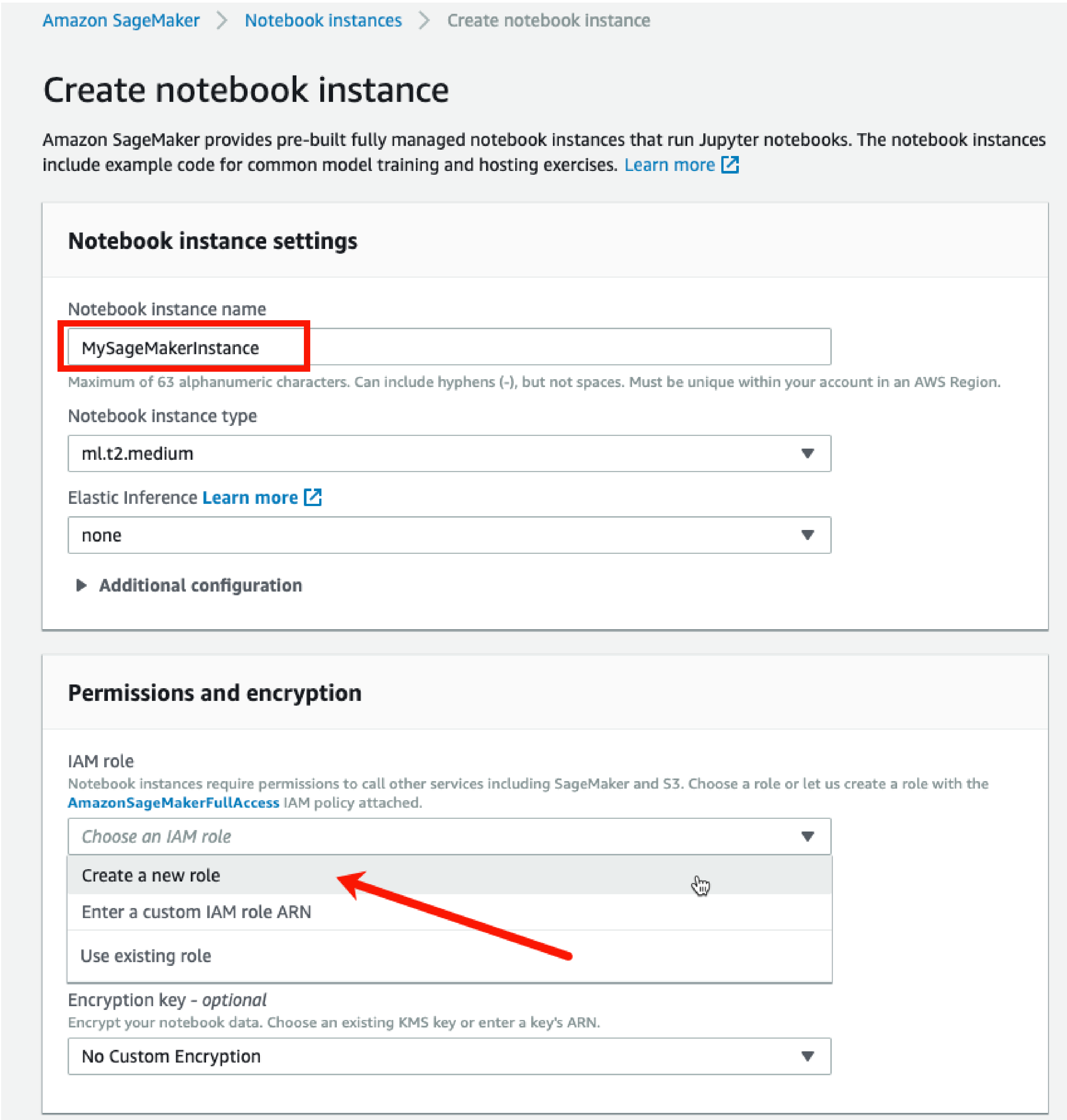


(click to enlarge)

2b. On the **Create notebook instance** page, enter a name in the **Notebook instance name** field. This tutorial uses *MySageMakerInstance* as the instance name, but you can choose a different name, if desired.

For this tutorial, you can keep the default **Notebook instance type** of *ml.t2.medium*.

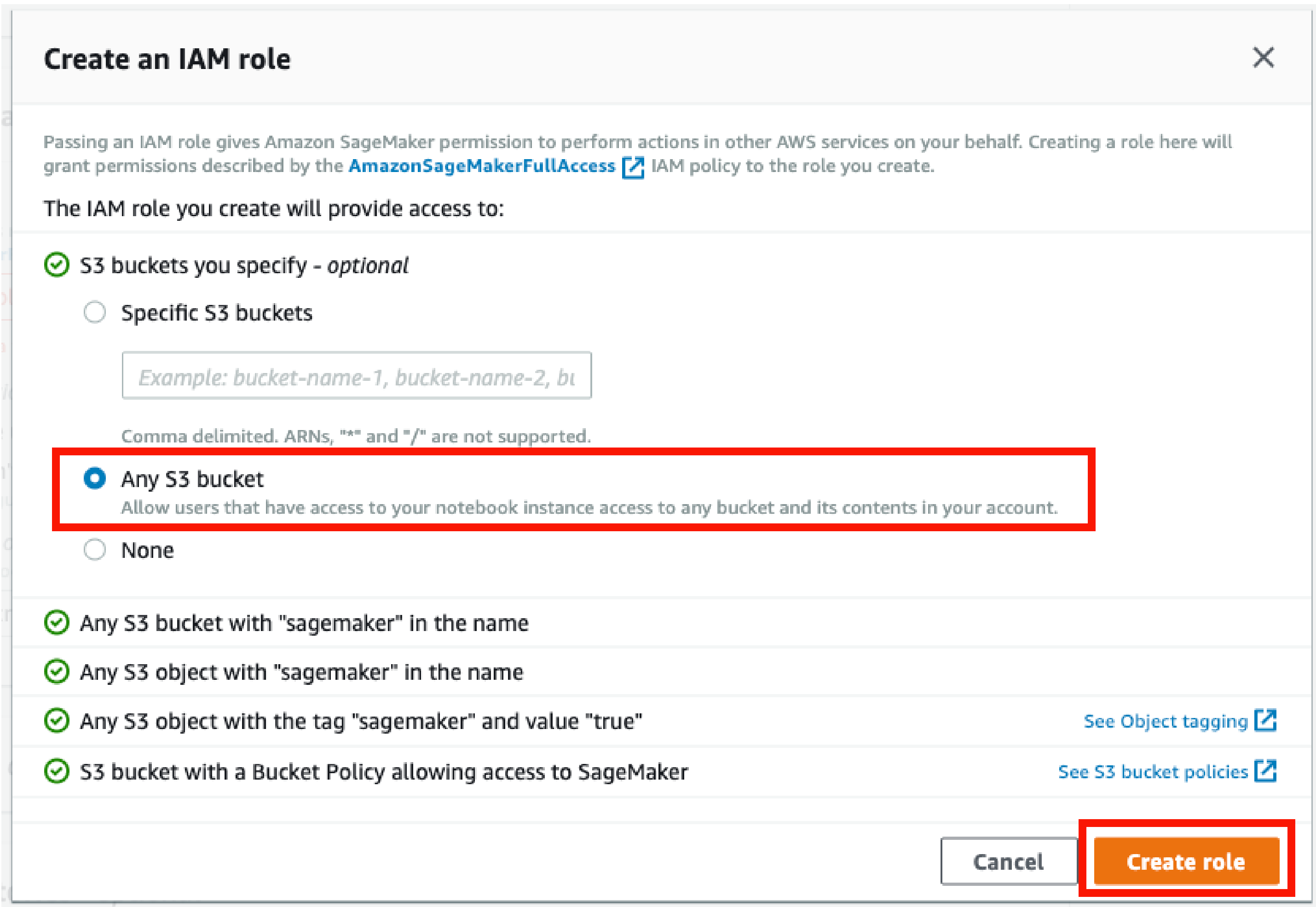
create a role with the required permissions and assign it to your instance. Alternately, you can choose an existing IAM role in your account for this purpose.



(click to enlarge)

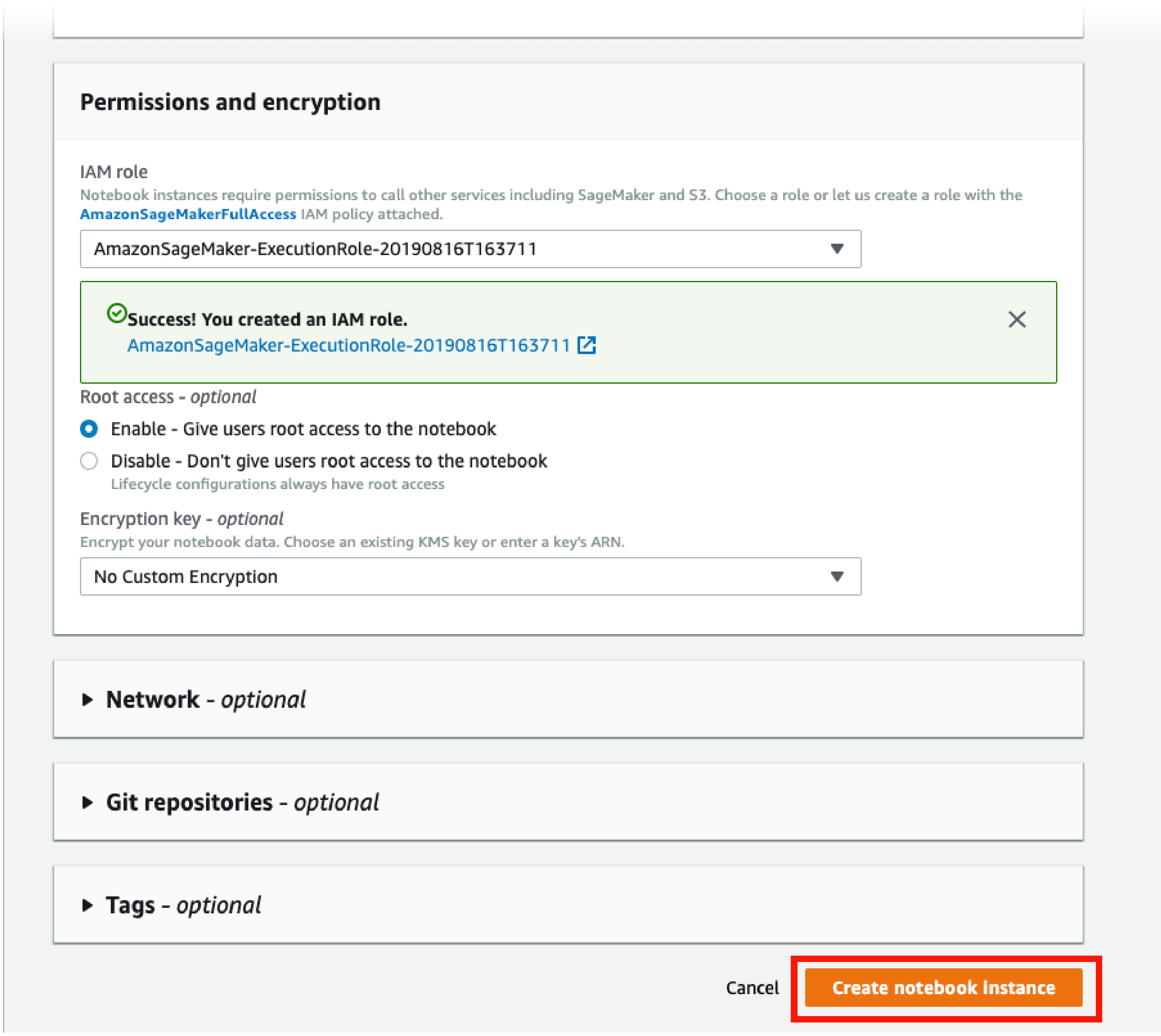
2c. In the **Create an IAM role** box, select **Any S3 bucket.** This allows your Amazon SageMaker instance to access all S3 buckets in your account. Later in this tutorial, you'll be creating a new S3 bucket. However, if you have a bucket you'd want to use instead, select **Specific S3 buckets** and specify the name of the bucket.

Choose **Create role**.



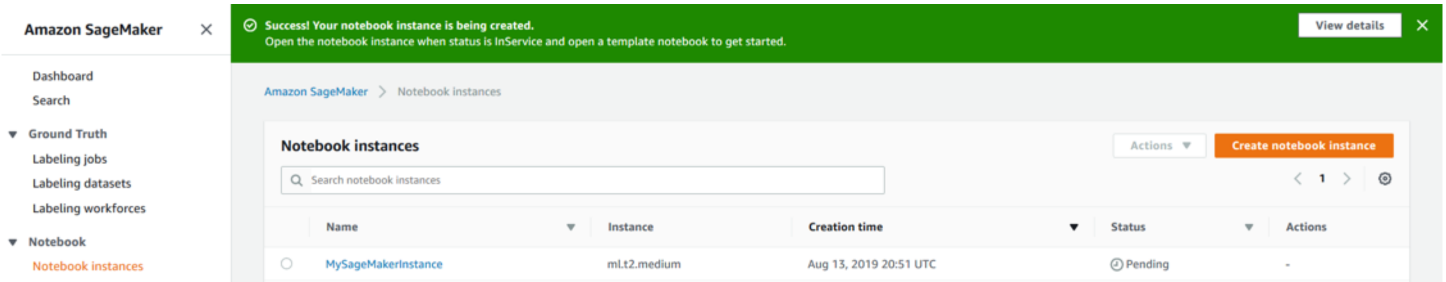
(click to enlarge)

2d. Notice that Amazon SageMaker created a role called *AmazonSageMaker-ExecutionRole-**** for you.



(click to enlarge)

2e. On the **Notebook instances** page, you should see your new *MySageMakerInstance* notebook instance in **Pending** status.



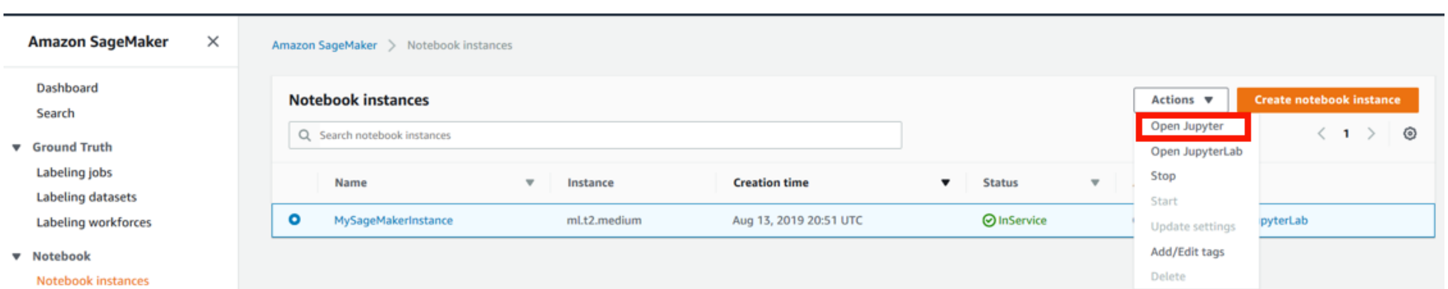
(click to enlarge)

Step 3. Prepare the data

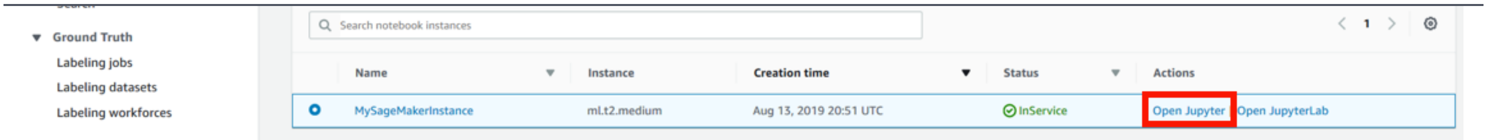
In this step you will use your Amazon SageMaker notebook to preprocess the data that you need to train your machine learning model.

3a. On the **Notebook instances** page, wait until *MySageMakerInstance* has transitioned from **Pending** to **InService** status.

After the status is **InService**, select *MySageMakerInstance* and open it using the **Actions** drop down menu, or by choosing **Open Jupyter** next to the **InService** status.

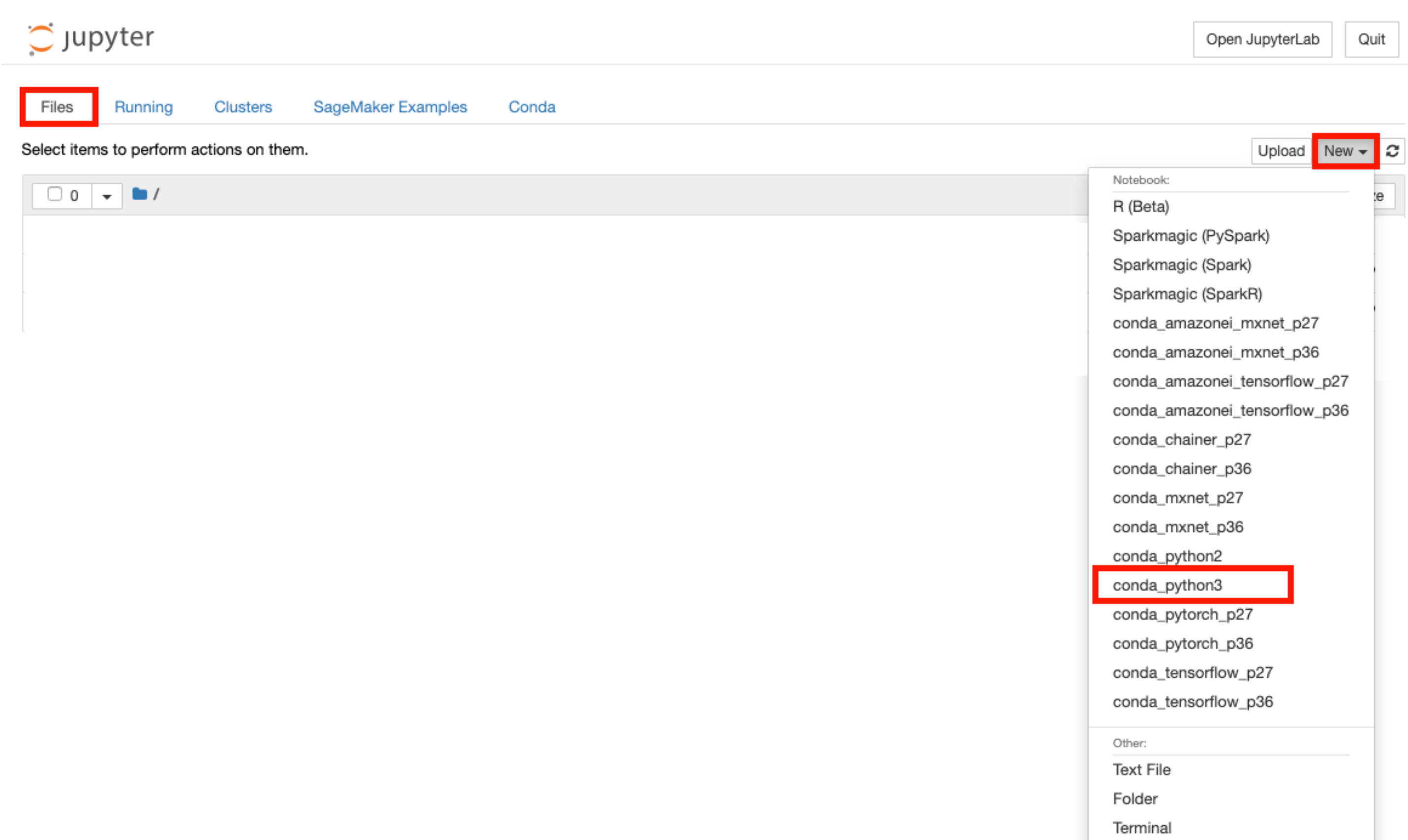


(click to enlarge)



(click to enlarge)

3b. After Jupyter opens, from the **Files** tab, choose **New** and then choose **conda_python3**.



(click to enlarge)

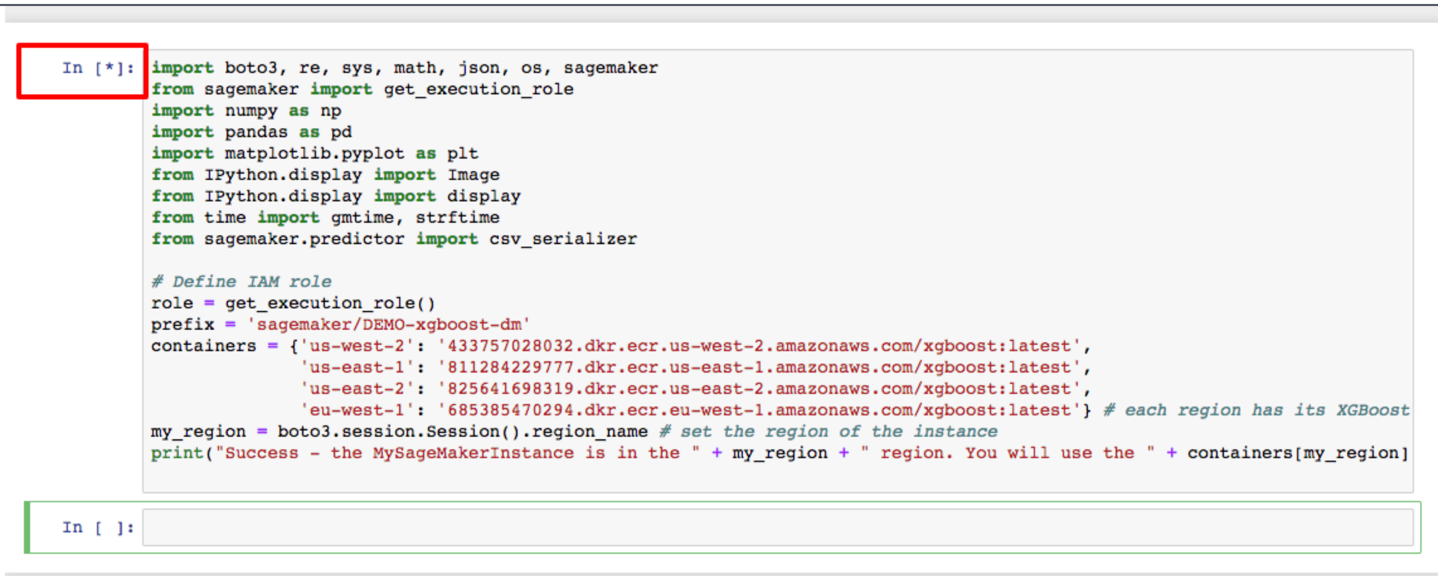
following code into the code cell in your instance and select **Run**.

While the code runs, an * appears between the square brackets as pictured in the first screenshot to the right. After a few seconds, the code execution will complete, the * will be replaced with the number **1**, and you will see a success message as pictured in the second screenshot to the right.

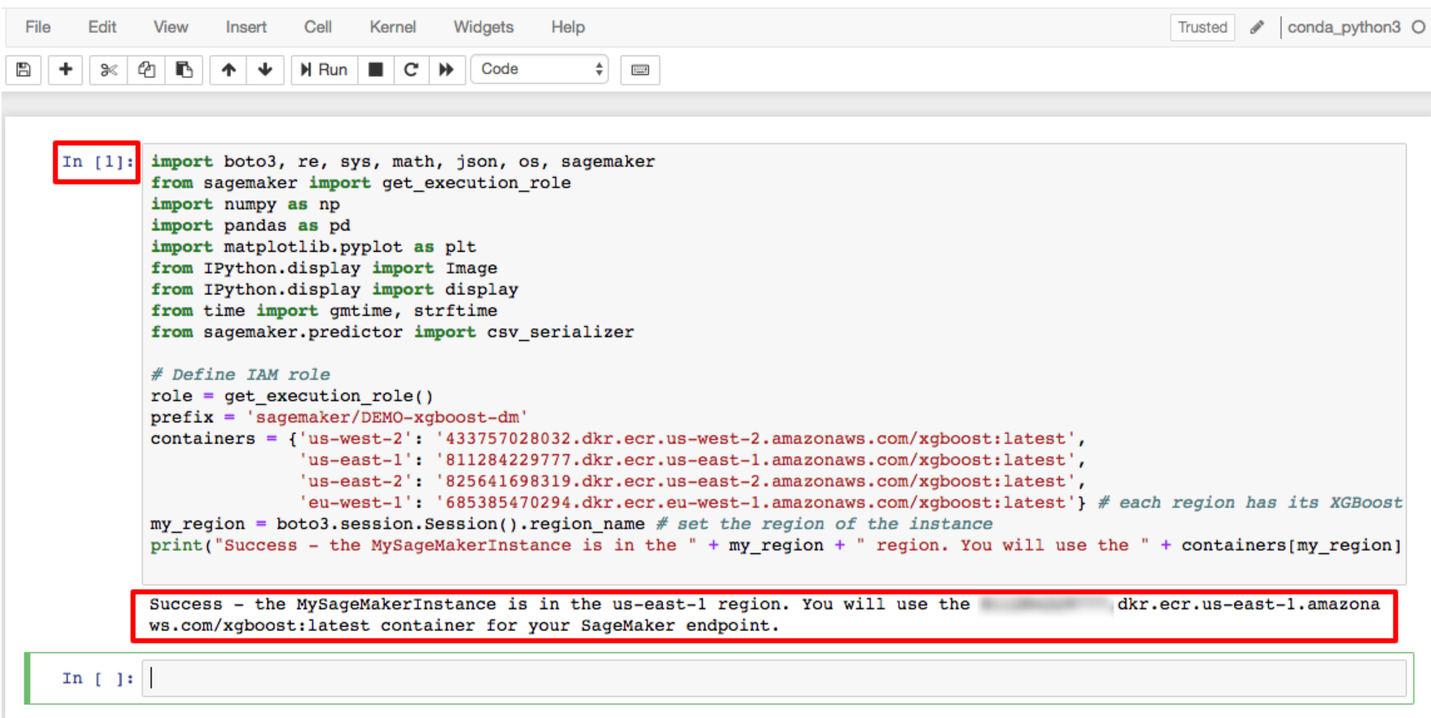
Python

```python
# import libraries
import boto3, re, sys, math, json, os, sagemaker, urllib.request
from sagemaker import get_execution_role
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from IPython.display import Image
from IPython.display import display
from time import gmtime, strftime
from sagemaker.predictor import csv_serializer

# Define IAM role
role = get_execution_role()
prefix = 'sagemaker/DEMO-xgboost-dm'
containers = {'us-west-2': '433757028032.dkr.ecr.us-west-2.amazonaws.com/xgboost:lat
              'us-east-1': '811284229777.dkr.ecr.us-east-1.amazonaws.com/xgboost:lat
              'us-east-2': '825641698319.dkr.ecr.us-east-2.amazonaws.com/xgboost:lat
              'eu-west-1': '685385470294.dkr.ecr.eu-west-1.amazonaws.com/xgboost:lat
my_region = boto3.session.Session().region_name # set the region of the instance
print("Success - the MySageMakerInstance is in the " + my_region + " region. You wil
```



(click to enlarge)



(click to enlarge)

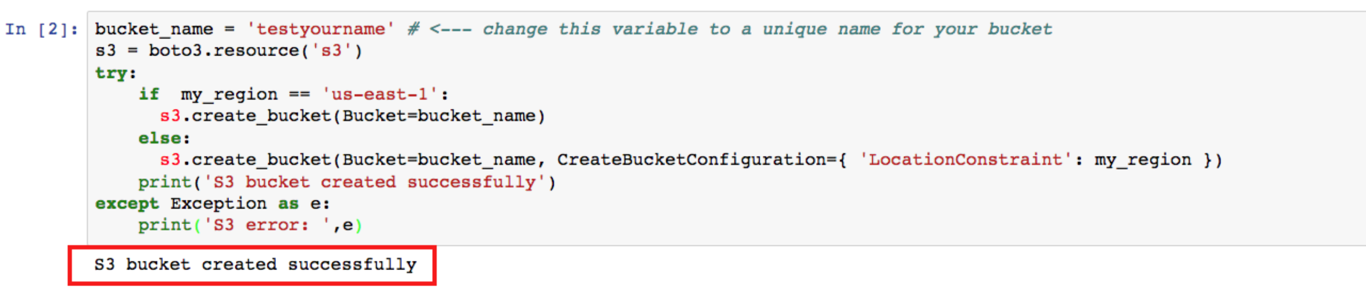
3d. In this step, you create an S3 bucket that will store your data for this tutorial.

bucket to make it unique. S3 bucket names must be globally unique and have some other restrictions and limitations.

Select **Run**. If you don't receive a success message, change the bucket name and try again.

Python

```python
bucket_name = 'your-s3-bucket-name' # <--- CHANGE THIS VARIABLE TO A UNIQUE NAME FOR
s3 = boto3.resource('s3')
try:
    if  my_region == 'us-east-1':
      s3.create_bucket(Bucket=bucket_name)
    else:
      s3.create_bucket(Bucket=bucket_name, CreateBucketConfiguration={ 'LocationCons
    print('S3 bucket created successfully')
except Exception as e:
    print('S3 error: ',e)
```



```
In [2]: bucket_name = 'testyourname' # <--- change this variable to a unique name for your bucket
        s3 = boto3.resource('s3')
        try:
            if  my_region == 'us-east-1':
              s3.create_bucket(Bucket=bucket_name)
            else:
              s3.create_bucket(Bucket=bucket_name, CreateBucketConfiguration={ 'LocationConstraint': my_region })
            print('S3 bucket created successfully')
        except Exception as e:
            print('S3 error: ',e)

        S3 bucket created successfully
```

(click to enlarge)

3e. Next, you need to download the data to your Amazon SageMaker instance and load it into a dataframe. Copy and **Run** the following code:

```
try:
  urllib.request.urlretrieve ("https://d1.awsstatic.com/tmt/build-train-deploy-machi
  print('Success: downloaded bank_clean.csv.')
except Exception as e:
  print('Data load error: ',e)

try:
  model_data = pd.read_csv('./bank_clean.csv',index_col=0)
  print('Success: Data loaded into dataframe.')
except Exception as e:
    print('Data load error: ',e)
```

(click to enlarge)

3f. Now, we will shuffle the data and split it into training data and test data.

The **training data** (70% of customers) will be used during the model training loop. We will use gradient-based optimization to iteratively refine the model parameters. Gradient-based optimization is a way to find model parameter values that minimize the model error, using the gradient of the model loss function.

The **test data** (remaining 30% of customers) will be used to evaluate the performance of the model, and measure how well the trained model generalizes to unseen data.

Copy the following code into a new code cell and select **Run** to shuffle and split the data:

```
train_data, test_data = np.split(model_data.sample(frac=1, random_state=1729), [int(
print(train_data.shape, test_data.shape)
```

```
In [4]: train_data, test_data = np.split(model_data.sample(frac=1, random_state=1729), [int(0.7 * len(model_data))])
        print(train_data.shape, test_data.shape)
        (28831, 61) (12357, 61)
```

(click to enlarge)

Step 4. Train the model from the data

In this step, you will train your machine learning model with the training dataset.

4a. To use an Amazon SageMaker pre-built XGBoost model, you will need to reformat the header and first column of the training data and load the data from the S3 bucket.

Copy the following code into a new code cell and select **Run** to reformat and load the data:

Python

```
pd.concat([train_data['y_yes'], train_data.drop(['y_no', 'y_yes'], axis=1)], axis=1)
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix, 'trai
s3_input_train = sagemaker.s3_input(s3_data='s3://{}/{}/train'.format(bucket_name, p
```

new code cell and select **Run**:

Python

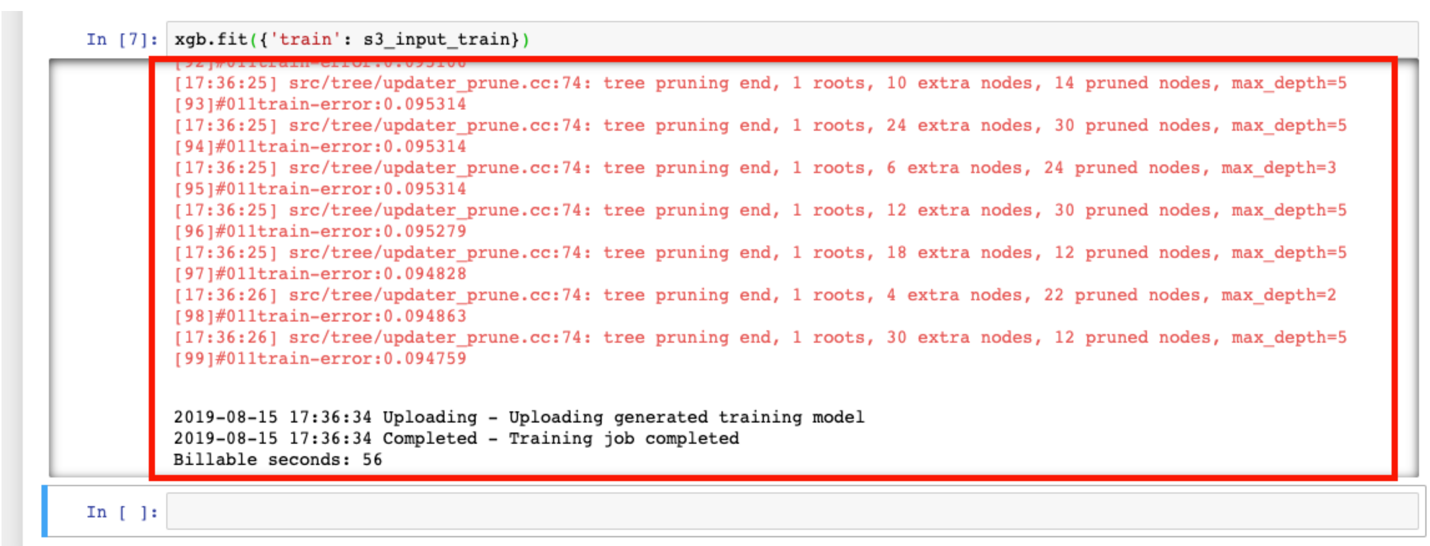
```
sess = sagemaker.Session()
xgb = sagemaker.estimator.Estimator(containers[my_region],role, train_instance_count
xgb.set_hyperparameters(max_depth=5,eta=0.2,gamma=4,min_child_weight=6,subsample=0.8
```

4c. With the data loaded and the XGBoost estimator set up, train the model using gradient optimization on a ***ml.m4.xlarge*** instance by copying the following code into the next code cell and selecting **Run.**

After a few minutes, you should start to see the training logs being generated.

Python

```
xgb.fit({'train': s3_input_train})
```



(click to enlarge)

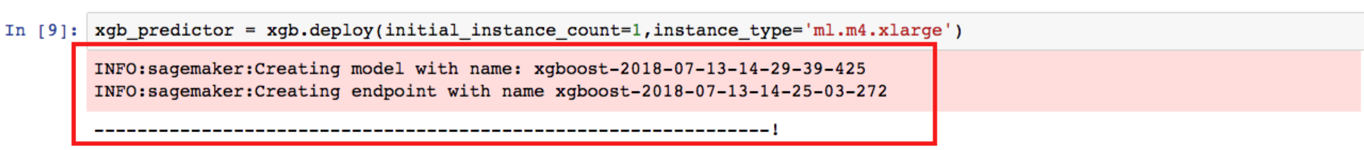
Step 5. Deploy the model

In this step, you will deploy the trained model to an endpoint, reformat then load the CSV data, then run the model to create predictions.

5a. To deploy the model on a server and create an endpoint that you can access, copy the following code into the next code cell and select **Run**:

Python

```python
xgb_predictor = xgb.deploy(initial_instance_count=1,instance_type='ml.m4.xlarge')
```



(click to enlarge)

5b. To predict whether customers in the test data enrolled for the bank product or not, copy the following code into the next code cell and select **Run**:

Python

```python
test_data_array = test_data.drop(['y_no', 'y_yes'], axis=1).values #load the data in
xgb_predictor.content_type = 'text/csv' # set the data type for an inference
xgb_predictor.serializer = csv_serializer # set the serializer type
predictions = xgb_predictor.predict(test_data_array).decode('utf-8') # predict!

```

```
In [20]: test_data_array = test_data.drop(['y_no', 'y_yes'], axis=1).values #load the data into an array
         xgb_predictor.content_type = 'text/csv' # set the data type for an inference
         xgb_predictor.serializer = csv_serializer # set the serializer type
         predictions = xgb_predictor.predict(test_data_array).decode('utf-8') # predict!
         predictions_array = np.fromstring(predictions[1:], sep=',') # and turn the prediction into an array
         print(predictions_array.shape)
(12357,)
```

(click to enlarge)

Step 6. Evaluate model performance

In this step, you will evaluate the performance and accuracy of the machine learning model.

6a. Copy and paste the code below and select **Run** to compare actual vs. predicted values in a table called a ***confusion matrix***.

Based on the prediction, we can conclude that you predicted a customer will enroll for a certificate of deposit accurately for 90% of customers in the test data, with a precision of 65% (278/429) for enrolled and 90% (10,785/11,928) for didn't enroll.

Python

```python
cm = pd.crosstab(index=test_data['y_yes'], columns=np.round(predictions_array), rown
tn = cm.iloc[0,0]; fn = cm.iloc[1,0]; tp = cm.iloc[1,1]; fp = cm.iloc[0,1]; p = (tp+
print("\n{0:<20}{1:<4.1f}%\n".format("Overall Classification Rate: ", p))
print("{0:<15}{1:<15}{2:>8}".format("Predicted", "No Purchase", "Purchase"))
print("Observed")
print("{0:<15}{1:<2.0f}% ({2:<}){3:>6.0f}% ({4:<})".format("No Purchase", tn/(tn+fn)
print("{0:<16}{1:<1.0f}% ({2:<}){3:>7.0f}% ({4:<}) \n".format("Purchase", fn/(tn+fn)
```

```
In [12]: cm = pd.crosstab(index=test_data['y_yes'], columns=np.round(predictions_array), rownames=['Observed'], colnames=['Predi
         tn = cm.iloc[0,0]; fn = cm.iloc[1,0]; tp = cm.iloc[1,1]; fp = cm.iloc[0,1]; p = (tp+tn)/(tp+tn+fp+fn)*100
         print("\n{0:<20}{1:<4.1f}%\n".format("Overall Classification Rate: ", p))
         print("{0:<15}{1:<15}{2:>8}".format("Predicted", "No Purchase", "Purchase"))
         print("Observed")
         print("{0:<15}{1:<2.0f}% ({2:<}){3:>6.0f}% ({4:<})".format("No Purchase", tn/(tn+fn)*100,tn, fp/(tp+fp)*100, fp))
         print("{0:<16}{1:<1.0f}% ({2:<}){3:>7.0f}% ({4:<}) \n".format("Purchase", fn/(tn+fn)*100,fn, tp/(tp+fp)*100, tp))
```

```
Overall Classification Rate: 89.5%

Predicted      No Purchase    Purchase
Observed
No Purchase    90% (10785)    35% (151)
Purchase        10% (1143)     65% (278)
```

(click to enlarge)

Step 7. Terminate your resources

In this step, you will terminate your Amazon SageMaker-related resources.

Important: Terminating resources that are not actively being used reduces costs and is a best practice. Not terminating your resources will result in a charge.

7a. To delete the Amazon SageMaker endpoint and the objects in your S3 bucket, copy, paste and **Run** the following code:

Python

```python
sagemaker.Session().delete_endpoint(xgb_predictor.endpoint)
bucket_to_delete = boto3.resource('s3').Bucket(bucket_name)
bucket_to_delete.objects.all().delete()
```

```
Out[22]: [{'ResponseMetadata': {'RequestId': '28E42783694CB6C0',
   'HostId': 'j7ajjzyseZHs5ruIrbmaKuXeJh/I7EjYGs9VADB9wY5HTbuG24AFetOjta603cWo39XPXBkf7XA=',
   'HTTPStatusCode': 200,
   'HTTPHeaders': {'x-amz-id-2': 'j7ajjzyseZHs5ruIrbmaKuXeJh/I7EjYGs9VADB9wY5HTbuG24AFetOjta603cWo39XPXBkf7XA=',
    'x-amz-request-id': '28E42783694CB6C0',
    'date': 'Thu, 15 Aug 2019 21:21:43 GMT',
    'connection': 'close',
    'content-type': 'application/xml',
    'transfer-encoding': 'chunked',
    'server': 'AmazonS3'},
   'RetryAttempts': 0},
  'Deleted': [{'Key': 'sagemaker/DEMO-xgboost-dm/output/xgboost-2019-08-15-18-39-29-757/output/model.tar.gz'},
   {'Key': 'sagemaker/DEMO-xgboost-dm/train/train.csv'}]}]
```


(click to enlarge)

You have learned how to use Amazon SageMaker to prepare, train, deploy and evaluate a machine learning model. Amazon SageMaker makes it easy to build ML models by providing everything you need to quickly connect to your training data and select the best algorithm and framework for your application, while managing all of the underlying infrastructure, so you can train models at petabyte scale.

Was this tutorial helpful?

Yes

No

Have more feedback? Tell us here

Sign In to the Console

Learn About AWS

What Is AWS?

Resources for AWS

Getting Started

Training and Certification

AWS Solutions Portfolio

Developers on AWS

Developer Center

SDKs & Tools

.NET on AWS

What Is a Container?

What Is a Data Lake?

AWS Cloud Security

What's New

Blogs

Press Releases

Analyst Reports

AWS Partner Network

PHP on AWS

Javascript on AWS

Help

Contact Us

AWS Careers

File a Support Ticket

Knowledge Center

AWS Support Overview

Legal

Sign In to the Console

Amazon is an Equal Opportunity Employer: *Minority / Women / Disability / Veteran / Gender Identity / Sexual Orientation / Age.*

Language

عربي |
Bahasa Indonesia |
Deutsch |
English |
Español |
Français |
Italiano |
Português |
Tiếng Việt |
Türkçe |
Русский |
ไทย |
日本語 |

中文 (繁體)

Privacy
|
Site Terms
|
© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.