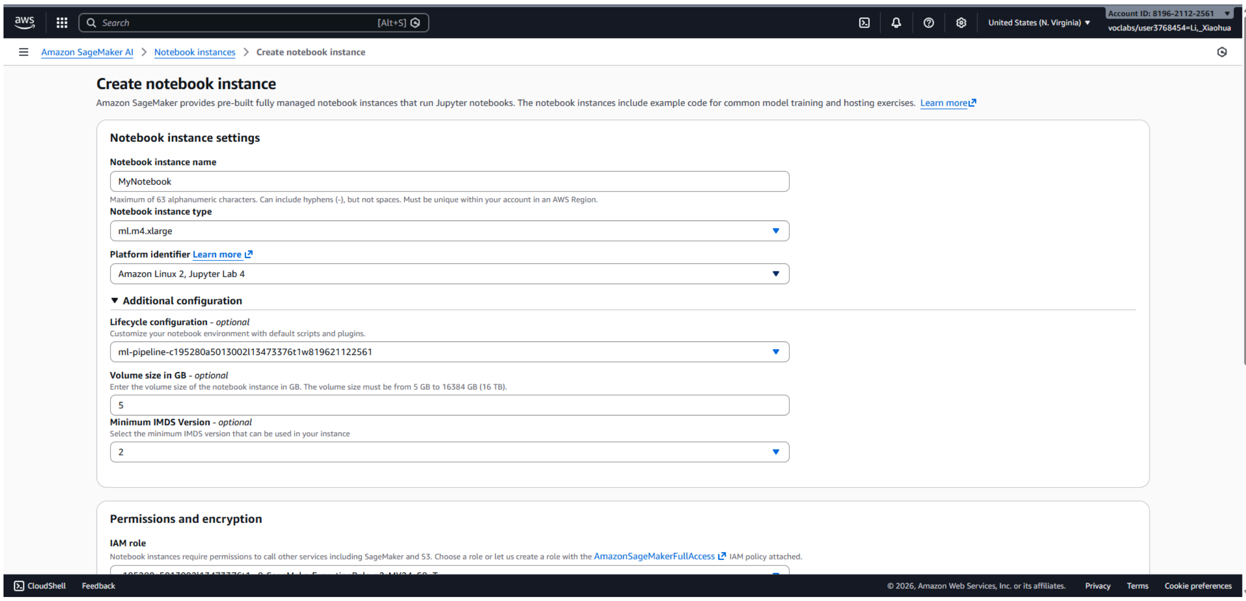
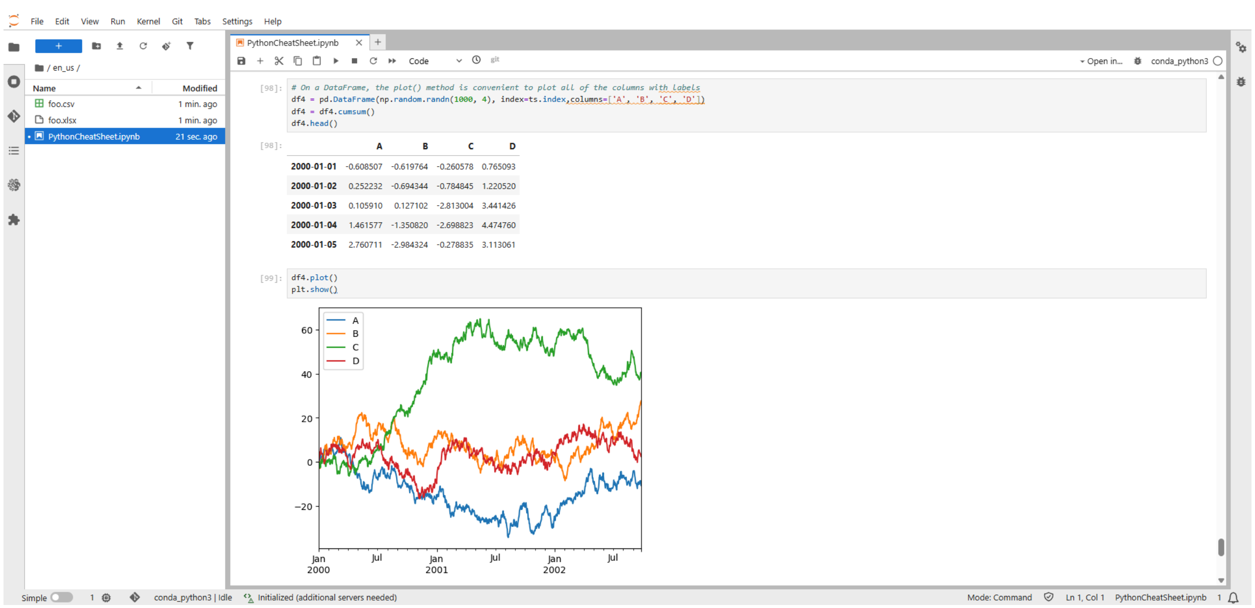
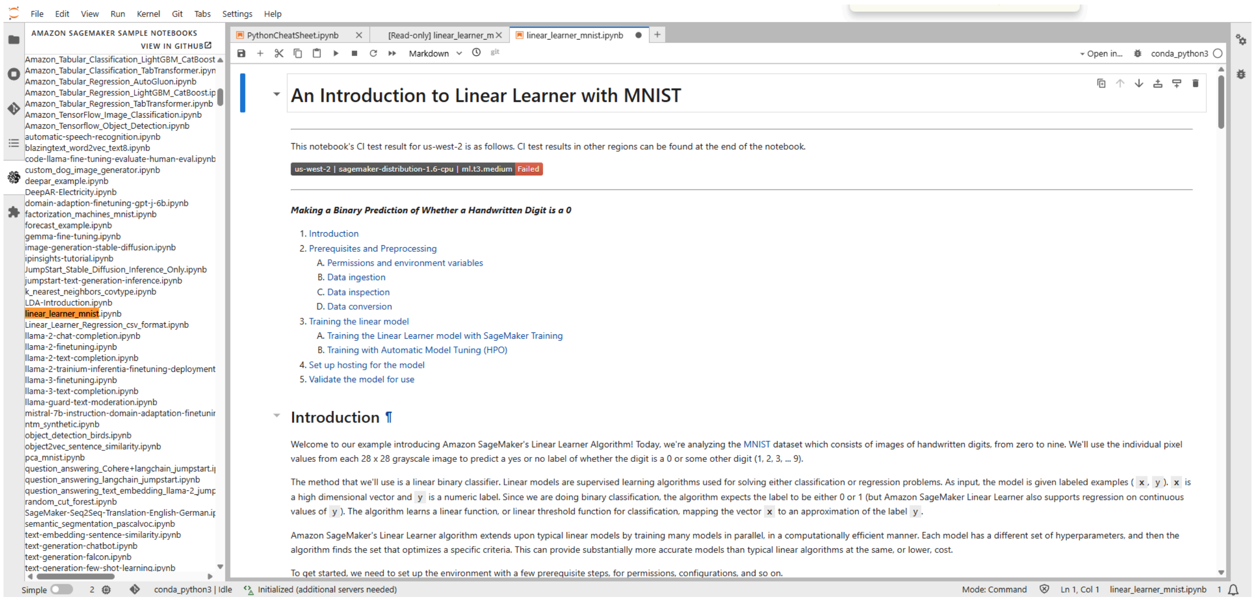
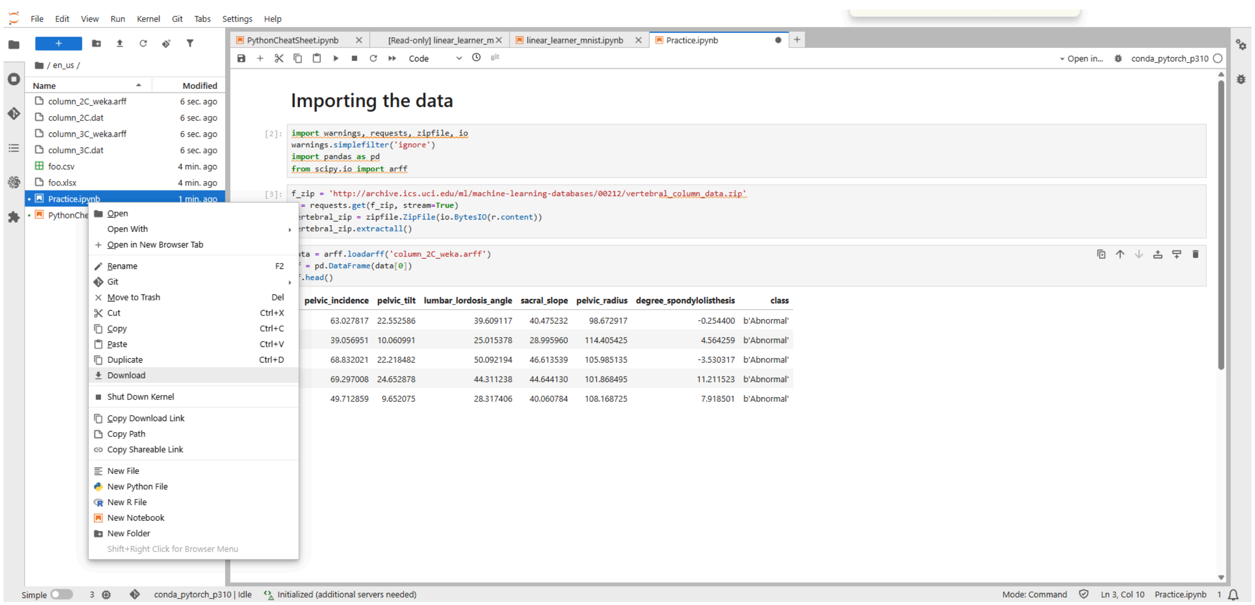
Lab 3.1 - Amazon SageMaker - Creating and importing data

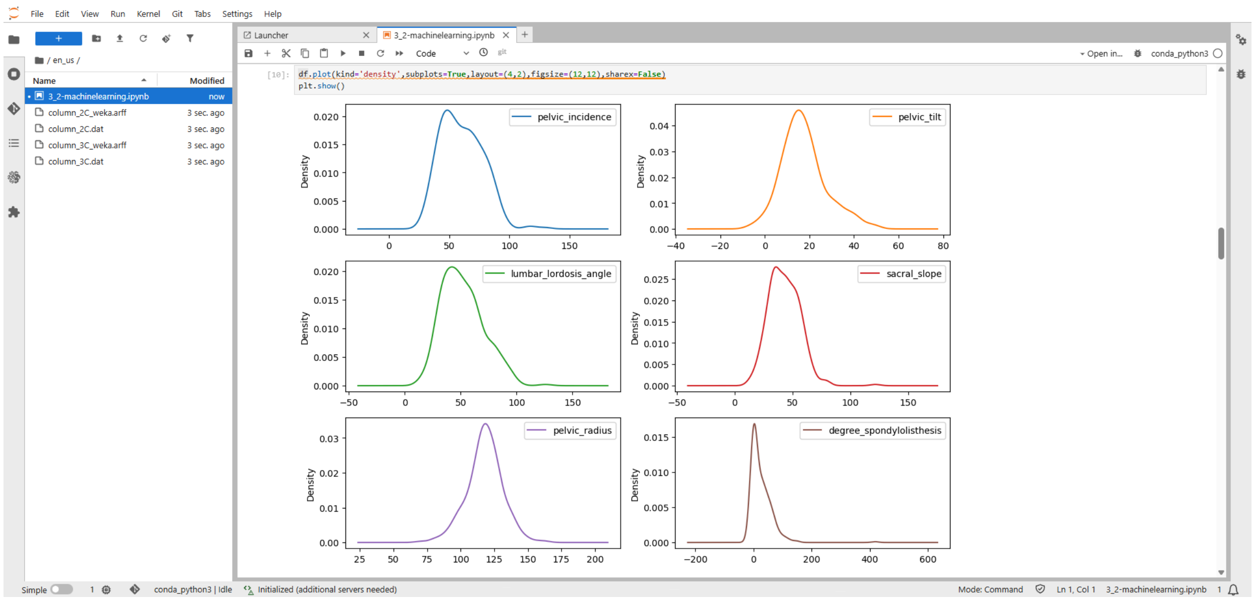


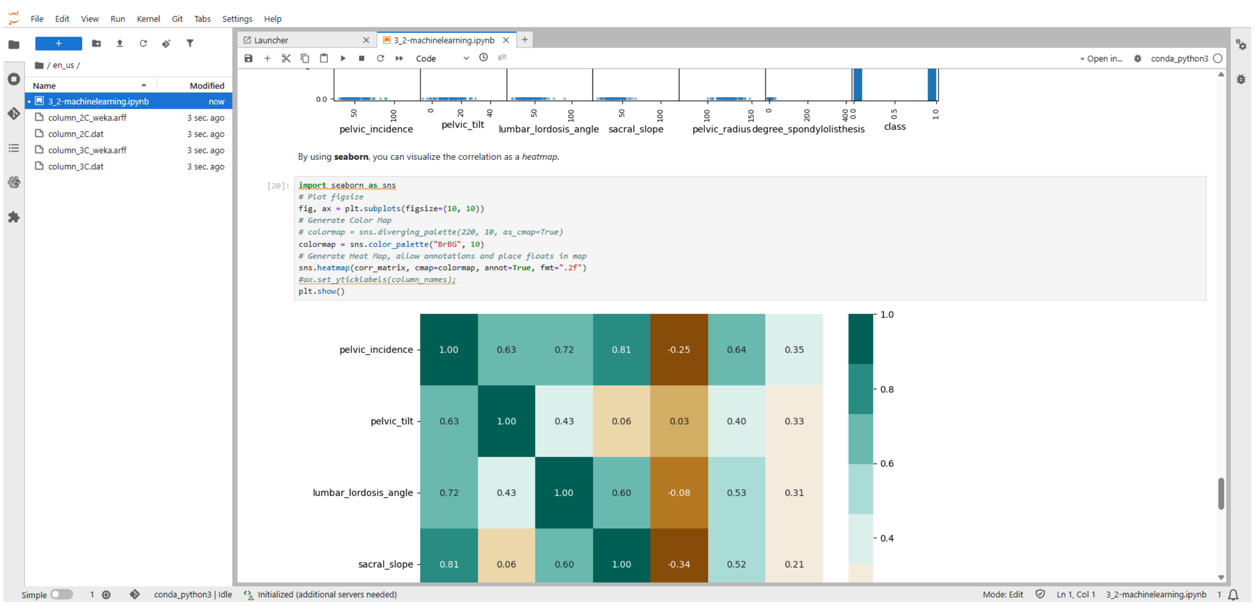



File Edit View Run Kernel Git Tabs Settings Help
AMAZON SAGEMAKER SAMPLE NOTEBOOKS
VIEW IN GITHUB
Amazon_Tabular_Classification_LightGBM_CatBoost
Amazon_Tabular_Classification_TabTransformer.ipyn
Amazon_Tabular_Regression_AutoGluon.ipynb
Amazon_Tabular_Regression_LightGBM_CatBoost.ip
Amazon_Tabular_Regression_TabTransformer.ipynb
Amazon_TensorFlow_Image_Classification.ipynb
Amazon_Tensorflow_Object_Detection.ipynb
automatic-speech-recognition.ipynb
blazingtext_word2vec_text8.ipynb
code-llama-fine-tuning-evaluate-human-eval.ipynb
custom_dog_image_generator.ipynb
deepar_example.ipynb
DeepAR-Electricity.ipynb
domain-adaption-finetuning-gpt-j-6b.ipynb
factorization_machines_mnist.ipynb
forecast_example.ipynb
gemma-fine-tuning.ipynb
image-generation-stable-diffusion.ipynb
ipinsights-tutorial.ipynb
JumpStart_Stable_Diffusion_Inference_Only.ipynb
jumpstart-text-generation-inference.ipynb
k_nearest_neighbors_covtype.ipynb
LDA-Introduction.ipynb
linear_learner_mnist.ipynb
Linear_Learner_Regression_csv_format.ipynb
llama-2-chat-completion.ipynb
llama-2-finetuning.ipynb
llama-2-text-completion.ipynb
llama-2-trainium-inferentia-finetuning-deployment
llama-3-finetuning.ipynb
llama-3-text-completion.ipynb
llama-guard-text-moderation.ipynb
mistral-7b-instruction-domain-adaptation-finetunir
ntm_synthetic.ipynb
object_detection_birds.ipynb
object2vec_sentence_similarity.ipynb
pca_mnist.ipynb
question_answering_Cohere+langchain_jumpstart.i
question_answering_langchain_jumpstart.ipynb
question_answering_text_embedding_llama-2_jump
random_cut_forest.ipynb
SageMaker-Seq2Seq-Translation-English-German.ip
semantic_segmentation_pascalvoc.ipynb
text-embedding-sentence-similarity.ipynb
text-generation-chatbot.ipynb
text-generation-falcon.ipynb
text-generation-few-shot-learning.ipynb
PythonCheatSheet.ipynb
[Read-only] linear_learner_m
linear_learner_mnist.ipynb
Markdown
Open in...
conda_python3
An Introduction to Linear Learner with MNIST
This notebook's CI test result for us-west-2 is as follows. CI test results in other regions can be found at the end of the notebook.
us-west-2 | sagemaker-distribution-1.6-cpu | ml.t3.medium Failed
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
A. Training the Linear Learner model with SageMaker Training
B. Training with Automatic Model Tuning (HPO)
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.
Simple 2 conda_python3 | Idle Initialized (additional servers needed)
Mode: Command Ln 1, Col 1 linear_learner_mnist.ipynb 1


File Edit View Run Kernel Git Tabs Settings Help
/ en_us /
Name Modified
column_2C_weka.arff 6 sec. ago
column_2C.dat 6 sec. ago
column_3C_weka.arff 6 sec. ago
column_3C.dat 6 sec. ago
foo.csv 4 min. ago
foo.xlsx 4 min. ago
Practice.ipynb 1 min. ago
PythonChe
Open
Open With
Open in New Browser Tab
Rename F2
Git
Move to Trash Del
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Duplicate Ctrl+D
Download
Shut Down Kernel
Copy Download Link
Copy Path
Copy Shareable Link
New File
New Python File
New R File
New Notebook
New Folder
Shift+Right Click for Browser Menu
PythonCheatSheet.ipynb
[Read-only] linear_learner_m
linear_learner_mnist.ipynb
Practice.ipynb
Code
Open in...
conda_pytorch_p310
Importing the data
[2]: import warnings, requests, zipfile, io
warnings.simplefilter('ignore')
import pandas as pd
from scipy.io import arff
[3]: f_zip = 'http://archive.ics.uci.edu/ml/machine-learning-databases/00212/vertebral_column_data.zip'
= requests.get(f_zip, stream=True)
rtebral_zip = zipfile.ZipFile(io.BytesIO(r.content))
rtebral_zip.extractall()
ta = arff.loadarff('column_2C_weka.arff')
= pd.DataFrame(data[0])
.head()
pelvic_incidence pelvic_tilt lumbar_lordosis_angle sacral_slope pelvic_radius degree_spondylolisthesis class
63.027817 22.552586 39.609117 40.475232 98.672917 -0.254400 b'Abnormal'
39.056951 10.060991 25.015378 28.995960 114.405425 4.564259 b'Abnormal'
68.832021 22.218482 50.092194 46.613539 105.985135 -3.530317 b'Abnormal'
69.297008 24.652878 44.311238 44.644130 101.868495 11.211523 b'Abnormal'
49.712859 9.652075 28.317406 40.060784 108.168725 7.918501 b'Abnormal'
Simple 3 conda_pytorch_p310 | Idle Initialized (additional servers needed)
Mode: Command Ln 3, Col 10 Practice.ipynb 1

Lab 3.2 - Amazon SageMaker - Exploring Data





Lab 3.3 - Amazon SageMaker - Encoding Categorical Data

Dataset attributions

This dataset was obtained from: Dua, D. and Graff, C. (2019). UCI Machine Learning Repository (http://archive.ics.uci.edu/ml). Irvine, CA: University of California, School of Information and Computer Science.

Step 1: Importing and exploring the data

You will start by examining the data in the dataset.

To get the most out of this lab, read the instructions and code before you run the cells. Take time to experiment!

Start by importing the pandas package and setting some default display options.

[1]:
```
import pandas as pd

pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)
```

Next, load the dataset into a pandas DataFrame.

The data doesn't contain a header, so you will define those column names in a variable that's named `col_names` to the attributes listed in the dataset description.

[2]:
```
url = "imports-85.csv"
col_names=['symboling','normalized-losses','fuel-type','aspiration','num-of-doors','body-style','drive-wheels','engine-location','wheel-base',
                                 'length','width','height','curb-weight','engine-type','num-of-cylinders','engine-size',
                                 'fuel-system','bore','stroke','compression-ratio','horsepower','peak-rpm','city-mpg','highway-mpg','price']

df_car = pd.read_csv(url,sep=',',names = col_names ,na_values="?",  header=None)
```

First, to see the number of rows (instances) and columns (features), you will use `shape`.

[3]:
```
df_car.shape
```

[3]:
```
(205, 25)
```

Next, examine the data by using the `head` method.

[4]:
```
df_car.head(5)
```

To make it easier to view the dataset when you start encoding, drop the columns that you won't use.

[31]:
```
df_car.columns
```

[31]:
```
Index(['symboling', 'normalized-losses', 'fuel-type', 'aspiration', 'num-of-doors', 'body-style', 'drive-wheels', 'engine-location', 'wheel-base', 'length', 'width', 'height', 'curb-weight', 'engine-type', 'num-of-cylinders', 'engine-size', 'fuel-system', 'bore', 'stroke', 'compression-ratio', 'horsepower', 'peak-rpm', 'city-mpg', 'highway-mpg', 'price'], dtype='object')
```

[7]:
```
# df_car = df_car[[ 'aspiration', 'num-of-doors',  'drive-wheels',  'num-of-cylinders']].copy()
```

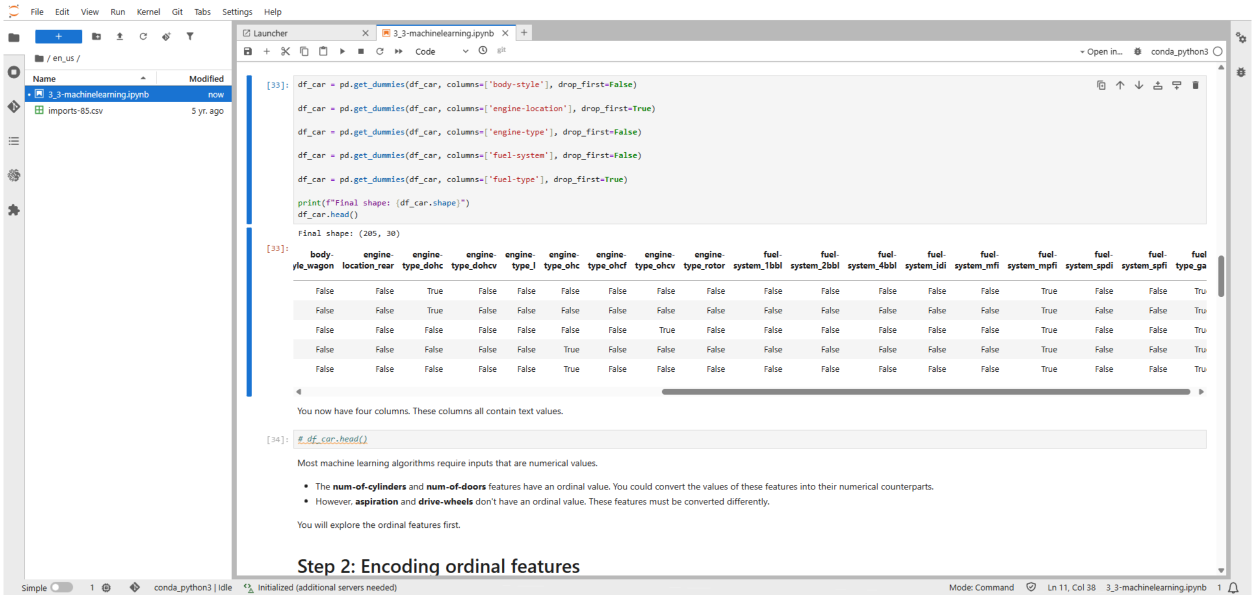
[32]:
```
columns_to_keep = ['symboling', 'fuel-type', 'aspiration', 'num-of-doors',
                   'body-style', 'drive-wheels', 'engine-location',
                   'engine-type', 'num-of-cylinders', 'fuel-system',
                   'price', 'horsepower', 'city-mpg']

df_car = df_car[columns_to_keep]
df_car.head()
```

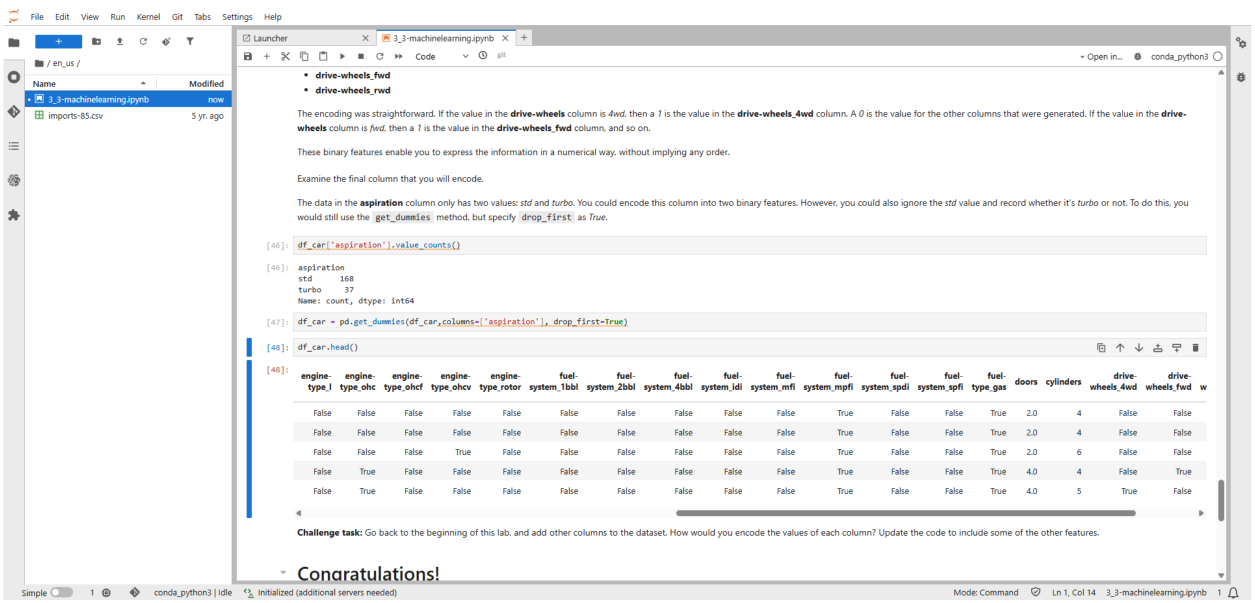
[32]:

	symboling	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	engine-type	num-of-cylinders	fuel-system	price	horsepower	city-mpg
0	3	gas	std	two	convertible	rwd	front	dohc	four	mpfi	13495.0	111.0	21
1	3	gas	std	two	convertible	rwd	front	dohc	four	mpfi	16500.0	111.0	21
2	1	gas	std	two	hatchback	rwd	front	ohcv	six	mpfi	16500.0	154.0	19
3	2	gas	std	four	sedan	fwd	front	ohc	four	mpfi	13950.0	102.0	24
4	2	gas	std	four	sedan	4wd	front	ohc	five	mpfi	17450.0	115.0	18

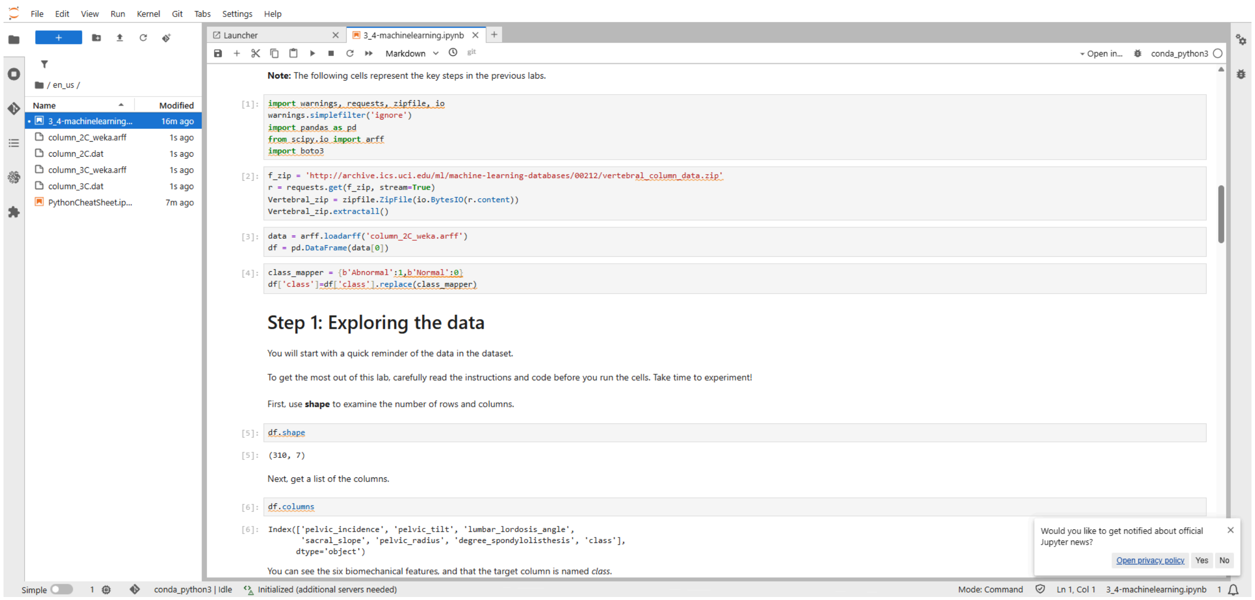
[33]:
```
df_car = pd.get_dummies(df_car, columns=['body-style'], drop_first=False)

df_car = pd.get_dummies(df_car, columns=['engine-location'], drop_first=True)

df_car = pd.get_dummies(df_car, columns=['engine-type'], drop_first=False)

df_car = pd.get_dummies(df_car, columns=['fuel-system'], drop_first=False)

df_car = pd.get_dummies(df_car, columns=['fuel-type'], drop_first=True)

print(f"Final shape: {df_car.shape}")
df_car.head()
```

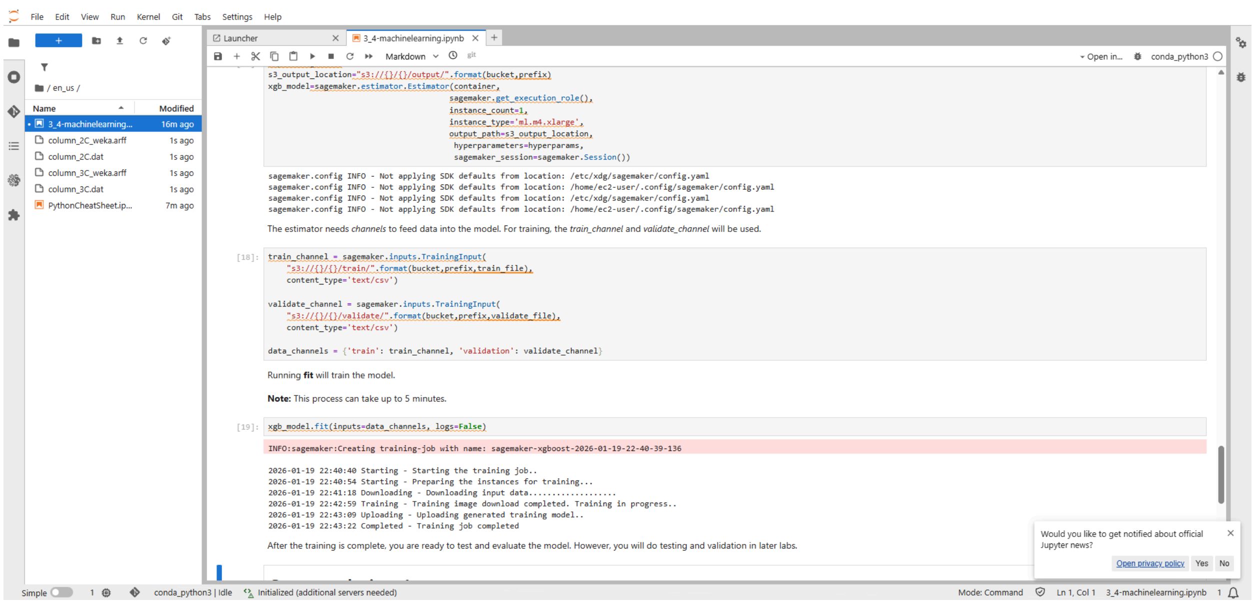
```
Final shape: (205, 30)
```



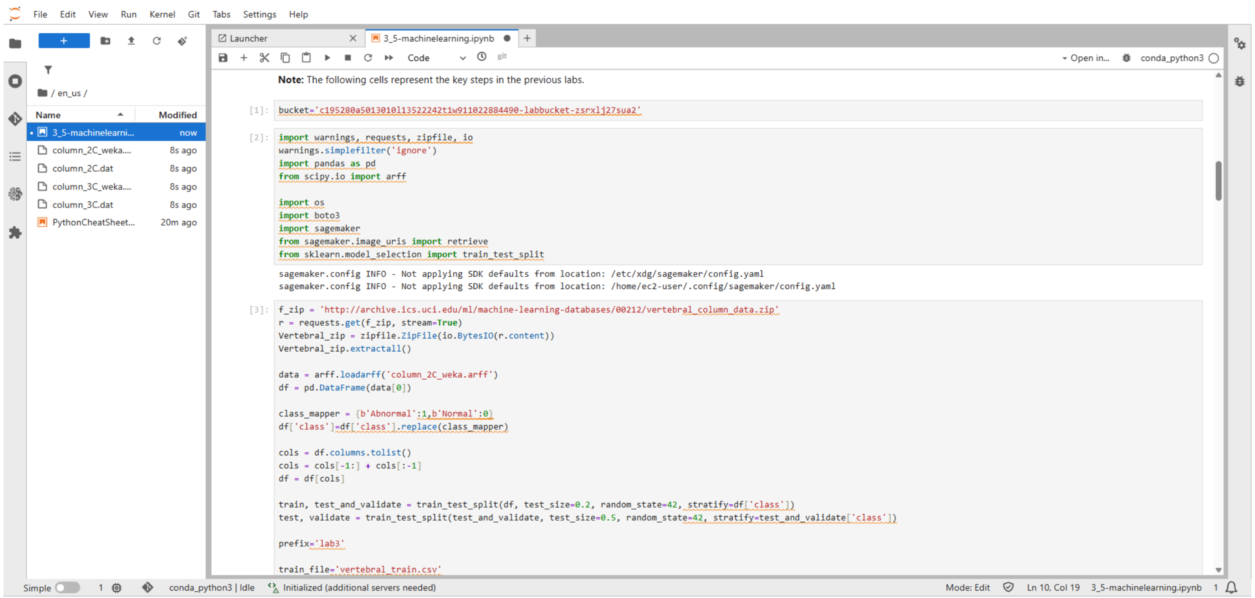


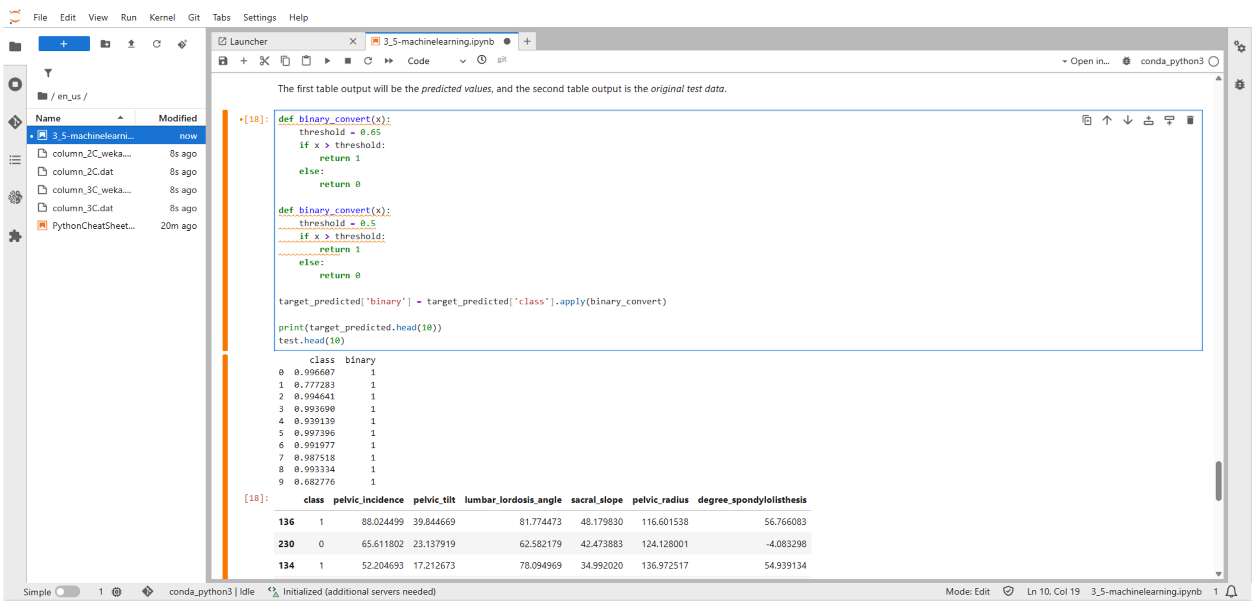
Lab 3.4 - Amazon SageMaker - Training a model





Lab 3.5 - Amazon SageMaker - Deploying a model

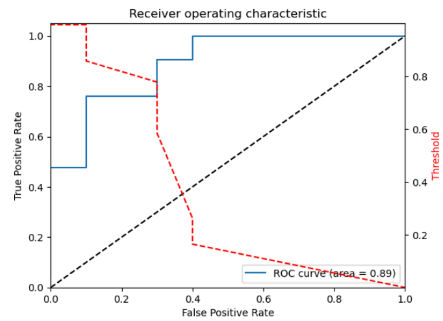




Lab 3.6 - Amazon SageMaker - Generating model performance metrics

3_6-machinelearning.ipynb

```python
plt.legend(loc="lower right")

roc_auc = auc(fpr, tpr)

if thresholds_finite.size > 0:
    ax2 = plt.gca().twinx()
    ax2.plot(fpr_finite, thresholds_finite, markeredgecolor='r', linestyle='dashed', color='r')
    ax2.set_ylabel('Threshold', color='r')
    ax2.set_ylim([thresholds_finite[-1], thresholds_finite[0]])
    ax2.set_xlim([fpr_finite[0], fpr_finite[-1]])

plt.show()
```



3_6-machinelearning.ipynb

Step 1: Exploring the results

The output from the model will be a probability. You must first convert that probability into one of the two classes, either *0* or *1*. To do this, you can create a function to perform the conversion. Note the use of the threshold in the function.

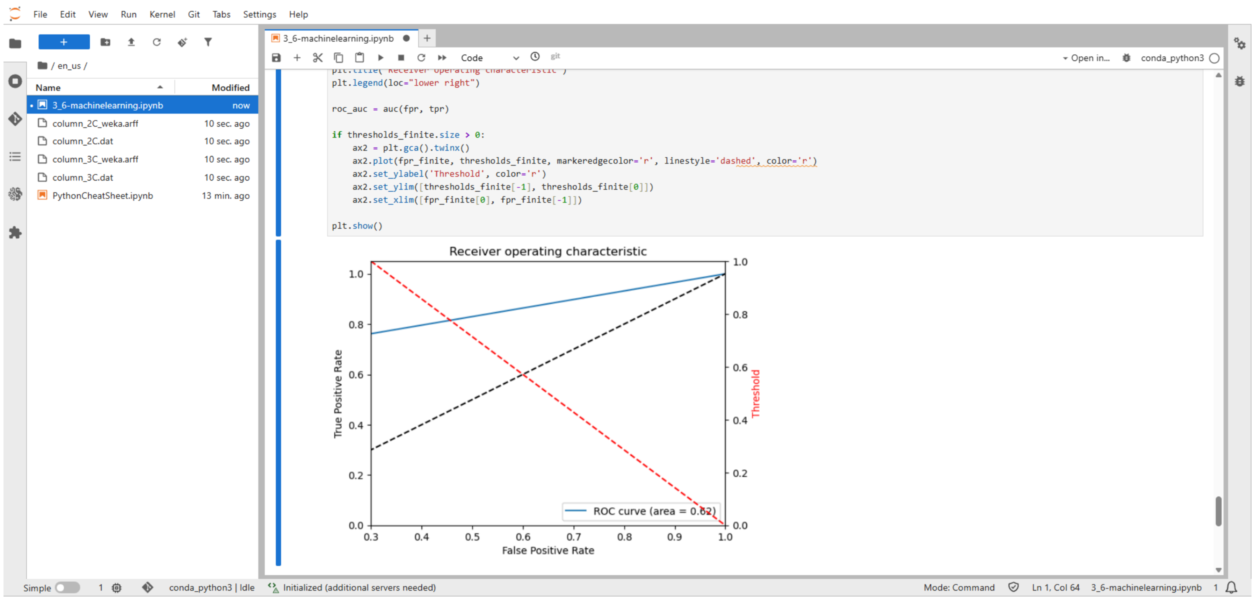
[42]:

```python
def binary_convert(x):
    # threshold = 0.3
    # threshold = 0.25
    threshold = 0.75
    if x > threshold:
        return 1
    else:
        return 0

target_predicted_binary = target_predicted['class'].apply(binary_convert)

print(target_predicted_binary.head(5))
test.head(5)
```

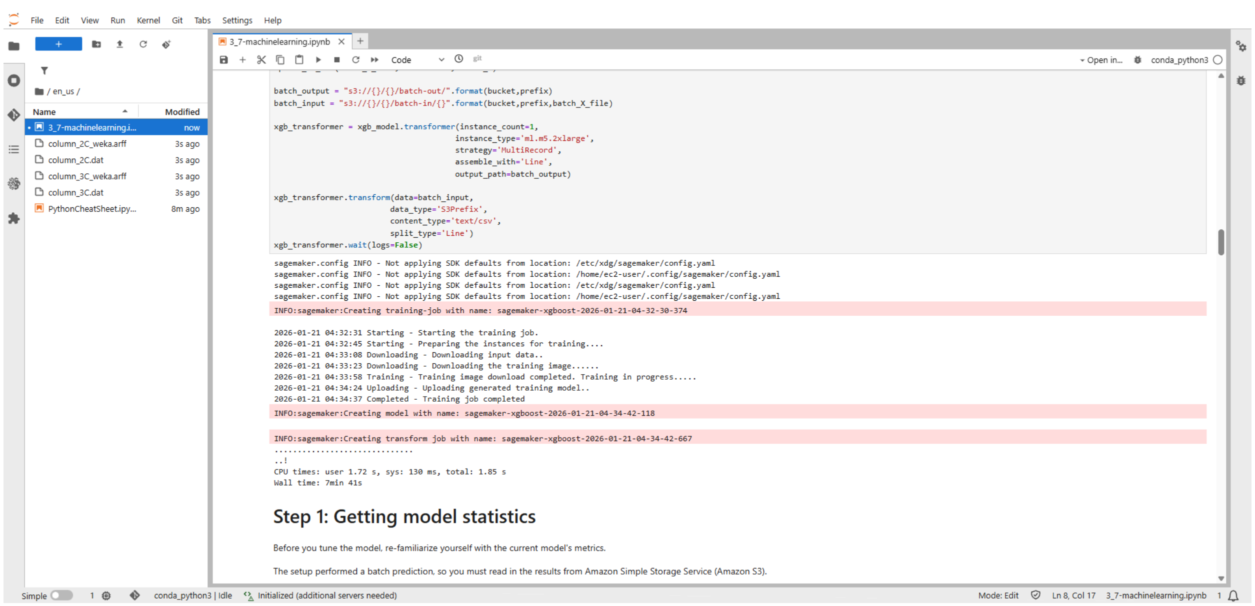
```
0    1
1    1
2    1
3    1
4    1
Name: class, dtype: int64
```

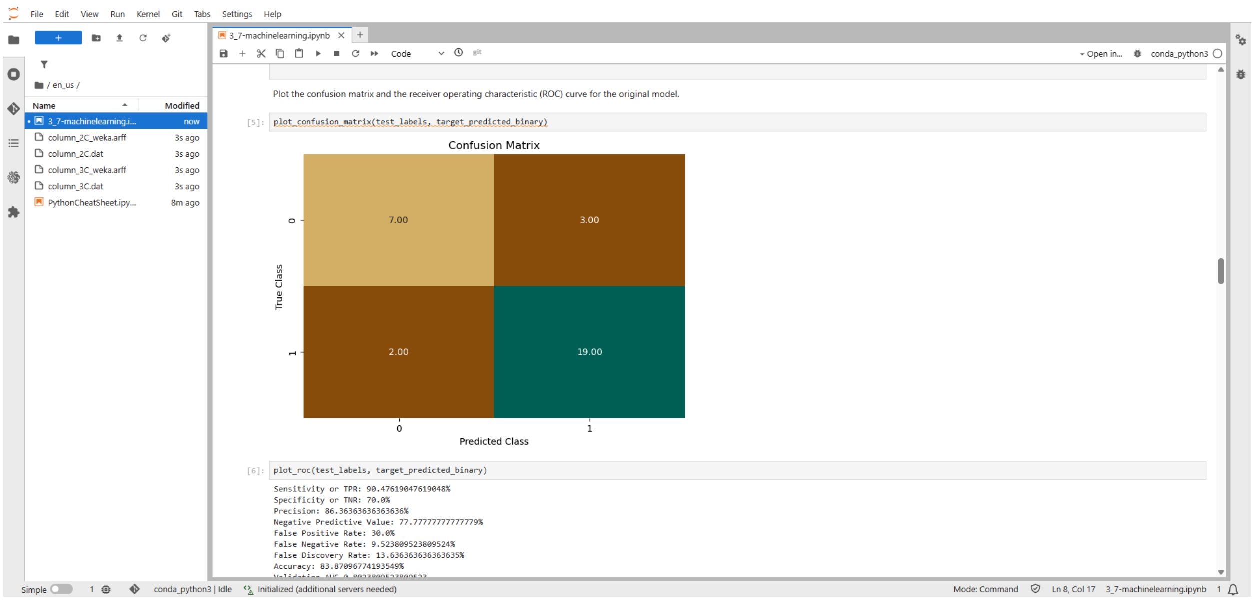
[42]:

	class	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degree_spondylolisthesis
136	1	88.024499	39.844669	81.774473	48.179830	116.601538	56.766083
230	0	65.611802	23.137919	62.582179	42.473883	124.128001	-4.083298
134	1	52.204693	17.212673	78.094969	34.992020	136.972517	54.939134
130	1	50.066786	9.120340	32.168463	40.946446	99.712453	26.766697
47	1	41.352504	16.577364	30.706191	24.775141	113.266675	-4.497958

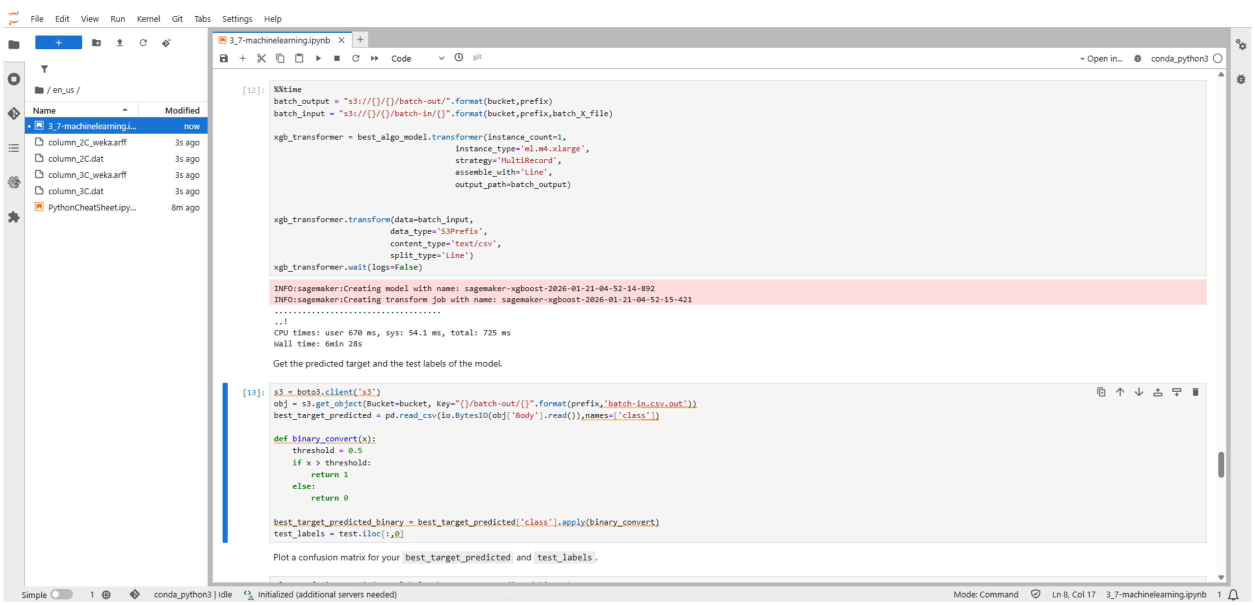


Lab 3.7 - Amazon SageMaker - Hyperparameter Tuning

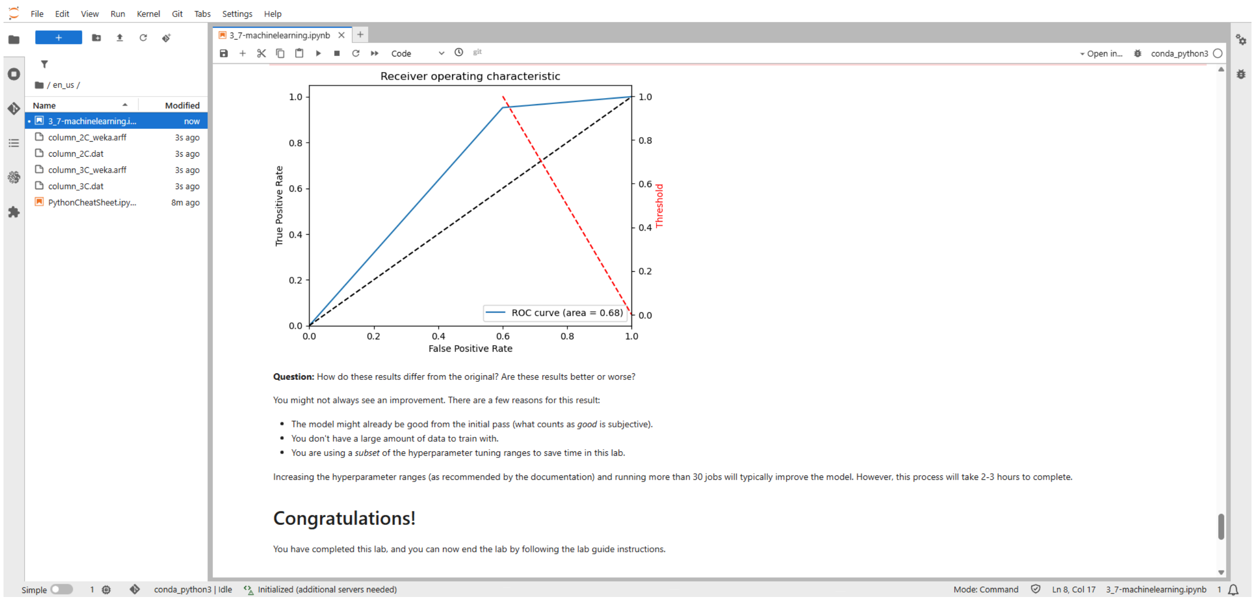




File Edit View Run Kernel Git Tabs Settings Help
3_7-machinelearning.ipynb
Code
Open in...
conda_python3
/ en_us /
Name
Modified
3_7-machinelearning.i...
now
column_2C_weka.arff
3s ago
column_2C.dat
3s ago
column_3C_weka.arff
3s ago
column_3C.dat
3s ago
PythonCheatSheet.ipy...
8m ago
Plot the confusion matrix and the receiver operating characteristic (ROC) curve for the original model.
[5]: plot_confusion_matrix(test_labels, target_predicted_binary)
Confusion Matrix
True Class
0
1
7.00
3.00
2.00
19.00
0
1
Predicted Class
[6]: plot_roc(test_labels, target_predicted_binary)
Sensitivity or TPR: 90.47619047619048%
Specificity or TNR: 70.0%
Precision: 86.36363636363636%
Negative Predictive Value: 77.77777777777779%
False Positive Rate: 30.0%
False Negative Rate: 9.523809523809524%
False Discovery Rate: 13.636363636363635%
Accuracy: 83.87096774193549%
Simple
conda_python3 | Idle
Initialized (additional servers needed)
Mode: Command
Ln 8, Col 17
3_7-machinelearning.ipynb



File Edit View Run Kernel Git Tabs Settings Help
3_7-machinelearning.ipynb
Code
Open in...
conda_python3
/ en_us /
Name
Modified
3_7-machinelearning.i...
now
column_2C_weka.arff
3s ago
column_2C.dat
3s ago
column_3C_weka.arff
3s ago
column_3C.dat
3s ago
PythonCheatSheet.ipy...
8m ago
[12]: %%time
batch_output = "s3://{}/{}/batch-out/".format(bucket,prefix)
batch_input = "s3://{}/{}/batch-in/{}".format(bucket,prefix,batch_X_file)
xgb_transformer = best_algo_model.transformer(instance_count=1,
instance_type='ml.m4.xlarge',
strategy='MultiRecord',
assemble_with='Line',
output_path=batch_output)
xgb_transformer.transform(data=batch_input,
data_type='S3Prefix',
content_type='text/csv',
split_type='Line')
xgb_transformer.wait(logs=False)
INFO:sagemaker:Creating model with name: sagemaker-xgboost-2026-01-21-04-52-14-892
INFO:sagemaker:Creating transform job with name: sagemaker-xgboost-2026-01-21-04-52-15-421
....
..!
CPU times: user 670 ms, sys: 54.1 ms, total: 725 ms
Wall time: 6min 28s
Get the predicted target and the test labels of the model.
[13]: s3 = boto3.client('s3')
obj = s3.get_object(Bucket=bucket, Key="{}/batch-out/{}".format(prefix,'batch-in.csv.out'))
best_target_predicted = pd.read_csv(io.BytesIO(obj['Body'].read()),names=['class'])
def binary_convert(x):
threshold = 0.5
if x > threshold:
return 1
else:
return 0
best_target_predicted_binary = best_target_predicted['class'].apply(binary_convert)
test_labels = test.iloc[:,0]
Plot a confusion matrix for your best_target_predicted and test_labels.
Simple
conda_python3 | Idle
Initialized (additional servers needed)
Mode: Command
Ln 8, Col 17
3_7-machinelearning.ipynb


File Edit View Run Kernel Git Tabs Settings Help
/ en_us /
Name
Modified
3_7-machinelearning.i... now
column_2C_weka.arff 3s ago
column_2C.dat 3s ago
column_3C_weka.arff 3s ago
column_3C.dat 3s ago
PythonCheatSheet.ipy... 8m ago
3_7-machinelearning.ipynb
Code
Open in...
conda_python3
Receiver operating characteristic
True Positive Rate
False Positive Rate
Threshold
ROC curve (area = 0.68)
0.0
0.2
0.4
0.6
0.8
1.0
Question: How do these results differ from the original? Are these results better or worse?
You might not always see an improvement. There are a few reasons for this result:
The model might already be good from the initial pass (what counts as good is subjective).
You don't have a large amount of data to train with.
You are using a subset of the hyperparameter tuning ranges to save time in this lab.
Increasing the hyperparameter ranges (as recommended by the documentation) and running more than 30 jobs will typically improve the model. However, this process will take 2-3 hours to complete.
Congratulations!
You have completed this lab, and you can now end the lab by following the lab guide instructions.
Simple
1
conda_python3 | Idle
Initialized (additional servers needed)
Mode: Command
Ln 8, Col 17
3_7-machinelearning.ipynb
1