

Name of Student: Pushkar Sane		
Roll Number: 45		Lab Assignment Number: 5 & 6
Title of Lab Assignment: Feature Engineering, one hot Encoding, Normalization, Standardization, EDA using SageMaker DataWrangler Linear and Multiple Linear Regression using SageMaker.		
DOP: 16-03-2024		DOS: 16-03-2024
CO Mapped: CO3	PO Mapped: PO2, PO3, PO4, PO5, PO6, PO7, PSO1, PSO2	Signature:

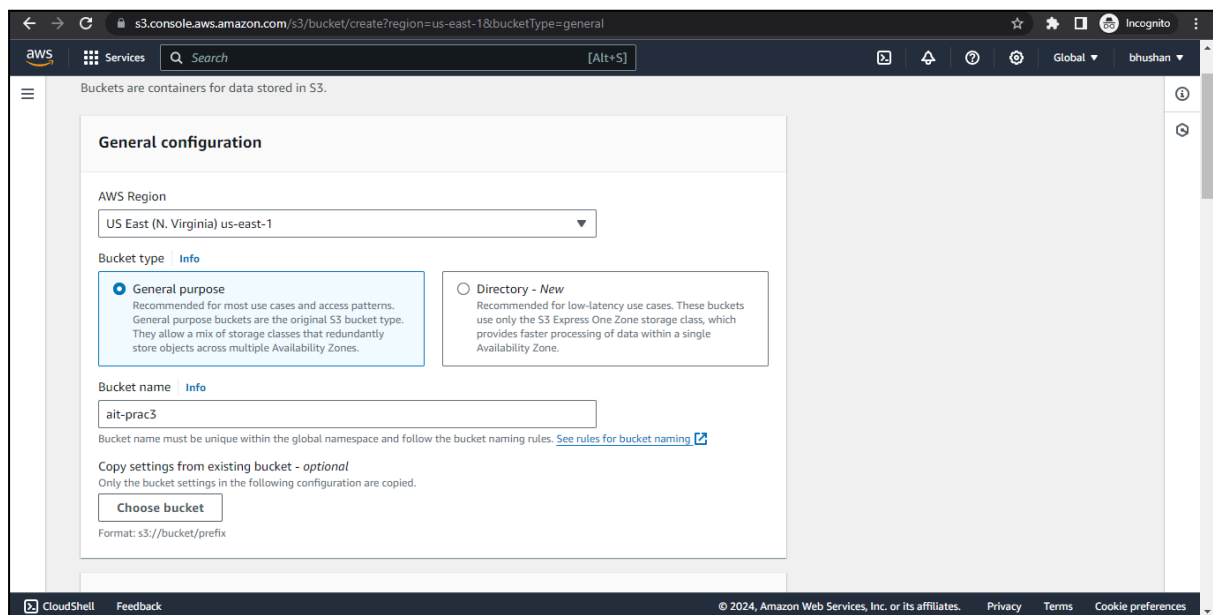
Practical 5 & 6

Aim: Feature Engineering, one hot Encoding, Normalization, Standardization, EDA using SageMaker DataWrangler Linear and Multiple Linear Regression using SageMaker.

Description:

Prerequisite

Create an S3 bucket and keep aside (it will be used later)



s3.console.aws.amazon.com/s3/bucket/create?region=us-east-1&bucketType=general

General configuration

AWS Region
US East (N. Virginia) us-east-1

Bucket type [Info](#)

☒ **General purpose**
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

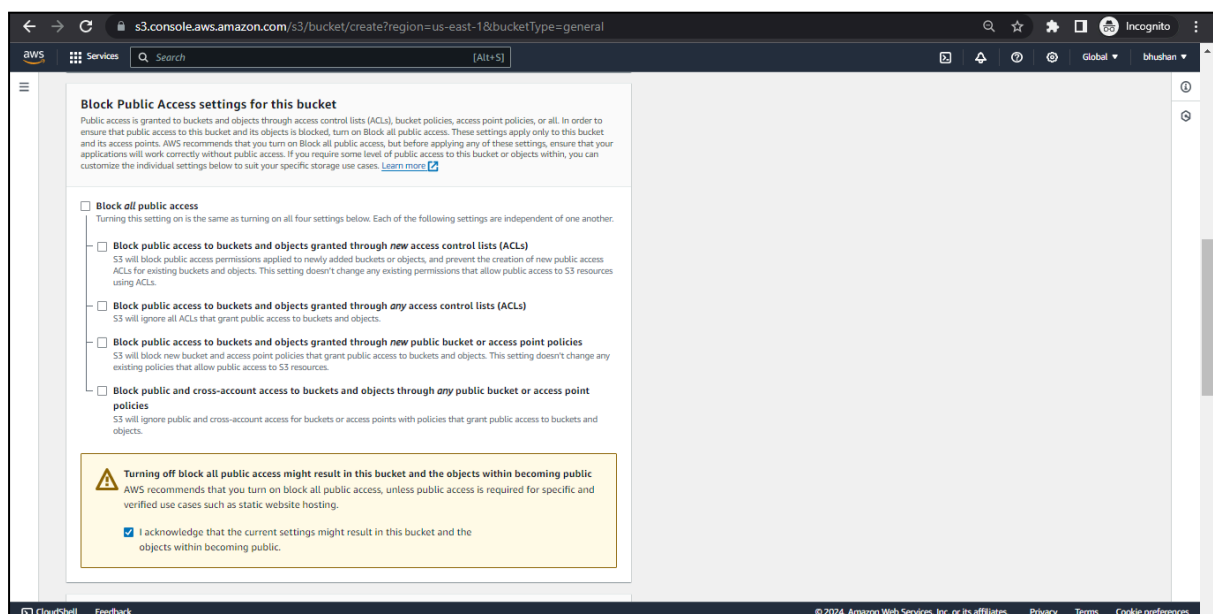
☐ **Directory - New**
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name [Info](#)
ait-prac3
Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

Format: s3://bucket/prefix



s3.console.aws.amazon.com/s3/bucket/create?region=us-east-1&bucketType=general

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☒ **Block all public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

☐ **Block public access to buckets and objects granted through new access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

☐ **Block public access to buckets and objects granted through any access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.

☐ **Block public access to buckets and objects granted through new public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

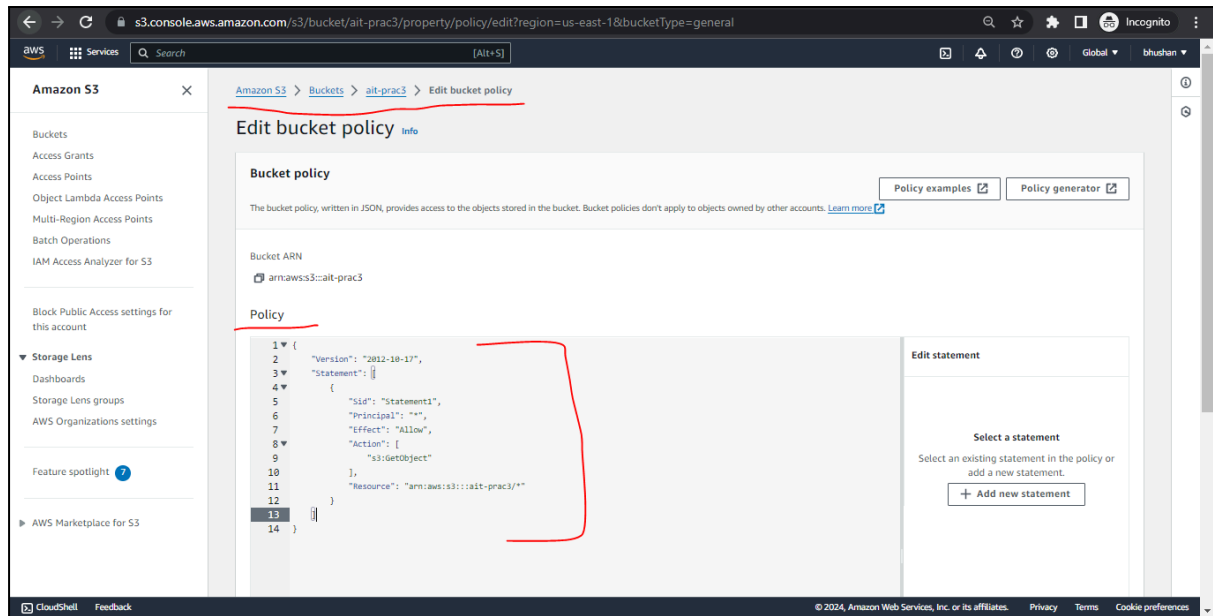
☐ **Block public and cross-account access to buckets and objects through any public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

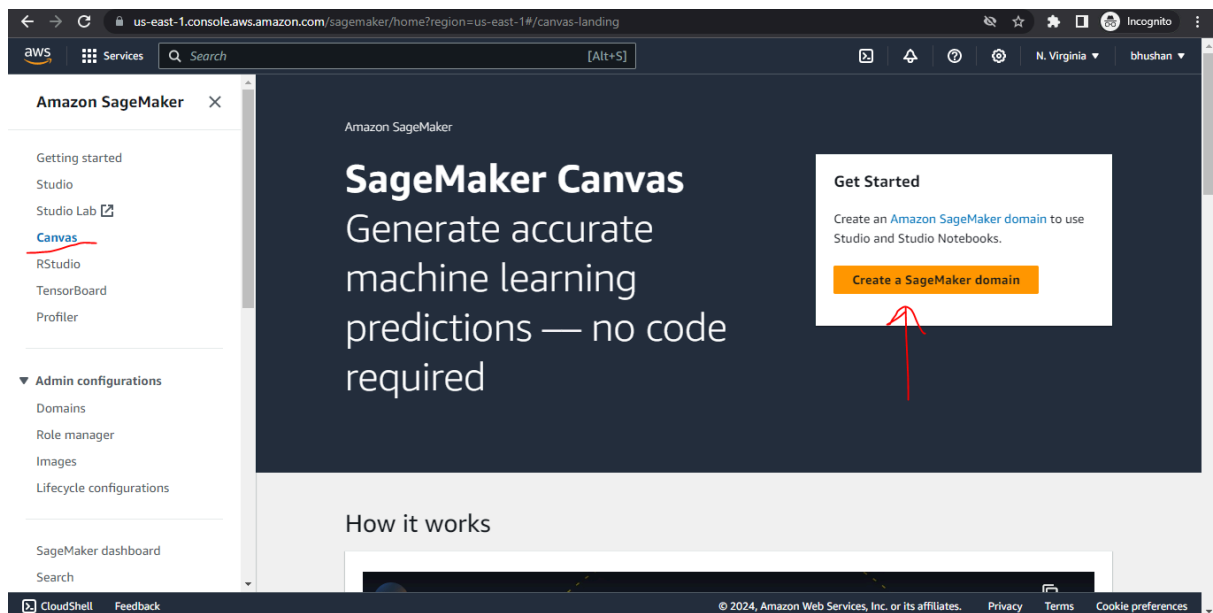
☒ I acknowledge that the current settings might result in this bucket and the objects within becoming public.

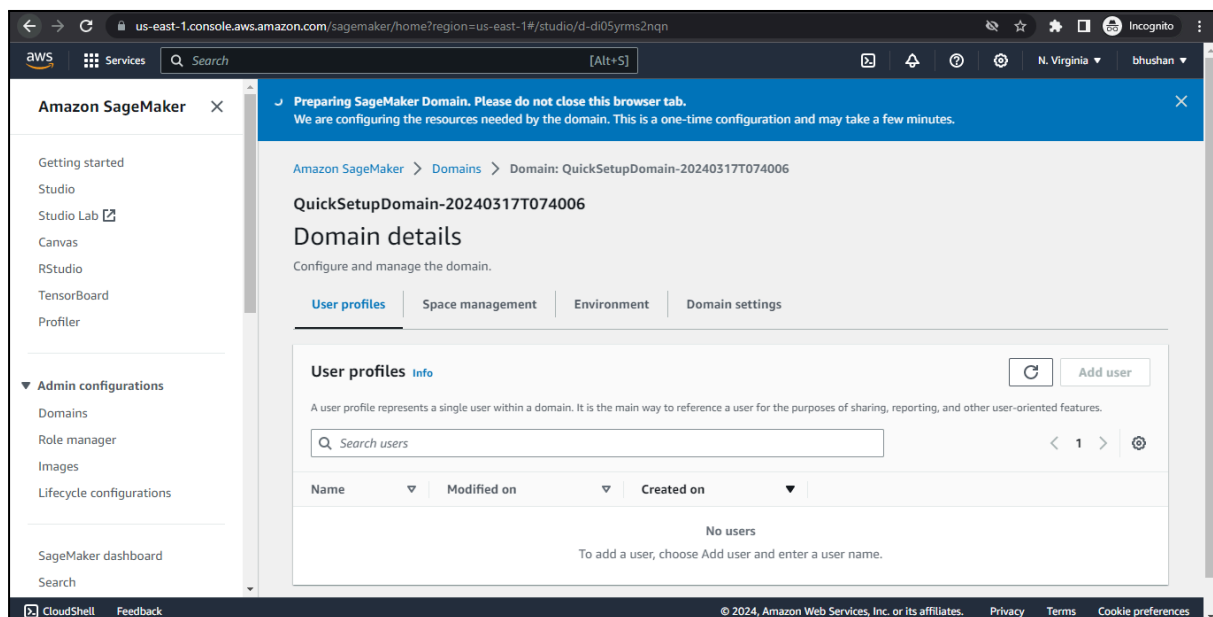
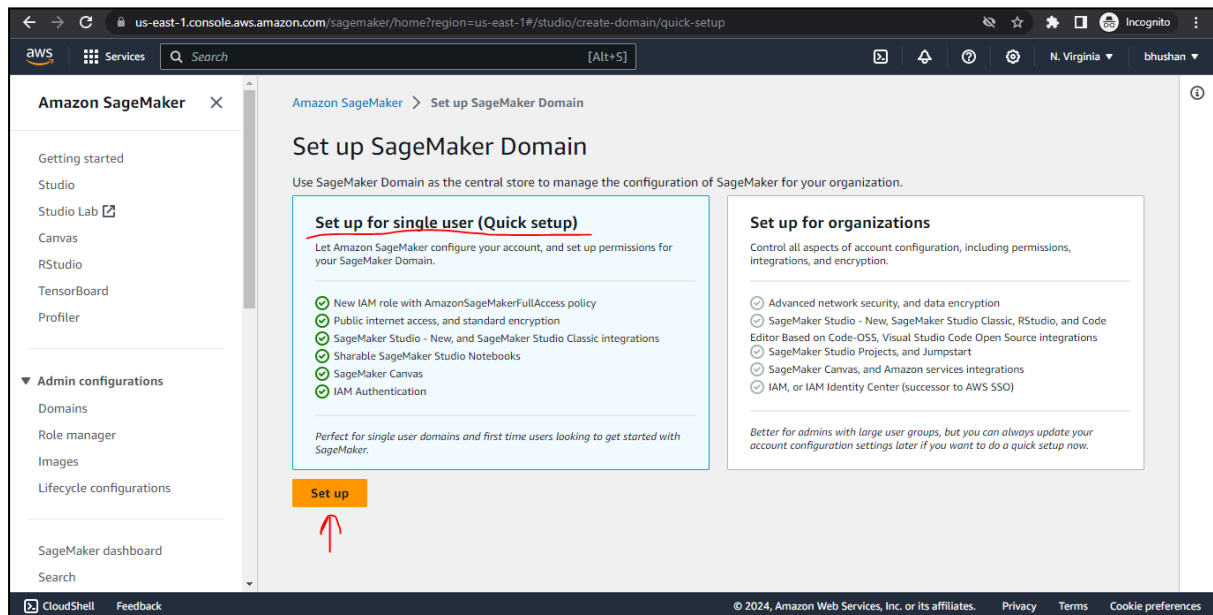
Click on "create" to create the bucket

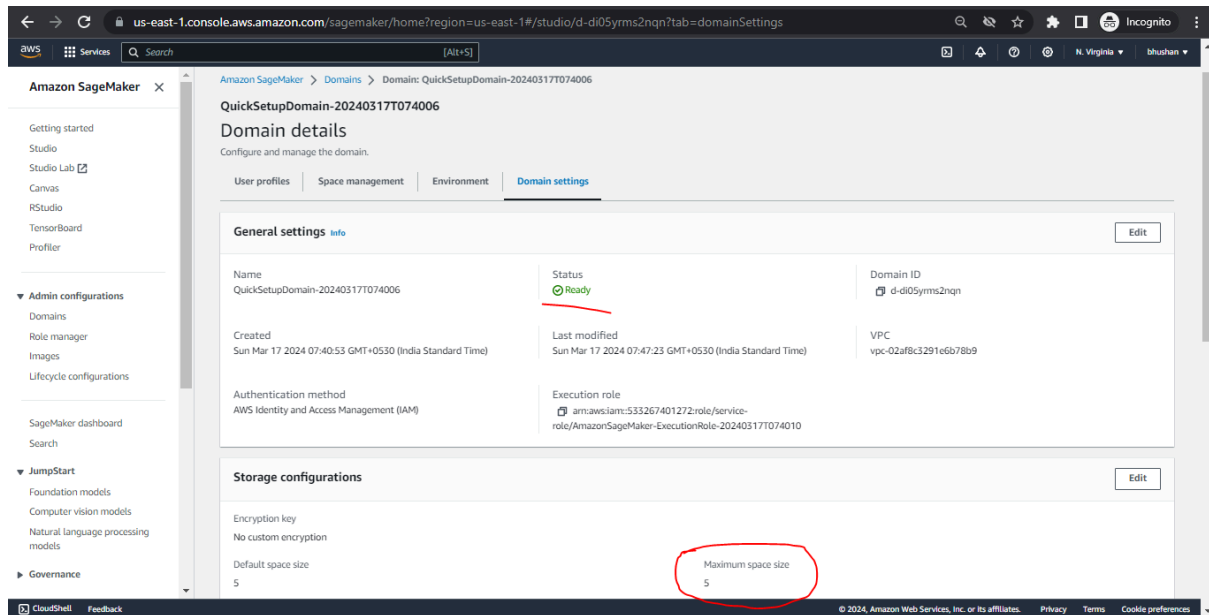
Change the permissions



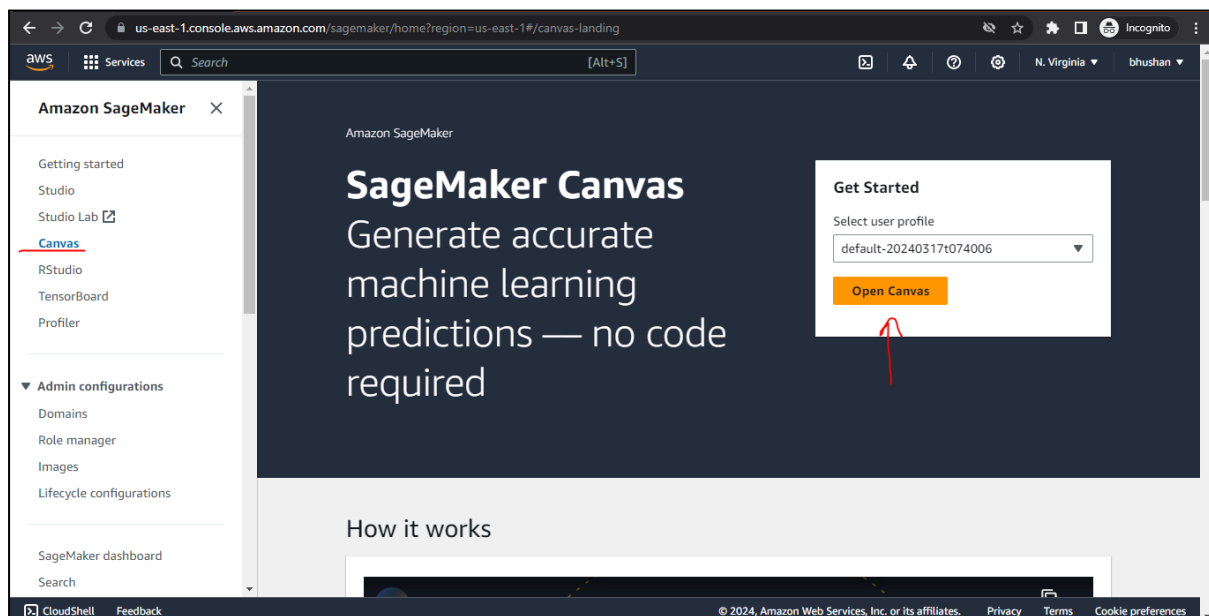
SageMaker Canvas



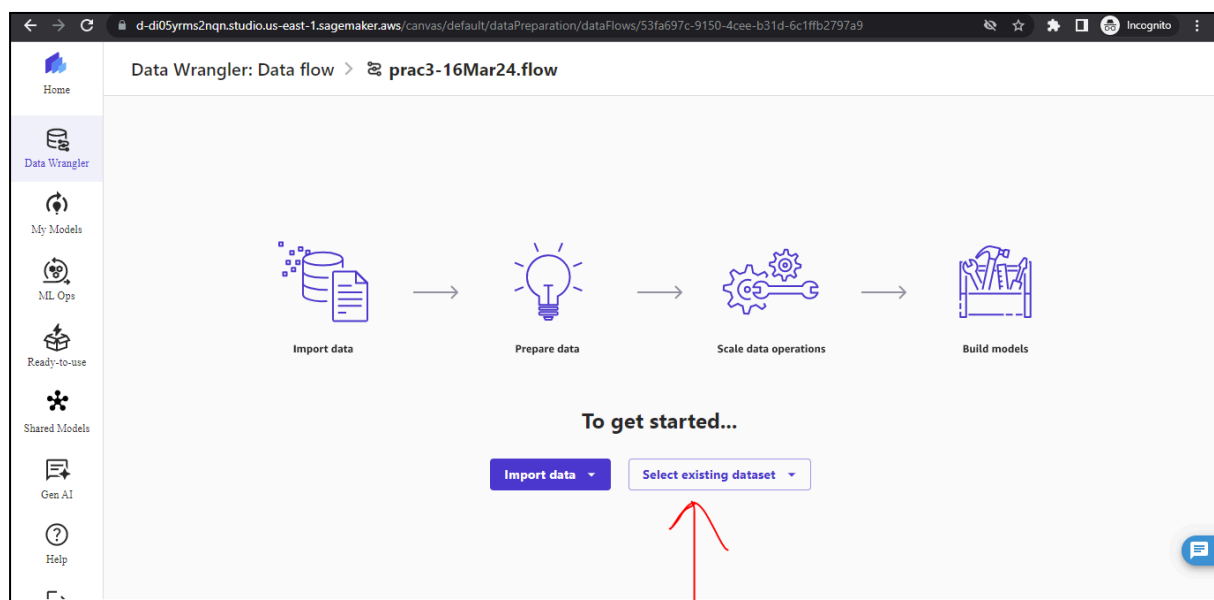
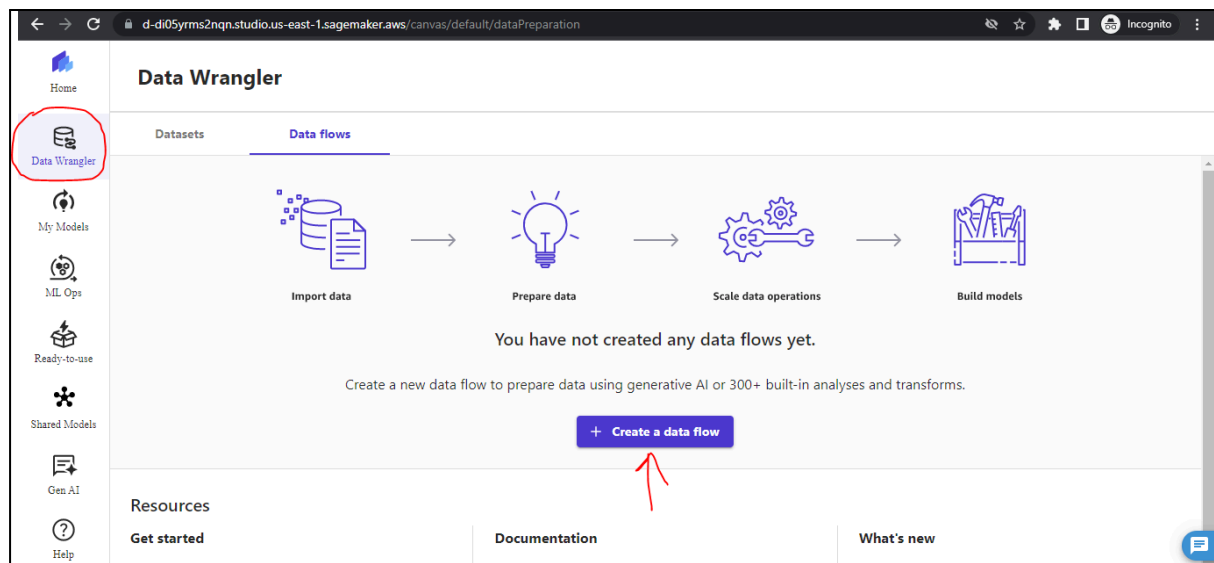


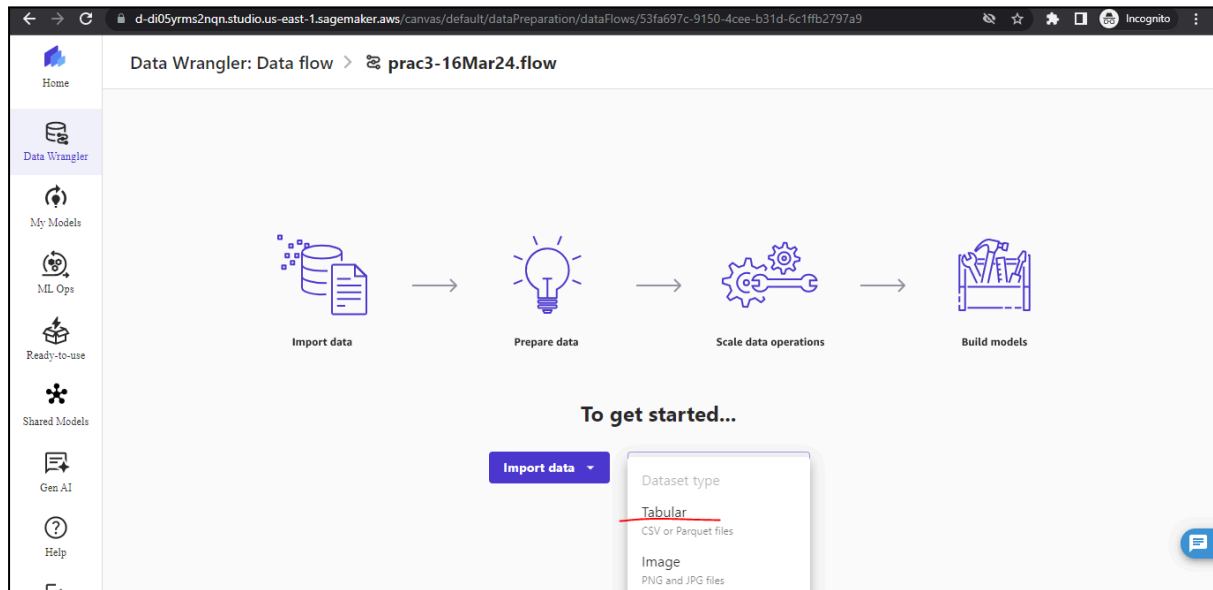


Data Preparation using SageMaker DataWrangler



Import Dataset

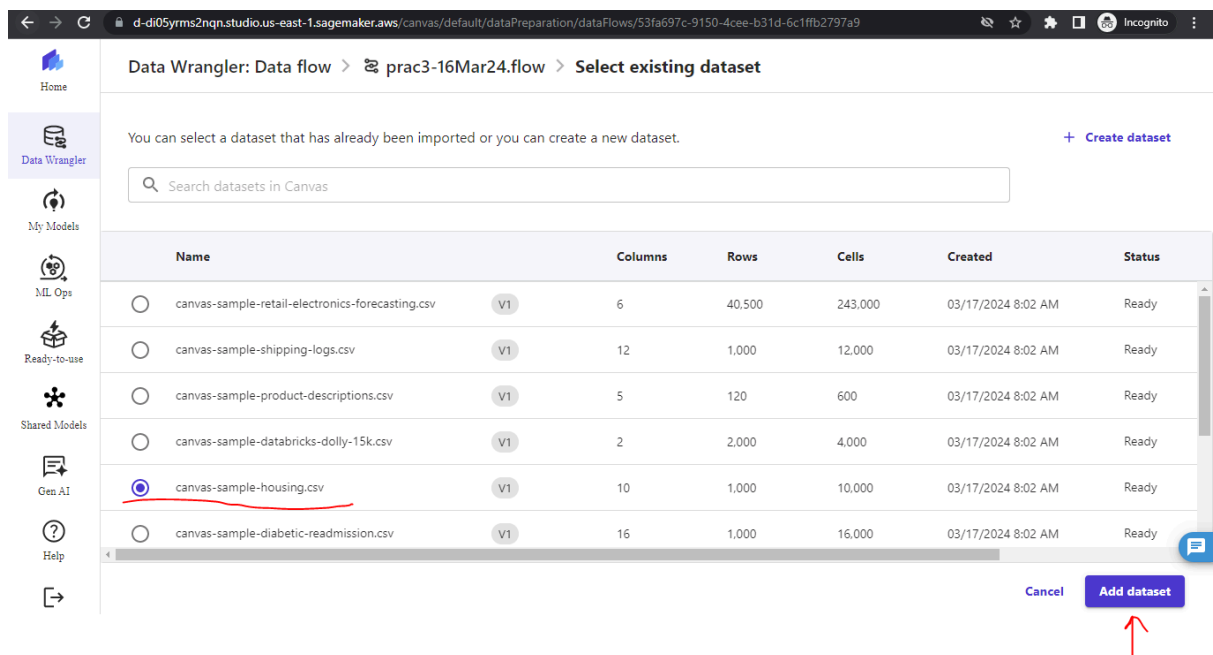




Reference for sample datasets explanation:

<https://docs.aws.amazon.com/sagemaker/latest/dg/canvas-sample-datasets.html>

canvas-sample-housing.csv: This dataset contains data on the characteristics tied to a given housing price. You can use this dataset to predict housing prices. Use the **median_house_value** column as the target column, and use the numeric prediction model type with this dataset. To learn more about building a model with this dataset, see the [SageMaker Canvas workshop page](#). This is the California housing dataset obtained from the [StatLib repository](#).



Delete irrelevant columns

The screenshot shows the AWS Data Wrangler interface for a data flow named 'prac3-16Mar24.flow' with a canvas sample housing.csv. The interface is divided into three main sections: a left sidebar with navigation icons, a central workspace, and a right-hand 'Steps' panel.

Central Workspace:

- Step 2. Data types:** Displays histograms for 'longitude (float)', 'latitude (float)', and 'housing_median_age'. Below the histograms is a table showing the first 2,000 rows of data.
- Table Data:**

longitude (float)	latitude (float)	housing_median_age
-122.23	37.88	41
-122.22	37.86	21
-122.24	37.85	52
-122.25	37.85	52
-122.25	37.85	52

☒ Show in-column visualizations for the first 2,000 rows. Visualize the full dataset. [Run Data quality and insights report](#)

Steps Panel:

- + Add step** (indicated by a red arrow)
- 1. S3 Source
- 2. Data types

This screenshot shows the same AWS Data Wrangler interface, but with the 'Steps' panel expanded to show additional data transformation options.

Steps Panel (Expanded):

- Handle missing:** Replace, drop, or add indicators for missing values.
- Handle outliers:** Remove or replace outlier numeric and categorical values.
- Handle structured column:** Flatten JSON and perform other operations on structured data.
- Manage columns:** Move, drop, duplicate or rename columns in the dataset.
- Manage rows:** Sort the rows, randomly shuffle their order, remove duplicates, remove any empty rows, or filter out/drop any rows that do not match a specified pattern.
- Manage vectors:** Expand or create vector columns.
- Parse column as type:** Cast a column to a new data type.
- Process numeric:** Transform numeric values to improve machine learning model performance.
- Sampling**

Data Wrangler: Data flow > prac3-16Mar24.flow > canvas-sample-housing.csv

Step 2. Data types

longitude (float)	latitude (float)	housing_median_age
-122.34 - -121.61	37.47 - 37.9	2 - 52
-122.23	37.88	41
-122.22	37.86	21
-122.24	37.85	52
-122.25	37.85	52

☒ Show in-column visualizations for the first 2,000 rows. Visualize the full dataset, [Run Data quality and insights report](#)

Manage columns

Move, drop, duplicate or rename columns in the dataset. [Learn more.](#)

Transform Required

Drop column × ▼

Columns to drop Required

latitude × longitude × × ▼

[Clear](#) [Preview](#) [Add](#)

Ordinal Encoding for categorical column

Data Wrangler: Data flow > prac3-16Mar24.flow > canvas-sample-housing.csv

Step 3. Drop column

population (float)	households (float)	median_income (float)	median_house_value (float)	ocean_proximity (string)
18 - 12203	7 - 3701	0.4999 - 13.499	60000 - 5.0000e+5	3 Categories
322	126	8.3252	452600	NEAR BAY
2401	1138	8.3014	356900	NEAR BAY
496	177	7.2574	352100	NEAR BAY
558	219	5.6431	341300	NEAR BAY
565	259	3.8462	342200	NEAR BAY
413	193	4.0368	269700	NEAR BAY
1094	514	3.5591	299200	NEAR BAY

Steps

- 1. S3 Source
- 2. Data types
- 3. Drop column

[+ Add step](#)

Data Wrangler: Data flow > prac3-16Mar24.flow > canvas-sample-housing.csv

Step 3. Drop column

population (float)	households (float)	median_income (float)	median_house_value (float)	ocean_proximity (string)
18 - 12203	7 - 3701	0.4999 - 13.499	60000 - 5.0000e+5	3 Categories
322	126	8.3252	452600	NEAR BAY
2401	1138	8.3014	356900	NEAR BAY
496	177	7.2574	352100	NEAR BAY
558	219	5.6431	341300	NEAR BAY
565	259	3.8462	342200	NEAR BAY
413	193	4.0368	269700	NEAR BAY
1094	514	3.5591	299200	NEAR BAY
1157	647	3.12	241400	NEAR BAY
1206	595	2.0804	226700	NEAR BAY
1551	714	3.6912	261100	NEAR BAY
910	402	3.2031	261500	NEAR BAY
1504	734	3.2705	241800	NEAR BAY
1098	468	3.075	213500	NEAR BAY

Add transform

Search transforms

Frequently used

- Manage columns**
Move, drop, duplicate or rename columns in the dataset.
- Custom**
Custom formula
Define a new column using a Spark SQL expression to query data in the current dataframe.
Custom transform
Use PySpark, Pandas, or PySpark (SQL) to define custom transformations.
- Standard**
Balance data
Balance the data for binary classification problems using random oversampling, random undersampling or SMOTE.
Dimensionality Reduction
For the top K principal components, trains a model to project vectors to a lower dimensional space.
Encode categorical
Convert categorical variables to numeric or vector representations.
Featureize date/time
Encode date/time values to numeric and vector representations. Extract features from dates, such as year, day, week, and more.
Featureize text
Generate vector representations from natural language text.

The screenshot shows the AWS Data Wrangler interface. The main panel displays 'Step 3. Drop column' with a table of data. The table has columns: population (float), households (float), and median_income (float). The data is as follows:

population (float)	households (float)	median_income (float)
322	126	8.3252
2401	1138	8.3014
496	177	7.2574
558	219	5.6431
565	259	3.8462
413	193	4.0368

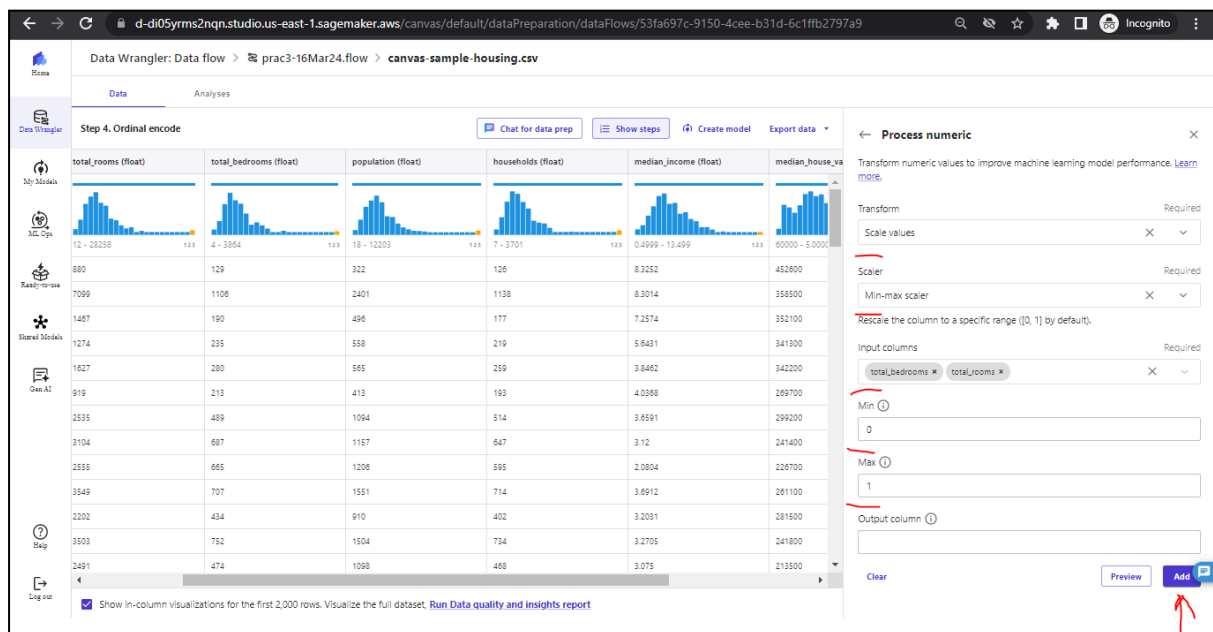
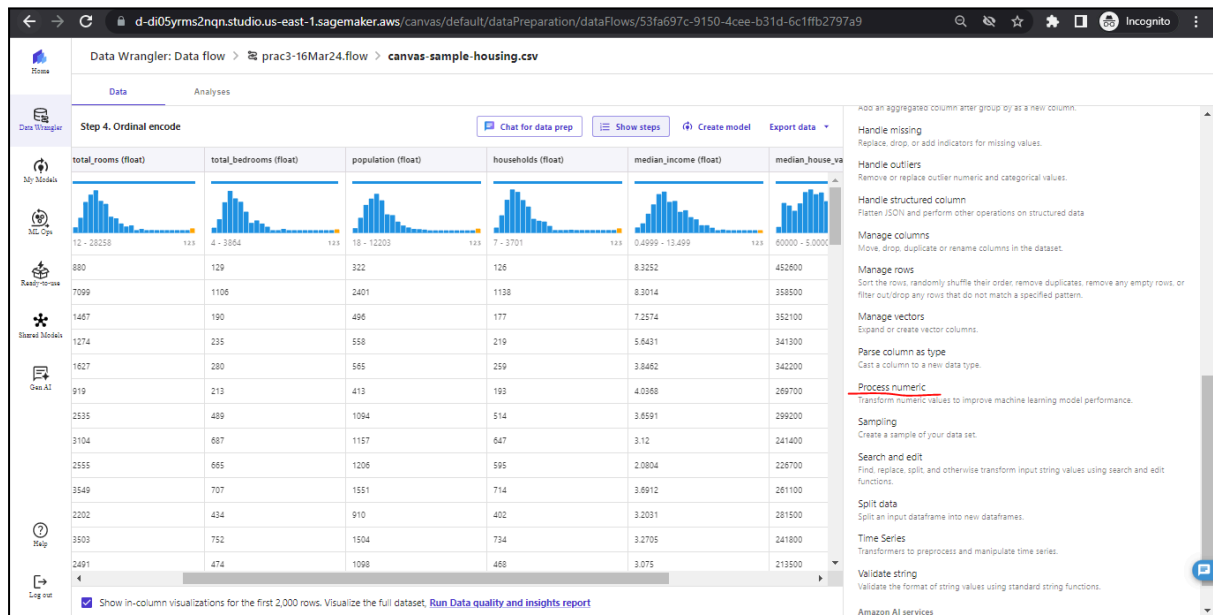
On the right, the 'Encode categorical' panel is open. It shows the 'Transform' dropdown set to 'Ordinal encode'. The 'Input columns' dropdown is set to 'ocean_proximity'. The 'Output column' field is empty. The 'Invalid handling strategy' dropdown is set to 'Skip'. There are 'Clear', 'Preview', and 'Add' buttons at the bottom of the panel. A red arrow points to the 'Add' button.

Min-Max Scaling

The screenshot shows the AWS Data Wrangler interface. The main panel displays 'Step 4. Ordinal encode' with a table of data. The table has columns: total_rooms (float), total_bedrooms (float), population (float), households (float), median_income (float), and median_house_value (float). The data is as follows:

total_rooms (float)	total_bedrooms (float)	population (float)	households (float)	median_income (float)	median_house_value (float)
880	129	322	126	8.3252	452600
7099	1106	2401	1138	8.3014	358600
1467	190	496	177	7.2574	352100
1274	235	558	219	5.6431	341300
1627	280	565	259	3.8462	342200
919	213	413	193	4.0368	269700
2535	489	1094	514	3.6591	299200
3104	687	1157	647	3.12	241400
2555	665	1206	595	2.0804	228700
3549	707	1551	714	3.6912	261100
2202	434	910	402	3.2031	281500
3503	752	1504	734	3.2705	241800
2491	474	1098	468	3.075	213500

On the right, the 'Steps' panel is open. It shows a list of steps: 1. S3 Source, 2. Data types, 3. Drop column, and 4. Ordinal encode. A red arrow points to the 'Add step' button at the top of the panel.



EDA using SageMaker DataWrangler

Table Summary

Data Wrangler: Data flow > prac3-16Mar24.flow > canvas-sample-housing.csv

Table Summary: Untitled

No Preview available

	median_income (float)	median_house_value (float)	ocean_proximity (float)
	8.3252	452600	0
	8.3014	358500	0
	7.2574	352100	0
	5.6431	341300	0

Create analysis

Analysis type: Table Summary (Required)

A limit of 20,000 rows is used for this analysis.

Analysis name: Untitled

Clear Preview Add

Data Wrangler: Data flow > prac3-16Mar24.flow > canvas-sample-housing.csv

Table Summary: Untitled

summary (string)	count (string)	mean (string)	stddev
median_income	1000	3.8429933000000025	1.81
median_house_value	1000	209518.612	890
ocean_proximity	1000	0.14	0.42

Create analysis

Analysis type: Table Summary (Required)

A limit of 20,000 rows is used for this analysis.

Analysis name: Untitled

Clear Preview Add

Checking if the target column is normally distributed (bell shaped graph)

Incognito

Data Wrangler: Data flow > prac3-16Mar24.flow > canvas-sample-housing.csv

Data Analyses

Histogram: Untitled

No Preview available

	population (float)	households (float)	median_income (float)	median_house_value (float)	ocean_proximity (float)
	322	126	8.3252	452600	0
	2401	1138	8.3014	358500	0
	496	177	7.2574	352100	0
	558	219	5.6431	341300	0
	565	259	3.8462	342200	0
	413	193	4.0368	269700	0
	1094	514	3.6591	299200	0
	1157	647	3.12	241400	0

Create analysis

Analysis type: Histogram

Analysis name: Untitled

X axis: median_house_value

Color by: Select...

Facet by: Select...

Clear Preview Add

Incognito

Data Wrangler: Data flow > prac3-16Mar24.flow > canvas-sample-housing.csv

Data Analyses

Histogram: Untitled

	population (float)	households (float)	median_income (float)	median_house_value (float)	ocean_proximity (float)
	322	126	8.3252	452600	0
	2401	1138	8.3014	358500	0
	496	177	7.2574	352100	0
	558	219	5.6431	341300	0
	565	259	3.8462	342200	0

Create analysis

Analysis type: Histogram

Analysis name: Untitled

X axis: median_house_value

Color by: Select...

Facet by: Select...

Clear Preview Add

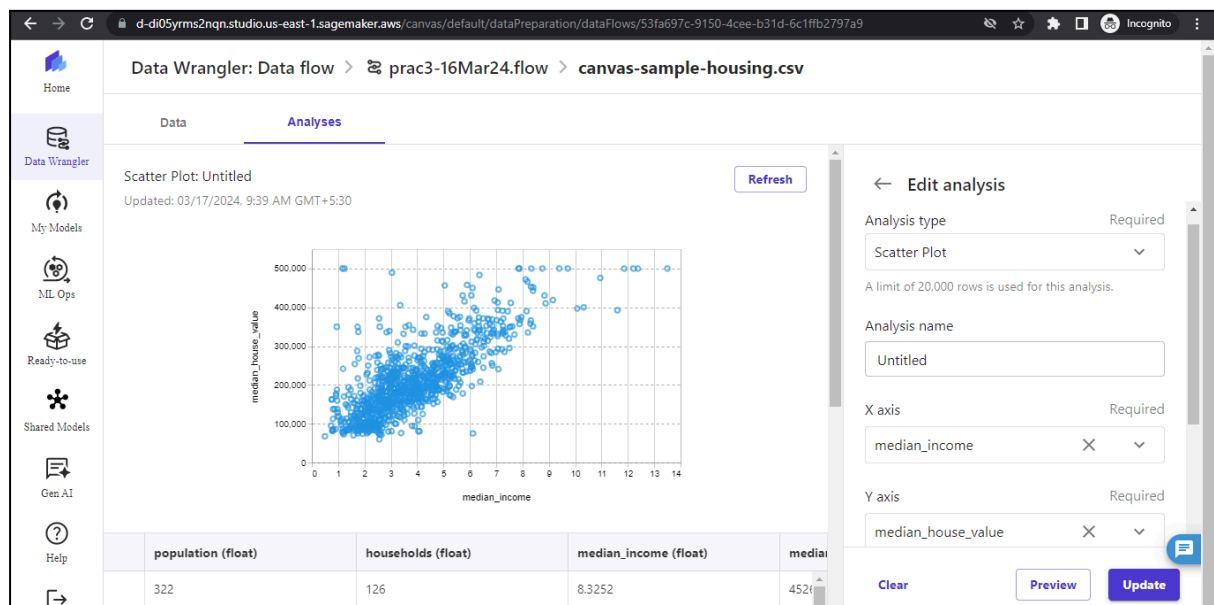
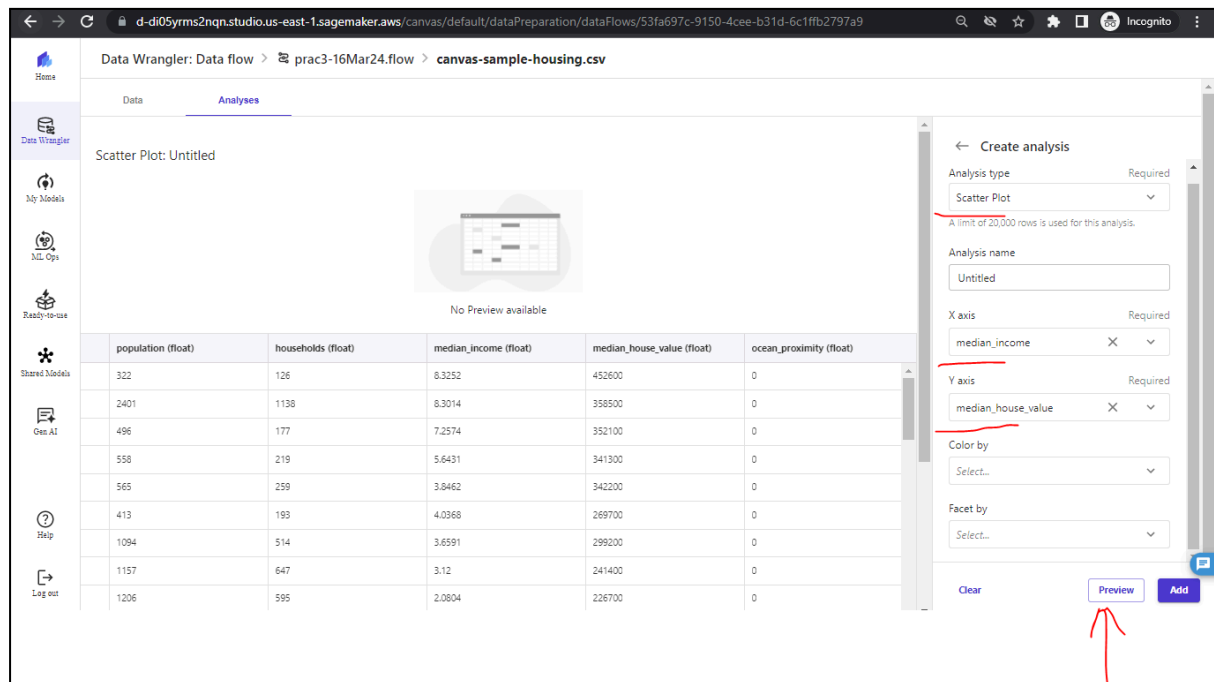
Data is normally distributed but slightly skewed towards the left

Since this is a sample dataset, we can proceed with a slightly skewed data also

Checking if the relationship between the predictor and target variable is linear

The **dependent variable** is called the **target** and the **independent variable** is called the **predictor**

The **dependent variable (target)** depends on the **independent variable (predictor)**



The columns' relation is roughly linear

The relationship looks roughly linear, so we can proceed with the linear regression.

Export prepared data

The screenshot shows the Amazon SageMaker Data Wrangler interface. The top navigation bar indicates the current data flow is 'prac3-16Mar24.flow' and the dataset is 'canvas-sample-housing.csv'. The left sidebar contains navigation options: Home, Data Wrangler, My Models, ML Ops, Ready-to-use, Shared Models, Gen AI, Help, and a search icon. The main area is divided into 'Data' and 'Analyses' tabs. The 'Analyses' tab is active, showing a scatter plot titled 'Scatter Plot: Untitled' with the x-axis labeled 'median_income' and the y-axis labeled 'median_house_value'. The plot shows a positive correlation between income and house value. Below the plot is a table with the following data:

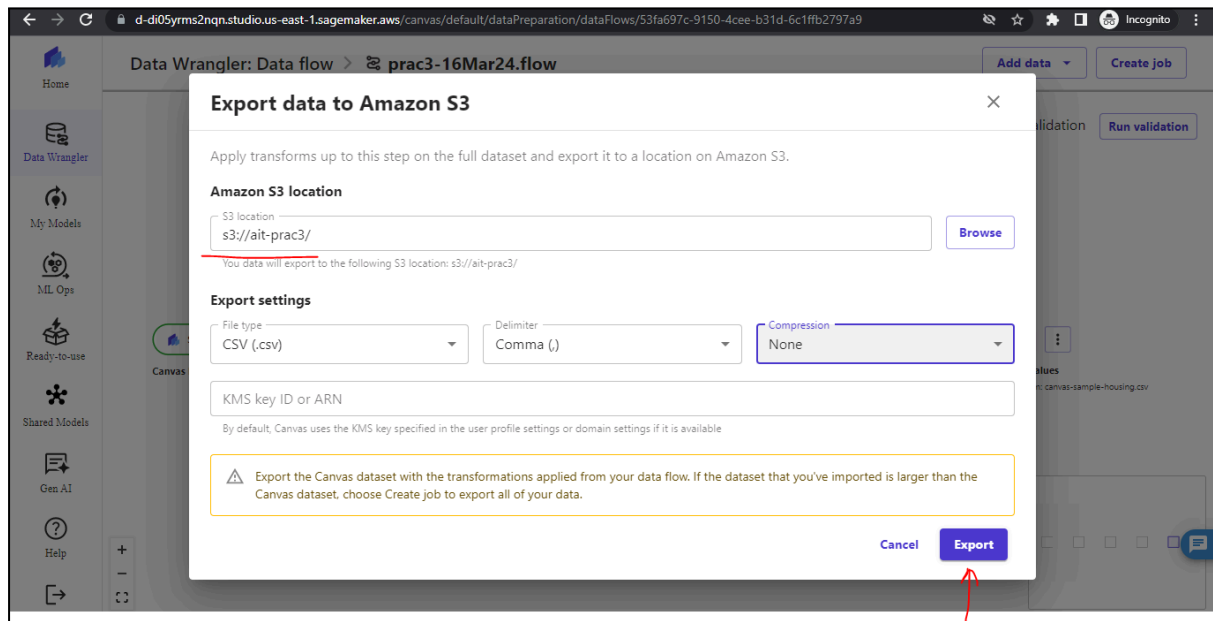
population (float)	households (float)	median_income (float)	median_house_value (float)
322	126	8.3252	452000

On the right side, the 'Edit analysis' panel is visible. It includes fields for 'Analysis type' (Scatter Plot), 'Analysis name' (Untitled), 'X axis' (median_income), and 'Y axis' (median_house_value). A red circle highlights the 'Edit analysis' button, and a red arrow points to the 'Clear' button below it.

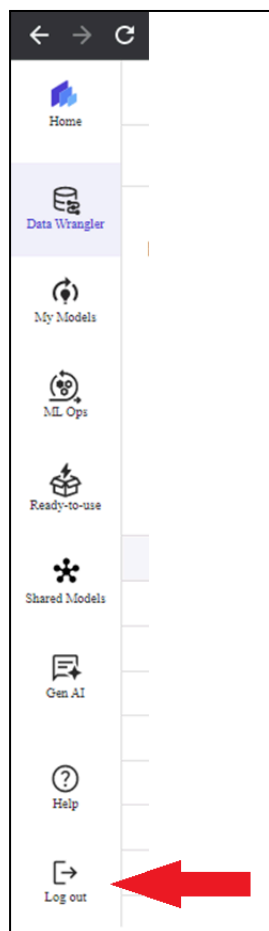
The screenshot shows the Amazon SageMaker Data Wrangler interface at 'Step 5. Scale values'. The 'Data' tab is selected, and the 'Export data' button is highlighted with a red arrow. A dropdown menu is open, showing 'Canvas dataset' and 'Amazon S3' as options. The main area displays histograms for 'housing_median_age (float)', 'total_rooms (float)', and 'total_bedrooms (float)'. Below the histograms is a table with the following data:

housing_median_age (float)	total_rooms (float)	total_bedrooms (float)	median_house_value (float)
41	0.030730014869362034	0.03238341968911917	322
21	0.25090278269489485	0.2854922279792746	2401
52	0.05151171847341217	0.04818652849740932	496
52	0.04467889258656093	0.05984455958549222	558
52	0.05717623734334065	0.07150259067357513	565

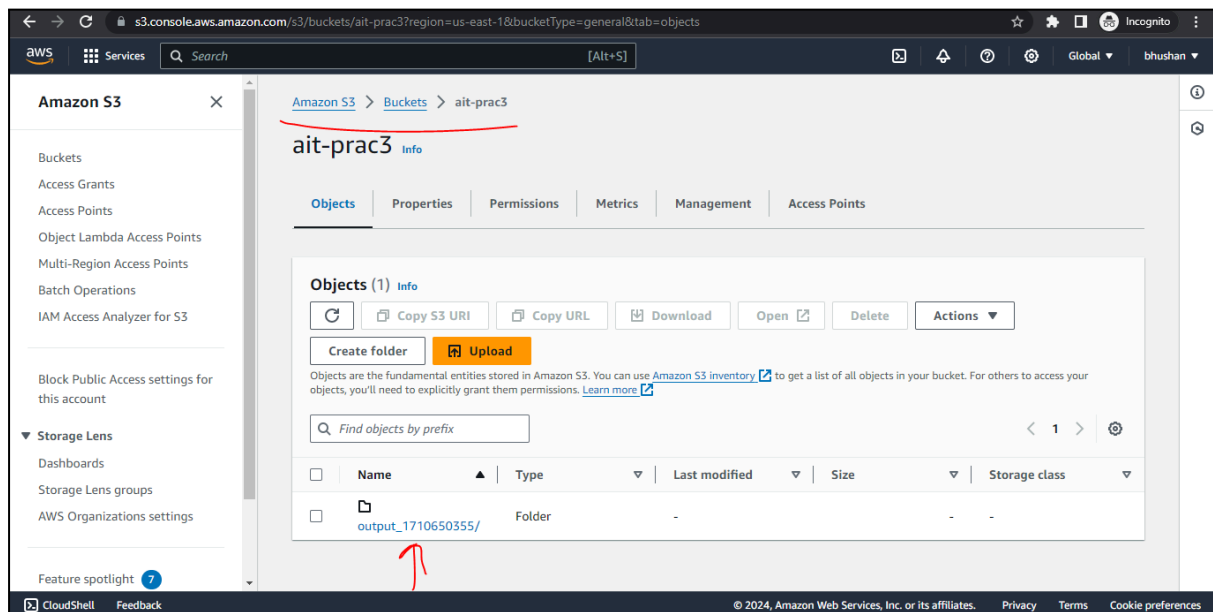
At the bottom, there is a checkbox labeled 'Show in-column visualizations for the first 2,000 rows. Visualize the full dataset, Run Data quality and insights report'.



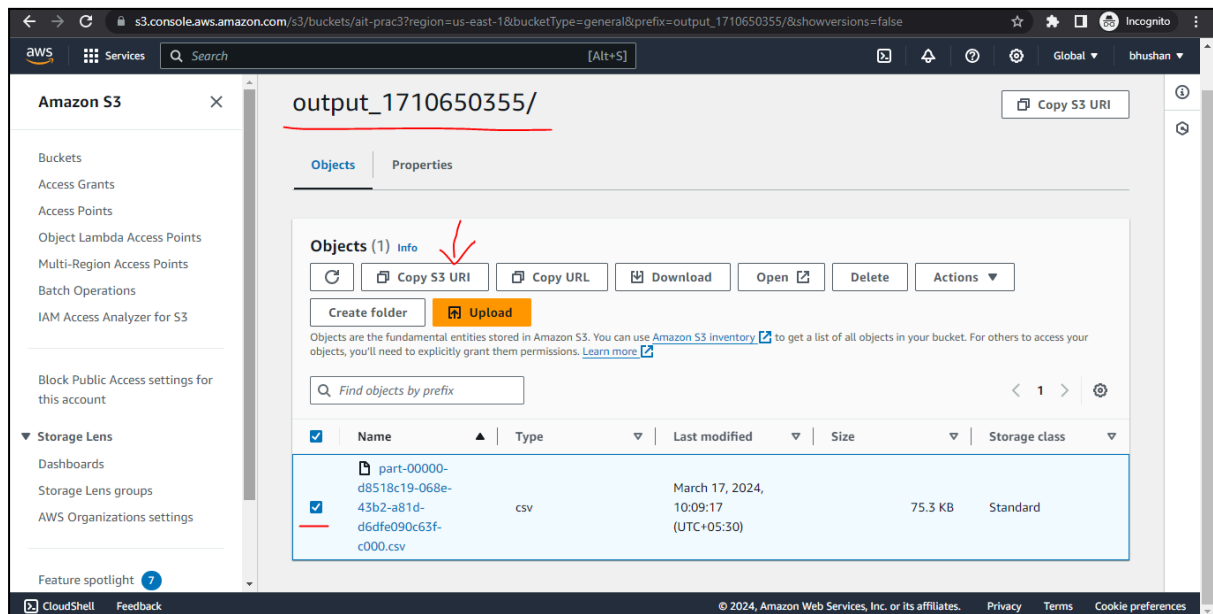
Logout of the Canvas after the data is exported



Important: Delete the SageMaker-domain-user and the SageMaker-domain
Locate the exported CSV file in the bucket

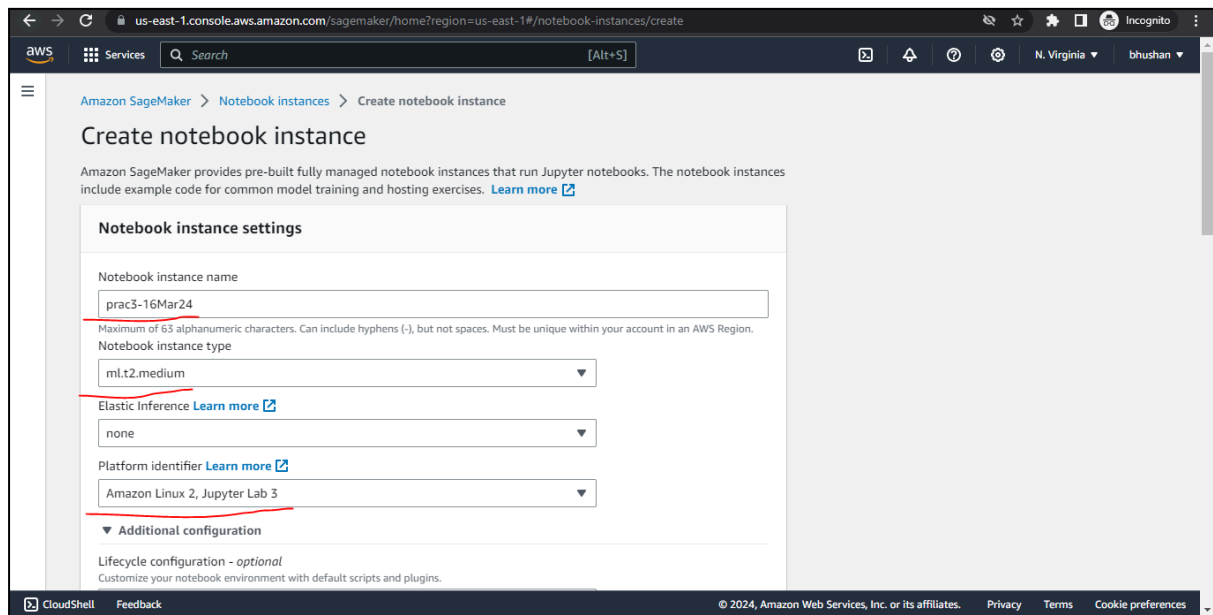
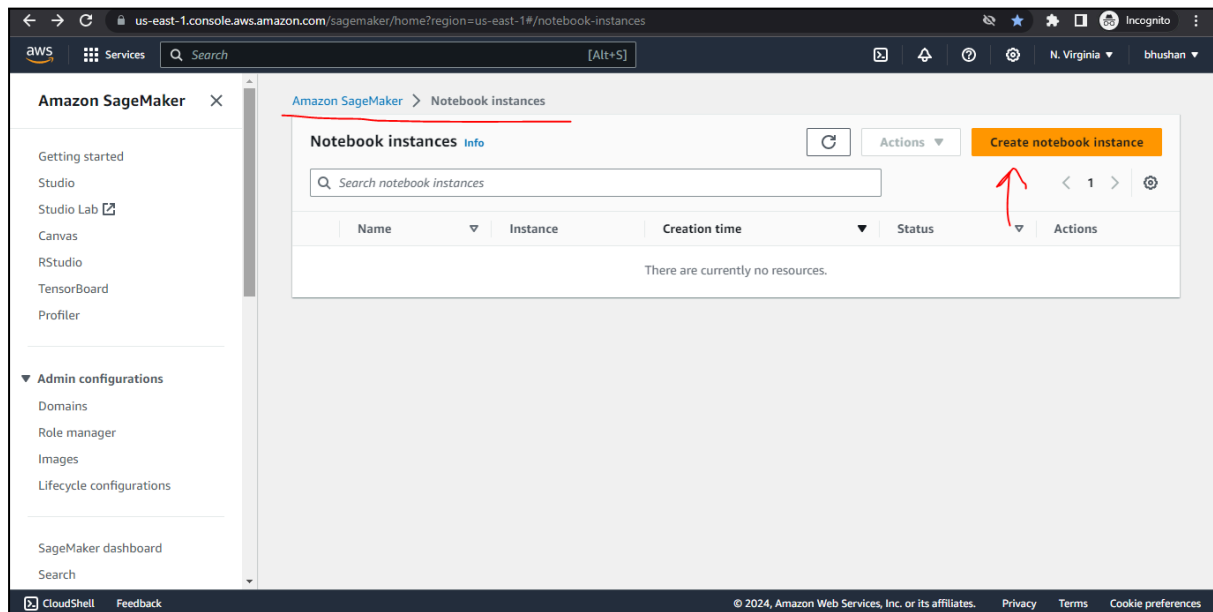


Copy the S3 URI of the csv file (to be used later in the python program)



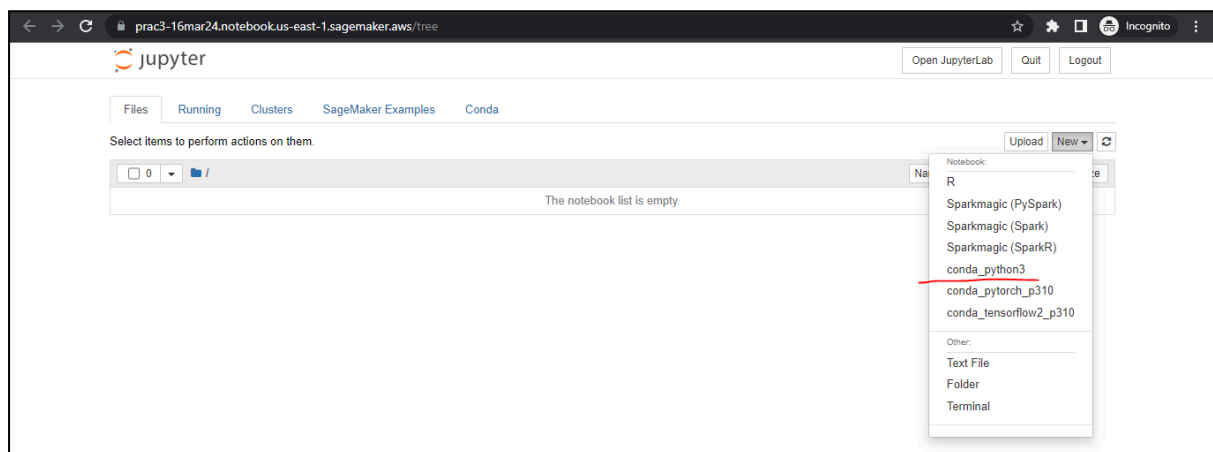
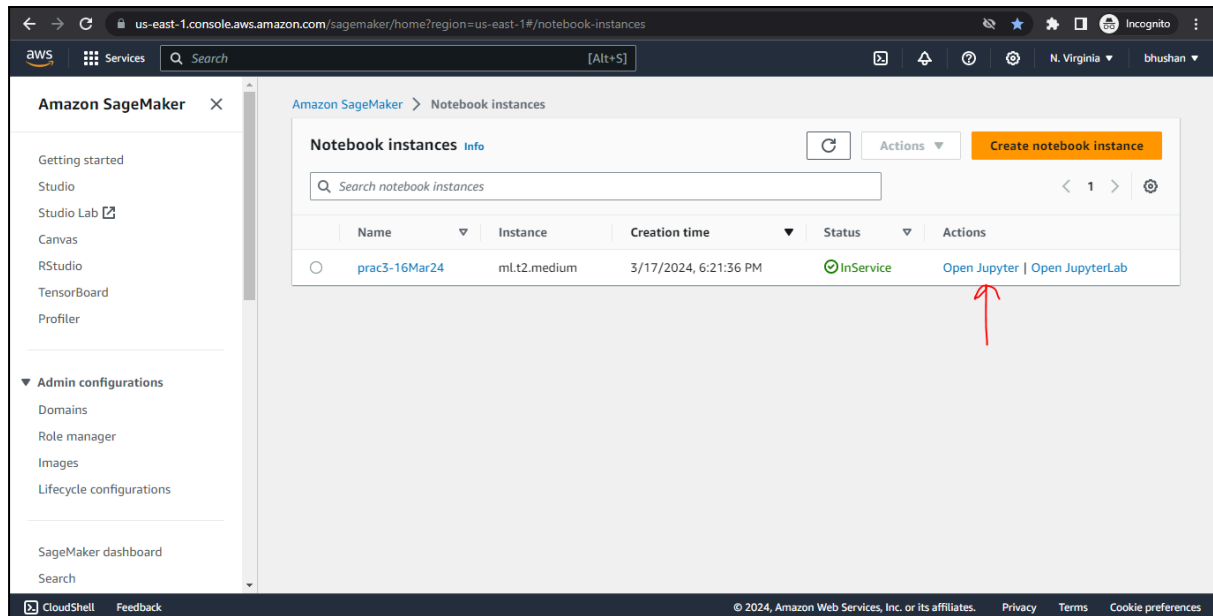
Simple Linear Regression using SageMaker

Create a Notebook



Click on “create” to create the notebook

Open in Jupyter to add the code



Add the following code

```
# Using linear regression algorithm from sklearn library
```

```
import pandas as pd
```

```
import boto3
```

```
from io import StringIO
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.linear_model import LinearRegression
```

```
from sklearn.metrics import r2_score
```

```
# Load the dataset
```

```
s3_bucket = 'ait-prac3'
```

```
s3_key =
'output_1710650355/part-00000-d8518c19-068e-43b2-a81d-d6dfe090c63f-c000.csv'
s3_client = boto3.client('s3')
response = s3_client.get_object(Bucket=s3_bucket, Key=s3_key)
data = pd.read_csv(response['Body'])
# data = pd.read_csv(response['Body'], na_values=['NA', 'N/A', 'NaN', ''])
# Split the data into features (X) and target variable (y)
X = data[['median_income']]
y = data[['median_house_value']] # Notice the double square brackets to keep it as a
DataFrame
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Initialize and train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)
# Make predictions on the test set
predictions = model.predict(X_test)
#print(predictions)
# Extract the coefficients
# slope = model.coef_[0][0]
# intercept = model.intercept_[0]
# Print the equation of the linear regression line
# print(f"Linear Regression Equation: y = {slope:.2f}x + {intercept:.2f}")
## Evaluate the model's accuracy
test_predictions = model.predict(X_test)
r_squared = r2_score(y_test, test_predictions)
print("R-squared:", r_squared)
# using SageMaker's built-in Linear Regression algorithm
# import sagemaker
# from sagemaker import get_execution_role
# from sagemaker.amazon.amazon_estimator import get_image_uri
# from sagemaker.session import s3_input
# Define your S3 bucket and prefix
# bucket = 'your-s3-bucket'
# prefix = 'your-prefix'
```

```
# Set up SageMaker session and role
# sagemaker_session = sagemaker.Session()
# role = get_execution_role()
# Specify the location of your dataset in S3
# train_data = 's3://{}/{}/{}'.format(bucket, prefix, 'canvas-sample-housing.csv')
# Create a LinearLearner estimator
# linear_container = get_image_uri(sagemaker_session.boto_region_name, 'linear-learner')
# linear = sagemaker.estimator.Estimator(linear_container,

#                                     role,

#                                     train_instance_count=1,

#                                     # train_instance_type='ml.m4.xlarge',

#                                     train_instance_type="",

#                                     output_path='s3://{}/{}/output'.format(bucket, prefix),

#                                     sagemaker_session=sagemaker_session)

# Set hyperparameters

# linear.set_hyperparameters(predictor_type='regressor',

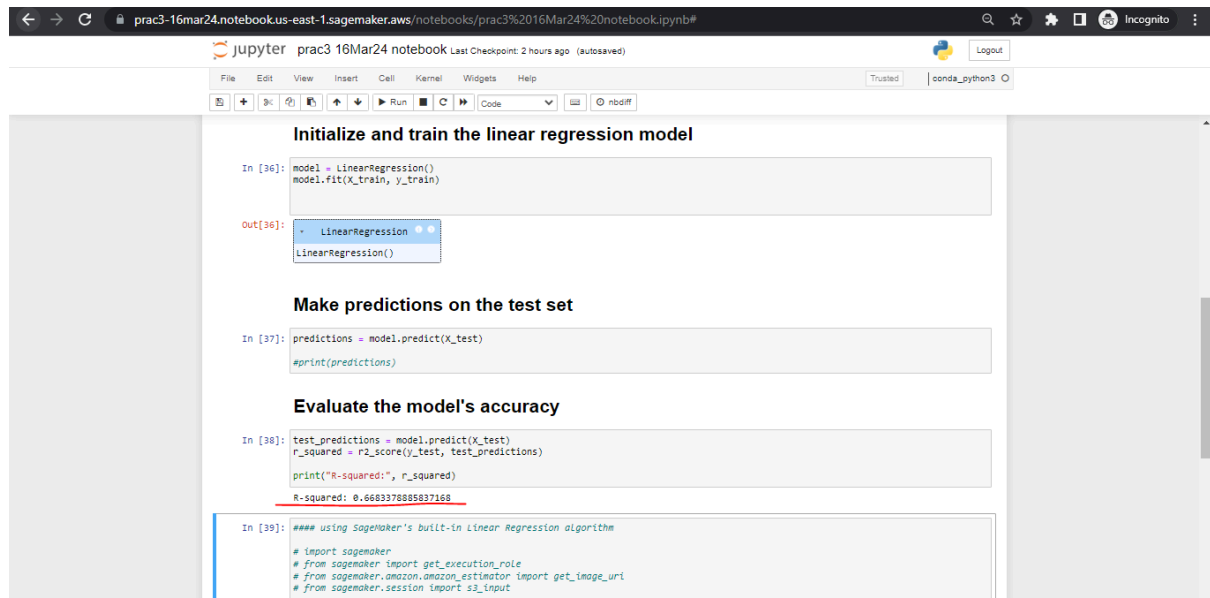
#                             mini_batch_size=50,

#                             epochs=3)

# Train the model

# linear.fit({'train': s3_input(train_data, content_type='text/csv')})
```

Output



The screenshot shows a Jupyter Notebook titled "prac3 16Mar24 notebook" with the following code cells:

```
In [36]: model = LinearRegression()
         model.fit(X_train, y_train)

Out[36]: LinearRegression()
```

Initialize and train the linear regression model

```
In [37]: predictions = model.predict(X_test)
         #print(predictions)
```

Make predictions on the test set

```
In [38]: test_predictions = model.predict(X_test)
         r_squared = r2_score(y_test, test_predictions)
         print("R-squared:", r_squared)

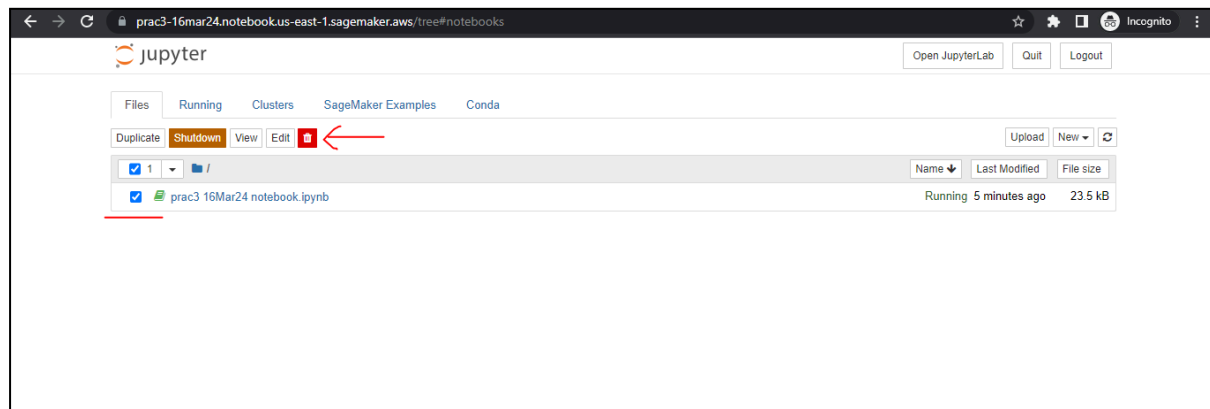
R-squared: 0.66837885837168
```

Evaluate the model's accuracy

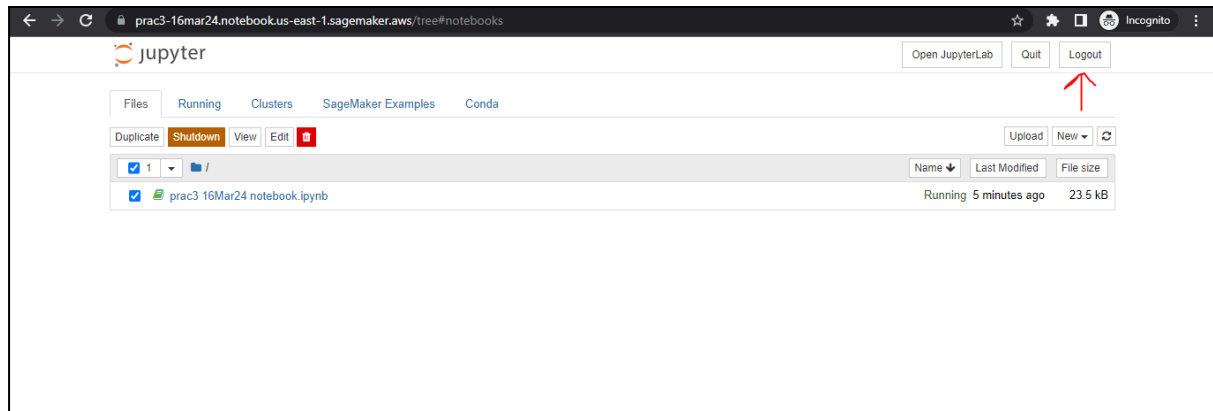
```
In [39]: ### using SageMaker's built-in Linear Regression algorithm
         # import sagemaker
         # from sagemaker import get_execution_role
         # from sagemaker.amazon.amazon_estimator import get_image_uri
         # from sagemaker.session import s3_input
```

Since the R^2 value is 0.66, we can say that the linear regression model is 66% accurate

Delete the Jupyter Notebook



Logout from Jupyter



CLEAN UP

1. Stop all the services that are running.
2. Delete all the resources that were created for this practical (including the buckets and the notebooks, etc.)

Conclusion: We saw data Preparation and EDA using AWS SageMaker DataWrangler, linear regression using SageMaker.