

STUDENT 1: 11355 | ASAD TARIQ SHEIKH

STUDENT 2: 10718 | MUZAMMIL AHMED

CLASS ID: 114195

COURSE: INTRODUCTION TO BIG DATA

INSTRUCTOR: MUHAMMAD AHMAD

CRYPTOCURRENCY PROJECT

STEP 1: TERMINAL (docker-compose)

```
asad@asad-Latitude:~/Desktop/IBDProjectAsad/minilake-main$ docker-compose up -d
postgres is up-to-date
master-node is up-to-date
worker-node1 is up-to-date
jupyter-node is up-to-date
historyserver is up-to-date
asad@asad-Latitude:~/Desktop/IBDProjectAsad/minilake-main$ █
```

STEP 2: DOCKER (All the images running)

The screenshot shows the Docker Desktop interface with the following details:

- Containers**: The main view displays a list of running containers.
- Container CPU usage**: 764.34% / 800% (8 cores available).
- Container memory usage**: 3.37GB / 7.48GB.
- Search bar**: Search for images, containers, volumes, extensions and more...
- Filter**: Only show running containers.
- Delete**: Red button to delete selected containers.
- Actions**: Columns for Name, Image, Status, CPU (%), Port(s), Last started, and Actions (⋮, ⏺, ⏻).
- Selected Container**: **minilake-main** (Running 5/5, 764.34%, 41 seconds ago).
- Other Containers**:
 - worker-node1** (Running, 248.31%, 41 seconds ago)
 - postgres** (Running, 0.06%, 41 seconds ago)
 - master-node** (Running, 390.48%, 41 seconds ago)
 - jupyter-node** (Running, 0%, 41 seconds ago)
 - historyserver** (Running, 125.49%, 41 seconds ago)
- Extensions**: Shows extensions like minilake and others.
- Add Extensions**: Button to add new extensions.
- Status Bar**: Engine running, RAM 4.60 GB, CPU 99.87%, Disk 38.28 GB avail. of 62.67 GB, Signed in.
- Bottom Right**: Version v4.25.0, 2 notifications.

Docker Desktop Update to latest

Search for images, containers, volumes, extensions and more... ⌘K

sheikh...

Containers

Images

Volumes

Dev Environments BETA

Docker Scout

Learning center

Extensions •

Add Extensions

Images Give feedback

Local Hub Artifactory EARLY ACCESS

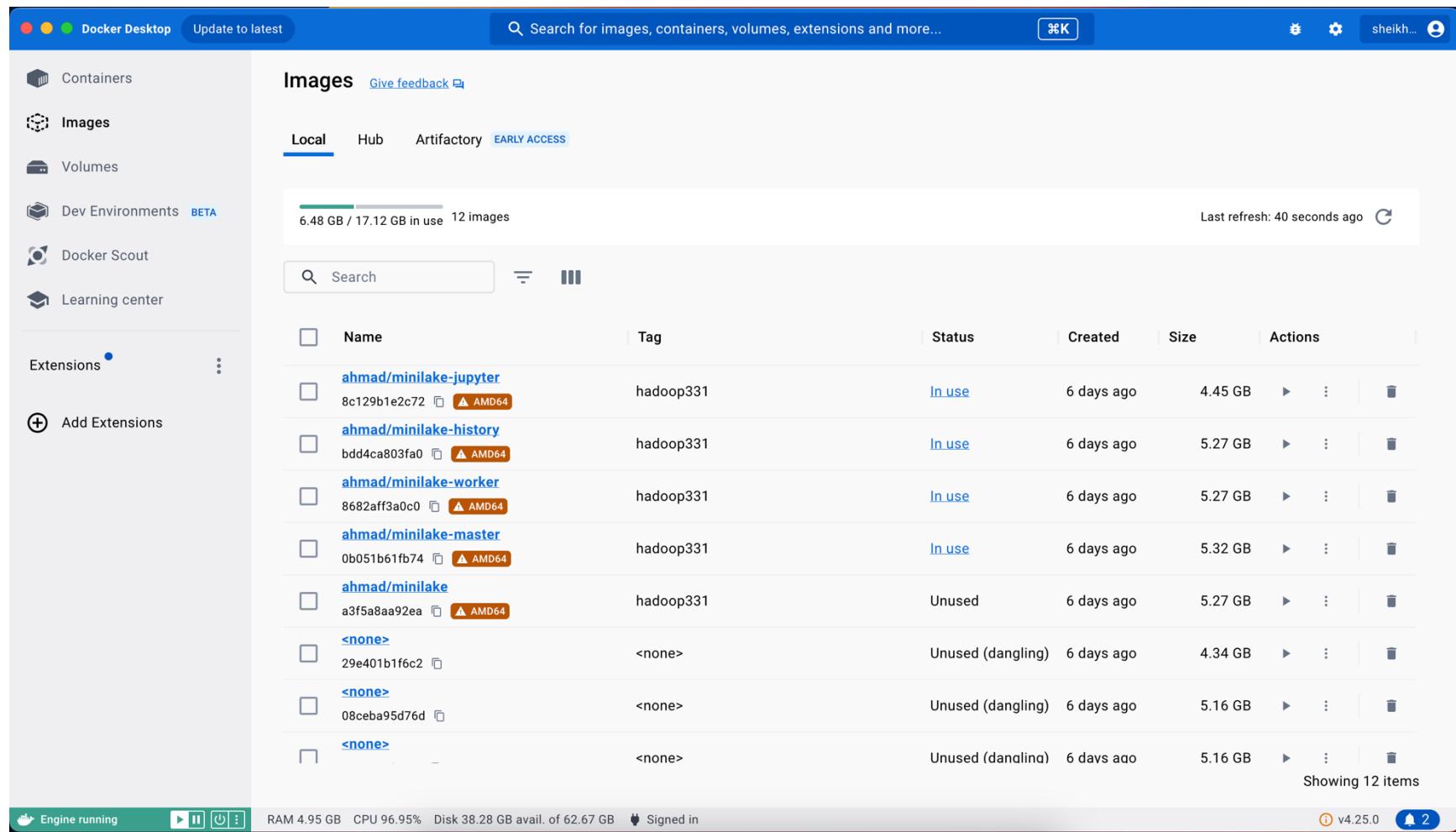
6.48 GB / 17.12 GB in use 12 images Last refresh: 40 seconds ago

Search

Name	Tag	Status	Created	Size	Actions
ahmad/minilake-jupyter 8c129b1e2c72	hadoop331	In use	6 days ago	4.45 GB	▶ ⋮ 🗑
ahmad/minilake-history bdd4ca803fa0	hadoop331	In use	6 days ago	5.27 GB	▶ ⋮ 🗑
ahmad/minilake-worker 8682aff3a0c0	hadoop331	In use	6 days ago	5.27 GB	▶ ⋮ 🗑
ahmad/minilake-master 0b051b61fb74	hadoop331	In use	6 days ago	5.32 GB	▶ ⋮ 🗑
ahmad/minilake a3f5a8aa92ea	hadoop331	Unused	6 days ago	5.27 GB	▶ ⋮ 🗑
<none> 29e401b1f6c2	<none>	Unused (dangling)	6 days ago	4.34 GB	▶ ⋮ 🗑
<none> 08ceba95d76d	<none>	Unused (dangling)	6 days ago	5.16 GB	▶ ⋮ 🗑
<none>	<none>	Unused (dangling)	6 days ago	5.16 GB	▶ ⋮ 🗑

Showing 12 items

Engine running RAM 4.95 GB CPU 96.95% Disk 38.28 GB avail. of 62.67 GB Signed in v4.25.0 2





Quit

Files Running Clusters

Select items to perform actions on them.

Upload New ▾



<input type="checkbox"/> 0	<input type="checkbox"/> /	Name	Last Modified	File size
<input type="checkbox"/>	<input type="checkbox"/> pyspark.ipynb		7 days ago	7.34 kB
<input type="checkbox"/>	<input type="checkbox"/> SparkLab(python).ipynb		a day ago	46.2 kB
<input type="checkbox"/>	<input type="checkbox"/> Untitled.ipynb		7 days ago	12.4 kB
<input type="checkbox"/>	<input type="checkbox"/> bitcoin.csv		seconds ago	206 kB
<input type="checkbox"/>	<input type="checkbox"/> usd.csv		seconds ago	206 kB

STEP 3: HIVE: CREATE TABLE `CRYPTO`

jupyter

```
(base) root@root:~# hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/hive/lib/log4j-slf4j-impl-2.17.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/hadoop/share/hadoop/common/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLogge
rBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 5ff86310-9fbf-440b-8dbf-333dd07b65ea

Logging initialized using configuration in jar:file:/opt/hive/lib/hive-common-3.1.3.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution en
gine (i.e. spark, tez) or using Hive 1.X releases.Hive Session ID = 71e20623-35f6-46e9-a5b9-8ae0b52a9208

hive> CREATE TABLE IF NOT EXISTS Crypto(
    > daate date,
    > open double,
    > high double,
    > low double,
    > cloose double,
    > volume double,
    > currency string)
    > COMMENT 'CryptoData'
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ',';
OK
Time taken: 0.895 seconds
hive> LOAD DATA INPATH '/tmp/spark/usd.csv' INTO TABLE Crypto;
Loading data to table default.crypto
OK
Time taken: 0.926 seconds
hive> █
```

STEP 4: PYSPARK: LOAD DATA INTO HDFS

jupyter SparkLab(python) Last Checkpoint: 2 hours ago (unsaved changes)

Not Trusted | Python 3 (ipykernel) O

Cryptocurrency Dataset

11355 | Asad Tariq Sheikh

10718 | Muzammil Ahmed

```
In [96]: import findspark
findspark.init()

In [97]: import pyspark
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName('jupyter-spark') \
    .enableHiveSupport() \
    .getOrCreate()
sc = spark.sparkContext

24/01/06 17:09:11 WARN SparkSession: Using an existing Spark session; only runtime SQL configurations will take effect.

In [98]: # Checking if spark is working or not
spark
```

Out[98]: `SparkSession - hive
SparkContext

Spark UI
Version
v3.3.1
Master
yarn
AppName
jupyter-spark`

```
In [99]: !pwd
/notebook
```

Loading Data into HDFS

```
In [100]: !hdfs dfs -put /notebook/usd.csv /tmp/spark/
!hadoop fs -ls /tmp/spark

put: '/tmp/spark/usd.csv': File exists
Found 22 items
-rw-r--r-- 2 root supergroup 64067991 2024-01-06 14:34 /tmp/spark/CAvideos.csv
```

jupyter SparkLab(python) Last Checkpoint: 2 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Help

Not Trusted Python 3 (ipykernel)

In [100]: `!hdfs dfs -put /notebook/usd.csv /tmp/spark/
!hadoop fs -ls /tmp/spark`

```
put: `/tmp/spark/usd.csv': File exists  
Found 22 items  
-rw-r--r-- 2 root supergroup 64067991 2024-01-06 14:34 /tmp/spark/CAvideos.csv  
-rw-r--r-- 2 root supergroup 63040138 2024-01-06 14:34 /tmp/spark/DEvideos.csv  
-rw-r--r-- 2 root supergroup 51424788 2024-01-06 14:34 /tmp/spark/FRvideos.csv  
-rw-r--r-- 2 root supergroup 53213441 2024-01-06 14:34 /tmp/spark/GBvideos.csv  
-rw-r--r-- 2 root supergroup 59600439 2024-01-06 14:34 /tmp/spark/INvideos.csv  
-rw-r--r-- 2 root supergroup 28740747 2024-01-06 14:34 /tmp/spark/JPvideos.csv  
-rw-r--r-- 2 root supergroup 34835868 2024-01-06 14:34 /tmp/spark/KRvideos.csv  
-rw-r--r-- 2 root supergroup 45191541 2024-01-06 14:34 /tmp/spark/MXvideos.csv  
-rw-r--r-- 2 root supergroup 76268286 2024-01-06 14:34 /tmp/spark/RUvideos.csv  
-rw-r--r-- 2 root supergroup 62756152 2024-01-06 14:34 /tmp/spark/USvideos.csv  
drwxr-xr-x - root supergroup 0 2024-01-06 14:50 /tmp/spark/df_1.parquet  
drwxr-xr-x - root supergroup 0 2024-01-06 14:50 /tmp/spark/df_10.parquet  
drwxr-xr-x - root supergroup 0 2024-01-06 14:50 /tmp/spark/df_2.parquet  
drwxr-xr-x - root supergroup 0 2024-01-06 14:50 /tmp/spark/df_3.parquet  
drwxr-xr-x - root supergroup 0 2024-01-06 14:50 /tmp/spark/df_4.parquet  
drwxr-xr-x - root supergroup 0 2024-01-06 14:50 /tmp/spark/df_5.parquet  
drwxr-xr-x - root supergroup 0 2024-01-06 14:50 /tmp/spark/df_6.parquet  
drwxr-xr-x - root supergroup 0 2024-01-06 14:50 /tmp/spark/df_7.parquet  
drwxr-xr-x - root supergroup 0 2024-01-06 14:50 /tmp/spark/df_8.parquet  
drwxr-xr-x - root supergroup 0 2024-01-06 14:50 /tmp/spark/df_9.parquet  
-rw-r--r-- 2 root supergroup 1185018 2024-01-06 12:43 /tmp/spark/indexProcessed.csv  
-rw-r--r-- 2 root supergroup 205588 2024-01-06 13:13 /tmp/spark/usd.csv
```

In [101]: `rrdd = sc.textFile('hdfs://tmp/spark/usd.csv')`

In [102]: `rrdd.foreach(lambda f: print(f))`

In [103]: `rrdd.collect()`

```
Out[103]: ['Date,Open,High,Low,Close,Volume,Currency',  
'18/07/2010,0.0,0.1,0.1,0.1,175,USD',  
'19/07/2010,0.1,0.1,0.1,0.1,574,USD',  
'20/07/2010,0.1,0.1,0.1,0.1,262,USD',  
'21/07/2010,0.1,0.1,0.1,0.1,575,USD',  
'22/07/2010,0.1,0.1,0.1,0.1,2160,USD',  
'23/07/2010,0.1,0.1,0.1,0.1,2493,USD',  
'24/07/2010,0.1,0.1,0.1,0.1,496,USD',  
'25/07/2010,0.1,0.1,0.1,0.1,1551,USD',  
'26/07/2010,0.1,0.1,0.1,0.1,877,USD',  
'27/07/2010,0.1,0.1,0.1,0.1,3374,USD',  
'28/07/2010,0.1,0.1,0.1,0.1,4390,USD',  
'29/07/2010,0.1,0.1,0.1,0.1,0.1,8058,USD',  
'30/07/2010,0.1,0.1,0.1,0.1,3021,USD',  
'31/07/2010,0.1,0.1,0.1,0.1,0.1,4022,USD',  
'01/08/2010,0.1,0.1,0.1,0.1,2681,USD',  
'02/08/2010,0.1,0.1,0.1,0.1,3599,USD',  
'03/08/2010,0.1,0.1,0.1,0.1,9821,USD',
```

STEP 5: PRINT DATA FROM HIVE & FIND MEAN, MEDIAN, MODE, & CORRELATION

jupyter SparkLab(python) Last Checkpoint: 2 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Help

Not Trusted Python 3 (ipykernel) ○

Printing Data from Hive

```
In [104]: df = spark.sql("SELECT * FROM Crypto")
In [105]: df.show()
```

date	open	high	low	close	volume	currency
null	null	null	null	null	null	Currency
null	0.01	0.1	0.1	0.1	75.0	USD
null	0.1	0.1	0.1	0.1	574.0	USD
null	0.1	0.1	0.1	0.1	262.0	USD
null	0.1	0.1	0.1	0.1	575.0	USD
null	0.1	0.1	0.1	0.1	2160.0	USD
null	0.1	0.1	0.1	0.1	2403.0	USD
null	0.1	0.1	0.1	0.1	496.0	USD
null	0.1	0.1	0.1	0.1	1551.0	USD
null	0.1	0.1	0.1	0.1	877.0	USD
null	0.1	0.1	0.1	0.1	3374.0	USD
null	0.1	0.1	0.1	0.1	14390.0	USD
null	0.1	0.1	0.1	0.1	8058.0	USD
null	0.1	0.1	0.1	0.1	3021.0	USD
null	0.1	0.1	0.1	0.1	4022.0	USD
null	0.1	0.1	0.1	0.1	2601.0	USD
null	0.1	0.1	0.1	0.1	3599.0	USD
null	0.1	0.1	0.1	0.1	9821.0	USD
null	0.1	0.1	0.1	0.1	3494.0	USD
null	0.1	0.1	0.1	0.1	5034.0	USD

only showing top 20 rows

Mean, Median, Mode

```
In [106]: import pandas as pd
# Load the CSV file into a DataFrame
file_path = 'usd.csv'
df = pd.read_csv(file_path, encoding='ascii')

# Calculate mean, median, and mode for all numeric columns
summary_stats = pd.DataFrame(index=['mean', 'median', 'mode'])
for column in df.select_dtypes(include=['float64', 'int64']).columns:
    summary_stats[column] = [df[column].mean(), df[column].median(), df[column].mode()[0]]

# Output the summary statistics
print(summary_stats)
```

	Open	High	Low	Close	Volume
mean	8302.221855	8531.97138	8043.473869	8306.969457	1.395788e+07
median	683.600000	706.50000	657.550000	685.000000	6.762050e+04
mode	0.100000	0.10000	0.100000	0.100000	0.000000e+00

STEP 6: MACHINE LEARNING MODEL & PREDICTION

jupyter SparkLab(python) Last Checkpoint: 2 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Help

Not Trusted Python 3 (ipykernel) ○

Correlation

```
In [107]: # Calculate the correlation matrix for the relevant columns
relevant_columns = ['Open', 'High', 'Low', 'Close', 'Volume']
correlation_matrix = df[relevant_columns].corr()

# Output the correlation matrix
print(correlation_matrix)
```

	Open	High	Low	Close	Volume
Open	1.000000	0.999589	0.999221	0.998996	0.182761
High	0.999589	1.000000	0.999158	0.999556	0.182589
Low	0.999221	0.999158	1.000000	0.999468	0.183437
Close	0.998996	0.999556	0.999468	1.000000	0.182242
Volume	0.182761	0.182589	0.183437	0.182242	1.000000

Using PySpark Linear Regression

```
In [108]: from pyspark.sql import SparkSession
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.regression import LinearRegression
from pyspark.ml.evaluation import RegressionEvaluator
import urllib.request

In [109]: spark = SparkSession.builder.appName("LinearRegressionExample").getOrCreate()
24/01/06 17:09:27 WARN SparkSession: Using an existing Spark session; only runtime SQL configurations will take effect.

In [110]: url = "https://raw.githubusercontent.com/asadsheikh1/MachineLearningPredictions/main/usd.csv"
local_file_path = "/usd.csv"
urllib.request.urlretrieve(url, local_file_path)

Out[110]: ('/usd.csv', <http.client.HTTPMessage at 0x7f22f7d77340>

In [111]: df = pd.read_csv(local_file_path)

In [112]: !hdfs dfs -ls hdfs://master/
Found 6 items
-rw-r--r-- 2 root supergroup 1185010 2024-01-06 13:38 hdfs://master/indexProcessed.csv
drwxr-xr-x - root supergroup 0 2024-01-06 08:22 hdfs://master/spark
drwxrwx--- - root supergroup 0 2024-01-06 12:55 hdfs://master/tmp
-rw-r--r-- 2 root supergroup 205580 2024-01-06 13:26 hdfs://master/usd.csv
drwxr-xr-x - root supergroup 0 2024-01-06 08:29 hdfs://master/user
drwxr-xr-x - root supergroup 0 2024-01-06 08:22 hdfs://master/var

In [113]: !hdfs dfs -put /notebook/usd.csv /
put: '/usd.csv': File exists
```

jupyter SparkLab(python) Last Checkpoint: 2 hours ago (autosaved)

File Edit View Insert Cell Kernel Help

Not Trusted Python 3 (ipykernel)

```
In [113]: !hdfs dfs -put /notebook/usd.csv /
put: `/usd.csv': File exists
```

```
In [114]: data = spark.read.csv(local_file_path, header=True, inferSchema=True)
```

```
In [115]: data
Out[115]: DataFrame[Date: string, Open: double, High: double, Low: double, Close: double, Volume: bigint, Currency: string]
```

```
In [116]: data.printSchema()
data.show()
```

```
root
 |-- Date: string (nullable = true)
 |-- Open: double (nullable = true)
 |-- High: double (nullable = true)
 |-- Low: double (nullable = true)
 |-- Close: double (nullable = true)
 |-- Volume: long (nullable = true)
 |-- Currency: string (nullable = true)

+-----+---+---+---+---+---+
|    Date|Open|High|Low|Close|Volume|Currency|
+-----+---+---+---+---+---+
|18/07/2010| 0.0| 0.1|0.1| 0.1|    75|   USD|
|19/07/2010| 0.1| 0.1|0.1| 0.1|   574|   USD|
|20/07/2010| 0.1| 0.1|0.1| 0.1|   262|   USD|
|21/07/2010| 0.1| 0.1|0.1| 0.1|   575|   USD|
|22/07/2010| 0.1| 0.1|0.1| 0.1|  2160|   USD|
|23/07/2010| 0.1| 0.1|0.1| 0.1|  2403|   USD|
|24/07/2010| 0.1| 0.1|0.1| 0.1|   496|   USD|
|25/07/2010| 0.1| 0.1|0.1| 0.1|  1551|   USD|
|26/07/2010| 0.1| 0.1|0.1| 0.1|   877|   USD|
|27/07/2010| 0.1| 0.1|0.1| 0.1|  3374|   USD|
|28/07/2010| 0.1| 0.1|0.1| 0.1|  4390|   USD|
|29/07/2010| 0.1| 0.1|0.1| 0.1|  8058|   USD|
|30/07/2010| 0.1| 0.1|0.1| 0.1|  3021|   USD|
|31/07/2010| 0.1| 0.1|0.1| 0.1|  4022|   USD|
|01/08/2010| 0.1| 0.1|0.1| 0.1|  2601|   USD|
|02/08/2010| 0.1| 0.1|0.1| 0.1|  3599|   USD|
|03/08/2010| 0.1| 0.1|0.1| 0.1|  9821|   USD|
|04/08/2010| 0.1| 0.1|0.1| 0.1|  3494|   USD|
|05/08/2010| 0.1| 0.1|0.1| 0.1|  5034|   USD|
|06/08/2010| 0.1| 0.1|0.1| 0.1|  1395|   USD|
+-----+---+---+---+---+---+
```

only showing top 20 rows

```
In [117]: assembler = VectorAssembler(inputCols=['Open', 'High', 'Low', 'Volume'], outputCol='features')
```

jupyter SparkLab(python) Last Checkpoint: 2 hours ago (autosaved)

File Edit View Insert Cell Kernel Help

Not Trusted Python 3 (ipykernel)

```
In [117]: assembler = VectorAssembler(inputCols=['Open', 'High', 'Low', 'Volume'], outputCol='features')
data = assembler.transform(data)

In [118]: data = data.withColumnRenamed('Close', 'label')

In [119]: train_data, test_data = data.randomSplit([0.8, 0.2], seed=1234)

In [120]: lr = LinearRegression(featuresCol='features', labelCol='label')

In [121]: lr_model = lr.fit(train_data)
24/01/06 17:09:35 WARN Instrumentation: [d6ac34b7] regParam is zero, which might cause numerical instability and overfitting.

In [122]: predictions = lr_model.transform(test_data)

In [123]: evaluator = RegressionEvaluator(labelCol='label', predictionCol='prediction', metricName='rmse')
rmse = evaluator.evaluate(predictions)
evaluator = RegressionEvaluator(labelCol='label', predictionCol='prediction', metricName='rmse')
rmse = evaluator.evaluate(predictions)

In [124]: print("Mean Absolute Error: ", metrics.mean_absolute_error(Y_test, y_pred))
print("Mean Squared Error: ", metrics.mean_squared_error(Y_test, y_pred))
print("Root Mean Squared Error: ", np.sqrt(metrics.mean_squared_error(Y_test, y_pred)))
print("Coefficient of Determination (R2 Score): ", metrics.r2_score(Y_test, y_pred))
Mean Absolute Error: 0.0008327265398649373
Mean Squared Error: 2.8538136120404585e-06
Root Mean Squared Error: 0.001689323418425394
Coefficient of Determination (R2 Score): 0.9999378230963039



## Using Linear Regression


In [125]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LinearRegression
from sklearn import metrics
import warnings
from sklearn import metrics
warnings.filterwarnings("ignore")

In [126]: data = pd.read_csv("https://raw.githubusercontent.com/asadsheikh1/MachineLearningPredictions/main/usd.csv")

In [52]: data
```

jupyter SparkLab(python) Last Checkpoint: 2 hours ago (autosaved)

File Edit View Insert Cell Kernel Help

Not Trusted Python 3 (ipykernel) O

warnings.filterwarnings("ignore")

In [126]: data = pd.read_csv("https://raw.githubusercontent.com/asadsheikh1/MachineLearningPredictions/main/usd.csv")

In [52]: data

Out[52]:

	Date	Open	High	Low	Close	Volume	Currency
0	18/07/2010	0.0	0.1	0.1	0.1	75	USD
1	19/07/2010	0.1	0.1	0.1	0.1	574	USD
2	20/07/2010	0.1	0.1	0.1	0.1	262	USD
3	21/07/2010	0.1	0.1	0.1	0.1	575	USD
4	22/07/2010	0.1	0.1	0.1	0.1	2160	USD
...
4415	19/08/2022	23201.6	23202.3	20807.8	20831.3	339472	USD
4416	20/08/2022	20830.7	21357.4	20784.6	21138.9	206943	USD
4417	21/08/2022	21138.9	21692.4	21077.4	21517.2	177522	USD
4418	22/08/2022	21516.8	21517.4	20912.1	21416.3	251833	USD
4419	23/08/2022	21416.5	21458.2	21271.2	21309.0	251696	USD

4420 rows × 7 columns

In [53]: data.isnull().sum()

Out[53]:

	Date	Open	High	Low	Close	Volume	Currency
	0	0	0	0	0	0	0

In [54]: data.describe()

Out[54]:

	Open	High	Low	Close	Volume
count	4420.000000	4420.000000	4420.000000	4420.000000	4.420000e+03
mean	8302.221855	8531.971380	8043.473869	8306.969457	1.395788e+07
std	14598.398937	14992.605971	14146.499234	14599.046649	1.645973e+08
min	0.000000	0.100000	0.000000	0.100000	0.000000e+00
25%	122.500000	129.975000	118.300000	122.800000	3.013600e+04
50%	683.600000	706.500000	657.550000	685.000000	6.762050e+04
75%	8877.250000	9122.700000	8666.900000	8886.075000	1.769402e+05
max	67528.700000	68990.600000	66334.900000	67527.900000	4.468697e+09

jupyter SparkLab(python) Last Checkpoint: 2 hours ago (autosaved)

File Edit View Insert Cell Kernel Help

Not Trusted Python 3 (ipykernel)

In [54]: `data.describe()`

Out[54]:

	Open	High	Low	Close	Volume
count	4420.000000	4420.000000	4420.000000	4420.000000	4.420000e+03
mean	8302.221855	8531.971380	8043.473869	8305.969457	1.395788e+07
std	14598.398937	14992.605971	14146.499234	14599.046649	1.645973e+08
min	0.000000	0.100000	0.000000	0.100000	0.000000e+00
25%	122.500000	129.975000	118.300000	122.800000	3.013600e+04
50%	683.600000	706.500000	657.550000	685.000000	6.762050e+04
75%	8877.250000	9122.700000	8666.900000	8886.075000	1.769402e+05
max	67528.700000	68990.600000	66334.900000	67527.900000	4.468697e+09

In [55]: `data.nunique()`

Out[55]:

	Date	Open	High	Low	Close	Volume	Currency	dtype
Count	4420	3462	3453	3463	3469	4369	1	int64

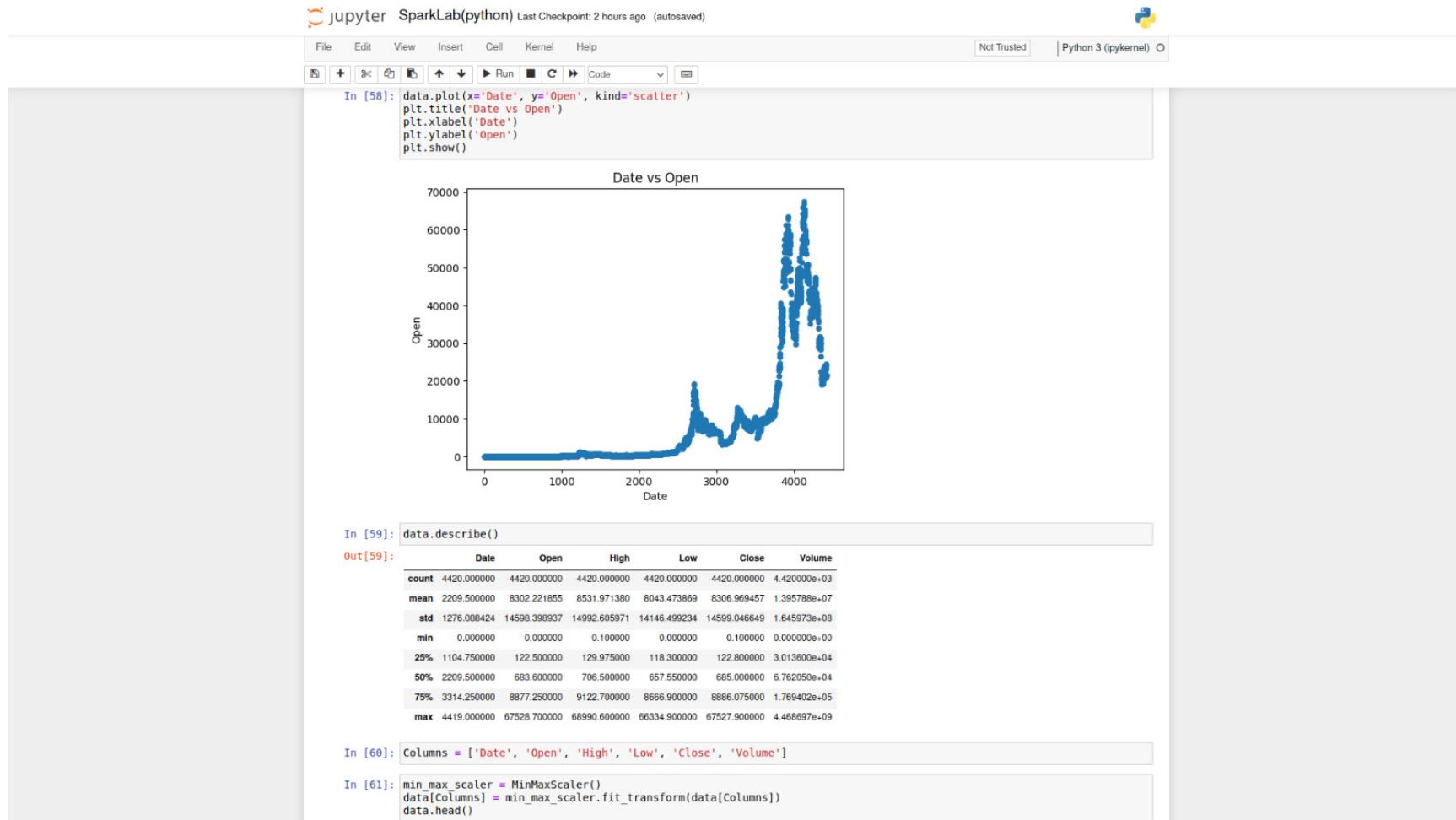
In [56]: `data.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4420 entries, 0 to 4419
Data columns (total 7 columns):
 #   Column   Non-Null Count  Dtype  
 --- 
 0   Date     4420 non-null   object 
 1   Open     4420 non-null   float64
 2   High    4420 non-null   float64
 3   Low     4420 non-null   float64
 4   Close   4420 non-null   float64
 5   Volume  4420 non-null   int64  
 6   Currency 4420 non-null   object  
dtypes: float64(4), int64(1), object(2)
memory usage: 241.8+ KB
```

In [57]: `data['Date'] = pd.to_datetime(data['Date'])
data['Date'] = (data['Date'] - data['Date'].min()).dt.days`

In [58]: `data.plot(x='Date', y='Open', kind='scatter')
plt.title('Date vs Open')
plt.xlabel('Date')
plt.ylabel('Open')
plt.show()`

Date vs Open



jupyter SparkLab(python) Last Checkpoint: 2 hours ago (autosaved)

File Edit View Insert Cell Kernel Help

Not Trusted Python 3 (ipykernel)

In [60]: Columns = ['Date', 'Open', 'High', 'Low', 'Close', 'Volume']

In [61]: min_max_scaler = MinMaxScaler()
data[Columns] = min_max_scaler.fit_transform(data[Columns])
data.head()

Out[61]:

	Date	Open	High	Low	Close	Volume	Currency
0	0.000000	0.000000	0.0	0.000002	0.0	1.678342e-08	USD
1	0.000226	0.000001	0.0	0.000002	0.0	1.284491e-07	USD
2	0.000453	0.000001	0.0	0.000002	0.0	5.863007e-08	USD
3	0.000679	0.000001	0.0	0.000002	0.0	1.286729e-07	USD
4	0.000905	0.000001	0.0	0.000002	0.0	4.833624e-07	USD

In [62]: data['Currency'] = (data['Currency'] == 'USD').astype(int)

In [63]: X = data.drop(['Open'], axis=1)
Y = data['Open']

In [64]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=30)

Linear Regression

In [65]: model = LinearRegression()
model.fit(X_train, Y_train)

Out[65]:

```
+--> LinearRegression  
|  
| LinearRegression()  
|  
+--> LinearRegression
```

In [66]: X.size

Out[66]: 26520

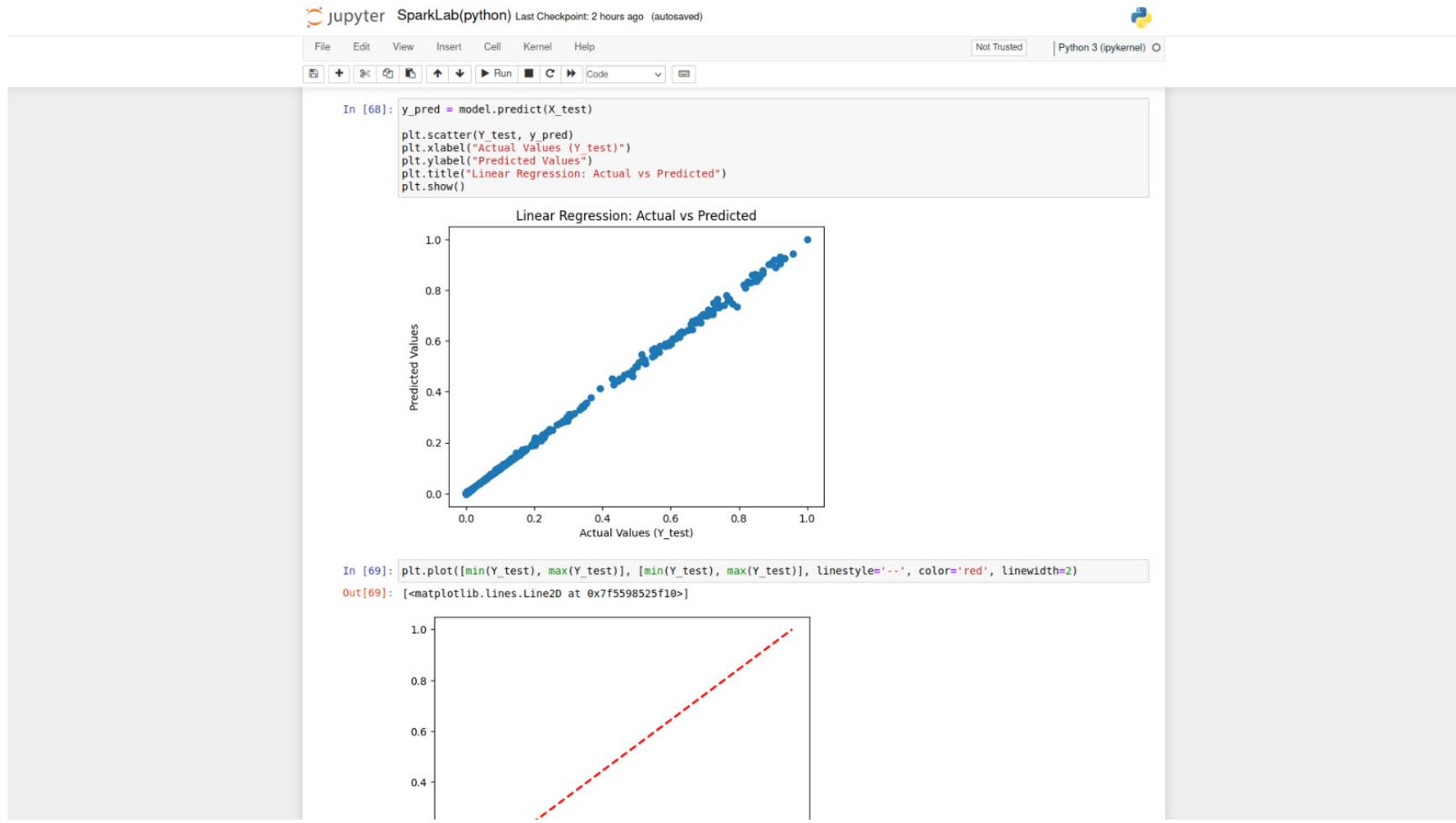
In [67]: Y.size

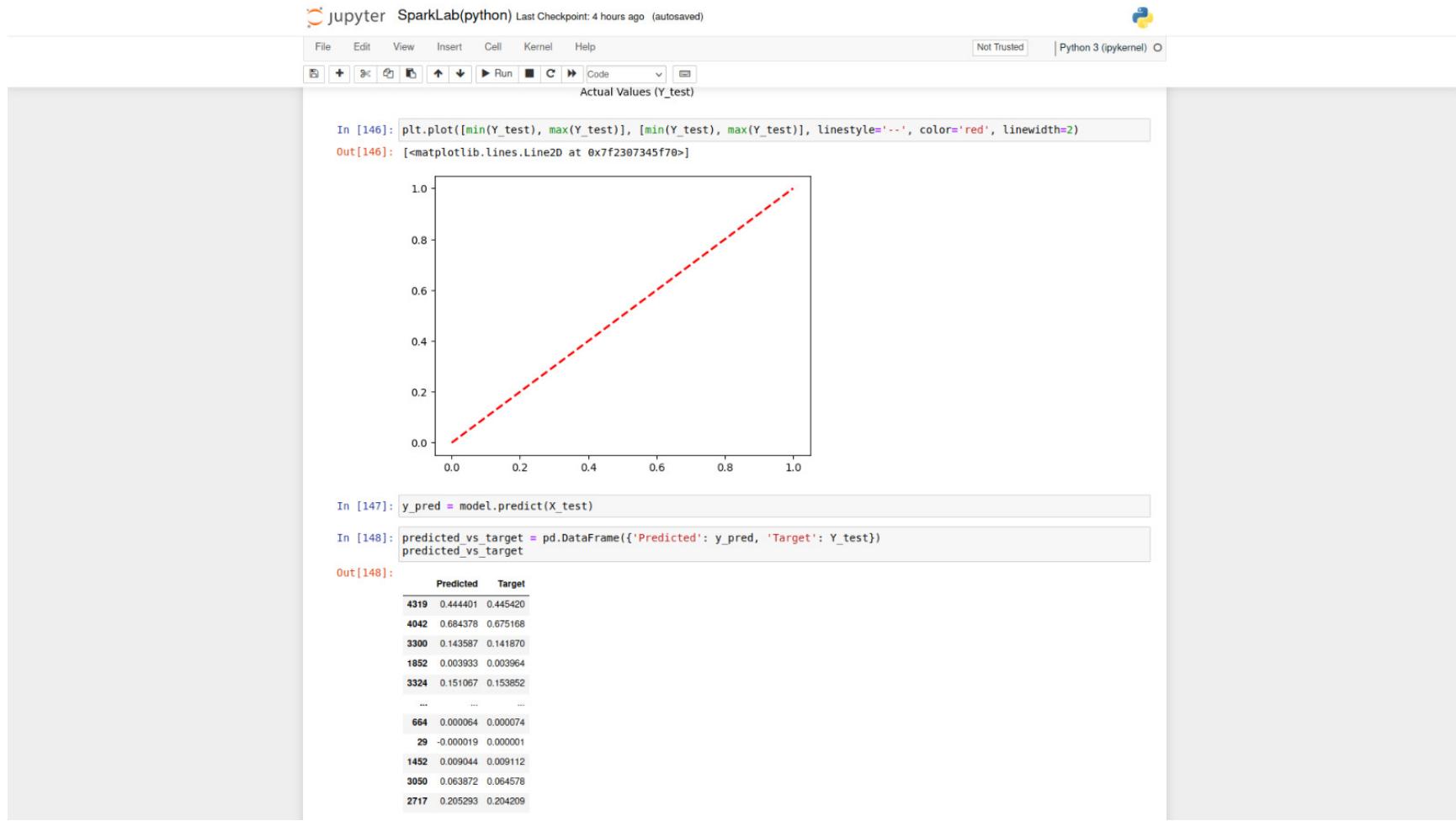
Out[67]: 4420

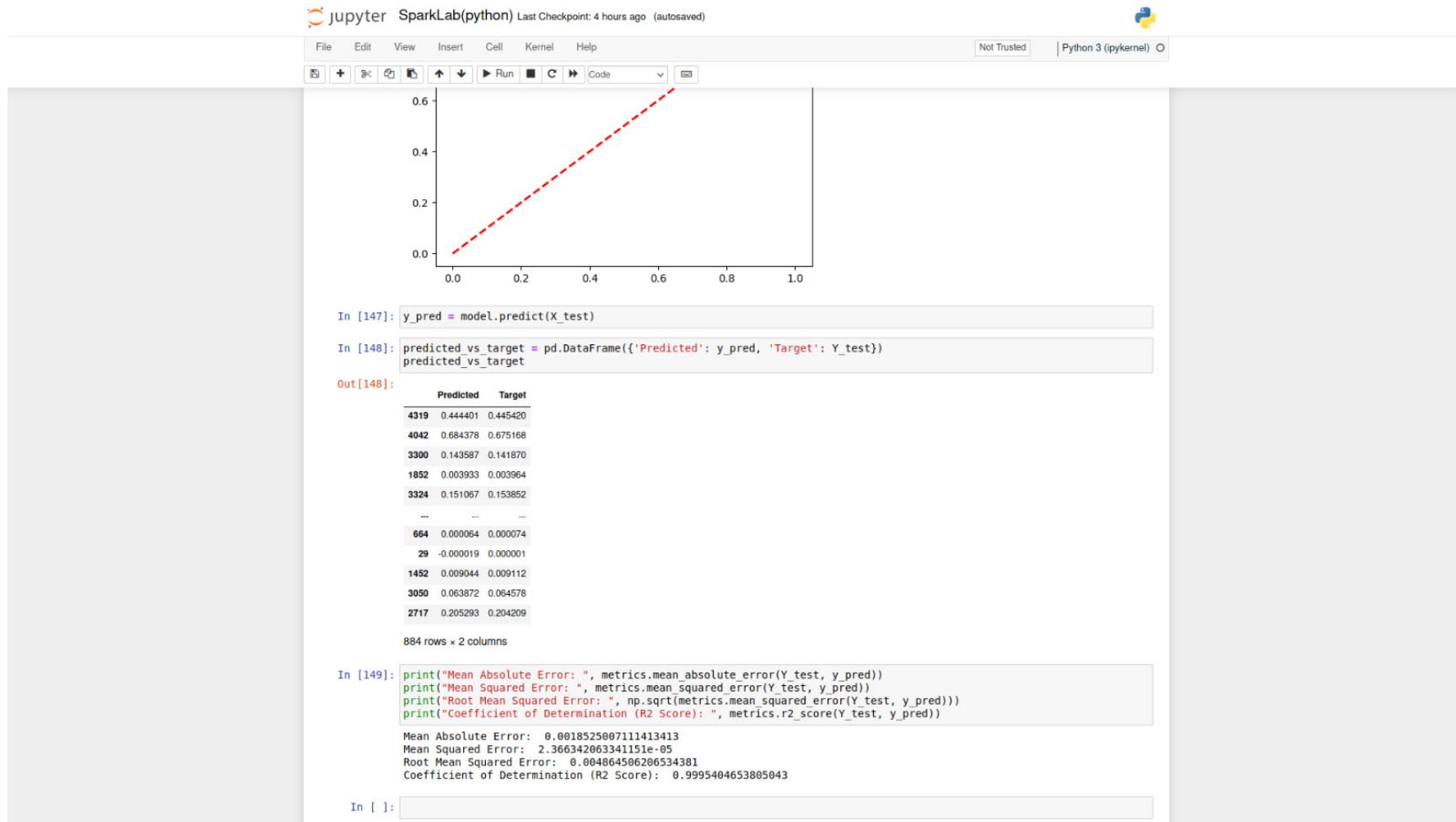
In [68]: y_pred = model.predict(X_test)

plt.scatter(Y_test, y_pred)
plt.xlabel("Actual Values (Y_test)")
plt.ylabel("Predicted Values")
plt.title("Linear Regression: Actual vs Predicted")
plt.show()

Linear Regression: Actual vs Predicted







STEP 7: DASHBOARDING THROUGH FLASK

The screenshot shows a dark-themed code editor interface with a sidebar containing icons for file operations like copy, paste, and search. The main area displays a Python script named `main.py`:

```
main.py
1  from flask import Flask, render_template
2  import pandas as pd
3
4  app = Flask(__name__)
5
6  data = pd.read_csv('data.csv')
7
8  @app.route('/')
9  def dashboard():
10     return render_template('dashboard.html', data=data)
11
12 if __name__ == '__main__':
13     app.run(debug=True)
14
```

Below the code editor is a terminal window showing the output of running the script:

```
PROBLEMS OUTPUT TERMINAL PORTS COMMENTS DEBUG CONSOLE Python Debug Console + ×
conda activate base
(base) asadsheikh@Asads-MacBook-Pro Cryptocurrency Dataset % conda activate base
>cd/python/debugpy/adapt...
* Serving Flask app 'main'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 826-019-290
127.0.0.1 - - [06/Jan/2024 23:19:31] "GET / HTTP/1.1" 200 -
```

The terminal also shows a status bar at the bottom with information like line count, character count, encoding, and code analysis tools.

DATA.CSV FILE

```
data.csv x
data.csv
1 date,Open,High,Low,Close,Volume,Currency
2 18/07/2010,0,0.1,0.1,0.1,75,USD
3 19/07/2010,0.1,0.1,0.1,0.1,574,USD
4 20/07/2010,0.1,0.1,0.1,0.1,262,USD
5 21/07/2010,0.1,0.1,0.1,0.1,575,USD
6 22/07/2010,0.1,0.1,0.1,0.1,2168,USD
7 23/07/2010,0.1,0.1,0.1,0.1,2403,USD
8 24/07/2010,0.1,0.1,0.1,0.1,496,USD
9 25/07/2010,0.1,0.1,0.1,0.1,1551,USD
10 26/07/2010,0.1,0.1,0.1,0.1,877,USD
11 27/07/2010,0.1,0.1,0.1,0.1,3374,USD
12 28/07/2010,0.1,0.1,0.1,0.1,4398,USD
13 29/07/2010,0.1,0.1,0.1,0.1,8058,USD
14 30/07/2010,0.1,0.1,0.1,0.1,3021,USD
15 31/07/2010,0.1,0.1,0.1,0.1,4022,USD
16 01/08/2010,0.1,0.1,0.1,0.1,2601,USD
17 02/08/2010,0.1,0.1,0.1,0.1,3599,USD
18 03/08/2010,0.1,0.1,0.1,0.1,9821,USD
19 04/08/2010,0.1,0.1,0.1,0.1,3494,USD
20 05/08/2010,0.1,0.1,0.1,0.1,5834,USD
21 06/08/2010,0.1,0.1,0.1,0.1,1395,USD
22 07/08/2010,0.1,0.1,0.1,0.1,2619,USD
23 08/08/2010,0.1,0.1,0.1,0.1,2201,USD
24 09/08/2010,0.1,0.1,0.1,0.1,13631,USD
25 10/08/2010,0.1,0.1,0.1,0.1,1310,USD
26 11/08/2010,0.1,0.1,0.1,0.1,14061,USD
27 12/08/2010,0.1,0.1,0.1,0.1,2062,USD
28 13/08/2010,0.1,0.1,0.1,0.1,3592,USD
29 14/08/2010,0.1,0.1,0.1,0.1,4404,USD
30 15/08/2010,0.1,0.1,0.1,0.1,4463,USD
31 16/08/2010,0.1,0.1,0.1,0.1,10731,USD
32 17/08/2010,0.1,0.1,0.1,0.1,13186,USD
33 18/08/2010,0.1,0.1,0.1,0.1,2954,USD
34 19/08/2010,0.1,0.1,0.1,0.1,741,USD
35 20/08/2010,0.1,0.1,0.1,0.1,4200,USD
36 21/08/2010,0.1,0.1,0.1,0.1,10444,USD
37 22/08/2010,0.1,0.1,0.1,0.1,18649,USD
38 23/08/2010,0.1,0.1,0.1,0.1,4297,USD
39 24/08/2010,0.1,0.1,0.1,0.1,6712,USD
40 25/08/2010,0.1,0.1,0.1,0.1,4229,USD
41 26/08/2010,0.1,0.1,0.1,0.1,3878,USD
42 27/08/2010,0.1,0.1,0.1,0.1,9818,USD
43 28/08/2010,0.1,0.1,0.1,0.1,6174,USD
44 29/08/2010,0.1,0.1,0.1,0.1,3173,USD
45 30/08/2010,0.1,0.1,0.1,0.1,34193,USD
46 31/08/2010,0.1,0.1,0.1,0.1,14887,USD
47 01/09/2010,0.1,0.1,0.1,0.1,7165,USD
48 02/09/2010,0.1,0.1,0.1,0.1,8151,USD
```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Plain Text Codemod: [...] ⌂

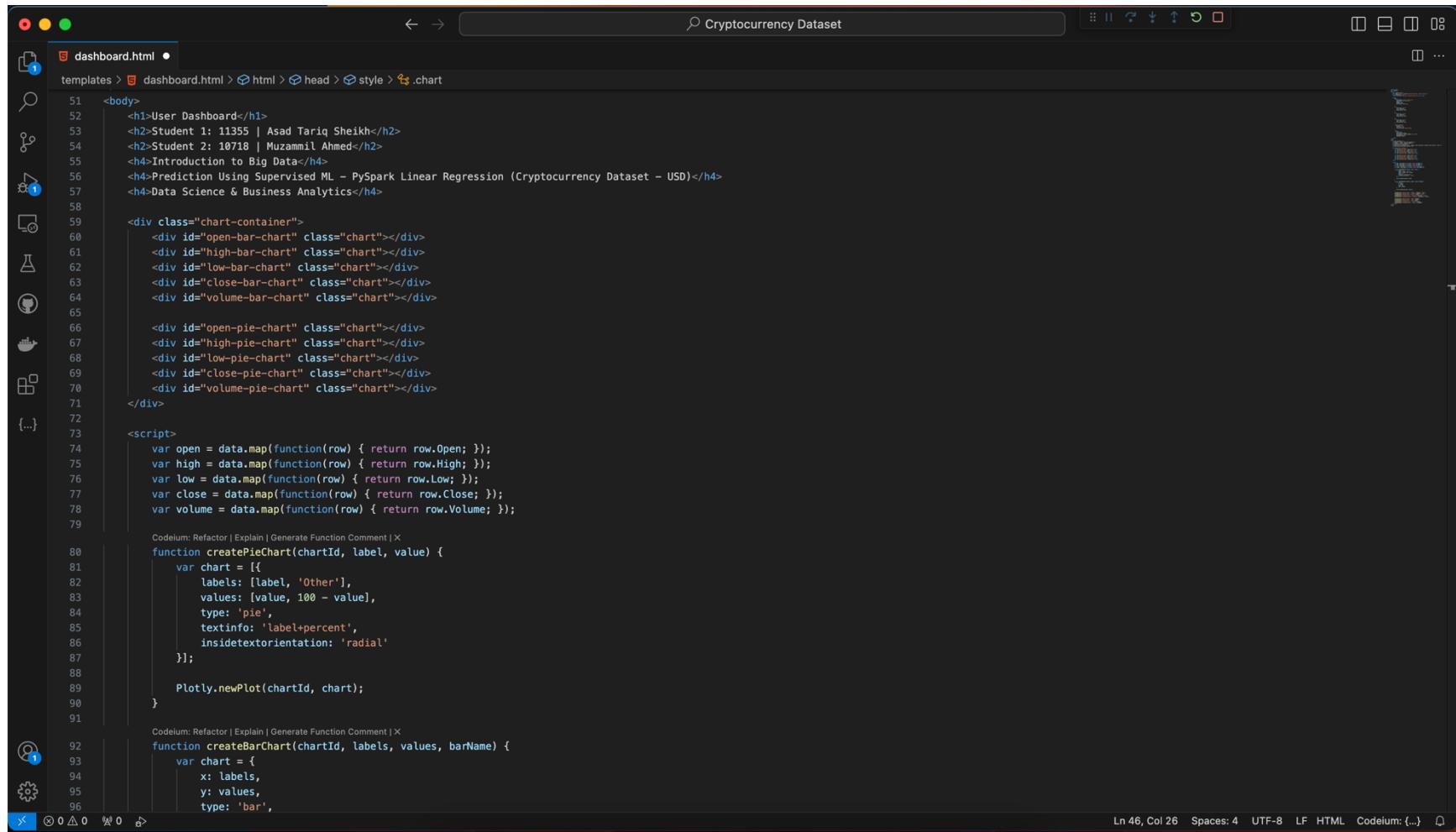
Cryptocurrency Dataset

```
dashboard.html •
templates > dashboard.html > html > body > script
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Dashboard</title>
7      <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
8
9      <style>
10         body {
11             font-family: 'Arial', sans-serif;
12             background-color: #f5f5f5;
13             margin: 0;
14             padding: 20px;
15             box-sizing: border-box;
16         }
17
18         h1 {
19             text-align: center;
20             color: #3498db;
21             margin-bottom: 30px;
22         }
23
24         h2 {
25             text-align: center;
26             color: #528e19;
27             margin-bottom: 24px;
28         }
29
30         h4 {
31             text-align: center;
32             color: #463f92;
33             margin-bottom: 24px;
34         }
35
36         .chart-container {
37             display: flex;
38             flex-wrap: wrap;
39             justify-content: space-around;
40         }
41
42         .chart {
43             width: 45%;
44             margin-bottom: 20px;
45             box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
46             border-radius: 8px;
47             overflow: hidden;
48         }

```

Ln 75, Col 1 Spaces: 4 UTF-8 LF HTML Codeium: {}

DASHBOARD.HTML FILE



```
templates > dashboard.html > head > style > .chart
51  <body>
52    <h1>User Dashboard</h1>
53    <h2>Student 1: 11355 | Asad Tariq Sheikh</h2>
54    <h2>Student 2: 10718 | Muzammil Ahmed</h2>
55    <h4>Introduction to Big Data</h4>
56    <h4>Prediction Using Supervised ML - PySpark Linear Regression (Cryptocurrency Dataset - USD)</h4>
57    <h4>Data Science & Business Analytics</h4>
58
59    <div class="chart-container">
60      <div id="open-bar-chart" class="chart"></div>
61      <div id="high-bar-chart" class="chart"></div>
62      <div id="low-bar-chart" class="chart"></div>
63      <div id="close-bar-chart" class="chart"></div>
64      <div id="volume-bar-chart" class="chart"></div>
65
66      <div id="open-pie-chart" class="chart"></div>
67      <div id="high-pie-chart" class="chart"></div>
68      <div id="low-pie-chart" class="chart"></div>
69      <div id="close-pie-chart" class="chart"></div>
70      <div id="volume-pie-chart" class="chart"></div>
71    </div>
72
73    <script>
74      var open = data.map(function(row) { return row.Open; });
75      var high = data.map(function(row) { return row.High; });
76      var low = data.map(function(row) { return row.Low; });
77      var close = data.map(function(row) { return row.Close; });
78      var volume = data.map(function(row) { return row.Volume; });
79
80      Codeium: Refactor | Explain | Generate Function Comment | X
81      function createPieChart(charId, label, value) {
82        var chart = [
83          {
84            labels: [label, 'Other'],
85            values: [value, 100 - value],
86            type: 'pie',
87            textinfo: 'label+percent',
88            insidetextorientation: 'radial'
89          }];
90
91        Plotly.newPlot(charId, chart);
92      }
93
94      Codeium: Refactor | Explain | Generate Function Comment | X
95      function createBarChart(charId, labels, values, barName) {
96        var chart = {
97          x: labels,
98          y: values,
99          type: 'bar',
100        }
```

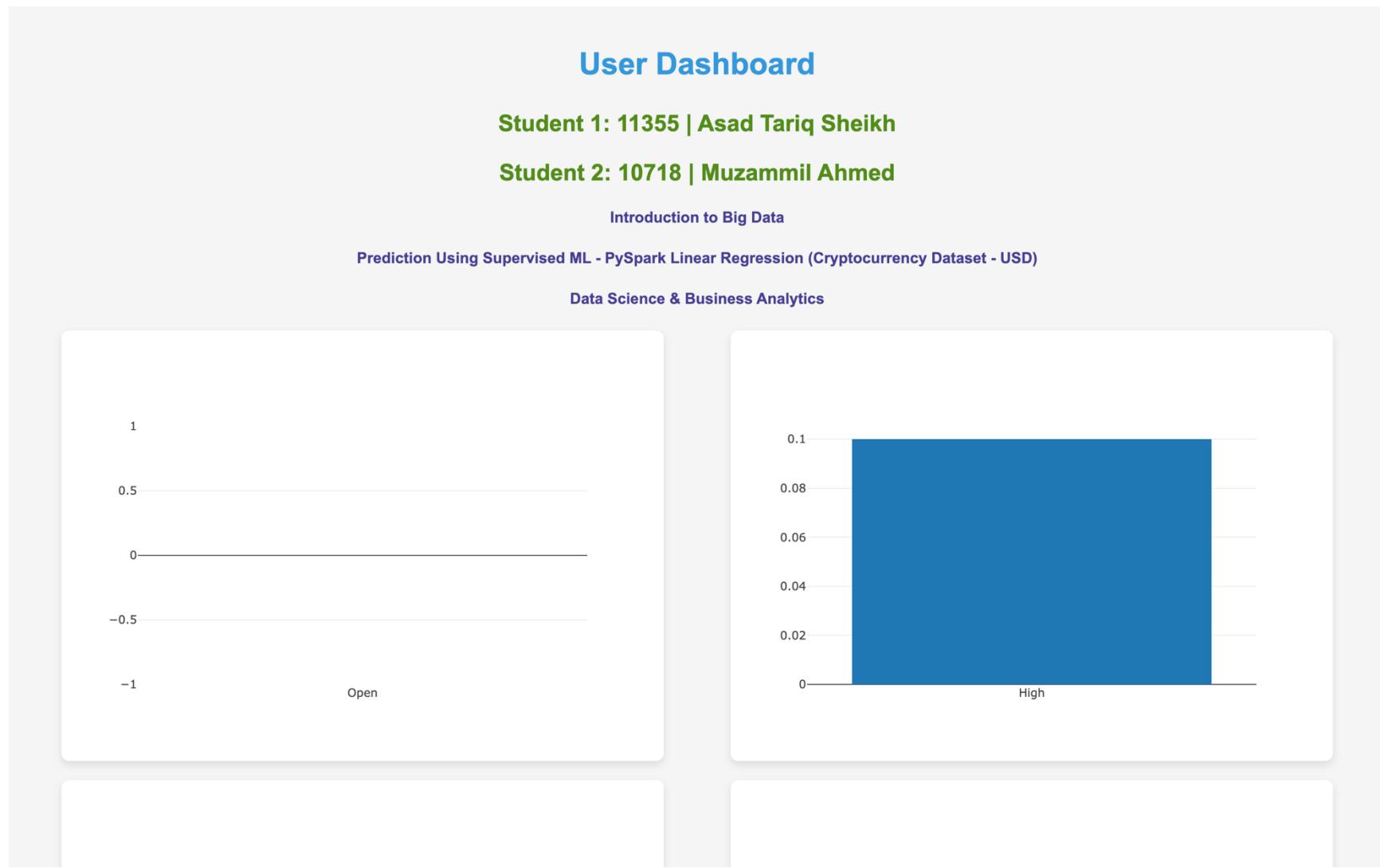
Ln 46, Col 26 Spaces: 4 UTF-8 LF HTML Codeium: (...) ⌂

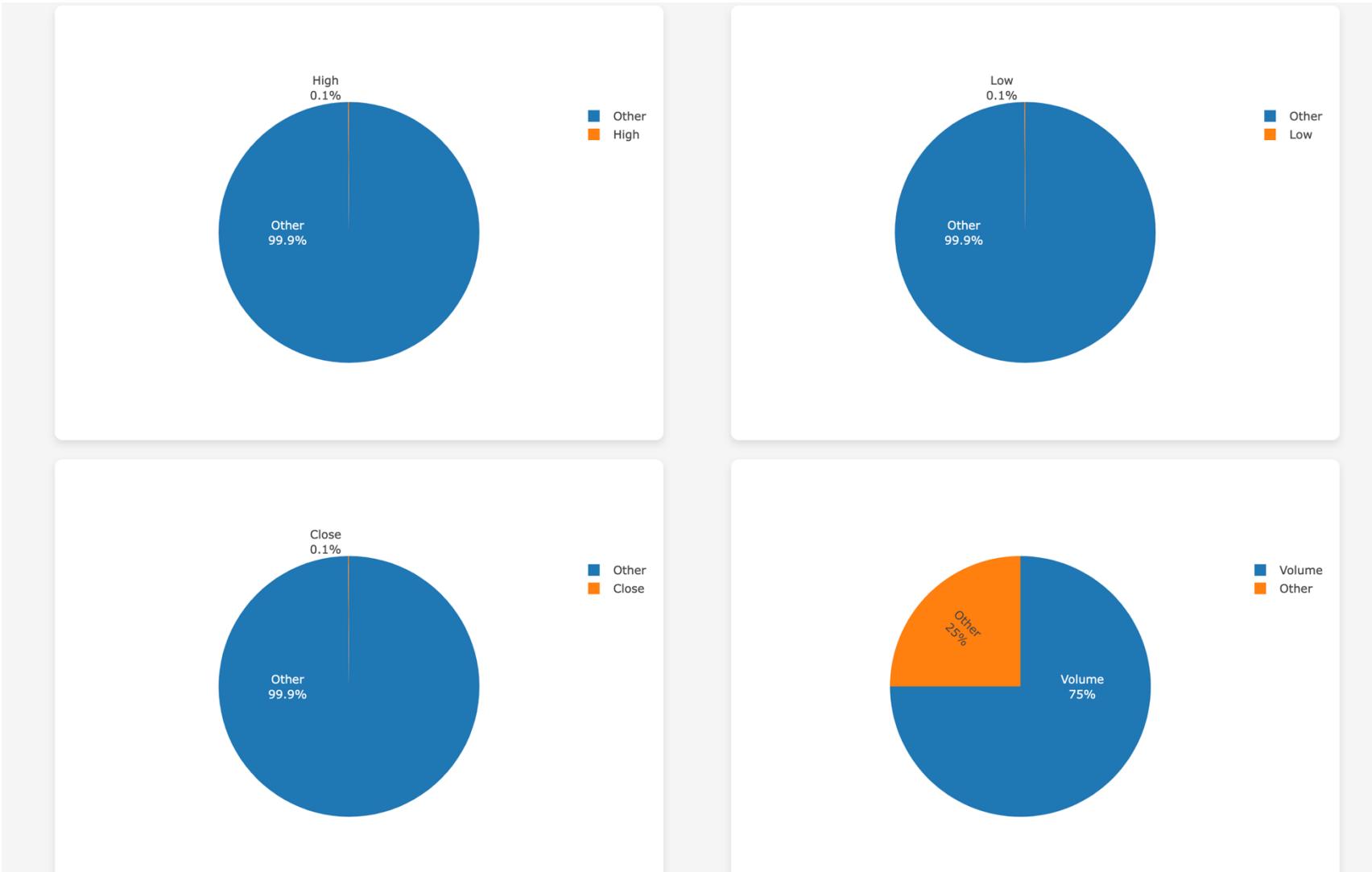
Cryptocurrency Dataset

```
dashboard.html
templates > dashboard.html > html > head > style > .chart
71   </div>
72
73   <script>
74     var open = data.map(function(row) { return row.Open; });
75     var high = data.map(function(row) { return row.High; });
76     var low = data.map(function(row) { return row.Low; });
77     var close = data.map(function(row) { return row.Close; });
78     var volume = data.map(function(row) { return row.Volume; });
79
80     function createPieChart(chartId, label, value) {
81       var chart = [
82         labels: [label, 'Other'],
83         values: [value, 100 - value],
84         type: 'pie',
85         textinfo: 'label+percent',
86         insidetextorientation: 'radial'
87       ];
88
89       Plotly.newPlot(chartId, chart);
90     }
91
92     function createBarChart(chartId, labels, values, barName) {
93       var chart = {
94         x: labels,
95         y: values,
96         type: 'bar',
97         name: barName,
98       };
99
100      Plotly.newPlot(chartId, [chart]);
101    }
102
103    createBarChart('open-bar-chart', ['Open'], [open[0]], 'Open');
104    createBarChart('high-bar-chart', ['High'], [high[0]], 'High');
105    createBarChart('low-bar-chart', ['Low'], [low[0]], 'Low');
106    createBarChart('close-bar-chart', ['Close'], [close[0]], 'Close');
107    createBarChart('volume-bar-chart', ['Volume'], [volume[0]], 'Volume');
108
109    createPieChart('open-pie-chart', 'Open', open[0]);
110    createPieChart('high-pie-chart', 'High', high[0]);
111    createPieChart('low-pie-chart', 'Low', low[0]);
112    createPieChart('close-pie-chart', 'Close', close[0]);
113    createPieChart('volume-pie-chart', 'Volume', volume[0]);
114
115  </script>
116 </body>
117 </html>
```

Ln 46, Col 26 Spaces: 4 UTF-8 LF HTML Codeium: {}

STEP 8: SHOWING DASHBOARD (BAR & PIE GRAPHS)





GITHUB LINK: DOCKER FILE USED

<https://github.com/shaikhahmad/miniLake>

YOUTUBE VIDEO LINK (IN CASE THE VIDEO FILE IS DAMAGED)

<https://youtu.be/MYA7njFURCY>

CONFIGURATION OF MACHINE USED FOR THE PROJECT

Core i7 6th Generation

Quad Core Processor

256Gb SSD

2GB AMD FirePro GPU

RAM: 16GB

Architecture: x86_64

CPU op-mode(s): 32-bit, 64-bit

Address sizes: 39 bits physical, 48 bits virtual

Byte Order: Little Endian

CPU(s): 8

On-line CPU(s) list: 0-7

Vendor ID: GenuineIntel

Model name: Intel(R) Core(TM) i7-6820HQ CPU @ 2.70GHz

CPU family: 6

Model: 94

Thread(s) per core: 2

Core(s) per socket: 4

Socket(s): 1

Stepping: 3

00:02.0 VGA compatible controller: Intel Corporation HD Graphics 530 (rev 06) (prog-if 00 [VGA controller])

asad-latitude

description: Laptop

product: Latitude E5570 (06DF)

vendor: Dell Inc.

serial: FKN8JC2

width: 64 bits

capabilities: smbios-3.0.0 dmi-3.0.0 smp vsyscall32

configuration: boot=normal chassis=laptop family=Latitude sku=06DF uuid=4c4c4544-004b-4e10-8038-c6c04f4a4332

*-core

description: Motherboard

product: 06YF8N

vendor: Dell Inc.

physical id: 0

version: A00

serial: /FKN8JC2/CN1296366U0028/

*-firmware
description: BIOS
vendor: Dell Inc.
physical id: 0
version: 1.34.3
date: 11/20/2022
size: 64KiB
capacity: 16MiB