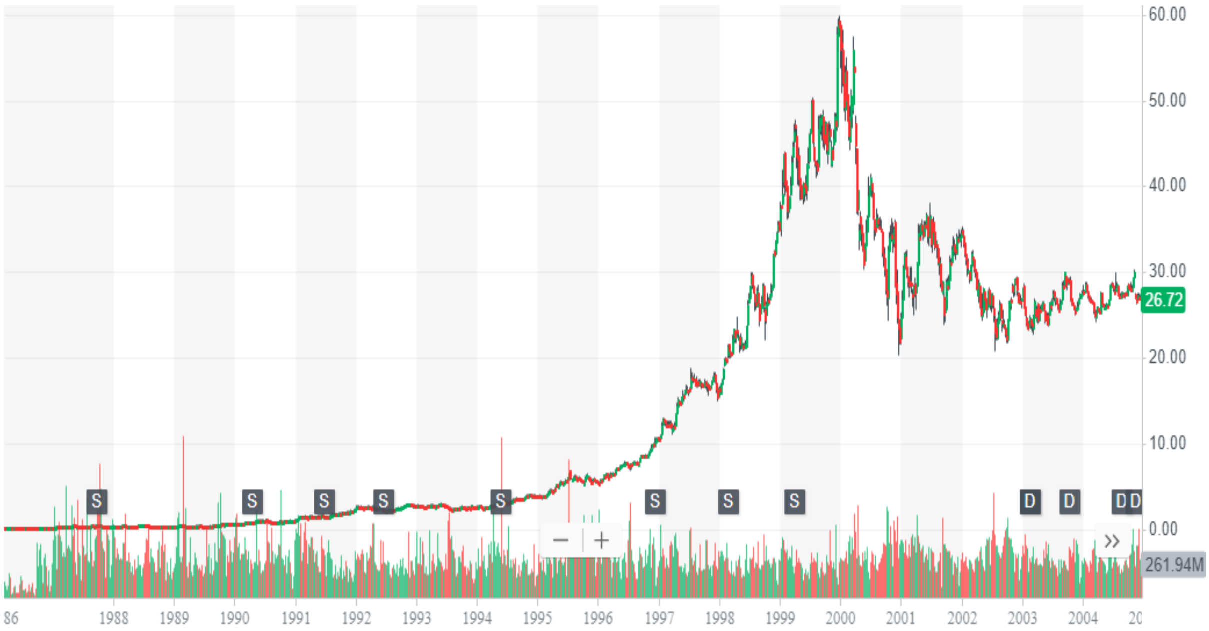
STOCK PRICE PREDICTION

Team Member: **M.Madhan kumar**

Phase 01: Document Submission

PROJECT: STOCK PRICE PREDICTION

STOCK PRICE PREDICTION



Problem Definition and Design Thinking:

In this part you will need to understand the problem statement and create a document on what have you understood and how will you proceed ahead with solving the problem. Please think on a design and present in form of a document.

Problem Definition:

The problem is to build a predictive model that forecasts stock prices based on historical market data. The goal is to create a tool that assists investors in making well-informed decisions and optimizing their investment strategies. This project involves data collection, data preprocessing, feature engineering, model selection, training, and evaluation.

Design Thinking:

1. Data Collection:

Data collection is the process of gathering historical stock market data, which includes essential features like date, open price, close price, volume, and other relevant indicators. This data serves as the foundation for building a stock price prediction model.

2. Data Preprocessing:

Data preprocessing refers to the steps taken to clean and prepare the collected stock market data for analysis. This includes handling missing values, removing outliers, and converting categorical features into numerical representations, ensuring that the data is in a suitable format for modeling.

3. Feature Engineering:

Feature engineering involves creating additional features or variables that have the potential to enhance the predictive power of the stock price prediction model. This can include calculating moving averages, incorporating technical indicators (e.g., Relative Strength Index), and creating lagged variables that capture historical price movements.

4. Model Selection:

Model selection is the process of choosing appropriate algorithms or methods for time series forecasting in the context of stock price prediction. Common choices include AutoRegressive Integrated Moving Average (ARIMA) models, Long Short-Term Memory (LSTM) neural networks, or other machine learning techniques tailored to time series data.

5. Model Training:

Model training involves using the preprocessed historical stock market data to teach the selected forecasting model how to make predictions. During this phase, the model learns patterns and relationships in the data that will enable it to make future stock price predictions.

6. Evaluation:

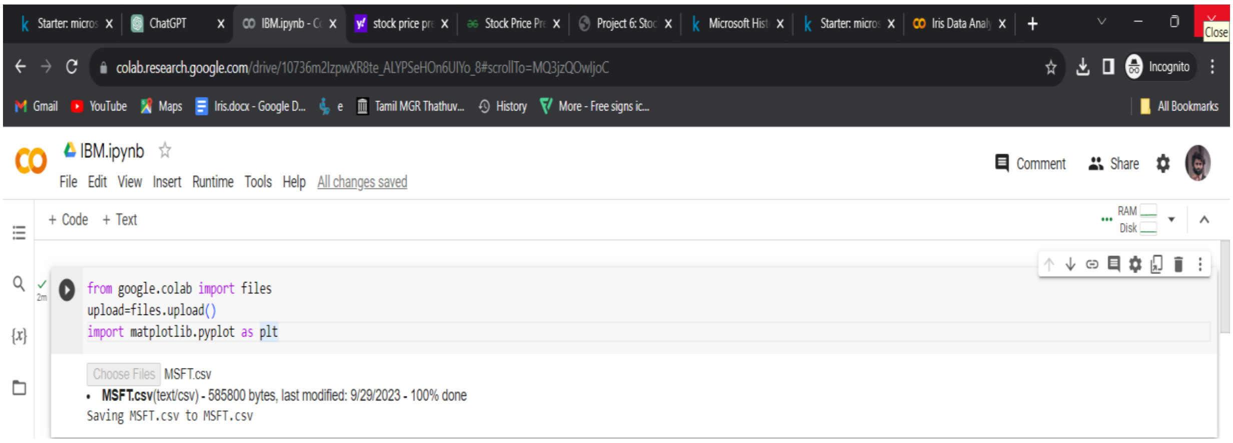
Evaluation is the assessment of the stock price prediction model's performance. This typically involves comparing the model's predictions to actual stock prices over a specified evaluation period. Common evaluation metrics in time series forecasting include Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), which quantify the accuracy of the model's predictions.

Data source:

IMPORTING .CSV FILE:

```
from google.colab import files
upload=files.upload()
import matplotlib.pyplot as plt
```

SUCCESSFULLY IMPORTED:



```
from google.colab import files
upload=files.upload()
import matplotlib.pyplot as plt
```

Choose Files MSFT.csv

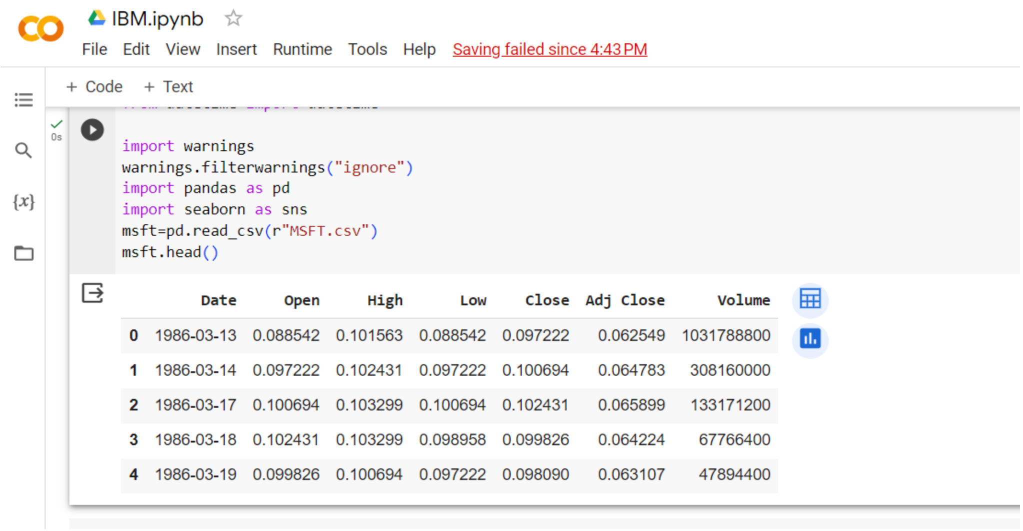
- **MSFT.csv**(text/csv) - 585800 bytes, last modified: 9/29/2023 - 100% done

```
Saving MSFT.csv to MSFT.csv
```

READING DATA IN CSV FILE:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import tensorflow as tf
from tensorflow import keras
import seaborn as sns
import os
from datetime import datetime

import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import seaborn as sns
msft=pd.read_csv(r"MSFT.csv")
msft.head()
```

OUTPUT:


IBM.ipynb
File Edit View Insert Runtime Tools Help Saving failed since 4:43 PM
+ Code + Text

```python
from datetime import datetime

import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import seaborn as sns
msft=pd.read_csv(r"MSFT.csv")
msft.head()
```

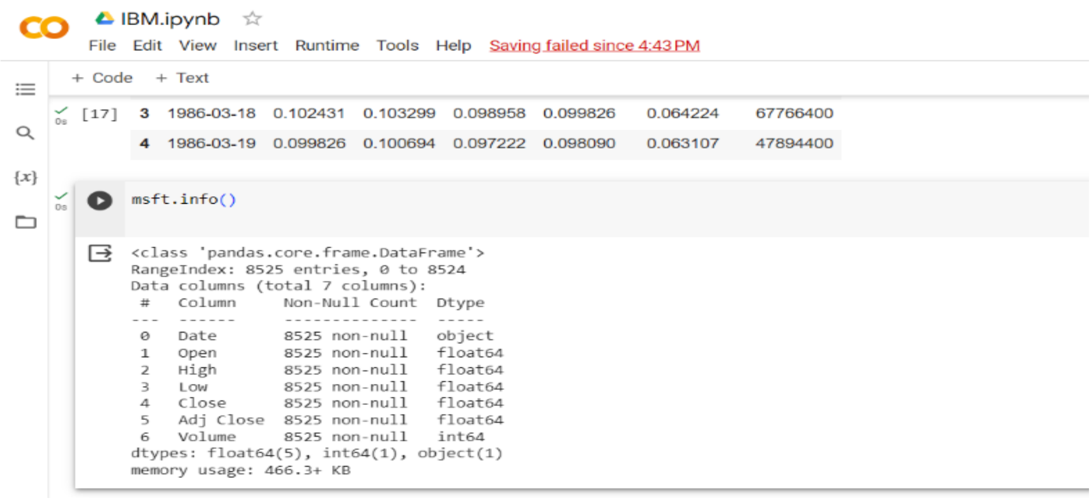
	Date	Open	High	Low	Close	Adj Close	Volume
0	1986-03-13	0.088542	0.101563	0.088542	0.097222	0.062549	1031788800
1	1986-03-14	0.097222	0.102431	0.097222	0.100694	0.064783	308160000
2	1986-03-17	0.100694	0.103299	0.100694	0.102431	0.065899	133171200
3	1986-03-18	0.102431	0.103299	0.098958	0.099826	0.064224	67766400
4	1986-03-19	0.099826	0.100694	0.097222	0.098090	0.063107	47894400

DATA PROCESSING:

MSFT.CSV INFO( ):

```python
msft.info()
```

OUTPUT:


IBM.ipynb
File Edit View Insert Runtime Tools Help Saving failed since 4:43 PM
+ Code + Text
[17]

3	1986-03-18	0.102431	0.103299	0.098958	0.099826	0.064224	67766400
4	1986-03-19	0.099826	0.100694	0.097222	0.098090	0.063107	47894400

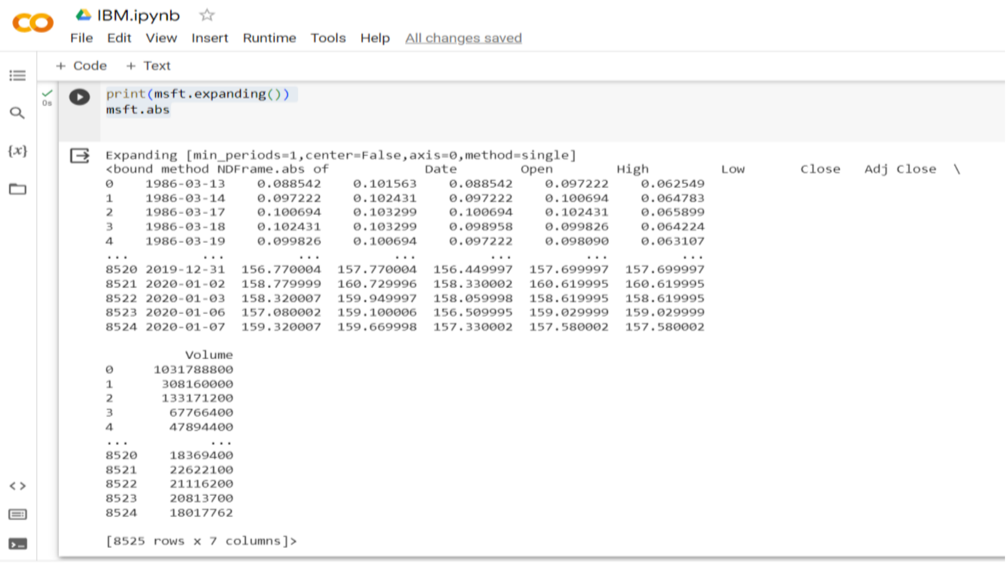
```python
msft.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8525 entries, 0 to 8524
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       8525 non-null   object
 1   Open       8525 non-null   float64
 2   High       8525 non-null   float64
 3   Low        8525 non-null   float64
 4   Close      8525 non-null   float64
 5   Adj Close  8525 non-null   float64
 6   Volume     8525 non-null   int64
dtypes: float64(5), int64(1), object(1)
memory usage: 466.3+ KB
```

EXPANSION OF CSV FILE:

```
print(msft.expanding())
msft.abs
```

OUTPUT:



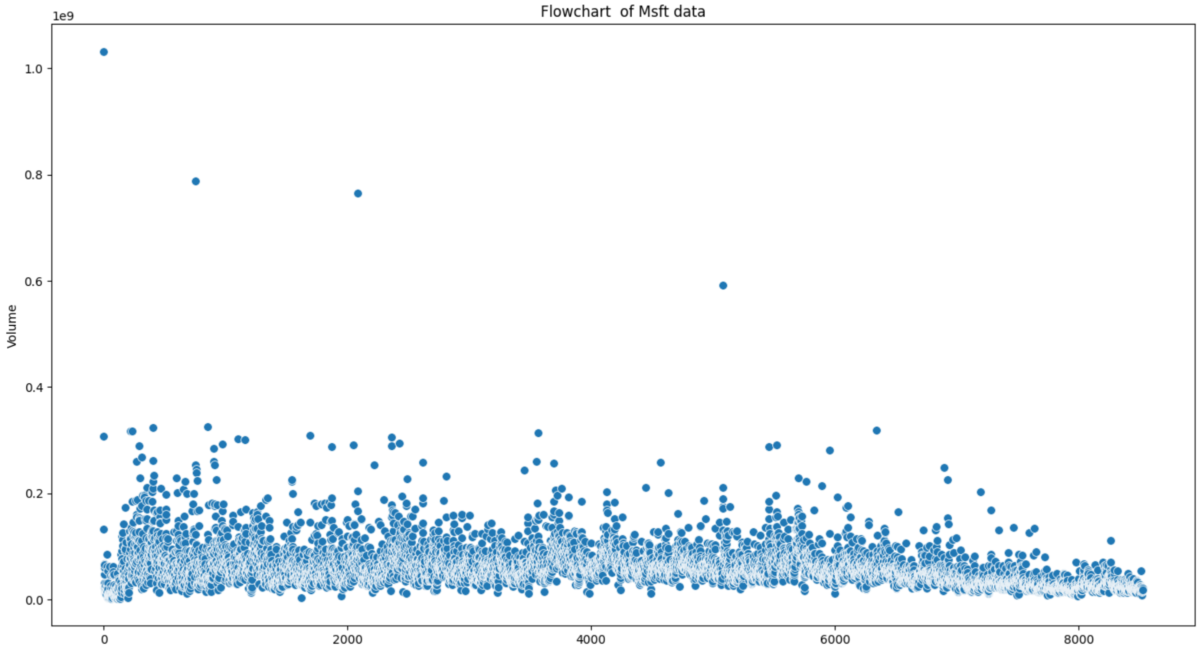
```
print(msft.expanding())
msft.abs
```

```
Expanding [min_periods=1,center=False,axis=0,method=single]
<bound method NDFrame.abs of             Date      Open      High       Low     Close   Adj Close  \
0     1986-03-13    0.088542    0.101563    0.088542    0.097222    0.062549
1     1986-03-14    0.097222    0.102431    0.097222    0.100694    0.064783
2     1986-03-17    0.100694    0.103299    0.100694    0.102431    0.065899
3     1986-03-18    0.102431    0.103299    0.098958    0.099826    0.064224
4     1986-03-19    0.099826    0.100694    0.097222    0.098090    0.063107
...          ...         ...         ...         ...         ...         ...
8520 2019-12-31  156.770004  157.770004  156.449997  157.699997  157.699997
8521 2020-01-02  158.779999  160.729996  158.330002  160.619995  160.619995
8522 2020-01-03  158.320007  159.949997  158.059998  158.619995  158.619995
8523 2020-01-06  157.080002  159.100006  156.509995  159.029999  159.029999
8524 2020-01-07  159.320007  159.669998  157.330002  157.580002  157.580002

          Volume
0     1031788800
1      308160000
2      133171200
3       67766400
4       47894400
...          ...
8520    18369400
8521    22622100
8522    21116200
8523    20813700
8524    18017762

[8525 rows x 7 columns]>
```

FLOW CHART OF VOLUME:

```
import seaborn as sns
plt.figure(figsize=(17,9))
plt.title('Flowchart  of Msft data')
sns.scatterplot(msft['Volume'],s=50)
```

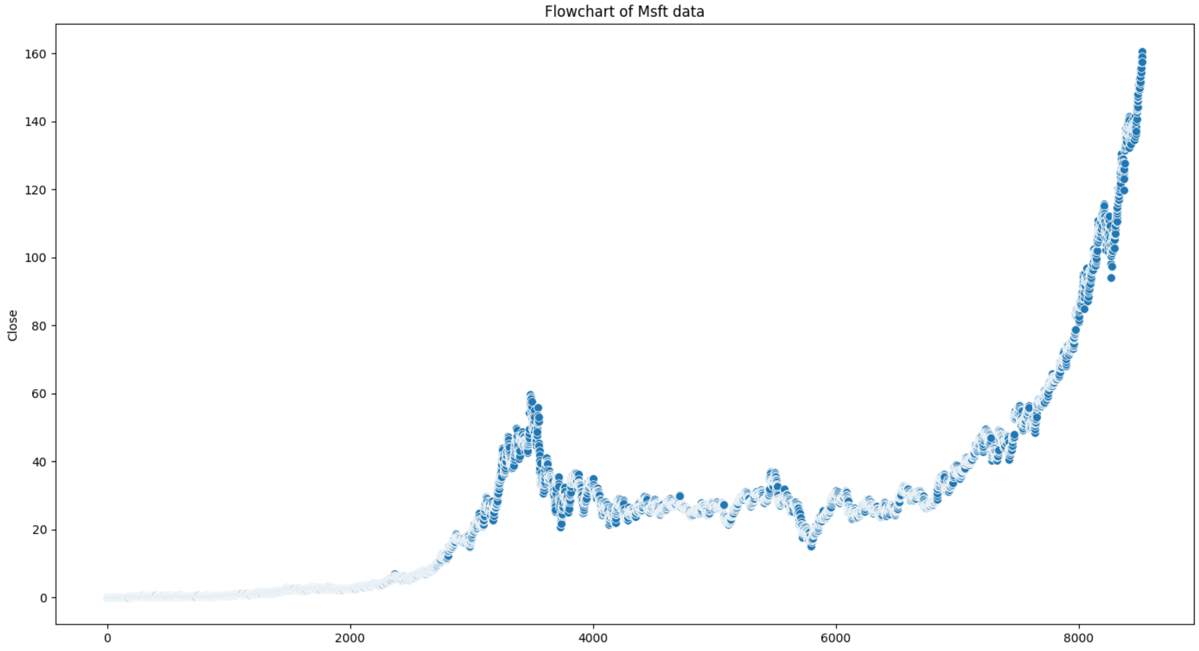
OUTPUT:



FLOWCHART OF CLOSE COLUMN:

```
plt.figure(figsize=(17,9))
plt.title('Flowchart of Msft data')
sns.scatterplot(msft['Close'],s=50)
```

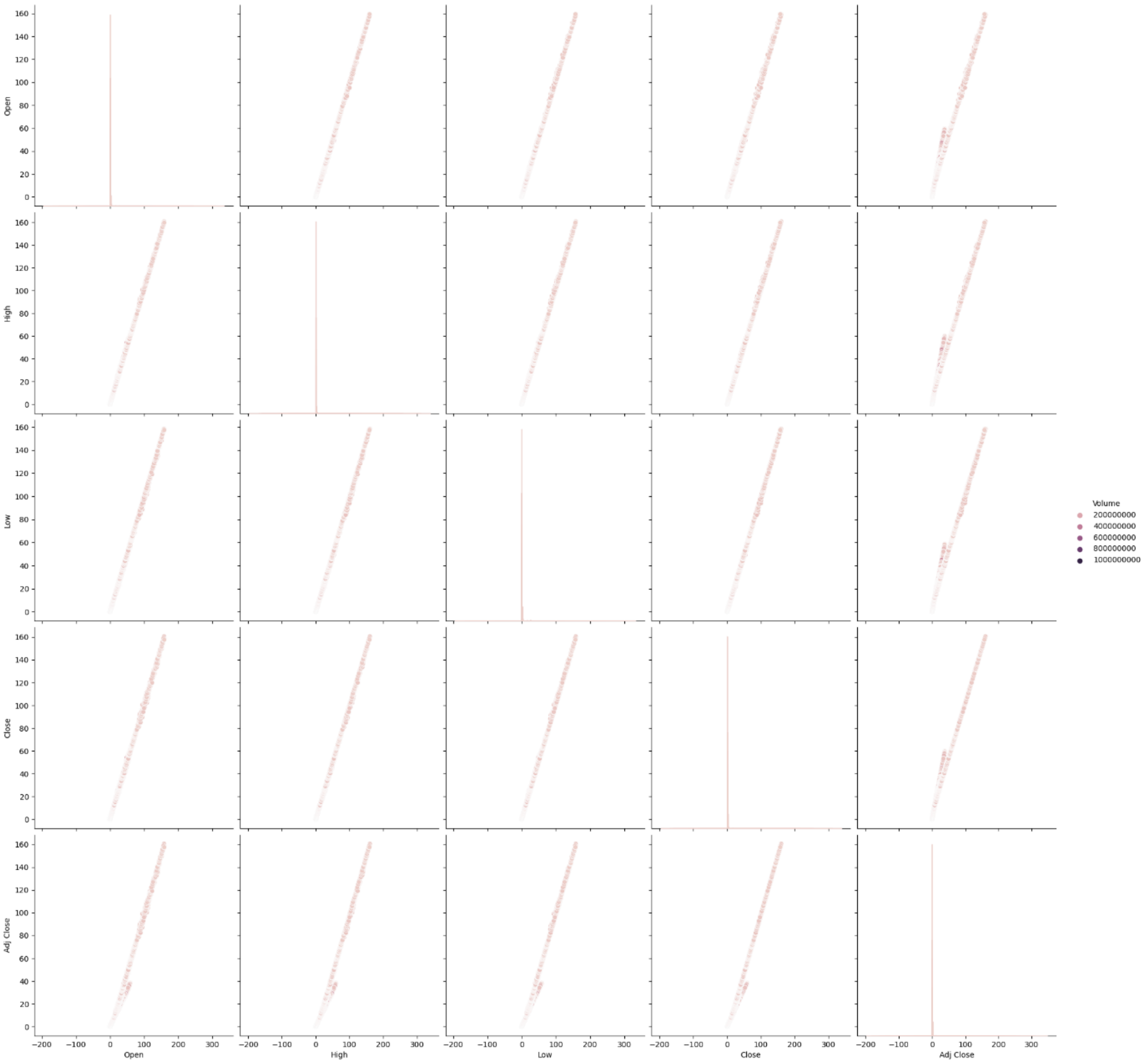
OUTPUT:



PAIRPLOTTING OF CSV FILE:

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.pairplot(data=msft,hue="Volume",height=4)
```

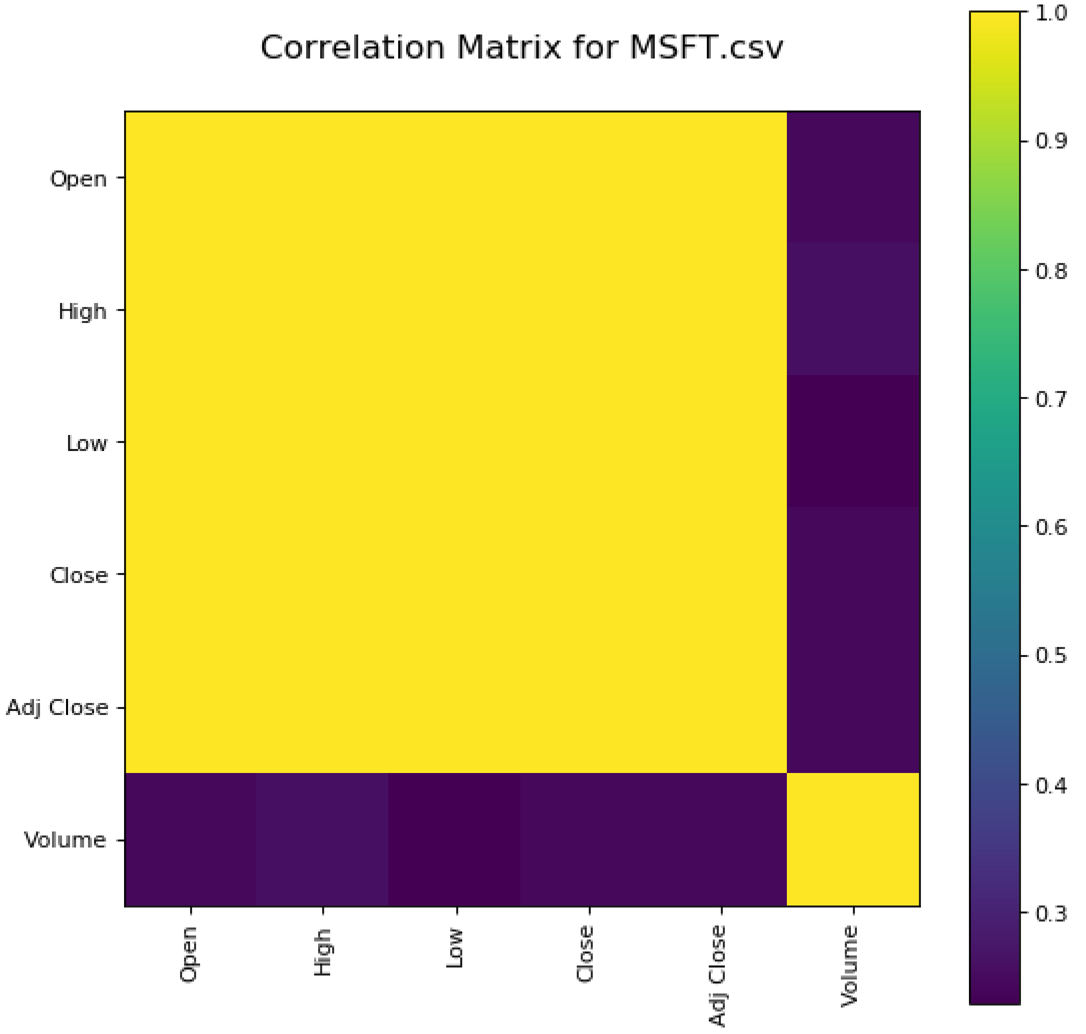
OUTPUT:



CORRELATION MATRIX:

```
plotCorrelationMatrix(df1, 8)
```

OUTPUT:



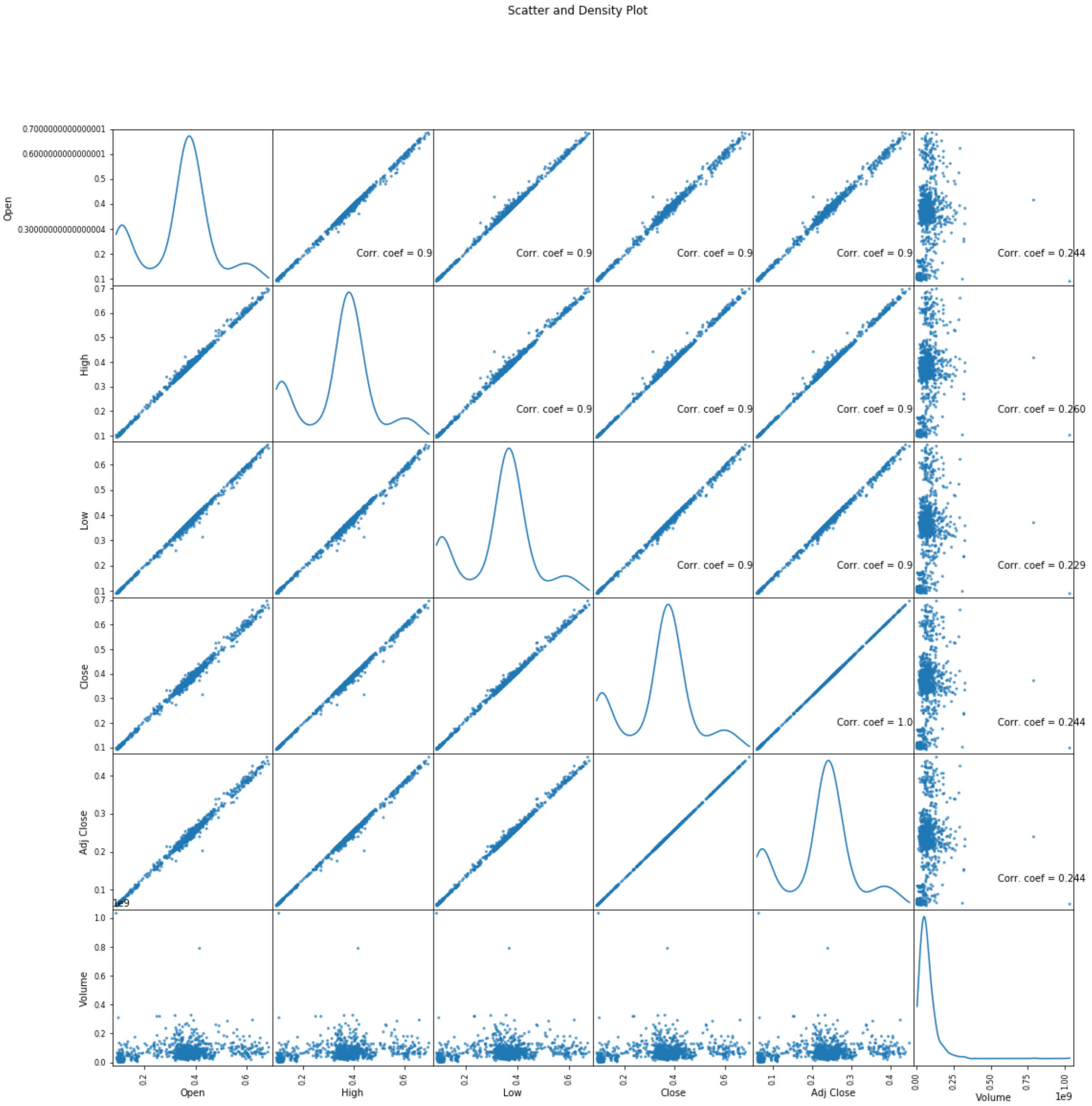
H

Scatter and density plots:

In [10]:

```
plotScatterMatrix(df1, 18, 10)
```

OUTPUT:



CONCLUSION:

Hence the stock price predictions using MSFT.csv file has been operated in many ways and shows the market price predictions based on the data.Here from the data’s in the csv file is compared from one another and flow chart is drawn.

…THANK YOU…