



Microsoft Stock Price Prediction

<https://github.com/magnusaghe/ANA500>

Magnus Aghe
20 August 2023





Problem Statement

- A common feature of the stock market is its volatility and dynamism. For an individual stock, volatility represents the measure of the frequency and magnitude of movements in its price. These fluctuations are often used by investors to gauge risk and to help in the prediction of future price movements.
- In this project, a Microsoft stock price dataset (from April 1, 2015, to March 31, 2021) is analyzed to interrogate which features have the largest effect on its price. Additionally, machine learning models will be used to predict Microsoft stock price movement.

Objectives

- To know the features that have the greatest effect on the Microsoft stock price.
- To decompose the stock price as a time series data in terms of trend, seasonality, and residuals.
- To measure the volatility in terms of percentage change of daily returns in the price.
- To use deep learning regression methods like recurrent neural networks (RNN) implemented with Long Short Term Memory (LSTM) building blocks for forecasting the stock price.
- To use autocorrelation and ARIMA models.





Hypothesis Formulation

Null Hypothesis (H_0):

- There is no significant association between the closing price of the stock (close) and the attributes:- opening price (open), highest daily price (high), lowest daily price (low), and number of daily traded shares (volume).

Alternative Hypothesis (H_1):

- There is significant association between the closing price of the stock (close) and the attributes:- opening price (open), highest daily price (high), lowest daily price (low), and number of daily traded shares (volume).

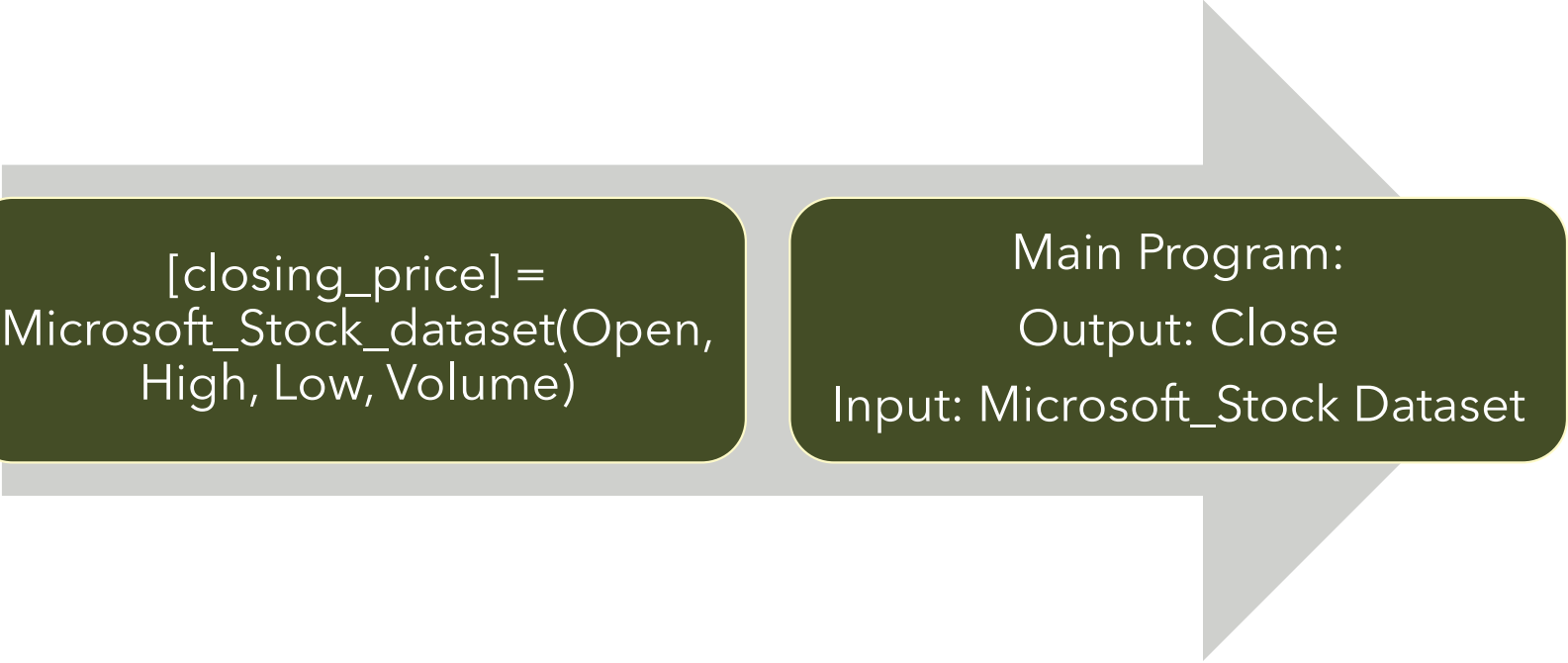
Top-Down Program Design

- The overall task is to develop a Python program that takes a Microsoft stock price dataset and outputs a predicted closing price.
- The program will be implemented using Jupyter notebook and will use libraries such as NumPy, Pandas, matplotlib, seaborn, scikit-learn, and other packages.
- The steps that will be taken to perform the task (Data Science Process) are:
 1. Acquire - Identify and load the dataset
 2. Prepare - Explore and pre-process the data
 3. Analyze - Select analytical techniques, and build models
 4. Report - Communicate results
 5. Act - Apply results, connect results with the chosen business question and objective(s).

Microsoft Stock Price Prediction by Magnus Aghe -
ANA500 Micro-Project



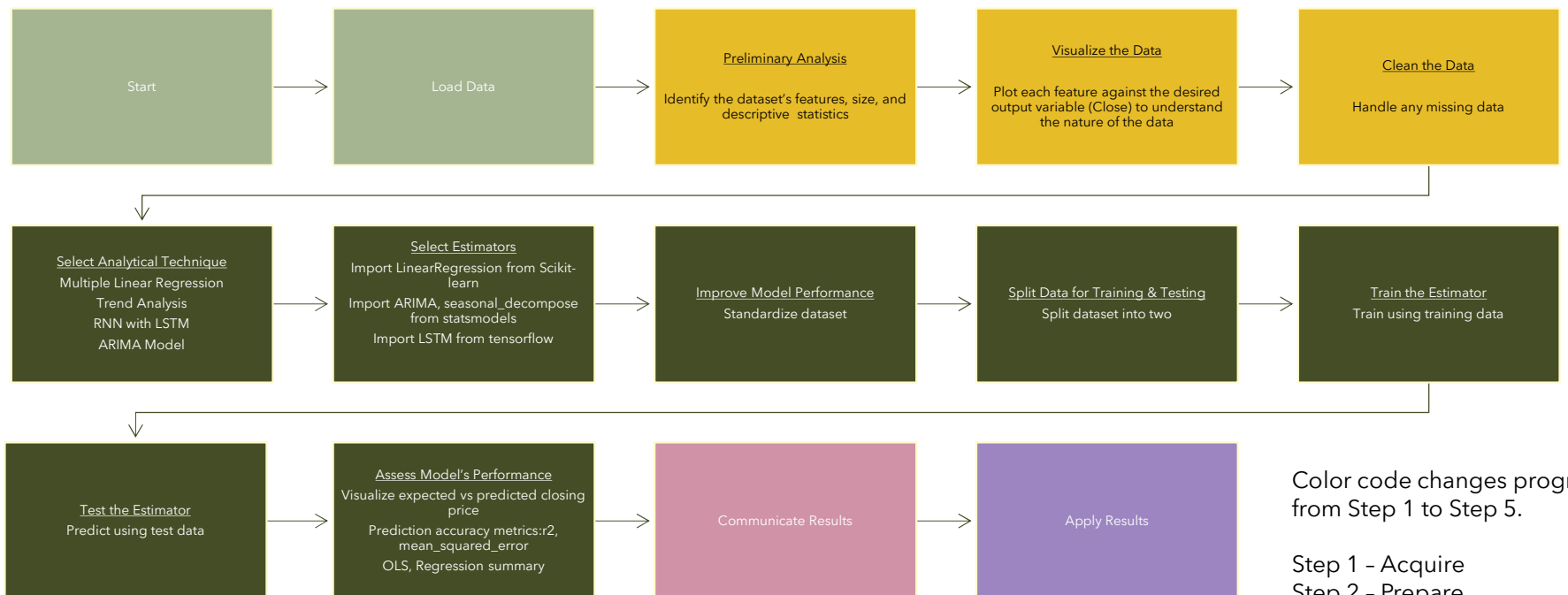
Hierarchy Chart



```
[closing_price] =  
Microsoft_Stock_dataset(Open,  
High, Low, Volume)
```

Main Program:
Output: Close
Input: Microsoft_Stock Dataset

Flow Chart



Color code changes progressively from Step 1 to Step 5.

Step 1 - Acquire
Step 2 - Prepare
Step 3 - Analyze
Step 4 - Report
Step 5 - Act



Acquire

- The dataset used in this study was acquired through Kaggle.com, the world's largest online community of data scientists and machine learning practitioners, owned by Google. Here, people can participate in data science competitions, post projects, and acquire open-source datasets. The dataset contains Microsoft stock price over a six-year period from April 1, 2015, to March 31, 2021.
- Data source:
<https://www.kaggle.com/datasets/vijayvvenkitesh/microsoft-stock-time-series-analysis>

Acquire, cont.

Import NumPy, pandas.

```
In [1]: import warnings
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

warnings.filterwarnings("ignore")
```

```
In [2]: # Load the data
Microsoft_Stock_dataset = pd.read_csv("Microsoft_Stock.csv", parse_dates=[0])
```

Load the data

Microsoft_Stock_dataset = pd.read_csv("Microsoft_Stock.csv", parse_dates=[0])

Prepare

- 'df.head()' function outputs the column headers and the first five rows of the dataset.

```
In [3]: df = Microsoft_Stock_dataset  
df.head()
```

Out[3]:

| | Date | Open | High | Low | Close | Volume |
|---|---------------------|-------|-------|-------|-------|----------|
| 0 | 2015-04-01 16:00:00 | 40.60 | 40.76 | 40.31 | 40.72 | 36865322 |
| 1 | 2015-04-02 16:00:00 | 40.66 | 40.74 | 40.12 | 40.29 | 37487476 |
| 2 | 2015-04-06 16:00:00 | 40.34 | 41.78 | 40.18 | 41.55 | 39223692 |
| 3 | 2015-04-07 16:00:00 | 41.61 | 41.91 | 41.31 | 41.53 | 28809375 |
| 4 | 2015-04-08 16:00:00 | 41.48 | 41.69 | 41.04 | 41.42 | 24753438 |

Prepare, cont.

- Shape & Features: The dataset has 1511 rows and 6 features/attributes.

```
In [7]: df.shape
```

```
Out[7]: (1511, 6)
```

```
In [8]: df.columns
```

```
Out[8]: Index(['Date', 'Open', 'High', 'Low', 'Close', 'Volume'], dtype='object')
```


- Datatype:

```
In [11]: # Datatype info  
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 1511 entries, 0 to 1510  
Data columns (total 6 columns):  
#   Column  Non-Null Count  Dtype  
---  ---  
0   Date    1511 non-null   datetime64[ns]  
1   Open    1511 non-null   float64  
2   High    1511 non-null   float64  
3   Low     1511 non-null   float64  
4   Close   1511 non-null   float64  
5   Volume  1511 non-null   int64  
dtypes: datetime64[ns](1), float64(4), int64(1)  
memory usage: 71.0 KB
```

Prepare, cont.

- Summary Statistics:

```
In [5]:  #summary statistics  
df.describe()
```

Out[5]:

| | Open | High | Low | Close | Volume |
|-------|-------------|-------------|-------------|-------------|--------------|
| count | 1511.000000 | 1511.000000 | 1511.000000 | 1511.000000 | 1.511000e+03 |
| mean | 107.385976 | 108.437472 | 106.294533 | 107.422091 | 3.019863e+07 |
| std | 56.691333 | 57.382276 | 55.977155 | 56.702299 | 1.425266e+07 |
| min | 40.340000 | 40.740000 | 39.720000 | 40.290000 | 1.016120e+05 |
| 25% | 57.860000 | 58.060000 | 57.420000 | 57.855000 | 2.136213e+07 |
| 50% | 93.990000 | 95.100000 | 92.920000 | 93.860000 | 2.662962e+07 |
| 75% | 139.440000 | 140.325000 | 137.825000 | 138.965000 | 3.431962e+07 |
| max | 245.030000 | 246.130000 | 242.920000 | 244.990000 | 1.352271e+08 |



Start of Micro- Project 2

Microsoft Stock Price Prediction by Magnus Aghe - ANA500 Micro-Project

Prepare, cont.

- Checking for missing values.
- Checking for unique values.

```
In [9]: # check for null values  
df.isnull().sum()
```

```
Out[9]: Date      0  
       Open      0  
       High      0  
       Low       0  
       Close     0  
       Volume    0  
       dtype: int64
```

- There are no missing values in the dataset.

```
In [10]: # Unique values
```

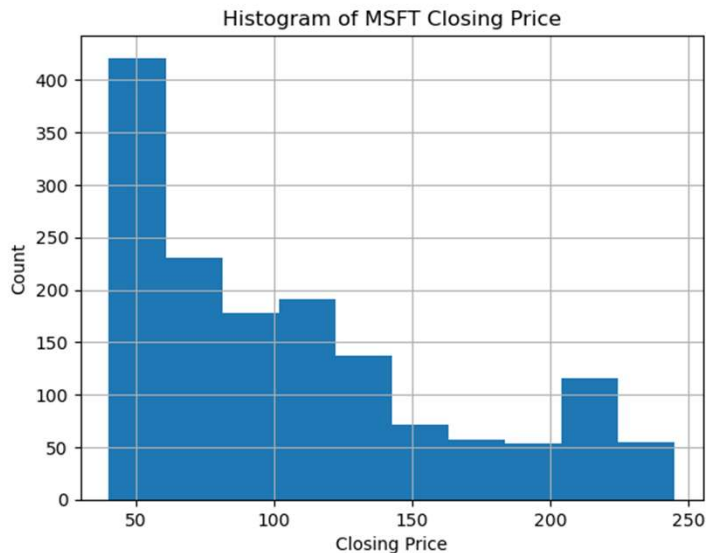
```
df.nunique().sort_values(ascending=True)
```

```
Out[10]: Low      1397  
        Close    1398  
        High     1400  
        Open     1409  
        Date      1511  
        Volume    1511  
        dtype: int64
```

The values in every variable are very diverse.

Analyze

Exploratory Data Analysis - Visualizing the Data



The stock price closed at about 50 dollars more than 400 times, almost double the nearest closing price.

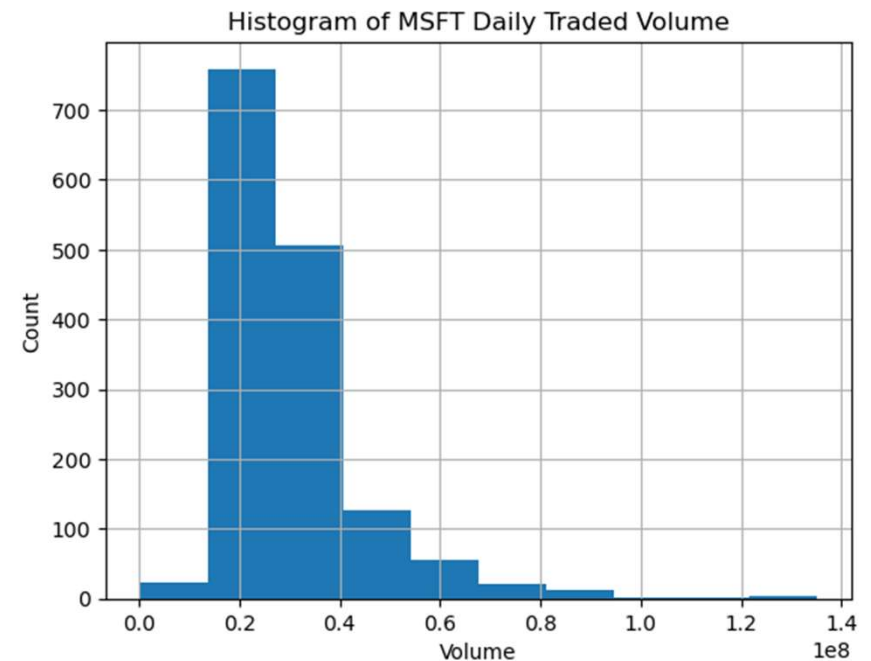
Box plot to visualize the distribution of closing price



Analyze, cont.

Distribution of Daily Traded Volume of MSFT Stock

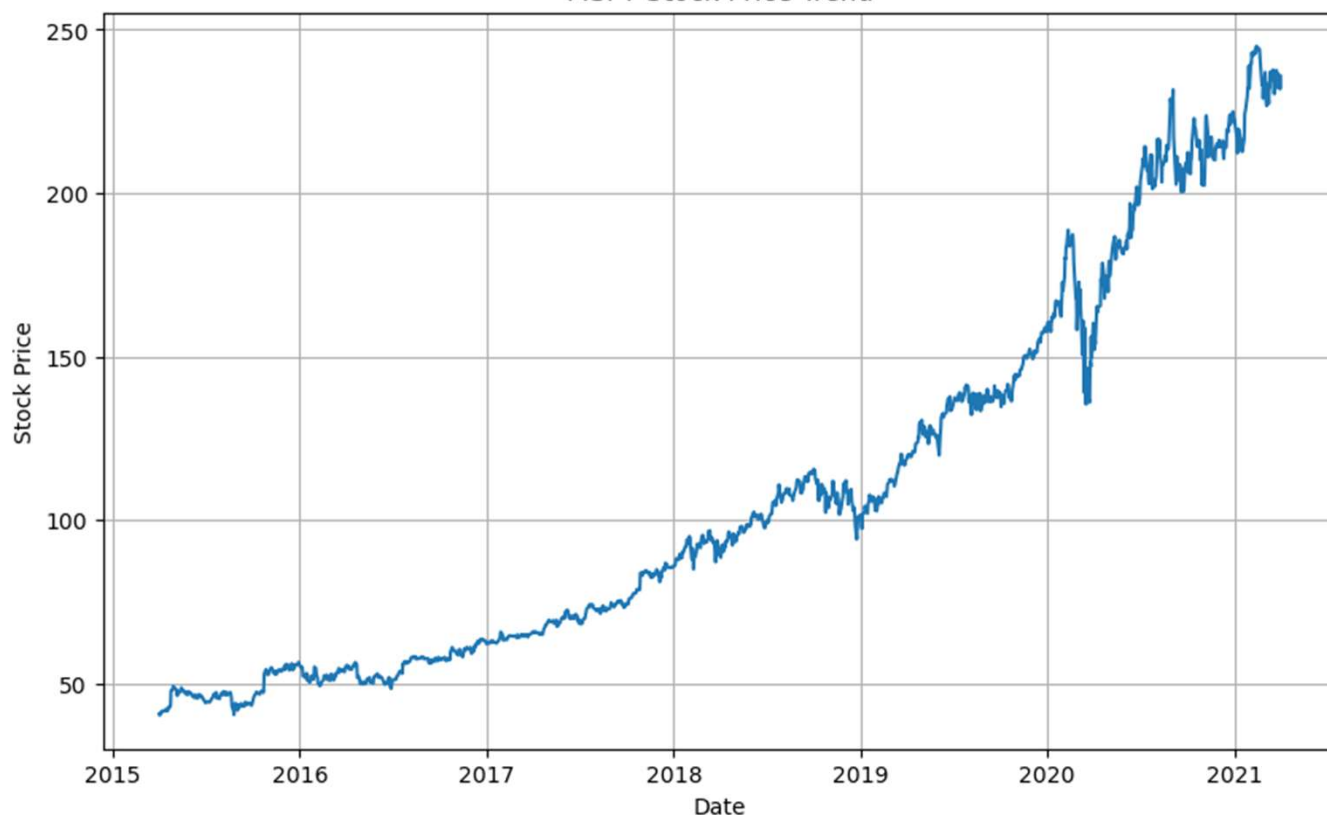
- The mean daily traded volume was 30.198 million shares.
- Minimum was 101,612 shares while maximum daily traded volume was 135.227 million shares.





Analyze, cont.

MSFT Stock Price Trend



Microsoft Stock Price Prediction by Magnus Aghe -
ANA500 Micro-Project

Time Series Plot of MSFT Closing Price

- We see an overall upward trend of MSFT stock price in the period under review (2015 - 2021).

Analyze, cont.

Scatterplot & Correlation Matrix

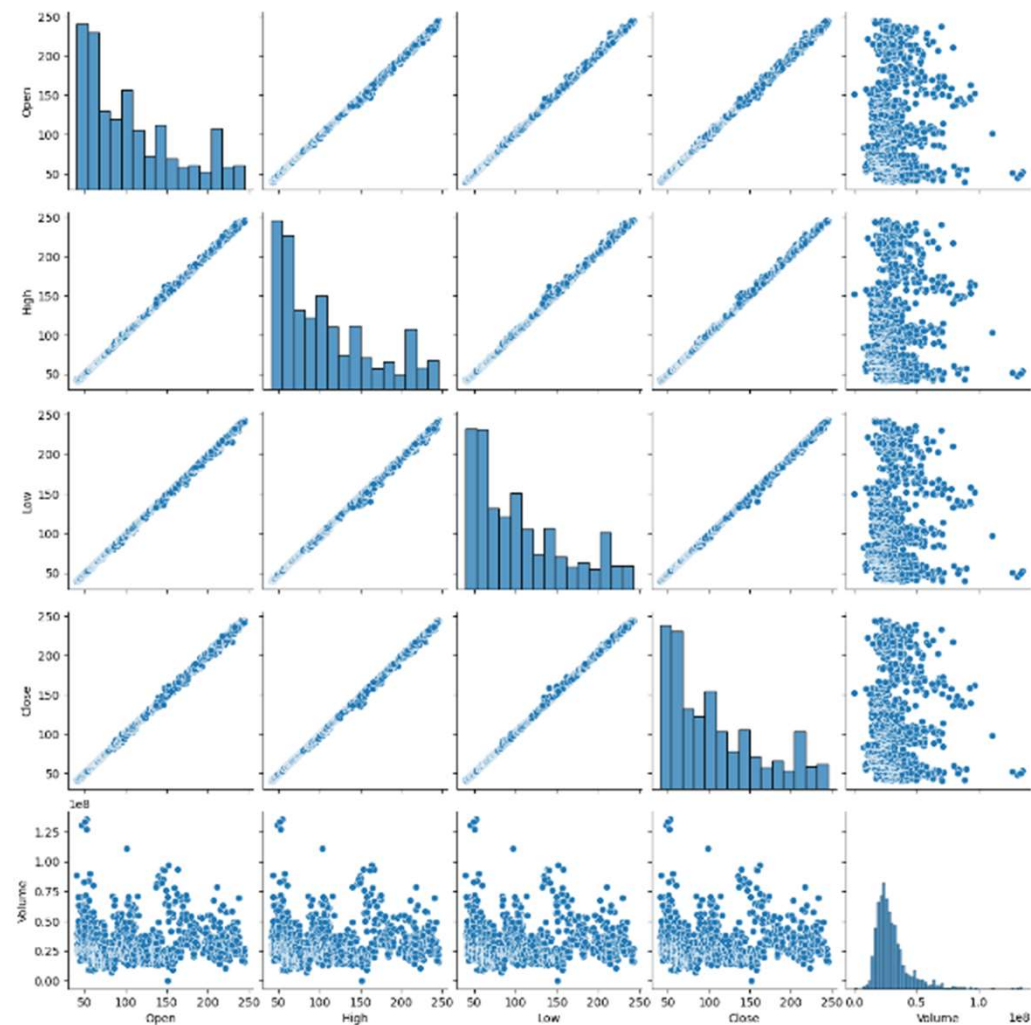
| | Open | High | Low | Close | Volume |
|--------|------|------|------|-------|--------|
| Open | 1.00 | 1.00 | 1.00 | 1.00 | 0.05 |
| High | 1.00 | 1.00 | 1.00 | 1.00 | 0.06 |
| Low | 1.00 | 1.00 | 1.00 | 1.00 | 0.04 |
| Close | 1.00 | 1.00 | 1.00 | 1.00 | 0.05 |
| Volume | 0.05 | 0.06 | 0.04 | 0.05 | 1.00 |

Microsoft S

```
In [17]: # Let's visualize the relationship between the quantitative variables (Open, High, Low, Close, Volume).
```

```
# Scatterplot matrix  
sns.pairplot(df)  
plt.show
```

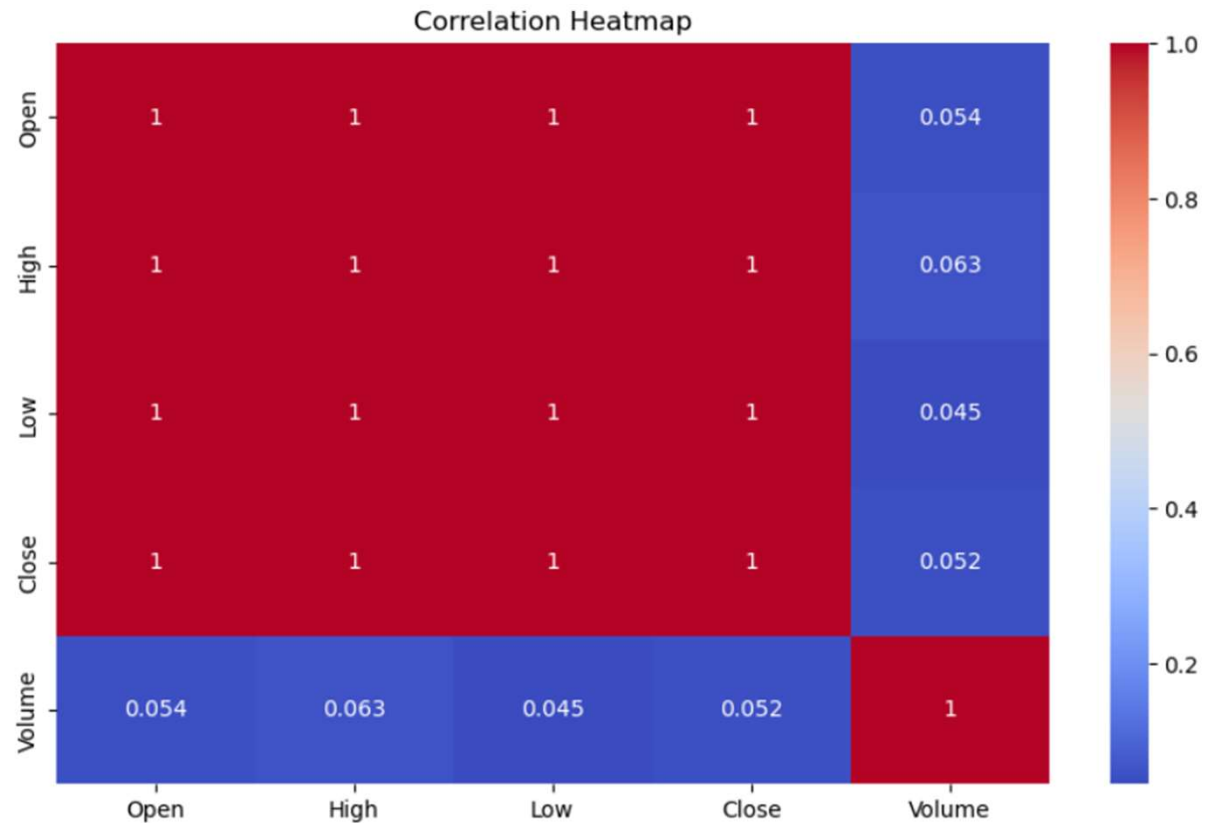
```
Out[17]: <function matplotlib.pyplot.show(close=None, block=None)>
```



Analyze, cont.

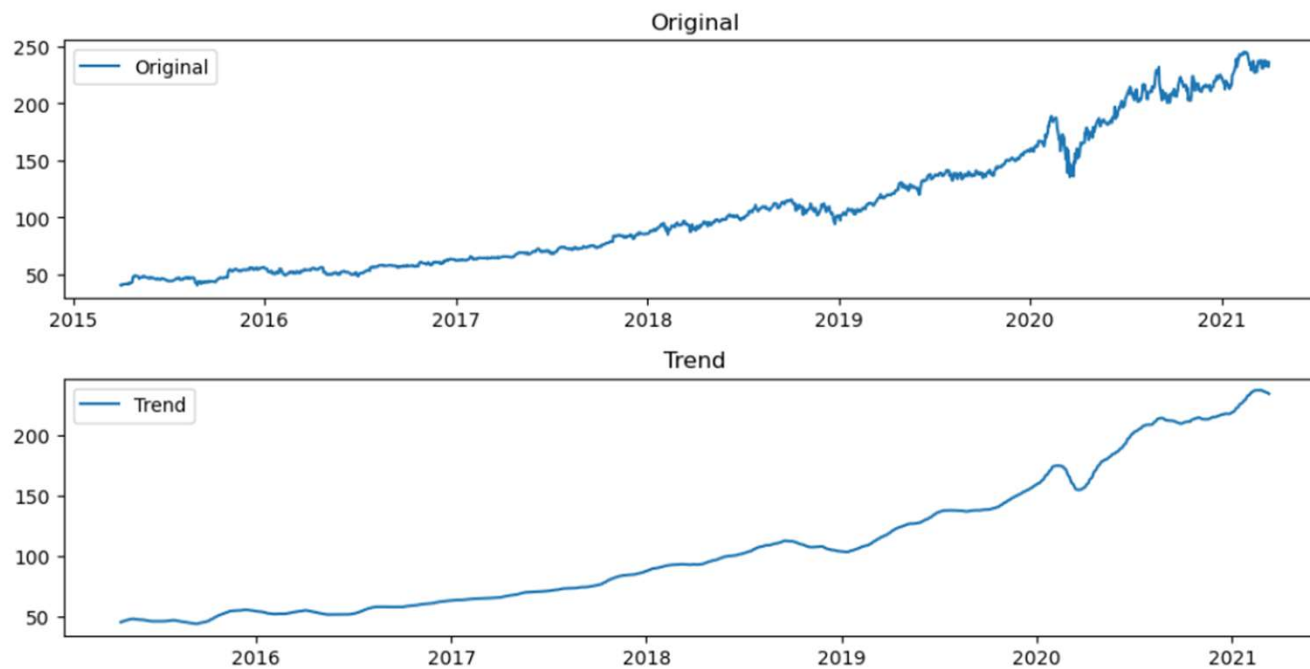
Correlation Heatmap

- There is a perfect correlation between the opening, high, low, and closing prices of MSFT.
- Between these prices and the volume traded, the correlation is much smaller.
- The correlation between trading volume and closing stock price movements is $0.052 = 5.2\%$



Analyze, cont.

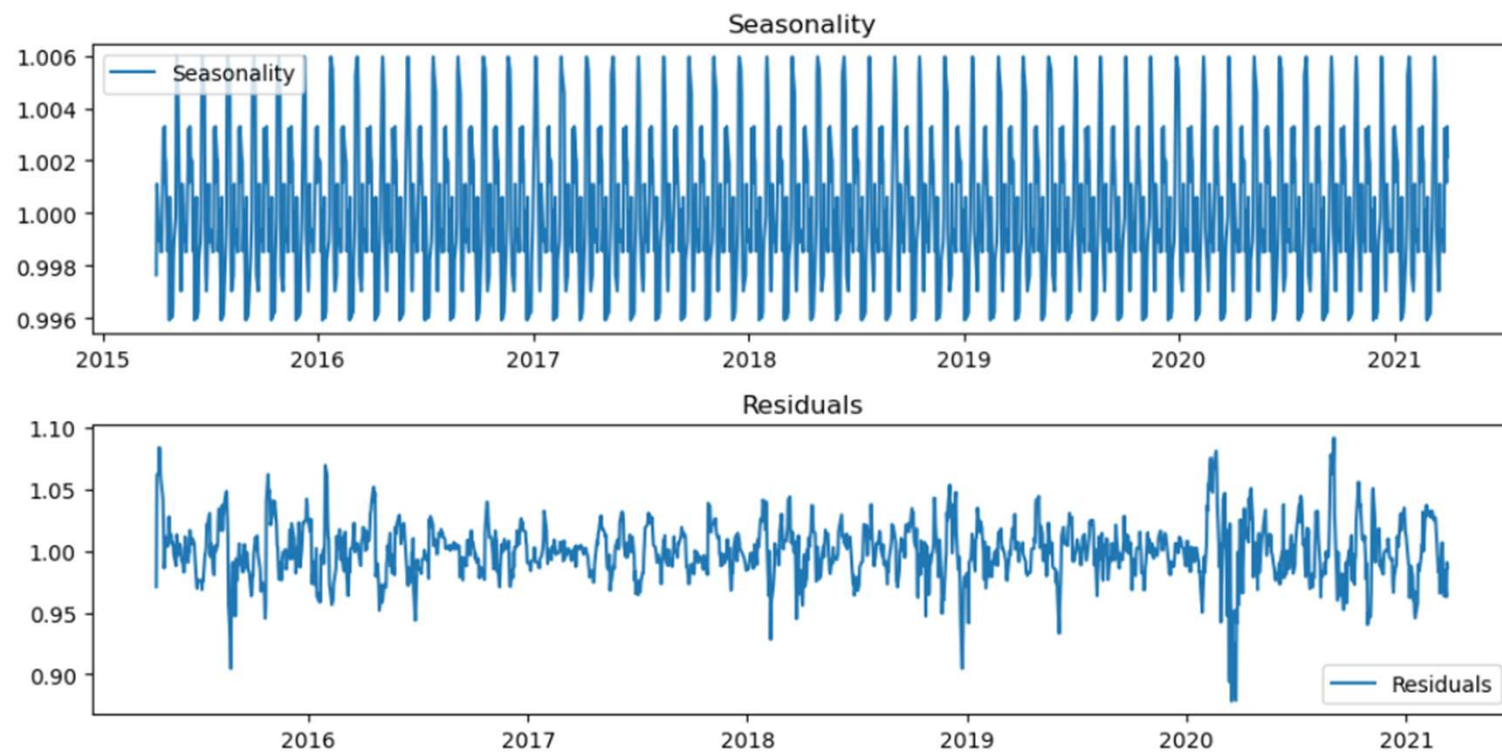
Seasonal Decomposition of the Time Series - Trend



Microsoft Stock Price Prediction by Magnus Aghe -
ANA500 Micro-Project

Analyze, cont.

Seasonal Decomposition of the Time Series – Seasonality and Residuals



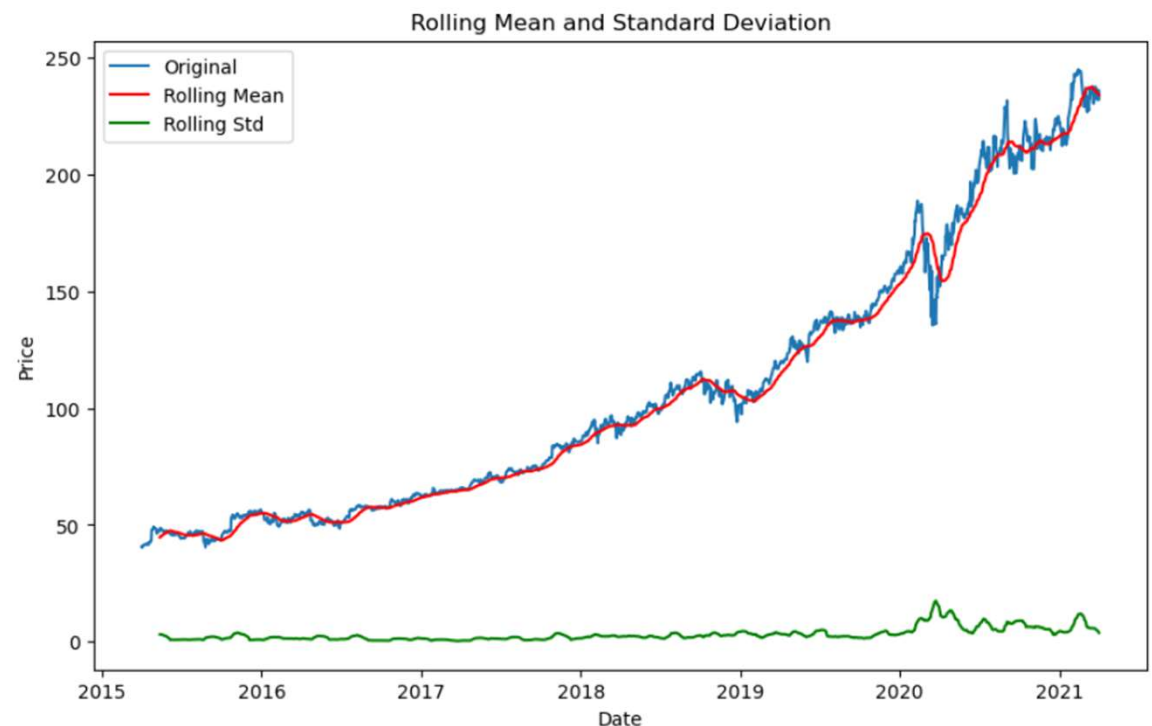
Analyze, cont.

Plotting the 30-day moving average (Rolling Mean) for MSFT

```
# 30-day moving average (Rolling mean) and standard deviation

rolling_mean = df['Close'].rolling(window=30).mean()
rolling_std = df['Close'].rolling(window=30).std()

plt.figure(figsize=(10, 6))
plt.plot(df['Close'], label='Original')
plt.plot(rolling_mean, label='Rolling Mean', color='r')
plt.plot(rolling_std, label='Rolling Std', color='g')
plt.legend(loc='best')
plt.title('Rolling Mean and Standard Deviation')
plt.xlabel('Date')
plt.ylabel('Price')
plt.show()
```

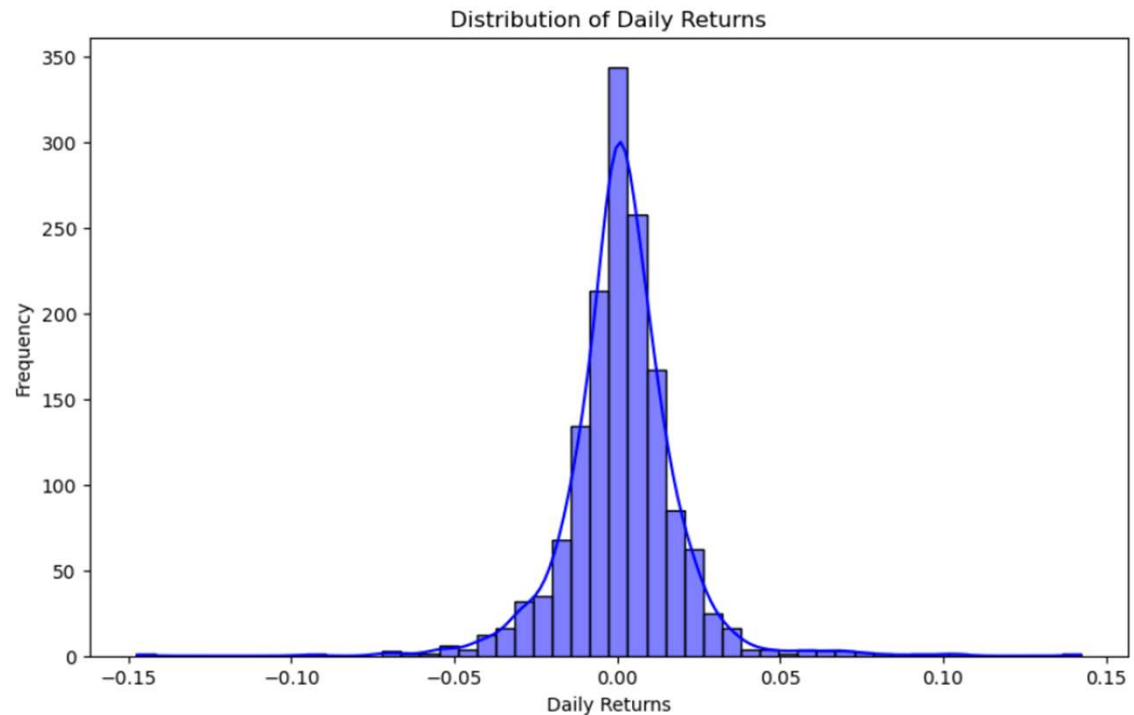


Analyze, cont.

Average Daily Return and Volatility (Standard Deviation)

Average Daily Return: 0.00132
Volatility: 0.01745

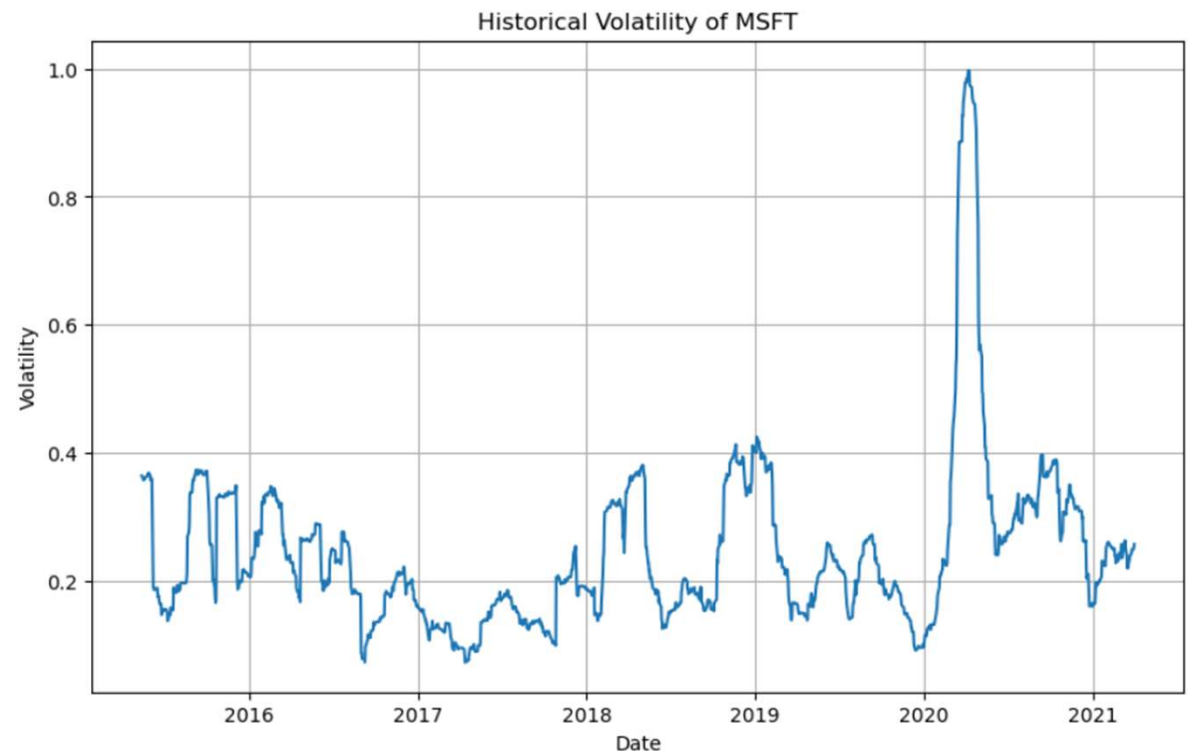
- The average daily return of MSFT stock price is 0.132%.
- The volatility of daily returns is at 1.75%.



Analyze, cont.

Historical Volatility of MSFT

- 2020/2021 witnessed unusual volatility compared to other years.

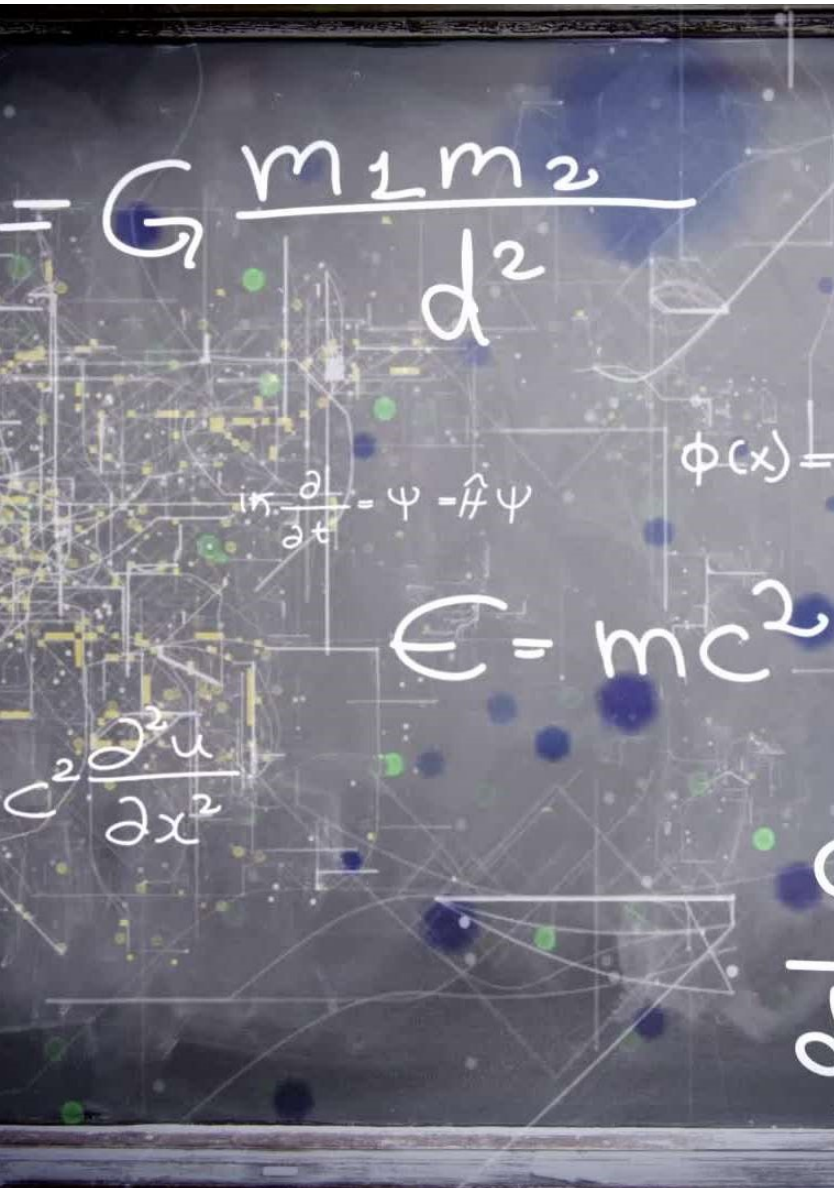


Report

- The year 2020/2021 witnessed undue volatility in the closing price of MSFT compared to other years under review.
- With an average daily return of 0.132%, and average volatility of 1.75%, it is easier to predict how much MSFT stocks can swing.
- There is an overall yearly upward trajectory of MSFT stock price in the year under review (2015 - 2021).
- The closing stock price is perfectly correlated with opening price, high and low daily prices.

Microsoft Stock Price Prediction by Magnus Aghe -
ANA500 Micro-Project





Report, cont.

- Multiple linear regression would not be suitable for predicting the closing MSFT price, due to high collinearity with the predictor variables.
- Other machine learning models (such as RNNs, LSTM, Autocorrelation and ARIMA) will be explored next for predictive modeling.



Start of Micro- Project 3

Micro-Project 3

Objectives

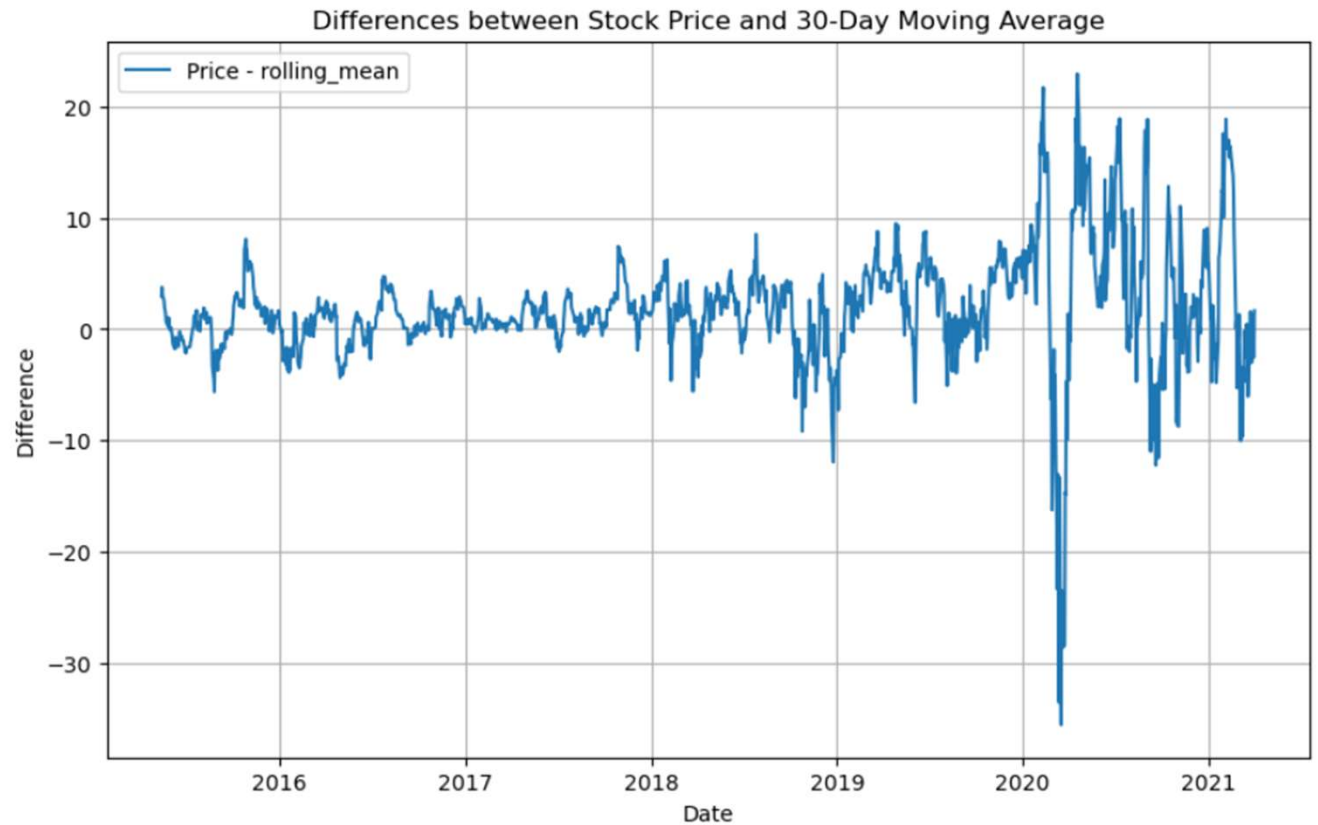
- To use **Moving Average** to predict the closing price of Microsoft stock.
- To use **Linear Regression** to predict the closing price of Microsoft stock using one predictor variable in the time series (Date).
- To use another machine learning regression technique - **Autoregressive Integrated Moving Average (ARIMA)** to predict the closing price of Microsoft stock.

Microsoft Stock Price Prediction by Magnus Aghe -
ANA500 Micro-Project



Analyze

- This reveals the difference between the daily closing price of Microsoft stock and the 30-day moving average from 2015 to 2021. The biggest differences occurred in 2020.



Analyze, cont.

Moving Average Predictive Model

With the moving average predictive model, the RMSE value is 69.38. As seen from the plot, the predicted data is away from the actual data.

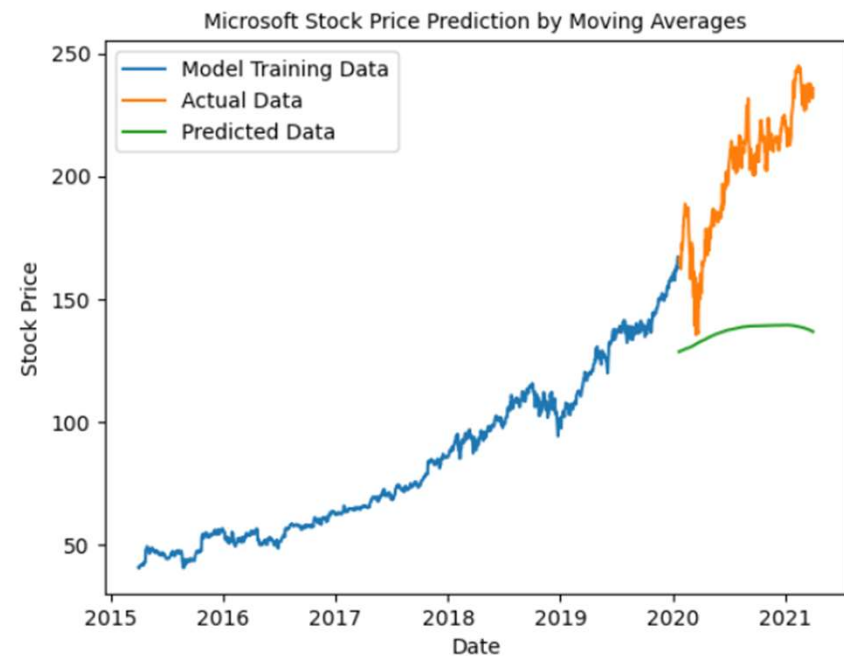
Training and Validation data is split in the ratio 80:20.

-----MICROSOFT STOCK PRICE PREDICTION BY MOVING AVERAGE-----

Shape of Training Set (1209, 1)

Shape of Validation Set (302, 1)

RMSE value on validation set: 69.38484924906464



Analyze, cont.

Linear Regression Predictive Model

Target variable = Close

Predictor variable = Date

Training and Validation data is split in the ratio 80:20.

```
def linear_regression_prediction(df):
    shape=df.shape[0]
    df_new=df[['Close']]
    df_new.head()
    train_set=df_new.iloc[:ceil(shape*0.80)]
    valid_set=df_new.iloc[ceil(shape*0.80):]
    print('-----')
    print('-----MICROSOFT STOCK PRICE PREDICTION BY LINEAR REGRESSION-----')
    print('-----')
    print('Shape of Training Set',train_set.shape)
    print('Shape of Validation Set',valid_set.shape)
    train=train_set.reset_index()
    valid=valid_set.reset_index()
    x_train = train['Date'].map(dt.datetime.toordinal)
    y_train = train[['Close']]
    x_valid = valid['Date'].map(dt.datetime.toordinal)
    y_valid = valid[['Close']]
    #implement linear regression
    model = LinearRegression()
    model.fit(np.array(x_train).reshape(-1,1),y_train)
    preds = model.predict(np.array(x_valid).reshape(-1,1))
    rms=np.sqrt(np.mean(np.power((np.array(valid_set['Close'])-preds),2)))
    print('RMSE value on validation set:',rms)
    print('-----')
    print('-----')
    valid_set['Predictions'] = preds
    plt.plot(train_set['Close'])
    plt.plot(valid_set[['Close', 'Predictions']])
    plt.xlabel('Date',size=10)
    plt.ylabel('Stock Price',size=10)
    plt.title('Microsoft Stock Price Prediction by Linear Regression',size=10)
    plt.legend(['Model Training Data','Actual Data','Predicted Data'])
```

```
linear_regression_prediction(df)
```

Analyze, cont.

Linear Regression Predictive Model

With linear regression, the RMSE value is 56.96.

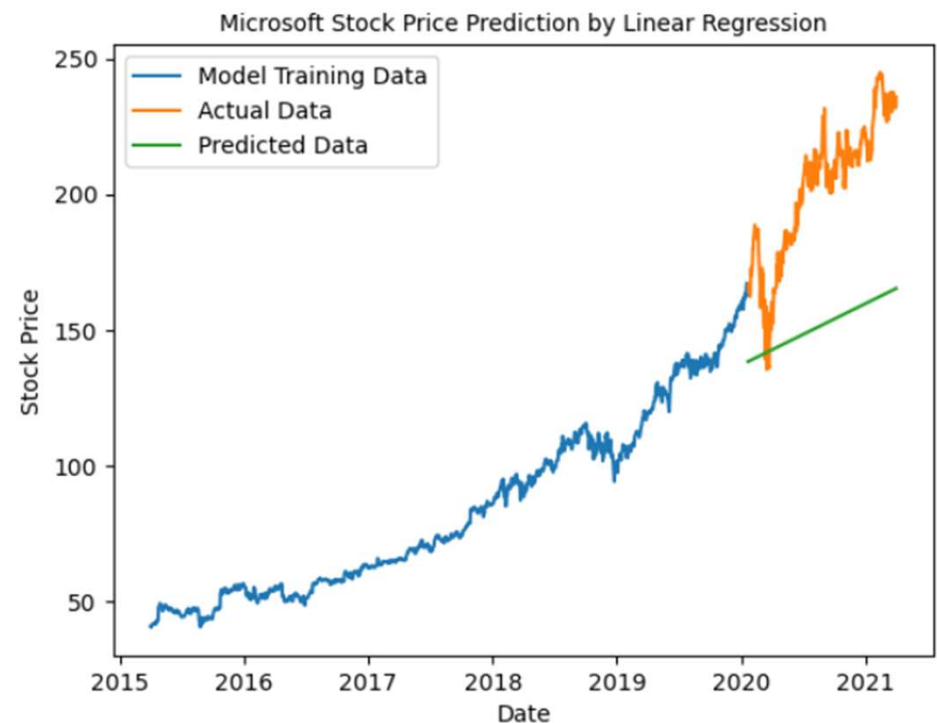
This is better than the moving average prediction in terms of accuracy and as seen from the plot, the predicted data is a bit closer to the actual data.

-----MICROSOFT STOCK PRICE PREDICTION BY LINEAR REGRESSION-----

Shape of Training Set (1209, 1)

Shape of Validation Set (302, 1)

RMSE value on validation set: 56.96175596453962





Analyze, cont.

- **ARIMA Predictive Model**

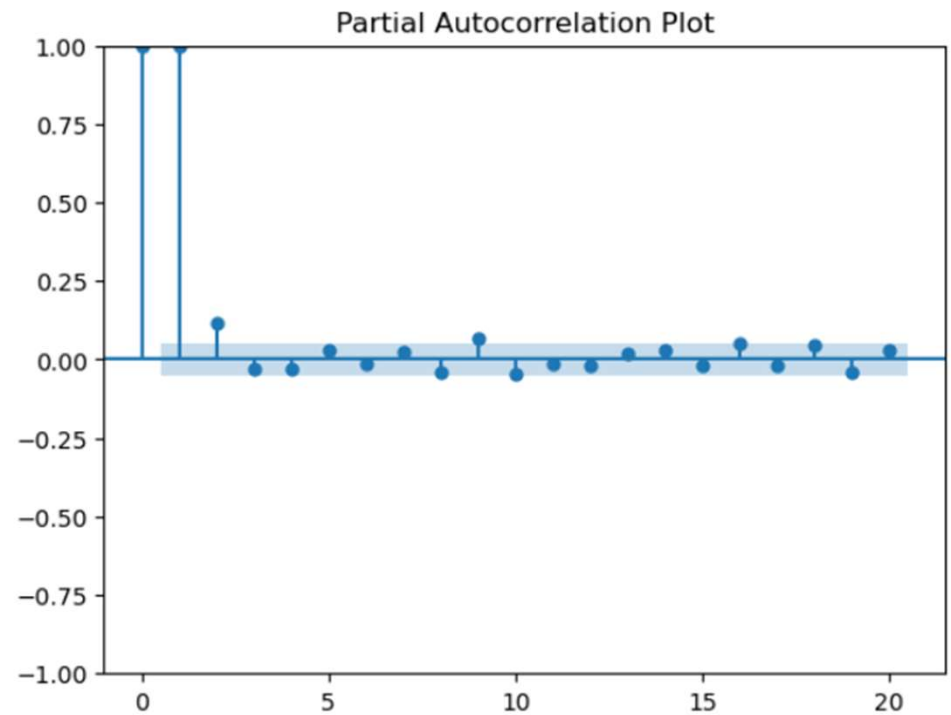
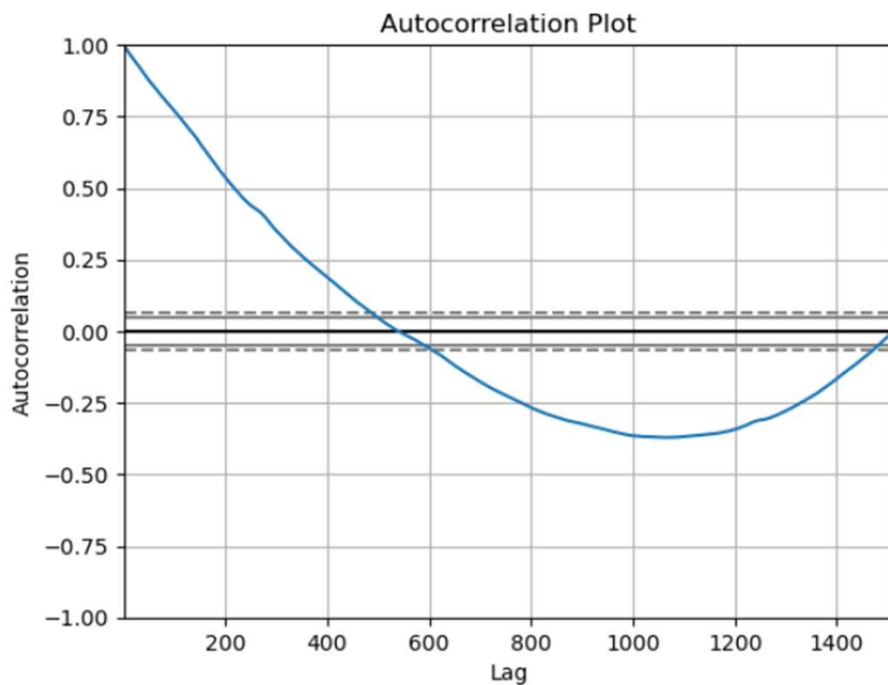
ARIMA is a popular method for time series forecasting. The models use past values to predict future values, and has three important parameters namely, p (past values used to forecast the next value), q (past forecast errors used to predict future values), and d (order of differencing).

Analyzing autocorrelation and partial autocorrelation plots to determine p , d , and q .

Analyze, cont.

• Autocorrelation

- This is the correlation between values of a time series in neighborhood periods. It describes a relationship between the series and itself.



Analyze, cont.

ARIMA Predictive Model

With ARIMA, for an arbitrary (p, d, q) order of (1, 2, 3), the RMSE value is 18.60.

This is much better than both linear regression models and the moving average predictive model.

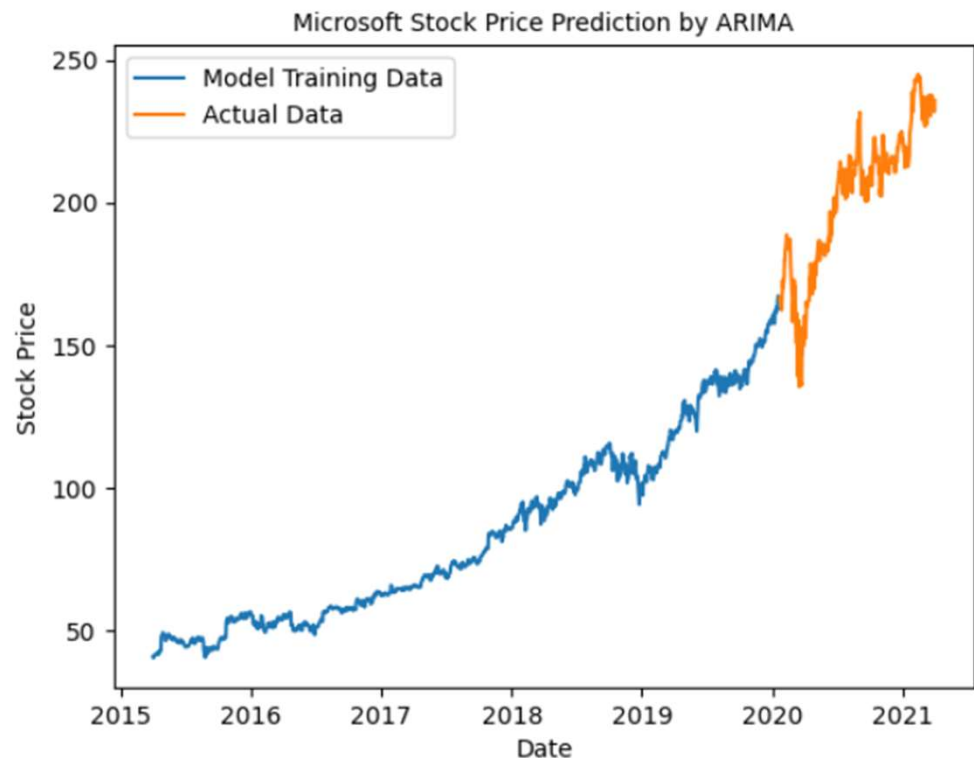
Training and Validation data is split in the ratio 80:20.

-----MICROSOFT STOCK PRICE PREDICTION BY ARIMA-----

Shape of Training Set (1209, 1)

Shape of Validation Set (302, 1)

RMSE value on validation set: 18.60488784602582



Analyze, cont.

ARIMA Predictive Model

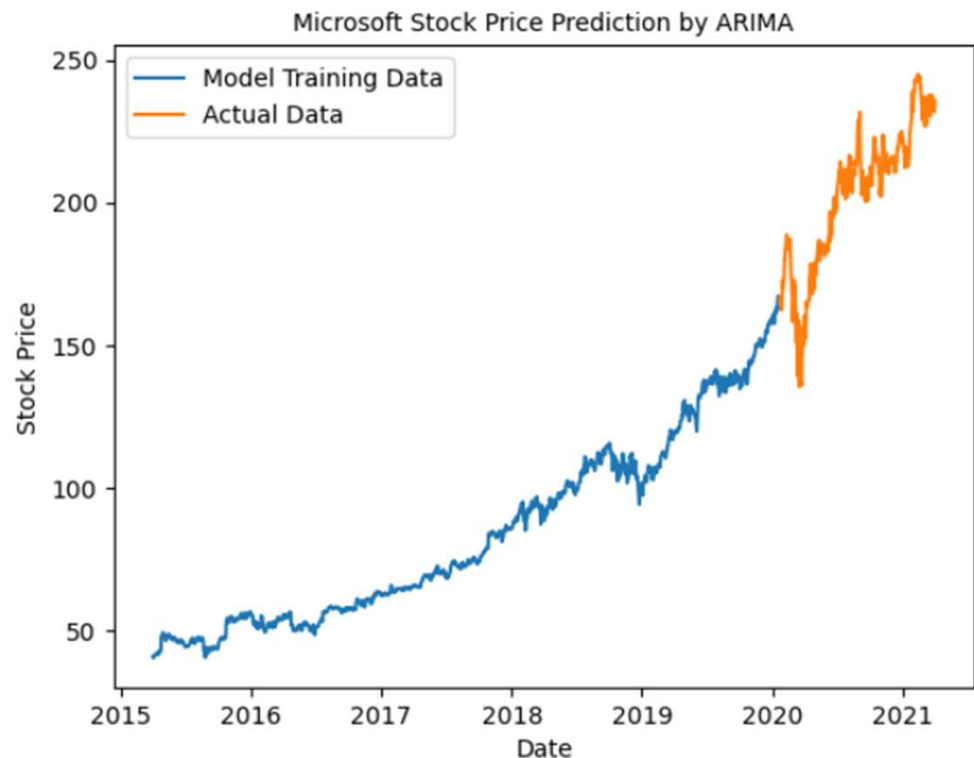
However, we can find the best ARIMA model, based on the order of p, d, q .

The best ARIMA model has an RMSE of 17.09.

The p, d, q order is (5, 2, 1)

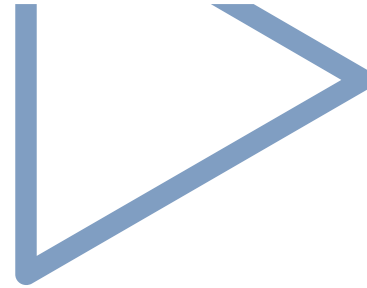
This is a better model than the ones we have explored so far.

Best ARIMA order: (5, 2, 1) with AIC: 3918.801366924544
RMSE: 17.09



Report

- Of the three machine learning techniques that we used to develop models to predict the price of Microsoft stock, ARIMA model was the best as it had the lowest root mean square error (RMSE) value of 17.09.
- This makes it suitable to use for prediction compared to Moving Average or Linear Regression.
- However, in the next micro-project, we shall explore Long Short Term Memory (LSTM) method of recurrent neural networks to see if predictive accuracy can be improved upon.



Act

- Our first objective was to know the features which had the greatest effect on the closing Microsoft stock price. The closing stock price was perfectly correlated with opening price, high and low daily prices (value = 1), and so all three had equal effect on the closing price.
- We therefore reject the null hypothesis (H_0) of no significant association between the closing price of Microsoft stock and the attributes, open, high, low, and volume.
- We decomposed the Microsoft stock price in terms of trend, seasonality and residuals and plotted the series.
- We measured volatility in terms of percentage change of daily returns in the price. The average daily return was 0.132% and volatility of daily returns was 1.75%.
- 2020/2021 witnessed unusual volatility compared to the other years in the series.
- Our best model thus far, is the ARIMA model for predicting the price of Microsoft stock. It had the least RMSE (17.09) and would be deployed for future forecast, unless there is a better model. We shall explore the LSTM model to see if it would give us better prediction.

Microsoft Stock Price Prediction by Magnus Aghe -
ANA500 Micro-Project





Start of *Micro-Project* 4

Objective

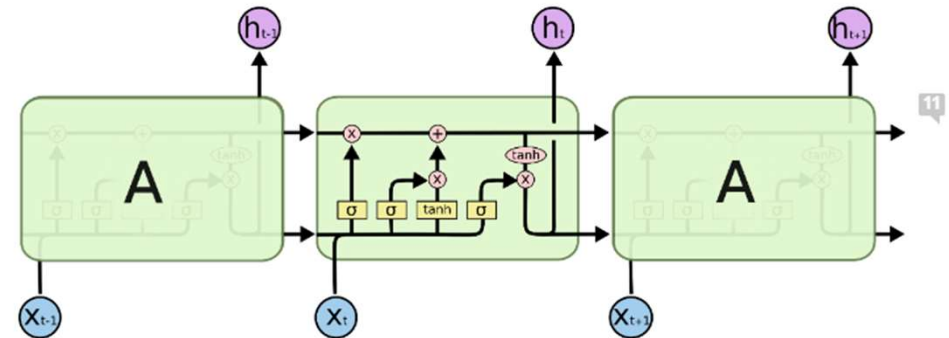
- To use deep learning regression methods like recurrent neural networks (RNN) implemented with Long Short-Term Memory (LSTM) building blocks for forecasting the stock price.



Analyze

Long Short-Term Memory (LSTM)

- LSTM is a popular recurrent neural network (RNN) infrastructure which addresses the vanishing gradient problem of RNNs.
- It has cells which has three (3) gates namely, the input gate, the output gate, and the forget gate.
- These gates control the flow of information which is needed to predict the output in the network.



The repeating module in an LSTM contains four interacting layers.

Analyze, cont.

Long Short-Term Memory (LSTM)

- The input gate adds information to the cell state.
- The forget gate removes the information that is no longer required by the model.
- The output gate selects the information to be shown as output.

Using Neural Network's LSTM in TensorFlow to Predict Microsoft Stock Price

```
: import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
sns.set(rc={'figure.figsize':(16,8)})
sns.set(font_scale=1.3)
plt.style.use('fivethirtyeight')

from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, LSTM, RepeatVector, TimeDistributed
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau

from sklearn.metrics import mean_squared_error
import math
from math import floor,ceil,sqrt
import datetime as dt

import warnings
warnings.filterwarnings('ignore')

: url = 'https://raw.githubusercontent.com/magnusaghe/ANA500/main/Microsoft_Stock.csv'
df = pd.read_csv(url, parse_dates=[0])
```

- Implementing LSTM as a black-box:-

Analyze, cont.

Long Short-Term Memory (LSTM)

- The model is trained with 50 epochs and batch size of 32.
- Training and Validation data is split in the ratio 80:20.

```
-----MICROSOFT STOCK PRICE PREDICTION BY LONG SHORT TERM MEMORY (LSTM)-----
Shape of Training Set (1209, 1)
Shape of Validation Set (302, 1)
Epoch 1/50
37/37 - 8s - loss: 0.0058 - 8s/epoch - 219ms/step
Epoch 2/50
37/37 - 3s - loss: 1.5205e-04 - 3s/epoch - 80ms/step
Epoch 3/50
37/37 - 3s - loss: 1.1017e-04 - 3s/epoch - 88ms/step
Epoch 4/50
37/37 - 4s - loss: 1.0808e-04 - 4s/epoch - 111ms/step
Epoch 5/50
37/37 - 3s - loss: 1.0977e-04 - 3s/epoch - 81ms/step
```

Microsoft Stock Price Predictio
ANA500 Micro-I

```
def lstm_prediction(df):
    shape=df.shape[0]
    df_new=df[['Close']]
    df_new.head()
    dataset = df_new.values
    train=df_new[:ceil(shape*0.80)]
    valid=df_new[ceil(shape*0.80):]
    print('-----MICROSOFT STOCK PRICE PREDICTION BY LONG SHORT TERM MEMORY (LSTM)-----')
    print('Shape of Training Set',train.shape)
    print('Shape of Validation Set',valid.shape)
    scaler = MinMaxScaler(feature_range=(0, 1))
    scaled_data = scaler.fit_transform(dataset)
    x_train, y_train = [], []
    for i in range(40,len(train)):
        x_train.append(scaled_data[i-40:i,0])
        y_train.append(scaled_data[i,0])
    x_train, y_train = np.array(x_train), np.array(y_train)
    x_train = np.reshape(x_train, (x_train.shape[0],x_train.shape[1],1))
    model = Sequential()
    model.add(LSTM(units=128, return_sequences=True, input_shape=(x_train.shape[1],1)))
    model.add(LSTM(units=64))
    model.add(Dense(1))
    model.compile(loss='mean_squared_error', optimizer='adam')
    model.fit(x_train, y_train, epochs=50, batch_size=32, verbose=2)
    inputs = df_new[len(df_new) - len(valid) - 40:].values
    inputs = inputs.reshape(-1,1)
    inputs = scaler.transform(inputs)
    X_test = []
    for i in range(40,inputs.shape[0]):
        X_test.append(inputs[i-40:i,0])
    X_test = np.array(X_test)
    X_test = np.reshape(X_test, (X_test.shape[0],X_test.shape[1],1))
    closing_price = model.predict(X_test)
    closing_price = scaler.inverse_transform(closing_price)
    rms=np.sqrt(np.mean(np.power((valid-closing_price),2)))
    print('RMSE value on validation set:',rms)
    print('-----')
    valid['Predictions'] = closing_price
    plt.plot(train['Close'])
    plt.plot(valid[['Close','Predictions']])
    plt.xlabel('Date',size=15)
    plt.ylabel('Stock Price',size=15)
    plt.title('Microsoft Stock Price Prediction by Long Short Term Memory (LSTM)',size=15)
    plt.legend(['Model Training Data','Actual Data','Predicted Data'])
    plt.figure(figsize=(10, 6))
```

lstm_prediction(df)

Analyze, cont.

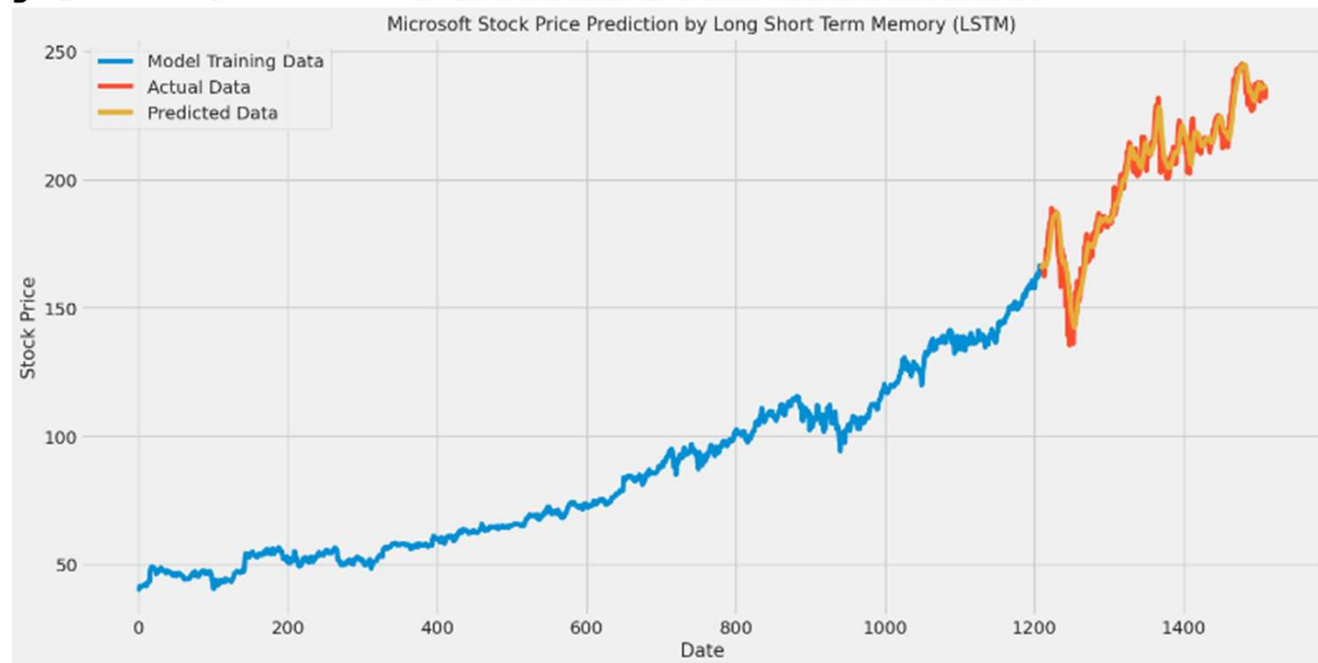
Long Short-Term Memory (LSTM)

With LSTM, the RMSE value is 5.69

This is so far the best model!

We can see the predicted data (in orange) fitting well on the actual data (in red).

```
Epoch 47/50
37/37 - 4s - loss: 8.4196e-05 - 4s/epoch - 114ms/step
Epoch 48/50
37/37 - 3s - loss: 8.2170e-05 - 3s/epoch - 81ms/step
Epoch 49/50
37/37 - 3s - loss: 9.0878e-05 - 3s/epoch - 78ms/step
Epoch 50/50
37/37 - 3s - loss: 7.3390e-05 - 3s/epoch - 78ms/step
10/10 [=====] - 2s 54ms/step
RMSE value on validation set: Close 5.692778
dtype: float64
```



<Figure size 1000x600 with 0 Axes>

Report

- Of all predictive machine learning techniques that we used to develop models to predict the price of Microsoft stock, **LSTM is the best model as it has the least root mean square error (RMSE) value of 5.69.**
- The next best was ARIMA model with an optimal RMSE of 17.09.
- Linear regression had an RMSE of 56.96; and...
- Moving Average had the least RMSE of 69.38.



Act

- Our first objective was to know the features which had the greatest effect on the closing Microsoft stock price. The closing stock price was perfectly correlated with opening price, high and low daily prices (value = 1), and so all three had equal effect on the closing price.
- We therefore reject the null hypothesis (H_0) of no significant association between the closing price of Microsoft stock and the attributes, open, high, low, and volume.
- We decomposed the Microsoft stock price in terms of trend, seasonality and residuals and plotted the series.
- We measured volatility in terms of percentage change of daily returns in the price. The average daily return was 0.132% and volatility of daily returns was 1.75%.
- 2020/2021 witnessed unusual volatility compared to the other years in the series.
- Our overall best model is the LSTM model for predicting the price of Microsoft stock. It had the least RMSE (5.69) and would be deployed.
- It is important to note that other external factors such as media reports of Microsoft's performance, and general news on the economy can add to speculative effects which can also contribute to the volatility of the stock.

Microsoft Stock Price Prediction by Magnus Aghe -
ANA500 Micro-Project





THANK YOU!