



Delhi Technological University

M&S CO-207 Project

on

Stock Market Simulator

Stock Market Simulator

by

Name	Roll no.	Email ID
Siddhant Bikram Shah	2K19/CO/374	siddhantbikramshah_2k19co374@dtu.ac.in
Rahul Anand	2K19/CO/303	rahulanand_2k19co303@dtu.ac.in

A comprehensive project report has been submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology

in

Computer Engineering

Under the supervision of and submitted to

Ms. Madhuri Yadav

Delhi Technological University, formerly

Delhi College of Engineering

Department of Computer Science Engineering

Delhi Technological University, Shahbad Daultpur, Main Bawana

Road , Delhi-110042, India

Declaration



"We Do hereby declare that this submission is our own work conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute and that, to the best of our knowledge and belief, it contains no material previously written by another neither person nor material (data, theoretical analysis, figures, and text) which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text."

Siddhant Bikram Shah

2k19/CO/374

Rahul Anand

2k19/CO/303

Acknowledgement

I would like to express my special thanks and gratitude to my teacher Ms. Madhuri Yadav as well as our college **Delhi Technological University(DTU)** which gave me the golden opportunity to do this wonderful project on the topic “Stock Market Simulator”, which also helped me in doing a lot of research and discovering many new things on the subject matter.

Secondly, I would also like to thank my parents, friends and classmates who helped me a lot in finalizing this project within the limited time frame which had been given to us. I am really thankful to all of them.

My special thanks goes to my university i.e, Delhi Technological University for providing me such a great platform and such good teachers for overlooking my project and its progress. Having such a great curriculum which helps us to improve our learning skill as well as presenting our work in form of report, such as this kind of project work will definitely go a long way in my quest to improve my skills.

Abstract



“How can I utilize my money in a smart way so that my future is safe?” Most wise people in the world would answer that question in a similar way- Invest in the stock market.

In this era of technology and easily available knowledge, we can effortlessly get informed about the stock market from the comfort of our own homes.

The program offered below can help us predict and simulate stock market trends by looking at previous stock tendencies.

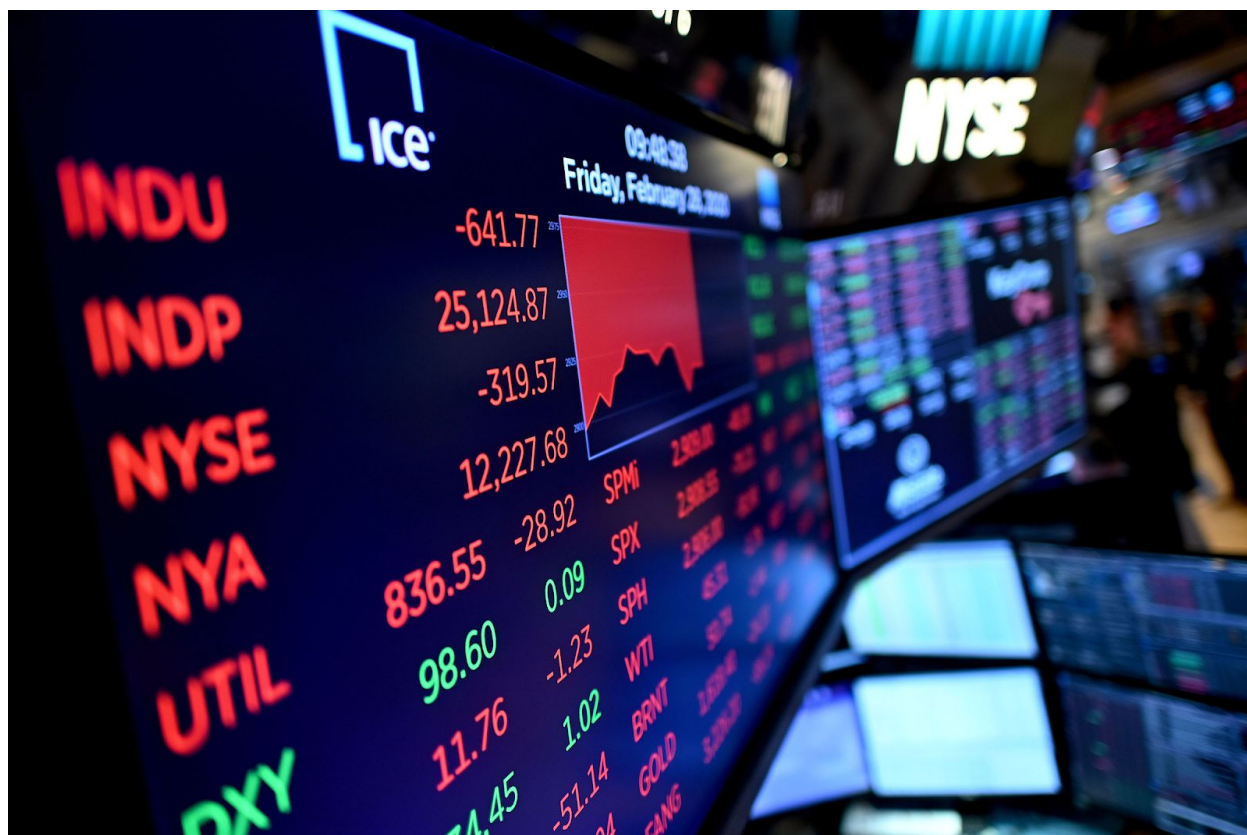
This project consists of two parts:

1. Building a stock market screener
2. Building a stock market simulator

Table of contents


1. Introduction.....	6
2. Description.....	8
3. Programs.....	13
4. Code.....	23
5. Conclusion.....	28
6. References.....	29

Introduction



A stock or share represents ownership of a person in a company and a proportionate claim on its assets and earnings. In today's world, there are thousands of companies with millions of shares to their name, which makes it a good idea for any person to contemplate investing in the stock market. In fact, the richest men in the world have reached that position through investments in the stock market and still invest the majority of their wealth.

The mere thought of investing in the stock market is enough to frighten anyone. This can be attributed to the numerous public cases of stockholders netting a loss on their investments. However, the truth remains that even though it carries a risk,



investing in the stock market is one of the most efficient ways to build up one's net worth.

As it is such a risky ordeal, investing in the stock market must be done in a disciplined manner, with proper planning and forethought. That is where the concept of **Modelling** and **Simulation** comes into play.

Simulation is the process of imitating a real-world process in a controlled environment. It represents the operation of the system over time.

A **Model** is a representative of an object or system and it explains phenomena that might not be easy to experience directly. We use modelling to figure out the result of a simulation.

Simulation can help us to foresee or at least understand the stock market trends and make a prediction on what turns the stock market will take.

With the help of simulation and modelling, we aim to

1. Build a Stock Market Screener that plots stock market tendencies and inclinations on a graph.
2. Build a Stock Market Simulator that predicts and simulates stock market trends by looking at previous stock tendencies.

Description

History and Background

Stock markets were started when countries in the New World began trading with each other. To most people, the name Wall Street is synonymous with stock exchange. The market on Wall Street opened May 17, 1792 on the corner of Wall Street and Broadway. Twenty-four supply brokers signed the Buttonwood Agreement outside 68 Wall St. in New York, underneath a buttonwood tree. On March 8, 1817 the group renamed itself the New York Stock and Exchange Board and moved off the street into 40 Wall St. The organization that would define the world's economic future was born.

Today, there are many stock exchanges worldwide, each supplying the capital necessary to support industry growth. Without these vital funds, many revolutionary ideas would never become a reality, nor would fundamental improvements be made to existing products. In addition, the stock market creates personal wealth and financial stability through private investment, allowing individuals to fund their retirement and or other ventures.

What is the Stock Market?

Stock markets are where individual and institutional investors come together to buy and sell shares in a public venue. Nowadays these exchanges exist as electronic marketplaces. Share prices are set by supply and demand in the market as buyers and sellers place orders.

The primary purpose of a stock market is to regulate the exchange of stocks, as well as other financial assets. Such regulation ensures a fair environment for not only investors, but also the corporations whose stocks are traded in the market.

The following stock market terms are used in this project-

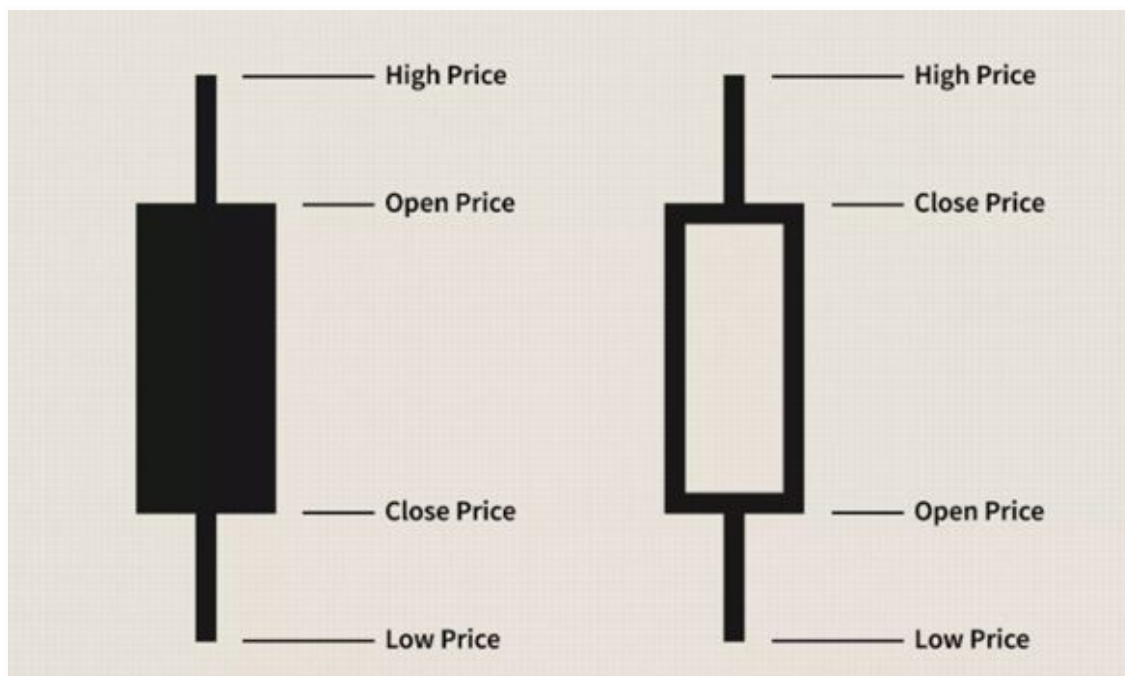
1. High price(maximum price in a given time period)
2. Low price(minimum price in a given time period)
3. Open(Price at which a stock began trading in a given time period)
4. Close(Price at which a stock ended trading in a given time period)
5. Volume(Total amount of trading activity)
6. Adjusted values(which factor in corporate actions such as dividends, stock splits, and new share issuance)

What are Stock Market Screeners?

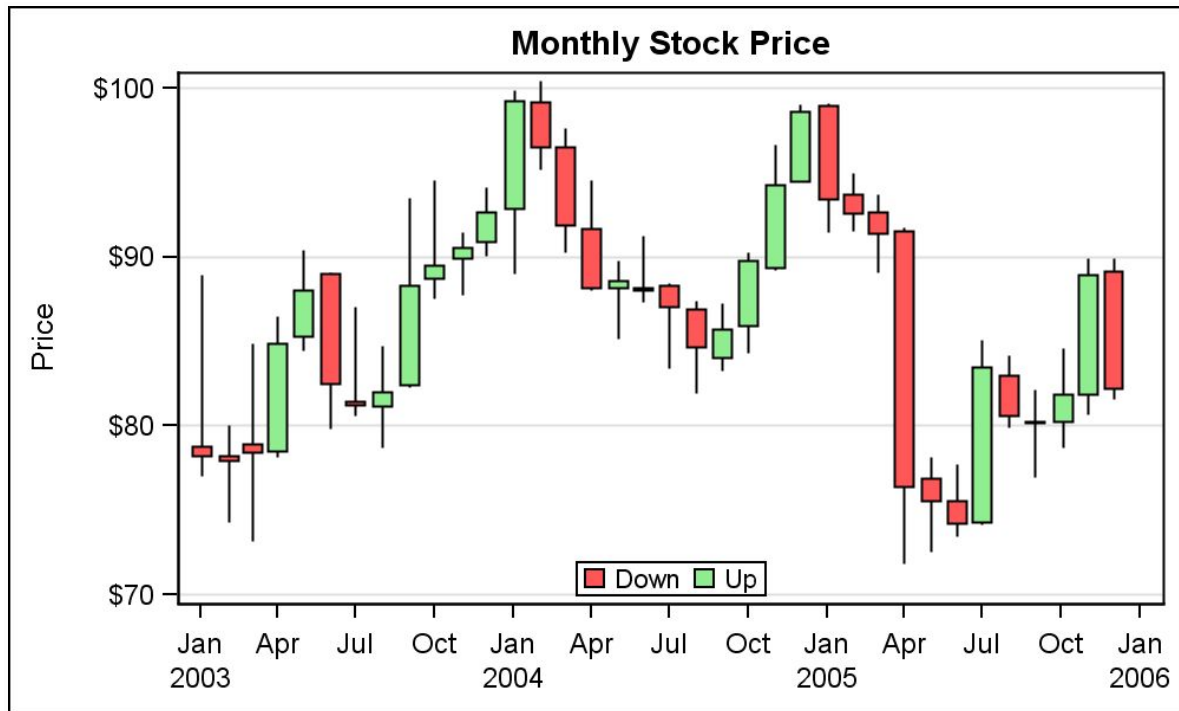
Stock Market Screeners help us to visualize the prices related to the stock market so that we can easily compare them to prices in the past, see the uptrend or downtrend, etc. which would help us decide what action is the best and safest at the right time.

In our Stock Market Screener, we will be using a CandleStick OHLC graph to display information. This style of graph allows us to display OHLC- Open, High, Low and Close prices for that particular time period at once, compared to traditional graphs, which would be congested if more than one type of data were included.

The following diagram explains how to interpret a candlestick graph:



Here, we have a sample stock market screen:



Green indicates the stock is trading higher than the previous day's close.

Red indicates the stock is trading lower than the previous day's close.

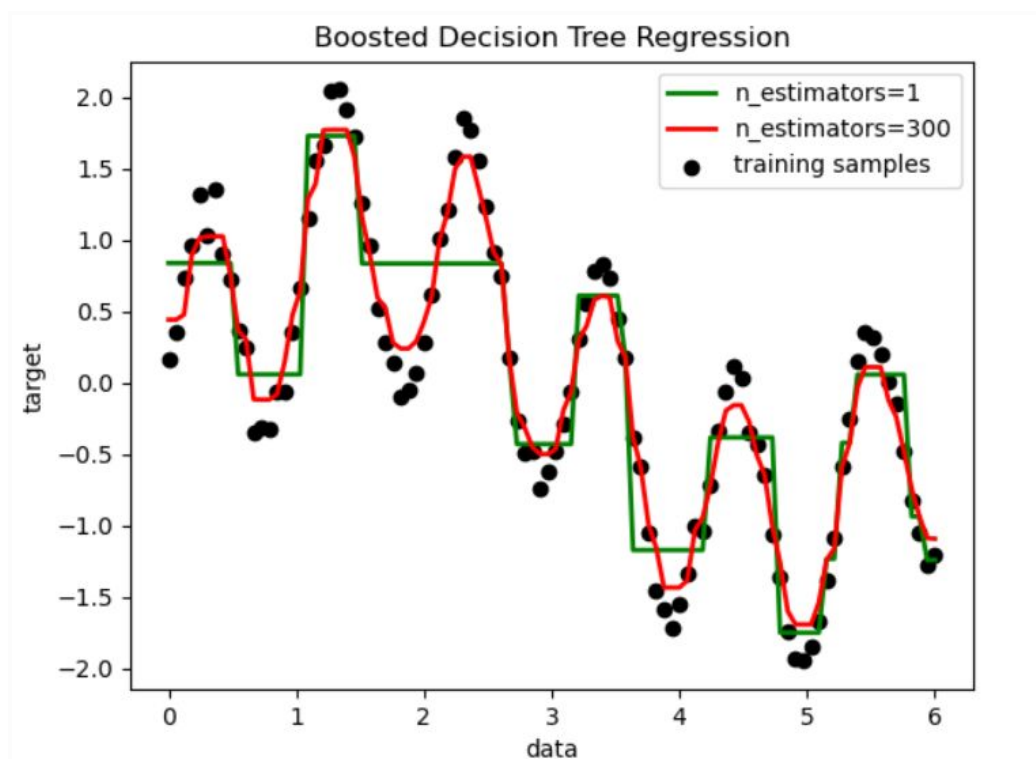
These graphs are condensed with data that we can interpret at a glance if we get acclimated with reading stock market screens.

How can we predict future stock market trends?

We predict future stock market trends using **Decision Tree Regression**.

The decision tree is the simplest, yet the most powerful algorithm in **machine learning**. Decision tree uses a flow chart like tree structure to **predict** the output on the basis of input or situation described by a set of properties.

Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output. Continuous output means that the output/result is not discrete, i.e., it is not represented just by a discrete, known set of numbers or values.



Programs

This project has two main goals:

1. To build a Stock Market Screener that plots stock market tendencies and inclinations on a graph.
2. To build a Stock Market Simulator that predicts and simulates stock market trends by looking at previous stock tendencies.

Specifications

We have implemented the following tools in our project:

1. **Python**- This is a simple and robust programming language that is easy to maintain and develop, which is why it was chosen.
2. **PyCharm**- A Python IDE that is simple, quick, efficient and also comes with Matplotlib integration and support.
3. **Matplotlib**- Matplotlib is a plotting library for Python. It provides an Object Oriented API for embedding plots into applications.

Some of the modules used in this project are:

1. **datetime**- It allows us to use date and time functions in our program, which is used for fetching and displaying data in certain timeframes, which is an essential component of a stock market screener. For now, we have used the timeframe of 1st January 2020 to 11th November 2020, just for reference.

2. **matplotlib**- It is a plotting library for Python which provides an Object Oriented API for embedding plots into applications. Basically, it helps us to visualize and plot the values and prices of stocks that we have fetched from the internet into the graph.
3. **pandas-datareader**- Fetches large dataframes with historical stock data(here, stocks from Apple have been fetched by using Apple's stock name 'AAPL') from Yahoo! Finance which can be later plotted on a graph by using matplotlib.
4. **numpy**- NumPy, short for Numerical Python is a python library used for working with arrays. It also has functions for working in the domain of linear algebra, fourier transform, and matrices.
5. **sklearn**- Scikit-Learn is a module that has functions for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python.
6. **mpl_finance**- Matplotlib finance is a module that contains various utility functions for the visualization of financial data.

To accomplish the goals of this project, we have prepared two separate programs which serve different purposes.

We have also added more functionalities and salient features to the programs since our last progress report.

Program 1- Stock Market Screener

1. Defining start and end dates

Since we have imported the datetime module, defining the start and end dates we intend to use on is simple.

For this program, we have chosen to focus on the timeframe of 1st January 2020 to 11th November 2020, but changing this date to any day in the past is very simple.

In our IDE, it looks like this:

```
11  
12     start=period.datetime(2020,1,1)  
13     end=period.datetime(2020,11,11)  
14
```

2. Fetching stock data from the internet

By using pandas_datareader, we can fetch historical stock data for various companies in the form of dataframes (tables with Open, Close, High, Low etc. prices listed according to their respective dates).

We have chosen to use the stock history of Apple(AAPL) and to use Yahoo! Finance(yahoo) as the data source.

```
13  
14     dataframe=net.DataReader('AAPL','yahoo',start,end)  
15
```


3. Resampling daily data into weekly intervals

For simplicity and to avoid clustering, we have chosen to display and store the information we fetched from the internet as weekly data.

```
16 dataframe_ohlc=dataframe['Adj Close'].resample('7D').ohlc()  
17 dataframe_volume=dataframe['Volume'].resample('7D').sum()  
18 print(dataframe_ohlc.head())
```

Then, the data frame is displayed in the output screen for us to easily reference.

	open	high	low	close
Date				
2019-12-31	72.192863	73.840042	72.192863	73.704819
2020-01-07	73.358185	77.923538	73.358185	77.923538
2020-01-14	76.871323	78.358696	76.541885	78.358696
2020-01-21	77.827667	78.481621	75.954315	75.954315
2020-01-28	78.103012	79.737900	75.883018	75.883018

4. Plotting the data and displaying the graph

Now that we have fetched and resampled the data from the internet, we hand over the dataframe to matplotlib, which plots and visualizes the data.

```
23 ax1=mpl.subplot2grid((6,1),(0,0),rowspan=5,colspan=1)
24 ax1.xaxis_date()
25
26 candlestick_ohlc(ax1,dataframe_ohlc.values,width=2,colorup='g')
27
28 mpl.show()
```

Finally, we get a graph that looks like this:



The condensed weekly stock OHLC data is displayed neatly and clearly in Candlestick form in the graph.

Program 2- Stock Market Simulator

1. Defining start and end dates

Similar to the last program, we have chosen to focus on the timeframe of 1st January 2020 to 11th November 2020.

In our IDE, it looks like this:

```
11
12     start=period.datetime(2020,1,1)
13     end=period.datetime(2020,11,11)
14
```

2. Fetching stock data from the internet

Similar to the last program, we have chosen to use the stock history of Apple(AAPL) and to use Yahoo! Finance(yahoo) as the data source.

```
13
14     dataframe=net.DataReader('AAPL','yahoo',start,end)
15
```

3. Create a new column in dataframe

We create a new column 'Prediction' in our dataframe, which takes input from the pre-existing 'Close' column, just shifted 60 days or 2 months prior to the end date.

```
17     days=60
18     dataframe['Prediction']=dataframe[['Close']].shift(-days)
19     print(dataframe.head())
20     print()
```

The output dataframe looks like this:

Date	High	Low	Open	...	Volume	Adj Close	Prediction
2019-12-31	73.419998	72.379997	72.482498	...	100805600.0	72.192863	61.935001
2020-01-02	75.150002	73.797501	74.059998	...	135480400.0	73.840042	63.702499
2020-01-03	75.144997	74.125000	74.287498	...	146322800.0	73.122154	63.572498
2020-01-06	74.989998	73.187500	73.447502	...	118387200.0	73.704819	60.227501
2020-01-07	75.224998	74.370003	74.959999	...	108872000.0	73.358185	61.232498

4. Split our prediction column into train and test into our X_train, X_test, Y_train and Y_test variables

We split our prediction column(in variables X and Y) using a train and test method, where the size of test is 25% and size of train is 75% of the whole dataframe.

We train the model using the training set and we test the model using the testing set. So, the train part of the dataframe is fed into the DecisionTreeRegressor() function.

```

30 X_train, X_test, Y_train, Y_test=train_test_split(X,Y,test_size=0.25)
31
32 tree=DecisionTreeRegressor().fit(X_train,Y_train)

```

5. Predict stock market trends for the last 60 days

Now, we store the prediction column of the last 60 days of the dataframe into our X_future variable. Then, we feed this X_future variable to our tree.predict() function as a parameter, which makes our final predictions based on sklearn's decision tree regression model.

Then, the final prediction is stored in the result variable.

```
34 X_future=dataframe.drop(['Prediction'],1)[:days]
35 X_future=X_future.tail(days)
36 X_future=np.array(X_future)
37 print(X_future)
38 print()
39
40 tree_prediction=tree.predict(X_future)
41 print(tree_prediction)
42 print()
43
44 result=tree_prediction
```

6. Plotting our final predictions on the graph

We create a variable `vdata` to store the actual value of stock prices on that day.

Then, we use `matplotlib` to plot the actual values of stock as well as our predicted values for that particular day on a graph, so we can compare how our predictions hold up to the actual values of the stock.

```
46 vdata=dataframe[X.shape[0]:]
47 vdata['Prediction']=result
48
49 mlp.title('Apple')
50 mlp.ylabel('Close Price in USD($)')
51 mlp.plot(dataframe['Close'])
52 mlp.plot(vdata[['Close', 'Prediction']])
53 mlp.legend(['Original Prices', 'Actual Prices', 'Predicted Prices'])
54 mlp.show()
```

We get an output like this:

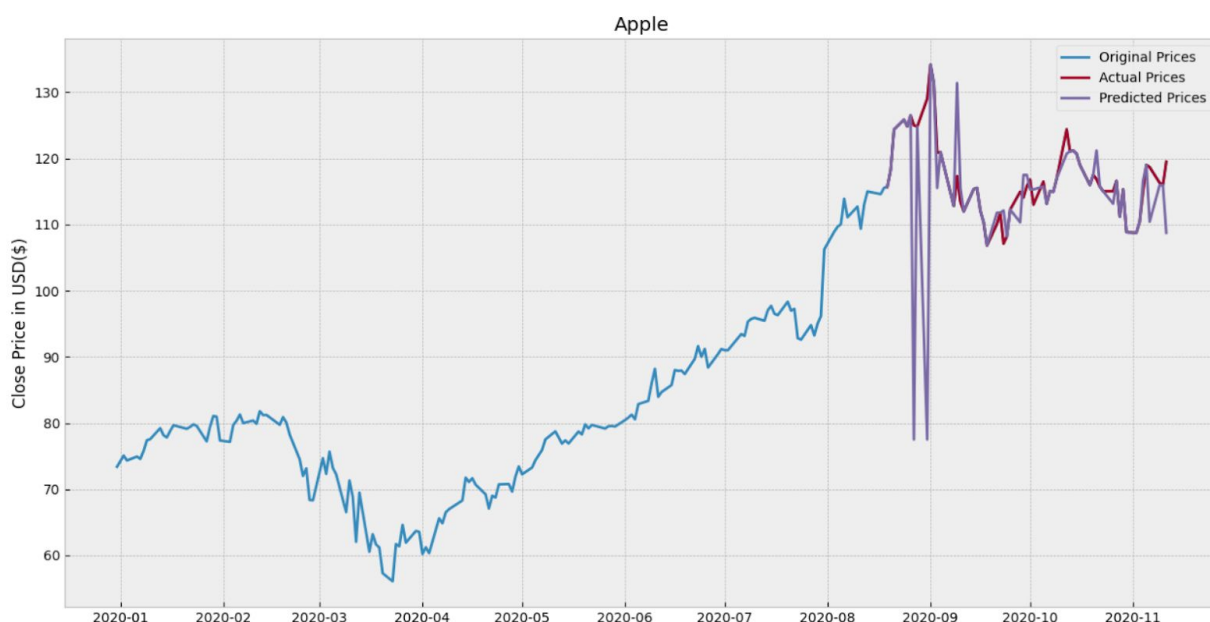


The **purple** line represents our predicted prices.

The **red** line represents the actual prices for that day.

The **blue** line represents the train values of the dataframe which we have used to train our Decision Tree Regression predictor function.

In the above example, it is clear that our program is able to predict stock market trends relatively accurately.



However, sometimes the program fails to correctly predict the stock market trends on that day. We can see that the purple line and red line are far apart from each other.

This is because our Decision Tree Progression predictor function has not been able to make accurate decisions based on past stock market tendencies.

This is just a testament to the volatility of the stock market, how stock trends can change regularly and how hard it is to predict stock price tendencies because these changes are brought on by real people performing stock transactions in real time.

Code

Program 1- Stock Market Screener

```
import datetime as period
import matplotlib.pyplot as mpl
from matplotlib import style
from mpl_finance import candlestick_ohlc
import matplotlib.dates as mplperiod
import pandas as pd
import pandas_datareader.data as net
style.use('bmh')

start=period.datetime(2020,1,1)
end=period.datetime(2020,11,11)

dataframe=net.DataReader('AAPL','yahoo',start,end)

dataframe_ohlc=dataframe['Adj Close'].resample('7D').ohlc()
dataframe_volume=dataframe['Volume'].resample('7D').sum()
print(dataframe_ohlc.head())
dataframe_ohlc.reset_index(inplace=True)
dataframe_ohlc['Date']=dataframe_ohlc['Date'].map(mplperiod.date2num)

ax1=mpl.subplot2grid((6,1),(0,0),rowspan=5,colspan=1)
```



```
ax1.xaxis_date()
```

```
candlestick_ohlc(ax1,dataframe_ohlc.values,width=2,colorup='g'
```

```
mpl.show())
```

Output

	open	high	low	close
Date				
2019-12-31	72.192863	73.840042	72.192863	73.704819
2020-01-07	73.358185	77.923538	73.358185	77.923538
2020-01-14	76.871323	78.358696	76.541885	78.358696
2020-01-21	77.827667	78.481621	75.954315	75.954315
2020-01-28	78.103012	79.737900	75.883018	75.883018



Program 2- Stock Market Simulator

```
import datetime as period
from matplotlib import style
import pandas_datareader.data as net
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as mlp

style.use('bmh')

start=period.datetime(2020,1,1)
end=period.datetime(2020,11,11)

dataframe=net.DataReader('AAPL','yahoo',start,end)

days=60
dataframe['Prediction']=dataframe[['Close']].shift(-days)
print(dataframe.head())
print()

ndf=dataframe['Close']
print(ndf.head())
print()

X=np.array(dataframe.drop(['Prediction'],1))[:-days]
```

```
Y=np.array(dataframe['Prediction'])[:-days]

X_train, X_test, Y_train, Y_test=train_test_split(X,Y,test_size=0.25)

tree=DecisionTreeRegressor().fit(X_train,Y_train)

X_future=dataframe.drop(['Prediction'],1)[:-days]
X_future=X_future.tail(days)
X_future=np.array(X_future)
print(X_future)
print()

tree_prediction=tree.predict(X_future)
print(tree_prediction)
print()

result=tree_prediction

vdata=dataframe[X.shape[0]:]
vdata['Prediction']=result

mlp.title('Apple')
mlp.ylabel('Close Price in USD($)')
mlp.plot(dataframe['Close'])
mlp.plot(vdata[['Close','Prediction']])
mlp.legend(['Original Prices','Actual Prices','Predicted Prices'])
mlp.show()
```

Output

	High	Low	Open	...	Volume	Adj Close	Prediction
Date				...			
2019-12-31	73.419998	72.379997	72.482498	...	100805600.0	72.192863	61.935001
2020-01-02	75.150002	73.797501	74.059998	...	135480400.0	73.840042	63.702499
2020-01-03	75.144997	74.125000	74.287498	...	146322800.0	73.122154	63.572498
2020-01-06	74.989998	73.187500	73.447502	...	118387200.0	73.704819	60.227501
2020-01-07	75.224998	74.370003	74.959999	...	108872000.0	73.358185	61.232498



Conclusion

If we want to make investments towards our future, having a good grip on how the stock market works is essential.

The stock market simulator we have created has worked satisfactorily by training the decision tree regression model to make decisions using a training set of variables.

Projects like these will help the common man to get more information on the stock market in a simple to understand way since a lot of this information is visualized, which helps people understand it quickly and lastingly.

Personally, we have learned a great deal about Modelling and Simulation, the Stock market, the Python programming language and many other related topics.

Future Scope

In this project, we have implemented the Decision Tree Regression method of predicting the future. As mentioned earlier in this project, this model sometimes fails to accurately predict stock market trends. In the future, more accurate prediction methods can be devised, so that we can correctly predict future stock market tendencies even better.

References

- I. Advanced Matplotlib Charting Series Tutorials for Python
<https://pythonprogramming.net/advanced-matplotlib-graphing-charting-tutorial/>
- II. Python Programming Tutorial- How to make a Stock Screener
<https://www.youtube.com/watch?v=Y4GHgJlQnk>
- III. Making a Stock Screener with Python!
<https://towardsdatascience.com/making-a-stock-screener-with-python-4f591b198261?gi=1bd5a3be36d4>
- IV. Python Programming for Finance- Playlist
<https://youtube.com/watch?v=2BrpKpWwT2A&list=PLQVvva00QuDcOdF96TBtRtuQksErCEBYZ&index=1>
- V. Investopedia- Stock Market
<https://www.investopedia.com/terms/s/stockmarket.asp>