STOCK PRICE PREDICTION

A Project Work Synopsis

Submitted in the partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE IN BIG DATA

Submitted by:
SUSHANT BISHT
20BCS6148

Under the Supervision of:

GURPREET SINGH PANESAR

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413, PUNJAB

2020-2024

ACKNOWLEDGEMENT

I would like to express my deep gratitude to my project guide Gurpreet Singh Panesar, Department of Computer Science and Engineering, for his/her guidance with unsurpassed knowledge and immense encouragement. I am grateful to Mr. Aman Kaushik, the Head of the Department, Computer Science, and Engineering in Big Data, for providing me with the required facilities for the completion of the project work.

I express my thanks to Project Coordinator Gurpreet Singh Panesar, for his continuous support and encouragement. I thank all teaching faculty of the Department of CSE-Big Data, whose suggestions during reviews helped me in the accomplishment of my project.

I would like to thank my parents, friends, and classmates for their encouragement throughout my project period. Last but not the least, I thank everyone for supporting me directly or indirectly in completing this project successfully.

PROJECT STUDENT

SUSHANT BISHT - 20BCS6148

DECLARATION

I, Sushant Bisht of fourth semester B.E., in the department of Computer Science and Engineering-Big Data from CHANDIGARH UNIVERSITY, Gharuan, Mohali, hereby declare that the project work entitled STOCK PRICE PREDICTION is carried out by me and submitted in partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science Engineering-Big Data during the academic year 2020-2024 and has not been submitted to any other university for the award of any kind of degree.

PROJECT STUDENT

SUSHANT BISHT - 20BCS6148

ABSTRACT

In this project, I attempt to implement a machine learning approach to predict stock prices. Machine learning is effectively implemented at forecasting stock prices. The objective is to predict the stock prices to make more informed and accurate investment decisions. We propose a stock price prediction system that integrates mathematical functions, machine learning, and other external factors to achieve better stock prediction accuracy and issue profitable trades.

There are two types of stocks. You may know of intraday trading by the commonly used term "day trading." Interday traders hold securities positions from at least one day to the next and often for several days to weeks or months.

Keywords:Random Forest Regression, ML, DL, Trade Open, Trade Close, Trade Low, Trade High

Table of Contents

Chapter 1: Introduction

Chapter 2: Literature Survey

Chapter 3: Proposed System

Chapter 4: Conclusion

5. REFERENCES

1 INTRODUCTION

The financial market is a dynamic and composite system where people can buy and sell currencies, stocks, equities, and derivatives over virtual platforms supported by brokers. The stock market allows investors to own shares of public companies through trading either by exchange or over-the-counter markets. This market has given investors the chance of gaining money and having a prosperous life through investing small initial amounts of money, low risk compared to the risk of opening a new business or the need for a high salary career. Stock markets are affected by many factors causing uncertainty and high volatility in the market. Although humans can take orders and submit them to the market, automated trading systems (ATS) that are operated by the implementation of computer programs can perform better and with higher momentum in submitting orders than any human. However, to evaluate and control the performance of ATSs, the implementation of risk strategies and safety measures applied based on human judgments are required. Many factors are incorporated and considered when developing an ATS, for instance, trading strategy to be adopted, complex mathematical functions that reflect the state of a specific stock, machine learning algorithms that enable the prediction of the future stock value, and specific news related to the stock being analyzed.

.

1.1 MOTIVATION FOR WORK

Businesses primarily run over customer satisfaction, customer reviews about their products. Shifts in sentiment on social media have been shown to correlate with shifts in stock markets. Identifying customer grievances thereby resolving them leads to customer satisfaction as well as the trustworthiness of an organization. Hence there is a necessity for an unbiased automated system to classify customer reviews regarding any problem.

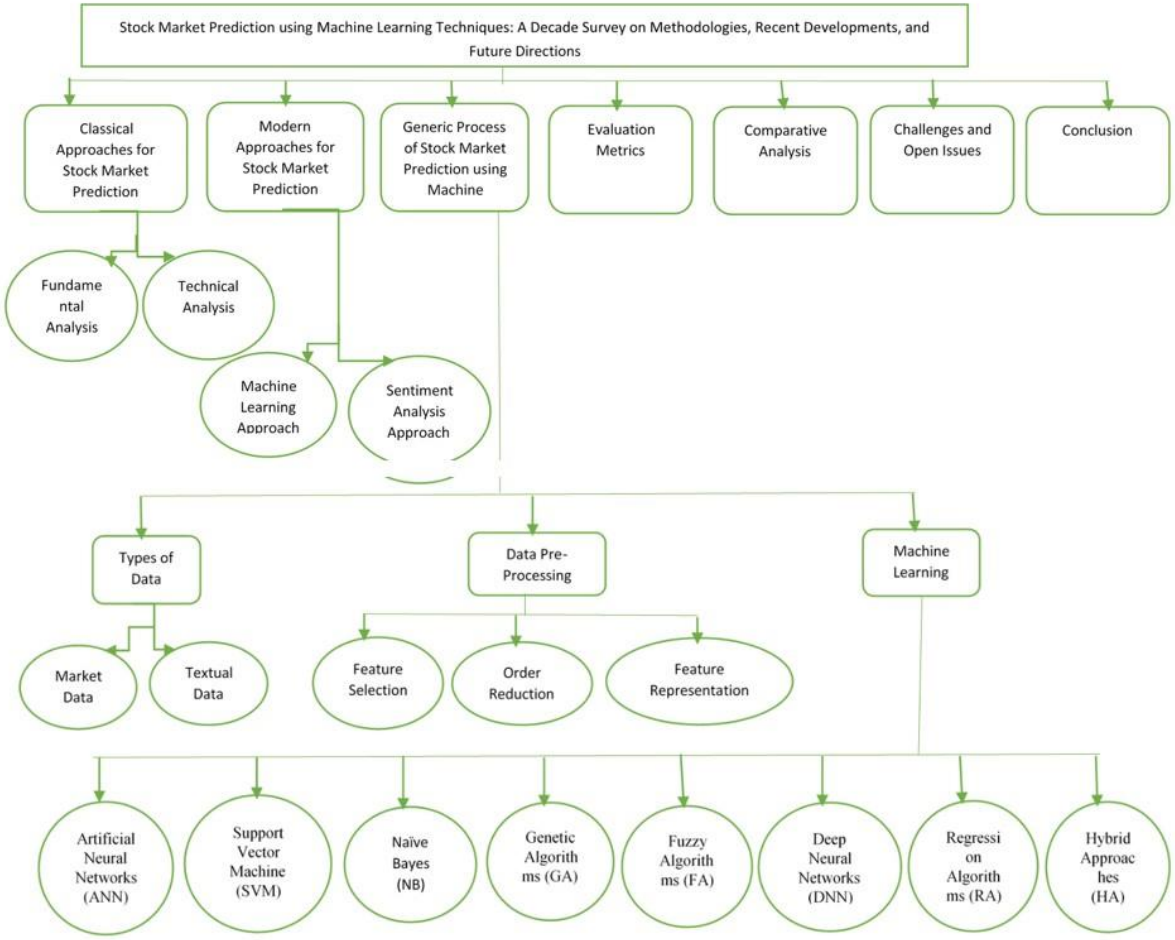
In today's environment where we're justifiably suffering from data overload (although this does not mean better or deeper insights), companies might have mountains of customer feedback collected; but for mere humans, it's still impossible to analyze it manually without any sort of error or bias. Oftentimes, companies with the best intentions find themselves in an insights vacuum. You know you need insights to inform your decision making and you know that you're lacking them, but don't know how best to get them. Sentiment analysis provides some answers into what the most important issues are, from the perspective of customers, at least. Because sentiment analysis can be automated, decisions can be made based on a significant amount of data rather than plain intuition.

1.2 PROBLEM STATEMENT

- Most people don't know the relationship between all financial assets.
- It is difficult to predict what drives the price of each asset.
- It is difficult to find a link between financial assets and the real economy.
- The aim of the project is to examine a number of different forecasting techniques to predict future stock returns based on past returns to construct a portfolio of multiple stocks in order to diversify the risk. We do this by applying supervised learning methods for stock price forecasting by interpreting the seemingly

chaotic market data.

2 LITERATURE SURVEY



2.1 INTRODUCTION

"What other people think" has always been an important piece of information for most of us during the decision-making process. The Internet and the Web have now (among other things) made it possible to find out about the opinions and experiences of those in the vast pool of people that are neither our acquaintances nor well-known professional critics — that is, people we have never heard of. And conversely, more and more people are making their opinions available to strangers via the Internet. The interest that individual users show in online opinions about products and services, and the potential influence such opinions wield, is something that is the driving force for this area of interest. And there are many challenges involved in this process that needs to be walked all over to attain proper outcomes out of them. In this survey, I have analyzed the basic methodology that usually happens in this process and measures that are to be taken to overcome the challenges being faced.

2.2 STOCK MARKET PREDICTION USING MACHINE LEARNING

Stock market prediction is an act of trying to determine the future value of a stock other financial instrument traded on a financial exchange. This paper explains the prediction of a stock using Machine Learning. The technical and fundamental or the time series analysis is used by most of the stockbrokers while making the stock predictions. The programming language is used to predict the stock market using machine learning is Python. In this paper, I have proposed a Machine Learning (ML) approach that will be trained from the available stocks data and gain intelligence and then uses the acquired knowledge for an accurate prediction.

3 METHODOLOGIES

- Collect data from source
- Normalize Data
- Scale Data
- Create a Training and Testing dataset
- Train all the models using the training dataset
- Test the model and determine the score
- Predict the stocks using trained models
- Compare the predictions of all models using a graph
- Choose the best model between the algorithm
- Save trained model
- Display the model on the website

- Show prediction of stock prices

3.1 PROPOSED SYSTEMS

The prediction methods can be roughly divided into two categories, statistical methods and artificial intelligence methods. Statistical methods include the logistic regression model, ARCH model, etc., support vector machine, recurrent neural network, etc. We use Random Forest Regression Algorithm for this.

Hardware Requirements:

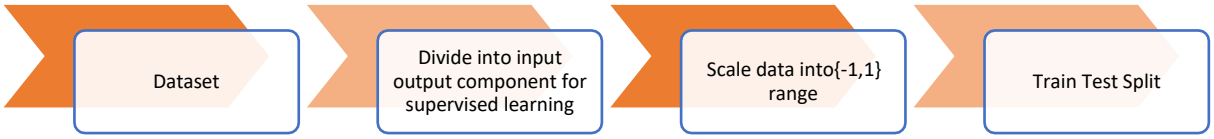
- RAM: 4 GB
- Storage: 500 GB
- CPU: 2 GHz or faster
- Architecture: 32-bit or 64-bit

Software Requirements:

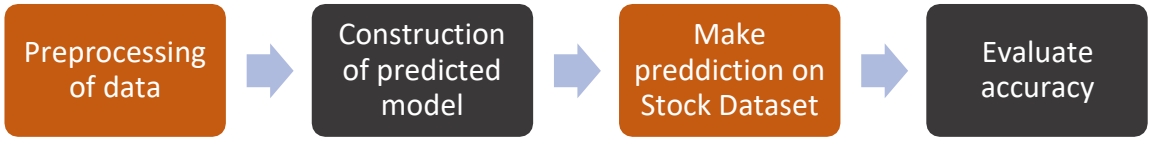
- Python 3.5 in Google Collab is used for data pre-processing, model training, and prediction.
- Operating System: Windows 10 and above or Linux based OS or MAC OS

System Architecture:

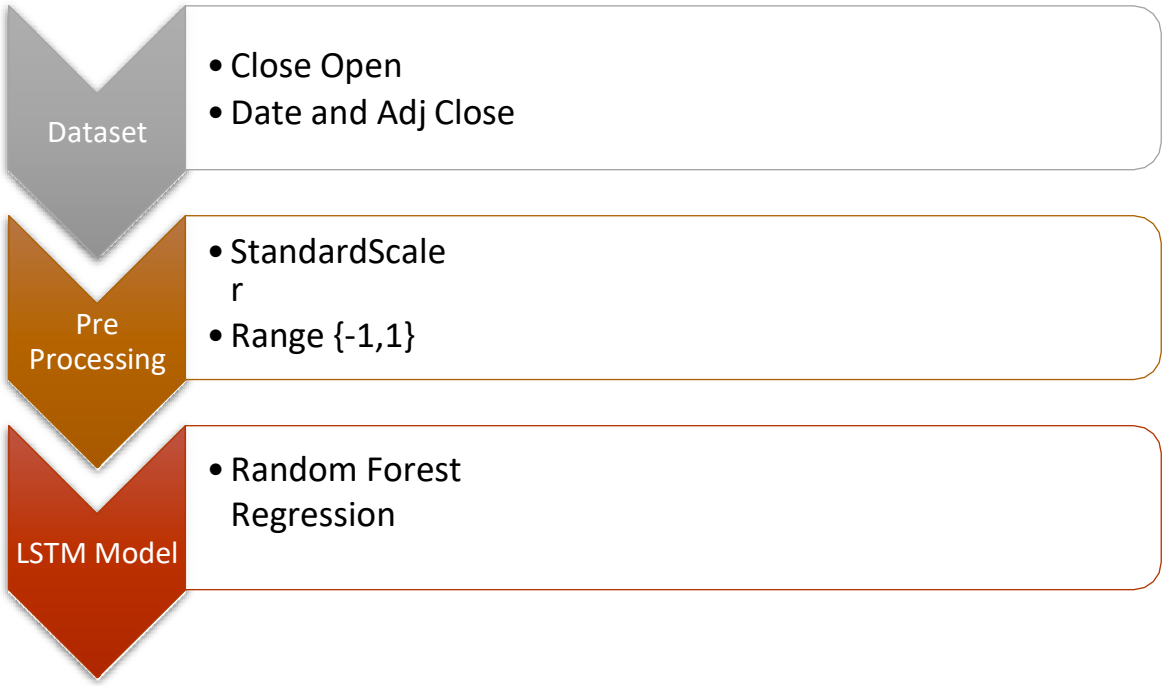
1) Preprocessing of data



2) Overall Architecture:



3) Flowchart:



Sample Code:

```
import NumPy as np

import pandas as pd


import matplotlib.pyplot as plt

import pandas_datareader as data
```

#moving average for 100 values

ma100 = df.Close.rolling(100).mean()

ma100

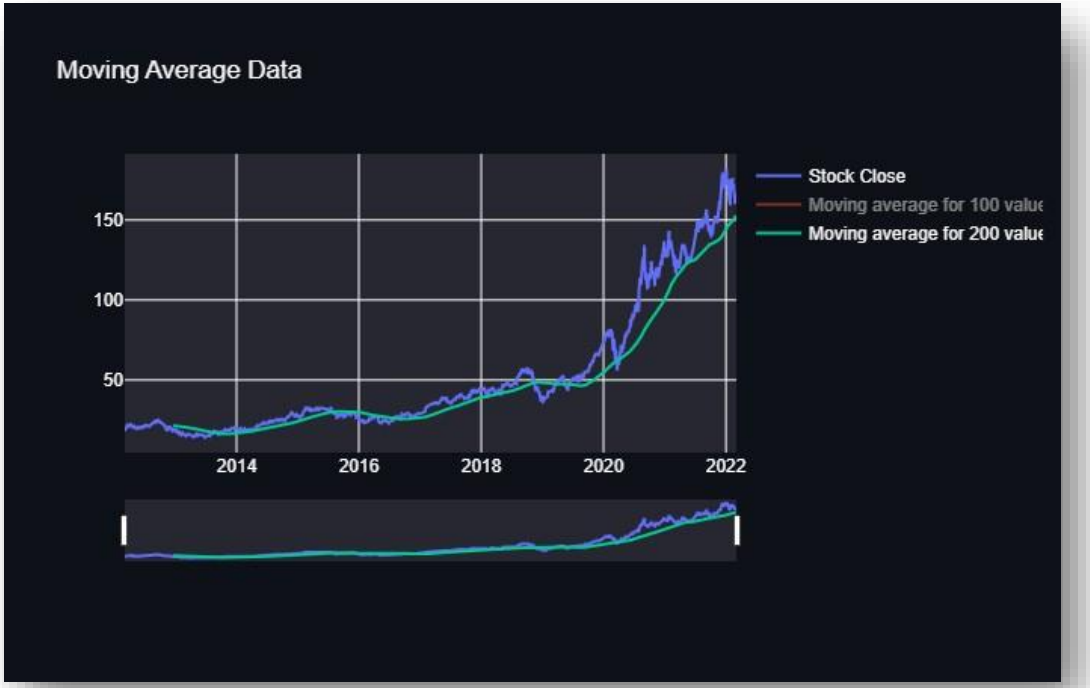
now n the below dataset you can see that the moving average for the first 100 terms is null but for the next, all value its have some integer.

plt.figure(figsize=(12,6))

plt.plot(df.Close)

plt.plot(ma100,'r')

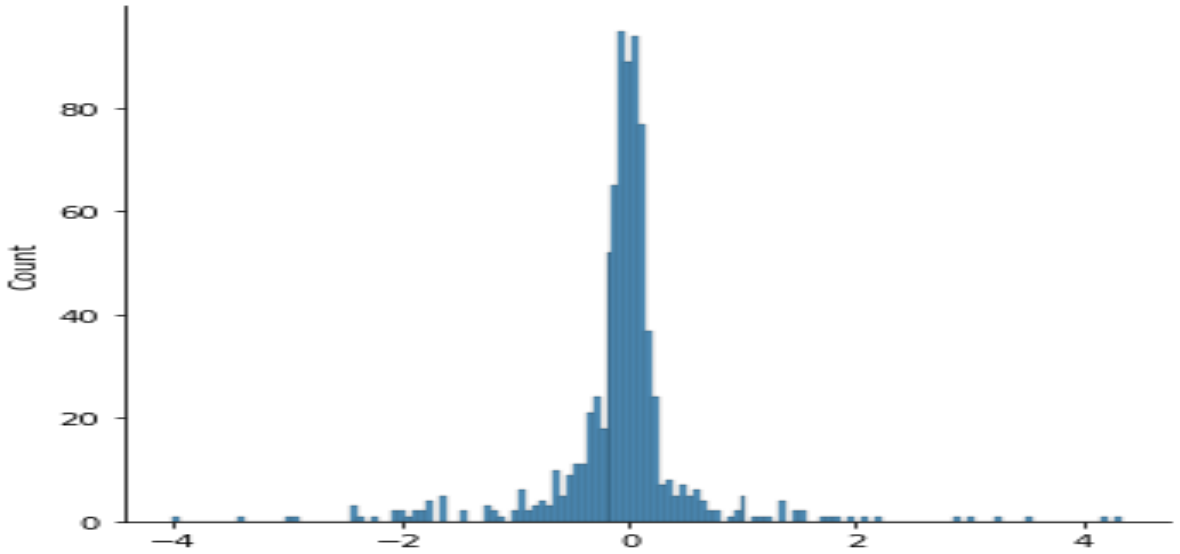

```
ma200 = df.Close.rolling(200).mean()

ma200

plt.figure(figsize=(12,6))

plt.plot(df.Close)

plt.plot(ma100,'r')

plt.plot(ma200,'y')
```




```
from sklearn.ensemble import RandomForestRegressor
regression = RandomForestRegressor(n_estimators=10, random_state=0)
regression.fit(x_train, y_train)
```

```
np.concatenate((y_pred.reshape(len(y_pred), 1), y_test.reshape(len(y_test), 1)), 1)
```

```
sns.displot(y_pred-y_test)
```



App.py(Front End):

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from plotly import graph_objs as go
import yfinance as yf
from sklearn.metrics import r2_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor
import streamlit as st
import datetime


stocks =("AAPL","GOOG","MSFT","SBIN.NS","KOTAKBANK.NS","AXISBANK.NS","INFY.NS","BTC-INR")
st.set_page_config(layout="wide")
def side_display():
    with st.sidebar:
```

```
        st.title("Dashboard😊")
        st.subheader("To compare the stocks of more than one company 📊")
        dropdown= st.multiselect('Pick your assets',stocks)

        start_date = st.date_input("Select Starting Date",datetime.date.today())
        end_date = st.date_input("Select Ending Date",datetime.date.today())
    start_date =None if start_date>=datetime.date.today() else start_date
    end_date =None if end_date>=datetime.date.today() else end_date
    return start_date,end_date,dropdown

@st.cache
def load_data(ticker,start,end):
    #st.download_button('Download',data.to_csv())
    data = yf.download(ticker,start,end)
    data.reset_index(inplace=True)
    return data

def relativeret(df):
    rel  =df.pct_change()
    cumret =(1+rel).cumprod()-1
    cumret = cumret.fillna(0)
    return cumret

def show_update(open_val,close_val,high_val,low_val,ov,hv,lv,cv):
    kpi1, kpi2, kpi3, kpi4= st.columns(4)
    kpi1.metric(label="$OPEN",value=open_val[len(open_val)-1],delta=ov)
    kpi2.metric(label="$High",value=high_val[len(high_val)-1],delta=hv)
    kpi3.metric(label="$Low",value=low_val[len(low_val)-1],delta=lv)
    kpi4.metric(label="$Close",value=close_val[len(close_val)-1],delta=cv)

st.title("Stock Price Prediction ")
st.subheader("- using Random Forest Regressor🚀📈by sushant Bisht ")



selected_stocks = st.selectbox("Select Dataset for prediction ",stocks)
```

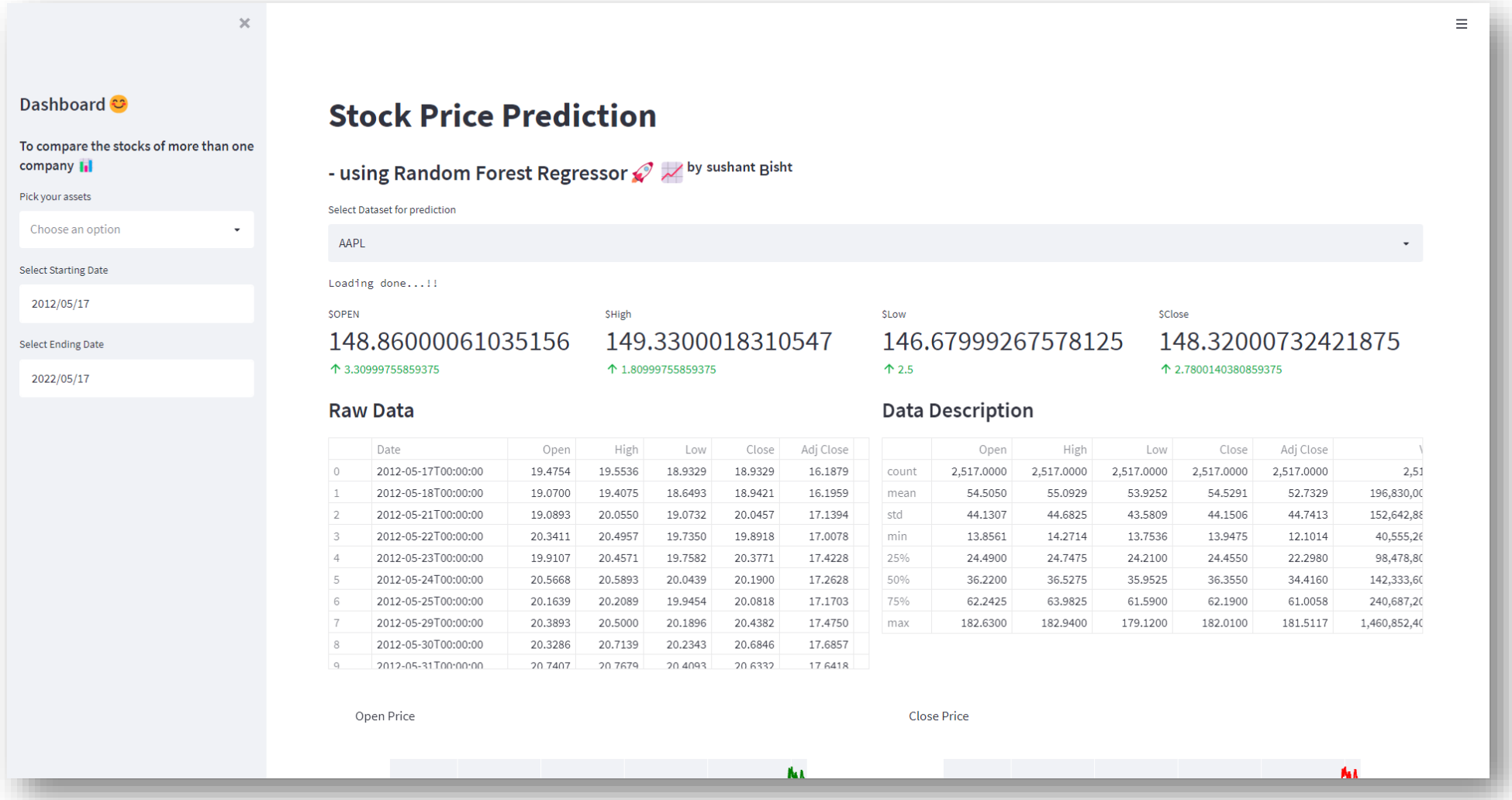
```
start_date,end_date,dropdown=side_display()
data_load_state = st.text("Load data.....")
data = load_data(selected_stocks,start_date,end_date)
data_load_state.text("Loading done...!!")

# for latest open values
open_val = data['Open'].values
ov = open_val[len(open_val)-1] - open_val[len(open_val)-2]

#for latest high values
high_val = data['High'].values
hv = high_val[len(high_val)-1] - high_val[len(high_val)-2]

low_val = data['Low'].values
lv = low_val[len(low_val)-1] - low_val[len(low_val)-2]

close_val = data['Close'].values
cv = close_val[len(close_val)-1] - close_val[len(close_val)-2]

show_update(open_val,close_val,high_val,low_val,ov,hv,lv,cv)


if len(dropdown)>0:
    df = relativeret(yf.download(dropdown,start_date,end_date)['Adj Close'])
    st.line_chart(df)




full_display,details_display = st.columns(2)
full_display.subheader("Raw Data")
full_display.write(data)

details_display.subheader('Data Description')
details_display.write(data.describe())
```

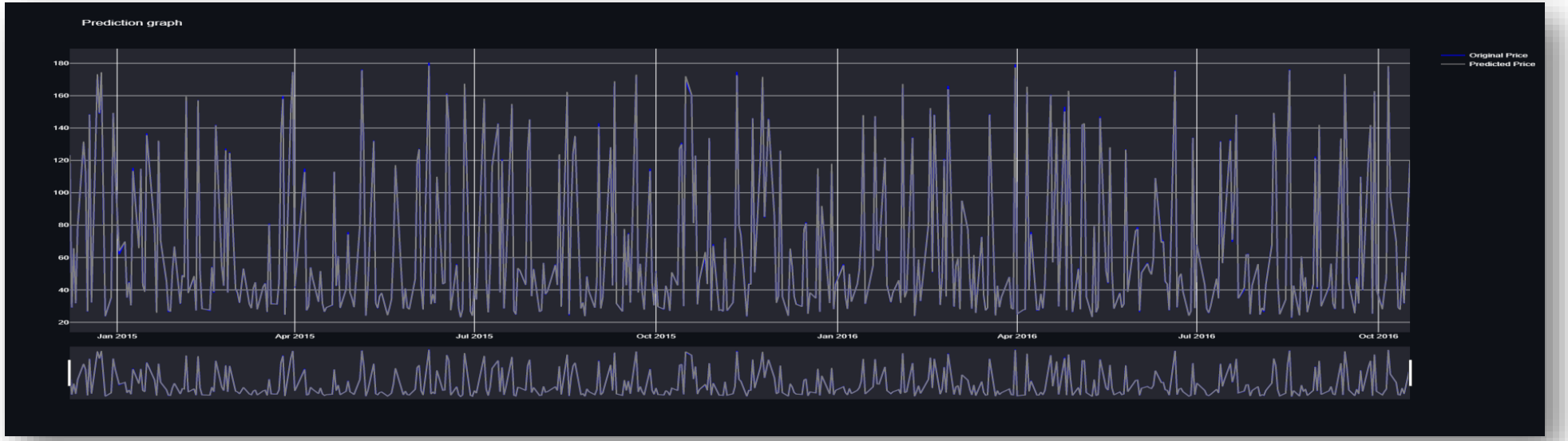
```
dataploy_open,dataplot_close = st.columns(2)
def plot_open_dataset():
    fig = go.Figure()
    fig.add_trace(go.Scatter(x=data['Date'],y=data['Open'],name="Stock Open",line=dict(color='green')))
    fig.layout.update(title_text ="Open Price",xaxis_rangeslider_visible=True)
    dataploy_open.plotly_chart(fig)

def plot_close_dataset():
    fig = go.Figure()
    fig.add_trace(go.Scatter(x=data['Date'],y=data['Close'],name="Stock Close",line=dict(color='red')))
    fig.layout.update(title_text ="Close Price",xaxis_rangeslider_visible=True)
    dataplot_close.plotly_chart(fig)

plot_open_dataset()
plot_close_dataset()

raw_data_col,moving_average_col = st.columns(2)

def plot_raw_data(data):
    fig = go.Figure()
    fig.add_trace(go.Scatter(x=data['Date'],y=data['Open'],name="Stock Open"))
    fig.add_trace(go.Scatter(x=data['Date'],y=data['Close'],name="Stock Close"))
    fig.layout.update(title_text ="Time series data",xaxis_rangeslider_visible=True)
    raw_data_col.plotly_chart(fig)

def moving_average(data):
    ma100 = data.Close.rolling(100).mean()
    ma200 = data.Close.rolling(200).mean()
    fig = go.Figure()
    fig.add_trace(go.Scatter(x=data['Date'],y=data['Close'],name="Stock Close"))
    fig.add_trace(go.Scatter(x=data['Date'],y=ma100,name="Moving average for 100 values"))
    fig.add_trace(go.Scatter(x=data['Date'],y=ma200,name="Moving average for 200 values"))
    fig.layout.update(title_text ="Moving Average Data",xaxis_rangeslider_visible=True)
    moving_average_col.plotly_chart(fig)
def plot_pridicted_graph(data,y_test,y_pred):
    fig = go.Figure()
```

```
    fig.add_trace(go.Scatter(x= data['Date'], y=y_test, name="Original Price", line=dict(color='blue')))
    fig.add_trace(go.Scatter(x=data['Date'], y=y_pred, name="Predicted Price", line=dict(color='grey')))
    fig.layout.update(title_text ="Prediction graph",xaxis_rangeslider_visible=True)
    st.plotly_chart(fig)

plot_raw_data(data)
moving_average(data)
df = data.drop(['Date','Adj Close'], axis=1)

x = data.iloc[:,1:4].values
y = data.iloc[:,4].values

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=0)

sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)

regression = RandomForestRegressor(n_estimators=10, random_state=0)
regression.fit(x_train, y_train)

y_pred = regression.predict(x_test)

print(np.concatenate((y_pred.reshape(len(y_pred), 1), y_test.reshape(len(y_test), 1)), 1))

st.subheader("Here test data and Predicted data you can see how similar they are !!")

full_display,details_display = st.columns(2)
full_display.subheader("Test Data")
full_display.write(y_test.reshape(len(y_test), 1))

details_display.subheader('Predicted Data')
details_display.write(y_pred.reshape(len(y_pred), 1))

st.subheader("R2 Score for model : ")
st.text(r2_score(y_test,y_pred))
```

```
with st.form(key=" form1"):
    st.subheader("want to predict close price for any open price ?")
    open_input = st.text_input("Open ",198.779999)
    high_input = st.text_input("High ",199.990005)
    low_input = st.text_input("Low ",197.619995)
    input_values = []
    final_input = []
    input_values.append(open_input)
    input_values.append(high_input)
    input_values.append(low_input)
    final_input.append(input_values)

    final_input=sc.transform(final_input)
    st.form_submit_button(label="Predict")

    st.subheader('Predicted Closing Price ')
    st.write(regression.predict(final_input))

st.subheader("R2 Score for model : ")
st.text(r2_score(y_test,y_pred))

# st.subheader("prediction")
# fig2 =plt.figure(figsize=(50,12))
# plt.plot(y_test,'g',label='Original Price')
# plt.plot(y_pred,'y',label='Predicted Price')
# plt.xlabel('Date')
# plt.ylabel('Close')
# plt.legend()
# st.pyplot(fig2)

plot_pridicted_graph(data, y_test, y_pred)
```

Input and Output:





Here test data and Predicted data you can see how similar they are !!

Test Data

	0
0	17.2511
1	0.9933
2	0.3320
3	125.9100
4	21.2057
5	0.4152
6	0.1496
7	2.0357
8	17.6132
9	0.3728

Predicted Data

	0
0	17.3388
1	0.9334
2	0.3343
3	125.2998
4	21.1125
5	0.4248
6	0.1501
7	2.0383
8	17.3014
9	0.3693

CONCLUSION AND FUTURE WORK

Conclusion:

In this project, we are predicting the closing stock price of any given organization, we developed a web application for predicting close stock price using Random Forest Regression algorithms model for prediction. We have applied datasets belonging to Google, Apple, Microsoft, Infosys, and Reliance Stocks and achieved above 95% accuracy for these datasets.

Future work:

- We want to extend this application for predicting cryptocurrency trading.
- We want to add sentiment analysis for better analysis.

REFERENCES

[1] Yahoo Finance ww.yahoo.com

2.1 Literature Review Summary

Table 2.1: Literature review summary

Year and citation	Article Title	Purpose of the study	Tools/ Software used	Comparison of technique done	Source (Journal/ Conference)	Findings	Data set (if used)	Evaluation parameters
2022	Stock Price Prediction	Forecast the Price Trend	Machine Learning Android Studio Python Heroku Git	We usually test our data with a Machine learning algorithm which provides best prediction	StackOverflow	Yahoo Finance	Apple, Google, Microsoft, SBI, Kotak and many more	displot