CMPE256
Individual Project Report
Stock Recommendation using Twitter Sentiment Analysis

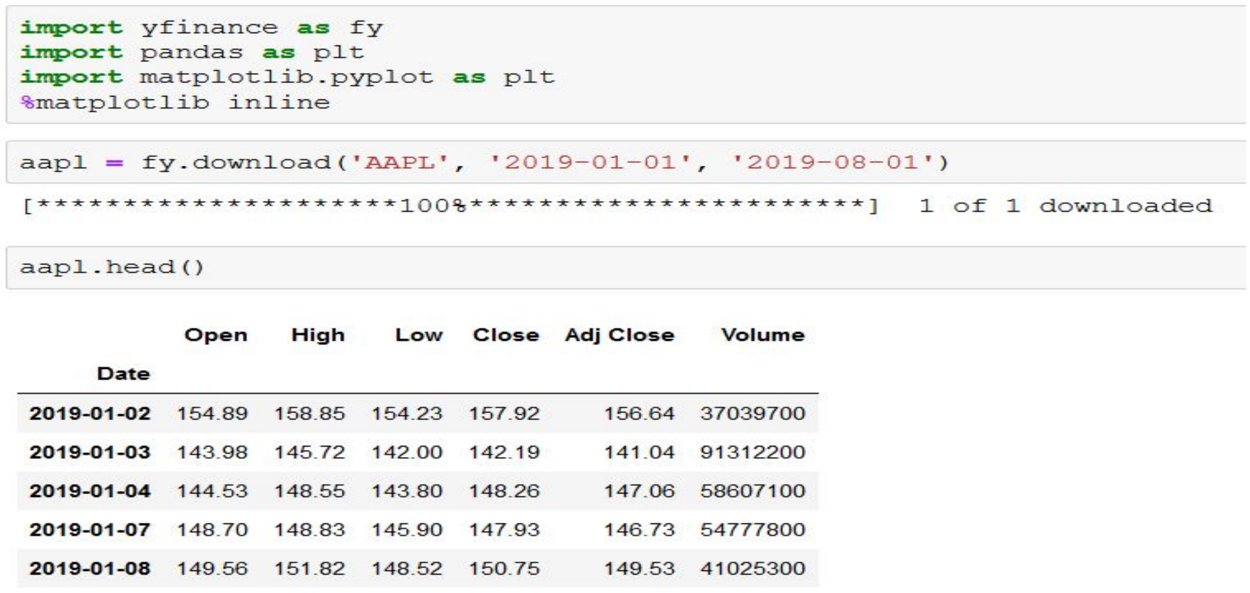
Name: Sai Sandeep Jyothula
SJSU ID: 013821418
San Jose State University

1. Introduction:

In many situations, investors find it difficult to decide whether to invest in a particular stock or not. In general, many factors are taken into consideration by the investors before investing in a stock of a random company. Initially I will predict the stock prices for various companies and then to help the investors to make a decision, I have built a stock recommendation system which take the sentiment of tweets(positive or negative) related to a particular stock and then recommend to the investor based on the output whether to invest in that particular stock or not.

2. Data collection:

For this particular project I had to collect two datasets to work on. One related to the stock information of various companies and the other is the tweets related to those companies. So to collect the stock information I have used the Yahoo! Finance API. Using this API I have retrieved the stock information related to a particular company. For example to retrieve stock information related to Apple inc. from Yahoo finance I have used the following lines of code.



```
import yfinance as fy
import pandas as plt
import matplotlib.pyplot as plt
%matplotlib inline
```

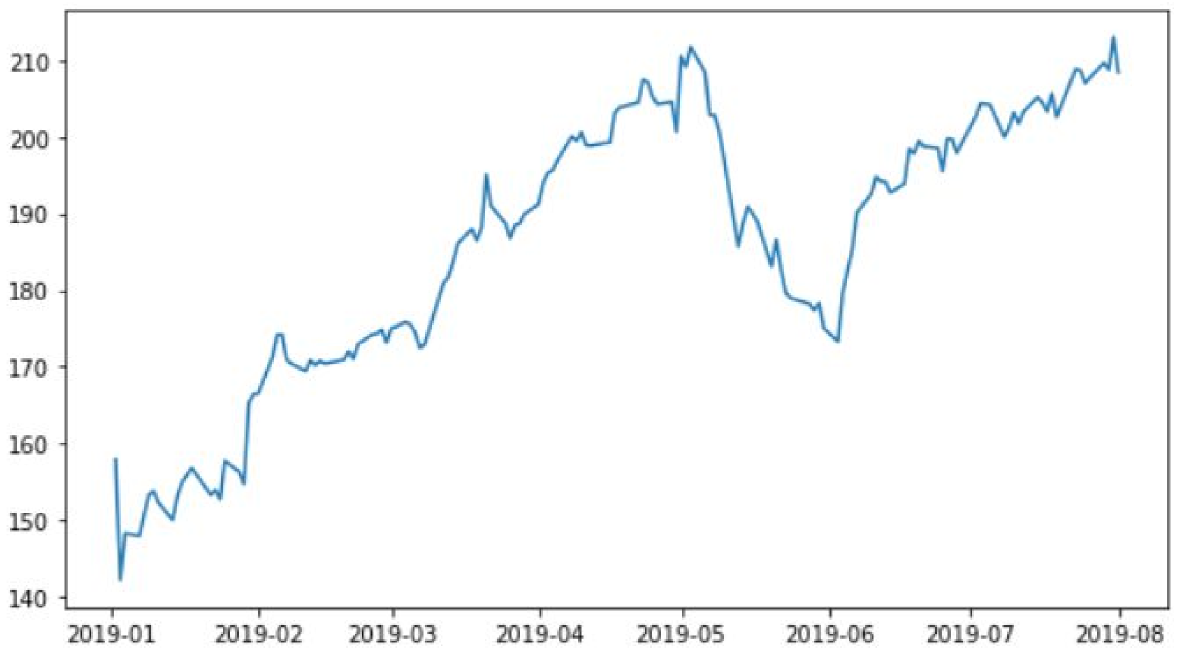
```
aapl = fy.download('AAPL', '2019-01-01', '2019-08-01')
```

```
[*********************100%***********************]  1 of 1 downloaded
```

```
aapl.head()
```

Date	Open	High	Low	Close	Adj Close	Volume
2019-01-02	154.89	158.85	154.23	157.92	156.64	37039700
2019-01-03	143.98	145.72	142.00	142.19	141.04	91312200
2019-01-04	144.53	148.55	143.80	148.26	147.06	58607100
2019-01-07	148.70	148.83	145.90	147.93	146.73	54777800
2019-01-08	149.56	151.82	148.52	150.75	149.53	41025300

I have plotted the results obtained from Yahoo finance to get a better understanding of the stock prices of that company.



The twitter data, that is the tweets related to that particular company(for example Apple inc here) is obtained from the Twitter API.

3. Model:

3.1 Model for Stock forecasting:

Once the stock data is collected the first step in building the model is to predict the stock prices for the coming days based on the previously collected stock data. This is done using the following lines of code. The first step in this process is to validate the user input i.e., to check whether the company code given by the user is a valid one.

```
def check_stock_symbol(flag=False, companies_file='companylist.csv'):
    df = pd.read_csv(companies_file, usecols=[0])

    while flag is False:
        symbol = input('Enter a stock symbol to retrieve data from: ').upper()
        for index in range(len(df)):
            if df['Symbol'][index] == symbol:
                flag = True
    return flag, symbol
```

Now, the next step is to create the Pandas DataFrame of the introduced symbol stock market values from the last year from now. The information is retrieved from Yahoo! Finance as said earlier. Once the data frame is created, I started modelling it to use the prediction algorithms from scikit-learn, here I am using LinearRegression because it has a better accuracy score overall. The model for the forecast

```
def get_stock_data(symbol, from_date, to_date):
    data = yf.download(symbol, start=from_date, end=to_date)
    df = pd.DataFrame(data=data)

    df = df[['Open', 'High', 'Low', 'Close', 'Volume']]
    df['HighLoad'] = (df['High'] - df['Close']) / df['Close'] * 100.0
    df['Change'] = (df['Close'] - df['Open']) / df['Open'] * 100.0

    df = df[['Close', 'HighLoad', 'Change', 'Volume']]
    return df
```

Then I apply a preprocessing to the X values, that are the original values of the Close ones but without the forecast out values that are shifted up to 10% of the whole dataset. So I apply the cross-validation algorithm to establish the X and Y values for training and testing

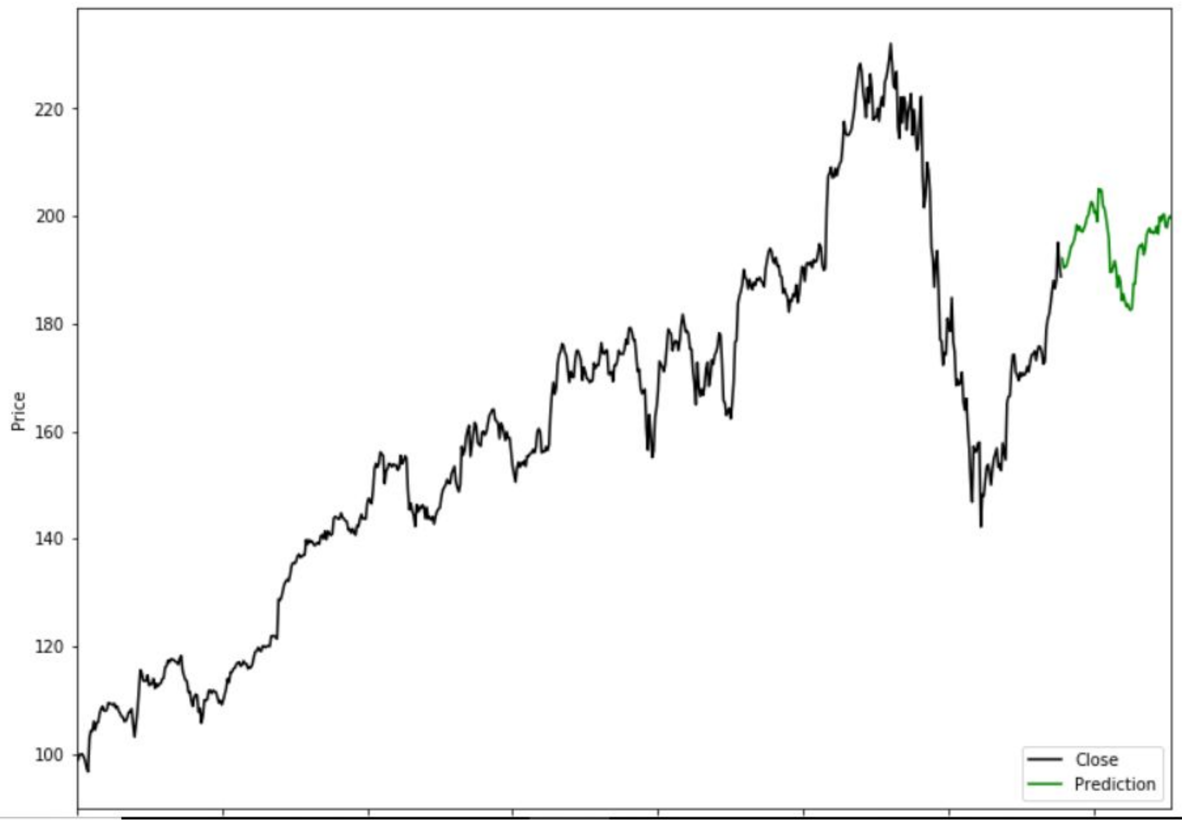
```
X = np.array(df.drop(['Label'], axis=1))
X = preprocessing.scale(X)
X_forecast = X[-forecast_out:]
X = X[:-forecast_out]

df.dropna(inplace=True)
y = np.array(df['Label'])

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5)
```

Finally I apply the Linear regression algorithm to the data set and then I plot it to see the representation of the forecast made.

```
clf = LinearRegression(n_jobs=-1)
clf.fit(X_train, y_train)
accuracy = clf.score(X_test, y_test)
forecast = clf.predict(X_forecast)
df['Prediction'] = np.nan

last_date = df.iloc[-1].name
last_date = dt.datetime.strptime(str(last_date), "%Y-%m-%d %H:%M:%S")

for pred in forecast:
    last_date += dt.timedelta(days=1)
    df.loc[last_date.strftime("%Y-%m-%d")] = [np.nan for _ in range(len(df.columns) - 1)] + [pred]
return df, forecast_out
```

For example, for Apple Inc. (AAPL) the prediction plot looks like this:

```
Enter a stock symbol to retrieve data from: aapl
Retrieving Stock Data from introduced symbol...
[*********************100%***********************]  1 of 1 downloaded
Forecasting stock DataFrame...
Plotting existing and forecasted values...
```



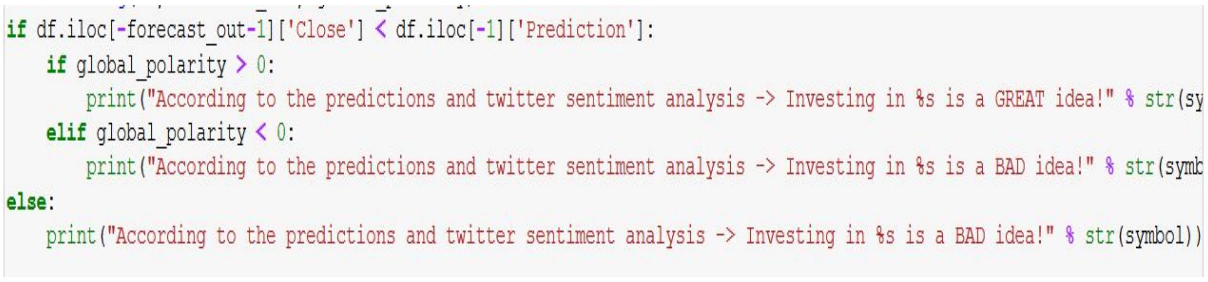
3.2 Retrieving tweets polarity:

Once the stock predictions has finished, the Twitter sentiment analysis starts and it retrieves a list of the last 100 tweets posted in english containing the symbol introduced and they are later stored in a list of Tweet class with the tweet's text and polarity from TextBlob.

```python
auth = tweepy.OAuthHandler(consumer_key,consumer_secret )
auth.set_access_token(access_token ,access_token_secret )
user = tweepy.API(auth, wait_on_rate_limit=True)

tweets = tweepy.Cursor(user.search, q=str(symbol), tweet_mode='extended', lang='en').items(num_of_tweets)

tweet_list = []
global_polarity = 0
for tweet in tweets:
    tw = tweet.full_text
    blob = TextBlob(tw)
    polarity = 0
    for sentence in blob.sentences:
        polarity += sentence.sentiment.polarity
        global_polarity += sentence.sentiment.polarity
    tweet_list.append(Tweet(tw, polarity))

global_polarity = global_polarity / len(tweet_list)
return global_polarity
```

3.3 Recommendations:

Now, to make the recommendation I check if the prediction was favorable compared to the last non-predicted value and if it is, then the global polarity is checked and if it is positive (>0.0 according to TextBlob values), the system recommendation is to invest in that symbol, if not, the recommendation is not to invest.

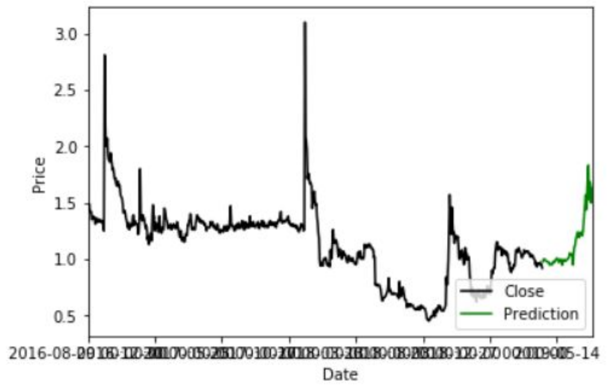


```
if df.iloc[-forecast_out-1]['Close'] < df.iloc[-1]['Prediction']:
    if global_polarity > 0:
        print("According to the predictions and twitter sentiment analysis -> Investing in %s is a GREAT idea!" % str(sy
    elif global_polarity < 0:
        print("According to the predictions and twitter sentiment analysis -> Investing in %s is a BAD idea!" % str(symb
else:
    print("According to the predictions and twitter sentiment analysis -> Investing in %s is a BAD idea!" % str(symbol))
```

4. Results:

Finally, the model will show the output as whether it is recommended to invest in the stocks of that particular company or not.

```
Enter a stock symbol to retrieve data from: ACST
Retrieving Stock Data from introduced symbol...
[*********************100%***********************]  1 of 1 downloaded
Forecasting stock DataFrame...
Plotting existing and forecasted values...
```



```
Retrieving ACST related tweets polarity...
Generating recommendation based on prediction & polarity...
According to the predictions and twitter sentiment analysis -> Investing in ACST is a GREAT idea!
```

Here the user is searching for Acasti Pharma, Inc. company. So my model will predict the stock prices for that company and also the tweets related to that company and finally give out a recommendation saying that Investing in ACST is GREAT idea!