Sentiment Analysis, Emotions and Predictability of Stock-Money Market

Yousuf Minhaj Siddiqui

MS (Data Science)

A project submitted in partial fullfillment of the requirements for the degree of Master of Science in Data Science at the National University of Computer & Emerging Sciences

****

Computer Science

National University of Computer & Emerging Sciences

2021

**Plagiarism Undertaking**

I take full responsibility for the research work conducted during the MS Project titled Sentiment Analysis, Emotions and Predictability of Stock-Money Market. I solemnly declare that the work presented in the project is done solely by me with no significant help from any other person; however, small help wherever taken is duly acknowledged. I have also written the complete thesis by myself. Moreover, I have not presented this thesis (or substantially similar research work) or any part of the project previously to any other degree-awarding institution within Pakistan or abroad.

I understand that the management of the National University of Computer and Emerging Sciences has a zero-tolerance policy towards plagiarism. Therefore, I as an author of the above-mentioned project, solemnly declare that no portion of my project has been plagiarized and any material used in the thesis from other sources is properly referenced. Moreover, the project does not contain any literal citing of more than 70 words (total) even by giving a reference unless I have the written permission of the publisher to do so. Furthermore, the work presented in the thesis is my own original work and I have positively cited the related work of the other researchers by clearly differentiating my work from their relevant work.

I further understand that if I am found guilty of any form of plagiarism in my project work even after my graduation, the University reserves the right to revoke my MS degree. Moreover, the University will also have the right to publish my name on its website that keeps a record of the students who plagiarized in their thesis work.

**Yousuf Minhaj Siddiqui (19K-0943)**

**Author’s Declaration**

I, **Yousuf Minhaj Siddiqui** hereby state that my MS Project named **Sentiment Analysis, Emotions and predictability of Stock/Money Market** is my own work and it has not been previously submitted by me for taking partial or full credit for the award of any degree at this University or anywhere else in the world. If my statement is found to be incorrect, at any time even after my graduation, the University has the right to revoke my MS degree.

**Yousuf Minhaj Siddiqui (19K-0943)**

Certificate of Approval



*It is certified that the research work presented in this thesis, entitled* ***“ Sentiment Analysis, Emotions & Predictability of Stock-Money Market ”*** *was conducted by* ***Yousuf Minhaj Siddiqui*** *under the supervision of*

***Dr. Muhammad Nouman Durrani***

*No part of this Project has been submitted anywhere else for any other degree.*

*This project is submitted to the* ***FAST School of Computing*** *in partial fulfillment of the requirements for the degree of Master of Science in* ***“Data Science”***

*at the*

*National University of Computer and Emerging Sciences*

*Karachi, PAKISTAN*

***14.01.2022***

Candidate Name: Yousuf Minhaj Siddiqui Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Examination Committee:

a) Name: <<External>> Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Designation, University,

b) Name: <<Internal>> Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Designation, University,

Supervisor:

c) Name: Muhammad Nouman Duraani Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Designation, University,

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Head, FAST School of Computing, National University of Computer and Emerging Sciences, Karachi

**Abstract**

Today's market globally and locally in Pakistan Stock market prediction is incredibly vital within the planning of business activities. Stock worth prediction has attracted several researchers in multiple disciplines as well as computer science, statistics, economics, finance, and operations research. Recent studies have shown that the bulk quantity of online data within the property right like Wikipedia usage pattern, news stories from the thought media, and social media discussions will have gained noticeable results of investors' opinions towards money markets. The responsibility of the procedure models on exchange prediction is important because it is incredibly sensitive to the economy and might directly lead to loss. In this project named “Sentiment Analysis, Emotions and Predictability for Stock Market”, we have a tendency to retrieve, extract, and analyze the consequences of stories' sentiments on the exchange. Our main contributions embody the event of a sentiment analysis and opinion mining lexicon for the money sector, the event of a dictionary-based sentiment analysis model along with machine learning algorithms, and therefore the analysis of the model for gauging the consequences of stories sentiments on stocks or the crypto currencies.

**Acknowledgment**

I am thankful to Almighty Allah who praise me with the ability to think, work and deliver what I was assigned to do. I am also thankful to our supervisor **Dr. Muhammad Nouman Durrani** who helps us throughout the project. I am also thankful to all my seniors, colleagues, and mentors, without their efforts this thesis may not have completed. I also acknowledge our teachers that throughout our studies, they help us and guide us with the most they can do. Besides, this thesis makes us realized the value of working together as a team and as a new experience in working environment, which challenges us ever minute. The whole project really brought us together to appreciate the true value of friendship and respect of each other. I would also thanks to our university, faculty members, and class fellows who helped us in many ways. I also extend my heartfelt thanks to our families and well-wishers.

**Contents**

Introduction11

Background & Related Work13

Methodology15

Tools & Platform17

Project Timelines17

Software Requirement Specification18

Introduction18

Scope18

Problem Definition19

Social Media Sentiment Analysis19

Overall Description19

Data Collection19

Environment Creation20

Required Languages20

Data Preparation20

Data Pre-Processing for YAHOO20

Data Pre-Processing for Crypto20

Data Pre-Processing for Twitter21

Sentiment Analysis21

Model Analysis21

System Features & Requirements23

Functional Requirement23

Non-Functional Requirement23

External Requirement23

Software Application24

System View25

Key Algorithms30

ARIMA Model30

LSTM Model32

LR Model35

Fetching & Cleaning Tweets37

Sentiments Recommendation40

Comparison42

Conclusion & Future Work43

References44

**List of Figures**

Block Diagram16

Project Timeline18

Model Overview21

Working & Behavior22

Dashboard View & Trends24

Dashboard View & Trends (Partial)25

Today Stock/Ticker Information (UI)25

Today Stock/Ticker Information25

Downloaded Data view from YAHOO25

ARIMA Prediction & RMSE26

Recent Trends & ARIMA Prediction26

ARIMA Actual & Predicted Trend26

LSTM Prediction & RMSE27

LSTM, Linear Regression & Prediction For Upcoming Day27

LSTM Actual & Predicted Trend27

LR Prediction & RMSE28

LR Actual & Predicted Trend28

Tokenized Tickers/Stock After Processing Tweets29

Recommendation After Sentiment Analysis29

Recommendation After Sentiment Analysis42

Chapter 1

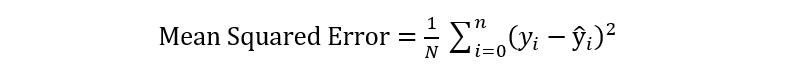
**Introduction**

All Stock Prediction is a challenging problem within the field of finance similarly as engineering, computer science and arithmetic. Due to its gain, it's attracted a lot of attention each from educational facet and business facet. Stock worth prediction has continually been a subject matter of interest for many investors and monetary analysts.

In last ten years, stock prices or crypto currencies popularity have experienced tremendous global growth resulting in much higher trading prices more over this might be because of the reaction of the people over recent years on social media, news and RSS live feeds. Money Market movement is explained as up and down of the markets. The upward shift represents positive returns, while the downward shift represents negative returns. In last couple of days Elon Musk, the founder and CEO of well know Automobile company and space research firm only tweets about a crypto currency on social media, only a minute later the trading price of that crypto currency reached at its peak. The distinctive nature of social media website makes it a valuable source for mining public views or emotions to predict the nature of money market. Recently a huge number of progress has been made to study emotions using social media but there is a limited research or platform available for the users where they could view the prediction and the investment on stocks and crypto currencies along with their market caps.

Most popular task in natural language processing area is sentiment classification which predicts sentiment’s opinion or emotion from a given corpus. This proposed project to predict the market with respect to the previous result of actual market along with the sentiment and emotions of the people helps the investors how, when and where to invest resulting in growth and stability of the economy in future.

We have calculated the MSE for all the models and then compare all the models by MSE using the formula



However, result show result show the MSE all Machine Learning algorithms between 6-14. On average we have predicted the company stock for the next opening day and also predicted the stocks based on twitter news for a week long.

Chapter 2

**Background and Related Work**

In last ten years, stock prices or crypto currencies popularity have experienced tremendous global growth resulting in much higher trading prices more over this might be because of the reaction of the people over recent years on social media, news and RSS live feeds [1]. Money Market movement is explained as up and down of the markets. The upward shift represents positive returns, while the downward shift represents negative returns. In last couple of days Elon Musk, the founder and CEO of well know Automobile company and space research firm only tweets about a crypto currency on social media, only a minute later the trading price of that crypto currency reached at its peak [4]. The distinctive nature of social media website makes it a valuable source for mining public views or emotions to predict the nature of money market. Recently a huge number of progress has been made to study emotions using social media but there is a limited research or platform available for the users where they could view the prediction and the investment on stocks and crypto currencies along with their market caps [5]. Most popular task in natural language processing area is sentiment classification which predicts sentiment’s opinion or emotion from a given corpus. This proposed project to predict the market with respect to the previous result of actual market along with the sentiment and emotions of the people helps the investors how, when and where to invest resulting in growth and stability of the economy in future [6].

There is a lot of work done for predicting the shifts in money market. Recently, a ton of attention-grabbing work has been done in the space of applying Machine Learning Algorithms for analyzing price patterns and predicting of money market. Most investors nowadays depend on Intelligent Trading Systems which help them in predicting prices based on various situations and conditions. Multiple researches take place which focusses on some point like Chinese sentiment analysis method which predicts the only stock market price in china [3]. Now a day’s crypto currencies have done major outbreak in market no one know when the crypto currencies will shift, some of the financial researchers had predicted the machine learning model to predict the prices of stocks and evaluating them using accuracy recall and F-1 score [2].

Zhu, M. at el. [10] focused on using a naïve bayes classifier and a linear regression model to predict the opening of stock prices for ten different companies and achieved a result of an accuracy of 52.2%. Further they concluded that the relationship between the public sentiment and stocks market movements.

Shah et al. [11] shows how current sentiment can be utilised to forecast the pharmaceutical market's changes. To forecast the movements, the authors of this paper utilised a dictionary-based sentiment analysis model that exclusively used sentiment from news. On the other hand, for our research, we focused on public sentiment, scraping for tweets about various companies throughout the previous n years. In addition, we employed a linear regression model to assess the model's performance in correctly predicting stock prices.

Chapter 3

**Methodology**

In this project we are focusing about the prediction and technical analysis of the market. Stock-money market forecasting is the method to determine the future value of company stock. Nowadays, a huge amount of valuable information related to the financial market is available on various media such as websites, twitter, Facebook, blogs and such others. In general, a stock price depends on two factors. One is fundamental factor and another one is technical factor. The fundamental factor mainly depends on the statistical data of a company. It includes reports, financial status of the company, the balance sheets, dividends and policies of the companies whose stock are to be observed. The technical factor includes the quantitative parameters like trend indicators, daily ups and downs, highest and lowest values of a day, volume of stock, indices, put/call ratios, etc. In technical factor the historical prices are considered for the forecasting. Initially the historical prices of the selected company are downloaded from the website. Various methods of stock level indicators are available to computing the stock value. Few of them are Moving Average, Stochastic RSI (Relative-Strength Index), Bollinger bands, Accumulation – Distribution, Typical Point (pivot point).

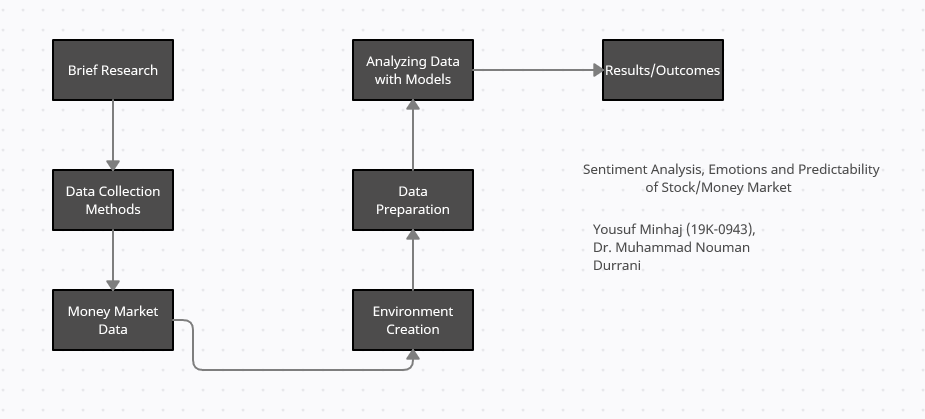


Figure : Block Diagram

Mainly the data collected from YAHOO and Crypto exchanges needs to be preprocessed to make it suitable. But there is a main problem that there is no data when the market is closed, so to overcome this problem we use simple formula Y = (xPrevious + xNext)/2 We are using two metrices which are useful for machine learning algorithms HLPCT (High Low Percentage) (HLPCT = High-low/low) PCTChange(Percentage Change) ( PCTChange = Close-open/open ).

The data collected from crypto exchange on some interval of time will be processed and the polarity and subjectivity will be calculated accordingly. Data Collected from Twitter needs to be preprocessed to make it suitable User Request the API to get Tweets from the Server. We use Twitter Api which is a REST Api the result will come in JSON format The Search API allows filtering based on language, region, geolocation and time. JSON objects that contain the tweets and their metadata. A variety of information, including username, time, location, retweets, and more. we use $ sign as a ticker to gather the most financial tweets. used TweetPy which is a wrapper for the Twitter API.

**Tools & Platform**

By analyzing the data obtained and compare the results obtained from that analysis respectively. In this project we use multiple Machine Learning Algorithms namely, Nave Bayes, Linear Regression, LSTM and ARIMA. Furthermore, after the results, we will evaluate these model’s performance using Precision, Recall and F-1 score. CV technique.

We will use the following tools and platform

1. Visual Studio
2. Python
3. Live feed API
4. Html 5
5. Jquery
6. Bootstrap

**Project Timeline**

|  |  |
| --- | --- |
| Task | TimeLine |
| Research Analysis | 1/Mar/2021 – 31/Mar/2021 |
| Data gathering Methods | 1/Apr/2021 – 04/May/2021 |
| Money Market Data | 5/May/2021 – 01/June/2021 |
| Preparing of Data | 01/June/2021 – 31/July/2021 |
| Environment Development | 05/May/2021 – 31/July/2021 |
| Executing Data Models | 01/Aug/2021 – 15/Sep/2021 |
| Analyzing Data with Model | 01/Sep/2021 – 30/Sep/2021 |
| Results/Outcomes | 01/Oct/2021 – 31/Oct/2021 |

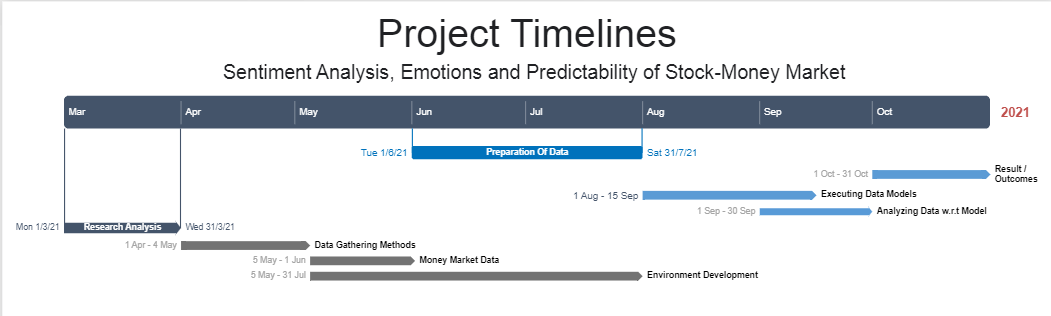


Figure : Project Timeline

**Software Requirement Specification**

**Introduction**

Today's market globally and locally in Pakistan Stock market prediction is incredibly vital within the planning of business activities. Stock worth prediction has attracted several researchers in multiple disciplines as well as computer science, statistics, economics, finance, and operations research. Recent studies have shown that the bulk quantity of online data within the property right like Wikipedia usage pattern, news stories from the thought media, and social media discussions will have gained noticeable results of investors' opinions towards money markets. The responsibility of the procedure models on exchange prediction is important because it is incredibly sensitive to the economy and might directly lead to loss. In this project named “Sentiment Analysis, Emotions and Predictability for Stock Market”, we tend to retrieve, extract, and analyze the consequences of stories' sentiments on the exchange. Our main contributions embody the event of a sentiment analysis and opinion mining lexicon for the money sector, the event of a dictionary-based sentiment analysis model along with machine learning algorithms, and therefore the analysis of the model for gauging the consequences of stories sentiments on stocks or the crypto currencies.

**Scope**

Our goal with this project is to is to provide a knowledge-intensive and computationally efficient coarse-grained analysis of Cryptocurrencies stock market values which can be analyzed by analyzing the social media comments and tweets related the stock market or crypto market date and to predict that how can it create the massive impact on the stock market as well as on the crypto market. As everyone knows that cryptocurrencies are becoming increasingly relevant in the financial world and can be considered as an emerging market. The high data availability of this market and very low barrier of entry, makes this an excellent subject that now adays people are very much interested in, particularly from social networks. This data can presumably be used to infer future human behavior, and therefore could be used to develop advantageous trading strategies [1,2] as has been shown in recent attempts to detect speculative bubbles in the cryptocurrency market using sentiment analysis [3]. Sentiment analysis has found widespread use in combination with social media, as social media is a good source of valuable and sentimental, however unstructured data itself is of little value for real world applications (IBM, 2017) and social media posts fall into this category. Therefore, sentiment analysis is the ideal tool to transform this unstructured data into tangible and processable information.

**Problem Definition**

Stock market attracts thousands of investors’ hearts from all around the world. The risk and profit of it has great charm and every investor wants to book profit from that. People use various methods to predict market volatility, such as K-line diagram analysis method, Point Data Diagram, Moving Average Convergence Divergence, even coin tossing, fortune telling, and so on. Now, all the financial data is stored digitally and is easily accessible. Availability of this huge amount of financial data in digital media creates appropriate conditions for a data mining research. The important problem in this area is to make effective use of the available data. [4]

**Social Media Sentiment Analysis**

Online social networks are now attracting a lot of attention not only from their users but also from researchers in various fields. Many researchers believe that the public mood or sentiment expressed in social media is related to financial markets. We propose to use trust among users as a filtering and amplifying mechanism for the social media to increase its correlation with financial data in the stock market. Therefore, we used the real stock market data as ground truth for our trust management system. We collected stock-related data (tweets) from Twitter, which is a very popular Micro-blogging forum, to see the correlation between the Twitter sentiment valence and abnormal stock returns. [5]

**Overall Description**

**Data Collection**

We are using two different sources for data collection in order to prediction of stock market as well as crypto market current rate.

1. YAHOO Finance API

Yahoo Finance API is a reliable source of stock market data. It also provides other financial information including market summaries, historical quotes, news feed and financial reports. The unofficial Yahoo Finance API can be accessed from Rakuten RapidAPI. [6]

YAHOO Finance API consist of the following Stock Values of Each Day:

1. Open
2. Close
3. High
4. Low
5. Twitter Search API

The Twitter's standard search API (search/tweets) allows simple queries against the indices of recent or popular Tweets and behaves similarly to. [7]

Twitter Search API consists of:

1. Tweet Id
2. Tweet Timestamp
3. Tweet Text

**Environment Creation**

Friendly Environment will be created for User which Includes the following:

1. A Dashboard (WebApp)
2. Python Runner & Python Engine

**Languages Required**

1. C# (.Net Core)

We have used C# .net core for rendering python scripts.

1. Python

Core Machine Learning Algorithms are written on python scripts which are fast and supports Data manipulation smoothly.

1. Jquery

Jquery is used to render view Realtime without refreshing the page.

1. Html

Html is used for creating view for user.

**Data Preparation**

**Data Pre-processing for YAHOO Finance**

Data Collected from YAHOO needs to be preprocessed to make it suitable.

But there is main problem that there is no data values are present in weekends and holiday or when market is closed.

To overcome the problem we use simple formula:

Y = (xPrevious + xNext)/2

We are using two metrices which are useful for machine learning algorithms

1. HLPCT(High Low Percentage)
2. HLPCT = High-low/low
3. PCTChange(Percentage Change)
4. PCTChange = Close-open/open

**Data Pre-processing for Crypto Exchange**

1. Data Collected from Exchanges needs to be preprocessed to make it suitable
2. User Request the API to get data from the Server.
3. We use binance Api which is a REST Api the result will come in JSON format

**Data Pre-processing for Twitter API**

1. Data Collected from Twitter needs to be preprocessed to make it suitable
2. User Request the API to get Tweets from the Server.
3. We use Twitter Api which is a REST Api the result will come in JSON format
4. The Search API allows filtering based on language, region, geolocation and time.
5. JSON objects that contain the tweets and their metadata.
6. A variety of information, including username, time, location, retweets, and more.
7. we use $ sign as a ticker to gather the most financial tweets.
8. used TweetPy which is a wrapper for the Twitter API

**Sentiment Analysis**

To perform the sentiment analysis of the data we use the NLKT library which is a Python library. First, we divide the data from spaces to obtain the list of individual words of the tweets, then we use each word from the tweet as feature to train the classifier. Secondly, we remove the stop words by the national language container stop words dictionary. Thirdly, we tent to manage the twitter symbols, as we know that the twitter tweets contain many symbols other than text such as ‘hashtag’. For example, the words start with a hashtag (#) will not be filtered out because they may contain any crucial information about the tweet. Then the training sets are formed in three categories such as, positive sentiment query, negative sentiment query and neutral. Then the words are filtered and moved to the classifies desired group.

**Model Analysis**

To analyze the data obtained and compare the results obtained from that analysis respectively. In this project we use multiple Machine Learning Algorithms namely, Nave Bayes, Linear Regression, LSTM and ARIMA. Furthermore, after the results, we will evaluate these model’s performance using Precision, Recall and F-1 score. CV technique.

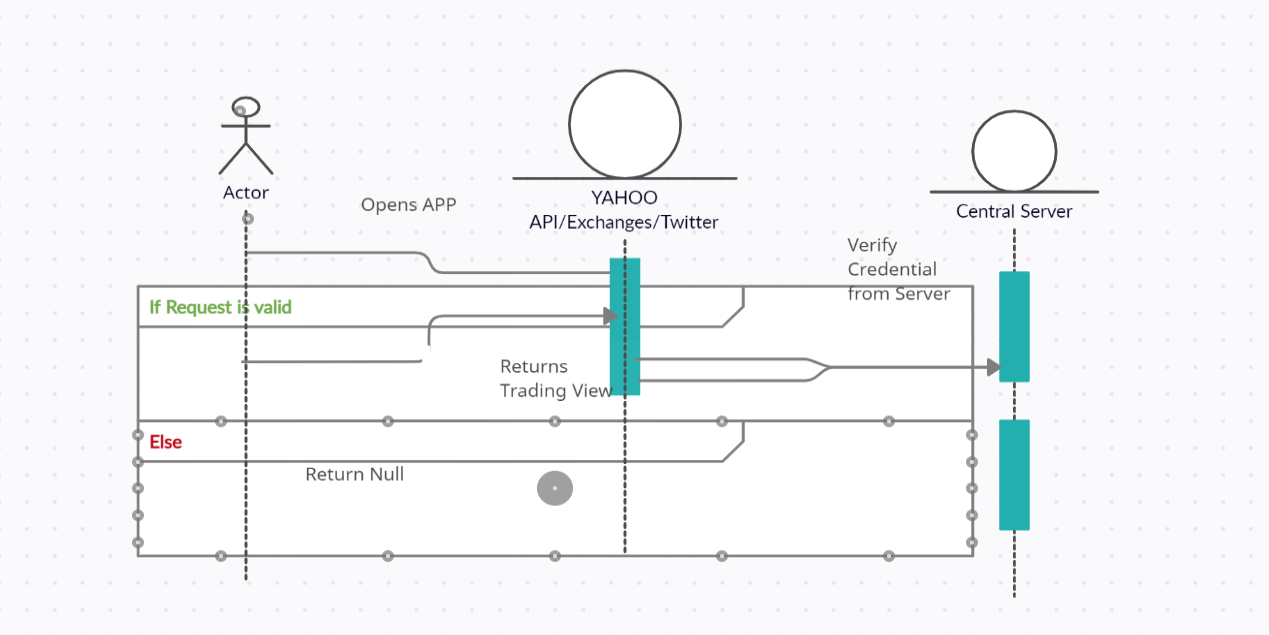


Figure : Model Overview

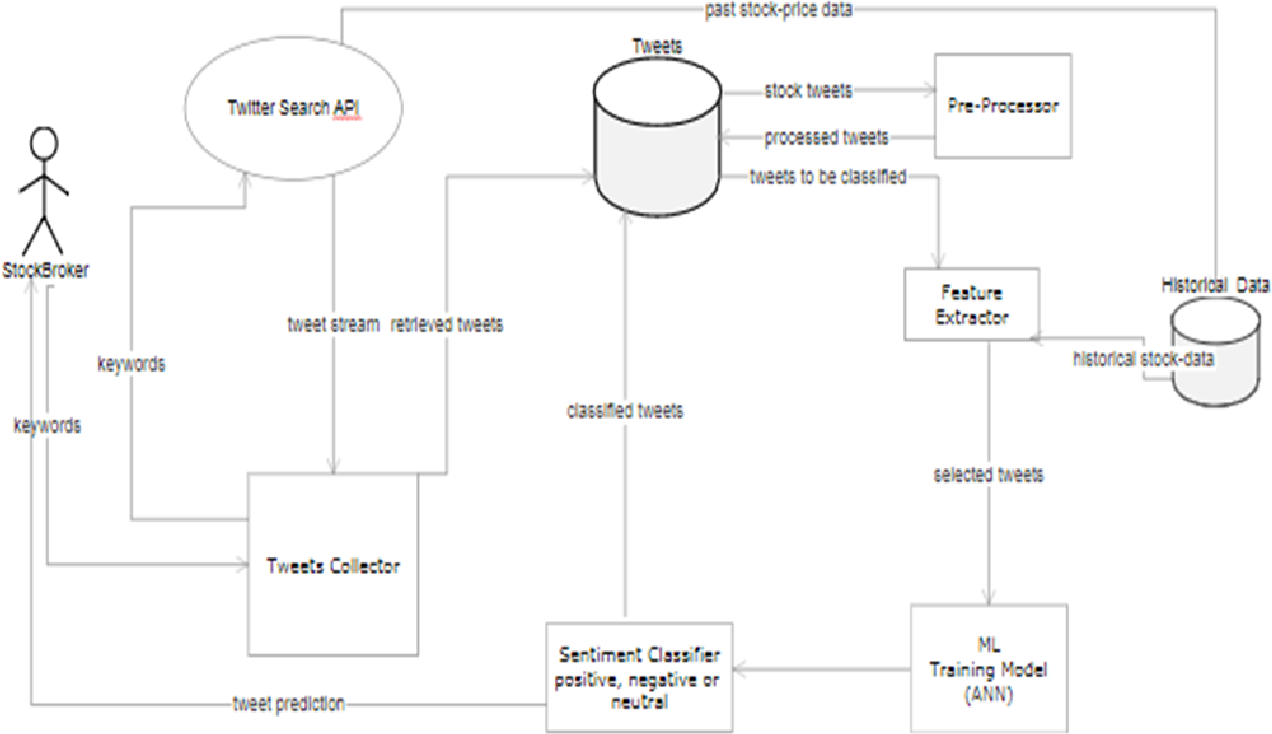


Figure : Working & Behavior

**System Features & Requirement**

**Functional Requirement**

The key Functionality of the project are described below:

1. Fluctuation Analysis of Money Market Prices.
2. Analyzing and Processing Data using ML.
3. Better view and understanding for user.
4. Predictability for every stock with trends.
5. Predictability using Live Feeds Twitter Sentiments and News.
6. Evaluation and Validation of Predicted Results.
7. Detailed Reporting

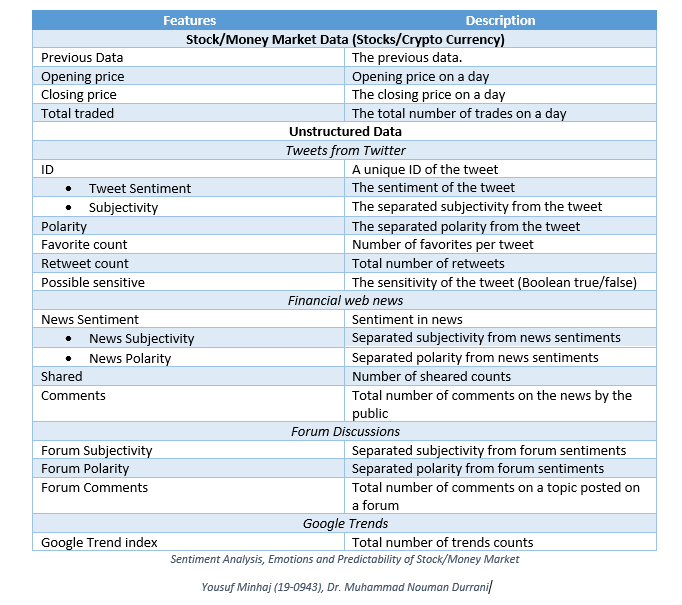
**Non-Functional Requirement**

The project is user friendly as it displays the desired result to the user.

1. System will respond quick
2. The project can run smoothly there will be no glitches in design.
3. This project can run in various web browsers which support the system environment.

**External Requirement**

This Project helps investors in efficient way, the requirement are as follows:



Chapter 4

**Software Application**

Most popular task in natural language processing area is sentiment classification which predicts sentiment’s opinion or emotion from a given corpus. This proposed project to predict the market with respect to the previous result of actual market along with the sentiment and emotions of the people helps the investors how, when and where to invest resulting in growth and stability of the economy in future.

Our system consists of dashboard which give a clear view to the user about the current opening, high, low and closing of the market of each ticker/stock. Trends will also help the user to express their thoughts by brainstorming it. After clicking in each ticker/stock our project analyzes the ticker/stock and perform the analysis on each of the product. Firstly, our project downloads the entire set of historic as well as current market of the ticker/stock from the YAHOO finance API and clean the data to shape it for processing. After dataset is imported successfully our system will download the tweets based on the ticker/stock value from Twitter API and process it using the NLTK library, once the processing is completed then its time to sort the sentiments out of it.

Once the system performs the specific task then the algorithm will take place and here in this case three algorithm runs simultaneously ARIMA Time Series Model, Long-Short Term Memory (LSTM) Model and Linear Regression Model. All the model based on the dataset which we collected from YAHOO and Twitter train the data set and predict the ticker/stock price with some round mean square error which will be displayed for the user to plan correctly.

We also uses the sentiment for the tweets to predict the ticker/stock prices for upcoming week.

**System View**

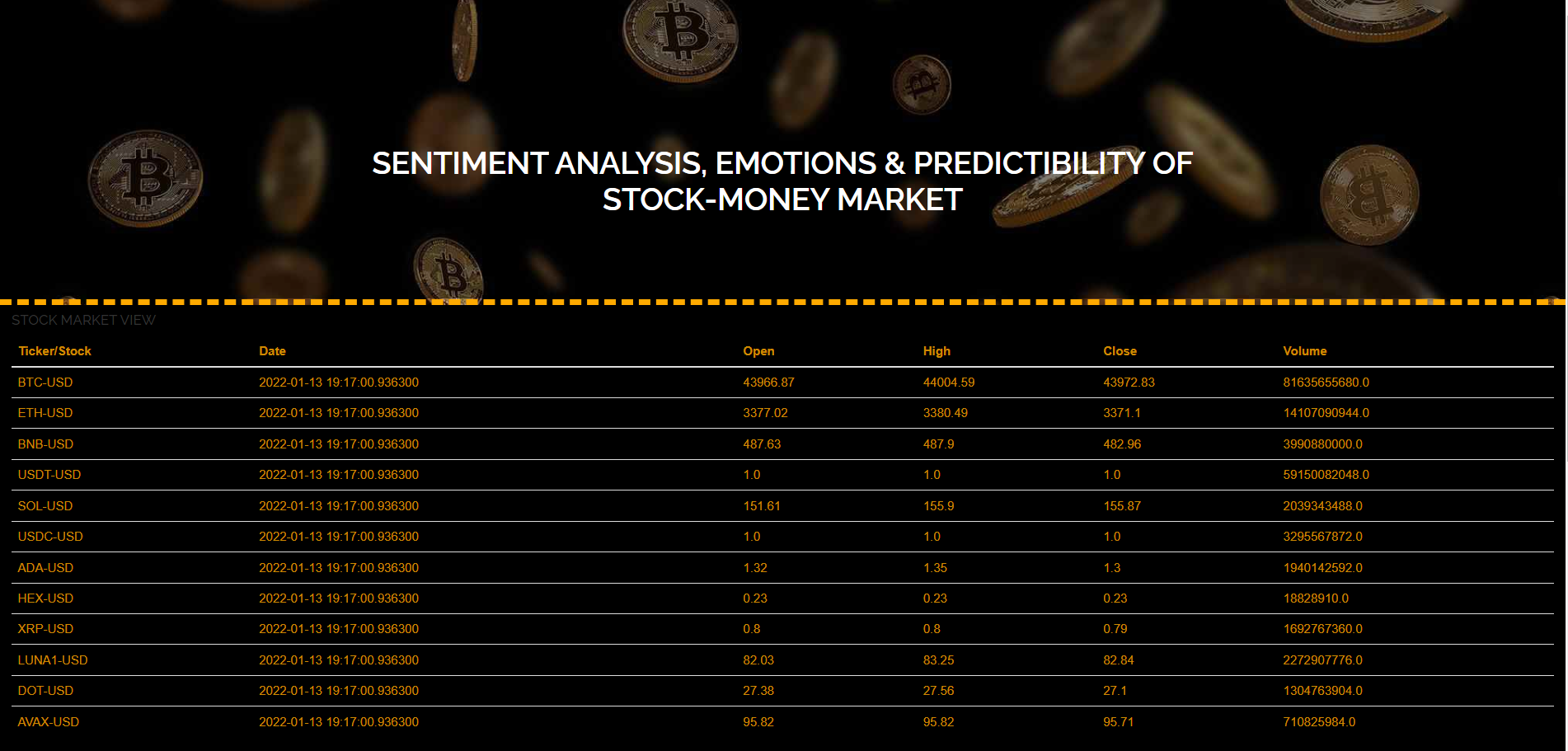
****

Figure 5 : Dashboard View & Trends

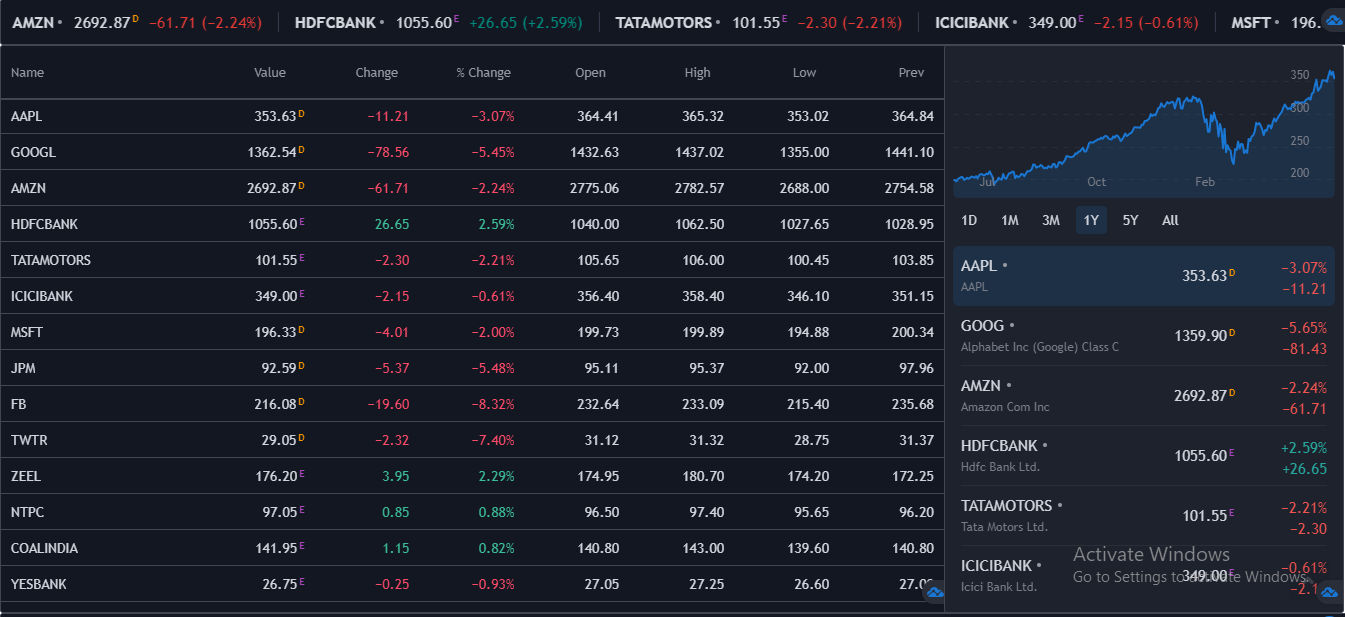
****

Figure : Dashboard View & Trends (Partial)

Our Dashboard consist of current Stock trends along with the closing position. It is feasible for the customers to look and made decisions based on the dashboard.

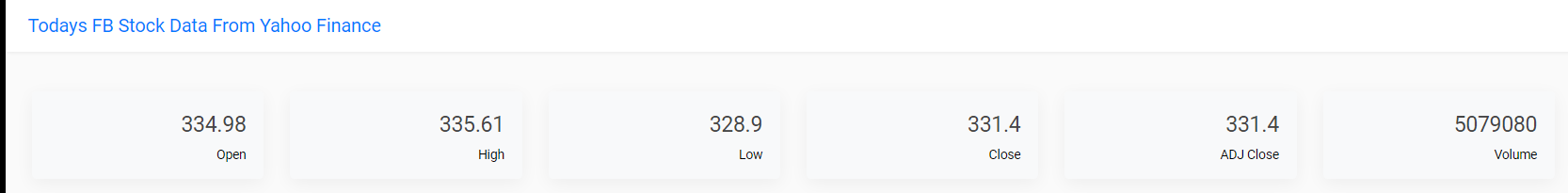


Figure 7 : Todays Ticker/Stock Information (UI)

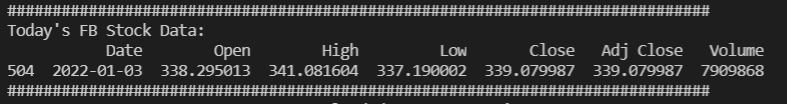


Figure : Todays Ticker/Stock Information

After Clicking to any stock ticker from the dashboard the processing started along with the design algorithm to download, predict and recommend the current and future treads with Machine Learning Algorithms.

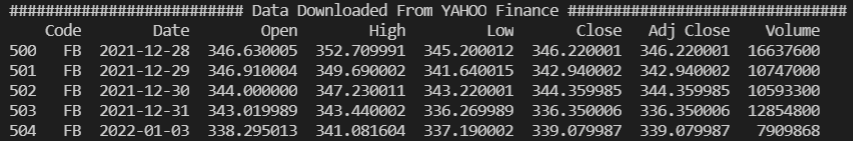


Figure : Downloaded Data View from Yahoo

Data downloaded from Yahoo Finance API for the selected ticker or stock.

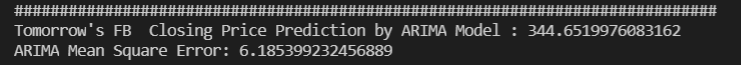


Figure : ARIMA Prediction & RMSE

Processing the downloaded data from Yahoo to ARMIA model which is a model used for time series prediction and plot the correlation and auto correlation trends/charts.

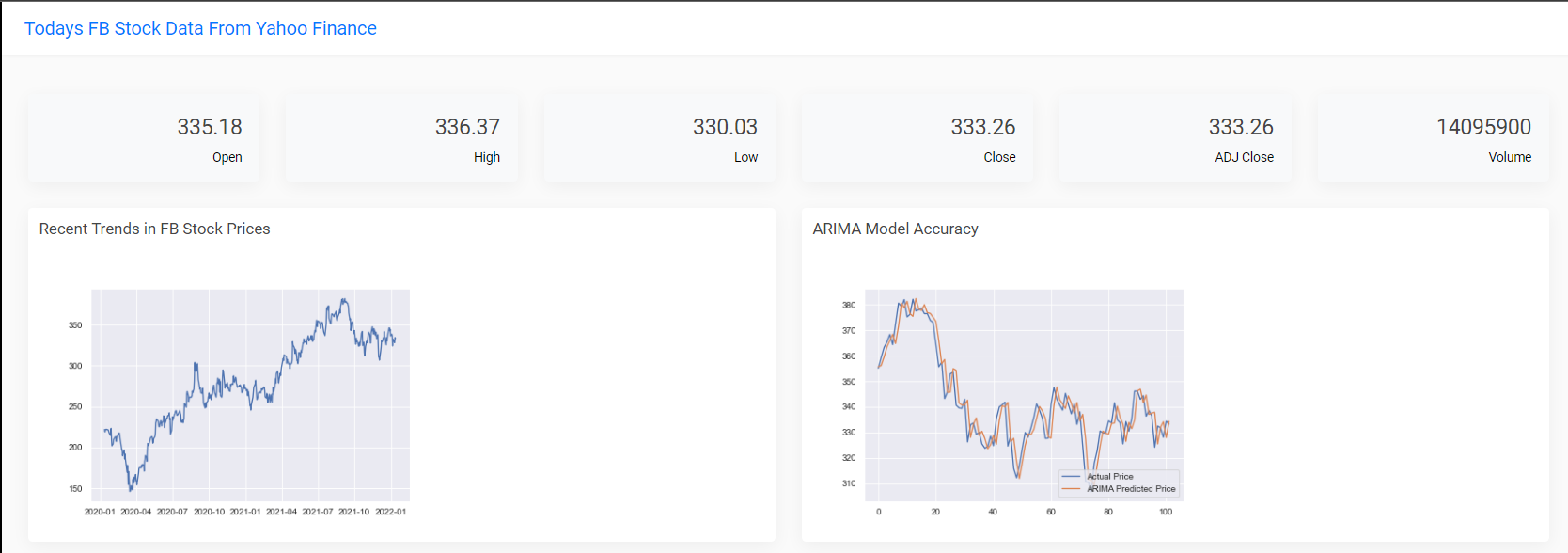


Figure 11 : Recent Trends & ARIMA Prediction (UI)



Figure : ARIMA Actual & Predicted Trend

Trends show the correlation between the actual and predicted price using ARIMA time series algorithm.

Graphical user interface, application, website

Description automatically generated

Figure 13 : LSTM, Linear Regression & Prediction For upcoming day

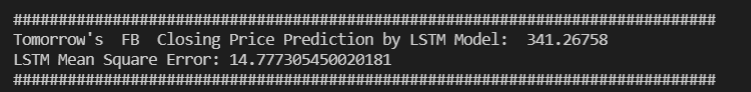


Figure : LSTM Prediction & RMSE

On second, we have inputted the same dataset for processing to Long Short Term Memory Machine Learning Algorithm.



Figure : LSTM Actual & Predicted Trend

Trends show the correlation between the actual and predicted price using LSTM algorithm which provides greater accuracy for demand forecasters which result in better decision making.

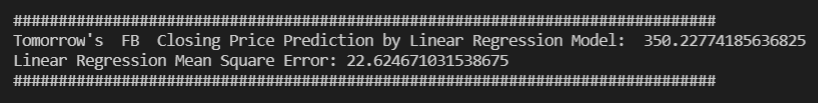


Figure : LR Prediction & RMSE

Lastly, we process the same dataset with Linear Regression Model to evaluate the results between models.

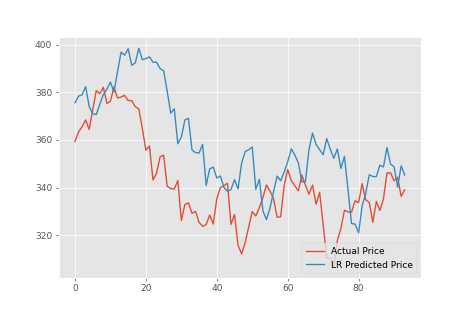


Figure : LR Actual & Predicted Trend

Trends show the interchangeably relationship with the forecasted independent or depended variable. we use this because it is more versatile and has wide applicability.

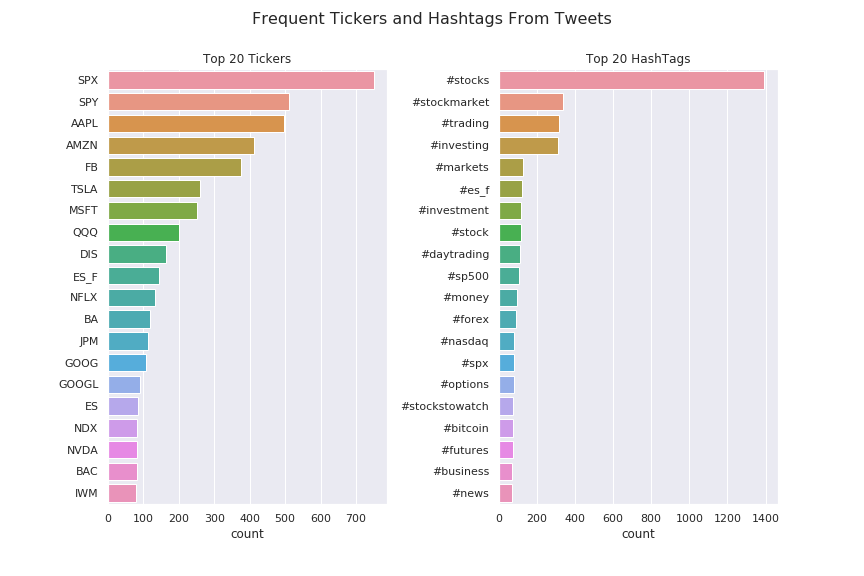


Figure : Tokenized Tickers/Stocks After Processing Tweets

We have also collected the tweets from twitter API and process using the NLTK library for data cleaning, detecting parts of speech, tokenizing and sentiment analysis of that tweets.

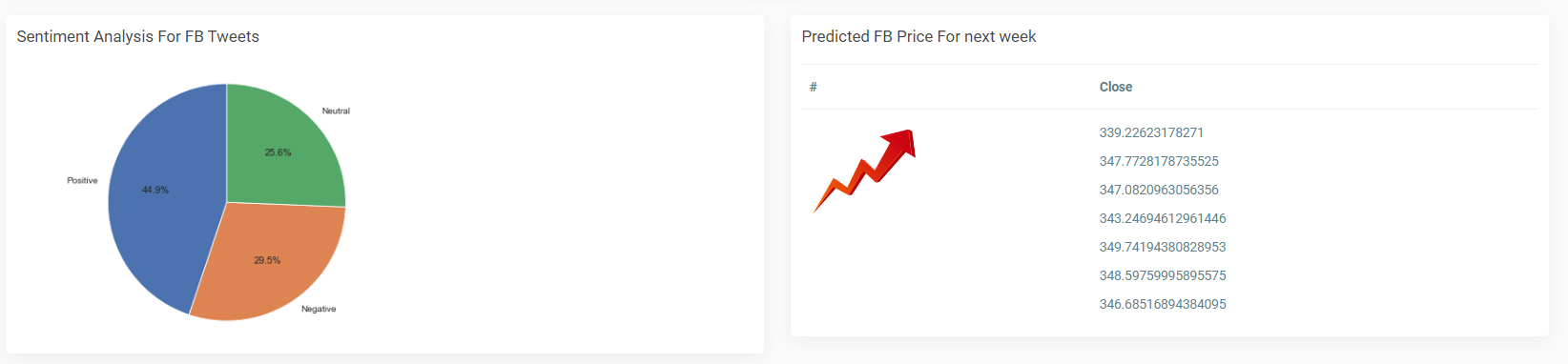


Figure : Recommendation after Sentiment Analysis

Further after performing data cleansing and detecting and tokenizing parts of speech we have concluded the forecasted price of this ticker/stock for a week.

**Key Algorithms**

**ARIMA Model**

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ARIMA TIME SERIES SECTION \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def ArimaAlgorithm\_Process(dataFrame):

        print("################# Started ARIMA Time Series Model Evaluation #################")

        uniqueValues = dataFrame["Code"].unique()

        len(uniqueValues)

        dataFrame=dataFrame.set\_index("Code")

        #for daily basis

        def dateParser(x):

            return datetime.strptime(x, '%Y-%m-%d')

        def ModelProcessARIMA(train, test):

            history = [x for x in train]

            predictions = list()

            for t in range(len(test)):

                model = ARIMA(history, order=(6,1 ,0))

                model\_fit = model.fit(disp=0)

                output = model\_fit.forecast()

                yhat = output[0]

                predictions.append(yhat[0])

                obs = test[t]

                history.append(obs)

            return predictions

        for company in uniqueValues[:10]:

            data=(dataFrame.loc[company,:]).reset\_index()

            data['Price'] = data['Close']

            dateQuantity = data[['Price','Date']]

            dateQuantity.index = dateQuantity['Date'].map(lambda x: dateParser(x))

            dateQuantity['Price'] = dateQuantity['Price'].map(lambda x: float(x))

            dateQuantity = dateQuantity.fillna(dateQuantity.bfill())

            dateQuantity = dateQuantity.drop(['Date'],axis =1)

            fig = plt.figure(figsize=(7.2,4.8),dpi=65)

            plt.plot(dateQuantity)

            plt.savefig('ActualTrends.png')

            plt.close(fig)

            quantity = dateQuantity.values

            size = int(len(quantity) \* 0.80)

            train, test = quantity[0:size], quantity[size:len(quantity)]

            #fit in model

            predictions = ModelProcessARIMA(train, test)

            #plot graph

            fig = plt.figure(figsize=(7.2,4.8),dpi=65)

            plt.plot(test,label='Actual Price')

            plt.plot(predictions,label='ARIMA Predicted Price')

            plt.legend(loc=4)

            plt.savefig('ARIMATrends.png')

            plt.close(fig)

            print()

            print("==============================================================================")

            arimaPrediction=predictions[-2]

            print("Tomorrow's",quote," Closing Price Prediction by ARIMA Model :",arimaPrediction)

            #rmse calculation

            arimaMSE = math.sqrt(mean\_squared\_error(test, predictions))

            print("ARIMA Mean Square Error:",arimaMSE)

            print("==============================================================================")

            return arimaPrediction, arimaMSE

**LSTM Model**

 #\*\*\*\*\*\*\*\*\*\*\*\*\* Long-Short Term Memory Model Section \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def LstmAlgorithm\_Process(dataFrame):

        print("################# Started LSTM Model Evaluation #################")

        #Split data into training set and test set

        trainDataset=dataFrame.iloc[0:int(0.8\*len(dataFrame)),:]

        testDataset=dataFrame.iloc[int(0.8\*len(dataFrame)):,:]

        ############# NOTE #################

        #PREDICT STOCK PRICES OF NEXT N DAYS, STORE PREVIOUS N DAYS IN MEMORY WHILE TRAINING

        # HERE N=7

        trainSet=dataFrame.iloc[:,4:5].values# 1:2, to store as numpy array else Series obj will be stored

        #select cols using above manner to select as float64 type, view in var explorer

        #Feature Scaling

        from sklearn.preprocessing import MinMaxScaler

        sc=MinMaxScaler(feature\_range=(0,1))#Scaled values btween 0,1

        trainSet\_scaled=sc.fit\_transform(trainSet)

        #In scaling, fit\_transform for training, transform for test

        #Creating data stucture with 7 timesteps and 1 output.

        #7 timesteps meaning storing trends from 7 days before current day to predict 1 next output

        X\_train=[]#memory with 7 days from day i

        y\_train=[]#day i

        for i in range(7,len(trainSet\_scaled)):

            X\_train.append(trainSet\_scaled[i-7:i,0])

            y\_train.append(trainSet\_scaled[i,0])

        #Convert list to numpy arrays

        X\_train=np.array(X\_train)

        y\_train=np.array(y\_train)

        X\_forecast=np.array(X\_train[-1,1:])

        X\_forecast=np.append(X\_forecast,y\_train[-1])

        #Reshaping: Adding 3rd dimension

        X\_train=np.reshape(X\_train, (X\_train.shape[0],X\_train.shape[1],1))#.shape 0=row,1=col

        X\_forecast=np.reshape(X\_forecast, (1,X\_forecast.shape[0],1))

        #For X\_train=np.reshape(no. of rows/samples, timesteps, no. of cols/features)

        #Building RNN

        from keras.models import Sequential

        from keras.layers import Dense

        from keras.layers import Dropout

        from keras.layers import LSTM

        #Initialise RNN

        regressor=Sequential()

        #Add first LSTM layer

        regressor.add(LSTM(units=50,return\_sequences=True,input\_shape=(X\_train.shape[1],1)))

        #units=no. of neurons in layer

        #input\_shape=(timesteps,no. of cols/features)

        #return\_seq=True for sending recc memory. For last layer, retrun\_seq=False since end of the line

        regressor.add(Dropout(0.1))

        #Add 2nd LSTM layer

        regressor.add(LSTM(units=50,return\_sequences=True))

        regressor.add(Dropout(0.1))

        #Add 3rd LSTM layer

        regressor.add(LSTM(units=50,return\_sequences=True))

        regressor.add(Dropout(0.1))

        #Add 4th LSTM layer

        regressor.add(LSTM(units=50))

        regressor.add(Dropout(0.1))

        #Add o/p layer

        regressor.add(Dense(units=1))

        #Compile

        regressor.compile(optimizer='adam',loss='mean\_squared\_error')

        #Training

        regressor.fit(X\_train,y\_train,epochs=25,batch\_size=32 )

        #For lstm, batch\_size=power of 2

        #Testing

        ###testDataset=pd.read\_csv('Google\_Stock\_Price\_Test.csv')

        real\_stock\_price=testDataset.iloc[:,4:5].values

        #To predict, we need stock prices of 7 days before the test set

        #So combine train and test set to get the entire data set

        dtTotal=pd.concat((trainDataset['Close'],testDataset['Close']),axis=0)

        testingSet=dtTotal[ len(dtTotal) -len(testDataset) -7: ].values

        testingSet=testingSet.reshape(-1,1)

        #-1=till last row, (-1,1)=>(80,1). otherwise only (80,0)

        #Feature scaling

        testingSet=sc.transform(testingSet)

        #Create data structure

        X\_test=[]

        for i in range(7,len(testingSet)):

            X\_test.append(testingSet[i-7:i,0])

            #Convert list to numpy arrays

        X\_test=np.array(X\_test)

        #Reshaping: Adding 3rd dimension

        X\_test=np.reshape(X\_test, (X\_test.shape[0],X\_test.shape[1],1))

        #Testing Prediction

        pSTPRice=regressor.predict(X\_test)

        #Getting original prices back from scaled values

        pSTPRice=sc.inverse\_transform(pSTPRice)

        fig = plt.figure(figsize=(7.2,4.8),dpi=65)

        plt.plot(real\_stock\_price,label='Actual Price')

        plt.plot(pSTPRice,label='LSTM Predicted Price')

        plt.legend(loc=4)

        plt.savefig('LSTMTrend.png')

        plt.close(fig)

        LSTMMSE = math.sqrt(mean\_squared\_error(real\_stock\_price, pSTPRice))

        #Forecasting Prediction

        forecasted\_stock\_price=regressor.predict(X\_forecast)

        #Getting original prices back from scaled values

        forecasted\_stock\_price=sc.inverse\_transform(forecasted\_stock\_price)

        predLSTM=forecasted\_stock\_price[0,0]

        print()

        print("=============================================================================")

        print("Tomorrow's ",quote," Closing Price Prediction by LSTM Model: ",predLSTM)

        print("LSTM Mean Square Error:",LSTMMSE)

        print("=============================================================================")

        return predLSTM,LSTMMSE

**LR Model**

 #\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* LINEAR REGRESSION MODEL SECTION \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def LinearRefAlgorithm\_Process(dataFrame):

        print("################# Started Linear Regression Model Evaluation #################")

        #No of days to be forcasted in future

        forcastOut = int(7)

        #Price after n days

        dataFrame['Close after n days'] = dataFrame['Close'].shift(-forcastOut)

        #New dataFrame with only relevant data

        dataFrame\_new=dataFrame[['Close','Close after n days']]

        #Structure data for train, test & forecast

        #lables of known data, discard last 35 rows

        y =np.array(dataFrame\_new.iloc[:-forcastOut,-1])

        y=np.reshape(y, (-1,1))

        #all cols of known data except lables, discard last 35 rows

        X=np.array(dataFrame\_new.iloc[:-forcastOut,0:-1])

        #Unknown, X to be forecasted

        xForecasted=np.array(dataFrame\_new.iloc[-forcastOut:,0:-1])

        #Traning, testing to plot graphs, check accuracy

        X\_train=X[0:int(0.8\*len(dataFrame)),:]

        X\_test=X[int(0.8\*len(dataFrame)):,:]

        y\_train=y[0:int(0.8\*len(dataFrame)),:]

        y\_test=y[int(0.8\*len(dataFrame)):,:]

        # Feature Scaling===Normalization

        from sklearn.preprocessing import StandardScaler

        sc = StandardScaler()

        X\_train = sc.fit\_transform(X\_train)

        X\_test = sc.transform(X\_test)

        xForecasted=sc.transform(xForecasted)

        #Training

        clf = LinearRegression(n\_jobs=-1)

        clf.fit(X\_train, y\_train)

        #Testing

        y\_test\_pred=clf.predict(X\_test)

        y\_test\_pred=y\_test\_pred\*(1.04)

        import matplotlib.pyplot as plt2

        fig = plt2.figure(figsize=(7.2,4.8),dpi=65)

        plt2.plot(y\_test,label='Actual Price' )

        plt2.plot(y\_test\_pred,label='LR Predicted Price')

        plt2.legend(loc=4)

        plt2.savefig('LRTrend.png')

        plt2.close(fig)

        LRMSE = math.sqrt(mean\_squared\_error(y\_test, y\_test\_pred))

        #Forecasting

        forcastedSet = clf.predict(xForecasted)

        forcastedSet=forcastedSet\*(1.04)

        mean=forcastedSet.mean()

        LRPred=forcastedSet[0,0]

        print()

        print("==============================================================================")

        print("Tomorrow's ",quote," Closing Price Prediction by Linear Regression Model: ",LRPred)

        print("Linear Regression Mean Square Error:",LRMSE)

        print("=============================================================================")

        return dataFrame, LRPred, forcastedSet, mean, LRMSE

**Fetching & Cleaning Tweets**

def TweetsSentiments\_Process(symbol):

        StockTicker = pd.read\_csv('Yahoo-Finance-Ticker-Symbols.csv')

        StockFullName = StockTicker[StockTicker['Ticker']==symbol]

        symbol = StockFullName['Name'].to\_list()[0][0:12]

        auth = tweepy.OAuthHandler(ct.consumer\_key, ct.consumer\_secret)

        auth.set\_access\_token(ct.access\_token, ct.access\_token\_secret)

        user = tweepy.API(auth)

    file\_name = 'tweets\_labelled\_09042020\_16072020.csv'

data = pd.read\_csv(file\_name, sep=';').set\_index('id')

data.shape

data.head()

ticker\_pattern = re.compile(r'(^\$[A-Z]+|^\$ES\_F)')

ht\_pattern = re.compile(r'#\w+')

TickerDic = collections.defaultdict(int)

ht\_dic = collections.defaultdict(int)

for text in data['text']:

    for word in text.split():

        if ticker\_pattern.fullmatch(word) is not None:

            TickerDic[word[1:]] += 1

        word = word.lower()

        if ht\_pattern.fullmatch(word) is not None:

            ht\_dic[word] += 1

            ticker\_df = pd.DataFrame.from\_dict(

    TickerDic, orient='index').rename(columns={0:'count'})\

    .sort\_values('count', ascending=False).head(20)

ht\_df = pd.DataFrame.from\_dict(

    ht\_dic, orient='index').rename(columns={0:'count'})\

    .sort\_values('count', ascending=False).head(20)

fig, ax = plt.subplots(1, 2, figsize=(12,8))

plt.suptitle('Frequent Tickers and Hashtags From Tweets', fontsize=16)

plt.subplots\_adjust(wspace=0.4)

sns.barplot(x=ticker\_df['count'], y=ticker\_df.index, orient='h', ax=ax[0])

ax[0].set\_title('Top 20 Tickers')

sns.barplot(x=ht\_df['count'], y=ht\_df.index, orient='h', ax=ax[1])

ax[1].set\_title('Top 20 HashTags')

plt.savefig('Frequent Tickers')

plt.show()

        tweets = dftweets

        #tweets = tweepy.Cursor(user.search, q=symbol, tweet\_mode='extended', lang='en',exclude\_replies=True).items(ct.num\_of\_tweets)

        TweetList = [] #List of tweets alongside polarity

        globalPolarity = 0 #Sentimental Polarity of all tweets === Sum of individual tweets

        tw\_list=[] #List of tweets only for web page

        #Count positiveSentimentitive, negativeSentimentative to plot pie chart

        positiveSentiment=0 #Num of positiveSentiment tweets

        negativeSentiment=1 #Num of negativeSentimentative tweets

        for tweet in tweets:

            count=20 #Num of tweets to be displayed on web page

            #Convert to Textblob format for assigning polarity

            tw2 = tweet.full\_text

            tw = tweet.full\_text

            #Clean

            tw=p.clean(tw)

            #print("-------------------------------CLEANED TWEET -----------------------------")

            #print(tw)

            #Replace &amp; by &

            tw=re.sub('&amp;','&',tw)

            #Remove :

            tw=re.sub(':','',tw)

            #print("-------------------------------TWEET AFTER REGEX MATCHING-----------------------------")

            #print(tw)

            #Remove Emojis and Hindi Characters

            tw=tw.encode('ascii', 'ignore').decode('ascii')

            #print("-------------------------------TWEET AFTER REMOVING NON ASCII CHARS-----------------------------")

            #print(tw)

            blob = TextBlob(tw)

            polarity = 0 #Polarity of single individual tweet

            for sentence in blob.sentences:

                polarity += sentence.sentiment.polarity

                if polarity>0:

                    positiveSentiment=positiveSentiment+1

                if polarity<0:

                    negativeSentiment=negativeSentiment+1

                globalPolarity += sentence.sentiment.polarity

            if count > 0:

                tw\_list.append(tw2)

            TweetList.append(Tweet(tw, polarity))

            count=count-1

        if len(TweetList) != 0:

            globalPolarity = globalPolarity / len(TweetList)

        else:

            globalPolarity = globalPolarity

        neutral=ct.num\_of\_tweets-positiveSentiment-negativeSentiment

        if neutral<0:

            negativeSentiment=negativeSentiment+neutral

            neutral=20

        print()

        print("==============================================================================")

        print("positiveSentimentitive Tweets :",positiveSentiment,"negativeSentimentative Tweets :",negativeSentiment,"Neutral Tweets :",neutral)

        print("==============================================================================")

        labels=['positiveSentimentitive','negativeSentimentative','Neutral']

        sizes = [positiveSentiment,negativeSentiment,neutral]

        explode = (0, 0, 0)

        fig = plt.figure(figsize=(7.2,4.8),dpi=65)

        fig1, ax1 = plt.subplots(figsize=(7.2,4.8),dpi=65)

        ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%', startangle=90)

        # Equal aspect ratio ensures that pie is drawn as a circle

        ax1.axis('equal')

        plt.tight\_layout()

        plt.savefig('SATrends.png')

        plt.close(fig)

        #plt.show()

        if globalPolarity>0:

            print()

            print("==============================================================================")

            print("Tweets Sentiment: Overall positive")

            print("==============================================================================")

            tw\_pol="Overall positiveSentimentitive"

        else:

            print()

            print("==============================================================================")

            print("Tweets Sentiment: Overall negative")

            print("==============================================================================")

            tw\_pol="Overall negativeSentimentative"

        return globalPolarity,tw\_list,tw\_pol,positiveSentiment,negativeSentiment,neutral

**Sentiments Recommendation**

def Predictions(dataFrame, globalPolarity,todayStock,mean):

        if todayStock.iloc[-1]['Close'] < mean:

            if globalPolarity > 0:

                TickIdea="Rising"

                ForcastedDecision="Need To BUY"

                print()

                print("=============================================================================")

                print("According to the Machine Learning Predictions and Sentiment Analysis of Tweets, a",TickIdea,"in",quote,"stock is expected => ",ForcastedDecision)

            elif globalPolarity <= 0:

                TickIdea="Falling"

                ForcastedDecision="Need to Sell"

                print()

                print("=============================================================================")

                print("According to the Machine Learning Predictions and Sentiment Analysis of Tweets, a",TickIdea,"in",quote,"stock is expected => ",ForcastedDecision)

        else:

            TickIdea="Falling"

            ForcastedDecision="Need to Sell"

            print()

            print("=================================================================================")

            print("According to the Machine Learning Predictions and Sentiment Analysis of Tweets, a",TickIdea,"in",quote,"stock is expected => ",ForcastedDecision)

        return TickIdea, ForcastedDecision

**Comparison**

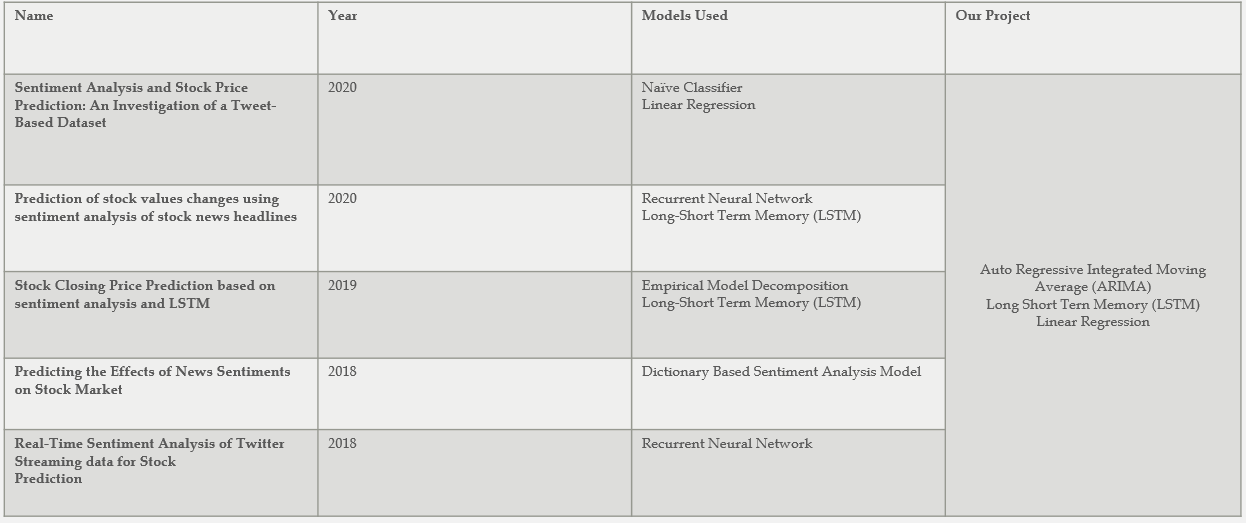


Figure 20 : Comparison

1. The regression approach turned out to be a better approach for predicting stock price as opposed to the classification approach.
2. This project shows that there is evidence of dependence between stock price and twitter sentiment.
3. This project also uses hybrid approach (lexicon base technique + machine learning algorithm).
4. Three of the well known time series models are used in this project.
5. However, this needs to be further investigated to accurately forecast a connection between social media and market behavior.

Chapter 5

**Conclusion & Future Work**

**Conclusion & Future Work**

Proposed system is a web application that allows users to conclude the opinion on the posts. Our application is simple to use and can respond to every type of user efficiently.as we see that older systems are web browsers and using older techniques. But we provide a web application that uses a hybrid technique (lexicon base technique + machine learning algorithm) which enhances the output. Our system also provides forecasting of the price and displays a friendly view to the user.

After reading and getting knowledge from the papers written on Stock Market data sentiment analysis, we conclude that the movements which behaves on sentiments are not good enough to predict the market due to the biasness factor.

In future we can contribute this to make Alert bot which automatically by the help of machine learning algorithms reminds the registered user to sell/buy the stock at a price which is forecasted. If a user wants automaticity, this system can provide the automatic buying and selling based on some threshold. Furthermore, we can add some more and advance machine learning algorithm so that the accuracy of the system increases.

1. **References**

[1] Nti, I.K., Adekoya, A.F. and Weyori, B.A., 2020. Predicting stock market price movement using sentiment analysis: Evidence from Ghana. *Applied Computer Systems*, *25*(1), pp.33-42.

[2] Larsen, J.I., 2010. *Predicting stock prices using technical analysis and machine learning* (Master's thesis, Institutt for datateknikk og informasjonsvitenskap).

[3] Chen, W., Cai, Y., Lai, K. and Xie, H., 2016, January. A topic-based sentiment analysis model to predict stock market price movement using Weibo mood. In *Web Intelligence* (Vol. 14, No. 4, pp. 287-300). IOS Press.

[4] Valencia, F., Gómez-Espinosa, A. and Valdés-Aguirre, B., 2019. Price movement prediction of cryptocurrencies using sentiment analysis and machine learning. *Entropy*, *21*(6), p.589.

[5] Steyn, D.H., Greyling, T., Rossouw, S. and Mwamba, J.M., 2020. *Sentiment, emotions and stock market predictability in developed and emerging markets* (No. 502). GLO discussion paper.

[6] Shah, D., Isah, H. and Zulkernine, F., 2019. Stock market analysis: A review and taxonomy of prediction techniques. *International Journal of Financial Studies*, *7*(2), p.26.

[7] Bharathi, S. and Geetha, A., 2017. Sentiment analysis for effective stock market prediction. *International Journal of Intelligent Engineering and Systems*, *10*(3), pp.146-154.

[8] D. Shah, H. Isah and F. Zulkernine, "Predicting the Effects of News Sentiments on the Stock Market," 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 2018, pp. 4705-4708, doi: 10.1109/BigData.2018.8621884.

[9] Jin, Z., Yang, Y. and Liu, Y., 2020. Stock closing price prediction based on sentiment analysis and LSTM. *Neural Computing and Applications*, *32*(13), pp.9713-9729.

[10] Mandapati, T. and Zhu, M., 2020. Sentiment analysis and stock price prediction : An investigation of tweet-based dataset. *Engineering Applications*, 85, pp.569-578.

[11] Dev Shah, Haruna Isah, and Farhana Zulkernine. Predicting the effects of news sentiments on the stock market,

2018