*A Project Report*

*on*

**Stock Trading Recommendation System**

*carried out as part of the course CC3270 Submitted by*

***Akruti Sinha 189303039  
Mahin Anup 189303032***

***VI semester B.Tech CCE***

*in partial fulfillment for the award of the degree*

*of*

**BACHELOR OF TECHNOLOGY**

In

**Computer & Communication Engineering**



**Department of Computer and Communication Engineering**

**School of Computing and Information Technology**

**Manipal University Jaipur**

**May 2021**

**CERTIFICATE**

This is to certify that the project entitled "**Stock Trading Recommendation System**" is a bonafide work carried out as part of the course ***(*CC3270) Minor Project Lab**, under my guidance by **Akruti Sinha, Mahin Anup** students of **B.Tech VI Semester** at the Department of Computer & Communication Engineering, Manipal University Jaipur, during the academic semester **VI**, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer & Communication Engineering, at MUJ, Jaipur.

Place: Jaipur, Rajasthan Dr. Deepak Sinwar

Date: 15/6/21 Signature of the Instructor

**DECLARATION**

I hereby declare that the project entitled “**Stock Trading Recommendation System**” submitted as part of the partial course requirements for the course **Minor Project Lab (CC3270)**, for the award of the degree of Bachelor of Technology in Computer & Communication Engineering at Manipal University Jaipur during the **VI (2021)** semester, has been carried out by us. We declare that the project has not formed the basis for the award of any degree, associateship, fellowship, or any other similar titles elsewhere.

Further, we declare that we will not share, re-submit or publish the code, idea, framework, and/or any publication that may arise out of this work for academic or profit purposes without obtaining the prior written consent of the Course Faculty Mentor and Course Instructor.

Signature of the Student: Akruti Sinha, Mahin Anup

Place: Jaipur

Date: 15/06/21

**ABSTRACT**

Recommendation systems for stock trading are of great significance to a layperson who wants to benefit from stock trading despite not having a seasoned trader's capacity or experience. Recommendation systems such as these can discover trends in stock price movements and can greatly complement a stock trader's decision-making process by creating stock recommendations based on the patterns thus discovered. It is considered to be a challenging task to trade successfully in stock markets. Significant stock trading experience and the capacity to spot trends in stock price movements are required to become a successful stock trader [6]. Therefore, a stock trading recommendation system that can assume the role of an 'expert' trader and generate stock buy/sell recommendations is of great value to a layperson who wants to profit by investing in stocks.

**TABLE OF CONTENTS**

**PARTICULARS PAGE**Certificate 1  
Candidate’s Declaration 2  
Abstract 3

1. **Introduction 6**  
   1.1 Scope of the Work  
   1.2 Product Scenarios
2. **Requirement Analysis 11**2.1 Functional Requirements   
   2.2 Non-functional Requirements   
   2.3 Use Case Scenarios   
   2.4 Other Software Engineering Methodologies
3. **System Design** **15**  
   3.1 Design Goals   
   3.2 System Architecture  
   3.3 Detailed Design Methodologies
4. **Work Done** **30**  
   4.1 Development Environment  
   4.2 Results and Discussion  
   4.3 Individual Contribution of project members 34
5. **Conclusion and Future 35**5.1. Proposed Work Plan of the project
6. **References Appendix 37**

**LIST OF FIGURES AND TABLES**

**Figure 1** Volatility of the Stock Market **8  
Table 1** Tabular Comparison of Existing Literature **13-14  
Figure 2** Backend Algorithm **16  
Figure 3** System Architecture **17  
Figure 4** Analysis of stock data from 2002 to 2020 **19  
Figure 5** Actual vs Predicted for FB Prophet **20  
Figure 6** Architecture of the LSTM **21  
Figure 7** Model Design of LSTM **21  
Figure 8** Actual vs Predicted for LSTM model **22  
Figure 9**  Actual vs Predicted for Arima Model. **23  
Figure 10** Actual vs Predicted for Linear Regression. **24  
Figure 11** Design of the LSTM model(NLP) **25  
Figure 12** Accuracy of LSTM model(NLP) **25  
Figure 13** Classification report of Naive Bayes **26  
Figure 14** Random Forest algorithm **27  
Figure 15** Classification report of random forest **27  
Figure 16** Accuracy of Logistic Regression **28  
Figure 17** Classification report of KNN **28  
Figure 18** Classification report of MLP classifier **29  
Table 2** Project Packages  **30-33**

**INTRODUCTION**

**1.1 Scope of the Work**

To those who want to trade in stocks but are constrained by their limited knowledge of stock market dynamics, recommendation systems that can tell the user when to buy and sell stocks can be of great assistance. Due to the nonlinear nature of movements in stock prices, it is considered an extremely daunting task. This recommendation system is responsible for identifying stock price trends on its own, enabling even a layman user who is not well versed in stock market behavior to trade consistently profitably. In general, stock market analysis is split into two parts - Fundamental Analysis and Technical Analysis.

* Based on its current business environment and financial performance, Fundamental Analysis involves analyzing the future profitability of the company.
* On the other hand, Technical Analysis involves reading the charts and using statistical evidence to recognize stock market trends.

This recommendation system aims to work on Technical Analysis which involves both Qualitative and Quantitative approaches. We also aim to assist the decisions of stock market traders, private investors, and investment managers by recommending investments in a group of equity stocks where there is clear evidence of potential gains from certain transactions.

Several individuals and scientific researchers have successfully used machine-learning methods. The neural network is a smart data mining technique that has been used by researchers in different fields over the last decade. In the current marketplace, stock market data prediction and assessment play a significant role.

A stock market is a place where a company's shares or stocks are traded. It is divided into two parts:

* the primary market
* the secondary market

The primary market is where new developments are launched in the market via Initial Public Offerings (IPOs).

Investors start trading equities that they already hold on the secondary market. The stock market has highly volatile and non-linear time-series data. A time series is a collection of data that is measured over time to determine the status of an activity. For stock market prediction, linear models such as AR, ARMA, and ARIMA have historically been used. The only issue with these models is that they only work for specific time-series data, i.e. the framework outlined for one company will not accomplish the same results well for another. [7] Stock market predictions are riskier than forecasting in other sectors due to the ambiguous and unpredictable nature of the stock market.

Stock exchanges are secondary markets where existing stockholders can sell their shares to potential buyers. It's vital to remember that organizations listed on stock exchanges don't buy and sell their stock on a regular basis (businesses may engage in stock buybacks8 or issue additional shares9, but these aren't day-to-day operations and typically take place outside of an exchange's framework). So, when you buy stock on the stock market, you're not buying it from the company; instead, you're buying it from another shareholder. When you sell your stock, you don't sell it back to the company; instead, you sell it to another investor.

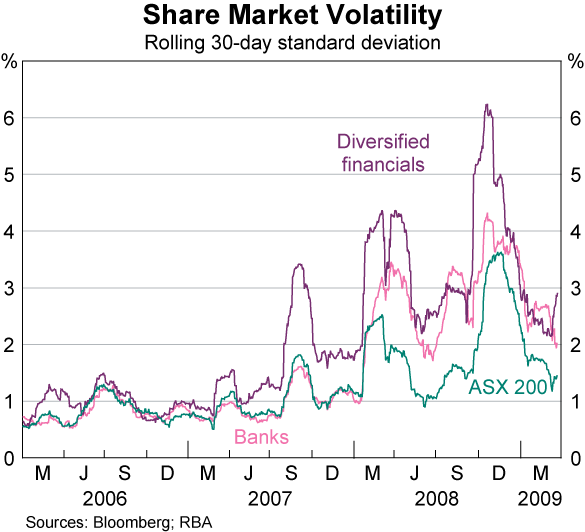


Fig.1 This chart showcases how volatile the stock market is

It is one of the primary reasons for the difficulty in predicting the stock market. This is where the use of deep-learning models in financial forecasting comes into play. The use of machine learning algorithms in DL models gave rise to the term "deep neural network." It is also known as ANN. ANNs are good approximators because they can learn and generalize from experience.[7]

We trained a deep learning model with Long Short Term Memory (LSTM) to analyze stock sentiment. Natural language processing (NLP) converts words (text) into numbers, which are then used to train an AI/ML model to make projections. In this project, we developed a machine learning model to analyze hundreds of Twitter tweets and news articles to forecast people's mentality toward a specific company or stock. The methodology could be used to instantaneously understand viewpoints from public Twitter posts, which could then be used as a factor when making stock buy/sell decisions.

DL is designed specifically for dealing with large amounts of data and performing complicated tasks where autonomous learning is required. It may be enticing to investors aiming to better their trading process because of its potential to identify complicated patterns from the data. Furthermore, DL and, in particular, LSTM appear to be a good choice from a linguistic standpoint, given their ability to remember previous words and phrases.

Sentiment analysis, which is concerned with the explanation and categorization of emotions within various sources of text data, is one of the most important NLP techniques used in financial prediction.

It is a research area that has been revitalized in the last years as a result of the advent of social media and the supply of inexpensive computing power. Market feelings, like goods and services, impact information flow and trading; thus, trading firms believe in financial gain based on price trend forecasts influenced by emotions in financial news. Is it possible to obtain predictive ability in the stock market depending on them? Even though the forecasting models published in the research papers have not been able to profit in the longer term, many hypotheses and meaningful observations have been drawn from the data of the capital markets. [8]

Many models have been developed and used in recent times, each with its very own set of advantages and disadvantages. Incredibly complex models, in particular, perform poorly, whereas simpler existing methods rely on strong hypotheses, such as a Gaussian distribution, that does not always apply in actual cases. Deep learning appears to be the best fit for this purpose because it can analyze large amounts of data, which NLP requires to understand the context and syntactical constructions.

**1.2 Product Scenarios**

Stock market analysis can be a difficult, multifaceted endeavor, but artificial intelligence and machine learning can help make it easier. Data processing, data classification, stock analysis, and pattern detection can all be aided by artificial intelligence and machine learning. Because studies show that the number of stock investors has expanded dramatically over the last decade, the timing is ripe for products that anticipate stock values. anywhere in between, it provides firewalls and intrusion protection, VPN and network access control, network traffic analysis, and secure web gateways.

In addition, very few studies have been carried out on the effectiveness of recommendation schemes in India, the sixth-largest economy in the world and home to one of the largest exchanges worldwide: the Bombay Stock Exchange (BSE).

Recommendation systems that can advise users on when to purchase and sell stocks can be extremely beneficial to those who want to trade stocks but are limited by their understanding of market dynamics.

**REQUIREMENT ANALYSIS**

**2.1 Functional Requirements**

Our functional requirements include a dataset from Kaggle containing NIFTY-50 Stock Market Data (2000 - 2020) - it contains stock prices.

NSE India data for the fifty stocks in the NIFTY-50 index. We used HTML, CSS, and JavaScript for the front end of the web design. The Flask framework is used for the backend. We also used the Heroku application to host the website. Python is used as the programming language, and the libraries are NumPy, Pandas, NLTK, Beautiful Soup, Matplotlib, and Scikit-learn.

NumPy is a Python library that is used to work with arrays. It also includes functions for working with linear algebra, the Fourier transform, and matrices.

Travis Oliphant created NumPy in 2005. It is an open-source project that you are free to use. NumPy is an abbreviation for Numerical Python.

Pandas is a Python library that is used to work with data sets. It includes tools for data analysis, cleaning, exploration, and manipulation. Wes McKinney created the name "Pandas" in 2008 as a reference to both "Panel Data" and "Python Data Analysis."

The Natural Language Toolkit, or NLTK for short, is a collection of libraries and programs written in Python for symbolic and statistical natural language processing in English. The Natural Language Toolkit (NLTK) is a collection of libraries and programs for statistical language processing. It is one of the most powerful NLP libraries, containing packages for making machines understand human language and respond appropriately to it.

Beautiful Soup is a Python package that allows you to parse HTML and XML documents. It generates a parse tree for parsed pages, which can be used to extract data from HTML and is useful for web scraping. It integrates with your preferred parser to provide idiomatic methods of navigating, searching and modifying the parse tree. It frequently saves programmers hours or even days of work.

Matplotlib was created by John Hunter (1968-2012), who, along with its many contributors, put in an incalculable amount of time and effort to create a piece of software that is used by thousands of scientists worldwide. Matplotlib is a Sponsored Project of NumFOCUS, a 501(c)(3) charitable organization in the United States. NumFOCUS provides fiscal, legal, and administrative support to Matplotlib to ensure the project's health and sustainability.

Scikit-learn (formerly scikits.learn and also known as sklearn) is a free Python machine learning library. [3] It includes support vector machines, random forests, gradient boosting, k-means, and DBSCAN as classification, regression, and clustering algorithms, and is designed to work with the Python numerical and scientific libraries NumPy and SciPy. Scikit-learn is primarily written in Python and heavily relies on NumPy for high-performance linear algebra and array operations. In addition, to improve performance, some core algorithms are written in Cython.

A Cython wrapper around LIBSVM is used to implement support vector machines; a similar wrapper around LIBLINEAR is used to implement logistic regression and linear support vector machines. Extending these methods with Python may be impossible in such cases.

Scikit-learn works well with a variety of other Python libraries, including Matplotlib and plotly for plotting, NumPy for array vectorization, Pandas data frames, SciPy, and many others.

**2.2 Non Functional Requirements**

There are no non-functional requirements.

**2.3 Use Case Scenarios**

For example, through a collaboration with a company, someone has integrated artificial intelligence and machine learning into its sensors. They can now carve files in transit across the network and upload them for review to the company malware conviction engine. The solution easily offers a "conviction" on any file that is suspected of being malicious. The solution then displays not only the alert but also the background associated with that file on the computer. The file name, hash values, and transfer protocol are all included in this. Furthermore, the console displays information from the company’s engine, such as threat scoring and the actions that caused the file to be flagged as malicious.

The solution not only informs the analyst about a threat but also provides background for how it got there, which is a higher-level advantage for security operations. This allows security professionals to address the current issue while also strengthening network security to avoid similar accidents in the future.

**2.4 Literature Review**

The stock trading recommendation systems will help improve the innovative technology available today that can contribute to more succinct decisions. It is important to understand their strengths and weaknesses as well as areas that require further study.

| **Study** | **Advantages** | **Disadvantages** |
| --- | --- | --- |
| Stock market prediction using machine learning techniques [1]. | ● Combines results from historical data, news, and Twitter feed.  ● Uses technical analysis like ARIMA and SMA. | ● This approach has chances of inaccuracy for market scenarios not covered in training data |
| An autonomous trader agent for the stock market based on an online sequential extreme learning machine ensemble [2]. | ● Focuses on short-term gains.  ● Produces results better than research proposed before. | ● Improvements can be made on choosing more features and making it more flexible |
| Machine Learning in Prediction of Stock Market Indicators Based on Historical Data and Data from Twitter Sentiment Analysis [3]. | ● States that the addition of Twitter sentiment analysis does not add any valuable information.  ● Take news feeds into consideration. | ● The initial results indicate that the addition of information from Twitter does not increase accuracy.  ● The training period and sentiment analysis algorithm need to be improved. |
| Machine Learning Techniques and Use of Event Information for Stock Market Prediction: A Survey and Evaluation [7] | ● Provides great insight into the proper implementation of sentiment analysis.  ● Discusses the use of feed-forward can everal neural networks. | ● Based on only short time frame predictions and recommendations of stock |

Table 1: Tabular Comparison of Existing Literature

**SYSTEM DESIGN**

**3.1 Design Goals**

Investors start trading equities that they already hold on the secondary market. The stock market has highly volatile and non-linear time-series data. A time series is a collection of data that is measured over time to determine the status of an activity. For stock market prediction, linear models such as AR, ARMA, and ARIMA have historically been used. The only issue with these models is that they only work for specific time-series data, i.e. the framework outlined for one company will not accomplish the same results well for another. [7] Stock market predictions are riskier than forecasting in other sectors due to the ambiguous and unpredictable nature of the stock market.

The first stage includes selecting a Machine Learning (ML) or Deep Learning (DL) model to forecast market prices. The final model will be selected after comparing the accuracy of ML models like Lasso Regression, Ridge Regression discussed in [5] and DL models like Long Short-term memory (LSTM) discussed in [6]. When the user enters the name of the stock then using the Yahoo Finance stock price API it gives back the recent stock prices in the form of a CSV file, then the CSV is loaded to the program and is put into the algorithm that returns the predicted price of the stock for the next 7 days. In the next stage, we would scrape the web for the stock's latest news and then use Natural Language Processing (NLP) to conduct sentiment analysis. This will assist us in deciding whether the news is positive or negative for the stock. We now have two separate variables influencing the recommendation of the stock. A Fuzzy logic module will be used to take all these variables into account and give us a final recommendation. The last stage of the project will be to integrate the entire recommendation system into an interactive website.

**3.2 System Architecture**

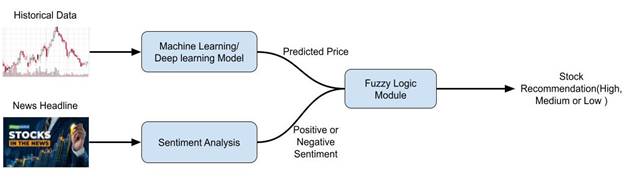
The proposed strategy shall include both quantitative and qualitative analysis of the stock market trends. The first stage includes selecting a Machine Learning (ML) or Deep Learning (DL) model to forecast market prices. The final model will be selected after comparing the accuracy of ML models like Lasso Regression, Ridge Regression discussed in [5] and DL models like Long Short-term memory (LSTM) discussed in [6]. In the next stage, we would scrape the web for the stock's latest news and then use Natural Language Processing (NLP) to conduct sentiment analysis. This will assist us in deciding whether the news is positive or negative for the stock. We now have two separate variables influencing the recommendation of the stock. A Fuzzy logic module will be used to take all these variables into account and give us a final recommendation. The last stage of the project will be to integrate the entire recommendation system into an interactive website. Fig. 1.1 showcases the proposed methodology in a concise format.

Fig.2 Backend Algorithm

When the user enters the name of the stock then using the Yahoo Finance stock price API it gives back the recent stock prices in the form of a CSV file, then the CSV is loaded to the program and is put into the algorithm that returns the predicted price of the stock for the next 7 days. It also shows a graph that shows the actual vs predicted graph and the rmse value of the algorithm. Simultaneously using the Twitter API we get the top 300 tweets about the stock in the form of JSON. It is then given to the algorithm and it returns the number of tweets that have a polarity of positive and negative. From the numbers, we can calculate the percentage of positive and negative tweets. Now we combine the results of the forecast prices and the sentiment analysis to give a recommendation for the stock, either to buy or sell the stocks. All this information is sent back to the frontend using JSON and represented intuitively can information several so that the user can understand the content easily.

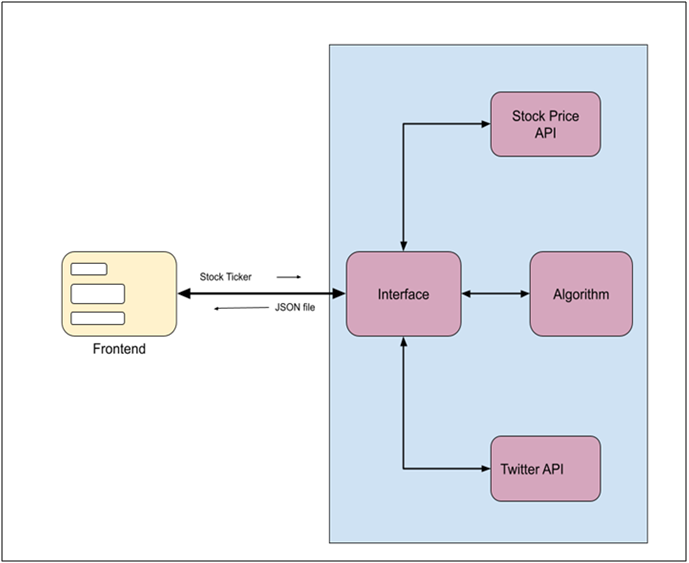


Fig 3: System Architecture

The recommendation may be associated with a potential risk. For example, when recommending stocks for purchase, users may wish to be risk-averse, preferring stocks that have lower expected growth, but also a lower risk of collapsing. On the other hand, users may be risk-seeking, preferring stocks that have a potentially high, even if less likely, profit. In such cases, we may wish to evaluate not only the (expected) value generated from a recommendation but also to minimize the risk. The standard way to evaluate risk-sensitive systems is by considering not just the expected utility, but also the utility variance.

**3.3 Detailed System Methodologies**

The proposed strategy included both quantitative and qualitative analyses of the stock market trend.

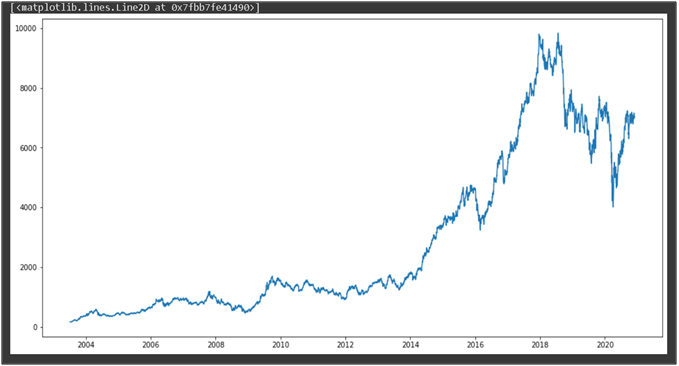
In general, stock market analysis is split into two parts - Fundamental Analysis and Technical Analysis.

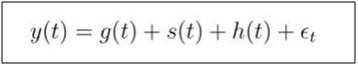
● Based on its current business environment and financial performance, Fundamental Analysis involves analyzing the future profitability of the company.

● On the other hand, Technical Analysis involves reading the charts and using statistical evidence to recognize stock market trends.

**Phase 1**

The first phase included selecting a Machine Learning (ML) or Deep Learning (DL) model with the best accuracy to forecast market prices. For the training of the model, we collected everyday stock prices from 2002 to 2020 using the yahoo finance website. For the purpose of training, we take an example stock like Maruti, ITC, TESLA, etc.

Fig 4: Analysis of stock data from 2002 to 2020

After preprocessing the data we feed it into the first model that is the Facebook Prophet algorithm. It is a time series model with three main model components: trend, seasonality, and holidays. They are combined in the following equation:

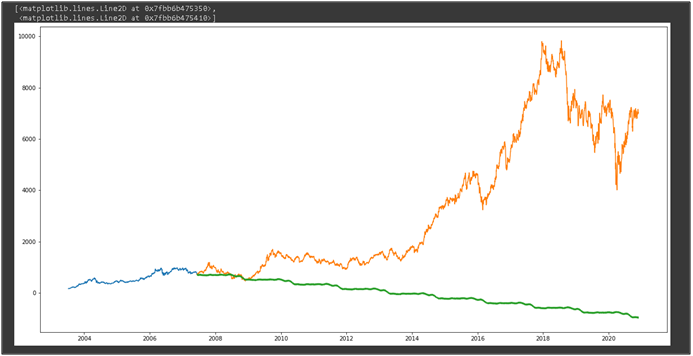
g(t): piecewise linear or logistic growth curve for modeling non-periodic changes in time series.

s(t): periodic changes (e.g. weekly/yearly seasonality).

h(t): effects of holidays (user-provided) with irregular schedules.

εt: error term accounts for any unusual changes not accommodated by the model.

After training and testing the model we got an RMSE value of 98.97**.**

Fig 5: Actual vs Predicted for FB Prophet

The second model we implemented is the LSTM Convolutional neural network. LSTMs are widely used for sequence prediction problems and have proven to be extremely effective. The reason they work so well is that LSTM is able to store past important information and forget the information that is not. LSTM has three gates:

* The input gate: The input gate adds information to the cell state.
* The forget gate: It removes the information that is no longer required by the model.
* The output gate: Output Gate at LSTM selects the information to be shown as output.

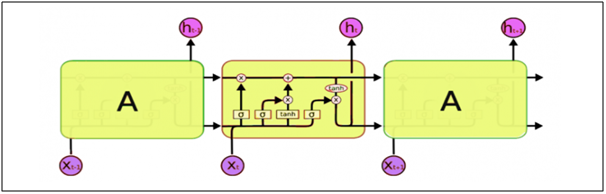
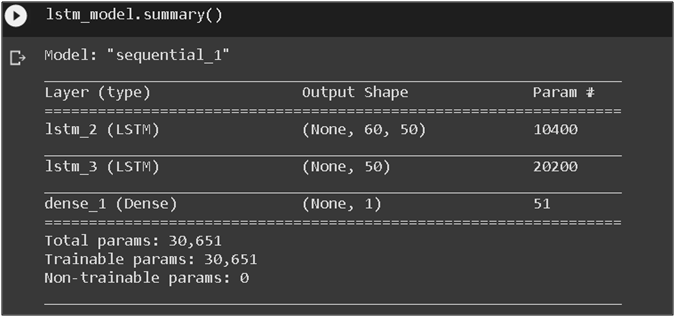
Fig 6: Architecture of the LSTM

Fig 7: Model Design of LSTM

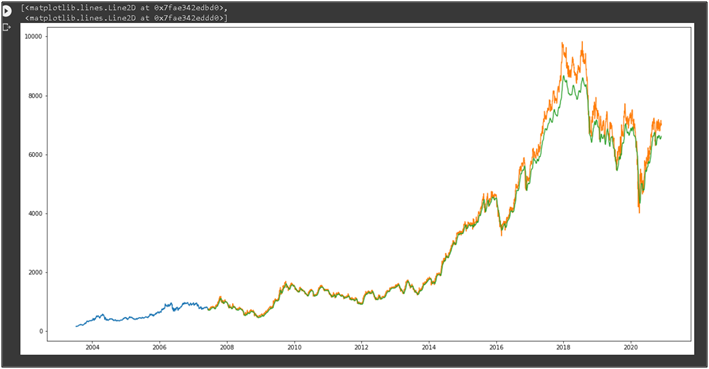
After training and testing the model we got an RMSE value of 11.77

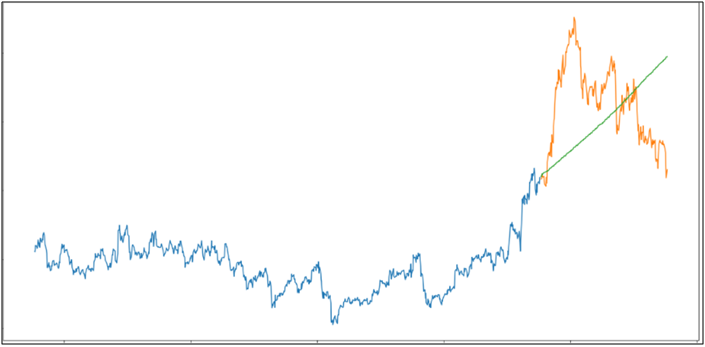
Fig 8: Actual vs Predicted for LSTM model

The third model that we implemented was the Auto-Arima model. ARIMA is a very popular statistical method for time series forecasting. ARIMA models take into account the past values to predict the future values. There are three important parameters in ARIMA

* p (past values used for forecasting the next value).
* q (past forecast errors used to predict the future values).
* d (order of differencing).

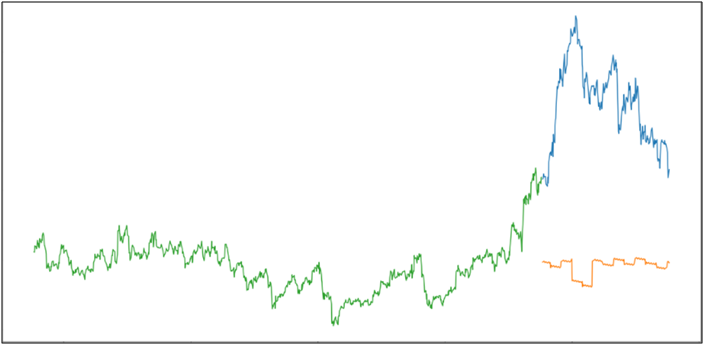
Parameter tuning for ARIMA consumes a lot of time. So we will use auto ARIMA which automatically selects the best combination of (p,q,d) that provides the least error.

After training and testing the model we got an RMSE value of 40.2

Fig 9: Actual vs Predicted for Arima Model.

The fourth algorithm we implemented was the Linear Regression algorithm. The most basic machine learning algorithm that can be implemented on this data is linear regression. The linear regression model returns an equation that determines the relationship between the independent variables and the dependent variable.

The equation for linear regression can be written as:  
After training and testing the model we got an RMSE value of 121.11.

Fig 10: Actual vs Predicted for Linear Regression.

**Phase 2**

The second phase of the project to compare different NLP models and select the best model for the sentiment analysis of tweets/news. We used a data set of stock tweets with 35k positive and 20k negative tweets. After the preprocessing of the data, we implemented 6 different algorithms for sentiment analysis which were:- LSTM, Naive Bayes, Random Forest, KNN, Logistic Regression, MLP classifier.

The first model that we used was the LSTM model. We used the base architecture of the model that we used to forecast the prices but made changes to different layers and hyperparameters to make it optimized for sentiment analysis.

Fig 11: Design of the LSTM model(NLP)

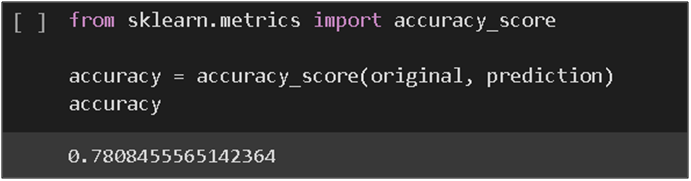
After testing an accuracy of 78.08% was achieved.

Fig: 12 Accuracy of LSTM model(NLP)

The second model that was implemented is the Naive Bayes. The Naive Bayes algorithm is a supervised learning algorithm, which is based on the Bayes theorem and used for solving classification problems. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

Where,

* P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.
* P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.
* P(A) is Prior Probability: Probability of hypothesis before observing the evidence.
* P(B) is Marginal Probability: Probability of Evidence.

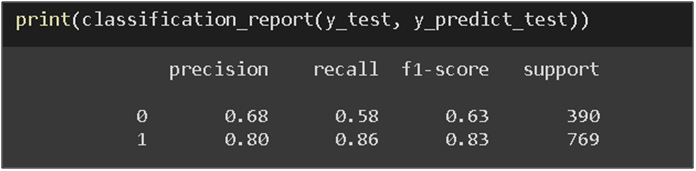
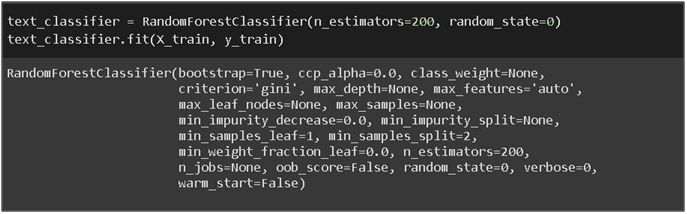
With this classifier, we achieved an accuracy of 86%, a precision of 80%, and an f1-score of 83%

Fig: 13 Classification report of Naive Bayes

The next algorithm that was implemented is the Random Forest algorithm. Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. In this project, the no. of estimators was equal to 200 with a random state equal to 0.

Fig: 14 Random Forest algorithm

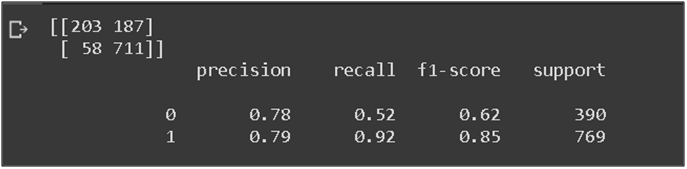
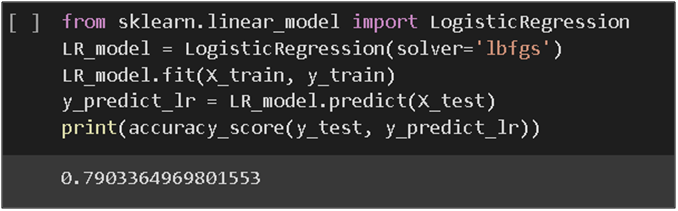
With this classifier, we achieved an accuracy of 78%, precision of 79%, and an f1-score of85%

Fig: 15 Classification report of random forest

The next algorithm that was implemented is the logistic regression algorithm. With this classifier, we achieved an accuracy of 79%.

Fig:16 Accuracy of Logistic Regression

The next algorithm we implemented was the KNN algorithm. The K-NN algorithm assumes the similarity between the new case/data and available cases and puts the new case into the category that is most similar to the available categories. K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well-suited category by using K- NN algorithm.

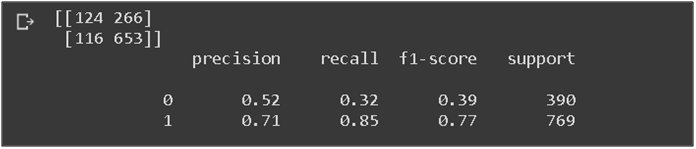
With this classifier, we achieved an accuracy of 69%, precision of 71%, and an f1-score of77%

Fig 17: Classification report of KNN

The next classifier that was implemented is the Multilayer Perceptron. They are composed of one or more layers of neurons. Data is fed to the input layer, there may be one or more hidden layers providing levels of abstraction, and predictions are made on the output layer, also called the visible layer.

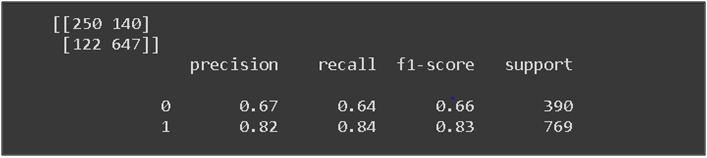
With this classifier, we achieved an accuracy of 77.3%, a precision of 82%, and an f1-score of 83%.

Fig:18 Classification report of MLP classifier

After comparing different ML models we have selected the LSTM model for the price forecasting because it gave the minimum RMSE value and the Naive Bayes classifier for the sentiment analysis of stock tweets/news as it gave the highest accuracy of all.

**WORK DONE**

**4.1 Development Environment**

Languages – Python, HTML, CSS, JavaScript.

Frameworks – Flask, Bootstrap.

IDE – PyCharm, Google Collaboratory.

Hosting platform – Heroku.

Hardware – 8GB Ram, Radeon AMD 360 graphics card, core i5 8th gen, and NVIDIA cloud GPU.

The two main IDEs used were the Google Collab for training-testing ML/DL models and PyCharm for creating and integrating the frontend and backend.

| Flask | 1.1.2 | cached-property | 1.5.2 |
| --- | --- | --- | --- |
| Jinja2 | 2.11.3 | cache tools | 4.2.2 |
| Keras | 2.4.3 | certifi | 2020.12.5 |
| Keras-Preprocessing | 1.1.2 | charsetthe | 4.0.0 |
| Markdown | 3.3.4 | click | 7.1.2 |
| MarkupSafe | 1.1.1 | constants | 0.6.0 |
| Pillow | 8.2.0 | cycler | 0.10.0 |
| PySocks | 1.7.1 | flatbuffers | 1.12 |
| PyYAML | 5.4.1 | gast | 0.3.3 |
| Werkzeug | 1.0.1 | google-auth | 1.30.0 |
| absl-py | 0.12.0 | google-auth-oauthlib | 0.4.4 |
| aiohttp | 3.7.4 | google-pasta | 0.2.0 |
| alpha-vantage | 2.3.1 | grpcio | 1.32.0 |
| astunparse | 1.6.3 | h5py | 2.10.0 |
| async-timeout | 3.0.1 | idna | 2.10 |
| Attrs | 21.2.0 | importlib-metadata | 4.0.1 |
| Idna | 2.10 | rsa | 4.7.2 |
| importlib-metadata | 4.0.1 | scikit-learn | 0.24.2 |
| itsdangerous | 1.1.0 | scipy | 1.6.3 |
| Joblib | 1.0.1 | setuptools | 56.1.0 |
| kiwisolver | 1.3.1 | statsmodels | 0.12.2 |
| Lxml | 4.6.3 | tensorboard | 2.5.0 |
| matplotlib | 3.4.2 | tensorboard-data-server | 0.6.1 |
| multidict | 5.1.0 | tensorboard-plugin-wit | 1.8.0 |
| multitasking | 0.0.9 | tensorflow | 2.4.1 |
| Nltk | 3.6.2 | tensorflow-estimator | 2.4.0 |
| numpy | 1.19.5 | termcolor | 1.1.0 |
| oauthlib | 3.1.0 | textblob | 0.15.3 |
| opt-einsum | 3.3.0 | tqdm | 4.60.0 |
| pandas | 1.2.4 | tweepy | 3.10.0 |
| Patsy | 0.5.1 | tweet-preprocessor | 0.6.0 |
| Pip | 19.0.3 | typing-extensions | 3.7.4.3 |
| preprocessor | 1.1.3 | urllib3 | 1.26.4 |
| protobuf | 3.16.0 | wheel | 0.36.2 |
| pyasn1 | 0.4.8 | wrapt | 1.12.1 |
| pyasn1-modules | 0.2.8 | yarl | 1.6.3 |
| pyparsing | 2.4.7 | yfinance | 0.1.59 |
| python-dateutil | 2.8.1 | zipp | 3.4.1 |
| Pytz | 2021.1 |  |  |
| Regex | 2021.4.4 |  |  |
| requests | 2.25.1 |  |  |
| requests-OAuth lib | 1.3.0 |  |  |

Table 2: Project Packages

**4.2 Results and Discussion**

Recommendation systems for stock trading are of great significance to a layperson who wants to benefit from stock trading despite not having a seasoned trader's capacity or experience. Recommendation systems such as these can discover trends in stock price movements and can greatly complement a stock trader's decision-making process by creating stock recommendations based on the patterns thus discovered. It is considered to be a challenging task to trade successfully in stock markets. Significant stock trading experience and the capacity to spot trends in stock price movements are required to become a successful stock trader [6]. Therefore, a stock trading recommendation system that can assume the role of an 'expert' trader and generate stock buy/sell recommendations is of great value to a layperson who wants to profit by investing in stocks.

We were able to obtain a superior return performance using this AutoARIMA model in conjunction with our simple stock trading approach than if we had just invested in SPY. However, thanks to the rapid stock market drop, we were able to perform fairly well in the end.

It's extremely likely that we wouldn't have gotten identical findings if we back tested on a different time frame or with different stocks. It's a good idea to start forward-testing this method at this point to have a better picture of our model's true performance.

**4.3 Individual Contribution of project members**

The tasks were distributed equally amongst us.

Akruti Sinha created the front end of the website, applied some of the Machine Learning algorithms, the presentations, and half of the project report.

Mahin Anup applied some Machine Learning, NLP algorithms and improved them, developed the back-end of the website. Also wrote half the project report.

**CONCLUSION AND FUTURE**

**5.1 Proposed Work Plan of the project**

Many factors influence stock prices, which result in a highly nonlinear structure. As a result, making stock trading predictions and recommendations is a difficult undertaking. In this research, a user-friendly stock trading prediction and recommendation system are proposed. Existing recommendation algorithm selection of the correct stock at the right moment is the most important factor in stock trading decision-making. However, with our proposed paradigm, this issue is reduced to a minimum. These are mostly based on the analysis of trading data and the forecasting of firm profits. Despite the fact that many studies reveal a strong association between investor sentiment and financial market developments, few recommendation theories based on sentiment have been developed. Different data extraction and analysis approaches have been investigated as part of this study.

This project includes a stock-recommendation system that analyses quarterly reports, news articles, and stock prices to suggest appropriate stocks for additional (human) examination based on user interest (e.g. resources or tech companies). To facilitate research of prospective n-bagger stocks, the system is intended for relevance, innovation, and serendipity (with configurable parameters).

Our future plan is:

* Integration of Spark to handle online learning and real-time data processing (continuous prediction)
* Create Recommenders for different time frames
* Integrate multiple higher-order features
* Create additional higher-order features (e.g. RNN predictions)
* Integrate Rule-Based approaches (e.g. implement Ben Graham Strategies)
* Balance Dataset for prediction
* Test additional NLP approaches (LSTM embeddings through character prediction)
* Bayesian Networks to measure confidence in stock predictions

**REFERENCES APPENDIX**

1. R. C. Cavalcante and A. L. I. Oliveira, "An autonomous trader agent for the stock market based on online sequential extreme learning machine ensemble," 2014 International Joint Conference on Neural Networks (IJCNN), Beijing, 2014, pp. 1424-1431.
2. A. Porshnev, I. Redkin and A. Shevchenko, "Machine Learning in Prediction of Stock Market Indicators Based on Historical Data and Data from Twitter Sentiment Analysis," 2013 IEEE 13th International Conference on Data Mining Workshops, Dallas, TX, 2013, pp. 440-444.
3. Analytics Vidhya Blog visited on 22.02.21.
4. M. Usmani, S. H. Adil, K. Raza, and S. S. A. Ali, "Stock market prediction using machine learning techniques," 2016 3rd International Conference on Computer and Information Sciences (ICCOINS), Kuala Lumpur, 2016, pp. 322-327
5. Stock prediction using ML visited on 20.02.21  
   <https://www.analyticsvidhya.com/blog/2018/10/predicting-stock-price-machine-learningnd-deep-learning-techniques-pyth>
6. Shetty, Nisha& Pathak, Ashish. (2017). Indian Stock Market Prediction Using Machine Learning and Sentiment Analysis. 10.1007/978-981-10-8055-5\_53.
7. Yang, Hongyang and Liu, Xiao-Yang and Wu, Qingwei, A Practical Machine Learning Approach for Dynamic Stock Recommendation (December 16, 2018). Available at SSRN:<https://ssrn.com/abstract=3302088> or [http://dx.doi.org/10.2139/ssrn.3302088](https://dx.doi.org/10.2139/ssrn.3302088)
8. P. D. Yoo, M. H. Kim and T. Jan, "Machine Learning Techniques and Use of Event Information for Stock Market Prediction: A Survey and Evaluation," International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06), Vienna, 2005, pp. 835-841
9. Hiransha, M., Gopalakrishnan, E. A., Menon, V. K., & Soman, K. P. (2018). NSE stock market prediction using deep-learning models. Procedia computer science, 132, 1351-1362.
10. Vicari, M., Gaspari, M. Analysis of news sentiments using natural language processing and deep learning. AI & Soc (2020). https://doi.org/10.1007/s00146-020-01111-x