



Department of Computer Science
BSCCS Final Year Project 2021-2022
Interim Report I

21CS068

**Stock Market Price Prediction with a combination of NLP and
Time Series**

(Volume __ of __)

**Student Name : VOBBILISSETTY Sai
Navyanth**

Student No. : 55357700

Programme Code : BSCEGU4

Supervisor : Dr. SONG, Linqi

Date : October 24, 2021

For Official Use Only

Table of Contents

1. Introduction	3
1. 1 Time Series Analysis	3
1. 2 Natural Language Processing	4
1. 3 Project Objective & Scope	4
2. Literature Review	5
2. 1 Stock Price Prediction Using Time Series Models	5
2. 1. 1 Analysis	5
2. 1. 2 Limitations	6
2. 2 Improvements in our potential solution	7
3. Preliminary design, solution, and system	7
3. 1 Data Collection and Storage	7
3. 2 Training Algorithms	8
3. 2. 1 ARIMA	8
3. 2. 2 LSTM (Long Short Term Memory)	9
3. 3 Web Application Development	10
4. Testing Procedures	10
4. 1 Unit and Integration Testing	10
4. 2 Backtesting by Using Virtual Trading Environment	11
5. Tasks to be done	12
6. Monthly Log	12
7. References	13

1. Introduction

In recent years, the number of retail investors investing in the stock market has significantly increased [4]. This rise can be attributed to the pandemic since retail investors became aware of and better informed about stock markets. Another factor that led to this is influential people like Elon Musk, Chamath Palihapitiya, etc.

Most retail investors invest based on the crowd/momentum and end up incurring losses. I have witnessed the losses of many people who had invested their own money into the market as first-time retail investors with little knowledge about the intricate workings of the market.

Moreover, it is very time-consuming for retail investors to keep track of the various companies and industries in the stock market.

As a solution to this problem, my groupmate (Varun) and I decided to work on this topic for our final year project as a group, where we aim to assist retail investors to predict better as to whether they should invest in a particular company by checking the company's recent news and historical data to forecast the price of the stock.

The two leading technologies we will be using in our FYP are:

1. Time Series Analysis/Forecasting (implemented by me)
2. NLP (Natural Language Processing) (implemented by my groupmate)

1. 1 Time Series Analysis

Time series is a series of observations recorded at regular time intervals. The time interval can be hourly, daily, weekly, monthly, quarterly, and annually. Depending on the problem, there might be time series analysis where time intervals can be as small as seconds or minutes [10].

In our project, we would use time series to predict/forecast the price of a particular company's stock in the future, using the stock's price in the past. This is a quantitative way of predicting the

price of a stock and does not consider any sentiment of how the company is performing in the market.

1. 2 Natural Language Processing

According to IBM [6], Natural Language Processing is a branch of Artificial Intelligence, which gives computers the ability to understand the text and spoken words in much the same way as humans.

NLP is used in multiple scenarios [1]. Here are some of them:

1. Text Search
2. Machine Translation
3. Summarizing group of text
4. Sentiment Analysis

In this project, we will be focusing mainly on sentiment analysis of news data. Sentiment analysis is the process of calculating the sentiment of a given text using different methods. The sentiment can be either positive, negative, or neutral.

1. 3 Project Objective & Scope

Objective: We are creating a fully working framework to predict the price of a stock, from acquiring historical data from yahoo finance to cleaning the data to train our algorithms (mainly LSTM and ARIMA). In the end, create a simple web app to display the results.

Scope: We intend to create multiple models and test which model is efficient. The main models we plan to implement are:

1. LSTM (Long Short Tem Memory) Neural Network
2. ARIMA (Auto-Regressive Integrated Moving Average)

We will backtest both these methods and decide which one should be used in our web app.

For the data, we will be using the top 10 companies present in the NASDAQ 100 index as of 19th September 2021.

2. Literature Review

The recent works in this area are mainly focused on predicting the price of a stock using NLP on stocks' news [8] or through time-series analysis of the historical prices of the stocks [5,7,12].

However, predictions solely based on this information might not be perfect. In the following section, I explain the reasons why these methods are not excellent.

2. 1 Stock Price Prediction Using Time Series Models

2. 1. 1 Analysis

There are different models one can use to perform time series analysis.

Some of them are:

1. ARIMA Model
2. PROPHET
3. KERAS with LSTM

The paper (Chouksey, 2019) has implemented all the above three methods for stock price prediction. The author conducted a detailed analysis of all three methods by comparing their Root Mean Square Errors (RMSE). In other words, the lower the error rate, the better.

They have analyzed five different Indian banks for 20 years taken from the Yahoo Finance library.

The detailed analysis shows that the KERAS with LSTM model performs the best with an average RMSE of 6% compared to 151%, 173% for PROPHET and ARIMA Model, respectively.

2. 1. 2 Limitations

There are multiple limitations in approaches like this. They are:

1. Lack of backtesting:

While implementing a trading strategy, backtesting is very important. Backtesting is basically to determine how good a trading strategy is by testing the strategy with historical data. In this way, one can find out how a particular strategy works [7].

2. Merging algorithms:

The research paper described above has only worked with one algorithm at a time. For example, in one scenario, ARIMA might be better than LSTM, while it might be different in another. If the author had tried to combine the outputs of both these algorithms, the output might have been better. This process of combining models is called ensemble learning.

Ensemble learning combines multiple models or classifiers to improve the accuracy of a particular problem. So, models trained with ensemble learning are generally better than models which are static [11].

3. Performance it in real life:

The author doesn't implement the strategy in real life to show how it performs. For example, a paper trading account can be opened, dedicated to a trading strategy that would determine how the portfolio is performing.

In this way, we can financially analyze how the strategy would perform in real life if implemented with real money. But, this is not done in most research papers, as they mainly focus on the technical implementation.

2. 2 Improvements in our potential solution

We mainly tackle the three limitations that are described above. To overcome the first problem, we do extensive backtesting to our algorithm. As we already have a stock's historical price, we can backtest the strategy using the data.

To overcome the second limitation, both the algorithms developed by my groupmate and I would be combined in the end to provide better results. In this way, we won't have to depend on only one algorithm.

Finally, to overcome the third limitation, we implement our strategies in real life using online stock brokers that provide free paper money, which can be used to test.

3. Preliminary design, solution, and system

3. 1 Data Collection and Storage

We will be using the 'yfinance' library in python to get the historical data of stock. The primary focus in this project would be the top 10 companies of NASDAQ 100. They are:

We will extract the past ten years of data of these ten companies and use it to train our algorithms.

We will be using a MongoDB database located in a local Heroku server. The data we extract will be stored in this MongoDB database to be referenced anytime when needed.

In addition, I will be helping my groupmate in extracting news data, as it is comparatively difficult. We will be extracting news data from different websites like Bloomberg, Reuters, etc., and save it in the MongoDB database. We will build a scraper in python and run it every day so that it can extract the required data every day.

3. 2 Training Algorithms

We will be implementing two kinds of algorithms:

1. ARIMA
2. LSTM

3. 2. 1 ARIMA

Auto-Regressive Integrated Moving Average is a statistical method to analyze and forecast time series data. Lets breakdown ARIMA to understand better:

Auto-Regression (AR): This algorithm determines the relationship between the present value or observation and few lagged values or observations.

Integrated (I): Subtracting an observation/value from an observation/value at another time step.

This is used to make the time series stationary. A stationary time series data is a series of data whose mean and standard deviation is constant throughout the series. ARIMA cannot work if the data is not stationary. That is why ARIMA converts the data to stationary beforehand.

Moving Average (MA): This is an algorithm that uses the relationship between an observation or value and a residual error by applying the moving average of previous observations or values.

ARIMA mainly consists of 3 parameters p, d , and q .

' p ', also known as 'lag order' represents the number of previous observations or lagged observations.

' d ' represents the order of differencing needs to be done in Integrated (I).

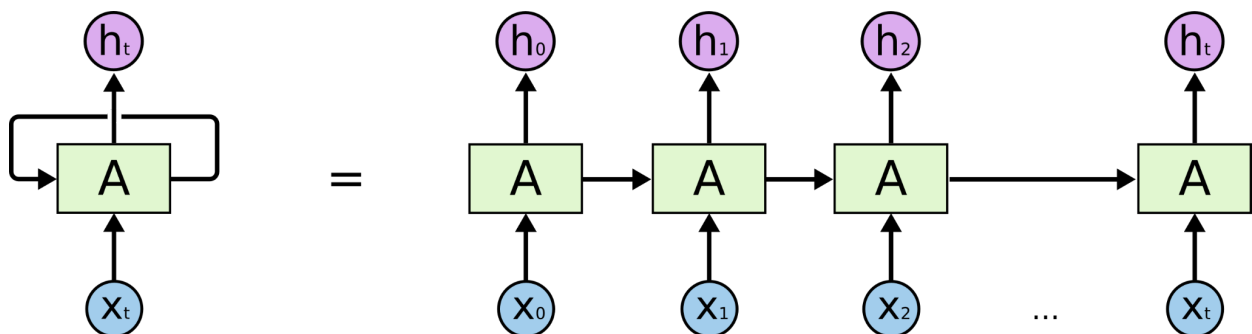
' q ' represents the size of the moving average window.

We will be tuning these three parameters and finding the best combination that would fit our training data.

We will mainly use the 'statsmodels' library in python, which provides many statistical models, where ARIMA is one of them. We will use hyperparameter tuning using 10-fold cross-validation to find the best values of ' p ', ' d ', and ' q '.

3. 2. 2 LSTM (Long Short Term Memory)

LSTM networks are a type of RNN (Recurrent Neural Network, which is used to learn from the experience for tasks like speech recognition, language translation, etc.) used to learn and deal with long-term dependencies. Traditional RNNs generally fail when there are long-term dependencies. For example, if asked to translate a very long sentence, an RNN would fail to do so. This is where LSTMs are utilized. They come with all the utilities an RNN can provide and have additional features that an RNN fails to provide.



It is important to learn from past experience in this project and provide it to future data. That is why we will be using LSTMs. We will use the closing price of a stock and train it to our LSTM network using a moving window of 30-40 steps.

In other words, using the past 30-40 days of stock price, our trained LSTM model will predict the following day's price. The number of days, moving period is a hyperparameter. With our current knowledge, we think it would be better to have 30-40 days as our training set, but it would change as we backtest our algorithm.

We will use the PyTorch framework to build our LSTM network. PyTorch was developed by Facebook. Our LSTM network would have initially five hidden layers and 50 neurons, and our data will be normalized to be between 0-1 using the MinMax Scaler from the 'scikit-learn' library. This will make our data normalized. Later, we will update our training data in such a way that the input would be a sequence of prices of the previous 30-40 days, and the output

would be the price of the following day. The hyperparameters which need to be tuned for the network would be the number of layers, number of epochs, mini-batch size. etc.

Once both the algorithms are done training, we will backtest our models and find which one works better, either LSTM or ARIMA. Later, we will store the model as a pickle file in our MongoDB database. This final model will be used in the web app for the predictions.

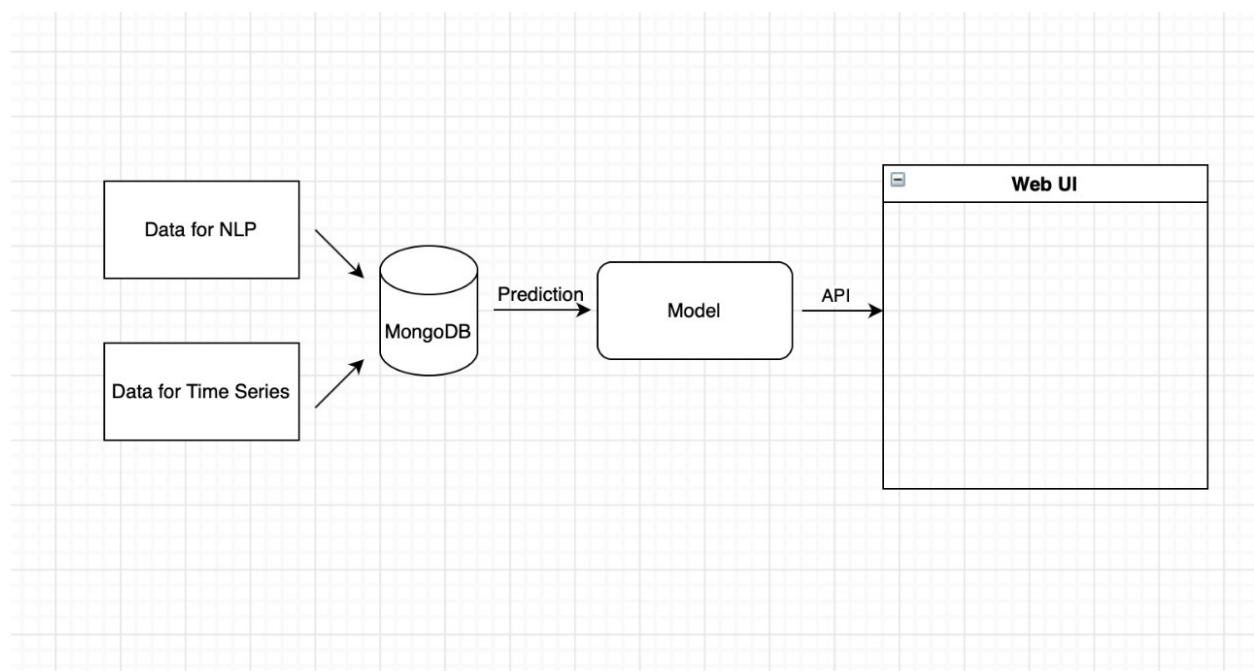
3. 3 Web Application Development

Once the model training part is done, we will build a simple web app using the JavaScript framework Vue.js for the front-end and a python framework called flask for the backend.

In the front end:

- We will allow the user to select a company from a dropdown list of the ten companies.
- A graph showing how the stock performed in the past 30-40 days.
- The prediction of the stock's price, along with a signal showing whether to buy/sell/hold the particular stock.

In the backend, we will connect the web app to the MongoDB database, containing the final models and the data. In addition, the predictions will be stored in another database in order to let the user view the predictions over time. This would be the final workflow of our project:



4. Testing Procedures

We test the whole framework in two different ways:

1. Unit and Integration Testing
2. Backtesting by Using a Virtual Trading Environment

4.1 Unit and Integration Testing

Our framework consists of multiple interdependent modules. So, it is crucial to perform Unit Testing of smaller modules and Integration testing whenever needed (when combining multiple smaller modules). If one of the modules fails to work, the whole framework will break down. So, it is important to do these steps.

We will be using two python libraries, 'unittest' and 'integrate', to perform Unit Test and Integration tests, respectively, in python. But other than these two, we will also write custom unit test cases, in order to test our machine learning model, if it returns good results for base cases. For example, one test case can be, if a hypothetical stock stays constant everyday, which means, the price of the stock is constant everyday, then the prediction should also be the same constant. But, if our model fails to do so, it means it failed the uni test. Similarly, we can add more test cases, to test our model.

4.2 Backtesting by Using Virtual Trading Environment

Any Machine Learning algorithm or research paper dedicated to predicting stock prices (for example, the research paper explained in Literature Review) would calculate the MSE (Mean Squared Error) of their predictions and analyze how their algorithm worked. But, in real-life trading scenarios, there are various factors to consider if a trading strategy works well, like transaction costs, bid-ask spread, time lag, etc.

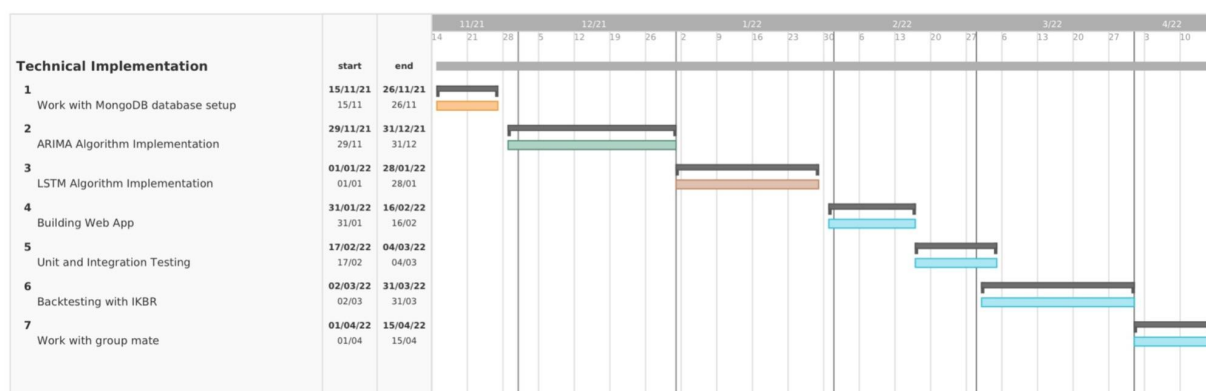
This is our primary source of analyzing and testing the performance of our models. Interactive Brokers (IBKR) is a US trading platform used to trade various kinds of securities like stocks,

options, futures, currencies, bonds, and funds. IKBR provides a free API and allows users to have a virtual/paper trading account.

In IKBR, we will open a paper trading account with virtual cash of \$100,000. We will maintain the portfolio of the ten stocks of NASDAQ 100, and we will make trades solely based on the predictions from our model. We will continue this process for every trading day in a month. In the end, we will calculate different financial ratios, like the Sharpe ratio, hit ratio, the net return, etc., to evaluate our portfolio in real life.

5. Tasks to be done

At the time of writing this report, research on various frameworks for web apps and neural networks has been done. After today, I will be following the tasks as mentioned in the below Gantt chart.



According to the schedule, ARIMA and LSTM algorithms need to be implemented by the end of January 2022. By the end of April 2022, Testing and Webapp implementation needs to be done.

6. Monthly Log








Monthly Log (October 2021)

Current Time: Nov 12, 2021 (Fri) 2:03:16 AM

Submission Deadline: Oct 31, 2021 (Sun) 11:59:59 PM

Date Last Edited: Nov 9, 2021 (Tue) 6:42 PM

Content*

  **B** *I* U     

Researched deep learning models for time series analysis.

Research on web stack for building the website.

7. References

1. Avenga. (2021, October 12). *Natural language processing: Tasks and application areas*. Avenga. Retrieved October 24, 2021, from <https://www.avenga.com/magazine/natural-language-processing-application-areas/>.
2. Chouksey, S. (2019). *Stock price prediction using time series models - DBS esource home*. Stock price prediction using time series models. Retrieved October 24, 2021, from https://esource.dbs.ie/bitstream/handle/10788/3830/msc_chouksey_s_2019.pdf?sequence=1&isAllowed=y.
3. Chouksey, S. (2019). *Stock price prediction using time series models - DBS esource home*. Stock price prediction using time series models. Retrieved October 24, 2021, from https://esource.dbs.ie/bitstream/handle/10788/3830/msc_chouksey_s_2019.pdf?sequence=1&isAllowed=y.
4. Fitzgerald, M. (2021, August 18). *A large chunk of the retail investing crowd started during the pandemic, Schwab survey shows*. CNBC. <https://www.cnbc.com/2021/04/08/a-large-chunk-of-the-retail-investing-crowd-got-their-start-during-the-pandemic-schwab-survey-shows.html>
5. Hoare, T. (2019, July 11). *Stock price prediction using time series models*. Dublin Business School. <https://esource.dbs.ie/handle/10788/3830>
6. IBM Cloud Education. (n.d.). *What is natural language processing?* IBM. Retrieved October 24, 2021, from <https://www.ibm.com/cloud/learn/natural-language-processing>.
7. Kuepper, J. (2021, September 13). *The importance of backtesting trading strategies*. Investopedia. Retrieved October 24, 2021, from <https://www.investopedia.com/articles/trading/05/030205.asp>.
8. Mehtab, S. (2019, December 9). *A Robust Predictive Model for Stock Price Prediction Using Deep*. . . ArXiv.Org. <https://arxiv.org/abs/1912.07700>
9. Mehtab, Sidra & Sen, Jaydip. (2020). *A Time Series Analysis-Based Stock Price Prediction Framework Using Machine Learning and Deep Learning Models*.
10. Prabhakaran, S. (2021, October 8). *Time series analysis in python - A comprehensive guide with examples - ML+*. Machine Learning Plus. Retrieved October 24, 2021, from <https://www.machinelearningplus.com/time-series/time-series-analysis-python/>.

11. Polikar, R. (n.d.). Ensemble learning. Scholarpedia. Retrieved October 24, 2021, from http://www.scholarpedia.org/article/Ensemble_learning.
12. Shen, J. (2020, August 28). *Short-term stock market price trend prediction using a comprehensive deep learning system*. Journal of Big Data. <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00333-6>