Capstone Proposal

Machine Learning Engineer Nanodegree

Parameswaran Iyer
March 4, 2019

Domain Background

In early 1600s The Dutch East India Company became the first company to issue bonds and shares of stock to general public and thus, it became the first official publicly traded company. This gave birth to the concept of stock exchange in a large scale.

Today almost every developed and developing countries have stock exchanges where in millions of transactions happens on a daily basis.
Since childhood, the enormous scale at which these stock exchange operate has been a very astonishing to me, and yet this boils down to the simple fact of trading, i.e. buy at a lower price and sell at a higher price.

We can find similar researches in the following links:

- https://www.sciencedirect.com/science/article/pii/S1877050918307828
- https://www.researchgate.net/publication/259240183_A_Machine_Learning_Model_for_Stock_Market_Prediction
- https://acadpubl.eu/jsi/2017-115-6-7/articles/8/12.pdf

Problem Statement

If we leave all the complexity of the stock market for some time, the sole purpose in stock market is to make a profit by trading in stocks. This is done simply by buying a certain amount of stock at a lower price and selling at a higher price. This being the usual way, there is also another way which is called as 'Short Sell'. The basic principle of short selling is to sell the stock, which you **DO NOT OWN**, first at a higher cost and later buy it at a lower price and reimburse it to the broker.

In either of the case, the profit is made by the difference in the buying price and the selling price. Hence, in order to make a profit, it becomes necessary to analyze the performance of the stock over a time period and predict the closing price of that stock.

Therefore, the problem to be solved is *to predict the closing price of given stock using the historical data of the same using concepts and techniques of machine learning.*

Datasets and inputs

For this project, the data is taken from Yahoo! Finance [2]. Unfortunately, because of lack of official API any longer, we are using a third-party package called as ‘fix-yahoo-finance’ [6] which takes the data from the Yahoo! Finance webpage and returns the same. The data will be taken from 01-01-1997 to 31-12-2017. The dataset consists of 5196 non-null rows.

Note – As the library is a third-party package, sometimes there may be issue which can be lead to failure while fetching the data. If the issues are too often, then instead of using the package we would download the historical data CSV file directly from the Yahoo! Finance and use it in the input.

There are 3 inputs which is necessary for accessing the data:

1. The ticker for the company whose stock price needs to be predicted. This has to be the same as that in Yahoo! Finance webpage for the corresponding ticker.
2. Start date – Date from which the data needs to be provided.
3. End date – Date till which the data needs to be provided.

The dataset contains the following columns:

- Date – The date for which the data is given.
- Open – The opening price of the stock for given date.
- High – The highest price of stock on that given date.
- Low – The lowest price of stock on that given date.
- Close – The closing price of the stock for given date.
- Adjusted Close – The close price which is adjusted according to the announcement of the company after the stock market closes.
- Volume – The number of stock traded on that given date.

Solution Statement

The task for this project, is to build a stock price predictor which takes inputs for certain consecutive days and returns the stock closing price for the corresponding next day. The potential solution would be to create a neural network which learns the trends in the stock prices from the start date of '01-01-1997' to the end date of '31-12-2017' and predict the stock price with a marginal error with inputs of past few days, for example 4 days, and predict the closing stock price of next day.

This in turn makes it easy for the investor to learn the behaviors of the stock price and make a calculated investment accordingly.

Benchmark Model

In order to compare the final output, we would use K-nearest neighbors regressor model. Apart from the regressor model, statistics from Google Finance, Yahoo! Finance and Moneycontrol can be used to gain and insight for the short term stock pricing.

Evaluation Metrics

The model will be evaluated with Mean Squared Error, Mean Absolute Percentage Error and Mean Absolute Error

Mean Squared Error [3]:

$$\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \tilde{y}_i)^2$$

Mean Absolute Error [4]:

$$\mathrm{MAE} = \frac{1}{n} \sum_{j=1}^{n} |y_j - \hat{y}_j|$$

Mean Absolute Percentage Error [5]:

$$\mathrm{MAPE} = \frac{\sum \frac{|A-F|}{A} \times 100}{N}$$

Here, in this project, the model results can be considered satisfactory if the Mean Absolute Percentage Error is less than 2%.
This is mainly because stock market works in real time and in most of the days, the economic conditions and political conditions of the country also favor in the performance of the stock market. Moreover, the news about the company also affects the stock market.
For example, a war like situation will make the stock market plummet which in turn reduces the stock price. Since these situations cannot always be known beforehand, hence it would be very difficult to predict the exact closing stock price

Project Design

The approach to the problem would be to create a model in a modular way, such that it can later be directly used with a UI. It is to be remembered that the main aim of this project is to create a neural network which predicts the stock price and hence more focus would be given at the same and UI part would be approached only if time permits.

The backend for the project would be divided into following segments:

1. Data Collection.
2. Data Pre-processing and Normalization.
3. Creating feature set and output set.
4. Splitting the data set into training, validation and testing set.
5. Defining the model architecture.
6. Compiling and training model.
7. Testing the model.
8. Predictions.

Data Collection:
The data would primarily be used of NSE and BSE stock exchanges. For this, third party package will be used or the data would be directly downloaded from the Yahoo! Finance website.

Data Preprocessing and Normalization:
All the null values will be removed and the data will be normalized for better performance of the model.

Creating feature set and output set and splitting the data:
Feature set will consist of 'Open', 'High', 'Low', 'Close' for last *n* number of days and the output set will consist of the 'Close' stock price for the *n+1*th day.

The dataset would be split into training, validation and testing set. While splitting the data, care would be taken that the data is not shuffled. The data would be split into 80-20, 80% being training and validation set and 20% being testing set. Of the 80%, the first half would be used for training and the second half for validation. Therefore, training set would consist of 2076 samples, validation would consist of 2077 and testing would consist of 1039. As mentioned earlier, the dataset would not be shuffled and the first 80% would be training and validation set and the rest 20% as testing set.

Defining Model Architecture:
For the model architecture, 2 models will be primarily taken into consideration. One of them will be a simple neural network which would consists of Dense and Flatten layers with RELU activation and the other would be a Recurring Neural Network consisting of LSTM architecture.

Compiling, Training and Testing model.
The model would be compiled with the callbacks for ModelCheckpoint [7] to save the best weights and EarlyStopping [7] in order to avoid overfitting.

References

1. https://en.wikipedia.org/wiki/Stock_market
2. https://finance.yahoo.com/
3. https://en.wikipedia.org/wiki/Mean_squared_error
4. https://en.wikipedia.org/wiki/Mean_absolute_error
5. https://en.wikipedia.org/wiki/Mean_absolute_percentage_error
6. https://pypi.org/project/fix-yahoo-finance/
7. https://keras.io/callbacks/