PREDICTION OF STOCK PRICE USING MACHINE LEARNING

Name: Gopikha Menon Email: gopikha25@gmail.com

Third year Computer Engineering,
Vidyavardhini's College of Engineering and Technology

ABSTRACT: High authenticity and utmost precision is the key factor in prognosticating or predicting the stock market. The technical or the time series analysis is used by most of the stockbrokers while making the predictions. The inefficiency in these methods leads to the research of finding the most effective prediction model that generates the most accurate prediction with the lowest error percentage. The paper evaluates various machine learning techniques and algorithm employed to upgrade the legibility of stock price prediction.

1. INTRODUCTION

Stock market exchange provides the imperative results about the share and events related to business accomplishments. Stock price prediction is often deemed as one of the most arduous tasks in the financial world. This is owing to the existence of many attributes particularly political and economic shocks, the comprehensive opinion of people towards the company and economic outlook hence changing the trend of stock values. We can classify this under non-linear series forecasting. During recent years machine learning and artificial intelligence have played a considerable role in creating algorithms that help to predict the steering of the stock market. The stock market has enormous historical data that varies with trade date, which is time-series data, but the LSTM (Long-Short Term Memory) model predicts future price of stock within a short-time period with higher accuracy when the dataset features a huge amount of knowledge. LSTM plays a huge role as it has some contextual state cells that act as long-term and short-term memory cells which allows it to take both past inputs and current inputs in account to predict the next value, because of this it can predict stock values with minimal error.

2. DATA COLLECTION

For training the algorithm, we will collect the data set of APPLE, GOOGLE, MICROSOFT from YAHOO FINANCE STOCK DATA. It consists of data set attributes of a specific company and former stock rates and investors' increase rate and company rate of growth. The key values are considered to predict the analysis.

DATA DESCRIPTION

Date	High	Low	Open	Close	Volume	Adj Close
2012-01-03	14.732143	14.607143	14.621429	14.686786	302220800.0	12.650659
2012-01-04	14.810000	14.617143	14.642857	14.765714	260022000.0	12.718646
2012-01-05	14.948214	14.738214	14.819643	14.929643	271269600.0	12.859850
2012-01-06	15.098214	14.972143	14.991786	15.085714	318292800.0	12.994284
2012-01-09	15.276786	15.048214	15.196429	15.061786	394024400.0	12.973674
...	...	...	...	...	...	...
2021-02-22	129.720001	125.599998	128.009995	126.000000	103916400.0	126.000000
2021-02-23	126.709999	118.389999	123.760002	125.860001	158273000.0	125.860001
2021-02-24	125.559998	122.230003	124.940002	125.349998	111039900.0	125.349998
2021-02-25	126.459999	120.540001	124.680000	120.989998	148199500.0	120.989998
2021-02-26	124.849998	121.199997	122.589996	121.260002	164320000.0	121.260002

The data has been collected during the tenure of '2012-01-01' to '2021-02-28'. The attributes are open, close, adjusted close, volume highest and lowest values of a given day (in USD).

We can observe that there are not values for all the days in the given period. This is a result of inconsistent recording of data.

We visualize the closing price history as we are going to review the close price to predict the values.

CLEANING AND PREPARING DATA

Cleaning process takes considerable amount of time. Invalid data such as negative values, misprinted values and duplicated values are to be discarded. We use **pandas** to obtain data from **yahoo finance.** As our target value is the Close value, a target data frame is created with only close column. The data is then normalized/scaled and converted so that all the values lie between 0 and 1.

Date	Close
2012-01-03	14.686786
2012-01-04	14.765714
2012-01-05	14.929643
2012-01-06	15.085714
2012-01-09	15.061786
...	...
2021-02-22	126.000000
2021-02-23	125.860001
2021-02-24	125.349998
2021-02-25	120.989998
2021-02-26	121.260002

We find the number of rows to train our model and transform the dataset into values using numpy array and dataset is split into two parts for training and testing.

```
array ([[0.00572147],
        [0.00633231],
        [0.00760099],
        ...,
        [0.862165],
        [0.82842213],
        [0.83051175]])
```

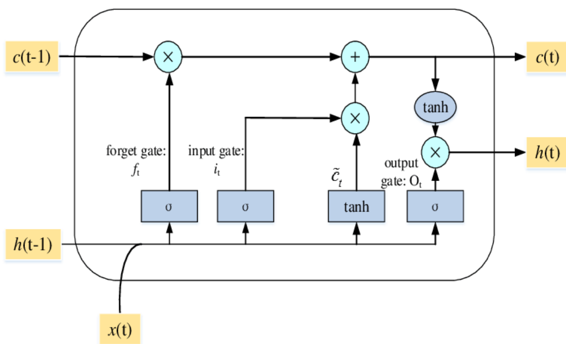
3. MACHINE LEARNING MODELS

Will be using various forecasting techniques LSTM or LONG-SHORT TERM MEMORY. The parameters like RMSE, MAE and MSE are considered to verify the performance of these models.

LONG-SHORT TERM MEMORY (LSTM)

LSTM (Long Short-Term Memory Network) is a type of neural recurrent network capable of retrieving past information and, while predicting future values, takes this past information into account. It is proficient in learning order dependencies in sequence prediction problems. A Sequential is also a type of neural network model that has many forms of simple neural network structure and is constantly in loops that can go from one state to another at a time when the structure of the neural network is very easy.

The classic LSTM architecture is characterized by recurrent linear cell state enveloped by non-linear layers that supply input and compiling output from it. Principally, the cell state operates in collaboration with 4 gating layers, which are often alluded to as forget, (2x) input, and output gates.



The forget gate can select which values of the old cell state to get rid of, depending on the current input results. The two input gates (often referred to as i and j) work with each other to determine contents to be added to the cell state based on the input. i and j tend to have different activation functions, which we conceptually assume to be used to demonstrate the vector scaling and the candidate values to be updated to the cell state.

Finally, the output gate decides which parts of the cell state should be transferred to the output. Note that in the case of classic LSTMs, output h consists of hidden layer activations (which may well be subject to additional layers for grouping) and input comprises of the previous hidden state output as well as any new data x received at the current time level.

As time progresses it is less likely that performance will rely on very obsolete inputs, these inputs will be overlooked by their forgot gates, which is just a multiplicative factor of 0.9, that is, in 12 steps the factor will be 0.9^{12}=0.282. The equations used for forget gates, input and output gates, allow LSTM to recognize previous inputs when evaluating the succeeding output and forget gate equation helps to forget very old inputs, are given below.

$$ft = \sigma(wf\,(ht-1, xt) + bf\,)$$
$$it = \sigma(wi(ht-1, xt) + bi)$$
$$ot = \sigma(wo(ht-1, xt) + b0)$$

In the training methodology, we use LSTM to forecast the closing prices of the stock next day. The first step is to scrap the Yahoo Finance data. The data is then scaled to fit the ranging between zero and 1.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Equation for scaling

We have preprocessed our data, segregated it into training and testing data set transformed and stored it in numpy arrays. Now the LSTM model developed. The LSTM architecture that we are trying to design would be a multi-layer sequential model. We will incorporate two layers of LSTM to our model, followed by a dense layer that forecasts future stock prices. We add a **dense layer** at the end of the model to make it more robust. The number of neurons in the dense layer can be set to 1 since we want to estimate a single value in the output or any other number of values we want to estimate. We used Adam for optimizer and trained 3 epochs.

The **Adam optimizer** is one of the most commonly used optimizers for deep learning. When training with Adam the model usually converges tons faster than when using regular stochastic gradient descent (SGD), and Adam often requires less tuning of the training rate compared to SGD with momentum. Adam improves on SGD with momentum by (in addition to momentum) also computing adaptive learning rates for every parameter that's tuned. This means that when using Adam there's less got to modify the training rate during the training than when using SGD.

The estimated close values were then compared with original close values and the **Root Mean Squared Error** (RMSE) was computed to obtain the error value and determine the precision.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

Equation of RMSE

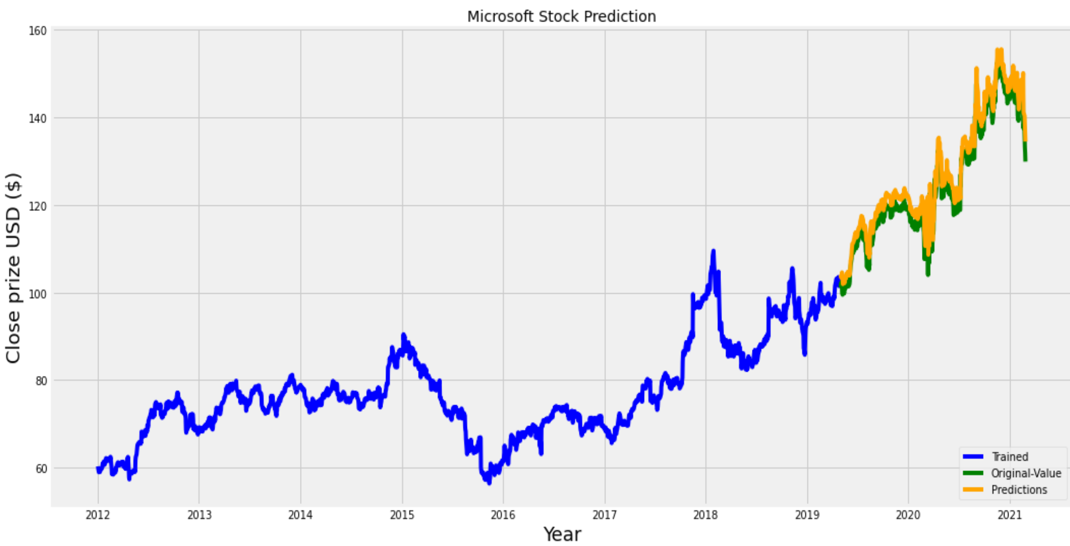
Finally, we need to transform our data to a three-dimensional format that can be used as an input to the LSTM. In order to make forecasts, we basically need to call the predictive method in the model we have trained. Since we have scaled our data, the results produced by the LSTM are also scaled. We ought to revert the scaled prediction back to their true values for which **Inverse transform method** of the scalar object created, is employed.

A graph is prepared to visualize the relation between the observed and the literal values.

EXPERIMENTAL RESULTS:

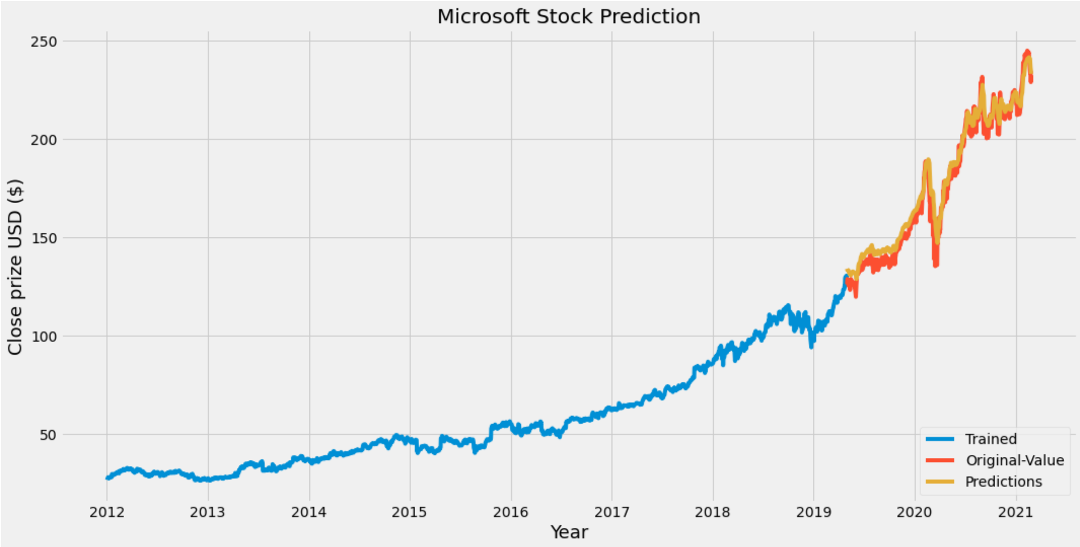
We have tested the approach with Walmart, Microsoft and Apple.

1. Walmart Stock Prediction:



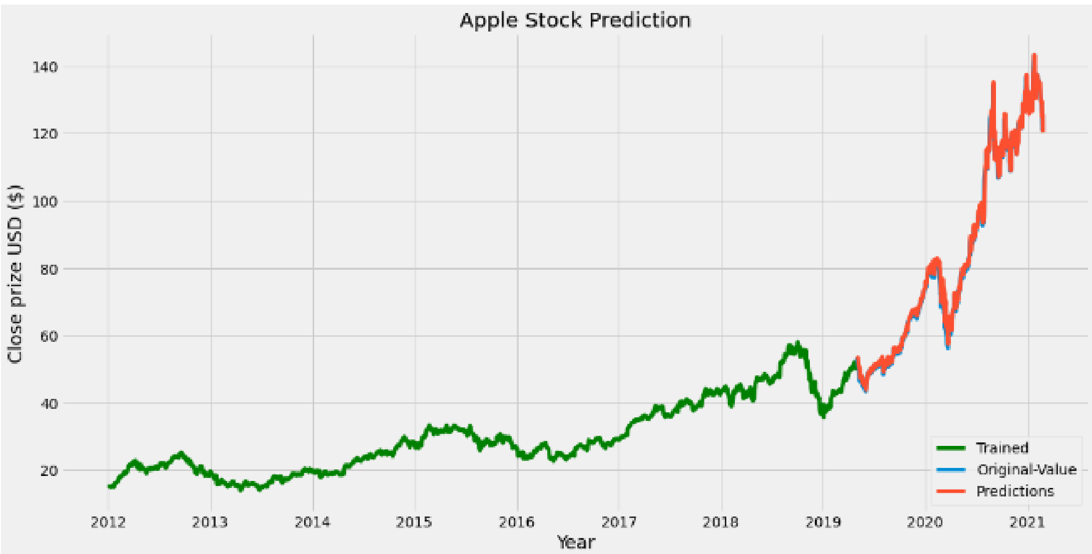
Comparison between predicted and actual values of stock for Walmart.Inc

2. Microsoft Stock Prediction:



Comparison between predicted and actual values of stock for Microsoft Corp.

3. Apple Stock Prediction:



Comparison between predicted and actual values of stock for Apple.Inc

The tables following represents the predictions made by the algorithm of recent periods and the real stock prices of those dates. The last five forecasts as seen in the tables below shows the difference between the prediction and the real value is much reduced, it shows that the LSTM can estimate the next day's close value very precisely.

Date	Close	Predictions
2019-05-02	101.150002	99.943443
2019-05-03	102.080002	99.511177
2019-05-06	102.459999	100.048927
2019-05-07	101.300003	100.593437
2019-05-08	100.300003	99.961792
...	...	...
2021-02-22	137.690002	133.704636
2021-02-23	135.470001	133.693329
2021-02-24	133.210007	132.221939
2021-02-25	131.949997	130.201736
2021-02-26	129.919998	128.935532

Walmart Predictions

Date	Close	Predictions
2019-05-02	126.209999	134.085663
2019-05-03	128.899994	133.420303
2019-05-06	128.149994	133.375259
2019-05-07	125.519997	133.346252
2019-05-08	125.510002	132.731323
...	...	...
2021-02-22	234.509995	240.900696
2021-02-23	233.270004	238.667664
2021-02-24	234.550003	236.458572
2021-02-25	228.990005	235.200928
2021-02-26	232.380005	233.186081

Microsoft Predictions

Date	Close	Predictions
2019-05-02	52.287498	52.846581
2019-05-03	52.937500	52.845005
2019-05-06	52.119999	53.360538
2019-05-07	50.715000	52.705135
2019-05-08	50.724998	51.379726
...	...	...
2021-02-22	126.000000	129.551895
2021-02-23	125.860001	125.167831
2021-02-24	125.349998	125.490425
2021-02-25	120.989998	125.142357
2021-02-26	121.260002	120.319267

Apple Predictions

4. CONCLUSION:

There are several variables influencing the stock market, and stock data has the characteristics of significant fluctuations and non-linearity. In this article, we discussed the use of the LSTM neural network to predict stock prices. The model structure, the algorithm framework and the experiment design are presented. LSTM has proved to be the best algorithm to predict stock values as it takes into account previous values, but still uses forget gates to delete very old values as it is unlikely that the next result will rely on them, making it quite accurate.

5. FUTURE WORK:

There are number of further directions that can be investigated beginning from this project. The first one is to include social media sites to get the public sentiment which can enhance the accuracy of prediction. Second, we will consider the changes in the international situation. Finally, LSTM could be optimized even more. The dimension reduction of the input parameters and optimizing the loss calculation method to improve the average accuracy of the model to predict stock.