STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

EM-623 Final project

Topic: Prediction of Stock Market with Machine learning concepts.

Mentor: Feng Liu

Course: Data Science and Knowledge Discovery

Members: Harsh Shah, Viraj Patel, Rushik Gandhi

Table of contents

Introduction

1.1 Problem statement and study objectives

Investment is the key/main source of income for majority of population around the world. Data shows almost 56% of U.S. population invests in stock market and out of this a large number of people have investments in shares and mutual funds. Here in this investment decision is made using machine learning LSTM technique which is very famous and trusted technique for predicting stock market for company BMW. Patterns of daily change is being analysed and then LSTM is performed for almost 10,000 samples for a year that 252 working market days. And main objective was to try to reduce the error in current LSTM models and find lowest percentile just being on safer side to analyse what could be the loss in worst possible situation. Also what percentage of investment should be invested and what can be the expected return. Graphs for all the parameters are being plotted for better visualization and understanding. Hence performance of AMAZON is being analysed and predicted and based on that decision is displayed. How money should be invested is also presented as a suggestion. What can be the range of expected returns? What can be the risk factor? And this model was validated with comparing the projected value with actual market value.

Prediction of stock trend has long been an intriguing topic and is extensively studied by researchers from different fields. Simple statistical analysis of financial data provides some insights but, in recent years, investment companies have increasingly used various forms of artificial intelligence (AI) systems to look for patterns in massive amounts of real time equity and economic data. Machine learning, a well-established algorithm in a wide range of applications, has been extensively studied for its potentials in prediction of financial markets. Today stock market analysis or prediction is the most difficult task to perform. Though many models and concepts have been developed and it is a lot more easier than previous times when there were no proper technologies and lack of knowledge and also no proper data management. But in this field still there are many variables which are hard to overcome like sentiments, economies etc which affect the stocks. Hence it becomes very difficult to predict. There are many existing models which use methods like LSTM, EMA, SMA and all but all models have credibility with their accuracy level. One of the important factor and variable that affect the stock market is sentiments and various efforts have been done to break the boundaries by incorporating external information through fresh financial news or personal internet posts such as Twitter. These approaches, known as sentiment analysis, replies on the attitudes of several key figures or successful analysts in the markets to interpolate the minds of general investors. Despite its success in some occasions, sentiment analysis may fail when some of the people are biased, or positive opinions follow past good performance instead of suggesting promising future markets.

1.2 Dataset description

Here we are using dataset from Yahoo and we have choose BMW as our company. Here the start date of the dataset is from 2000 to 2021. Larger the dataset more accurate the model. As whole technique is predicting prices based on data. We didn't require to clean data as it was clean and variables were also less only date, closing prices and volumes of shares. We directly extracted the data from the yahoo by using the libraries of python which was very easy to use. We used libraries like 'pandas_datareader', and within this we used 'data_source' was Yahoo. Start dates and end dates were also specified within this function. All raw data will be managed by Python. We will also use existing python libraries for designing a machine learning model. Our accuracy mainly depend on the amount of data we are provided hence we will try to extract maximum amount of data for making the model accurate. Also try to predict the patterns and what factors affect the pattern by putting in our machine learning model. All techniques will be referred from previous machine learning model. All algorithms will be referred from previous published machine learning paper and will test on various data sets.

For understanding the data and the closing price we plotted the basic relationship of closing price for the graphs to analyse the basic trends. Size of data is also very important for the formulation and accuracy of the model. Hence we took largest possible data set from where the company was listed to get more accurate model and results. Hence all trends and graphs are plotted. And base on this model is formulated and conclusion is drawn.

1.3 Machine learning model and working

Our program uses an artificial recurrent neural network called Long Short Term Memory(LSTM), to predict the closing stock price of a corporation (BMW), using the past 60 days stock price. Unlike standard feet for neural networks LSTM has feedback connections it can not only process single datapoint like images but also entire sequences of data such as speech or video. LSTMs are widely used for sequence prediction problems and have proven to be extremely effective and the reason it works so well is because the LSTM is able to store past information that is important and forgive information that is not important.

The machine learning libraries that we used were "MinMaxScaler" from sklearn, to scale our dataset for the model. The model used was "Sequential" from keras and layers added to the model for our analysis was "Dense" and "LSTM" again from keras.

Using datareader from pandas, we were able to extract the data from yahoo finance for the BMW stock. Our data frame is shown below.

Date	High	Low	Open	Close	Volume	Adj Close
2000-01-03	32.000000	28.660000	31.000000	29.490000	1335338.0	16.296951
2000-01-04	29.600000	27.730000	29.500000	28.299999	1564056.0	15.639324
2000-01-05	28.100000	27.000000	27.500000	27.740000	1359518.0	15.329853
2000-01-06	28.400000	27.000000	27.270000	27.650000	1064746.0	15.280117
2000-01-07	28.299999	27.270000	27.760000	27.600000	1575704.0	15.252487
...	...	...	...	...	...	...
2021-12-23	89.459999	87.980003	88.000000	89.169998	823550.0	89.169998
2021-12-27	90.110001	88.610001	88.720001	90.000000	396304.0	90.000000
2021-12-28	90.690002	89.750000	90.089996	89.949997	442096.0	89.949997
2021-12-29	89.970001	88.870003	89.889999	89.199997	419820.0	89.199997
2021-12-30	89.500000	88.120003	89.389999	88.489998	598323.0	88.489998

5625 rows × 6 columns

Fig. 1

Next, we visualized the data using matplotlib to see the trends in the dataset. This visualization was for intial understanding patterns and change in price for future approach for our model. The visual representation looks like follows:

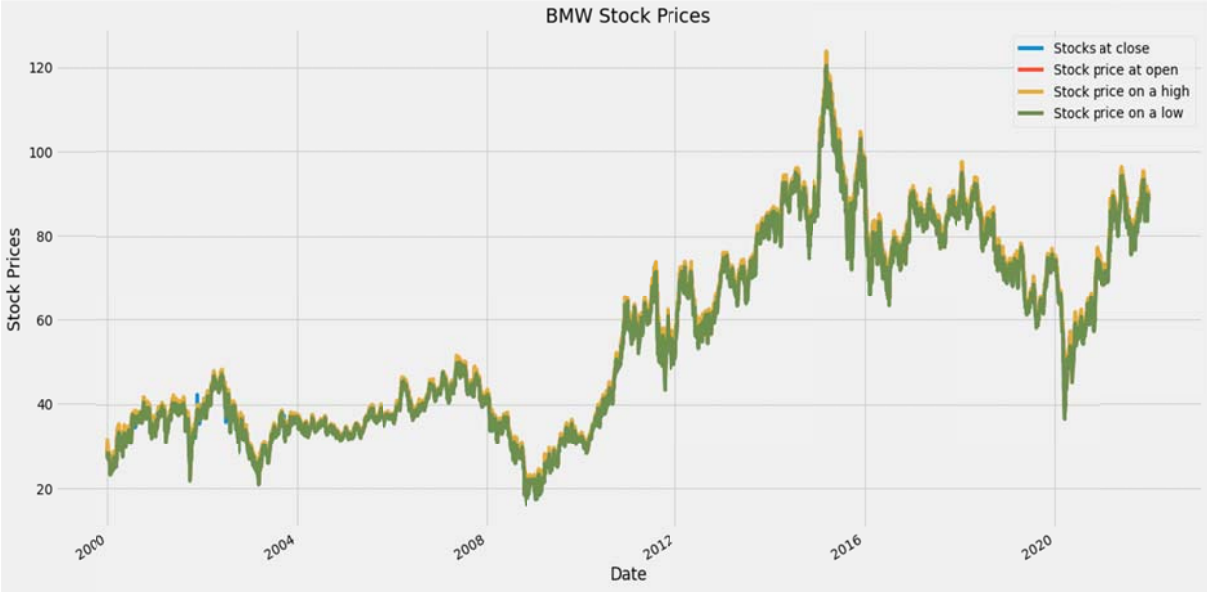

Fig.2

As we can notice the above graph looks very chaotic, and for our analysis since we are only concerned with the closing price of the stock, we edited the graph to just display the closing price and the result is shown below.

Fig.3

We can notice the trend in the closing price of the BMW stock over the years. It has mostly been increasing, with little dips in the prices every 4-5 years, but as we can see if we would have purchased a BMW stock in the year 2000 at the price of about 30 dollars, our money would have almost tripled by now.

We than split the data into training dataset and testing dataset, with 80% of the dataset being the training data which will train our model. After which we scaled the data using the "MinMaxScaler". It is usually a good practice to perform pre-processing on the input data, like scaling or normalization, before using it for a model. We scaled our data to have values between (0,1). "Fit_transform" was used to get the maximum and minimum value for scaling the dataset.

```
array([[0.11794239],
       [0.10666918],
       [0.10136414],
       ...,
       [0.69069721],
       [0.68359225],
       [0.67686622]])
```

Fig.4

Then we created our training datasets, with “x_train” being our dependent variable and “y_train” being our dependent variable or our prediction. With each pass through it will take the past 60 days for the x_train and for every 60 days a prediction value will be stored in y_train variable. Since LSTM model only accepts 3 dimensional inputs in terms of number of features, samples, and time stamps, and our input is only 2 dimensional, we converted it to 3 dimensional using the “np.reshape” function. We then created our sequential model with 1st layer being the LSTM with 50 neurons, return sequence being true and shape of 60 timestamps and 1 feature. The 2nd layer is another LSTM with 50 neurons with return sequence being false. The 3rd layer is a dense layer with 25 neurons. The 4th layer is dense layer again. Next, the model was compiled using ‘adam’ optimizer and RMSE (root mean squared error” as loss to check for the accuracy of the model, the closer the value is to 0 the more accurate our model is. The batch size and epochs for our training dataset is 1. Epochs is the number of iterations when an entire dataset is passed forward and backward through a neural network.

Next, we executed the model and calculated the RMSE score.

```
#Get the models predicted price values
predictions = model.predict(x_test)
predictions = scaler.inverse_transform(predictions)
```

```
#Get the root mean squared error (RMSE)
rmse = np.sqrt(np.mean(((predictions - y_test)**2)))
rmse
```

```
1.7249846735190644
```

Fig.5

As, you can see our RMSE score is 1.72, which is very close to zero signifying a very high accuracy of our model. The performance can be visualized in the graph below. According to our findings it seems that size of data set also plays an important role in measuring the error and it seems larger the data set more accurate the model. Here the root mean squared mean error method is being used to calculate the error. This value is pretty small which means our model has very less error. Larger the value the more unstable is the model. If rmse score is 0 means the model is perfect. Results are also being compared with present day prices to validate the model. Here rmse will always be positive indicating predictions will always a bit higher.

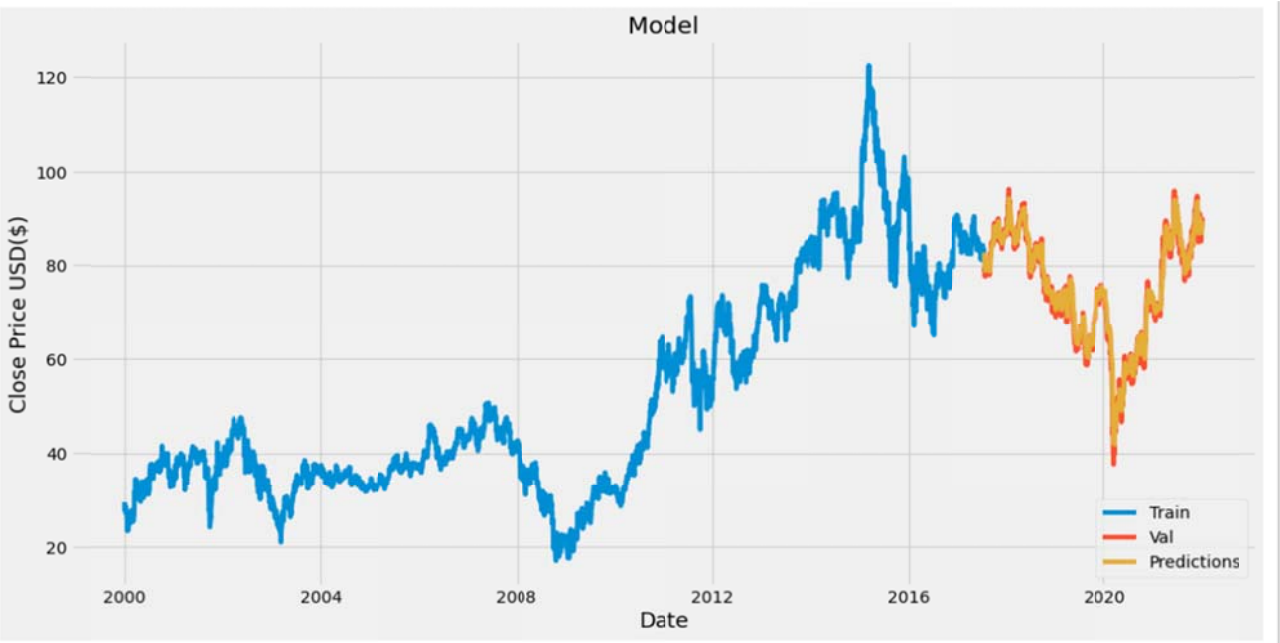

Fig.6

The predicted values shown in yellow are very close to the actual values shown in red. The numbers can be seen below. Here graph is plotted to have an idea about actual value and predicted one and it can be visualize that as lines are very close to each other which is self explanatory that there is very small error.

Date	Close	Predictions
2017-07-24	78.940002	82.803818
2017-07-25	79.430000	81.555428
2017-07-26	79.110001	80.719612
2017-07-27	78.500000	80.182396
2017-07-28	77.849998	79.740059
...	...	...
2021-12-23	89.169998	87.149124
2021-12-27	90.000000	87.890411
2021-12-28	89.949997	88.785552
2021-12-29	89.199997	89.394554
2021-12-30	88.489998	89.477646

Fig.7

For one final test we tested our model for a very recent date stock of BMW. Our model predicted the stock price for 29th April to be 77.85 USD, whereas the actual stock price come that date was 78.51 USD, indicating a very low error and high accuracy of our model. The same can be seen in the image below.

```
#Get the predicted scaled price
pred_price = model.predict(X_test)
#undo the the scaling
pred_price = scaler.inverse_transform(pred_price)
print(pred_price)
```

```
[[77.855545]]
```

```
# Get the quote
bmw_quote2 = web.DataReader('BMW.DE', data_source='yahoo', start='2022-04-29', end='2022-04-29')
print(bmw_quote2['Close'])
```

```
Date
2022-04-29    78.510002
```

Fig.8

1.4 Result and Conclusion

As we performed and design our whole model using LSTM technique it seems this is very accurate techniques for prediction of stock market analysis and we use root mean square error method to calculate the error which is also called as RMSE score. Here the analysis is done on BMW stock. And after analysing it seems that length of dataset also plays an important role in accuracy of the model. And on analysing basic trends there was a bit change in closing price over a period.

Statistical results were as follows as we got RMSE score as 1.72 as lower the RMSE score more accurate the model and it can't be negative as we are using mean square method and RMSE score be 0 means the model is perfect. As for the BMW data set our score was 1.72 which is 90% accuracy of the model. As this is self explanatory from the graph(Fig.6) as lines coincides with the actual value and predicted value which indicates there is very less error. Here almost 20% dataset is used for validation with the actual market value. On taking a random date of 29th april and comparing price it was found the actual price was $78.51 and our predicted price was $77.85 which is pretty close with very less error margin.

References

1. https://ieeexplore.ieee.org/abstract/document/8862225

2. https://link.springer.com/chapter/10.1007/978-981-10-7245-1_38

3. https://www.sciencedirect.com/science/article/pii/S0957417422001452

4. https://ieeexplore.ieee.org/abstract/document/7783235

5. https://neptune.ai/blog/predicting-stock-prices-using-machine-learning