Ryan Senoune

260989415

Deliverable 3

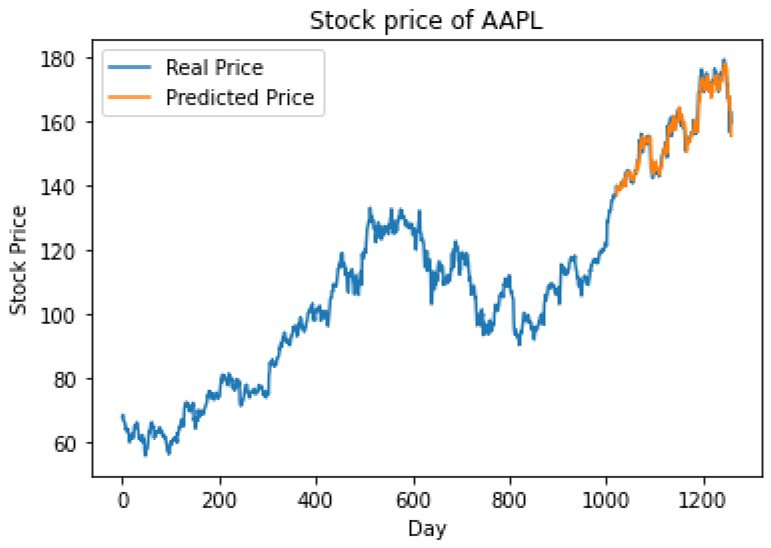
Final training results

In order to increase the accuracy of my model, I added additional features to my data. In total, I experimented with 89 new features, all of them being technical indicators. I used a Python library named “ta”, which allowed me to derive new metrics from my data such as volatility, momentum, trend and more. At the end, I decided to keep the following 9 features: open price, close price, low, high, volume, and four technical indicators related to momentum.

I shifted the goal of my project, instead of predicting the price of the S&P500 index fund as a whole, my model will predict the price of each individual stock within the index. So, we can feed the data of any of 500 stocks to my model and get the forecast of its price.

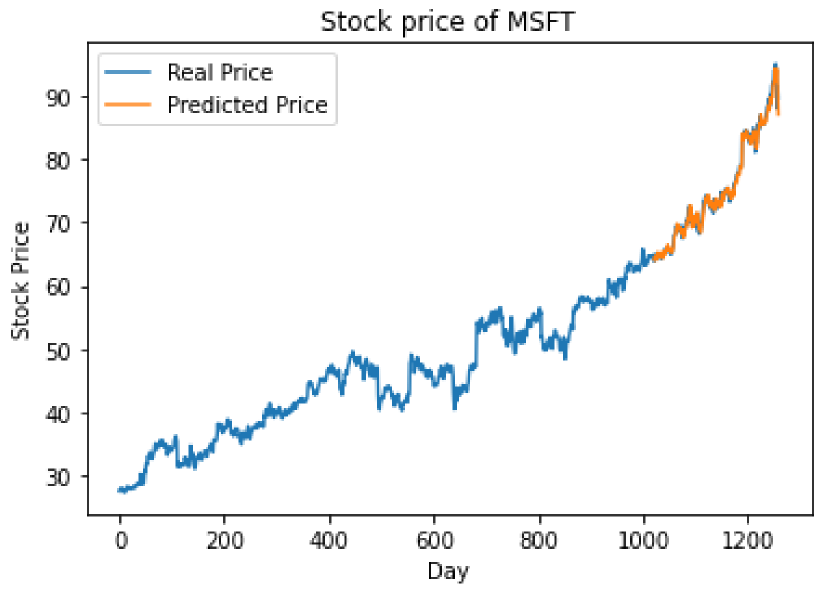
I trained my model on the stock prices of 468 stocks within the index. I removed additional stocks which I think did not have enough data. Here are the predictions on two example stocks along with the mean squared error.

```
Mean Squared Error
Training Score: 99.9731588801892
Testing Score: 156.14976347177497
```



```
movement accuracy: 49.789029535864984%
```

```
Mean Squared Error
Training Score: 46.047796408642895
Testing Score: 74.94422878985333
```



```
movement accuracy: 56.9620253164557%
```

As we can see, my model is quite accurate. I predicted the price of all 504 companies within the index fund and calculated the mean squared error for each of them. The average mean square error for a company is

```
Average mean squared error for training data: 79.6556405102408
Average mean squared error for testing data: 101.18239262826874
```

Even if my model is accurate, I would like to know if it is able to predict the price movement of a stock (positive or negative). On its own, the mean squared error has limited meaning because even a simple model could achieve a high accuracy. For example, if the model only outputs the same price as the last day before the prediction, it would still get a high accuracy, as the stock prices don't typically move that much day to day.

I found that my model was able to predict the correct movement of a stock price 49.8% of the time. If I have the time to tweak with the model more, I'd like to improve the movement accuracy even if it comes at the cost of the mean squared error accuracy.

Final demonstration proposal

I do not have any experience with web development, so I still have some research to do. For now, my plan would be to have a landing page where users can select one stock in the S&P500. Users would be shown the short-term forecast of the stock price.

In order to make the prediction, my model would require the latest information about the stock. I think I could scrape that information from a website like yahoo or any other platform that has an API. Since my model uses the past 14 days to make a prediction, I won’t need that much data per prediction.

If I have the time, I could also show a slightly more long-term forecast of the stock. I would use the LSTM model to generate the predicted price for many days and make a graph.