Ethan Paek & Brandon Quant

Dr. Yi Fang

COEN 140 - T/Th 10:20 AM -12:00 PM

12 June 2020

Final Project Report

Introduction

For our project, we wanted to create an accurate stock market predictor based on historical stock price data from a particular company. By using techniques that we've learned in class and ones outside of class, we hope to find a method that results in a low RMSE and accurately predicts the price of any stock on a given day.

There are two main reasons as to why we were motivated to do this project:

1. Lots of numerical data readily available

Although we will elaborate on this in the Datasets section, Yahoo! Finance allows anyone to download all historical data of any stock as a CSV file. There are thousands of stocks on the stock market; some companies have even been active in the stock market for several decades. Thus, by selectively choosing the proper companies, we are able to use thousands of numerical data points at our disposal for training and testing.

2. Creating an accurate stock price predictor results in smarter investment decisions

If we are able to create an extremely accurate stock market predictor, this would allow us to easily make better decisions on which stocks we should invest in. By evaluating which stocks will do poorly and which stocks will do well, this would give

us an indication of which companies' stocks to buy or sell. In other words, the successful prediction of a stock's future price could yield significant profit for us in real-life.

Related Work

Linear regression and ridge regression is explained through the material covered in the class, which is why we do not include outside information on these models. Long Short-Term Memory (LSTM) is a recurrent neural network that is capable of learning long-term dependencies. LSTM is able to remember information over long periods of time and is good for time-based data. In deep neural networks, when running the backpropagation algorithm to update the weights of the nodes, we may run into the Vanishing Gradient problem or the Exploding Gradient problem, which is when our values get exponentially smaller or larger. This causes problems in our training data and thus LSTM was created as a solution to the short-term memory problem.

We want to use data from 1-2 months from the past in order to predict the closing price of a specific day, which is why we thought LSTM would be a good model for stock market predictions. Listed below are references that explain LSTM more in-depth:

- https://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21
- https://medium.com/deep-math-machine-learning-ai/chapter-10-1-deepnlp-lstm-long-short-term-memory-networks-with-math-21477f8e4235

Methodology

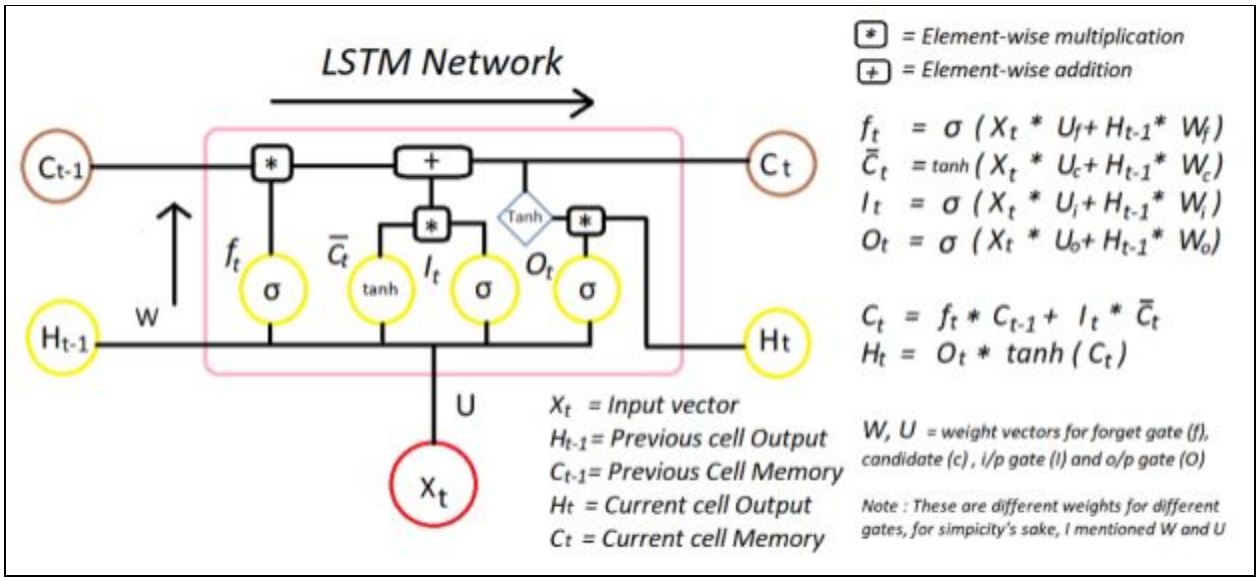
To begin, we created a simple benchmark to compare with our machine learning models by calculating the mean closing price of the last 30 days for a given stock. We could then compare these results with the actual closing price for a particular day for our training and testing datasets.

Next, we implemented linear regression as our first machine learning model since it is fairly simple to understand and apply. In addition, since our data is based on continuous variables, we believe that this was a good statistical data analysis tool to start off with. To calculate our predicted closing prices we used the following formula: $\mathbf{w = (x^T x)^{-1} x^T y}$, where w = our predicted closing prices, x = the dataset of all traits that correspond to a certain day's closing price, and y = the dataset of all closing prices.

After conducting linear regression on each of our chosen stocks, we found that there was a significant issue of overfitting on our predictions. Thus, we then chose to implement ridge regression to try and resolve this problem by using the following formula: $\mathbf{w = (x^T x + \lambda I)^{-1} (x^T y)}$, where w, x, and y are the same variables as linear regression, I = the identity matrix for our respective x, and λ = the optimal lambda that was found through k-fold cross-validation. To implement k-fold cross-validation, we used a k of 5 as we did in our labs. To find the optimal lambda, we initialized λ as 50 and divided the value in half over 15 total iterations. Then, we

calculated the RMSE for each of our lambdas which is defined as the following: RMSE = $\sqrt{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2/n}$ and set our optimal λ as the one with the lowest RMSE.

Finally, we implemented LSTM by using TensorFlow. To do this, we began by normalizing all data points for a respective stock between 0 and 1. We created a new three-dimensional training dataset for each of our respective companies with 60 timesteps and used this data to train our model which can be visualized as the following:



After this, we gathered our predicted closing prices and compared them with our actual closing prices.

To compare our predicted closing prices and actual closing prices on our testing sets with each of our machine learning models, we used RMSE.

Datasets

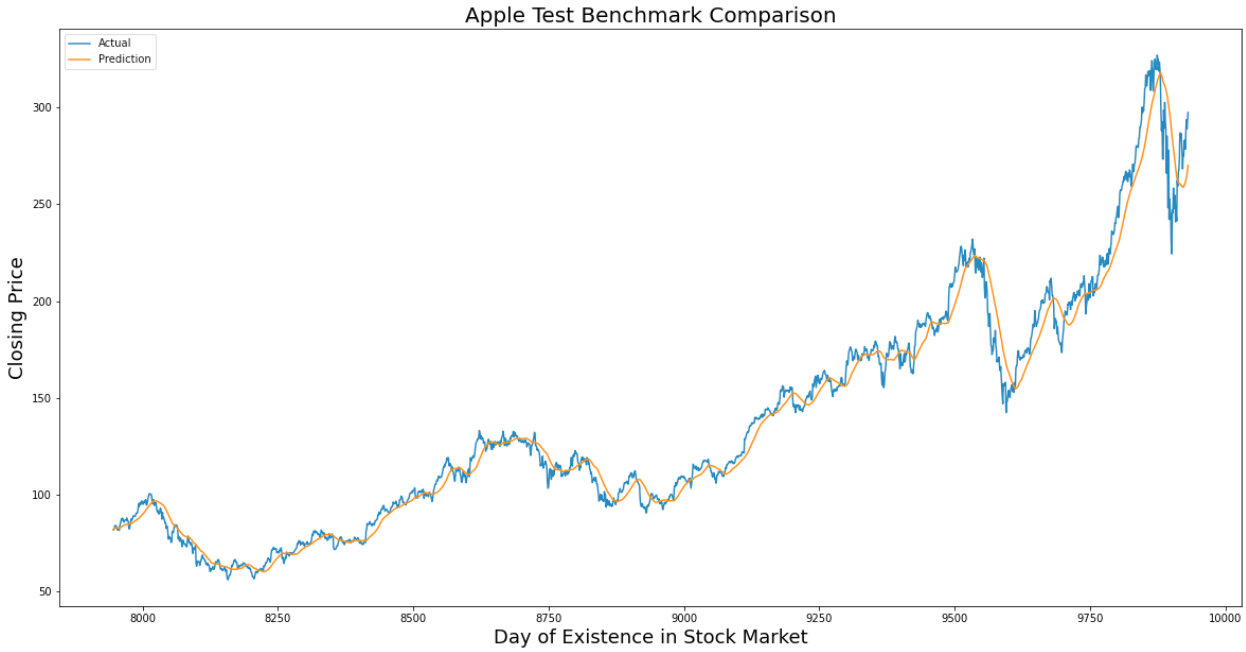
We first started with the historical stock prices for Apple, Intel, and Microsoft, which can all be downloaded as CSV files from Yahoo! Finance. For each of our stocks, our training datasets are based on the first 80% of its existence on the stock market and the other 20% as our testing set. Our training size ranged from 6800 instances to 8000 instances and our test size ranged from 1700 instances to 2000 instances. There are 6 features included in the CSV files, which are the Date, Open Price, High, Low, Closing Price, and the Adjusted Closing Price. Because we thought that these 6 features were not enough, we added 4 more features to our dataset. These features were the mean of the last 7 days, the mean of the last 30 days, the largest difference in closing prices of the last 7 days, and the largest difference in closing prices of the last 30 days. We dropped the Date feature since our data was already sorted in chronological order and we need continuous values for regression models. Here is an example of Apple's overall stock price since its day of existence:

Experiments

It is important to note that we have decided to only include our graphs for the Apple stock since there are a total of 12 graphs which we feel is too many for this report.

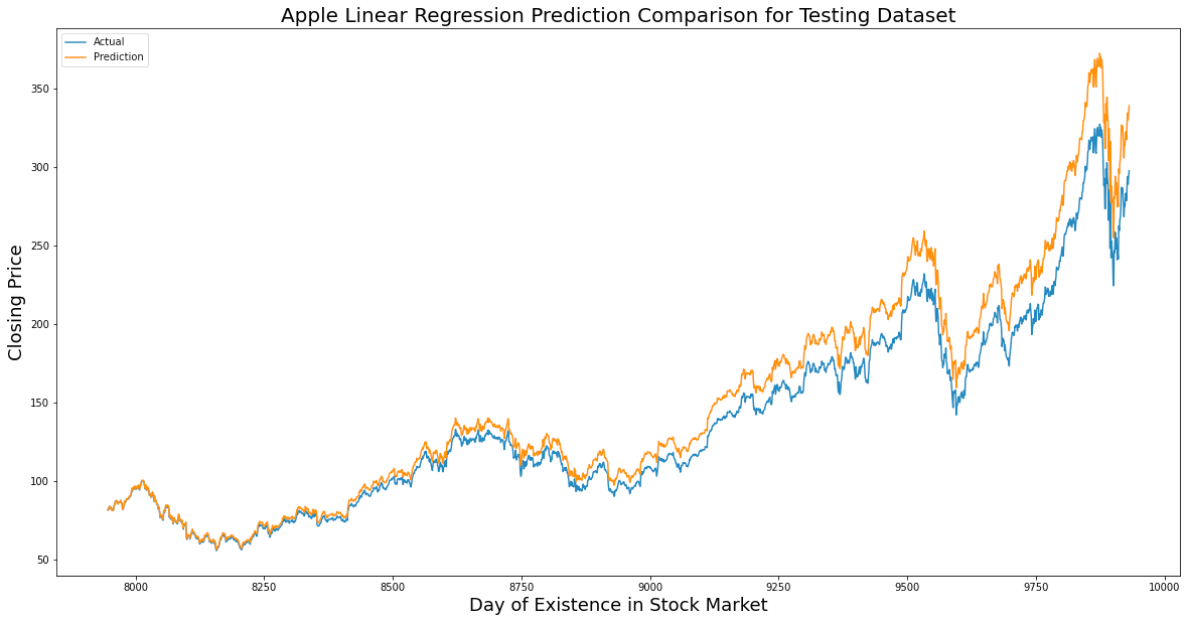
Benchmark

	Training RMSE	**Testing RMSE**
Apple	1.059	8.919
Intel	1.671	2.095
Microsoft	1.483	4.197



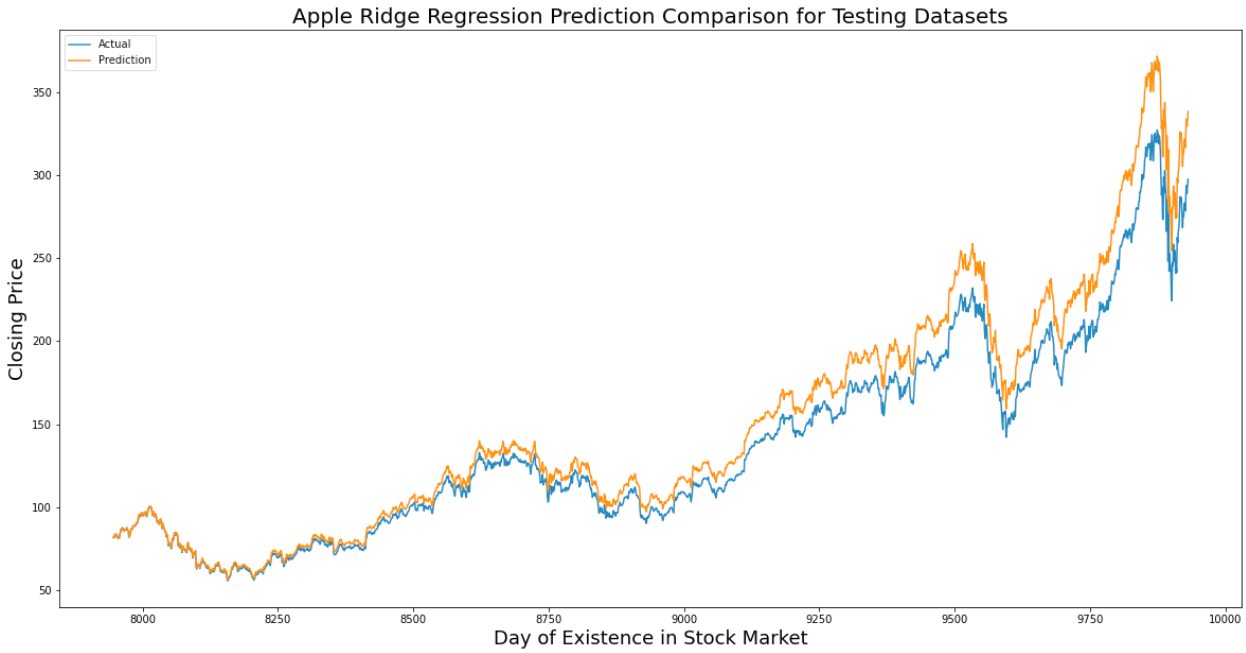
Linear Regression

	Training RMSE	**Testing RMSE**
Apple	0.0343	16.4502
Intel	0.2026	0.7505
Microsoft	0.1801	0.6455



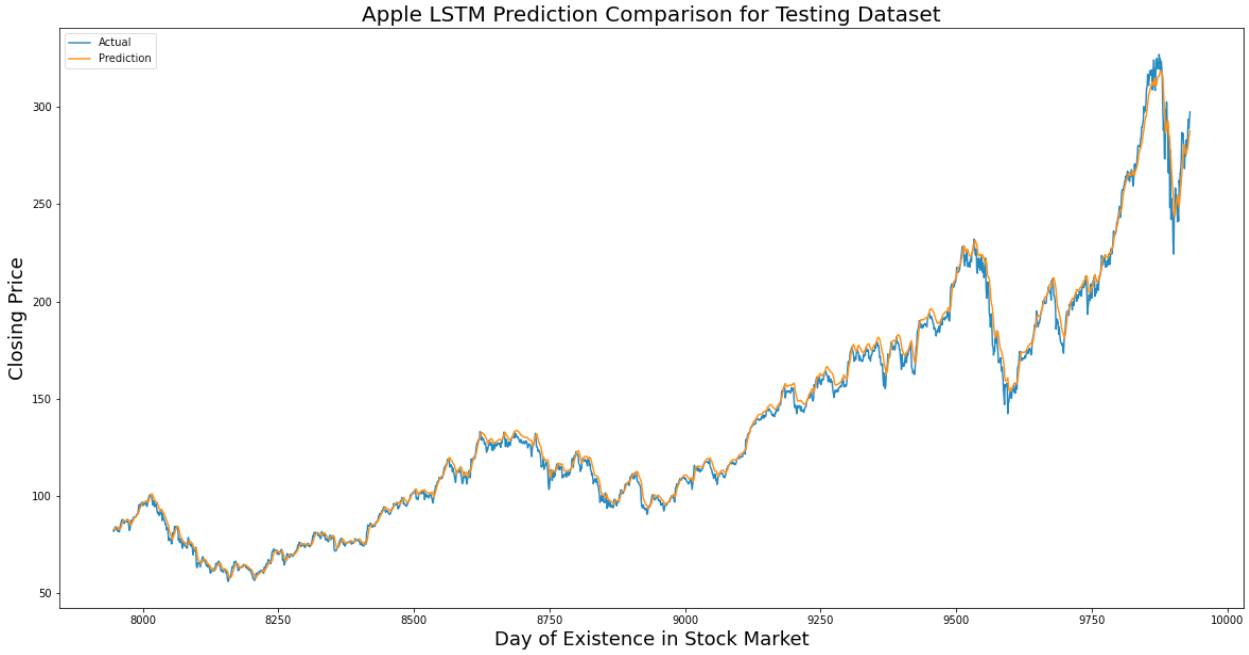
Ridge Regression

	Training RMSE	**Testing RMSE**
Apple	0.0343	16.2047
Intel	0.1985	0.6657
Microsoft	0.1801	0.6454



LSTM

	Testing RMSE
Apple	4.4270
Intel	0.5943
Microsoft	0.5732



As we can see, with each implementation, we were able to lower our RMSE for each of our stock datasets. Clearly, we can see that our models outperformed our benchmark, except for the case of Apple's stock, which only outperformed when we used LSTM. We believe that the main reason for this is because of an incredibly low RMSE in regard to the training dataset when we fit our models since it remains a fairly flat line over this time period.

Conclusion and Future Work

From our results, even though LSTM performed the best, we would not use this model in real life because of the high RMSE values. We learned that even complex models such as LSTM are not good at predicting the stock market and that predicting the stock market closing prices is a difficult task.

For further work, we would like to explore longer forecast horizons, where we would use more information over a longer period of time to see if that would improve our results. Additionally, we would want to experiment with other more complicated machine learning models that would perform better than our models used. Lastly, we would want to incorporate news classification for each respective company into our models. Since positive or negative news normally correlates to an increase or decrease in the stock prices, adding this to our model would definitely help improve our results.