Can a Long Short-Term Memory Model Produce Accurate Stock Price Predictions?:

A Deep Learning Approach to Predicting Apple Inc. Stock Price

Shadi Chamseddine
Department of Economics
Carleton University
100937807

Nick Dirienzo
Department of Health Sciences
Carleton University
100978404

Christopher Lee
Department of Economics
Carleton University
100937241

Abstract—**Since the inception of the first stock market, various models, techniques and methods have been developed to try and gain an edge in stock investing. Yet, the ever changing and complex nature of the stock markets have rendered these approaches unreliable. With the tremendous growth in computing power today, a new approach to cracking the code for stock price prediction has started to take shape. Traders have begun to explore computational solutions in an attempt to predict movements in stock prices. However, to date, there has been a lack of innovation in the application of deep learning techniques that make accurate predictions feasible. We propose a Long Short-Term Memory (LSTM) model approach to address this innovation gap. The LSTM is developed to capture pertinent underlying data related to stock price trends based on historical stock movements, as well as key features drawn from correlated assets and technical indicators. We show how this deep learning recurrent neural network model is able to accurately predict the trends in future stock price movements. We obtain a forecast error of just 3.1% for our LSTM predictions, which is considerably more accurate than the forecast error of the current standard benchmark ARIMA model which comes in at 6.8%.**

INTRODUCTION

The mechanisms of various stock markets have long been studied by investment brokers, bankers, and individual investors alike in an effort to maximize the returns on their investments. To this stage, there have been a variety of techniques, models, and methods used to try and gain an edge in the stock market, yet all have been proven to be unreliable due to the ever changing and complex nature of the stock market itself. As such, there remains a gap between the existing models that fail to address the intricacies and flows of the stock market and those that can self update and adjust as real time information flows through the system.

With the remarkable growth in computing power over the past decade, many traders have begun to explore computational solutions to try and crack the stock market puzzle. Though, what is evident from the current literature is the majority of stock market "prediction" models that are used today rely on historical data and predict solely based on past market behaviour. This results in models that can only "predict" events that have occurred in the past. As a result, these models perform poorly for actual stock price prediction. For this reason, we wish to improve such methods to produce real and accurate stock price predictions.

The aim of this study is to take the current analysis found in the literature a step further and actually predict stock price movements beyond the historical dataset. We are thus interested in determining whether or not data science techniques such as deep learning can be applied in a financial setting to provide insights into the data that can assist in accurately making future stock price predictions.

LITERATURE REVIEW

Our study begins with an extensive review of the existing literature and work done in the area of machine learning stock prediction algorithms. Through this research, we determine which existing elements may be useful to incorporate into our study, and where we can make our own contributions to this literature. We find the majority of machine learning stock market "prediction" models that exist today rely on historical data that require a training dataset and testing dataset. The main limitations with these models is that they cannot predict beyond the historical dataset, and thus, they do not have practical use beyond predicting events which have already occurred. This generally makes them weak prediction models. We find this to be the result of insufficient innovation in modifying existing machine learning techniques to allow for prediction capability to date. However, this does not mean that the literature itself is weak. In fact, it provides a solid foundation in an emerging market for data science, and serves as the preliminary basis for our work, while leaving us with room to make our own contributions to the literature.

Hargreaves and Hao [1] explored whether the application of a personal trading strategy or a data mining approach will outperform the Australian stock market over a certain time period. To narrow down the scope of their research they identified the best performing sector by comparing all sector price trends against the ASX (Australian Ordinary Index) for the latest 3 months. This turned out to be the industrial sector. From here they used 4 different stock selection strategies - a personal trading strategy, a

price trading strategy (C5.0 decision tree), a growth trading strategy (CHAID decision tree), and a growth and value trading strategy (neural network). These strategies used the same 5 variables for consistency, each strategy chose the best 6 performing stocks from the industrial sector. They found each of these strategies performed well with high sensitivity rates, signalling a high rate of correctly classifying stocks with an increasing trend. In addition the returns from each of these strategies are higher than the return from the ASX. The authors show how data mining techniques can be used to create classification models with stock data from the industrial sector in the ASX.

Liu and Malik [2] proposed an alternative method of stock market prediction using data mining and Neural Networks (NNs). The proposed framework begins with choosing the optimal stock choice. Within the New York Stock Exchange (NYSE) there are many stocks to choose from, thus an easier method would involve choosing the highest performing sector, a composite index of sorts. Exchange-Traded Funds (ETFs) were selected as they are mostly used to track a basket of assets of a specific sector, albeit still traded like a stock on an exchange. Split amongst sectors, Liu and Malik chose the most stable ETF, the SPDR Consumer Staples (XLP) ETF as it outperformed most sectors both before and after the 2008 market crash. The NN framework involved an input layer of 22 neurons and an output layer of 2 nodes. The 22 neurons include the previous 13 weekly Closing prices, in addition to the Open, High and Low prices of the last 3 weeks. The outputs are split into the Action node and Profit node. The Action output recommends either a buy long, sell short, or no action, while the Profit node provides the predicted profit of such an action. The NN was trained on approximately 3 years of historical data. While the predicted profits only had a correlation coefficient of 0.11 with the true profits made on investments, this proposed framework is a good start in the goal of producing a buy/sell strategy to optimize overall profits as when to buy, hold, or sell are all equally important. Authors demonstrated 1. NNs can be used to approximate the non-linear relationship between historical data and future performance of stock prices, and 2. A new framework based on NNs to invest in the stock market.

As outlined in the reviewed works, data science applications to this point have seen general success in the financial market, and therefore it gives us confidence in being able to produce positive results with our stock market prediction models.

DATA

Our primary data of interest for prediction is the Apple Inc. stock which is traded on the Nasdaq stock exchange. We chose Apple Inc. stock because it is a well established company on the stock exchange with decades worth of data. We data scrape the daily closing stock price for Apple Inc. stocks from Yahoo Finance for the period of 1990 to 2019, which provides us with 30 continuous years of the Apple Inc. daily stock data. This allows us to sequentially segment the data into a training set (using the first 80% of the data) and a testing set (using the last 20% of the data), while also encompassing a large time period. The ordering of the data is important because although stock price movement has some inherent randomness, it is not completely random as outlined by the random-walk hypothesis. The stock price in period T+1 is dependent on many factors, including its price at many periods into the past. For example, if a stock has been overvalued, its previous prices matter immensely for what its stock prices will eventually become.

To supplement our data analysis, we will incorporate correlated assets as features into our models. Correlated assets involve the movement in stock price of other businesses in the same industry, or movement of composite indices. The stock prices we will include in the analysis are those of rival companies in the same industry such as Alphabet, Microsoft, LG, and Samsung. This is because competitors can have a direct impact on each other's stock prices and product releases. For our composite indices we will use the market index values for the Dow Jones, NASDAQ Composite, and S&P 500. Dow Jones, is a stock market index which measures the stock performance of the 30 largest companies listed on stock exchanges in the United States. The S&P 500 is a stock market index which measures the stock performance of the 500 largest companies listed on stock exchanges in the United States, and the NASDAQ Composite is a stock market index of the common stocks and similar securities listed on the Nasdaq stock market. These are the 3 most followed indices in US stock markets and we will use this data because it gives a general idea of the overall movement in the stock market. When these indices increase, most of the stock market is typically appreciating in value and vice versa, so they provide us with a benchmark to gauge the movement and performance of how the overall stock market is doing.

In addition to our correlated assets, we also include some technical indicators into the model as additional features. Technical indicators include the volatility index of the market, Bollinger Bands, the volume of stocks traded in the market or a particular stock, the 7 and 21 day Moving Averages (MA), the Exponential Moving Average (EMA), and the Moving Average Convergence Divergence (MACD). Bollinger Bands characterize the prices and volatility over time for a stock. Market/stock volume refers to the number of shares traded in a given time period for the market/stock of interest. 7 and 21 day MA takes the arithmetic mean over the past 7 and 21 days. An EMA is a MA which places greater weight and importance on the more recent stock prices. The MACD is a

trend-following indicator which shows the relationship between 2 MAs of a stock's price. The MACD is calculated by subtracting the 26-period EMA from the 12-period EMA. We will also use the VIX volatility index, which is a popular measure of the stock market's expectation of volatility. These will all give us useful insight into technical and fundamental factors that will impact stock price movements, and also help us control the importance of different factors to the analysis. The data for all of these aforementioned features are available on Yahoo Finance and are scrapped using the "Quantmod" package in R.

DATA ANALYSIS & METHODOLOGY

Prior to exploring the methodology of our models, it is important to highlight the statistical robustness checks performed in order to ensure the validity of our models given the aforementioned dataset. As our models are fed a variety of different data vectors that make up our Apple Inc. stock price and data features, we are presented with the challenge of having to ensure our models do not become susceptible to issues with biasness, consistency, and efficiency which can adversely impact our prediction accuracy. As such, we conduct robustness checks to ensure our models will not suffer from issues such as autocorrelation, non-stationarity, and correlation between our residuals.

As mentioned, the price of a stock tomorrow is dependent on the price of that stock many periods into the past. As such, we can make a strong logical assumption there is going to be high levels of autocorrelation between stock prices across many periods of lags. Figure 1 plots the autocorrelation function (ACF) which allows us to statistically confirm our assumption of autocorrelation. We can see the autocorrelation between our stock prices is tremendously high (close to 1) even across 30+ lags (over a month into the past). This suggests we have strong autocorrelation between our stock prices that we need to account for when choosing and developing our prediction models.

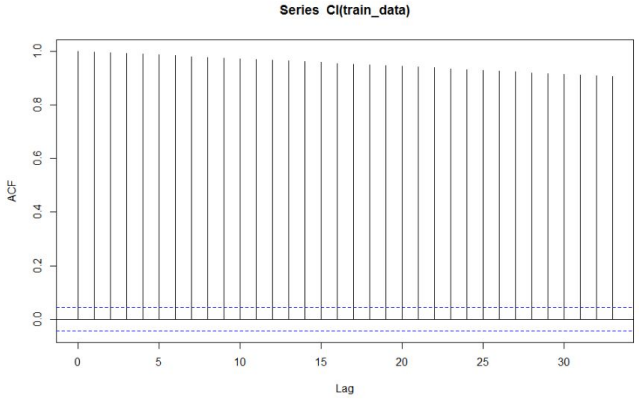

Figure 1: Autocorrelation Function

The partial autocorrelation function (PACF) on the other hand specifies the correlation between the residuals across periods. Just like we do not want to have autocorrelation between our explanatory variables, we also do not want our residuals to be correlated across periods. Unlike the ACF however, it is not intuitively evident whether or not residual correlation will be an issue in our models, so we will need to statistically test this with a PACF. In our case, plotting the PACF in Figure 2, we can see the autocorrelation between our residuals quickly declines after the first lag. Since this decrease is substantial enough that the PACF value falls between the critical values indicated by the 2 blue dotted lines in Figure 2, it indicates partial autocorrelation will not be a problem in our models.

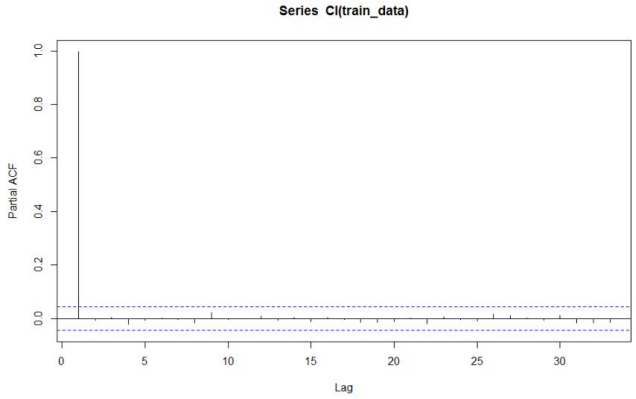

Figure 2: Partial Autocorrelation Function

Another key issue we need to be careful of is non-stationarity. Stationarity itself ensures the process generating our time-series does not change over time. Across a time-series dataset such as a stock price or composite index, the parameters we may be interested in modelling such as the mean or variance typically change over time. However, to be able to accurately make predictions through our models, we need to pull out the patterns, trends, and associations in the data, which requires our data to be stationary. When looking at the movement of stock prices, it is clear both visually and intuitively that our dataset is non-stationary and we will need to account for this issue. Once again, in the interest of being statistically methodical, we perform an Augmented Dickey-Fuller test to confirm our assumption of non-stationarity.

The results of the Augmented Dickey-Fuller test in Figure 3 confirm our hypothesis of non-stationary data, as the p-value of 0.6779 means we fail to reject the null hypothesis of

non-stationarity. The silver lining here is that the issue of non-stationarity can be mitigated without negatively impacting our models. To do this, we simply take the first difference of our data rather than the actual stock prices and process those values through our models. When the predictions are completed on the differenced data, we can invert the differencing to get our final results back as a stock price rather than a difference.

```
        Augmented Dickey-Fuller Test

data:  Cl(train_data)
Dickey-Fuller = -1.7647, Lag order = 12, p-value = 0.6779
alternative hypothesis: stationary
```

Figure 3: Augmented Dickey-Fuller Table

The final robustness check we want to investigate prior to modelling is residual correlation. Having a large R-squared value does not necessarily indicate a good time-series model, as we could just be overfitting the model, so we need to look further. Having correlated residuals in time series analysis means inferences about levels, means, variances, or any other types of parameters we may be interested in modelling may be spurious with an unknown direction of bias, even with a large R-squared. In our case, the ordering of data is meaningful which would make residual correlation a significant issue that would result in our models being highly inefficient. It is also important to note this inefficiency, which results in a very high prediction variance does not decrease, even if we increase the sample size. Therefore, we need to check for residual correlation using the Ljung-box test.

The results of the Ljung-box test are found in Figure 4 below. From our p-values at the 5, 10 and 15 lag levels respectively, our Ljung-box test fails to reject the null hypothesis that the residuals are random. This indicates residual correlation will not be an issue in our models, and thus we do not need to account for it.

```
        Box-Ljung test

data:  model_fit$residuals
X-squared = 5.4168, df = 5, p-value = 0.3672

        Box-Ljung test

data:  model_fit$residuals
X-squared = 12.838, df = 10, p-value = 0.2329

        Box-Ljung test

data:  model_fit$residuals
X-squared = 23.735, df = 15, p-value = 0.06973
```

Figure 4: Ljung-Box Test Results

As we have thoroughly investigated the data and worked out any issues potentially arising from our modelling, we can move into the methodology behind our study. In this paper, we explore 3 different model specifications:

1.) Benchmark Auto Regressive Integrated Moving Average (ARIMA) Model
2.) Historical Long Short-Term Memory (LSTM) model
3.) Modified LSTM model for future predictions beyond the historical dataset

Benchmark ARIMA Model:

The first model we develop is an ARIMA model. An ARIMA model is a generalization of an Autoregressive Moving Average (ARMA) model, which allows us to model data without a constant mean across the time-series. ARIMA models are typically used to either fit or predict future values of time series data, which makes them an appropriate model to begin our analysis with. In general, ARIMA models are highly regarded and commonly used for time-series modelling, and often do a good job at making short-term predictions. Therefore, we use it as the first model to generate a benchmark prediction that we can compare our LSTM to.

Our ARIMA model specifications are chosen via an Auto-ARIMA function that cycles through different model specifications to find the one that best fits the dataset. The three parameters which the model tries to fit are the regression of current price on past price (the number of auto-regressors), the degree to difference the data to make it stationary, and the calculation of past forecast errors.

In the case of our model, it is optimized with a (0,1,0) specification which means we have:

- 0 auto-regressors,
- A difference of 1 (first differencing the data),
- And a moving average with an order of 0

Historical LSTM Model:

Prior to jumping into LSTM models, it is important to have some background knowledge on how the NN architecture behind the LSTM works. A neural network maps input data points to output data points which have been processed through various abstract layers in the model. For LSTM models, they are a type of recurrent neural network with many similarities to the NN's explained above, but with a few key differences that are important for our analysis. First, unlike typical feedforward NN's, LSTM models

also have feedback connections which essentially gives it a long-term memory that can maintain and call-back data and results from previous periods. The second key difference is that LSTM models can go beyond processing only single data points, which allows us to process entire sequences of data together. This becomes particularly handy as we are interested in processing the entire sequence of Apple stock prices along with our technical indicators and correlated assets to determine how all of these features interact with one another to impact price trend movements. Within our LSTM, we have an input gate, which decides if the current memory cell state will be updated, the forget state which determines how much of the old state should be forgotten when updating the memory state and then the output state, which returns the result vector. The combination of the input gate and the forget gate eventually flows into our result sequence that returns our prediction.

It is important to note that with an LSTM model, the input sequence can be larger than the output sequence and vise versa. We will make use of this feature by inputting a vector of all of our data as an input, while getting back a single prediction value as our output. Using an encoder-decoder architecture, we can make use of the LSTM to take our sequence of input data, which in the case of this paper is a vector of data which includes the Apple stock price and all of the aforementioned features per day and pass it into the model through the use of the encoder. This vector of data is then activated by our sigmoid activation function and processed through the LSTM model before it is passed to the decoder which takes our processed vector and returns our output sequence, which in this case is the stock price prediction. This model is activated via 3 main layers:

- **LSTM** for adding the Long Short-Term Memory layer
- **Dense** layer to add full connectivity between the neural network layer, where each input node connects to each output node
- **Dropout** for adding dropout layers that randomly remove nodes in the hidden layer to prevent overfitting the data

LSTM Beyond the Historical Data:

For our main contribution, we will take our analysis a step further by modifying our LSTM code to allow for predictions beyond the historical dataset. Much of what was done and explained in the aforementioned section is carried over into this version of our LSTM model, but there are a few crucial modifications which allow us to be able to extend our model beyond the historical data.

First, we will replace the sigmoid activation function in our model with a rectified linear unit (ReLU) activation function for our LSTM training dataset. The reason we use ReLU in this case is because the sigmoid function we used in the historical LSTM does not allow the data to exceed the maximum stock price in our training dataset. We assume at some stage that a future value of Apple stock price may be higher than the maximum stock price in our dataset and thus we need to modify our activation function to allow for that possibility. Next, we will make use of our newly predicted values by inputting them back into our training model as features via an LSTM layer paired with a data feedback loop. So, as future predictions beyond the historical data are developed, the data feedback loop will impute these values back into the training set and use these values to further train the model. We then use this trained model to create predictions for the following periods based on the raw data, trends, and patterns that we can draw out of our previous period predictions. The model applies these adjustments through our LSTM model and ReLU activation functions, learning to improve prediction accuracy through each iteration.

RESULTS

There are a variety of ways that one can assess the prediction accuracy of a model. In our case, we are interested in examining the mean forecast error of our model predictions, which returns the disparity between our predicted stock price, and the actual stock price that we are trying to predict. This measure gives us a solid indication of how closely our predictions are tracking the true stock price both from a directional standpoint as well as a magnitude standpoint. The goal is to minimize the mean forecast error which indicates that our model is doing a good job of making accurate predictions.

We begin with the results of our benchmark ARIMA model, which is plotted in Figure 5 below.

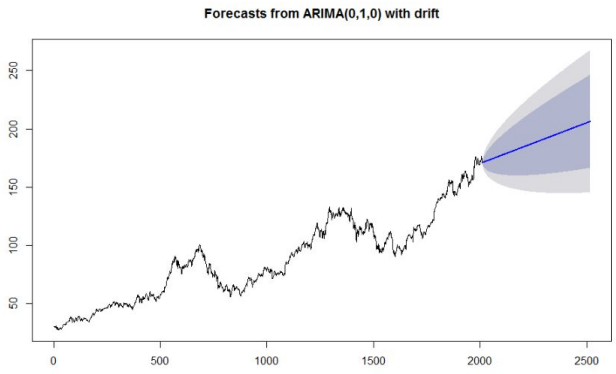

Figure 5: Benchmark ARIMA Model Prediction

We see that the ARIMA model results in a fairly linear prediction (blue line) for the value of future stock prices, with a high degree of variance indicated by the blue shading (80% confidence interval) and grey shading (95% confidence interval). Plotting our

ARIMA prediction results against the true Apple stock prices as shown in Figure 6 allows us to determine the accuracy of our forecast using the ARIMA model. It is evident that the ARIMA model is not doing a particularly good job of predicting the stock price or the movements in the trend of the stock price. Overall, this results in a forecast error of 6.8%. While this is not a large forecasting error in general cases, it can be problematic in the context of stock prediction, where the margins to make a profit on a stock trade can be very small. It is clear the ARIMA model in general will not be good for stock price prediction, however as mentioned it will serve its purpose as a model which we can benchmark against.

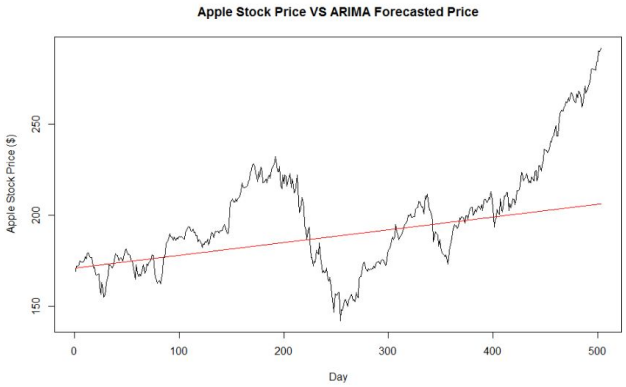

Figure 6: ARIMA Prediction vs Actual Apple Stock Price

Next, we move on to our historical LSTM model. Using our ARIMA model prediction results as the benchmark, we are interested in determining whether our LSTM model will be more successful at capturing the direction and magnitude of the movement in stock prices, as well as producing a lower mean forecast error. We begin by running our sequential model across 30 years of data, using the first 24 years to train the model and the last 6 years to test the model. We then fit our model across 30 epochs and reset the state after each loop.

The results for our historical LSTM model predictions are displayed in Figure 7. With the red line representing the prediction, and the black line representing the actual stock price, we see that the model did an outstanding job at following the trends and magnitude in the true stock price movements. This resulted in a very small mean forecast error of just 0.8%, which is 7.5 times more accurate than the ARIMA model prediction.

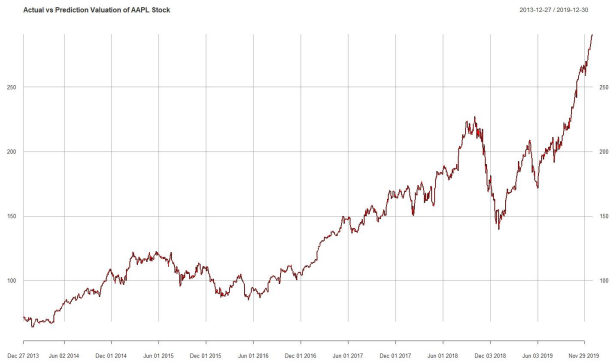

Figure 7: Historical LSTM Prediction vs Apple Stock Price

These results although very precise are also somewhat deceiving. The reason why the predicted price lines up so perfectly with the actual stock prices is because the test data is fed into the model to generate the predicted stock prices. Thus, although the model is learning and making predictions to some degree, a great deal of this prediction accuracy is dependent on the LSTM model fitting it's predictions to the test data, which are the results we are trying to predict. Therefore, these models are completely dependent on knowing the stock price they are trying to predict to make an accurate prediction. It is clear that if these models were able to truly make predictions that accurately, then it would be making many people including ourselves very rich, but that is not the case.

As we mentioned, the vast majority of machine learning stock "prediction" algorithms that we found in the literature use the above method. Therefore, they will get accurate results when trying to predict something that has already occured, but when actually trying to predict into the future the model will produce extremely poor results. Our contribution to the literature is to address this issue by taking the analysis a step further. As outlined in the methodology section, we alter the LSTM algorithm to make real future predictions that are not based on fitting to the underlying data we are trying to predict.

For the purposes of this model, historical data refers to any data prior to December 30th, 2019. Our goal is to predict the daily stock price of Apple across the first 2 months of 2020. We will use all 30 years of our data as training data for our model, with no test set. The lack of test set insures that we don't have any future data we are trying to fit our model to. In the context of this research, we still want to use the first 2 months of 2020 stock price data to test our predictions against. However this data will not be pulled into the system until our model is completely finished making all of its predictions. This way, the model has no knowledge of any stock prices beyond 2019, and we only pull those values in after to compare to our future prediction.

The results from this prediction model can be seen in Figure 8. Once again, the black line represents the actual stock price and the red line represents the predicted stock price.

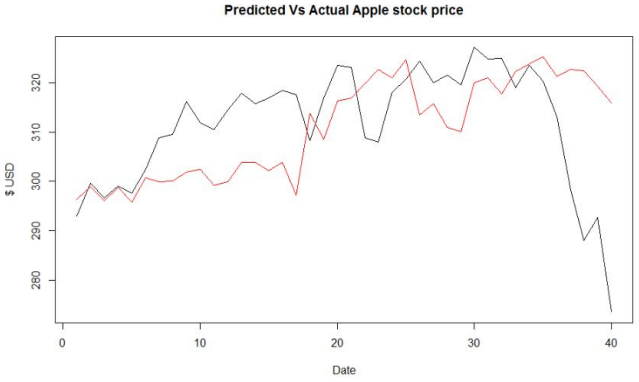

Figure 8: LSTM Prediction Vs Actual Apple Stock Price

We can see that remarkably the model is doing a very good job at picking up the trends in stock price movement, which we believe is a result of capturing the main features that will impact the stock price moving up or down. The only issue seems to be the magnitude in which the stock price moves, and we can attribute that to two main issues. The first is that there are so many factors that can impact the stock price, that we are not capturing all of them as features in our model, and thus the model seems to be underestimating the magnitude of price shifts in most instances. The second issue being that we are not capturing non-intrinsic factors such as the general sentiment of a particular stock in the marketplace which also has an impact on the rate at which the stock price moves. Take the end of Figure 8 for example, where the impact of the global COVID-19 pandemic began to be seen in the stock market. Our model's correlated asset and technical indicator features correctly picked up on the decline of stock price, but the lack of sentiment analysis meant that we did not capture how big of a negative impact the virus would have on the stock price. Overall, we have seen positive results from our stock prediction LSTM model, with a forecast error of just 3.1%, which is over 2 times more accurate than our benchmark ARIMA predictions. In addition, unlike the LSTM that can only forecast based on historical data, this LSTM model is useful for making actual predictions about future values that we have no knowledge about.

IMPLICATIONS

As we have seen, our LSTM model for future predictions is better than both the LSTM that is reliant on historical data, as well as our benchmark ARIMA model. When it comes to our benchmark, the LSTM does a much better job at predicting the stock price, as indicated by the lower forecast error. It also does a better job at forecasting the movements in price trends ultimately making it a better model. When it comes to comparing to the historical LSTM, although it has a lower forecast error, our model actually has the ability to make predictions about future values which are unknown to the model, whereas the historical LSTM model cannot.

The drawback of our model is that it is too reliant on using fundamental features to predict the impact on stock price movement. This presents a question about whether or not the inclusion of some sentiment analysis can help improve the prediction accuracy by both better capturing trend movements as well as the magnitude of those movements. From an economic perspective, stock prices are determined using all current and past information, as well as what people expect to happen in the future. Given this, we would assume that fundamentals as well as sentiment would be priced in before the model is able to act. Another issue we could argue when it comes to sentiment analysis, is that it is not advanced enough at this moment to always determine the true sentiment towards a stock and the type of magnitude that sentiment could have on stock prices. For instance, the COVID-19 pandemic has had tremendous negative effects on the majority of stock prices, however the stock price of companies that provide essential services, internet, streaming services and virtual meeting software among others have benefited from the COVID-19 situation and seen their stock prices appreciate. We argue that sentiment analysis is not yet advanced enough to accurately determine how sentiment towards a stock may impact its valuation. All that said, we believe including some sentiment analysis can improve prediction accuracy in general.

Although our LSTM model has shown very positive results in predicting stock price movements, we do not believe it is at the stage where a trader should be confident enough to make trades as per the model predictions. The stock market isn't always rational, so even if all fundamental and non-intrinsic signs point to the stock market moving in one direction, it may theoretically move in the other direction. Therefore, it is quite risky to trust that such a model will always give you the correct prediction. It can however be useful in assisting traders navigate through the uncertainty of stock price movements by providing insights on trends in big data which are not available without the use of such machine learning algorithms. Therefore, we believe that our results indicate that there is a future for the field of data science in the financial world including the stock market.

CONCLUSIONS

In this paper, we showed how we can use a deep learning recurrent neural network model to be able to accurately predict stock price movements. Although we tried predicting Apple stock prices in this study, these models can be easily modified to predict

any stock that is traded on a public stock exchange by making some small adjustments to the correlated assets to fit the industry of the stock you wish to predict. Overall, our LSTM model for future predictions beyond the historical data did a particularly good job at being able to predict the trends in stock price movement, but fell short on capturing the full magnitude of those movements. The main reasons were that we were not picking up all of the features that may impact the stock price, and in addition, we did not have any sort of parameter such as sentiment analysis that would capture the non-intrinsic impacts on stock price. Overall, this resulted in a forecast error of 3.1%, which was significantly more accurate than the forecast error of our benchmark ARIMA model which came in at 6.8%.

Future iterations of such work would benefit from including some sentiment analysis that could capture some of the non-intrinsic factors that impact stock price movement, which we believe can help mitigate some of this issue. These include:

- Adding in Google search data as we believe there's a relationship between search trends, search volume, and stock price movement.
- Using sentiment analysis through a Bidirectional Encoder Representations from Transformers (BERT) model to capture news, social media and other non-fundamental influencers that play a large role in stock price movements.

REFERENCES

[1] C. Hargreaves and Y. Hao. "Does the Use of Technical and Fundamental Analysis Improve Stock Choice? : A Data Mining Approach applied to the Australian Stock Market". In Statistics in Science, Business and Engineering (ICSSBE), 2012 International Conference on, 2012.

[2] C. Liu and H. Malik. "A New Investment Strategy Based on Data Mining and Neural Networks". Neural Networks (IJCNN), 2014 International Joint Conference on. IEEE, 2014.

[3] A. Nayak. Predicting Stock Price with LSTM. (March 2019). Retrieved from https://towardsdatascience.com/predicting-stock-price-with-lstm-13af86a74944

[4] J.R. Quinlan. "C4.5: Programs for Machine Learning". Morgan Kaufmann Publishers, 1993.

[5] T. Kim, H.Y. Kim.(2019) "Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data", PLOS ONE Journals