Table of Contents

Stock Prediction ARIMA vs LSTM

Introduction

One of the ways to ensure higher returns than saving in the back is by investing in the stock market. Large financial institutions as well as individuals would like to maximize returns on their money, and one of the substantial ways this can be achieved is through the investing in the stock market. There are various indicators and tools that are available to help financial investors and traders make calculated decisions on what stocks to buy but these indicators but even with these it has been a very hard task to predict the future prices of stocks. As technology has advanced the financial industry has been interested in using machine learning in price prediction, pricing and managing entire portfolio of assists and investment process. With increase in computational efficiency and deep learning the applications are very versatile and we would like to explore the use of these new tools in stock prediction. With recent combination of machine learning models and statistics algorithms have been made that can reveal intricate patterns that are non-linear.

In this project we will look into a statistical based prediction algorithm ARIMA in particular and compare it to the the technologically advanced neural network deep learning system LSTM RNN in particular and see how it performs and if there is true advancement between the traditional and advanced method of stock prediction.

Objective

The main objective of this project is to access and investigate with forecasting model produce the best prediction results, by analyzing the model with lower forecast errors using mean square error and mean average error. With modern technology deep learning is able to patterns and structure in data that are more complex this project seeks to investigate its effectiveness in stock prediction.

Data

Standard stock price data has five columns the high, low, open, close, adjusted close and date. These dates do not include the weekend and holidays. The data ranges form 2012-01-05 to 2019-12-31. About a total of 8 years, making a total of 2010 rows with 6 columns. The high signifies the maximum price for the day while the low indicates the minimum price for the day, the open is the price the stock was at the beginning of the day, the close is the price at the end of the day, the adjusted is the price at the end of the day but factors in anything that might affect the stock price after the market closes.

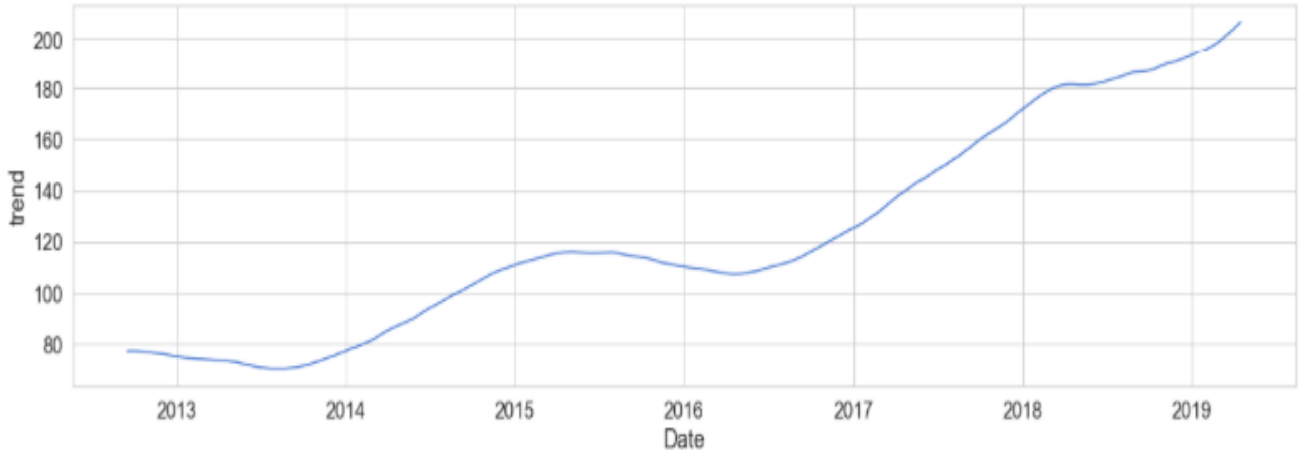
Figure2: close price 2012 - 2019

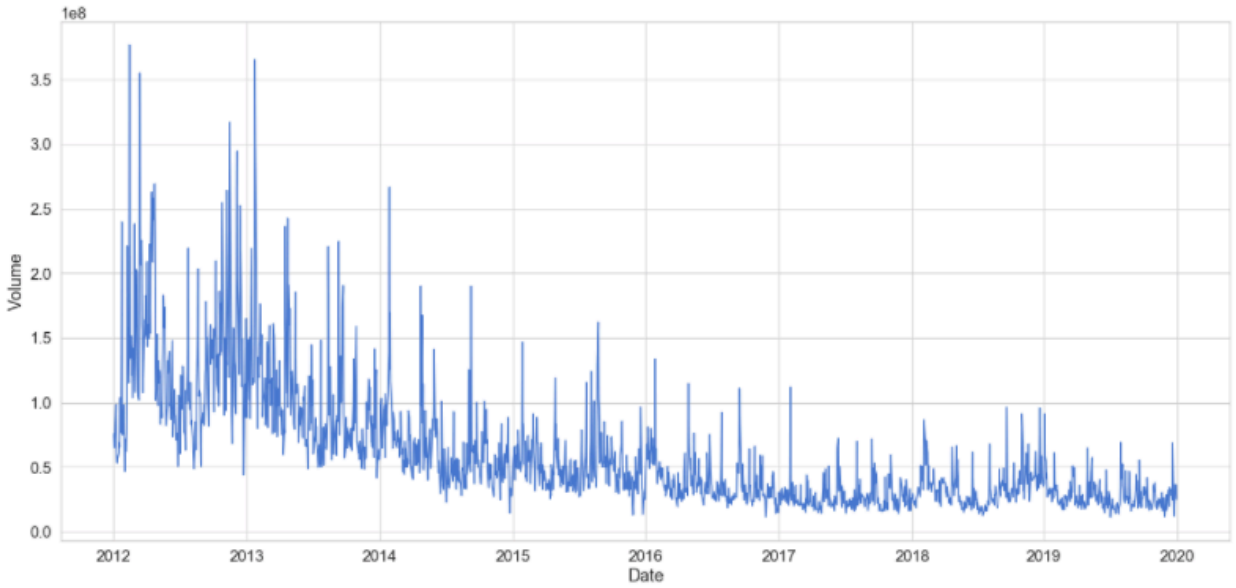
Data Exploration

Data exploration with regards to stock charts is not broad as the main technique is to use past data to predict the present. The only feature used in this study would be the daily closing costs of the stock.

1. Decomposition of Time Series

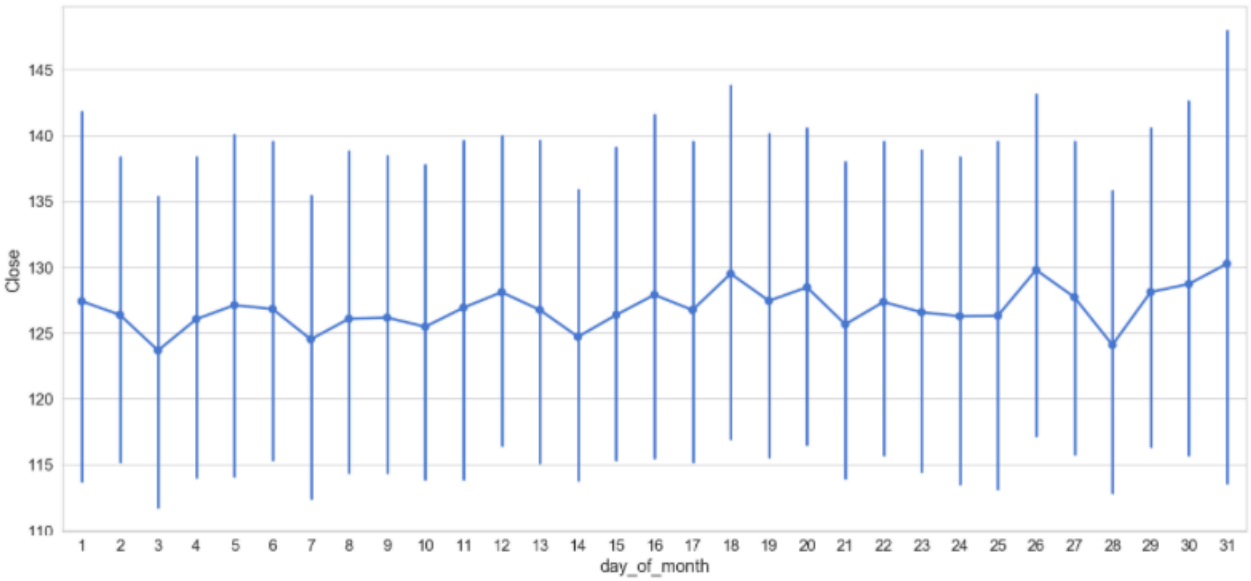
A series needs to be decomposed in order to ensure that there are some patterns and that the series is not majorly white noise which cannot be used for or casting.





a. Seasonal Chart from this chart we can see a cyclic pattern that occurs but over the a period of a year.
b. Trend: shows the overall direction of the apple stock price over the course of years
c. Residuals this is the unexplained noise that is found in the data, over those number of years after you remove the effect of the trend and the

3. Check for Patterns over smaller time range
    a. In order to look more closely at data on closer basis we check if we can find any pattern in the time series over the span of a month. From this chart there is no constant pattern that occurs monthly.
    b. This chart tells us high high the buying and selling interests are , here we see volume of shares traded over the years has reduced considerably.



Stock Prediction Models

1. ARIMA

ARIMA stands for 'AutoRegressive Integrated Moving Average', is a forecasting algorithm based on the idea that the information in the past values of the time series can alone be used to predict the future values. Hence the RIMA model captures temporal structures in time series data. ARIMA models must be non- seasonal and exhibits a patterns and is not white noise. An ARIMA model uses it own lag as predictors that is why it is called Auto regressive means it is a linear regression they will work best when they are not correlated and are independent of each other so we much make the series stationary to achieve this.

An Auto regressive model (AR) depends on its own lag $Y_{t.}$. That is, Yt is a function of the 'lags of Yt'. the model is as follows

Predicted Yt = Constant + Linear combination Lags of Y (up to p lags) + Linear Combi-

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_p Y_{t-p} + \epsilon_1$$

nation of Lagged forecast errors (up to q lags).

Hence the p, d and q variables need to be identified. p is the order of the AR term, q is the order of the MA term and d is the number of differencing required to make the time series stationary.

where, $Y*{t-1}$ is the lag1 of the series, $\beta*1$ is the coefficient of lag1 that the model estimates and α is the intercept term, also estimated by the model.

Steps in carrying out in ARIMA forecast :

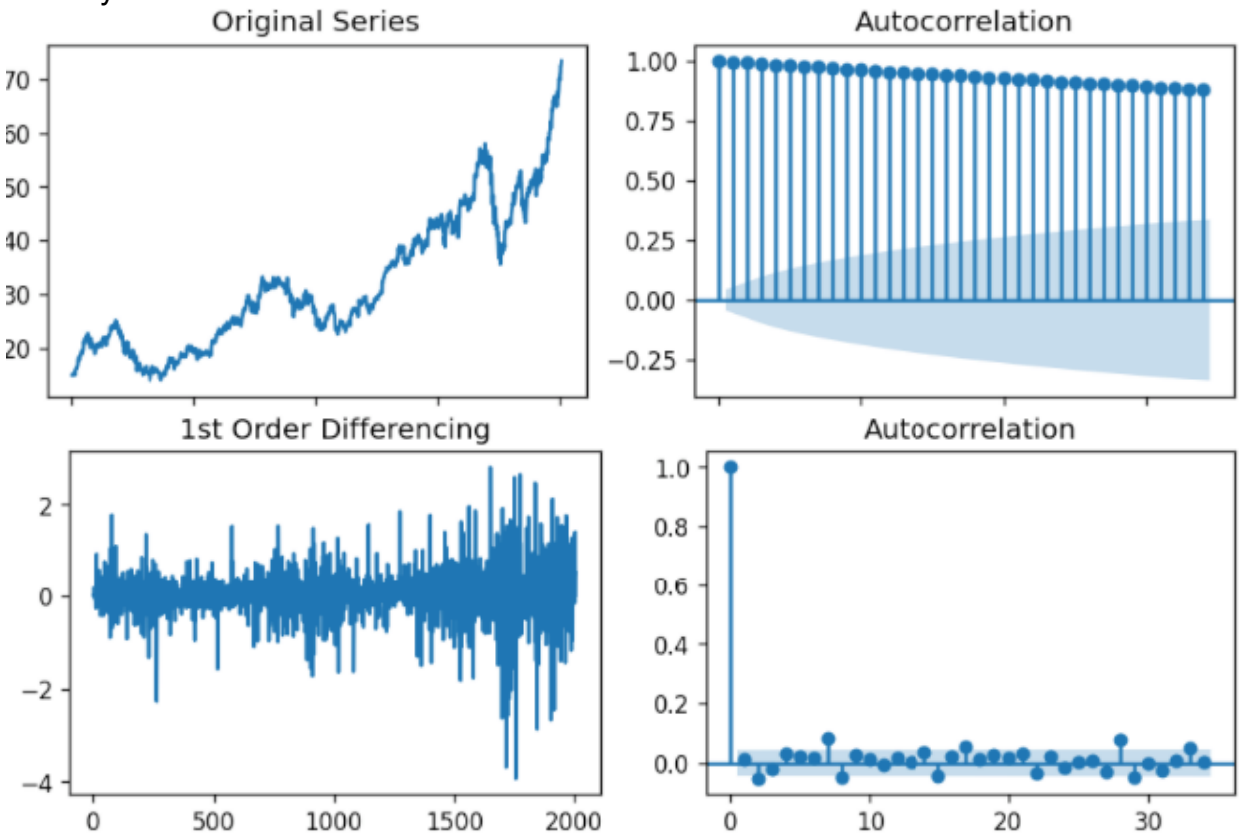
1. Import the dataset.
2. Subset data set to target variable only

3. Build a train and test set.
4. Check stationarity in data and act accordingly
5. Find the order of differencing (d)
6. Find the order of AR terms (p)
7. Find the order of the MA term (q)
8. Build models.
9. Validate the model with the test set.

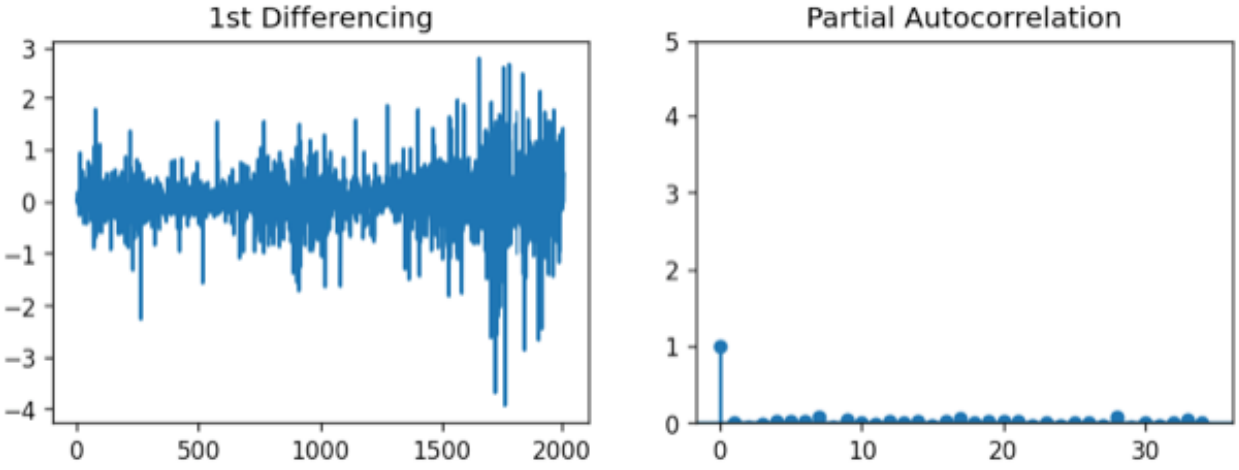
Methodology

In this project the main idea is to find the adequate p,q,d terms using the ACF, PACF plots. This should be done after the series is found to be or made stationary. We use the augmented Dickey–Fuller test (ADF) test to check if the time series is stationary. The summary of the process is as follows

The series doesn't look stationary as the autocorrelation plot is showing an increasing trend. Now we will use the augmented dickey fuller (ADF) test to check if the series are stationary. The results of the ADF test is a p value of 0.99 which means we cannot cannot reject the null hypothesis of a unit root which suggests the series is not stationary. For our series it reaches a stationarity with one order of differencing since the the autocorrelation plot os close to 0 and is ranging above and below the 0 level very randomly.



Next we find the if the model needs AR terms we can determine this by looking at the partial autocorrelation plot (PACF). The PACF is a correlation between the series and its lag. Looking at the PACF of we can see that only the first value is above the blue region which means that the the p of 1 is appropriate.



Order of AR term (p)

To Find the the MA term we must look at the ACF plot the MA term is technically the error of the lagged term. From the diagram below we see that the first lag is above the blue area hence it means we use q as 1. Since we have the terms p, d, q we can create the ARIMA model. From the summary table we see that the AR term and MA term are both significant since they are both 0.00 so our model does not need to be changed.

ARIMA Model Results

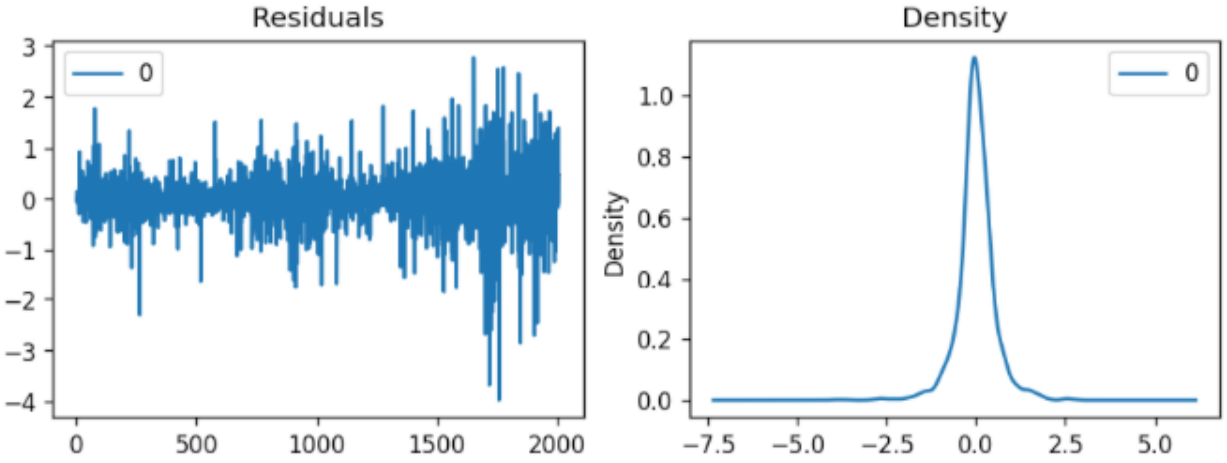
Dep. Variable:	D.value	No. Observations:	2009
Model:	ARIMA(1, 1, 1)	Log Likelihood	-1581.391
Method:	css-mle	S.D. of innovations	0.532
Date:	Tue, 05 Jan 2021	AIC	3170.782
Time:	21:58:48	BIC	3193.203
Sample:	1	HQIC	3179.013

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.0291	0.012	2.428	0.015	0.006	0.053
ar.L1.D.value	-0.8352	0.120	-6.946	0.000	-1.071	-0.600
ma.L1.D.value	0.8547	0.113	7.552	0.000	0.633	1.077

Roots

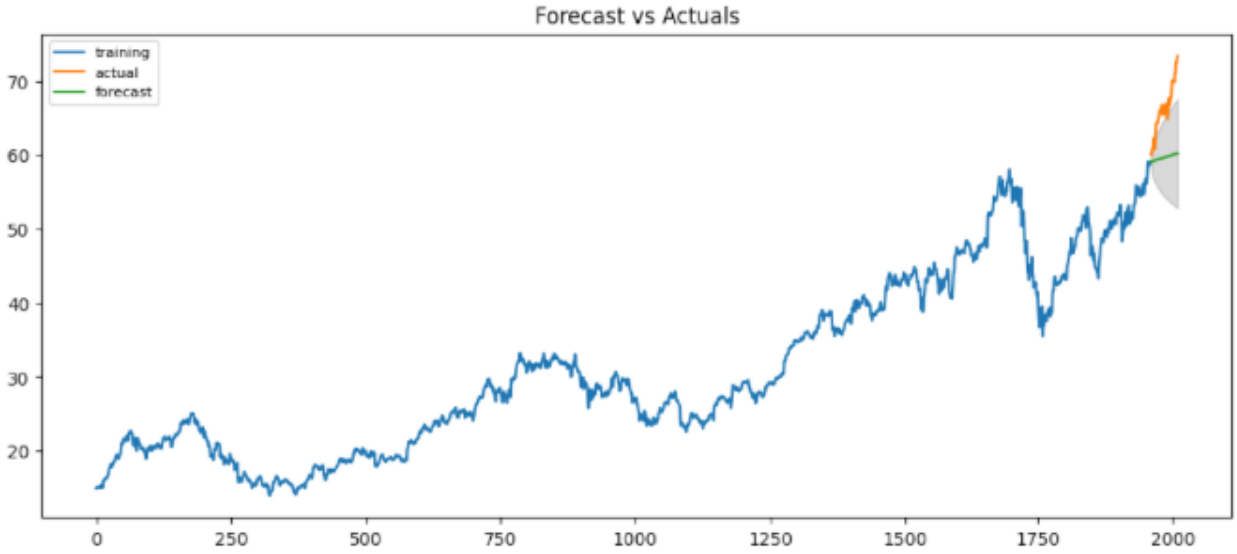
	Real	Imaginary	Modulus	Frequency
AR.1	-1.1973	+0.0000j	1.1973	0.5000
MA.1	-1.1700	+0.0000j	1.1700	0.5000

Also we can check the Residuals and the density plot here we confirm that one order of differencing is enough as the residual errors seem fine with near zero mean and uniform variance, and the density plot shows and uniform distribution. The next steps are to train the model and use it to forecast into the future.



These values though on some occasions coincide with the test values most times the predictions are off, this algorithm may need to be tweaked to further increase accuracy.

The test and training sets were created with the train to be 1960 days before and the train set to predict 50 days out. The forecast command in the statsmodel tools will be used to predict values into the future with an out of sample prediction that is not being influenced by in-sample values.



ARIMA forecast

2. Long Short-Term Memory Network (LSTM)

The Long Short-Term Memory network, or LSTM network, is a recurrent neural network (RNN) that is trained using sequential observations learned from the earlier stages to forecast future trends. Although the LSTM is a type of RNN it is special because it can store past data and memorize it. LSTM can capture long term influences, with a component it has called memory block which replaces the traditional artificial neurons in the hidden layer by the memory cell. The LSTM works by managing the blocks state using gate structure.

There are 3 gates the forget gate that conditionally decides what information to keep or discard it has output of 0 or 1 where 0 means “completely ignore”, the input gate which conditionally decides which input to update the state of the memory block and the output gate which decides conditionally what to output based on the input and memory of the block.

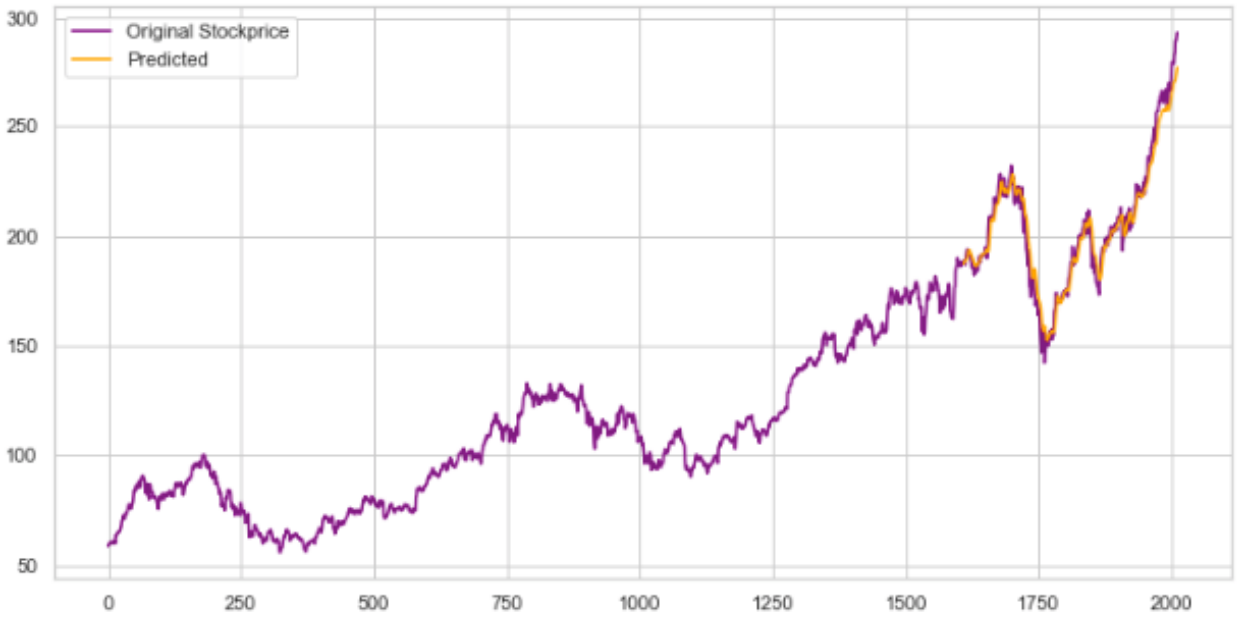
Steps in carrying out in LSTM forecast :

1. Import the dataset.
2. Subset data set to target variable only
3. Convert to numpy array and reshape
4. Build a train and test set.
5. Feature scale the data.
6. Create data step to predict next day price.
7. Build model.
8. Validate the model with the test set.

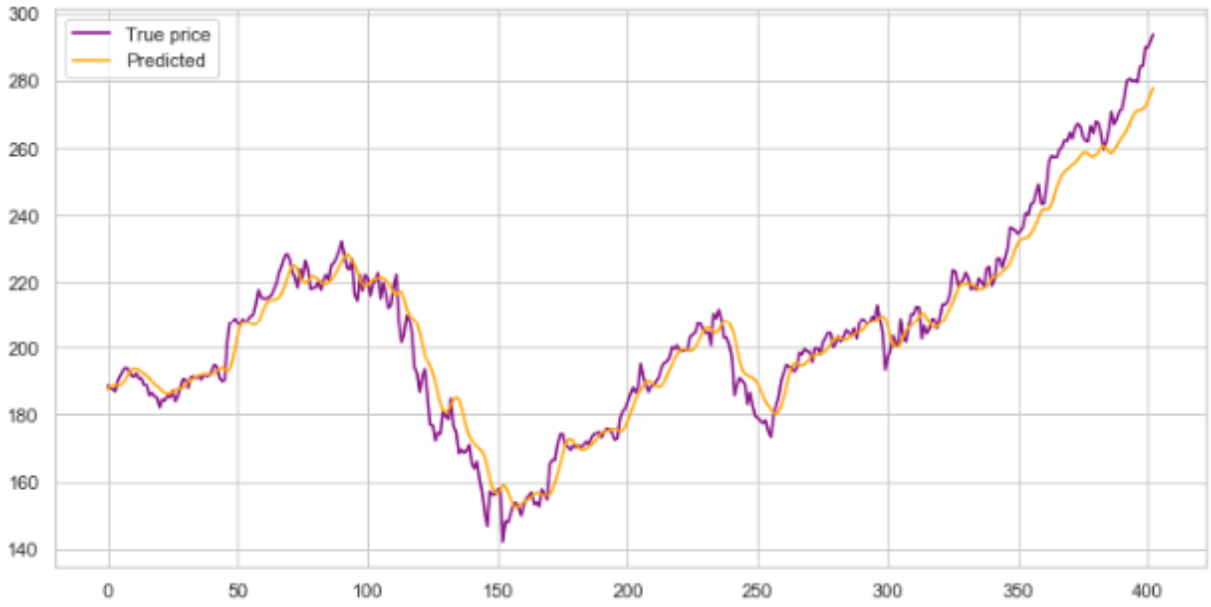
There are some important terms in LSTM which are batch size and time steps The batch size refers to the number of samples the neural network analyses before it updated the weights, very small batch size makes the model run slower however larger back sizes take up memory. Time steps are how many units in back time you want the network to analyze before it predicts for the next stock. In this project they were 50 time steps to predict 1 step. And the The test train dataset is divided with 20% test and 80% training. LSTMs usually accepts a 3d array array format so after the time steps are created the test and train set are reshaped into a 3d array.

Methodology

The training and testing set was divided into 80% 20%. Then the data was feature scaled and normalized. The LSTM model is created with keras and consist of 4 types of layers; a sequential layer for initialization, the LSTM layer, the dropout layer to prevent overfitting and the dense layer for adding a densely connected neural network layer is normally at the end of the model. The LSTM layer comprises of 3 arguments



the units which is the dimensionality of the output space, the return sequences which determines whether to return the last output in the output sequence, or the full sequence so when the last layer is input the sequence should not be returned, and the



input shape which is the shape of the training data set. The drop out layers were specified at 20% hence 20% of the layers will be dropped. The dense layer is then specified. This model uses ADAM optimizer this was chosen because it combines the perks of two other optimizers: ADAgrad and RMSprop and sets the loss as the mean square error.

Assessment metric

1. MSE:

The mean square error is the a measure normally used for a prediction model it is measures the average of the square residuals ie difference between then actual and predicted values. Since the errors are the larger errors have more weight on the score.

$$\mathrm{MSE} = \frac{\sum_{i=0}^{n}(y_i - \hat{y}_i)^2}{n}$$

2. MAE:

the mean absolute error is the arithmetic average of the absolute errors or the average of the absolute values of the deviation. This metric can tells us the size of an error you can expect for the forecast.

$$\mathrm{MAE} = \frac{\sum_{i=0}^{n}|y_i - \hat{y}_i|}{n}$$

Final results

Metric	ARIMA	LSTM	% Reduction
MSE	53.24	2.9	89
MAE	6.61	1.3	67

Table 1

The results are reported in table 1there was considerable better performance with the LSTM than the ARIMA in all evaluation metrics % reduction for the RMSE and MAE respectively is 62% and 67%.

Conclusion:

For the LSTM model the loss function ranged from 0.0404 to 8.17e-04 these are fairly good values. However when looking at the plot for the actual versus the predicted the predicted prices were close, always slightly above or below the actual prices. The ex-

act predicted values would not be good for trading as they are mostly not perfectly accurate but indicated the overall trend and direction so could be somewhat effective for stock predictions. This LSTM model could be altered and improved upon to generate more accurate results and from this study it seems LSTM models could be useful in stock predictions.

For the ARIMA model From the chart the ARIMA model (1,1,1) seems to give the right direction however the forecast was not in the 95% confidence interval. So this model would be considered fairly inaccurate. After analyzing and checking the the ACF and PACF plots we see that the ARIMA model (1,1,1) was adequate and the AR and MA terms are both 0.00 and gave a pale of the below the standard alpha of 0.05 we can say that the chosen values are accurate.

For our final conclusion we see that the LSTM model is considerably better however is not accurate enough to produce prices that can be used to actually on stock as the predictions were not accurate enough to place market orders on since stocks deal directly with money the predicts need to near perfect for these values to be used. New technology is an improvement however more work needs to be before more trust can placed on them.

References

1. https://reader.elsevier.com/reader/sd/pii/S1877050920304865?token=375651CB02BB699B28C5CE44F3C39BDA96E60C8B269C0EAA6-DA52D8D6F978C8E0A3703F40EF06280AB98F80BFA48490C
2. https://blog.usejournal.com/stock-market-prediction-by-recurrent-neural-network-on-lstm-model-56de700bff68
3. https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/
4. https://joydeep31415.medium.com/common-metrics-for-time-series-analysis-f3-ca4b29fe42
5. https://par.nsf.gov/servlets/purl/10186768
6.