Method 2

1 Problem statement

In this method, we are using the Artificial Recurrent Neural Network called Long Short-Term Memory to predict the closing price of a corporation called Apple Inc. using the past 60 days stock price. Long Short-Term Memory that's uses the time-series model of the Machine Learning to predict the long values. We are using the Keras model of the python library. We are importing the data from Yahoo! Finance. It is the Financial market web site that provides the financial news, data and commentary including stock quotes, press releases financial reports and original contents. We have used libraries like math function to do the operations for the Keras layers, Pandas library is used the get the financial data from the yahoo finance. To show the actual data with the time-series model as we are applying the long short time memory to plot the data points on the graph. Python programming language has the high-performance library called pandas which are open source and is easy to use data structures and data analysis tools.

2 Data processing

We have imported the dataset using web data reader in which we have Apple Inc. stock quotes. While importing the database we can give the other companies data using their market initial that are used in the finance market. To read the data from the online finance market there are many data sources available Google Finance, Tiingo, Stooq, Nasdaq and MSN money. To get the data we used the starting of the data set by giving the start date and end using the last day of the historical quotes and show the complete data set. In the given data set we get the High, Low, Open, Close, Volume, Adj. Close and date values of the stocks of that company. In the bottom of the dataset, we can see how many rows and columns are used in the dataset. By calling the shape function we can also get the number of the rows and columns in the data set. We can see information about the data set we get the in the class pandas core frame data Frame with the how many entries are there start and end date in the Date time Index. All the data in the columns are in the form of the float format data type, and non-null Count shows that how many empty values are in the given column and total usage of memory used.

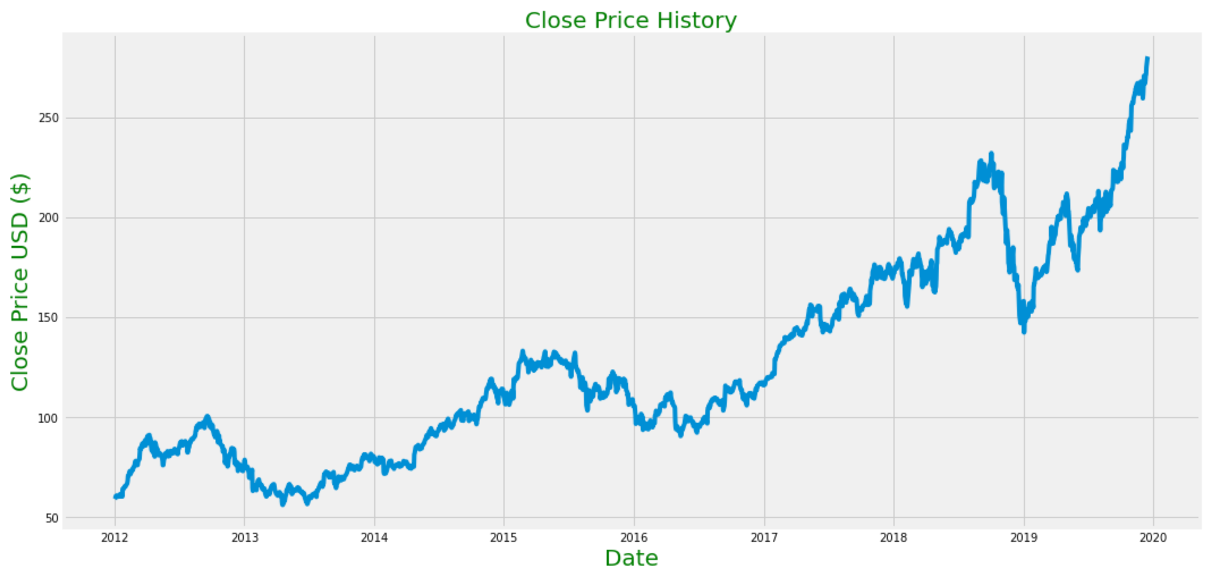

Fig – Graph showing the closing stock price history of Apple Inc.

In the above figure of Closing Price History, we can see the Apple Corporation stock prices, in the graph, it is clearly showing that these prices have the uptrend in which it has continuously grown. From Series & Panel, we can get the Data Frame which is a data structure provided by pandas' library, we get the table of the rows and columns with the 2-dimensional structure. The figure contains Dates as the times with the series with each year on the X-axis and prises with the United States Dollars on the Y-axis. As the Stock buyer, if we want to buy stocks and sells them at the right point, we can analyse this by closing data, for that we are using time-series modelling. The Machine Learning models called long short-term memory we have used to predict the last

60 days of the closing price with the validation, the actual price of the data and predicted values of the data in this model. This model will not just a machine learning, but we also must think about the stochastic or random process model.

3 Model building

Now we will create the new data frame with only “Close” column using the filter function. Then after filtering the dataset with close values, we will convert these values into a NumPy array and store it in the new variable. We have the array of the data that’s is the Closing price, now we will train the data using the math. Ceil function and it will show the number of rows is to be trained within the data set by diving the data into the training data which we have calculated it as the 80% of data would be the training data. It defined the C standards which are provided by the mathematical function. These math functions are not used for the complex numbers, so we have just used the float numbers for the close values. When we use the NumPy array and convert the data into the filtered data that is not the complex data math. Ceil (X) in which x must be non-float values and this function will give the smallest integer greater than or equal to x when we return ceiling of x.

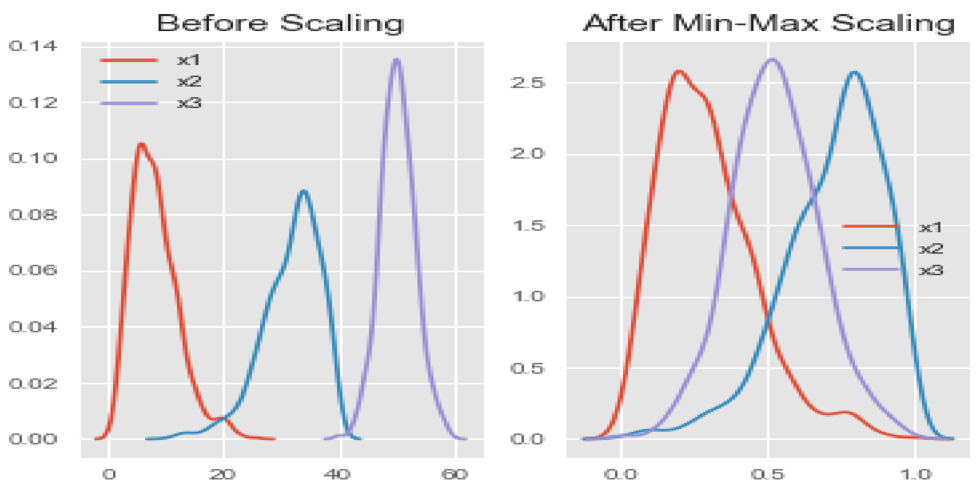

Figure A – MinMaxScaler

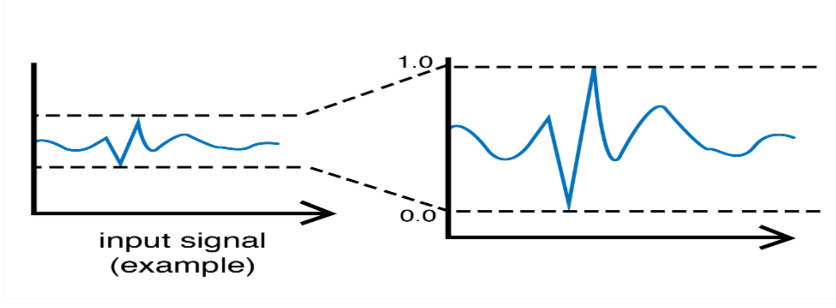

Figure B - Plaquette-MinMaxScaler

We have trained the data and know the length of the training data which how many rows is to be trained. In figure A we have shown the data frames which have data samples X1, X2, X3 with positive skew, negative skew and no skew respectively that is shown before scaling. When we Scale the data and do some pre-processing on the data using MinMaxScaler and fit scalar data in the transform and that will shrink the data in the range on 0 and 1 and that is shown in the after Min-Max Scaling. In this trained data now, we will scale the data to normalization, transformation and processing on the input data that will help the model. To scale the by using the scaler function we will use the MinMaxScaler by giving the scale it in the range of 0 and 1. The MinMaxScaler has the following formula:

$$X = x_i - \min(x) / \max(x) - \min(x)$$

In the MinMaxScaler object, it essentially shrinks the range of the distributed data such that the range is 0 and 1 in other ways if it has negative values it will have the range of -1 to 1. In the given formula X is the rescaled values of the data set, x_i is the original value which can be trained, $\min(x)$ is the minimum value in the feature and $\max(x)$ is the maximum value in the feature which rescales feature values to between range 0 and 1. The distribution is done works well in the Gaussian or when the standard deviation is very small otherwise MinMaxScaler works with a normal distribution. In Figure B we can see that remapping the filtering unit with regularizing incoming signals into a new interval of [0,1]. While scaling data in the range the minimum values of the signals is mapped to 0 and the maximum values are mapped to 1, it keeps track of the minimum and the maximum values can take the signal and rescale it with the range.

4 Model compiling

An artificial recurrent neural network (RNN) architecture which is used for deep learning it has Long short-term memory (LSTM) that gives feedback connections which are standard feed-forward neural networks. The sequence of data points (such as speech or video) is also processed by LSTM as single data points (such as images). Long Short-Term Memory works in the sequential prediction problems and has the good ability to be extremely effective because it stores past information that is important and forgets information which is not important. Input Gate, Forget Gate, Input Modulation Gate and Output Gate are commonly used architecture of long short-term memory.

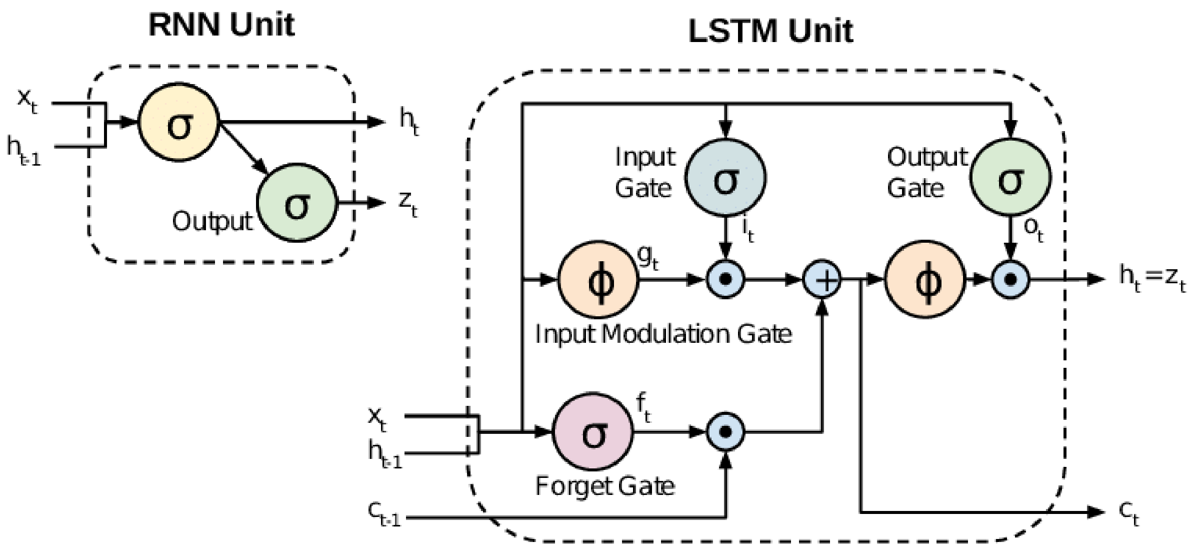

Fig - The different Gates used in RNN and LSTM.

Now we will create the training data set for that we will create variable to store the scaled training data set. We have provided the data from 0 to the training data which we have splinted and shows values in the arrow. Now we will split the data into the X and Y training data, where X is the independent training variable or feature variable, Y is the dependent variable or target variable. We gave the range of 60 because we are, we are predicting the 60 days of the data. When we train append of the 60 days when of data using the append train, if we change the values of the 60 it gives the same number of values when we give one number bigger it will get the second pass with the value. In this, we will get x values as several days that is 60 days and y value would be the next value that module to predict.

The data set from 0 to index 59 that are total 60 values and the second column has the values from the data set from index 1 to index 60 that's is 60 values and will give again 60 values which are stored in the first column that is 'x-train' data set. The 61st value is stored at index 60 in which 62nd value will be stored in the index 62 of the data for the next two values in the 'y-train' data set. LSTM model gets the converted independent training data set 'x-train' and dependent training data set 'y-train' in NumPy arrays respectively. We will reshape the data because we have the 2-Dimensional data and for the LSTM model we required 3-Dimensional form of data that is number of **samples**, number of **time steps** and number of **features**.

5 Model fitting

ADAM and Mean Squared Error (MSE)

After getting trained data and reshaped the given data we will train the LSTM model with Sequential values, for that are providing two LSTM layers with 50 neurons that are several samples and two Dense layers of 25 neurons that are several time steps and one neuron as several features. Now we will compile the model by using mean squared error (MSE) as the loss function of the model and Adam as the optimizer. We will train the model using training data sets, it is also known as fit. In the model to compile the LSTM model, we are giving Adam optimizer while compiling and we are giving the loss function as the mean squared error (MSE). The optimizer is used to improve the point loss function, and loss function is how the model on which we did the training.

To train deep neural network Adam has adaptive learning to optimize algorithms which are specifically designed. Adam computes individual learning rates on different parameters because it has an adaptive learning rate method.

Properties of Adam

1. Each iteration is approximately connected to the step size hyperparameter with actual step size taken by Adam. The previous unintuitive rate of hyper-parameter which could have the understanding to get this property with intuitive.
2. While going through areas with tiny gradients such as saddle points or ravines can help to get invariant to the magnitude of the gradient of step size of the Adam update rule. It gives mostly negative values of them when they make trouble for SGD with momentum to reach.
3. Adam can involve with the advantage of Adagrad, that performs with gradient and RMSprop can give best results on-line setting. While working with both Sparse gradient and RMSprop will give priority to use its broader range of tasks. The combination of RMSprop and SGD with momentum has all the properties of Adam.

Mean Squared Error (MSE)

Mean Squared Error (MSE) is also known as the mean squared deviation (MSD), it is estimated unobserved quantity, which is produced while compiling the program, it measures the average of squares of the errors. The estimated value and the actual values of the error which has difference are called as average squared. The loss function for the regression has mostly used feature of Mean squared error (MSE).

$$MSE = L(y, \hat{y}) = \frac{1}{N}\sum_{i=0}^{N}(y - \hat{y}_i)^2$$

The squared difference between true and predicted value has the mean overseen of the loss function, in this above equation ŷ value is predicted value as from a least-squares fit. A sample of N data points on all variables that are generated by a vector of n predictions, the variable is being predicted by observing the vector y, the MSE is the mean $\left(\frac{1}{n}\sum_{i=0}^{n} \quad\right)$ of the squares of the errors $(y - \hat{y}_i)^2$. The model and variation explained by randomness have detailed and gives one to partition variation of the dataset into variation while analysing the performance of linear regression have confident evident of the mathematical benefits of mean squared. The mean squared error is used because the mean target value gives optimal prediction when MSE is sensitive towards outliners and it has the same input feature values. When we use normalised distributed data around a mean value which have conditioned on the input values gives the good target value by using mean squared error (MSE). We can use mean squared values when we want large errors to be significantly (quadratically) more penalized than small ones which use MSE in regression, while we guess the target value giving them condition on input and will be normally distributed. In this example we are predicting the 60days value of the stocks the Closing price is continuously increasing, and for that, we need regression so that MSE is used as the loss function.

6 Train Dataset and Model prediction

Now we will train our model using model fit which is another name for the which is in our input data set. In this batch size is the total number of the training examples present in a single batch. The number of epochs is several iterations when the entire data set is passed forward and backwards to a neural network. In some programs we see that we have three different codes with the same kind of terms like in our research we are applying three different algorithms which have same functions of machine learning, to find out how can we use three different terms we will know about the Gradient Descent.

Gradient Descent

Gradient Descent is used to get the best results with a minimum of the curve for that we use the **Iterative** optimization algorithms which are used in machine learning. The rate of inclination or declination of slop is known as gradient, the instance of descending is known as Descent. We have to run the iterative algorithms on result values multiple times so that we the most optimal results, that we have shown in the following figures with passing the different cost values with parameters and it gives the curves for the line that we have shown in the module output with the labelled data. More the under-fitted graph which is given by the iterative quality of the gradient descent will make the graph fit optimally to the data.

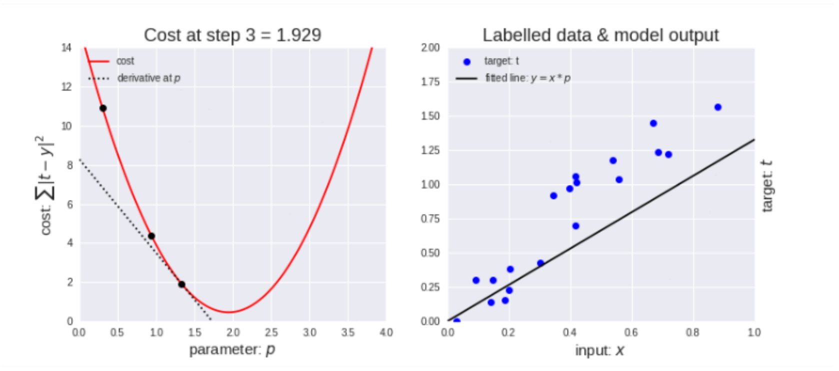

Fig - Iterative Optimization on labelled data at cost 3.

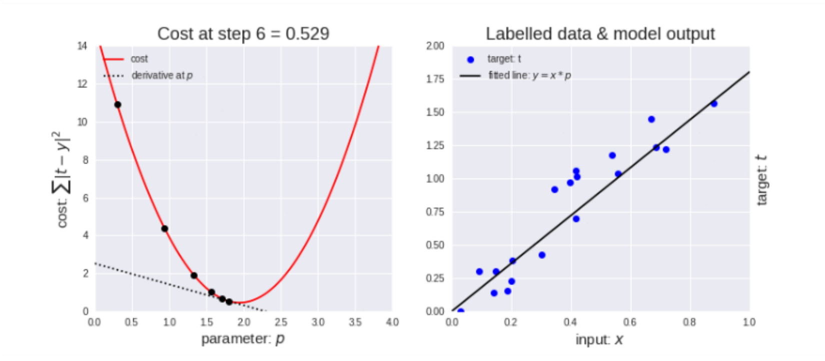

Fig - Iterative Optimization on labelled data at cost 6.

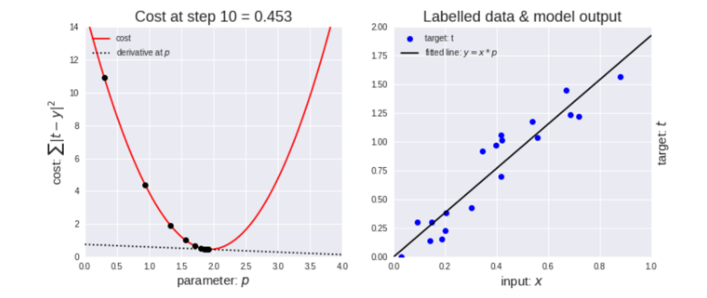

Fig - Iterative Optimization on labelled data at cost 10.

The Learning rate parameter is also known as Gradient descent. In the above figure, we can see that when the point goes down then the learning rate will also become small with a shorter size of the steps, and vice-versa in initial cost at steps are bigger which means the learning will go higher size. When the cost is decreasing then the **Cost** function will also be decreasing. In the financial market we can see that **Loss** function is decreasing it means the loss is decreasing. In short, the **Cost** and **Loss** represent the same things that are good things that is our loss/cost is also decreasing. To overcome these problems, we used the terminologies like epochs, batch size and iterations, that are used to give data which is so big in the stepwise to our computer at once. While giving the data to the computer we need to divide the data into smaller size one by one, with that at the end of every step to fit it to the given data we update some of the weights of neural networks.

Epochs

One Epoch is when an ENTIRE dataset is passed forward and backwards through the neural network only ONCE. We must divide the dataset into several smaller batches because providing one epoch is the too big dataset to give to the computer. In our example, we are passing the 60days value from the data set to train the value which is not the big data for the normal computer to perform for that we must pass the full dataset multiple times to the same neural network. We are using the **Gradient Descent** which is an **iterative** process to perform on the limited dataset so that we can optimise the learning with the graph, hence updating the more weights with the single-pass or one epoch is not that much effective.

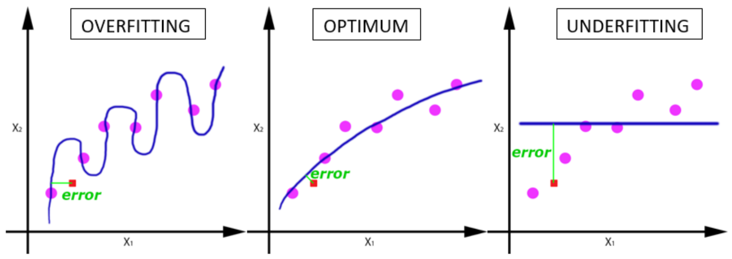

Fig - One epoch leads to underfitting of the curve in the graph.

In the figure shown above, we can see that when we increase the number of epochs then the neural network has been changed the greater number of there times weight, to curve in the graph from underfitting to optimal to overfitting with the reverse sequence. With our example, we can see that for our data set we have used only 60days value to train and fit the model, as we can say that for different data set we will pass the different number of epochs to get the right curve for the diverse dataset.

Batch Size

A total number of training examples present in a single batch, it is important to differentiate with terms batch size and number of batches. We have used the batch when we **divide the dataset into Number of Batches or sets or parts** because we can't give the entire dataset to our neural network on our computer. In general, to understand the batch concept we can see that if we have all the closing price in our dataset then it will no use to get the best result, so we have divided it into training and testing dataset.

Iterations

Iterations are the number of batches needed to complete one epoch; we should know multiplications tables. The number of iterations is the same as the number of batches for one epoch. To understand this, we consider that we have 5000 training examples, that means we can divide training examples into batches of 1000 then we have taken the 5 iterations to complete 1 epoch, where Batch size is 1000 and iterations is 5, for 1 complete epoch.

Now we will create a testing dataset, for that we will create a new array which contains the next values remaining values of our dataset which are not used in the training dataset. When we have the test data that would scale the data which is contained data from the index that is represented in our example we will get by from full dataset minus length of training data subtracting the number of days which is 60 in our example. We are creating two datasets that are for x and y tests respectively, in the x test which contains the past 60 values, y test dataset will have the all the values which our module wants to predict for that rest of the data, that's are from all the columns from the Closing price minus length of the training data. We have provided the append to the data set from the position i-60, that means it will add cases/observations to a dataset. When two or more variables have the same name then we use append function, for our dataset we are using the same categorical data.

In the next step, we will convert our test data into the final test data, for that we will convert the independent test data set into NumPy array so that we can use this test data for testing the Long Short-Term Memory model. As we have the 2-dimensional array in the test data so we will convert new test data by using **Reshape**, so that we will get the data in the form of 3-dimensional. Now we will get the predicted values for x test values from the model using the new test data. We create the variable called **Prediction** in which we will get the same values in which we have inversely transformed y test dataset. When we have the predicted values in the new variable called **Predictions** will fit that in the model then we will unscale the values as it has the same values as y test dataset have, we will get new **Prediction** values which are based on the x test dataset.

7 Result visualization

Now we will get the root mean squared error (RMSE) which as a good measure of how accurate the model predicts; it is the standard deviation of residual and lower values of RMSE will indicate a better fit. Then we get the values closer to the 0 that indicates the models predicted the closer or actual values from the dataset which we have used, as we found that the lower the values the better the model performed, we also know the flow of the data performed well if uses different matrices.

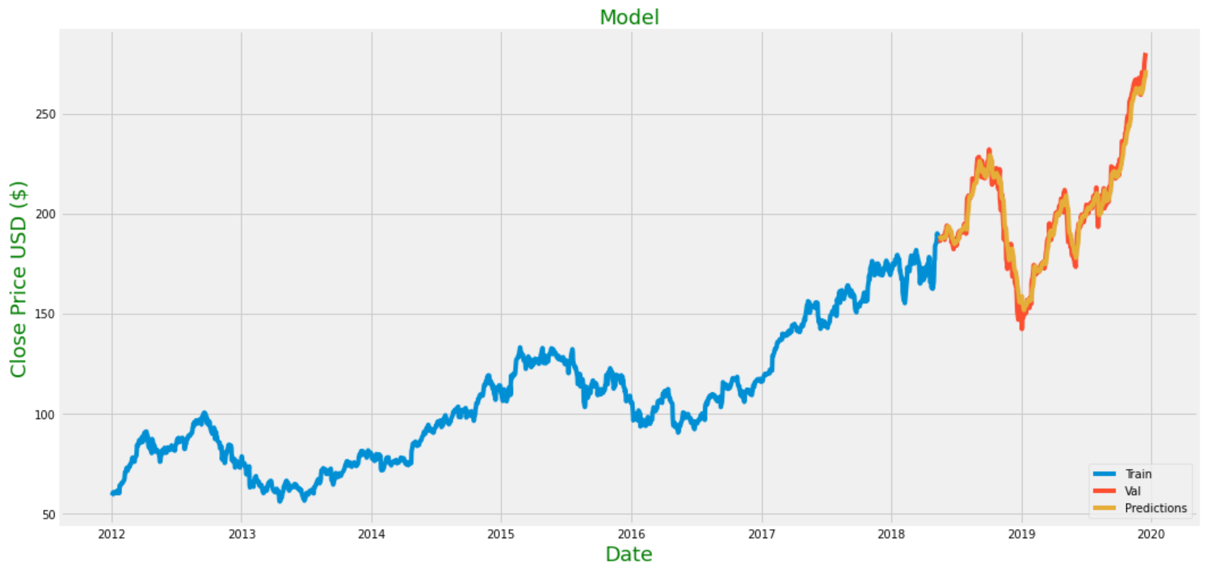

Fig - Graph showing the training (train), actual (valid) and predicted (predictions) prices.

In the above graph, we can see Training values that are in the blue, red are the actual values which are validation values and the orange it shows the predicted values. Through the valid parameter, we can see the three columns that are with the last 60 days value on these last stocks of the Apple corporation that are Dates, Closing Prices and the Predictions which are printed. We have used the Long Short Temporary Memory we have predicted the values with the values for actual (close) and predicted (predictions) price.

To predict the closing stock price for the one date using the Long Short-Term memory.

In this procedure, we will test our model with multiple ways and get the predicted value of Apple Inc. for the one day. So that we will get the data from the Yahoo Finance for Apple Inc. and will fit the data into the data frame. We will filter the Closing price and get the closing price and we will store it in the new data frame, when we get the 60 days closing price values and that values which are stored in the data frame we will put these values in the array. We will scale the data into the values between 0 and 1 for that we are using the last 60 days closing value. While scaling the data the variable we are calling is a scaler, we are not using the fit transform because we are using MinMax values when we first transformed the data in the previous procedure.

When we will get the NumPy array and reshape it and put the input data in the model for that we will create the empty list and append the last 60-day values.we will reshape the x test data in the changing same in 3-dimensional data as our data is in 2-dimensional values, and our LSTM model is expecting the 3-dimensional data. We will create the new variable called predict and give that x test data, and undo scaling and we will inverse transform the predicted price. We will get the predicted price when we input the data into this LSTM model. Now we will print the actual values of the closing price for that day.

https://readthedocs.org/projects/pandas-datareader/downloads/pdf/latest/

https://riptutorial.com/Download/pandas.pdf

https://docs.python.org/3.2/library/math.html

https://benalexkeen.com/feature-scaling-with-scikit-learn/

https://jovianlin.io/feature-scaling/

https://sofapirate.github.io/Plaquette/MinMaxScaler.html

https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c

https://en.wikipedia.org/wiki/Mean_squared_error

https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/mean-squared-error

https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9