**Introduction**

As per the problem statement to predict the return of the stock on 30th trading day, we have used a LSTM (Long Short-Term Memory) network for a time series analysis to iteratively predict the closing price of the stocks on 30th trading day i.e. 16th May.

Recurrent neural networks (RNN) have proved one of the most powerful models for processing sequential data.

Long Short-Term memory is one of the most successful RNNs architectures. LSTM introduces the memory cell, a unit of computation that replaces traditional artificial neurons in the hidden layer of the network. With these memory cells, networks are able to effectively associate memories and input remote in time, hence suit to grasp the structure of data dynamically over time with high prediction capacity.

In this document, we have used the historical data (previous 5 years) of YESBANK.NS and used it for training and validation process.

**Approach**

We have used LSTM (long short-term memory) recurrent neural networks used in order to perform financial time series forecasting on return data on the given stock data.

Regression with LSTM network models. Predicting 1-day ahead

market closing price and appending it to the dataset till we get closing price of 30th trading day.

**Technologies Used:**

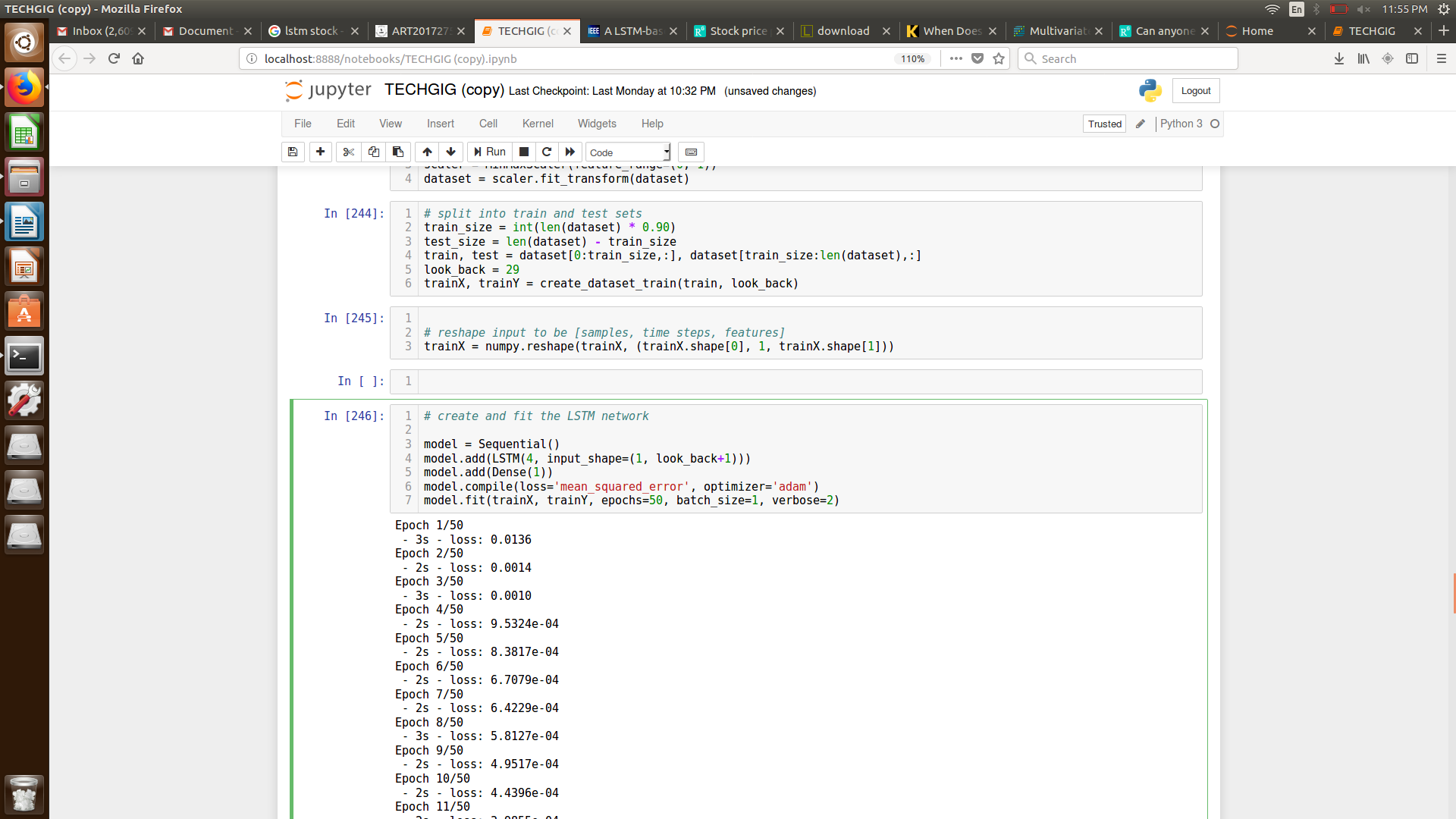
* Python
* Keras(Used for creating LSTM)
* Tensorflow(backend for keras)
* Numpy
* Pandas(Data Analysis)
* Sci-kit Learn(Min-Max scaler,Mean-squared error)
* Matplotlib(Plotting graphs)

**Why LSTM?**

LSTM can in principle model nonlinearities automatically for Time series analysis which we would need to explicitly model using transformations in linear regression.

**LSTM STRUCTURE**

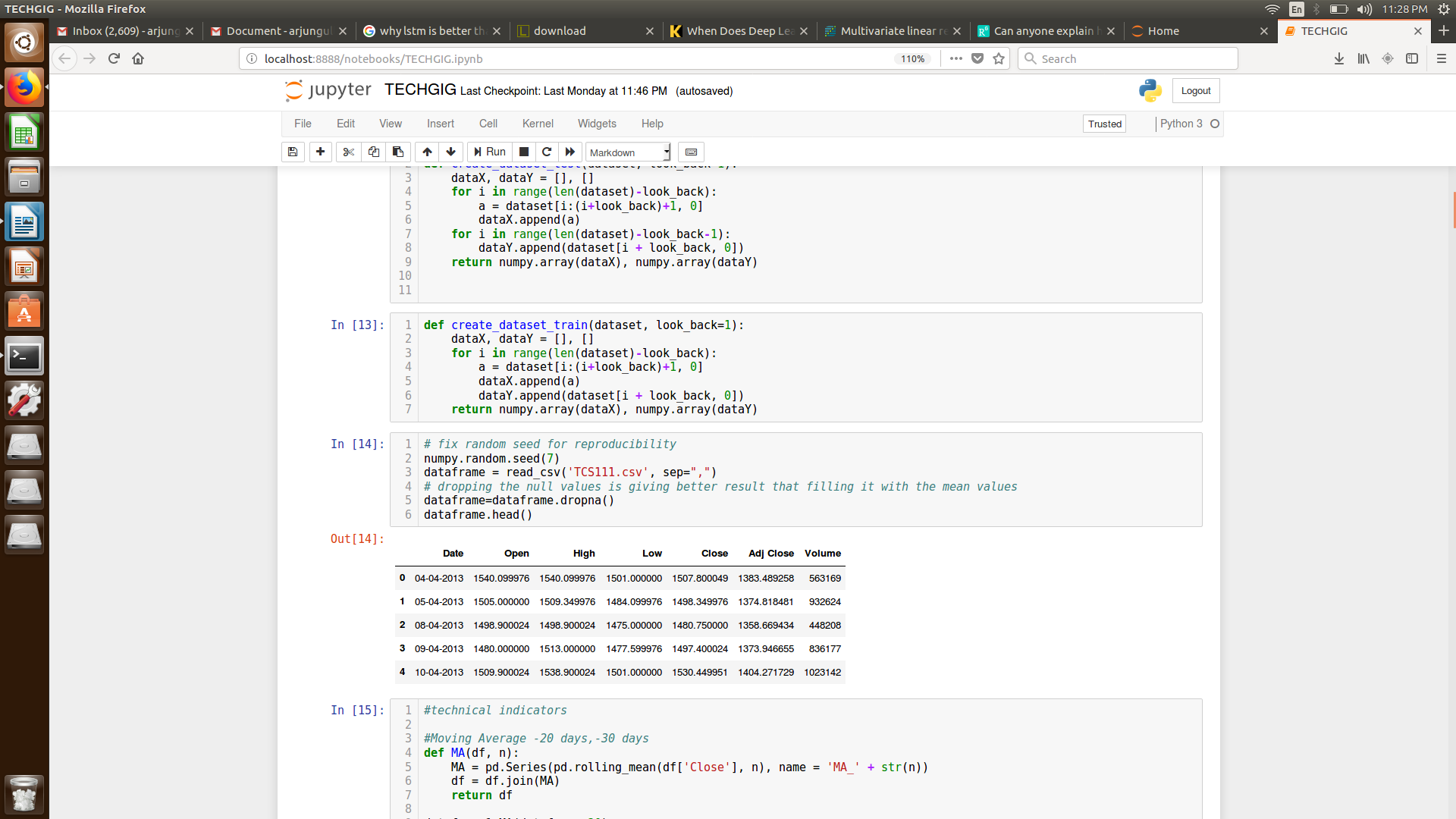
Our LSTM model is composed of a sequential input layer followed by 1 LSTM layers and dense layer.



**DATASET ANALYSIS AND FILTERING**

* Handling Null values:

The null values were filled with the average of the values before and after the null value. With this prediction was made and the accuracy was found lesser than the approach in which we drop the null columns completely.

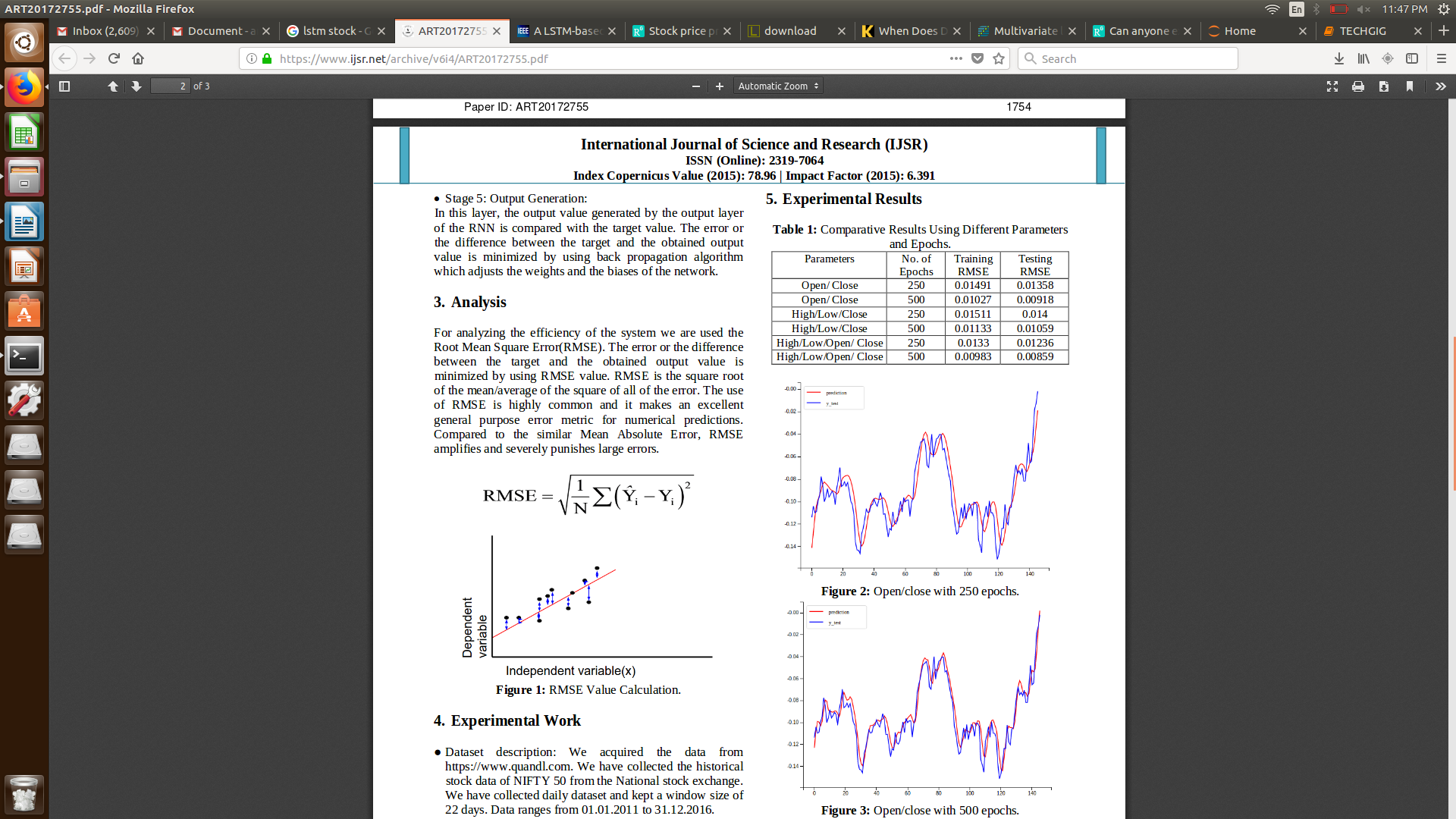


* Technical Indicators:

The Technical indicators were derived using the python maths library. We wrote separate function for each technical indicator because of Talib’s dependency on Visual C++ tools.

**ERROR CALCULATION**

For analyzing the efficiency of the system we are used the Root Mean Square Error(RMSE). The error or the difference between the target and the obtained output value is minimized by using RMSE value. RMSE is the square root of the mean/average of the square of all of the error.The use of RMSE is highly common and it makes an excellent general purpose error metric for numerical predictions. Compared to the similar Mean Absolute Error, RMSE amplifies and severely punishes large errors.

****

**LSTM for Time Series Analysis of Stock Prices**

There are various steps involved in making this LSTM network.

* Firstly, the preprocessing of data was done which included the normalization of data using the sklearn’s MinMaxScaler function for better accuracy, the data was cleaned i.e. the null values were removed from the dataset



Figure 1: Code for dataset Normalization

* Secondly, the dataset for split into the train and the test sets. The train data was 90-92% of the dataset while the rest was given to the test set. For validation, the test set did’nt include the input values to predict the closing price on 30th trading day.
* Thirdly, we have created a dataset with the input parameters. The input parameters for our LSTM network is previous 29 days closing price of the stock. The output parameter is the closing price of the stock on the 30th day.
* Next, we train an RNN (Recurrent Neural Net) LSTM network to predict the values. Our LSTM network is composed of a sequential input layer followed by a dense layer.

*\*Note: The parameters passed in LSTM were tweaked and are slightly different for every company’s dataset, which was done in order to maximize prediction accuracy.*

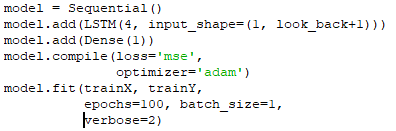


Figure 2: LSTM Network

**Analysis**

For analysing the efficiency of the system, we are used the Root Mean Square Error(RMSE). The error or the difference between the target and the obtained output value is minimized by using RMSE value. RMSE is the square root of the mean/average of the square of all of the error.



Figure 3: RMSE on YESBANK.NS dataset

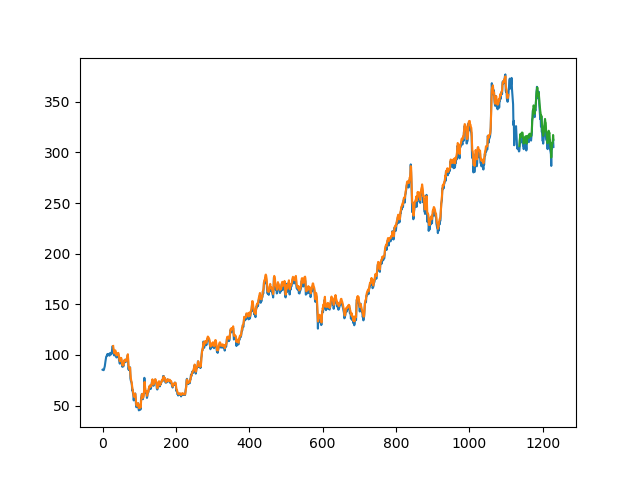
We have used the Technical Indicators for analysis but haven’t used any of them for predicting the values. Since we are not directly calculating the values of 30th trading i.e. 16th May but iteratively calculating the values from day 1 i.e. 5th April, therefore using more variables as input parameter increases the RMSE on the predicted values.

**Experimental Results**

For YESBANK.NS dataset, the graph shows the actual values received for different number of epoch and batch sizes.

1. **Number of epoch=50**

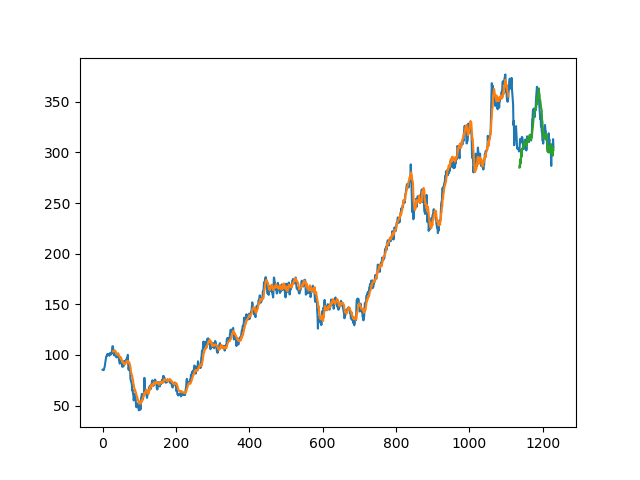
**Batch Size=1**

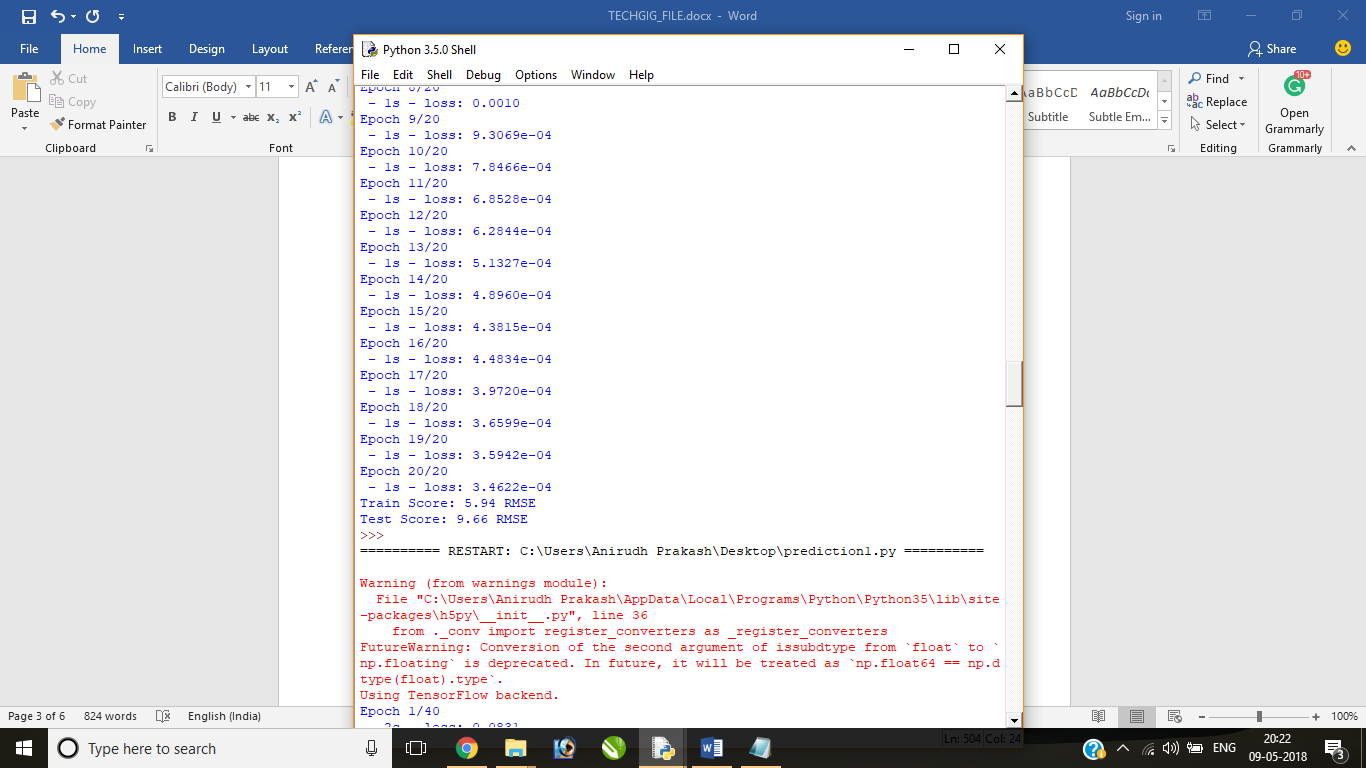




1. **Number of epoch=20**

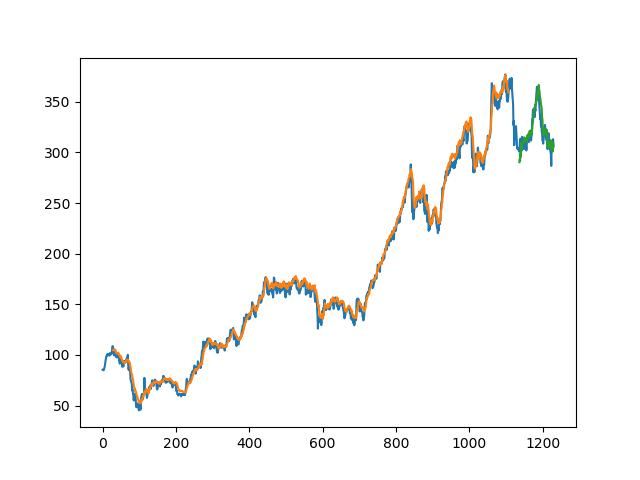
**Batch Size=10**

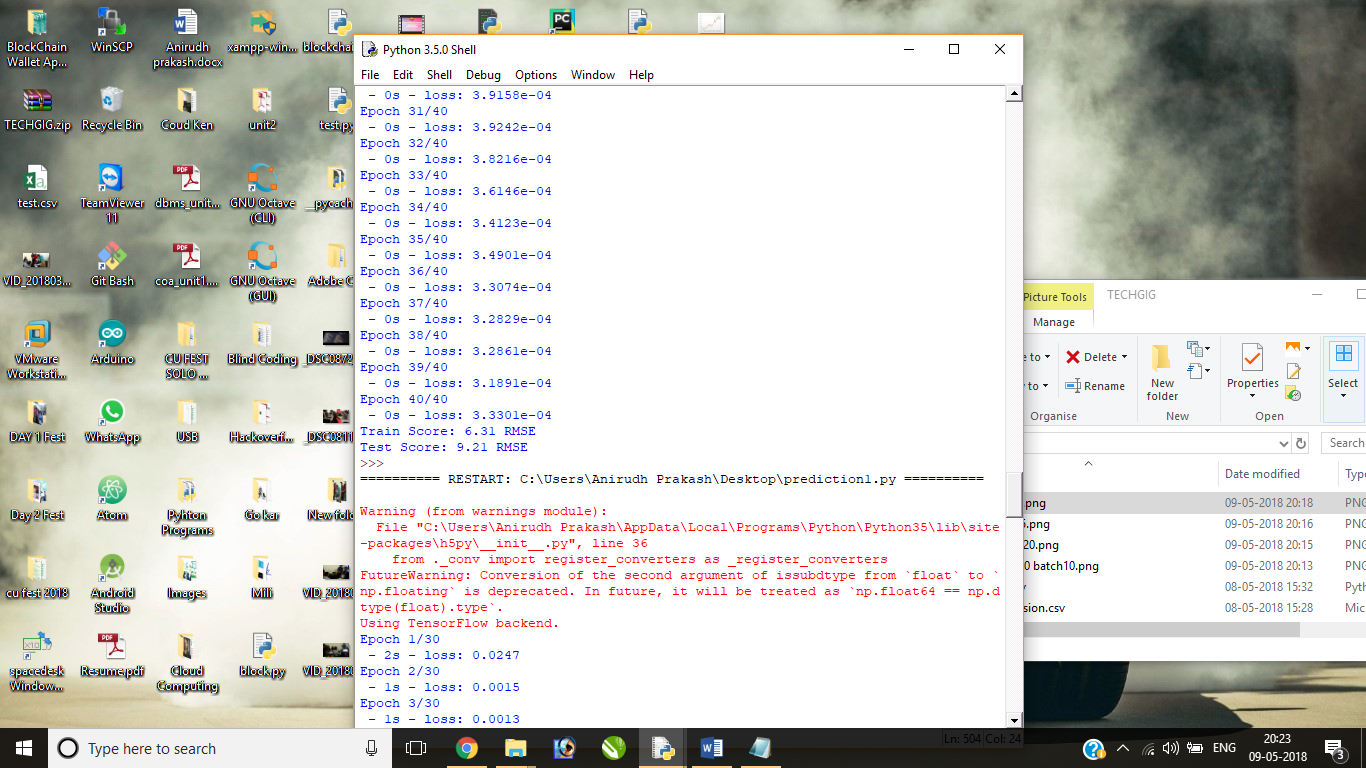




1. **Number of epoch=40**

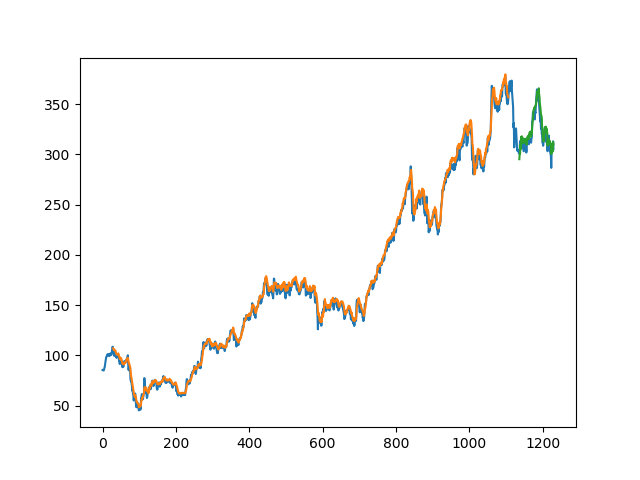
**Batch Size=20**

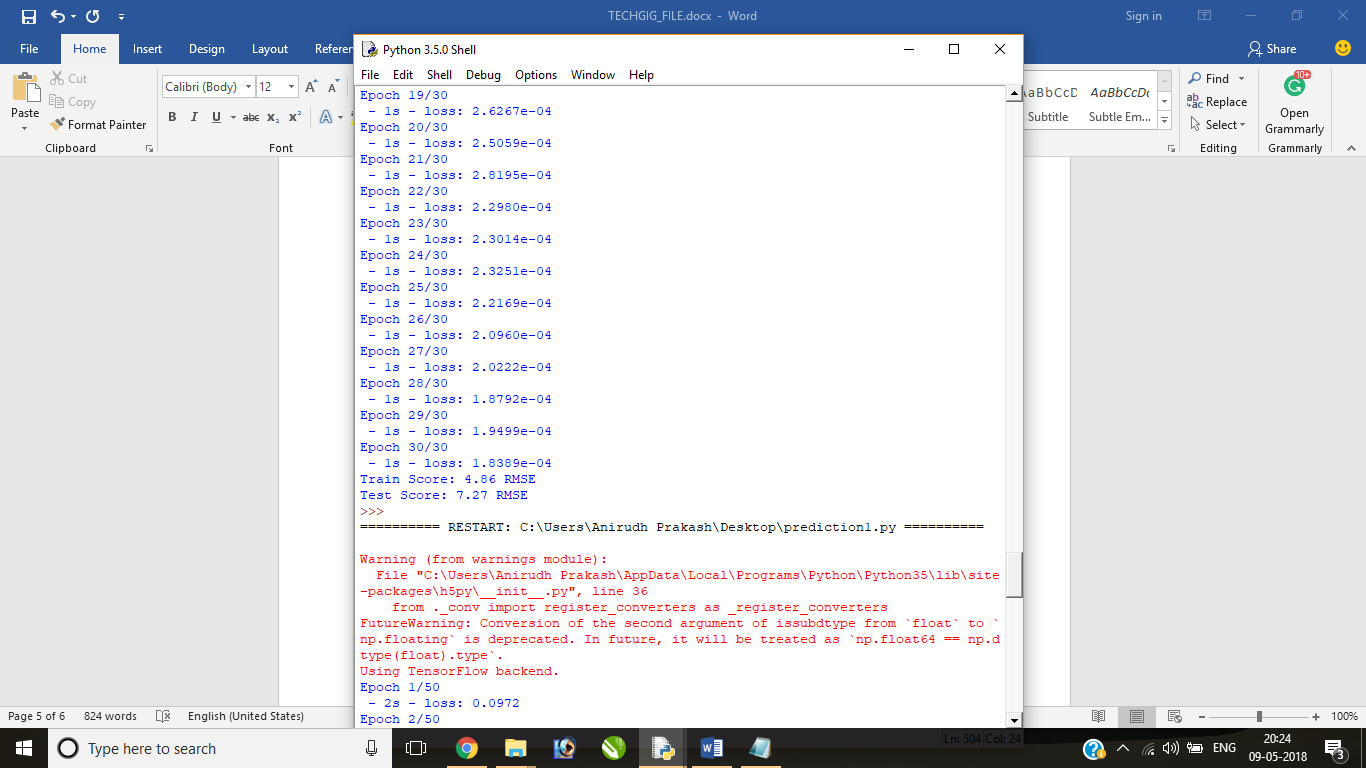




1. **Number of epoch=30**

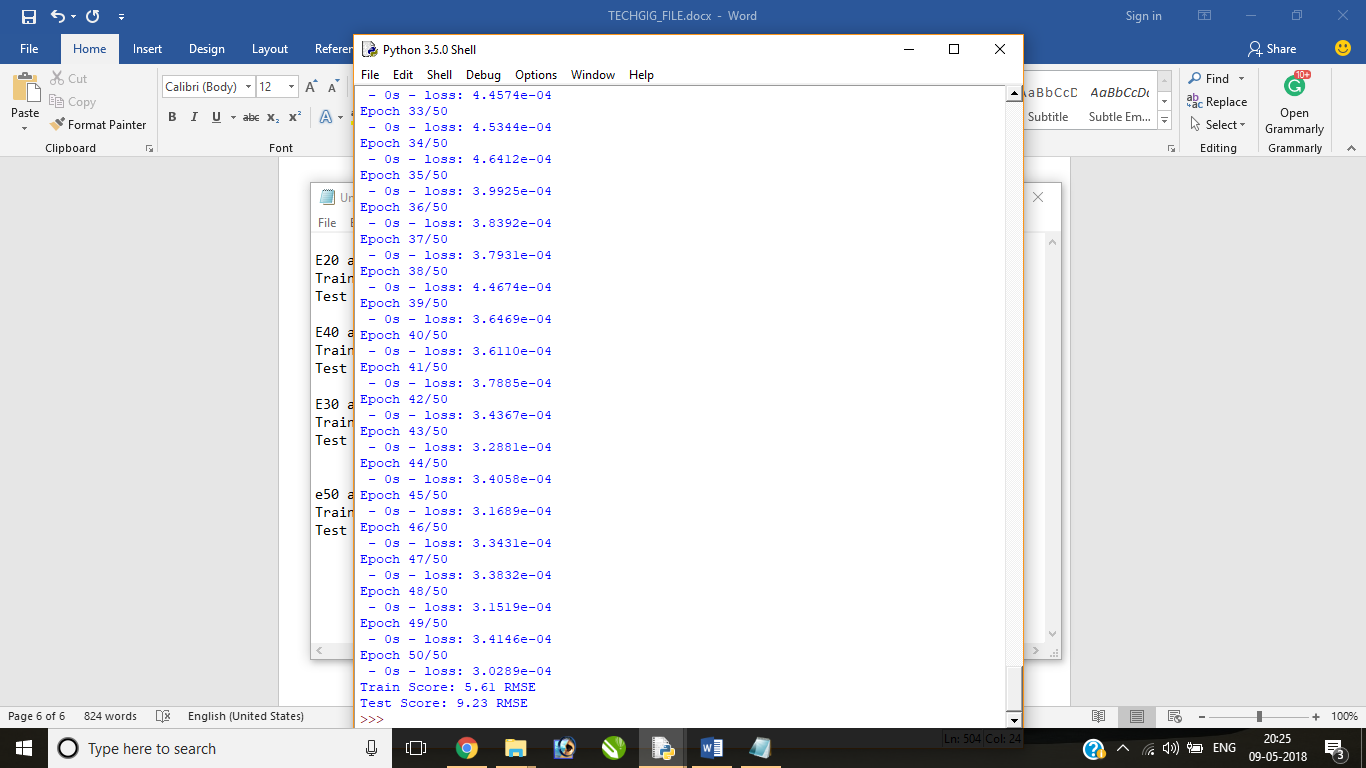
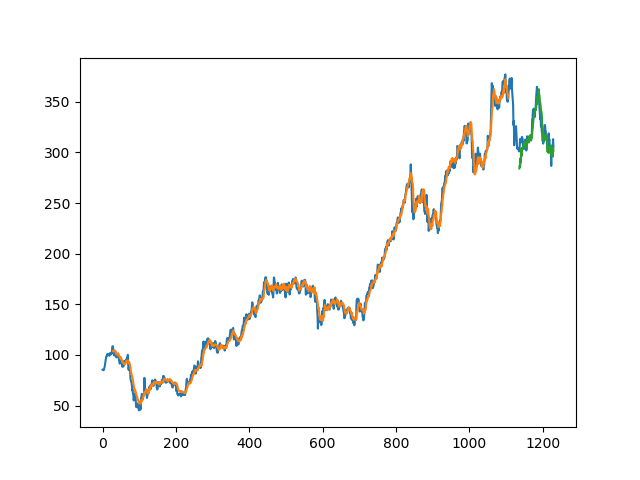
**Batch Size=5**





1. **Number of epochs=50**

**Batch Size =25**



In the above graph of close prices along with the predicted values graph of the train and the test set. The orange graph is the graph of predicted values on train set and the green graph is the graph of predicted values on the test set.

As we can see from the graphs, the model gives the minimum RMSE when the Number of epochs=50 and batch size=1. Hence the first arrangement was used for training the model.