PROJECT REPORT ON STOCK PREDICTION USING RECURRENT NEURAL NETWORKS.

Project By:
ABISHEK LAKSHMIRATHAN

Introduction: The project uses a forecasting model (recurrent neural network model) made up of LSTM layers to predict the future prices of a stock based on its previous closing prices as the predictor constraint.

Scope:

- The model created in the project can be used to study the variations in the stock prices of a group of companies and to show how the previous data co-relates with the future closing prices of a stock.
- The comparison of the actual prices (stock prices which we know) and the prices predicted by the model can be used to calculate the accuracy of the model and to understand how the recurrent neural network model deals with time-series data.

Dependencies:

- Tensorflow Backend for Machine Learning (on top of Keras)
- Python Packages used: Keras, Numpy, Pandas, Scikit Learn, Urllib and Matplotlib.

Working:

Data collection:

- The data was retrieved using Quandl API which included data over the time-period of 16 Years. The data included the Daily closing prices of the stock specified. I have Used Microsoft and Apple stock for our training purposes.
- The API provides us with http responses which is then decoded to 'UTF-8', cleaned and then stored in a CSV format which is then fed to the model.

Loading the Data:

- The load_data() function is responsible for feeding the data to the model. It is based on three parameters which include:
    - The File Name**:** This is the CSV generated during the data collection phase.
    - The Sequence Length: This is the value we use to split the data into categories.
    - Normalized Window: Every sequence category is fed to the window which is normalized to reflect percentage changes from the start of that window. This is done by normalize_windows function where the formula used to normalize the data is given by:

$$n_i = \left(\frac{p_i}{p_0}\right) - 1$$

Building the Model:

- It is a sequential model having one input layer consisting of a sequence of size 50 which is fed into a LSTM Layer with 50 Neurons.
- This layer in-turn feeds into another LSTM layer with 100 neurons which then feeds into a fully connected normal layer of 1 neuron with Linear Activation Function which will be used to give the prediction of the next time stamp.
- We used RMSProp as our optimizer and Mean Squared error(MSE) as the loss function.

Training the Model:

- We have divided the data into training and testing data. Training set uses 90% of the data available, 10 epochs have been run on 3613 samples of data having the run time batch size of 512.

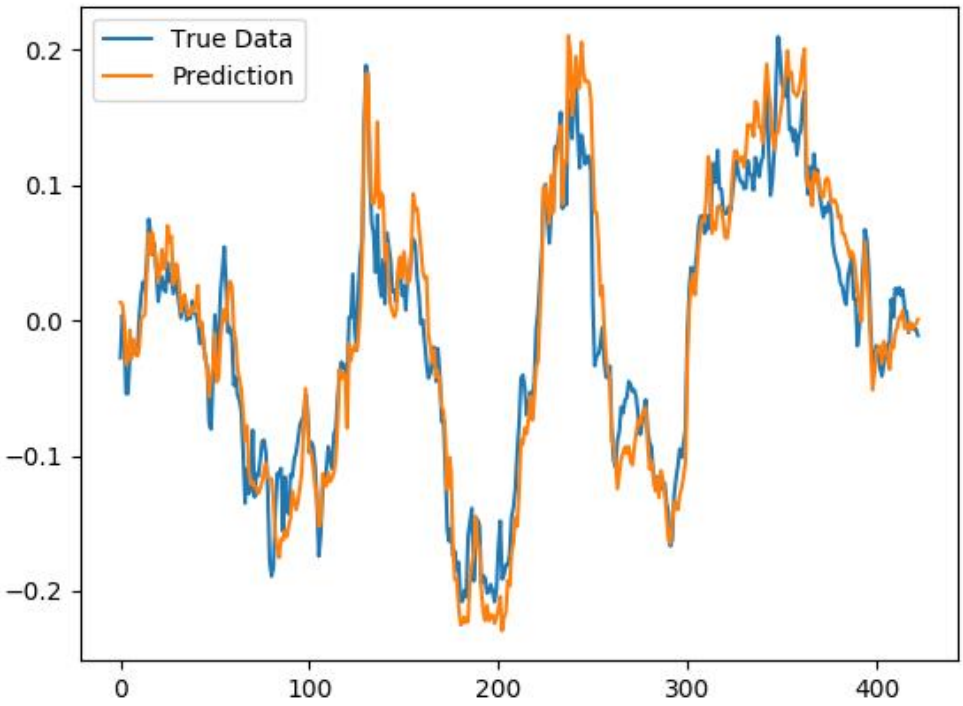
Testing the Model:

- The testing data consists of 10% of the available data and the Root Mean Squared error values have been calculated after running 10 epochs.
- The model has an average error of 40 prices in the training dataset and 18 prices in testing dataset.

```
Train Score: 39.66 RMSE
Test Score: 17.40 RMSE
```

Visualization:

- The plot consists of the original dataset (closing prices of stocks) comparison with predicted prices of the stocks.



This plot represents the testing data comparison where the data is normalized between -0.2 and 0.2 for 428 values.

Improvisations that can be implemented:

- Using more predictive constraints rather than only the history of closing prices such as: International impacts, Government factors and Volume of stocks Purchased and sold for each day.
- Larger size of training data so that the model used is more accurate.