Data Creation

The dataset is created from the dataset provided in the assignment. The dataset has 'Open', 'High', 'Low', 'Volume', 'Close/Last' and Date for each sample. The dataset is created by considering the last Three days as features for 'Open', 'High', 'Low' and 'Volume' values. In addition, in order to convert the problem as supervised learning problem the next day 'Open' value are considered as the Target. So in total we have 12 features for each of the last three days for Open', 'High', 'Low' and 'Volume' values and a 'Target' Column for the next days 'Open' value. The new dataset is created into a pandas dataframe by following this pattern and thus resulting it in a new dataset for the prediction problem.

Preprocessing

In the Preprocessing, the first step is to check if the data is in the right format by checking the data types. In the dataset, in the 'Close/Last' column the price values have '$' sign in them this should be removed. In addition, the dataset is also checked for missing values as it is a sequential dataset there is a chance that we might have some missing values. In this case there are no missing values. After this, the next step is to scale the dataset. As the volume column has very large values as compared to other columns so scaling of the dataset is required. In order to do the scaling MinMaxScaler Scaling function from Sklearn (https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html) which scales the entire dataset. In order to pass the dataset to the Neural Network reshaping of the dataset is required. Therefore the data is reshaped into size of (X_train_size, 12, 1) so that it can be feed to the Neural Network for learning and predicting the next day's Stock Price.

Steps of Choosing the Right Design

In order to choose the right design several expermentation with lstm was performed. First only one LSTM layer with 100 units along with one Dense layer with one Neuron in it was used for prediction with 20 Epochs initially. The loss for this design was quite high and it was decreasing slowly and it indicated the underfitting problem. The predicted output was a flat line and the Network was not predicting properly as it was predicting the same value for all the outputs. In the second step, the 2 more layers of LSTM with 50 units in it was added to see the results and with this design the loss was decreasing as expected and the output was better as compared to the previous design. However, the loss was still high and then the number of Epochs was increased to 2000 to check the performance of the network. After the training for 2000 Epochs the loss values dropped but the output was still not close to the test data so another anlysis into the network showed that the loss value was still higher so another layer of LSTM with 50 units in it was added and the Network was retrained for the training dataset. With this the results were quite satisfactory as the loss value decreased considerably along with closed output to the test dataset. This is the approach taken for creating the design.

Design Details

The Network design is shown below along with all the design choices.

```
self.Lstm_model = Sequential()
self.Lstm_model.add(LSTM(units = 100, return_sequences= True, input_shape = X_train_s
cl_reshape.shape[1],1)))
self.Lstm_model.add(Dropout(0.2))
self.Lstm_model.add(LSTM(units = 50, return_sequences=True))
self.Lstm_model.add(Dropout(0.2))
self.Lstm_model.add(LSTM(units = 50, return_sequences=True))
self.Lstm_model.add(Dropout(0.2))
self.Lstm_model.add(LSTM(units = 50))
self.Lstm_model.add(Dropout(0.2))
self.Lstm_model.add(Dense(units = 1))

self.epochs = 2000
self.batch_size = 32
self.Lstm_model.compile(optimizer = 'adam', loss = 'mean_squared_error')
self.history_Lstm_model = self.Lstm_model.fit(X_train_reshape, y_train, epochs = sel
f.epochs, batch_size = self.batch_size)
```

The details for the Network is shown above. The number of LSTM layers used is 4 along with one Dense layer with one unit in it. In addition, Dropout of 20% is used in order for it to avoid overfitting. The training and testing dataset is reshaped for the model. The number of Epochs used for training is 2000 along with a batch size of 32. The Optimizer for the Network is 'adam' along with MSE 'mean_squared_error' as the loss function type. The optimization algorithm used in this design is 'adam' which is mainly used in place of Stochastic Gradient Method to update the weights for the Network. One of the main advantage of using this optimization technique here is that the dataset is non-stationary and as from the ADAM (https://arxiv.org/abs/1412.6980) research paper it is clear that the 'adam' algorithm works better for non-stationary objectives. In addition, it is easirer to implement, Computationally more efficient with little memory requirements. Therefore, 'adam' is used in this design. In addition, for the loss function MSE is used mainly to make the problem Mathematically more suitable for solving the optimization problem.

Output From Training Loop with Comments

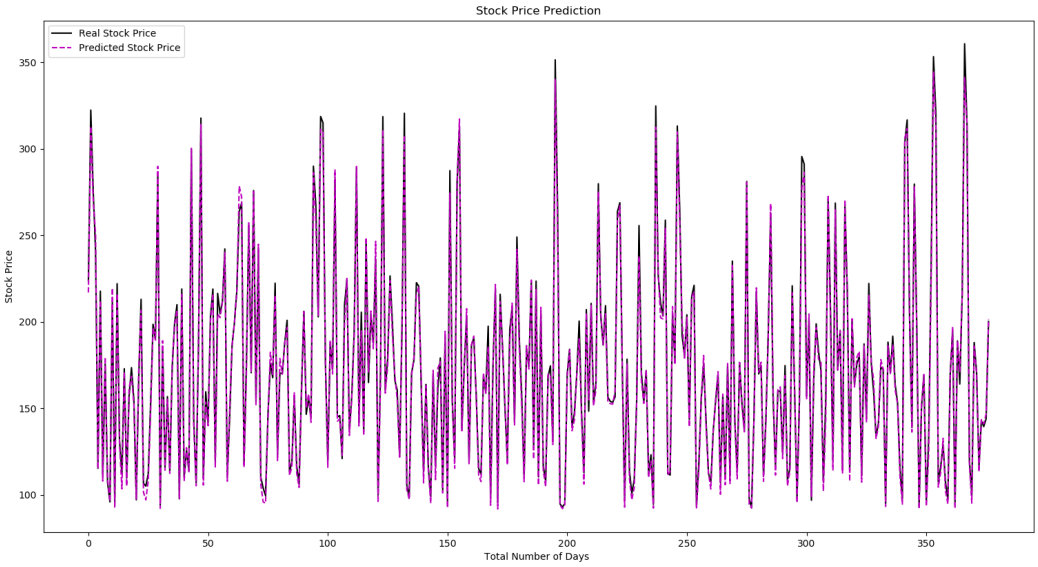
The Training output from the first 5 Epochs and last 5 Epochs is shown below

```
Epoch 1/2000
879/879 [==============================] - 5s 6ms/step - loss: 32418.8634
Epoch 2/2000
879/879 [==============================] - 2s 3ms/step - loss: 29928.2698
Epoch 3/2000
879/879 [==============================] - 2s 3ms/step - loss: 29206.7645
Epoch 4/2000
879/879 [==============================] - 2s 3ms/step - loss: 28646.6213
Epoch 5/2000
879/879 [==============================] - 2s 3ms/step - loss: 28134.5308
....
Epoch 1996/2000
879/879 [==============================] - 2s 3ms/step - loss: 151.6558
Epoch 1997/2000
879/879 [==============================] - 2s 3ms/step - loss: 173.0100
Epoch 1998/2000
879/879 [==============================] - 2s 3ms/step - loss: 167.7131
Epoch 1999/2000
879/879 [==============================] - 2s 3ms/step - loss: 189.6705
Epoch 2000/2000
879/879 [==============================] - 2s 3ms/step - loss: 150.9006
```

The output from the Training loop is clear that the loss decreases from Epoch 1 and it decreases as expected. Towards the Last 5 Epochs of the Network the loss is quite lower as compared to the First 5.

Output From the Testing With Plot and Comments

The Output image from the Neural Network is shown below. It has both the original test data Stock prices and the preditcted stock prices.



The Preditcted Stock price is shown in 'magenta' color and the Original Stock price is shown in 'black' color. From the graph it is clear that the Preditcted Stock Price is very close to the Original Stock Price as both the plots overlaps.

If more days used as Features

If the number of Features are increased by taking more days into consideration then the Network will not be able to capture the recent trend in the dataset and might fail to predict an accurate next day value. In other words the mean absolute percent error (MAPE) will increase and lead to inaccurate value being predicted by the Network.