Machine Learning Nanodegree

Capstone Proposal (Kushal Sharma)

Domain Background

Stock market prediction is the act of trying to determine the future value of a company stock or other financial instrument traded on an exchange. A stock market, equity market or share market is the aggregation of buyers and sellers of stocks (also called shares), which represent ownership claims on businesses; these may include securities listed on a public stock exchange as well as those only traded privately. The successful prediction of a stock's future price could yield significant profit.

History has shown that the price of stocks and other assets is an important part of the dynamics of economic activity, and can influence or be an indicator of social mood. An economy where the stock market is on the rise is considered to be an up-and-coming economy. The stock market is often considered the primary indicator of a country's economic strength and development.

With the advent of the digital computer, stock market prediction has since moved into the technological realm. The most prominent technique involves the use of artificial neural networks(ANNs) and Genetic Algorithms. The most common form of ANN in use for stock market prediction is the feed forward network utilizing the backward propagation of errors algorithm to update the network weights. These networks are commonly referred to as Backpropagation networks. Another form of ANN that is more appropriate for stock prediction is the time recurrent neural network (RNN) or time delay neural network (TDNN).

Problem Statement

In this project the challenge is to predict the return of a stock, given the history of the past few days/months/years. Provided N-day window of time, days N-2, N-1, N, N+1, and N+2. We have returns in days N-2, N-1, N, and we have to predict the returns in the rest of days N+1 and N+2. The goal of this project is to reasonably predict the price of individual stocks (with confidence interval) for the Indian Stock Market over a short period of time in the future, provided the historical data for that stock.

Datasets and Inputs

The dataset used in this challenge is taken from quandl. The query for data is constructed using two variables i.e. ticker and exchange. Ticker is the code name of the stock we need to get the data for (example TECHM for Tech Mahindra), exchange is the name of the exchange we want to fetch data for (example NSE for National Stock Exchange). The above example will return us the data for Tech Mahindra separated at day intervals for all the years.

The response for the above data call will be something similar to 6 columns below in the table, i.e. date, open, high, low, close and volume for the stock. Open is the opening price of the stock on that day. High is the highest price of the stock on that day. Low is the lowest price of the stock on that day. Close is the closing price for the stock on that day. Volume is the volume of the stock traded on that day.

	Date	Open	High	Low	Close	Volume
0	2015-08-03 15:30:00	523.50	533.50	521.50	529.00	811778.00
1	2015-08-04 15:30:00	540.00	549.00	524.15	525.00	2145670.00
2	2015-08-05 15:30:00	552.85	558.80	542.00	543.90	2640419.00
3	2015-08-06 15:30:00	532.50	557.05	531.10	556.00	1547635.00

The input will be a dataframe with two columns: ds and y. The ds (datestamp) column should be of a format expected by Pandas, ideally YYYY-MM-DD for a date or YYYY-MM-DD HH:MM:SS for a timestamp. The X column must be numeric, and represents the adjusted value of the stock we wish to predict.

Solution Statement

The solution to the problem will be predicting the future value of the given stock such that the value fall in a good confidence interval and is as accurate as the real value for the prediction timeframe. The solution will be to build a LSTM model using Keras and TensorFlow. The Long Short-Term Memory network or LSTM network is a type of recurrent neural network used in deep learning because very large architectures can be successfully trained. The trained model will be first tested on a validation data set to measure the performance while training; and then on the test data set after complete training to measure the actual performance of the model. The final aim of the project will be to predict the future value of the given stock for given N number of days in future.

Benchmark Model

The benchmark model used for this problem statement will be Prophet model by facebook. Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit. The original solution model and benchmark model will both be evaluated by comparing the predicted price with the original price at the given dates to predict. One metric that we will use will be root mean squared error of the original and predicted values. Also will be will be evaluating the models by using a validation period to calculate the profit/loss in that duration if we chose to play the market with N number of shares.

Evaluation Metrics

The evaluation metric that we will be using in this project is root-mean-square error (RMSE). The root-mean-square deviation (RMSD) or root-mean-square error (RMSE) (or sometimes root-mean-squared error) is a frequently used measure of the differences between values (sample values) predicted by a model or an estimator and the values observed.

$$\mathrm{RMSD}(\hat{\theta}) = \sqrt{\mathrm{MSE}(\hat{\theta})} = \sqrt{\mathrm{E}((\hat{\theta} - \theta)^2)}.$$

RMSD is always non-negative, and a value of 0 (never achieved in practice) would indicate a perfect fit to the data. In general, a lower RMSD is better than a higher one.

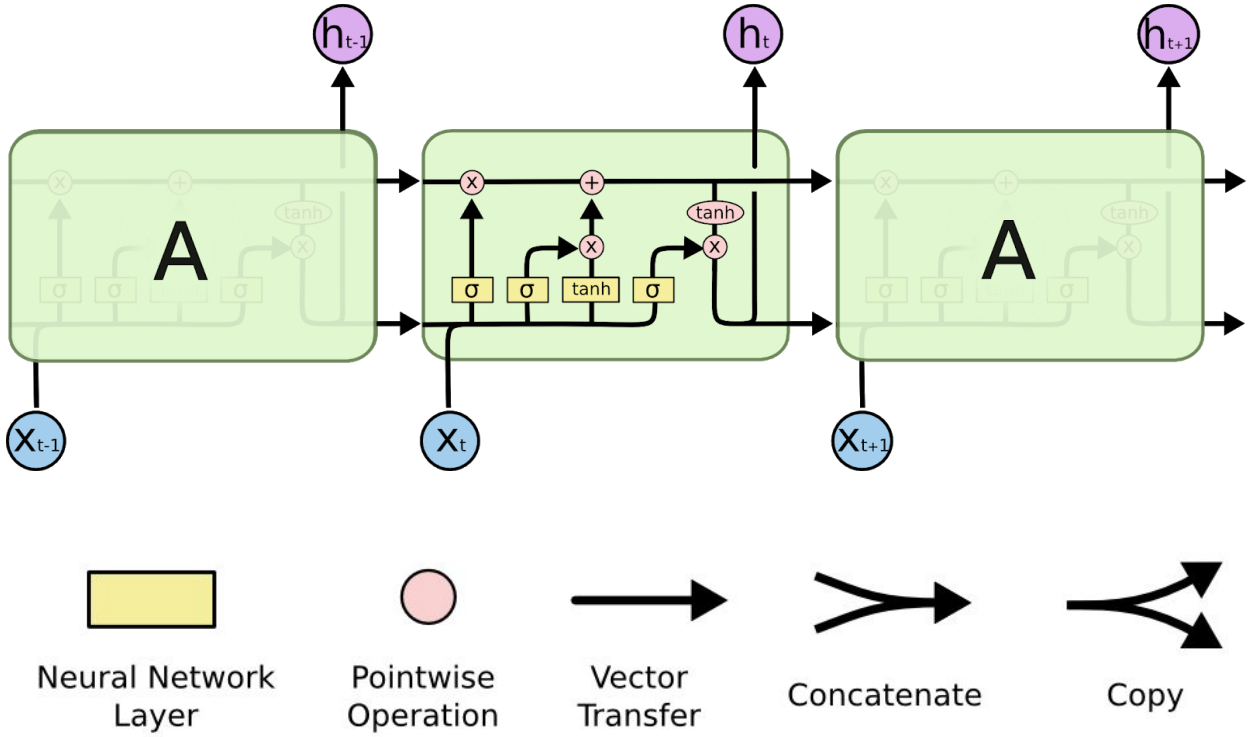
Project Design

Load the dataset from network call or csv file as a Pandas dataframe.

Normalize the data i.e rescale the data to the range of 0-to-1. We will normalize the dataset using the MinMaxScaler preprocessing class from the scikit-learn library.

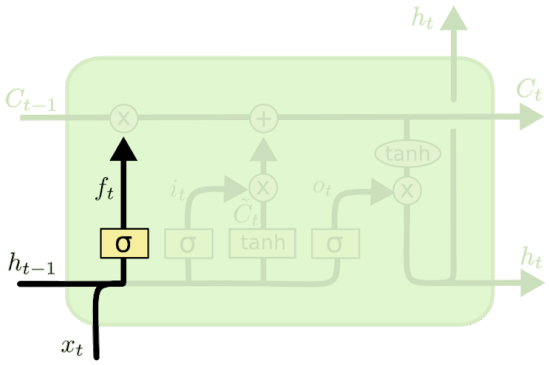
Split the ordered dataset into train and test datasets and modify dataset such that it fits the shape of LSTM network input [samples, time steps, features]

A typical LSTM network is comprised of different memory blocks called cells. There are two states that are being transferred to the next cell; the cell state and the hidden state. The memory blocks are responsible for remembering things and manipulations to this memory is done through three major mechanisms, called gates.



Forget gate

Forget gate is responsible for removing information from the cell state. The information that is no longer required for the LSTM to understand things or the information that is of less importance is removed via multiplication of a filter. This is required for optimizing the performance of the LSTM network.

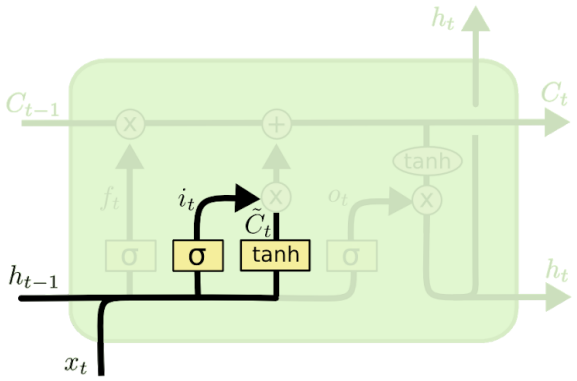


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

h_t-1 is the hidden state from the previous cell or the output of the previous cell and x_t is the input at that particular time step. The given inputs are multiplied by the weight matrices and a bias is added. After this, the sigmoid function is applied to this value. The sigmoid function outputs a vector, with values ranging from 0 to 1, corresponding to each number in the cell state. If a '0' is output for a particular value in the cell state, it means that the forget gate wants the cell state to forget that piece of information completely. Similarly, a '1' means that the forget gate wants to remember that entire piece of information. This vector output from the sigmoid function is multiplied to the cell state.

Input gate

The input gate is responsible for the addition of information to the cell state. First it regulates what values need to be added to the cell state by involving a sigmoid function.

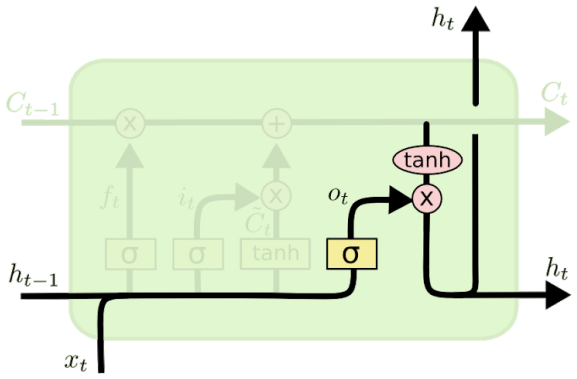


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

This is similar to the forget gate and acts as a filter for all the information from h_t-1 and x_t. Then it creates a vector containing all possible values that can be added (as perceived from h_t-1 and x_t) to the cell state. This is done using the tanh function, which outputs values from -1 to +1. Lastly, the value of the regulatory filter (the sigmoid gate) is multiplied to the created vector (the tanh function) and then this information is added to the cell state via addition operation.

Output gate

The output gate selects useful information from the current cell state and show it as an output. It creates a vector after applying tanh function to the cell state, thereby scaling the values to the range -1 to +1.



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Then it makes a filter using the values of h_t-1 and x_t, such that it can regulate the values that need to be output from the vector created above. This filter again employs a sigmoid function. Lastly it multiplies the value of this regulatory filter to the vector created using the tanh function, and sending it out as a output along with to the hidden state of the next cell.

Generate predictions using the model for both train and test dataset and use the evaluation metric to determine the accuracy of model. Lastly, predict the next value in sequence (i.e. price of the stock for the coming day).

References

- http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://facebook.github.io/prophet/
- https://en.wikipedia.org/wiki/Root-mean-square_deviation
- https://en.wikipedia.org/wiki/Long_short-term_memory
- https://en.wikipedia.org/wiki/Stock_market_prediction
- https://www.quandl.com/