Udacity Machine Learning Engineer Capstone Project Proposal

Using LSTMs for Stock Price Prediction

Ho Jun Liang

December 2020

Domain Background

Investors make investment decisions based on expectations of stock prices in the future. Often, this prediction is done by creating financial/valuation models or developing a qualitative expectation based on current affairs that may affect the stock prices. The former would more popularly be referred to as technical analysis, while the latter is popularly known as fundamental analysis. Here, an attempt will be made to tackle technical analysis using machine learning.

We will be exploring the use of the Long Short-Term Memory (LSTM) model, a deep learning sequential model. Jia (2016) investigated the effectiveness of LSTM networks for the use case of stock price prediction and showed that it was effective. This is because LSTMs are a specific type of recurrent neural network which have a form of 'memory' that makes use of previous time events to help to predict the next time event. This makes it quite suitable for the use case of stock price prediction, where data is in the form of time series.

Recurrent neural networks have two distinct and well-known problems – vanishing gradients and exploding gradients (Bengio et al., 1994) – because these networks tend to be very deep. The use of LSTMs helps to alleviate these problems, and is "achieved by an efficient, gradient-based algorithm for an architecture enforcing constant … error flow through internal states of special units" (Hochreiter and Schmidhuber, 1997).

Another problem common to neural networks would be that of overfitting. This is exacerbated by the fact that stock data has limited data points. Overfitting can be caused by an overly complex model and also by insufficient data used for training. Overfitting can be mitigated by applying the following strategies – making use of more training data, reducing the dimensionality of the model using Principal Component Analysis (PCA) and using regularisation techniques. Additional training data can be obtained by using features generated from other component stocks in the S&P 500 which may have correlated movements with the target stock. PCA has been shown to improve the accuracy of recurrent neural networks because it reduces dimensionality and hence the complexity of the model (Berradi and Lazaar, 2019). Dropout is a form of regularisation for dealing with the problem of overfitting by randomly dropping units and connections during training of the neural network (Srivastava et al., 2014). However, it may not be good for use in LSTM models as some 'memory' may be 'forgotten' if these units are dropped.

Problem Statement

The problem being investigated would be the question 'What is a stock's price in a certain specified future time?'. This project seeks to predict stock prices of a stock based on historical time-series data.

Datasets and Inputs

The historical time-series data mentioned before are the inputs and consist of daily trading data over a certain date range. These inputs will be taken from Yahoo! Finance's web API, and would have the following metrics: (1) opening price ('Open'), (2) highest price the stock traded at ('High'), (3) how many stocks were traded ('Volume') and (4) closing price adjusted for stock splits and dividends ('Adjusted Close'). The independent variables will be Open, High and Volume; the dependent variable (or target variable) will be Adjusted Close. The data to be extracted from the web API would be for all S&P 500 component stocks.

To maximise the number of data points (given that stock data is relatively limited), the period for extraction will be from 1 January 2000 to 30 November 2020. This should give 5261 data points, with the relevant metrics mentioned above for all 505 S&P 500 component stocks.

Below is an extract of the data points for a 5-day period for a stock ticker 'MMM':

	MMM_open	MMM_high	MMM_low	MMM_close	MMM_adjclose	MMM_volume
0	0.025821	0.025843	0.025792	0.025813	0.026489	-0.009411
1	0.004039	0.003955	0.004142	0.004079	0.002834	0.005582
2	-0.016311	-0.015952	-0.016796	-0.016381	-0.009636	0.035670
3	-0.000315	-0.000336	-0.000408	-0.000399	-0.002631	0.010229
4	-0.014047	-0.013882	-0.014300	-0.014138	-0.010881	0.008620

Solution Statement

This project aims to predict stock prices using recurrent neural networks, which is a type of neural network very suited to handle data with sequence dependence (time series data is an example of data with sequence dependence). In particular, the project will make use of Long Short-Term Memory ('LSTM') networks in Python using the Keras deep learning library.

Benchmark Model

The benchmark model to be used is linear regression. Linear regression is suitable for time series data as well, and has the same inputs as LSTM. It will serve as a performance benchmark for the LSTM model.

Evaluation Metrics

Given that this is a regression problem, some suitable evaluation metrics like R-square and root mean squared error ('RMSE') will be used. R-square is a measure of how much the variation in the dependent variable is influenced by the variation in the independent variables. RMSE is a measure of the average deviation of the predicted value as opposed to the mean of the actual value. The desired result for R-square is for it to be large (as that implies that there is a strong relationship between the dependent and independent variables); that for RMSE is for it to be small (as that implies that the model is accurate with the predicted not deviating much from the actual).

Project Design

1. Create Github repository and link to AWS.
2. Import relevant Python libraries for use.
3. Import data from Yahoo! Finance for all S&P500 stocks.
4. Perform exploratory data analysis to identify if there needs to be normalisation and other pre-processing to be done. This could possibly be done using the Matplotlib library.
5. Split the pre-processed data set into training and testing data sets.
6. Create the benchmark linear regression model using scikit-learn and note the evaluation metrics for it – as mentioned earlier, these should be R-squared and RMSE.
7. Create the LSTM model using Keras and note the evaluation metrics for it. Vary the relevant hyperparameters to increase effectiveness.
8. Compare and assess the effectiveness of the two models in the prediction of stock prices.
9. For future development: serve this model into a web application for use using AWS API Gateway, AWS Lambda etc.

References

1. Jia H. (2016). Investigation into the effectiveness of long short-term memory networks for stock price prediction. arXiv preprint arXiv :1603.07893
2. Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 5(2), 157-166. doi:10.1109/72.279181
3. Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735-1780. doi:10.1162/neco.1997.9.8.1735
4. Berradi, Z., & Lazaar, M. (2019). Integration of Principal Component Analysis and Recurrent Neural Network to Forecast the Stock Price of Casablanca Stock Exchange. Procedia Computer Science, 148, 55-61. doi:10.1016/j.procs.2019.01.008
5. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15(1).