Machine Learning Engineer Nanodegree Capstone Proposal

Machine Learning Approach to Stock Price Prediction

Zhirui Wang

December 27, 2017

Domain Background

Stock price prediction is a popular topic throughout last century. Using statistical methods and stochastic analysis to make stock price prediction is the mainstream in last 30 years, while using machine learning techniques to predict stock price is becoming more and more prevalence in recent years.
In their research *Support Vector Machines for Prediction of Futures Prices in Indian Stock Market*, Shom Prasad Das and Sudarsan Padhy discuss Back Propagation Technique and Support Vector Machine Technique to predict futures prices traded in Indian stock market. The reported NSME for all the futures stock index taken into consideration fall in the range of 0.9299 to 1.1521, which is a decent result.
In their study *A deep learning framework for financial time series using stacked auto encoders and long-short term memory*, Wei Boa, Jun Yue and Yulei Rao discuss combining wavelet transforms (WT), stacked auto encoders (SAEs) and long-short term memory (LSTM) to forecast stock index prices. The performance is reported across a year worth of data across various types of market and report Mean absolute percentage error (MAPE), correlation coefficient (R) and Theil's inequality coefficient (Theil U). In developed market they predict CSI 300 index, with average value of MAPE and Theil U of WSAEs-LSTM as 0.019 and 0.013.
It seems like machine learning and even deep learning methods can yield good result in stock price prediction. That is why I would like to try my own methods in this project.

Problem Statement

The project aims to create a machine learning model that can predict stock price by using historical information as a time-series data. The task is to build a stock price predictor that takes daily trading data over a certain date range as input, and outputs projected estimates for given query dates. The inputs will contain multiple metrics, such as opening price (Open), highest price the stock traded at (High), how many stocks were traded (Volume) and closing price adjusted for stock splits and dividends (Adjusted Close); my system will need to predict the Adjusted Close price.

Datasets and Inputs

In this project, I will use all the stocks in the S&P 500 as inputs and target. For each stock, the data will contain `Open`, `High`, `Low`, `Close`, `Adj Close` and `Volume`(already explained in the problem statement part) six variables. The `Adj Close` of each stock can be the target, and rest variables of this stock and all the variables of other stocks can be inputs. In order not to include structure break in the data set, we should pick a relatively stable time period, which ends up been from 2009 to 2017.
We will get the symbol list of all S&P 500 stocks from Wikipedia, and then use Python module `pandas-datareader` to query the prices and volumes from Yahoo-Finance.

Solution Statement

In this project, I would like to use recurrent neural network to solve the problem. I will use Long-short term memory network as the model, the stock that the user choose as the target, and all the historical time series

data of the target stock itself and other stocks as inputs. I will use sequence length of 1 month or 2 month as the length of the LSTM network(this will be a hyperparameter). The prediction will be one-step-ahead stock price of the target stock. Once the model is trained, the user can choose what is the 1-month or 2-month period they want to take as input, and the model can predict the next date's adjusted close price.

Benchmark Model

The benchmark model for this project would be a linear regression. Including lagged features of the dependent variable and other exogenous features in a linear regression is called autoregressive model with exogenous inputs, which is proved to be very successful in statistical time series modeling. This benchmark will use exact the same input as our LSTM network model, and provide a benchmark performance for the LSTM.

Evaluation Metrics

This is a regression problem, so the metrics of choice would be R-square and root-mean-squared-error. R-square can provide how much variation in the dependent variable can be explained by the variation in the independent variables. Root-mean-squared-error can provide what is the average deviation of the prediction from the true value, and it can be compared with the mean of the true value to see whether the deviation is large or small.

Project Design

For each of the stocks, a model will be built separately for that stock's `Adj Close`. Let us call this target stock's `Adj Close` price as target price.
First we will use `pandas-datareader` module to query all the `Open`, `High`, `Low`, `Close`, `Adj Close` and `Volume` from S&P 500 stocks, from 2009 to 2017. Because there are 500 stocks in S&P 500, and each stock has 6 variables, which makes us have 2999 features in our dataset. So the first we should do is feature selection. I will use Pearson correlation to calculate the correlation of target price with all the other 2999 features, and select 100 features that has the highest correlation with the target price. If there is a tie, just randomly pick one because these tied features should have quite similar information provided.
After feature selection, the remaining 100 time series should be turned into sequences. We may think today's target price will only depend on the features till one-month ago, then the sequence length would be one month. This is a hyperparameter we need to tune, and I just assume it is one-month in this proposal. The sequence generation will be as follows: set a rolling window of length 20 days(approximately the number of trading days in a month), move this window along the time series, and record the data in each step as the feature sequence for the next day. This will gives us 2000 features in our feature set.
After we have generated our feature sets, we can input these features into the benchmark linear regression and LSTM networks. I will use scikit-learn's `LinearRegression` class to do the linear regression, and use Keras's `LSTM` function to build the neural networks. The networks will have two or three layers, each with hidden states dimension of 100-50 or 150-100-50. Each layer will use Relu as activation function, and each layer will have a dropout layer with keep probability of 80% in order to regularize. The output layer will have dimension of 1, and linear activation function.
I will split the data into 2009 to 2016 as training, and 2017 as validation data. I will use R-square and RMSE to check the model performance, on both benchmark model and LSTM model. Once a successful model is built, the user can choose which 20 days of data they want to use as input, and the model will predict the next date's target price.

Reference

Das, Shom Prasad, and Sudarsan Padhy. "Support vector machines for prediction of futures prices in Indian stock market." International Journal of Computer Applications 41.3 (2012).

Bao, Wei, Jun Yue, and Yulei Rao. "A deep learning framework for financial time series using stacked autoencoders and long-short term memory." PloS one 12.7 (2017): e0180944.