Machine Learning Engineer
Nanodegree

Capstone Proposal

Investment and Trading

Stock Price Prediction

Krzysztof Mazur

mazurkrzysztof.k@gmail.com

February 2, 2017

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. The successful prediction of a stock's future price could yield significant profit. The efficient-market hypothesis suggests that stock prices reflect all currently available information and any price changes that are not based on newly revealed information thus are inherently unpredictable. Others disagree and those with this viewpoint possess myriad methods and technologies which purportedly allow them to gain future price information. (?, ?).

Different strategies are applied to gain competitive advantage over other market participants like machine learning, bespoke hardware solutions and even infrastructure investments. For example in HFT[1] sophisticated hardware push the boundries of what is possible in shorter and shorter amount of time (not milliseconds but microseconds[2]). Even a slightly better improvement can yield substantial profits. In this study focus is placed on a longer term; starting from one day prediction as opposed to intra-day prediction.

In software area multitude of algorithms exist such as Logistic Regression to forecast the direction and strength of Stock Market Movement (?, ?), SVMs[3] (?, ?) to predict future price, artificial neural networks (?, ?) including RNNs[4] (?, ?) and many more.

Such intensive studies prove that the topic is highly attractive for researchers and trading companies across the globe. The motivation behind studying and predicting the stock market movements are not only backed by financial gratifications but also by interesting conclusions that can be made from crowd and behavioural psychology analysis, sentiment analysis and interconnections between different news mediums and even general time-series forecasting. All those branches can benefit from the stock market prediction research which is very appealing problem to solve and improve upon.

Problem Statement

The goal of this work is to implement a stock market predictor which for given ticker symbols and dates predicts the price of the supplied tickers on that days. The predictor should first build a model using machine learning algorithms and then calculate ticker prices. Basically the problem is to create a function that maps ticker, date pairs to corresponding ticker, price pairs. To be more precise: create a function that maps date range and list of tickers to a function which maps list of tickers and dates to prices for those tickers.

$$f : ([d_0, d_1], [t_0, t_1, ...]) \mapsto \{g : [t'_0, t'_1, ...], [d'_0, d'_1, ...] \mapsto [p_0, p_1, ...]\}$$

Where $d_0, d_1, ...$ are days, $t_0, t_1, ...$ are ticker symbols and $p_0, p_1, ...$ are prices.

The evaluation metric is some measure of difference between predicted and real price e.g. *coefficient of determination* $\boldsymbol{R^2}$. The results can be easily reproducible by setting seed values

[1] High Frequency Trading

[2] Algo-logic claims that their system goes below sub-microsecond for tick-to-trade. One microsecond is the time the light travels approximately 300 meters.

[3] Support Vector Machines

[4] Recursive Neural Networks

for algorithms and system state and because the past is already written when it comes to ticker price history.

Datasets and Inputs

The dataset(s) and/or input(s) to be used in the project are thoroughly described. Information such as how the dataset or input is (was) obtained, and the characteristics of the dataset or input, should be included. It should be clear how the dataset(s) or input(s) will be used in the project and whether their use is appropriate given the context of the problem.

For this work the *Yahoo! Finance API* will be used as the source of datasets. It's convenient to use as it provides wide range of ticker symbols, is fast and gives automatically adjusted close price. Format of the dataset is also a good fit for the Pandas DataFrames as it is column-based. There are seven columns that can be used: Date, Open, High, Low, Close, Volume, Adj Close. Of course the dataset is not perfect as there are only the companies that are successful and didn't go bankrupt. This imposes a survivor bias in any machine learning algorithm that can be used - the algorithm will know about only successful companies and will won't capture the characteristics of companies that are collapsing. Datasets with companies that are no longer on stock market are available but are difficult to acquire or are very expensive because it takes a lot of work to select, gather, clean and put it all together. For the purpose of this work *Yahoo! Finance API* is sufficient. The fundamental features of companies is more difficult to acquire and is not provided as a daily data (rather quarterly) so it won't be used in this study.

Input values such as ticker symbols and dates for those tickers are provided by the user.

Solution Statement

In this paper the Long-Short Term Memory Neural Network (LSTM) with Discrete Wavelet Transformation (DWT) will be used to predict stock value. Feature extraction will be made with DWT and wavelet filters will be analysed and compared in order to find the best match for financial series domain.

LSTM Neural Networks are specific type of Recurrent Neural Networks (RNN) which are growing in popularity recently. In general, RNNs are used specifically in time-series domain as they have the ability to remember previous values from the time-series which makes them a perfect match for financial time-series prediction.

The solution will use *k-fold* cross-validation against multiple tickers in order to ensure the stability of the model and consistent results. Besides that the model parameters such as hidden layer neurons, learning rate, etc. will be evaluated to find the best solutions.

Benchmark Model

In order the compare the solution proposed in this paper the benchmark will be made against existing models and compared with the built algorithm. The benchmark models that will be used are very popular SVMs and Random Forests.

Evaluation Metrics

Ideally the stock predictor can be used in an actual automatic stock trading. The closer the predicted price value is to the real value the better score the model should have. In regression models *coefficient of determination* $\boldsymbol{R^2}$ is often used as it provides a measure of how well future samples are likely to be predicted correctly by the model:

$$\boldsymbol{R^2(y, \hat{y}) = 1 - \frac{\sum_{i=0}^{n-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{n-1}(y_i - \bar{y})^2}}$$

$$\boldsymbol{\bar{y} = \frac{1}{n}\sum_{i=0}^{n-1} y_i}$$

where $\boldsymbol{\hat{y}_i}$ is predicted value of $\boldsymbol{i}$-th sample, $\boldsymbol{y_i}$ is the corresponding true value, $\boldsymbol{n}$ is the number of samples.

The $\boldsymbol{R^2}$ metric is not perfect and has its limitations. It can't determine whether the coefficient estimates and predictions are biased and doesn't indicate whether a regression model is adequate. Therefore it's required to asses residual plots before making conclusions.

Although $\boldsymbol{R^2}$ has some drawbacks for the stock prediction problem it's a strong candidate since it reflects prediction precision considerably well.

Project Design

In order to build good and correct model the analysis needs to be divided in several steps:

1. Data aquisition - The dataset is freely available online on demand due to Yahoo! Finance API but despite its convenience it doesn't contain stocks that went bankrupt. A discussion should be made on the surviror biases.
2. Data wrangling - It's highly likely that the datasets will contain some missing values (dates that company didn't trade on Stock Market Exchange), scaled values, etc. Those kind of issues have to be resolved before feeding them into the model.
3. Feature transformations - Sometimes it's useful to have derived values (e.g. Moving Average Convergence/Divergence) to add as a feature as it may increase the overall performance of the algorithm.
4. Data analysis and feature selection - Some values might be not necessary in the training and predicting the future price due to for example high correlation between the features. The analysis should be made which features should end up in the processing pipeline.
5. Pipeline creation - The model will be implemented as a straightforward pipeline taking input values and giving predictions. Usage of this model should be convenient to the user and it should provide the ability to check the model performance
6. Model training - This step will train the proposed model to gain insights how it behaves during different conditions.
7. Results analysis and summary - The last but not least is to analyse results, compare them against other models, make conclusions and present them in a easy to understand manner.

References