Machine Learning Engineer Nanodegree

Renato L. F. Cunha

May 28th, 2017

Stock Price Indicator Capstone Proposal

Domain Background

Statistical models and machine learning have been used in various domains, ranging from baseball player performance prediction to stock prediction. In the latter case, investment firms, hedge funds and small investors develop or follow financial models to understand and, to some extend, predict market behavior to make profitable investments.

Using machine learning with stock data is particularly interesting, because we have access to daily stock information for periods spanning decades.

Problem Statement

The problem tackled by this project is that of predicting stock prices for future dates given historical data about such stock items. 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). The objective in this project is to predict the Adjusted Close price. The simplest solution would be to predict the mean value of the adjusted close price, but clearly we can strive to do better than that.

Datasets and Inputs

From the problem statement, it should be clear that what one would want to use to solve such a problem would be stock data itself. Fortunately, there are several sources for historical stock price, such as Yahoo! Finance, Bloomberg and Quandl, to name a few.

The easiest way to obtain the data during development of the model is to download CSV files from Yahoo stock data. For a production system, one could

use the Yahoo! Query Language to download the historical data on-the-fly to train the models. Other APIs include the Bloomberg API and the Quandl API.

Independent of the API or data source used, the data is a daily time series, with the needed information in separate "columns". (Yahoo! Query Language returns results as XML, but one can trivially convert that to tabular data.) Columns are the metrics mentioned above, such as Open, High, Low, Close, Volume and Adjusted Close.

Solution Statement

My proposed solution is a machine learning model that outputs the predicted Adjusted Close value of a set of stock items for a ticker symbol (e.g. GOOG, AAPL, etc.). The output is a number: the value in dollars of the stock.

Benchmark Model

As aforementioned, the simplest model that makes some sense is the mean of stock prices. A slightly better model would be one that outputs the *rolling* mean value of the stock. I will compare my model's performance with the rolling mean output for different values of window sizes, such as one week, 15 days and a month. Since means output the same unit of the input data, the benchmark model also outputs its values in dollars.

Evaluation Metrics

Some possible evaluation metrics are: Mean Squared Error (MSE) of predictions and accuracy within some percentage of the actual value.

Project Design

The workflow would basically be:

1. Download data to implement and validate the predictor;
2. Actually implement the predictor;
3. Test and measure performance of the predictor;
4. Implement more user-friendly user interface.

Stock predictor

For the stock predictor I intend to implement:

- A training interface that accepts a data range `(start_date, end_date)` and a list of ticker symbols (e.g. GOOG, AAPL), and builds a model of stock behavior. Your code should read the desired historical prices from the data source of your choice.
- A query interface that accepts a list of dates and a list of ticker symbols, and outputs the predicted stock prices for each of those stocks on the given dates. Note that the query dates passed in must be after the training date range, and ticker symbols must be a subset of the ones trained on.

For the initial training, as input data I will download CSVs from Yahoo! Finance for different companies. After that I will build the model. Some alternatives are:

- Linear regression: this is simple and, if it works well, this is a good model. Just fit a line and extrapolate to find your prediction.
- Generalized Linear Models using the one-parameter exponential families are a generalization of linear regression and can also be used.
- A more complex model for time series data is the ARIMA (autoregressive integrated moving average) model.
- LSTM (long short-term memory neural network) models are another model suited for time-series data that will be investigated.

Testing and performance measurement

A basic run of the core system would involve one call to the training interface, and one or more calls to the query interface. I will implement a train-test cycle to measure the performance of my model and use it to test prediction accuracy for query dates at different intervals after the training end date, such as the day immediately after training end date, 7 days later, 14 days, 28 days, etc.

User interface

I will implement a browser-based interface to let users specify stocks they are interested in. With this the interface will fetch the data, train the model and predict stock prices at predefined intervals.