Machine Learning Engineer Nanodegree

Capstone Proposal

Yong Yu March 11st, 2017

Proposal

Domain Background

Bitcoin has became one of the hottest topic in main stream slowly ever since silk road (https://en.wikipedia.org/wiki/Silk_Road_(marketplace) was shut down by FBI. If you haven't heard of it, you can check it here (https://en.wikipedia.org/wiki/Bitcoin). In short, you can think Bitcoin as a distributed currency system, although it is behaving more like an investment today as more and more people are trading instead of using them. It was 2013 when I first heard about Bitcoin, and ever since then I became an enthusiast about its future. I also bought my first Bitcoin when I first heard about it, and started my trading bitcoins ever since. After studying about machine learning, I think it would be interesting to apply it into trading bitcoin, and see if I could beat the market.

In 2014, a group of researchers from MIT (https://arxiv.org/pdf/1410.1231v1.pdf) claimed that they doubled their bitcoin investment using a Bayseian regression model. Same year, researchers from Stanford (http://cs229.stanford.edu/proj2014/Isaac%20Madan,%20Shaurya%20Saluja,%20Aojia%20Zhao,Automated%2(extended this experiment with more training features and three algorithms, GML, SVM and Random Forest, to find out that Random Forest worked the best with a ten minutes interval, resulted in an accuracy score of 0.574. My goal in this report is to examine their methods and see if I could reproduce their results.

Problem Statement

The problem here I am trying to solve is to predict bitcoin's price over a given time window, which could be days, hours, minutes, or seconds. In specific, I want to build three models to predict bitcoin's price with an accuracy over 50%, each has a time window of 60 minutes, 30 minutes, and 10 minutes. Several common machine learning algorithms applied in trading will be experimented to find out the best one. The current choices are GLM, SVM, Random Forest, and RNN.

Datasets and Inputs

All the data used in this project are from GDAX's API (https://docs.gdax.com/). The dataset has the following six fields,

- datetime - precision down to one second;
- open price
- close price
- high price
- low price
- volume

All these six fields will be used as features for training. There are half a year's dataset available, ranging from July 1st, 2016 to Dec 31st, 2016.

Solution Statement

To best predict the price of bitcoin, four different machine leanring algorithms will be explored in this project,

- GLM, generalized linear model;
- SVM, support vector machine;
- Random Forest;
- RNN, recurrent neural network.

There are three time windows applied, 10 minutes, 30 minutes, and 60 minutes. Each of them will use the same training set as inputs, and the outputs are price at any given time t, the future price t + m, where m = 10, 30 or 60 minutes.

We will use the first 70% of the original data as training set, and the rest will be used as testing set.

Benchmark Model

There is limited research on how well a model can perform on predicting bitcoin't price. Since bitcoin is fairly new, its market cap size is small which could be easily manipulated, its difficult to find a benchmark for its price prediction. Currently, the researchers from Stanford claimed an accuracy of 0.574 with a ten minute time window using random forest. The detailed results are as following,

STATISTIC	10 SECOND GLM	10 MINUTE GLM	10 MINUTE RANDOM FOREST
Sensitivity (TPR)	0.5429	0.524	0.540
Specificity (TNR)	0.577	0.576	0.619
Precision (PPV)	0.574	0.551	0.581
Accuracy (ACC)	0.085	0.539	0.574

This 10 minutes result shall be used as the benchmark in this project.

Evaluation Metrics

The evaluation metrics will be used here is accuracy score. It is defined as,

$$accuracy = \frac{number\ of\ correctly\ predicted\ labels}{total\ number\ of\ labels}$$

Project Design

To solve this problem, one should identify it as a supervised learning problem. When we say we want to do time series prediction, we need to precisely define the range of time we want to predict. Here, our goal is that as data coming in every minute, the model could predict the price change for the next 10, 30, and 60 minutes. So this would be an online, real time model used to predict the price of bitcoin.

Although there were only two researches found for this particular topic, the general price prediction is a wide field that has a lot research papers. Particularly, for stock price prediction, ideas like strength index could be used here too. However, a lot of the techniques and indexes found in stock price prediction won't be used here due to the lacking of theorical prove, or the increasing complexity of the model. This can be saved for future explore.

Before we go and get some real data, the next thing is to pick out a few algorithms which have been used in research before. We will apply a set of chosen algorithms, compare their performance based on accurary score, and find out the best one. Based on the research, the models chosen are generalized linear model, support vector machine, random forest and recurrent neural network.

The next step is to find data for this project. Nearly every trading platform for bitcoin has API for aquiring trading data, which includes high, low, open, close, volume and datetime. However, since it is a rule of thumb that we always check and clean the data we got, we some basic rules to validate and preprocess our dataset. For the dataset, we can make API calls to Coinbase and get a half year's trading records, recorded every second. Although this dataset is of the higher quality data you can find on the interest, the following process will be applied,

- validate each field, make sure they have the right type. For example, all the price fields should be float or digit, the amount field should be int, and the datetime should be using IOS format;
- it is expected that not every second would have a corresponding record, that is why we want to use each minute to predict the price movement, not every second. To do this, it would need us to combine the data into a subset using one-minute window. If, for example, the platform was down for a given time period, we would smooth the blanks by averaging the begining and ending prices of the blanks.
- split the data into traing set and testing set.

After the data is cleaned and validated, we should repeat the following steps for each model,

- fit and train the model;
- grid search with cross validation on the model;
- find the best performance one using accuracy score;

After all the models being trained and testes, we can then choose the best one as our final model.