**CS230 Project Midterm Report:   
Crypto Exchange Price Prediction using Limit Order Book**

|  |  |  |
| --- | --- | --- |
| **Ben Gilboa**  **(SUID# - 06278930)** | **Tamal Biswas**  **(SUID# — 05107984)** | **Ashwin Selka Padmanabhan (SUID# — 06246676)** |
| **Stanford University**  **Winter 2018** | | |

**Abstract**

*This midterm report reviews the problem that the project is intended to investigate and solve. It explains the dataset format and acquisition process along with the initial model that we started to build and evaluate. The last section describes the remaining activities, methods and tasks that we plan to do for completing the project*

# Introduction

High frequency trading or Algo trading is gaining significant momentum in stock exchanges. In today’s market sizable portion of the daily traded volume is done by specialized companies using those techniques. In the elaborated stock market it is almost impossible for individuals not using heavy machinery and very fast access to data to gain any advantage as margins and arbitrages are closed in fraction of a second.

The rise of the crypto market and exchanges might reveal opportunities that are long gone in the stock market for small scale algorithmic trading.

In this project we explore and develop deep machine learning model that predict the future price of digital asset such as bitcoin. We intend to build a machine learning RNN (Recurrent Neural Network) that predict the future price of a tradable and volatile digital asset such as the Bitcoin. As input to the model we will use limit order book data along with other historic indications for demand as supply to develop our predictor. Although we chose a digital asset for this project, the principals and methods we develop are transferable to any asset that is tradable in an exchange.

# Prior work

TBD

# Dataset Characteristics and Acquisition

The data that is primarily used in for our predictor is the data from limit order book.

## Limit order book

This is a ledger maintained by the exchange of all limit orders that are pending. The order book has a sorted list of all bid and ask orders with the quantity and associated price for each order. It is usually presented graphically as accumulative plot of all bid and ask orders.

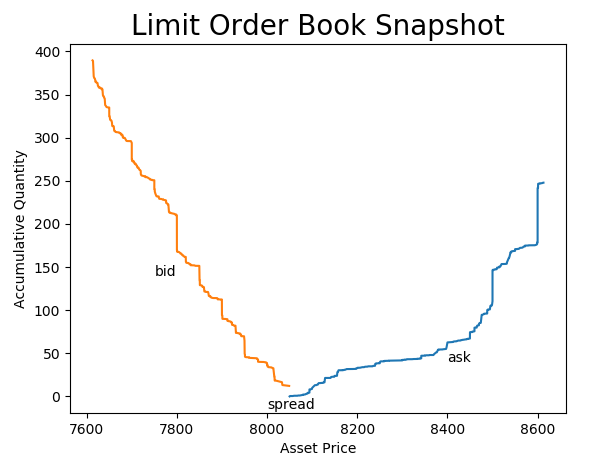


Figure 1: Limit Order Book Snapshot

The limit order book snapshot represent the demand and supply in the market in a certain point in time. In the above figure, it is clearly seen that the demand is “stronger”. There are much more buyers who are wiling to buy the asset for a price that is lower by 3% from last price than sellers who are willing to sell in a price that is higher by 3% than the last price. This might indicate that the price is about to increase. We look at the 500 highest bid orders and the 500 lowest ask orders in every snapshot of the order book.

## Bitcoin historical price

Apart from the limit order book we also look at the corresponding bitcoin price. This is basically the “last” price of a transaction at the same time when the order book was sampled. This data will serve both as features in the training examples as well as in generating the classifier for price increase or decrease. Consider a point in time ‘t0’ that corresponds to sample in our dataset ‘s0’. By considering certain number of examples (s-1, s-2, …, s-n) we get historical feature to the training set. By considering the samples (s1, s2, …, sn) we build our label for the classifier.

## Order history

In addition to the bitcoin price history and the limit order book history, we have data that represent the last 100 orders that were placed in the exchange. We plan to check if this data contribute to the prediction. The data contains the number of bids and asks and the accumulative quantities of each. For example, one training example contains 60 bids at total of 5 bitcoins and 40 asks at total of 3 bitcoin. The delta time of these last 100 orders is also known to us and might add value.

## Data acquisition

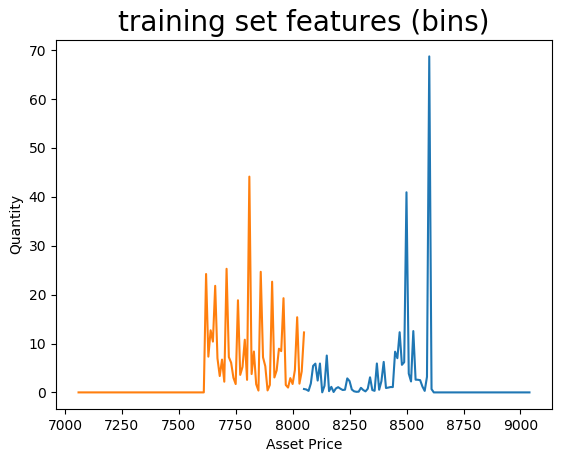
We obtain the above data by sampling the Bittrex exchange every 1 minute using the API it provides and storing the data. We obtained so far over 20,000 samples that represent 2 weeks of trading data. The data is not 100% consecutive as sometimes the software crash for several reasons.

# Initial model

As a starting point we use only the limit order book to predict future price increase or decrease and we use only one snapshot of the order book meaning that we predict a future change based on the current status without looking at the history.

Since every order in the book has 2 parameters (quantity and price) we cant use it as is. We apply a small modification to the data to extract a training example. We define “bins” of 10$ and we sum the quantities that relate to each bin. From 500 bid orders we create 100 bins that represent the last price down to last price minus 1000$. Figure 2 present a result of the bining process and a visual representation of one training example that we feed to the initial NN. It is easy to observe that this training example corresponds to the one used in figure 1. After bining the data, we end up with 200 features for every training example.

For the labels we have the last Bitcoin price that corresponds to every training example. We make it a classification problem by comparing the next value of the bitcoin (1 min into the future) to the current price. If the price increased the label is ‘1’ and if decreased or same it is ‘0’. This classification is very naïve and will not result in a successful trading strategy but it is good simple classification for initial design.

Figure 2: sample of one training example after structured in bins 

## NN Architecture

The objective of this initial phase is to find the correlation and validate the data from the order book as valid predictor. For that we use a fully connected network of the shape in figure 3.

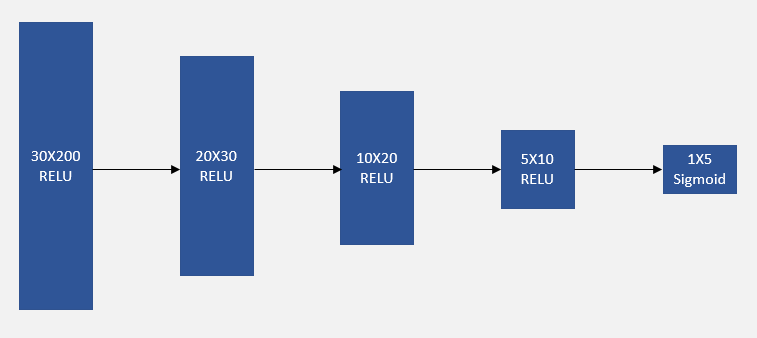


Figure 3: First NN Architecture

## Initial Results

We spent few iterations adding hidden layers and hidden units before deciding on this 5 layers model. We used about 21,000 training examples and shuffled them. Then we defined the training / dev sets as 80%/20%. For the labels we compared bitcoin price 10min into the future to the current price and derived the label.

After adjusting the learning rate and number of ephocs we achieved about 80% accuracy on the training set and 60% on the dev set. The 80% accuracy is very encouraging result for us but the high variance is clearly a concern. Trying to add regularization or using dropout did not help to reduce variance at all. It only increased the bias.