High Frequency Trading Price Prediction using LSTM Recursive Neural Networks

Karol Dzitkowski, Tomasz Kozicki

# Introduction

The aim of the project was to implement two neural networks using Keras python library. One of them is a LSTM RNN and the second is MLP. In implementation each of these operates on the same input feature vectors and output class variables.

# Data structure

The feature vector that is passed to the neural network consists of:

1. Time - time since previous event (ask or bid)
2. Price - price of last ask or bid
3. MidPrice - mean of highest bid and lowest ask
4. Volume – for last order
5. Side - ask or bid
6. AskPriceDiff - difference in price from last ask
7. BidPriceDiff - difference in price from last bid
8. AvgAskPrice - average ask price in order book
9. AvgBidPrice - average bid price in order book

Each instance of such vector results from processing by the Order Book consecutive messages from Message Book.

The response variable is a category number indicating whether the transaction price for a given stock has dropped, stayed the same or risen in the next moment.

# Program structure

As part of the project the following parts were implemented separately and integrated continuously:

## Test data providing

As a basic data that our program operates on we have chosen a NYSE OpenBook Ultra data set from 03/04/2013. The data for given stock symbol is parsed from large binary file and can be conveniently saved to the database and/or file. Then this message book is transformed into an order book message by message to capture the transaction price changes in each moment of time, as well as the history of other stock parameters for further building of the features.

## LSTM recurrent neural network

## Multi-Layer perceptron

## Feature selection

A method of performing feature selection with greedy forward selection using cross validation. From it gets testing data and neural network in parameters and returns a list of numbers of features to use.

## Crossvalidation

A method of using cross validation with N passes that can be used with different neural networks using different features. It returns a vector of error rates or a vector of vectors of predictions of test data. Data is split in proportion defined by a parameter.

## Performance comparison

A method of comparing performance of two neural networks. Used t-student metric to compare, and it’s pvalue to say if difference is significant. Used mean of error rates to say which one performed better. As input used a vector of error rates from cross validation of two neural networks on the same data.

# Test run results

## Description

Attached is text file “CIBA\_earlyResults.txt” which contains execution of the program for 19k events of the ‘AIG’ stock order book.

Both RNN and MLP report high accuracy from the beginning of iterations. This is very suspicious behaviour that can be explained by the data set characteristics in the next paragraph. As we can see the cross validation correctly divides the samples for training and testing and the feature selection chooses the best features. Due to no negligible differences in performance the program reports that RNN and MLP are not significantly different.

## Imbalanced data set

The problem that occurred during test phase turned out to be originating with data approach.

As the response value for our vector of features we have choosen indicator wheter the transaction price for a given stock has dropped, stayed the same or risen. This results in response variable being mainly dominated by the second case, because transactions on NYSE OpenBook occur much less often than we initialy expected, making below ~10% of overall entries.

Above leads to the poor learning performance of the neural networks. It’s just easier for them to assume no transaction price change since it’s the correct choice most of the time.