Machine Learning Project Report

# Definition

**Project Overview**

*Student provides a high-level overview of the project in layman’s terms. Background information such as the problem domain, the project origin, and related data sets or input data is given.*

The domain background for this project is the application of machine learning tools to predict prices for financial securities. For this project I took inspiration from a codepiece uploaded to machine learning platform Kaggle at <https://www.kaggle.com/kimy07/eurusd-15-minute-interval-price-prediction/data> and written by kimy07.

Our aim is to extend the analysis done in above code to a more granular (sub-second) dataset for EUR/USD with longer history (back to 2000). This should allow me to refine the model used by kimy07 and make better predictions.

From the literature on predicting financial security prices it appears the LSTM model (**L**ong **S**hort **T**erm **M**emory Network) can achieve better prediction capability compared to a simple Recurrent Neural Network and linear regression (http://colah.github.io/posts/2015-08-Understanding-LSTMs/) so we will use this approach. Other attempts at price prediction include bitcoin price prediction (<http://trap.ncirl.ie/2496/1/seanmcnally.pdf>), general evaluation of reinforcement learning algorithms on the foreign exchange market (<https://www.doc.ic.ac.uk/teaching/distinguished-projects/2015/j.cumming.pdf>), and stock market price prediction using LSTM specifically (<http://ieeexplore.ieee.org/document/7966019/>).

The dataset for this project is downloaded from <http://www.histdata.com/download-free-forex-data/?/ascii/tick-data-quotes> and consists of tick data - that is sub-second price quotes - for the EUR/USD currency pair since the beginning of 2000 up to the present day.

**Problem Statement**

*The problem which needs to be solved is clearly defined. A strategy for solving the problem, including discussion of the expected solution, has been made.*

The problem we will attempt to solve is price prediction based on historical tick data in the EUR/USD market.

This is a regression problem, as we try to predict future prices based on past information encoded in state variables, and those variables are continuous.

The inputs for the problem are the millisecond date-time stamp, the bid and ask prices, as well as features I will derive from these such as open, close, high, low, bid-offer spread.

**Metrics**

*Metrics used to measure performance of a model or result are clearly defined. Metrics are justified based on the characteristics of the problem.*

The success of any solution will be measured by looking at the error of predicted out-of-sample next prices.

Given that the problem is stated as a regression, I will use the mean absolute error and the mean squared error as evaluation metrics. I would hope to achieve a MAE of less than 0.0005.

# Analysis

**Data Exploration**

*If a dataset is present, features and calculated statistics relevant to the problem have been reported and discussed, along with a sampling of the data. In lieu of a dataset, a thorough description of the input space or input data has been made. Abnormalities or characteristics about the data or input that need to be addressed have been identified.*

The dataset is sourced from <http://www.histdata.com/download-free-forex-data/?/ascii/tick-data-quotes>. It contains for the EUR/USD currency pair tick by tick bid and ask price data for every day since 2000.

I will use date, bid and ask prices for this problem. Date is a millisecond datetime stamp, bid and ask are floating point numbers. Thus all three can be said to be continuous. The dataset has around 126 million rows. The outcome variable is the future price in the next period. Given I have many examples, I will split the data 80-10-10 into training, validation and testing.

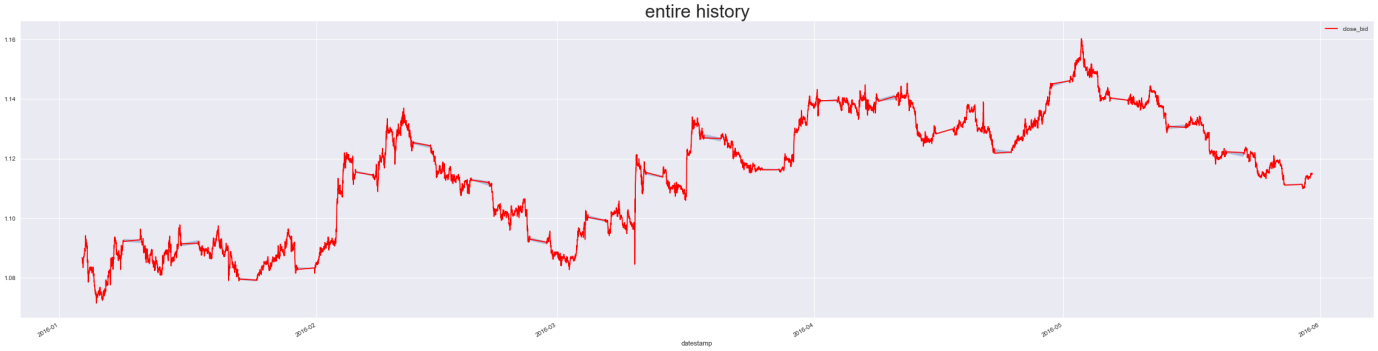
To prevent the algorithm from using future information to predict previous prices, I will attach actual future prices as target labels to each historical window I use for training the model. I will experiment with different window ranges as training sets, each being a multiple of 15 minutes.

Regarding class balances, there should not be any price groups in this dataset – it is unlikely that there is a “preferred price” at which the currency pair trades. To stationarize the data, i.e. to give no particular preference to absolute price levels, we will predict returns instead of absolute price values. This also leads to balanced classes, as both positive and negative return between ticks are approximately equally likely. However, there is a slightly fat tail of very negative returns, so I may have to deal with those.

**Exploratory Visualization**

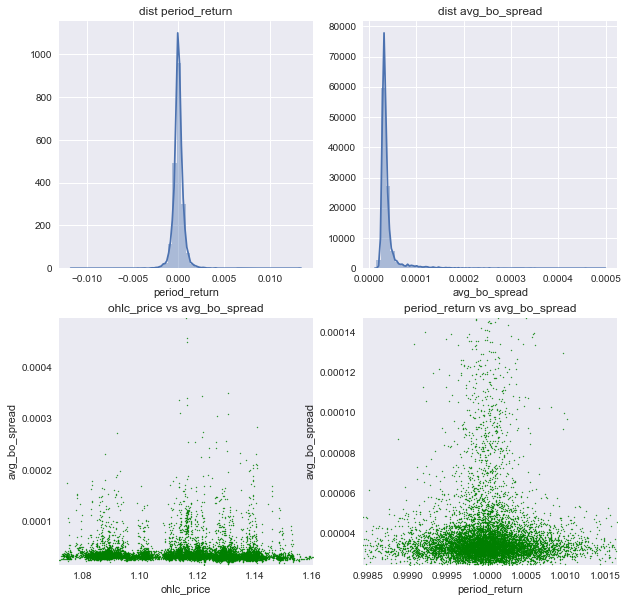
*A visualization has been provided that summarizes or extracts a relevant characteristic or feature about the dataset or input data with thorough discussion. Visual cues are clearly defined.*

Below the performance history of eurusd:



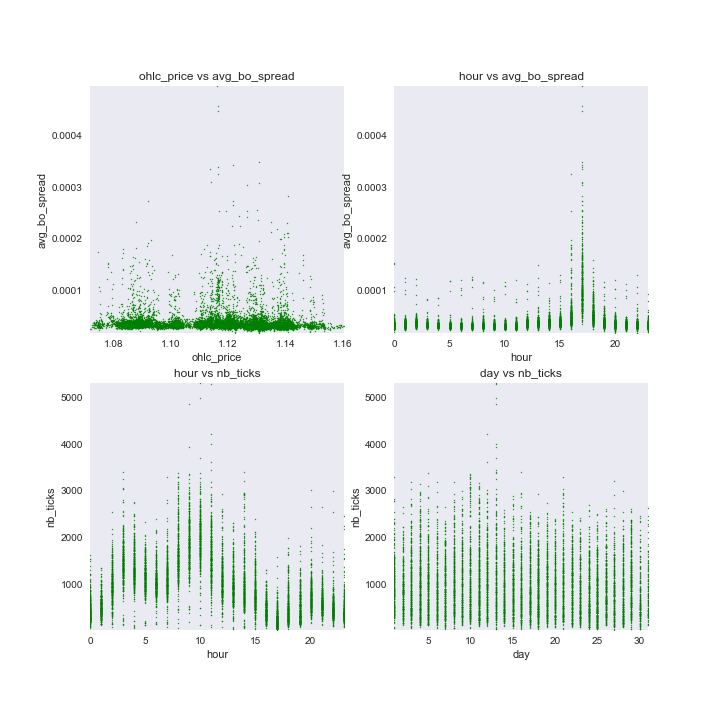
There are periods where the price doesn’t move (weekends), this should be accounted for when building the model.

After building features, distributions are this:

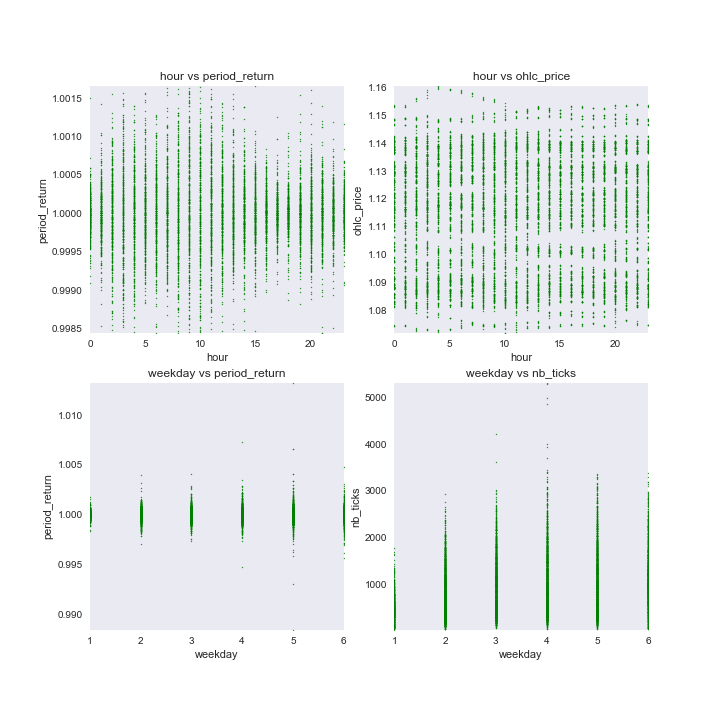


Prices are evenly distributed across the spectrum, returns are symmetrical around 0, the bid offer spread has a positive fat tail. This is expected as bid offer spreads will rise during market strain as liquidity dries up.

Next, we can see there is seasonality in the dataset, as the average bo spread changes with the hour, it seems highest and most spread out around 5 pm. It is also visible certain prices have higher bid offer spreads, although this could be down to market events coinciding with a certain price.

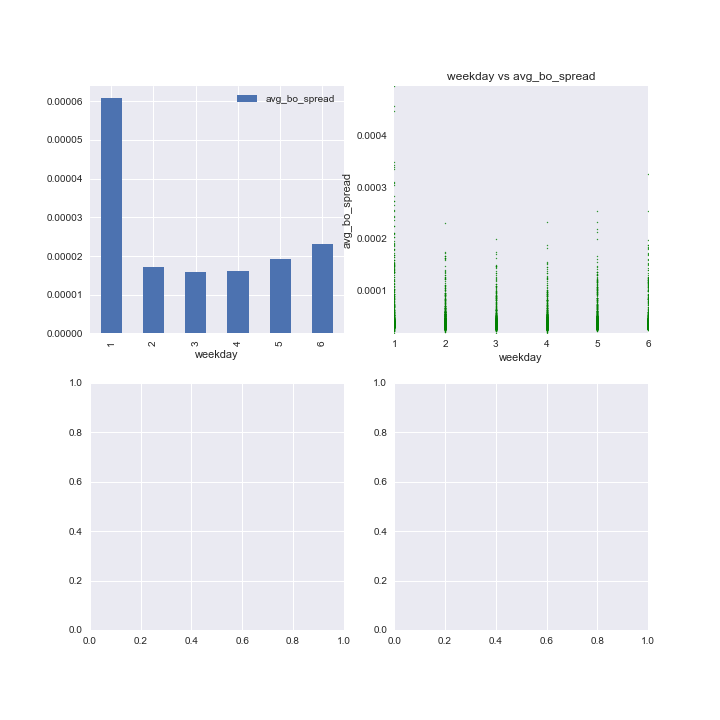


Further:



We can see here the Thursday has the largest variance in number of ticks, Monday clearly has a low absolute number of ticks. The period returns seems more narrow in the early hours and around 5pm, but there are no price patterns by the hour.

Finally, average bo spread is generally higher on Mondays:



As a result of above, I have incorporated the hour, day etc as features so that any model could use them to improve prediction accuracy.

**Algorithms and Techniques**

*Algorithms and techniques used in the project are thoroughly discussed and properly justified based on the characteristics of the problem.*

Algorithms we tried are linear regression and PCA for dimensionality reduction and feature engineering as a preprocessing step for the subsequent neural network.

Another approach to find useful features was a random forest regressor to help extract feature importances.

I used the LSTM to make the final prediction.

**Benchmark**

*Student clearly defines a benchmark result or threshold for comparing performances of solutions obtained.*

As benchmark the Kaggle code created by kimy, a contributor on machine learning platform Kaggle, will be used: <https://www.kaggle.com/kimy07/eurusd-15-minute-interval-price-prediction/notebook>. The benchmark model attempted price prediction with bid price data on EUR/USD with a fairly coarse 15 minute observation window

# Methodology

**Data Preprocessing**

*All preprocessing steps have been clearly documented. Abnormalities or characteristics about the data or input that needed to be addressed have been corrected. If no data preprocessing is necessary, it has been clearly justified.*

To attempt a solution, the dataset is loaded into a sql database. Next data cleanliness is checked, to account for nulls, positive and negative outliers and to create a final, clean dataset.

The dataset is enriched by creating additional features for my state space, such as:

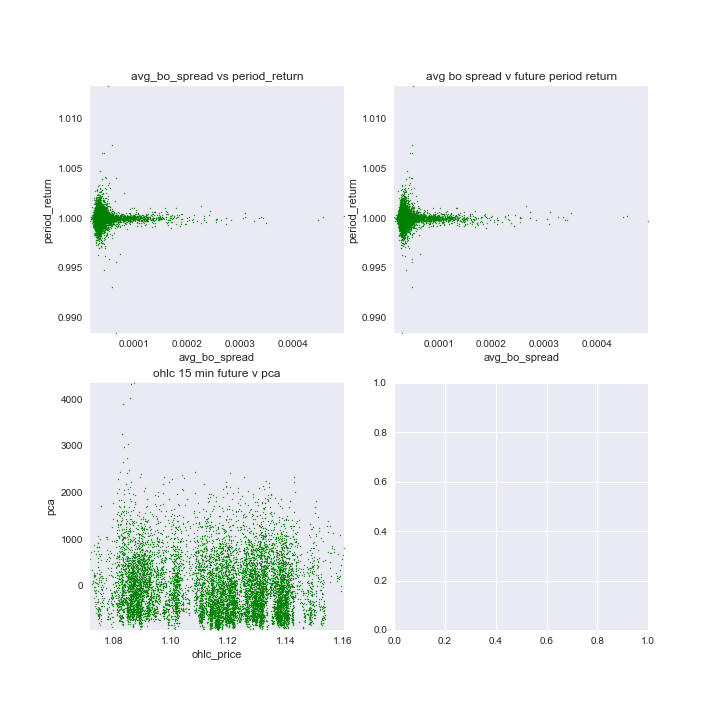
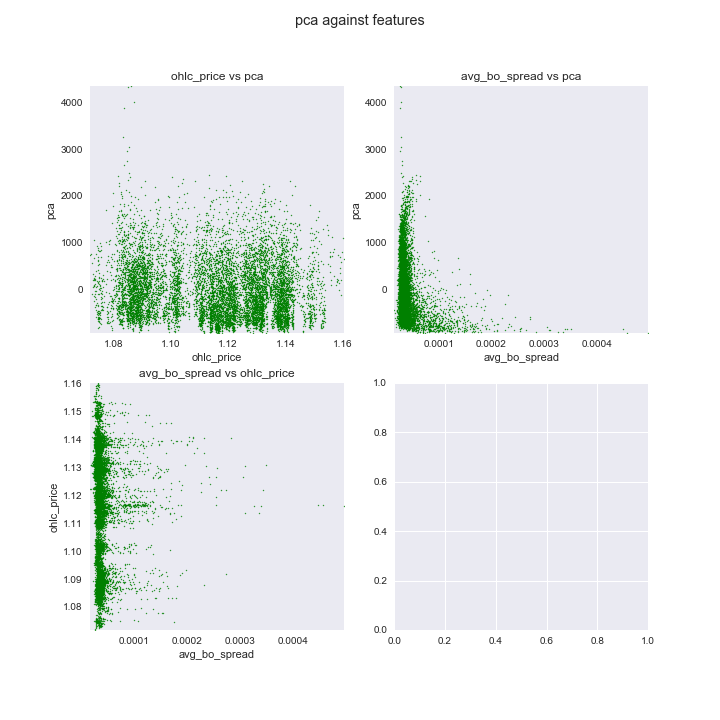
* returns
* bid-offer spreads
* period high, low, average, median, standard deviation

The tricky bit here is that prediction from the previous tick row is straightforward, as each feature can be used directly. However, using several tick rows to cover a 15 minute lookback window requires a decision how to use the features of each row.

One way would be to group the data into 15 minute intervals, with a number of aggregates to describe price behaviour during this time. We will initially attempt this, and only resort to the tick by tick prediction if error is too high.

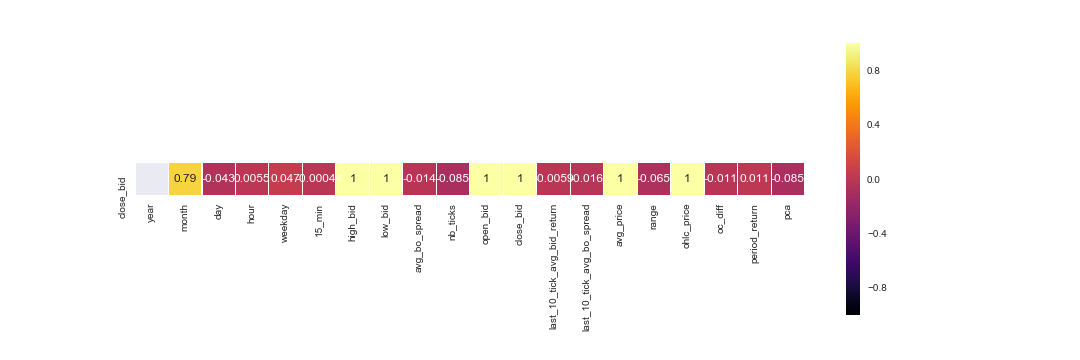
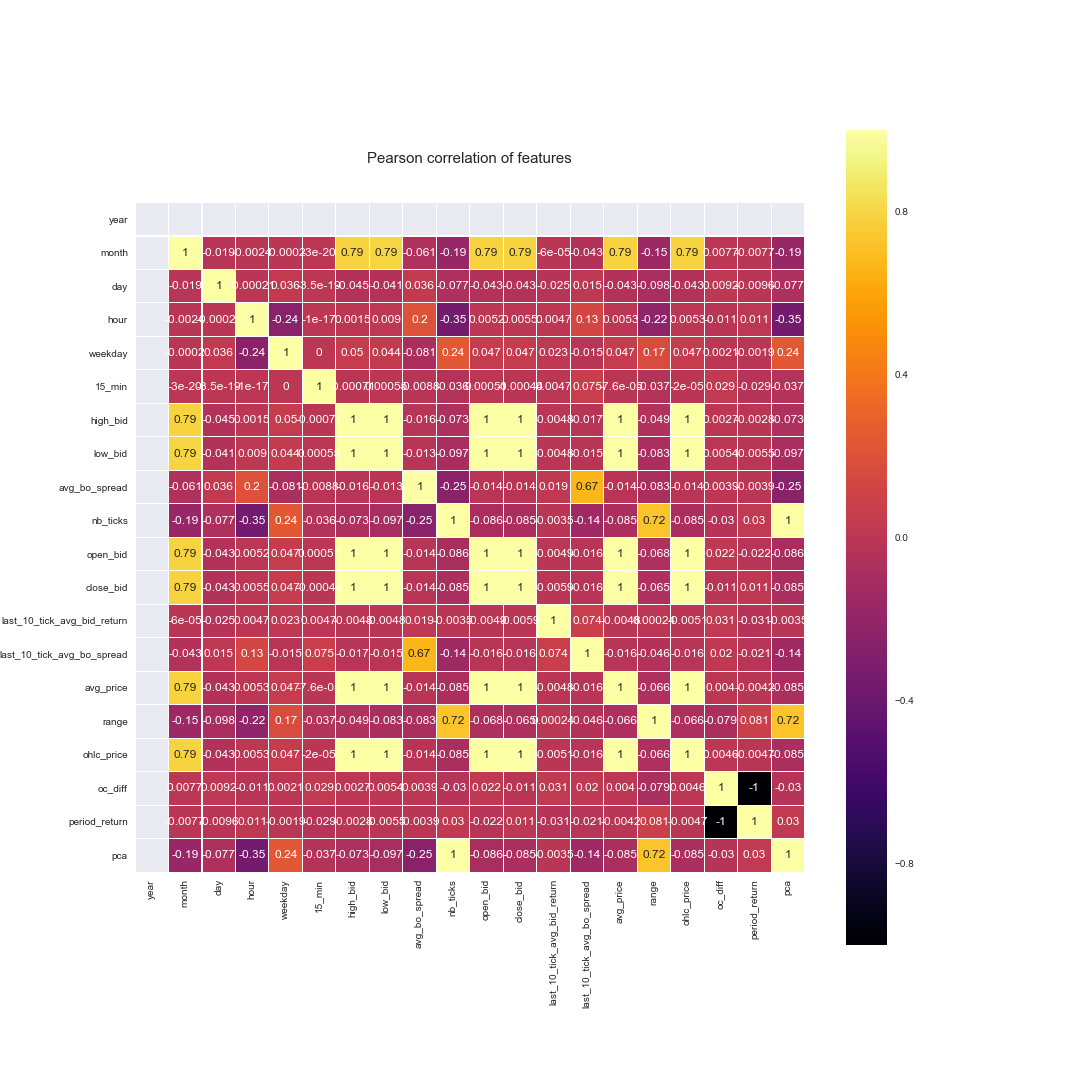
PCA:

PCA was used for feature engineering, trying out the first dimension of PCA as a feature:



The results suggest that PCA does not add extra structure to the dataset.

A feature correlation matrix gave the following output:



It can be seen that the best correlation to close is close itself, or else one of the other price composites for a 15 minute window. This is a trivial result, as the close price enters into those aggregates. Other observations are that the time and the number of ticks are correlated, which corroborates our initial findings that time provides structure to the data.

Random Forest:

A random forest regression suggests that the best feature to divide the dataset is the close price itself, by far:

Feature ranking:

close\_bid 0.926392

ohlc\_price 0.024249

avg\_price 0.022592

high\_bid 0.022400

low\_bid 0.003085

open\_bid 0.000675

last\_10\_tick\_avg\_bo\_spread 0.000088

last\_10\_tick\_avg\_bid\_return 0.000085

range 0.000064

avg\_bo\_spread 0.000060

pca 0.000056

nb\_ticks 0.000053

oc\_diff 0.000041

period\_return 0.000039

hour 0.000037

day 0.000037

weekday 0.000021

15\_min 0.000017

month 0.000009

Linear Regression:

Running linear regression, this is the result:

oc\_diff -0.204211

nb\_ticks -0.096250

low\_bid -0.017280

last\_10\_tick\_avg\_bo\_spread -0.005564

avg\_bo\_spread -0.003196

avg\_price -0.003091

last\_10\_tick\_avg\_bid\_return -0.002060

weekday -0.000060

year 0.000000

month 0.000062

15\_min 0.000098

day 0.000105

hour 0.000380

high\_bid 0.012807

range 0.031856

pca 0.096251

period\_return 0.185722

ohlc\_price 0.199248

open\_bid 0.300989

close\_bid 0.505757

The main factors are the open close difference, the close bid, the open bid and the period return. PCA and the number of ticks get about 10% each. The other factors are marginal.

Evaluating prediction error on linear regression gives us this:

**Implementation**

*The process for which metrics, algorithms, and techniques were implemented with the given datasets or input data has been thoroughly documented. Complications that occurred during the coding process are discussed.*

Coding: any coding issues? The main process issues was how to keep track of a long jupyter notebook. For that I added a table of contents plugin. To plot many different correlations, I wrote functions that would plot into a matplotlib subplot and x and y numpy arrays I gave them.

**Refinement**

*The process of improving upon the algorithms and techniques used is clearly documented. Both the initial and final solutions are reported, along with intermediate solutions, if necessary.*

The initial solution was … In the final solution I found it would be better to predict the return class (positive, negative or zero), instead of the exact price. However, my models seems to have a bug in that it always predicts the same thing. Given that in my project proposal, binary classification was out of scope and I proposed this as a regression problem, I did not investigate further.

# Results

**Model Evaluation and Validation**

The final model’s qualities — such as parameters — are evaluated in detail. Some type of analysis is used to validate the robustness of the model’s solution.

Linear Regression:

['mse train all feature: ', 4.4530569e-07]

['mse test all feature: ', 6.5385599e-08]

['mae train all feature: ', 0.00042667281]

['mae test all feature: ', 0.0002031199]

['mean avg bo spread: ', 3.9868658581951545e-05]

['how often sign of price change is same: ', 0.53398058252427183]

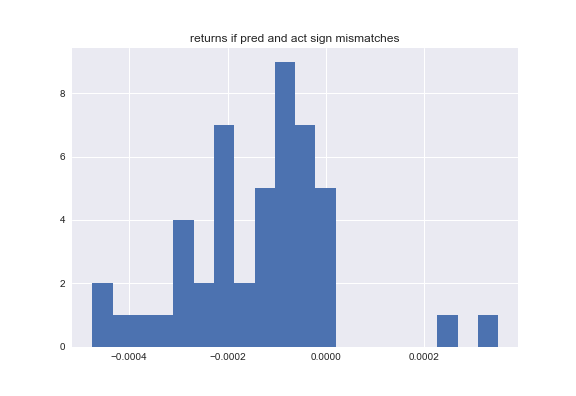
['if same sign, how often is actual better than 0.1 percent in both directions: ', 0.0]

['if same sign, how often is actual better than predicted in both directions: ', 0.96363636363636362]

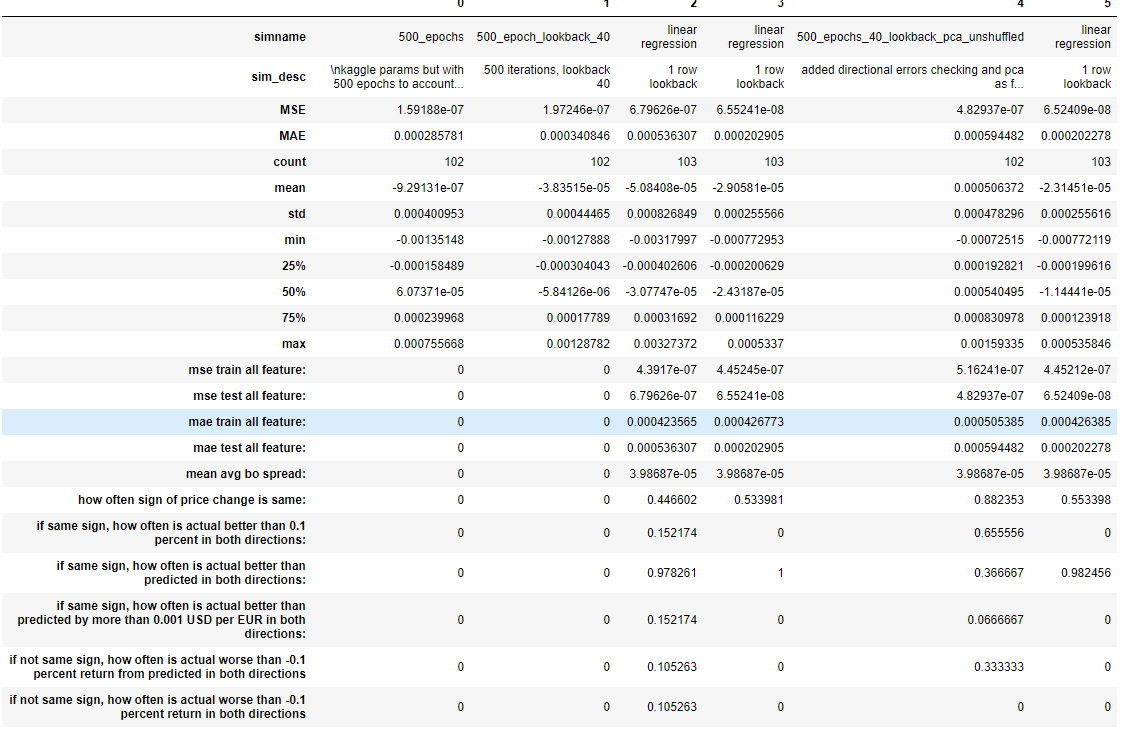
['if same sign, how often is actual better than predicted by more than 0.001 USD per EUR in both directions: ', 0.0]

['if not same sign, how often is actual worse than -0.1 percent return from predicted in both directions', 0.0]

['if not same sign, how often is actual worse than -0.1 percent return in both directions', 0.0]



Here a comparison of the errors of the different simulations:



I finally check the performance of the LSTM model, to see if it improves on any of the previous:

There seems to be no improvement.

I also tried the LSTM model using binary classification, but it gives me the same prediction for every class, so I am probably doing something wrong.

**Justification**

*The final results are compared to the benchmark result or threshold with some type of statistical analysis. Justification is made as to whether the final model and solution is significant enough to have adequately solved the problem.*

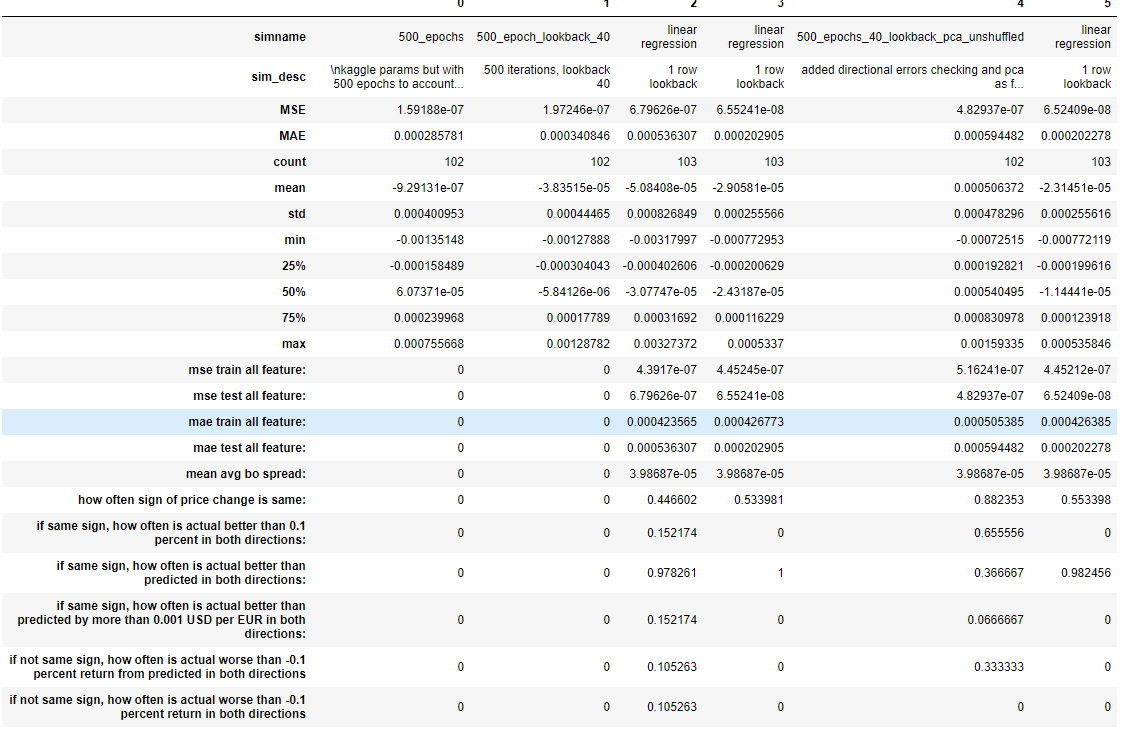
The final solution does not present a significant improvement on the initial solution. This is possibly down to a lack of explanatory parameters in the price history only – one might have to take into account other financial variables such as news.

# Conclusion

**Free-Form Visualization**

A visualization has been provided that emphasizes an important quality about the project with thorough discussion. Visual cues are clearly defined.

Show the error comparison here between the various methods:



**Reflection**

Student adequately summarizes the end-to-end problem solution and discusses one or two particular aspects of the project they found interesting or difficult.

It was very interesting to investigate different properties of the dataset. It was difficult to get down the prediction error, possibly because I am solving for the wrong measure. A much more practical measure would be to predict whether I get the direction of the next price movement right, rather than its value. This would give less importance to small errors in value.

**Improvement**

Discussion is made as to how one aspect of the implementation could be improved. Potential solutions resulting from these improvements are considered and compared/contrasted to the current solution.

# Quality

**Presentation**

Project report follows a well-organized structure and would be readily understood by its intended audience. Each section is written in a clear, concise and specific manner. Few grammatical and spelling mistakes are present. All resources used to complete the project are cited and referenced.

**Functionality**

Code is formatted neatly with comments that effectively explain complex implementations. Output produces similar results and solutions as to those discussed in the project.