Summary for 1min approach

Minda Yang
minda.yang@columbia.edu

Notice: Please refer to the README.txt for the summary of the codes.
Codes available at: https://github.com/mindaxiaoke/temporary

1. Data cleaning:

Logics:

The logic for data cleaning is to assume a distribution of the daily return. One simple approach is to plot the histogram of the daily return and exclude the extreme returns that fall too far away from the majority distribution. One sample was classified as noisy and excluded from the data.

Notice:

Given the dynamics of the security market, sometimes we need to validate the data after cleaning with another data source to verify that we are not excluding the real data.

Future:

Collect the 1 min data for this security from another source and verify the correctness of the data cleaning.

Results:

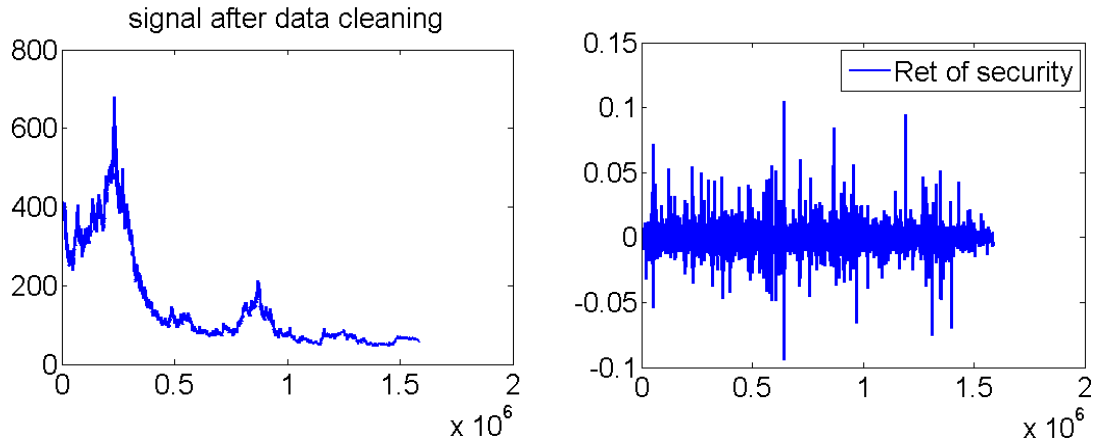


Figure 1. The minute bar price for this security (left) and the return (right) after data cleaning.

2. Data preparation

Logics:

i. First I filled in the missing minutes assuming the prices were unchanged. But if the gap is larger than 30 minutes, then I regard the data to be separate and there's no filling in.

ii. All the continuous trading minutes are taken as one trading block (notice that one

trading block could contain minutes from two days, for example, 11:00pm of day 1 to 8pm of day 2).

iii. I excluded the trading blocks with less than 90 minutes, assuming these blocks should be further examined.

iv. For the input features, I'm using previous 30 minutes of security return per minute (30-dimensional) and the normalized high, low and average price of the security up till the current minute for this trading block. So the total input dimension is 33.

v. The prediction target is the return of the next 4 minutes. For example, if we were looking at the first 35 minutes of one trading block, we would be using the 1-30 minutes return (the return for the first minute is 0) to predict the return of 31-35 minutes. (5 states: HH, H, F, L, LL)

vi. For testing the pnl, I'm using a very simple model right now. I assume I have 500 dollars and invest 100 dollars (long or short) per minute according to the prediction being positive or negative.

Notice:

The boundaries for the five states of return were set to the 20,40,60 and 80 percentiles of the distribution of return. The purpose of this is to ensure an even distribution of the returns in the five states. But keep in mind that the distribution is sometimes subject to the trend of the security, for example, the distribution/percentiles could be different when the security price is trending up vs. trending down.

Future:

With better understanding of the context for this security, it's possible to further examine the data (especially the trading blocks with less number of minutes) and include more data for testing the deep learning approach.

3. Deep learning

Logics:

I tested deep neural networks with 1-5 hidden layers. For the model with 4 hidden layers I also tested different number of units per layer.

To avoid over-training, I used both **early stopping** (stops the training of the network when the performance ceases to increase on the validation set) and **L2 regularization** (apply penalty for the sum of squared weights of all the units).

Notice:

As the number of hidden layers increases with fixed number of units, we are introducing more parameters into the deep neural networks. The approach I tested here is to first find the optimal number of hidden layers and then finds the optimal number of units per layer. Sometimes it requires a subtler parameter search (such as grid search) to find the optimal set of parameters.

Results:

The networks were trained on the first 8 folds of the data and validated on the 9^{th} fold using early stopping. The best performance for the parameters tested was achieved by

using a neural network with 4 hidden layers and 20 units per hidden layer.

The predicted return is converted by multiplying the posterior probability of the output layer with the boundaries of the return. For example, if the posterior probability for the output layer is [0.1 0.2 0.3 0.3 0.1] and the boundaries are [-0.001 -0.0005 0 0.0005 0.001], then the predicted return is set as 0.0005.

The accuracy for predicting up/down of the return:

Table 1. The accuracy for prediction of up/down

Train set	68.45%
Valid set	65.34%
Test set	64.61%

The correlation between the predicted return and the real return:

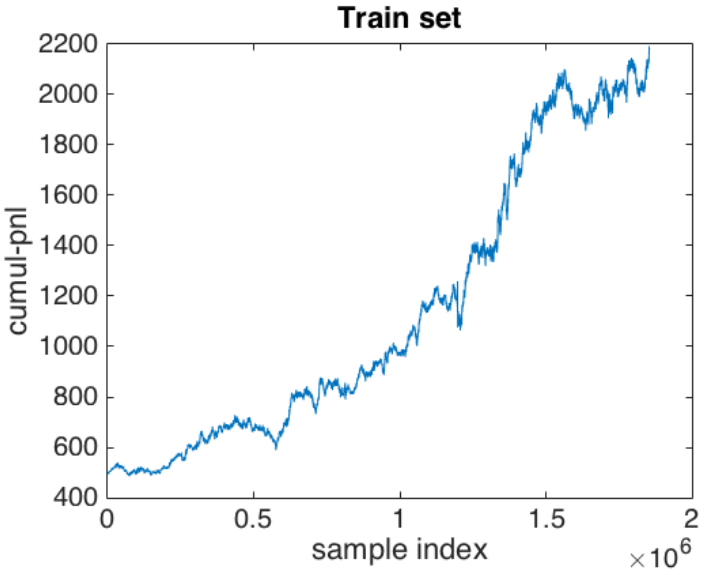
Table 2. The correlation between predicted return and real return

Train set	0.0036
Valid set	0.0008
Test set	0.0008

Notice that to test the significance of the correlation value we can perform some statistical test (for example, generate random predictions and calculate p-value).

To simulate the real performance, I used a very simple approach:

Assume we start with 500 dollars and for each minute we get into long/short positions according to the prediction being positive/negative. Then the pnl we get will be the real return of the 4-minute period.



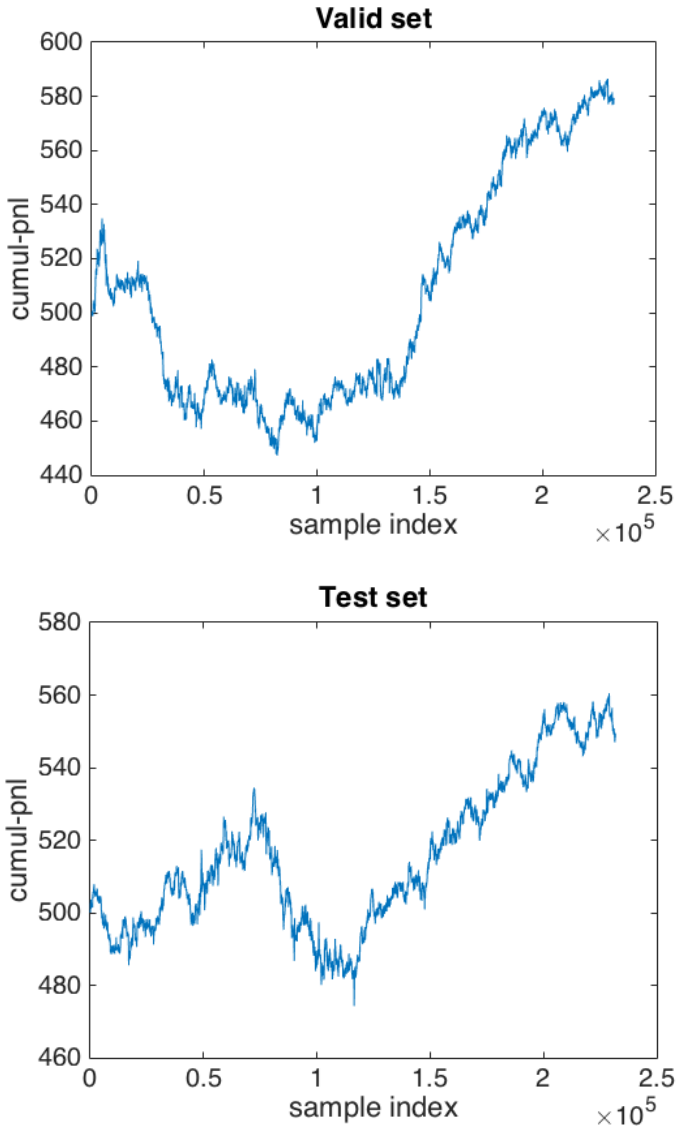


Figure 2. The cumulative performance for the training set (top), validation set (middle) and test set (bottom).

4. Conclusion

In this approach I tested the performance of deep learning models on predicting the return of one security. This approach applies early stopping and L2 regularization to constrain the over-fitting in the training set. This is my first attempt to make an intraday trading signal so a lot of things could be further improved.

Future expansion of this project could be a grid search of the optimal parameters, another regularization method such as dropout, and/or making better trading signals/positions based on the prediction.