Stock Prediction Model

Lu Tao

March 14, 2017

I. Introduction

In this project, I am trying to use machine learning techniques to make stock prediction. I find that the data set 7 is the financial data with daily close prices in US market. And, actually I am working on stock prediction project in chinese stock markets and I am trying to make better models.

The stock prediction problem is a complicated but interesting problem, it is highly challenging problem because even human can not handle the problem. And on the other hand, the market is dynamic and highly efficient, whenever there is something profitable chances, investors will practice arbitrage and make the strategy we found disappeared.

Potentially, we can try to introduce more data more or less related to financial market into our model. Because the neural network model have a high capacity and tolerence to additional dimensions. For example, from the perspective of information system, we can leverage the tremendous socail media data into the model. And also, we can try to filter some finacial events by extract infromation from news or analysts reports. From the perspective of economics, we can try to get economics information from both micro side or marcro side to predict the firm future performance in a more plausible way.

II. Data

In the project, I use financial data from chinese stock market from 2000 to 2016. The data is basically composed by two parts, the technical part and the fundamental part. The technical part, we have stocks' daily open, close, high, low prices and free turn overs, which cover all the information that delivered by candlestick chart. The fundamental part, we have stocks' price-earning ratios, price-bookvalue ratios, equity values, and industry sector numbers.

A. Chinese Stock Market

I will briefly describe the stock market in China. There are two stock exchanges in china, Shanghai and Shenzhen stock exchange. There are more than 2,900 stocks have been traded, and in my database, I have 2,934 stocks in 4,020 days. And because of these are new stocks newly issued in the time period of my dataset, so I have totally 6,931,966 rows in my dataset.

B. Potential Increasement

Further, we can process the input data more carefully. Although, we are looking forward that the neural network can automatically generate features for us, but actually the pre-processing will increase the model perfomance a lot. Especially, we can construct more useful features as input by using other statistical techniques, such as some sorts of PCAs. And also we can use some time series analysis to extract different features.

III. Methods

In this project, I use neural network to train the model. I will simply tell the model specification.

A. Data Pre-processing

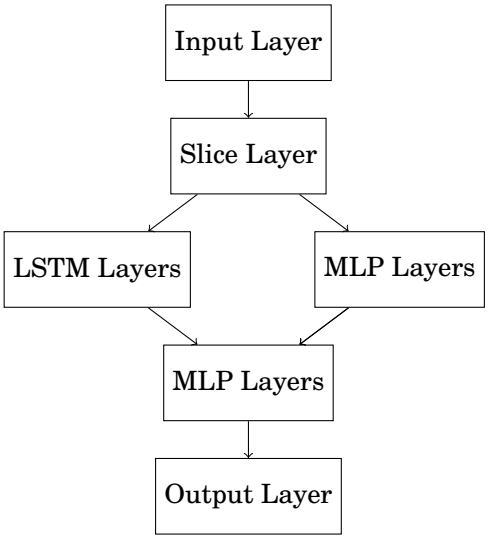
Before training, the input data should be pre-processed in the following steps.

- For time series data, I use 200 days technical history data as input
- For all the imput data, I normalized all none dummy variables
- For output label, I classify the samples into 3 categories based on stocks' performances in future 10 days: The wining group, the equal group, and the losing group. (I also tried different time periods)

Finally, I get totally 7 varibles in 200 days before, and totally have 1400 input dimensions. And also I have 37 categorical input dimensions.

B. Network Structure

The network structure is established as the following figure. After the input layer, the two parts of data, the time series data and categorical data. The time series data is fed into LSTM RNN network, and the categorical data is fed into MLP network. And the sub-network outputs are combined together and fed into another MLP network.



C. Training Issues

First, I use some regularizations trying to overcome the overfitting issue. As we all know, for a high dimensional neural network model, the overfitting problem is an important issue, especially in a problem that the distributions can be dynamic over time. I use both L1 and L2 loss function in the model to srinkage

and implement some batchnorm layers and dropout layers in the model, which is proved to be useful to alleviate the overfitting.

The updating function I choose is Nesterov momentum, which is better than SGD in dealing with a problem which is a high dimensional optimization problem. There are also some other updating functions I can also test. Given the updating function, the learning rate is kind of important in the optimization procedure, basically, we can set the learning rate higher in the first several iterations and tune the rate smaller and smaller in following steps to balance the efficiency and the accuracy.

I use GPUs which is already highly efficient because of the parallel computation to training the stock prediction model, and usually the training period take about several hours once. And If we just use CPUs to train the model, it may takes about ten times more time than GPUs.

IV. Rudimentary Results

In this section, I will show some rudimentary results in my project. I am still on the progress of training the model, so I list the results of one of the models I trained. This model I use two layers of LSTM with 64 filters for the time series input, two layers of MLP with 32 filters for catagorical data, and two layers of MLP with 128 filters before output layer.

First, Figure 1 is the loss curve. As we can see, loss of training data is always decreasing and the validation loss decreases first and after about 250 to 300 iterations, the validation loss starts to increase which means we should stop training the model at the point.

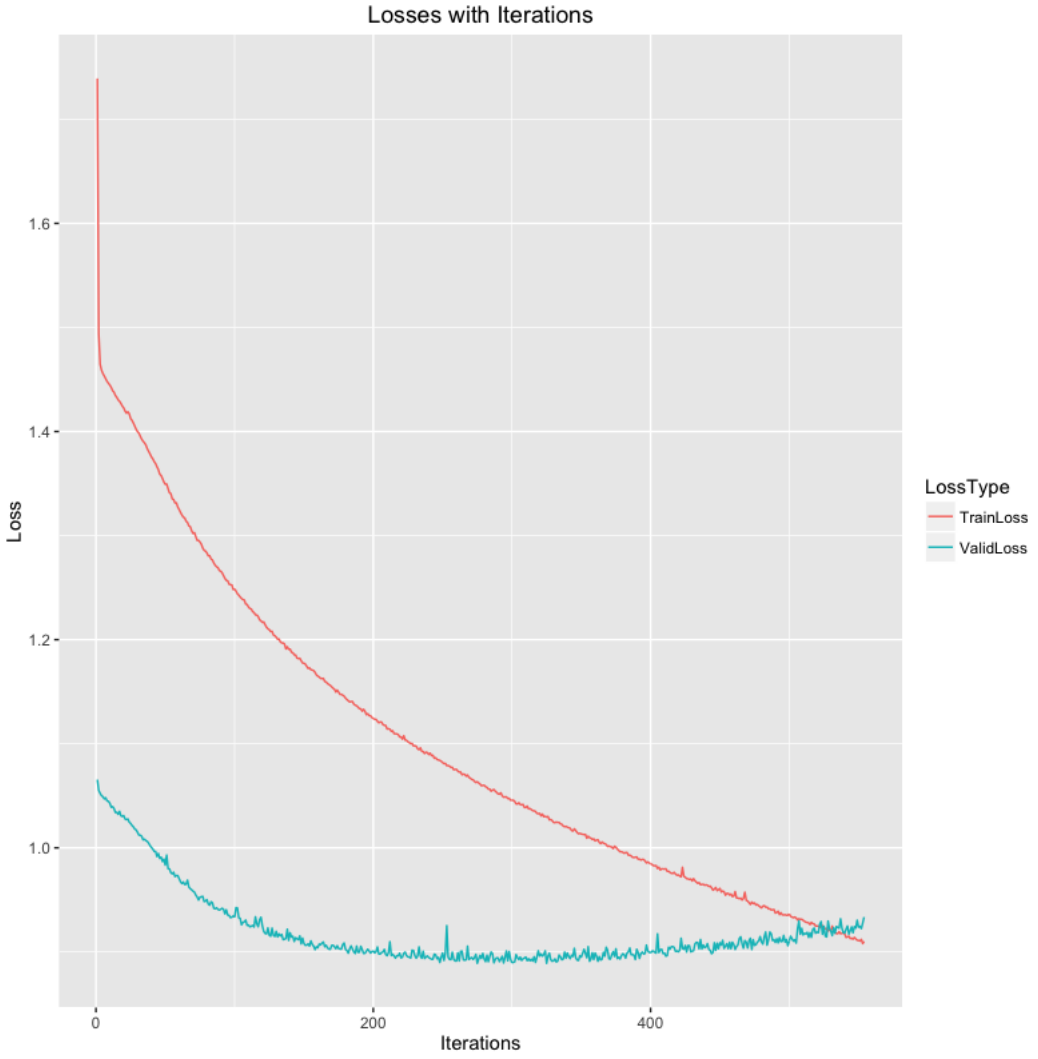


Figure 1: Loss Curve

Then, Figure 2 is the accuracy curve. The accucy increases through training, and after about 300 iterations, the accuracy of the validation data stop increasing and has an accuracy at about 58%. Consider we have an output labels with

3 categories, the 58% is kind of good prediction.

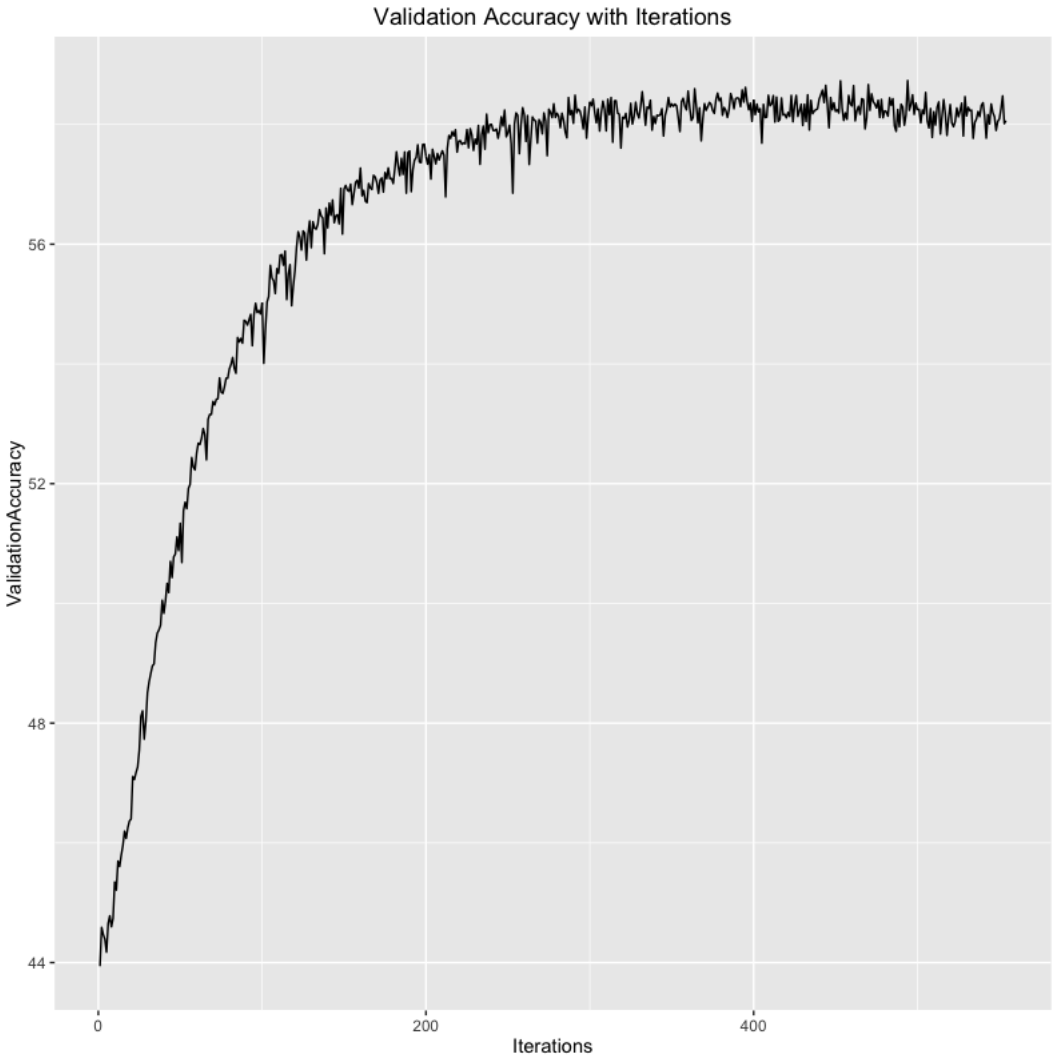


Figure 2: Accuracy Curve

Finally, Figure 3 is the ROC curve. And the AUC number is 0.7348.

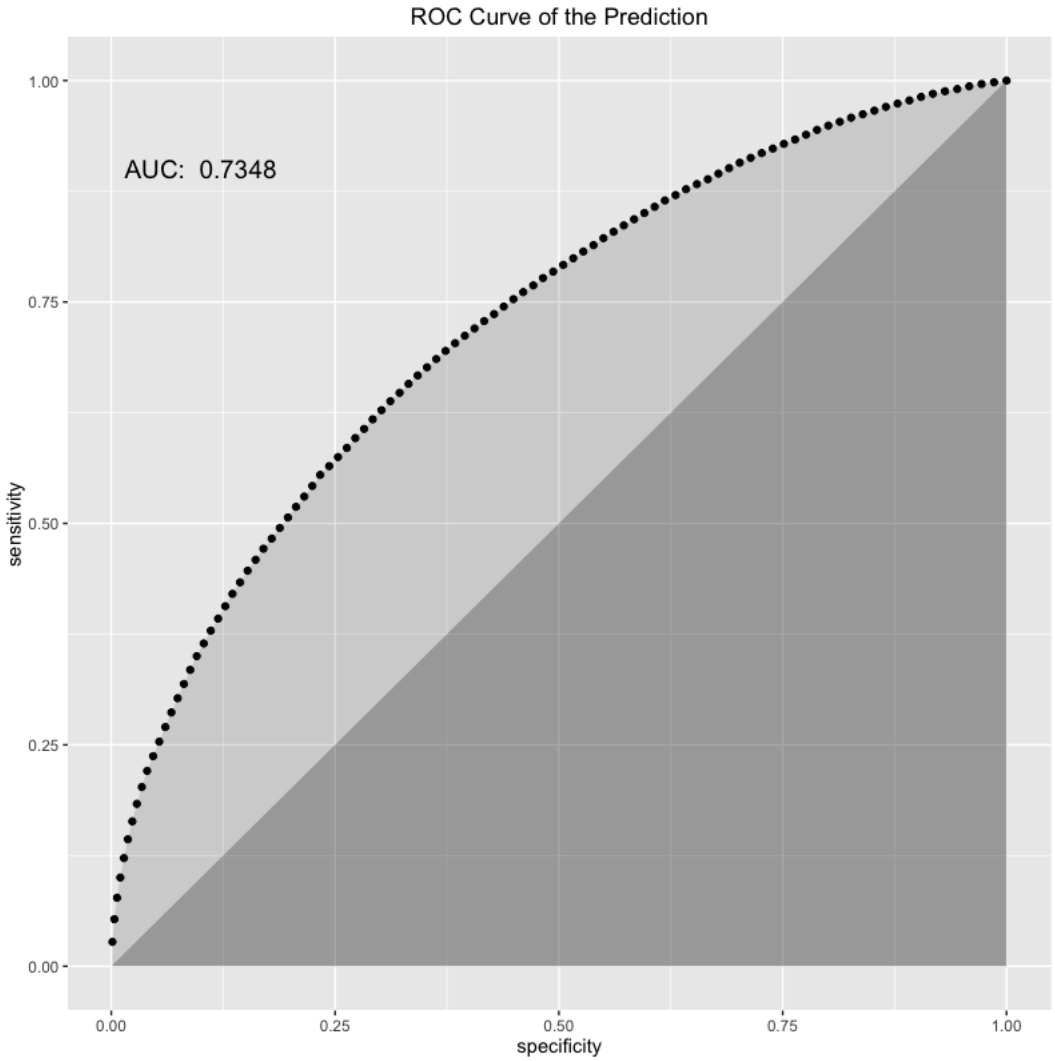


Figure 3: ROC Curve

V. Conclusions and Remarks

From now on, I just complete a rudimentary model which has a slightly good prediction model, which is already a progress in financial predictions. And I will continue working on the project to test more techniques and inputs for the model. There are something we can do further:

- Introducing more data into the input set.
- Pre-process the input data to extract more ex-ante features.
- Changing the network stucture and tune the hyper parameters.
- Do more scientific back tests on the trading strategy.