Boston University

Predicting Stock Return

January 31, 2020

Author

Cong Cao

Contents

1. Introduction

In the financial market, stock return and direction predictions are crucial for making investment decisions. Machine learning techniques are considered to be one of the most efficient statistical approaches in data predictions, and has been widely used in asset management firm to generate alphas. In this project, we tried to apply multiple machine learning methods to evaluate to forecast companies' the one-month stock returns and the next month's grow-fall directions.

In the original dataset, various types of financial and accounting features, such as percentage, currency and ratio, are covered to make predictions. The next few paragraphs of the paper is organized in the following way. First we described the data set given to us. Next, we conduct model and variable selection on the original data set and altered data set. In Section 4 we conducted hyperparameter tuning for chosen model. Next, we discussed the meaning of some of the most important features in our model and conclude our research.

2. Data Description

The raw data that consists of monthly stock returns of a company, S&P 500 return and volatility, sentiment measures, and several firm fundamentals from December 2014 to October 2018. We adjusted the data table in the following manner so that it is more suitable for predicting stock returns:

1. We ordered data by company ID and date.
2. We shifted all columns labeled end of month down by one row (Lag 1) to be consistent with start of the month data.
3. We transformed the industry classification to ordinal data

The justification for adding lag 1 to end of month data is that we believe it is reasonable to treat the month end data of last month as the start of month data of the current month, and this subsequently provides us with more varieties of data to train the machine learning model. The rationale of transforming industry classification is that we believe this variable is a significant factor in predicting stock returns, and should be added to the training list.

3. Model and Variable Selection

3.1 Stock Return Forecast

For selecting models of stock return forecast, we began with simple models such as LASSO and ridge regression and evaluate the model performance using MSE and R-square. We applied elastic net regression, which linearly combines the L1 and L2 penalties of the lasso and ridge methods. The formula is given as

$$\min_{\beta_0,\beta} \frac{1}{N} \sum_{i=1}^{N} w_i l\left(y_i, \beta_0 + \beta^T x_i\right) + \lambda \left[(1-\alpha)\|\beta\|_2^2/2 + \alpha\|\beta\|_1\right]$$

where $l(y, \eta)$ is the negative log-likelihood contribution for observation i. The elastic-net penalty is controlled by α, and bridges the gap between lasso ($\alpha = 1$) and ridge ($\alpha = 0$). λ controls the overall strength of the penalty.

The model is cross-validated to identify optimal parameters that minimizes MSE. The tuning parameters include (α) and (λ). The prediction result of the model has a R^2 of about -22%.

Our second attempt is to use XGBoost. Boosting build an ensemble of shallow and weak successive trees with each tree learning and improving on the previous. The rationale of using boosting is that it often provides high predictive accuracy, with multiple hyperparameter that make the function fit flexible. We test the boosting machine with rounds = 100, training month set to first 36 months and testing months set to last 10 months, and other parameters set to default to evaluate its accuracy.

Models	XGboost	Elastic-net regression
R2	-4.5%	-22.42%

Table 3.1: Model accuracy for 36 to 10 month splitting

3.2 Direction Forecast

3.2.1 Variable selection

We adopted principal component analysis, R^2 and information coefficient (IC) approaches to reduce the data dimensions and select variables. By using PCA, the number of features

were reduced to 54, which captures 99% of cross sectional variation of the original data. For the R^2 and information coefficients approaches, the data was grouped in the same month first, and then for each month, we computed the R^2 of regressions and correlations between each feature and return direction, ranked the R^2 and information coefficients, and finally selected 70 features for our R-squared and 66 features for the IC. Finally, we prepared the original, PCA, R^2 and IC datasets as our inputs to fit models.

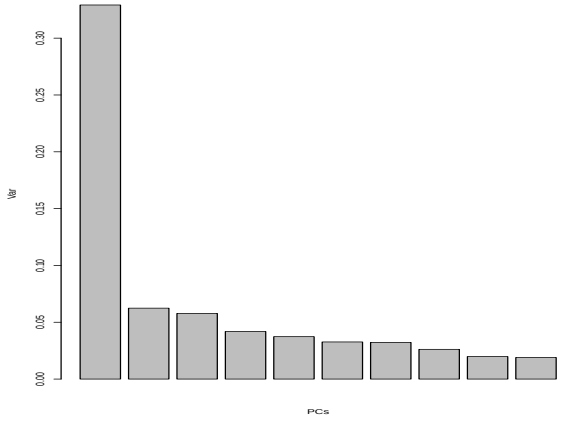

Figure 3.1: First 10 PCAs

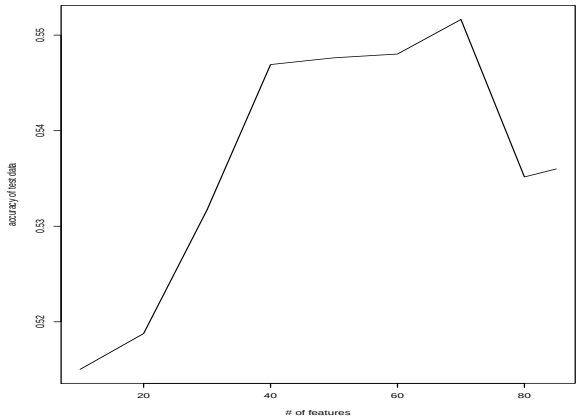

Figure 3.2: R^2 vs Features

3.2.2 Model Selection

We have tried several models (SVM, KNN, Random Forest, Neural Networks, XGBoost,etc.) that may work well for classification. Among them, XGBoost showed the best performance.

First we conducted the analysis on the original dataset. We split our dataset into groups of six months, with first five months in each group as training set and last month as testing set.

Models	SVM	Random Forest	GBM	Neural Networks	XGBoost
Train Accuracy	70.83%	100%	52.39%	60.85%	47.61%
Test Accuracy	48.86%	49.26%	49.23%	52.35%	48.60%

Table 3.2: Model accuracy for 5 to 1 month splitting

Due to small size of dataset, most of the models perform below 50% of test accuracy. Besides that, high train accuracy implies the strong overfitting of the models; more information might need. We also notice that neural networks model requires more time on training and predicting the data comparing with the others, and XGBoost model is the fastest.

Then, we use the whole dataset with first 36 months as training set and last 10 months as testing test. The output is summarized in the following table.

Models	SVM	Random Forest	GBM	Neural Networks	XGBoost
Train Accuracy	69.46%	100%	52.39%	60.94%	66.44%
Test Accuracy	51.57%	49.36%	52.55%	54.58%	56.06%

Table 3.3: Model accuracy for 36 to 10 months splitting

The results showed XGBoost perform the best among the others with highest test accuracy, reasonable train accuracy and least training time.

XGBoost stands for extreme gradient boosting. It is an implementation of gradient boosted decision trees designed for speed and performance. Gradient boosting is an approach where new models are created that predict the residuals or errors of prior models and then added together to make the final prediction. It uses a gradient descent algorithm to minimize the loss when adding new models.

Finally, we trained all the models on the R^2, IC and PCA datasets. The results are too long to present here. The models trained on original dataset performed the best among all. This may due to the fact that model selection method may lose information contained in the original dataset, which can reduce the test accuracy.

As a result, we choose to use the original dataset with XGBoost in the following analysis.

4. Hyperparameter Tuning

We use tree booster for prediction. To achieve a better performance, we apply the grid search method to tune the parameters. The parameters to be tuned are:

- nrounds: Maximum number of iterations
- max_depth: Maximum depth of a tree
- eta: Step size shrinkage used in update to prevents over-fitting, this can be viewed as learning rate

Within the training data set, we apply the time series cross validation to find the optimal set of parameters.

This method prevents the look-ahead bias. If we use the traditional cross-validation and randomly divide the training set into K folds, train the model on $K-1$ folds, then these $K-1$ folds may contain future information about the K-th fold since we ignore the time structure. Moreover, the growing size of training data allows us to accumulate more information to train our model.

It is conducted in the following way:

1. Extract the first 36 months from the adjusted data set as the training set. We will for now ignore the last 10 months of the data set.
2. Select month end returns (RETMONTH_end) as training response Y (train.y), and all other data in the table except month end return as training inputs (train.x).
3. Separate training set into overlapping groups with expanding size. The beginning group starts with first 4 months of data. The next group has first 8 months of data. Each group of data in the training set will include 4 extra months of new data compared to the previous group.
4. The testing set was chosen as the next 4 months of the last month of the training set, with a constant size. For instance, the first testing group includes data from 5th to 8th month, the second group includes data from 9th to 12th month, the third group includes data from 13th to 16th month, etc.

Note the number of month in each group can be altered. However, if the number is too small, the whole cross-validation process would take too long.

For one set of parameters, the model is separately trained with groups of training data prepared as above and evaluated against the corresponding group of testing data.

For instance in the case of stock return forecast, in the first iteration, the 1st group of training data (1th-4th month) will be used to train the model. The prediction results are evaluated against 1st group of testing data (5th-8th month) to obtain mean square of error. In the second iteration, the 2nd group of training data (1th-8th month) will be used for training, and the prediction results are evaluated against 2nd group of testing data (9th-12th month). The mean square error from the 2nd group is added on the 1st group. The iteration continues for the rest of the groups. The total mean square error was then compared against that of from other hyper parameters.

Below is an overview of the performance of each set of parameters, from which we can see that the test accuracy is not an increasing function of model complexity. Too much complexity may incur over-fitting problem, which reduces the performance for test data.

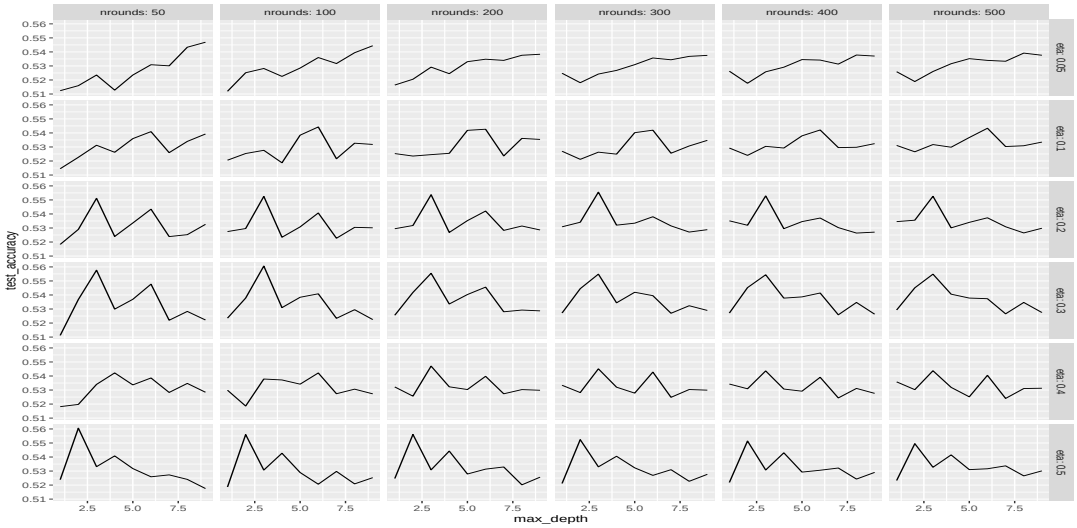

Figure 4.1: Parameter Tuning Result for Classification

We do it for the forecast prediction as well. From Figure 4.2 we can tell a small eta generally yields a better results compared to a larger eta. As the maximum depth increase, MSE will also increases. This increase is not monotonic, in some cases, larger maximum depth will yield better result. From the figure the impact of nrounds is not significant after a certain threshold. However, if we look at the value of the MSE we find in general larger nrounds will yield slightly better result, combing with certain level of max_depth. For our training set, $\eta = 0.01, nrounds = 500, max\_depth = 4$ yield the best result. Later, we did a cross validation on the whole data set, the result is similar except maximum depth becomes 2.

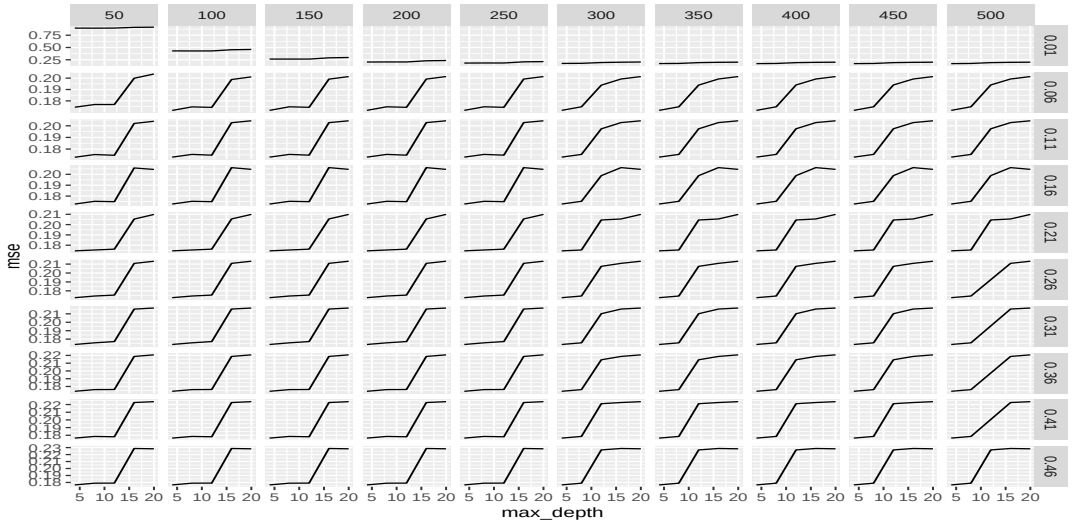

Figure 4.2: Parameter Tuning Result for Prediction

For classification problem, to take a closer look at how model performs for the change of each parameter, we draws the graphs below. These graphs also include the in-sample accuracy, which gives us a rough relationship of out-of-sample performance and in-sample one.

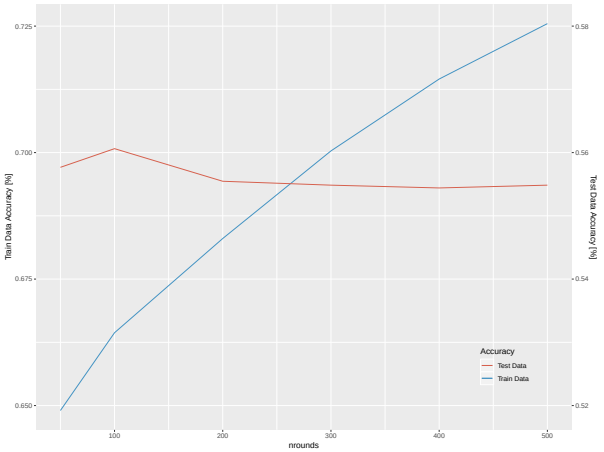

Figure 4.3: XGBoost nrounds vs accuracy

From Figure 4.3, we can see that the test data accuracy is rather flat. When the other parameters are chosen as optimal, change in nrounds doesn't pose a significant influence on the out-of-sample accuracy.

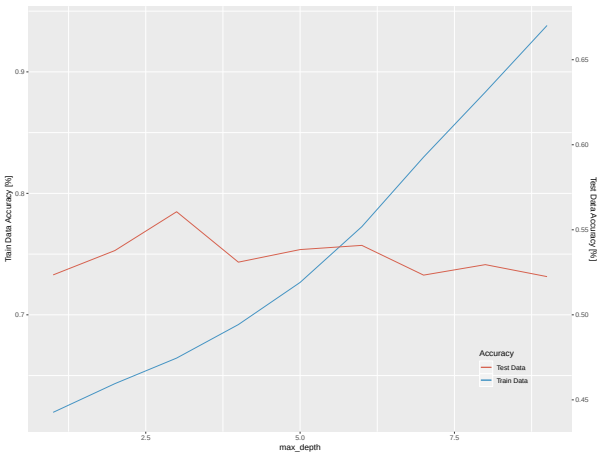

Figure 4.4: XGBoost max_depth vs accuracy

From Figure 4.4, we can see that the test data accuracy is downward sloping. When other parameters are chosen as optimal, change in max_depth reduces the test data accuracy. This implies that a more complex model with more numbers of depths reduces the prediction accuracy.

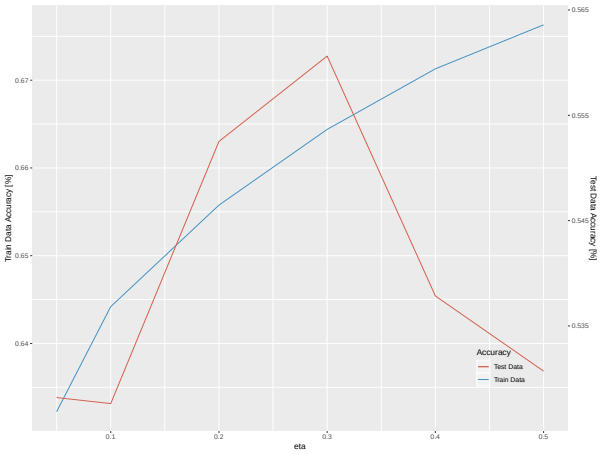

Figure 4.5: XGBoost eta vs accuracy

From Figure 4.5, we see that testing data accuracy increases with eta up until eta is around 0.3, then decreases with eta. The graph implies that eta has a huge impact on the test data accuracy. A medium learning rate is optimal for the model.

5. Feature Importance

Figure 5.1 shows the relative importance of the features that contribute to the model prediction result. It indicates that the features sentiment_neutral_end, sentiment_bullish_end, realize_vol_spx_end, retmonth_spx_end, sentiment_bearish_end, ebt_start, marketcap_start, industry, adj_Close_end and netmargin_start have the highest influence on the classification.

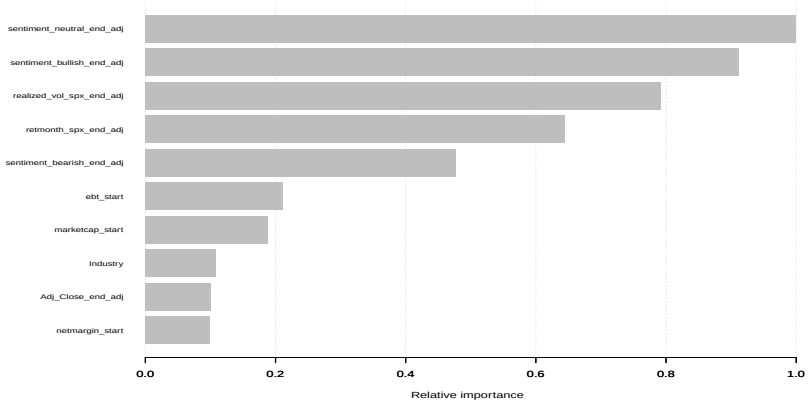

Figure 5.1: Feature Relative Importance

We can tell that the largest group of importance features are about the individual investors' sentiments, which represent the investors' opinions about the future growth of the company. This is reasonable, since the investors' opinion at the end of this period affect their investment decisions for the next period. And those investment activities will directly dominate the growth or falling of the company's stock.

The second largest group of importance features are the monthly return and realized volatility of the S&P 500 index of current period, which represents the trend of the market. The influence on the prediction result is reasonable since there is a high correlation between individual companies and the market.

The other important features are Earning before tax, market capitalization, industry sectors, adjust closing price, profit margin, sorting by importance. Earning before tax and profit margin are the indicators that indicate the individual companies' operational conditions. Those indicators are revealed in the financial report of the individual company, and represent the profitability of the company during the current period. The individual investors will react accordingly to those information, and affect the tendency of the companies' stock price.

Market capitalization refers to the total dollar market value of a company's outstanding shares of stock. It is important since it shows the size of a company, which is a basic determinant of various characteristics the investors may be interested in, including risk. In practice, small-cap companies may have more risk exposure than large-cap companies, this may lead to higher fluctuation of the company's stock price.

The indicator 'industry' show the specific industry sectors the companies are in. This is of great importance since the features of different industry sectors may vary. For example, the Oil & Gas Equipment & service industry may rise and fall along with the overall economic activities. However, the economic environment may have little effect on the biotechnology industry. So, taking industry sectors into account may help us make more accurate predictions.

6. Conclusion

In this paper, elastic-net regression and XGBoost models was introduced to predict stock returns for the first part, and XGBoost model obtained a better accuracy with relative higher R^2 comparing with other models. We applied time-series cross validation method for parameter tuning to avoid look-ahead bias of time-series model. With a learning rate ($\eta = 0.01$), maximum depth ($max\_depth = 2$), and number of trees ($nrounds = 500$), the output model has an R-square of around 2.7% in predicts stock returns during out-of-sample testing. We found that a slow learning rate and a moderate number of shallow trees produces the most accurate prediction without running into over-fitting issue. The small R-square also implies that it is very hard to predict stock returns accurately. However, any model with an positive out-sample R-square is a major breakthrough since the model is better than mean in minimizing mean square of error. This essentially suggests that the model can accurately predicts the mean of next month stock return and therefore a model with a positive out-sample R-square is a major achievement. From the result of our model, we believe further out-sample testing is necessary to investigate the accuracy of the model and conclude whether it can sustainably maintain a positive R-square as achieved in the out-of-sample testing.

For the second part, we applied different variable selection methods and trained data with multiple machine learning models, and chose XGBoost model because of better direction prediction. Parameter tuning is also extremely essential to prevent over-fitting and improve the forecast performance. With 0.3 learning rate, the model reached its optimal test data accuracy for classification problem. We found that features such as investors' sentiments and realized volatility of S&P 500, play an important role in predicting stock return. However, further research needs to be conducted before we can finally conclude which model or which features will lead to best result for stock return forecast.