STOCK RETURN PREDICTION VIA MACHINE LEARNING

Wenxuan Ma
Institute of Statistics and Big Data
Renmin University of China
mawenxuan@ruc.edu.cn

May 10, 2022

1 Introduction

Predicting stock return is one of the most acknowledged problems in the financial field. However, market efficiency forces return variation to be dominated by unforecastable news. This issue makes the prediction task tricky. In recent years, there has been an increasing interest in conducting neural network models (NNs) to predict risk premiums. Compared with traditional empirical methods, NNs have more advantages in handling high dimensionality and own more abundant function forms. But due to this flexibility, there is a great possibility of overfitting the model. Meanwhile, lack of interpretability makes such models appear as black boxes to economists.

Tree-structured models have a clear advantage in model interpretability because of their modeling construction processes. However, far too little attention has been paid to applying tree-structured models in asset pricing. [2] has proposed Panel Trees to construct profitable investments. They train the model iteratively and globally with a fixed tree structure. Their work has well-combined machine learning and asset pricing.

In [1], they conduct a series of traditional statistical models and machine learning methods to measure the asset risk premiums. They focus on the comparative analysis of these models' predictive performance. From their study, the tree-based models and neural network models outperformed other models. They also find the important variables that most dominate the asset risk premiums prediction. And they also provide evidence for economic gains from machine learning forecasts in the form of portfolio Sharpe ratios.

In this project, I will focus on the stock return prediction via machine learning methods as in [1]. And I will apply a new tree-based model which integrates the neural network into the tree-based model. The aim is to see whether this kind of method will help to predict stock return.

2 Data Information

I obtain the monthly total individual equity returns from CRSP [Services] which begins in January 1990 and ends in December 2020. The corresponding characteristics data are available from Dacheng Xiu's Web site [Xiu] which begins in March 1957 and ends in December 2021. The characteristics listed here have been studied in the previous cross-section of stock pieces of literature.

In this project, I assume the panel of stocks is balanced and replace the missing characteristics with the cross-sectional median at each month for each stock respectively. For the remaining missing variables, I use median interpolation. Then I merge these two datasets by date and permno. I call the new dataset 'stock.csv' which has 2279269 observations and 95 characteristics. The period is from 1990-01-31 to 2020-12-31, a total of 31 years. The number of stocks is 23,099 during this period.

In the following parts, I will conduct exploratory data analysis and show the data details from three levels, which are stock yearly performance, cross-section performance, and stock level time-series performance.

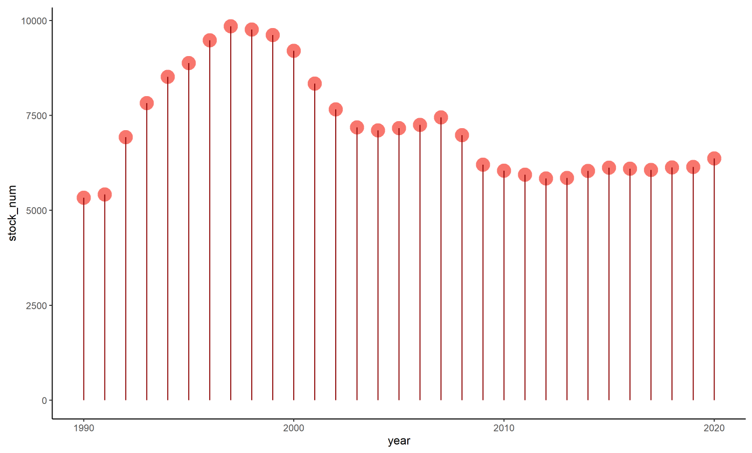

Figure 1: The number of stocks during 1990-2020

2.1 Yearly Performance

From figure 1 we can see that the number of stocks changes every year. It peaked around 1997 and gradually decline to a stable level. So there is a need to refit the predictive models every year, or even every month.

2.2 Cross Section Performance

Here I fix the date to 2020-12-31. There are 5,885 stocks on this day. First, I use the condition number test to detect whether there is multicollinearity between features. The condition number above 100 means that the regression may have severe multicollinearity. This result also seems reasonable because there are many lagged variables in this stock panel data, such as momentum. Therefore, when we want to use multiple characteristics to predict stock return, there is a need for us to choose models which can deal with the multicollinearity problem.

Then I select a small set of characteristics including the target variable return to check their correlation, which are log market equity (mvel1), book-to-market (bm), dividend to price (dy), earnings to price (ep), cash flow to price ratio (cfp), asset growth (agr), absolute accruals (absacc), stock momentum (mom12m), return on assets (roaq), return on equity (roeq), sales growth (sgr), leverage (lev), organizational capital (orgcap), share turnover (turn), idiosyncratic return volatility (idiovol), and market beta (beta).

In the previous studies, stocks with low characteristics like log market equity (mvel1), asset growth (agr), absolute accruals (absacc), sales growth (agr), share turnover (turn), or idiosyncratic return volatility (idiovol), have abnormally high average returns, and stocks have a positive association with the other characteristics. However, in the correlation coefficient figure 2, we can hardly see this inverse proportional relation. Besides, in figure 2 the correlation coefficients between stock return and these characteristics are far too small which confirms that the stock return can be hardly captured. As well as the correlation coefficients between characteristics are relatively high which lends support to the multicollinearity.

2.3 Stock Level Time-series Performance

In this part, I will show the time series changing performance of six stocks, Apple (AAPL), Microsoft (MSFT), Bank of America (BAC), Google (GOOG), Amazon (AMZN), and IBM (IBM).

Figure 3 shows that the fluctuation of different stocks is different. And it can be found that the listing time of different stocks is also different. For example, Amazon was on the market in 1997, Google was on the market in 2004, and other four companies were on the market before 1990. Hence, there is a need to refit the model once a year because different stocks are listed or delisted every year.

Figure 4 demonstrates the box plots of these six stocks. We can clearly find that the average return of Apple and Amazon is relatively high, while that of IBM is the lowest. Interestingly, the volatility of Apple and Amazon is high, while that of IBM is the lowest. This is also consistent with expectations, that is, the investment risk of stocks with high returns is high, and the investment risk of stocks with low returns is low.

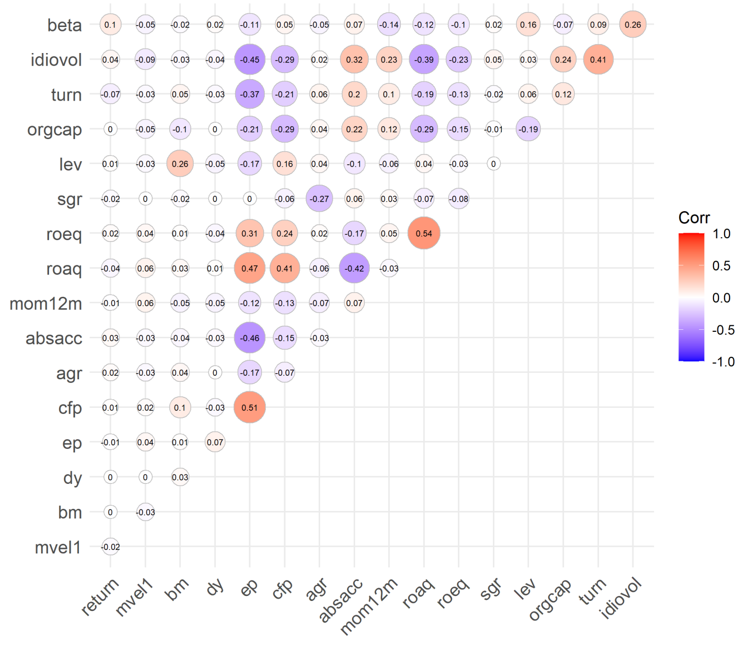

Figure 2: The correlation coefficient between stock return and several characteristics

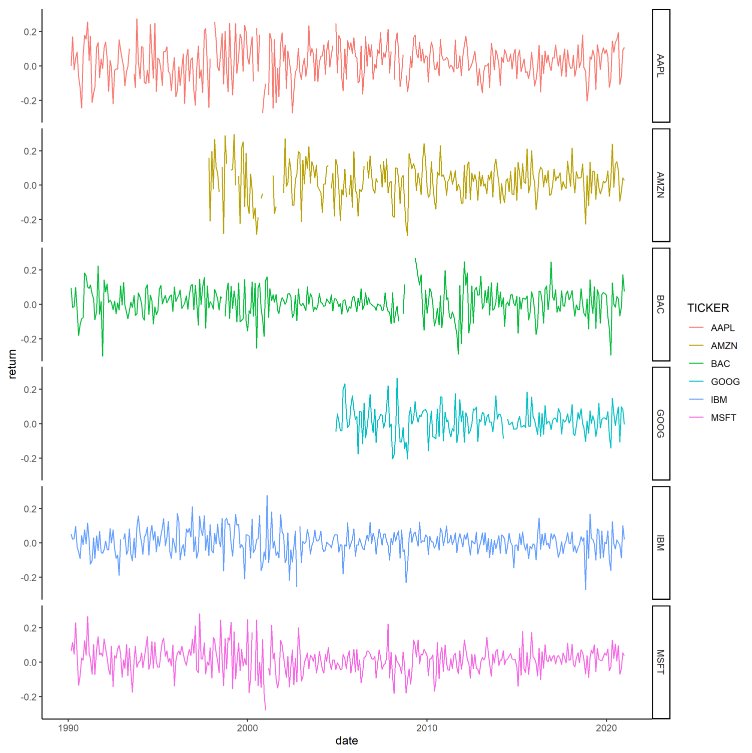

Figure 3: Time series return changes of six stocks

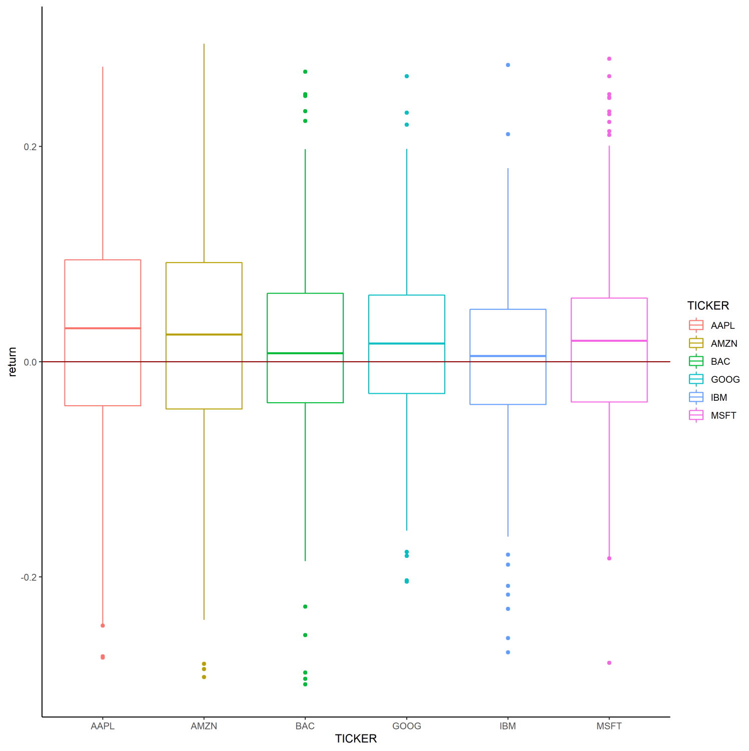

Figure 4: Box plots of six stocks

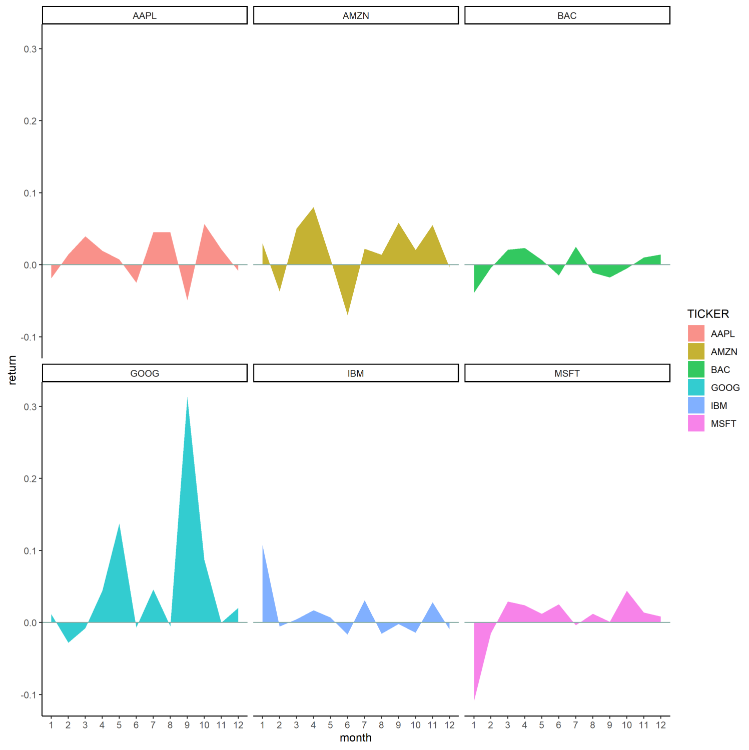

Figure 5: Average return in each month

Figure 5 displays the average return for each month. The results of the six stocks all show seasonal fluctuation. The change in some stocks is relatively more apparent. Based on this phenomenon, we should refit models once a month.

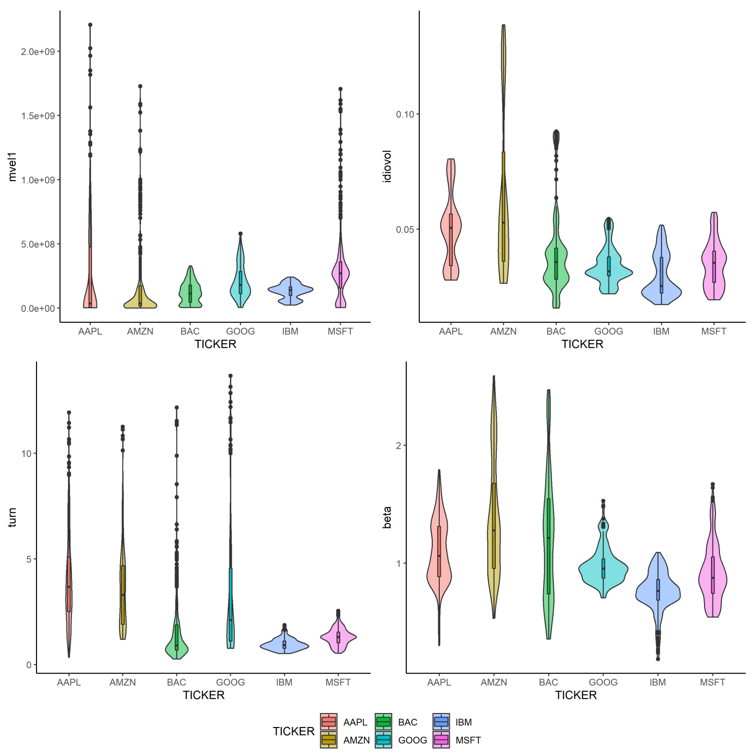

Figure 6: The violin diagram of four variables

Figure 6 is the violin diagram of four variables, Size, Momentum, Turnover, and Market Beta. We can find that the distribution of these characteristics is distinct in different stocks. Therefore, in the following section, I will construct different models to depict the relationship between stock returns.

3 Models

In asset pricing prediction task, the general model form is an additive prediction error model [1]:

$$r_{i,t+1} = \mathbb{E}_t(r_{i,t+1}) + \varepsilon_{i,t+1}, \tag{1}$$

where

$$\mathbb{E}_t(r_{i,t+1}) = g^\star(z_{i,t}). \tag{2}$$

Stocks are indexed as $i = 1, \cdots, N_t$ and times are indexed as $t = 1, \cdots, T$. Characteristics are $z_{i,t}$. Next we will discuss the diverse function forms of $g^\star(z_{i,t})$.

3.1 Linear Regression

In linear models, $g^\star(z_{i,t})$ is assumed to be the linear combination of characteristics:

$$g\left(z_{i,t}; \theta\right) = z_{i,t}' \theta. \tag{3}$$

The corresponding parameters are optimized by minimizing the mean squared error (MSE):

$$\mathcal{L}(\theta) = \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} \left(r_{i,t+1} - g\left(z_{i,t}; \theta\right)\right)^2. \tag{4}$$

3.2 Ridge Regression

The model setting of ridge regression is the same as that of linear regression. Differently, it adds a penalty to the loss function:

$$\mathcal{L}(\theta;\alpha) = \mathcal{L}(\theta) + \phi(\theta;\alpha), \tag{5}$$

where

$$\phi(\theta;\alpha) = \alpha \sum_{j=1}^{P} \theta_j^2. \tag{6}$$

Ridge regression uses the l_2 parameter penalization which encourages the coefficient estimates closer to zero. In the objective function, the hyperparameter α can be chosen by cross-validation.

3.3 Regression Tree

In this part, I will use CART as the representative of the regression tree model. One of the most attractive merits of a tree-based model is that it can capture the interaction of predictors and make the model interpretable. It manages to find the homogenous groups (data at the same leaf nodes) based on different split criteria. Like CART, it finds the best split to minimize the MSE. The models on the leaves are constant-valued functions. After building up an initial tree, CART will post prune the tree structure to balance the model accuracy and complexity based on cross-validation.

The prediction of a tree $\mathcal{T}$ with K leaves and depth L is

$$g\left(z_{i,t};\theta,K,L\right) = \sum_{k=1}^{K} \theta_k \mathbf{1}_{\{z_{i,t}\in C_k(L)\}}. \tag{7}$$

3.4 Ensemble Methods

3.4.1 Bagging

Random forest is an ensemble method that takes average prediction of the individual trees. Here we use CART as the base leaner to predict the stock return. The individual trees are built up by a certain number of bootstrap samples of the original data.

3.4.2 Boosting

Another well-known ensemble method is boosting. The representative boosting trees are Gradient Boosting Decision Tree (GBDT) and XGBoost. They train a sequence of decision trees. The final result is the summation of these trees.

3.5 Neural Networks

Neural network models are one of the most powerful supervised learning methods in recent years. In this project, I will use the same network structure setting as that in []. There are four neural network models. The simplest neural network, named NN1, has one hidden layer with 32 neurons. NN2 has two hidden layers with 32 and 16 neurons respectively. NN3 has three hidden layers with 32, 16, and 8 neurons respectively. NN4 has four hidden layers with 32, 16, 8, and 4 neurons respectively. We choose the rectified linear unit (ReLU) as our nonlinear activation function, which is defined as

$$\mathrm{ReLU}(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{otherwise} \end{cases}. \tag{8}$$

We choose the Adam algorithm to optimize the parameters of the neural network. We set the learning rate to 0.01 and batch size to 128. To avoid the overfitting problem, we use early stopping to decide when to stop fitting the model.

3.6 A New Tree-based Model

Apart from the models introduced before, we now present a new tree-based model to predict the stock return. The function form of this model is different from that of the previous:

$$\mathbf{r}_t = \sum_{m=1}^{M} \left(g_m^{\star}\left(\mathbf{z}_t\right) + \varepsilon_m\right) \mathbf{1}_{\mathbf{z}_t\in\mathcal{R}_m}. \tag{9}$$

As a tree-based model, it has three aspects distinct from CART. In the view of the splitting criterion, this model identifies different regions by splitting, while CART aims to decrease the sum of squared errors. In the view of modeling on the leaf, this model trains neural networks on the leaves, while CART has constant-valued functions on the leaves. In the view of the pruning algorithm, this model has the trait of auto-pruning, so it does not need the post pruning procedure like CART.

4 Experiment

4.1 Data Splitting

Considering the computation cost, we only conduct prediction for five years and refit the model by one year. We split the data into three parts. The training dataset is from 2000 to 2009, a total of ten years. The validation dataset is from 2010 to 2015, a total of five years. The testing dataset is from 2016 to 2020, a total of five years. We refit the models once a year and then roll the training dataset and validation dataset forward to increase them by one-year correspondingly.

4.2 Predictive Performance Evaluation

Here we choose two evaluations to assess the predictive performance of various models. The first one is the root mean squared errors (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{NT}\sum_{(i,t)\in\mathcal{T}_3}\left(r_{i,t+1} - \widehat{r}_{i,t+1}\right)^2}, \tag{10}$$

where $\mathcal{T}_3$ indicates the testing dataset.

The other metric is out-of sample R^2:

$$R^2_{\text{oos}} = 1 - \frac{\sum_{(i,t)\in\mathcal{T}_3}\left(r_{i,t+1} - \widehat{r}_{i,t+1}\right)^2}{\sum_{(i,t)\in\mathcal{T}_3} r^2_{i,t+1}}. \tag{11}$$

5 Results

5.1 Comparison of RMSE

Figure 7 clearly illustrates that among all these methods, Xgboost has the worst predictive performance. And the other methods' RMSEs do not have an obvious gap between each other. Therefore, we need to consider another performance metrics to compare these methods.

5.2 Comparision of R^2_{oos}

Table 1: **Yearly percentage** R^2_{oos}: The Bold fonts represent the best two performance. The integers in parentheses indicate the rank of models.

Model	2016	2017	2018	2019	2020	Average
OLS	-9.8120(10)	-12.2760(10)	0.9103(8)	-2.8203(9)	-7.4674(10)	-6.218(10)
Ridge	-1.6871(8)	-2.9609(8)	0.9579(7)	-2.7222(8)	-7.2243(9)	-3.5011(8)
CART	0.0868(5)	0.0138(5)	1.2273(6)	-0.0333(3)	-3.0452(6)	-0.8223(6)
RF	0.2431(4)	**0.2811**(1)	1.3832(4)	**-0.0058**(2)	-2.8207(5)	-0.6527(5)
GBDT	-0.9174(7)	-0.4793(7)	**2.3930**(1)	-1.5020(5)	-4.0521(7)	-1.4866(7)
XGBoost	-74.0202(11)	-171.8072(11)	-136.2014(11)	-109.2651(11)	-236.2871(11)	-159.5(11)
NN1	-1.1692(9)	-6.5893(9)	-3.4164(10)	-3.9118(10)	-5.9849(8)	-4.4364(9)
NN2	0.5417(3)	0.0858(4)	1.3164(5)	-0.1403(4)	-2.6560(4)	-0.6099(4)
NN3	**0.7031**(1)	0.1014(3)	1.6758(3)	-0.2408(7)	**-1.2257**(2)	**-0.0533**(1)
NN4	-0.0235(6)	-0.1266(6)	-0.1826(9)	-0.1809(6)	**-0.1456**(1)	**-0.1356**(2)
New Tree	**0.5923**(2)	**0.1583**(2)	**1.8184**(2)	**0.0014**(1)	-2.1647(3)	-0.2877(3)

From table 1, we can get several conclusions. From the single-year performance and the average five years performance, the ordinary least squares get the worst outcome. After adding a penalty, the performance of the ridge regression becomes better. For the tree-based models, the random forest has a better effect, than the single tree model,

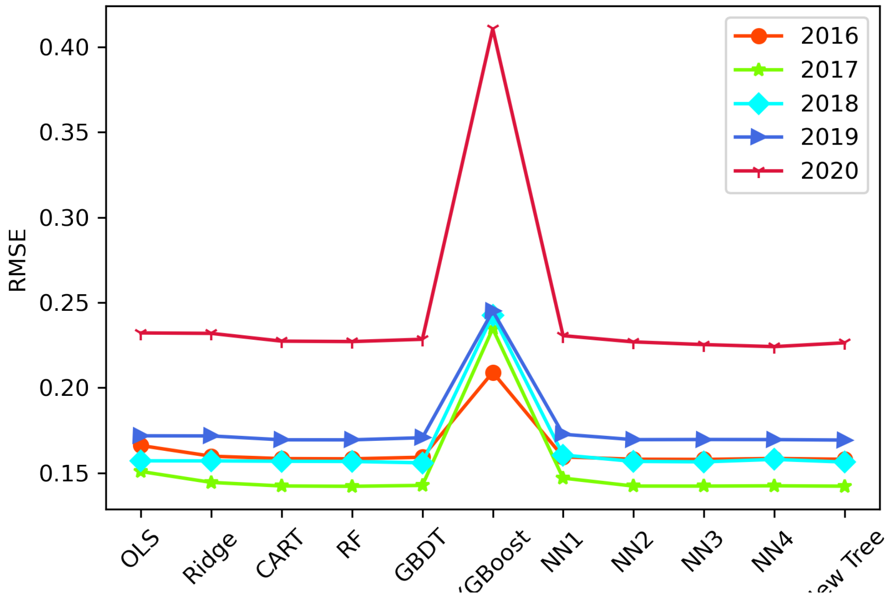

Figure 7: The predictive RMSE of these methods.

CART. Surprisingly, boosting trees' predictive performance is not as good as expected. Especially, Xgboost's predictive performance is far worse than other methods. While in other prediction tasks, Xgboost is always the better model. From my perspective, the reason why GBDT and XGBoost are not as good as other methods is that hyperparameter tuning is very important. However, I only focus on two hyperparameters, the minimum data size on the leaf nodes and the number of estimators in each ensemble method. And the performance of neural network models can compete with the tree-based models. Meanwhile, the new tree-based model also has impressive performance.

5.3 Interpretability of Tree-based Models

In this project, tree-based models are all built up in a parallel-axis way. That is we only partition the data based on one variable every time. This kind of construction endows tree models with interpretable ability. In figure **??** we show the results of CART and the new tree-based model in 2016 to illustrate their interpretable ability.

6 Summary

In this section, we summarize the experiment results from two perspectives, prediction performance, and interpretable ability. From the result of RMSEs and R^2_{oos}s we can find that the new tree-based model, random forest, and NN3 get the best performance. The result of linear regression is bad while adding a penalty function can improve the outcome. Surprisingly, the boosting models are not excellent as expected. Because in most prediction tasks, GBDT and XGBoost usually outperform other models. When referring to the model interpretability, we can focus on the model forms. The traditional statistical models, like linear regression and ridge regression, are interpretable. Because the parameters in the models have statistical meanings. Tree-based models possess interpretability due to the modeling processes. Although neural network models have high-quality performance in prediction, they cannot interpret their models.

References

[1] Gu, S., Kelly, B., and Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies*, 33(5):2223–2273.

[2] He, X., Cong, L. W., Feng, G., and He, J. (2021). Asset pricing with panel trees under global split criteria. *Available at SSRN 3949463*.

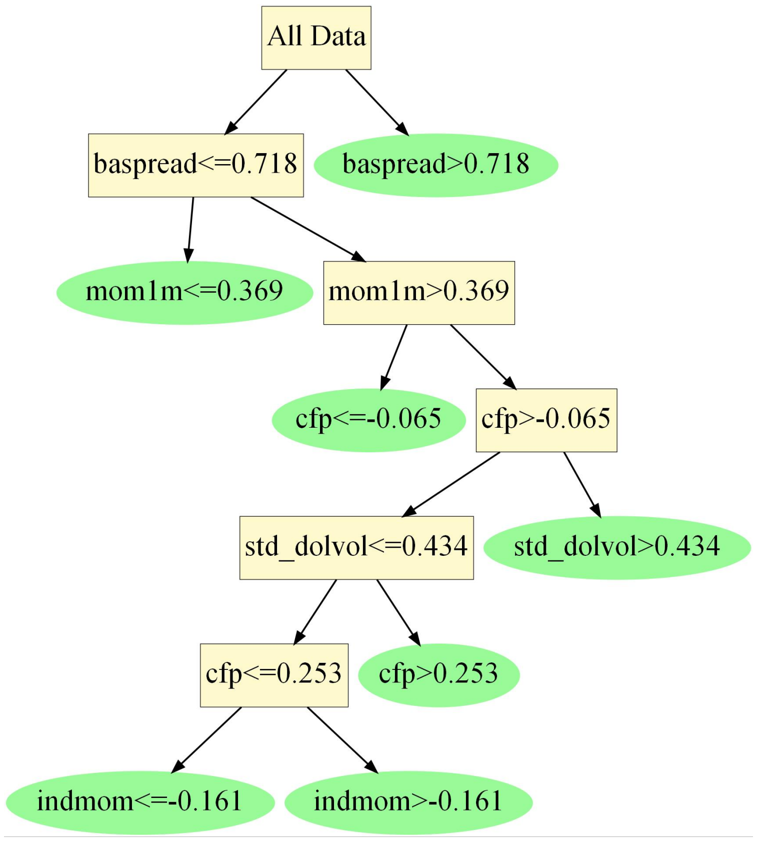

Figure 8: CART structure in 2016

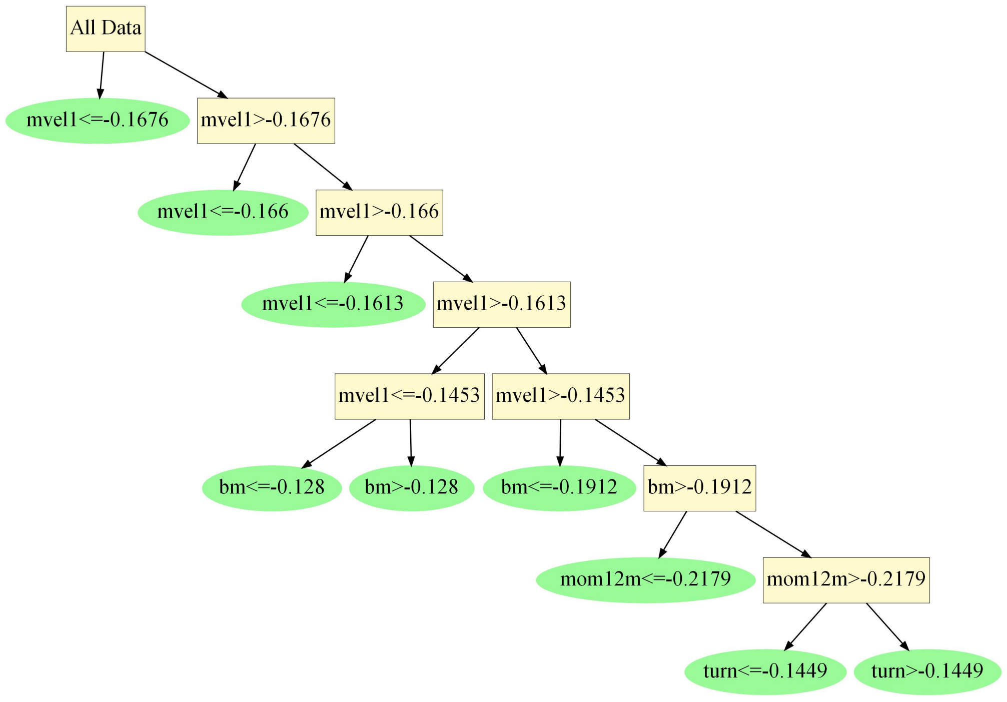

Figure 9: New tree-based model structure in 2016

[Services] Services, W. R. D. Wharton research data services. `https://wrds-www.wharton.upenn.edu/`.

[Xiu] Xiu, D. Dacheng xiu's web site. `tps://dachxiu.chicagobooth.edu/`.