**Forecasting Stock Returns Using Random Forests and Gradient Boosting**

Tian Song

April 2020

Abstract

Forecasting returns is an important but difficult task for stock market investors due to the random nature of the financial markets. Traditional regression-based methods assume a linear relationship between expected stock returns and predictors, which can be problematic for analyzing large complex data. We perform two machine learning methods, random forests and gradient boosting, to forecast return. These two machine learning methods improve prediction accuracy at both stock level and portfolio level by including nonlinear effects and predictor interactions, although their forecasts are still noisy at the stock level. Random forests and gradient boosting also identify a different set of key return predictors. Both random forests and gradient boosting pick short-term reversal (SREV) and investment-to-assets ratio (IA) as the top two predictors, while linear regression picks net stock issues (NSI) and market cap (MC).

## **I. Introduction**

Forecasting returns is an important task for stock market investors. However, it is extremely difficult due to the random nature of the financial markets. Traditionally, researchers and practitioners use regression-based models to forecast returns. These models generally assume a linear relationship between expected stock returns and predictors, which can be problematic for analyzing large complex data. In contrast, the new methods in machine learning and big data analytics allow flexible functional forms and are often designed to handle high-dimensional data. Thus, it is interesting to see if they can be used to improve stock return forecasts. In this project, we apply two machine learning methods, random forests and gradient boosting, to forecast returns and compare their performance with that of linear regression.

## **II. Data**

Our data consists of the largest 1,000 US stocks with available information on ten well-known stock return predictors from January 1995 to December 2019. Stock returns are monthly and the ten predictors are measured and updated at the beginning of each month. The ten predictors are widely used in the cross-sectional stock return literature (e.g., see Hou, Xue, and Zhang 2015). Table 1 describes the variables and presents their summary statistics.

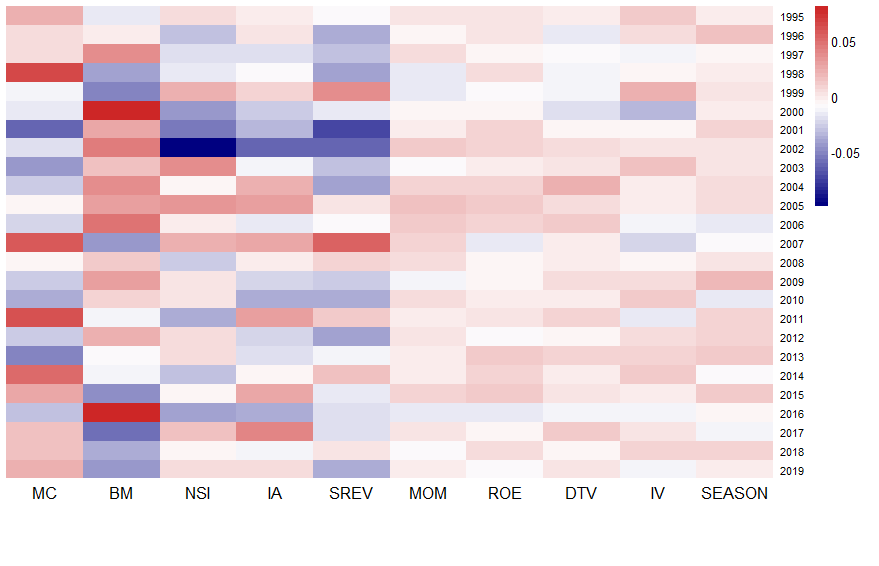
**Table 1. Variable Description and Summary Statistics**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Variable | Description | Mean | S.D. | Min | Median | Max |
| PERMNO | unique stock ID | - | - | - | - | - |
| Date | date (end of month) | - | - | - | - | - |
| RET | monthly stock return | 0.01 | 0.12 | -0.98 | 0.01 | 2.60 |
| MC | market cap ($M) | 8,689 | 22,324 | 163 | 1,862 | 158,918 |
| BM | book value-to-market value | 0.52 | 0.37 | 0.03 | 0.43 | 2.00 |
| NSI | net stock issues | 0.03 | 0.10 | -0.14 | 0.01 | 0.56 |
| IA | investment-to-assets | 0.15 | 0.32 | -0.31 | 0.07 | 1.91 |
| SREV | short-term reversal | 0.02 | 0.11 | -0.29 | 0.01 | 0.39 |
| MOM | momentum | 0.12 | 0.57 | -0.81 | 0.04 | 2.64 |
| ROE | return on equity | -0.01 | 0.14 | -0.91 | 0.02 | 0.24 |
| DTV | dollar trading volume ($M) | 10.20 | 34.75 | 0.00 | 0.26 | 249.71 |
| IV | idiosyncratic volatility (%) | 3.02 | 2.40 | 0.06 | 2.33 | 13.82 |
| SEASON | seasonality | 0.01 | 0.15 | -0.36 | 0.00 | 0.56 |

The return predictors predict returns across stocks, not over time. Moreover, the cross-sectional distributions of these predictors can change systematically over time, making their values less comparable over time. Thus, we follow Gu, Kelly, and Xiu (2019) and cross-sectionally rank all predictors each month and then transform these ranks into the [-1, 1] interval. The resulting transformed predictors will be used in our analysis.

Figure 1 shows a heatmap of the cross-sectional correlations between returns and (transformed) predictors from 1995 to 2019.[[1]](#footnote-1) On average, four predictors (MC, NSI, IA, and SREV) negatively forecast returns, while the other six positively forecast stock returns, consistent with the literature (Hou et al. 2015). However, the sign of predictability can change over time. For example, high BM stocks outperform low BM stocks in the first half of 2000s, but underperform in late 1990s and late 2010s.

**Figure 1. Correlation Heatmap**

**III. Methods**

**Models**

We forecast returns using three models: linear regression, random forests, and gradient boosting. The linear regression with ordinary least squares (OLS) estimation is widely used in the cross-sectional stock return literature (e.g., Fama and MacBeth 1973) and can serve as the benchmark model for comparison. We perform two versions of linear regression. First, following common practice (e.g., Lewellen 2015), all ten predictors are included in *a multiple regression* and no variable selection is performed. Second, we construct *a forecast combination* based on univariate regressions. Following Han, He, Rapach, and Zhou (2018), we conduct univariate regressions with one predictor at a time and then take the average of the forecasts from all univariate regressions as our overall forecast.

Random forests are a modification of a more general procedure known as “bagging” (Breiman, 2001). The bagging procedure draws different bootstrap samples of the original data, fit a separate regression tree to each sample, and then average their predictions. To reduce the correlation among trees in different bootstrap samples, random forests builds a large collection of de-correlated trees by randomly selecting a subset of predictors for splitting at each branch.

Like random forests, gradient boosting is also an ensemble method that combines predictions from many different trees. However, it builds an ensemble of shallow and weak successive trees with each tree learning and improving on the previous one.

**Estimation Procedure**

As shown by the correlations in Figure 1, the relationship between stock returns and predictors can change significantly over time. To account for such time variations, we estimate the models using 11-year rolling windows. For each rolling window, we split the data into a 10-year training sample and a 1-year testing sample. We build models in the training sample and evaluate their out-of-sample predictions in the testing sample. Then we move the rolling window forward by one year and repeat the process. The detailed procedure is as follows:

1. Use the first 10 years (1995 – 2004) as the training sample to build models. For the ML models, the sample is further divided into training and validation subsamples for tuning hyperparameters.
2. Evaluate the model’s performance in the following year (2005).
3. Moving forward by one year, use the second 10 years (1996 – 2005) to train models and test the model’s performance in the following year (2006).
4. Repeat this process until the end of sample period.

At the end of the procedure, our “combined” testing period covers the 15 years from 2005 to 2019.

**Tuning for ML Models**

The machine learning methods rely on the choice of hyperparameters, e.g., the number of predictors used for splitting in random forests, the number of trees, the depth of trees, and the learning rate in boosting. Such choices are critical to the performance of machine learning methods as they control the model’s complexity. As such, selecting the hyperparameters (“tuning”) is an important step in machine learning.

For random forests and boosting tree, we further divide our training data into two subsamples, training and validation. First, using the training sample, we estimate the model with a specific set of hyperparameters. Then, we calculate the prediction for validation sample using the estimated model from the training sample. Finally, we calculate the prediction errors for the validation sample, and iteratively search for hyperparameters that minimize the prediction errors.

To select the best hyperparameters, we create a grid and loop through each hyperparameter combination. This grid search tends to be computationally intensive. In this project, we use the “h2o” package in R for model tuning. “h2o” is a powerful and efficient java-based interface that provides parallel distributed algorithms, which allows us to tune our models more efficiently. Table 2 shows the set of hyperparameters and values used for the ML models.

**Table 2. Hyperparameters For Random Forests and Gradient Boosting**

|  |  |
| --- | --- |
| Random Forests | Gradient Boosting |
| Max\_depth = (1,2,3,4,5,6)  Mtries = (3,4,5)  Ntrees = 300 | Max\_depth = (1,2,3,5)  Learning Rate = (0.01, 0.05, 0.1)  Ntrees = 5000 |

**Model Evaluation**

We evaluate model performance using the following criteria:

1. Stock-level prediction. Since the models are estimated at the stock level, it is natural to compare the average magnitude of prediction errors for individual stock returns. To facilitate interpretation, we use the root mean squared error (RMSE), which has the same unit as returns. RMSE is defined as:

,

in which is the actual return for stock and is the forecasted return. We compute RMSE each year and report its values and summary stats for our 15-year testing sample.

1. Portfolio-level prediction. Individual stock returns are known to be extremely volatile and thus hard to predict with much accuracy. However, investors often hold portfolios, which helps average out the noise in individual stock returns. Thus, it is interesting to see how our return prediction performs at the portfolio level. Following Gu et al., at the beginning of each month, we sort stocks into ten portfolios (“deciles”) based on their forecasted returns. Then, we compute the equal-weighted average portfolio return using actual stock returns during the month and compute the return spread between the bottom and top deciles (“10-1”). If our return prediction is accurate, stocks in Decile 10 should outperform stocks in Decile 1, which leads to a significant “10-1” spread. We calculate the “10-1” spread for each month in our testing sample and report the mean, standard deviation, and the Sharpe ratios for the time-series of the spreads. In particular, the Sharpe ratio is a widely used metric for evaluating the performance of stock portfolios. It is defined as the average of excess returns (investment return minus the risk-free rate) divided by the volatility of excess returns. Intuitively, average excess return is the reward investors earn and the volatility is the risk they bear. Thus, the Sharpe ratio represents the reward-to-volatility ratio. The higher the Sharpe ratio, the better the portfolio performs. Formally, we calculate the annualized Sharpe ratio (SR) as follows:

,

in which is the return on the “10-1” spread portfolio, and is the multiplier for annualization (as the returns are monthly).

1. Predictor importance. We also compare the relative importance of predictors in each model. Given the proliferation of predictors, identify the most reliable ones can be useful for investors.

## **IV. Results**

In this section, we compare return predictions from four methods using the criteria described in the previous section.

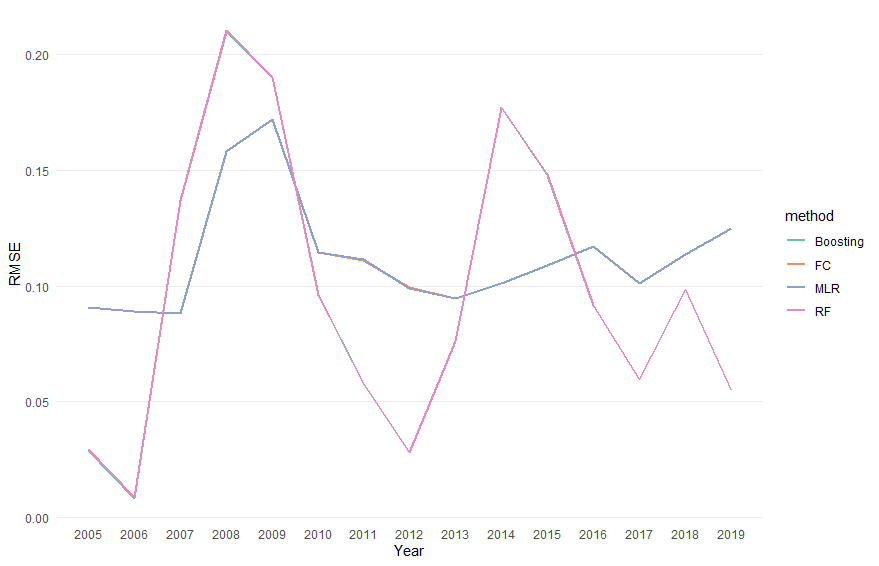
**Stock-level Prediction**

Table 3 and Figure 2 report the RMSE of prediction at the stock level for each testing sample. The two linear regression methods yield almost identical results. Similarly, results for the two machine learning forecasts are also close. On average, random forests and gradient boosting perform somewhat better than linear regression, with RMSE at 0.098/0.097 versus 0.112 for linear regression. However, the accuracy of these two machine learning methods varies greatly over time, especially when the market is volatile. For example, during the financial crisis of 2008-2009, random forests and gradient boosting yield higher RMSEs than linear regression. It is also worth noting that all four methods generate RMSEs an order of magnitude larger than the average monthly stock return (around 1%). This is not very surprising given the noisiness of individual stock returns, which has been well acknowledged in the literature.

**Table 3. Stock-level Prediction RMSE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Year** | **Multiple Linear Regression (MLR)** | **Forecast Combination**  **(FC)** | **Random**  **Forests**  **(RF)** | **Gradient Boosting**  **(Boosting)** |
| 2005 | 0.091 | 0.091 | 0.029 | 0.029 |
| 2006 | 0.089 | 0.089 | 0.009 | 0.008 |
| 2007 | 0.088 | 0.088 | 0.137 | 0.137 |
| 2008 | 0.158 | 0.158 | 0.210 | 0.210 |
| 2009 | 0.172 | 0.172 | 0.190 | 0.190 |
| 2010 | 0.114 | 0.114 | 0.096 | 0.096 |
| 2011 | 0.111 | 0.111 | 0.058 | 0.058 |
| 2012 | 0.099 | 0.099 | 0.028 | 0.028 |
| 2013 | 0.095 | 0.095 | 0.076 | 0.076 |
| 2014 | 0.101 | 0.101 | 0.177 | 0.177 |
| 2015 | 0.109 | 0.109 | 0.148 | 0.148 |
| 2016 | 0.117 | 0.117 | 0.092 | 0.091 |
| 2017 | 0.101 | 0.101 | 0.060 | 0.060 |
| 2018 | 0.114 | 0.114 | 0.098 | 0.098 |
| 2019 | 0.125 | 0.125 | 0.055 | 0.055 |
| **Mean** | **0.112** | **0.112** | **0.098** | **0.097** |
| **SD** | **0.024** | **0.024** | **0.062** | **0.062** |
| **Min** | **0.088** | **0.088** | **0.009** | **0.008** |
| **Median** | **0.109** | **0.109** | **0.092** | **0.091** |
| **Max** | **0.172** | **0.172** | **0.210** | **0.210** |

**Figure 2. Out-of-Sample RMSEs from 2005 to 2019**



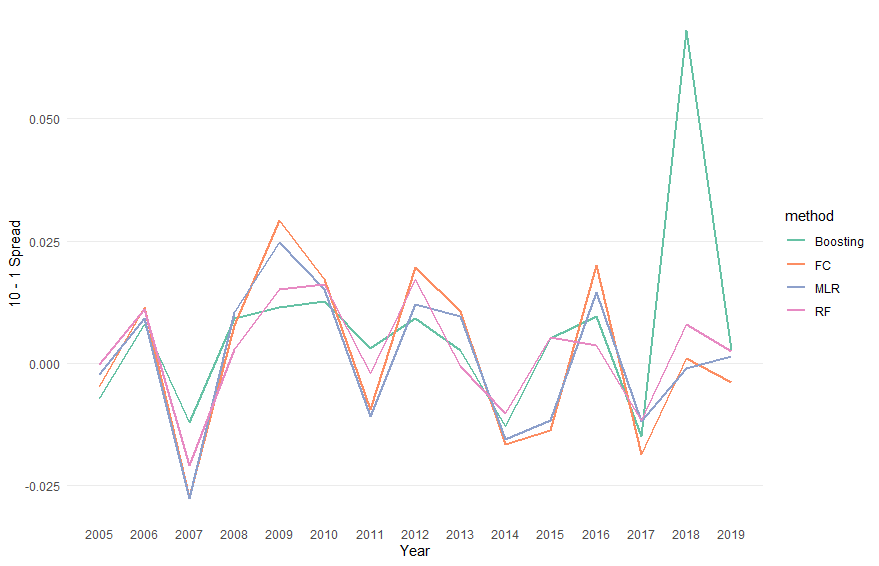
**Portfolio-Level Prediction**

We also compare forecasting performance of four methods at the portfolios level. Table 4 reports the return spread between top and bottom portfolios (“10-1” spread) for each testing sample, as well as the mean, standard deviation, and the annualized Sharpe ratios for the entire time-series of the spreads. If a model predicts stock returns well, the top-minus-bottom portfolio strategy should have a higher return, a lower volatility, and a high Sharpe ratio. At the portfolio level, the machine learning methods, especially gradient boosting, perform substantially better than the traditional methods. The average “10-1” spread is the highest for gradient boosting at 0.6% per month versus 0.2% for random forests and 0.1% for linear regression. The standard deviation is the lowest for random forecasts at 1.1% per month versus 1.4%/1.7% for linear regression and 1.9% for gradient boosting. Overall, the annualized Sharpe ratio is only 0.083/0.102 for linear regression but increase to 0.219 for random forests and 0.446 for gradient boosting. Finally, a caveat is that gradient boosting performs particularly well in 2018 but is largely comparable to random forests.

**Table 4. Return Spread between Top and Bottom Portfolios**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Year** | **Multiple Linear Regression (MLR)** | **Forecast Combination**  **(FC)** | **Random**  **Forests**  **(RF)** | **Gradient Boosting (Boosting)** |
| 2005 | -0.002 | -0.005 | 0.000 | -0.007 |
| 2006 | 0.009 | 0.011 | 0.011 | 0.008 |
| 2007 | -0.028 | -0.027 | -0.021 | -0.012 |
| 2008 | 0.010 | 0.008 | 0.003 | 0.009 |
| 2009 | 0.025 | 0.029 | 0.015 | 0.011 |
| 2010 | 0.015 | 0.017 | 0.016 | 0.013 |
| 2011 | -0.011 | -0.009 | -0.002 | 0.003 |
| 2012 | 0.012 | 0.020 | 0.017 | 0.009 |
| 2013 | 0.010 | 0.011 | -0.001 | 0.003 |
| 2014 | -0.015 | -0.017 | -0.010 | -0.013 |
| 2015 | -0.012 | -0.014 | 0.005 | 0.005 |
| 2016 | 0.015 | 0.020 | 0.004 | 0.010 |
| 2017 | -0.012 | -0.019 | -0.012 | -0.015 |
| 2018 | -0.001 | 0.001 | 0.008 | 0.068 |
| 2019 | 0.001 | -0.004 | 0.002 | 0.003 |
| **Mean** | **0.001** | **0.001** | **0.002** | **0.006** |
| **SD** | **0.014** | **0.017** | **0.011** | **0.019** |
| **Annualized Sharpe Ratio** | **0.083** | **0.102** | **0.219** | **0.446** |

**Figure 3. Return Spread from 2005 to 2019**



**Predictor Importance**

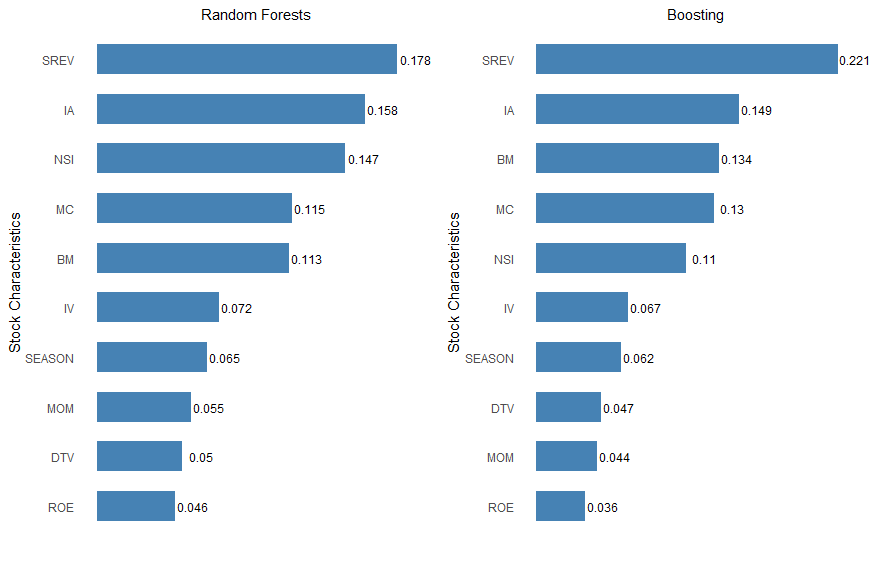
Finally, we investigate which predictors contribute the most power to return forecasts. For multiple regression, following Fama and MacBeth (1973), we calculate the time-series averages and (simple) t-statistics of regression coefficient estimates.[[2]](#footnote-2) In Table 5, based on the magnitude of estimates and their statistical significance, net stock issues (NSI), market cap (MC), and short-term reversal (SREV) are the top three predictors.

**Table 5. Multiple Regression Coefficient Estimates**

|  |  |  |
| --- | --- | --- |
|  | Mean | t |
| Intercept | 0.009944 | 5.21 |
| NSI | -0.001866 | -2.00 |
| MC | -0.002176 | -1.92 |
| SREV | -0.002838 | -1.71 |
| SEASON | 0.000258 | 1.63 |
| IA | -0.000507 | -1.26 |
| BM | 0.001519 | 1.25 |
| IV | -0.000182 | -0.54 |
| ROE | 0.000078 | 0.45 |
| MOM | 0.000166 | 0.35 |
| DTV | 0.000075 | 0.11 |

For random forests and gradient boosting, predictor importance is determined by the relative influence of each predictor in H2o package: whether the predictor was selected during splitting in the tree building process and how much the squared error (over all trees) improves as a result. H2o reports the normalized version of importance, in which predictor importance are normalized to sum up to one. In our analysis, we use this normalized importance from each testing sample, and calculate the average importance over the entire testing period. Figure 3 shows the relative importance of predictors in random forests and gradient boosting.

**Figure 4. Predictor Importance**



Both random forests and gradient boosting select short-term reversal (SREV) and investment-to-assets ratio (IA) as the top two predictors. However, random forests select net stock issues (NSI) as the third most important predictor while gradient boosting selects book-to-market (B/M). Both traditional and ML methods agree that SREV and NSI are the most important predictors. However, the ML methods find IA to be more informative while the traditional methods favor market cap (MC).

## **V. Conclusion**

We apply two machine learning methods, random forests and gradient boosting, to forecast stock returns. By including nonlinear effects and predictor interactions, the ML methods can improve the accuracy of return predictions at both the stock level and the portfolio level. However, their forecasts are still noisy at the stock level. The results in this study suggests that machine learning is a promising tool for empirical finance research.

## **References**

Breiman, Leo, 2001, Random forests, *Machine Learning* 45, 5–32.

Fama, Eugene F, and James D. MacBeth, 1973, Risk, Return, and Equilibrium: Empirical Tests, *Journal of Political Economy* 81, 607–636.

Gu, Shihao, Bryan Kelly, and Dacheng Xiu, 2019, Empirical Asset Pricing via Machine Learning, *Review of Financial Studies*, forthcoming.

Han, Yufeng, Ai He, David Rapach, and Guofu Zhou, 2018, Firm Characteristics and Expected Stock Returns, working paper.

Hou, Kewei, Chen Xue, and Lu Zhang, 2015, Digesting Anomalies, *Review of Financial Studies* 28, 650–705.

Lewellen, Johnathan, 2015, The Cross-section of Expected Stock Returns, *Critical Finance Review* 4, 1–44.

1. We calculate the correlation between predictors and returns each month and take the average within each year. [↑](#footnote-ref-1)
2. We do not report results for forecast combination, because each variable enters into regression separately and we cannot directly compare their *relative* importance. [↑](#footnote-ref-2)