MF850 Final Project

Prediction of One-Month Stock Return

Jing Li - E1 Section
Qing Chen - E2 Section
You Xie - E2 Section
Jie Zhu - E2 Section
Haomiao Yu - E1 Section
Mingqian Xu - E2 Section

Questrom School of Business
December 16, 2019

Contents

1 Introduction

1.1 Goal of Our Project

The prediction of stock returns is one of the most important processes for investors to make investment decisions. One way to perform stock return forecasting is to use machine learning methods. In this project, our two main goals are to use machine learning tools to construct and train forecasting models that can accurately predict the one-month stock return of a company, as well as the grow-fall indicator. For our first task, we construct regression models to obtain exact prediction values; for the second task, we construct classification models to decide whether a stock will grow or fall in the future.

1.2 Data and Methodology

With monthly stock return data and monthly characteristics of companies and U.S. stock market during the time span between December 2014 and October 2018, we are required to make stock return forecasts for U.S. companies over the course of a month after October 2018 and before November 2019.

First of all, we decide to consider our data as cross-sectional data instead of panel data for the reason that in most cases, investors are more concerned about how the cross-section of stock returns are affected by fundamental and market factors such as company size, earnings–price ratio and market volatility.

We conduct both linear methods and nonlinear methods and then adopt the better ones. Considering the analysis above, we will discuss in detail about three parts, i.e. data processing, prediction of stock price (LASSO and Neural Network) and prediction of up and down (Logistic Regression and Random Forest), for the rest of our report.

2 Data Processing

2.1 Feature Classification

First and foremost, we differentiated the discrete and continuous variables in terms of the description and the unit type of the indicators and also their given value. In our observation, all data could be processed together step by step as continuous variables, except compid and industry indicators. "Compid", as an anonymized company identifier, which has no effect on our forecasts of stock returns in our model assumption, was discarded before the end of our data processing. "Industry" provides the industry of the firm and we planned to add industry dummies into our variables data. However, there are 146 different industries

(dummy variables) given by this indicator. Too meticulous classification brought us a large amount of new variables and resulted in instability of our regression. Due to the reasons mentioned above, this indicator was dropped with "Compid" and "Date" before variable selection.

2.2 Data Shifting

Since the columns with a postfix "_end" and with a postfix "_start" illustrate whether a column data entry is available at the end or start of a month respectively. Meanwhile, we can not use the end month(future) data as known information to do forecasts together with the start month data. We shifted the column data with a postfix "_end" for the current month upward as the start data for the next month, for our goal to use the start of a month data to construct predictions of the return value and direction at the end of the month. At the same time, we named our shifted column data the same as before, but replaced the "_end" postfix with the "_start" instead for easy recognition. After shifting all features marked with the postfix "_end" , we have turned all our variables into lagged variables which are always available.

2.3 Removal of "Size Effect"

Similar to the original phenomenon of size effect that the small cap shares to outperform large caps, the characteristics of companies may vary widely due to differences in their sizes. We have to remove such a kind of more generalized "size effect" in order to improve the comparability between companies and to obtain more meaningful features. Thus, we chose 50 features listed as below:

```
"assets_start", "debtusd_start", "capex_start", "cashnequsd_start", "cor_start", "consolinc_start",
"depamor_start", "deposits_start", "ebitdausd_start", "ebitusd_start", "ebt_start",
"equityusd_start", "ebt_start", "fcf_start", "gp_start", "intangibles_start",
"intexp_start", "invcap_start", "inventory_start", "investments_start", "liabilities_start",
"ncf_start", "ncfbus_start", "ncfcommon_start", "ncfdebt_start", "ncfdiv_start",
"ncff_start", "ncfi_start", "ncfinv_start", "ncfo_start", "ncfx_start",
"netinc_start", "opex_start", "opinc_start", "payables_start", "ppnenet_start",
"prefdivis_start", "retearn_start","revenueusd_start","sbcomp_start","sgna_start",
"sharesbas_start","shareswa_start","shareswadil_start","tangibles_start","taxexp_start",
"workingcapital_start", "rnd_start", "taxassets_start","taxliabilities_start")
```

Figure 1: Features with "Size Effect"

Then, we divided these column data by each company's market capitalization to do data neutralization based on its size.

2.4 Normalization

In order to avoid the influence brought by features' different ranges and observations' different time spans, we normalized all data according to the same "Date" according to the following method:

$$Scaled\ X_{t,j}^{i} = \frac{X_{t,j}^{i} - \overline{X_t^i}}{s_t^i}$$

where $X_{t,j}^{i}$ denotes the value of the i^{th} feature's j^{th} observation at time t; $\overline{X_t^i}$ and s_t^i respectively denote the mean value and the standard deviation of the i^{th} feature at time t.

2.5 Feature Crosses

In this step, we tried to add more variables that may be statistically significant,considering cross effect, a phenomenon in which two or more effects are coupled. Therefore, we multiply each other column data one by one, except "Date", "compid", "Industry", "RETMONTH_end" to find new variables that contribute much to our prediction of stock value.

3 Stock Return Prediction

3.1 Feature Selection

The feature selection part was proceeded as follows:

We took all the candidate variables and put them in the lasso regression model to help us do the variable selection. We now had 3165 candidate variables after data aggregation and transformation. We would like to select approximately 20 effective variables from all the candidate variables. To achieve this, we set the hyperparameter, λ, as $\exp^{-6}$, and we obtained 19 selected features. To further understand and visualize the corresponding relationships between our selected features and the response return, we did a feature plot, part of which is listed below.

```
("depamor_start"    ,        "fcf_start"         ,       "ncfo_start"     ,
 "netinc_start"  ,          "opinc_start"      ,        "retearn_start"     ,
 "rnd_start"      ,         "sbcomp_start"  ,           "sgna_start"        ,
 "retmonth_spx_start" ,     "capex_start_ncfx_start"  , "depamor_start_fxusd_start",
  "fcf_start_fxusd_start" ,  "fxusd_start_opinc_start"  , "fxusd_start_retearn_start",
 "fxusd_start_sgna_start",   "gp_start_sps_start"     ,  "rnd_start_sbcomp_start" ,
 "tbvps_start_volume_start" )
```

Figure 2: Features Selected by LASSO Regression

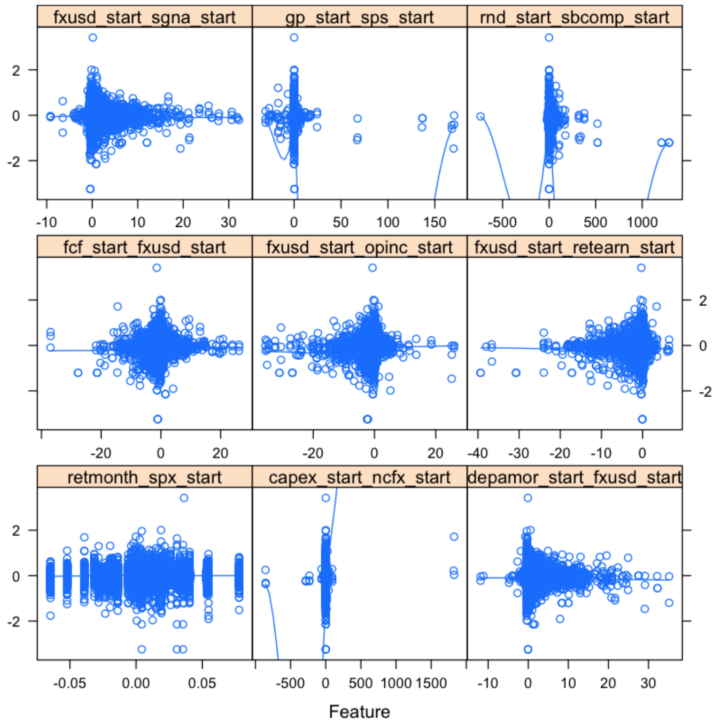

Figure 3: Selected Features and Target Variables

From the feature plot of our selected variable, we detected highly non-linear relationships between these variables and the response, which indicated that a generic linear model was not suffice to predict the response with our selected variables. Thus, we need turn to some non-linear model.

Besides, we plotted the correlation matrix of our selected variables and did a linear combination test, which is shown below:

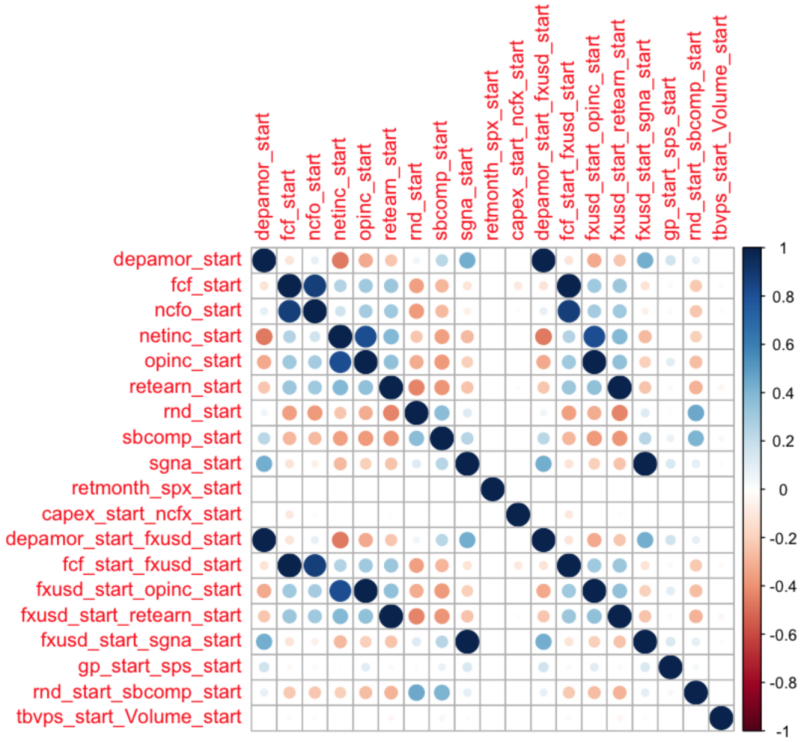

Figure 4: Correlation Matrix of Selected Variables

Both linear combination test and the correlation matrix plot indicated that our selected variables were not suffering from severe linear dependence among each other, which validated the usefulness of lasso selection scheme.

```
In [21]: comboInfo <- findLinearCombos(proc_data)
         comboInfo

$linearCombos

$remove
NULL
```

Figure 5: Results of Linear Combination Test

3.2 Model Development and Justification

We employed Neural Network as our primary stock return forecast model. Neural Network is a relatively flexible framework that can address non-linearity and additivity issues that generic linear regression model can't. The model training part was proceeded as follows:

3.2.1 Tuning Hyperparameter

From Introduction to Neural Networks for Java (second edition) by Jeff Heaton:

Problems that require two hidden layers are rarely encountered. However, neural networks with two hidden layers can represent functions with any kind of shape. There is currently no theoretical reason to use neural networks with any more than two hidden layers. In fact, for many practical problems, there is no reason to use any more than one hidden layer.

Empirically, Neural Network with one hidden layer is enough to address our problem. Thus the main hyperparameter we need to choose is the number of nodes in the hidden layer. From geometric pyramid rule of Masters (1993), the number of nodes should be around $\sqrt{19}$. And we tuned the number of nodes by 8-fold cross validation through the parameter grid (2, 3, 4, 5). The number of nodes with the minimum MSE through cross validation was 3 (the cross validation and tuning hyperparameter was done through the "caret" package in R).

3.2.2 Model Training

Once we have determined the hyperparameter, we employed the "neuralnet" function in package "neuralnet" to train our model. We split the whole sample data into training data and testing data by 80/20. And we obtained an in-sample R^2 of 2.72% and out-of-sample R^2 of 1.78%. More importantly, we obtained a ranked IC of 0.1377 between our prediction and the test sample return, justifying the soundness of our model. To obtain our final model for prediction, we used the whole data to train our model, and we visualized the neural network as below.

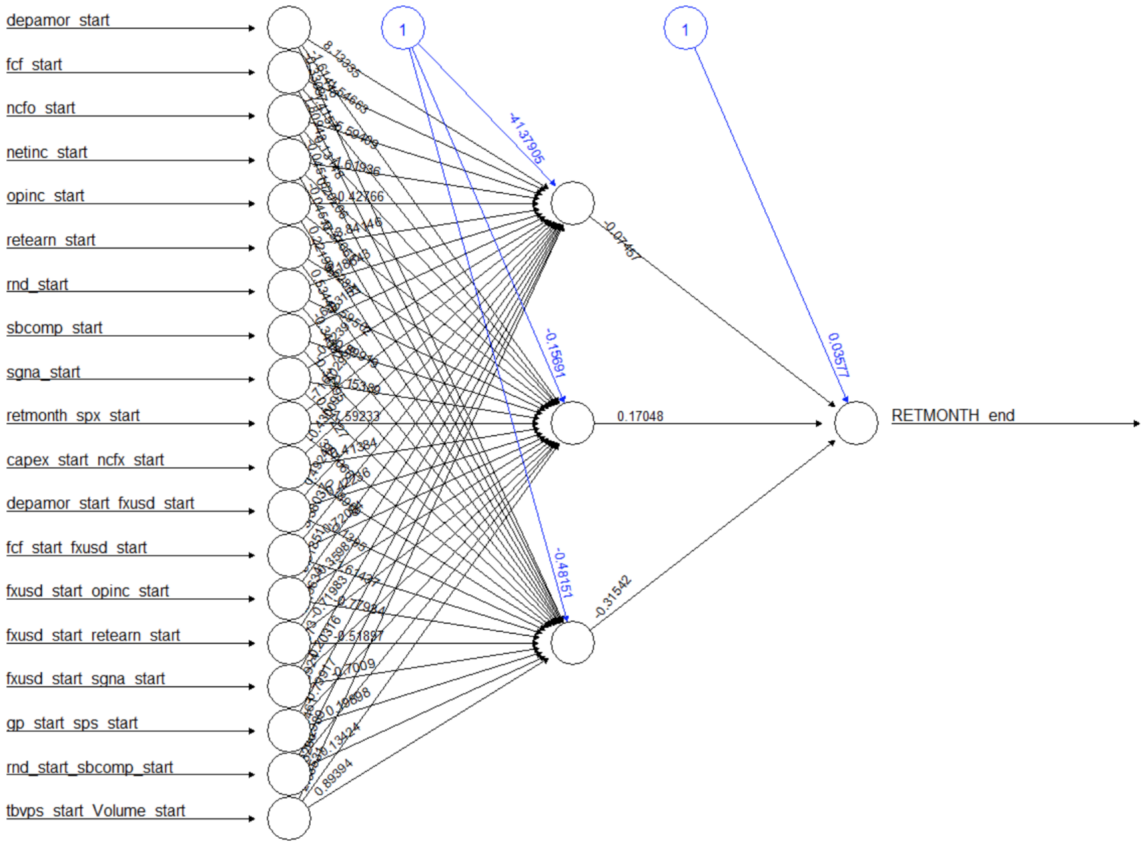

Figure 6: Visualized Neural Network

4 Stock Trend Prediction

The stock market series are dynamic and noisy in nature marketing, and the price movement is regarded as a random process with fluctuations. Here, our goal is to implement an approach to predict the future trend of stock prices either up or down over the course of the month. In this session, we will use a mix of machine learning tools to predict the movement starting with a relatively simple algorithm like logistic regression, and then move on to an advanced technique like random forests.

The data set contains data that span between December 2014 and October 2018. We randomly choose 80% of the data as our training data and regard the remaining 20% as our testing data. As shown in the following table, there are about 50% of data that has up and down trend both on training and testing data set, which indicates unbiased trend in our sample data.

Table 1: Training and Testing Data Sets

Trend	Up	Down
Training dataset	50.50%	49.50%
Testing dataset	50.41%	49.59%

4.1 Feature Selection

Before we start to run the prediction models, the most important thing is to do the variable selections. Given the provided features - market data and firm fundamentals, not all features are meaningful. The problem comes down to how many and which features to select considering the computation efficiency and prediction accuracy. We select features using the following steps.

4.1.1 Logistic regression on all features

Logistic regression model is very useful for binary classification problems. It is our first choice, and the regression model could help us to find features with the most importance in prediction.

$$l = \log_b \frac{p}{1-p} = \beta_0 + \sum_{i=1}^{n} \beta_i x_i$$

Therefore, our first step is to run logistic regression on all of the variables, including technical indicators and fundamental indicators. We expect to use the ranking of the variable importance to help us pick most significant features, and here we use the function called *varlmp*() in the *carpet* package to perform the variable importance. Variable importance is a very useful indicator in how much a model use the variable to make a relatively accurate prediction, and a variable with a high value in variable importance contribute more in the model.

4.1.2 Logistic regression on top 13 features

Based on our subjective judgment, we finally select top 13 indicators including 6 technical indicators and 7 fundamental indicators according to the importance which are listed as Table 2.

Table 2: Variable Importance

Technical Variables		Fundamental Variables	
Variable	Importance	Variable	Importance
sentiment_neutral	14.08	pe1	2.10
sentiment_bearish	14.08	sps	1.81
Adj_volumn	3.15	tbvps	1.35
retmonth_spx	3.78	gp	1.35
realized_vol_spx	3.17	fcfps	0.75
Close	1.94	pb	0.50
		divyield	0.07

The correlation plot of these 13 indicators is shown as below. Positive correlations are displayed in blue and negative correlations in red color. Color intensity is proportional to the correlation coefficients.

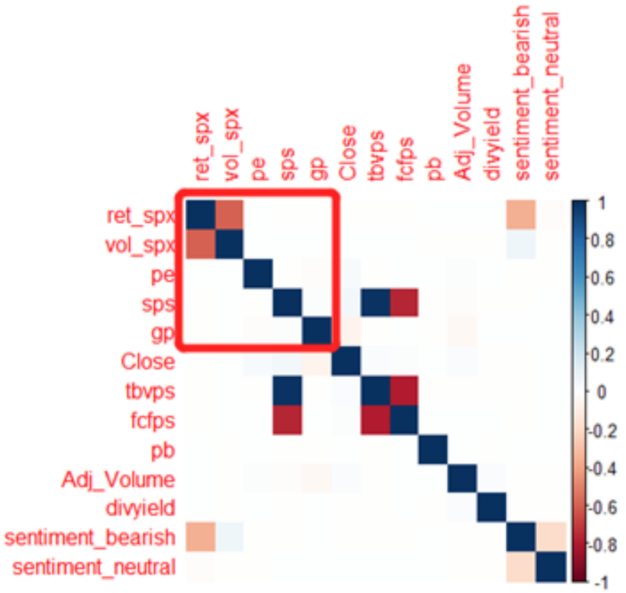

Figure 7: Correlation Matrix

We run regression on selected 13 variables again and compare the results in Table 2. We set 0.2 as threshold, which means any forecast that is greater than 0.2 is viewed as “up” otherwise “down”.

Table 3: Test Data Prediction Accuracy

Regression on all variables	52.38%
Regression on selected 13 variables	49.76%
Regression on top 5 variables	49.95%

4.1.3 Select Top 5 Features

One of the shortcomings of logistic regression is that it uses linear relationship for prediction and lose the non-linear relationships. Random forest is an alternative method for classification capturing non-linear relations. However, the computational cost is much higher. Therefore, we have to select limited top variables that can produce the largest prediction accuracy in the test data set.

To select the top five indicators with the most prediction ability, we delete the indicators one by one and see the prediction accuracy with the remaining 12 indicators. The greater the decrease in accuracy implies the greater the importance of the deleted indicators. As shown in Table 4. In this way, we select the top five indicators and the correlation plot is circled in the Figure1. To have a clear idea, the correlation matrix of the top 5 features shows in Table 5, these feature has little correlations with each other, indicating different dimensions of information.

Table 4: Prediction Accuracy of Indicators

Indicators	Accuracy without the indicator	Accuracy decrease
retmonth_spx_start	49.68%	0.27%
realized_vol_spx_start	49.70%	0.25%
gp_start	49.86%	0.09%
pe1_start	49.88%	0.08%
sps_start	49.92%	0.03%
tbvps_start	49.94%	0.01%
Close_start	49.94%	0.01%
fcfps_start	49.94%	0.01%
pb_start	49.94%	0.01%
Adj_Volume_start	49.95%	0.00%
divyield_start	49.97%	-0.02%
sentiment_bearish_start	49.99%	-0.04%
sentiment_neutral_start	49.99%	-0.04%

Table 5: Correlation Matrix of Top 5 Features

	retmonth_spx_start	realized_vol_spx_start	pe1_start	sps_start	gp_start
retmonth_spx_start	1.0000	-0.5914	0.0019	0.0019	0.0011
realized_vol_spx_start	-0.5914	1.0000	-0.0011	-0.0028	0.0007
pe1_start	0.0019	-0.0011	1.0000	-0.0008	-0.0111
sps_start	0.0019	-0.0028	-0.0008	1.0000	0.0186
gp_start	0.0011	0.0007	-0.0111	0.0186	1.0000

4.2 Random Forest Forecast

Utilizing the logistic regression method, selecting the top 5 features generates 49.95% out-of-sample prediction accuracy listed in the last row of Table 2, higher than the accuracy using 13 variables.

4.2.1 Default parameter settings

Based on the top 5 features, we start to build up the random forest model to predict if a stock will go up or down in value in the following month, and we believe that the advanced algorithm will outperform the logistic regression model, that is, random forest method will give us higher prediction accuracy than 49.95% due to the ability to capture the non-linear relations built under various trees.

Under the training data set, we perform a random forest algorithm using default settings in R, 500 trees and one variable randomly sampled as candidates at each split and maximum number of terminal nodes trees in the forest can have, that is, "ntree = 500, mtry =1".

In this task, the monthly return data as our target output, if the predicted value is positive, we set "y=1" and it is regarded as going up, otherwise, we set "y=0" and predict that the trend falls over a course of months.

This results in accuracy of 75.98% for training data set and accuracy of 62.31% for the testing data set. This prediction based on random forest model is far higher than the one obtained from logistic regression, which is 49.95%.

4.2.2 Distribution of Tree Minimal Depth

The minimal depth of a variable in a tree stands for the depth of the node which splits on that variable. The value implies the amount of observations that are divided into groups based on this variable. If the value is low, then the amount will be large. Given the figure of Distribution of Minimal Depth and its Mean, we could see the importance of the features we choose.

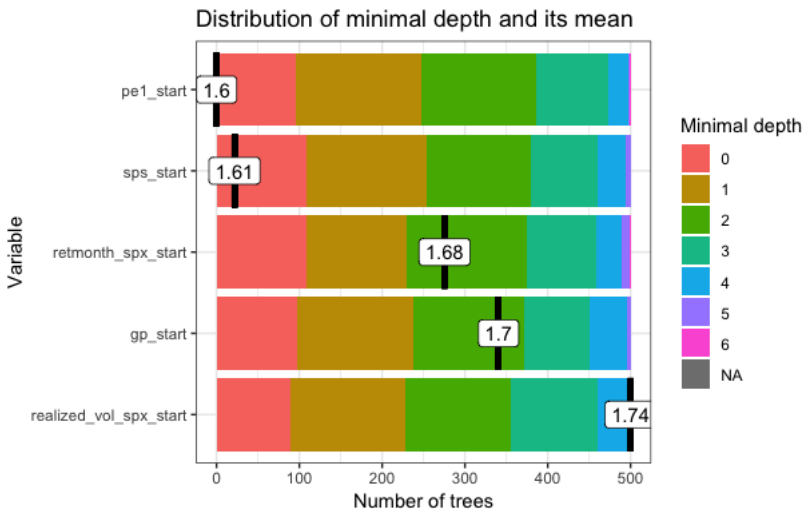

Figure 8: Distribution of Minimal Depth and its Mean

4.2.3 Parameter Selection

Considering the computational efficiency, we plan to test the prediction accuracy under different values of parameters including number of trees (ntree), number of variables randomly sampled as candidates at each split (mtry) and maximum number of terminal nodes (maxnodes).

We plot a figure with x-axis number of trees, and y-axis errors. From the Figure3, we figure out that when the number of trees is larger than 200, the mean of squared residuals (errors) become stable around 0.019.

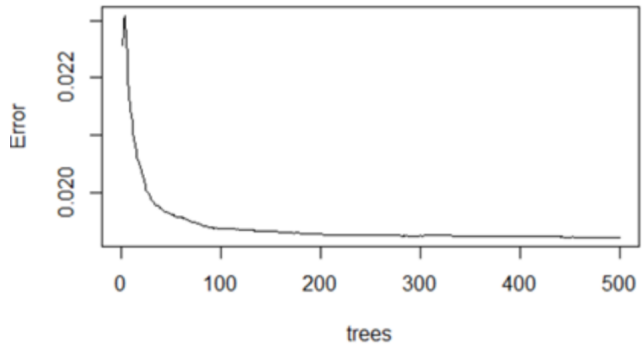

Figure 9: Tree number and Errors

We set "ntree=200" and maximum number of nodes as default and start to test various values of "mtry" as 1, 2, 3. The testing result is listed below as Table 4.

Table 6: Prediction Accuracy with "ntree=200" and "maxnodes" as Default

mtry	1	2	3
Training	68.22%	83.49%	83.30%
Testing	62.32%	61.45%	61.42%

Increasing "mtry" leads to the increase in the accuracy of training, but decrease in the accuracy of testing. The increase in the difference of accuracy implies a more serious situation in overfitting. Therefore, we choose "mtry = 1" to minimize the overfitting problem and result in the highest accuracy for various data sets.

Next, the value of "maxnodes" can be maximum possible, so we want to locate an optimal value so that we could maximize the time consuming. Setting "ntree = 200" and "mtry =1", we test different values of "maxnodes" as 100, 200, 300, 500, 800, 1000, 1500, 2000. By the following figure, we notice that the accuracy for either training or testing is in a growth trend with the increase in maximum nodes, but the accuracy is a bit lower under maximum 2000 nodes than that under 1500.

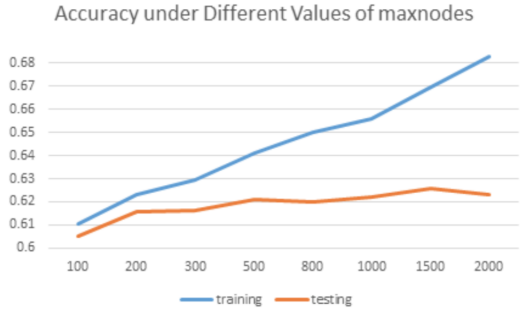

Figure 10: Accuracy Under Different Maxnodes

Therefore, we choose "maxnodes = 1500", the accuracy of testing is 62.57%, which is higher than result under default setting. Therefore, the parameters we finally choose are "ntree = 200, mtry =1, maxnodes = 1500". The prediction accuracy under this setting is as below :

		Data	
		Up	Down
Predict	Up	36.43%	23.45%
	Down	13.98%	26.14%

Figure 11: Accuracy of Testing Data Set Under Final Setting

5 Reference

1. "R Random Forest Tutorial with Example - Guru99." 28 Nov. 2019, https://www.guru99.com/r-random-forest-tutorial.html. Accessed 13 Dec. 2019.

2. "Feature selection techniques with R - Dataaspirant." 15 Jan. 2018, https://dataaspirant.com/2018/01/15/feature-selection-techniques-r/. Accessed 13 Dec. 2019.

3. "Stock market trend predictions using random forests - RPubs." 15 Aug. 2018,
https://rpubs.com/raaraa/412512. Accessed 13 Dec. 2019.

4. "Visualize correlation matrix using correlogram - Easy ... - STHDA."
http://www.sthda.com/english/wiki/visualize-correlation-matrix-using-correlogram. Accessed 13 Dec. 2019.

5. "Feature Selection methods with example (Variable selection ...." 1 Dec. 2016,
https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/. Accessed 13 Dec. 2019.

6. "Understanding random forests with randomForestExplainer."
https://cran.rstudio.com/web/packages/randomForestExplainer/vignettes/randomForestExplainer.html. Accessed 13 Dec. 2019.