Classification using Random Forest in R

Team MAKE R GREAT AGAIN

Danish Chappanga Thottathil

Mownika Chalichama

Siddharth Kajampady

Sindhushree Sonnad Matt

Yoganand Guttikonda

11/30/2016

**TABLE OF CONTENTS**

Executive Summary…………………………………………………………………………….. 2

Multiclass Classification……………………………………………………………………….. 2

Decision Trees………………………………………………………………………………….. 3

Bootstrap Aggregating…………………………………………………………………………. 6

Random Forest Algorithm……………………………………………………………………... 7

About the Data Set……………………………………………………………………………... 8

Building Random Forest in R………………………………………………………………….. 9

Model Diagnostics……………………………………………………………………………...10

**Executive Summary**

Classification is a technique widely used in multiple fields of science, commerce, mathematics or even in one’s day to day basis. It increases visibility in the given information to make informed decisions. Product classification plays an important role in retail businesses. It helps marketers channel their efforts based on consumers’ purchasing behavior. A retail/e-commerce business can use these buying habits to design their marketing strategies for a clearly defined target audience. Therefore, application of analytics and machine learning techniques play a pivotal role in this segment of a retail business.

Our team researched and implemented one of the classification analysis methods: Random Forest, on a dataset provided by Otto Group, one of the biggest e-commerce companies based in Europe. The dataset was open source and was extracted from Kaggle. In this white paper, we introduce the concepts leading to Random Forest algorithm and present the implementation of Random Forest algorithm on the dataset.

**Multiclass Classification**

Multiclass classification refers to classification task with more than two classes. Multiclass classification assumes that each sample is assigned to one and only one label: It is often confused with multilabel classification where the labels are not mutually exclusive.

E.g.: Classify a set of images of animals which may be lions, dogs, or elephants.

An animal can be either be a lion or a dog but not both at the same time.

The different multiclass algorithms available are:

Naive Bayes, LDA (Linear discriminant analysis) and QDA (Quadratic discriminant analysis), Decision Trees, Random Forests, Nearest Neighbors

The underlying concept of the multiclass classification problem is to build multiple binary classifiers for each class. The two strategies involved in building this binary classifier are

* One - Vs – rest

This strategy involves in building one classifier for each class. Each classifier assigns positive to the class members and negative to the non-class members.

* One – Vs – One

In One – Vs – One (OVO), we train the data by building N(N-1)/2 classifiers, one classifier receives a sample pairs of classes. So, this classifier helps to distinguish between the classes and upon voting system the class which receives highest number of positives is predicted.

The above classification strategies are extended to solve the multi-classification problems using the different multiclass algorithms.

**Decision Trees**

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. It works for both categorical and continuous input and output variables. In this technique, we split the population or sample into two or more homogeneous sets based on most significant splitter / differentiator in input variables.

**Types of Decision Trees:**

Types of decision tree is based on the type of target variable we have. It can be of two types:

Categorical Variable Decision Tree: Decision Tree which has categorical target variable then it called as categorical variable decision tree.

Continuous Variable Decision Tree: Decision Tree has continuous target variable then it is called as Continuous Variable Decision Tree.

**How does a classification decision tree work?**

Classification trees are used when dependent variable is categorical. Multi classification trees are used when the dependent variable is categorical and has many levels of classification.

The decision of making strategic splits heavily affects a tree’s accuracy. The decision criteria are different for classification and regression trees.

Decision trees use multiple algorithms to decide to split a node in two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that purity of the node increases with respect to the target variable. Decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes.

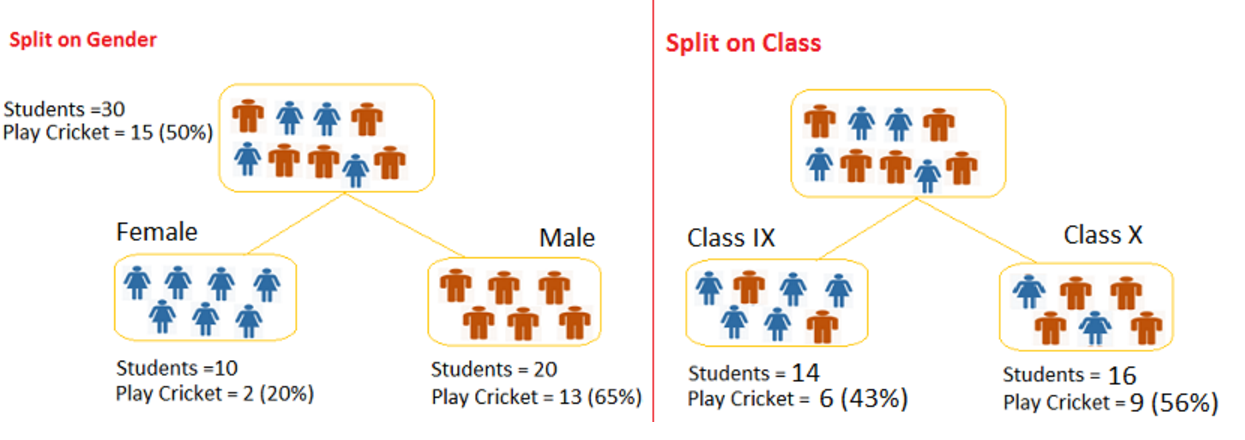
Generally, ID3 algorithm is used to build the decision tree for the dataset and GINI Index as a parameter to decide the splits in the node.

**Steps to Calculate Gini index for a split:**

Calculate Gini for sub-nodes, using formula sum of square of probability for success and failure (p^2+q^2).

Calculate Gini for split using weighted Gini score of each node of that split

Example: If we want to segregate the students based on target variable (playing cricket or not). In the snapshot below, we split the population using two input variables Gender and Class. Now, we want to identify which split is producing more homogeneous sub-nodes using Gini index.

Calculate, Gini for sub-node Female = (0.2) ^2+(0.8) ^2=0.68

Gini for sub-node Male = (0.65) ^2+(0.35) ^2 = 0.55

Calculate weighted Gini for Split Gender = (10/30)\*0.68+(20/30)\*0.55 = 0.59

Similar for Split on Class:

Gini for sub-node Class IX = (0.43)^2 + (0.57)^2 = 0.51

Gini for sub-node Class X = (0.56)^2+(0.44)^2=0.51

Calculate weighted Gini for Split Class = (14/30)\*0.51+(16/30)\*0.51 = 0.51

Above, you can see that Gini score for Split on Gender is higher than Split on Class, hence, the node split will take place on Gender.

**Advantages of Decision trees:**

* Easy to Understand
* Useful in Data exploration
* Less data cleaning required
* Data type is not a constraint
* Non-Parametric Method

**Disadvantages of Decision trees:**

* Over fitting
* Not fit for continuous variables

**Bootstrap Aggregating (Bagging)**

Bootstrap aggregating (abbreviated as bagging), is a machine learning meta-algorithm (An algorithm which is used to build/manipulate another algorithm), which has been designed to improve the accuracy and decrease the variance of the predictions for statistical classification and regression methods. To understand bagging, we will first look at an important technique known as bootstrap.

Bootstrap is a powerful statistical method for estimating a quantity from a sample of data. This method uses random sampling of data and with replacement. Bagging is an application of the bootstrap procedure to a high variance machine learning algorithm, typically decision trees.

Let’s assume we have a sample dataset of 1000 observations and we are using a decision tree algorithm. Bagging of the tree algorithm would work as follows.

1. Create many (e.g. 100) random sub-samples of our dataset with replacement.

2. Train a tree model on each sample.

3. Given a new dataset, calculate the average predictions from all different trees for a quantitative prediction or for a categorical prediction, take the most frequent class of the predictions from each model.

The only parameters when bagging decision trees is the number of samples and hence the number of trees to include.

Just like decision trees, bagging can also be used for classification, regression problems and subset selection in linear regression.

Random forest is one such type of bagging algorithms.

**Random Forest Algorithm**

Random forest can be described as ‘bootstrapping algorithm with Decision tree model’. This is one of the most popular machine learning algorithms used for classification, and therefore widely used for categorical response variables. The basic idea in Random Forest algorithm is to first develop multiple decision trees based on random selection of data and random selection of variables, which forms a ‘random forest’. A final prediction of the response variable is then made for each observation in the data. The final prediction is a function of each prediction made by the decision trees, typically the mode or mean of the different predictions.

To explain the approach of random forest in layman terms, if you are uncertain about watching a certain movie, instead of taking an opinion from a single person you consult with multiple people and make an informed decision based on majority votes.

Random forest algorithm is an ensemble approach where the multiple decision trees who are individually weak learners, come together to form a strong learner.

**Working of the algorithm:**

Consider a supervised setup with the observed data points:

D = (x1,y1)…….(xn,yn)

We create bootstrap sub samples Di from D, where i = 1, 2,…p using bagging approach.

We then construct trees Ti from Di which constitutes our random forest.

The decision trees Ti are constructed such that, at each node we choose a random subset of features(variables) and only consider splitting of trees on those features in the subset. This is called random subspace.

E.g., for p number of trees if there are m number of features in a subset, when we are considering the splits at each node we will only consider the best split in the m features

We then classify the observations by considering the majority votes across all the trees in the random forest.

**Advantages of Random Forest over Decision tree models:**

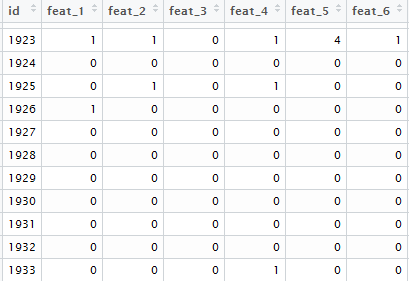
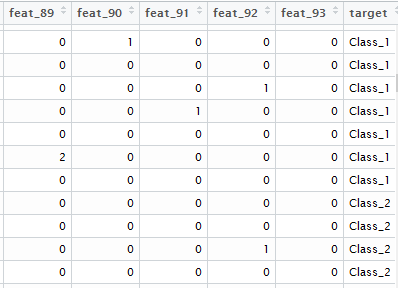
Decision tree has complications of over-fitting and ignorance of a variable in case of small sample size and large p-value. This is overcome in the method of random forests. Random forest also gives much more accurate predictions.

Random forest can handle large data set with higher dimensionality. It can handle thousands of input variables and identify most significant variables so it is considered as one of the dimensionality reduction methods.

Although random forest comes at the expense of some loss of interpretability, it generally boosts the performance of the final model.

**About the Data Set**

The dataset consists of 93 product features of Otto group, one of the biggest e-commerce companies. Each product belongs to a certain product group. The objective is to build a predictive model that accurately classifies these products into correct categories based on the product features. All the products need to be classified to either one of the nine product categories present in the data.

 . . . . 

**Building Random Forest in R**

The R package “randomForest” is used to create and analyze random forests. It is based on Breiman’s random forest algorithm for classification.



formula: Describes the model to be fitted. Here the target variable the product classification that need to be predicted and the predictors are the 93 features present the dataset.

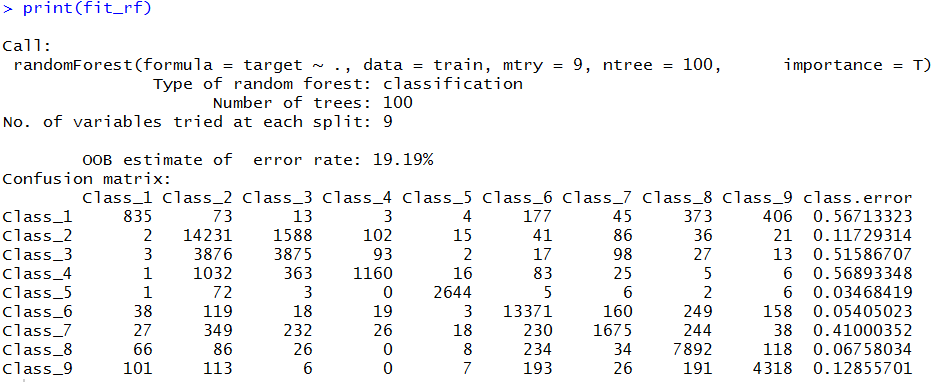
data: The dataframe containing the variables (93 features) that need to be trained by the random forest model

mtry: Number of variables randomly sampled at each split

ntree: Number of trees to grow

importance: Should the importance of the variables be assessed?

Initially, a random forest model is built with 100 trees and using 9 random variables at each split. The output is as follows:

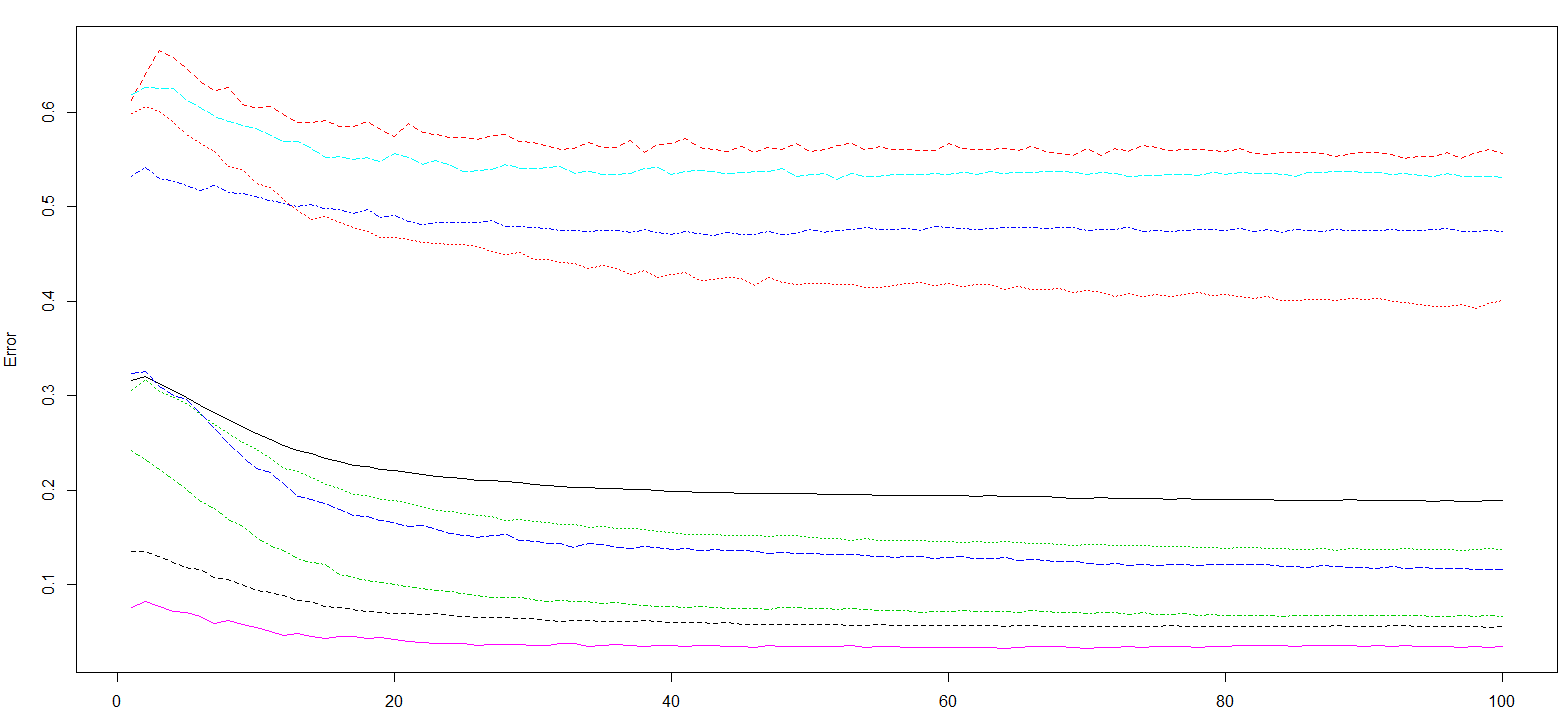


In random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error. It is estimated internally, during the run. Each tree is constructed using a different bootstrap sample from the original data. About one-third of the cases are left out of the bootstrap sample and not used in the construction of the kth tree. Put each case left out in the construction of the kth tree down the kth tree to get a classification. In this way, a test set classification is obtained for each case in about one-third of the trees. At the end of the run, take j to be the class that got most of the votes every time case n was OOB. The proportion of times that j is not equal to the true class of n averaged over all cases is the OOB error estimate.

The class error in the output shows that class 1,3,4 and 7 have high error rate and are not accurately classified by the model.

**Model Diagnostics**

The plot function shows the decrease in error with respect to the number of trees built. Higher the number of trees, higher the computational time for building the model. But typically, the higher number of trees reduce the error as error gets averaged among the trees.



Thus, the optimal number of trees can be identified from the decrease in the error with respect to increase in number of trees.

**tuneRF Function**

The “tuneRF” function is used to tune the parameters of the random forest model. Using this function, we can, starting with the default value of mtry, search for the optimal value (with respect to Out-of-Bag error estimate) of mtry for randomForest



x: Data frame of predictor variables (93 feature variables)

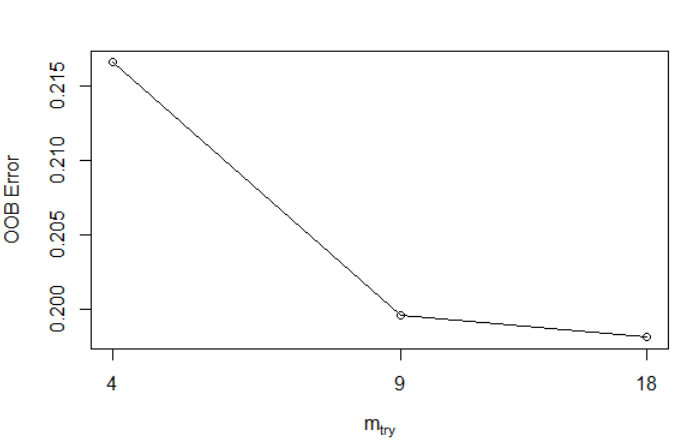
y: Data frame of response variable (target)

mtryStart: Starting value of mtry

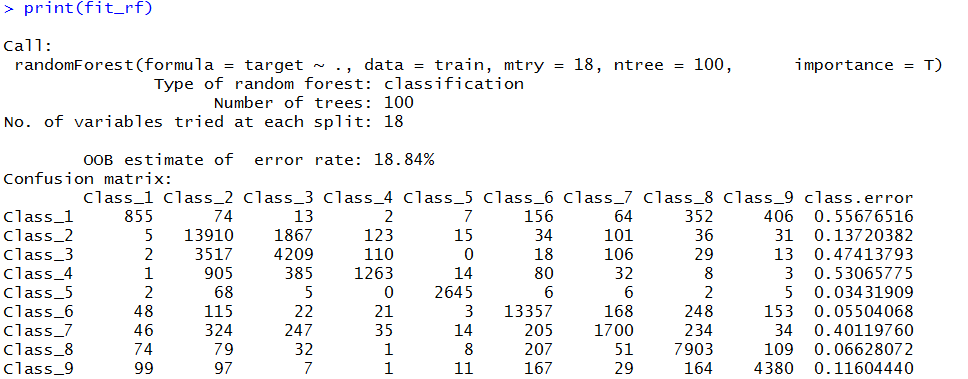
stepFactor: Increment to mtry at each iteration

improve: the (relative) improvement in OOB error must be by this much for the search to continue

The function provides a plot of OOB error rate v/s the mtry and thus helps to identify the optimal number of predictor variables that need to be sampled for building trees for each bootstrap sample.



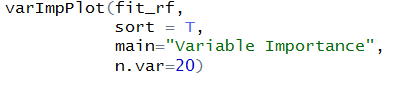
The random forest can thus be re-run using mtry = 18 decrease the OOB error rate and further increase the accuracy of the model.



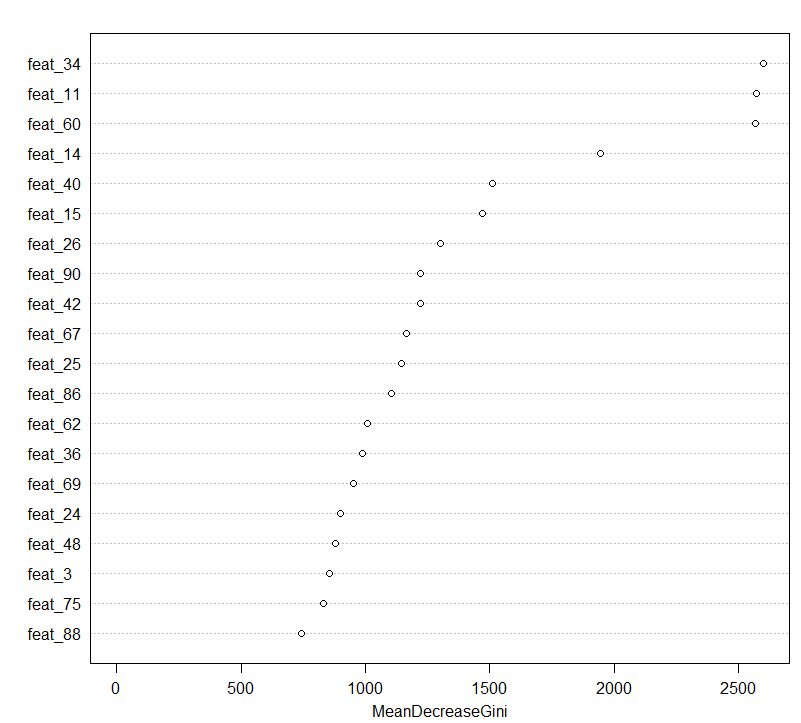
Rerunning the model shows that the OOB error rate has improved compared to the previous model.

**Variable Importance Plot:**

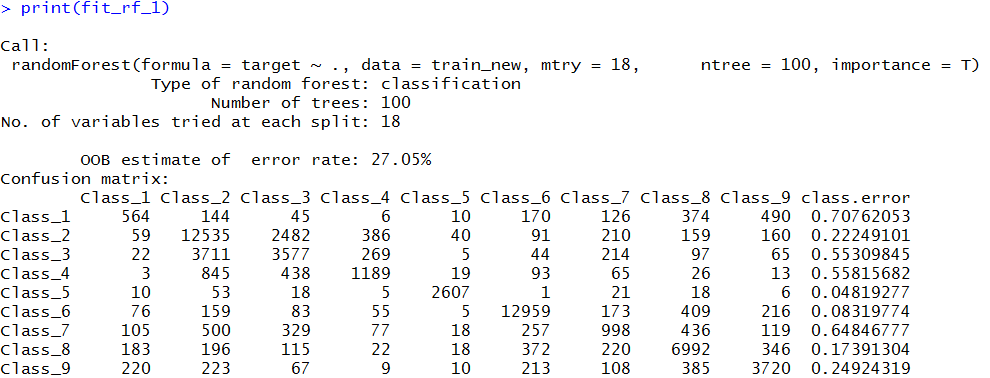
Variable importance is calculated based on mean decrease in GINI index. Gini index says, if we select two items from a population at random then they must be of same class and probability for this is 1 if population is pure.



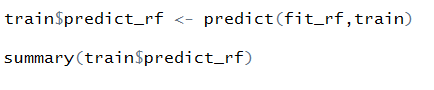
The above varImpPlot function shows the top 20 variables based on their mean decrease in GINI score.



The model can be rerun using the most influencing features to decrease the complexity. We ran the model the using the top 20 most important variables based on the mean decrease in GINI score. But it was observed that the OOB rate increased to 27%

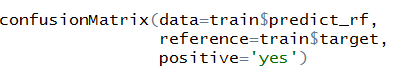


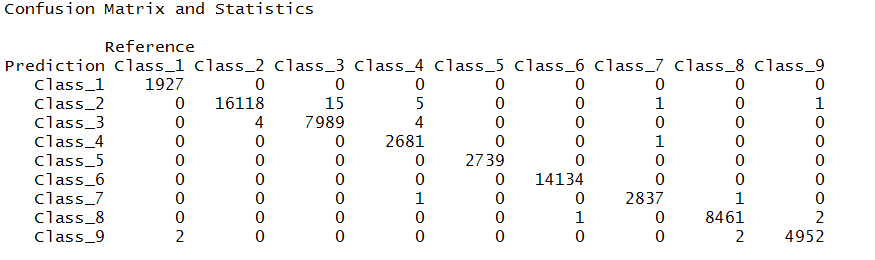
Finally, we selected the model keeping all the feature variables to predict the target. However, variables can be removed based on decreasing GINI score decrease the complexity and computation time if the OOB rate is not significantly affected.

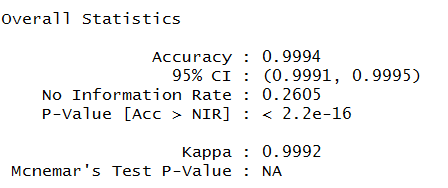


Using the predict function, the target class can be predicted.

Confusion matrix function from the caret package is used to calculate the overall statistics of the model







From the confusion matrix generated, we can observe the overall statistics at a Class level. The accuracy and other statistics are great, which explains further why random forest is widely popular.