**A Short Introduction To The Random Forest Ensemble Model:**

David Goody

1. Random forest is a machine learning algorithm that creates a large number of decision trees[[1]](#footnote-1) and then combines their results to create an overall result. A key aspect of the approach is that the methodology starts each stage by selecting a subset of the cases[[2]](#footnote-2) (typically two-thirds of the overall dataset) and then selecting a subset of the variables. It then fits the best decision tree for that subset of cases and variables. This process is repeated hundreds of time to give a range of simple and different datasets.
2. The output from the bagging algorithm takes the average result of all of these different decision trees to an overall prediction of an event occurring or a dataset belonging to a particular class (typically a percentage from 0% to 100%). By consulting a large range of different models the output can be incredibly fine-tuned. By fitting on subsets of the data the chance of over-fitting is reduced but not removed (the approach is known as cross-validation).
3. The accuracy of a random forest model is calculated by applying each of the decision trees to the third of the dataset that they wasn’t used to fit tree. By looking at how accurately the model makes predictions on this “new” data we can calculate the out of bag error estimate. This error estimate is unbiased.
4. This error method can be extended to judge the importance of individual variables through a method known as permutation importance. This works by looking at how much better an individual variable is than random guessing within each decision tree it appears in. Compute the standard errors and z-scores for each of these to quantify the importance and statistical significance of each of the variables.
5. An alternative way to judge variable importance is by applying the Gini coefficient. The Gini coefficient is a measure of inequality – frequently used to judge the gap between rich and poor in countries. In the context of a random forest model each time we split the data using a variable we want to create a more starkly segmented dataset with the “yes” and “no” outcomes split into different groups. By measuring the variation in inequality within the dataset using the Gini coefficient we can quantify how effectively each of the variables splits the data into starkly defined categories. This is typically presented by saying if we removed variable x the inequality (Gini coefficient) would reduce by y on average. The bigger the reduction – the move important the variable is. Permutation importance and Gini importance typically give consistent outcomes.

**Comparison Of Decision Tree Methodologies**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Decision Trees** | **Bagging** | **Boosting** | **Random Forest** |
| **No. of trees** | One | Many | Many | Many |
| **Weighting of trees** | - | Equal | Variable | Equal |
| **Variables in scope** | All | All | All | Randomly selected for each tree |
| **Cases used for fitting** | All training data | Random selection without replacement | All training data | Random selection with replacement |

**Further Reading**

Leo Breiman and Adele Cutler - <http://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm>

Wikipedia - <http://en.wikipedia.org/wiki/Random_forest>

**Annex A – Applying the random forest model in R**

#EXAMPLE OF APPLYING RANDOM FOREST IN R

#This example uses is based on the passenger list from the Titanic. We are trying to

#predict who survived and who died based on their age (Age), the fare they paid (Fare)

#gender (Female), embarkation point (South), how many siblings/spouses in each party (SibSp)

#how many parents/children in each passengers party (Parch)

#The dataset is used is publically available. It is the basis of a machine learning

#example on Kaggle (https://www.kaggle.com/c/titanic-gettingStarted).

##########################################################################

#SET-UP

#Install the random forest package

install.packages(randomForest)

require(randomForest) #For random forest (ensemble) models

#Set the working directory where you will be loading and saving data from

setwd("/Users/datascientist3/Desktop/R Packages/Kaggle Titanic")

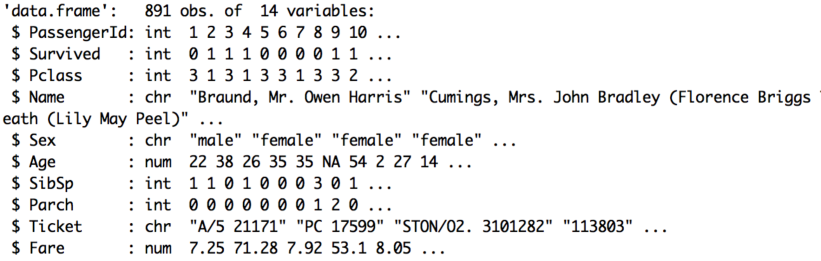
#Load Titanic dataset that we will train the data on

Titanic.Train <- read.csv("titanic\_train\_kaggle.csv", stringsAsFactors=FALSE)

#HAVE A LOOK AT THE DATA YOU ARE GOING TO USE

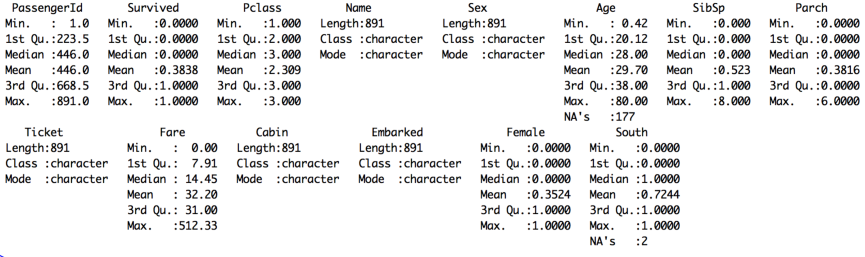
#Review structure of the data (shows field type and outputs form first few cases)

str(Titanic.Train)



#Look at a summary of the data (shows distribution of data for numeric fields)

summary(Titanic.Train)



#Flag survived variables as a factor (yes/no) rather than a continuous numeric variable

Titanic.Train$Survived <- factor(Titanic.Train$Survived)

#Convert text fields to dummy variables with 1/0 flags for use in equation

#Convert gender field into a 1/0 flag for female status

Titanic.Train$Female[Titanic.Train$Sex=='female'] <- 1

Titanic.Train$Female[Titanic.Train$Sex=='male'] <- 0

#Convert embarkation point 1/0 flag for those boarding at Southampton

Titanic.Train$South[Titanic.Train$Embarked=='S'] <- 1

Titanic.Train$South[Titanic.Train$Embarked!='S'] <- 0

#Over-write NA value for age with average as Lasso package can handle NA values

Titanic.Train$Age[is.na(Titanic.Train$Age)] <- 29.70

#FIT THE MODEL

#Apply Random Forest model to predict whether passengers survived based on the 6 variables

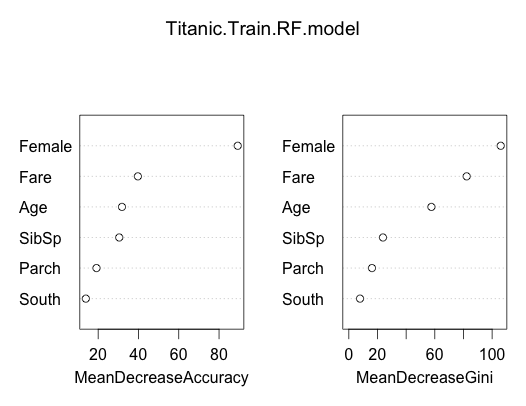
Titanic.Train.RF.model <- randomForest(Survived ~ Female + Age + SibSp + Parch + Fare + South, #Equation

data=Titanic.Train, #Data source

importance=TRUE) #Keep data on variable importance

#Show the importance of each of the variables

varImpPlot(Titanic.Train.RF.model)



#APPLY THE MODEL

#We can take the model we've fitted and apply it to our training data

#This will give us a percentage value for each record.

#This is our prediction of how likely they were to survive

#The predict function is of the form predict([model name], [output required])

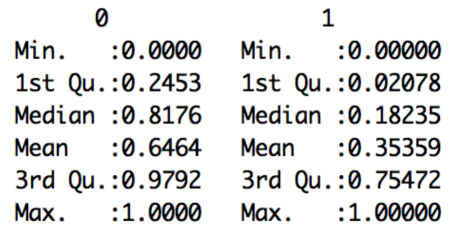
#In this case probability gives us the % chance that they survive according to our model

Titanic.Train.RF.Preds <- predict(Titanic.Train.RF.model,type="prob")

#Look at a summary of our predictions. The % chance of survival varies from 5% to 100%

summary(Titanic.Train.RF.Preds)

**Did Not Survive Survived**



#Match the predictions back onto the main dataset to create a new file

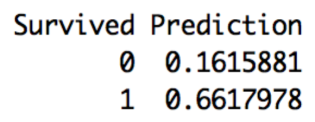
Titanic.Train.With.Predictions <- data.frame(Titanic.Train, Prediction = Titanic.Train.RF.Preds[,2])

#Show the average prediction chance of survival for those who did live or die (Survived = 1 or 0)

#Model gives an average prediction of 16% for those who died and 66% for those who lived

aggregate(Prediction ~ Survived, data=Titanic.Train.With.Predictions,

FUN=function(x) {sum(x)/length(x)})



#Look at an individual result - the fifth case in the dataset

#Mr William Henry Allen did not survive (Survived = 0)

#Our model gave him a 0% chance of survival. Not very good odds! The model is doing OK here.

Titanic.Train.With.Predictions [5,] #Square brackets mean take 5th row (then comma) show all columns



#We can save our results to a csv file

write.csv(Titanic.Train.With.Predictions, file="Titanic.Train.With.Predictions.RF.csv")

1. This paper assumes an understanding of decision trees. Please read to the “Short Introduction To Decision Trees” paper before reading this paper. [↑](#footnote-ref-1)
2. The subset of cases is created with replacement. This means the same results may be selected multiple times when fitting the model. [↑](#footnote-ref-2)