Random Forest Method II

Learn How Ensemble Learning Can be Used for Predictive Modeling

Contents

Case Study – Employee Churn Model

Background

- A company has comprehensive database of its past and present workforce, with information on their demographics, education, experience and hiring background as well as their work profile. The management wishes to see if this data can be used for predictive analysis, to control attrition levels.

Objective

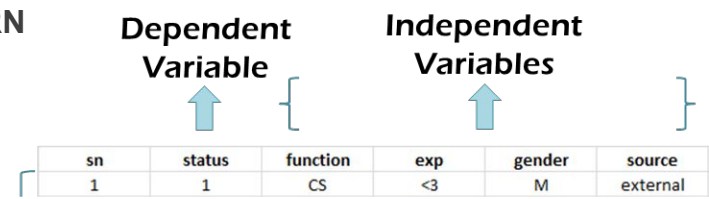
- To develop an Employee Churn model via Random Forest Method

Available Information

- Sample size is 83
- **Gender, Experience Level** (<3, 3-5 and >5 years), **Function** (Marketing, Finance, Client Servicing (CS)) **and Source** (Internal or External) are independent variables
- **Status** is the dependent variable (=1 if employee left within 18 months from joining date)

Data Snapshot

EMPLOYEE CHURN DATA



Columns	Description	Type	Measurement	Possible values
sn	Serial Number	-	-	-
status	= 1 If the Employee Left Within 18 Months of Joining = 0 Otherwise	Integer	1,0	2
function	Employee Job Profile	Character	CS, FINANCE, MARKETING	3
exp	Experience in Years	Character	<3,3-5,>5	3
gender	Gender of the Employee	Character	M,F	2
source	Whether the Employee was Appointed via Internal or External Links	Character	external, internal	2

Random Forest in R

```
# Installing Package, Importing and Readying the Data

install.packages("randomForest")
library(randomForest)

empdata<-read.csv("EMPLOYEE CHURN DATA.csv",header=T)

empdata$status<-as.factor(empdata$status)
```

Since it's a classification problem, dependent variable is converted to factor variable using **as.factor().**

Random Forest in R

```
# Run Random Forest
churn_rf<-randomForest(status~function.+exp+gender+source,
data=empdata,
                       mtry=2,ntree=100,importance=TRUE,
cutoff=c(0.6,0.4))
```

- **randomForest()** implements Breiman's random forest algorithm, for classification and regression.
- The first argument in the function is **formula=** describing the model to be fitted. It can also take **x**, data frame or matrix of predictors.
- **data=** gives the data object.
- **mtry=** Number of variables randomly sampled as candidates at each split. Note that the default values are different for classification (sqrt(p) where p is number of variables in x) and regression (p/3).
- **ntree=** Specifies the number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times.
- **importance=** logical, (default is FALSE) tells R whether variable importance is to be assessed or not.
- **cutoff=** This argument is specific to classification only. A vector of length equal to number of classes. The 'winning' class for an observation is the one with the maximum ratio of proportion of votes to cutoff. Default is 1/k where k is the number of classes (i.e. Majority vote wins).

Random Forest in R – Output

```
# Output
churn_rf
```

```
> churn_rf

Call:
 randomForest(formula = status ~ function. + exp + gende
(0.6, 0.4))
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 2

        OOB estimate of  error rate: 28.92%
Confusion matrix:
   0  1 class.error
0 36 14   0.2800000
1 10 23   0.3030303
```

Interpretation :
Model calculates the OOB error.

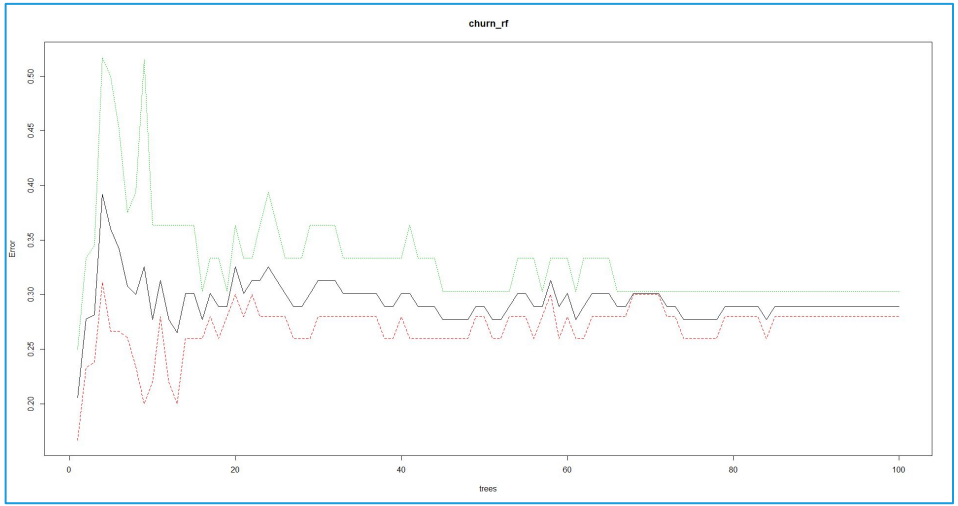
Note : Since the samples are generated randomly, the outputs will vary slightly for different devices.

Random Forest in R

```
# Decision Trees Error Rate
plot(churn_rf)

# Output
```

plot() of a **randomForest()** object returns a plot of the error rates or MSE of the object.



Interpretation :

- Plot shows error rates for all 100 decision trees.
- Black line shows the overall OOB error rate.
- Coloured lines show error rates for each class.

* In case of regression, the plot shows only the black line, overall OOB MSE error.

Random Forest in R – Prediction

```
# Adding Predictions as a new column to original data
empdata$pred <- predict(churn_rf,empdata)
head(empdata)
```

```
# Output
```

	sn	status	function.	exp	gender	source	pred
1	1	1	CS	<3	M	external	1
2	2	1	CS	<3	M	external	1
3	3	1	CS	>=3 and <=5	M	internal	0
4	4	1	CS	>=3 and <=5	F	internal	0
5	5	1	CS	<3	M	internal	1
6	6	1	CS	>5	M	external	1

Random Forest in R – Variable Importance

```
# Importance Matrix
churn_rf$importance
```

randomForest() object contains variable importance matrix, if **importance=T** in the function.

```
# Output
```

	0	1	MeanDecreaseAccuracy	MeanDecreaseGini
function.	0.06742038	0.04707436	0.056724526	7.391617
exp	0.12162621	0.10965367	0.113936498	12.987221
gender	0.01061312	-0.01354428	0.001747229	1.989162
source	0.02297498	-0.01021166	0.009771908	2.586050

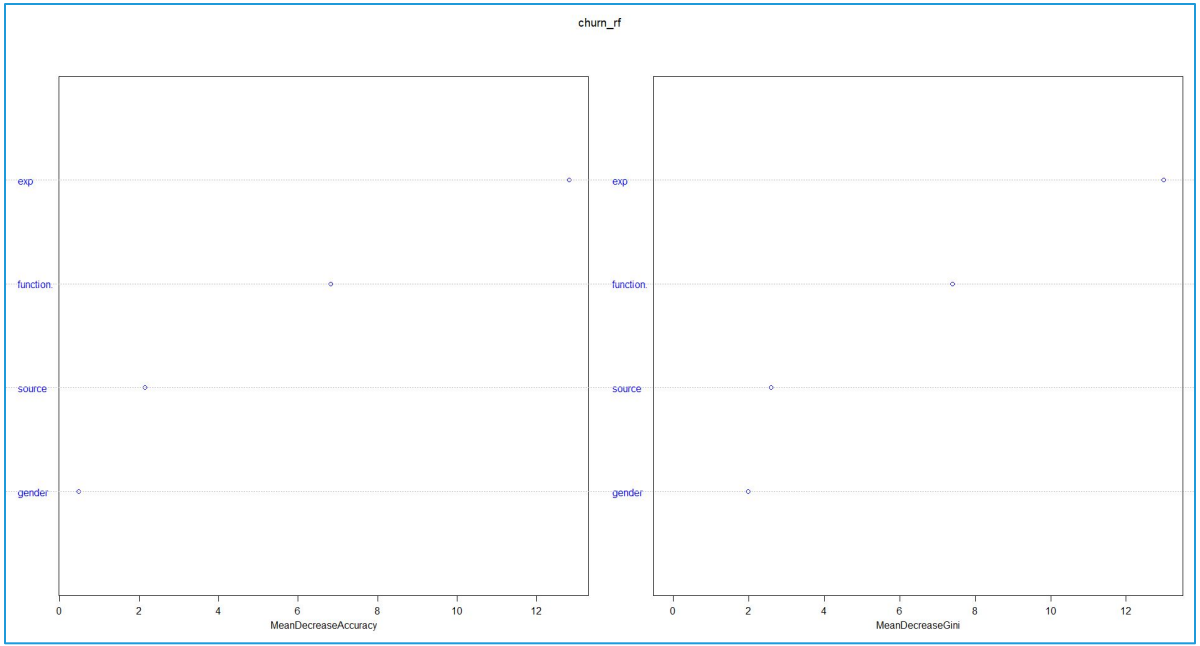
- Experience has highest importance.
- The first two columns are the class-specific measures computed as mean decrease in accuracy.
- The next column is the mean decrease in accuracy over all classes & the mean decrease in Gini index.

```
# Variable Importance Plot
varImpPlot(churn_rf, col="blue")
```

varImpPlot() in package **"randomForest"** returns a dot chart of variable importance measured by a random forest.

Random Forest in R – Variable Importance

```
# Output
```



Quick Recap

Random Forest Method	• Its an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the class's output by individual trees • Random forests also work for regression problems • The method combines Breiman's “Bagging" idea and the random selection of features
Random Forest in R	• **`randomForest()`** in package “**`randomForest`**” runs random forest analysis • The output can generate variable importance and confusion matrix