Data Mining Assignment – Neural Network and Random Forests

DM Group1

Authors: Amrish, Pradeep, Krishna, Balaji Venkat and Santhosh

Date : 23-Jul-2017

Contents

[Scope 3](#_Toc488620383)

[To Check whether Ensemble Model Performance outperforms the individual RF & NN model 3](#_Toc488620384)

[library(ineq) Data Import getwd() 3](#_Toc488620385)

[Exploratory Data analysis (c) 4](#_Toc488620386)

[Removing Insignificant and unusable data from the Source 14](#_Toc488620387)

[Attrition Propotion 22](#_Toc488620388)

[Common for both NN and RF 22](#_Toc488620389)

[Sampling 22](#_Toc488620390)

[Split the data in Dev & Hold Out sample (70:30) 22](#_Toc488620391)

[Data Count on Development and Hold Out 22](#_Toc488620392)

[Development Sample 22](#_Toc488620393)

[Holdout Sample 22](#_Toc488620394)

[Proportions comparison 22](#_Toc488620395)

[Propotion in Dev 22](#_Toc488620396)

[Propotion in Holdout 23](#_Toc488620397)

[PopulationPropotion 23](#_Toc488620398)

[Building Neural Network Model 23](#_Toc488620399)

[Cleaning the Extreme Values and Scaling it 25](#_Toc488620400)

[Distribution is not Proper, so, let us scale the data 25](#_Toc488620401)

[Validating Neural Network 27](#_Toc488620402)

[Basic Confusion Matrix 27](#_Toc488620403)

[Detailed Results 27](#_Toc488620404)

[Scoring Hold-out data using NN 27](#_Toc488620405)

[Random Forest 29](#_Toc488620406)

[Ploting Random Forest 29](#_Toc488620407)

[Importance of variables that are used for Random Forest 29](#_Toc488620408)

[Tuning Random Forest 31](#_Toc488620409)

[Validating RF model 32](#_Toc488620410)

[Area under Curve 32](#_Toc488620411)

[Gini Coefficient 32](#_Toc488620412)

[Confusion matrix 33](#_Toc488620413)

[Scoring for Hold Out Samples 33](#_Toc488620414)

[Ensemble Model of Neural net and RF 34](#_Toc488620415)

[Averaging 34](#_Toc488620416)

[DevData 34](#_Toc488620417)

[Comparison Study 34](#_Toc488620418)

[Ensemble vs Actual 34](#_Toc488620419)

[Conclusion 35](#_Toc488620420)

[Summary 35](#_Toc488620421)

[RCode 36](#_Toc488620422)

# Scope

a) Data Import (Target variable is "Attrition" column)

b) Split the data in Dev & Hold Out sample (70:30)

c) Perform Exploratory Data Analysis

d) Identify columns which are of no use. drop those columns

e) Write Hypothesis and validate the Hypothesis

f) Build Neural Network Model (Development sample)

g) Validate NN model on Hold Out. If need be improvize

h) Build Random Forest Model

i) Validate RF Model

j) Compare NN with RF

k) Combine NN and RF into Ensemble Model

# To Check whether Ensemble Model Performance outperforms the individual RF & NN model

Required Libraries

library(scales)  
library(RColorBrewer)  
library(neuralnet)  
library(googleVis)

library(data.table)  
library(caret)

library(ROCR)

library(randomForest)

# library(ineq) Data Import getwd()

## [1] "C:/Home/Work/GreatLakes/Data Mining/Assignments/NeuralNet and RF"

setwd("C:/Home/Work/GreatLakes/Data Mining/Assignments/NeuralNet and RF")  
darkCol = brewer.pal(9, "Set1")  
  
HRSourceData = read.table("C:/Home/Work/GreatLakes/Data Mining/Assignments/NeuralNet and RF/1452762979\_586\_\_HR\_Employee\_Attrition\_Data.csv", header = T, sep=",")  
  
  
  
#Data Preparation  
# Target data is a factor of String, converting it to Integer of 0 or 1  
Target\_Attrition = as.vector(HRSourceData$Attrition)  
Target\_Attrition = replace(Target\_Attrition,Target\_Attrition=="No",0)  
Target\_Attrition = replace(Target\_Attrition,Target\_Attrition=="Yes",1)  
Target\_Attrition = as.integer(Target\_Attrition)  
HRSourceData$TargetAttrition = Target\_Attrition

# Exploratory Data analysis (c)

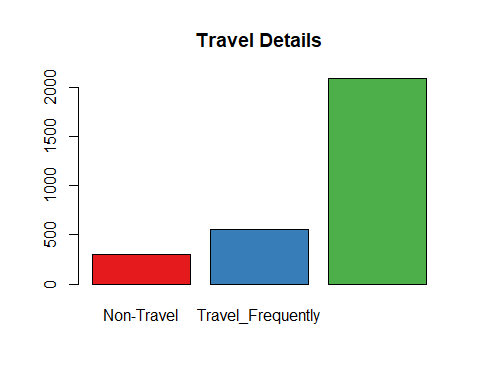
#plot(gvisTable(HRSourceData))  
names(HRSourceData)

## [1] "Age" "Attrition"   
## [3] "BusinessTravel" "DailyRate"   
## [5] "Department" "DistanceFromHome"   
## [7] "Education" "EducationField"   
## [9] "EmployeeCount" "EmployeeNumber"   
## [11] "EnvironmentSatisfaction" "Gender"   
## [13] "HourlyRate" "JobInvolvement"   
## [15] "JobLevel" "JobRole"   
## [17] "JobSatisfaction" "MaritalStatus"   
## [19] "MonthlyIncome" "MonthlyRate"   
## [21] "NumCompaniesWorked" "Over18"   
## [23] "OverTime" "PercentSalaryHike"   
## [25] "PerformanceRating" "RelationshipSatisfaction"  
## [27] "StandardHours" "StockOptionLevel"   
## [29] "TotalWorkingYears" "TrainingTimesLastYear"   
## [31] "WorkLifeBalance" "YearsAtCompany"   
## [33] "YearsInCurrentRole" "YearsSinceLastPromotion"   
## [35] "YearsWithCurrManager" "TargetAttrition"

str(HRSourceData)

## 'data.frame': 2940 obs. of 36 variables:  
## $ Age : int 41 49 37 33 27 32 59 30 38 36 ...  
## $ Attrition : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 1 1 1 1 1 ...  
## $ BusinessTravel : Factor w/ 3 levels "Non-Travel","Travel\_Frequently",..: 3 2 3 2 3 2 3 3 2 3 ...  
## $ DailyRate : int 1102 279 1373 1392 591 1005 1324 1358 216 1299 ...  
## $ Department : Factor w/ 3 levels "Human Resources",..: 3 2 2 2 2 2 2 2 2 2 ...  
## $ DistanceFromHome : int 1 8 2 3 2 2 3 24 23 27 ...  
## $ Education : int 2 1 2 4 1 2 3 1 3 3 ...  
## $ EducationField : Factor w/ 6 levels "Human Resources",..: 2 2 5 2 4 2 4 2 2 4 ...  
## $ EmployeeCount : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ EmployeeNumber : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ EnvironmentSatisfaction : int 2 3 4 4 1 4 3 4 4 3 ...  
## $ Gender : Factor w/ 2 levels "Female","Male": 1 2 2 1 2 2 1 2 2 2 ...  
## $ HourlyRate : int 94 61 92 56 40 79 81 67 44 94 ...  
## $ JobInvolvement : int 3 2 2 3 3 3 4 3 2 3 ...  
## $ JobLevel : int 2 2 1 1 1 1 1 1 3 2 ...  
## $ JobRole : Factor w/ 9 levels "Healthcare Representative",..: 8 7 3 7 3 3 3 3 5 1 ...  
## $ JobSatisfaction : int 4 2 3 3 2 4 1 3 3 3 ...  
## $ MaritalStatus : Factor w/ 3 levels "Divorced","Married",..: 3 2 3 2 2 3 2 1 3 2 ...  
## $ MonthlyIncome : int 5993 5130 2090 2909 3468 3068 2670 2693 9526 5237 ...  
## $ MonthlyRate : int 19479 24907 2396 23159 16632 11864 9964 13335 8787 16577 ...  
## $ NumCompaniesWorked : int 8 1 6 1 9 0 4 1 0 6 ...  
## $ Over18 : Factor w/ 1 level "Y": 1 1 1 1 1 1 1 1 1 1 ...  
## $ OverTime : Factor w/ 2 levels "No","Yes": 2 1 2 2 1 1 2 1 1 1 ...  
## $ PercentSalaryHike : int 11 23 15 11 12 13 20 22 21 13 ...  
## $ PerformanceRating : int 3 4 3 3 3 3 4 4 4 3 ...  
## $ RelationshipSatisfaction: int 1 4 2 3 4 3 1 2 2 2 ...  
## $ StandardHours : int 80 80 80 80 80 80 80 80 80 80 ...  
## $ StockOptionLevel : int 0 1 0 0 1 0 3 1 0 2 ...  
## $ TotalWorkingYears : int 8 10 7 8 6 8 12 1 10 17 ...  
## $ TrainingTimesLastYear : int 0 3 3 3 3 2 3 2 2 3 ...  
## $ WorkLifeBalance : int 1 3 3 3 3 2 2 3 3 2 ...  
## $ YearsAtCompany : int 6 10 0 8 2 7 1 1 9 7 ...  
## $ YearsInCurrentRole : int 4 7 0 7 2 7 0 0 7 7 ...  
## $ YearsSinceLastPromotion : int 0 1 0 3 2 3 0 0 1 7 ...  
## $ YearsWithCurrManager : int 5 7 0 0 2 6 0 0 8 7 ...  
## $ TargetAttrition : int 1 0 1 0 0 0 0 0 0 0 ...

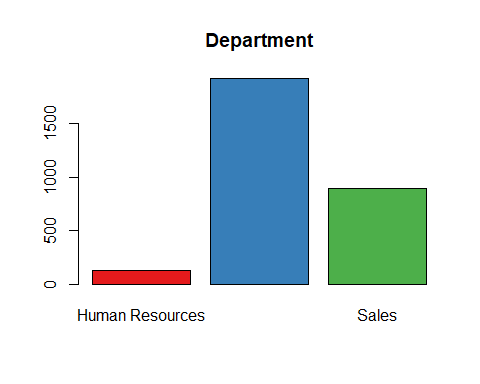
#List of Numerical Variable  
# Age : COnt  
# DailyRate : Ratio  
# Distance from Home : Ratio  
# Education : Ordinal  
# Employee Count : Discrete  
# Employee Number : Nominal  
# Employee Satisfaction : Ordinal  
# Hourly Rate : Cont, Ratio  
# Job Involvement : Discrete  
# Job Level : Ordinal  
# Job Satisfaction : Ordinal  
# Monthly Income : Ratio, Cont  
# Monthly Rate : Ration, cont  
# Num Company Worked : Discrete  
# Percent Salaray Hike : Interval, Cont  
# Performance Rating : Ordinal  
# Relationship Satisfaction : Ordinal  
# Standard Hours : Cont  
# Stock Option Level : Ordinal  
# Total Working Years : Discrete  
# Training Times Last Year : Discrete  
# Work Life Balance : Ordinal  
# Years at Company : Cont, Interval  
# Years in Current Role : Cont, Interval  
# Years Since Last Promotion : Cont, Interval  
# Years with Current Manager : Cont, Interval  
  
#List of Categorical Variable  
#BusinessTravel  
#Department  
#EducationField  
#Gender  
#JobRole  
#MaritalStatus  
#Over18  
#OverTime  
  
#Business Travel  
plot(HRSourceData$BusinessTravel, col=darkCol, main = "Travel Details")



BusinessTravel = data.frame(summary(HRSourceData$BusinessTravel))  
round((BusinessTravel/sum(BusinessTravel))\*100,2)

## summary.HRSourceData.BusinessTravel.  
## Non-Travel 10.20  
## Travel\_Frequently 18.84  
## Travel\_Rarely 70.95

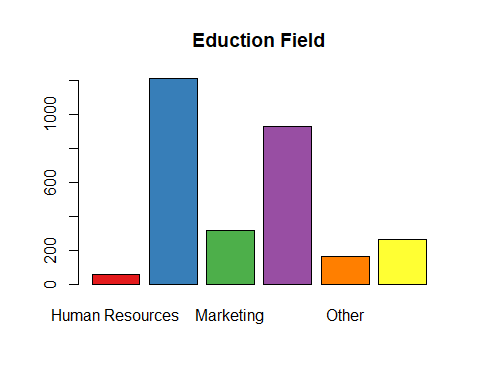
#Department  
plot(HRSourceData$Department, col=darkCol, main="Department")



vDepartment = data.frame(summary(HRSourceData$Department))  
round((vDepartment/sum(vDepartment))\*100,2)

## summary.HRSourceData.Department.  
## Human Resources 4.29  
## Research & Development 65.37  
## Sales 30.34

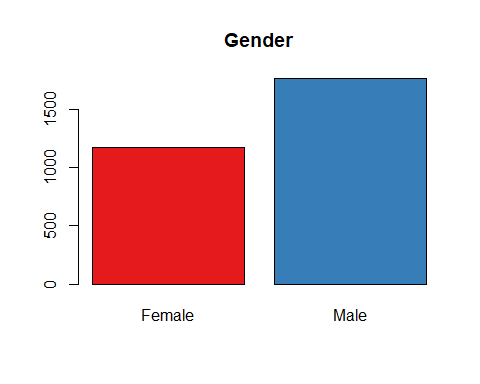
#EducationField  
plot(HRSourceData$EducationField, col=darkCol, main="Eduction Field")



vEducationField = data.frame(summary(HRSourceData$EducationField))  
round((vEducationField/sum(vEducationField))\*100,2)

## summary.HRSourceData.EducationField.  
## Human Resources 1.84  
## Life Sciences 41.22  
## Marketing 10.82  
## Medical 31.56  
## Other 5.58  
## Technical Degree 8.98

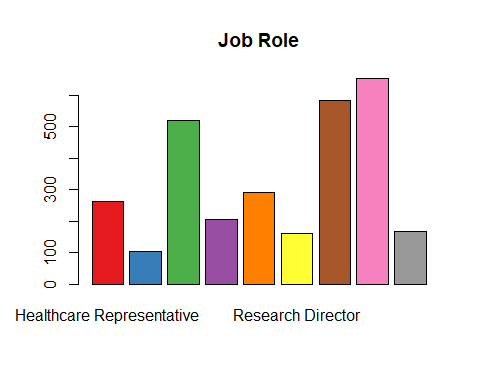
#Gender  
plot(HRSourceData$Gender, col=darkCol,main="Gender")



vGender = data.frame(summary(HRSourceData$Gender))  
round((vGender/sum(vGender))\*100,2)

## summary.HRSourceData.Gender.  
## Female 40  
## Male 60

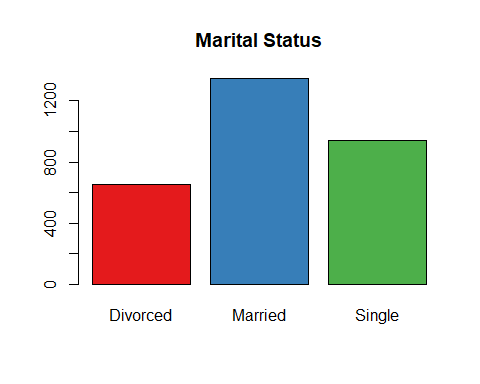
#JobRole  
plot(HRSourceData$JobRole, col=darkCol, main="Job Role")



vJobRole = data.frame(summary(HRSourceData$JobRole))  
round((vJobRole/sum(vJobRole))\*100,2)

## summary.HRSourceData.JobRole.  
## Healthcare Representative 8.91  
## Human Resources 3.54  
## Laboratory Technician 17.62  
## Manager 6.94  
## Manufacturing Director 9.86  
## Research Director 5.44  
## Research Scientist 19.86  
## Sales Executive 22.18  
## Sales Representative 5.65

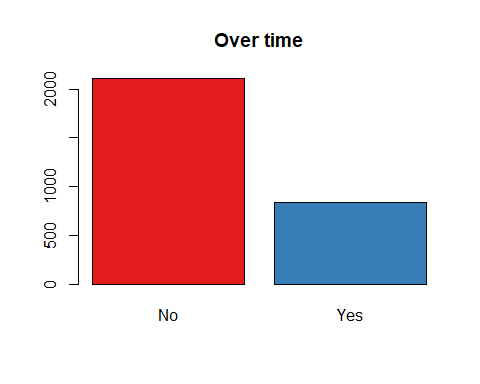
#MaritalStatus  
plot(HRSourceData$MaritalStatus, col=darkCol, main="Marital Status")



vMaritalStatus = data.frame(summary(HRSourceData$MaritalStatus))  
round((vMaritalStatus/sum(vMaritalStatus))\*100,2)

## summary.HRSourceData.MaritalStatus.  
## Divorced 22.24  
## Married 45.78  
## Single 31.97

#OverTime  
plot(HRSourceData$OverTime, col=darkCol, main="Over time")



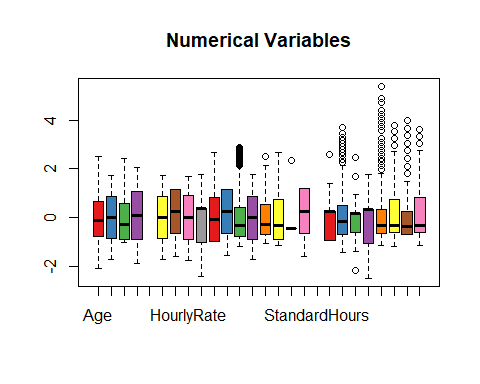
vOverTime = data.frame(summary(HRSourceData$OverTime))  
round((vOverTime/sum(vOverTime))\*100,2)

## summary.HRSourceData.OverTime.  
## No 71.7  
## Yes 28.3

#Finding Ouliers in Cont data  
x = subset(HRSourceData,   
 select = c("Age",  
 "DailyRate",  
 "DistanceFromHome",  
 "Education",  
 "EmployeeCount",  
 "EmployeeNumber",  
 "EnvironmentSatisfaction",  
 "HourlyRate",  
 "JobInvolvement",  
 "JobLevel",  
 "JobSatisfaction",  
 "MonthlyIncome",  
 "MonthlyRate",  
 "NumCompaniesWorked",  
 "PercentSalaryHike",  
 "PerformanceRating",  
 "RelationshipSatisfaction",  
 "StandardHours",  
 "StockOptionLevel",  
 "TotalWorkingYears",  
 "TrainingTimesLastYear",  
 "WorkLifeBalance",  
 "YearsAtCompany",  
 "YearsInCurrentRole",  
 "YearsSinceLastPromotion",  
 "YearsWithCurrManager"))  
class(x)

## [1] "data.frame"

y = scale(x[,])  
boxplot(y, col=darkCol, main="Numerical Variables")



# Removing Insignificant and unusable data from the Source

#Before splitting the data as development and holdout, let us convert the categorical variables to continous variables.  
# Get rid of variables that are of no use  
# StandardHours and Over18 are having same values for all observations, so we can remove those variables.  
CleanedHRData = HRSourceData[,!(names(HRSourceData) %in% c("EmployeeCount", "StandardHours","Over18","EmployeeNumber"))]  
  
# Converting Categorical Variables to COnt  
#BusinessTravel  
mBusinessTravel = model.matrix(~ BusinessTravel - 1, data = CleanedHRData)  
head(mBusinessTravel)

## BusinessTravelNon-Travel BusinessTravelTravel\_Frequently  
## 1 0 0  
## 2 0 1  
## 3 0 0  
## 4 0 1  
## 5 0 0  
## 6 0 1  
## BusinessTravelTravel\_Rarely  
## 1 1  
## 2 0  
## 3 1  
## 4 0  
## 5 1  
## 6 0

CleanedHRData = data.frame(CleanedHRData, mBusinessTravel)  
CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("BusinessTravel"))]  
  
#Department  
mDepartment = model.matrix(~ Department - 1, data = CleanedHRData)  
head(mDepartment)

## DepartmentHuman Resources DepartmentResearch & Development  
## 1 0 0  
## 2 0 1  
## 3 0 1  
## 4 0 1  
## 5 0 1  
## 6 0 1  
## DepartmentSales  
## 1 1  
## 2 0  
## 3 0  
## 4 0  
## 5 0  
## 6 0

CleanedHRData = data.frame(CleanedHRData, mDepartment)  
CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("Department"))]  
  
#EducationField  
mEducationField = model.matrix(~ EducationField - 1, data = CleanedHRData)  
head(mEducationField)

## EducationFieldHuman Resources EducationFieldLife Sciences  
## 1 0 1  
## 2 0 1  
## 3 0 0  
## 4 0 1  
## 5 0 0  
## 6 0 1  
## EducationFieldMarketing EducationFieldMedical EducationFieldOther  
## 1 0 0 0  
## 2 0 0 0  
## 3 0 0 1  
## 4 0 0 0  
## 5 0 1 0  
## 6 0 0 0  
## EducationFieldTechnical Degree  
## 1 0  
## 2 0  
## 3 0  
## 4 0  
## 5 0  
## 6 0

CleanedHRData = data.frame(CleanedHRData, mEducationField)  
CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("EducationField"))]  
  
#Gender  
mGender = model.matrix(~ Gender - 1, data = CleanedHRData)  
head(mGender)

## GenderFemale GenderMale  
## 1 1 0  
## 2 0 1  
## 3 0 1  
## 4 1 0  
## 5 0 1  
## 6 0 1

CleanedHRData = data.frame(CleanedHRData, mGender)  
CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("Gender"))]  
  
#JobRole  
mJobRole = model.matrix(~ JobRole - 1, data = CleanedHRData)  
head(mJobRole)

## JobRoleHealthcare Representative JobRoleHuman Resources  
## 1 0 0  
## 2 0 0  
## 3 0 0  
## 4 0 0  
## 5 0 0  
## 6 0 0  
## JobRoleLaboratory Technician JobRoleManager  
## 1 0 0  
## 2 0 0  
## 3 1 0  
## 4 0 0  
## 5 1 0  
## 6 1 0  
## JobRoleManufacturing Director JobRoleResearch Director  
## 1 0 0  
## 2 0 0  
## 3 0 0  
## 4 0 0  
## 5 0 0  
## 6 0 0  
## JobRoleResearch Scientist JobRoleSales Executive  
## 1 0 1  
## 2 1 0  
## 3 0 0  
## 4 1 0  
## 5 0 0  
## 6 0 0  
## JobRoleSales Representative  
## 1 0  
## 2 0  
## 3 0  
## 4 0  
## 5 0  
## 6 0

CleanedHRData = data.frame(CleanedHRData, mJobRole)  
CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("JobRole"))]  
  
#MaritalStatus  
mMaritalStatus = model.matrix(~ MaritalStatus - 1, data = CleanedHRData)  
head(mMaritalStatus)

## MaritalStatusDivorced MaritalStatusMarried MaritalStatusSingle  
## 1 0 0 1  
## 2 0 1 0  
## 3 0 0 1  
## 4 0 1 0  
## 5 0 1 0  
## 6 0 0 1

CleanedHRData = data.frame(CleanedHRData, mMaritalStatus)  
CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("MaritalStatus"))]  
  
#OverTime  
mOverTime = model.matrix(~ OverTime - 1, data = CleanedHRData)  
head(mOverTime)

## OverTimeNo OverTimeYes  
## 1 0 1  
## 2 1 0  
## 3 0 1  
## 4 0 1  
## 5 1 0  
## 6 1 0

CleanedHRData = data.frame(CleanedHRData, mOverTime)  
CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("OverTime"))]  
  
CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("Attrition"))]  
CleanedHRData = cbind(CleanedHRData[,(names(CleanedHRData) %in% c("TargetAttrition"))],  
 CleanedHRData[,!(names(CleanedHRData) %in% c("TargetAttrition"))])  
names(CleanedHRData)[1] = "TargetAttrition"  
  
  
  
  
str(CleanedHRData)

## 'data.frame': 2940 obs. of 52 variables:  
## $ TargetAttrition : int 1 0 1 0 0 0 0 0 0 0 ...  
## $ Age : int 41 49 37 33 27 32 59 30 38 36 ...  
## $ DailyRate : int 1102 279 1373 1392 591 1005 1324 1358 216 1299 ...  
## $ DistanceFromHome : int 1 8 2 3 2 2 3 24 23 27 ...  
## $ Education : int 2 1 2 4 1 2 3 1 3 3 ...  
## $ EnvironmentSatisfaction : int 2 3 4 4 1 4 3 4 4 3 ...  
## $ HourlyRate : int 94 61 92 56 40 79 81 67 44 94 ...  
## $ JobInvolvement : int 3 2 2 3 3 3 4 3 2 3 ...  
## $ JobLevel : int 2 2 1 1 1 1 1 1 3 2 ...  
## $ JobSatisfaction : int 4 2 3 3 2 4 1 3 3 3 ...  
## $ MonthlyIncome : int 5993 5130 2090 2909 3468 3068 2670 2693 9526 5237 ...  
## $ MonthlyRate : int 19479 24907 2396 23159 16632 11864 9964 13335 8787 16577 ...  
## $ NumCompaniesWorked : int 8 1 6 1 9 0 4 1 0 6 ...  
## $ PercentSalaryHike : int 11 23 15 11 12 13 20 22 21 13 ...  
## $ PerformanceRating : int 3 4 3 3 3 3 4 4 4 3 ...  
## $ RelationshipSatisfaction : int 1 4 2 3 4 3 1 2 2 2 ...  
## $ StockOptionLevel : int 0 1 0 0 1 0 3 1 0 2 ...  
## $ TotalWorkingYears : int 8 10 7 8 6 8 12 1 10 17 ...  
## $ TrainingTimesLastYear : int 0 3 3 3 3 2 3 2 2 3 ...  
## $ WorkLifeBalance : int 1 3 3 3 3 2 2 3 3 2 ...  
## $ YearsAtCompany : int 6 10 0 8 2 7 1 1 9 7 ...  
## $ YearsInCurrentRole : int 4 7 0 7 2 7 0 0 7 7 ...  
## $ YearsSinceLastPromotion : int 0 1 0 3 2 3 0 0 1 7 ...  
## $ YearsWithCurrManager : int 5 7 0 0 2 6 0 0 8 7 ...  
## $ BusinessTravelNon.Travel : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ BusinessTravelTravel\_Frequently : num 0 1 0 1 0 1 0 0 1 0 ...  
## $ BusinessTravelTravel\_Rarely : num 1 0 1 0 1 0 1 1 0 1 ...  
## $ DepartmentHuman.Resources : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ DepartmentResearch...Development: num 0 1 1 1 1 1 1 1 1 1 ...  
## $ DepartmentSales : num 1 0 0 0 0 0 0 0 0 0 ...  
## $ EducationFieldHuman.Resources : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ EducationFieldLife.Sciences : num 1 1 0 1 0 1 0 1 1 0 ...  
## $ EducationFieldMarketing : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ EducationFieldMedical : num 0 0 0 0 1 0 1 0 0 1 ...  
## $ EducationFieldOther : num 0 0 1 0 0 0 0 0 0 0 ...  
## $ EducationFieldTechnical.Degree : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ GenderFemale : num 1 0 0 1 0 0 1 0 0 0 ...  
## $ GenderMale : num 0 1 1 0 1 1 0 1 1 1 ...  
## $ JobRoleHealthcare.Representative: num 0 0 0 0 0 0 0 0 0 1 ...  
## $ JobRoleHuman.Resources : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ JobRoleLaboratory.Technician : num 0 0 1 0 1 1 1 1 0 0 ...  
## $ JobRoleManager : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ JobRoleManufacturing.Director : num 0 0 0 0 0 0 0 0 1 0 ...  
## $ JobRoleResearch.Director : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ JobRoleResearch.Scientist : num 0 1 0 1 0 0 0 0 0 0 ...  
## $ JobRoleSales.Executive : num 1 0 0 0 0 0 0 0 0 0 ...  
## $ JobRoleSales.Representative : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ MaritalStatusDivorced : num 0 0 0 0 0 0 0 1 0 0 ...  
## $ MaritalStatusMarried : num 0 1 0 1 1 0 1 0 0 1 ...  
## $ MaritalStatusSingle : num 1 0 1 0 0 1 0 0 1 0 ...  
## $ OverTimeNo : num 0 1 0 0 1 1 0 1 1 1 ...  
## $ OverTimeYes : num 1 0 1 1 0 0 1 0 0 0 ...

#Test for Significance using Anova  
fit1 = aov(TargetAttrition ~ ., data = CleanedHRData)  
summary(fit1)

## Df Sum Sq Mean Sq F value Pr(>F)   
## Age 1 10.08 10.077 98.845 < 2e-16 \*\*\*  
## DailyRate 1 1.20 1.201 11.779 0.000607 \*\*\*  
## DistanceFromHome 1 2.38 2.381 23.353 1.42e-06 \*\*\*  
## Education 1 0.00 0.000 0.004 0.951795   
## EnvironmentSatisfaction 1 3.95 3.946 38.703 5.65e-10 \*\*\*  
## HourlyRate 1 0.03 0.033 0.322 0.570205   
## JobInvolvement 1 6.14 6.138 60.205 1.18e-14 \*\*\*  
## JobLevel 1 4.62 4.619 45.311 2.02e-11 \*\*\*  
## JobSatisfaction 1 4.54 4.542 44.552 2.96e-11 \*\*\*  
## MonthlyIncome 1 0.02 0.024 0.239 0.624739   
## MonthlyRate 1 0.17 0.172 1.685 0.194320   
## NumCompaniesWorked 1 3.53 3.533 34.653 4.39e-09 \*\*\*  
## PercentSalaryHike 1 0.20 0.199 1.948 0.162920   
## PerformanceRating 1 0.10 0.104 1.024 0.311695   
## RelationshipSatisfaction 1 0.62 0.615 6.034 0.014094 \*   
## StockOptionLevel 1 6.94 6.942 68.097 2.34e-16 \*\*\*  
## TotalWorkingYears 1 0.66 0.659 6.465 0.011051 \*   
## TrainingTimesLastYear 1 1.34 1.339 13.133 0.000295 \*\*\*  
## WorkLifeBalance 1 1.39 1.393 13.665 0.000223 \*\*\*  
## YearsAtCompany 1 0.03 0.030 0.298 0.585317   
## YearsInCurrentRole 1 2.51 2.512 24.641 7.30e-07 \*\*\*  
## YearsSinceLastPromotion 1 2.36 2.360 23.152 1.57e-06 \*\*\*  
## YearsWithCurrManager 1 1.92 1.917 18.808 1.49e-05 \*\*\*  
## BusinessTravelNon.Travel 1 2.47 2.470 24.229 9.03e-07 \*\*\*  
## BusinessTravelTravel\_Frequently 1 4.06 4.061 39.835 3.19e-10 \*\*\*  
## DepartmentHuman.Resources 1 0.04 0.043 0.421 0.516369   
## DepartmentResearch...Development 1 3.38 3.377 33.124 9.55e-09 \*\*\*  
## EducationFieldHuman.Resources 1 0.36 0.364 3.573 0.058835 .   
## EducationFieldLife.Sciences 1 0.16 0.159 1.559 0.211916   
## EducationFieldMarketing 1 0.11 0.112 1.098 0.294718   
## EducationFieldMedical 1 1.10 1.101 10.802 0.001026 \*\*   
## EducationFieldOther 1 0.98 0.979 9.603 0.001961 \*\*   
## GenderFemale 1 0.49 0.491 4.817 0.028255 \*   
## JobRoleHealthcare.Representative 1 0.53 0.532 5.216 0.022452 \*   
## JobRoleHuman.Resources 1 0.02 0.025 0.245 0.620641   
## JobRoleLaboratory.Technician 1 2.01 2.006 19.674 9.52e-06 \*\*\*  
## JobRoleManager 1 0.00 0.002 0.018 0.891877   
## JobRoleManufacturing.Director 1 0.11 0.115 1.126 0.288661   
## JobRoleResearch.Director 1 0.00 0.003 0.032 0.858377   
## JobRoleResearch.Scientist 1 0.35 0.353 3.462 0.062910 .   
## JobRoleSales.Executive 1 2.57 2.568 25.193 5.50e-07 \*\*\*  
## MaritalStatusDivorced 1 0.10 0.099 0.973 0.324081   
## MaritalStatusMarried 1 3.25 3.251 31.893 1.79e-08 \*\*\*  
## OverTimeNo 1 25.59 25.589 250.999 < 2e-16 \*\*\*  
## Residuals 2895 295.14 0.102   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

#Following Variables came out as insignificant, it can be dropped from the data set.  
# Df Sum Sq Mean Sq F value Pr(>F)  
#Education 1 0.00 0.000 0.004 0.952001   
#HourlyRate 1 0.03 0.033 0.320 0.571858   
#MonthlyIncome 1 0.02 0.024 0.237 0.626222   
#MonthlyRate 1 0.17 0.172 1.671 0.196237   
#PercentSalaryHike 1 0.20 0.199 1.931 0.164728   
#PerformanceRating 1 0.10 0.104 1.015 0.313771   
#YearsAtCompany 1 0.03 0.030 0.295 0.586925   
#JobRoleHuman.Resources 1 0.03 0.026 0.252 0.615773   
#JobRoleManager 1 0.01 0.006 0.060 0.806153   
#MaritalStatusDivorced 1 0.11 0.110 1.065 0.302052   
CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("Education","HourlyRate",  
 "MonthlyIncome","MonthlyRate",  
 "PercentSalaryHike", "PerformanceRating",  
 "YearsAtCompany","JobRoleHuman.Resources",  
 "JobRoleManager", "MaritalStatusDivorced"))]  
fit2 = aov(TargetAttrition ~ ., data = CleanedHRData)  
summary(fit2)

## Df Sum Sq Mean Sq F value Pr(>F)   
## Age 1 10.08 10.077 98.733 < 2e-16 \*\*\*  
## DailyRate 1 1.20 1.201 11.765 0.000612 \*\*\*  
## DistanceFromHome 1 2.38 2.381 23.326 1.44e-06 \*\*\*  
## EnvironmentSatisfaction 1 3.94 3.940 38.604 5.94e-10 \*\*\*  
## JobInvolvement 1 6.17 6.170 60.449 1.04e-14 \*\*\*  
## JobLevel 1 4.59 4.594 45.008 2.35e-11 \*\*\*  
## JobSatisfaction 1 4.46 4.464 43.736 4.46e-11 \*\*\*  
## NumCompaniesWorked 1 3.53 3.535 34.632 4.44e-09 \*\*\*  
## RelationshipSatisfaction 1 0.58 0.580 5.681 0.017211 \*   
## StockOptionLevel 1 7.11 7.107 69.635 < 2e-16 \*\*\*  
## TotalWorkingYears 1 0.61 0.610 5.973 0.014587 \*   
## TrainingTimesLastYear 1 1.34 1.340 13.125 0.000296 \*\*\*  
## WorkLifeBalance 1 1.40 1.397 13.684 0.000220 \*\*\*  
## YearsInCurrentRole 1 1.73 1.729 16.943 3.96e-05 \*\*\*  
## YearsSinceLastPromotion 1 3.10 3.104 30.411 3.80e-08 \*\*\*  
## YearsWithCurrManager 1 1.22 1.217 11.925 0.000562 \*\*\*  
## BusinessTravelNon.Travel 1 2.47 2.469 24.190 9.21e-07 \*\*\*  
## BusinessTravelTravel\_Frequently 1 4.17 4.169 40.848 1.91e-10 \*\*\*  
## DepartmentHuman.Resources 1 0.08 0.081 0.792 0.373623   
## DepartmentResearch...Development 1 3.37 3.370 33.021 1.01e-08 \*\*\*  
## EducationFieldHuman.Resources 1 0.35 0.351 3.436 0.063893 .   
## EducationFieldLife.Sciences 1 0.17 0.171 1.671 0.196284   
## EducationFieldMarketing 1 0.11 0.108 1.061 0.303092   
## EducationFieldMedical 1 1.02 1.022 10.018 0.001566 \*\*   
## EducationFieldOther 1 1.03 1.029 10.080 0.001514 \*\*   
## GenderFemale 1 0.49 0.486 4.759 0.029230 \*   
## JobRoleHealthcare.Representative 1 0.57 0.573 5.617 0.017856 \*   
## JobRoleLaboratory.Technician 1 2.11 2.107 20.640 5.77e-06 \*\*\*  
## JobRoleManufacturing.Director 1 0.16 0.164 1.609 0.204785   
## JobRoleResearch.Director 1 0.01 0.010 0.095 0.758519   
## JobRoleResearch.Scientist 1 0.23 0.233 2.282 0.130957   
## JobRoleSales.Executive 1 1.16 1.159 11.353 0.000763 \*\*\*  
## JobRoleSales.Representative 1 1.33 1.326 12.989 0.000319 \*\*\*  
## MaritalStatusMarried 1 1.25 1.246 12.203 0.000484 \*\*\*  
## MaritalStatusSingle 1 1.97 1.971 19.314 1.15e-05 \*\*\*  
## OverTimeNo 1 25.80 25.799 252.776 < 2e-16 \*\*\*  
## Residuals 2903 296.29 0.102   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

## Attrition Propotion

PopulationPropotion = sum(CleanedHRData$TargetAttrition)/nrow(CleanedHRData) percent(PopulationPropotion)

"16.1%"

## Common for both NN and RF

## Sampling

sampleIndex = sample(nrow(CleanedHRData), nrow(CleanedHRData)\*.7) #Splitting of Data #Training Data HrDev = CleanedHRData[sampleIndex,] #Testing Data HrHoldOut = CleanedHRData[-sampleIndex,]

# Split the data in Dev & Hold Out sample (70:30)

## Data Count on Development and Hold Out

### Development Sample

dim(HrDev)

[1] 2058 42

### Holdout Sample

dim(HrHoldOut)

[1] 882 42

### Proportions comparison

### Propotion in Dev

DevPropotion = sum(HrDev$TargetAttrition)/(nrow(HrDev)) DevPropotion

[1] 0.1598639456

### Propotion in Holdout

HOPropotion = sum(HrHoldOut$TargetAttrition)/(nrow(HrDev)) HOPropotion

[1] 0.07045675413

### PopulationPropotion

[1] 0.1612244898

rbind(PopulationPropotion,DevPropotion, HOPropotion) # Comparison of the Distribution of Data.

PopulationPropotion 0.16122448980

DevPropotion 0.15986394558

HOPropotion 0.07045675413

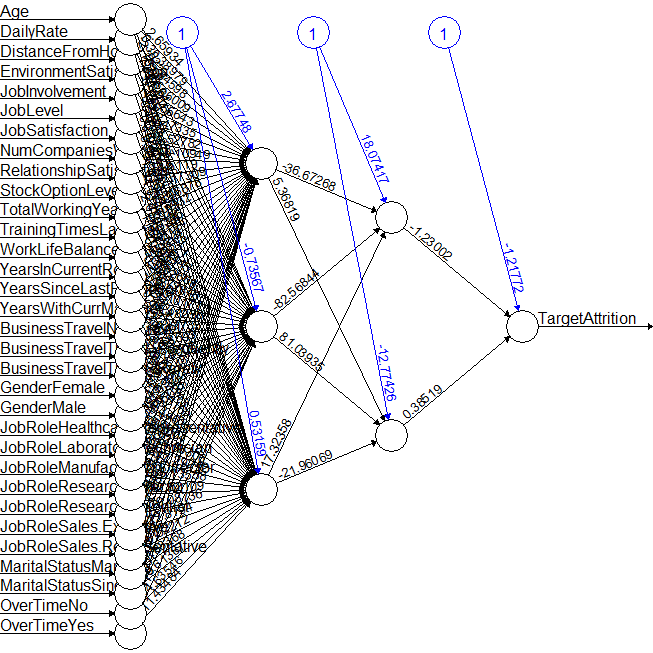
# Building Neural Network Model

For Building the Neural net, we have to be keen in selecting following parameters

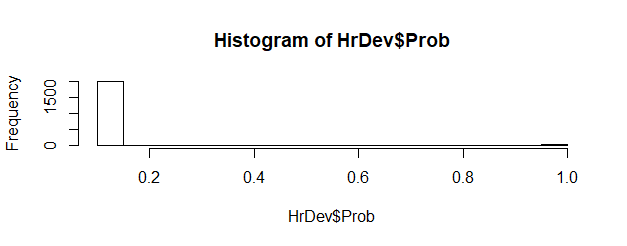
* number hidden layers
* Number of neuron (tumbrule is sqrt)
* epoh
* Activation Function
* avoiding overfitting.
* function for dealing with error
* threshold - important factor that decides over fitting.
* stopmax
* learning rate

names(CleanedHRData) nn1 = neuralnet(TargetAttrition ~ Age + DailyRate + DistanceFromHome + EnvironmentSatisfaction + JobInvolvement + JobLevel + JobSatisfaction + NumCompaniesWorked + RelationshipSatisfaction + StockOptionLevel + TotalWorkingYears + TrainingTimesLastYear + WorkLifeBalance + YearsInCurrentRole + YearsSinceLastPromotion + YearsWithCurrManager + BusinessTravelNon.Travel + BusinessTravelTravel\_Frequently + BusinessTravelTravel\_Rarely + GenderFemale + GenderMale + JobRoleHealthcare.Representative + JobRoleLaboratory.Technician + JobRoleManufacturing.Director + JobRoleResearch.Director + JobRoleResearch.Scientist + JobRoleSales.Executive + JobRoleSales.Representative + MaritalStatusMarried + MaritalStatusSingle + OverTimeNo + OverTimeYes , data = HrDev, hidden = c(3,2), linear.output = FALSE, err.fct = "sse", lifesign = "full", lifesign.step = 10, threshold = 0.01, stepmax = 4000)

plot(nn1)



nn1Prob = as.numeric(nn1Prob)

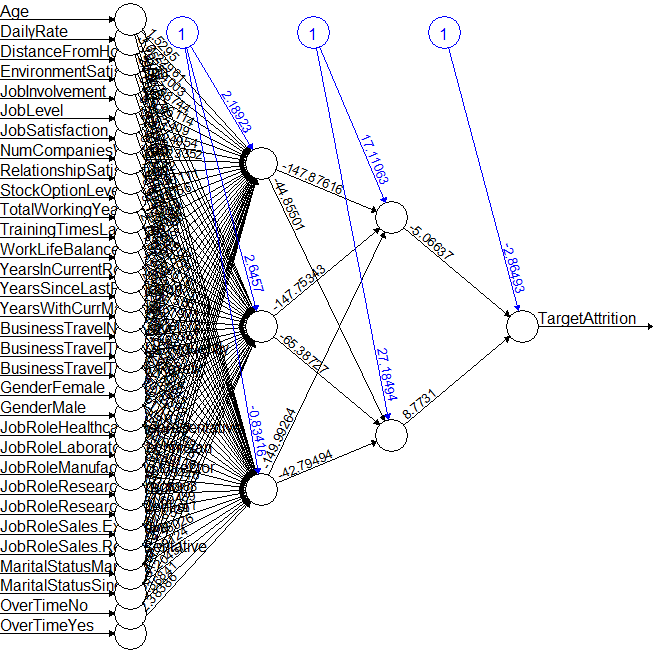


## Cleaning the Extreme Values and Scaling it

TargetAttrition = HrDev[,1] TargetAttrition HRDevScaledData = scale(HrDev[,-1]) HRDevScaledData = cbind(TargetAttrition,HRDevScaledData ) #plot(gvisTable(data.frame(HRDevScaledData)))

## Distribution is not Proper, so, let us scale the data

nn2 = neuralnet(TargetAttrition ~ Age + DailyRate + DistanceFromHome + EnvironmentSatisfaction + JobInvolvement + JobLevel + JobSatisfaction + NumCompaniesWorked + RelationshipSatisfaction + StockOptionLevel + TotalWorkingYears + TrainingTimesLastYear + WorkLifeBalance + YearsInCurrentRole + YearsSinceLastPromotion + YearsWithCurrManager + BusinessTravelNon.Travel + BusinessTravelTravel\_Frequently + BusinessTravelTravel\_Rarely + GenderFemale + GenderMale + JobRoleHealthcare.Representative + JobRoleLaboratory.Technician + JobRoleManufacturing.Director + JobRoleResearch.Director + JobRoleResearch.Scientist + JobRoleSales.Executive + JobRoleSales.Representative + MaritalStatusMarried + MaritalStatusSingle + OverTimeNo + OverTimeYes , data = HRDevScaledData, hidden = c(3,2), linear.output = FALSE, err.fct = "sse", lifesign = "full", lifesign.step = 10, threshold = 0.01, stepmax = 4000)



HRDevScaledData\_df = as.data.frame(HRDevScaledData)

HRDevScaledData\_dfnet.result[[1]])

quantile(HRDevScaledData\_df$Prob, c(0,10,20,30,40,50,60,70,80,90,100)/100)

hist(HRDevScaledData\_df$Prob)

# Validating Neural Network

## Basic Confusion Matrix

HRDevScaledData\_dfProb>0.16,1,0) with(HRDevScaledData\_df, table(TargetAttrition,Predicted.Score))

Predicted.Score

TargetAttrition 0 1

0 1680 41

1 96 241

## Detailed Results

confusionMatrix(table(HRDevScaledData\_dfTargetAttrition), dnn = c("Predicted Attrition","Actual Attrition"))

Confusion Matrix and Statistics

0 1

0 1680 96

1 41 241

**Accuracy : 0.9334305**

95% CI : (0.9217871, 0.9438196)

No Information Rate : 0.8362488

P-Value [Acc > NIR] : < 0.00000000000000022204

Kappa : 0.7398624

Mcnemar's Test P-Value : 0.000003958845

Sensitivity : 0.9761766

Specificity : 0.7151335

Pos Pred Value : 0.9459459

Neg Pred Value : 0.8546099

Prevalence : 0.8362488

Detection Rate : 0.8163265

Detection Prevalence : 0.8629738

Balanced Accuracy : 0.8456551

'Positive' Class : 0

## Scoring Hold-out data using NN

HTargetAttrition = HrHoldOut[,1] HTargetAttrition HRHoldOutScaledData = scale(HrHoldOut[,-1]) HRHoldOutScaledData = cbind(HTargetAttrition,HRHoldOutScaledData ) HoldOutOutput = compute(nn2,HRHoldOutScaledData[,c("Age","DailyRate","DistanceFromHome","EnvironmentSatisfaction","JobInvolvement","JobLevel","JobSatisfaction","NumCompaniesWorked","RelationshipSatisfaction","StockOptionLevel","TotalWorkingYears","TrainingTimesLastYear","WorkLifeBalance","YearsInCurrentRole","YearsSinceLastPromotion","YearsWithCurrManager","BusinessTravelNon.Travel","BusinessTravelTravel\_Frequently","BusinessTravelTravel\_Rarely","GenderFemale","GenderMale","JobRoleHealthcare.Representative","JobRoleLaboratory.Technician","JobRoleManufacturing.Director","JobRoleResearch.Director","JobRoleResearch.Scientist","JobRoleSales.Executive","JobRoleSales.Representative","MaritalStatusMarried","MaritalStatusSingle","OverTimeNo","OverTimeYes")]) HRHoldOutScaledData\_df = as.data.frame(HRHoldOutScaledData) HRHoldOutScaledData\_dfnet.result[,1] HRHoldOutScaledData\_dfProb>0.16,1,0) cm\_HRHOldout = confusionMatrix(table(HRHoldOutScaledData\_dfHTargetAttrition))

Confusion Matrix and Statistics – Hold Out Data

0 1

0 708 53

1 29 92

**Accuracy : 0.9070295**

95% CI : (0.8859137, 0.9253754)

No Information Rate : 0.8356009

P-Value [Acc > NIR] : 0.0000000006009261

Kappa : 0.6375137

Mcnemar's Test P-Value : 0.01108762

Sensitivity : 0.9606513

Specificity : 0.6344828

Pos Pred Value : 0.9303548

Neg Pred Value : 0.7603306

Prevalence : 0.8356009

Detection Rate : 0.8027211

Detection Prevalence : 0.8628118

Balanced Accuracy : 0.7975670

'Positive' Class : 0

# Random Forest

rfHRDevScaledData\_df = HRDevScaledData\_df[,-c(43:45)] RF <- randomForest(as.factor(TargetAttrition) ~ ., data = rfHRDevScaledData\_df[,-1], ntree=500, mtry = 3, nodesize = 10, importance=TRUE) print(RF)

## Ploting Random Forest

plot(RF) legend("topright", c("OOB", "0", "1"), text.col=1:6, lty=1:3, col=1:3) title(main="Error Rates Random Forest HR data - Development") RF$err.rate

Call:

randomForest(formula = as.factor(TargetAttrition) ~ ., data = rfHRDevScaledData\_df[, -1], ntree = 500, mtry = 3, nodesize = 10, importance = TRUE)

Type of random forest: classification

Number of trees: 500

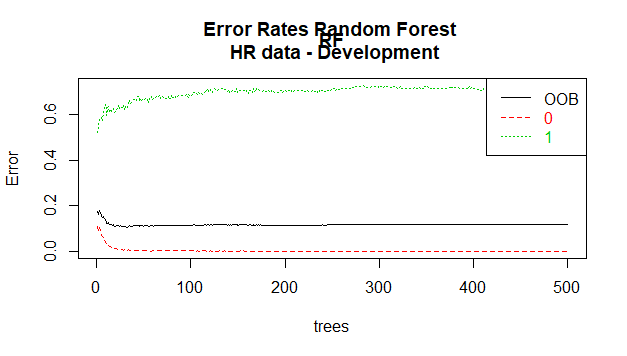
No. of variables tried at each split: 3

**OOB estimate of error rate: 11.52%**

Confusion matrix:

0 1 class.error

0 1719 2 0.001162115049

1 235 102 0.697329376855

## Importance of variables that are used for Random Forest

impVar <- round(randomForest::importance(RF), 2) impVar[order(impVar[,3], decreasing=TRUE),]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Variabkes** | **0** | **1** | **MeanDecreaseAccuracy** | **MeanDecreaseGini** |
| Age | 17.15 | 19.98 | 23.59 | 24.92 |
| TotalWorkingYears | 15.81 | 18.73 | 22.67 | 21.19 |
| YearsWithCurrManager | 13.58 | 17.99 | 21.62 | 16.57 |
| DistanceFromHome | 14.46 | 17.68 | 20.3 | 17.62 |
| OverTimeNo | 17.25 | 19.52 | 20.21 | 12.98 |
| EnvironmentSatisfaction | 14.67 | 18.44 | 19.67 | 11.53 |
| OverTimeYes | 16.24 | 18.2 | 19.24 | 12.99 |
| JobLevel | 12.36 | 18.2 | 18.28 | 13.14 |
| WorkLifeBalance | 12.7 | 16.78 | 18.16 | 11.76 |
| JobInvolvement | 12.55 | 16.21 | 18.15 | 11.04 |
| YearsInCurrentRole | 10.99 | 16.06 | 17.64 | 13.35 |
| StockOptionLevel | 12.8 | 16.46 | 17.54 | 10.85 |
| DailyRate | 10.94 | 15.66 | 17.43 | 19.5 |
| NumCompaniesWorked | 12.27 | 14.51 | 16.98 | 12.83 |
| JobSatisfaction | 12.84 | 12.71 | 16.68 | 9.56 |
| RelationshipSatisfaction | 12.44 | 14.27 | 16.07 | 9.28 |
| MaritalStatusSingle | 9.92 | 15.34 | 14.95 | 7.4 |
| YearsSinceLastPromotion | 9.8 | 11.3 | 14.22 | 9.34 |
| TrainingTimesLastYear | 8.91 | 13.6 | 13.86 | 9.56 |
| BusinessTravelTravel\_Frequently | 9.59 | 12.2 | 12.92 | 5.33 |
| MaritalStatusMarried | 6.66 | 12.02 | 12.15 | 3.51 |
| DepartmentSales | 8.11 | 8.79 | 11.26 | 3.53 |
| JobRoleLaboratory.Technician | 4.44 | 12.42 | 11.02 | 4.66 |
| DepartmentResearch...Development | 7.45 | 9.62 | 10.94 | 3.33 |
| JobRoleSales.Representative | 6 | 11.33 | 10.79 | 5.16 |
| GenderFemale | 7.95 | 8.68 | 10.28 | 2.82 |
| GenderMale | 7.85 | 8.06 | 10.08 | 2.97 |
| EducationFieldLife.Sciences | 6.71 | 8.8 | 9.96 | 3.06 |
| JobRoleResearch.Scientist | 8.07 | 5.51 | 9.88 | 2.86 |
| BusinessTravelTravel\_Rarely | 6.83 | 9.26 | 9.71 | 3.21 |
| JobRoleSales.Executive | 4.78 | 9.68 | 9.16 | 2.78 |
| EducationFieldMedical | 6.49 | 7.78 | 9.08 | 3.02 |
| JobRoleResearch.Director | 4.13 | 8.41 | 7.89 | 1.02 |
| EducationFieldMarketing | 4.79 | 7.89 | 7.83 | 2.71 |
| BusinessTravelNon.Travel | 5.09 | 7.29 | 7.46 | 1.74 |
| JobRoleManufacturing.Director | 2.84 | 7.95 | 7 | 1.58 |
| EducationFieldTechnical.Degree | 4.17 | 7.06 | 6.75 | 3.13 |
| EducationFieldOther | 4.93 | 4.4 | 6.55 | 1.73 |
| EducationFieldHuman.Resources | 1.72 | 5.94 | 4.76 | 1.07 |
| JobRoleHealthcare.Representative | 1.35 | 7.12 | 4.75 | 1.56 |
| DepartmentHuman.Resources | 2.27 | 4.86 | 4.37 | 1.35 |

## Tuning Random Forest

tRF <- tuneRF(x = rfHRDevScaledData\_df[,-c(1)], y=as.factor(rfHRDevScaledData\_df$TargetAttrition), mtryStart = 3, ntreeTry=100, stepFactor = 2, improve = 0.001, trace=TRUE, plot = TRUE, doBest = TRUE, nodesize = 150, importance=FALSE )

rfHRDevScaledData\_df$predict.class <- predict(tRF, rfHRDevScaledData\_df, type="class") rfHRDevScaledData\_df$predict.score <- predict(tRF, rfHRDevScaledData\_df, type="prob") class(rfHRDevScaledData\_df$predict.score)

mtry = 3 OOB error = 16.33%

Searching left ...

mtry = 2 OOB error = 16.33%

0 0.001

Searching right ...

mtry = 6 OOB error = 16.23%

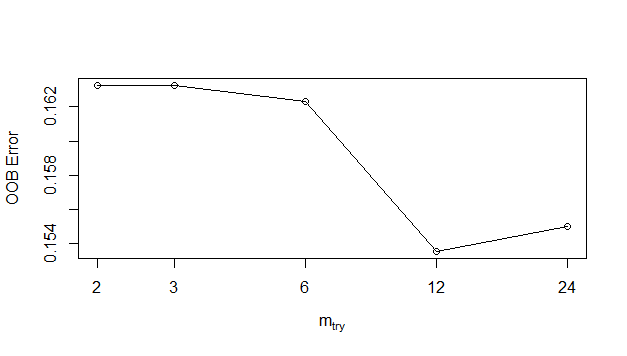
0.005952380952 0.001

mtry = 12 OOB error = 15.35%

0.05389221557 0.001

mtry = 24 OOB error = 15.5%

-0.009493670886 0.001

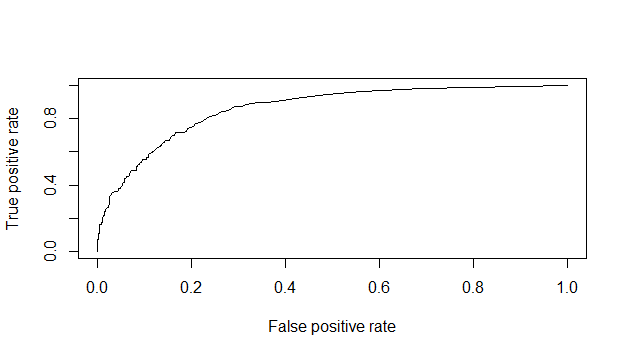


# Validating RF model

pred <- prediction(rfHRDevScaledData\_dfTargetAttrition) perf <- performance(pred, "tpr", "fpr") plot(perf) #Kolomorgov- Smirnof test KS <- max(attr(perf, 'y.values')[[1]]-attr(perf, 'x.values')[[1]])

KS

[1] 0.581083043



## Area under Curve

auc <- performance(pred,"auc"); auc <- as.numeric([auc@y.values](mailto:auc@y.values)) auc

[1] 0.856178447

## Gini Coefficient

gini = ineq(rfHRDevScaledData\_df$predict.score[,2], type="Gini") gini

[1] 0.7797781085

## Confusion matrix

confusionMatrix(table(rfHRDevScaledData\_dfpredict.class))

0 1

0 1718 4

1 303 33

**Accuracy : 0.850826**

95% CI : (0.8346926, 0.8659529)

No Information Rate : 0.9820214

P-Value [Acc > NIR] : 1

Kappa : 0.1493921

Mcnemar's Test P-Value : <0.0000000000000002

Sensitivity : 0.85007422

Specificity : 0.89189189

Pos Pred Value : 0.99767712

Neg Pred Value : 0.09821429

Prevalence : 0.98202138

Detection Rate : 0.83479106

Detection Prevalence : 0.83673469

Balanced Accuracy : 0.87098306

'Positive' Class : 0

## Scoring for Hold Out Samples

rfHRHoldOutScaledData = as.data.frame(HRHoldOutScaledData)

rfHRHoldOutScaledData$predict.class <- predict(tRF, rfHRHoldOutScaledData[,-1], type="class") rfHRHoldOutScaledData$predict.score <- predict(tRF, rfHRHoldOutScaledData[,-1], type="prob")

confusionMatrix(rfHRHoldOutScaledDatapredict.class)

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 740 4

1 121 17

**Accuracy : 0.8582766**

95% CI : (0.8334968, 0.8806333)

No Information Rate : 0.9761905

P-Value [Acc > NIR] : 1

Kappa : 0.1799438

Mcnemar's Test P-Value : <0.0000000000000002

Sensitivity : 0.8594657

Specificity : 0.8095238

Pos Pred Value : 0.9946237

Neg Pred Value : 0.1231884

Prevalence : 0.9761905

Detection Rate : 0.8390023

Detection Prevalence : 0.8435374

Balanced Accuracy : 0.8344948

'Positive' Class : 0

# Ensemble Model of Neural net and RF

## Averaging

## DevData

AverageProb\_Ensemble = (HRDevScaledData\_dfpredict.score[,2])/2 predict.score\_Ensemble = ifelse(AverageProb\_Ensemble>0.16,1,0) EnsembleModels = cbind(HRDevScaledData\_dfProb, HRDevScaledData\_dfpredict.score[,2], rfHRDevScaledData\_df$predict.class, AverageProb\_Ensemble, predict.score\_Ensemble) EnsembleModels = as.data.frame(EnsembleModels) names(EnsembleModels) = c("Target", "NeuralNet\_prob", "Neuralnet\_Prediected", "RF\_Probability", "RF\_prediected", "Ensemble\_Prob", "Ensemble\_Prediected") class(EnsembleModels) str(EnsembleModels)

## Comparison Study

## Ensemble vs Actual

confusionMatrix(EnsembleModelsEnsemble\_Prediected) #Accuracy is 91%

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 1590 132

1 79 257

Accuracy : 0.8974733

95% CI : (0.8835536, 0.9102486)

No Information Rate : 0.8109815

P-Value [Acc > NIR] : < 0.00000000000000022204

Kappa : 0.6471451

Mcnemar's Test P-Value : 0.0003438209

Sensitivity : 0.9526663

Specificity : 0.6606684

Pos Pred Value : 0.9233449

Neg Pred Value : 0.7648810

Prevalence : 0.8109815

Detection Rate : 0.7725948

Detection Prevalence : 0.8367347

Balanced Accuracy : 0.8066673

'Positive' Class : 0

# Conclusion

1. Random forest model provides lower overall accuracy and higher miss classification rate for the

given data compared to neural network and ensemble model

2. Ensemble model is marginally higher in overall accuracy than neural network model. Ensemble

also has lower miss classification rate and close to ideal KS statistics than others.

3. When compared with unbalanced data (skewed) set all models classification accuracy is above

90% as mentioned below

# Summary

Ensemble model outperforms Random Forest model when compared with most of model performance metrics like overall accuracy, miss classification error, area under curve percentage, Gini value and KS statistics whereas performs marginally better than Neural Network model in balanced data set. Hence, Null hypothesis (H0) is rejected and Alternative hypothesis (H1) is accepted.

# RCode

# a) Data Import (Target variable is "Attrition" column)

# b) Split the data in Dev & Hold Out sample (70:30)

# c) Perform Exploratory Data Analysis

# d) Identify columns which are of no use. drop those columns

# e) Write Hypothesis and validate the Hypothesis

# f) Build Neural Network Model (Development sample)

# g) Validate NN model on Hold Out. If need be improvize

# h) Build Random Forest Model

# i) Validate RF Model

# j) Compare NN with RF

# k) Combine NN and RF into Ensemble Model

# l) Check whether Ensemble Model Performance outperforms the individual RF & NN model

library(scales)

library(RColorBrewer)

library(neuralnet)

library(googleVis)

library(data.table)

library(caret)

library(ROCR)

library(randomForest)

library(ineq)

getwd()

setwd("C:/Home/Work/GreatLakes/Data Mining/Assignments/NeuralNet and RF")

darkCol = brewer.pal(9, "Set1")

HRSourceData = read.table("C:/Home/Work/GreatLakes/Data Mining/Assignments/NeuralNet and RF/1452762979\_586\_\_HR\_Employee\_Attrition\_Data.csv", header = T, sep=",")

#Data Preparation

# Target data is a factor of String, converting it to Integer of 0 or 1

Target\_Attrition = as.vector(HRSourceData$Attrition)

Target\_Attrition = replace(Target\_Attrition,Target\_Attrition=="No",0)

Target\_Attrition = replace(Target\_Attrition,Target\_Attrition=="Yes",1)

Target\_Attrition = as.integer(Target\_Attrition)

HRSourceData$TargetAttrition = Target\_Attrition

#plot(gvisTable(HRSourceData))

names(HRSourceData)

str(HRSourceData)

#List of Numerical Variable

# Age : COnt

# DailyRate : Ratio

# Distance from Home : Ratio

# Education : Ordinal

# Employee Count : Discrete

# Employee Number : Nominal

# Employee Satisfaction : Ordinal

# Hourly Rate : Cont, Ratio

# Job Involvement : Discrete

# Job Level : Ordinal

# Job Satisfaction : Ordinal

# Monthly Income : Ratio, Cont

# Monthly Rate : Ration, cont

# Num Company Worked : Discrete

# Percent Salaray Hike : Interval, Cont

# Performance Rating : Ordinal

# Relationship Satisfaction : Ordinal

# Standard Hours : Cont

# Stock Option Level : Ordinal

# Total Working Years : Discrete

# Training Times Last Year : Discrete

# Work Life Balance : Ordinal

# Years at Company : Cont, Interval

# Years in Current Role : Cont, Interval

# Years Since Last Promotion : Cont, Interval

# Years with Current Manager : Cont, Interval

#List of Categorical Variable

#BusinessTravel

#Department

#EducationField

#Gender

#JobRole

#MaritalStatus

#Over18

#OverTime

#Business Travel

plot(HRSourceData$BusinessTravel, col=darkCol, main = "Travel Details")

BusinessTravel = data.frame(summary(HRSourceData$BusinessTravel))

round((BusinessTravel/sum(BusinessTravel))\*100,2)

#Department

plot(HRSourceData$Department, col=darkCol, main="Department")

vDepartment = data.frame(summary(HRSourceData$Department))

round((vDepartment/sum(vDepartment))\*100,2)

#EducationField

plot(HRSourceData$EducationField, col=darkCol, main="Eduction Field")

vEducationField = data.frame(summary(HRSourceData$EducationField))

round((vEducationField/sum(vEducationField))\*100,2)

#Gender

plot(HRSourceData$Gender, col=darkCol,main="Gender")

vGender = data.frame(summary(HRSourceData$Gender))

round((vGender/sum(vGender))\*100,2)

#JobRole

plot(HRSourceData$JobRole, col=darkCol, main="Job Role")

vJobRole = data.frame(summary(HRSourceData$JobRole))

round((vJobRole/sum(vJobRole))\*100,2)

#MaritalStatus

plot(HRSourceData$MaritalStatus, col=darkCol, main="Marital Status")

vMaritalStatus = data.frame(summary(HRSourceData$MaritalStatus))

round((vMaritalStatus/sum(vMaritalStatus))\*100,2)

#OverTime

plot(HRSourceData$OverTime, col=darkCol, main="Over time")

vOverTime = data.frame(summary(HRSourceData$OverTime))

round((vOverTime/sum(vOverTime))\*100,2)

#Finding Ouliers in Cont data

x = subset(HRSourceData,

select = c("Age",

"DailyRate",

"DistanceFromHome",

"Education",

"EmployeeCount",

"EmployeeNumber",

"EnvironmentSatisfaction",

"HourlyRate",

"JobInvolvement",

"JobLevel",

"JobSatisfaction",

"MonthlyIncome",

"MonthlyRate",

"NumCompaniesWorked",

"PercentSalaryHike",

"PerformanceRating",

"RelationshipSatisfaction",

"StandardHours",

"StockOptionLevel",

"TotalWorkingYears",

"TrainingTimesLastYear",

"WorkLifeBalance",

"YearsAtCompany",

"YearsInCurrentRole",

"YearsSinceLastPromotion",

"YearsWithCurrManager"))

class(x)

y = scale(x[,])

boxplot(y, col=darkCol, main="Numerical Variables")

#Before splitting the data as development and holdout, let us convert the categorical variables to continous variables.

# Get rid of variables that are of no use

# StandardHours and Over18 are having same values for all observations, so we can remove those variables.

CleanedHRData = HRSourceData[,!(names(HRSourceData) %in% c("EmployeeCount", "StandardHours","Over18","EmployeeNumber"))]

# Converting Categorical Variables to COnt

#BusinessTravel

mBusinessTravel = model.matrix(~ BusinessTravel - 1, data = CleanedHRData)

head(mBusinessTravel)

CleanedHRData = data.frame(CleanedHRData, mBusinessTravel)

CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("BusinessTravel"))]

#Department

mDepartment = model.matrix(~ Department - 1, data = CleanedHRData)

head(mDepartment)

CleanedHRData = data.frame(CleanedHRData, mDepartment)

CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("Department"))]

#EducationField

mEducationField = model.matrix(~ EducationField - 1, data = CleanedHRData)

head(mEducationField)

CleanedHRData = data.frame(CleanedHRData, mEducationField)

CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("EducationField"))]

#Gender

mGender = model.matrix(~ Gender - 1, data = CleanedHRData)

head(mGender)

CleanedHRData = data.frame(CleanedHRData, mGender)

CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("Gender"))]

#JobRole

mJobRole = model.matrix(~ JobRole - 1, data = CleanedHRData)

head(mJobRole)

CleanedHRData = data.frame(CleanedHRData, mJobRole)

CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("JobRole"))]

#MaritalStatus

mMaritalStatus = model.matrix(~ MaritalStatus - 1, data = CleanedHRData)

head(mMaritalStatus)

CleanedHRData = data.frame(CleanedHRData, mMaritalStatus)

CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("MaritalStatus"))]

#OverTime

mOverTime = model.matrix(~ OverTime - 1, data = CleanedHRData)

head(mOverTime)

CleanedHRData = data.frame(CleanedHRData, mOverTime)

CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("OverTime"))]

CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("Attrition"))]

CleanedHRData = cbind(CleanedHRData[,(names(CleanedHRData) %in% c("TargetAttrition"))],

CleanedHRData[,!(names(CleanedHRData) %in% c("TargetAttrition"))])

names(CleanedHRData)[1] = "TargetAttrition"

str(CleanedHRData)

#Test for Significance using Anova

fit1 = aov(TargetAttrition ~ ., data = CleanedHRData)

summary(fit1)

#Following Variables came out as insignificant, it can be dropped from the data set.

# Df Sum Sq Mean Sq F value Pr(>F)

#Education 1 0.00 0.000 0.004 0.952001

#HourlyRate 1 0.03 0.033 0.320 0.571858

#MonthlyIncome 1 0.02 0.024 0.237 0.626222

#MonthlyRate 1 0.17 0.172 1.671 0.196237

#PercentSalaryHike 1 0.20 0.199 1.931 0.164728

#PerformanceRating 1 0.10 0.104 1.015 0.313771

#YearsAtCompany 1 0.03 0.030 0.295 0.586925

#JobRoleHuman.Resources 1 0.03 0.026 0.252 0.615773

#JobRoleManager 1 0.01 0.006 0.060 0.806153

#MaritalStatusDivorced 1 0.11 0.110 1.065 0.302052

CleanedHRData = CleanedHRData[,!(names(CleanedHRData) %in% c("Education","HourlyRate",

"MonthlyIncome","MonthlyRate",

"PercentSalaryHike", "PerformanceRating",

"YearsAtCompany","JobRoleHuman.Resources",

"JobRoleManager", "MaritalStatusDivorced"))]

fit2 = aov(TargetAttrition ~ ., data = CleanedHRData)

summary(fit2)

#Attrition Propotion

PopulationPropotion = sum(CleanedHRData$TargetAttrition)/nrow(CleanedHRData)

percent(PopulationPropotion)

#Common for both NN and RF

#Sampling

sampleIndex = sample(nrow(CleanedHRData), nrow(CleanedHRData)\*.7)

#Splitting of Data

#Training Data

HrDev = CleanedHRData[sampleIndex,]

#Testing Data

HrHoldOut = CleanedHRData[-sampleIndex,]

#Exloratory Data analysis (c)

#Data Count on Development and HOld Out

#Development Sample

dim(HrDev)

#Holdout Sample

dim(HrHoldOut)

#Propotions comparison

#Propotion in Dev

DevPropotion = sum(HrDev$TargetAttrition)/(nrow(HrDev))

DevPropotion

#Propotion in Holdout

HOPropotion = sum(HrHoldOut$TargetAttrition)/(nrow(HrDev))

HOPropotion

PopulationPropotion

rbind(PopulationPropotion,DevPropotion, HOPropotion) # Comparison of the Distribution of Data.

# Building Neural Network Model

# For Building the Neural net, we have to be keen in selecting following parameters

# number hidden layers

# Number of neuron (tumbrule is sqrt)

# epoh

# Activation Function

# avoiding overfitting.

# function for dealing with error

# threshold - important factor that decides over fitting.

# stopmax

# learning rate

names(CleanedHRData)

nn1 = neuralnet(TargetAttrition ~ Age + DailyRate + DistanceFromHome + EnvironmentSatisfaction + JobInvolvement + JobLevel + JobSatisfaction + NumCompaniesWorked + RelationshipSatisfaction + StockOptionLevel + TotalWorkingYears + TrainingTimesLastYear + WorkLifeBalance + YearsInCurrentRole + YearsSinceLastPromotion + YearsWithCurrManager + BusinessTravelNon.Travel + BusinessTravelTravel\_Frequently + BusinessTravelTravel\_Rarely + GenderFemale + GenderMale + JobRoleHealthcare.Representative + JobRoleLaboratory.Technician + JobRoleManufacturing.Director + JobRoleResearch.Director + JobRoleResearch.Scientist + JobRoleSales.Executive + JobRoleSales.Representative + MaritalStatusMarried + MaritalStatusSingle + OverTimeNo + OverTimeYes ,

data = HrDev,

hidden = c(3,2),

linear.output = FALSE,

err.fct = "sse",

lifesign = "full",

lifesign.step = 10,

threshold = 0.01,

stepmax = 4000)

plot(nn1)

nn1$net.result[[1]]

HrDev$Prob = as.numeric(nn1$net.result[[1]])

hist(HrDev$Prob)

#Cleaning the Extreme Values

TargetAttrition = HrDev[,1]

TargetAttrition

HRDevScaledData = scale(HrDev[,-1])

HRDevScaledData = cbind(TargetAttrition,HRDevScaledData )

#plot(gvisTable(data.frame(HRDevScaledData)))

#Distribution is not Proper, so, let us scale the data

nn2 = neuralnet(TargetAttrition ~ Age + DailyRate + DistanceFromHome + EnvironmentSatisfaction + JobInvolvement + JobLevel + JobSatisfaction + NumCompaniesWorked + RelationshipSatisfaction + StockOptionLevel + TotalWorkingYears + TrainingTimesLastYear + WorkLifeBalance + YearsInCurrentRole + YearsSinceLastPromotion + YearsWithCurrManager + BusinessTravelNon.Travel + BusinessTravelTravel\_Frequently + BusinessTravelTravel\_Rarely + GenderFemale + GenderMale + JobRoleHealthcare.Representative + JobRoleLaboratory.Technician + JobRoleManufacturing.Director + JobRoleResearch.Director + JobRoleResearch.Scientist + JobRoleSales.Executive + JobRoleSales.Representative + MaritalStatusMarried + MaritalStatusSingle + OverTimeNo + OverTimeYes ,

data = HRDevScaledData,

hidden = c(3,2),

linear.output = FALSE,

err.fct = "sse",

lifesign = "full",

lifesign.step = 10,

threshold = 0.01,

stepmax = 4000)

plot(nn2)

HRDevScaledData\_df = as.data.frame(HRDevScaledData)

HRDevScaledData\_df$Prob = as.numeric(nn2$net.result[[1]])

quantile(HRDevScaledData\_df$Prob, c(0,10,20,30,40,50,60,70,80,90,100)/100)

hist(HRDevScaledData\_df$Prob)

#Basic Confusion Matrix

HRDevScaledData\_df$Predicted.Score = ifelse(HRDevScaledData\_df$Prob>0.16,1,0)

with(HRDevScaledData\_df, table(TargetAttrition,Predicted.Score))

#Detailed Results

confusionMatrix(table(HRDevScaledData\_df$Predicted.Score, HRDevScaledData\_df$TargetAttrition), dnn = c("Predicted Attrition","Actual Attrition"))

#Scoring Hold-out data using NN

HTargetAttrition = HrHoldOut[,1]

HTargetAttrition

HRHoldOutScaledData = scale(HrHoldOut[,-1])

HRHoldOutScaledData = cbind(HTargetAttrition,HRHoldOutScaledData )

HoldOutOutput = compute(nn2,HRHoldOutScaledData[,c("Age","DailyRate","DistanceFromHome","EnvironmentSatisfaction","JobInvolvement","JobLevel","JobSatisfaction","NumCompaniesWorked","RelationshipSatisfaction","StockOptionLevel","TotalWorkingYears","TrainingTimesLastYear","WorkLifeBalance","YearsInCurrentRole","YearsSinceLastPromotion","YearsWithCurrManager","BusinessTravelNon.Travel","BusinessTravelTravel\_Frequently","BusinessTravelTravel\_Rarely","GenderFemale","GenderMale","JobRoleHealthcare.Representative","JobRoleLaboratory.Technician","JobRoleManufacturing.Director","JobRoleResearch.Director","JobRoleResearch.Scientist","JobRoleSales.Executive","JobRoleSales.Representative","MaritalStatusMarried","MaritalStatusSingle","OverTimeNo","OverTimeYes")])

HRHoldOutScaledData\_df = as.data.frame(HRHoldOutScaledData)

HRHoldOutScaledData\_df$Prob = HoldOutOutput$net.result[,1]

HRHoldOutScaledData\_df$Predicted.Score = ifelse(HRHoldOutScaledData\_df$Prob>0.16,1,0)

confusionMatrix(table(HRHoldOutScaledData\_df$Predicted.Score, HRHoldOutScaledData\_df$HTargetAttrition))

cm\_HRHOldout = confusionMatrix(table(HRHoldOutScaledData\_df$Predicted.Score, HRHoldOutScaledData\_df$HTargetAttrition))

names(HRDevScaledData\_df)

#Random Forest

rfHRDevScaledData\_df = HRDevScaledData\_df[,-c(43:45)]

RF <- randomForest(as.factor(TargetAttrition) ~ ., data = rfHRDevScaledData\_df[,-1],

ntree=500, mtry = 3, nodesize = 10,

importance=TRUE)

print(RF)

#Ploting Random Forest

plot(RF)

legend("topright", c("OOB", "0", "1"), text.col=1:6, lty=1:3, col=1:3)

title(main="Error Rates Random Forest \n HR data - Development")

RF$err.rate

#Importance of variabels that are used for Random Forest

impVar <- round(randomForest::importance(RF), 2)

impVar[order(impVar[,3], decreasing=TRUE),]

#tuning Random Forest

## Tuning Random Forest

tRF <- tuneRF(x = rfHRDevScaledData\_df[,-c(1)],

y=as.factor(rfHRDevScaledData\_df$TargetAttrition),

mtryStart = 3,

ntreeTry=100,

stepFactor = 2,

improve = 0.001,

trace=TRUE,

plot = TRUE,

doBest = TRUE,

nodesize = 150,

importance=FALSE

)

rfHRDevScaledData\_df$predict.class <- predict(tRF, rfHRDevScaledData\_df, type="class")

rfHRDevScaledData\_df$predict.score <- predict(tRF, rfHRDevScaledData\_df, type="prob")

class(rfHRDevScaledData\_df$predict.score)

#Evaluating The RF model

pred <- prediction(rfHRDevScaledData\_df$predict.score[,2], rfHRDevScaledData\_df$TargetAttrition)

perf <- performance(pred, "tpr", "fpr")

plot(perf)

#Kolomorgov- Smirnof test

KS <- max(attr(perf, 'y.values')[[1]]-attr(perf, 'x.values')[[1]])

KS

#ROCR

pred = prediction(rfHRDevScaledData\_df$predict.score[,2], rfHRDevScaledData\_df$TargetAttrition)

pref = performance(pred,"tpr","fpr")

plot(pref)

#Area under Curve

auc <- performance(pred,"auc");

auc <- as.numeric(auc@y.values)

auc

## Gini Coefficient

gini = ineq(rfHRDevScaledData\_df$predict.score[,2], type="Gini")

gini

## Confusion matrix

confusionMatrix(table(rfHRDevScaledData\_df$TargetAttrition,rfHRDevScaledData\_df$predict.class))

#Scoring for Hold Out Samples

rfHRHoldOutScaledData = as.data.frame(HRHoldOutScaledData)

rfHRHoldOutScaledData$predict.class <- predict(tRF, rfHRHoldOutScaledData[,-1], type="class")

rfHRHoldOutScaledData$predict.score <- predict(tRF, rfHRHoldOutScaledData[,-1], type="prob")

confusionMatrix(rfHRHoldOutScaledData$HTargetAttrition, rfHRHoldOutScaledData$predict.class)

#Ensemble Model of Neural net and RF

#Averaging

#DevData

AverageProb\_Ensemble = (HRDevScaledData\_df$Prob + rfHRDevScaledData\_df$predict.score[,2])/2

predict.score\_Ensemble = ifelse(AverageProb\_Ensemble>0.16,1,0)

EnsembleModels = cbind(HRDevScaledData\_df$TargetAttrition,

HRDevScaledData\_df$Prob,

HRDevScaledData\_df$Predicted.Score,

rfHRDevScaledData\_df$predict.score[,2],

rfHRDevScaledData\_df$predict.class,

AverageProb\_Ensemble,

predict.score\_Ensemble)

EnsembleModels = as.data.frame(EnsembleModels)

names(EnsembleModels) = c("Target",

"NeuralNet\_prob",

"Neuralnet\_Prediected",

"RF\_Probability",

"RF\_prediected",

"Ensemble\_Prob",

"Ensemble\_Prediected")

class(EnsembleModels)

str(EnsembleModels)

#Comparison Study

#Ensemble vs Actual

confusionMatrix(EnsembleModels$Target, EnsembleModels$Ensemble\_Prediected)

#Accuracy is 91%