Data Mining Assignment Report 2

Date: 7 July 2016

Document control

Response Strategic Leadership: Rajesh Jakhotia
Response Manager: Dr. P K Vishwanathan
Author: Anshuman Biswal,Vishnu G,Bharat Pai,Akash Alex,Himanshu Jain
Version: 1
Status: Final

Purpose of this Data Analysis Report: to develop analytical business model using Random Forest & Neural Network Technique using HR_Employee_Attrition_Data.csv

Table of contents

1. Introduction

The data has information about a company's employee details from the responses in the exit interviews and has the details of the employees attrition data set

1.1 Purpose of the analysis

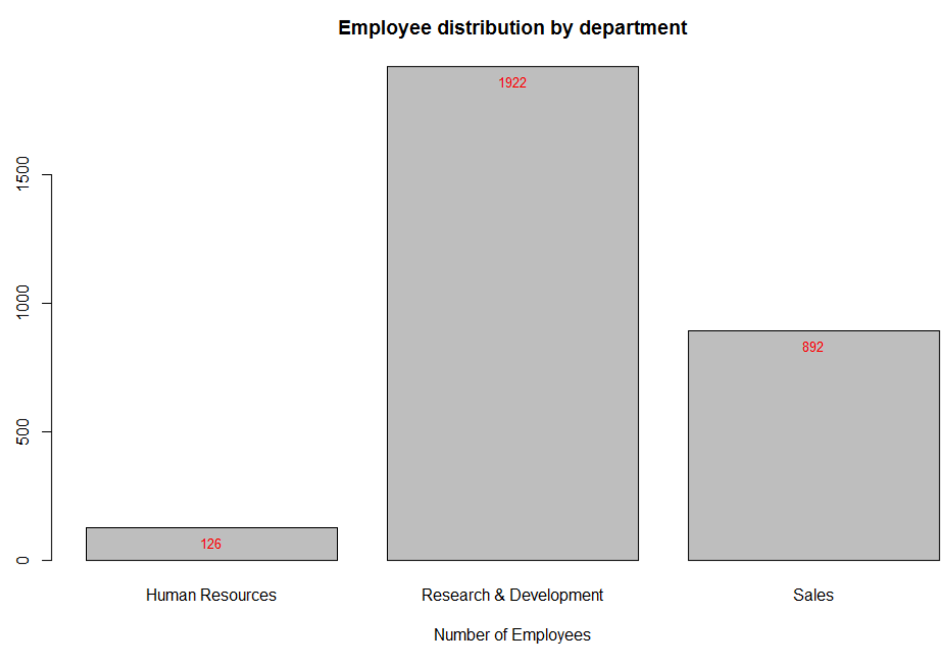
To develop an analytic model for the company to predict the Attrition value using the dataset provided ; HR_Employee_Attrition_Data.csv

2. Discussion

2.1 Exploratory Data Analysis

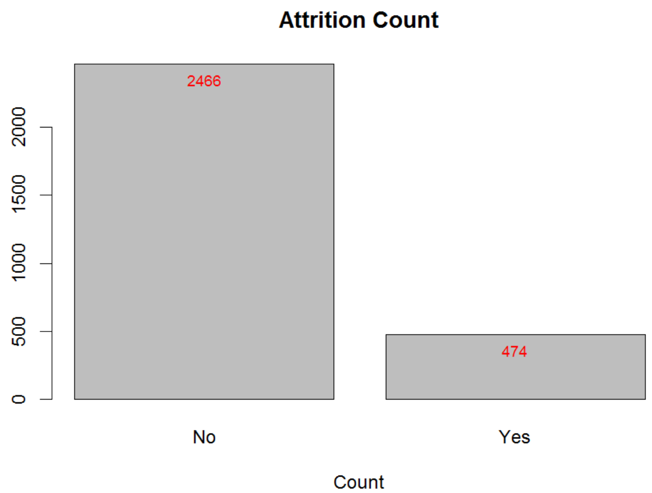
1> The data set has 2940 observations and each observation has data for 35 variables.

2.1.1 Distribution of dataset by department



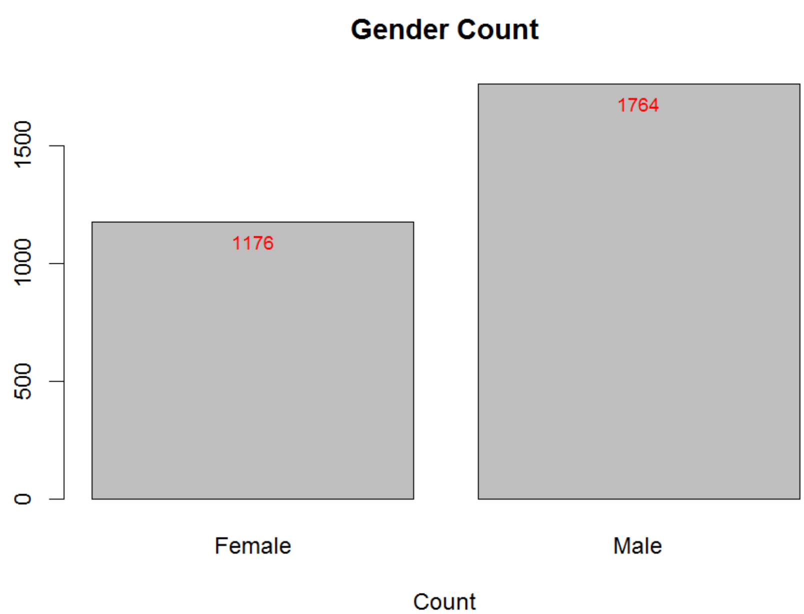
2> The data set has data of 1922 Research and Development department employees, 892 Sales employees and 126 Human Resources employees

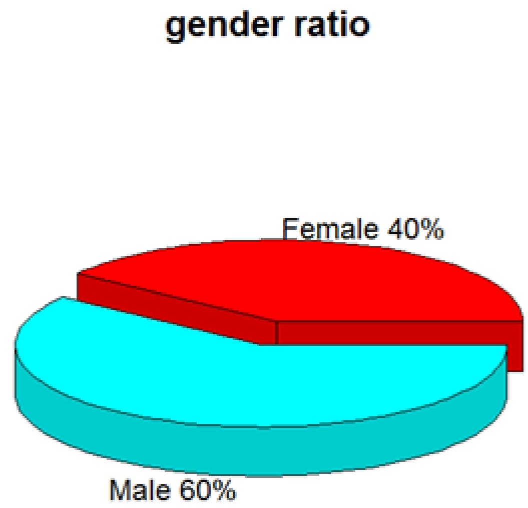
2.1.2 Distribution of dataset by Attrition



3> From the above plot out of 2510 entries 474 has a Attrition value of Yes and 2466 has Attrition value as No

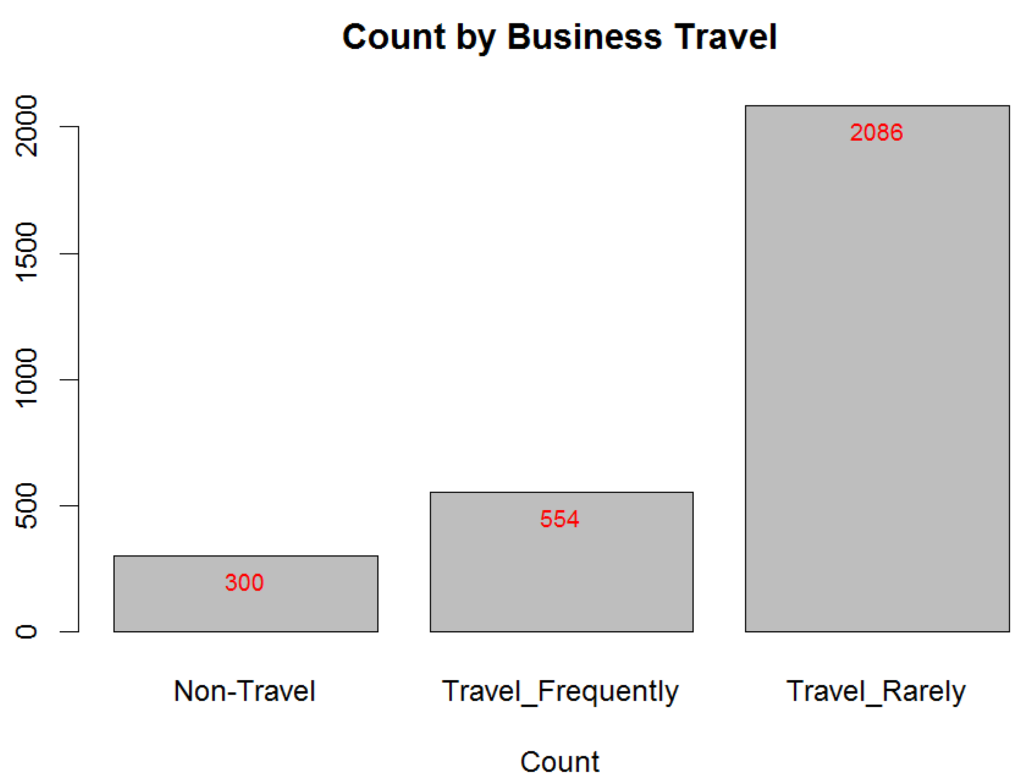
2.1.3 Distribution of dataset by Gender

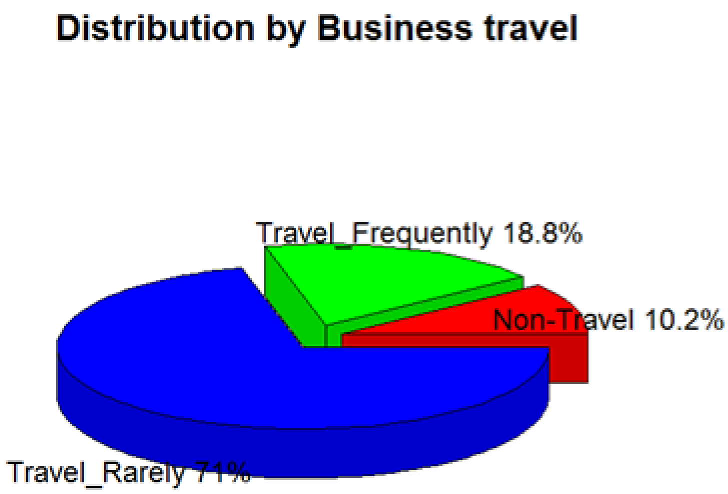




4> The data consists of 1764 male and 1176 female employees which implies that the data consists of 60% of male employees and 40% of female employees

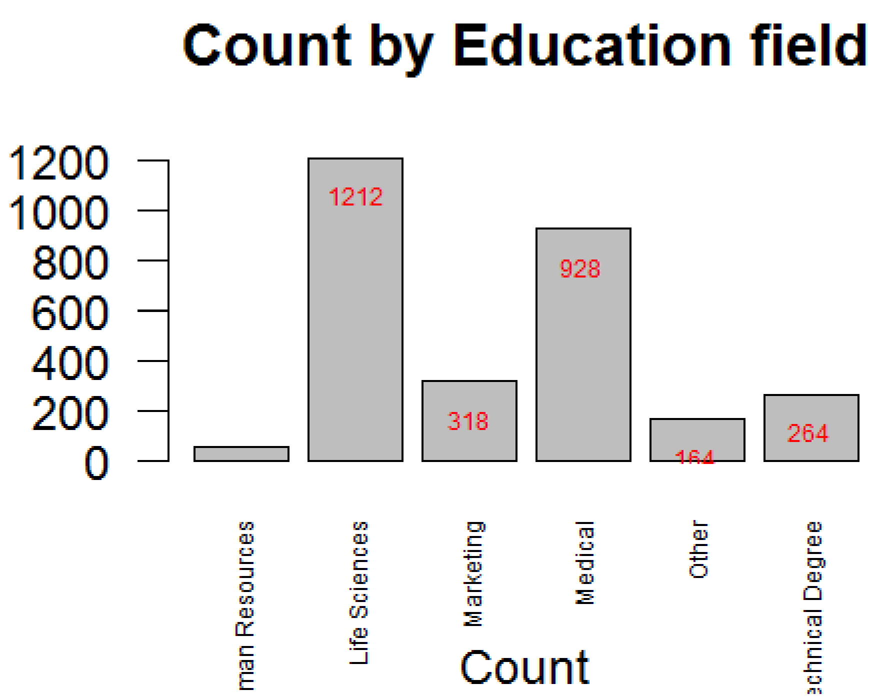
2.1.4 Distribution of dataset by Business travel

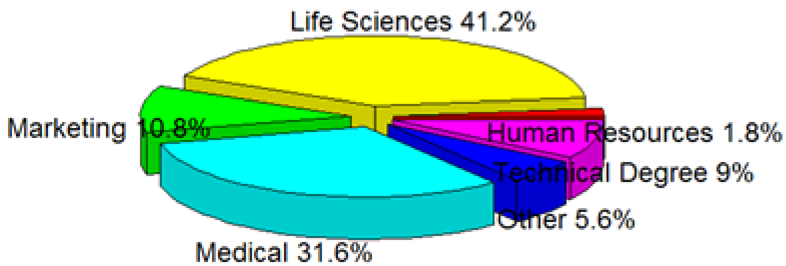




5> From the above two graphs it is seen that the data has 71% of the employees who travel rarely,18.8% of employess travel frequently and only 10.2% of employees don't travel at all

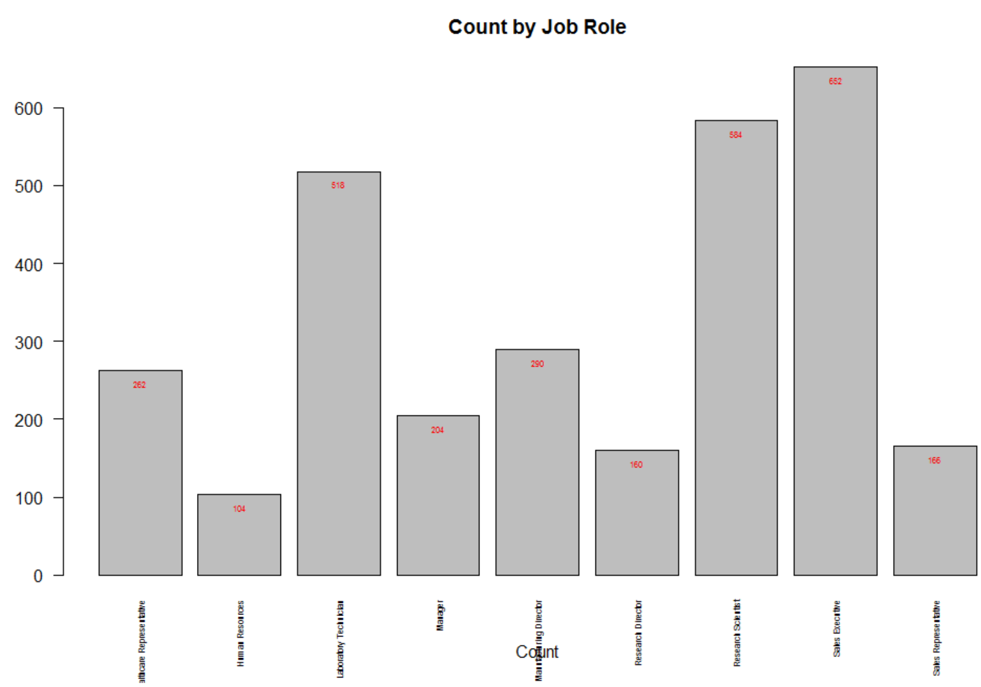
2.1.5 Distribution of dataset by Education field

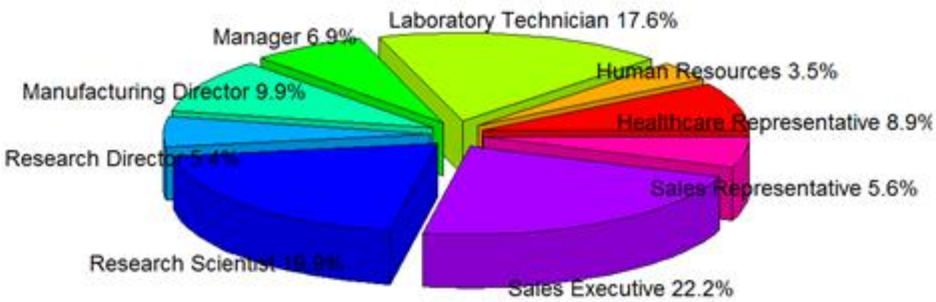




6> The data consists of employees with the following distribution of education field 41.2% of employees are from Life Sciences subject which counts to 1212,31.6% are from medical background which accounts to 928 count, 318 employees are from marketing ed ucation background which accounts to 10.8% of the data set,264 employees have technica l degree which accounts to 9% of population,164 employees have other education field w hich is 5.6% of data and 1.8% of the population are from Human Resources stream which accounts to only 54 employees

2.1.6 Distribution of dataset by job role





7> As per above pie and bar plot it is seen that the data consists of 22.2% of Sales Executive and 19.9 percent of data are of research scientists. Human Resource accounts for 3.5% of data as per job role. As per job role the contribution of each role in the data set are as fo llows

Healthcare Representative = 262 (8.9% of data set)

Human Resources = 104 (3.5% of data set)

Laboratory Technician = 518 (17.6% of data set)

Manager = 204 (6.9% of data set)

Manufacturing Director = 290 (9.9% of data set)

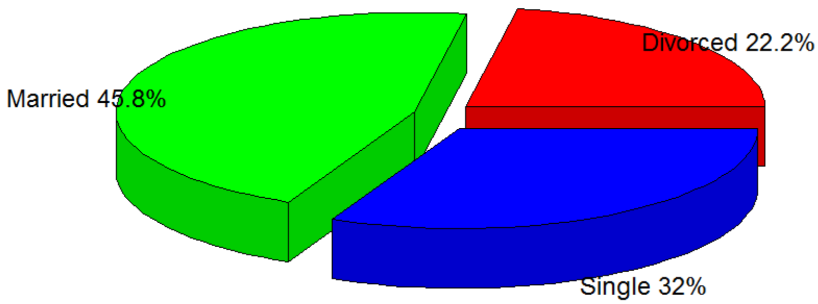
Research Director = 160 (5.4% of data set)

Research Scientist = 584 (19.9% of data set)

Sales Executive = 652 (22.2% of data set)

Sales Representative =166 (5.6% of data set)

2.1.7 Distribution of dataset by marital status



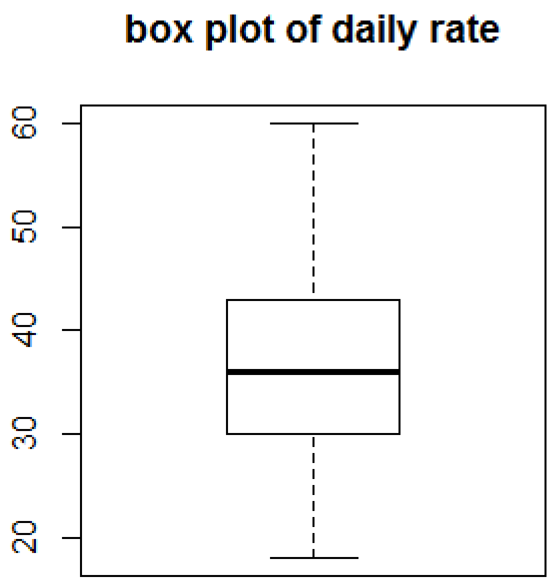
8> 45.8% of the population are married ,32% are single and 22.2% of the dataset are divorce d.

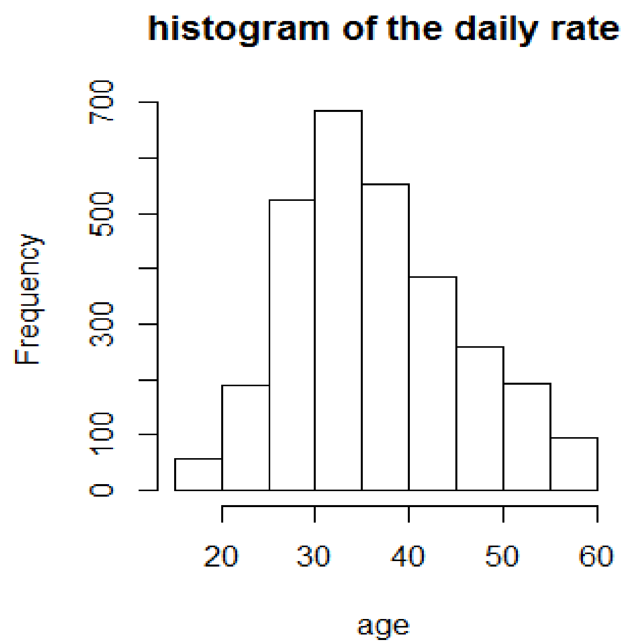
9> It is good to see that there is no employee whose age is below 18 years of age. All 2940 e mployees, whose data were taken are of 18 years and above age

2.1.8 Analysis of dataset for the age variable

10> It is inferred that the age variable is not much skewed. From the histogram it is seen tha t the maximum age is 60 and minimum is 18 years . Median is at 36. 75% of the data has a ge below 43. 50% of the age lie above 36 and 50% lie between 30 and 36 years. Mean is s lightly greater than median means the data might be very slightly positively skewed.

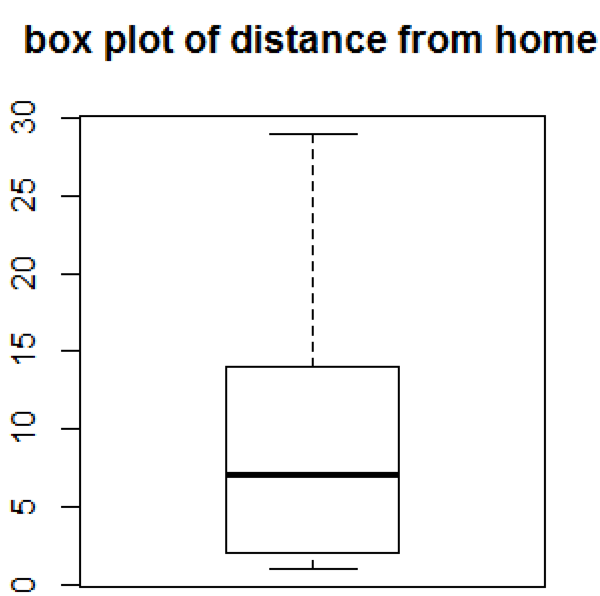
2.1.9 Analysis of dataset for daily rate variable

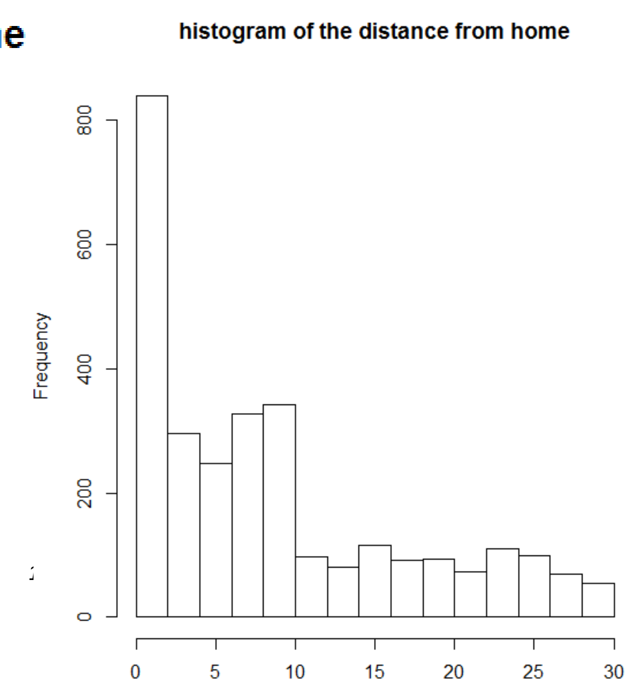




11> It is inferred that the daily rate variable is not much skewed. From the histogram it is seen that the maximum daily rate is 1499 unit and minimum is 102 unit. median is at 802. 75% of the data has daily rate below 1157. 50% of the daily rate lie above 1157 and 50% lie between 465 and 802. Mean is slightly greater than median means the data might be very slightly positively skewed.

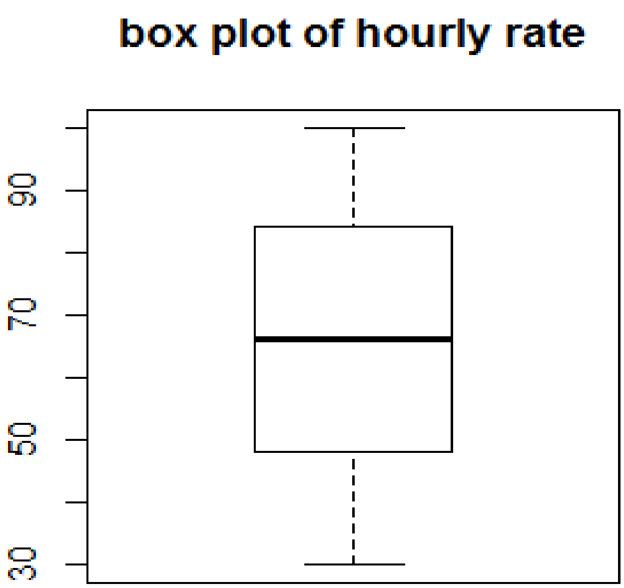
2.1.10 Analysis of dataset for distance from home variable

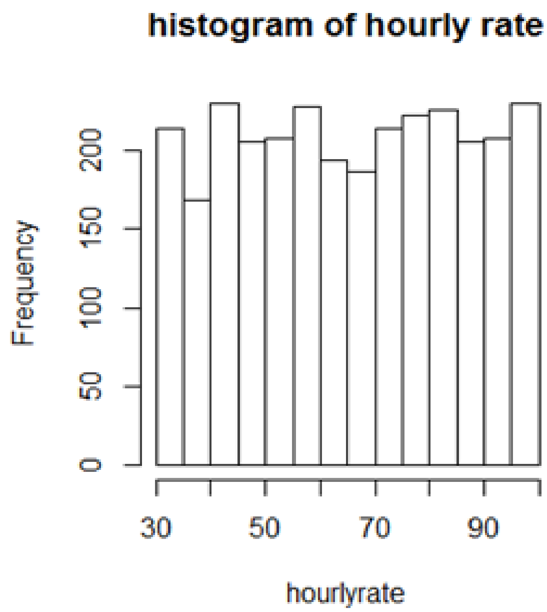




12> It is inferred that the distance from home variable is skewed. From the histogram it is se en that the data is positively skewed. maximum distance from home value is 29 unit and m inimum is 1unit. median is at 7unit. 75% of the data has distance from home value below 14unit. 50% of the data has distance from home value lie above 9.193unit and 50% of val ue lie between 2unit and 7unit. Mean is greater than median means the data is positively s kewed.

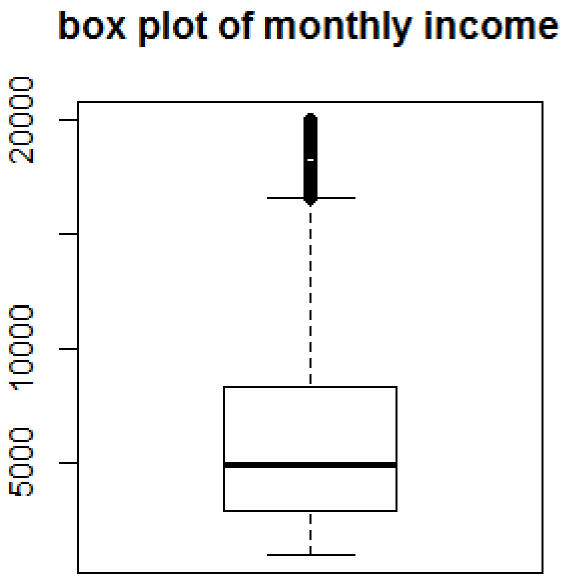
2.1.11 Analysis of dataset for hourly rate variable

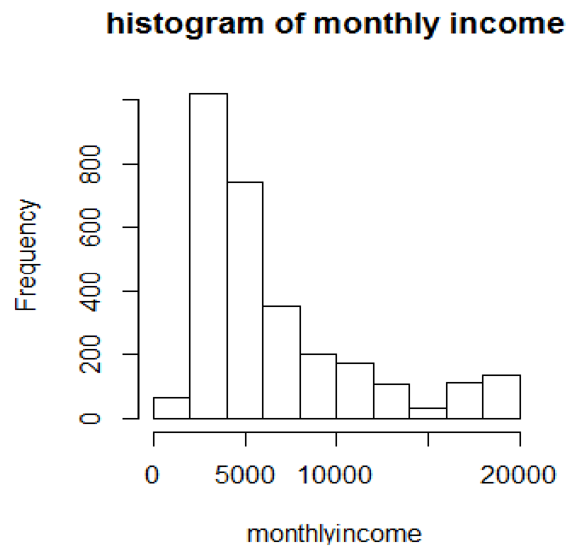




13> It is inferred that the hourly rate variable is almost normal. From the histogram it is seen that the maximum hourly rate is 100 unit and minimum is 30 unit. median is at 66. 75% o f the data has hourly rate below 84unit. 50% of the hourly rate lie above 65.89unitand 50 % lie between 48 and 66.

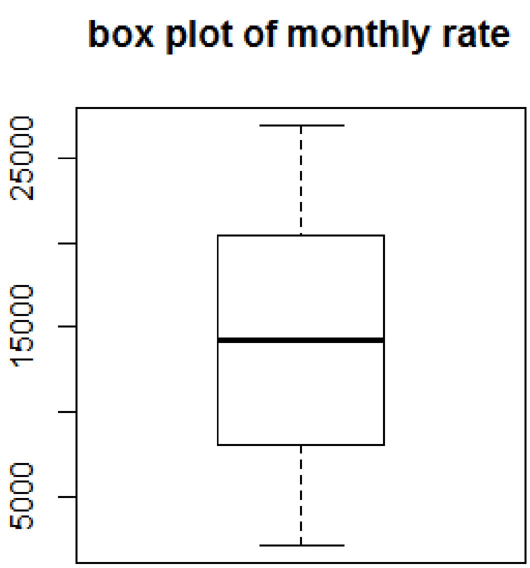
2.1.12 Analysis of dataset for monthly income variable

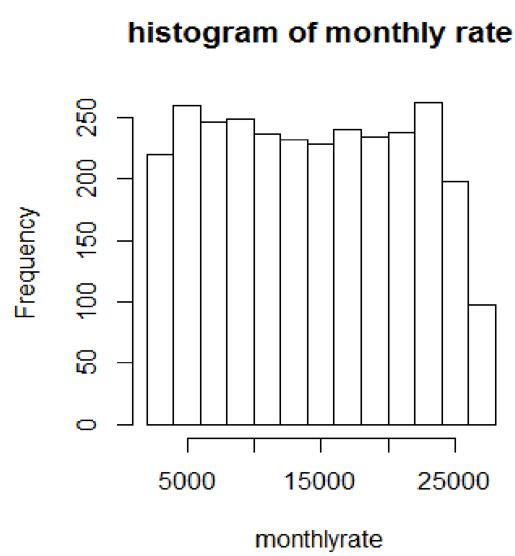




14> It is inferred that the monthly income is skewed and also there are more outliers. From t he histogram it is seen that the minimum monthly income is 1009 and maximum is 19999 which is an outlier. As there are outliers so we cannot consider any summary data.

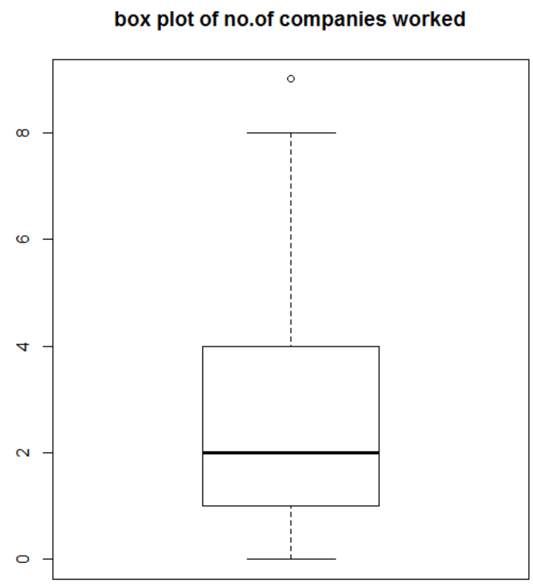
2.1.13 Analysis of dataset for monthly rate variable

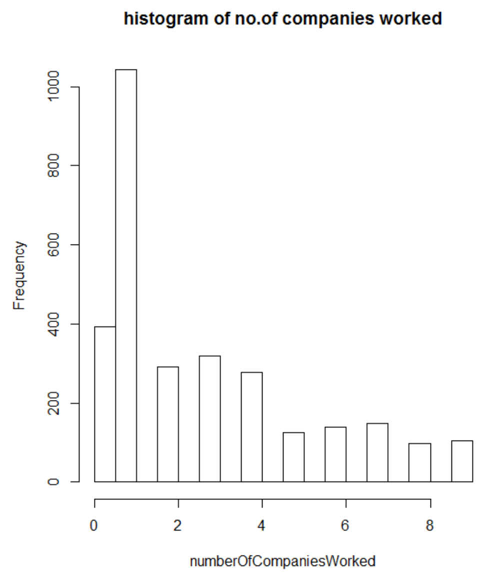




15> It is inferred that the monthlyrate variable is almost normal. From the histogram it is seen that the maximum monthly rate is 26999 unit and minimum is 2094 unit. median is at 14236unit. 75% of the data has monthly rate below 20462unit. 50% of the hourly rate lie above 14313unit and 50% of data lie between 8045 and 14236

2.1.14 Analysis of dataset for no. of companies variable





16> From the boxplot it is inferred that the NoOFCompaniesWorked variable is skewed. From the histogram it is seen that the minimum value is 0 and maximum value is 9. median is at 2. 75% of the data has value below 4. 50% of the value lie above 2.693 and 50% of data lie between 1 and 2

2.1.15 Analysis of dataset for other variable

PercentSalaryHike	TotalWorkingYears	TrainingTimesLastYear
# Min. : 11	Min. : 0.00	Min. :0.000
# 1st Qu.: 12	1st Qu.: 6.00	1st Qu.:2.000
# Median : 14	Median :10.00	Median :3.000
# Mean : 15.21	Mean :11.28	Mean :2.799
# 3rd Qu.: 18	3rd Qu.:15.00	3rd Qu.:3.000
# Max. :25	Max. :40.00	Max. :6.000

17> PercentSalaryHike: The data seems to be normal and max value is 25 and min is 11.75% of data has value below 18 and 50% of data have values above 15.21 and 50% of data lie between 12 and 14

18> TotalWorkingHours: Mean is slightly greater than median so data seems to be slightly positively skewed. 75% od data has values less than 15 and 50% of data lie above 11.28 and 50% lie between 6 and 10. Minimum value is 0 and maximum is 40.

19> TrainingTimeLastYear: Minimum value of this variable is 0 and maximum is 6. 75% of data lie below 3 and 50% of data lie above 2.799.50% of data lie between 2 and 3.Mean is slightly less than 3 ,so data is bit negatively skewed.

```
# YearsAtCompany      YearsInCurrentRole    YearsSinceLastPromotion   YearsWithCurrManager
# Min.   : 0.000      Min.   : 0.000        Min.   : 0.000            Min.   : 0.000
# 1st Qu.: 3.000      1st Qu.: 2.000        1st Qu.: 0.000            1st Qu.: 2.000
# Median : 5.000      Median : 3.000        Median : 1.000            Median : 3.000
# Mean   : 7.008      Mean   : 4.229        Mean   : 2.188            Mean   : 4.123
# 3rd Qu.: 9.000      3rd Qu.: 7.000        3rd Qu.: 3.000            3rd Qu.: 7.000
# Max.   :40.000      Max.   :18.000        Max.   :15.000            Max.   :17.000
```

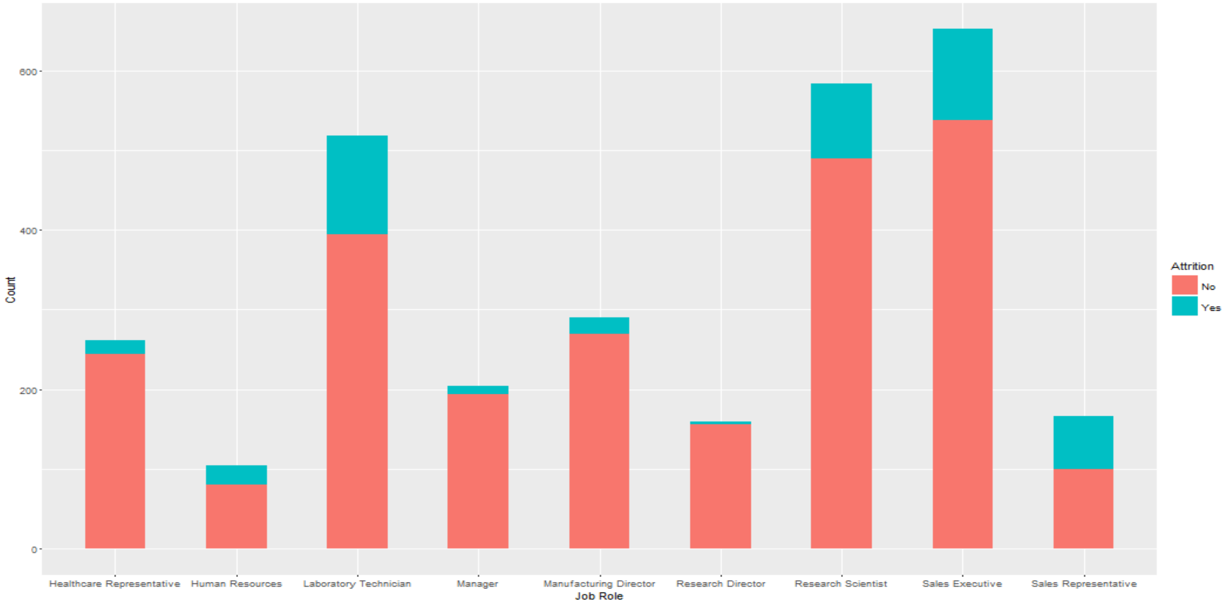
20> Years at company: Minimum is 0 and maximum is 40. Mean is greater than median says data is bit positively skewed.75% of data lie below 9 years.50% of data lie above 7.008 years and 50% of data lie between 3 and 5 years.

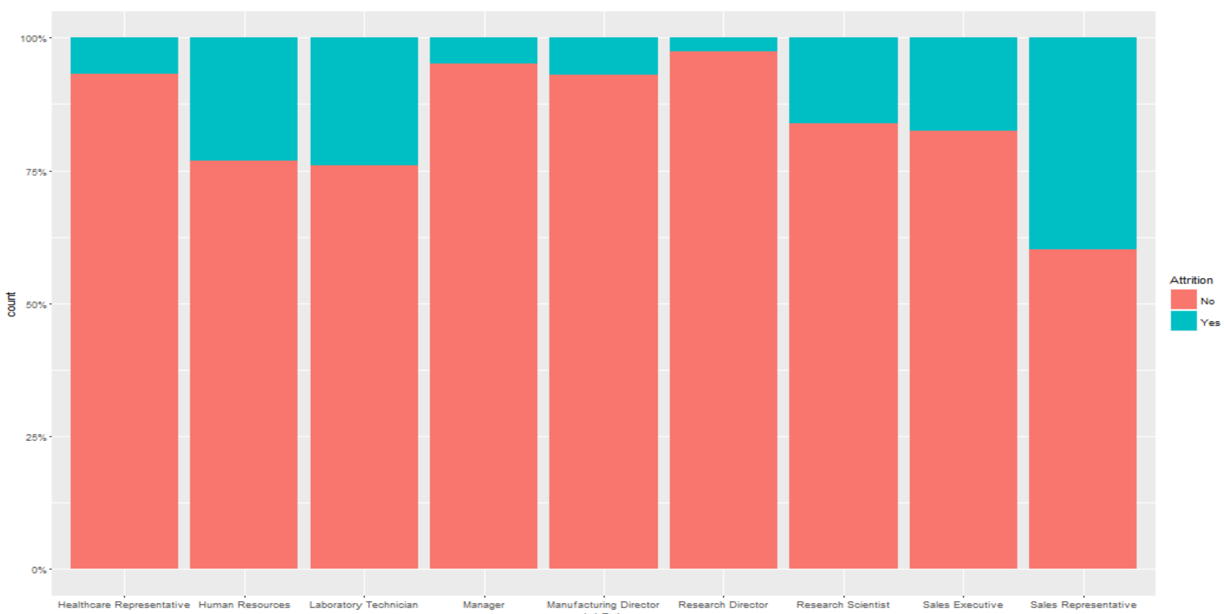
21> Years in current role: Minimum is 0 and maximum is 18.75% of data lie below 7 . 50% of data lie between 2 and 3 and 50% of data lie above 4.229. Mean greater than median says data is positively skewed.

22> Years since last promotion:minimum is 0 and max is 15.mean is greater than median so data is positively skewed.75% of data lie below 3. 50% of data lie above 2.188 and 50% of data lie between 0 and 1.

23> YearsWithCurrentManager: Minimum is 0 and maximum is 17. 75% of data lie below 7 . 50% of data lie between 2 and 3 and 50% of data lies above 4.123.Mean is greater than median says data is positively skewed.

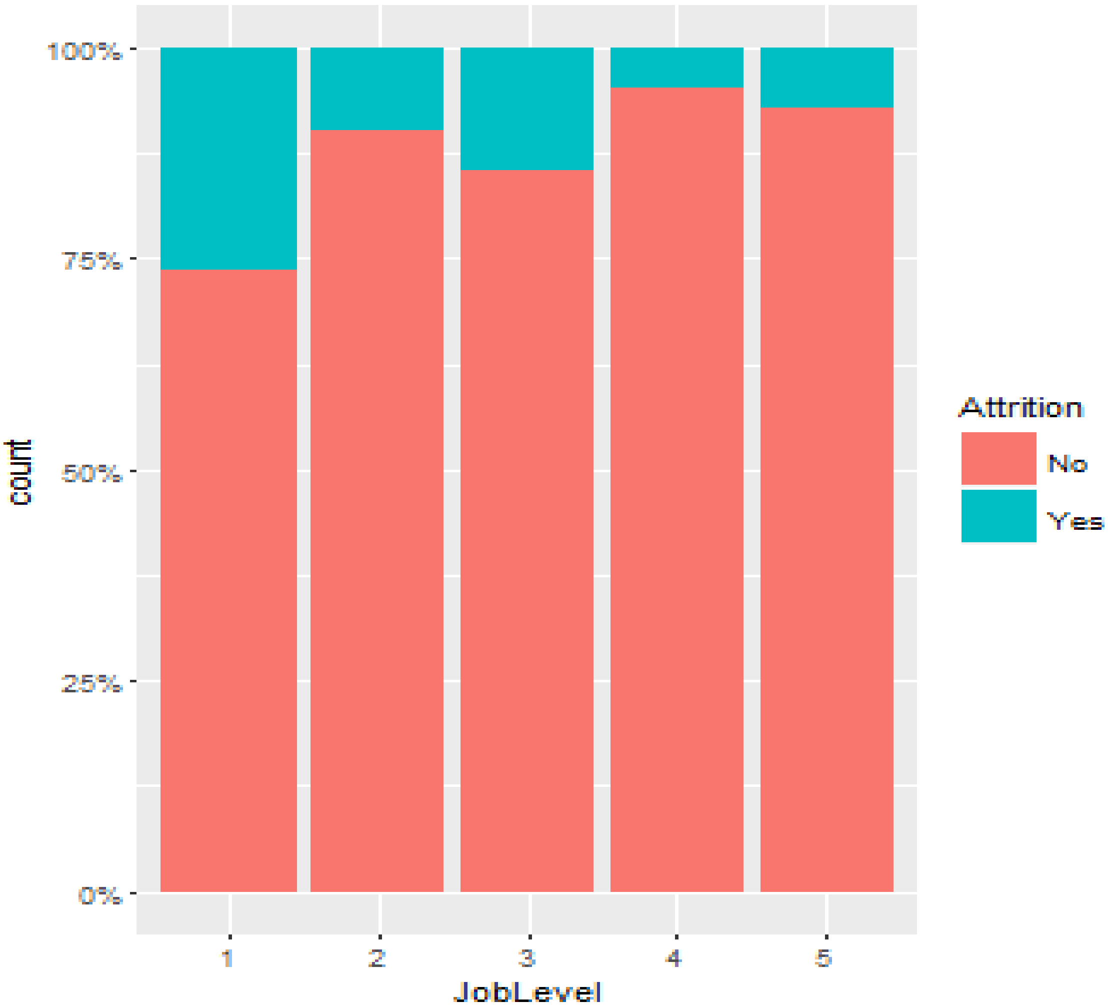
2.1.16 Attrition rate across job roles:





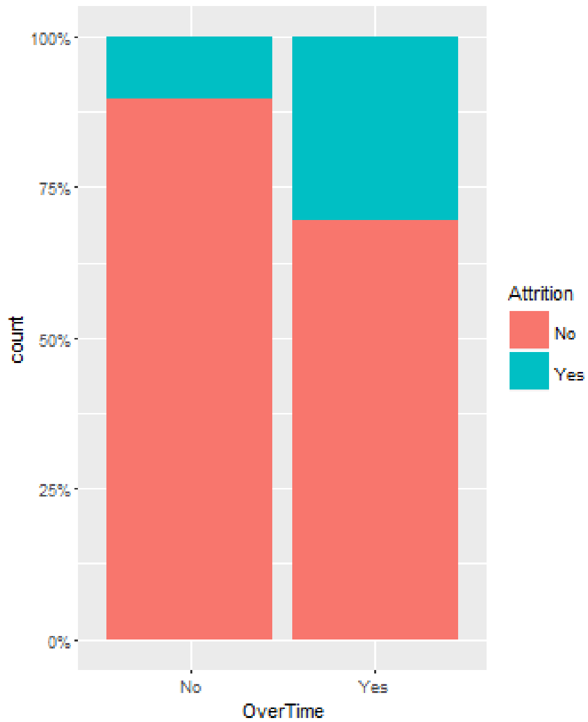
24> Sales Representative job role has significantly more attrition relative to other job roles

2.1.17 Attrition rate across job level:



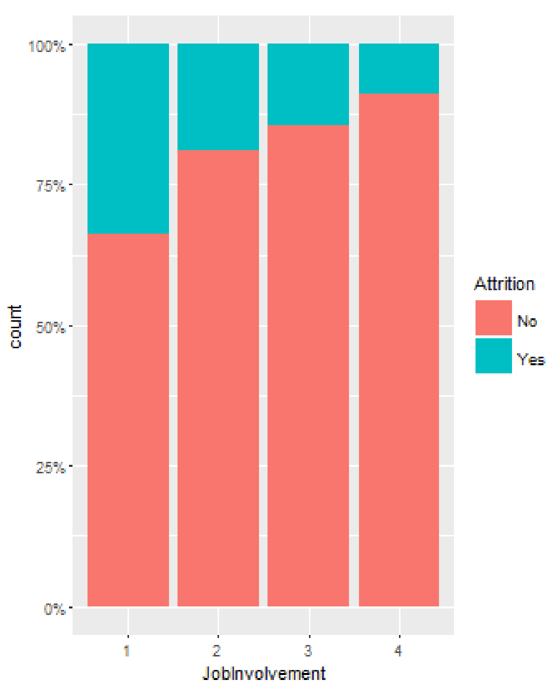
25> Job level 1 has higher attrition rates over other levels and after job level1 job level 3 has maximum attritions

2.1.18 Attrition rate because of Overtime values:



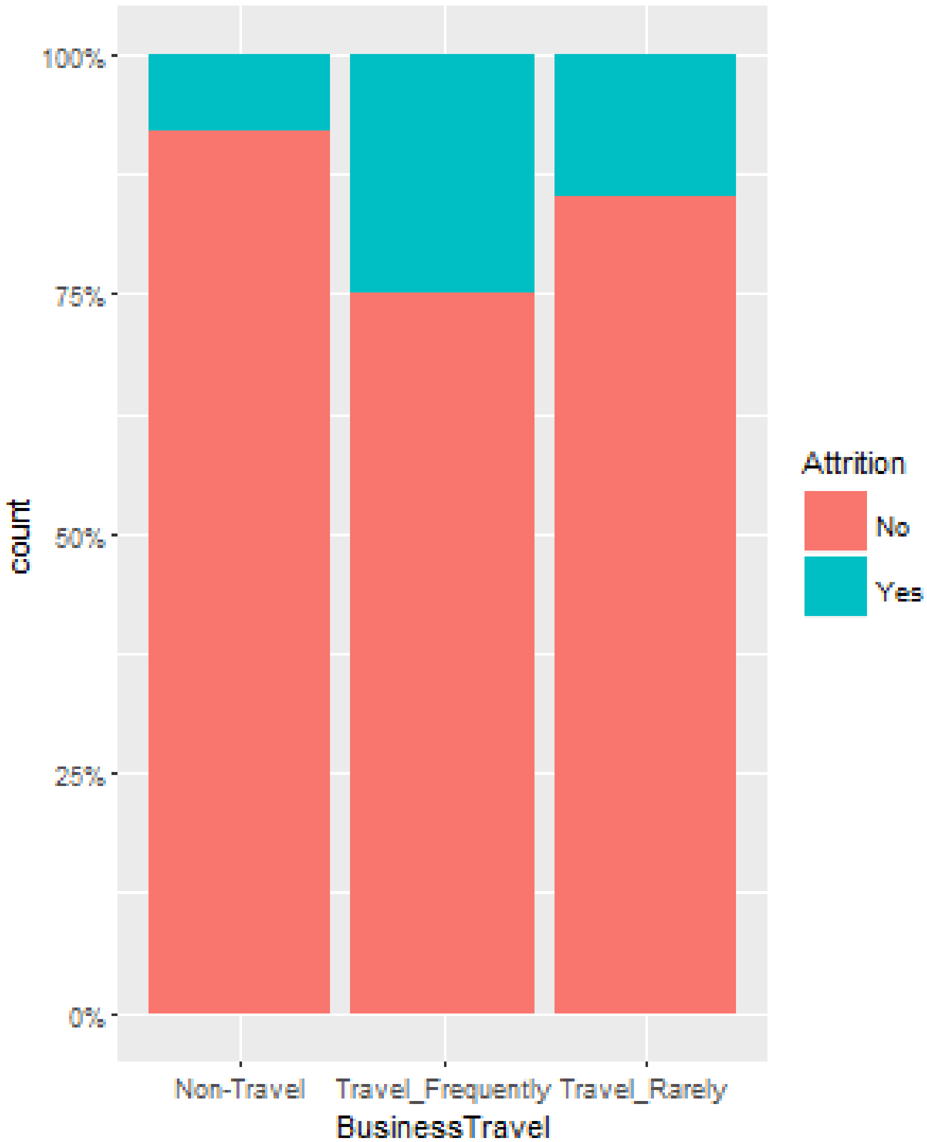
26> Attrition rate is higher in those who does overtime than the employees who does not do overtime.

2.1.19 Attrition rate because of job involvement:



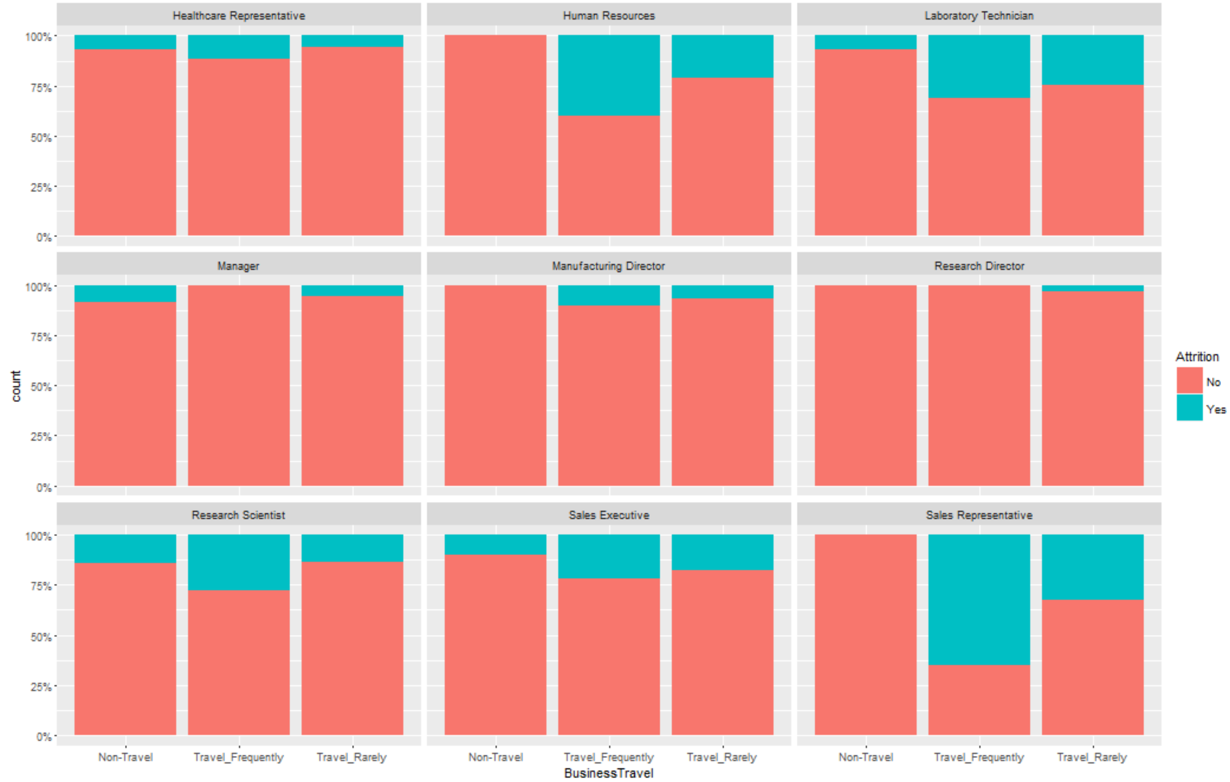
27> JobInvolvement = 1 has higher attrition than the other.

2.1.20 Attrition rate because of amount of employee travels:



28> Attrition is high in frequent traveller employee

2.1.21 Attrition rate because of job roles with the frequency of travel:



29> From the above plot it is inferred that attrition is more in Sales representative role who t ravel frequently and there is no attrition seen in Non-travelling Sales representative.

30> We see no attrition in non-travelling HR,MD, research director .

31> Contradictory to above results we see Managers are happy who travel a lot i.e the data s ays there is no attrition in Manager job role who travel frequently.

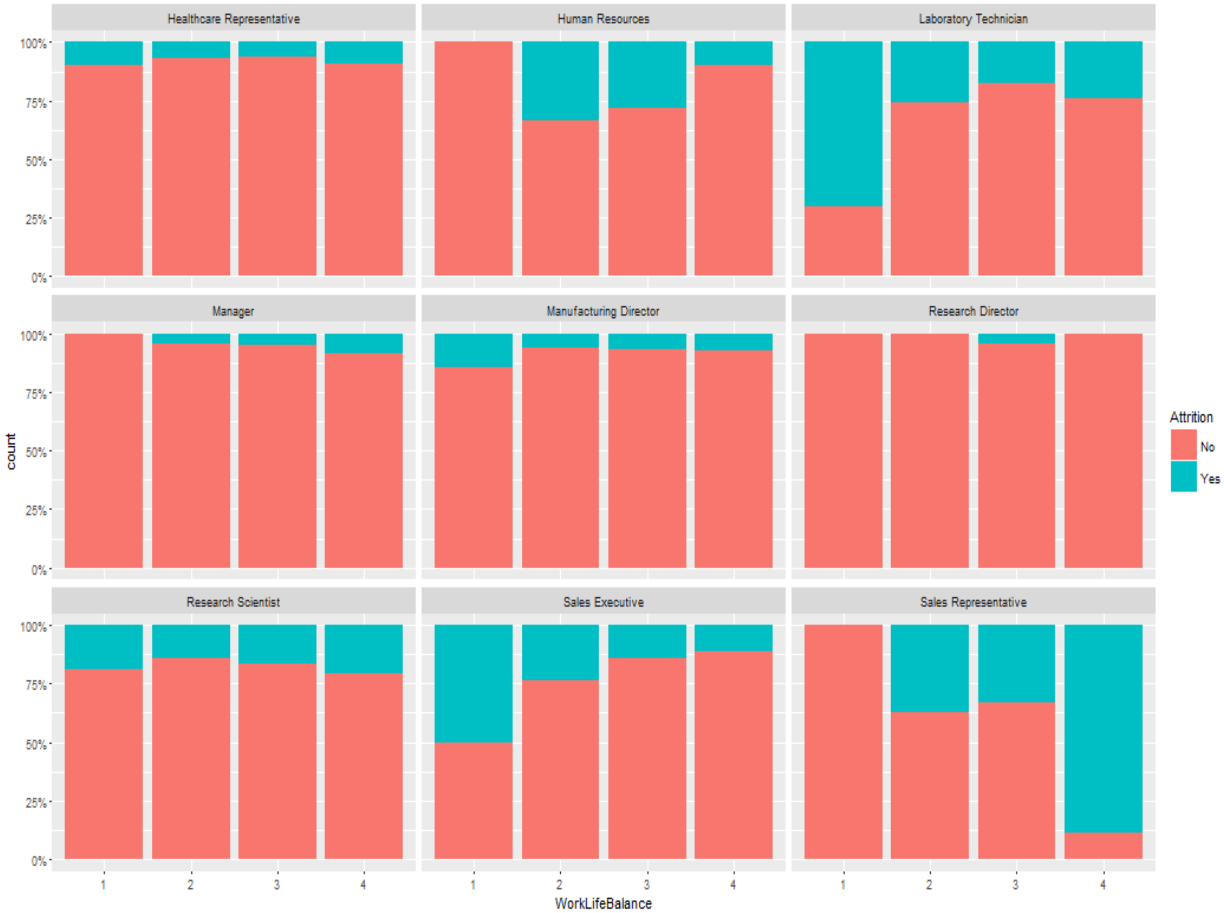
32> Among all the job roles we see there is very minimal attrition rate in Research Director t han in other job roles.

33> After non travelling sales representative the data says the second highest attrition is in H uman Resource Job role who travel frequently.

2.1.22 Attrition rate because of job roles with the overtime:

34> There is high attrition among hourly sales representatives and lab technicians

2.1.23 Attrition rate because of job roles with different work life balance:



35> Attrition rate is high With low work life balance For Lab Technician roles.But the opposite is seen For Sales representative

2.1.24 Hypothesis Tests:

36> H0 = There is no correlation between JobRole and Attrition or Attrition is independent of Job role

37> Ha = There is strong correlation between JobRole and Attrition or Attrition is dependent on Job role

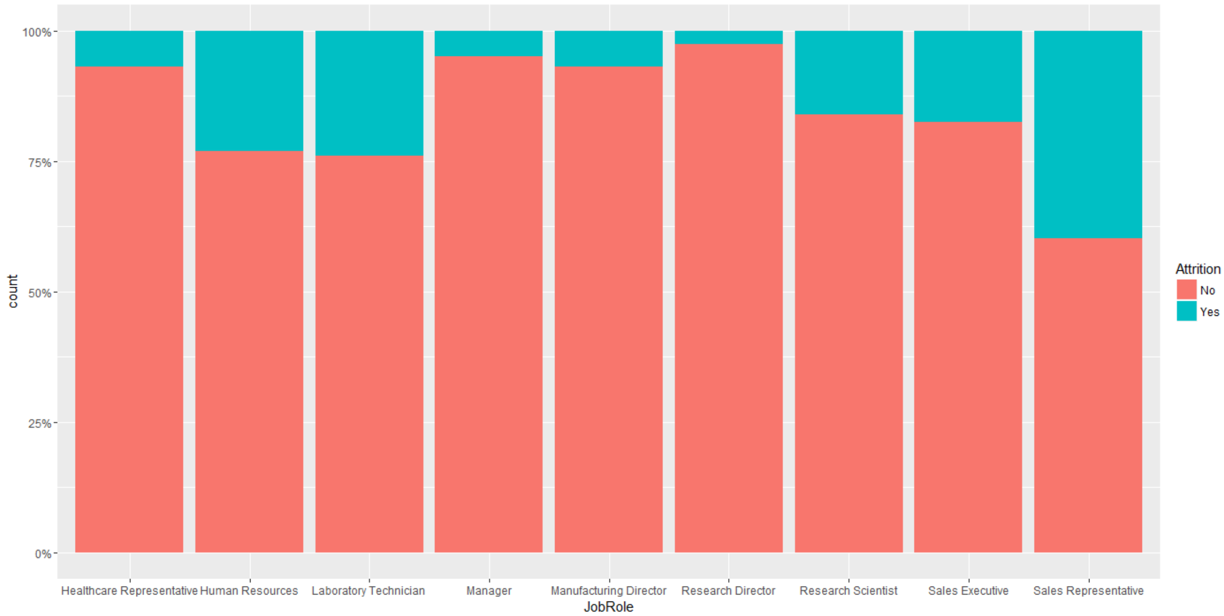
38>

39> . Pearson's Chi-squared test

40> data: tbl.jrattr

41> X-squared = 172.38, df = 8, p-value < 2.2e-16

42> Conclusion: Since the p value of chi square test of independence is less than 0.05, so we reject the null hypothesis and accept the alternate hypothesis. This means Attrition value is not independent of Job role



From the above graph we conclude that the attrition rate varies with the Job role and Sales Representative job role has significantly more attrition relative to other job roles

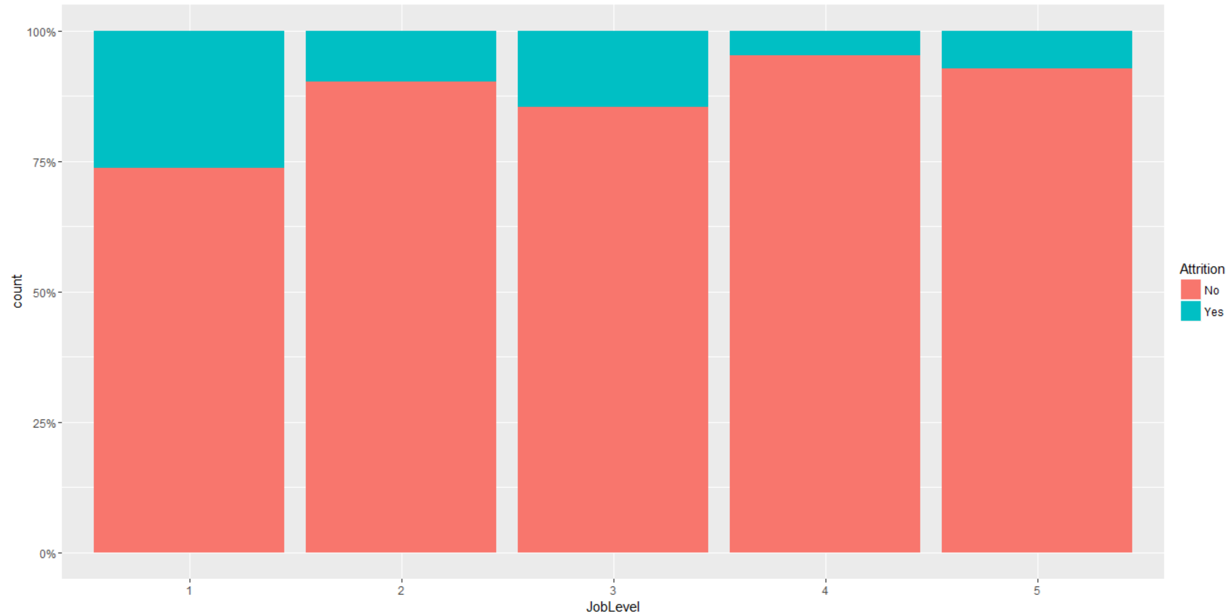
1> H0 = There is no correlation between JobRole and Attrition or Attrition is independent of Job role

Ha = There is strong correlation between JobRole and Attrition or Attrition is dependent on Job role

Pearson's Chi-squared testdata: tbl.jlattr

X-squared = 145.06, df = 4, p-value < 2.2e-16

Conclusion: Since the p value of chi square test of independence is less than 0.05, so we reject the null hypothesis and accept the alternate hypothesis. This means Attrition value is not independent of Job level and there is some correlation between the two



From the above figure it is evident that the Attrition rate differs with the Job level type and hence they are dependent of each other and it is seen that Job level 1 has high attrition then the other job level

2> Ho= Overtime doesnot affect the Attrition or Attrition is independent of Overtime or there is no strong correlation between Overtime and Attrition

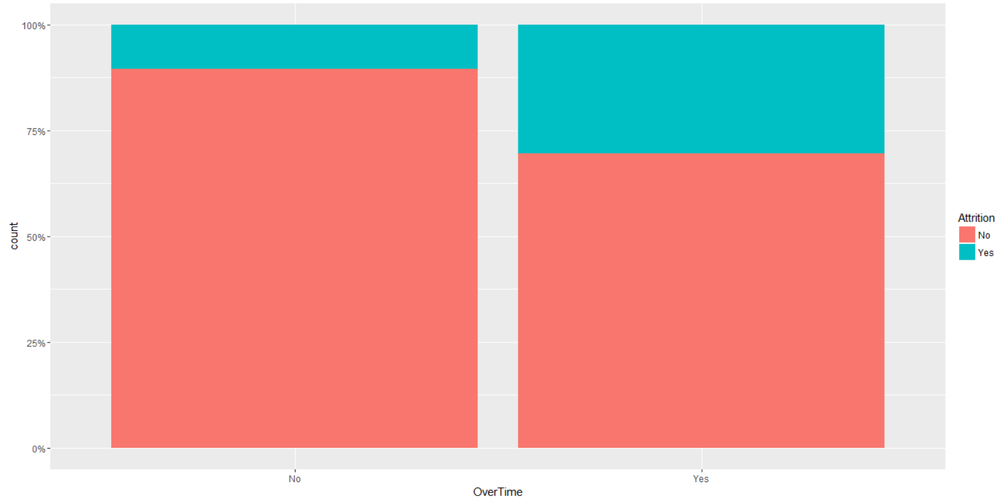
Ha= There is a strong correlation between Attrition and Overtime or Attrion and Overtime are not independent or Overtime affect the attrition rate

Pearson's Chi-squared test with Yates' continuity correction

data: tbl.otattr

X-squared = 176.61, df = 1, p-value < 2.2e-16

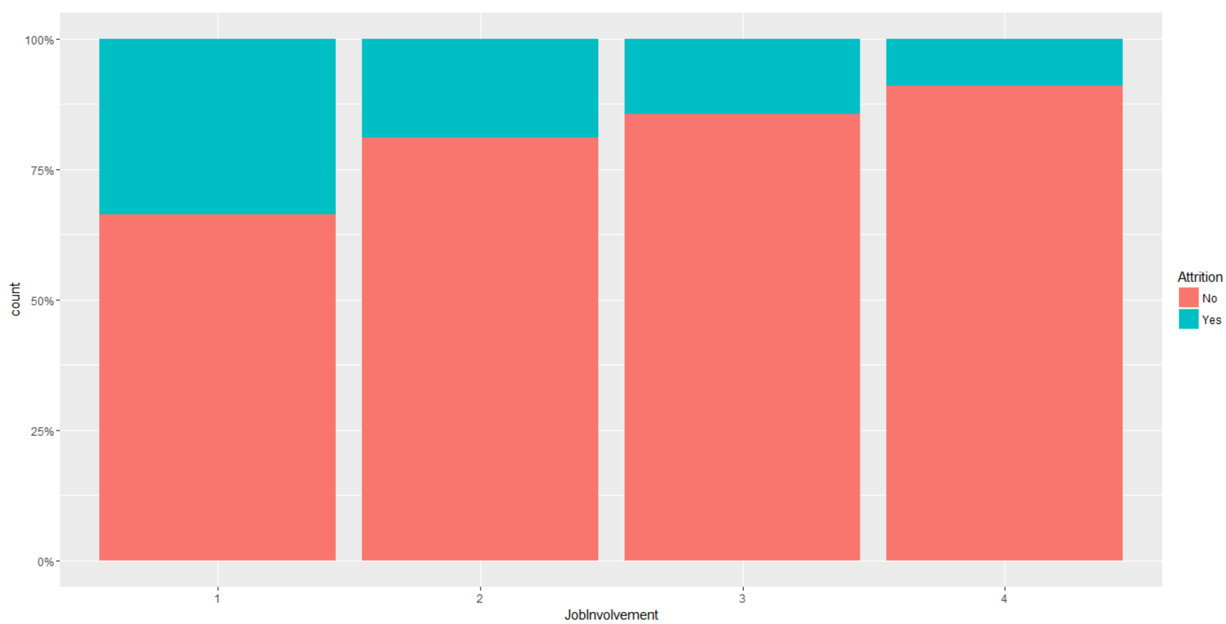
Conclusion: Since the p value of chi square test of independence is less than 0.05, so we reject the null hypothesis and accept the alternate hypothesis. This means Attrition value is dependent on the Overtime variable and there is some correlation between the two.



From the graph it is seen that overtime affects the attrition and Attrition rate is higher in those who does overtime than the employees who does not do overtime.

3> Ho= Job involvement doesnot affect the Attrition or Attrition is independent of JobInvolvement or there is no strong corelation between JobInvolvement and Attrition
Ha= There is a strong corelation between Attrition and JobInvolvement or Attrion and JobInvolvement are not independent or JobInvolvement affect the attrition rate

Conclusion: Since the p value of chi square test of independence is less than 0.05, so we reject the null hypothesis and accept the alternate hypothesis. This means Attrition value is dependent on the JobInvolvement variable and there is some correlation between the two



From the above figure it is also evident that Attrition value changes with that of the Job Involvement level and has some corelation and its seen that JobInvolvement = 1 has higher attrition than the other.

4> Ho= Business Travel doesnot affect the Attrition or Attrition is independent of Business Travel or there is no strong corelation between Business Travel and Attrition

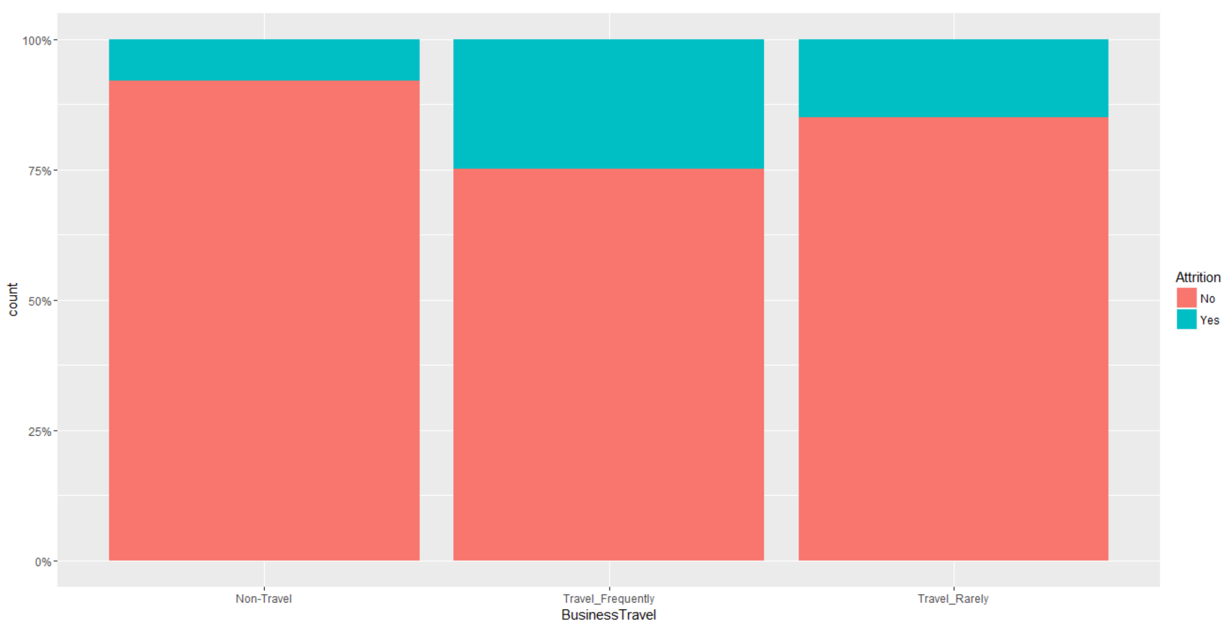
Ha= There is a strong corelation between Attrition and Business Travel or Attrion and Business Travel are not independent or Business Travel affect the attrition rate

Pearson's Chi-squared test

data: tbl.btnattr

X-squared = 48.365, df = 2, p-value = 3.146e-11

Conclusion: Since the p value of chi square test of independence is less than 0.05, so we reject the null hypothesis and accept the alternate hypothesis. This means Attrition value is dependent on the Business travel variable and there is some correlation between the two.



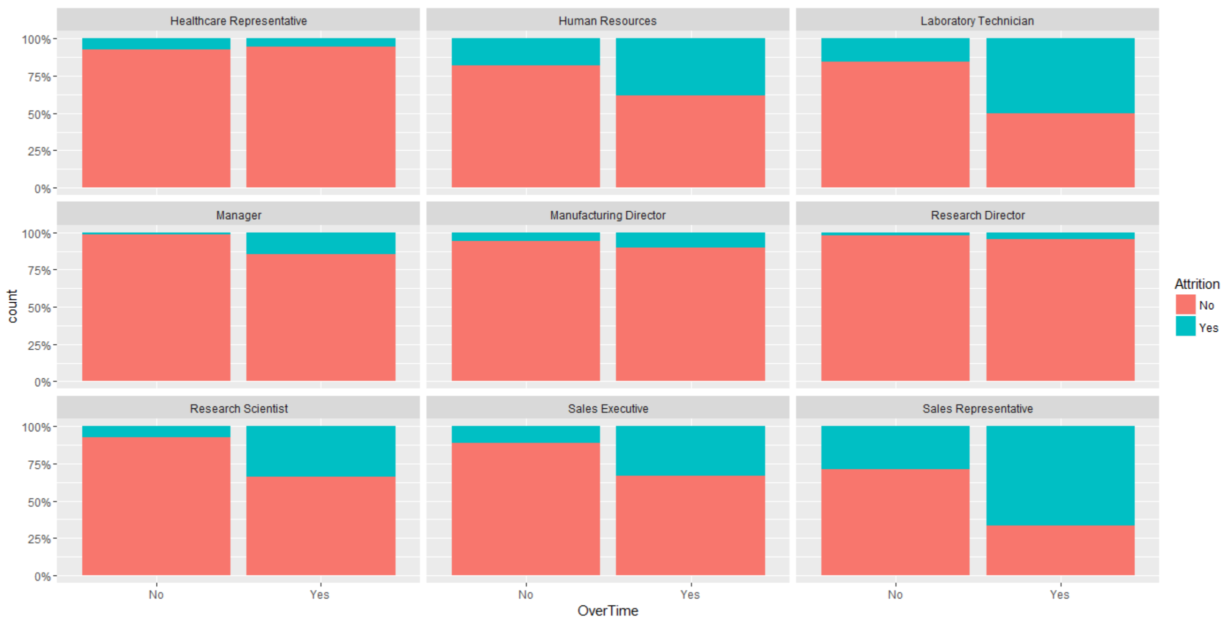
From the above graph it is clear that Business Travel; has influence on the Attrition and it is observed that the Attrition is high in frequent traveler employee.

5> Does Attrition rate depends on Jobrole and Business travel?

From the above graph it is seen that the attrition rate differs for every job role and Business travel. Attrition is more in Sales representative role who travel frequently and there is no attrition seen in Non-Travelling Sales representative. We see no attrition in non-travelling HR, MD, research director.

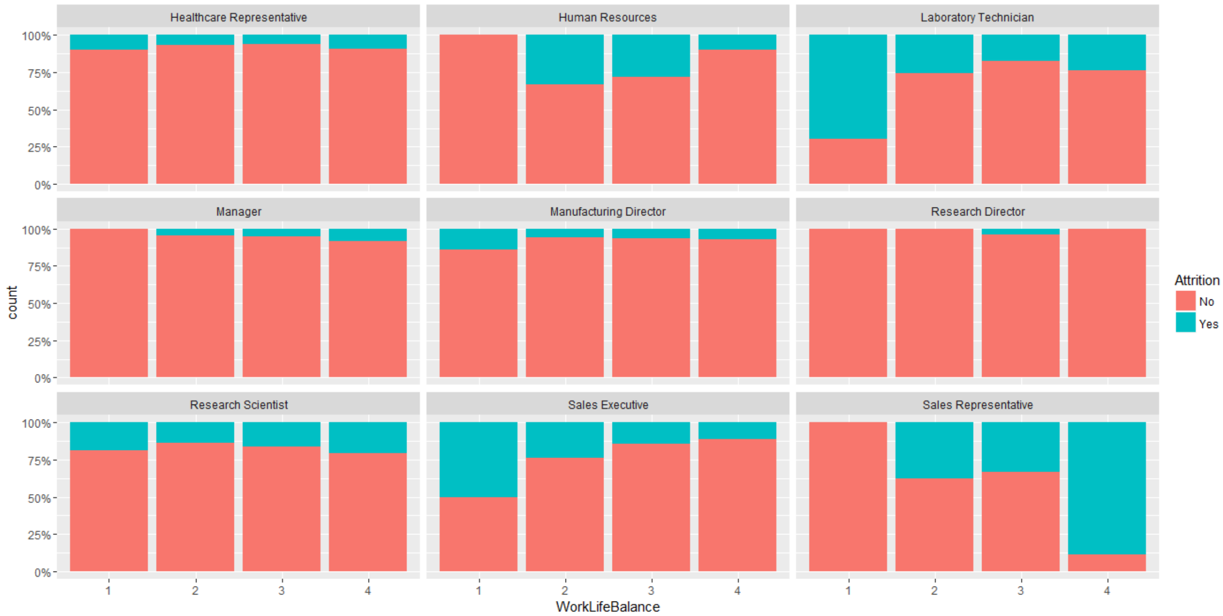
Contradictory to above results we see Managers are happy who travel a lot i.e the data says there is no attrition in Manager job role who travel frequently. Among all the job roles we see there is very minimal attrition rate in Research Director than in other job roles. After non travelling sales representative the data says the second highest attrition is in Human Resource Job role who travel frequently.

6> Does Job roles with overtime has effect on Attrition rate?



From the above graph it is seen that the Attrition rate differs for every job role and also it depends on the Overtime variable for the respective Job role. There is high attrition among hourly sales representatives and lab technicians.

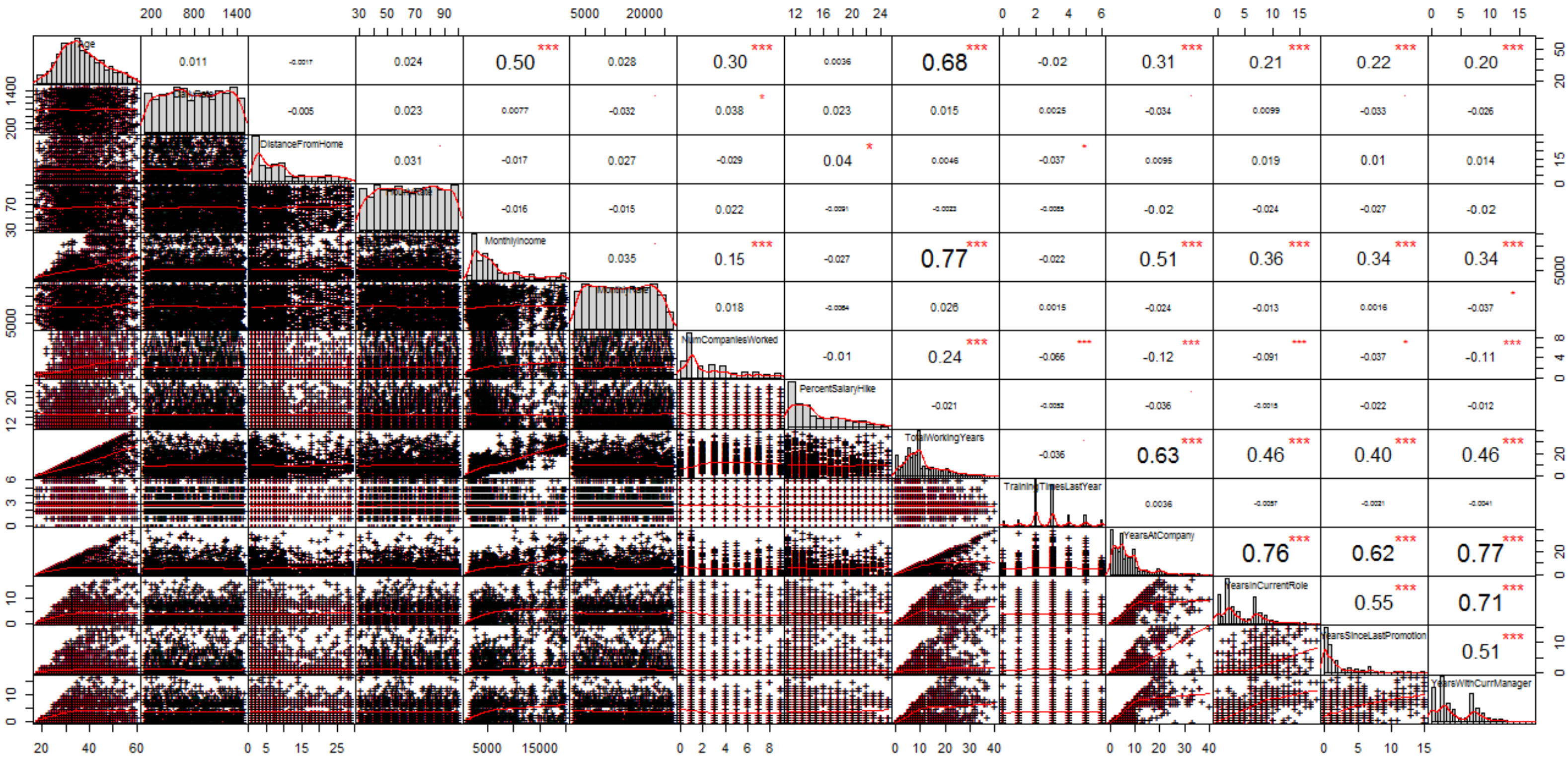
1> Does Job roles with work life balance has effect on Attrition rate?



From the above graph we see that the Attrition rate differs with the level of worklife balance for every job role. Attrition rate is high with low work life balance For Lab Technician roles But the opposite is seen For Sales representative.

7> The Strongly correlated variables are as follows

```
First.Variable        Second.Variable Correlation
117           MonthlyIncome       TotalWorkingYears    0.7728932
193          YearsAtCompany    YearsWithCurrManager    0.7692124
165          YearsAtCompany      YearsInCurrentRole    0.7587537
194      YearsInCurrentRole    YearsWithCurrManager    0.7143648
113                     Age       TotalWorkingYears    0.6803805
149       TotalWorkingYears          YearsAtCompany    0.6281332
179          YearsAtCompany YearsSinceLastPromotion    0.6184089
180      YearsInCurrentRole YearsSinceLastPromotion    0.5480562
145           MonthlyIncome          YearsAtCompany    0.5142848
195 YearsSinceLastPromotion    YearsWithCurrManager    0.5102236
```



From the graph it is seen that the most highly correlated variables with more than 60% and above corelation factor are MonthlyIncome - TotalWorkingYears with corelation value of 77.2%,YearsAtCompany-YearsWithCurrManager with corelation value of 76.9%,YearsAtCompany-YearsInCurrentRole with corelation value of 75.8%,YearsInCurrentRole-YearsWithCurrManager with corelation value of 71.4%,Age-TotalWorkingYears with corelation value of 68.03%, TotalWorkingYears- YearsAtCompany with corelation value of 62.8% and YearsAtCompany-YearsSinceLastPromotion with corelation value of 61.8%

2.1.25 Principal Component Analysis

Lets check the SD of each variable

Age Attrition BusinessTravel DailyRate

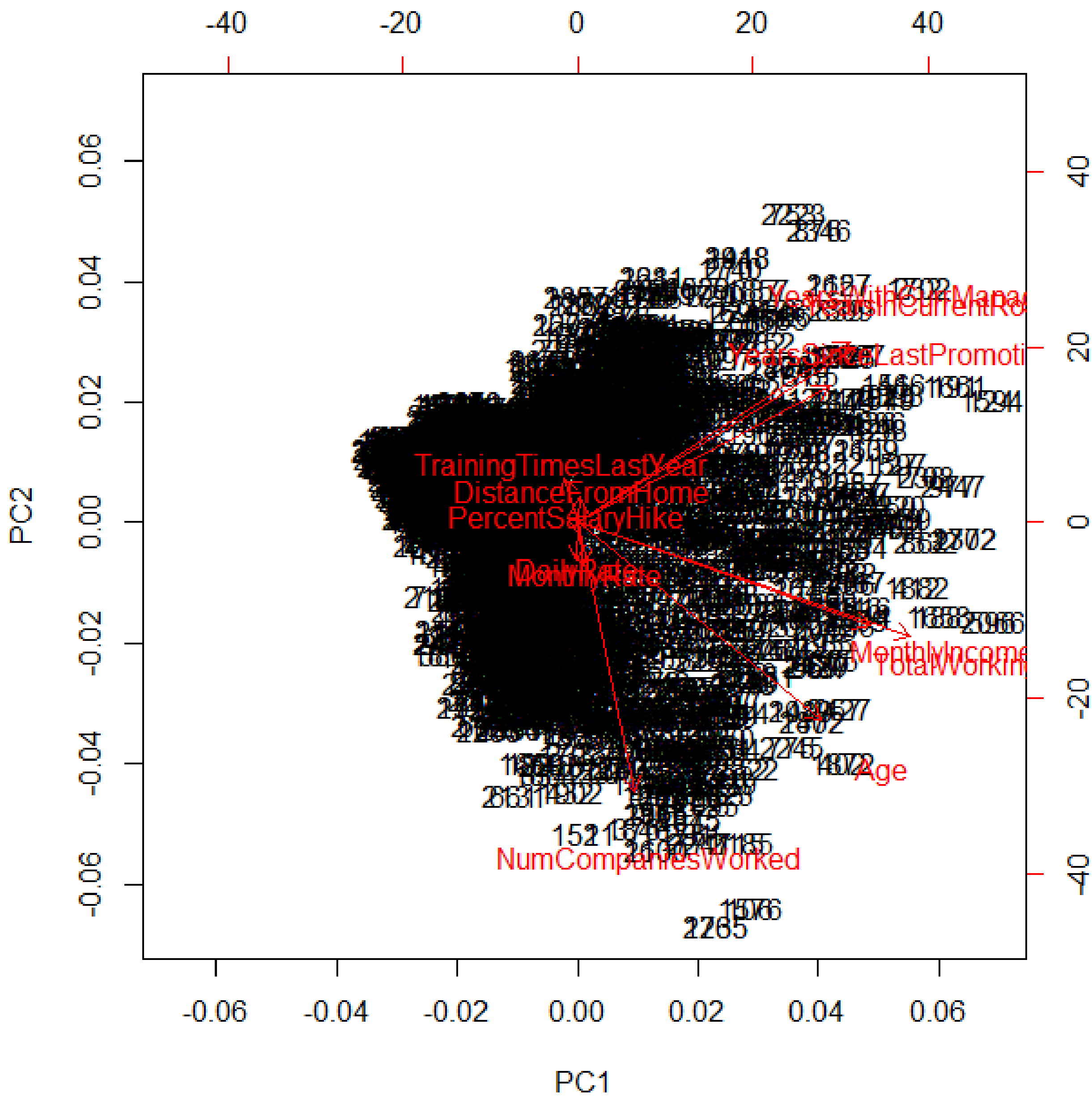
9.1338192 0.3678004 0.6653417 403.4404468

Department DistanceFromHome Education EducationField

```
# 0.5277025            8.1054851                 1.0239907            1.3311426
# EmployeeCount        EmployeeNumber  EnvironmentSatisfaction          Gender
# 0.0000000                848.8492210          1.0928962           0.4899813
# HourlyRate       JobInvolvement            JobLevel            JobRole
# 20.3259687           0.7114401          1.1067516          2.4614024
# JobSatisfaction        MaritalStatus        MonthlyIncome        MonthlyRate
# 1.1026585              0.7299965        4707.1557696        7116.5750213
# NumCompaniesWorked        Over18            OverTime       PercentSalaryHike
# 2.4975840                    0.0000000          0.4505298           3.6593150
# PerformanceRating RelationshipSatisfaction      StandardHours      StockOptionLevel
# 0.3607621            1.0810249                      0.0000000           0.8519317
# TotalWorkingYears    TrainingTimesLastYear       WorkLifeBalance     YearsAtCompany
# 7.7794579              1.2890513                    0.7063556           6.1254828
# YearsInCurrentRole  YearsSinceLastPromotion     YearsWithCurrManager
# 3.6225206              3.2218820                    3.5675290
```

We see significant difference in SD in our variables so we have to scale the data. After scaling do the principal component analysis using prcomp(). We get the following output in R

```
Importance of components:
                          PC1    PC2     PC3     PC4     PC5    PC6     PC7     PC8
Standard deviation     1.8142 1.2453 1.03035 1.02270 0.99942 0.9785 0.96476 0.84998
Proportion of Variance 0.2743 0.1292 0.08847 0.08716 0.08324 0.0798 0.07756 0.06021
Cumulative Proportion  0.2743 0.4035 0.49199 0.57915 0.66239 0.7422 0.81975 0.87996
                           PC9    PC10    PC11    PC12
Standard deviation     0.72492 0.68553 0.53132 0.40344
Proportion of Variance 0.04379 0.03916 0.02353 0.01356
Cumulative Proportion  0.92375 0.96291 0.98644 1.00000
```



The eigen values are

[1] 3.2914796 1.5508375 1.0616146 1.0459190 0.9988408 0.9575479 0.9307533 0.7224697

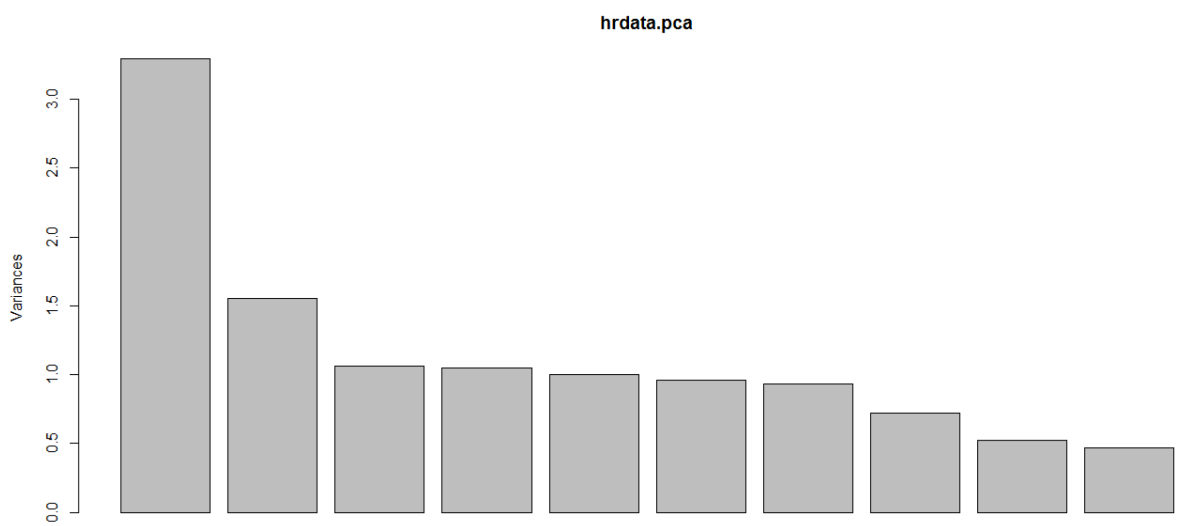
[9] 0.5255099 0.4699578 0.2823046 0.1627652

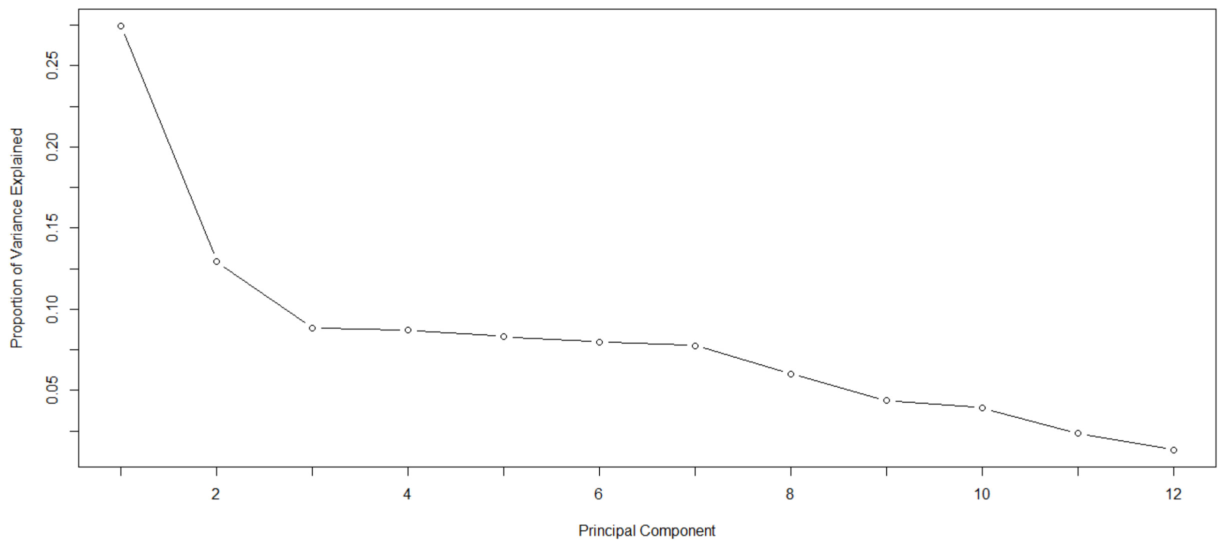
We aim to find the components which explain the maximum variance. This is because, we want to retain as much information as possible using these components. So, higher is the explained variance, higher will be the information contained in those components. To compute the proportion of variance explained by each component, we simply divide the variance by sum of total variance. This results in:

[1] 0.27428997 0.12923646 0.08846788 0.08715992 0.08323673 0.07979566 0.07756278

[8] 0.06020581 0.04379249 0.03916315 0.02352538 0.01356377

Conclusion : This shows that first principal component explains 27.42% variance. Second component explains 12.9% variance. Third component explains 8.8% variance. Eleventh component explains 2.35% of variance and so on. So, how do we decide how many components should we select for modeling stage ? The answer to this question is provided by a scree plot. A scree plot is used to access components or factors which explains the most of variability in the data. It represents values in descending order.





From the plot it is seen that with 4 components 98% of variance is explained.

Loadings:

	PC1	PC2	PC3	PC4
Age	0.3537335204	-0.418007927	0.001681452	-0.001359940
DailyRate	-0.0008826506	-0.080486552	-0.034204585	0.675682794
DistanceFromHome	0.0040633438	0.054622454	0.712349192	-0.052044453
MonthlyIncome	0.4243089257	-0.218637273	-0.032136918	-0.031911877
MonthlyRate	0.0087236589	-0.090546509	0.247761583	-0.612553730
NumCompaniesWorked	0.0836576773	-0.570555936	-0.007606365	0.049613257
PercentSalaryHike	-0.0149980573	0.007563436	0.480423799	0.393656758
TotalWorkingYears	0.4826005231	-0.240805001	-0.002966526	-0.003773748
TrainingTimesLastYear	-0.0190527826	0.095400170	-0.444618650	-0.047194685
YearsInCurrentRole	0.4047538720	0.371130843	0.019841283	0.051458477
YearsSinceLastPromotion	0.3662521658	0.288333153	-0.001661180	-0.031632560
YearsWithCurrManager	0.3953289416	0.382430537	0.003376557	0.034417369

	PC5
Age	-0.07207694
DailyRate	-0.15549880
DistanceFromHome	-0.05178562

```
MonthlyIncome           -0.04900642
MonthlyRate             -0.38657669
NumCompaniesWorked       0.08691496
PercentSalaryHike       -0.45365456
TotalWorkingYears       -0.03321397
TrainingTimesLastYear   -0.77350590
YearsInCurrentRole       0.01975632
YearsSinceLastPromotion  0.03124058
YearsWithCurrManager     0.04430146
```

Conclusion: From the Loadings it is found that Here we see component 1 seems to be influenced DailyRate,MonthlyIncome, TotalWorkingInYears, YearsInCurrentRole, YearsSinceLastPromotion , YearsWithCurrManager . Component 2 is influenced mainly by Age , YearsInCurrentRole, TrainingTimesLastYear . C3 by MonthlyIncome, PercentSalaryHike , TrainingTimesLastYear.C4 by DistanceFromHome , PercentSalaryHike ,MonthlyRate, C5 MonthlyRate, PercentSalaryHike , TrainingTimesLastYear, YearsWithCurrManager

2.1.26 Factor Analysis

```
Call:
factanal(x = hrdatascaled.clean.numericonly, factors = 3, rotation = "varimax")


Uniquenesses:
                    Age               DailyRate        DistanceFromHome
                  0.465                   0.991                   1.000
             HourlyRate           MonthlyIncome             MonthlyRate
                  0.997                   0.367                   0.996
     NumCompaniesWorked       PercentSalaryHike       TotalWorkingYears
                  0.711                   0.995                   0.026
         YearsAtCompany      YearsInCurrentRole YearsSinceLastPromotion
                  0.112                   0.291                   0.580
   YearsWithCurrManager
                  0.292
```

```
Loadings:
                         Factor1 Factor2 Factor3
Age                       0.129   0.681  -0.233
DailyRate
DistanceFromHome
HourlyRate
MonthlyIncome             0.211   0.757   0.125
MonthlyRate
NumCompaniesWorked       -0.120   0.291  -0.436
PercentSalaryHike
TotalWorkingYears         0.318   0.934
YearsAtCompany            0.785   0.410   0.324
YearsInCurrentRole        0.801   0.222   0.136
YearsSinceLastPromotion   0.576   0.243   0.172
YearsWithCurrManager      0.793   0.222   0.171

               Factor1 Factor2 Factor3
SS loadings      2.397   2.322   0.457
Proportion Var   0.184   0.179   0.035
Cumulative Var   0.184   0.363   0.398

Test of the hypothesis that 3 factors are sufficient.
The chi square statistic is 120.29 on 42 degrees of freedom.
The p-value is 1.77e-09
```

Conclusion: From the Test of the hypothesis that 3 factors are sufficient.The chi square statistic is 120.29 on 42 degrees of freedom. The p-value is 1.77e-09 . Near the bottom of the output, we can see that the significance level of the chi square fit statistic is very small. This indicates that the hypothesis of perfect model fit is rejected. Since we are in a purely exploratory vein, let’s fit a 4 factor model

```
Call:
factanal(x = hrdatascaled.clean.numericonly, factors = 4, rotation = "varimax")

Uniquenesses:
                    Age               DailyRate        DistanceFromHome
                  0.387                   0.992                   0.999
             HourlyRate           MonthlyIncome             MonthlyRate
                  0.998                   0.005                   0.996
     NumCompaniesWorked      PercentSalaryHike       TotalWorkingYears
                  0.763                   0.996                   0.114
         YearsAtCompany      YearsInCurrentRole YearsSinceLastPromotion
                  0.025                   0.063                   0.594
   YearsWithCurrManager
                  0.351

Loadings:
                        Factor1 Factor2 Factor3 Factor4
Age                      0.299   0.709   0.136
DailyRate
DistanceFromHome
HourlyRate
MonthlyIncome            0.502   0.342   0.789
MonthlyRate
NumCompaniesWorked               0.432          -0.195
PercentSalaryHike
TotalWorkingYears        0.599   0.653   0.311
YearsAtCompany           0.902                   0.392
YearsInCurrentRole       0.939                  -0.200
YearsSinceLastPromotion  0.620                   0.141
YearsWithCurrManager     0.787                   0.156

               Factor1 Factor2 Factor3 Factor4
SS loadings      3.409   1.253   0.759   0.297
Proportion Var   0.262   0.096   0.058   0.023
Cumulative Var   0.262   0.359   0.417   0.440

Test of the hypothesis that 4 factors are sufficient.
The chi square statistic is 45.88 on 32 degrees of freedom.
The p-value is 0.0533
```

Conclusion: Test of the hypothesis says that 4 factors are sufficient. The chi square statistic is 45 .88 on 32 degrees of freedom. The p-value is 0.0533 Near the bottom of the output, we can see th at the significance level of the chi square fit statistic is quite ok. This indicates that the hypothesis of perfect model fit can be accepted. Since we are in a purely exploratory vein, let's fit a 5 factor model and see whether it is better. Test of the hypothesis that 5 factors are sufficient. The chi squ are statistic is 30.45 on 23 degrees of freedom for 5 factors. The p-value is 0.137. Test of the hyp othesis that 6 factors are sufficient. The chi square statistic is 17.48 on 15 degrees of freedom. Th e p-value is 0.291. For factor 7 the factAnal() was not able to optimize. So we will go with the 6 f actor.

We can "clean up" the factor pattern in several ways. One way is to hide small loadings, to redu ce the visual clutter in the factor pattern. Another is to reduce the number of decimal places from 3 to 2. A third way is to sort the loadings to make the simple structure more obvious. After cleani ng it looks as follow

```
Call:
factanal(x = hrdatascaled.clean.numericonly, factors = 6, rotation = "varimax")

Uniquenesses:
                    Age               DailyRate        DistanceFromHome
                   0.45                    0.99                    0.82
             HourlyRate           MonthlyIncome             MonthlyRate
                   0.99                    0.35                    0.98
     NumCompaniesWorked       PercentSalaryHike       TotalWorkingYears
                   0.70                    0.99                    0.00
         YearsAtCompany      YearsInCurrentRole YearsSinceLastPromotion
                   0.07                    0.17                    0.55
   YearsWithCurrManager
                   0.31

Loadings:
                        Factor1 Factor2 Factor3 Factor4 Factor5 Factor6
YearsAtCompany           0.84    0.37                    -0.25
YearsInCurrentRole       0.88
YearsSinceLastPromotion  0.62
YearsWithCurrManager     0.78
Age                              0.63    0.35
MonthlyIncome            0.31    0.71
TotalWorkingYears        0.39    0.91
DailyRate
DistanceFromHome                                 0.39
HourlyRate
MonthlyRate
NumCompaniesWorked               0.25    0.44
PercentSalaryHike

               Factor1 Factor2 Factor3 Factor4 Factor5 Factor6
SS loadings       2.79    2.00    0.37    0.20    0.16    0.13
Proportion Var    0.21    0.15    0.03    0.02    0.01    0.01
Cumulative Var    0.21    0.37    0.40    0.41    0.42    0.43

Test of the hypothesis that 6 factors are sufficient.
The chi square statistic is 17.48 on 15 degrees of freedom.
The p-value is 0.291
```

3. Random Forest

1. Create the development and validation sample

```
hrdata.sample$split <- runif(nrow(hrdata.sample), 0, 1);

hrdata.sample <- hrdata.sample[order(hrdata.sample$split),]

#Now, if you view the hrdata.sample dataset again you would notice a new column added at the end

#Now, you can split the dataset to training and testing as follows

hrdata.train  <- hrdata.sample[which(hrdata.sample$split <= 0.7),]

hrdata.test <- hrdata.sample[which(hrdata.sample$split > 0.7),]

c(nrow(hrdata.train), nrow(hrdata.test))

#remove the split columns used for partitioning from both train and test data set

hrdata.train <- hrdata.train[,-c(33)]

hrdata.test <- hrdata.test[,-c(33)]

str(hrdata.train)

str(hrdata.test)
```

2> After creating the development and test sample ,convert all variables to factor variable by binning the numerical varaiables using R.

3> Then check if the factor variables has same level or not for each factor variable in dev and test sample. If not then make the levels of each factor variable same in both the development and test sample.

4> Find the best ntree and mtry .Caution: this algorithms takes 48 hours to populate the values for the Hr attrition csv file

```
new.modellist <- list()
for (ntree in c(500,1000, 1500, 2000, 2500)) {
  set.seed(seed)
  print(ntree)
  for(mtryt in c(1:15)){
    tunegrid=expand.grid(.mtry=mtryt)
```

```
key <-toString(paste(toString(ntree),toString(mtryt),sep = "_"))
print(key)
print(tunegrid)
fit <- train(train.new$Attrition~., data=train.new, method="rf", metric="Accuracy",
tuneGrid=tunegrid, trControl=control, ntree=ntree)
print("new fit added")
new.modellist[[key]] <- fit
}
```

5> Next compare the results to get the best mean accuracy value for ntree and mtry value.

```
results <- resamples( new.modellist)
summary(results)
```

```
Call:
summary.resamples(object = results)

Models: 500_3, 500_4, 500_5, 500_6, 500_7, 500_8, 500_9, 500_10, 500
_11, 500_12, 500_13, 500_14, 500_15, 1000_3, 1000_4, 1000_5, 1000_6,
1000_7, 1000_8, 1000_9, 1000_10, 1000_11, 1000_12, 1000_13, 1000_14,
1000_15, 1500_3, 1500_4, 1500_5, 1500_6, 1500_7, 1500_8, 1500_9, 150
0_10, 1500_11, 1500_12, 1500_13, 1500_14, 1500_15, 2000_3, 2000_4, 2
000_5, 2000_6, 2000_7
Number of resamples: 10

Accuracy
          Min. 1st Qu. Median   Mean 3rd Qu.   Max. NA's
500_3   0.8439  0.8447 0.8447 0.8450  0.8447 0.8488    0
500_4   0.8488  0.8544 0.8544 0.8562  0.8591 0.8641    0
500_5   0.8786  0.9029 0.9051 0.9057  0.9078 0.9272    0
500_6   0.9122  0.9369 0.9416 0.9383  0.9490 0.9515    0
500_7   0.9272  0.9367 0.9369 0.9383  0.9417 0.9515    0
500_8   0.9223  0.9369 0.9465 0.9441  0.9539 0.9612    0
500_9   0.9320  0.9369 0.9416 0.9461  0.9501 0.9709    0
500_10  0.9223  0.9367 0.9417 0.9417  0.9465 0.9660    0
500_11  0.9272  0.9367 0.9417 0.9446  0.9466 0.9757    0
500_12  0.9223  0.9330 0.9440 0.9436  0.9502 0.9709    0
500_13  0.9171  0.9248 0.9345 0.9349  0.9417 0.9612    0
500_14  0.9175  0.9356 0.9489 0.9436  0.9515 0.9612    0
500_15  0.9320  0.9380 0.9442 0.9446  0.9501 0.9612    0
1000_3  0.8439  0.8447 0.8447 0.8450  0.8447 0.8488    0
1000_4  0.8447  0.8495 0.8544 0.8533  0.8575 0.8592    0
1000_5  0.8786  0.8943 0.8981 0.8994  0.9064 0.9223    0
1000_6  0.9126  0.9266 0.9417 0.9373  0.9454 0.9612    0
1000_7  0.9175  0.9320 0.9366 0.9393  0.9502 0.9563    0
1000_8  0.9126  0.9367 0.9369 0.9398  0.9478 0.9610    0
1000_9  0.9272  0.9369 0.9415 0.9402  0.9454 0.9563    0
1000_10 0.9223  0.9320 0.9367 0.9412  0.9527 0.9660    0
1000_11 0.9223  0.9320 0.9440 0.9417  0.9466 0.9659    0
1000_12 0.9078  0.9283 0.9369 0.9364  0.9466 0.9612    0
1000_13 0.9223  0.9380 0.9417 0.9470  0.9635 0.9709    0
1000_14 0.9223  0.9330 0.9466 0.9465  0.9611 0.9709    0
1000_15 0.9272  0.9333 0.9417 0.9422  0.9465 0.9612    0
1500_3  0.8439  0.8447 0.8447 0.8450  0.8447 0.8488    0
```

1500_4	0.8447	0.8495	0.8495	0.8518	0.8526	0.8683	0
1500_5	0.8835	0.8956	0.9100	0.9077	0.9160	0.9320	0
1500_6	0.9175	0.9248	0.9345	0.9368	0.9502	0.9563	0
1500_7	0.9175	0.9381	0.9466	0.9417	0.9512	0.9515	0
1500_8	0.9223	0.9296	0.9392	0.9393	0.9452	0.9612	0
1500_9	0.9171	0.9295	0.9393	0.9397	0.9466	0.9612	0
1500_10	0.9272	0.9332	0.9369	0.9388	0.9453	0.9515	0
1500_11	0.9175	0.9333	0.9369	0.9407	0.9502	0.9608	0
1500_12	**0.9122**	**0.9381**	**0.9490**	**0.9451**	**0.9563**	**0.9612**	**0**
1500_13	0.9223	0.9369	0.9440	0.9432	0.9514	0.9563	0
1500_14	0.9320	0.9380	0.9442	0.9431	0.9466	0.9515	0
1500_15	0.9078	0.9367	0.9417	0.9436	0.9550	0.9709	0
2000_3	0.8439	0.8447	0.8447	0.8450	0.8447	0.8488	0
2000_4	0.8447	0.8457	0.8516	0.8537	0.8617	0.8689	0
2000_5	0.8835	0.8894	0.9003	0.9018	0.9114	0.9272	0
2000_6	0.9078	0.9233	0.9296	0.9320	0.9454	0.9512	0
2000_7	0.9223	0.9318	0.9369	0.9407	0.9537	0.9612	0

Kappa

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
500_3	0.0000	0.00000	0.00000	0.00000	0.00000	0.0000	0
500_4	0.0000	0.10120	0.10120	0.11430	0.14870	0.1944	0
500_5	0.3211	0.50340	0.51040	0.51790	0.53610	0.6569	0
500_6	0.5676	0.71170	0.73340	0.71580	0.77550	0.7880	0
500_7	0.6569	0.71160	0.71170	0.71790	0.73560	0.7880	0
500_8	0.6282	0.71170	0.76320	0.74750	0.79980	0.8352	0
500_9	0.6847	0.71170	0.73350	0.75760	0.78170	0.8798	0
500_10	0.6282	0.70410	0.73790	0.73460	0.76330	0.8578	0
500_11	0.6569	0.70410	0.73790	0.74880	0.76330	0.9012	0
500_12	0.6282	0.68290	0.75050	0.74380	0.78180	0.8798	0
500_13	0.5983	0.64230	0.69820	0.69710	0.73560	0.8352	0
500_14	0.5985	0.70430	0.77210	0.74470	0.78800	0.8352	0
500_15	0.6847	0.71820	0.75060	0.75210	0.77650	0.8352	0
1000_3	0.0000	0.00000	0.00000	0.00000	0.00000	0.0000	0
1000_4	0.0000	0.05168	0.10120	0.08610	0.10390	0.1488	0
1000_5	0.3211	0.44310	0.46950	0.47450	0.52280	0.6282	0
1000_6	0.5678	0.64720	0.73790	0.70970	0.75700	0.8352	0
1000_7	0.5985	0.68470	0.70650	0.72170	0.78180	0.8119	0
1000_8	0.5678	0.71160	0.71170	0.72410	0.76890	0.8299	0
1000_9	0.6569	0.71170	0.73330	0.72800	0.75700	0.8119	0
1000_10	0.6282	0.68470	0.70660	0.73100	0.79340	0.8578	0
1000_11	0.6282	0.68470	0.74610	0.73430	0.76330	0.8577	0
1000_12	0.5361	0.66100	0.71130	0.70370	0.76330	0.8352	0
1000_13	0.6282	0.71600	0.73790	0.76090	0.84630	0.8798	0
1000_14	0.6282	0.68290	0.76290	0.75750	0.83520	0.8798	0
1000_15	0.6569	0.69150	0.73790	0.73760	0.76330	0.8352	0
1500_3	0.0000	0.00000	0.00000	0.00000	0.00000	0.0000	0
1500_4	0.0000	0.05168	0.05168	0.07030	0.05309	0.2381	0
1500_5	0.3602	0.45160	0.55190	0.53000	0.57920	0.6847	0
1500_6	0.5985	0.64230	0.69820	0.70730	0.77980	0.8119	0
1500_7	0.5985	0.71830	0.76330	0.73480	0.78610	0.7880	0
1500_8	0.6282	0.67060	0.72470	0.72150	0.75090	0.8352	0
1500_9	0.5983	0.66810	0.73230	0.72520	0.76330	0.8352	0
1500_10	0.6569	0.69150	0.71560	0.72190	0.75690	0.7880	0
1500_11	0.5985	0.69150	0.71170	0.72960	0.78180	0.8298	0
1500_12	0.5676	0.71830	0.77560	0.75140	0.81190	0.8352	0
1500_13	0.6282	0.71170	0.75050	0.74330	0.78620	0.8119	0
1500_14	0.6847	0.71600	0.75060	0.74410	0.76330	0.7880	0
1500_15	0.5361	0.71160	0.73790	0.74280	0.80420	0.8798	0

```
2000_3   0.0000 0.00000 0.00000 0.00000 0.00000 0.0000     0
2000_4   0.0000 0.01291 0.05262 0.08852 0.17110 0.2383     0
2000_5   0.3602 0.40090 0.48510 0.48960 0.55990 0.6569     0
2000_6   0.5361 0.63520 0.67080 0.68070 0.75700 0.7809     0
2000_7   0.6506 0.67620 0.71170 0.73110 0.79970 0.8352     0
```

From the result we find that the mean accuracy of 94.51% ,which is good when ntree is 1500 and with mtry of 12

6> Build the model using random forest

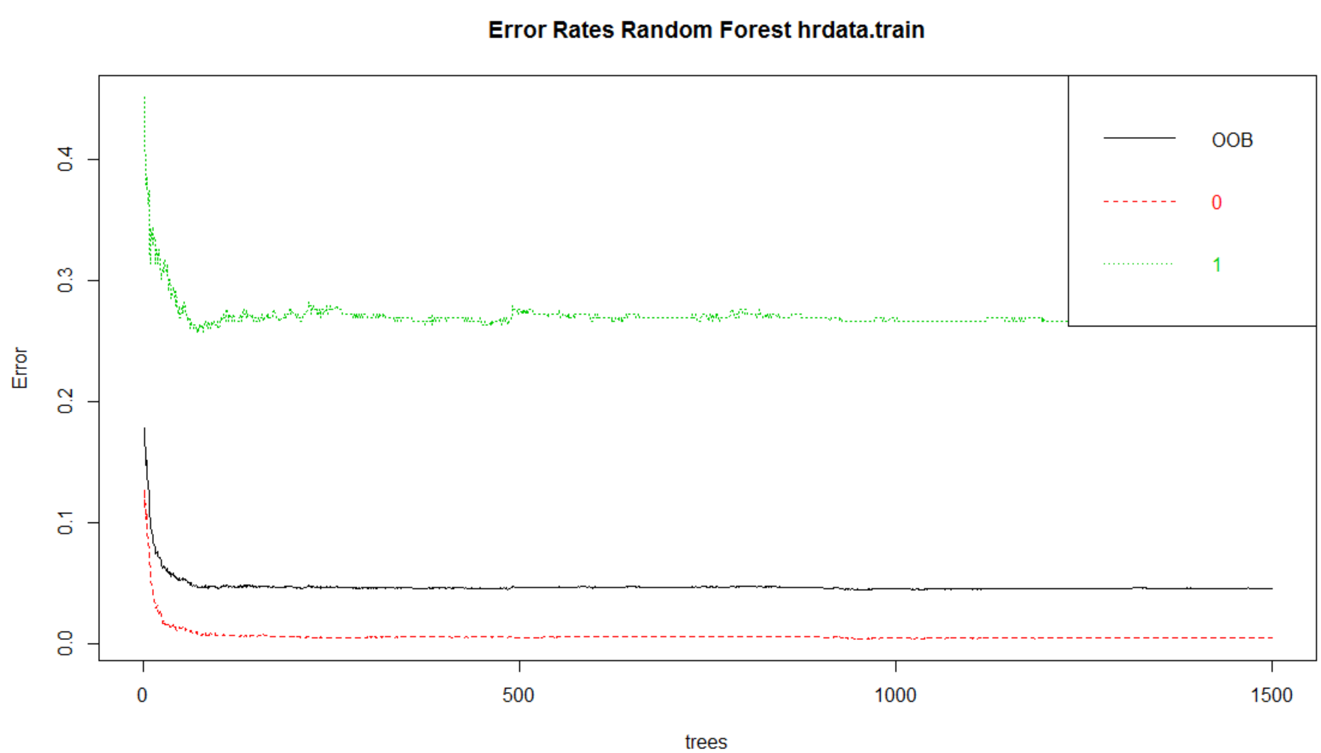
arf <- randomForest(Attrition ~ .,data = train.new,importance = TRUE,proximity = TRUE,ntree =1500, mtry=12, keep.forest = TRUE)

print(arf)

```
Call:
 randomForest(formula = Attrition ~ ., data = train.new, importance
= TRUE,      proximity = TRUE, ntree = 1500, mtry = 12, keep.forest
= TRUE)
               Type of random forest: classification
                     Number of trees: 1500
No. of variables tried at each split: 12

        OOB estimate of  error rate: 4.57%
Confusion matrix:
      No Yes class.error
No  1730   9 0.005175388
Yes   85 234 0.266457680
```

7> Plot the model



From the summary and the plot it is found that the OOB is 4.57% which is very good

8> Validating the model using deciling and rank order

```
decile <- function(x){
  deciles <- vector(length=10)
  for (i in seq(0.1,1,.1)){
    deciles[i*10] <- quantile(x, i, na.rm=T)
  }
  return (
    ifelse(x<deciles[1], 1,
           ifelse(x<deciles[2], 2,
                  ifelse(x<deciles[3], 3,
                         ifelse(x<deciles[4], 4,
                                ifelse(x<deciles[5], 5,
                                       ifelse(x<deciles[6], 6,
                                              ifelse(x<deciles[7], 7,
                                                     ifelse(x<deciles[8], 8,
                                                            ifelse(x<deciles[9], 9, 10
                                                            ))))))))))
}

train.new$deciles <- decile(train.new$predict.score[,2])
library(plyr)
x = (count(train.new$Attrition == "No"))$freq + (count(train.new$Attrition == "No"))$freq
x[1]
## Ranking code
train.new$Attrition = factor(
  train.new$Attrition,
  levels = c("Yes", "No"),
  labels = c(1, 0)
)
train.new$Attrition <-
  as.numeric(as.character(train.new$Attrition))
library(data.table)
```

```
tmp_DT = data.table(train.new)
rank <- tmp_DT[, list(
  cnt = length(Attrition),
  cnt_resp = sum(Attrition),
  cnt_non_resp = sum(Attrition == 1)),
  by=deciles][order(-deciles)]
rank$rrate <- round(rank$cnt_resp * 100 / rank$cnt,2);
rank$cum_resp <- cumsum(rank$cnt_resp)
rank$cum_non_resp <- cumsum(rank$cnt_non_resp)
rank$cum_rel_resp <- round(rank$cum_resp / sum(rank$cnt_resp),2);
rank$cum_rel_non_resp <- round(rank$cum_non_resp / sum(rank$cnt_non_resp),2);
rank$ks <- abs(rank$cum_rel_resp - rank$cum_rel_non_resp);
View(rank)
```

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	207	207	207	100.0	207	207	0.65	0.65	0
2	9	204	112	112	54.9	319	319	1.00	1.00	0
3	8	208	0	0	0.0	319	319	1.00	1.00	0
4	7	221	0	0	0.0	319	319	1.00	1.00	0
5	6	194	0	0	0.0	319	319	1.00	1.00	0
6	5	211	0	0	0.0	319	319	1.00	1.00	0
7	4	210	0	0	0.0	319	319	1.00	1.00	0
8	3	203	0	0	0.0	319	319	1.00	1.00	0
9	2	210	0	0	0.0	319	319	1.00	1.00	0
10	1	190	0	0	0.0	319	319	1.00	1.00	0

From the rank order we find that the first decile rrrate is 100% and has 65% of attrition value.Simillarly second decile's rrate is 54.9% and has 100% of attrition value.

3.1 Performance of random forest

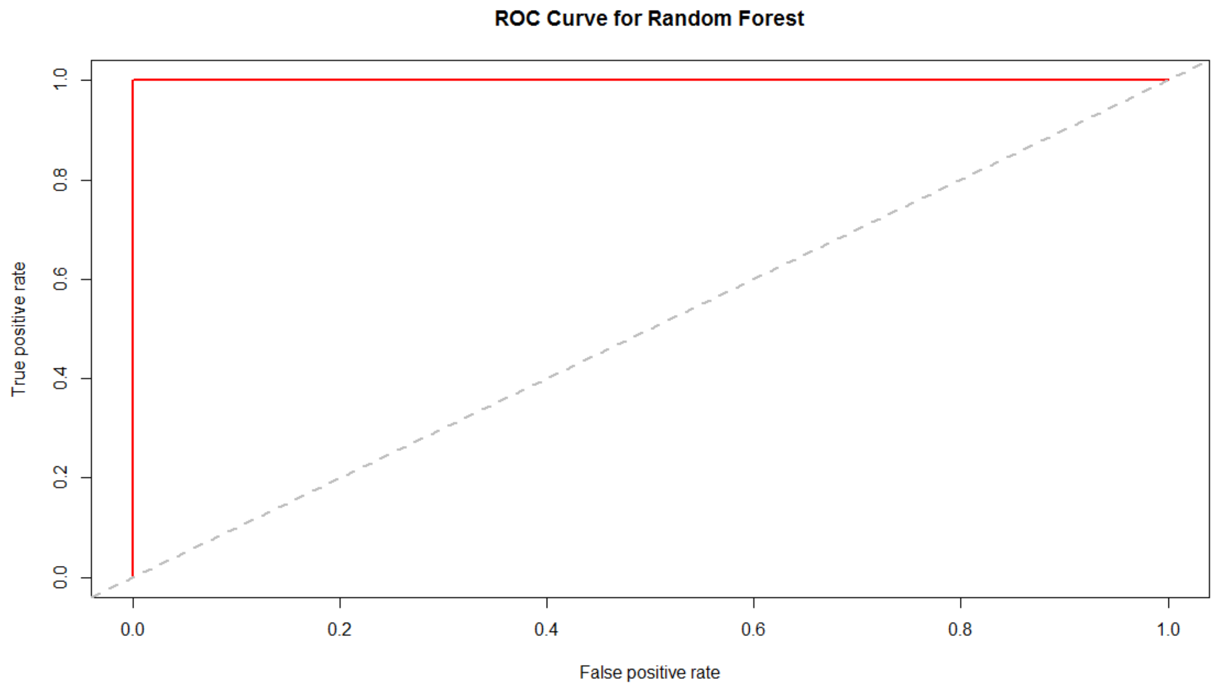
```
library(ROCR)
testp4 <- predict(arf,train.new,type = 'prob')[,2]
pred4 <- prediction(testp4,train.new$Attrition)
```

3.1.1 Performance in terms of true and false positive rates

```
perf4 <- performance(pred4,"tpr","fpr")
```

Plot the curve

```
plot(perf4,main="ROC Curve for Random Forest",col=2,lwd=2)
abline(a=0,b=1,lwd=2,lty=2,col="gray")
```



3.2 Compute area under curve

```
KS <- max(attr(perf4, 'y.values')[[1]]-attr(perf4, 'x.values')[[1]])
auc <- performance(pred4,"auc");
auc <- as.numeric(auc@y.values)
minauc<-min(round(auc, digits = 2))
maxauc<-max(round(auc, digits = 2))
minauct <- paste(c("min(AUC) = "),minauc,sep="")
maxauct <- paste(c("max(AUC) = "),maxauc,sep="")
```

KS value is 1,auc value is 1,minauc,maxauc all values are 1

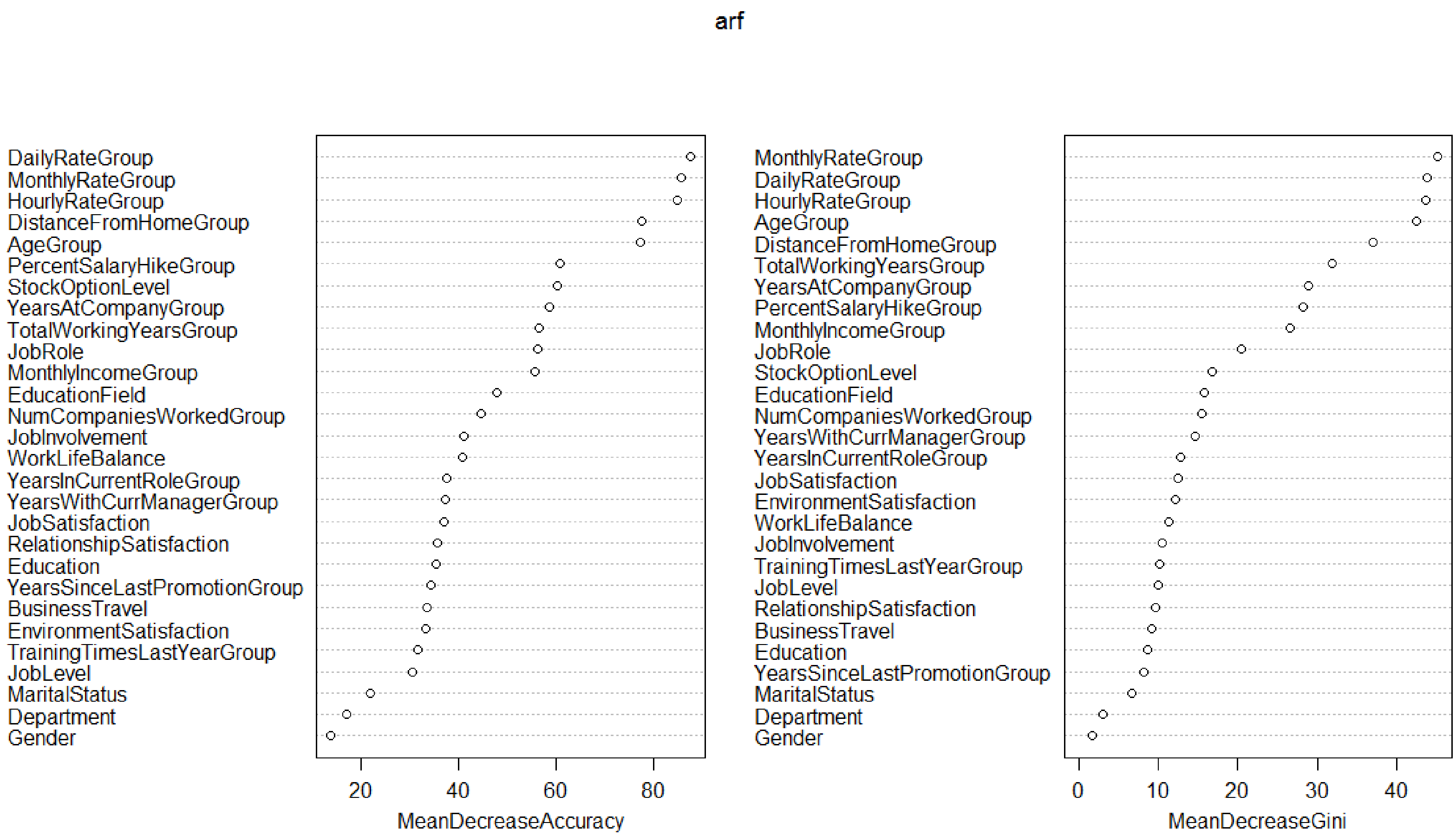
9>

3.3 Important variable selection

10> Since the model is validated and the performance was found to be good so we can find out the important variables

print(importance(arf,type = 2))

varImpPlot(arf)



Important variables that contribute to the Attrition as per the data set and random forest are

Number of companies worked, EducationField, MonthlyIncome, JobRole, TotalWorkingYears, YearsAtCompany, StockOptionLevel, PercentSalaryHike,Age, DistanceFromHome, HourlyRate, MonthlyRate, DailyRate.

Least important variables are Marital Status, Department and Gender

4. Neural Network

4.1 Using nnet package

1> Read the .csv file and create the hrdata data frame

```
setwd("E:\\PGPBA\\PGPBA-
GreatLakes\\Modules\\DataMining\\Assignment\\Assignment2")
hrdata = read.csv("HR_Employee_Attrition_Data.csv",
           header = TRUE,
           sep = ",")
```

2> We see that there are a total of 2940 observations with 35 variables. There is no NA values in the data.Some variables needs to be converted to factor variables. Lets convert some variables into factor variable.

```
 hrdata$Education <- as.factor(hrdata$Education)
hrdata$EnvironmentSatisfaction <-
  as.factor(hrdata$EnvironmentSatisfaction)
hrdata$JobInvolvement <- as.factor(hrdata$JobInvolvement)
hrdata$JobSatisfaction <- as.factor(hrdata$JobSatisfaction)
hrdata$PerformanceRating <- as.factor(hrdata$PerformanceRating)
hrdata$RelationshipSatisfaction <-
  as.factor(hrdata$RelationshipSatisfaction)
hrdata$WorkLifeBalance <- as.factor(hrdata$WorkLifeBalance)
hrdata$JobLevel <- as.factor(hrdata$JobLevel)
hrdata$StockOptionLevel <- as.factor(hrdata$StockOptionLevel)
```

3> Creating Development and Validation Sample

```
hrdata.sample$split <- runif(nrow(hrdata.sample), 0, 1)
hrdata.sample <- hrdata.sample[order(hrdata.sample$split), ]
```

4> Now, if you view the hrdata.sample dataset again you would notice a new column added at the end.Now, you can split the dataset to training and testing as follows

```
hrdata.train  <- hrdata.sample[which(hrdata.sample$split <= 0.7), ]
hrdata.test <- hrdata.sample[which(hrdata.sample$split > 0.7), ]
c(nrow(hrdata.train), nrow(hrdata.test))
```

5> Remove the split columns used for partitioning from both train and test data set

```
hrdata.train <- hrdata.train[, -c(32)]

hrdata.test <- hrdata.test[, -c(32)]
```

6> Convert the numerical values to factor by binning them as follows

```
ApplyQuantile <- function(x) {
  cut(x, breaks = c(quantile(hrdata.train$Age, probs = seq(0, 1, by = 0.10))), include.lowest =
TRUE)
}
str(hrdata.train)
hrdata.new.train = hrdata.train
hrdata.train$AgeGroup = sapply(hrdata.train$Age, ApplyQuantile)
str(hrdata.train)
```

7> The above code is of Age variable,similarly do it for other numerical variables.

8> Fit a Single Hidden Layer Neural Network using Least Squares using nnet package

```
train.nnet <-
  nnet(
    Attrition ~ .,
    train.new,
    size = 3,
    rang = 0.07,
    Hess = FALSE,
    decay = 15e-4,
    maxit = 250
  )
```

9> Use TEST data for testing the trained model

```
test.nnet <- predict(train.nnet, test.new, type = ("class"))
```

10> MisClassification Confusion Matrix

```
table(test.new$Attrition, test.nnet)
```

```
 test.nnet
      No Yes
No   573 154
Yes   54 101
```

Misclassification accuracy = 208/882 = 0.2358 = 23.58% and Classification Accuracy is 76.42%

11> One can maximize the Accuracy by changing the "size" while training the neural network. SIZE refers to the number of nodes in the hidden layer.

12> We find 149 as the row number which break ties at random (Maximum position in vector) using following command

```
which.is.max(test.nnet)
```

4.2 Using Multinomial log linear models through multinom method

13> Use Multinomial Log Linear models using Neural Networks

```
train.mlln <- multinom(Attrition ~ ., train.new)
```

14> USe TEST data for testing the trained model

```
test.mlln <- predict(train.mlln, test.new)
```

15> Misclassification or Confusion Matrix

```
table(test.new$Attrition, test.mlln)
```

```
   test.mlln
      No Yes
No   232 495
Yes    5 150
```

16> We find that the misclassification error is 500/882 = 56.68 and Classification accuracy is 43.32%

4.3 Using neuralnet package

17> Check for all Input Independent Variables to be Integer or Numeric or complex matrix or vector arguments. If they are not any one of these, then tranform them accordingly

```
str(hrdata.train)
str(hrdata.test)
```

18> It can be observed that all are either integer or factor. Now these factors have to be transformed to numeric. One cannot use directly as.numeric() to convert factors to numeric as it has limitations. First, Lets convert factors having character levels to numeric levels.

```
hrdata.transform$Attrition = factor(
  hrdata.transform$Attrition,
  levels = c("Yes", "No"),
  labels = c(1, 0)
)
hrdata.transform$BusinessTravel = factor(
  hrdata.transform$BusinessTravel,
  levels = levels(hrdata.transform$BusinessTravel),
  labels = c(1, 2, 3)
)
```

19> The above is just an example,similarly do for other variables. See Appendix section for complete code.

20> Now convert these numerical factors into numeric

```
hrdata.transform$Attrition <-
  as.numeric(as.character(hrdata.transform$Attrition))
hrdata.transform$BusinessTravel <-
  as.numeric(as.character(hrdata.transform$BusinessTravel))
```

21> The above is just an example,similarly do for other variables. See Appendix section for complete code.

22> Now all the variables are wither intergers or numeric. Now we shall partition the data into train and test data

```
library(caret)
set.seed(1234567)
train2 <-
  createDataPartition(hrdata.transform$Attrition, p = 0.7, list = FALSE)
trainnew <- hrdata.transform[train2, ]
testnew <- hrdata.transform[-train2, ]
```

23> In neural network it is adviced to scale the data if the data is not properly distributed or is not normal and we know from the EDA and PCA that the variables has high standard deviation and also it is not normal. So scale the variables except the Attrition variable.

```
x <- hrdata.transform[,-2]
nn.devscaled <- scale(x)

nn.devscaled <- cbind(hrdata.transform[2], nn.devscaled)
```

24> Now create the scaled training and test sample set

```
set.seed(891023)
train3 <-  createDataPartition(nn.devscaled$Attrition, p = 0.7, list = FALSE)
scaledTraindata <- nn.devscaled[train3, ]
scaledTestdata <- nn.devscaled[-train3, ]
```

25> Now lets run the neuralnet model on Train dataset

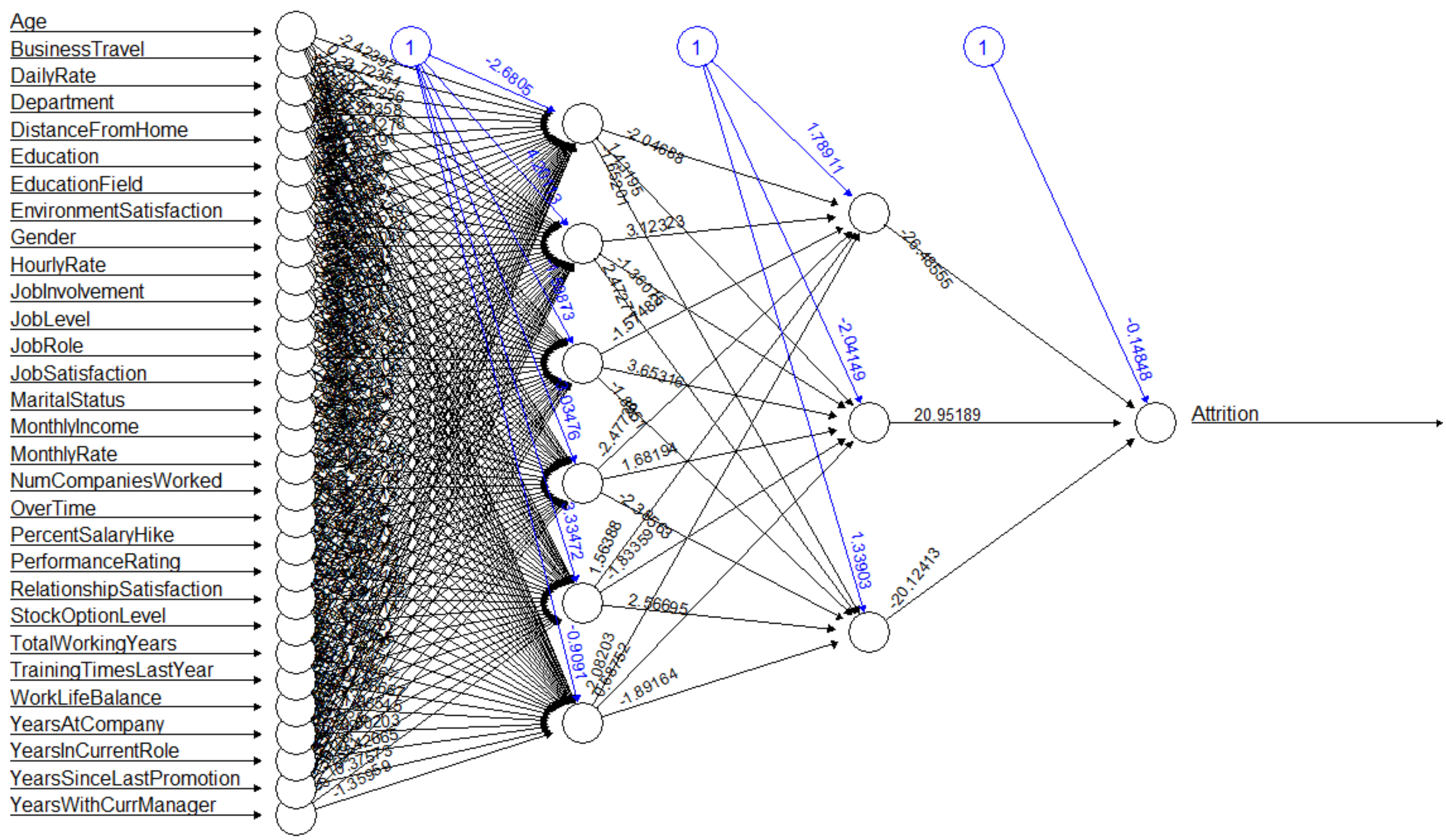
```
trainnew.nnbp <-
 neuralnet(
  Attrition ~ Age + BusinessTravel + DailyRate + Department + DistanceFromHome +
   Education + EducationField + EnvironmentSatisfaction + Gender + HourlyRate +
   JobInvolvement + JobLevel + JobRole + JobSatisfaction + MaritalStatus +
   MonthlyIncome + MonthlyRate + NumCompaniesWorked + OverTime + PercentSalaryHike +
   PerformanceRating + RelationshipSatisfaction + StockOptionLevel + TotalWorkingYears +
   TrainingTimesLastYear + WorkLifeBalance + YearsAtCompany + YearsInCurrentRole +
   YearsSinceLastPromotion + YearsWithCurrManager,
  data = scaledTraindata,
  hidden = c(6,3),
  threshold = 0.01,
  err.fct = "sse",
  linear.output = FALSE,
  lifesign = "full",
  lifesign.step = 10,
  stepmax = 1e6
 )
```

26> The distribution of the estimated results

```
   0%                                  1%
0.00000000000000000005941683911 0.00000000000000000006164678296
                             5%                             10%
0.00000000000000000009932743439 0.00000000000000000015584611023
                            25%                            50%
0.00000000000000000630070656892 0.00000000000004003685180272121
                            75%                             90%
0.00000910327779685689418218141 9 0.99530290693586420669447534237 4
                            95%                             99%
0.99999657495208948976284091259 0 0.99999999613376089335758933884 8
                           100%
0.99999999887026902456454990897 3
```

27> We find the probability distribution is good.

28> Smoother the Curve- Better is the model prediction. Plot the trained Neural Network



29> Check your prediction accuracy of training model. Select the columns from the data set that were used to build the model.

```
columnc = c(
  "Age",
  "BusinessTravel",
  "DailyRate",
  "Department",
  "DistanceFromHome",
  "Education",
  "EducationField",
  "EnvironmentSatisfaction",
  "Gender",
  "HourlyRate",
  "JobInvolvement",
  "JobLevel",
  "JobRole" ,
  "JobSatisfaction",
  "MaritalStatus",
  "MonthlyIncome" ,
  "MonthlyRate",
  "NumCompaniesWorked",
  "OverTime" ,
  "PercentSalaryHike",
  "PerformanceRating",
  "RelationshipSatisfaction",
  "StockOptionLevel",
  "TotalWorkingYears",
  "TrainingTimesLastYear",
  "WorkLifeBalance" ,
  "YearsAtCompany" ,
  "YearsInCurrentRole" ,
  "YearsSinceLastPromotion",
  "YearsWithCurrManager"
)
testnew2 <- subset(scaledTestdata, select = columnc)
```

```
testnew.nnbp <- compute(trainnew.nnbp, testnew2, rep = 1)
```

30> Create the misclassification confusion matrix

```
testnew2 <- subset(scaledTestdata, select = columnc)
testnew.nnbp <- compute(trainnew.nnbp, testnew2, rep = 1)
## MisClassification Confusion Matrix
table(testnew$Attrition, testnew.nnbp$net.result)
cbind(testnew$Attrition, testnew.nnbp$net.result)
print(testnew.nnbp)

## Error Computation
misClassTable = data.frame(Attrition = scaledTraindata$Attrition,
                           Predict.score = trainnew.nnbp$net.result[[1]])
misClassTable$Predict.class = ifelse(misClassTable$Predict.score > 0.21, 1, 0)
with(misClassTable, table(Attrition, Predict.class))
```

```
         Predict.class
 Attrition    0    1
         0 1702   14
         1   68  274
```

Misclassification is 82

misclassification error = 82/2058=0.0398 = 3.98%

Classification Accuracy=96.02%

```
library(e1071)
confusionMatrix(misClassTable$Attrition, misClassTable$Predict.class)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1702   14
         1   68  274

               Accuracy : 0.9601555
                 95% CI : (0.9507817, 0.9681874)
    No Information Rate : 0.8600583
    P-Value [Acc > NIR] : < 0.00000000000000022204

                  Kappa : 0.8465224
 Mcnemar's Test P-Value : 0.000000004831592

            Sensitivity : 0.9615819
            Specificity : 0.9513889
         Pos Pred Value : 0.9918415
         Neg Pred Value : 0.8011696
             Prevalence : 0.8600583
```

```
         Detection Rate : 0.8270165
   Detection Prevalence : 0.8338192
      Balanced Accuracy : 0.9564854

       'Positive' Class : 0
```

```
sum((misClassTable$Attrition - misClassTable$Predict.score) ^ 2) / 2
```

31> The error was calculated to be of value 41.00971486

32> Lets do the rank order with deciling

```
decile <- function(x){
 deciles <- vector(length=10)
 for (i in seq(0.1,1,.1)){
  deciles[i*10] <- quantile(x, i, na.rm=T)
 }
 return (
  ifelse(x<deciles[1], 1,
      ifelse(x<deciles[2], 2,
          ifelse(x<deciles[3], 3,
              ifelse(x<deciles[4], 4,
                  ifelse(x<deciles[5], 5,
                      ifelse(x<deciles[6], 6,
                          ifelse(x<deciles[7], 7,
                              ifelse(x<deciles[8], 8,
                                  ifelse(x<deciles[9], 9, 10
                                  ))))))))))
}

## deciling
misClassTable$deciles <- decile(misClassTable$Predict.score)

## Ranking code
library(data.table)
tmp_DT = data.table(misClassTable)
rank <- tmp_DT[, list(
 cnt = length(Attrition),
 cnt_resp = sum(Attrition),
 cnt_non_resp = sum(Attrition == 1)) ,
```

```
 by=deciles][order(-deciles)]
rank$rrate <- round (rank$cnt_resp / rank$cnt,2);
rank$cum_resp <- cumsum(rank$cnt_resp)
rank$cum_non_resp <- cumsum(rank$cnt_non_resp)
rank$cum_rel_resp <- round(rank$cum_resp / sum(rank$cnt_resp),2);
rank$cum_rel_non_resp <- round(rank$cum_non_resp / sum(rank$cnt_non_resp),2);
rank$ks <- abs(rank$cum_rel_resp - rank$cum_rel_non_resp)
library(scales)
rank$rrate <- percent(rank$rrate)
rank$cum_rel_resp <- percent(rank$cum_rel_resp)
rank$cum_rel_non_resp <- percent(rank$cum_rel_non_resp)
View(rank)
```

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_
1	10	207	193	193	93%	193	193	56%	56%
2	9	205	81	81	40%	274	274	80%	80%
3	8	206	4	4	2%	278	278	81%	81%
4	7	205	6	6	3%	284	284	83%	83%
5	6	206	6	6	3%	290	290	85%	85%
6	5	206	10	10	5%	300	300	88%	88%
7	4	205	15	15	7%	315	315	92%	92%
8	3	206	9	9	4%	324	324	95%	95%
9	2	207	11	11	5%	335	335	98%	98%
10	1	205	7	7	3%	342	342	100%	100%

33>

First decile has got 93% and is capturing 56% of total attrition value

34> Similalry second decile has got 40% and is capturing 80% of total attrition value

35> Calculate the MSE value

```
pr.nn1 <- compute(trainnew.nnbp,testnew2)
pr.nn1_ <- pr.nn1$net.result*(max(scaledTestdata$Attrition)-
min(scaledTestdata$Attrition))+min(scaledTestdata$Attrition)
test.r1 <- (scaledTestdata$Attrition)*(max(scaledTestdata$Attrition)-
min(scaledTestdata$Attrition))+min(scaledTestdata$Attrition)
MSE.nn1 <- sum((test.r1 - pr.nn1_)^2)/nrow(scaledTestdata)
```

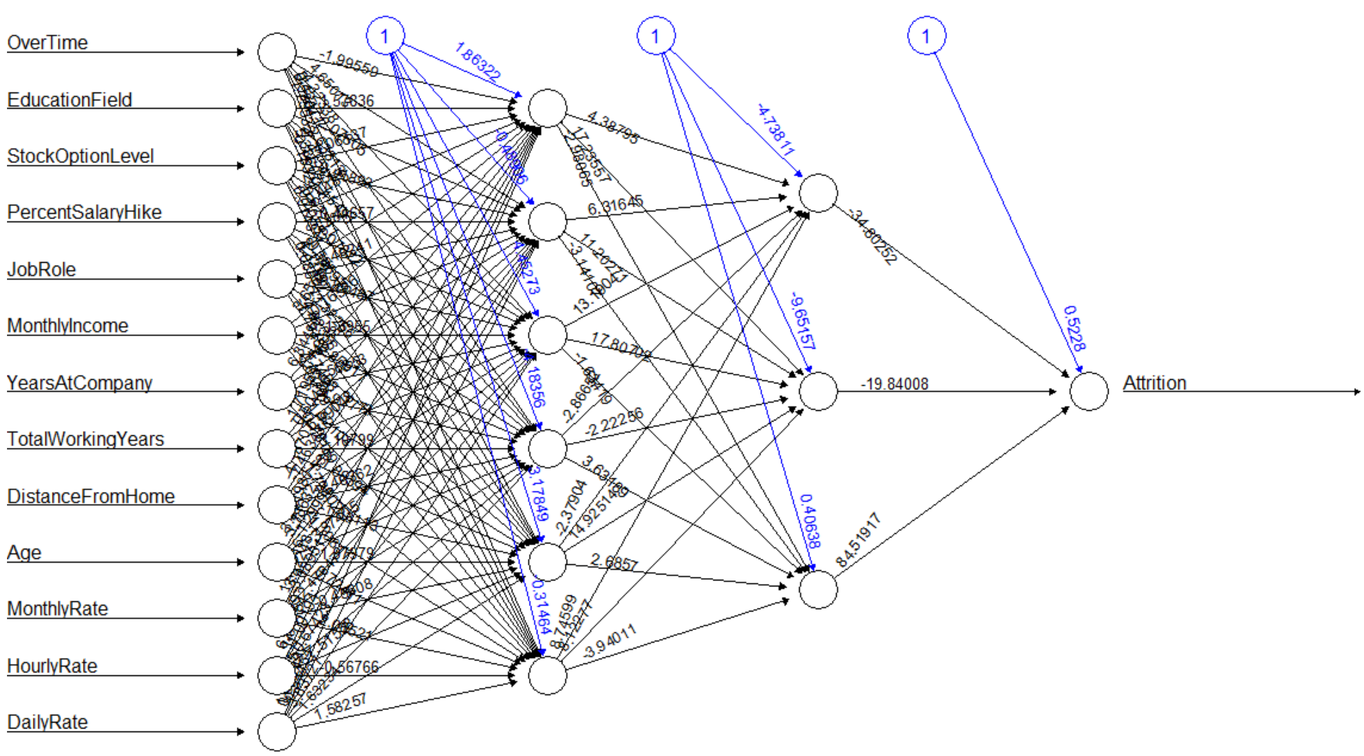
36> The MSE value was found to be 0.0865

4.4 Neural network with another model

37> Now lets change the model by selecting variables that are important as per random forest and lets see if the model improves or not

```
trainnew.nnbp1 <-
 neuralnet(
  Attrition ~
OverTime+EducationField+StockOptionLevel+PercentSalaryHike+JobRole+MonthlyIncome+YearsAtCompany+TotalWorkingYears+DistanceFromHome+Age+MonthlyRate+HourlyRate+DailyRate,
  data = scaledTraindata,
  hidden = c(6,3),
  threshold = 0.01,
  err.fct = "sse",
  linear.output = FALSE,
  lifesign = "full",
  lifesign.step = 10,
  stepmax = 1e6
 )
```

38> The model looks as follows



39> Performance evaluation

```
misClassTable1 = data.frame(Attrition = scaledTraindata$Attrition,
                            Predict.score = trainnew.nnbp1$net.result[[1]])
misClassTable1$Predict.class = ifelse(misClassTable1$Predict.score > 0.21, 1, 0)
with(misClassTable1, table(Attrition, Predict.class))
```

```
         Predict.class
Attrition    0    1
        0 1687   29
        1  102  240
```

Misclassification is 131

Misclassification error =131/2058=0.0636=6.4%

Classification Accuracy=93.6%(approx)

40> Confusion matrix

```
confusionMatrix(misClassTable1$Attrition, misClassTable1$Predict.class)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1687   29
         1  102  240

               Accuracy : 0.936346
                 95% CI : (0.9249196, 0.9465089)
    No Information Rate : 0.8692906
    P-Value [Acc > NIR] : < 0.000000000000000022204

                  Kappa : 0.7488472
 Mcnemar's Test P-Value : 0.0000000003161003

            Sensitivity : 0.9429849
            Specificity : 0.8921933
         Pos Pred Value : 0.9831002
         Neg Pred Value : 0.7017544
             Prevalence : 0.8692906
         Detection Rate : 0.8197279
   Detection Prevalence : 0.8338192
      Balanced Accuracy : 0.9175891

       'Positive' Class : 0
```

41> Error Calculation

```
sum((misClassTable1$Attrition - misClassTable1$Predict.score) ^ 2) / 2
```

The Error was calculated to be 65.512 % in this model

42> Now lets see the rank order with deciling

```
misClassTable1$deciles <- decile(misClassTable1$Predict.score)

## Ranking code
tmp_DT = data.table(misClassTable1)
rank <- tmp_DT[, list(
  cnt = length(Attrition),
  cnt_resp = sum(Attrition),
  cnt_non_resp = sum(Attrition == 1)) ,
  by=deciles][order(-deciles)]
rank$rrate <- round (rank$cnt_resp / rank$cnt,2);
rank$cum_resp <- cumsum(rank$cnt_resp)
rank$cum_non_resp <- cumsum(rank$cnt_non_resp)
rank$cum_rel_resp <- round(rank$cum_resp / sum(rank$cnt_resp),2);
rank$cum_rel_non_resp <- round(rank$cum_non_resp / sum(rank$cnt_non_resp),2);
rank$ks <- abs(rank$cum_rel_resp - rank$cum_rel_non_resp)
library(scales)
rank$rrate <- percent(rank$rrate)
rank$cum_rel_resp <- percent(rank$cum_rel_resp)
rank$cum_rel_non_resp <- percent(rank$cum_rel_non_resp)

View(rank)
```

Rank

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	206	177	177	86%	177	177	52%	52%	0
2	9	207	63	63	30%	240	240	70%	70%	0
3	8	206	12	12	6%	252	252	74%	74%	0
4	7	205	17	17	8%	269	269	79%	79%	0
5	6	206	14	14	7%	283	283	83%	83%	0
6	5	206	14	14	7%	297	297	87%	87%	0
7	4	205	11	11	5%	308	308	90%	90%	0
8	3	206	11	11	5%	319	319	93%	93%	0
9	2	206	9	9	4%	328	328	96%	96%	0
10	1	205	14	14	7%	342	342	100%	100%	0

43> First decile has rrate of 86% and accounts for 52% of Attrition value

44> Second decile has rrate of 30% and accounts for 70% of Attrition value

45> MSE value for this model is calculated to be of 0.182

```
column=c(20,8,24,14,21,17,28,25,6,2,18,11,4)
pr.nn <- compute(trainnew.nnbp1,scaledTestdata[,column])

pr.nn_ <- pr.nn$net.result*(max(scaledTestdata$Attrition)-
min(scaledTestdata$Attrition))+min(scaledTestdata$Attrition)
test.r <- (scaledTestdata$Attrition)*(max(scaledTestdata$Attrition)-
min(scaledTestdata$Attrition))+min(scaledTestdata$Attrition)

MSE.nn <- sum((test.r - pr.nn_)^2)/nrow(scaledTestdata)
MSE.nn
```

46> We then compare the two MSEs

```
"0.0865254853333442 0.181853376270181"
```

47> From the above we conclude that the first model is better than the second model.

5. Conclusion

1> From section 2.1 it is seen that the data set has 2940 observations and each observation has data for 35 variables.

2> From section 2.1.1 the data set has data of 1922 Research and Development department employees, 892 Sales employees and 126 Human Resources employees.

3> From section 2.1.2 it is seen that out of 2510 entries 474 has a Attrition value of Yes and 2466 has Attrition value as No.

4> From section 2.1.3 it is found that the data consists of 1764 male and 1176 female employees which implies that the data consists of 60% of male employees and 40% of female employees.

5> From section 2.1.4 it is seen that the data has 71% of the employees who travel rarely,18.8% of employoess travel frequently and only 10.2% of employees don't travel at all.

6> From section 2.1.5 it is seen that 41.2% of employees are from Life Sciences subject which counts to 1212,31.6% are from medical background which accounts to 928 count, 318 employees are from marketing education background which accounts to 10.8% of the data set,264 employees have technical degree which accounts to 9% of population,164 employees have other education field which is 5.6% of data and 1.8% of the population are from Human Resources stream which accounts to only 54 employees.

7> From section 2.1.6 it is seen that the data consists of 22.2% of Sales Executive and 19.9 percent of data are of research scientists. Human Resource accounts for 3.5% of data as per job role. As per job role the contribution of each role in the data set are as follows

Healthcare Representative = 262 (8.9% of data set)

Human Resources = 104 (3.5% of data set)

Laboratory Technician = 518 (17.6% of data set)

Manager = 204 (6.9% of data set)

Manufacturing Director = 290 (9.9% of data set)

Research Director = 160 (5.4% of data set)

Research Scientist = 584 (19.9% of data set)

Sales Executive = 652 (22.2% of data set)

Sales Representative =166 (5.6% of data set)

8> From section 2.1.7 it is seen that 45.8% of the population are married ,32% are single and 22.2% of the dataset are divorced and it is good to see that there is no employee whose age is below 18 years of age. All 2940 employees, whose data were taken are of 18 years and above age.

9> From section 2.1.8 it is inferred that the age variable is not much skewed. From the histogram it is seen that the maximum age is 60 and minimum is 18 years . Median is at 36. 75% of the data has age below 43. 50% of the age lie above 36 and 50% lie between 30 and 36 years. Mean is slightly greater than median means the data might be very slightly positively skewed.

10> From section 2.1.9 it is inferred that the daily rate variable is not much skewed. From the histogram it is seen that the maximum daily rate is 1499 unit and minimum is 102 unit. median is at 802. 75% of the data has daily rate below 1157. 50% of the daily rate lie above 1157 and 50% lie between 465 and 802. Mean is slightly greater than median means the data might be very slightly positively skewed.

11> From section 2.1.10 it is inferred that the distance from home variable is skewed. From the histogram it is seen that the data is positively skewed. maximum distance from home value is 29 unit and minimum is 1unit. median is at 7unit. 75% of the data has distance from home value below 14unit. 50% of the data has distance from home value lie above 9.193unit and 50% of value lie between 2unit and 7unit. Mean is greater than median means the data is positively skewed.

12> From section 2.1.11 it is inferred that the hourly rate variable is almost normal. From the histogram it is seen that the maximum hourly rate is 100 unit and minimum is 30 unit. median is at 66. 75% of the data has hourly rate below 84unit. 50% of the hourly rate lie above 65.89unitand 50% lie between 48 and 66.

13> From section 2.1.12 it is inferred that the monthly income is skewed and also there are more outliers. From the histogram it is seen that the minimum monthly income is 1009 and maximum is 19999 which is an outlier. As there are outliers so we cannot consider any summary data.

14> From section 2.1.13 it is inferred that the monthlyrate variable is almost normal. From the histogram it is seen that the maximum monthly rate is 26999 unit and minimum is 2094 unit. median is at 14236unit. 75% of the data has monthly rate below 20462unit. 50% of the hourly rate lie above 14313unit and 50% of data lie between 8045 and 14236.

15> From section 2.1.14 it is inferred that the NoOFCompaniesWorked variable is skewed. From the histogram it is seen that the minimum value is 0 and maximum value is 9. median is at 2. 75% of the data has value below 4. 50% of the value lie above 2.693 and 50% of data lie between 1 and 2.

16> From section 2.1.16 it is seen that Sales Representative job role has significantly more attrition relative to other job roles.

17> From section 2.1.17 it is seen that Job level 1 has higher attrition rates over other levels and after job level1 job level 3 has maximum attritions.

18> From section 2.1.18 it is seen that Attrition rate is higher in those who does overtime than the employees who does not do overtime.

19> From section 2.1.19 it is seen that JobInvolvement = 1 has higher attrition than the other.

20> From section 2.1.20 it is seen that Attrition is high in frequent traveller employee.

21> From section 2.1.21 it is concluded that attrition is more in Sales representative role who travel frequently and there is no attrition seen in Non-travelling Sales representative.We see no attrition in non-travelling HR,MD, research director .Contradictory to above results we see Managers are happy who travel a lot i.e the data says there is no attrition in Manager job role who travel frequently.Among all the job roles we see there is very minimal attrition rate in Research Director than in other job roles.After non travelling sales representative the data says the second highest attrition is in Human Resource Job role who travel frequently.

22> From section 2.1.22 it is seen that there is high attrition among hourly sales representatives and lab technicians.

23> From section 2.1.23 it is seen that Attrition rate is high With low work life balance For Lab Technician roles.But the opposite is seen For Sales representative.

24> From section 2.1.24 it is seen that highly correlated variables with more than 60% and above corelation factor are MonthlyIncome - TotalWorkingYears with corelation value of 77.2%,YearsAtCompany-YearsWithCurrManager with corelation value of 76.9%,YearsAtCompany-YearsInCurrentRole with corelation value of 75.8%,YearsInCurrentRole-YearsWithCurrManager with corelation value of 71.4%,Age-TotalWorkingYears with corelation value of 68.03%, TotalWorkingYears- YearsAtCompany with corelation value of 62.8% and YearsAtCompany-YearsSinceLastPromotion with corelation value of 61.8%

25> From section 2.1.25 and Principal Component Analysis it is seen that the first principal component explains 27.42% variance. Second component explains 12.9% variance. Third component explains 8.8% variance. Eleventh component explains 2.35% of variance and so on. It is found that Here we see component 1 seems to be influenced DailyRate,MonthlyIncome, TotalWorkingInYears, YearsInCurrentRole, YearsSinceLastPromotion , YearsWithCurrManager . Component 2 is influenced mainly by Age , YearsInCurrentRole, TrainingTimesLastYear . C3 by MonthlyIncome, PercentSalaryHike , TrainingTimesLastYear.C4 by DistanceFromHome , PercentSalaryHike ,MonthlyRate, C5 MonthlyRate, PercentSalaryHike , TrainingTimesLastYear, YearsWithCurrManager

26> From section 2.1.26 and Factor Analysis it is seen that the important factors are Age,DailyRate,DistanceFromHome,MonthlyRate,MonthlyIncome,Hourlyrate,NumberOfCompa niesWorked,PercentSalaryHike,YearsAtCompany,YearsInCurrentRole,YearsSinceLastPromotio n ,YearsWithCurrManager

27> From section 3 ,the Random forest model was built whose OOB error was 4.57%

28> From section 3.1 , the performance of Random Forest model is as follows KS value is 1,auc value is 1,minauc,maxauc all values are 1

29> From section 3.2 it is seen that Important variables that contribute to the Attrition as per the data set and random forest are

Number of companies worked, EducationField, MonthlyIncome, JobRole, TotalWorkingYears, YearsAtCompany, StockOptionLevel, PercentSalaryHike,Age, DistanceFromHome, HourlyRate, MonthlyRate, DailyRate.

Least important variables are Marital Status, Department and Gender

30> From section 4.3 the neural network model had the classification accuracy of 96.02% and No information rate of 86% and MSE value was 0.0865

31> From section 4.4 the neural network model had the classification accuracy of 93.63% and no information rate of 86.93% and MSE value was found to be 0.182

31> From section 4.3 and 4.4 it is found that the following model is best as per neural network

Attrition ~ Age + BusinessTravel + DailyRate + Department + DistanceFromHome +Education + EducationField + EnvironmentSatisfaction + Gender + HourlyRate +JobInvolvement + JobLevel + JobRole + JobSatisfaction + MaritalStatus +MonthlyIncome + MonthlyRate + NumCompaniesWorked + OverTime + PercentSalaryHike +PerformanceRating + RelationshipSatisfaction + StockOptionLevel + TotalWorkingYears +TrainingTimesLastYear + WorkLifeBalance + YearsAtCompany + YearsInCurrentRole +YearsSinceLastPromotion + YearsWithCurrManager

6. Appendix

6.1 Appendix 1

```
library(ggplot2)
library(scales)
library(dplyr)
library(reshape)
library(PerformanceAnalytics)
library(Hmisc)
library(caTools)
setwd("E:\\PGPBA\\PGPBA-GreatLakes\\Modules\\DataMining\\Assignment\\Assignment2")
hrdata = read.csv("HR_Employee_Attrition_Data.csv",header = TRUE,sep = ",")
dim(hrdata)
str(hrdata)

#'data.frame':	2940 obs. of  35 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 ...
```

```
# we see that there are a total of 2940 observations with 35 variables. There is no NA values in the data.Some variables needs to be converted to factor variables.

#Lets convert some variables into factor variable.

hrdata = hrdata[,-c(9,10,27)]

hrdata$Education <- as.factor(hrdata$Education)
```

```
hrdata$EnvironmentSatisfaction <-
  as.factor(hrdata$EnvironmentSatisfaction)
hrdata$JobInvolvement <- as.factor(hrdata$JobInvolvement)
hrdata$JobSatisfaction <- as.factor(hrdata$JobSatisfaction)
hrdata$PerformanceRating <- as.factor(hrdata$PerformanceRating)
hrdata$RelationshipSatisfaction <-
  as.factor(hrdata$RelationshipSatisfaction)
hrdata$WorkLifeBalance <- as.factor(hrdata$WorkLifeBalance)
hrdata$JobLevel <- as.factor(hrdata$JobLevel)
hrdata$StockOptionLevel <- as.factor(hrdata$StockOptionLevel)

str(hrdata)

# 'data.frame':   2940 obs. of  35 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               : Factor w/ 5 levels "1","2","3","4",..: 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 : Factor w/ 4 levels "1","2","3","4": 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          : Factor w/ 4 levels "1","2","3","4": 3 2 2 3 3 3 4 3 2 3 ...
# $ JobLevel                : Factor w/ 5 levels "1","2","3","4",..: 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        : Factor w/ 4 levels "1","2","3","4": 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      : Factor w/ 2 levels "3","4": 1 2 1 1 1 1 2 2 2 1 ...
# $ RelationshipSatisfaction: Factor w/ 4 levels "1","2","3","4": 1 4 2 3 4 3 1 2 2 2 ...
# $ StandardHours          : int  80 80 80 80 80 80 80 80 80 80 ...
# $ StockOptionLevel       : Factor w/ 4 levels "0","1","2","3": 1 2 1 1 2 1 4 2 1 3 ...
# $ 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        : Factor w/ 4 levels "1","2","3","4": 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 ...

sapply(hrdata, is.numeric)
# Age              Attrition          BusinessTravel              DailyRate
# TRUE                FALSE                FALSE                 TRUE
# Department        DistanceFromHome            Education          EducationField
# FALSE                TRUE                FALSE                FALSE
# EmployeeCount          EmployeeNumber  EnvironmentSatisfaction          Gender
# TRUE                TRUE                FALSE                FALSE
# HourlyRate         JobInvolvement              JobLevel            JobRole
# TRUE                FALSE                FALSE                FALSE
# JobSatisfaction          MaritalStatus          MonthlyIncome        MonthlyRate
```

```
# FALSE                FALSE                TRUE                TRUE
# NumCompaniesWorked    Over18            OverTime      PercentSalaryHike
# TRUE                FALSE                FALSE                TRUE
# PerformanceRating RelationshipSatisfaction        StandardHours       StockOptionLevel
# FALSE                FALSE                TRUE                FALSE
# TotalWorkingYears    TrainingTimesLastYear       WorkLifeBalance        YearsAtCompany
# TRUE                TRUE                FALSE                TRUE
# YearsInCurrentRole  YearsSinceLastPromotion    YearsWithCurrManager
# TRUE                TRUE                TRUE

summary(hrdata)
counts <- table(hrdata$Department)
bp = barplot(counts, main="Employee distribution by department",
      xlab="Number of Employees")
## Add text at top of bars
text(x = bp, y = counts, label = counts, pos = 1, cex = 0.8, col = "red")
## Add x-axis labels

countsAttrition <- table(hrdata$Attrition)
bpAttr = barplot(countsAttrition, main="Attrition Count",
          xlab="Count")
## Add text at top of bars
text(x = bpAttr, y = countsAttrition, label = countsAttrition, pos = 1, cex = 0.8, col = "red")

 countsbygender <- table(hrdata$Gender)
 bpSex = barplot(countsbygender, main="Gender Count", xlab="Count")
 ## Add text at top of bars
 text(x = bpSex, y = countsbygender, label = countsbygender, pos = 1, cex = 0.8, col = "red")

 library(plotrix)
```

```
slices <- table(hrdata$Gender)
lbls <- names(slices)
pct <- prop.table(slices)*100
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
pie3D(slices,labels = lbls,explode = 0.1, main="gender ratio")


countsbyBusinessTravle <- table(hrdata$BusinessTravel)
bpBt = barplot(countsbyBusinessTravle, main="Count by Business Travel", xlab="Count")
## Add text at top of bars
text(x = bpBt, y = countsbyBusinessTravle, label = countsbyBusinessTravle, pos = 1, cex = 0.8, col =
"red")
lbls <- names(countsbyBusinessTravle)
pct <- round(prop.table(countsbyBusinessTravle)*100,1)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
pie3D(countsbyBusinessTravle,labels = lbls,explode = 0.1, main="Distribution by Business travel")



countsbyEF <- table(hrdata$EducationField)
bpBt = barplot(countsbyEF, main="Count by Education field", xlab="Count",las=2,cex.names =
0.5,beside = TRUE)
## Add text at top of bars
text(x = bpBt, y = countsbyEF, label = countsbyEF, pos = 1, cex = 0.5, col = "red")


lbls <- names(countsbyEF)
pct <- round(prop.table(countsbyEF)*100,1)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
oldpar=par()
par(cex=0.3)
```

```
pie3D(countsbyEF,labels = lbls,explode = 0.1, main="Distribution by Education field")
dev.off()
par(oldpar)

summary(hrdata$EducationField)

countsbyjobrole <- table(hrdata$JobRole)
bpJr = barplot(countsbyjobrole, main="Count by Job Role", xlab="Count",las=2,cex.names = 0.5,beside
= TRUE)
## Add text at top of bars
text(x = bpJr, y = countsbyjobrole, label = countsbyjobrole, pos = 1, cex = 0.5, col = "red")

lbls <- names(countsbyjobrole)
pct <- round(prop.table(countsbyjobrole)*100,1)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
oldpar=par()
par(cex=0.3)
pie3D(countsbyjobrole,labels = lbls,explode = 0.1, main="Distribution by Job Role")
dev.off()
par(oldpar)
summary(hrdata$JobRole)

distrByMaritalStat <- table(hrdata$MaritalStatus)
lbls <- names(distrByMaritalStat)
pct <- round(prop.table(distrByMaritalStat)*100,1)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
oldpar=par()
par(cex=0.3)
```

```
pie3D(distrByMaritalStat,labels = lbls,explode = 0.1, main="Distribution by Marital Status")

dev.off()

par(oldpar)


# Age        Attrition          BusinessTravel   DailyRate                    Department
DistanceFromHome

# Min.   :18.00   No :2466   Non-Travel       : 300   Min.   : 102.0   Human Resources       : 126   Min.   :
1.000

# 1st Qu.:30.00   Yes: 474   Travel_Frequently: 554   1st Qu.: 465.0   Research & Development:1922   1st
Qu.: 2.000

# Median :36.00              Travel_Rarely    :2086   Median : 802.0   Sales                 : 892   Median : 7.000

# Mean   :36.92                                       Mean   : 802.5                               Mean   : 9.193

# 3rd Qu.:43.00                                       3rd Qu.:1157.0                               3rd Qu.:14.000

# Max.   :60.00                                       Max.   :1499.0                               Max.   :29.000

#

# Education        EducationField EmployeeCount EmployeeNumber   EnvironmentSatisfaction    Gender

# 1: 340    Human Resources :  54   Min.   :1     Min.   :   1.0   1:568                   Female:1176

# 2: 564    Life Sciences   :1212   1st Qu.:1     1st Qu.: 735.8   2:574                   Male  :1764

# 3:1144    Marketing       : 318   Median :1     Median :1470.5   3:906

# 4: 796    Medical         : 928   Mean   :1     Mean   :1470.5   4:892

# 5:  96    Other           : 164   3rd Qu.:1     3rd Qu.:2205.2

#           Technical Degree: 264   Max.   :1     Max.   :2940.0

#

# HourlyRate     JobInvolvement JobLevel                    JobRole    JobSatisfaction  MaritalStatus

# Min.   : 30.00   1: 166         1:1086   Sales Executive          :652   1:578           Divorced: 654

# 1st Qu.: 48.00   2: 750         2:1068   Research Scientist       :584   2:560           Married :1346

# Median : 66.00   3:1736         3: 436   Laboratory Technician    :518   3:884           Single  : 940

# Mean   : 65.89   4: 288         4: 212   Manufacturing Director   :290   4:918

# 3rd Qu.: 84.00                  5: 138   Healthcare Representative:262

# Max.   :100.00                           Manager                  :204

#                                          (Other)                  :430
```

```
# MonthlyIncome    MonthlyRate    NumCompaniesWorked Over18   OverTime   PercentSalaryHike  PerformanceRating
# Min.   : 1009   Min.   : 2094   Min.   :0.000      Y:2940   No :2108   Min.   :11.00     3:2488
# 1st Qu.: 2911   1st Qu.: 8045   1st Qu.:1.000               Yes: 832   1st Qu.:12.00     4: 452
# Median : 4919   Median :14236   Median :2.000                          Median :14.00
# Mean   : 6503   Mean   :14313   Mean   :2.693                          Mean   :15.21
# 3rd Qu.: 8380   3rd Qu.:20462   3rd Qu.:4.000                          3rd Qu.:18.00
# Max.   :19999   Max.   :26999   Max.   :9.000                          Max.   :25.00
#
# RelationshipSatisfaction StandardHours StockOptionLevel TotalWorkingYears TrainingTimesLastYear WorkLifeBalance
# 1:552                    Min.   :80    0:1262           Min.   : 0.00     Min.   :0.000         1: 160
# 2:606                    1st Qu.:80    1:1192           1st Qu.: 6.00     1st Qu.:2.000         2: 688
# 3:918                    Median :80    2: 316           Median :10.00     Median :3.000         3:1786
# 4:864                    Mean   :80    3: 170           Mean   :11.28     Mean   :2.799         4: 306
#                          3rd Qu.:80                     3rd Qu.:15.00     3rd Qu.:3.000
#                          Max.   :80                     Max.   :40.00     Max.   :6.000
#
# YearsAtCompany   YearsInCurrentRole YearsSinceLastPromotion YearsWithCurrManager
# Min.   : 0.000   Min.   : 0.000     Min.   : 0.000          Min.   : 0.000
# 1st Qu.: 3.000   1st Qu.: 2.000     1st Qu.: 0.000          1st Qu.: 2.000
# Median : 5.000   Median : 3.000     Median : 1.000          Median : 3.000
# Mean   : 7.008   Mean   : 4.229     Mean   : 2.188          Mean   : 4.123
# 3rd Qu.: 9.000   3rd Qu.: 7.000     3rd Qu.: 3.000          3rd Qu.: 7.000
# Max.   :40.000   Max.   :18.000     Max.   :15.000          Max.   :17.000
```

```
age = hrdata[,1]
boxplot(age,main = "box plot of age")
hist(age,main = "histogram of the age")

dailyrate = hrdata[,4]
```

```
boxplot(age,main = "box plot of daily rate")

hist(age,main = "histogram of the daily rate")


distanceFromHome = hrdata[,6]

boxplot(distanceFromHome,main = "box plot of distance from home")

hist(distanceFromHome,main = "histogram of the distance from home")


hourlyrate = hrdata[,13]

boxplot(hourlyrate,main = "box plot of hourly rate")

hist(hourlyrate,main = "histogram of hourly rate")


monthlyincome = hrdata[,19]

boxplot(monthlyincome,main = "box plot of monthly income")

hist(monthlyincome,main = "histogram of monthly income")


monthlyrate = hrdata[,20]

boxplot(monthlyrate,main = "box plot of monthly rate")

hist(monthlyrate,main = "histogram of monthly rate")


numberOfCompaniesWorked = hrdata[,21]

boxplot(numberOfCompaniesWorked,main = "box plot of no.of companies worked")

hist(numberOfCompaniesWorked,main = "histogram of no.of companies worked")



#Hypothesis

#Ho= Job role doesnot affect the Attrition or Attrition is independent of jobrole or there is no strong corelation between jobrole and Attrition

#Ha= There is a strong corelation between Attrition and JobRole or Attrion and job role are not independent or job role affect the attrition rate

library("MASS")

tbl.jrattr=table(hrdata$JobRole,hrdata$Attrition)
```

chisq.test(tbl.jrattr)

Pearson's Chi-squared test

#

data: tbl.jrattr

X-squared = 172.38, df = 8, p-value < 2.2e-16

#Since p<0.05 so we reject null and accept alternate hypothesis.We conclude that there is strong corelation between Attrition and job role

#Attrition rate across job roles

ggplot(hrdata, aes(x = JobRole, fill = Attrition)) + stat_count(width = 0.5) +

  xlab("Job Role") + ylab("Count") + labs(fill = "Attrition")

ggplot(hrdata, aes(x = JobRole)) + geom_bar(aes(fill = Attrition), position = 'fill') +

  scale_y_continuous(labels = percent_format())

#Ho= Job level doesnot affect the Attrition or Attrition is independent of job level or there is no strong corelation between joblevel and Attrition

#Ha= There is a strong corelation between Attrition and JobLevel or Attrion and job level are not independent or job level affect the attrition rate

tbl.jlattr=table(hrdata$Attrition,hrdata$JobLevel)

chisq.test(tbl.jlattr)

Pearson's Chi-squared test

#

data: tbl.jlattr

X-squared = 145.06, df = 4, p-value < 2.2e-16

#Attrition rate across job labels

ggplot(hrdata, aes(x = JobLevel)) + geom_bar(aes(fill = Attrition), position = 'fill') +

  scale_y_continuous(labels = percent_format())

#Ho= Overtime doesnot affect the Attrition or Attrition is independent of Overtime or there is no strong corelation between Overtime and Attrition

#Ha= There is a strong corelation between Attrition and Overtime or Attrion and Overtime are not independent or Overtime affect the attrition rate

tbl.otattr=table(hrdata$OverTime,hrdata$Attrition)

chisq.test(tbl.otattr)

Pearson's Chi-squared test with Yates' continuity correction

#

data: tbl.otattr

X-squared = 176.61, df = 1, p-value < 2.2e-16

#p value is less than 0.05 and hence we reject the null and accept the alt. hypothesis

#Attrition rate by overtime value

ggplot(hrdata, aes(x = OverTime)) + geom_bar(aes(fill = Attrition), position = 'fill') +

  scale_y_continuous(labels = percent_format())

#Ho= Job involvement doesnot affect the Attrition or Attrition is independent of JobInvolvement or there is no strong corelation between JobInvolvement and Attrition

#Ha= There is a strong corelation between Attrition and JobInvolvement or Attrion and JobInvolvement are not independent or JobInvolvement affect the attrition rate

tbl.jinattr=table(hrdata$Attrition,hrdata$JobInvolvement)

tbl.jinattr

chisq.test(tbl.jinattr)

#p value is less than 0.05 and hence we reject the null and accept the alt. hypothesis

#Attrition rate by JobInvolvement

ggplot(hrdata, aes(x = JobInvolvement)) + geom_bar(aes(fill = Attrition), position = 'fill') +

  scale_y_continuous(labels = percent_format())

#Ho= Business Travel doesnot affect the Attrition or Attrition is independent of Business Travel or there is no strong corelation between Business Travel and Attrition

#Ha= There is a strong corelation between Attrition and Business Travel or Attrion and Business Travel are not independent or Business Travel affect the attrition rate

```
tbl.btnattr=table(hrdata$Attrition,hrdata$BusinessTravel)

print(chisq.test(tbl.btnattr))

#p value is less than 0.05 and hence we reject the null and accept the alt. hypothesis


#Attrition rate by BusinessTravel

ggplot(hrdata, aes(x = BusinessTravel)) + geom_bar(aes(fill = Attrition), position = 'fill') +

  scale_y_continuous(labels = percent_format())



##Attrition rate by Business Travel and Jobrole

library(reshape)

hrdata.m = melt(hrdata)

ggplot(hrdata.m, aes(x = BusinessTravel)) + geom_bar(aes(fill = Attrition),position = 'fill') +

  scale_y_continuous(labels = percent_format()) + facet_wrap( ~ JobRole)


##Attrition rate by overtime and Jobrole

ggplot(hrdata.m, aes(x = OverTime)) + geom_bar(aes(fill = Attrition),position = 'fill') +

  scale_y_continuous(labels = percent_format()) + facet_wrap( ~ JobRole)


##Attrition rate by Worklife balance and Jobrole

ggplot(hrdata.m, aes(x = WorkLifeBalance)) + geom_bar(aes(fill = Attrition),position = 'fill') +

  scale_y_continuous(labels = percent_format()) + facet_wrap( ~ JobRole)



# Code: standard deviation to determine if we need to scale

sapply(hrdata, sd)
```

```
# Age                  Attrition         BusinessTravel            DailyRate
# 9.1338192              0.3678004             0.6653417            403.4404468
# Department       DistanceFromHome            Education          EducationField
# 0.5277025              8.1054851             1.0239907              1.3311426
# EmployeeCount          EmployeeNumber  EnvironmentSatisfaction             Gender
# 0.0000000            848.8492210             1.0928962              0.4899813
# HourlyRate         JobInvolvement              JobLevel               JobRole
# 20.3259687              0.7114401             1.1067516              2.4614024
# JobSatisfaction          MaritalStatus          MonthlyIncome        MonthlyRate
# 1.1026585              0.7299965          4707.1557696          7116.5750213
# NumCompaniesWorked         Over18              OverTime       PercentSalaryHike
# 2.4975840              0.0000000             0.4505298              3.6593150
# PerformanceRating RelationshipSatisfaction        StandardHours        StockOptionLevel
# 0.3607621              1.0810249             0.0000000              0.8519317
# TotalWorkingYears    TrainingTimesLastYear        WorkLifeBalance     YearsAtCompany
# 7.7794579              1.2890513             0.7063556              6.1254828
# YearsInCurrentRole  YearsSinceLastPromotion     YearsWithCurrManager
# 3.6225206              3.2218820             3.5675290

#We see significant difference in SD in our variables so lets scale the data
backup = hrdata
numeric.columns = backup[,unlist(lapply(backup,is.numeric))]
scaled.numeric.columns = scale(numeric.columns)
backup[,unlist(lapply(backup,is.numeric))] =  scaled.numeric.columns
hrdatascaled = backup
hrdatascaled.clean = hrdatascaled[,-c(9,27)]
hrdatascaled.clean.numericonly = hrdatascaled.clean[,unlist(lapply(hrdatascaled.clean,is.numeric))]

# Code: get the most highly correlated variables
```

```
hrdataHighCorr <- function(df)
{
  # find the correlations
  cor.matrix <- cor(df)
  # set the correlations on the diagonal or lower triangle to zero,
  # so they will not be reported as the highest ones:
  diag(cor.matrix) <- 0
  cor.matrix[lower.tri(cor.matrix)] <- 0
  # flatten the matrix into a dataframe for easy sorting
  fm <- as.data.frame(as.table(cor.matrix))
  # assign human-friendly names
  names(fm) <- c("First.Variable", "Second.Variable","Correlation")
  # sort and print the top n correlations
  df = fm[order(abs(fm$Correlation),decreasing = T),]
  res = subset(df, ((df$Correlation >= 0.5 &
                     df$Correlation <= 1) |
                    (df$Correlation >= -1 & df$Correlation <= -0.5)
  ))
  res
}
hrdataHighCorr(numeric.columns)
#            First.Variable        Second.Variable Correlation
# 194         MonthlyIncome      TotalWorkingYears   0.7728932
# 286        YearsAtCompany   YearsWithCurrManager   0.7692124
# 252        YearsAtCompany     YearsInCurrentRole   0.7587537
# 287    YearsInCurrentRole   YearsWithCurrManager   0.7143648
# 188                   Age      TotalWorkingYears   0.6803805
# 233     TotalWorkingYears         YearsAtCompany   0.6281332
# 269        YearsAtCompany YearsSinceLastPromotion   0.6184089
# 270    YearsInCurrentRole YearsSinceLastPromotion   0.5480562
```

```
# 228          MonthlyIncome         YearsAtCompany   0.5142848
# 288 YearsSinceLastPromotion    YearsWithCurrManager   0.5102236
#mydata = numeric.columns[,c(1,7,12,14,15,16,17)]
library(PerformanceAnalytics)

correlationf <-
  function (R, histogram = TRUE, method = c("pearson", "kendall",
                                            "spearman"), ...)
  {
    x = checkData(R, method = "matrix")
    if (missing(method))
      method = method[1]
    panel.cor <-
      function(x, y, digits = 2, prefix = "", use = "pairwise.complete.obs",
               method, cex.cor, ...) {
        usr <- par("usr")
        on.exit(par(usr))
        par(usr = c(0, 1, 0, 1))
        r <- cor(x, y, use = use, method = method)
        txt <- format(c(r, 0.123456789), digits = digits)[1]
        txt <- paste(prefix, txt, sep = "")
        if (missing(cex.cor))
          cex <- 0.8 / strwidth(txt)
        test <- cor.test(x, y, method = method)
        Signif <- symnum(
          test$p.value, corr = FALSE, na = FALSE,
          cutpoints = c(0, 0.001, 0.01, 0.05, 0.1, 1), symbols = c("***",
                                                                   "**", "*", ".", " ")
        )
        text(0.5, 0.5, txt, cex = cex * (abs(r) + 1) / 1.3)
```

```
      text(0.8, 0.8, Signif, cex = cex, col = 2)
    }
  f <- function(t) {
    dnorm(t, mean = mean(x), sd = sd.xts(x))
  }
  hist.panel = function(x, ...) {
    par(new = TRUE)
    hist(
      x, col = "light gray", probability = TRUE, axes = FALSE,
      main = "", breaks = "FD"
    )
    lines(density(x, na.rm = TRUE), col = "red", lwd = 1)
    rug(x)
  }
  if (histogram)
    pairs(
      x, gap = 0, lower.panel = panel.smooth, upper.panel = panel.cor,
      diag.panel = hist.panel, method = method, ...
    )
  else
    pairs(
      x, gap = 0, lower.panel = panel.smooth, upper.panel = panel.cor,
      method = method, ...
    )
 }

correlationf(numeric.columns, histogram = TRUE, pch = '+',cex.cor.scale = 100)

#PCA
backup.hrdata = hrdata
```

hrdatanumericonly = backup.hrdata[,unlist(lapply(backup.hrdata,is.numeric))]

hrdatanumericonly = hrdatanumericonly[,-c(4,11)]

hrdata.pca <- prcomp(hrdatanumericonly,scale. = T)

print(hrdata.pca)

summary(hrdata.pca)

biplot(hrdata.pca)

std_dev <- hrdata.pca$sdev

pr_var <- std_dev ^ 2

#Eigen Values

pr_var

[1] 4.0168448 1.6502343 1.0689120 1.0599450 1.0240371 1.0013568 0.9772699 0.9449862 0.9168311 0.7223793 0.5306896

[12] 0.4698434 0.2832211 0.1933840 0.1400655

#We aim to find the components which explain the maximum variance. This is because, we want to retain as much information as possible using these components. So, higher is the explained variance, higher will be the information contained in those components.

#To compute the proportion of variance explained by each component, we simply divide the variance by sum of total variance. This results in:

prop_varex <- pr_var / sum(pr_var)

prop_varex

[1] 0.26778965 0.11001562 0.07126080 0.07066300 0.06826914 0.06675712 0.06515133 0.06299908 0.06112207 0.04815862

[11] 0.03537931 0.03132289 0.01888140 0.01289226 0.00933770

#This shows that first principal component explains 26.8% variance. Second component explains 11% variance. Third component explains 7.12% variance.Eleventh component explains 3.5% of variance and so on. So, how do we decide how many components should we select for modeling stage ?

#The answer to this question is provided by a scree plot. A scree plot is used to access components or factors which explains the most of variability in the data. It represents values in descending order.

```
plot(prop_varex, xlab = "Principal Component", ylab = "Proportion of Variance Explained",type = "b")

minor.tick(nx = 1,tick.ratio = 1)

minor.tick(ny = 1,tick.ratio = 1)
```

#From the plot it is seen that ~4 components 98% of variance is explained

```
plot(hrdata.pca)
```

#From the plot it is confirmed that ~4 components 98% of variance is explained

```
hrdata.pca$rotation[,1:5]
```

#Here we see component 1 seems to be influenced TotalWorkingYears and YearsAtTheCompany. Component 2 is influenced mainly by Age and NumberOfCompaniesWorked. C3 by DistanceFromHome,PercentSalaryHike and TrainingTimesLastYear.

#C4 by DailyRate and MonthlyRate, C5 by hourly rate and salary hike.

#C6 by SalaryHike and TrainingTime,C7 by monthly rate, C8 by Daily rate and Hourly rate, C9 by Distance from Home,C10 by NumberOfCompaniesWorked

#C11 by YearsSinceLastPromotion and C12 by Age,MonthlyIncome,YearsWithCurrentManager

#Lets do some factor analysis

```
fit.3 <-

  factanal(hrdatascaled.clean.numericonly,factors = 3,rotation = "varimax")

fit.3


fit.4 <-

  factanal(hrdatascaled.clean.numericonly,factors = 4,rotation = "varimax")

fit.4


fit.5 <-

  factanal(hrdatascaled.clean.numericonly,factors = 5,rotation = "varimax")

fit.5
```

```
fit.6 <-
  factanal(hrdatascaled.clean.numericonly,factors = 6,rotation = "varimax")
fit.6
fit.7 <-  factanal(hrdatascaled.clean.numericonly,factors = 7,rotation = "varimax")

# We can "clean up" the factor pattern in several ways. One way is to hide small
# loadings, to reduce the visual clutter in the factor pattern. Another is to reduce the
# number of decimal places from 3 to 2. A third way is to sort the loadings to make the
# simple structure more obvious. The following command does all three.
print(fit.6, digits = 2, cutoff = .2, sort = TRUE)




#RandomForest
set.seed(123)
str(hrdata)
hrdata.sample = hrdata


# Creating Development and Validation Sample
hrdata.sample$split <- runif(nrow(hrdata.sample), 0, 1);
hrdata.sample <- hrdata.sample[order(hrdata.sample$split),]

#Now, if you view the hrdata.sample dataset again you would notice a new column added at the end
#Now, you can split the dataset to training and testing as follows
hrdata.train  <- hrdata.sample[which(hrdata.sample$split <= 0.7),]
hrdata.test <- hrdata.sample[which(hrdata.sample$split > 0.7),]
c(nrow(hrdata.train), nrow(hrdata.test))
#remove the split columns used for partitioning from both train and test data set
```

```
hrdata.train <- hrdata.train[,-c(33)]

hrdata.test <- hrdata.test[,-c(33)]

str(hrdata.train)

str(hrdata.test)


ApplyQuantile <- function(x) {

 cut(x,breaks = c(quantile(hrdata.train$Age,probs = seq(0,1,by = 0.10))),include.lowest = TRUE)

}

str(hrdata.train)

hrdata.new.train = hrdata.train

hrdata.train$AgeGroup = sapply(hrdata.train$Age,ApplyQuantile)

str(hrdata.train)


ApplyQuantile <- function(x) {

 cut(x,breaks = c(quantile(

  hrdata.train$DailyRate,probs = seq(0,1,by = 0.10)

 )),include.lowest = TRUE)

}

hrdata.train$DailyRateGroup = sapply(hrdata.train$DailyRate,ApplyQuantile)

str(hrdata.train)


ApplyQuantile <- function(x) {

 cut(x,breaks = c(unique(

  quantile(

   hrdata.train$DistanceFromHome,probs = seq(0,1,by = 0.10),na.rm = TRUE

  )

 )),include.lowest = TRUE)

}

hrdata.train$DistanceFromHomeGroup = sapply(hrdata.train$DistanceFromHome,ApplyQuantile)

str(hrdata.train)
```

```
ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$HourlyRate,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$HourlyRateGroup = sapply(hrdata.train$HourlyRate,ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$MonthlyIncome,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$MonthlyIncomeGroup = sapply(hrdata.train$MonthlyIncome,ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$MonthlyRate,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$MonthlyRateGroup = sapply(hrdata.train$MonthlyRate,ApplyQuantile)
str(hrdata.train)
```

```
ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$NumCompaniesWorked,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$NumCompaniesWorkedGroup =
sapply(hrdata.train$NumCompaniesWorked,ApplyQuantile)
str(hrdata.train)


ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$PercentSalaryHike,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$PercentSalaryHikeGroup = sapply(hrdata.train$PercentSalaryHike,ApplyQuantile)
str(hrdata.train)


ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$TotalWorkingYears,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$TotalWorkingYearsGroup = sapply(hrdata.train$TotalWorkingYears,ApplyQuantile)
```

```
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$TrainingTimesLastYear,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$TrainingTimesLastYearGroup = sapply(hrdata.train$TrainingTimesLastYear,ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$YearsAtCompany,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$YearsAtCompanyGroup = sapply(hrdata.train$YearsAtCompany,ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$YearsInCurrentRole,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$YearsInCurrentRoleGroup = sapply(hrdata.train$YearsInCurrentRole,ApplyQuantile)
```

```
str(hrdata.train)


ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$YearsSinceLastPromotion,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$YearsSinceLastPromotionGroup =
sapply(hrdata.train$YearsSinceLastPromotion,ApplyQuantile)
str(hrdata.train)


ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$YearsWithCurrManager,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.train$YearsWithCurrManagerGroup =
sapply(hrdata.train$YearsWithCurrManager,ApplyQuantile)
str(hrdata.train)


ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.train$YearsWithCurrManager,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
```

```
hrdata.train$YearsWithCurrManagerGroup =
sapply(hrdata.train$YearsWithCurrManager,ApplyQuantile)

str(hrdata.train)


backup.hrdata.train = hrdata.train

hrdata.train.allfactorvar = backup.hrdata.train[,unlist(lapply(backup.hrdata.train,is.factor))]

backup.hrdata.train.allfactorvar = hrdata.train.allfactorvar

str(hrdata.train.allfactorvar)

backup.hrdata.train = hrdata.train


ApplyQuantile <- function(x) {
  cut(x,breaks = c(quantile(hrdata.test$Age,probs = seq(0,1,by = 0.10))),include.lowest = TRUE)
}

str(hrdata.test)

hrdata.new.test = hrdata.test

hrdata.test$AgeGroup = sapply(hrdata.test$Age,ApplyQuantile)

str(hrdata.test)


ApplyQuantile <- function(x) {
  cut(x,breaks = c(quantile(
    hrdata.test$DailyRate,probs = seq(0,1,by = 0.10)
  )),include.lowest = TRUE)
}

hrdata.test$DailyRateGroup = sapply(hrdata.test$DailyRate,ApplyQuantile)

str(hrdata.test)


ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$DistanceFromHome,probs = seq(0,1,by = 0.10),na.rm = TRUE
```

```
    )
  )),include.lowest = TRUE)
}
hrdata.test$DistanceFromHomeGroup = sapply(hrdata.test$DistanceFromHome,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$HourlyRate,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.test$HourlyRateGroup = sapply(hrdata.test$HourlyRate,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$MonthlyIncome,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.test$MonthlyIncomeGroup = sapply(hrdata.test$MonthlyIncome,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$MonthlyRate,probs = seq(0,1,by = 0.10),na.rm = TRUE
```

```
    )
  )),include.lowest = TRUE)
}
hrdata.test$MonthlyRateGroup = sapply(hrdata.test$MonthlyRate,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$NumCompaniesWorked,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.test$NumCompaniesWorkedGroup = sapply(hrdata.test$NumCompaniesWorked,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$PercentSalaryHike,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.test$PercentSalaryHikeGroup = sapply(hrdata.test$PercentSalaryHike,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$TotalWorkingYears,probs = seq(0,1,by = 0.10),na.rm = TRUE
```

```
    )
  )),include.lowest = TRUE)
}
hrdata.test$TotalWorkingYearsGroup = sapply(hrdata.test$TotalWorkingYears,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$TrainingTimesLastYear,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.test$TrainingTimesLastYearGroup = sapply(hrdata.test$TrainingTimesLastYear,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$YearsAtCompany,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.test$YearsAtCompanyGroup = sapply(hrdata.test$YearsAtCompany,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$YearsInCurrentRole,probs = seq(0,1,by = 0.10),na.rm = TRUE
```

```
    )
  )),include.lowest = TRUE)
}
hrdata.test$YearsInCurrentRoleGroup = sapply(hrdata.test$YearsInCurrentRole,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$YearsSinceLastPromotion,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.test$YearsSinceLastPromotionGroup =
sapply(hrdata.test$YearsSinceLastPromotion,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
      hrdata.test$YearsWithCurrManager,probs = seq(0,1,by = 0.10),na.rm = TRUE
    )
  )),include.lowest = TRUE)
}
hrdata.test$YearsWithCurrManagerGroup = sapply(hrdata.test$YearsWithCurrManager,ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x,breaks = c(unique(
    quantile(
```

```
    hrdata.test$YearsWithCurrManager,probs = seq(0,1,by = 0.10),na.rm = TRUE
   )
  )),include.lowest = TRUE)
}
hrdata.test$YearsWithCurrManagerGroup = sapply(hrdata.test$YearsWithCurrManager,ApplyQuantile)
str(hrdata.test)

backup.hrdata.test = hrdata.test
hrdata.test.allfactorvar = backup.hrdata.test[,unlist(lapply(backup.hrdata.test,is.factor))]
backup.hrdata.test.allfactorvar = hrdata.test.allfactorvar
str(hrdata.test.allfactorvar)
str(hrdata.train.allfactorvar)

#making the levels of some factors equal for test and train data
hrdata.test.allfactorvar$DistanceFromHomeGroup <-
as.character(hrdata.test.allfactorvar$DistanceFromHomeGroup)
hrdata.test.allfactorvar$YearsAtCompanyGroup =
as.character(hrdata.test.allfactorvar$YearsAtCompanyGroup)
hrdata.test.allfactorvar$isTest <- rep(1,nrow(hrdata.test.allfactorvar))
hrdata.train.allfactorvar$isTest <- rep(0,nrow(hrdata.train.allfactorvar))

fullSet <- rbind(hrdata.test.allfactorvar,hrdata.train.allfactorvar)
fullSet$DistanceFromHomeGroup <- as.factor(fullSet$DistanceFromHomeGroup)
fullSet$YearsAtCompanyGroup <- as.factor(fullSet$YearsAtCompanyGroup)

test.new <- fullSet[fullSet$isTest==1,]
test.new = test.new[,-33]
str(test.new)

train.new <- fullSet[fullSet$isTest==0,]
train.new = train.new[,-33]
```

```
library(randomForest)
library(randomForestSRC)
set.seed(2016)

# Manual Search
library(caret)
library(e1071)
train.new=train.new[,-13]
train.new=na.omit(train.new)
library(functional)
train.new[apply(train.new, 1, Compose(is.finite, all)),]
control <- trainControl(method="repeatedcv", number=10, repeats=3)
#tunegrid=expand.grid(.mtry=c(1:15), .ntree=c(500,1000, 1500, 2000, 2500))
seed=2016

#Lest find the best ntree and mtry .Caution: this algorithms takes 48 hours to populate the values for the
Hr attrition csv file
new.modellist <- list()
for (ntree in c(500,1000, 1500,2000,2500)) {
  set.seed(seed)
  print(ntree)
  for(mtryt in c(3:15)){
    tunegrid=expand.grid(.mtry=mtryt)
    key <-toString(paste(toString(ntree),toString(mtryt),sep = "_"))
    print(key)
    print(tunegrid)
    fit <- train(train.new$Attrition~., data=train.new, method="rf", metric="Accuracy", tuneGrid=tunegrid,
trControl=control, ntree=ntree)
    print("new fit added")
    new.modellist[[key]] <- fit
```

```
  }

}
save.image(file="hr_rf.RData")
# compare results
results <- resamples( new.modellist)
summary(results)
#From the resukt we find that the mean accuracy of 94.51% ,which is good when ntree is 1500 and with mtry of 12


arf <-  randomForest(  Attrition ~ .,data = train.new,importance = TRUE,proximity = TRUE,ntree =1500, mtry=12, keep.forest = TRUE)
print(arf)


plot(arf,main = "")
legend(
  "topright", c("OOB", "0", "1"), text.col = 1:6, lty = 1:3, col = 1:3
)
title(main = "Error Rates Random Forest hrdata.train")
arf$err.rate


#pred= predict(arf,test.new)
#table(pred,test.new$Attrition)
# pred    No  Yes
# No  1739    0
# Yes    0  319
#Misclassification rate is 0/2058 which is awesome


#validating the model


## Scoring syntax
```

```
train.new$predict.class <- predict(arf, train.new, type="class")

train.new$predict.score <- predict(arf, train.new, type="prob")

head(train.new)


## deciling


decile <- function(x){

  deciles <- vector(length=10)

  for (i in seq(0.1,1,.1)){

    deciles[i*10] <- quantile(x, i, na.rm=T)

  }

  return (

    ifelse(x<deciles[1], 1,

           ifelse(x<deciles[2], 2,

                  ifelse(x<deciles[3], 3,

                         ifelse(x<deciles[4], 4,

                                ifelse(x<deciles[5], 5,

                                       ifelse(x<deciles[6], 6,

                                              ifelse(x<deciles[7], 7,

                                                     ifelse(x<deciles[8], 8,

                                                            ifelse(x<deciles[9], 9, 10

                                                            ))))))))))

}


train.new$deciles <- decile(train.new$predict.score[,2])

library(plyr)

x = (count(train.new$Attrition == "No"))$freq + (count(train.new$Attrition == "No"))$freq

x[1]

## Ranking code

train.new$Attrition = factor(
```

```
  train.new$Attrition,
  levels = c("Yes", "No"),
  labels = c(1, 0)
)
train.new$Attrition <-
  as.numeric(as.character(train.new$Attrition))
library(data.table)
tmp_DT = data.table(train.new)
rank <- tmp_DT[, list(
  cnt = length(Attrition),
  cnt_resp = sum(Attrition),
  cnt_non_resp = sum(Attrition == 1)),
  by=deciles][order(-deciles)]
rank$rrate <- round(rank$cnt_resp * 100 / rank$cnt,2);
rank$cum_resp <- cumsum(rank$cnt_resp)
rank$cum_non_resp <- cumsum(rank$cnt_non_resp)
rank$cum_rel_resp <- round(rank$cum_resp / sum(rank$cnt_resp),2);
rank$cum_rel_non_resp <- round(rank$cum_non_resp / sum(rank$cnt_non_resp),2);
rank$ks <- abs(rank$cum_rel_resp - rank$cum_rel_non_resp);
View(rank)




#RandomForest performance


library(ROCR)


testp4 <- predict(arf,train.new,type = 'prob')[,2]
```

```
pred4 <- prediction(testp4,train.new$Attrition)

#performance in terms of true and false positive rates

perf4 <- performance(pred4,"tpr","fpr")

#plot the curve

plot(perf4,main="ROC Curve for Random Forest",col=2,lwd=2)

abline(a=0,b=1,lwd=2,lty=2,col="gray")

#compute area under curve

KS <- max(attr(perf4, 'y.values')[[1]]-attr(perf4, 'x.values')[[1]])

auc <- performance(pred4,"auc");

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


minauc<-min(round(auc, digits = 2))

maxauc<-max(round(auc, digits = 2))

minauct <- paste(c("min(AUC) = "),minauc,sep="")

maxauct <- paste(c("max(AUC) = "),maxauc,sep="")


#important factors

#plot variable importance

# The variable importance plot lists variables in terms of importance using the decrease in

# accuracy metric, of loss of predictive power if the variable is dropped, vs. the importance

# in terms of Gini index, a measure of separation of classes.


print(importance(arf,type = 2))

varImpPlot(arf)


save.image(file="hr_rf.RData")
```

6.2 Appendix 2

```
library(ggplot2)

library(scales)
```

```
library(dplyr)

library(reshape)

library(PerformanceAnalytics)

library(Hmisc)

library(caTools)

setwd("E:\\PGPBA\\PGPBA-GreatLakes\\Modules\\DataMining\\Assignment\\Assignment2")

hrdata = read.csv("HR_Employee_Attrition_Data.csv",

                  header = TRUE,

                  sep = ",")


hrdata = hrdata[, -c(9, 10, 22, 27)]

hrdata.transform = hrdata

str(hrdata)

# we see that there are a total of 2940 observations with 35 variables. There is no NA values in the data.Some variables needs to be converted to factor variables.

#Lets convert some variables into factor variable.

hrdata$Education <- as.factor(hrdata$Education)

hrdata$EnvironmentSatisfaction <-

  as.factor(hrdata$EnvironmentSatisfaction)

hrdata$JobInvolvement <- as.factor(hrdata$JobInvolvement)

hrdata$JobSatisfaction <- as.factor(hrdata$JobSatisfaction)

hrdata$PerformanceRating <- as.factor(hrdata$PerformanceRating)

hrdata$RelationshipSatisfaction <-

  as.factor(hrdata$RelationshipSatisfaction)

hrdata$WorkLifeBalance <- as.factor(hrdata$WorkLifeBalance)

hrdata$JobLevel <- as.factor(hrdata$JobLevel)

hrdata$StockOptionLevel <- as.factor(hrdata$StockOptionLevel)


str(hrdata)
```

```
#Neural net
set.seed(123)
hrdata.sample = hrdata


# Creating Development and Validation Sample
hrdata.sample$split <- runif(nrow(hrdata.sample), 0, 1)

hrdata.sample <- hrdata.sample[order(hrdata.sample$split), ]

#Now, if you view the hrdata.sample dataset again you would notice a new column added at the end
#Now, you can split the dataset to training and testing as follows
hrdata.train  <- hrdata.sample[which(hrdata.sample$split <= 0.7), ]
hrdata.test <- hrdata.sample[which(hrdata.sample$split > 0.7), ]
c(nrow(hrdata.train), nrow(hrdata.test))
#remove the split columns used for partitioning from both train and test data set
hrdata.train <- hrdata.train[, -c(32)]
hrdata.test <- hrdata.test[, -c(32)]
str(hrdata.train)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(quantile(hrdata.train$Age, probs = seq(0, 1, by = 0.10))), include.lowest = TRUE)
}
str(hrdata.train)
hrdata.new.train = hrdata.train
hrdata.train$AgeGroup = sapply(hrdata.train$Age, ApplyQuantile)
str(hrdata.train)
```

```
ApplyQuantile <- function(x) {
  cut(x, breaks = c(quantile(
    hrdata.train$DailyRate, probs = seq(0, 1, by = 0.10)
  )), include.lowest = TRUE)
}
hrdata.train$DailyRateGroup = sapply(hrdata.train$DailyRate, ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$DistanceFromHome,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.train$DistanceFromHomeGroup = sapply(hrdata.train$DistanceFromHome,
ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$HourlyRate,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
```

```
}
hrdata.train$HourlyRateGroup = sapply(hrdata.train$HourlyRate, ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$MonthlyIncome,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.train$MonthlyIncomeGroup = sapply(hrdata.train$MonthlyIncome, ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$MonthlyRate,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.train$MonthlyRateGroup = sapply(hrdata.train$MonthlyRate, ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
```

```
    quantile(
      hrdata.train$NumCompaniesWorked,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.train$NumCompaniesWorkedGroup = sapply(hrdata.train$NumCompaniesWorked,
ApplyQuantile)
str(hrdata.train)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$PercentSalaryHike,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.train$PercentSalaryHikeGroup = sapply(hrdata.train$PercentSalaryHike, ApplyQuantile)
str(hrdata.train)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$TotalWorkingYears,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
```

```
  )), include.lowest = TRUE)
}
hrdata.train$TotalWorkingYearsGroup = sapply(hrdata.train$TotalWorkingYears, ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$TrainingTimesLastYear,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.train$TrainingTimesLastYearGroup = sapply(hrdata.train$TrainingTimesLastYear,
ApplyQuantile)
str(hrdata.train)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$YearsAtCompany,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.train$YearsAtCompanyGroup = sapply(hrdata.train$YearsAtCompany, ApplyQuantile)
str(hrdata.train)
```

```
ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$YearsInCurrentRole,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.train$YearsInCurrentRoleGroup = sapply(hrdata.train$YearsInCurrentRole,
ApplyQuantile)
str(hrdata.train)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$YearsSinceLastPromotion,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.train$YearsSinceLastPromotionGroup = sapply(hrdata.train$YearsSinceLastPromotion,
ApplyQuantile)
str(hrdata.train)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.train$YearsWithCurrManager,
      probs = seq(0, 1, by = 0.10),
```

```
    na.rm = TRUE
   )
  )), include.lowest = TRUE)
}
hrdata.train$YearsWithCurrManagerGroup = sapply(hrdata.train$YearsWithCurrManager, ApplyQuantile)
str(hrdata.train)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
   quantile(
    hrdata.train$YearsWithCurrManager,
    probs = seq(0, 1, by = 0.10),
    na.rm = TRUE
   )
  )), include.lowest = TRUE)
}
hrdata.train$YearsWithCurrManagerGroup = sapply(hrdata.train$YearsWithCurrManager, ApplyQuantile)
str(hrdata.train)


backup.hrdata.train = hrdata.train
hrdata.train.allfactorvar = backup.hrdata.train[, unlist(lapply(backup.hrdata.train, is.factor))]
backup.hrdata.train.allfactorvar = hrdata.train.allfactorvar
str(hrdata.train.allfactorvar)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(quantile(hrdata.test$Age, probs = seq(0, 1, by = 0.10))), include.lowest = TRUE)
}
str(hrdata.test)
```

```
hrdata.new.test = hrdata.test

hrdata.test$AgeGroup = sapply(hrdata.test$Age, ApplyQuantile)

str(hrdata.test)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(quantile(
    hrdata.test$DailyRate, probs = seq(0, 1, by = 0.10)
  )), include.lowest = TRUE)
}
hrdata.test$DailyRateGroup = sapply(hrdata.test$DailyRate, ApplyQuantile)

str(hrdata.test)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$DistanceFromHome,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$DistanceFromHomeGroup = sapply(hrdata.test$DistanceFromHome,
ApplyQuantile)

str(hrdata.test)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$HourlyRate,
      probs = seq(0, 1, by = 0.10),
```

```
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$HourlyRateGroup = sapply(hrdata.test$HourlyRate, ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$MonthlyIncome,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$MonthlyIncomeGroup = sapply(hrdata.test$MonthlyIncome, ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$MonthlyRate,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$MonthlyRateGroup = sapply(hrdata.test$MonthlyRate, ApplyQuantile)
str(hrdata.test)
```

```
ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$NumCompaniesWorked,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$NumCompaniesWorkedGroup = sapply(hrdata.test$NumCompaniesWorked, ApplyQuantile)
str(hrdata.test)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$PercentSalaryHike,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$PercentSalaryHikeGroup = sapply(hrdata.test$PercentSalaryHike, ApplyQuantile)
str(hrdata.test)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$TotalWorkingYears,
```

```
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$TotalWorkingYearsGroup = sapply(hrdata.test$TotalWorkingYears, ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$TrainingTimesLastYear,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$TrainingTimesLastYearGroup = sapply(hrdata.test$TrainingTimesLastYear,
ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$YearsAtCompany,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
```

```
hrdata.test$YearsAtCompanyGroup = sapply(hrdata.test$YearsAtCompany, ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$YearsInCurrentRole,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$YearsInCurrentRoleGroup = sapply(hrdata.test$YearsInCurrentRole, ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$YearsSinceLastPromotion,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$YearsSinceLastPromotionGroup = sapply(hrdata.test$YearsSinceLastPromotion,
ApplyQuantile)
str(hrdata.test)

ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
```

```
    quantile(
      hrdata.test$YearsWithCurrManager,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$YearsWithCurrManagerGroup = sapply(hrdata.test$YearsWithCurrManager,
ApplyQuantile)
str(hrdata.test)


ApplyQuantile <- function(x) {
  cut(x, breaks = c(unique(
    quantile(
      hrdata.test$YearsWithCurrManager,
      probs = seq(0, 1, by = 0.10),
      na.rm = TRUE
    )
  )), include.lowest = TRUE)
}
hrdata.test$YearsWithCurrManagerGroup = sapply(hrdata.test$YearsWithCurrManager,
ApplyQuantile)
str(hrdata.test)


backup.hrdata.test = hrdata.test
hrdata.test.allfactorvar = backup.hrdata.test[, unlist(lapply(backup.hrdata.test, is.factor))]
backup.hrdata.test.allfactorvar = hrdata.test.allfactorvar
str(hrdata.test.allfactorvar)
str(hrdata.train.allfactorvar)
```

```
#making the levels of some factors equal for test and train data
hrdata.test.allfactorvar$TotalWorkingYearsGroup <-
  as.character(hrdata.test.allfactorvar$TotalWorkingYearsGroup)
hrdata.test.allfactorvar$isTest <-
  rep(1, nrow(hrdata.test.allfactorvar))
hrdata.train.allfactorvar$isTest <-
  rep(0, nrow(hrdata.train.allfactorvar))

fullSet <- rbind(hrdata.test.allfactorvar, hrdata.train.allfactorvar)
fullSet$TotalWorkingYearsGroup <-
  as.factor(fullSet$TotalWorkingYearsGroup)

test.new <- fullSet[fullSet$isTest == 1, ]
test.new = test.new[, -32]
str(test.new)

train.new <- fullSet[fullSet$isTest == 0, ]
train.new = train.new[, -32]
str(train.new)

library(nnet)
## 1. Fit a Single Hidden Layer Neural Network using Least Squares
train.nnet <-
  nnet(
    Attrition ~ .,
    train.new,
    size = 3,
    rang = 0.07,
    Hess = FALSE,
    decay = 15e-4,
```

```
  maxit = 250
 )
## Use TEST data for testing the trained model

test.nnet <- predict(train.nnet, test.new, type = ("class"))

## MisClassification Confusion Matrix

table(test.new$Attrition, test.nnet)

## One can maximize the Accuracy by changing the "size" while training the neural network. SIZE refers to the number of nodes in the hidden layer.

which.is.max(test.nnet)  ## To Find which row break ties at random (Maximum position in vector)


##2. Use Multinomial Log Linear models using Neural Networks

train.mlln <- multinom(Attrition ~ ., train.new)

##USe TEST data for testing the trained model

test.mlln <- predict(train.mlln, test.new)

##Misclassification or Confusion Matrix

table(test.new$Attrition, test.mlln)


##3. Training Neural Network Using neuralnet

library(neuralnet)


## Check for all Input Independent Variables to be Integer or Numeric or complex matrix or vector arguments. If they are not any one of these, then tranform them accordingly

str(hrdata.train)

str(hrdata.test)


## It can be observed that all are either integer or factor. Now these factors have to be transformed to numeric.

## One cannot use directly as.numeric() to convert factors to numeric as it has limitations.

## First, Lets convert factors having character levels to numeric levels
```

```
hrdata.transform$Attrition = factor(
  hrdata.transform$Attrition,
  levels = c("Yes", "No"),
  labels = c(1, 0)
)
hrdata.transform$BusinessTravel = factor(
  hrdata.transform$BusinessTravel,
  levels = levels(hrdata.transform$BusinessTravel),
  labels = c(1, 2, 3)
)
hrdata.transform$Department = factor(
  hrdata.transform$Department,
  levels = levels(hrdata.transform$Department),
  labels = c(1, 2, 3)
)
hrdata.transform$EducationField = factor(
  hrdata.transform$EducationField,
  levels = levels(hrdata.transform$EducationField),
  labels = c(1, 2, 3, 4, 5, 6)
)
hrdata.transform$Gender = factor(
  hrdata.transform$Gender,
  levels = levels(hrdata.transform$Gender),
  labels = c(1, 2)
)
hrdata.transform$JobRole = factor(
  hrdata.transform$JobRole,
  levels = levels(hrdata.transform$JobRole),
  labels = c(1, 2, 3, 4, 5, 6, 7, 8, 9)
)
```

```
hrdata.transform$MaritalStatus = factor(
  hrdata.transform$MaritalStatus,
  levels = levels(hrdata.transform$MaritalStatus),
  labels = c(1, 2, 3)
)
hrdata.transform$OverTime = factor(
  hrdata.transform$OverTime,
  levels = levels(hrdata.transform$OverTime),
  labels = c(1, 2)
)

## Now convert these numerical factors into numeric
hrdata.transform$Attrition <-
  as.numeric(as.character(hrdata.transform$Attrition))
hrdata.transform$BusinessTravel <-
  as.numeric(as.character(hrdata.transform$BusinessTravel))
hrdata.transform$Department <-
  as.numeric(as.character(hrdata.transform$Department))
hrdata.transform$EducationField <-
  as.numeric(as.character(hrdata.transform$EducationField))
hrdata.transform$Gender <-
  as.numeric(as.character(hrdata.transform$Gender))
hrdata.transform$JobRole <-
  as.numeric(as.character(hrdata.transform$JobRole))
hrdata.transform$MaritalStatus <-
  as.numeric(as.character(hrdata.transform$MaritalStatus))
hrdata.transform$OverTime <-
  as.numeric(as.character(hrdata.transform$OverTime))
hrdata.transform$PerformanceRating <-
  as.numeric(as.character(hrdata.transform$PerformanceRating))
```

```
hrdata.transform$RelationshipSatisfaction <-
  as.numeric(as.character(hrdata.transform$RelationshipSatisfaction))
hrdata.transform$StockOptionLevel <-
  as.numeric(as.character(hrdata.transform$StockOptionLevel))
hrdata.transform$WorkLifeBalance <-
  as.numeric(as.character(hrdata.transform$WorkLifeBalance))
hrdata.transform$Education <-
  as.numeric(as.character(hrdata.transform$Education))
hrdata.transform$WorkLifeBalance <-
  as.numeric(as.character(hrdata.transform$WorkLifeBalance))
hrdata.transform$EnvironmentSatisfaction <-
  as.numeric(as.character(hrdata.transform$EnvironmentSatisfaction))
hrdata.transform$JobInvolvement <-
  as.numeric(as.character(hrdata.transform$JobInvolvement))
hrdata.transform$JobLevel <-
  as.numeric(as.character(hrdata.transform$JobLevel))
hrdata.transform$JobSatisfaction <-
  as.numeric(as.character(hrdata.transform$JobSatisfaction))


str(hrdata.transform)
## Now all the variables are wither intergers or numeric
## Now we shall partition the data into train and test data
library(caret)
set.seed(1234567)
train2 <-
  createDataPartition(hrdata.transform$Attrition, p = 0.7, list = FALSE)
trainnew <- hrdata.transform[train2, ]
testnew <- hrdata.transform[-train2, ]
str(trainnew)
```

```
str(testnew)


str(hrdata)
#scale the data
x <- hrdata.transform[,-2]
nn.devscaled <- scale(x)

nn.devscaled <- cbind(hrdata.transform[2], nn.devscaled)

set.seed(891023)
train3 <-  createDataPartition(nn.devscaled$Attrition, p = 0.7, list = FALSE)
scaledTraindata <- nn.devscaled[train3, ]
scaledTestdata <- nn.devscaled[-train3, ]

## Now lets run the neuralnet model on Train dataset
trainnew.nnbp <-
  neuralnet(
    Attrition ~ Age + BusinessTravel + DailyRate + Department + DistanceFromHome +
      Education + EducationField + EnvironmentSatisfaction + Gender + HourlyRate +
      JobInvolvement + JobLevel + JobRole + JobSatisfaction + MaritalStatus +
      MonthlyIncome + MonthlyRate + NumCompaniesWorked + OverTime + PercentSalaryHike
+
      PerformanceRating + RelationshipSatisfaction + StockOptionLevel + TotalWorkingYears +
      TrainingTimesLastYear + WorkLifeBalance + YearsAtCompany + YearsInCurrentRole +
      YearsSinceLastPromotion + YearsWithCurrManager,
    data = scaledTraindata,
    hidden = c(6,3),
    threshold = 0.01,
    err.fct = "sse",
    linear.output = FALSE,
```

```
    lifesign = "full",
    lifesign.step = 10,
    stepmax = 1e6
  )
summary(trainnew.nnbp)
## The distribution of the estimated results

quantile(trainnew.nnbp$net.result[[1]], c(0,1,5,10,25,50,75,90,95,99,100)/100)
##(Smoother the Curve- Better is the model prediction)
## Plot the trained Neural Network

plot(trainnew.nnbp, rep = "best")

## To check your prediction accuracy of training model
columnc = c(
  "Age",
  "BusinessTravel",
  "DailyRate",
  "Department",
  "DistanceFromHome",
  "Education",
  "EducationField",
  "EnvironmentSatisfaction",
  "Gender",
  "HourlyRate",
  "JobInvolvement",
  "JobLevel",
  "JobRole" ,
  "JobSatisfaction",
  "MaritalStatus",
```

```
  "MonthlyIncome" ,
  "MonthlyRate",
  "NumCompaniesWorked",
  "OverTime" ,
  "PercentSalaryHike",
  "PerformanceRating",
  "RelationshipSatisfaction",
  "StockOptionLevel",
  "TotalWorkingYears",
  "TrainingTimesLastYear",
  "WorkLifeBalance" ,
  "YearsAtCompany" ,
  "YearsInCurrentRole" ,
  "YearsSinceLastPromotion",
  "YearsWithCurrManager"
)


testnew2 <- subset(scaledTestdata, select = columnc)
testnew.nnbp <- compute(trainnew.nnbp, testnew2, rep = 1)
## MisClassification Confusion Matrix
table(testnew$Attrition, testnew.nnbp$net.result)
cbind(testnew$Attrition, testnew.nnbp$net.result)
print(testnew.nnbp)

## Error Computation
misClassTable = data.frame(Attrition = scaledTraindata$Attrition,
                           Predict.score = trainnew.nnbp$net.result[[1]])
misClassTable$Predict.class = ifelse(misClassTable$Predict.score > 0.21, 1, 0)
with(misClassTable, table(Attrition, Predict.class))
```

```
library(e1071)
confusionMatrix(misClassTable$Attrition, misClassTable$Predict.class)
sum((misClassTable$Attrition - misClassTable$Predict.score) ^ 2) / 2



decile <- function(x){
  deciles <- vector(length=10)
  for (i in seq(0.1,1,.1)){
    deciles[i*10] <- quantile(x, i, na.rm=T)
  }
  return (
    ifelse(x<deciles[1], 1,
           ifelse(x<deciles[2], 2,
                  ifelse(x<deciles[3], 3,
                         ifelse(x<deciles[4], 4,
                                ifelse(x<deciles[5], 5,
                                       ifelse(x<deciles[6], 6,
                                              ifelse(x<deciles[7], 7,
                                                     ifelse(x<deciles[8], 8,
                                                            ifelse(x<deciles[9], 9, 10
                                                            ))))))))))
}

## deciling
misClassTable$deciles <- decile(misClassTable$Predict.score)

## Ranking code
library(data.table)
tmp_DT = data.table(misClassTable)
```

```
rank <- tmp_DT[, list(
  cnt = length(Attrition),
  cnt_resp = sum(Attrition),
  cnt_non_resp = sum(Attrition == 1)) ,
  by=deciles][order(-deciles)]
rank$rrate <- round (rank$cnt_resp / rank$cnt,2);
rank$cum_resp <- cumsum(rank$cnt_resp)
rank$cum_non_resp <- cumsum(rank$cnt_non_resp)
rank$cum_rel_resp <- round(rank$cum_resp / sum(rank$cnt_resp),2);
rank$cum_rel_non_resp <- round(rank$cum_non_resp / sum(rank$cnt_non_resp),2);
rank$ks <- abs(rank$cum_rel_resp - rank$cum_rel_non_resp)
library(scales)
rank$rrate <- percent(rank$rrate)
rank$cum_rel_resp <- percent(rank$cum_rel_resp)
rank$cum_rel_non_resp <- percent(rank$cum_rel_non_resp)

View(rank)
rank
#First decile has got 92% and is capturing 55% of total attrition value
#similalry second decile has got 40% and is capturing 79% of total attrition value



pr.nn1 <- compute(trainnew.nnbp,testnew2)


pr.nn1_ <- pr.nn1$net.result*(max(scaledTestdata$Attrition)-
min(scaledTestdata$Attrition))+min(scaledTestdata$Attrition)
test.r1 <- (scaledTestdata$Attrition)*(max(scaledTestdata$Attrition)-
min(scaledTestdata$Attrition))+min(scaledTestdata$Attrition)


MSE.nn1 <- sum((test.r1 - pr.nn1_)^2)/nrow(scaledTestdata)
MSE.nn1
```

```
# 0.08652548533


#Now lets change the model by selecting variables that are important as per random forest and lets see if the model improves or not


trainnew.nnbp1 <-
  neuralnet(
    Attrition ~ OverTime+EducationField+StockOptionLevel+PercentSalaryHike+JobRole+MonthlyIncome+YearsAtCompany+TotalWorkingYears+DistanceFromHome+Age+MonthlyRate+HourlyRate+DailyRate,
    data = scaledTraindata,
    hidden = c(6,3),
    threshold = 0.01,
    err.fct = "sse",
    linear.output = FALSE,
    lifesign = "full",
    lifesign.step = 10,
    stepmax = 1e6
  )
summary(trainnew.nnbp1)
plot(trainnew.nnbp1, rep = "best")


#Performance evaluation


misClassTable1 = data.frame(Attrition = scaledTraindata$Attrition,
                            Predict.score = trainnew.nnbp1$net.result[[1]])
misClassTable1$Predict.class = ifelse(misClassTable1$Predict.score > 0.21, 1, 0)
with(misClassTable1, table(Attrition, Predict.class))


confusionMatrix(misClassTable1$Attrition, misClassTable1$Predict.class)
```

```
#Error calculation
sum((misClassTable1$Attrition - misClassTable1$Predict.score) ^ 2) / 2




#decile
## deciling
misClassTable1$deciles <- decile(misClassTable1$Predict.score)

## Ranking code
tmp_DT = data.table(misClassTable1)
rank <- tmp_DT[, list(
  cnt = length(Attrition),
  cnt_resp = sum(Attrition),
  cnt_non_resp = sum(Attrition == 1)) ,
  by=deciles][order(-deciles)]
rank$rrate <- round (rank$cnt_resp / rank$cnt,2);
rank$cum_resp <- cumsum(rank$cnt_resp)
rank$cum_non_resp <- cumsum(rank$cnt_non_resp)
rank$cum_rel_resp <- round(rank$cum_resp / sum(rank$cnt_resp),2);
rank$cum_rel_non_resp <- round(rank$cum_non_resp / sum(rank$cnt_non_resp),2);
rank$ks <- abs(rank$cum_rel_resp - rank$cum_rel_non_resp)
library(scales)
rank$rrate <- percent(rank$rrate)
rank$cum_rel_resp <- percent(rank$cum_rel_resp)
rank$cum_rel_non_resp <- percent(rank$cum_rel_non_resp)

View(rank)
rank
#First decile has rrate of 80% and accounts for 48% of Attrition value
```

```
#Second decile has rrrate of 42% and accounts for 73% of Attrition value

#OverTime+EducationField+StockOptionLevel+PercentSalaryHike+JobRole+MonthlyIncome+YearsAtCompany+TotalWorkingYears+DistanceFromHome+Age+MonthlyRate+HourlyRate+DailyRate,

column=c(20,8,24,14,21,17,28,25,6,2,18,11,4)

pr.nn <- compute(trainnew.nnbp1,scaledTestdata[,column])


pr.nn_ <- pr.nn$net.result*(max(scaledTestdata$Attrition)-min(scaledTestdata$Attrition))+min(scaledTestdata$Attrition)

test.r <- (scaledTestdata$Attrition)*(max(scaledTestdata$Attrition)-min(scaledTestdata$Attrition))+min(scaledTestdata$Attrition)


MSE.nn <- sum((test.r - pr.nn_)^2)/nrow(scaledTestdata)

MSE.nn

#we then compare the two MSEs



print(paste(MSE.nn1,MSE.nn))

save.image(file="hr_neuralnet.RData")
```