

```
library(tidyverse)
library(dplyr)
library(MASS)
library(ggplot2)
library(GGally)
library(DMwR)
library(car)
library(e1071)
library(caret)
library(cowplot)
library(caTools)
library(pROC)
library(ggcorrplot)
library(lattice)
library(sm)
library(Hmisc)
library(asbio)
library(MVA)
library(Hotelling)
library(Amelia)
library(grid)
library(gridExtra)
library(PerformanceAnalytics)
library(stats)
library(factoextra)
library(corrplot)
library(devtools)
library(data.table)
library(psych)
```

```
install.packages("cluster", lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")
```

```
library(cluster)
```

```
##Importing Data and initial analyses
```

```
#Importing csv file from a location
```

```
attr<- read.csv("C:/Users/sanja/Desktop/Assignments/MVA/HR Data.csv")
```

```
attr <- as.data.frame(attr)
```

```
glimpse(attr)
```

```
#Dimension of the dataset
```

```
dim(attr)
```

```
#View the first 5 rows of the dataset
```

```
head(attr)
```

```
summary(attr)
```

```
#Rename the Age column
```

```
colnames(attr)[1] <- "Age"
```

```
#Calculating the number of null values in each of the columns
```

```
colSums(sapply(attr,is.na))
```

```
missmap(attr,main="Missing Values VS Observed")
```

```
#Removing redundant columns
```

```
attr$EmployeeNumber<- NULL
```

```
attr$StandardHours <- NULL
```

```
attr$Over18 <- NULL
```

```
attr$EmployeeCount <- NULL
```

```
#Converting data type of categorical column
```

```
attr$Education <- factor(attr$Education)
```

```
attr$EnvironmentSatisfaction <- factor(attr$EnvironmentSatisfaction)
```

```

attr$JobInvolvement <- factor(attr$JobInvolvement)
attr$JobLevel <- factor(attr$JobLevel)
attr$JobSatisfaction <- factor(attr$JobSatisfaction)
attr$PerformanceRating <- factor(attr$PerformanceRating)
attr$RelationshipSatisfaction <- factor(attr$RelationshipSatisfaction)
attr$StockOptionLevel <- factor(attr$StockOptionLevel)
attr$WorkLifeBalance <- factor(attr$WorkLifeBalance)

str(attr)

#Assigning categorical and numerical variable to temporary variable
catvar<-c('BusinessTravel','Department','Education','EducationField','EnvironmentSatisfaction','Gender',
          'JobRole','JobInvolvement','JobLevel','JobSatisfaction',

'MaritalStatus','PerformanceRating','RelationshipSatisfaction','StockOptionLevel','WorkLifeBalance')
numvar<-c('Age','DailyRate','DistanceFromHome','HourlyRate',
          'MonthlyIncome','MonthlyRate','NumCompaniesWorked','PercentSalaryHike','TotalWorkingYears',
          'TrainingTimesLastYear','YearsAtCompany',
          'YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager')

##Exploratory Data Analysis

#Vizualization of Attrition

attr %>%
  group_by(Attrition) %>%
  tally() %>%
  ggplot(aes(x =Attrition,y = n,fill=Attrition)) +
  geom_bar(stat = "identity") +
  theme_minimal()+
  labs(x="Attrition", y="Count of Attrition")+
  ggtitle("Attrition")+

```

```
geom_text(aes(label = n), vjust = -0.5, position = position_dodge(0.9))
```

```
#Influence of features on Attrition
```

```
ggplot(data=attr, aes(attr$Age)) +  
  geom_histogram(breaks=seq(20, 50, by=2),  
    col="red",  
    aes(fill=..count..))+  
  labs(x="Age", y="Count")+  
  scale_fill_gradient("Count", low="yellow", high="dark red")
```

```
#Checking for distributions in numerical columns
```

```
#The qqPlot show a few extreme outliers which break the assumption of 95% confidence
```

```
#normal distribution
```

```
par(mfrow = c(1,2))  
hist(attr$Age,xlab="",main = 'Histogram of Age',freq = FALSE)  
lines(density(attr$Age,na.rm = T))  
rug(jitter(attr$Age))  
qqPlot(attr$Age,main='Normal QQ plot of Age')  
par(mfrow=c(1,1))
```

```
par(mfrow = c(1,2))  
hist(attr$DailyRate,xlab="",main = 'Histogram of DailyRate',freq = FALSE)  
lines(density(attr$DailyRate,na.rm = T))  
rug(jitter(attr$DailyRate))  
qqPlot(attr$DailyRate,main='Normal QQ plot of DailyRate')  
par(mfrow=c(1,1))
```

```
par(mfrow = c(1,2))
```

```
hist(attr$DistanceFromHome,xlab="",main = 'Histogram of DistanceFromHome',freq = FALSE)
lines(density(attr$DistanceFromHome,na.rm = T))
rug(jitter(attr$DistanceFromHome))
qqPlot(attr$DistanceFromHome,main='Normal QQ plot of DistanceFromHome')
par(mfrow=c(1,1))
```

```
par(mfrow = c(1,2))
hist(attr$HourlyRate,xlab="",main = 'Histogram of HourlyRate',freq = FALSE)
lines(density(attr$HourlyRate,na.rm = T))
rug(jitter(attr$HourlyRate))
qqPlot(attr$HourlyRate,main='Normal QQ plot of HourlyRate')
par(mfrow=c(1,1))
```

```
par(mfrow = c(1,2))
hist(attr$MonthlyIncome,xlab="",main = 'Histogram of Monthly Income',freq = FALSE)
lines(density(attr$MonthlyIncome,na.rm = T))
rug(jitter(attr$MonthlyIncome))
qqPlot(attr$MonthlyIncome,main='Normal QQ plot of Monthly Income')
par(mfrow=c(1,1))
```

```
# monthly_income_lm = lm(attr$MonthlyIncome~attr$Attrition,data= attr)
# monthly_income_res = resid(monthly_income_lm)
# monthly_income_res_dt=data.table('res'=monthly_income_res)
# ggplot(monthly_income_res_dt,aes(x=res,y=attr$Attrition))+geom_point()
par(mfrow = c(1,2))
hist(attr,attr$NumCompaniesWorked,xlab="",main = 'Histogram of NumCompaniesWorked',freq =
FALSE)
lines(density(attr$NumCompaniesWorked,na.rm = T))
rug(jitter(attr$NumCompaniesWorked))
```

```
qqPlot(attr$NumCompaniesWorked,main='Normal QQ plot of NumCompaniesWorked')  
par(mfrow=c(1,1))
```

```
# par(mfrow = c(1,2))  
# hist(attr$PerformanceRating,xlab='',main = 'Histogram of PerformanceRating',freq = FALSE)  
# lines(density(attr$PerformanceRating,na.rm = T))  
# rug(jitter(attr$PerformanceRating))  
# qqPlot(attr$PerformanceRating,main='Normal QQ plot of PerformanceRating')  
# par(mfrow=c(1,1))
```

```
par(mfrow = c(1,2))  
hist(attr$PercentSalaryHike,xlab='',main = 'Histogram of PercentSalaryHike',freq = FALSE)  
lines(density(attr$PercentSalaryHike,na.rm = T))  
rug(jitter(attr$PercentSalaryHike))  
qqPlot(attr$PercentSalaryHike,main='Normal QQ plot of PercentSalaryHike')  
par(mfrow=c(1,1))
```

```
par(mfrow = c(1,2))  
hist(attr$TrainingTimesLastYear,xlab='',main = 'Histogram of TrainingTimesLastYear',freq = FALSE)  
lines(density(attr$TrainingTimesLastYear,na.rm = T))  
rug(jitter(attr$TrainingTimesLastYear))  
qqPlot(attr$TrainingTimesLastYear,main='Normal QQ plot of TrainingTimesLastYear')  
par(mfrow=c(1,1))
```

```
par(mfrow = c(1,2))  
hist(attr$YearsAtCompany,xlab='',main = 'Histogram of YearsAtCompany',freq = FALSE)  
lines(density(attr$YearsAtCompany,na.rm = T))  
rug(jitter(attr$YearsAtCompany))  
qqPlot(attr$YearsAtCompany,main='Normal QQ plot of YearsAtCompany')
```

```
par(mfrow=c(1,1))
```

```
par(mfrow = c(1,2))
```

```
hist(attr$YearsInCurrentRole,xlab="",main = 'Histogram of YearsInCurrentRole',freq = FALSE)
```

```
lines(density(attr$YearsInCurrentRole,na.rm = T))
```

```
rug(jitter(attr$YearsInCurrentRole))
```

```
qqPlot(attr$YearsInCurrentRole,main='Normal QQ plot of YearsInCurrentRole')
```

```
par(mfrow=c(1,1))
```

```
par(mfrow = c(1,2))
```

```
hist(attr$YearsSinceLastPromotion,xlab="",main = 'Histogram of YearsSinceLastPromotion',freq = FALSE)
```

```
lines(density(attr$YearsSinceLastPromotion,na.rm = T))
```

```
rug(jitter(attr$YearsSinceLastPromotion))
```

```
qqPlot(attr$YearsSinceLastPromotion,main='Normal QQ plot of YearsSinceLastPromotion')
```

```
par(mfrow=c(1,1))
```

```
par(mfrow = c(1,2))
```

```
hist(attr$YearsWithCurrManager,xlab="",main = 'Histogram of YearsWithCurrManager',freq = FALSE)
```

```
lines(density(attr$YearsWithCurrManager,na.rm = T))
```

```
rug(jitter(attr$YearsWithCurrManager))
```

```
qqPlot(attr$YearsWithCurrManager,main='Normal QQ plot of YearsWithCurrManager')
```

```
par(mfrow=c(1,1))
```

```
#Boxplot distributions for our numeric columns
```

```
#The dashed line shows the mean and the dark center line shows the median
```

```
#Difference between these two lines depict the deviation from the central limit theorem
```

```
#Boxplot distributions for Age
```

```
boxplot(attr$Age, ylab = "Age")
```

```
rug(jitter(attr$Age), side = 2)
```

```
abline(h = mean(attr$Age, na.rm = T), lty = 2)
#Plotting the Age with 3 lines for mean, median and mean+std
plot(attr$Age, xlab = "")
abline(h = mean(attr$Age, na.rm = T), lty = 1)
abline(h = mean(attr$Age, na.rm = T) + sd(attr$Age, na.rm = T), lty = 2)
abline(h = median(attr$Age, na.rm = T), lty = 3)
#identify(attr$Age)
```

```
#Boxplot distributions for Daily rate
boxplot(attr$DailyRate, ylab = "DailyRate", outline = TRUE)
rug(jitter(attr$DailyRate), side = 2)
abline(h = mean(attr$DailyRate, na.rm = T), lty = 2)
#Plotting the DailyRate with 3 lines for mean, median and mean+std
plot(attr$DailyRate, xlab = "")
abline(h = mean(attr$DailyRate, na.rm = T), lty = 1)
abline(h = mean(attr$DailyRate, na.rm = T) + sd(attr$DailyRate, na.rm = T), lty = 2)
abline(h = median(attr$DailyRate, na.rm = T), lty = 3)
#identify(attr$DailyRate)
```

```
#Boxplot distributions for Distance from home
boxplot(attr$DistanceFromHome, ylab = "DistanceFromHome", outline = TRUE)
rug(jitter(attr$DistanceFromHome), side = 2)
abline(h = mean(attr$DistanceFromHome, na.rm = T), lty = 2)
#Plotting the Distance from home with 3 lines for mean, median and mean+std
plot(attr$DistanceFromHome, xlab = "")
abline(h = mean(attr$DistanceFromHome, na.rm = T), lty = 1)
abline(h = mean(attr$DistanceFromHome, na.rm = T) + sd(attr$DistanceFromHome, na.rm = T), lty = 2)
abline(h = median(attr$DistanceFromHome, na.rm = T), lty = 3)
#identify(attr$DistanceFromHome)
```



```
#Boxplot distributions for Monthly Income
```

```
boxplot(attr$MonthlyIncome, ylab = "Monthly Income")
```

```
rug(jitter(attr$MonthlyIncome), side = 2)
```

```
abline(h = mean(attr$MonthlyIncome, na.rm = T), lty = 2)
```

```
#Plotting the Monthly Income and Age with 3 lines for mean, median and mean+std
```

```
plot(attr$MonthlyIncome, xlab = "")
```

```
abline(h = mean(attr$MonthlyIncome, na.rm = T), lty = 1)
```

```
abline(h = mean(attr$MonthlyIncome, na.rm = T) + sd(attr$MonthlyIncome, na.rm = T), lty = 2)
```

```
abline(h = median(attr$MonthlyIncome, na.rm = T), lty = 3)
```

```
#identify(attr$MonthlyIncome)
```

```
#Boxplot distributions for NumCompaniesWorked
```

```
boxplot(attr$NumCompaniesWorked, ylab = "NumCompaniesWorked")
```

```
rug(jitter(attr$NumCompaniesWorked), side = 2)
```

```
abline(h = mean(attr$NumCompaniesWorked, na.rm = T), lty = 2)
```

```
#Plotting the NumCompaniesWorked with 3 lines for mean, median and mean+std
```

```
plot(attr$NumCompaniesWorked, xlab = "")
```

```
abline(h = mean(attr$NumCompaniesWorked, na.rm = T), lty = 1)
```

```
abline(h = mean(attr$NumCompaniesWorked, na.rm = T) + sd(attr$NumCompaniesWorked, na.rm = T), lty = 2)
```

```
abline(h = median(attr$NumCompaniesWorked, na.rm = T), lty = 3)
```

```
#identify(attr$NumCompaniesWorked)
```

```
#Boxplot distributions for PercentSalaryHike
```

```
boxplot(attr$PercentSalaryHike, ylab = "PercentSalaryHike")
```

```
rug(jitter(attr$PercentSalaryHike), side = 2)
```

```
abline(h = mean(attr$PercentSalaryHike, na.rm = T), lty = 2)
```

```
#Plotting the PercentSalaryHike with 3 lines for mean, median and mean+std
```

```
plot(attr$PercentSalaryHike, xlab = "")
abline(h = mean(attr$PercentSalaryHike, na.rm = T), lty = 1)
abline(h = mean(attr$PercentSalaryHike, na.rm = T) + sd(attr$PercentSalaryHike, na.rm = T), lty = 2)
abline(h = median(attr$PercentSalaryHike, na.rm = T), lty = 3)
#identify(attr$PercentSalaryHike)
```

#Boxplot distributions for TotalWorkingYears

```
boxplot(attr$TotalWorkingYears, ylab = "TotalWorkingYears")
rug(jitter(attr$TotalWorkingYears), side = 2)
abline(h = mean(attr$TotalWorkingYears, na.rm = T), lty = 2)
#Plotting the TotalWorkingYears with 3 lines for mean, median and mean+std
plot(attr$TotalWorkingYears, xlab = "")
abline(h = mean(attr$TotalWorkingYears, na.rm = T), lty = 1)
abline(h = mean(attr$TotalWorkingYears, na.rm = T) + sd(attr$TotalWorkingYears, na.rm = T), lty = 2)
abline(h = median(attr$TotalWorkingYears, na.rm = T), lty = 3)
#identify(attr$TotalWorkingYears)
```

#Boxplot distributions for TrainingTimesLastYear

```
boxplot(attr$TrainingTimesLastYear, ylab = "TrainingTimesLastYear")
rug(jitter(attr$TrainingTimesLastYear), side = 2)
abline(h = mean(attr$TrainingTimesLastYear, na.rm = T), lty = 2)
#Plotting the TrainingTimesLastYear with 3 lines for mean, median and mean+std
plot(attr$TrainingTimesLastYear, xlab = "")
abline(h = mean(attr$TrainingTimesLastYear, na.rm = T), lty = 1)
abline(h = mean(attr$TrainingTimesLastYear, na.rm = T) + sd(attr$TrainingTimesLastYear, na.rm = T), lty = 2)
abline(h = median(attr$TrainingTimesLastYear, na.rm = T), lty = 3)
```

```
#identify(attr$TrainingTimesLastYear)
```

```
#Boxplot distributions for YearsAtCompany
```

```
boxplot(attr$YearsAtCompany, ylab = "YearsAtCompany")
```

```
rug(jitter(attr$YearsAtCompany), side = 2)
```

```
abline(h = mean(attr$YearsAtCompany, na.rm = T), lty = 2)
```

```
#Plotting the Years at Company with 3 lines for mean, median and mean+std
```

```
plot(attr$YearsAtCompany, xlab = "")
```

```
abline(h = mean(attr$YearsAtCompany, na.rm = T), lty = 1)
```

```
abline(h = mean(attr$YearsAtCompany, na.rm = T) + sd(attr$YearsAtCompany, na.rm = T), lty = 2)
```

```
abline(h = median(attr$YearsAtCompany, na.rm = T), lty = 3)
```

```
#identify(attr$YearsAtCompany)
```

```
#Boxplot distributions for YearsInCurrentRole
```

```
boxplot(attr$YearsInCurrentRole, ylab = "YearsInCurrentRole")
```

```
rug(jitter(attr$YearsInCurrentRole), side = 2)
```

```
abline(h = mean(attr$YearsInCurrentRole, na.rm = T), lty = 2)
```

```
#Plotting the YearsInCurrentRole with 3 lines for mean, median and mean+std
```

```
plot(attr$YearsInCurrentRole, xlab = "")
```

```
abline(h = mean(attr$YearsInCurrentRole, na.rm = T), lty = 1)
```

```
abline(h = mean(attr$YearsInCurrentRole, na.rm = T) + sd(attr$YearsInCurrentRole, na.rm = T), lty = 2)
```

```
abline(h = median(attr$YearsInCurrentRole, na.rm = T), lty = 3)
```

```
#identify(attr$YearsInCurrentRole)
```

```
#Boxplot distributions for YearsSinceLastPromotion
```

```
boxplot(attr$YearsSinceLastPromotion, ylab = "YearsSinceLastPromotion")
```

```
rug(jitter(attr$YearsSinceLastPromotion), side = 2)
```

```
abline(h = mean(attr$YearsSinceLastPromotion, na.rm = T), lty = 2)
```

```
#Plotting the YearsSinceLastPromotion with 3 lines for mean, median and mean+std
```

```
plot(attr$YearsSinceLastPromotion, xlab = "")
abline(h = mean(attr$YearsSinceLastPromotion, na.rm = T), lty = 1)
abline(h = mean(attr$YearsSinceLastPromotion, na.rm = T) + sd(attr$YearsSinceLastPromotion, na.rm =
T), lty = 2)
abline(h = median(attr$YearsSinceLastPromotion, na.rm = T), lty = 3)
#identify(attr$YearsSinceLastPromotion)
```

#Boxplot distributions for YearsWithCurrManager

```
boxplot(attr$YearsWithCurrManager, ylab = "YearsWithCurrManager")
```

```
rug(jitter(attr$YearsWithCurrManager), side = 2)
```

```
abline(h = mean(attr$YearsWithCurrManager, na.rm = T), lty = 2)
```

#Boxplot distributions for YearsWithCurrManager

```
plot(attr$YearsWithCurrManager, xlab = "")
```

```
abline(h = mean(attr$YearsWithCurrManager, na.rm = T), lty = 1)
```

```
abline(h = mean(attr$YearsWithCurrManager, na.rm = T) + sd(attr$YearsWithCurrManager, na.rm =
T), lty = 2)
```

```
abline(h = median(attr$YearsWithCurrManager, na.rm = T), lty = 3)
```

```
#identify(attr$YearsWithCurrManager)
```

#Chi Plot for inspecting the independence

```
chi.plot(attr$MonthlyIncome, attr$Age)
```

```
plot(qc<-qchisq((1:nrow(attr)-1/2)/nrow(attr), df=))
```

#Plotting joint boxplots for various categories wrt numerical column Age

```
bwplot(attr$Department ~ attr$Age, data=attr, ylab='Department', xlab='Age')
```

```
bwplot(attr$Gender ~ attr$Age, data=attr, ylab='Gender', xlab='Age')
```

```
bwplot(attr$EducationField ~ attr$Age, data=attr, ylab='EducationField', xlab='Age')
```

```
bwplot(attr$JobRole ~ attr$Age, data=attr, ylab='JobRole', xlab='Age')
```

```
bwplot(attr$MaritalStatus ~ attr$MonthlyIncome, data=attr, ylab='MaritalStatus', xlab='Age')
```

```
bwplot(attr$BusinessTravel ~ attr$Age, data=attr, ylab='BusinessTravel', xlab='Age')
```

```

#Plotting stripplots for various categories wrt numerical column Age
bwplot(attr$Department ~ attr$Age, data=attr,panel=panel.bppplot,
       probs=seq(.01,.49,by=.01), datadensity=TRUE, ylab='Department',xlab='Age')
bwplot(attr$Gender ~ attr$Age, data=attr,panel=panel.bppplot,
       probs=seq(.01,.49,by=.01), datadensity=TRUE, ylab='Gender',xlab='Age')
bwplot(attr$EducationField ~ attr$Age, data=attr,panel=panel.bppplot,
       probs=seq(.01,.49,by=.01), datadensity=TRUE, ylab='EducationField',xlab='Age')
bwplot(attr$JobRole ~ attr$Age, data=attr,panel=panel.bppplot,
       probs=seq(.01,.49,by=.01), datadensity=TRUE, ylab='JobRole',xlab='Age')
bwplot(attr$MaritalStatus ~ attr$Age, data=attr,panel=panel.bppplot,
       probs=seq(.01,.49,by=.01), datadensity=TRUE, ylab='MaritalStatus',xlab='Age')
bwplot(attr$BusinessTravel ~ attr$Age, data=attr,panel=panel.bppplot,
       probs=seq(.01,.49,by=.01), datadensity=TRUE, ylab='BusinessTravel',xlab='Age')

data<-attr[,c('Age','DailyRate','DistanceFromHome','HourlyRate',
'MonthlyIncome','MonthlyRate','NumCompaniesWorked','PercentSalaryHike','TotalWorkingYears',
'TrainingTimesLastYear','YearsAtCompany',
'YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager')]
chart.Correlation(data,histogram = TRUE,pch=19)

#-----

##Creating Temporary Variables

#-----

#Converting double/int columns to numeric
numeric_col <- c("Age","DailyRate","DistanceFromHome","HourlyRate",
"MonthlyIncome","MonthlyRate","NumCompaniesWorked","PercentSalaryHike","TotalWorkingYears",
"TrainingTimesLastYear","YearsAtCompany",

```

```

      "YearsInCurrentRole", "YearsSinceLastPromotion", "YearsWithCurrManager")
attr[numeric_col] <- sapply(attr[numeric_col], as.numeric)

#Take out the numeric columns from categorical columns and storing them as a separate dataframe
attr_i <- attr[,c("Age", "DailyRate", "DistanceFromHome", "HourlyRate",
  "MonthlyIncome", "MonthlyRate", "NumCompaniesWorked", "PercentSalaryHike", "TotalWorkingYears",
  "TrainingTimesLastYear", "YearsAtCompany",
  "YearsInCurrentRole", "YearsSinceLastPromotion", "YearsWithCurrManager")]
attr_i <- data.frame(scale(attr_i))

#Creating temporary variables for the categorical data
#attr_c <- attr[, -c(2,3,5,8,10,11,12,13,14,15,19,21,22,23)]
#temporary <- data.frame(sapply(attr_c, function(x) data.frame(model.matrix(~x-1, data = attr_c))[, -1]))
#head(temporary)
#View(temporary)
#View(attr)

#Combining the temporary and the numeric columns and create the final dataset
#attr_final <- cbind(attr_i, temporary)
#head(attr_final)
#glimpse(attr_final)

#CorrelationMatrix
# cormatrix <- round(cor(attr_final), 4)
# cormatrix
#Heatmap for correlation matrix

```

#Negative correlations are shown in blue and positive in red

```
# col<- colorRampPalette(c("blue", "white", "red"))(20)
```

```
# heatmap(cormatrix, col=col, symm=TRUE)
```

##Test of Significance

#T-Test

#Null Hypothesis - The two means are equal

#Alternate Hypothesis - Difference in the two means is not zero

#pvalue >= 0.05, accept null hypothesis

#Or

#else accept the alternate hypothesis

#Univariate mean comparison using t test

#Monthly Income and Attrition

```
with(data=attr,t.test(attr$MonthlyIncome[attr$Attrition=="Yes"],attr$MonthlyIncome[attr$Attrition=="No"],var.equal=TRUE))
```

#HourlyRate and Attrition

```
with(data=attr,t.test(attr$HourlyRate[attr$Attrition=="Yes"],attr$HourlyRate[attr$Attrition=="No"],var.equal=TRUE))
```

#Daily Rate and Attrition

```
with(data=attr,t.test(attr$DailyRate[attr$Attrition=="Yes"],attr$DailyRate[attr$Attrition=="No"],var.equal=TRUE))
```

```
#Age and Attrition
```

```
with(data=attr,t.test(attr$Age[attr$Attrition=="Yes"],attr$Age[attr$Attrition=="No"],var.equal=TRUE))
```

```
#DistanceFromHome and Attrition
```

```
with(data =  
attr,t.test(attr$DistanceFromHome[attr$Attrition=="Yes"],attr$Age[attr$Attrition=="No"],var.equal =  
TRUE))
```

```
#Monthly Income and Gender
```

```
with(data =  
attr,t.test(attr$MonthlyIncome[attr$Gender=="Male"],attr$MonthlyIncome[attr$Gender=="Female"],va  
r.equal = TRUE))
```

```
#DistanceFromHome and Gender
```

```
with(data =  
attr,t.test(attr$DistanceFromHome[attr$Gender=="Male"],attr$DistanceFromHome[attr$Gender=="Fe  
male"],var.equal = TRUE))
```

```
#Multivariate mean comparison using Hotelling t test
```

```
#Monthly Income and gender
```

```
t2testgender <- hotelling.test(attr$MonthlyIncome + attr$DistanceFromHome ~ attr$Gender, data=attr)  
cat("T2 statistic =",t2testgender$stat[[1]],"\n")  
print(t2testgender)
```

```
#Monthly Income and Attrition
```

```
t2testattr <- hotelling.test(attr$MonthlyIncome + attr$DistanceFromHome ~ attr$Attrition, data=attr)  
cat("T2 statistic =",t2testattr$stat[[1]],"\n")
```



```

print(t2testattr)
attach(attr)
attach(attr_pca)

#PCA
#plot.new(); dev.off()
#Considering the numeric columns that will help to get variance in data
attr_pca <- attr[,numvar]
# solve the error "Figure margins too large"
par("mar")
par(mar=c(1,1,1,1))
#graphics.off()
#dev.off()
##Matrix Plots, Covariance and Correlations Plots
#ScatterPlot matrix
pairs(attr_pca[,10:14],pch=".",cex=1.5)

#Plotting correlation plot to understand the how feature are related to each other
correplot<-cor(attr_pca)
corrplot(correplot,method="circle")
#Finding the principal components of data
attr_pca_done <- princomp(attr_pca,scores = TRUE, cor = TRUE)
attr_pca_done
names(attr_pca_done)
head(attr_pca_done)
summary(attr_pca_done)
#Extract variance against features
eigenvalues<-attr_pca_done$sdev^2
eigenvalues

```

```
sum(eigenvalues)
names(eigenvalues) <- paste("PC",1:14,sep="")
eigenvalues
sumoflambdas <- sum(eigenvalues)
sumoflambdas
#Variance %
pctvar<- (eigenvalues/sumoflambdas)*100
pctvar
barplot(pctvar,main="scree plot",xlab="Principal Component",ylab="Percent Variation")
#Calculate cumulative of variance
cumvar <- cumsum(pctvar)
cumvar
matlambdas <- rbind(eigenvalues,pctvar,cumvar)
matlambdas

rownames(matlambdas) <- c("Eigenvalues","Prop. variance","Cum. prop. variance")
round(matlambdas,4)
```

```
#Loadings of principal Components
loadings(attr_pca_done)
attr_pca_done$loadings
plot(attr_pca_done)
eigenvec_attr<-attr_pca_done$rotation
#Visualize PCA using Scree plot
fviz_screepLOT(attr_pca_done, type='bar',main='Scree plot')
summary(attr_pca_done)
#Biplot of score variables
biplot(attr_pca_done)
```

```
#Scores of the components
```

```
attr_pca_done$scores[1:10,]
```

```
#Sample scores stored in attr_pca$x
```

```
#We need to calculate the scores on each of these components for each individual in our sample.
```

```
attr_pca_done$x
```

```
#x_pca$x
```

```
typ_pca <- cbind(data.frame(Attrition),attr_pca_done$x)
```

```
typ_pca
```

```
str(typ_pca)
```

```
#typ_pca
```

```
#T-Test-- We see that true difference in all the means is different from zero.
```

```
t.test(PC1~attr$Attrition,data=typ_pca)
```

```
t.test(PC2~attr$Attrition,data=typ_pca)
```

```
t.test(PC3~attr$Attrition,data=typ_pca)
```

```
t.test(PC4~attr$Attrition,data=typ_pca)
```

```
t.test(PC5~attr$Attrition,data=typ_pca)
```

```
t.test(PC6~attr$Attrition,data=typ_pca)
```

```
t.test(PC7~attr$Attrition,data=typ_pca)
```

```
t.test(PC8~attr$Attrition,data=typ_pca)
```

```
t.test(PC9~attr$Attrition,data=typ_pca)
```

```
t.test(PC10~attr$Attrition,data=typ_pca)
t.test(PC11~attr$Attrition,data=typ_pca)
t.test(PC12~attr$Attrition,data=typ_pca)
t.test(PC13~attr$Attrition,data=typ_pca)
t.test(PC14~attr$Attrition,data=typ_pca)
```

```
#F-Test #Testing Variation
```

```
#Variance Test- Test for variance
```

```
var.test(PC1~attr$Attrition,data=typ_pca)
var.test(PC2~attr$Attrition,data=typ_pca)
var.test(PC3~attr$Attrition,data=typ_pca)
var.test(PC4~attr$Attrition,data=typ_pca)
var.test(PC5~attr$Attrition,data=typ_pca)
var.test(PC6~attr$Attrition,data=typ_pca)
var.test(PC7~attr$Attrition,data=typ_pca)
var.test(PC8~attr$Attrition,data=typ_pca)
var.test(PC9~attr$Attrition,data=typ_pca)
var.test(PC10~attr$Attrition,data=typ_pca)
var.test(PC11~attr$Attrition,data=typ_pca)
var.test(PC12~attr$Attrition,data=typ_pca)
var.test(PC13~attr$Attrition,data=typ_pca)
var.test(PC14~attr$Attrition,data=typ_pca)
```

```
#Plotting the scores of Pricipal Component 1 and Principal component 2
```

```
plot(typ_pca$PC1, typ_pca$PC2,xlab="PC1:", ylab="PC2")
abline(h=0)
abline(v=0)
```

```
#Plotting the Variance of Principal Components
```

```
plot(eigenvalues ,xlab= "Component number", ylab = "Component variance", type = "l", main = "Scree diagram")
```

```
#Plotting the Log variance of COmponents
```

```
plot(log(eigenvalues), xlab = "Component number",ylab = "log(Component variance)", type="l",main = "Log(eigenvalue) diagram")
```

```
#Variance of the principal components
```

```
#View(attr_pca_done)
```

```
diag(cov(attr_pca_done$x))
```

```
#x_pca$x[,1]
```

```
#x_pca$x
```

```
#Plotting the scores
```

```
xlim <- range(attr_pca_done$x[,1])
```

```
plot(attr_pca_done$x,xlim=xlim,ylim=xlim)
```

```
#attr_pca_done$rotation[,1]
```

```
#attr_pca_done$rotation
```

```
#Variance plot for each component. We can see that all components play a dominant role.
```

```
plot(attr_pca_done)
```

```
#get the original value of the data based on PCA
```

```
center <- attr_pca_done$center
```

```
scale <- attr_pca_done$scale
```

```
new_attrition <- as.matrix(attr[,2])
```

```
new_attrition
```

```
drop(scale(new_attrition,center=center, scale=scale)%*%attr_pca_done$rotation[,2])
```

```
predict(attr_pca_done)[,2]
```

```
#The aboved two gives us the same thing. predict is a good function to know.
```

```
out <- sapply(10:14,
```

```
function(i){plot(attr$Attrition,attr_pca_done$x[,i],xlab=paste("PC",i,sep=""),ylab="Attrition")})
```

```
out
```

```
pairs(attr_pca_done$x[,10:14], ylim = c(-6,4),xlim = c(-6,4),panel=function(x,y,...){text(x,y,attr$Attrition)})
```

```
# covariance<-cov(attr_pca)
```

```
# cm<-colMeans(attr_pca)
```

```
# cm
```

```
# distance<-dist(scale(attr_pca,center=FALSE))
```

```
# d<-apply(attr_pca,MARGIN = 1,function(attr_pca)+t(attr_pca-cm)%*%solve(covariance)%*%(attr_pca-cm))
```

```
# plot(qc<-qchisq((1:nrow(attr_pca)-1/2)/nrow(attr_pca),df=14),sd<-sort(d),
```

```
#       xlab=expression(paste(chi[14]^2,"Quantile")),ylab="Ordered distances")
```

```
# ous<-which(rank(abs(qc-sd),ties="random")>nrow(attr_pca)-14)
```

```
# text(qc[ous],sd[ous]-1.5,ous)
```

```
# abline(a=0,b=1,col="orange")
```

```
# attr_pca_new<-log(attr_pca[,numvar])
```

```
# covariance<-cov(attr_pca_new)
```

```
# correlation<-cor(attr_pca_new)

# #colmeans

# cm_log<-colMeans(attr_pca_new)

# distance<-dist(scale,center=FALSE)

# d<-apply(attr_pca_new,MARGIN=1,function(attr_pca_new)+t(attr_pca_new - cm_log)%*%
#       solve(covariance)%*%(attr_pca_new - cm))

# plot(qc<-qchisq(1:nrow(paste(attr_pca_new)),df=14),sd<-sort(d))
```

#ClusTERING

#K-Means Clustering

#We implemented non hierchal clustering because of more than 1000 samples

```
attr_k <- read.csv("MVA/Attrition Dataset.csv")
```

```
attach(attr_k)
```

Standardizing the data with scale()

```
attr_std <- scale(attr_pca[,1:14])
```

K-means, k=2, 3, 4, 5, 6

Centers (k's) are numbers thus, 10 random sets are chosen

```
(kmeans2_attr_std <- kmeans(attr_std,2,nstart = 10))
```

```
# Computing the percentage of variation accounted for. Two clusters
```

```
perc.var.2 <- round(100*(1 - kmeans2_attr_std$betweenss/kmeans2_attr_std$totss),1)
```

```
names(perc.var.2) <- "Perc. 2 clus"
```

```
perc.var.2
```

```
# Computing the percentage of variation accounted for. Three clusters
```

```
(kmeans3_attr_std <- kmeans(attr_std,3,nstart = 10))
```

```
perc.var.3 <- round(100*(1 - kmeans3_attr_std$betweenss/kmeans3_attr_std$totss),1)
```

```
names(perc.var.3) <- "Perc. 3 clus"
```

```
perc.var.3
```

```
# Computing the percentage of variation accounted for. Four clusters
```

```
(kmeans4_attr_std <- kmeans(attr_std,4,nstart = 10))
```

```
perc.var.4 <- round(100*(1 - kmeans4_attr_std$betweenss/kmeans4_attr_std$totss),1)
```



```
names(perc.var.4) <- "Perc. 4 clus"
```

```
perc.var.4
```

```
# Computing the percentage of variation accounted for. Five clusters
```

```
(kmeans5_attr_std <- kmeans(attr_std,5,nstart = 10))
```

```
perc.var.5 <- round(100*(1 - kmeans5_attr_std$betweenss/kmeans5_attr_std$totss),1)
```

```
names(perc.var.5) <- "Perc. 5 clus"
```

```
perc.var.5
```

```
(kmeans6_attr_std <- kmeans(attr_std,6,nstart = 10))
```

```
# Computing the percentage of variation accounted for. Six clusters
```

```
perc.var.6 <- round(100*(1 - kmeans6_attr_std$betweenss/kmeans6_attr_std$totss),1)
```

```
names(perc.var.6) <- "Perc. 6 clus"
```

```
perc.var.6
```

```
#Factor Analysis
```

```
#parallel analysis suggest factor recommendation
```

```
parallel<-fa.parallel(attr_pca[,1:14],fm='minres',fa='fa')
```

```
#The gap between simulated data and actual data tends to be between 3 and 4
```

```
threefactor<-principal(attr_pca[,1:14],nfactors=3,rotate='varimax')
```

```
print(threefactor)
```

```
class(threefactor)
```

```
#Display factor values
```

```
threefactor$values
```

```
#Display factor loadings
```

```
threefactor$loadings
```

```
#communalities
```

```
threefactor$communality
```

```
#Rotated factor scores
```

```
head(threefactor$scores)
```

```
#round threefactor values
```

```
round(threefactor$values,3)
```

```
#Visualize the relationship and factor recommendations for simple structure
```

```
fa.diagram(threefactor)
```

```
colnames(threefactor$loadings)<- c("No.OfYears","PerformanceMetric","salaryMetric")
```

```
colnames(threefactor$loadings)
```

```
plot(threefactor)
```

```
#Multiple Regression
```

```
#install.packages("GGally")  
#install.packages("FFally")  
attach(attr)  
View(attr)  
attr[, c(2)] <- sapply(attr[, c(2)], as.numeric)
```

```
fit_attr<-  
lm(Attrition~Age+DailyRate+DistanceFromHome+HourlyRate+MonthlyIncome+MonthlyRate+NumComp  
aniesWorked+PercentSalaryHike+TotalWorkingYears+TrainingTimesLastYear+YearsAtCompany+YearsIn  
CurrentRole+YearsSinceLastPromotion+YearsWithCurrManager)  
fit_attr  
summary(fit_attr)  
coefficients(fit_attr)
```

```
#install.packages("GGally")  
#install.packages("FFally")  
library(GGally)  
confint(fit_attr,level=0.95)  
#Predicted Values  
fitted(fit_attr)  
residuals(fit_attr)  
#Anova table  
anova(fit_attr)  
vcov(fit_attr)  
temp<-influence.measures(fit_attr)  
temp  
View(temp)  
#Diagnostic Plot  
plot(fit_attr)
```

```
#Assessing Outliers
```

```
outlierTest(fit_attr)
```

```
qqPlot(fit_attr, main="QQ Plot")
```

```
# graphics.off()
```

```
# par(mfrow = c(1,2))
```

```
plot.new();
```

```
dev.off()
```

```
leveragePlots(fit_attr)
```

```
# Influential Observations
```

```
# added variable plots
```

```
avPlots(fit_attr)
```

```
# Normality of Residuals
```

```
# qq plot for studentized resid
```

```
qqPlot(fit_attr, main="QQ Plot")
```

```
# distribution of studentized residuals
```

```
library(MASS)
```

```
sresid <- studres(fit_attr)
```

```
hist(sresid, freq=FALSE,
```

```
    main="Distribution of Studentized Residuals")
```

```
xfit<-seq(min(sresid),max(sresid),length=40)
```

```
yfit<-dnorm(xfit)
```

```
lines(xfit, yfit)
```

```
#Non-constant Error Variance
```

```
# Evaluate homoscedasticity
```

```
# non-constant error variance test
```

```
ncvTest(fit_attr)
```

```
# plot studentized residuals vs. fitted values
```

```
spreadLevelPlot(fit_attr)
```

```
#Multi-collinearity
```

```
# Evaluate Collinearity
```

```
vif(fit_attr) # variance inflation factors
```

```
sqrt(vif(fit_attr)) > 2 # problem?
```

```
#Nonlinearity
```

```
# component + residual plot
```

```
crPlots(fit_attr)
```

```
# Ceres plots
```

```
#ceresPlots(fit_attr)
```

```
install.packages("gvlma")
```

```
library(gvlma)
```

```
gvmodel <- gvlma(fit_attr)
```

```
summary(gvmodel)
```

```
fit_attr
```

```
summary(fit_attr)
```

```
fit1<-fit_attr
```

```
fit2<-
```

```
lm(Attrition~Age+DailyRate+DistanceFromHome+MonthlyIncome+MonthlyRate+NumCompaniesWorked+PercentSalaryHike+TotalWorkingYears+TrainingTimesLastYear+YearsAtCompany+YearsInCurrentRole+YearsSinceLastPromotion+YearsWithCurrManager,data=attr)
```

```
summary(fit2)
```

```
fit3<-
```

```
lm(Attrition~Age+DailyRate+DistanceFromHome+MonthlyIncome+NumCompaniesWorked+PercentSalaryHike+TotalWorkingYears+TrainingTimesLastYear+YearsAtCompany+YearsInCurrentRole+YearsSinceLastPromotion+YearsWithCurrManager,data=attr)
```

```
summary(fit3)
```

```
fit4<-
```

```
lm(Attrition~Age+DailyRate+DistanceFromHome+MonthlyIncome+NumCompaniesWorked+TotalWorkingYears+TrainingTimesLastYear+YearsAtCompany+YearsInCurrentRole+YearsSinceLastPromotion+YearsWithCurrManager,data=attr)
```

```
summary(fit4)
```

```
fit5<-
```

```
lm(Attrition~Age+DailyRate+DistanceFromHome+MonthlyIncome+NumCompaniesWorked+TrainingTimesLastYear+YearsAtCompany+YearsInCurrentRole+YearsSinceLastPromotion+YearsWithCurrManager,data=attr)
```

```
summary(fit5)
```

```
fit6<-
```

```
lm(Attrition~Age+DailyRate+DistanceFromHome+MonthlyIncome+NumCompaniesWorked+TrainingTimesLastYear+YearsInCurrentRole+YearsSinceLastPromotion+YearsWithCurrManager,data=attr)
```

```
summary(fit6)
```

```
fit7<-
```

```
lm(Attrition~Age+DistanceFromHome+MonthlyIncome+NumCompaniesWorked+TrainingTimesLastYear+YearsInCurrentRole+YearsSinceLastPromotion+YearsWithCurrManager,data=attr)
```

```
summary(fit7)
```

```
fit8<-  
lm(Attrition~Age+DistanceFromHome+MonthlyIncome+NumCompaniesWorked+YearsInCurrentRole+YearsSinceLastPromotion+YearsWithCurrManager,data=attr)  
summary(fit8)
```

```
fit9<-  
lm(Attrition~Age+DistanceFromHome+MonthlyIncome+NumCompaniesWorked+YearsInCurrentRole+YearsSinceLastPromotion,data=attr)  
summary(fit9)
```

```
#Comparing model
```

```
anova(fit1,fit9)
```

```
step <- stepAIC(fit1, direction="both")
```

```
step$anova
```

```
attach(attr)
```

```
predict.lm(fit9, data.frame(Age=27,  
DistanceFromHome=10,MonthlyIncome=2000,NumCompaniesWorked=1,  
YearsInCurrentRole=3,YearsSinceLastPromotion=1))
```

