DDS Project 2

Tavin Weeda and Scott Frazier

12/10/2021

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.3     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```r
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

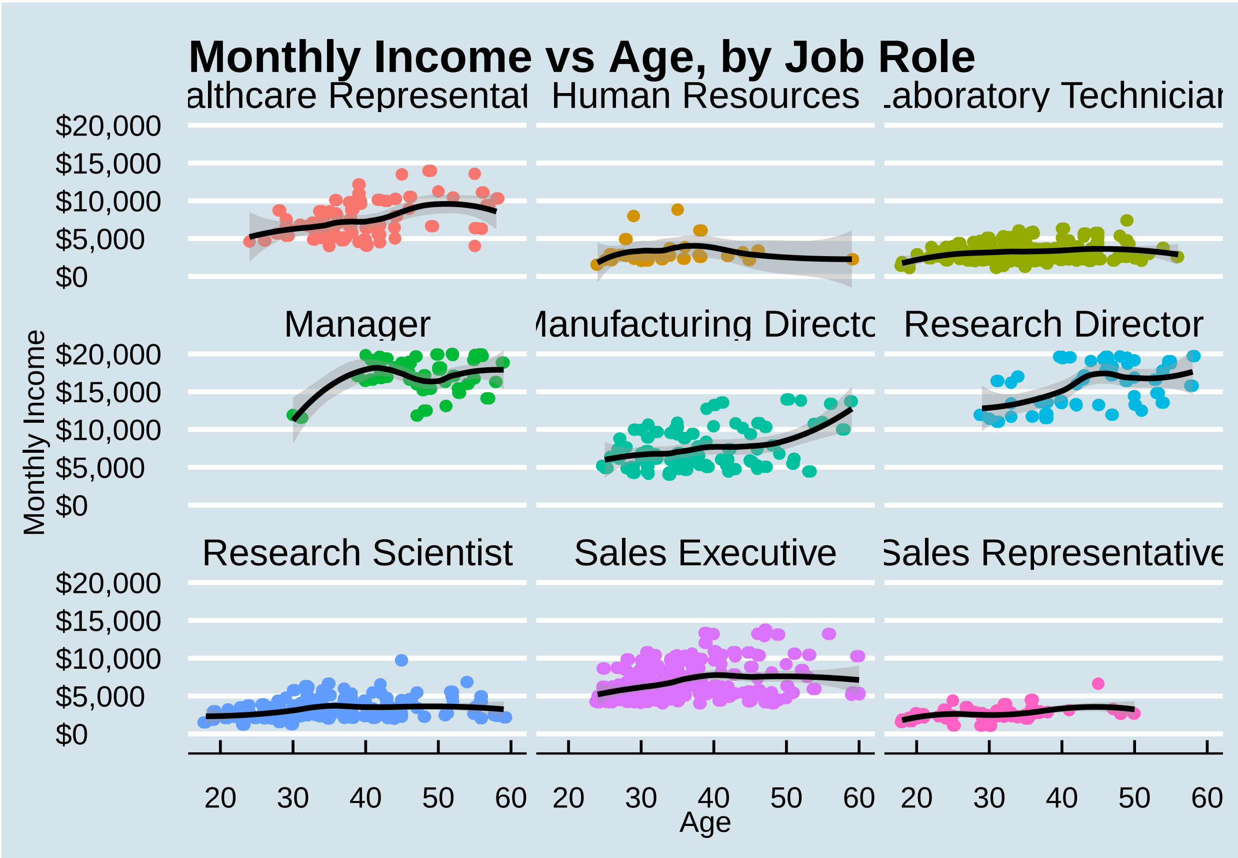
```
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(e1071)
library(class)
library(ggthemes)
set.seed(1234)
PREda <- read.csv("C:/Users/tavin/OneDrive/Desktop/DDS/Project 2/proj2main.csv")


##remove single level factors of ID, Employee Count, Over 18, Standard Hours, and Employee Number


PREda<-PREda[,c(-1,-10,-11,-23,-28)]
PREda$logIncome<-log(PREda$MonthlyIncome)
PREda<-PREda[,-18]

##Factor the categorical variables


PREda$Attrition<-as.factor(PREda$Attrition)
PREda$BusinessTravel<-as.factor(PREda$BusinessTravel)
PREda$Department<-as.factor(PREda$Department)
PREda$Education<-as.factor(PREda$Education)
PREda$EducationField<-as.factor(PREda$EducationField)
PREda$EnvironmentSatisfaction<-as.factor(PREda$EnvironmentSatisfaction)
PREda$Gender<-as.factor(PREda$Gender)
PREda$JobInvolvement<-as.factor(PREda$JobInvolvement)
PREda$JobLevel<-as.factor(PREda$JobLevel)
PREda$JobRole<-as.factor(PREda$JobRole)
PREda$JobSatisfaction<-as.factor(PREda$JobSatisfaction)
PREda$MaritalStatus<-as.factor(PREda$MaritalStatus)
PREda$OverTime<-as.factor(PREda$OverTime)
PREda$PerformanceRating<-as.factor(PREda$PerformanceRating)
PREda$RelationshipSatisfaction<-as.factor(PREda$RelationshipSatisfaction)
PREda$StockOptionLevel<-as.factor(PREda$StockOptionLevel)
PREda$TrainingTimesLastYear<-as.factor(PREda$TrainingTimesLastYear)
```

```r
PREda$WorkLifeBalance<-as.factor(PREda$WorkLifeBalance)
data<-PREda

##train/test split of 70/30 %.  the magic number 609 represents 70% of the data

index<-sample(1:dim(PREda)[1],609,replace=F)

train<-PREda[index,]
test<-PREda[-index,]

bayes.train<-train
bayes.test<-test
```

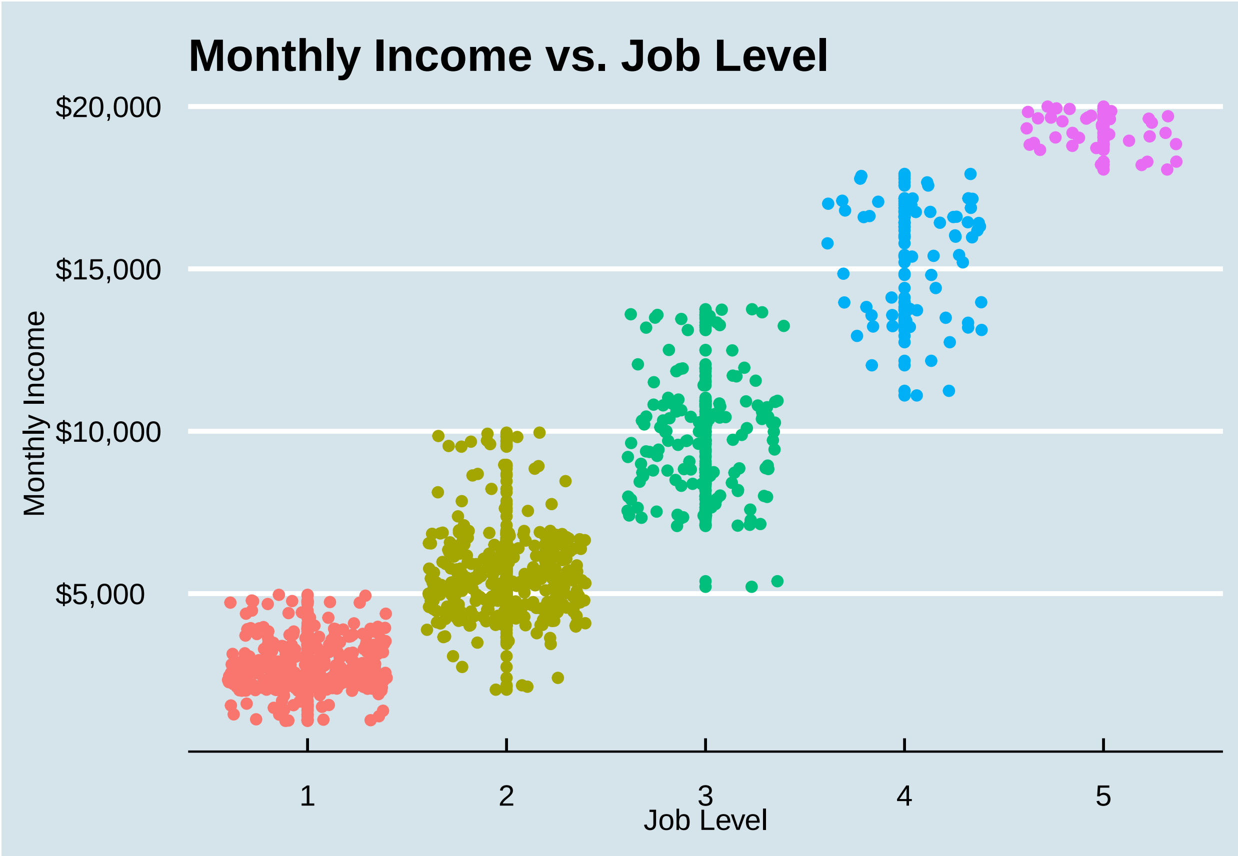
```r
##looking for interesting relationship

#Facet wrap of Age vs. MonthlyIncome, by Job Role
data %>% ggplot(aes(x=Age, y=MonthlyIncome, color=JobRole)) +
  geom_point() +
  geom_jitter() +
  geom_smooth(color = "black") +
  facet_wrap(~JobRole) + #facet wrap
  ggtitle("Monthly Income vs Age, by Job Role") +
  labs(y="Monthly Income") +
  scale_y_continuous(labels = scales::comma)+
  scale_y_continuous(labels=scales::dollar_format()) +
  theme_economist() +
  theme(legend.position = "None", axis.title.y=element_text(vjust=1.8))
```

```
## Scale for 'y' is already present. Adding another scale for 'y', which will
## replace the existing scale.

## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
## Monthly Income vs Job Level

#JobLevel vs MonthlyIncome
data %>% ggplot(aes(x=JobLevel, y=MonthlyIncome, color=JobLevel)) +
  geom_point() +
  geom_jitter() +
  ggtitle("Monthly Income vs. Job Level") +
  labs(y="Monthly Income", x="Job Level") +
  scale_y_continuous(labels = scales::comma)+
  scale_y_continuous(labels=scales::dollar_format()) +
  theme_economist() +
  theme(legend.position = "None", axis.title.y=element_text(vjust=1.8))
```

```
## Scale for 'y' is already present. Adding another scale for 'y', which will
## replace the existing scale.
```

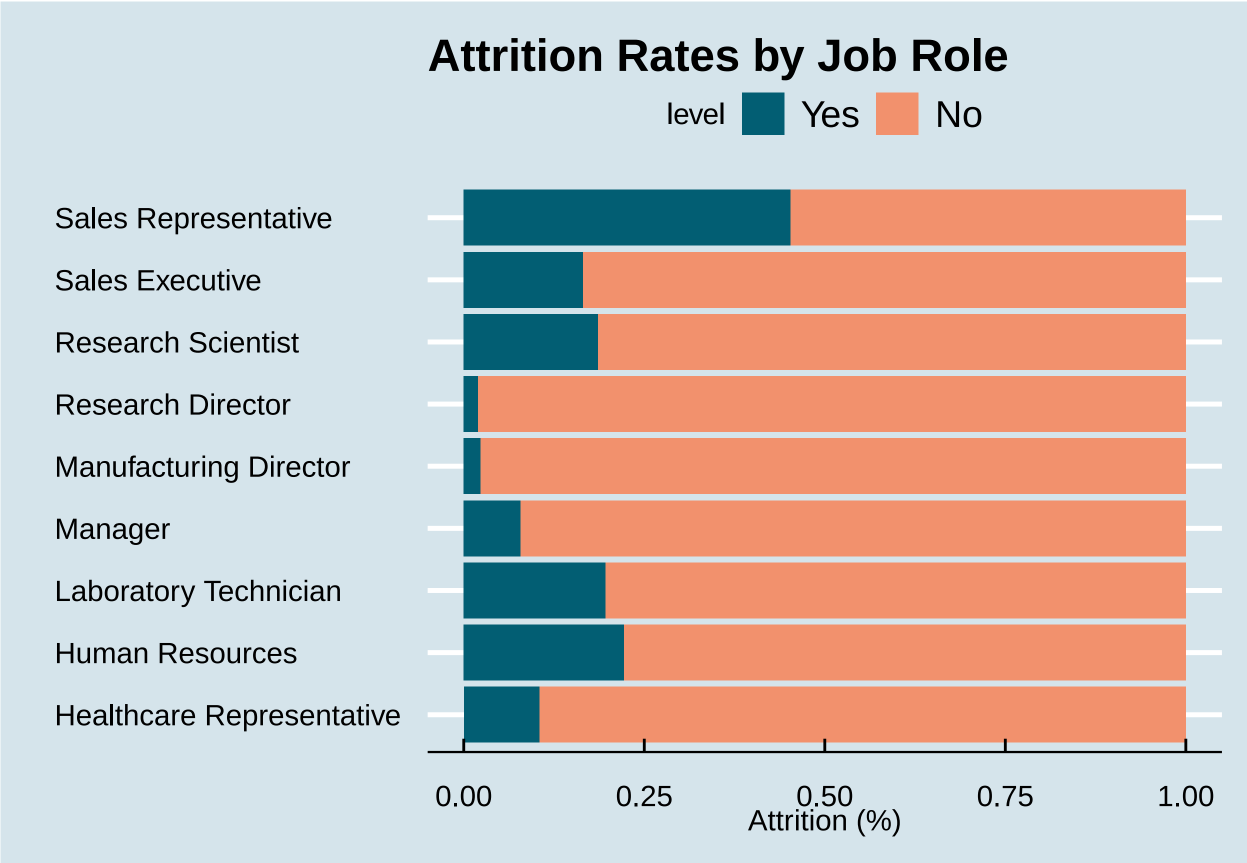



```
##Graphs for Attrition Slide

##Making dataframe for proportions of categorical variables

require(plyr)
```

```
## Loading required package: plyr

## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following object is masked from 'package:purrr':
##
##     compact
```

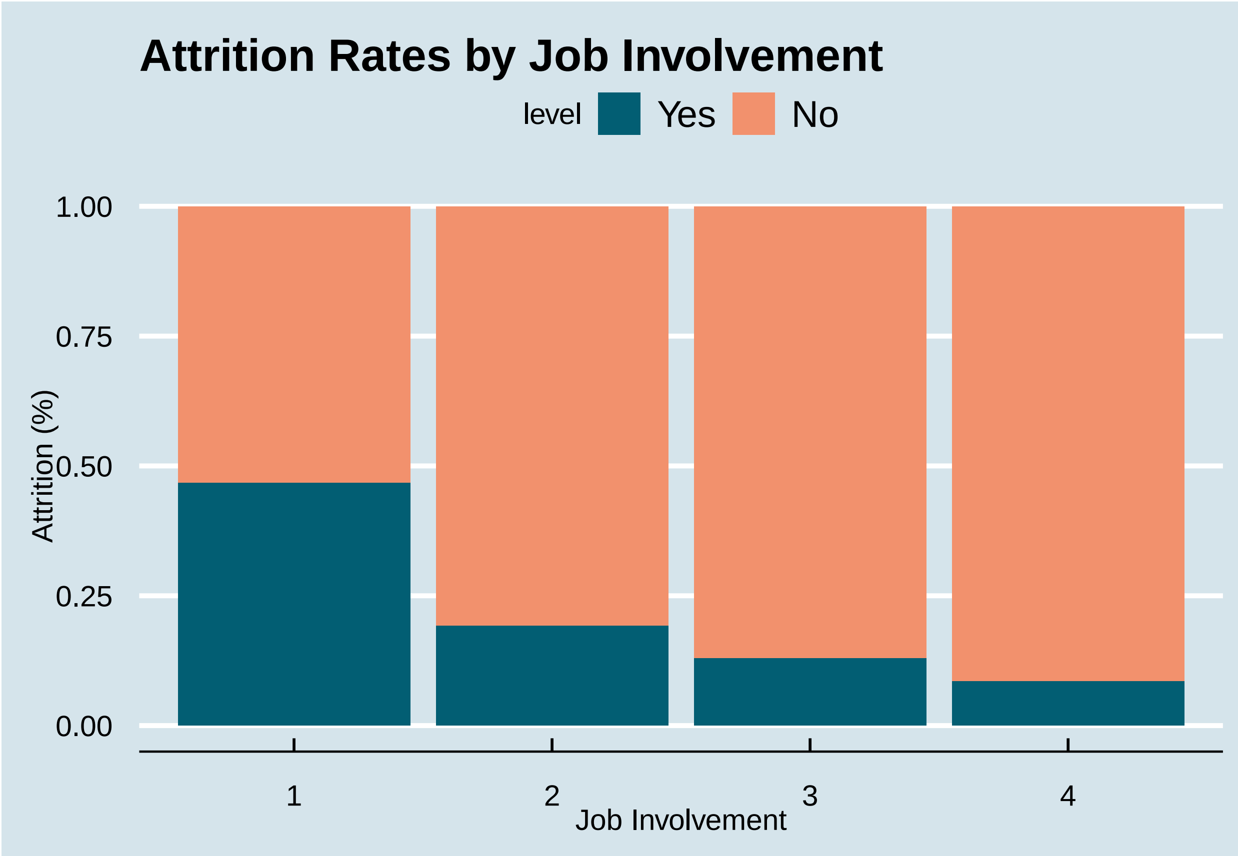
```r
##Job Role with proportion of Attrition


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


Education.Attrition.yes<-count(as.numeric(data$JobRole[data$Attrition=="Yes"]))
Education.Attrition.yes$level<-"Yes"
names(Education.Attrition.yes)<-c("Education","n","level")
Education.Attrition.yes$Education<-names(summary(data$JobRole))
Education.Attrition.no<-count(as.numeric(data$JobRole[data$Attrition=="No"]))
Education.Attrition.no$level<-"No"
names(Education.Attrition.no)<-c("Education","n","level")
Education.Attrition.no$Education<-names(summary(data$JobRole))
Education.Attrition<-rbind(Education.Attrition.yes,Education.Attrition.no)


  ##graph
Education.Attrition %>% ggplot(aes(x=as.factor(Education),y=n, fill=level)) +
  geom_col(position="fill") +
  scale_fill_manual(values = c("Yes" = "#025e73", "No"="#f2916d")) +
  ggtitle("Attrition Rates by Job Role")+xlab("")+ylab("Attrition (%)")+coord_flip()+theme_economist()
```



```r
##Job Involvement with proportion of Attrition
data$Education<-as.factor(data$Education)
Education.Attrition
```

```
##                    Education   n level
## 1  Healthcare Representative   8   Yes
## 2            Human Resources   6   Yes
## 3      Laboratory Technician  30   Yes
## 4                    Manager   4   Yes
## 5     Manufacturing Director   2   Yes
## 6          Research Director   1   Yes
## 7         Research Scientist  32   Yes
## 8            Sales Executive  33   Yes
## 9       Sales Representative  24   Yes
## 10 Healthcare Representative  68    No
## 11           Human Resources  21    No
## 12     Laboratory Technician 123    No
## 13                   Manager  47    No
## 14    Manufacturing Director  85    No
## 15         Research Director  50    No
## 16        Research Scientist 140    No
## 17           Sales Executive 167    No
## 18      Sales Representative  29    No
```

```r
Education.Attrition.yes<-count(as.numeric(data$JobInvolvement[data$Attrition=="Yes"]))
Education.Attrition.yes$level<-"Yes"
names(Education.Attrition.yes)<-c("Education","n","level")
Education.Attrition.no<-count(as.numeric(data$JobInvolvement[data$Attrition=="No"]))
Education.Attrition.no$level<-"No"
names(Education.Attrition.no)<-c("Education","n","level")
Education.Attrition<-rbind(Education.Attrition.yes,Education.Attrition.no)


##graph
  Education.Attrition %>% ggplot(aes(x=as.factor(Education),y=n, fill=level)) +
  geom_col(position="fill") +
  scale_fill_manual(values = c("Yes" = "#025e73", "No"="#f2916d")) +
  ggtitle("Attrition Rates by Job Involvement")+xlab("Job Involvement")+ylab("Attrition (%)")+theme_ecor
```

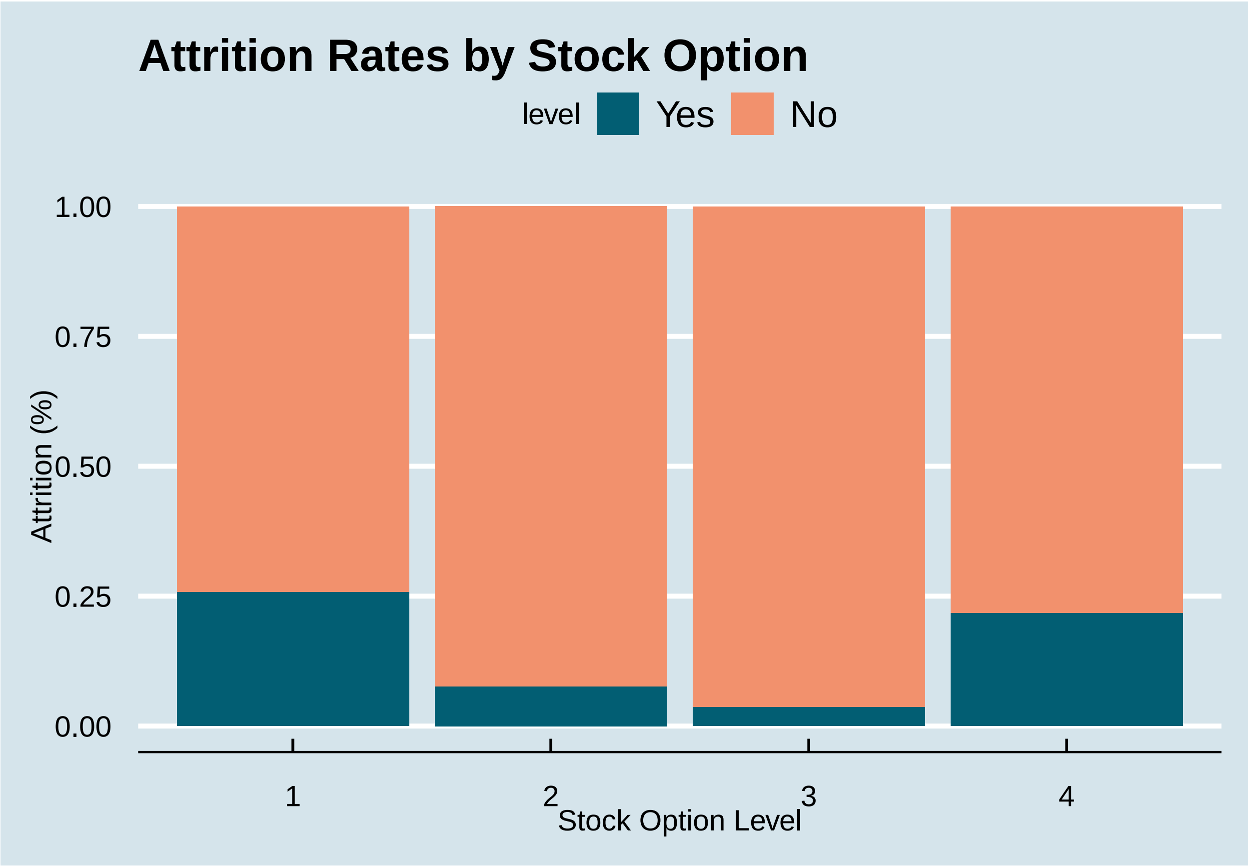



```
##Stock Option Level with proportion of Attrition
data$Education<-as.factor(data$Education)
Education.Attrition
```

```
##   Education   n level
## 1         1  22   Yes
## 2         2  44   Yes
## 3         3  67   Yes
## 4         4   7   Yes
## 5         1  25    No
## 6         2 184    No
## 7         3 447    No
## 8         4  74    No
```

```
Education.Attrition.yes<-count(as.numeric(data$StockOptionLevel[data$Attrition=="Yes"]))
Education.Attrition.yes$level<-"Yes"
names(Education.Attrition.yes)<-c("Education","n","level")
Education.Attrition.no<-count(as.numeric(data$StockOptionLevel[data$Attrition=="No"]))
Education.Attrition.no$level<-"No"
names(Education.Attrition.no)<-c("Education","n","level")
Education.Attrition<-rbind(Education.Attrition.yes,Education.Attrition.no)


##graph
  Education.Attrition %>% ggplot(aes(x=as.factor(Education),y=n, fill=level)) +
```

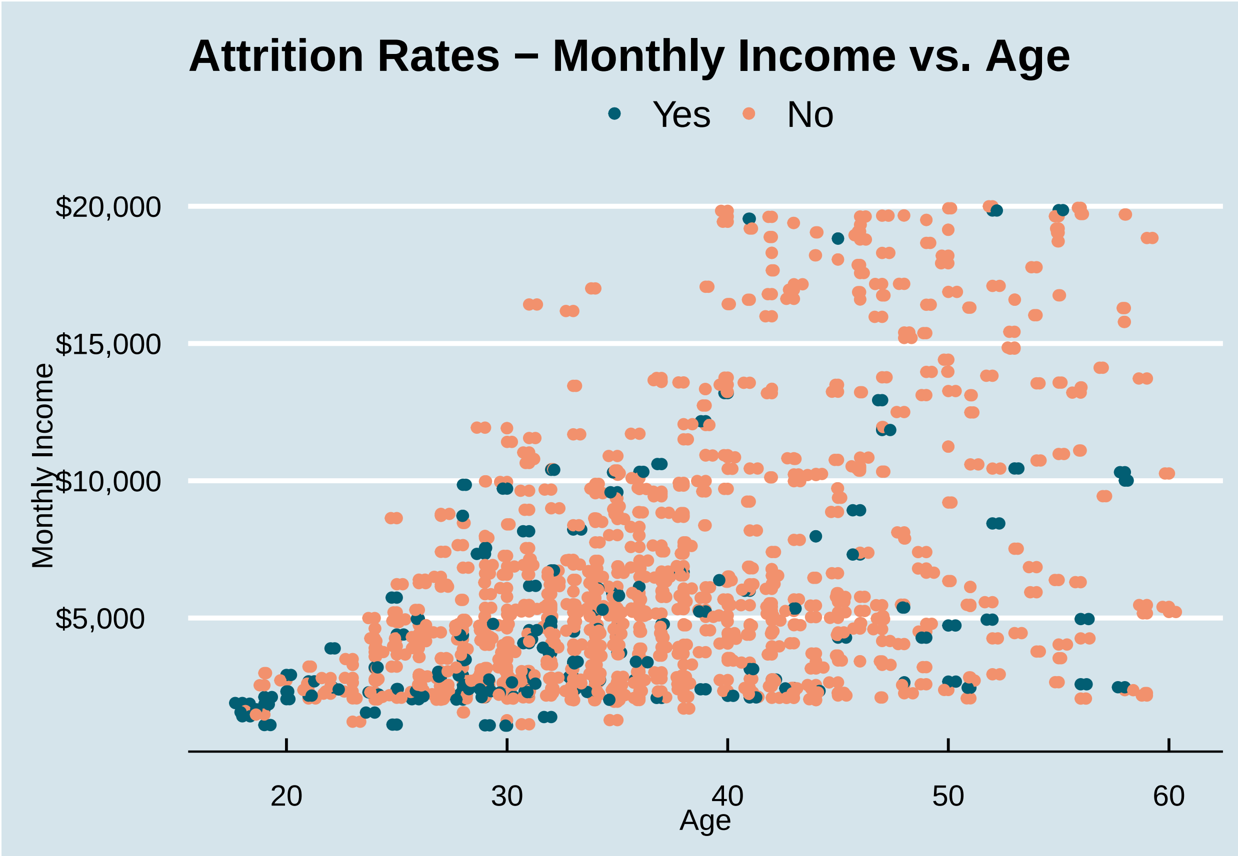
```
  geom_col(position="fill") +
  scale_fill_manual(values = c("Yes" = "#025e73", "No"="#f2916d")) +
  ggtitle("Attrition Rates by Stock Option")+xlab("Stock Option Level")+ylab("Attrition (%)")+theme_ecor
```




```
##override plyr
library(tidyverse)
```

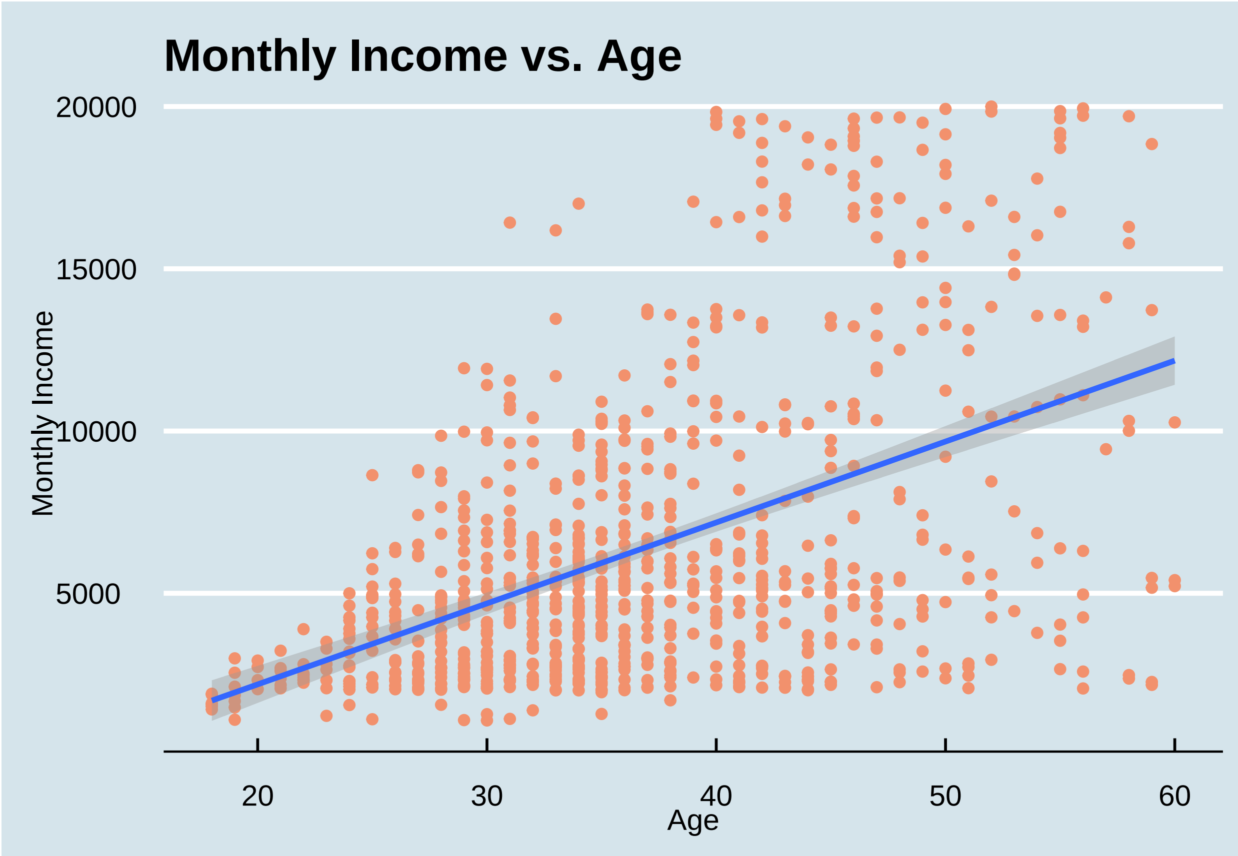
```
##showing an example of why KNN will most likely not perform well
##the data between yes and no for attrition appears to be randomly scattered and not any definite bound
data %>% ggplot(aes(x=Age, y=MonthlyIncome, color=Attrition)) +
  geom_point() +
  geom_jitter() +
  scale_color_manual(values = c("Yes" = "#025e73", "No"="#f2916d")) +
  ggtitle("Attrition Rates - Monthly Income vs. Age") +
  scale_y_continuous(labels = scales::comma) +
  scale_y_continuous(labels=scales::dollar_format()) +
  labs(y="Monthly Income") +
  theme_economist() +
  theme(legend.title = element_blank())
```

```
## Scale for 'y' is already present. Adding another scale for 'y', which will
## replace the existing scale.
```



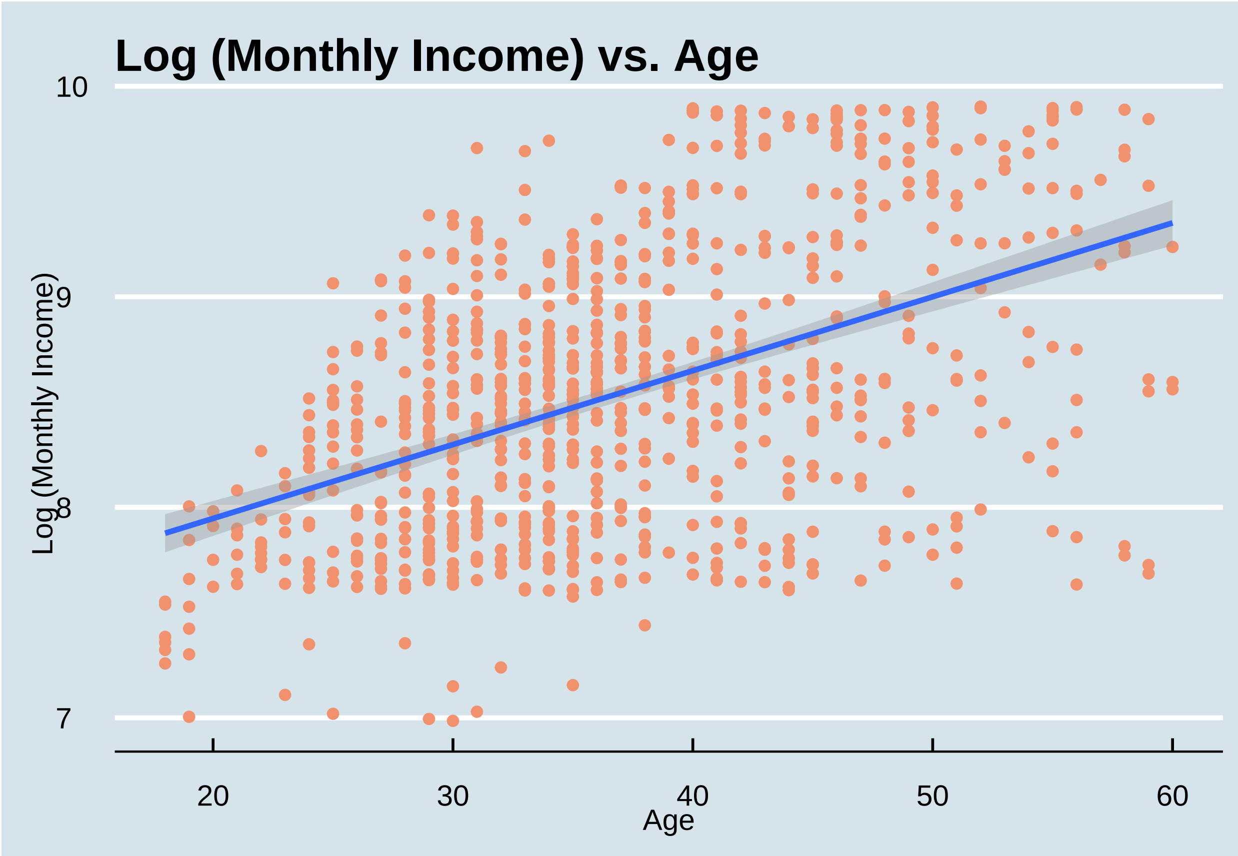

```
##Linear Regression showing non constant variance until log transformed

##just regular monthly income
data %>% ggplot(aes(x=Age, y=MonthlyIncome)) +
  geom_point(color="#f2916d") +
  ggtitle("Monthly Income vs. Age")+theme_economist()+geom_smooth(method="lm")+ylab("Monthly Income")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



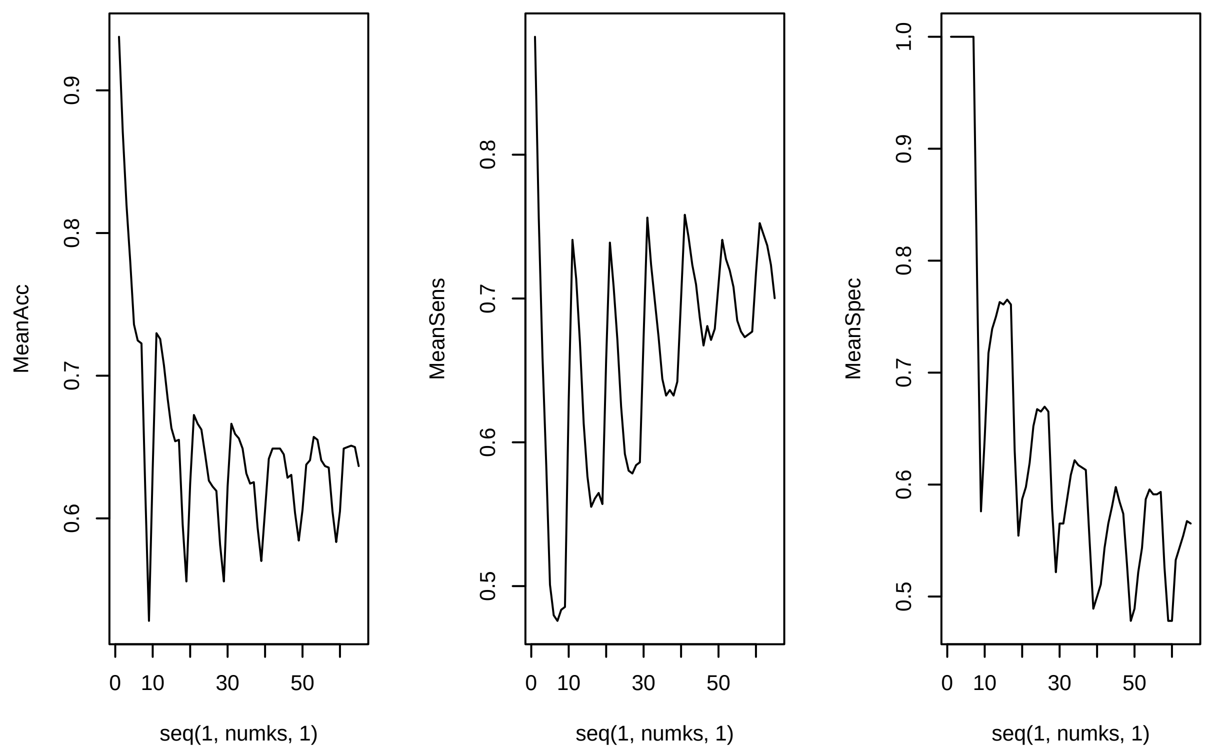

```
##log transformed Monthly income

data %>% ggplot(aes(x=Age, y=log(MonthlyIncome))) +
  geom_point(color="#f2916d") +
  ggtitle("Log (Monthly Income) vs. Age")+theme_economist()+geom_smooth(method="lm")+ylab("Log (Monthly
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
##KNN


##oversample minority group
x.1<-train[train$Attrition=="Yes",]
train.over<-train
train.over<-rbind(train.over,x.1)
train.over<-rbind(train.over,x.1)
train.over<-rbind(train.over,x.1)
train.over<-rbind(train.over,x.1)

##remove categorical predictors

  train.over<-train.over[,c(1,2,4,6,11,17,20,24,27,28,29,30,31)]

  test.cont<-test[,c(1,2,4,6,11,17,20,24,27,28,29,30,31)]

##scaling train

tempatt<-train.over[,2]

temp.2<-scale(train.over[,-2])
temp.2<-as.data.frame(temp.2)
temp.2$Attrition<-train.over[,2]

train.over<-temp.2
```

```r
    ##scaling test

temptest<-test.cont[,2]
temp.3<-scale(test.cont[,-2])
temp.3<-as.data.frame(temp.3)
temp.3$Attrition<-test.cont[,2]
test.cont<-temp.3

##tune k

iterations = 1
numks = 65
masterAcc = matrix(nrow = iterations, ncol = numks)
masterSens = matrix(nrow = iterations, ncol = numks)
masterSpec = matrix(nrow = iterations, ncol = numks)
for(j in 1:iterations)
{
  accs = data.frame(accuracy = numeric(30), k = numeric(30))

  for(i in 1:numks)
  {
    classifications = knn.cv(train.over[,-13],train.over$Attrition, prob = TRUE, k = i)
    table(classifications,train.over$Attrition)
    CM = confusionMatrix(table(classifications,train.over$Attrition))
    masterAcc[j,i] = CM$overall[1]
    masterSens[j,i]=CM$byClass[1]
    masterSpec[j,i]=CM$byClass[2]
  }
}
MeanAcc = colMeans(masterAcc)
MeanSens=colMeans(masterSens)
MeanSpec = colMeans(masterSpec)
par(mfrow=c(1,3))
plot(seq(1,numks,1),MeanAcc, type = "l")
plot(seq(1,numks,1),MeanSens, type = "l")
plot(seq(1,numks,1),MeanSpec, type = "l")
```




```r
##test data for KNN

classifications = knn(train.over[,-13],test.cont[,-13],train.over$Attrition, prob = TRUE, k = 3)
table(classifications,test.cont$Attrition)
```

```
##
## classifications  No Yes
##             No  144  24
##             Yes  69  24
```

```r
confusionMatrix(table(classifications,test.cont$Attrition))
```

```
## Confusion Matrix and Statistics
##
##
## classifications  No Yes
##             No  144  24
##             Yes  69  24
##
##                Accuracy : 0.6437
##                  95% CI : (0.5823, 0.7018)
##     No Information Rate : 0.8161
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1292
```

```
##
##  Mcnemar's Test P-Value : 5.053e-06
##
##             Sensitivity : 0.6761
##             Specificity : 0.5000
##          Pos Pred Value : 0.8571
##          Neg Pred Value : 0.2581
##              Prevalence : 0.8161
##          Detection Rate : 0.5517
##    Detection Prevalence : 0.6437
##       Balanced Accuracy : 0.5880
##
##        'Positive' Class : No
##
```

```
##remove monthly income for linear regression model

train<-train[,-17]

##Stepwise variable selection using AIC
fit.lm<-lm(logIncome~.,data=train)
step.lm<-fit.lm%>%stepAIC(trace=FALSE)
step.lm
```

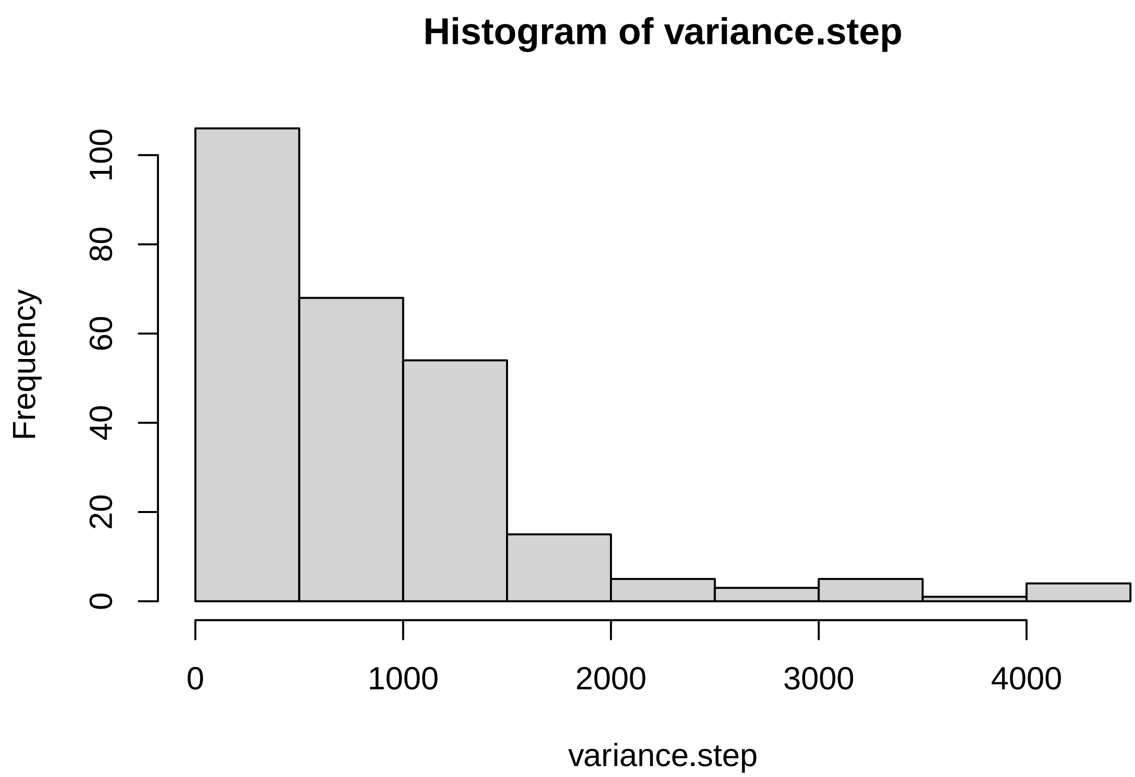
```
##
## Call:
## lm(formula = logIncome ~ Age + BusinessTravel + DailyRate + JobLevel +
##     JobRole + TotalWorkingYears + YearsInCurrentRole, data = train)
##
## Coefficients:
##                     (Intercept)                               Age
##                       7.868e+00                         1.871e-03
## BusinessTravelTravel_Frequently      BusinessTravelTravel_Rarely
##                       3.453e-02                         8.544e-02
##                       DailyRate                         JobLevel2
##                       5.222e-05                         5.159e-01
##                       JobLevel3                         JobLevel4
##                       9.778e-01                         1.209e+00
##                       JobLevel5          JobRoleHuman Resources
##                       1.324e+00                        -1.268e-01
##    JobRoleLaboratory Technician                    JobRoleManager
##                      -2.150e-01                         2.898e-01
##   JobRoleManufacturing Director          JobRoleResearch Director
##                       2.710e-02                         3.141e-01
##       JobRoleResearch Scientist            JobRoleSales Executive
##                      -1.793e-01                        -2.515e-02
##     JobRoleSales Representative                 TotalWorkingYears
##                      -2.430e-01                         5.165e-03
##              YearsInCurrentRole
##                       4.018e-03
```

```
fit.pred.step<-predict(step.lm,newdata=test,type="response")
```

```r
## RMSE calculation
RMSE<-mean((exp(test$logIncome)-exp(fit.pred.step))^2)%>%sqrt()
RMSE
```

```
## [1] 1158.702
```

```r
##variance of RMSE's
variance.step<-(exp(test$logIncome)-exp(fit.pred.step))^2%>%sqrt()
hist(variance.step)
```




```r
##vif

##Job level is high, but this is for prediction so we will go ahead with it.
vif(step.lm)
```

```
##                          GVIF Df GVIF^(1/(2*Df))
## Age                  1.927680  1        1.388409
## BusinessTravel       1.059714  2        1.014606
## DailyRate            1.023513  1        1.011688
## JobLevel            18.484918  4        1.439966
## JobRole             13.018557  8        1.173979
## TotalWorkingYears    4.516421  1        2.125187
## YearsInCurrentRole   1.464002  1        1.209959
```

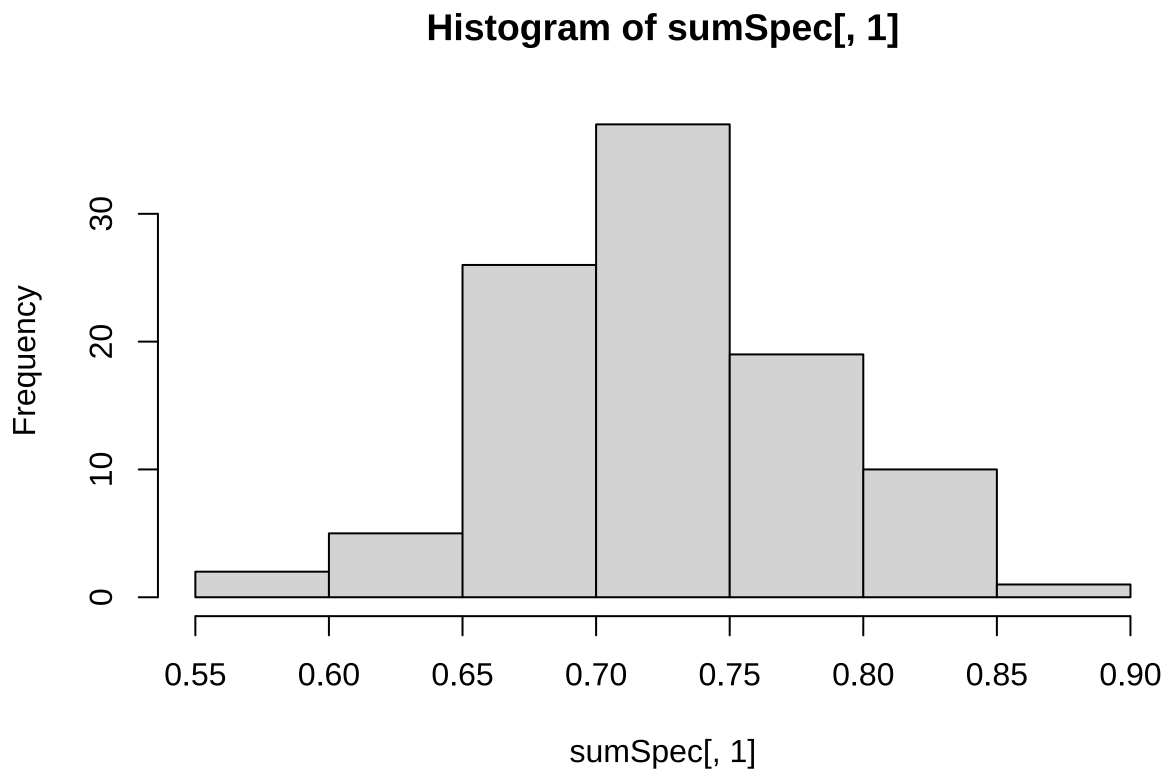
```r
##Naive Bayes
##remove highly correlated variables
sumSpec<-data.frame(Sens=c())
sumSens<-data.frame(Sens=c())

##Loops through 100 times with different train/test splits to get average sensitivity and specificity
for(x in 1:100){
index<-sample(1:dim(PREda)[1],609,replace=F)
train<-PREda[index,]
test<-PREda[-index,]

##these variables were removed in a forward-wise selection
##if a deleted variable had a noticeable change in the test metrics
##it was removed from the data set.
train<-train[,-c(29,5,4,8,21,22)]
test<-test[,-c(29,5,4,8,21,22)]
x.1<-train[train$Attrition=="Yes",]
train.over<-train
train.over<-rbind(train.over,x.1)
train.over<-rbind(train.over,x.1)
train.over<-rbind(train.over,x.1)
train.over<-rbind(train.over,x.1)
#train.over<-rbind(train.over,x.1)
#train<-train[,-c(5,4,8,18,22)]
#test<-test[,-c(5,4,8,18,22)]
#train<-train[,-c(5,14)]
#test<-test[,-c(5,14)]
model = naiveBayes(Attrition~.,data = train.over)
confusionMatrix(table(predict(model,test[,-2]),test$Attrition))
sumSpec<-rbind(sumSpec,confusionMatrix(table(predict(model,test[,-2]),test$Attrition))$byClass[2])
sumSens<-rbind(sumSens,confusionMatrix(table(predict(model,test[,-2]),test$Attrition))$byClass[1])
}

##mean of 100 iterations for sensitivity and specificity
mean(sumSpec[,1])
```

```
## [1] 0.726489
```

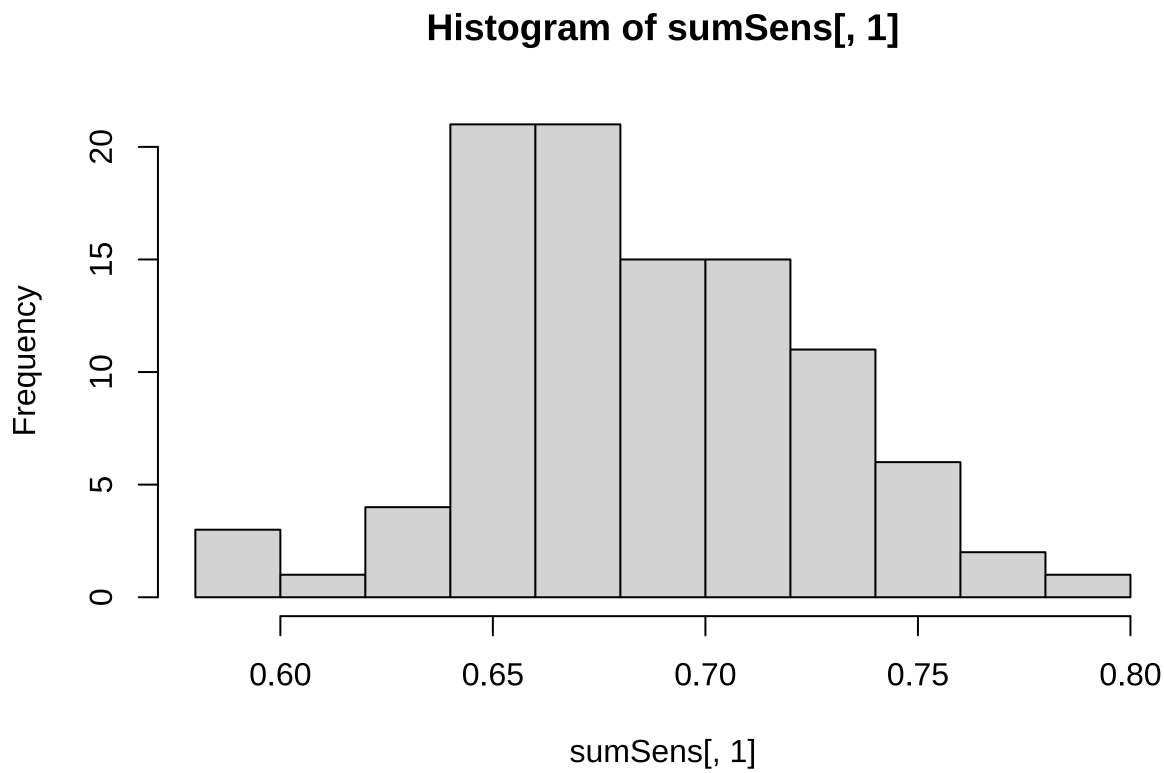
```r
mean(sumSens[,1])
```

```
## [1] 0.6842248
```

```r
hist(sumSpec[,1])
```




```
hist(sumSens[,1])
```




```r
##This gives the times out of 100 that the sensitivity was below .6 threshold (TRUE)
summary(sumSpec[1]<=.6)
```

```
##  X0.685714285714286
##  Mode :logical
##  FALSE:98
##  TRUE :2
```

```r
summary(sumSens[1]<=.6)
```

```
##  X0.650442477876106
##  Mode :logical
##  FALSE:97
##  TRUE :3
```

```r
##Naive bayes model
##This uses the original train test split (it was altered for the other models)
train<-bayes.train[,-c(29,5,4,8,21,22)]
test<-bayes.test[,-c(29,5,4,8,21,22)]

##This over samples the minority "Yes" class
x.1<-train[train$Attrition=="Yes",]
train.over<-train
train.over<-rbind(train.over,x.1)
```

```r
train.over<-rbind(train.over,x.1)
train.over<-rbind(train.over,x.1)
train.over<-rbind(train.over,x.1)

##This is the test set
model = naiveBayes(Attrition~.,data = train.over)
confusionMatrix(table(predict(model,test[,-2]),test$Attrition))
```

```
## Confusion Matrix and Statistics
##
##
##        No Yes
##   No  158  11
##   Yes  55  37
##
##                Accuracy : 0.7471
##                  95% CI : (0.6898, 0.7987)
##     No Information Rate : 0.8161
##     P-Value [Acc > NIR] : 0.9978
##
##                   Kappa : 0.3783
##
##  Mcnemar's Test P-Value : 1.204e-07
##
##             Sensitivity : 0.7418
##             Specificity : 0.7708
##          Pos Pred Value : 0.9349
##          Neg Pred Value : 0.4022
##              Prevalence : 0.8161
##          Detection Rate : 0.6054
##    Detection Prevalence : 0.6475
##       Balanced Accuracy : 0.7563
##
##        'Positive' Class : No
##
```