STAT 4198 Final Project

# STAT 4198 Data Mining

# Final Project

Jinwen Zhang Spring 2018

## Dataset input

data <- read.csv(file = "D:/file/STAT4198 Final/fordTrain.csv",stringsAsFactors=TRUE)  
indexes <- read.csv(file = "D:/file/STAT4198 Final/indexes.csv",header=FALSE, stringsAsFactors=TRUE)  
# G49803308  
last.two.digits<- 08  
k<-last.two.digits+1

newdata<-data.frame(data[data[,1]%in%indexes[k,], ])  
summary(newdata)

## TrialID ObsNum IsAlert P1   
## Min. : 0.0 Min. : 0.0 Min. :0.000 Min. : 22.35   
## 1st Qu.: 99.0 1st Qu.: 302.0 1st Qu.:0.000 1st Qu.: 31.59   
## Median :274.0 Median : 604.0 Median :1.000 Median : 34.13   
## Mean :245.3 Mean : 603.9 Mean :0.575 Mean : 35.22   
## 3rd Qu.:357.0 3rd Qu.: 906.0 3rd Qu.:1.000 3rd Qu.: 37.14   
## Max. :505.0 Max. :1210.0 Max. :1.000 Max. :100.15   
## P2 P3 P4 P5   
## Min. :-23.239 Min. : 504 Min. : 24.83 Min. : 0.04800   
## 1st Qu.: 9.988 1st Qu.: 788 1st Qu.: 50.00 1st Qu.: 0.09081   
## Median : 11.479 Median :1000 Median : 60.00 Median : 0.10379   
## Mean : 12.035 Mean :1017 Mean : 64.45 Mean : 0.21511   
## 3rd Qu.: 13.694 3rd Qu.:1200 3rd Qu.: 76.14 3rd Qu.: 0.14011   
## Max. : 29.082 Max. :2416 Max. :119.05 Max. :27.20220   
## P6 P7 P8 E1   
## Min. : 140.0 Min. : 22.42 Min. :0 Min. : 0.00   
## 1st Qu.: 648.0 1st Qu.: 68.49 1st Qu.:0 1st Qu.: 0.00   
## Median : 768.0 Median : 78.12 Median :0 Median : 0.00   
## Mean : 789.3 Mean : 80.59 Mean :0 Mean :10.95   
## 3rd Qu.: 876.0 3rd Qu.: 92.59 3rd Qu.:0 3rd Qu.:28.63   
## Max. :2676.0 Max. :428.57 Max. :0 Max. :38.62   
## E2 E3 E4 E5   
## Min. : 0.00 Min. :0.000 Min. :-250.000 Min. :0.00800   
## 1st Qu.: 0.00 1st Qu.:0.000 1st Qu.: -8.000 1st Qu.:0.01562   
## Median : 0.00 Median :0.000 Median : 0.000 Median :0.01600   
## Mean : 95.91 Mean :0.336 Mean : -2.336 Mean :0.01629   
## 3rd Qu.:196.63 3rd Qu.:0.000 3rd Qu.: 6.000 3rd Qu.:0.01688   
## Max. :359.82 Max. :4.000 Max. : 250.000 Max. :0.02394   
## E6 E7 E8 E9   
## Min. :260.0 Min. : 0.000 Min. :0.000 Min. :0.0000   
## 1st Qu.:350.0 1st Qu.: 0.000 1st Qu.:0.000 1st Qu.:1.0000   
## Median :366.0 Median : 1.000 Median :1.000 Median :1.0000   
## Mean :359.1 Mean : 1.503 Mean :1.252 Mean :0.8941   
## 3rd Qu.:367.0 3rd Qu.: 2.000 3rd Qu.:2.000 3rd Qu.:1.0000   
## Max. :513.0 Max. :20.000 Max. :9.000 Max. :1.0000   
## E10 E11 V1 V2   
## Min. : 0.00 Min. : 0.000 Min. : 0.00 Min. :-4.20000   
## 1st Qu.: 44.00 1st Qu.: 0.000 1st Qu.: 23.64 1st Qu.:-0.07000   
## Median : 65.00 Median : 0.000 Median : 99.88 Median : 0.00000   
## Mean : 60.16 Mean : 1.293 Mean : 75.73 Mean :-0.02632   
## 3rd Qu.: 72.00 3rd Qu.: 0.000 3rd Qu.:108.13 3rd Qu.: 0.07000   
## Max. :117.00 Max. :32.800 Max. :123.88 Max. : 2.83500   
## V3 V4 V5 V6   
## Min. : 240 Min. : 0.000 Min. :0.0000 Min. : 531   
## 1st Qu.: 255 1st Qu.: 1.488 1st Qu.:0.0000 1st Qu.:1068   
## Median : 497 Median : 3.019 Median :0.0000 Median :1982   
## Mean : 568 Mean : 34.079 Mean :0.1247 Mean :1708   
## 3rd Qu.: 767 3rd Qu.: 9.012 3rd Qu.:0.0000 3rd Qu.:2137   
## Max. :1023 Max. :479.981 Max. :1.0000 Max. :3466   
## V7 V8 V9 V10 V11   
## Min. :0 Min. : 0.00 Min. :0 Min. :1.000 Min. : 3.506   
## 1st Qu.:0 1st Qu.: 0.00 1st Qu.:0 1st Qu.:2.000 1st Qu.: 7.498   
## Median :0 Median :12.10 Median :0 Median :4.000 Median :10.796   
## Mean :0 Mean :12.18 Mean :0 Mean :3.276 Mean :11.179   
## 3rd Qu.:0 3rd Qu.:21.00 3rd Qu.:0 3rd Qu.:4.000 3rd Qu.:15.232   
## Max. :0 Max. :62.10 Max. :0 Max. :7.000 Max. :19.711

sum(is.na(newdata))

## [1] 0

The entire data has no missing value

## Exploratory Data Analysis

#### 1.Remove features which do not change their values

delete: P8, V7, V9

drops <- c("P8","V7","V9")  
newdata<-newdata[ , !(names(newdata) %in% drops)]

#### 2.For each feature

Due to the lack of dataset decription, I can only determine whether a variable is continuous of categorical by checking how many levels the variable has. For example, if a variable is continuous, then it could have as many levels as the sample size. Otherwise, if a variable is categorical, then it could have much fewer numbers of levels compared to the sample size.

TrialID: From the dataset description, variable “TrialID” is a code for a subject, which means such variable is categorical. And the specific levels for vairable “TrialID” are listing as following:

TrialID <- as.factor(newdata$TrialID)  
levels(TrialID)

## [1] "0" "8" "17" "29" "35" "36" "44" "60" "63" "78" "83"   
## [12] "91" "99" "100" "146" "155" "170" "180" "208" "225" "235" "244"  
## [23] "256" "258" "268" "274" "290" "291" "296" "308" "315" "320" "322"  
## [34] "324" "325" "338" "352" "357" "359" "392" "395" "405" "411" "416"  
## [45] "422" "425" "438" "450" "452" "505"

ObsNum: Based on the dataset description, variable “ObsNum” is the number of new sample. Therefore, this variable has no influence on the analysis. Range from 0 to 1211.

IsAlert: Based in the dataset description, this variable is the response variable. This variable only concludes two levels – 1 for having alert, and 0 for not having alert

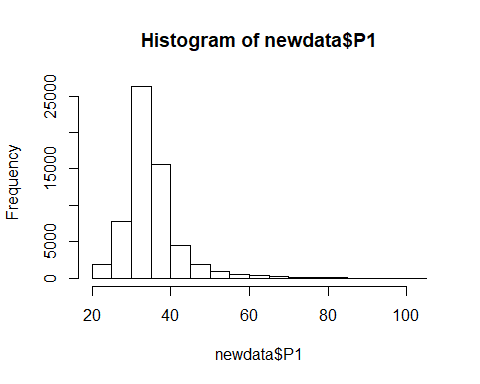
table(newdata$IsAlert)

##   
## 0 1   
## 25686 34756

There are 25686 observations have no alert, and 34756 onservations have alert.

Physical data: P1: Variable “P1” is continuous, and has symmetry distribution.

#P1 <- as.factor(newdata$P1)  
#levels(P1)  
hist(newdata$P1)

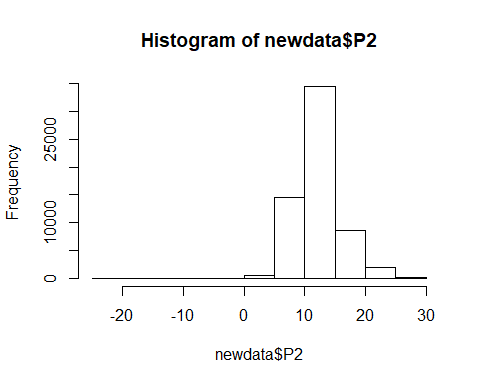


summary(newdata$P1)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 22.35 31.59 34.13 35.22 37.14 100.15

P2: Variable “P2” is continuous, and has symmetry distribution.

#P2 <- as.factor(newdata$P2)  
#levels(P2)  
hist(newdata$P2)

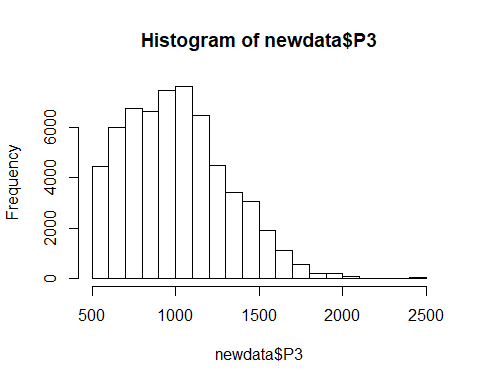


summary(newdata$P2)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -23.239 9.988 11.479 12.035 13.694 29.082

P3: Variable “P3” is continuous, and has symmetry distribution.

#P3 <- as.factor(newdata$P3)  
#levels(P3)  
hist(newdata$P3)

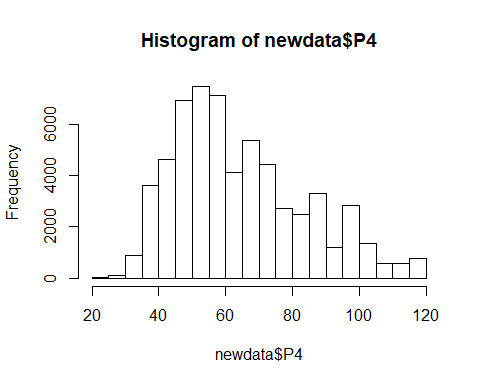


#barplot(table(newdata$P3),xlab= 'P3')  
summary(newdata$P3)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 504 788 1000 1017 1200 2416

P4: Variable “P4” is continuous, and has symmetry distribution.

#P4 <- as.factor(newdata$P4)  
#levels(P4)  
hist(newdata$P4)

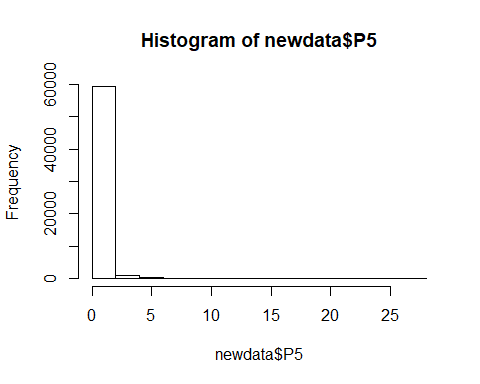


#barplot(table(newdata$P4),xlab= 'P4')  
summary(newdata$P4)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 24.83 50.00 60.00 64.45 76.14 119.05

P5: Variable “P5” is continuous, and has right-skewed distribution.

#P5 <- as.factor(newdata$P5)  
#levels(P5)  
hist(newdata$P5)

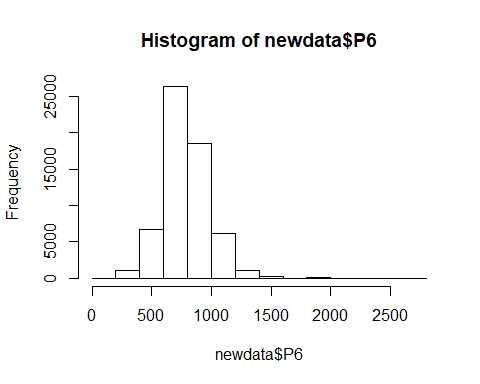


#barplot(table(newdata$P5),xlab= 'P5')  
summary(newdata$P5)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.04800 0.09081 0.10379 0.21511 0.14011 27.20220

P6: Variable “P6” is continuous, and has symmetry distribution.

#P6 <- as.factor(newdata$P6)  
#levels(P6)  
hist(newdata$P6)

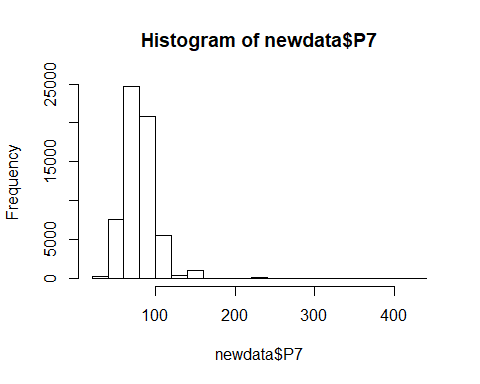


#barplot(table(newdata$P6),xlab= 'P6')  
summary(newdata$P6)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 140.0 648.0 768.0 789.3 876.0 2676.0

P7: Variable “P7” is continuous, and has symmetry distribution.

#P7 <- as.factor(newdata$P7)  
#levels(P7)  
hist(newdata$P7)

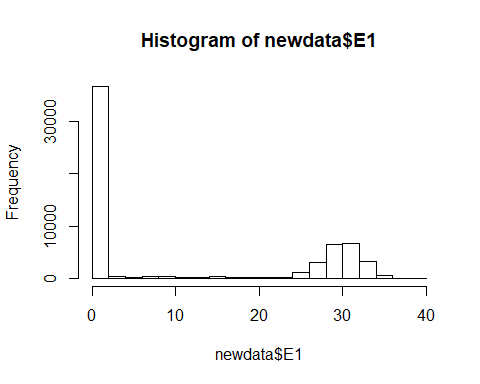


#barplot(table(newdata$P7),xlab= 'P7')  
summary(newdata$P7)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 22.42 68.49 78.12 80.59 92.59 428.57

E1: Variable “E1” is continuous, and has multimodality distribution.

#E1 <- as.factor(newdata$E1)  
#levels(E1)  
hist(newdata$E1)

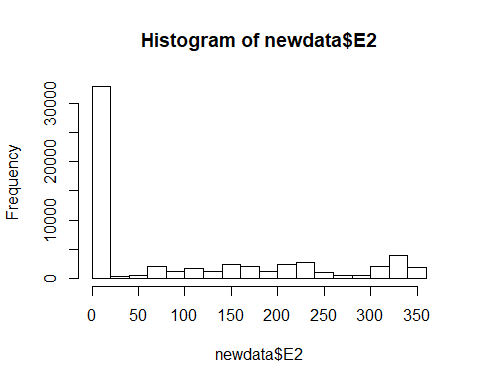


#barplot(table(newdata$E1),xlab= 'E1')  
summary(newdata$E1)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00 0.00 0.00 10.95 28.63 38.62

E2: Variable “E2” is continuous, and has multimodality distribution.

#E2 <- as.factor(newdata$E2)  
#levels(E2)  
hist(newdata$E2)



#barplot(table(newdata$E2),xlab= 'E2')  
summary(newdata$E2)

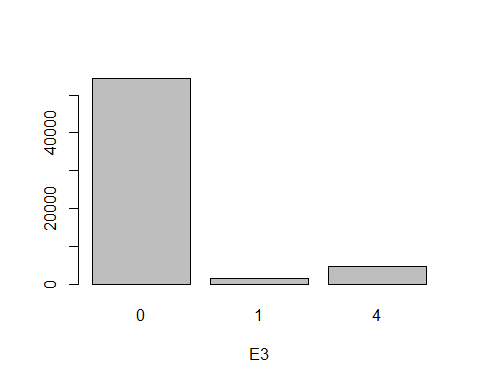
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00 0.00 0.00 95.91 196.63 359.82

E3: Variable “E3” is categorical, with 3 levels.

E3 <- as.factor(newdata$E3)  
levels(E3)

## [1] "0" "1" "4"

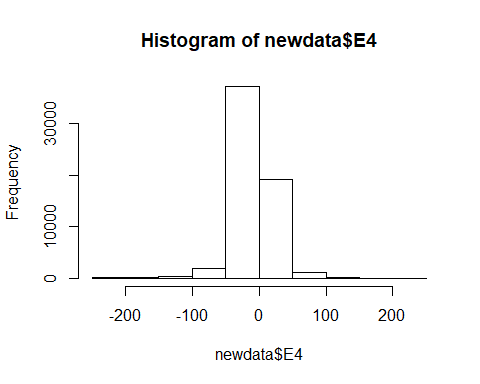
#hist(newdata$E3)  
barplot(table(newdata$E3),xlab= 'E3')



#summary(newdata$E3)

E4: Variable “E4” is continuous, and has symmetry distribution.

#E4 <- as.factor(newdata$E4)  
#levels(E4)  
hist(newdata$E4)

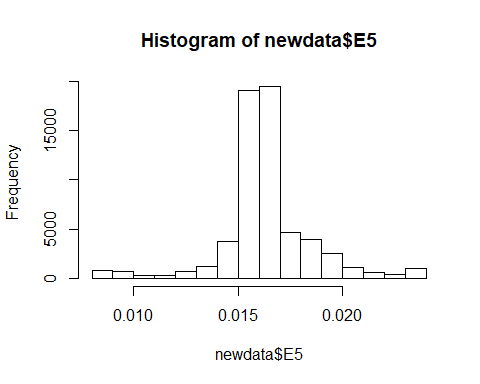


#barplot(table(newdata$E4),xlab= 'E4')  
summary(newdata$E4)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -250.000 -8.000 0.000 -2.336 6.000 250.000

E5: Variable “E5” is continuous, and has symmetry distribution.

#E5 <- as.factor(newdata$E5)  
#levels(E5)  
hist(newdata$E5)

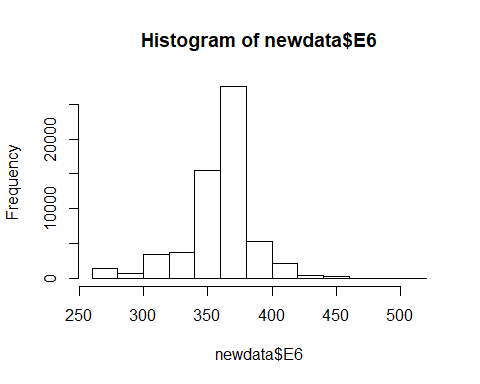


#barplot(table(newdata$E5),xlab= 'E5')  
summary(newdata$E5)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00800 0.01562 0.01600 0.01629 0.01688 0.02394

E6: Variable “E6” is continuous, and has symmetry distribution.

#E6 <- as.factor(newdata$E6)  
#levels(E6)  
hist(newdata$E6)



#barplot(table(newdata$E6),xlab= 'E6')  
summary(newdata$E6)

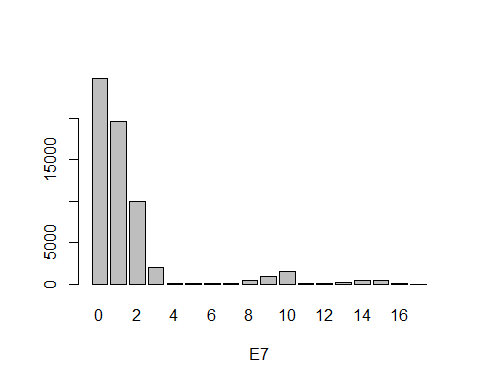
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 260.0 350.0 366.0 359.1 367.0 513.0

E7: Variable “E7” is categorial, with 18 levels. because all values of E7 are integers, and the values follow an order from 1 to 16 and the last one is 20.

E7 <- as.factor(newdata$E7)  
levels(E7)

## [1] "0" "1" "2" "3" "4" "5" "6" "7" "8" "9" "10" "11" "12" "13"  
## [15] "14" "15" "16" "20"

#hist(newdata$E7)  
barplot(table(newdata$E7),xlab= 'E7')



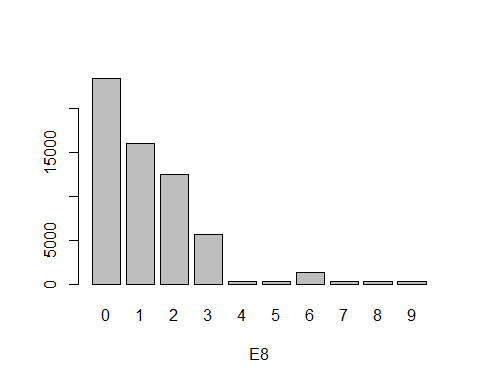
#summary(newdata$E7)

E8: Variable “E8” is categorial, with 10 levels.

E8 <- as.factor(newdata$E8)  
levels(E8)

## [1] "0" "1" "2" "3" "4" "5" "6" "7" "8" "9"

#hist(newdata$E8)  
barplot(table(newdata$E8),xlab= 'E8')



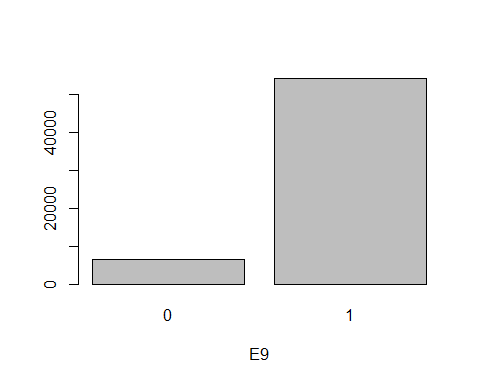
#summary(newdata$E8)

E9: Variable “E9” is categorial, with 2 levels.

E9 <- as.factor(newdata$E9)  
levels(E9)

## [1] "0" "1"

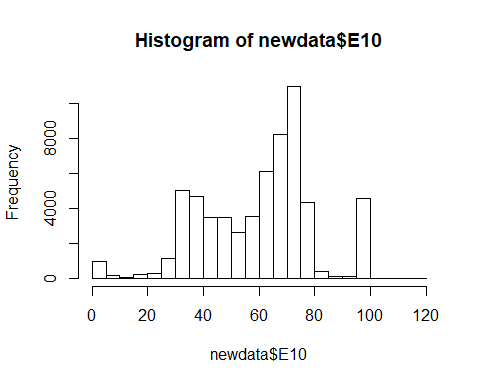
#hist(newdata$E9)  
barplot(table(newdata$E9),xlab= 'E9')



#summary(newdata$E9)

E10: Variable “E10” is continuous, and has multimodality distribution.

#E10 <- as.factor(newdata$E10)  
#levels(E10)  
hist(newdata$E10)

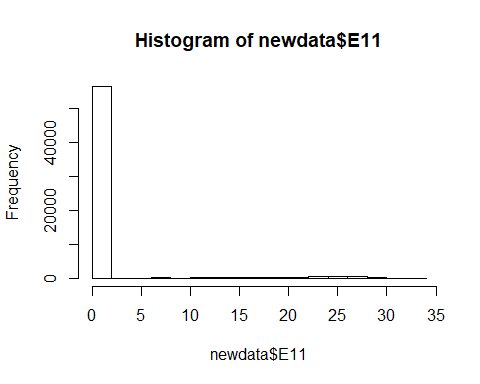


#barplot(table(newdata$E10),xlab= 'E10')  
summary(newdata$E10)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00 44.00 65.00 60.16 72.00 117.00

E11: Variable “E11” is continuous, and has right-skewed distribution.

#E11 <- as.factor(newdata$E11)  
#levels(E11)  
hist(newdata$E11)

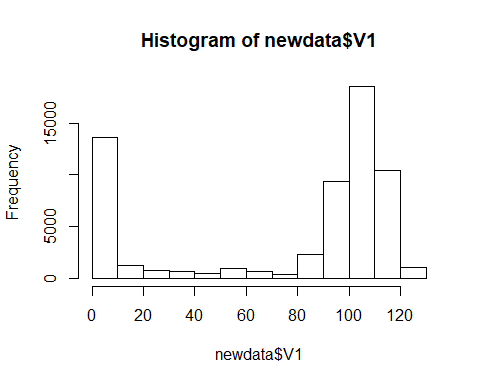


#barplot(table(newdata$E11),xlab= 'E11')  
summary(newdata$E11)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.000 0.000 0.000 1.293 0.000 32.800

V1: Variable “V1” is continuous, and has multimodality distribution.

#V1 <- as.factor(newdata$V1)  
#levels(V1)  
hist(newdata$V1)

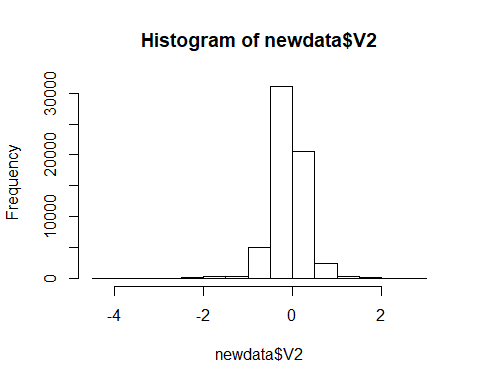


#barplot(table(newdata$V1),xlab= 'V1')  
summary(newdata$V1)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00 23.64 99.88 75.73 108.13 123.88

V2: Variable “V2” is continuous, and has symmetry distribution.

#V2 <- as.factor(newdata$V2)  
#levels(V2)  
hist(newdata$V2)

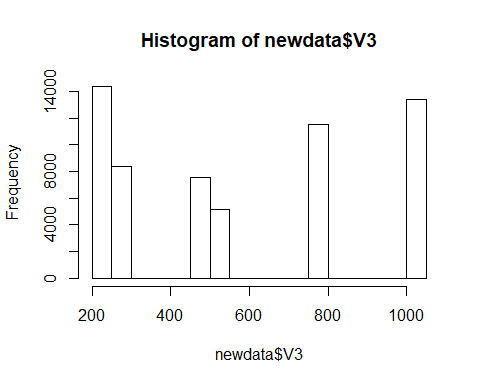


#barplot(table(newdata$V2),xlab= 'V2')  
summary(newdata$V2)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -4.20000 -0.07000 0.00000 -0.02632 0.07000 2.83500

V3: Variable “V3” is continuous, and has multimodality distribution.

#V3 <- as.factor(newdata$V3)  
#levels(V3)  
hist(newdata$V3)

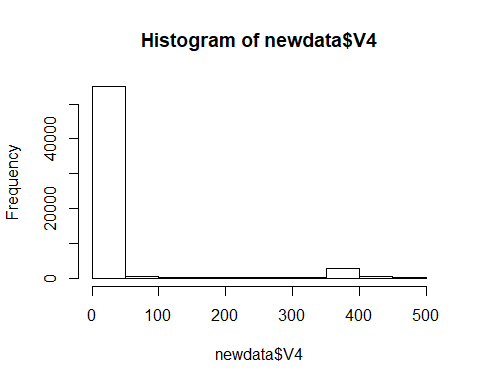


#barplot(table(newdata$V3),xlab= 'V3')  
summary(newdata$V3)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 240 255 497 568 767 1023

V4: Variable “V4” is continuous, and has right-skewed distribution.

#V4 <- as.factor(newdata$V4)  
#levels(V4)  
hist(newdata$V4)



#barplot(table(newdata$V4),xlab= 'V4')  
summary(newdata$V4)

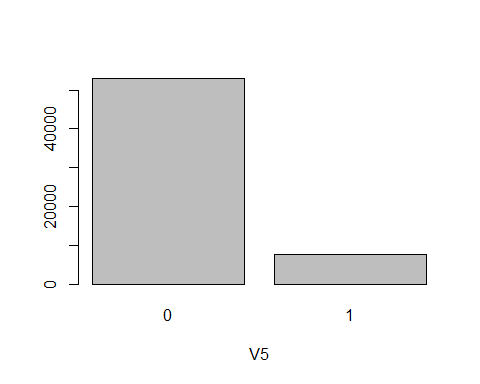
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.000 1.488 3.019 34.079 9.012 479.981

V5: Variable “V5” is categorial, with 2 levels.

V5 <- as.factor(newdata$V5)  
levels(V5)

## [1] "0" "1"

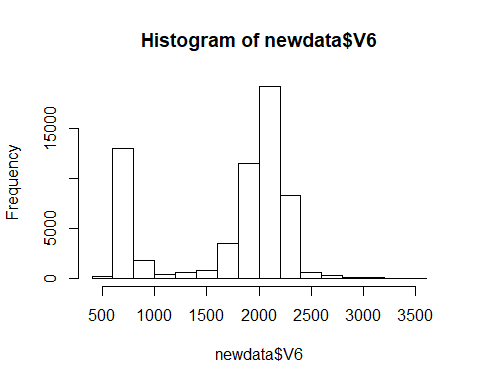
#hist(newdata$V5)  
barplot(table(newdata$V5),xlab= 'V5')



#summary(newdata$V5)

V6: Variable “V6” is continuous, and has multimodality distribution.

#V6 <- as.factor(newdata$V6)  
#levels(V6)  
hist(newdata$V6)

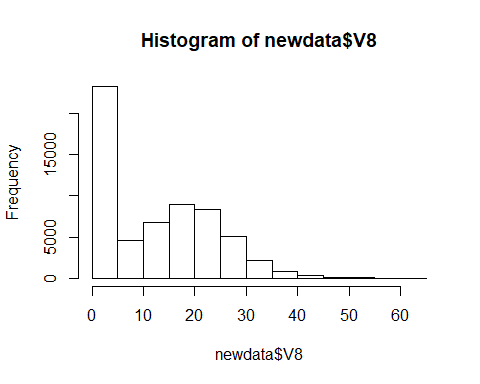


#barplot(table(newdata$V6),xlab= 'V6')  
summary(newdata$V6)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 531 1068 1982 1708 2137 3466

V8: Variable “V8” is continuous, and has right-skewed distribution.

#V8 <- as.factor(newdata$V8)  
#levels(V8)  
hist(newdata$V8)



#barplot(table(newdata$V8),xlab= 'V8')  
summary(newdata$V8)

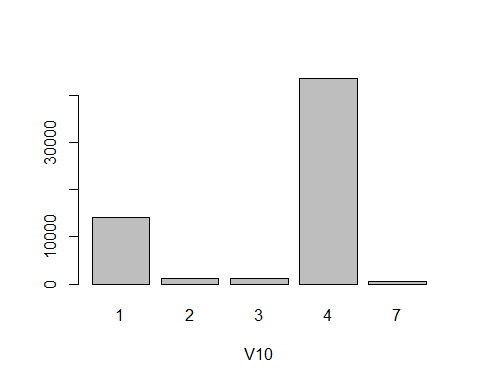
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00 0.00 12.10 12.18 21.00 62.10

V10: Variable “V10” is categorial, with 5 levels.

V10 <- as.factor(newdata$V10)  
levels(V10)

## [1] "1" "2" "3" "4" "7"

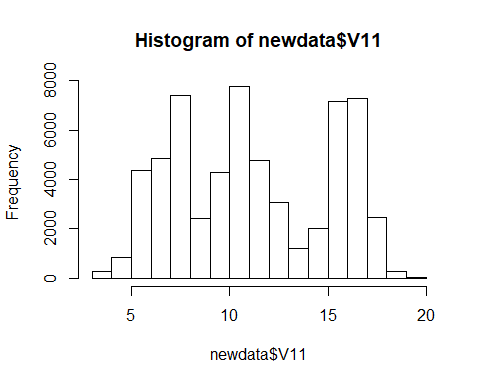
#hist(newdata$V10)  
barplot(table(newdata$V10),xlab= 'V10')



#summary(newdata$V10)

V11: Variable “V11” is continuous, and has multimodality distribution.

#V11 <- as.factor(newdata$V11)  
#levels(V11)  
hist(newdata$V11)

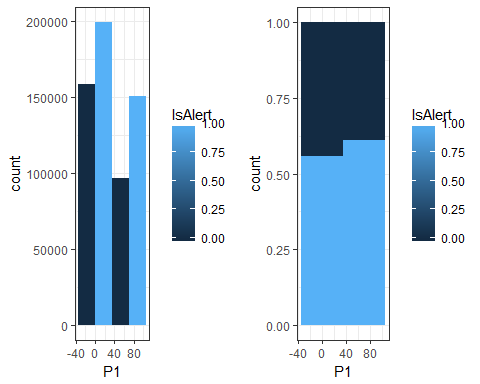


#barplot(table(newdata$V11),xlab= 'V11')  
summary(newdata$V11)

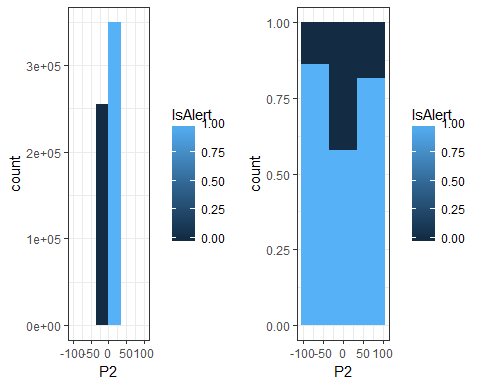
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 3.506 7.498 10.796 11.179 15.232 19.711

#### 3.Graphical investigation to whether the feature has a relationship with the response

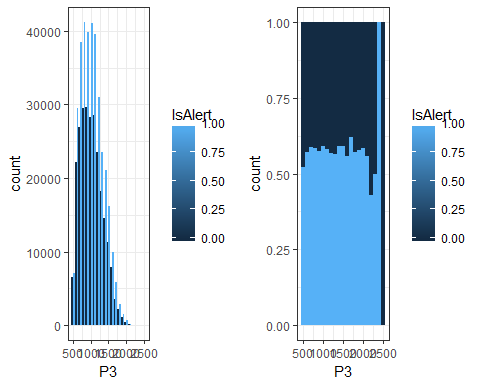
library(gridExtra)  
library(ggplot2)  
#P1  
p1\_1<-ggplot(data,aes(x=P1,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
p1\_2<-ggplot(data,aes(x=P1,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(p1\_1, p1\_2, ncol=2)



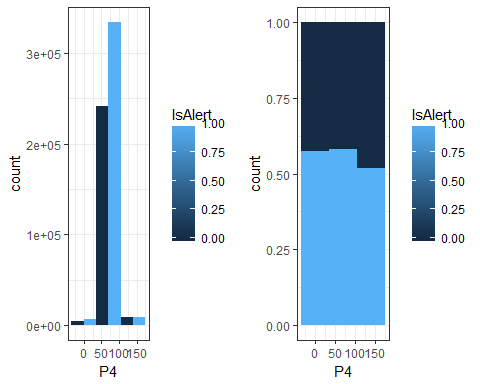
#P2  
p2\_1<-ggplot(data,aes(x=P2,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
p2\_2<-ggplot(data,aes(x=P2,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(p2\_1, p2\_2, ncol=2)



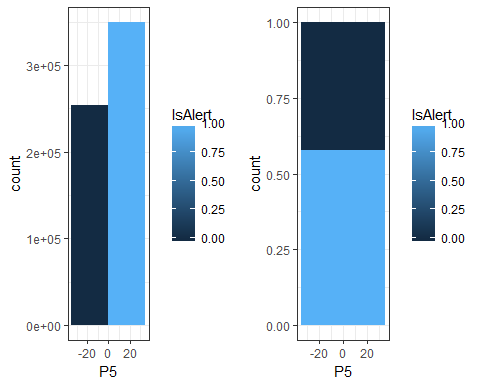
#P3  
p3\_1<-ggplot(data,aes(x=P3,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=100)+theme\_bw()  
p3\_2<-ggplot(data,aes(x=P3,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=100)+theme\_bw()  
grid.arrange(p3\_1, p3\_2, ncol=2)



#P4  
p4\_1<-ggplot(data,aes(x=P4,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
p4\_2<-ggplot(data,aes(x=P4,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(p4\_1, p4\_2, ncol=2)

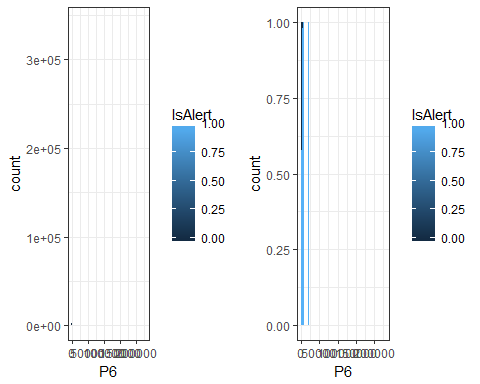


#P5  
p5\_1<-ggplot(data,aes(x=P5,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
p5\_2<-ggplot(data,aes(x=P5,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(p5\_1, p5\_2, ncol=2)

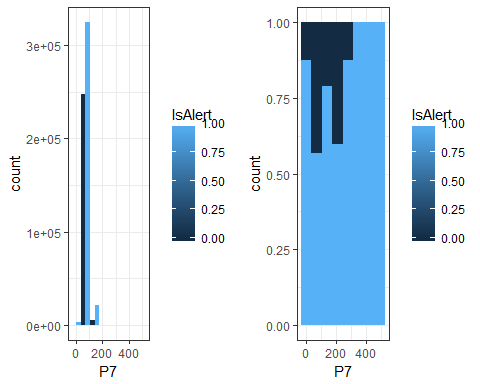


#P6  
p6\_1<-ggplot(data,aes(x=P6,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=1000)+theme\_bw()  
p6\_2<-ggplot(data,aes(x=P6,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=1000)+theme\_bw()  
grid.arrange(p6\_1, p6\_2, ncol=2)

## Warning: Removed 440 rows containing missing values (geom\_bar).

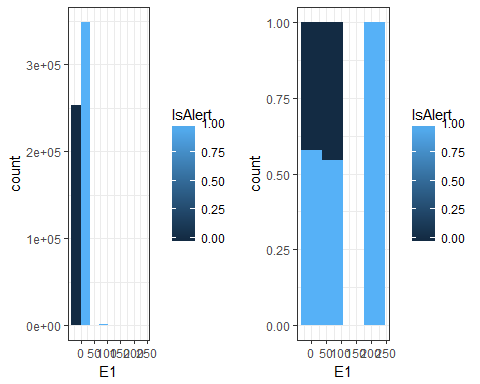


#P7  
p7\_1<-ggplot(data,aes(x=P7,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
p7\_2<-ggplot(data,aes(x=P7,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(p7\_1, p7\_2, ncol=2)

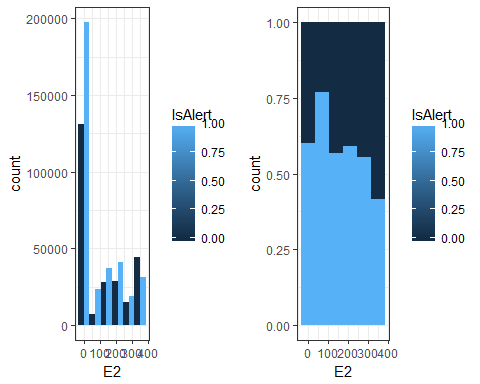
 Among all Physical data, the chang of P5 has no effect on the change of response variable, which indicates that P5 has no relationship with response variable

#E1  
e1\_1<-ggplot(data,aes(x=E1,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e1\_2<-ggplot(data,aes(x=E1,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e1\_1, e1\_2, ncol=2)

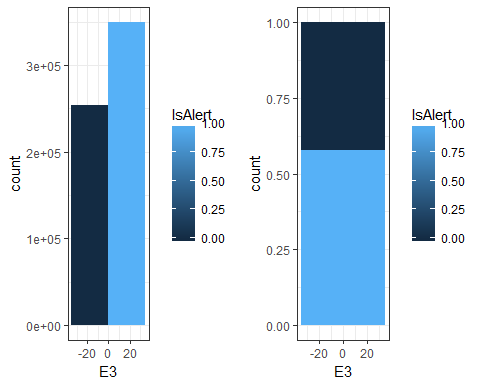
## Warning: Removed 2 rows containing missing values (geom\_bar).



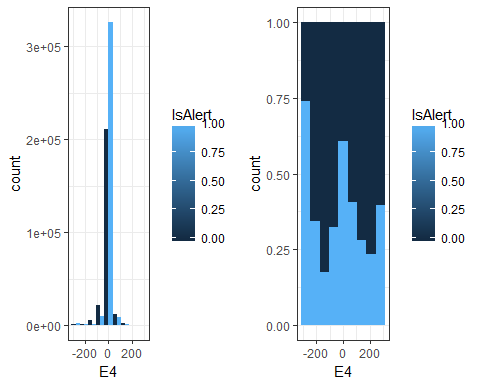
#E2  
e2\_1<-ggplot(data,aes(x=E2,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e2\_2<-ggplot(data,aes(x=E2,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e2\_1, e2\_2, ncol=2)



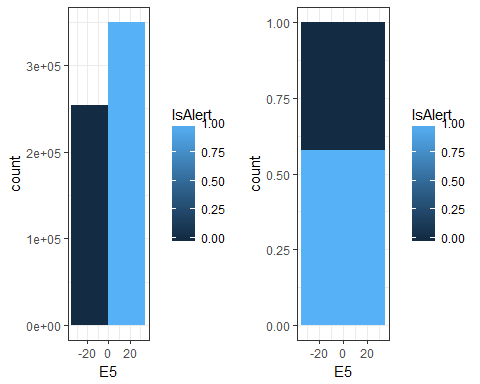
#E3  
e3\_1<-ggplot(data,aes(x=E3,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e3\_2<-ggplot(data,aes(x=E3,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e3\_1, e3\_2, ncol=2)



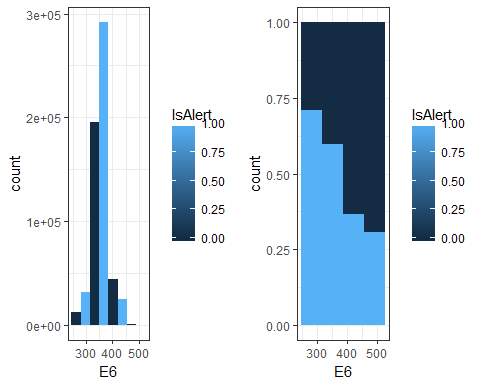
#E4  
e4\_1<-ggplot(data,aes(x=E4,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e4\_2<-ggplot(data,aes(x=E4,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e4\_1, e4\_2, ncol=2)



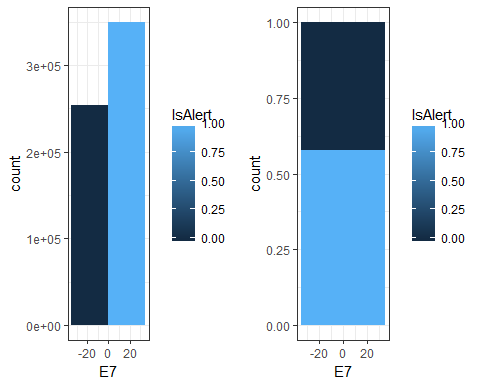
#E5  
e5\_1<-ggplot(data,aes(x=E5,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e5\_2<-ggplot(data,aes(x=E5,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e5\_1, e5\_2, ncol=2)



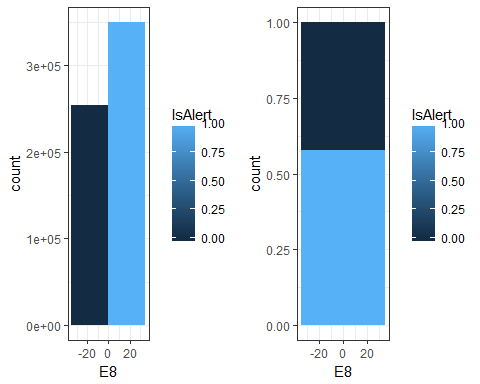
#E6  
e6\_1<-ggplot(data,aes(x=E6,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e6\_2<-ggplot(data,aes(x=E6,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e6\_1, e6\_2, ncol=2)



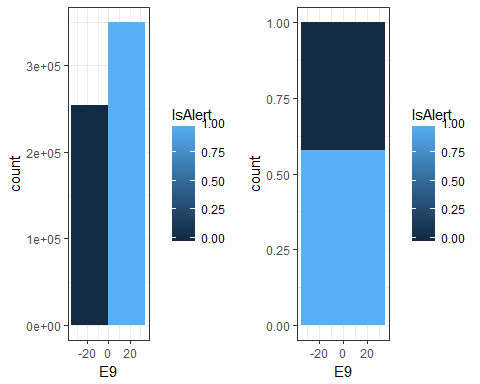
#E7  
e7\_1<-ggplot(data,aes(x=E7,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e7\_2<-ggplot(data,aes(x=E7,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e7\_1, e7\_2, ncol=2)



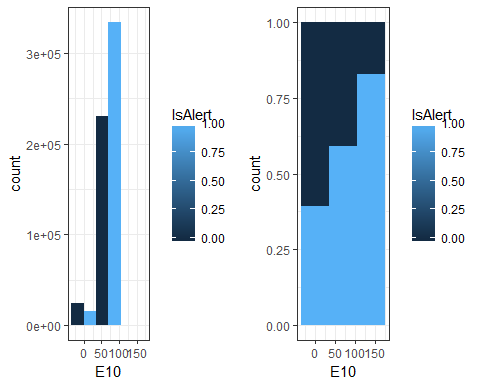
#E8  
e8\_1<-ggplot(data,aes(x=E8,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e8\_2<-ggplot(data,aes(x=E8,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e8\_1, e8\_2, ncol=2)



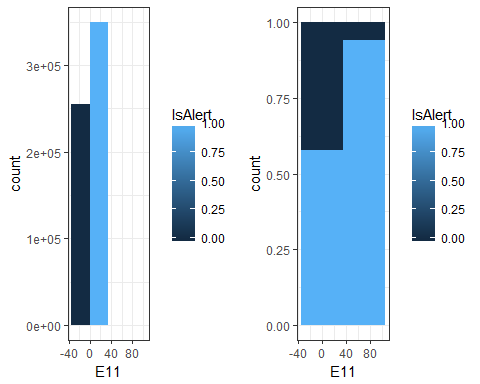
#E9  
e9\_1<-ggplot(data,aes(x=E9,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e9\_2<-ggplot(data,aes(x=E9,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e9\_1, e9\_2, ncol=2)



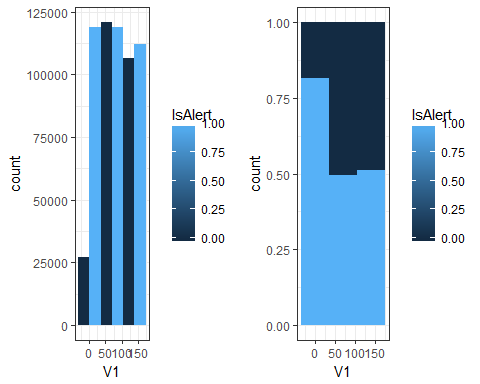
#E10  
e10\_1<-ggplot(data,aes(x=E10,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e10\_2<-ggplot(data,aes(x=E10,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e10\_1, e10\_2, ncol=2)



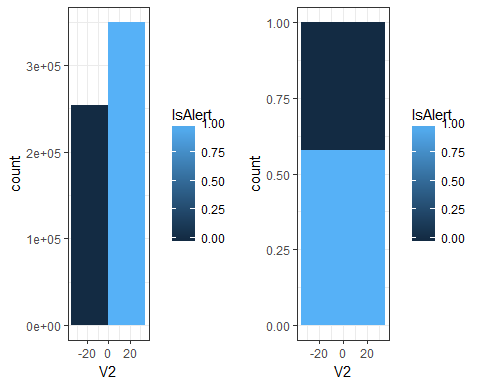
#E11  
e11\_1<-ggplot(data,aes(x=E11,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
e11\_2<-ggplot(data,aes(x=E11,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(e11\_1, e11\_2, ncol=2)

 E3, E5, E7, E8, E9 have no relationship with response variable. However, considered some of them are categorial variables, we cannot simply delete them from dataset.

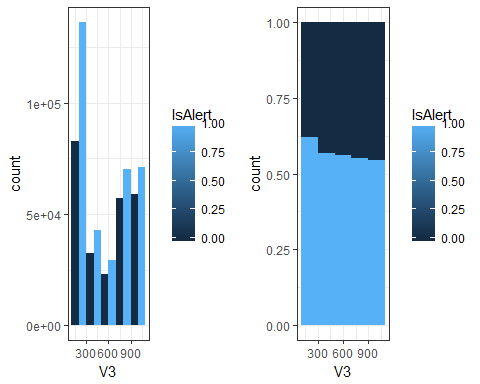
#V1  
v1\_1<-ggplot(data,aes(x=V1,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
v1\_2<-ggplot(data,aes(x=V1,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(v1\_1, v1\_2, ncol=2)



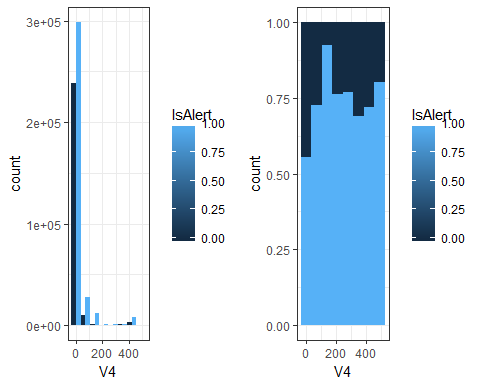
#V2  
v2\_1<-ggplot(data,aes(x=V2,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
v2\_2<-ggplot(data,aes(x=V2,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(v2\_1, v2\_2, ncol=2)



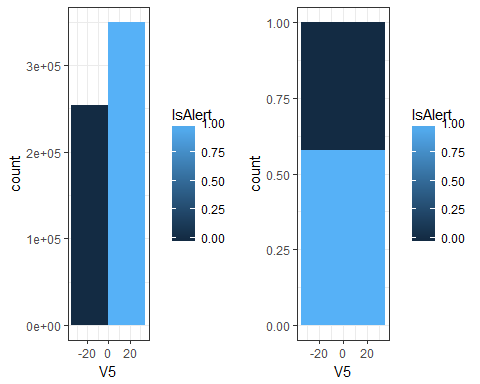
#V3  
v3\_1<-ggplot(data,aes(x=V3,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=200)+theme\_bw()  
v3\_2<-ggplot(data,aes(x=V3,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=200)+theme\_bw()  
grid.arrange(v3\_1, v3\_2, ncol=2)



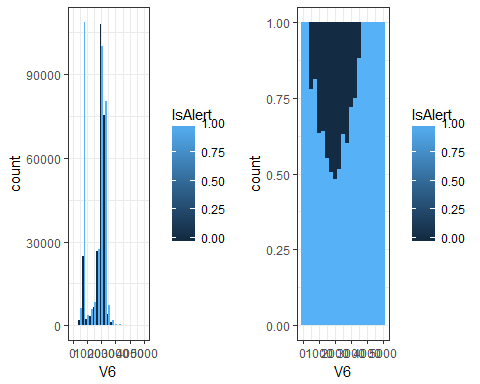
#V4  
v4\_1<-ggplot(data,aes(x=V4,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
v4\_2<-ggplot(data,aes(x=V4,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(v4\_1, v4\_2, ncol=2)



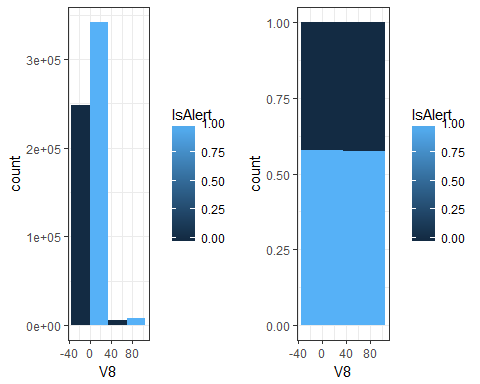
#V5  
v5\_1<-ggplot(data,aes(x=V5,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
v5\_2<-ggplot(data,aes(x=V5,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(v5\_1, v5\_2, ncol=2)



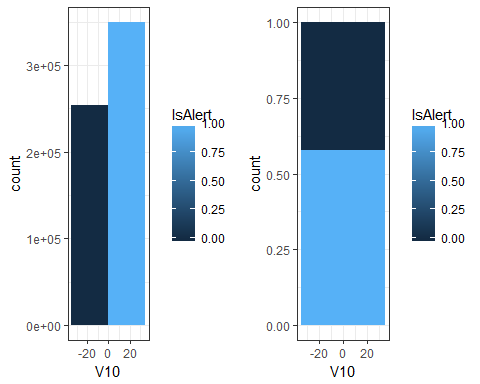
#V6  
v6\_1<-ggplot(data,aes(x=V6,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=250)+theme\_bw()  
v6\_2<-ggplot(data,aes(x=V6,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=250)+theme\_bw()  
grid.arrange(v6\_1, v6\_2, ncol=2)



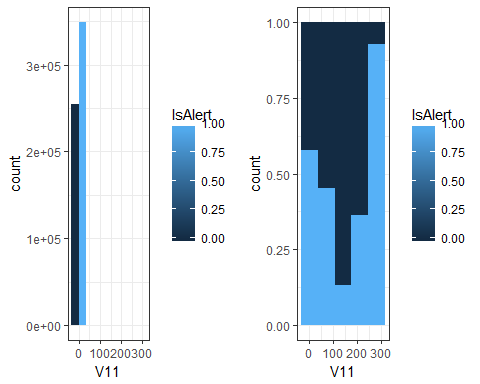
#V8  
v8\_1<-ggplot(data,aes(x=V8,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
v8\_2<-ggplot(data,aes(x=V8,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(v8\_1, v8\_2, ncol=2)



#V10  
v10\_1<-ggplot(data,aes(x=V10,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
v10\_2<-ggplot(data,aes(x=V10,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(v10\_1, v10\_2, ncol=2)

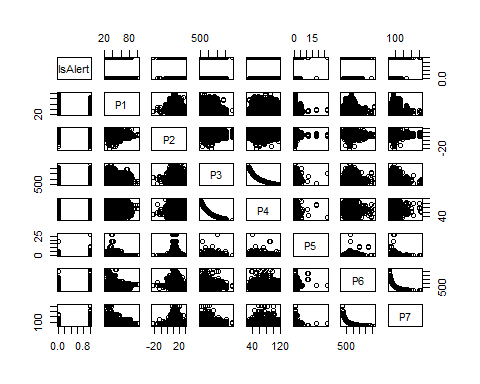


#V11  
v11\_1<-ggplot(data,aes(x=V11,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="dodge",binwidth=70)+theme\_bw()  
v11\_2<-ggplot(data,aes(x=V11,group=IsAlert,fill=IsAlert))+ geom\_histogram(position="fill",binwidth=70)+theme\_bw()  
grid.arrange(v11\_1, v11\_2, ncol=2)

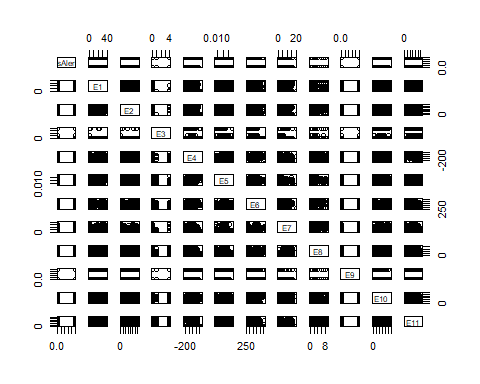
 V2, V5, V8, V10 have no relationship with response wariable. Similarly, we have to consider the categorial data.

### 4.Identify pairs of features which are related to each other

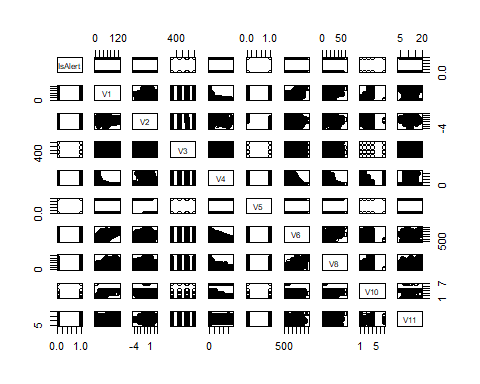
#P~  
pairs(newdata[,c(3,4,5,6,7,8,9,10)])



#E~  
pairs(newdata[,c(3,11, 12,13,14,15,16,17,18,19,20,21)])



#V~  
pairs(newdata[,c(3,22, 23,24,25,26,27,28, 29, 30)])



### 5. Dimension reduction

1). Remove the features which have no relationship with the response Becasue variables E1 to E11 are environmental data, which is reasonable to consider that certain environmental features could have no relationship with response variable. And to determine the specific variables which have no relationship with response variable, we need to check scatter plot matrix. And for those who have no relationship with response variable, the distribution of dots for “1” and “0” would be same. Therefore, I decide to delete the following variables: E1 E2 E5 E10 E11

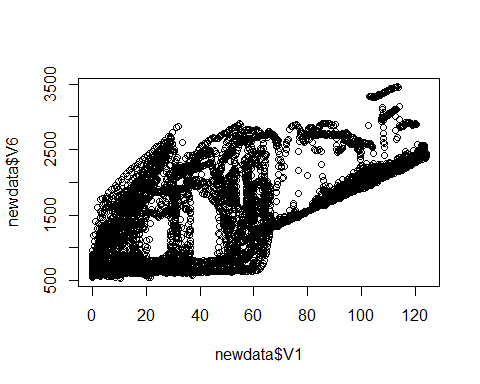
drops <- c("E1","E2","E5","E10","E11")  
newdata<-newdata[ , !(names(newdata) %in% drops)]

2). Remove a feature(s) that has a perfect relationship to another one P3 - P4, drop P4 P6 - P7, drop P7

drops <- c("P4","P7")  
newdata<-newdata[ , !(names(newdata) %in% drops)]

3)Reduce the number of features which are significantly related to each other in some non-linear way V1 - V6, drop V6

plot(newdata$V1, newdata$V6)



drops <- c("V6")  
newdata<-newdata[ , !(names(newdata) %in% drops)]

4)Transform a set of highly correlated features into a smaller set, by using wrapper method

#install.packages("caret", repos = "http://cran.r-project.org", dependencies = c("Depends", "Imports", "Suggests"))  
#install.packages("ggplot2", dependencies = TRUE)  
#install.packages("Rcpp", dependencies = TRUE)   
library(ggplot2)  
library(caret)

library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':  
##   
## margin

## The following object is masked from 'package:gridExtra':  
##   
## combine

temp\_train <- newdata[1:100,]  
temp\_train$IsAlert <- as.factor(temp\_train$IsAlert)  
#Feature selection using rfe in caret  
control <- rfeControl(functions = rfFuncs,method = "repeatedcv",repeats = 3, verbose = FALSE)  
outcomeName<-'IsAlert'  
predictors<-names(temp\_train)[!names(temp\_train) %in% outcomeName]  
result <- rfe(temp\_train[,predictors], temp\_train[,outcomeName],rfeControl = control)  
result

##   
## Recursive feature selection  
##   
## Outer resampling method: Cross-Validated (10 fold, repeated 3 times)   
##   
## Resampling performance over subset size:  
##   
## Variables Accuracy Kappa AccuracySD KappaSD Selected  
## 4 0.9926 0.9667 0.04057 0.1826 \*  
## 8 0.9796 0.9237 0.05089 0.2081   
## 16 0.9661 0.8806 0.05705 0.2228   
## 21 0.9661 0.8806 0.05705 0.2228   
##   
## The top 4 variables (out of 4):  
## E4, ObsNum, V1, E6

Now we have four variables which might have significant influence on response variable. they are E3, ObsNum, V1, and E6. However, F=four variables are insufficient to make analysis, So I will still keep as 22 variables.

Until now, we have 22 variables as following: P1:continuous P2:continuous P3:continuous P5:continuous P6:continuous E3:categorial, with 3 levels E4:continuous E6:continuous E7:categorial, with 18 levels E8:categorial, with 10 levels E9:categorial, with 2 levels V1:continuous V2:continuous V3:continuous V4:continuous V5:categorial, with 2 levels V8:continuous V10:categorial, with 5 levels V11:continuous

## Basic Model building

#Check indicator variables  
unique(newdata$E3)

## [1] 1 0 4

unique(newdata$E7)

## [1] 1 0 6 5 3 13 14 15 16 12 11 2 10 9 8 7 4 20

unique(newdata$E8)

## [1] 1 0 4 7 2 3 5 8 6 9

unique(newdata$E9)

## [1] 1 0

unique(newdata$V5)

## [1] 0 1

unique(newdata$V10)

## [1] 4 3 7 1 2

#Create indicator variable  
newdata$E3\_1 = c(rep(0, length(newdata$IsAlert)))  
newdata$E3\_0 = newdata$E3\_4 = newdata$E7\_low = newdata$E7\_high = newdata$E8\_low = newdata$E8\_high = newdata$E9\_0 = newdata$E9\_1 = newdata$V5\_0 = newdata$V5\_1 = newdata$V10\_low = newdata$V10\_high = newdata$E3\_1   
  
for (i in 1:length(newdata$IsAlert)) {  
#E3   
if(newdata$E3[i] == 1) newdata$E3\_1[i] <- 1  
if(newdata$E3[i] == 0) newdata$E3\_0[i] <- 1  
if(newdata$E3[i] == 4) newdata$E3\_4[i] <- 1  
  
#E7  
if(newdata$E7[i] <= 10) newdata$E7\_low[i] <- 1  
if(newdata$E7[i] > 10) newdata$E7\_high[i] <- 1  
  
#E8  
if(newdata$E8[i] <= 4) newdata$E7\_low[i] <- 1  
if(newdata$E8[i] > 4) newdata$E7\_high[i] <- 1  
  
#E9  
if(newdata$E9[i] == 1) newdata$E9\_1[i] <- 1  
if(newdata$E9[i] == 0) newdata$E9\_0[i] <- 1  
  
#V5  
if(newdata$V5[i] == 1) newdata$V5\_1[i] <- 1  
if(newdata$V5[i] == 0) newdata$V5\_0[i] <- 1  
  
#V10  
if(newdata$E8[i] <= 3) newdata$V10\_low[i] <- 1  
if(newdata$E8[i] > 3) newdata$V10\_high[i]<- 1  
}

# Minimax transform the continuous variables  
#P1  
newdata$P1\_mm <- (newdata$P1 - min(newdata$P1)) / (max(newdata$P1) - min(newdata$P1))  
#P2  
newdata$P2\_mm <- (newdata$P2 - min(newdata$P2)) / (max(newdata$P2) - min(newdata$P2))  
#P3  
newdata$P3\_mm <- (newdata$P3 - min(newdata$P3)) / (max(newdata$P3) - min(newdata$P3))  
#P5  
newdata$P5\_mm <- (newdata$P5 - min(newdata$P5)) / (max(newdata$P5) - min(newdata$P5))  
#P6  
newdata$P6\_mm <- (newdata$P6 - min(newdata$P6)) / (max(newdata$P6) - min(newdata$P6))  
#E4  
newdata$E4\_mm <- (newdata$E4 - min(newdata$E4)) / (max(newdata$E4) - min(newdata$E4))  
#E6  
newdata$E6\_mm <- (newdata$E6 - min(newdata$E6)) / (max(newdata$E6) - min(newdata$E6))  
#V1  
newdata$V1\_mm <- (newdata$V1 - min(newdata$V1)) / (max(newdata$V1) - min(newdata$V1))  
#V2  
newdata$V2\_mm <- (newdata$V2 - min(newdata$V2)) / (max(newdata$V2) - min(newdata$V2))  
#V3  
newdata$V3\_mm <- (newdata$V3 - min(newdata$V3)) / (max(newdata$V3) - min(newdata$V3))  
#V4  
newdata$V4\_mm <- (newdata$V4 - min(newdata$V4)) / (max(newdata$V4) - min(newdata$V4))  
#V8  
newdata$V8\_mm <- (newdata$V8 - min(newdata$V8)) / (max(newdata$V8) - min(newdata$V8))  
#V11  
newdata$V11\_mm <- (newdata$V11 - min(newdata$V11)) / (max(newdata$V11) - min(newdata$V11))

#new dataset  
mydata\_nn <- newdata[ , -c(4:22)]  
mydata\_nn <- mydata\_nn[, -c(1, 2)]  
mydata\_nn <- mydata\_nn[1:10000,]  
#training data  
training <- mydata\_nn[1:7000, ]  
#testing data  
testing <- mydata\_nn[7001:10000,]

#### Neural Network

1. Constructing and comparing model

library(nnet) # Requires package nnet  
set.seed(2018)  
myNewFrame <- data.frame(node = integer(0),maxit = integer(0), decay = integer(0) ,misscls.train = integer(0), misscls.test = integer(0))  
  
for (m in c(10, 100, 10000)) {  
 for (d in c(0, 0.1, 0.2)) {  
 for (s in c(5, 10, 15)) {  
 net.dat <- nnet(IsAlert~., data = training, size = s, decay = d, maxit = m)  
 #table(round(net.dat$fitted.values, 1))  
   
 estimated\_isAlert=as.numeric(net.dat$fitted.values>0.5)  
   
 T = table(estimated\_isAlert,training$IsAlert)  
   
 misscls.train<-sum((estimated\_isAlert - training$IsAlert)^2)/length(training$IsAlert)  
 misscls.train  
 presdicted.testing<-as.numeric(predict(net.dat,testing )>0.5)  
 Ttest = table(presdicted.testing, testing$IsAlert)  
 misscls.test<-sum((presdicted.testing - testing$IsAlert)^2)/length(testing$IsAlert)  
 misscls.test  
   
 df <- data.frame(s, m, d, misscls.train, misscls.test)  
 names(df)<-c("node", "maxit", "decay", "misscls.train", "misscls.test")  
 myNewFrame <- rbind(myNewFrame, df)  
   
 }  
 }  
}

## # weights: 141  
## initial value 1306.799428   
## final value 396.000000   
## converged  
## # weights: 281  
## initial value 2012.521226   
## final value 1744.000000   
## converged  
## # weights: 421  
## initial value 2549.624964   
## final value 1744.000000   
## converged  
## # weights: 141  
## initial value 2029.279725   
## iter 10 value 1281.854021  
## final value 1281.854021   
## stopped after 10 iterations  
## # weights: 281  
## initial value 2391.598699   
## iter 10 value 1495.701339  
## final value 1495.701339   
## stopped after 10 iterations  
## # weights: 421  
## initial value 1465.568100   
## iter 10 value 421.075809  
## final value 421.075809   
## stopped after 10 iterations  
## # weights: 141  
## initial value 2107.655542   
## iter 10 value 1503.434091  
## final value 1503.434091   
## stopped after 10 iterations  
## # weights: 281  
## initial value 3246.847111   
## iter 10 value 1527.330158  
## final value 1527.330158   
## stopped after 10 iterations  
## # weights: 421  
## initial value 2786.487949   
## iter 10 value 1678.682384  
## final value 1678.682384   
## stopped after 10 iterations  
## # weights: 141  
## initial value 1410.739557   
## final value 396.000000   
## converged  
## # weights: 281  
## initial value 1257.845833   
## final value 396.000000   
## converged  
## # weights: 421  
## initial value 1381.157690   
## iter 10 value 361.583794  
## iter 20 value 351.994633  
## iter 30 value 345.572434  
## iter 40 value 334.199224  
## iter 50 value 317.929717  
## iter 60 value 305.379378  
## iter 70 value 292.090277  
## iter 80 value 272.128459  
## iter 90 value 258.600778  
## iter 100 value 249.058926  
## final value 249.058926   
## stopped after 100 iterations  
## # weights: 141  
## initial value 1406.774649   
## iter 10 value 509.934431  
## iter 20 value 376.834567  
## iter 30 value 364.277056  
## iter 40 value 363.897512  
## iter 50 value 363.637781  
## iter 60 value 363.539481  
## iter 70 value 363.338802  
## iter 80 value 363.243296  
## iter 90 value 363.059820  
## iter 100 value 363.007993  
## final value 363.007993   
## stopped after 100 iterations  
## # weights: 281  
## initial value 1941.192597   
## iter 10 value 1576.397392  
## iter 20 value 430.834136  
## iter 30 value 380.908419  
## iter 40 value 365.992123  
## iter 50 value 364.144361  
## iter 60 value 363.447173  
## iter 70 value 363.247153  
## iter 80 value 363.168043  
## iter 90 value 363.029031  
## iter 100 value 362.885454  
## final value 362.885454   
## stopped after 100 iterations  
## # weights: 421  
## initial value 1343.584185   
## iter 10 value 420.587566  
## iter 20 value 366.720145  
## iter 30 value 363.656050  
## iter 40 value 363.260930  
## iter 50 value 363.146127  
## iter 60 value 362.951359  
## iter 70 value 362.762791  
## iter 80 value 362.564508  
## iter 90 value 362.474733  
## iter 100 value 362.442954  
## final value 362.442954   
## stopped after 100 iterations  
## # weights: 141  
## initial value 1390.434380   
## iter 10 value 428.396901  
## iter 20 value 374.731388  
## iter 30 value 369.920856  
## iter 40 value 369.261492  
## iter 50 value 368.834084  
## iter 60 value 368.635723  
## iter 70 value 368.576908  
## iter 80 value 368.574660  
## iter 90 value 368.574445  
## iter 100 value 368.574282  
## final value 368.574282   
## stopped after 100 iterations  
## # weights: 281  
## initial value 2328.470687   
## iter 10 value 1412.431944  
## iter 20 value 473.096669  
## iter 30 value 389.573877  
## iter 40 value 375.330528  
## iter 50 value 370.561815  
## iter 60 value 369.156849  
## iter 70 value 368.813227  
## iter 80 value 368.611964  
## iter 90 value 368.491813  
## iter 100 value 368.387404  
## final value 368.387404   
## stopped after 100 iterations  
## # weights: 421  
## initial value 2547.952114   
## iter 10 value 1679.176969  
## iter 20 value 466.734073  
## iter 30 value 374.596332  
## iter 40 value 368.846596  
## iter 50 value 368.383504  
## iter 60 value 368.255936  
## iter 70 value 368.223781  
## iter 80 value 368.218594  
## iter 90 value 368.212747  
## iter 100 value 368.206357  
## final value 368.206357   
## stopped after 100 iterations  
## # weights: 141  
## initial value 2850.854771   
## final value 1744.000000   
## converged  
## # weights: 281  
## initial value 2009.172362   
## final value 1744.000000   
## converged  
## # weights: 421  
## initial value 2526.940767   
## final value 1744.000000   
## converged  
## # weights: 141  
## initial value 3591.169029   
## iter 10 value 1572.849814  
## iter 20 value 415.484227  
## iter 30 value 369.150964  
## iter 40 value 364.322632  
## iter 50 value 364.124594  
## iter 60 value 363.844907  
## iter 70 value 363.673686  
## iter 80 value 363.564042  
## iter 90 value 363.423684  
## iter 100 value 363.213267  
## iter 110 value 362.876440  
## iter 120 value 362.771216  
## iter 130 value 362.539276  
## iter 140 value 362.469943  
## iter 150 value 362.455320  
## iter 160 value 362.446853  
## iter 170 value 362.443819  
## iter 180 value 362.438245  
## iter 190 value 362.436594  
## iter 200 value 362.436126  
## iter 210 value 362.432247  
## iter 220 value 362.430566  
## iter 230 value 362.429535  
## iter 240 value 362.428531  
## iter 250 value 362.428054  
## final value 362.428020   
## converged  
## # weights: 281  
## initial value 1292.209168   
## iter 10 value 663.247708  
## iter 20 value 387.227913  
## iter 30 value 366.227343  
## iter 40 value 364.219362  
## iter 50 value 363.657815  
## iter 60 value 363.264071  
## iter 70 value 363.023963  
## iter 80 value 362.916812  
## iter 90 value 362.815893  
## iter 100 value 362.717704  
## iter 110 value 362.601341  
## iter 120 value 362.550526  
## iter 130 value 362.518659  
## iter 140 value 362.490773  
## iter 150 value 362.462390  
## iter 160 value 362.447005  
## iter 170 value 362.436124  
## iter 180 value 362.429136  
## iter 190 value 362.423164  
## iter 200 value 362.417193  
## iter 210 value 362.412477  
## iter 220 value 362.408908  
## iter 230 value 362.406278  
## iter 240 value 362.405085  
## iter 250 value 362.404442  
## iter 260 value 362.403808  
## iter 270 value 362.403495  
## iter 280 value 362.401943  
## iter 290 value 362.401233  
## iter 300 value 362.400974  
## iter 310 value 362.400838  
## final value 362.400138   
## converged  
## # weights: 421  
## initial value 1401.399696   
## iter 10 value 408.648049  
## iter 20 value 373.341776  
## iter 30 value 365.027901  
## iter 40 value 363.788715  
## iter 50 value 363.333385  
## iter 60 value 363.180033  
## iter 70 value 363.060518  
## iter 80 value 362.752318  
## iter 90 value 362.519441  
## iter 100 value 362.463256  
## iter 110 value 362.450330  
## iter 120 value 362.444239  
## iter 130 value 362.441093  
## iter 140 value 362.436359  
## iter 150 value 362.427218  
## iter 160 value 362.422082  
## iter 170 value 362.418903  
## iter 180 value 362.416198  
## iter 190 value 362.414841  
## iter 200 value 362.413345  
## iter 210 value 362.412703  
## iter 220 value 362.412166  
## iter 230 value 362.411581  
## iter 240 value 362.411267  
## iter 250 value 362.411077  
## iter 260 value 362.410935  
## iter 270 value 362.410794  
## iter 280 value 362.410598  
## iter 290 value 362.410253  
## iter 300 value 362.409644  
## iter 310 value 362.409345  
## iter 320 value 362.409148  
## iter 330 value 362.408767  
## iter 340 value 362.408669  
## iter 350 value 362.408514  
## final value 362.408506   
## converged  
## # weights: 141  
## initial value 1324.431307   
## iter 10 value 413.369813  
## iter 20 value 371.472711  
## iter 30 value 368.742135  
## iter 40 value 368.586511  
## iter 50 value 368.575998  
## iter 60 value 368.575654  
## iter 70 value 368.574568  
## final value 368.573533   
## converged  
## # weights: 281  
## initial value 2848.781381   
## iter 10 value 1602.822447  
## iter 20 value 976.212020  
## iter 30 value 459.292679  
## iter 40 value 417.234153  
## iter 50 value 372.060190  
## iter 60 value 369.043194  
## iter 70 value 368.574584  
## iter 80 value 368.387506  
## iter 90 value 368.341836  
## iter 100 value 368.314161  
## iter 110 value 368.294447  
## iter 120 value 368.283180  
## iter 130 value 368.278502  
## iter 140 value 368.276798  
## iter 150 value 368.276392  
## iter 160 value 368.276280  
## iter 170 value 368.276145  
## final value 368.276009   
## converged  
## # weights: 421  
## initial value 1591.958396   
## iter 10 value 413.185743  
## iter 20 value 389.890075  
## iter 30 value 373.016553  
## iter 40 value 369.645385  
## iter 50 value 368.702181  
## iter 60 value 368.433782  
## iter 70 value 368.321278  
## iter 80 value 368.274693  
## iter 90 value 368.246836  
## iter 100 value 368.234059  
## iter 110 value 368.226857  
## iter 120 value 368.219032  
## iter 130 value 368.213343  
## iter 140 value 368.208683  
## iter 150 value 368.205179  
## iter 160 value 368.202874  
## iter 170 value 368.201806  
## iter 180 value 368.200935  
## iter 190 value 368.200451  
## iter 200 value 368.199774  
## iter 210 value 368.199313  
## iter 220 value 368.198918  
## iter 230 value 368.198663  
## final value 368.198511   
## converged

myNewFrame

## node maxit decay misscls.train misscls.test  
## 1 5 10 0.0 0.05657143 0.1033333  
## 2 10 10 0.0 0.24914286 0.1190000  
## 3 15 10 0.0 0.24914286 0.1190000  
## 4 5 10 0.1 0.24914286 0.1190000  
## 5 10 10 0.1 0.24914286 0.1190000  
## 6 15 10 0.1 0.05657143 0.1033333  
## 7 5 10 0.2 0.24914286 0.1190000  
## 8 10 10 0.2 0.24914286 0.1190000  
## 9 15 10 0.2 0.24914286 0.1190000  
## 10 5 100 0.0 0.05657143 0.1033333  
## 11 10 100 0.0 0.05657143 0.1033333  
## 12 15 100 0.0 0.04000000 0.1143333  
## 13 5 100 0.1 0.05657143 0.1033333  
## 14 10 100 0.1 0.05657143 0.1033333  
## 15 15 100 0.1 0.05657143 0.1033333  
## 16 5 100 0.2 0.05657143 0.1033333  
## 17 10 100 0.2 0.05657143 0.1033333  
## 18 15 100 0.2 0.05657143 0.1033333  
## 19 5 10000 0.0 0.24914286 0.1190000  
## 20 10 10000 0.0 0.24914286 0.1190000  
## 21 15 10000 0.0 0.24914286 0.1190000  
## 22 5 10000 0.1 0.05657143 0.1033333  
## 23 10 10000 0.1 0.05657143 0.1033333  
## 24 15 10000 0.1 0.05657143 0.1033333  
## 25 5 10000 0.2 0.05657143 0.1033333  
## 26 10 10000 0.2 0.05657143 0.1033333  
## 27 15 10000 0.2 0.05657143 0.1033333

Based on the output, I would choose the best model with 5 nodes, maxit as 100, and decay as 0.

best\_nn <- nnet(IsAlert~., data = newdata, size = 5, decay = 0, maxit = 100)

## # weights: 246  
## initial value 17630.923600   
## iter 10 value 14460.156899  
## iter 20 value 14199.861982  
## iter 30 value 13520.782083  
## iter 40 value 13451.497230  
## iter 50 value 13400.787369  
## iter 60 value 12849.881837  
## iter 70 value 12392.392073  
## iter 80 value 12073.870893  
## iter 90 value 11418.815823  
## iter 100 value 10581.357555  
## final value 10581.357555   
## stopped after 100 iterations

misscls.train<-sum((estimated\_isAlert - training$IsAlert)^2)/length(training$IsAlert)  
misscls.train

## [1] 0.05657143

presdicted.testing<-as.numeric(predict(net.dat,testing )>0.5)  
Ttest = table(presdicted.testing, testing$IsAlert)  
misscls.test<-sum((presdicted.testing - testing$IsAlert)^2)/length(testing$IsAlert)  
misscls.test

## [1] 0.1033333

presdicted.testing.all<-as.numeric(predict(net.dat, newdata )>0.5)  
Ttest = table(presdicted.testing.all, newdata$IsAlert)  
misscls.test.all<-sum((presdicted.testing.all - newdata$IsAlert)^2)/length(newdata$IsAlert)  
misscls.test.all

## [1] 0.3264452

the best Neural Network model has 5 nodes, decay = 0, maxit = 100, the missclasification rate for training data : 0.05657143 the missclasification rate for testing data : 0.1033333 the missclasification rate for entire data : 0.3264452

## Logistic Regression

#new data set  
drops <- c("E3", "E7", "E8", "E9" , "V5","V10")  
mydata\_lm<-newdata[ , !(names(newdata) %in% drops)]  
mydata\_lm <- mydata\_lm[ , -c(30:42)]

mydata\_lm <- mydata\_lm[1:1000,]  
#training data  
training <- mydata\_lm[1:700, ]  
#testing data  
testing <- mydata\_lm[701:1000,]  
  
summary(training)

## TrialID ObsNum IsAlert P1   
## Min. :0 Min. : 0.0 Min. :0.0000 Min. :25.38   
## 1st Qu.:0 1st Qu.:174.8 1st Qu.:1.0000 1st Qu.:34.15   
## Median :0 Median :349.5 Median :1.0000 Median :35.24   
## Mean :0 Mean :349.5 Mean :0.8457 Mean :35.89   
## 3rd Qu.:0 3rd Qu.:524.2 3rd Qu.:1.0000 3rd Qu.:37.35   
## Max. :0 Max. :699.0 Max. :1.0000 Max. :44.82   
## P2 P3 P5 P6   
## Min. : 6.137 Min. : 516 Min. :0.2335 Min. :556.0   
## 1st Qu.:10.936 1st Qu.: 816 1st Qu.:0.2672 1st Qu.:592.0   
## Median :12.783 Median :1084 Median :0.2789 Median :612.0   
## Mean :12.844 Mean :1081 Mean :0.2782 Mean :617.3   
## 3rd Qu.:14.753 3rd Qu.:1400 3rd Qu.:0.2919 3rd Qu.:632.0   
## Max. :18.524 Max. :1816 Max. :0.3140 Max. :732.0   
## E4 E6 V1 V2   
## Min. :-20.0000 Min. :260.0 Min. : 93.36 Min. :-0.6650   
## 1st Qu.: -6.0000 1st Qu.:317.0 1st Qu.: 97.01 1st Qu.: 0.0000   
## Median : 0.0000 Median :320.0 Median : 99.61 Median : 0.0000   
## Mean : 0.6857 Mean :319.9 Mean : 99.47 Mean : 0.0568   
## 3rd Qu.: 8.0000 3rd Qu.:324.0 3rd Qu.:102.30 3rd Qu.: 0.0700   
## Max. : 36.0000 Max. :332.0 Max. :104.97 Max. : 0.8750   
## V3 V4 V8 V11   
## Min. : 240.0 Min. : 0.000 Min. : 0.00 Min. :14.08   
## 1st Qu.: 240.0 1st Qu.: 3.019 1st Qu.: 7.95 1st Qu.:14.83   
## Median : 496.0 Median : 4.506 Median :14.70 Median :14.94   
## Mean : 536.8 Mean : 3.952 Mean :13.70 Mean :14.95   
## 3rd Qu.: 767.0 3rd Qu.: 4.506 3rd Qu.:20.60 3rd Qu.:15.05   
## Max. :1023.0 Max. :10.500 Max. :28.60 Max. :15.60   
## E3\_1 V10\_high V10\_low V5\_1   
## Min. :0.0000 Min. :0 Min. :1 Min. :0.000000   
## 1st Qu.:1.0000 1st Qu.:0 1st Qu.:1 1st Qu.:0.000000   
## Median :1.0000 Median :0 Median :1 Median :0.000000   
## Mean :0.9857 Mean :0 Mean :1 Mean :0.005714   
## 3rd Qu.:1.0000 3rd Qu.:0 3rd Qu.:1 3rd Qu.:0.000000   
## Max. :1.0000 Max. :0 Max. :1 Max. :1.000000   
## V5\_0 E9\_1 E9\_0 E8\_high   
## Min. :0.0000 Min. :0.0000 Min. :0.00000 Min. :0   
## 1st Qu.:1.0000 1st Qu.:1.0000 1st Qu.:0.00000 1st Qu.:0   
## Median :1.0000 Median :1.0000 Median :0.00000 Median :0   
## Mean :0.9943 Mean :0.9886 Mean :0.01143 Mean :0   
## 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:0.00000 3rd Qu.:0   
## Max. :1.0000 Max. :1.0000 Max. :1.00000 Max. :0   
## E8\_low E7\_high E7\_low E3\_4 E3\_0   
## Min. :0 Min. :0 Min. :1 Min. :0 Min. :0.00000   
## 1st Qu.:0 1st Qu.:0 1st Qu.:1 1st Qu.:0 1st Qu.:0.00000   
## Median :0 Median :0 Median :1 Median :0 Median :0.00000   
## Mean :0 Mean :0 Mean :1 Mean :0 Mean :0.01429   
## 3rd Qu.:0 3rd Qu.:0 3rd Qu.:1 3rd Qu.:0 3rd Qu.:0.00000   
## Max. :0 Max. :0 Max. :1 Max. :0 Max. :1.00000

logistic.model <- glm(IsAlert~., data = training, family = binomial(link="logit"))  
summary(logistic.model)

##   
## Call:  
## glm(formula = IsAlert ~ ., family = binomial(link = "logit"),   
## data = training)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.9021 0.1444 0.3130 0.5397 1.5659   
##   
## Coefficients: (11 not defined because of singularities)  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 3.733e+01 1.686e+03 0.022 0.982335   
## TrialID NA NA NA NA   
## ObsNum 8.934e-04 1.137e-03 0.786 0.432118   
## P1 7.744e-02 1.345e-01 0.576 0.564902   
## P2 2.972e-02 4.806e-02 0.618 0.536281   
## P3 -1.408e-03 4.476e-04 -3.145 0.001659 \*\*   
## P5 -4.477e+01 8.113e+00 -5.519 3.42e-08 \*\*\*  
## P6 2.204e-03 5.182e-03 0.425 0.670534   
## E4 2.433e-02 1.371e-02 1.774 0.076050 .   
## E6 5.952e-02 1.755e-02 3.392 0.000693 \*\*\*  
## V1 1.121e-01 5.888e-02 1.904 0.056877 .   
## V2 3.510e+00 1.104e+00 3.181 0.001467 \*\*   
## V3 -1.404e-03 4.887e-04 -2.873 0.004070 \*\*   
## V4 -6.796e-01 1.737e-01 -3.912 9.15e-05 \*\*\*  
## V8 2.384e-02 1.801e-02 1.323 0.185720   
## V11 -1.561e+00 1.743e+00 -0.896 0.370507   
## E3\_1 -1.559e+01 1.150e+03 -0.014 0.989190   
## V10\_high NA NA NA NA   
## V10\_low NA NA NA NA   
## V5\_1 1.474e+01 1.944e+03 0.008 0.993948   
## V5\_0 NA NA NA NA   
## E9\_1 -1.427e+01 1.232e+03 -0.012 0.990763   
## E9\_0 NA NA NA NA   
## E8\_high NA NA NA NA   
## E8\_low NA NA NA NA   
## E7\_high NA NA NA NA   
## E7\_low NA NA NA NA   
## E3\_4 NA NA NA NA   
## E3\_0 NA NA NA NA   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 602.10 on 699 degrees of freedom  
## Residual deviance: 452.36 on 682 degrees of freedom  
## AIC: 488.36  
##   
## Number of Fisher Scoring iterations: 16

To preserve only the significant variables: P3, P6, E4, E6, V1, V2, V3, V4, E9\_1

logistic.model.cur <- glm(IsAlert~P3 + P6 + E4 + E6 + V1 +V2 + V3 + V4 + E9\_1, data = training, family = binomial(link="logit"))  
summary(logistic.model.cur)

##   
## Call:  
## glm(formula = IsAlert ~ P3 + P6 + E4 + E6 + V1 + V2 + V3 + V4 +   
## E9\_1, family = binomial(link = "logit"), data = training)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.7354 0.2117 0.3612 0.6086 1.5315   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.755e+01 7.935e+02 -0.022 0.982356   
## P3 -1.734e-03 3.923e-04 -4.419 9.90e-06 \*\*\*  
## P6 3.365e-03 4.706e-03 0.715 0.474574   
## E4 3.596e-02 1.300e-02 2.767 0.005653 \*\*   
## E6 6.314e-02 1.528e-02 4.132 3.60e-05 \*\*\*  
## V1 1.599e-01 4.734e-02 3.377 0.000733 \*\*\*  
## V2 3.201e+00 1.025e+00 3.124 0.001786 \*\*   
## V3 -6.224e-04 4.015e-04 -1.550 0.121054   
## V4 -5.903e-01 1.251e-01 -4.720 2.36e-06 \*\*\*  
## E9\_1 -1.416e+01 7.935e+02 -0.018 0.985759   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 602.10 on 699 degrees of freedom  
## Residual deviance: 499.65 on 690 degrees of freedom  
## AIC: 519.65  
##   
## Number of Fisher Scoring iterations: 15

library(MASS)  
set.seed(10)  
birthwt.step <- stepAIC(logistic.model.cur, trace = 1, direction="both")

## Start: AIC=519.65  
## IsAlert ~ P3 + P6 + E4 + E6 + V1 + V2 + V3 + V4 + E9\_1  
##   
## Df Deviance AIC  
## - P6 1 500.18 518.18  
## - E9\_1 1 501.11 519.11  
## <none> 499.65 519.65  
## - V3 1 502.05 520.05  
## - E4 1 507.70 525.70  
## - V2 1 509.74 527.74  
## - V1 1 512.03 530.03  
## - E6 1 518.91 536.91  
## - P3 1 520.11 538.11  
## - V4 1 523.50 541.50  
##   
## Step: AIC=518.18  
## IsAlert ~ P3 + E4 + E6 + V1 + V2 + V3 + V4 + E9\_1  
##   
## Df Deviance AIC  
## - E9\_1 1 501.67 517.67  
## <none> 500.18 518.18  
## - V3 1 502.81 518.81  
## + P6 1 499.65 519.65  
## - E4 1 509.05 525.05  
## - V2 1 510.52 526.52  
## - V1 1 512.04 528.04  
## - E6 1 524.16 540.16  
## - V4 1 524.31 540.31  
## - P3 1 525.01 541.01  
##   
## Step: AIC=517.67  
## IsAlert ~ P3 + E4 + E6 + V1 + V2 + V3 + V4  
##   
## Df Deviance AIC  
## <none> 501.67 517.67  
## + E9\_1 1 500.18 518.18  
## - V3 1 504.60 518.60  
## + P6 1 501.11 519.11  
## - E4 1 510.48 524.48  
## - V1 1 513.18 527.18  
## - V2 1 513.32 527.32  
## - E6 1 525.59 539.59  
## - V4 1 526.80 540.80  
## - P3 1 527.33 541.33

birthwt.step$anova

## Stepwise Model Path   
## Analysis of Deviance Table  
##   
## Initial Model:  
## IsAlert ~ P3 + P6 + E4 + E6 + V1 + V2 + V3 + V4 + E9\_1  
##   
## Final Model:  
## IsAlert ~ P3 + E4 + E6 + V1 + V2 + V3 + V4  
##   
##   
## Step Df Deviance Resid. Df Resid. Dev AIC  
## 1 690 499.6475 519.6475  
## 2 - P6 1 0.5342874 691 500.1818 518.1818  
## 3 - E9\_1 1 1.4860335 692 501.6679 517.6679

the best Logistic Regression model has variables as following P3 + E4 + E6 + V1 + V2 + V3 + V4

logistic.model.best <- glm(IsAlert~P3 + E4 + E6 + V1 +V2 + V3 + V4, data = training, family = binomial(link="logit"))  
estimated\_isAlert<- predict(logistic.model.best, training, type = "response")  
misscls.train<-sum((estimated\_isAlert-training$IsAlert)^2)/ length(training$IsAlert)  
misscls.train

## [1] 0.1102006

presdicted.testing <- predict(logistic.model.best, testing, type = "response")  
misscls.test<-sum((presdicted.testing - testing$IsAlert)^2)/length(testing$IsAlert)  
misscls.test

## [1] 0.1379455

presdicted.testing.all <- predict(logistic.model.best, mydata\_lm, type = "response")  
misscls.test.all<-sum((presdicted.testing.all - mydata\_lm$IsAlert)^2)/length(mydata\_lm$IsAlert)  
misscls.test.all

## [1] 0.1185241

the missclasification rate for training data : 0.1102006 the missclasification rate for testing data : 0.1379455 the missclasification rate for entire data : 0.1185241

## Decision Tree

#new data set  
mydata\_tree <- newdata[ , 3:22]  
mydata\_tree <- mydata\_tree[1:1000,]  
#training data  
training <- mydata\_tree[1:700, ]  
#testing data  
testing <- mydata\_tree[701:1000,]  
  
summary(training)

## IsAlert P1 P2 P3   
## Min. :0.0000 Min. :25.38 Min. : 6.137 Min. : 516   
## 1st Qu.:1.0000 1st Qu.:34.15 1st Qu.:10.936 1st Qu.: 816   
## Median :1.0000 Median :35.24 Median :12.783 Median :1084   
## Mean :0.8457 Mean :35.89 Mean :12.844 Mean :1081   
## 3rd Qu.:1.0000 3rd Qu.:37.35 3rd Qu.:14.753 3rd Qu.:1400   
## Max. :1.0000 Max. :44.82 Max. :18.524 Max. :1816   
## P5 P6 E3 E4   
## Min. :0.2335 Min. :556.0 Min. :0.0000 Min. :-20.0000   
## 1st Qu.:0.2672 1st Qu.:592.0 1st Qu.:1.0000 1st Qu.: -6.0000   
## Median :0.2789 Median :612.0 Median :1.0000 Median : 0.0000   
## Mean :0.2782 Mean :617.3 Mean :0.9857 Mean : 0.6857   
## 3rd Qu.:0.2919 3rd Qu.:632.0 3rd Qu.:1.0000 3rd Qu.: 8.0000   
## Max. :0.3140 Max. :732.0 Max. :1.0000 Max. : 36.0000   
## E6 E7 E8 E9   
## Min. :260.0 Min. :0.0000 Min. :0.0000 Min. :0.0000   
## 1st Qu.:317.0 1st Qu.:1.0000 1st Qu.:1.0000 1st Qu.:1.0000   
## Median :320.0 Median :1.0000 Median :1.0000 Median :1.0000   
## Mean :319.9 Mean :0.9914 Mean :0.9914 Mean :0.9886   
## 3rd Qu.:324.0 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:1.0000   
## Max. :332.0 Max. :1.0000 Max. :1.0000 Max. :1.0000   
## V1 V2 V3 V4   
## Min. : 93.36 Min. :-0.6650 Min. : 240.0 Min. : 0.000   
## 1st Qu.: 97.01 1st Qu.: 0.0000 1st Qu.: 240.0 1st Qu.: 3.019   
## Median : 99.61 Median : 0.0000 Median : 496.0 Median : 4.506   
## Mean : 99.47 Mean : 0.0568 Mean : 536.8 Mean : 3.952   
## 3rd Qu.:102.30 3rd Qu.: 0.0700 3rd Qu.: 767.0 3rd Qu.: 4.506   
## Max. :104.97 Max. : 0.8750 Max. :1023.0 Max. :10.500   
## V5 V8 V10 V11   
## Min. :0.000000 Min. : 0.00 Min. :4 Min. :14.08   
## 1st Qu.:0.000000 1st Qu.: 7.95 1st Qu.:4 1st Qu.:14.83   
## Median :0.000000 Median :14.70 Median :4 Median :14.94   
## Mean :0.005714 Mean :13.70 Mean :4 Mean :14.95   
## 3rd Qu.:0.000000 3rd Qu.:20.60 3rd Qu.:4 3rd Qu.:15.05   
## Max. :1.000000 Max. :28.60 Max. :4 Max. :15.60

IsAlert=ifelse(mydata\_tree$IsAlert == 1," yes"," No ")  
mydata\_tree <- data.frame(mydata\_tree[,-1],IsAlert)  
summary(mydata\_tree)

## P1 P2 P3 P5   
## Min. :25.38 Min. : 6.137 Min. : 516 Min. :0.2335   
## 1st Qu.:34.46 1st Qu.:10.873 1st Qu.: 800 1st Qu.:0.2672   
## Median :35.81 Median :12.840 Median :1000 Median :0.2763   
## Mean :36.22 Mean :12.838 Mean :1048 Mean :0.2768   
## 3rd Qu.:37.59 3rd Qu.:14.629 3rd Qu.:1232 3rd Qu.:0.2896   
## Max. :44.82 Max. :20.608 Max. :1816 Max. :0.3321   
## P6 E3 E4 E6   
## Min. :556 Min. :0.00 Min. :-24.000 Min. :260   
## 1st Qu.:600 1st Qu.:1.00 1st Qu.: -6.000 1st Qu.:317   
## Median :616 Median :1.00 Median : 0.000 Median :320   
## Mean :625 Mean :0.99 Mean : 1.072 Mean :321   
## 3rd Qu.:632 3rd Qu.:1.00 3rd Qu.: 8.000 3rd Qu.:324   
## Max. :892 Max. :1.00 Max. : 36.000 Max. :393   
## E7 E8 E9 V1   
## Min. :0.00 Min. :0.000 Min. :0.000 Min. : 93.36   
## 1st Qu.:1.00 1st Qu.:1.000 1st Qu.:1.000 1st Qu.: 97.51   
## Median :1.00 Median :1.000 Median :1.000 Median :101.09   
## Mean :1.04 Mean :1.036 Mean :0.976 Mean :100.26   
## 3rd Qu.:1.00 3rd Qu.:1.000 3rd Qu.:1.000 3rd Qu.:102.86   
## Max. :6.00 Max. :7.000 Max. :1.000 Max. :106.11   
## V2 V3 V4 V5   
## Min. :-0.665000 Min. : 240.0 Min. : 0.000 Min. :0.000   
## 1st Qu.:-0.070000 1st Qu.: 255.0 1st Qu.: 1.488 1st Qu.:0.000   
## Median : 0.000000 Median : 752.0 Median : 3.019 Median :0.000   
## Mean : 0.005565 Mean : 595.7 Mean : 3.209 Mean :0.006   
## 3rd Qu.: 0.070000 3rd Qu.: 767.0 3rd Qu.: 4.506 3rd Qu.:0.000   
## Max. : 0.875000 Max. :1023.0 Max. :10.500 Max. :1.000   
## V8 V10 V11 IsAlert   
## Min. : 0.00 Min. :4 Min. :14.08 No :150   
## 1st Qu.: 8.40 1st Qu.:4 1st Qu.:14.87 yes:850   
## Median :15.20 Median :4 Median :15.02   
## Mean :14.01 Mean :4 Mean :15.01   
## 3rd Qu.:19.50 3rd Qu.:4 3rd Qu.:15.14   
## Max. :28.60 Max. :4 Max. :15.60

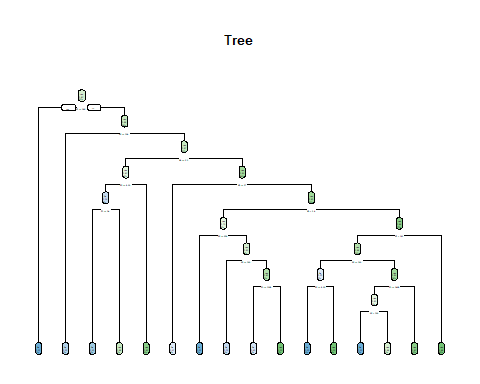
library(tree)

tree.mydata\_tree =tree(IsAlert~.,data=mydata\_tree )  
summary (tree.mydata\_tree )

##   
## Classification tree:  
## tree(formula = IsAlert ~ ., data = mydata\_tree)  
## Variables actually used in tree construction:  
## [1] "V1" "E6" "V4" "E4" "P1" "P6" "V8" "V2" "P5"  
## Number of terminal nodes: 25   
## Residual mean deviance: 0.098 = 95.55 / 975   
## Misclassification error rate: 0.023 = 23 / 1000

#plot(tree.mydata\_tree)  
#text(tree.mydata\_tree ,pretty =0)  
library(rpart.plot)

rpart.plot::rpart.plot(rpart(IsAlert~ .,data = mydata\_tree),main = "Tree")



ntrain=700  
set.seed(2020)  
train=sample(1: nrow(mydata\_tree ),ntrain )  
training = mydata\_tree[train,]  
testing=mydata\_tree[-train ,]  
tree.one =tree(IsAlert~.,data=training )  
summary (tree.one )

##   
## Classification tree:  
## tree(formula = IsAlert ~ ., data = training)  
## Variables actually used in tree construction:  
## [1] "V4" "V1" "E6" "E4" "P2" "V8" "P3" "P6" "V2" "P5" "P1"  
## Number of terminal nodes: 25   
## Residual mean deviance: 0.1242 = 83.81 / 675   
## Misclassification error rate: 0.02857 = 20 / 700

IsAlert.test=IsAlert[-train ]  
tree.pred=predict(tree.one, testing,type="class")  
table(tree.pred ,IsAlert.test)

## IsAlert.test  
## tree.pred No yes  
## No 29 12  
## yes 11 248

the missclasification rate for testing data :

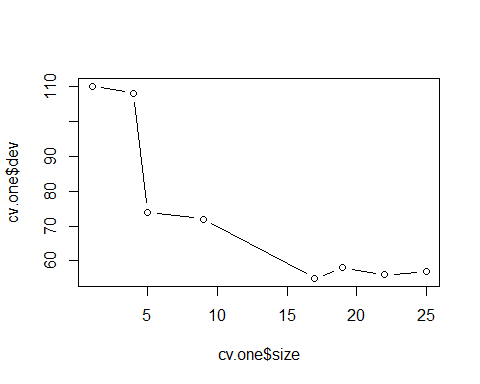
set.seed (2021)  
cv.one =cv.tree(tree.one ,FUN=prune.misclass )  
names(cv.one )

## [1] "size" "dev" "k" "method"

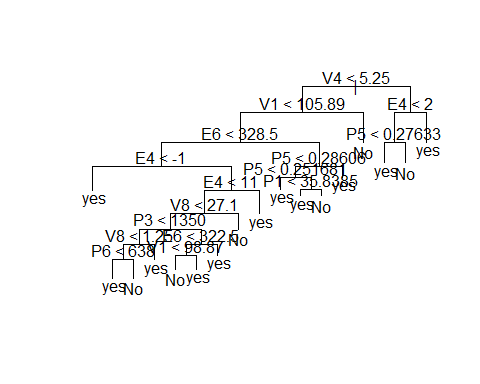
cv.one

## $size  
## [1] 25 22 19 17 9 5 4 1  
##   
## $dev  
## [1] 57 56 58 55 72 74 108 110  
##   
## $k  
## [1] -Inf 0.0000000 0.6666667 2.0000000 3.6250000 4.0000000  
## [7] 9.0000000 10.0000000  
##   
## $method  
## [1] "misclass"  
##   
## attr(,"class")  
## [1] "prune" "tree.sequence"

plot(cv.one$size ,cv.one$dev ,type="b")

 The best node size is 17.

mybest = 17  
prune.one = prune.misclass (tree.one ,best = mybest)  
plot(prune.one)  
text(prune.one ,pretty =0)



summary(prune.one)

##   
## Classification tree:  
## snip.tree(tree = tree.one, nodes = c(7L, 16L, 75L, 273L, 12L,   
## 549L))  
## Variables actually used in tree construction:  
## [1] "V4" "V1" "E6" "E4" "V8" "P3" "P6" "P5" "P1"  
## Number of terminal nodes: 17   
## Residual mean deviance: 0.243 = 166 / 683   
## Misclassification error rate: 0.03714 = 26 / 700

IsAlert.train=IsAlert[train ]  
tree.pred=predict(prune.one, training,type="class")  
table(tree.pred ,IsAlert.train)

## IsAlert.train  
## tree.pred No yes  
## No 89 5  
## yes 21 585

IsAlert.test=IsAlert[-train ]  
tree.pred=predict(prune.one, testing,type="class")  
table(tree.pred ,IsAlert.test)

## IsAlert.test  
## tree.pred No yes  
## No 26 5  
## yes 14 255

After we reduce the size of nodes,

the missclasification rate for training data:

the missclasification rate for testing data:

After reducing the size of nodes, the missclasification has reduced.

### Random Forest Boost

library(randomForest)  
bag.one =randomForest(IsAlert~.,data=mydata\_tree,subset =train , mtry=4, ntree =200,importance=TRUE)  
IsAlert.test=IsAlert[-train]  
tree.pred = predict ( bag.one ,newdata =mydata\_tree[-train ,],type="class")  
table(tree.pred , IsAlert.test)

## IsAlert.test  
## tree.pred No yes  
## No 33 0  
## yes 7 260

the missclasification rate for testing data:

## Esemble

library("mlbench")

## Warning: package 'mlbench' was built under R version 3.4.4

library("pROC")

## Warning: package 'pROC' was built under R version 3.4.4

## Type 'citation("pROC")' for a citation.

##   
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var

library(caret)  
#names(getModelInfo())

set.seed(2022)  
mydata\_last <- newdata[1:10000, ]  
mydata\_last$IsAlert <- as.factor(ifelse(mydata\_last$IsAlert == 0, "No", "Yes"))  
  
inTrain <- createDataPartition(y = mydata\_last$IsAlert, p = .75, list = FALSE)  
trainSet <- mydata\_last[ inTrain,]  
testSet <- mydata\_last[-inTrain,]  
  
#Defining the training controls for multiple models  
fitControl <- trainControl(  
 method = "cv",  
 number = 5,  
savePredictions = 'final',  
classProbs = T)  
  
#Defining the predictors and outcome  
#based on the result from logistic regression  
predictors<-c("P3", "E4", "E6", "V1", "V2", "V3", "V4")  
outcomeName<-'IsAlert'  
  
  
#Training the random forest model  
model\_rf<-train(trainSet[,predictors],trainSet[,outcomeName],method = 'rf')  
  
#Predicting using random forest model  
testSet$pred\_rf<-predict(object = model\_rf,testSet[,predictors],na.action = na.pass)  
  
  
#Checking the accuracy of the random forest model  
confusionMatrix(testSet$IsAlert,testSet$pred\_rf)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 504 21  
## Yes 1 1973  
##   
## Accuracy : 0.9912   
## 95% CI : (0.9867, 0.9945)  
## No Information Rate : 0.7979   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9731   
## Mcnemar's Test P-Value : 5.104e-05   
##   
## Sensitivity : 0.9980   
## Specificity : 0.9895   
## Pos Pred Value : 0.9600   
## Neg Pred Value : 0.9995   
## Prevalence : 0.2021   
## Detection Rate : 0.2017   
## Detection Prevalence : 0.2101   
## Balanced Accuracy : 0.9937   
##   
## 'Positive' Class : No   
##

#Training the knn model  
model\_knn<-train(trainSet[,predictors],trainSet[,outcomeName],method='knn')  
  
#Predicting using knn model  
testSet$pred\_knn<-predict(object = model\_knn,testSet[,predictors])  
  
#Checking the accuracy of the random forest model  
confusionMatrix(testSet$IsAlert,testSet$pred\_knn)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 377 148  
## Yes 71 1903  
##   
## Accuracy : 0.9124   
## 95% CI : (0.9006, 0.9232)  
## No Information Rate : 0.8207   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7209   
## Mcnemar's Test P-Value : 2.812e-07   
##   
## Sensitivity : 0.8415   
## Specificity : 0.9278   
## Pos Pred Value : 0.7181   
## Neg Pred Value : 0.9640   
## Prevalence : 0.1793   
## Detection Rate : 0.1509   
## Detection Prevalence : 0.2101   
## Balanced Accuracy : 0.8847   
##   
## 'Positive' Class : No   
##

#Training the Logistic regression model  
model\_lr<-train(trainSet[,predictors],trainSet[,outcomeName],method='glm')

#Predicting using Logistic regression model  
testSet$pred\_lr<-predict(object = model\_lr,testSet[,predictors])  
  
#Checking the accuracy of the random forest model  
confusionMatrix(testSet$IsAlert,testSet$pred\_lr)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 271 254  
## Yes 29 1945  
##   
## Accuracy : 0.8868   
## 95% CI : (0.8737, 0.8989)  
## No Information Rate : 0.88   
## P-Value [Acc > NIR] : 0.1548   
##   
## Kappa : 0.5951   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.9033   
## Specificity : 0.8845   
## Pos Pred Value : 0.5162   
## Neg Pred Value : 0.9853   
## Prevalence : 0.1200   
## Detection Rate : 0.1084   
## Detection Prevalence : 0.2101   
## Balanced Accuracy : 0.8939   
##   
## 'Positive' Class : No   
##

Random forst model has accuracy as 0.9912; knn model has accuracy as 0.9124; Logistic regression model has accuracy as 0.8868.

#Defining the training control  
fitControl <- trainControl(  
method = "cv",  
number = 10,  
savePredictions = 'final', # To save out of fold predictions for best parameter combinantions  
classProbs = T # To save the class probabilities of the out of fold predictions  
)  
  
  
#Training the random forest model  
model\_rf<-train(trainSet[,predictors],trainSet[,outcomeName],method='rf',trControl=fitControl,tuneLength=3)

#Training the knn model  
model\_knn<-train(trainSet[,predictors],trainSet[,outcomeName],method='knn',trControl=fitControl,tuneLength=3)

#Training the logistic regression model  
model\_lr<-train(trainSet[,predictors],trainSet[,outcomeName],method='glm',trControl=fitControl,tuneLength=3)

#Predicting the out of fold prediction probabilities for training data  
trainSet$OOF\_pred\_rf<-model\_rf$pred$Y[order(model\_rf$pred$rowIndex)]  
trainSet$OOF\_pred\_knn<-model\_knn$pred$Y[order(model\_knn$pred$rowIndex)]  
trainSet$OOF\_pred\_lr<-model\_lr$pred$Y[order(model\_lr$pred$rowIndex)]  
  
#Predicting probabilities for the test data  
testSet$OOF\_pred\_rf<-predict(model\_rf,testSet[predictors],type='prob')$Y  
testSet$OOF\_pred\_knn<-predict(model\_knn,testSet[predictors],type='prob')$Y  
testSet$OOF\_pred\_lr<-predict(model\_lr,testSet[predictors],type='prob')$Y  
  
#Predictors for top layer models   
predictors\_top<-c('OOF\_pred\_rf','OOF\_pred\_knn','OOF\_pred\_lr')   
  
#GBM as top layer model   
model\_gbm<-   
train(trainSet[,predictors\_top],trainSet[,outcomeName],method='gbm')

## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8380 nan 0.1000 0.0892  
## 2 0.7189 nan 0.1000 0.0599  
## 3 0.6286 nan 0.1000 0.0448  
## 4 0.5579 nan 0.1000 0.0355  
## 5 0.4975 nan 0.1000 0.0304  
## 6 0.4466 nan 0.1000 0.0254  
## 7 0.4043 nan 0.1000 0.0206  
## 8 0.3673 nan 0.1000 0.0190  
## 9 0.3345 nan 0.1000 0.0165  
## 10 0.3058 nan 0.1000 0.0141  
## 20 0.1507 nan 0.1000 0.0044  
## 40 0.0839 nan 0.1000 0.0005  
## 60 0.0738 nan 0.1000 0.0001  
## 80 0.0710 nan 0.1000 -0.0000  
## 100 0.0695 nan 0.1000 -0.0000  
## 120 0.0679 nan 0.1000 0.0001  
## 140 0.0665 nan 0.1000 -0.0001  
## 150 0.0660 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8388 nan 0.1000 0.0872  
## 2 0.7190 nan 0.1000 0.0609  
## 3 0.6279 nan 0.1000 0.0449  
## 4 0.5544 nan 0.1000 0.0359  
## 5 0.4942 nan 0.1000 0.0304  
## 6 0.4429 nan 0.1000 0.0259  
## 7 0.3989 nan 0.1000 0.0218  
## 8 0.3619 nan 0.1000 0.0186  
## 9 0.3291 nan 0.1000 0.0166  
## 10 0.3009 nan 0.1000 0.0141  
## 20 0.1467 nan 0.1000 0.0042  
## 40 0.0802 nan 0.1000 0.0003  
## 60 0.0661 nan 0.1000 -0.0000  
## 80 0.0619 nan 0.1000 -0.0001  
## 100 0.0583 nan 0.1000 -0.0001  
## 120 0.0538 nan 0.1000 0.0001  
## 140 0.0499 nan 0.1000 0.0000  
## 150 0.0491 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8368 nan 0.1000 0.0905  
## 2 0.7163 nan 0.1000 0.0606  
## 3 0.6251 nan 0.1000 0.0453  
## 4 0.5517 nan 0.1000 0.0364  
## 5 0.4910 nan 0.1000 0.0296  
## 6 0.4401 nan 0.1000 0.0250  
## 7 0.3969 nan 0.1000 0.0221  
## 8 0.3595 nan 0.1000 0.0190  
## 9 0.3272 nan 0.1000 0.0160  
## 10 0.2978 nan 0.1000 0.0142  
## 20 0.1429 nan 0.1000 0.0044  
## 40 0.0740 nan 0.1000 0.0007  
## 60 0.0590 nan 0.1000 -0.0001  
## 80 0.0545 nan 0.1000 -0.0001  
## 100 0.0494 nan 0.1000 0.0001  
## 120 0.0457 nan 0.1000 -0.0000  
## 140 0.0421 nan 0.1000 -0.0000  
## 150 0.0403 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8754 nan 0.1000 0.0867  
## 2 0.7536 nan 0.1000 0.0602  
## 3 0.6598 nan 0.1000 0.0466  
## 4 0.5831 nan 0.1000 0.0377  
## 5 0.5209 nan 0.1000 0.0317  
## 6 0.4671 nan 0.1000 0.0265  
## 7 0.4228 nan 0.1000 0.0220  
## 8 0.3843 nan 0.1000 0.0193  
## 9 0.3497 nan 0.1000 0.0171  
## 10 0.3207 nan 0.1000 0.0142  
## 20 0.1595 nan 0.1000 0.0041  
## 40 0.0878 nan 0.1000 0.0005  
## 60 0.0771 nan 0.1000 0.0000  
## 80 0.0736 nan 0.1000 -0.0000  
## 100 0.0717 nan 0.1000 -0.0001  
## 120 0.0702 nan 0.1000 0.0000  
## 140 0.0689 nan 0.1000 -0.0001  
## 150 0.0683 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8732 nan 0.1000 0.0882  
## 2 0.7499 nan 0.1000 0.0607  
## 3 0.6549 nan 0.1000 0.0463  
## 4 0.5789 nan 0.1000 0.0377  
## 5 0.5155 nan 0.1000 0.0309  
## 6 0.4623 nan 0.1000 0.0267  
## 7 0.4170 nan 0.1000 0.0226  
## 8 0.3779 nan 0.1000 0.0194  
## 9 0.3434 nan 0.1000 0.0172  
## 10 0.3137 nan 0.1000 0.0146  
## 20 0.1527 nan 0.1000 0.0041  
## 40 0.0835 nan 0.1000 0.0005  
## 60 0.0707 nan 0.1000 0.0001  
## 80 0.0661 nan 0.1000 -0.0001  
## 100 0.0602 nan 0.1000 -0.0000  
## 120 0.0570 nan 0.1000 0.0000  
## 140 0.0544 nan 0.1000 -0.0001  
## 150 0.0533 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8722 nan 0.1000 0.0899  
## 2 0.7481 nan 0.1000 0.0605  
## 3 0.6528 nan 0.1000 0.0479  
## 4 0.5762 nan 0.1000 0.0382  
## 5 0.5128 nan 0.1000 0.0312  
## 6 0.4595 nan 0.1000 0.0265  
## 7 0.4139 nan 0.1000 0.0229  
## 8 0.3743 nan 0.1000 0.0198  
## 9 0.3399 nan 0.1000 0.0173  
## 10 0.3104 nan 0.1000 0.0145  
## 20 0.1477 nan 0.1000 0.0044  
## 40 0.0764 nan 0.1000 0.0004  
## 60 0.0619 nan 0.1000 -0.0000  
## 80 0.0555 nan 0.1000 -0.0001  
## 100 0.0497 nan 0.1000 0.0000  
## 120 0.0465 nan 0.1000 0.0000  
## 140 0.0405 nan 0.1000 -0.0000  
## 150 0.0391 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8325 nan 0.1000 0.0872  
## 2 0.7131 nan 0.1000 0.0593  
## 3 0.6243 nan 0.1000 0.0449  
## 4 0.5512 nan 0.1000 0.0371  
## 5 0.4911 nan 0.1000 0.0302  
## 6 0.4412 nan 0.1000 0.0246  
## 7 0.3975 nan 0.1000 0.0217  
## 8 0.3600 nan 0.1000 0.0186  
## 9 0.3284 nan 0.1000 0.0158  
## 10 0.3005 nan 0.1000 0.0139  
## 20 0.1454 nan 0.1000 0.0039  
## 40 0.0770 nan 0.1000 0.0006  
## 60 0.0655 nan 0.1000 -0.0000  
## 80 0.0624 nan 0.1000 0.0000  
## 100 0.0605 nan 0.1000 -0.0001  
## 120 0.0591 nan 0.1000 -0.0000  
## 140 0.0584 nan 0.1000 -0.0000  
## 150 0.0581 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8305 nan 0.1000 0.0919  
## 2 0.7105 nan 0.1000 0.0601  
## 3 0.6192 nan 0.1000 0.0450  
## 4 0.5464 nan 0.1000 0.0370  
## 5 0.4863 nan 0.1000 0.0298  
## 6 0.4357 nan 0.1000 0.0256  
## 7 0.3918 nan 0.1000 0.0218  
## 8 0.3544 nan 0.1000 0.0186  
## 9 0.3218 nan 0.1000 0.0164  
## 10 0.2934 nan 0.1000 0.0139  
## 20 0.1411 nan 0.1000 0.0037  
## 40 0.0731 nan 0.1000 0.0002  
## 60 0.0597 nan 0.1000 0.0001  
## 80 0.0545 nan 0.1000 0.0001  
## 100 0.0496 nan 0.1000 -0.0001  
## 120 0.0482 nan 0.1000 0.0001  
## 140 0.0438 nan 0.1000 -0.0001  
## 150 0.0427 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8303 nan 0.1000 0.0872  
## 2 0.7098 nan 0.1000 0.0606  
## 3 0.6176 nan 0.1000 0.0456  
## 4 0.5448 nan 0.1000 0.0367  
## 5 0.4840 nan 0.1000 0.0299  
## 6 0.4331 nan 0.1000 0.0253  
## 7 0.3895 nan 0.1000 0.0217  
## 8 0.3518 nan 0.1000 0.0186  
## 9 0.3196 nan 0.1000 0.0164  
## 10 0.2910 nan 0.1000 0.0142  
## 20 0.1357 nan 0.1000 0.0043  
## 40 0.0656 nan 0.1000 0.0004  
## 60 0.0507 nan 0.1000 0.0001  
## 80 0.0437 nan 0.1000 -0.0001  
## 100 0.0379 nan 0.1000 -0.0000  
## 120 0.0353 nan 0.1000 -0.0001  
## 140 0.0330 nan 0.1000 0.0000  
## 150 0.0511 nan 0.1000 0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8413 nan 0.1000 0.0894  
## 2 0.7208 nan 0.1000 0.0593  
## 3 0.6284 nan 0.1000 0.0455  
## 4 0.5561 nan 0.1000 0.0362  
## 5 0.4953 nan 0.1000 0.0303  
## 6 0.4442 nan 0.1000 0.0259  
## 7 0.4018 nan 0.1000 0.0208  
## 8 0.3627 nan 0.1000 0.0192  
## 9 0.3299 nan 0.1000 0.0162  
## 10 0.3013 nan 0.1000 0.0143  
## 20 0.1442 nan 0.1000 0.0041  
## 40 0.0750 nan 0.1000 0.0005  
## 60 0.0662 nan 0.1000 0.0001  
## 80 0.0643 nan 0.1000 -0.0000  
## 100 0.0623 nan 0.1000 -0.0000  
## 120 0.0609 nan 0.1000 -0.0000  
## 140 0.0599 nan 0.1000 -0.0001  
## 150 0.0592 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8396 nan 0.1000 0.0899  
## 2 0.7188 nan 0.1000 0.0608  
## 3 0.6258 nan 0.1000 0.0460  
## 4 0.5519 nan 0.1000 0.0373  
## 5 0.4905 nan 0.1000 0.0304  
## 6 0.4383 nan 0.1000 0.0259  
## 7 0.3940 nan 0.1000 0.0221  
## 8 0.3563 nan 0.1000 0.0189  
## 9 0.3231 nan 0.1000 0.0166  
## 10 0.2942 nan 0.1000 0.0142  
## 20 0.1403 nan 0.1000 0.0039  
## 40 0.0733 nan 0.1000 0.0007  
## 60 0.0618 nan 0.1000 0.0001  
## 80 0.0569 nan 0.1000 -0.0001  
## 100 0.0480 nan 0.1000 -0.0002  
## 120 0.0439 nan 0.1000 -0.0000  
## 140 0.0414 nan 0.1000 -0.0002  
## 150 0.0407 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8389 nan 0.1000 0.0924  
## 2 0.7172 nan 0.1000 0.0610  
## 3 0.6249 nan 0.1000 0.0461  
## 4 0.5507 nan 0.1000 0.0366  
## 5 0.4891 nan 0.1000 0.0302  
## 6 0.4371 nan 0.1000 0.0258  
## 7 0.3929 nan 0.1000 0.0222  
## 8 0.3547 nan 0.1000 0.0191  
## 9 0.3214 nan 0.1000 0.0163  
## 10 0.2923 nan 0.1000 0.0145  
## 20 0.1349 nan 0.1000 0.0045  
## 40 0.0659 nan 0.1000 0.0004  
## 60 0.0525 nan 0.1000 -0.0016  
## 80 0.0445 nan 0.1000 -0.0000  
## 100 0.0405 nan 0.1000 -0.0001  
## 120 0.0361 nan 0.1000 0.0000  
## 140 0.0336 nan 0.1000 -0.0001  
## 150 0.0371 nan 0.1000 0.0002  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8513 nan 0.1000 0.0880  
## 2 0.7322 nan 0.1000 0.0586  
## 3 0.6403 nan 0.1000 0.0465  
## 4 0.5668 nan 0.1000 0.0358  
## 5 0.5055 nan 0.1000 0.0292  
## 6 0.4543 nan 0.1000 0.0259  
## 7 0.4112 nan 0.1000 0.0216  
## 8 0.3731 nan 0.1000 0.0189  
## 9 0.3412 nan 0.1000 0.0160  
## 10 0.3119 nan 0.1000 0.0143  
## 20 0.1544 nan 0.1000 0.0045  
## 40 0.0857 nan 0.1000 0.0006  
## 60 0.0752 nan 0.1000 -0.0000  
## 80 0.0713 nan 0.1000 0.0000  
## 100 0.0694 nan 0.1000 -0.0000  
## 120 0.0677 nan 0.1000 -0.0001  
## 140 0.0662 nan 0.1000 -0.0000  
## 150 0.0659 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8521 nan 0.1000 0.0896  
## 2 0.7301 nan 0.1000 0.0595  
## 3 0.6378 nan 0.1000 0.0452  
## 4 0.5638 nan 0.1000 0.0362  
## 5 0.5020 nan 0.1000 0.0308  
## 6 0.4507 nan 0.1000 0.0263  
## 7 0.4071 nan 0.1000 0.0216  
## 8 0.3684 nan 0.1000 0.0190  
## 9 0.3348 nan 0.1000 0.0166  
## 10 0.3056 nan 0.1000 0.0146  
## 20 0.1513 nan 0.1000 0.0040  
## 40 0.0827 nan 0.1000 0.0004  
## 60 0.0703 nan 0.1000 0.0001  
## 80 0.0635 nan 0.1000 -0.0001  
## 100 0.0585 nan 0.1000 -0.0001  
## 120 0.0546 nan 0.1000 -0.0000  
## 140 0.0520 nan 0.1000 -0.0001  
## 150 0.0502 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8501 nan 0.1000 0.0874  
## 2 0.7278 nan 0.1000 0.0616  
## 3 0.6354 nan 0.1000 0.0455  
## 4 0.5606 nan 0.1000 0.0371  
## 5 0.4993 nan 0.1000 0.0302  
## 6 0.4472 nan 0.1000 0.0260  
## 7 0.4029 nan 0.1000 0.0218  
## 8 0.3650 nan 0.1000 0.0188  
## 9 0.3315 nan 0.1000 0.0167  
## 10 0.3024 nan 0.1000 0.0144  
## 20 0.1443 nan 0.1000 0.0046  
## 40 0.0754 nan 0.1000 0.0004  
## 60 0.0622 nan 0.1000 0.0001  
## 80 0.0550 nan 0.1000 0.0000  
## 100 0.0459 nan 0.1000 0.0000  
## 120 0.0425 nan 0.1000 -0.0001  
## 140 0.0409 nan 0.1000 0.0001  
## 150 0.0391 nan 0.1000 0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8357 nan 0.1000 0.0877  
## 2 0.7160 nan 0.1000 0.0603  
## 3 0.6243 nan 0.1000 0.0451  
## 4 0.5515 nan 0.1000 0.0366  
## 5 0.4908 nan 0.1000 0.0295  
## 6 0.4404 nan 0.1000 0.0255  
## 7 0.3964 nan 0.1000 0.0216  
## 8 0.3581 nan 0.1000 0.0190  
## 9 0.3258 nan 0.1000 0.0156  
## 10 0.2971 nan 0.1000 0.0139  
## 20 0.1411 nan 0.1000 0.0046  
## 40 0.0723 nan 0.1000 0.0004  
## 60 0.0629 nan 0.1000 0.0000  
## 80 0.0607 nan 0.1000 0.0000  
## 100 0.0592 nan 0.1000 -0.0001  
## 120 0.0580 nan 0.1000 -0.0001  
## 140 0.0571 nan 0.1000 -0.0001  
## 150 0.0567 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8338 nan 0.1000 0.0902  
## 2 0.7128 nan 0.1000 0.0600  
## 3 0.6209 nan 0.1000 0.0465  
## 4 0.5476 nan 0.1000 0.0367  
## 5 0.4865 nan 0.1000 0.0312  
## 6 0.4352 nan 0.1000 0.0252  
## 7 0.3912 nan 0.1000 0.0217  
## 8 0.3537 nan 0.1000 0.0185  
## 9 0.3207 nan 0.1000 0.0164  
## 10 0.2918 nan 0.1000 0.0143  
## 20 0.1367 nan 0.1000 0.0044  
## 40 0.0696 nan 0.1000 0.0003  
## 60 0.0582 nan 0.1000 0.0001  
## 80 0.0520 nan 0.1000 -0.0001  
## 100 0.0495 nan 0.1000 -0.0002  
## 120 0.0471 nan 0.1000 -0.0001  
## 140 0.0458 nan 0.1000 -0.0001  
## 150 0.0449 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8340 nan 0.1000 0.0892  
## 2 0.7131 nan 0.1000 0.0611  
## 3 0.6207 nan 0.1000 0.0452  
## 4 0.5465 nan 0.1000 0.0366  
## 5 0.4852 nan 0.1000 0.0307  
## 6 0.4339 nan 0.1000 0.0248  
## 7 0.3896 nan 0.1000 0.0223  
## 8 0.3514 nan 0.1000 0.0192  
## 9 0.3182 nan 0.1000 0.0164  
## 10 0.2891 nan 0.1000 0.0143  
## 20 0.1341 nan 0.1000 0.0042  
## 40 0.0642 nan 0.1000 0.0005  
## 60 0.0513 nan 0.1000 0.0000  
## 80 0.0468 nan 0.1000 0.0000  
## 100 0.0434 nan 0.1000 0.0000  
## 120 0.0398 nan 0.1000 -0.0000  
## 140 0.0373 nan 0.1000 -0.0002  
## 150 0.0363 nan 0.1000 -0.0002  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8584 nan 0.1000 0.0904  
## 2 0.7349 nan 0.1000 0.0608  
## 3 0.6413 nan 0.1000 0.0488  
## 4 0.5653 nan 0.1000 0.0380  
## 5 0.5037 nan 0.1000 0.0305  
## 6 0.4510 nan 0.1000 0.0262  
## 7 0.4058 nan 0.1000 0.0224  
## 8 0.3667 nan 0.1000 0.0195  
## 9 0.3321 nan 0.1000 0.0172  
## 10 0.3032 nan 0.1000 0.0142  
## 20 0.1423 nan 0.1000 0.0043  
## 40 0.0709 nan 0.1000 0.0005  
## 60 0.0616 nan 0.1000 0.0000  
## 80 0.0590 nan 0.1000 -0.0002  
## 100 0.0573 nan 0.1000 -0.0001  
## 120 0.0556 nan 0.1000 -0.0000  
## 140 0.0549 nan 0.1000 -0.0000  
## 150 0.0545 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8567 nan 0.1000 0.0906  
## 2 0.7330 nan 0.1000 0.0606  
## 3 0.6384 nan 0.1000 0.0475  
## 4 0.5617 nan 0.1000 0.0369  
## 5 0.4988 nan 0.1000 0.0313  
## 6 0.4464 nan 0.1000 0.0262  
## 7 0.4010 nan 0.1000 0.0226  
## 8 0.3620 nan 0.1000 0.0190  
## 9 0.3280 nan 0.1000 0.0168  
## 10 0.2980 nan 0.1000 0.0145  
## 20 0.1368 nan 0.1000 0.0047  
## 40 0.0676 nan 0.1000 0.0004  
## 60 0.0554 nan 0.1000 0.0000  
## 80 0.0479 nan 0.1000 -0.0001  
## 100 0.0453 nan 0.1000 -0.0000  
## 120 0.0431 nan 0.1000 -0.0001  
## 140 0.0402 nan 0.1000 -0.0000  
## 150 0.0392 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8566 nan 0.1000 0.0915  
## 2 0.7327 nan 0.1000 0.0621  
## 3 0.6381 nan 0.1000 0.0468  
## 4 0.5614 nan 0.1000 0.0382  
## 5 0.4984 nan 0.1000 0.0318  
## 6 0.4450 nan 0.1000 0.0267  
## 7 0.3989 nan 0.1000 0.0229  
## 8 0.3597 nan 0.1000 0.0196  
## 9 0.3252 nan 0.1000 0.0170  
## 10 0.2952 nan 0.1000 0.0150  
## 20 0.1330 nan 0.1000 0.0040  
## 40 0.0621 nan 0.1000 0.0002  
## 60 0.0481 nan 0.1000 -0.0003  
## 80 0.0412 nan 0.1000 -0.0001  
## 100 0.0380 nan 0.1000 -0.0001  
## 120 0.0352 nan 0.1000 -0.0001  
## 140 0.0323 nan 0.1000 -0.0000  
## 150 0.0315 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8451 nan 0.1000 0.0880  
## 2 0.7241 nan 0.1000 0.0602  
## 3 0.6312 nan 0.1000 0.0466  
## 4 0.5564 nan 0.1000 0.0358  
## 5 0.4956 nan 0.1000 0.0293  
## 6 0.4446 nan 0.1000 0.0250  
## 7 0.4021 nan 0.1000 0.0216  
## 8 0.3635 nan 0.1000 0.0189  
## 9 0.3307 nan 0.1000 0.0163  
## 10 0.3031 nan 0.1000 0.0139  
## 20 0.1460 nan 0.1000 0.0049  
## 40 0.0743 nan 0.1000 0.0004  
## 60 0.0633 nan 0.1000 0.0001  
## 80 0.0611 nan 0.1000 -0.0000  
## 100 0.0593 nan 0.1000 -0.0000  
## 120 0.0582 nan 0.1000 -0.0000  
## 140 0.0571 nan 0.1000 -0.0001  
## 150 0.0568 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8426 nan 0.1000 0.0865  
## 2 0.7222 nan 0.1000 0.0596  
## 3 0.6295 nan 0.1000 0.0458  
## 4 0.5554 nan 0.1000 0.0368  
## 5 0.4941 nan 0.1000 0.0307  
## 6 0.4423 nan 0.1000 0.0255  
## 7 0.3976 nan 0.1000 0.0224  
## 8 0.3593 nan 0.1000 0.0191  
## 9 0.3261 nan 0.1000 0.0164  
## 10 0.2967 nan 0.1000 0.0146  
## 20 0.1405 nan 0.1000 0.0040  
## 40 0.0697 nan 0.1000 0.0006  
## 60 0.0573 nan 0.1000 0.0001  
## 80 0.0535 nan 0.1000 0.0001  
## 100 0.0502 nan 0.1000 -0.0000  
## 120 0.0485 nan 0.1000 -0.0000  
## 140 0.0462 nan 0.1000 0.0000  
## 150 0.0454 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8426 nan 0.1000 0.0910  
## 2 0.7205 nan 0.1000 0.0619  
## 3 0.6274 nan 0.1000 0.0468  
## 4 0.5528 nan 0.1000 0.0371  
## 5 0.4912 nan 0.1000 0.0305  
## 6 0.4390 nan 0.1000 0.0264  
## 7 0.3943 nan 0.1000 0.0221  
## 8 0.3562 nan 0.1000 0.0187  
## 9 0.3227 nan 0.1000 0.0168  
## 10 0.2933 nan 0.1000 0.0142  
## 20 0.1358 nan 0.1000 0.0042  
## 40 0.0647 nan 0.1000 0.0005  
## 60 0.0521 nan 0.1000 -0.0000  
## 80 0.0453 nan 0.1000 -0.0001  
## 100 0.0395 nan 0.1000 -0.0001  
## 120 0.0350 nan 0.1000 -0.0000  
## 140 0.0323 nan 0.1000 -0.0000  
## 150 0.1469 nan 0.1000 0.0002  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8549 nan 0.1000 0.0901  
## 2 0.7345 nan 0.1000 0.0600  
## 3 0.6420 nan 0.1000 0.0456  
## 4 0.5674 nan 0.1000 0.0368  
## 5 0.5069 nan 0.1000 0.0302  
## 6 0.4553 nan 0.1000 0.0255  
## 7 0.4124 nan 0.1000 0.0211  
## 8 0.3734 nan 0.1000 0.0190  
## 9 0.3394 nan 0.1000 0.0169  
## 10 0.3099 nan 0.1000 0.0146  
## 20 0.1546 nan 0.1000 0.0044  
## 40 0.0849 nan 0.1000 0.0005  
## 60 0.0739 nan 0.1000 0.0001  
## 80 0.0714 nan 0.1000 0.0001  
## 100 0.0702 nan 0.1000 -0.0001  
## 120 0.0693 nan 0.1000 -0.0000  
## 140 0.0682 nan 0.1000 -0.0000  
## 150 0.0677 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8540 nan 0.1000 0.0899  
## 2 0.7317 nan 0.1000 0.0608  
## 3 0.6383 nan 0.1000 0.0456  
## 4 0.5642 nan 0.1000 0.0366  
## 5 0.5027 nan 0.1000 0.0302  
## 6 0.4504 nan 0.1000 0.0261  
## 7 0.4065 nan 0.1000 0.0225  
## 8 0.3679 nan 0.1000 0.0197  
## 9 0.3342 nan 0.1000 0.0166  
## 10 0.3049 nan 0.1000 0.0144  
## 20 0.1500 nan 0.1000 0.0042  
## 40 0.0815 nan 0.1000 0.0003  
## 60 0.0661 nan 0.1000 0.0001  
## 80 0.0616 nan 0.1000 -0.0002  
## 100 0.0568 nan 0.1000 -0.0001  
## 120 0.0543 nan 0.1000 -0.0001  
## 140 0.0514 nan 0.1000 -0.0001  
## 150 0.0501 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8531 nan 0.1000 0.0899  
## 2 0.7308 nan 0.1000 0.0601  
## 3 0.6375 nan 0.1000 0.0468  
## 4 0.5634 nan 0.1000 0.0373  
## 5 0.5012 nan 0.1000 0.0312  
## 6 0.4486 nan 0.1000 0.0263  
## 7 0.4039 nan 0.1000 0.0221  
## 8 0.3647 nan 0.1000 0.0190  
## 9 0.3318 nan 0.1000 0.0161  
## 10 0.3030 nan 0.1000 0.0145  
## 20 0.1448 nan 0.1000 0.0042  
## 40 0.0739 nan 0.1000 0.0006  
## 60 0.0577 nan 0.1000 0.0000  
## 80 0.0513 nan 0.1000 -0.0002  
## 100 0.0482 nan 0.1000 -0.0001  
## 120 0.0455 nan 0.1000 0.0000  
## 140 0.0426 nan 0.1000 0.0001  
## 150 0.0407 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8526 nan 0.1000 0.0897  
## 2 0.7339 nan 0.1000 0.0601  
## 3 0.6417 nan 0.1000 0.0459  
## 4 0.5685 nan 0.1000 0.0367  
## 5 0.5065 nan 0.1000 0.0305  
## 6 0.4557 nan 0.1000 0.0254  
## 7 0.4115 nan 0.1000 0.0223  
## 8 0.3733 nan 0.1000 0.0187  
## 9 0.3402 nan 0.1000 0.0164  
## 10 0.3122 nan 0.1000 0.0138  
## 20 0.1524 nan 0.1000 0.0041  
## 40 0.0833 nan 0.1000 0.0005  
## 60 0.0726 nan 0.1000 0.0000  
## 80 0.0695 nan 0.1000 -0.0000  
## 100 0.0675 nan 0.1000 -0.0000  
## 120 0.0657 nan 0.1000 -0.0000  
## 140 0.0646 nan 0.1000 -0.0001  
## 150 0.0640 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8512 nan 0.1000 0.0886  
## 2 0.7304 nan 0.1000 0.0597  
## 3 0.6382 nan 0.1000 0.0462  
## 4 0.5635 nan 0.1000 0.0374  
## 5 0.5017 nan 0.1000 0.0300  
## 6 0.4493 nan 0.1000 0.0262  
## 7 0.4050 nan 0.1000 0.0224  
## 8 0.3670 nan 0.1000 0.0190  
## 9 0.3333 nan 0.1000 0.0168  
## 10 0.3040 nan 0.1000 0.0145  
## 20 0.1478 nan 0.1000 0.0039  
## 40 0.0805 nan 0.1000 0.0006  
## 60 0.0643 nan 0.1000 0.0003  
## 80 0.0584 nan 0.1000 -0.0001  
## 100 0.0543 nan 0.1000 -0.0001  
## 120 0.0503 nan 0.1000 -0.0001  
## 140 0.0478 nan 0.1000 -0.0000  
## 150 0.0464 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8512 nan 0.1000 0.0910  
## 2 0.7293 nan 0.1000 0.0606  
## 3 0.6363 nan 0.1000 0.0466  
## 4 0.5615 nan 0.1000 0.0385  
## 5 0.4997 nan 0.1000 0.0302  
## 6 0.4476 nan 0.1000 0.0260  
## 7 0.4039 nan 0.1000 0.0217  
## 8 0.3656 nan 0.1000 0.0189  
## 9 0.3316 nan 0.1000 0.0166  
## 10 0.3022 nan 0.1000 0.0144  
## 20 0.1436 nan 0.1000 0.0041  
## 40 0.0739 nan 0.1000 0.0002  
## 60 0.0596 nan 0.1000 -0.0001  
## 80 0.0622 nan 0.1000 -0.0000  
## 100 2701721.6555 nan 0.1000 0.0003  
## 120 inf nan 0.1000 nan  
## 140 inf nan 0.1000 nan  
## 150 inf nan 0.1000 nan  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8284 nan 0.1000 0.0878  
## 2 0.7109 nan 0.1000 0.0587  
## 3 0.6217 nan 0.1000 0.0439  
## 4 0.5493 nan 0.1000 0.0368  
## 5 0.4905 nan 0.1000 0.0295  
## 6 0.4409 nan 0.1000 0.0247  
## 7 0.3987 nan 0.1000 0.0213  
## 8 0.3621 nan 0.1000 0.0181  
## 9 0.3315 nan 0.1000 0.0154  
## 10 0.3036 nan 0.1000 0.0139  
## 20 0.1551 nan 0.1000 0.0044  
## 40 0.0903 nan 0.1000 0.0004  
## 60 0.0802 nan 0.1000 -0.0001  
## 80 0.0778 nan 0.1000 0.0000  
## 100 0.0759 nan 0.1000 -0.0001  
## 120 0.0746 nan 0.1000 0.0000  
## 140 0.0735 nan 0.1000 -0.0001  
## 150 0.0731 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8282 nan 0.1000 0.0885  
## 2 0.7089 nan 0.1000 0.0601  
## 3 0.6185 nan 0.1000 0.0451  
## 4 0.5465 nan 0.1000 0.0362  
## 5 0.4876 nan 0.1000 0.0292  
## 6 0.4381 nan 0.1000 0.0251  
## 7 0.3960 nan 0.1000 0.0209  
## 8 0.3594 nan 0.1000 0.0184  
## 9 0.3272 nan 0.1000 0.0162  
## 10 0.2993 nan 0.1000 0.0141  
## 20 0.1507 nan 0.1000 0.0038  
## 40 0.0861 nan 0.1000 0.0004  
## 60 0.0736 nan 0.1000 0.0001  
## 80 0.0679 nan 0.1000 -0.0001  
## 100 0.0636 nan 0.1000 -0.0000  
## 120 0.0585 nan 0.1000 -0.0000  
## 140 0.0561 nan 0.1000 -0.0001  
## 150 0.0547 nan 0.1000 0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8273 nan 0.1000 0.0915  
## 2 0.7082 nan 0.1000 0.0598  
## 3 0.6182 nan 0.1000 0.0450  
## 4 0.5463 nan 0.1000 0.0364  
## 5 0.4871 nan 0.1000 0.0301  
## 6 0.4367 nan 0.1000 0.0252  
## 7 0.3936 nan 0.1000 0.0211  
## 8 0.3563 nan 0.1000 0.0187  
## 9 0.3243 nan 0.1000 0.0157  
## 10 0.2966 nan 0.1000 0.0139  
## 20 0.1463 nan 0.1000 0.0039  
## 40 0.0805 nan 0.1000 0.0005  
## 60 0.0643 nan 0.1000 -0.0001  
## 80 0.0582 nan 0.1000 0.0002  
## 100 0.0540 nan 0.1000 -0.0001  
## 120 0.0516 nan 0.1000 -0.0000  
## 140 0.0476 nan 0.1000 0.0001  
## 150 0.0453 nan 0.1000 0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8434 nan 0.1000 0.0908  
## 2 0.7211 nan 0.1000 0.0608  
## 3 0.6279 nan 0.1000 0.0475  
## 4 0.5537 nan 0.1000 0.0380  
## 5 0.4920 nan 0.1000 0.0304  
## 6 0.4405 nan 0.1000 0.0258  
## 7 0.3965 nan 0.1000 0.0225  
## 8 0.3590 nan 0.1000 0.0189  
## 9 0.3252 nan 0.1000 0.0166  
## 10 0.2965 nan 0.1000 0.0146  
## 20 0.1374 nan 0.1000 0.0043  
## 40 0.0672 nan 0.1000 0.0006  
## 60 0.0562 nan 0.1000 0.0001  
## 80 0.0537 nan 0.1000 -0.0000  
## 100 0.0521 nan 0.1000 -0.0000  
## 120 0.0509 nan 0.1000 -0.0000  
## 140 0.0498 nan 0.1000 -0.0001  
## 150 0.0492 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8433 nan 0.1000 0.0941  
## 2 0.7209 nan 0.1000 0.0611  
## 3 0.6273 nan 0.1000 0.0470  
## 4 0.5527 nan 0.1000 0.0368  
## 5 0.4902 nan 0.1000 0.0312  
## 6 0.4384 nan 0.1000 0.0256  
## 7 0.3936 nan 0.1000 0.0226  
## 8 0.3553 nan 0.1000 0.0188  
## 9 0.3215 nan 0.1000 0.0169  
## 10 0.2924 nan 0.1000 0.0143  
## 20 0.1357 nan 0.1000 0.0042  
## 40 0.0628 nan 0.1000 0.0004  
## 60 0.0490 nan 0.1000 0.0003  
## 80 0.0450 nan 0.1000 -0.0000  
## 100 0.0420 nan 0.1000 -0.0000  
## 120 0.0397 nan 0.1000 0.0000  
## 140 0.0381 nan 0.1000 -0.0000  
## 150 0.0372 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8417 nan 0.1000 0.0892  
## 2 0.7183 nan 0.1000 0.0616  
## 3 0.6247 nan 0.1000 0.0465  
## 4 0.5496 nan 0.1000 0.0374  
## 5 0.4871 nan 0.1000 0.0300  
## 6 0.4350 nan 0.1000 0.0256  
## 7 0.3901 nan 0.1000 0.0218  
## 8 0.3516 nan 0.1000 0.0192  
## 9 0.3178 nan 0.1000 0.0163  
## 10 0.2888 nan 0.1000 0.0142  
## 20 0.1291 nan 0.1000 0.0043  
## 40 0.0563 nan 0.1000 0.0004  
## 60 0.0432 nan 0.1000 0.0000  
## 80 0.0391 nan 0.1000 -0.0001  
## 100 0.0362 nan 0.1000 -0.0001  
## 120 0.0336 nan 0.1000 -0.0001  
## 140 0.0316 nan 0.1000 0.0000  
## 150 0.0300 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8462 nan 0.1000 0.0869  
## 2 0.7257 nan 0.1000 0.0592  
## 3 0.6346 nan 0.1000 0.0447  
## 4 0.5607 nan 0.1000 0.0365  
## 5 0.4996 nan 0.1000 0.0302  
## 6 0.4493 nan 0.1000 0.0256  
## 7 0.4064 nan 0.1000 0.0212  
## 8 0.3667 nan 0.1000 0.0193  
## 9 0.3345 nan 0.1000 0.0157  
## 10 0.3052 nan 0.1000 0.0149  
## 20 0.1470 nan 0.1000 0.0044  
## 40 0.0791 nan 0.1000 0.0006  
## 60 0.0685 nan 0.1000 0.0001  
## 80 0.0663 nan 0.1000 -0.0001  
## 100 0.0643 nan 0.1000 -0.0000  
## 120 0.0627 nan 0.1000 -0.0000  
## 140 0.0614 nan 0.1000 -0.0000  
## 150 0.0608 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8442 nan 0.1000 0.0886  
## 2 0.7230 nan 0.1000 0.0599  
## 3 0.6308 nan 0.1000 0.0458  
## 4 0.5568 nan 0.1000 0.0360  
## 5 0.4961 nan 0.1000 0.0297  
## 6 0.4435 nan 0.1000 0.0261  
## 7 0.3991 nan 0.1000 0.0221  
## 8 0.3616 nan 0.1000 0.0187  
## 9 0.3282 nan 0.1000 0.0164  
## 10 0.2992 nan 0.1000 0.0146  
## 20 0.1439 nan 0.1000 0.0040  
## 40 0.0754 nan 0.1000 0.0005  
## 60 0.0607 nan 0.1000 0.0001  
## 80 0.0560 nan 0.1000 0.0001  
## 100 0.0532 nan 0.1000 -0.0001  
## 120 0.0503 nan 0.1000 0.0000  
## 140 0.0478 nan 0.1000 0.0000  
## 150 0.0470 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8441 nan 0.1000 0.0898  
## 2 0.7222 nan 0.1000 0.0612  
## 3 0.6294 nan 0.1000 0.0456  
## 4 0.5547 nan 0.1000 0.0361  
## 5 0.4931 nan 0.1000 0.0311  
## 6 0.4414 nan 0.1000 0.0258  
## 7 0.3972 nan 0.1000 0.0220  
## 8 0.3585 nan 0.1000 0.0191  
## 9 0.3251 nan 0.1000 0.0165  
## 10 0.2958 nan 0.1000 0.0145  
## 20 0.1366 nan 0.1000 0.0042  
## 40 0.0684 nan 0.1000 0.0007  
## 60 0.0540 nan 0.1000 0.0001  
## 80 0.0484 nan 0.1000 -0.0000  
## 100 0.0449 nan 0.1000 -0.0001  
## 120 0.0402 nan 0.1000 -0.0001  
## 140 0.0382 nan 0.1000 0.0000  
## 150 0.0370 nan 0.1000 0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8525 nan 0.1000 0.0882  
## 2 0.7313 nan 0.1000 0.0604  
## 3 0.6381 nan 0.1000 0.0464  
## 4 0.5636 nan 0.1000 0.0371  
## 5 0.5024 nan 0.1000 0.0306  
## 6 0.4507 nan 0.1000 0.0260  
## 7 0.4066 nan 0.1000 0.0220  
## 8 0.3679 nan 0.1000 0.0192  
## 9 0.3356 nan 0.1000 0.0160  
## 10 0.3064 nan 0.1000 0.0147  
## 20 0.1478 nan 0.1000 0.0041  
## 40 0.0805 nan 0.1000 0.0005  
## 60 0.0707 nan 0.1000 -0.0000  
## 80 0.0680 nan 0.1000 -0.0000  
## 100 0.0658 nan 0.1000 -0.0001  
## 120 0.0642 nan 0.1000 -0.0000  
## 140 0.0628 nan 0.1000 -0.0000  
## 150 0.0623 nan 0.1000 0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8511 nan 0.1000 0.0886  
## 2 0.7289 nan 0.1000 0.0617  
## 3 0.6362 nan 0.1000 0.0476  
## 4 0.5614 nan 0.1000 0.0375  
## 5 0.4997 nan 0.1000 0.0310  
## 6 0.4475 nan 0.1000 0.0262  
## 7 0.4026 nan 0.1000 0.0222  
## 8 0.3646 nan 0.1000 0.0190  
## 9 0.3309 nan 0.1000 0.0168  
## 10 0.3027 nan 0.1000 0.0143  
## 20 0.1449 nan 0.1000 0.0040  
## 40 0.0747 nan 0.1000 0.0006  
## 60 0.0604 nan 0.1000 -0.0004  
## 80 0.0556 nan 0.1000 -0.0001  
## 100 0.0515 nan 0.1000 -0.0003  
## 120 0.0462 nan 0.1000 -0.0001  
## 140 0.0444 nan 0.1000 -0.0001  
## 150 0.0432 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8495 nan 0.1000 0.0902  
## 2 0.7275 nan 0.1000 0.0611  
## 3 0.6340 nan 0.1000 0.0458  
## 4 0.5590 nan 0.1000 0.0367  
## 5 0.4973 nan 0.1000 0.0315  
## 6 0.4453 nan 0.1000 0.0263  
## 7 0.4009 nan 0.1000 0.0220  
## 8 0.3620 nan 0.1000 0.0192  
## 9 0.3277 nan 0.1000 0.0169  
## 10 0.2974 nan 0.1000 0.0147  
## 20 0.1405 nan 0.1000 0.0041  
## 40 0.0694 nan 0.1000 0.0005  
## 60 0.0548 nan 0.1000 0.0000  
## 80 0.0475 nan 0.1000 -0.0000  
## 100 0.0427 nan 0.1000 -0.0000  
## 120 0.0387 nan 0.1000 -0.0001  
## 140 0.0361 nan 0.1000 -0.0001  
## 150 0.0345 nan 0.1000 -0.0004  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8408 nan 0.1000 0.0879  
## 2 0.7230 nan 0.1000 0.0598  
## 3 0.6311 nan 0.1000 0.0458  
## 4 0.5573 nan 0.1000 0.0372  
## 5 0.4957 nan 0.1000 0.0304  
## 6 0.4451 nan 0.1000 0.0255  
## 7 0.4015 nan 0.1000 0.0222  
## 8 0.3637 nan 0.1000 0.0187  
## 9 0.3306 nan 0.1000 0.0160  
## 10 0.3028 nan 0.1000 0.0134  
## 20 0.1483 nan 0.1000 0.0043  
## 40 0.0822 nan 0.1000 0.0004  
## 60 0.0725 nan 0.1000 -0.0000  
## 80 0.0698 nan 0.1000 -0.0001  
## 100 0.0679 nan 0.1000 -0.0000  
## 120 0.0664 nan 0.1000 -0.0000  
## 140 0.0654 nan 0.1000 -0.0001  
## 150 0.0648 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8397 nan 0.1000 0.0895  
## 2 0.7193 nan 0.1000 0.0603  
## 3 0.6276 nan 0.1000 0.0465  
## 4 0.5541 nan 0.1000 0.0364  
## 5 0.4936 nan 0.1000 0.0307  
## 6 0.4417 nan 0.1000 0.0254  
## 7 0.3980 nan 0.1000 0.0222  
## 8 0.3604 nan 0.1000 0.0186  
## 9 0.3277 nan 0.1000 0.0162  
## 10 0.2990 nan 0.1000 0.0141  
## 20 0.1463 nan 0.1000 0.0037  
## 40 0.0772 nan 0.1000 0.0006  
## 60 0.0640 nan 0.1000 0.0002  
## 80 0.0602 nan 0.1000 -0.0001  
## 100 0.0547 nan 0.1000 -0.0000  
## 120 0.0519 nan 0.1000 -0.0001  
## 140 0.0492 nan 0.1000 -0.0000  
## 150 0.0516 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8388 nan 0.1000 0.0899  
## 2 0.7179 nan 0.1000 0.0594  
## 3 0.6266 nan 0.1000 0.0463  
## 4 0.5531 nan 0.1000 0.0373  
## 5 0.4922 nan 0.1000 0.0291  
## 6 0.4401 nan 0.1000 0.0259  
## 7 0.3962 nan 0.1000 0.0217  
## 8 0.3584 nan 0.1000 0.0184  
## 9 0.3254 nan 0.1000 0.0164  
## 10 0.2970 nan 0.1000 0.0139  
## 20 0.1428 nan 0.1000 0.0045  
## 40 0.0739 nan 0.1000 0.0005  
## 60 0.0590 nan 0.1000 0.0001  
## 80 0.0527 nan 0.1000 0.0000  
## 100 0.0487 nan 0.1000 0.0000  
## 120 0.0457 nan 0.1000 -0.0001  
## 140 0.0430 nan 0.1000 -0.0000  
## 150 0.0423 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8637 nan 0.1000 0.0902  
## 2 0.7420 nan 0.1000 0.0595  
## 3 0.6493 nan 0.1000 0.0477  
## 4 0.5752 nan 0.1000 0.0365  
## 5 0.5153 nan 0.1000 0.0293  
## 6 0.4624 nan 0.1000 0.0266  
## 7 0.4174 nan 0.1000 0.0224  
## 8 0.3790 nan 0.1000 0.0192  
## 9 0.3450 nan 0.1000 0.0169  
## 10 0.3166 nan 0.1000 0.0138  
## 20 0.1528 nan 0.1000 0.0044  
## 40 0.0826 nan 0.1000 0.0006  
## 60 0.0729 nan 0.1000 0.0000  
## 80 0.0701 nan 0.1000 -0.0001  
## 100 0.0683 nan 0.1000 0.0000  
## 120 0.0670 nan 0.1000 -0.0001  
## 140 0.0656 nan 0.1000 -0.0001  
## 150 0.0650 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8615 nan 0.1000 0.0894  
## 2 0.7396 nan 0.1000 0.0619  
## 3 0.6459 nan 0.1000 0.0463  
## 4 0.5703 nan 0.1000 0.0367  
## 5 0.5079 nan 0.1000 0.0317  
## 6 0.4557 nan 0.1000 0.0259  
## 7 0.4106 nan 0.1000 0.0226  
## 8 0.3717 nan 0.1000 0.0192  
## 9 0.3375 nan 0.1000 0.0170  
## 10 0.3082 nan 0.1000 0.0148  
## 20 0.1503 nan 0.1000 0.0040  
## 40 0.0795 nan 0.1000 0.0003  
## 60 0.0670 nan 0.1000 0.0000  
## 80 0.0621 nan 0.1000 0.0000  
## 100 0.0578 nan 0.1000 -0.0001  
## 120 0.0558 nan 0.1000 -0.0001  
## 140 0.0539 nan 0.1000 -0.0000  
## 150 0.0519 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8610 nan 0.1000 0.0878  
## 2 0.7389 nan 0.1000 0.0602  
## 3 0.6449 nan 0.1000 0.0461  
## 4 0.5693 nan 0.1000 0.0377  
## 5 0.5065 nan 0.1000 0.0312  
## 6 0.4538 nan 0.1000 0.0262  
## 7 0.4087 nan 0.1000 0.0224  
## 8 0.3695 nan 0.1000 0.0196  
## 9 0.3359 nan 0.1000 0.0170  
## 10 0.3056 nan 0.1000 0.0152  
## 20 0.1455 nan 0.1000 0.0041  
## 40 0.0721 nan 0.1000 0.0005  
## 60 0.0586 nan 0.1000 -0.0001  
## 80 0.0527 nan 0.1000 0.0001  
## 100 0.0489 nan 0.1000 -0.0001  
## 120 0.0527 nan 0.1000 -0.0025  
## 140 0.0410 nan 0.1000 0.0000  
## 150 0.0395 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8499 nan 0.1000 0.0880  
## 2 0.7294 nan 0.1000 0.0618  
## 3 0.6371 nan 0.1000 0.0459  
## 4 0.5631 nan 0.1000 0.0365  
## 5 0.5009 nan 0.1000 0.0306  
## 6 0.4497 nan 0.1000 0.0261  
## 7 0.4052 nan 0.1000 0.0223  
## 8 0.3674 nan 0.1000 0.0189  
## 9 0.3347 nan 0.1000 0.0165  
## 10 0.3057 nan 0.1000 0.0144  
## 20 0.1503 nan 0.1000 0.0046  
## 40 0.0829 nan 0.1000 0.0005  
## 60 0.0735 nan 0.1000 -0.0001  
## 80 0.0704 nan 0.1000 -0.0001  
## 100 0.0688 nan 0.1000 -0.0000  
## 120 0.0676 nan 0.1000 -0.0001  
## 140 0.0663 nan 0.1000 0.0000  
## 150 0.0659 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8480 nan 0.1000 0.0899  
## 2 0.7269 nan 0.1000 0.0615  
## 3 0.6338 nan 0.1000 0.0464  
## 4 0.5594 nan 0.1000 0.0374  
## 5 0.4972 nan 0.1000 0.0305  
## 6 0.4455 nan 0.1000 0.0262  
## 7 0.4013 nan 0.1000 0.0220  
## 8 0.3635 nan 0.1000 0.0186  
## 9 0.3305 nan 0.1000 0.0165  
## 10 0.3019 nan 0.1000 0.0146  
## 20 0.1463 nan 0.1000 0.0042  
## 40 0.0781 nan 0.1000 0.0006  
## 60 0.0650 nan 0.1000 0.0001  
## 80 0.0572 nan 0.1000 -0.0000  
## 100 0.0520 nan 0.1000 -0.0002  
## 120 0.0490 nan 0.1000 0.0000  
## 140 0.0458 nan 0.1000 -0.0001  
## 150 0.0450 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8483 nan 0.1000 0.0885  
## 2 0.7256 nan 0.1000 0.0608  
## 3 0.6329 nan 0.1000 0.0473  
## 4 0.5587 nan 0.1000 0.0379  
## 5 0.4970 nan 0.1000 0.0312  
## 6 0.4445 nan 0.1000 0.0265  
## 7 0.4004 nan 0.1000 0.0217  
## 8 0.3621 nan 0.1000 0.0194  
## 9 0.3287 nan 0.1000 0.0166  
## 10 0.2996 nan 0.1000 0.0144  
## 20 0.1429 nan 0.1000 0.0042  
## 40 0.0737 nan 0.1000 0.0005  
## 60 0.0526 nan 0.1000 0.0004  
## 80 0.0458 nan 0.1000 0.0000  
## 100 0.0410 nan 0.1000 -0.0000  
## 120 0.0379 nan 0.1000 -0.0000  
## 140 0.0351 nan 0.1000 -0.0000  
## 150 0.0340 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8527 nan 0.1000 0.0895  
## 2 0.7293 nan 0.1000 0.0613  
## 3 0.6355 nan 0.1000 0.0466  
## 4 0.5605 nan 0.1000 0.0370  
## 5 0.4980 nan 0.1000 0.0312  
## 6 0.4464 nan 0.1000 0.0257  
## 7 0.4028 nan 0.1000 0.0218  
## 8 0.3644 nan 0.1000 0.0191  
## 9 0.3302 nan 0.1000 0.0170  
## 10 0.3011 nan 0.1000 0.0147  
## 20 0.1383 nan 0.1000 0.0049  
## 40 0.0662 nan 0.1000 0.0005  
## 60 0.0561 nan 0.1000 -0.0001  
## 80 0.0531 nan 0.1000 0.0000  
## 100 0.0507 nan 0.1000 -0.0001  
## 120 0.0499 nan 0.1000 -0.0001  
## 140 0.0485 nan 0.1000 -0.0000  
## 150 0.0480 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8513 nan 0.1000 0.0880  
## 2 0.7277 nan 0.1000 0.0620  
## 3 0.6338 nan 0.1000 0.0468  
## 4 0.5581 nan 0.1000 0.0383  
## 5 0.4957 nan 0.1000 0.0308  
## 6 0.4433 nan 0.1000 0.0265  
## 7 0.3989 nan 0.1000 0.0220  
## 8 0.3595 nan 0.1000 0.0195  
## 9 0.3254 nan 0.1000 0.0173  
## 10 0.2955 nan 0.1000 0.0147  
## 20 0.1348 nan 0.1000 0.0041  
## 40 0.0644 nan 0.1000 0.0003  
## 60 0.0519 nan 0.1000 -0.0000  
## 80 0.0474 nan 0.1000 0.0000  
## 100 0.0439 nan 0.1000 -0.0000  
## 120 0.0427 nan 0.1000 -0.0001  
## 140 0.0412 nan 0.1000 -0.0001  
## 150 0.0397 nan 0.1000 -0.0002  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8503 nan 0.1000 0.0897  
## 2 0.7264 nan 0.1000 0.0620  
## 3 0.6325 nan 0.1000 0.0480  
## 4 0.5570 nan 0.1000 0.0373  
## 5 0.4937 nan 0.1000 0.0314  
## 6 0.4408 nan 0.1000 0.0261  
## 7 0.3953 nan 0.1000 0.0222  
## 8 0.3559 nan 0.1000 0.0193  
## 9 0.3213 nan 0.1000 0.0170  
## 10 0.2912 nan 0.1000 0.0149  
## 20 0.1286 nan 0.1000 0.0043  
## 40 0.0557 nan 0.1000 0.0004  
## 60 0.0428 nan 0.1000 -0.0000  
## 80 0.0385 nan 0.1000 0.0000  
## 100 0.0379 nan 0.1000 0.0002  
## 120 0.0345 nan 0.1000 -0.0000  
## 140 0.0315 nan 0.1000 -0.0001  
## 150 0.0301 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8660 nan 0.1000 0.0868  
## 2 0.7432 nan 0.1000 0.0605  
## 3 0.6502 nan 0.1000 0.0459  
## 4 0.5742 nan 0.1000 0.0383  
## 5 0.5118 nan 0.1000 0.0313  
## 6 0.4596 nan 0.1000 0.0261  
## 7 0.4145 nan 0.1000 0.0223  
## 8 0.3763 nan 0.1000 0.0192  
## 9 0.3432 nan 0.1000 0.0162  
## 10 0.3131 nan 0.1000 0.0148  
## 20 0.1504 nan 0.1000 0.0042  
## 40 0.0782 nan 0.1000 0.0006  
## 60 0.0681 nan 0.1000 0.0001  
## 80 0.0648 nan 0.1000 0.0000  
## 100 0.0625 nan 0.1000 0.0001  
## 120 0.0611 nan 0.1000 -0.0000  
## 140 0.0599 nan 0.1000 -0.0000  
## 150 0.0594 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8644 nan 0.1000 0.0891  
## 2 0.7417 nan 0.1000 0.0627  
## 3 0.6480 nan 0.1000 0.0464  
## 4 0.5723 nan 0.1000 0.0375  
## 5 0.5088 nan 0.1000 0.0314  
## 6 0.4551 nan 0.1000 0.0270  
## 7 0.4094 nan 0.1000 0.0225  
## 8 0.3693 nan 0.1000 0.0195  
## 9 0.3354 nan 0.1000 0.0167  
## 10 0.3058 nan 0.1000 0.0150  
## 20 0.1450 nan 0.1000 0.0043  
## 40 0.0745 nan 0.1000 0.0006  
## 60 0.0622 nan 0.1000 0.0000  
## 80 0.0580 nan 0.1000 0.0002  
## 100 0.0523 nan 0.1000 -0.0000  
## 120 0.0499 nan 0.1000 0.0000  
## 140 0.0463 nan 0.1000 -0.0002  
## 150 0.0452 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8630 nan 0.1000 0.0890  
## 2 0.7396 nan 0.1000 0.0618  
## 3 0.6445 nan 0.1000 0.0473  
## 4 0.5685 nan 0.1000 0.0379  
## 5 0.5048 nan 0.1000 0.0313  
## 6 0.4518 nan 0.1000 0.0264  
## 7 0.4059 nan 0.1000 0.0228  
## 8 0.3665 nan 0.1000 0.0199  
## 9 0.3325 nan 0.1000 0.0167  
## 10 0.3024 nan 0.1000 0.0150  
## 20 0.1404 nan 0.1000 0.0042  
## 40 0.0680 nan 0.1000 0.0004  
## 60 0.0551 nan 0.1000 0.0001  
## 80 0.0748 nan 0.1000 0.0001  
## 100 0.0712 nan 0.1000 0.0002  
## 120 inf nan 0.1000 nan  
## 140 inf nan 0.1000 nan  
## 150 inf nan 0.1000 nan  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8414 nan 0.1000 0.0860  
## 2 0.7227 nan 0.1000 0.0582  
## 3 0.6315 nan 0.1000 0.0448  
## 4 0.5588 nan 0.1000 0.0363  
## 5 0.4996 nan 0.1000 0.0296  
## 6 0.4486 nan 0.1000 0.0256  
## 7 0.4061 nan 0.1000 0.0216  
## 8 0.3678 nan 0.1000 0.0189  
## 9 0.3360 nan 0.1000 0.0156  
## 10 0.3068 nan 0.1000 0.0145  
## 20 0.1501 nan 0.1000 0.0040  
## 40 0.0821 nan 0.1000 0.0005  
## 60 0.0717 nan 0.1000 0.0001  
## 80 0.0693 nan 0.1000 -0.0001  
## 100 0.0676 nan 0.1000 -0.0001  
## 120 0.0664 nan 0.1000 -0.0000  
## 140 0.0653 nan 0.1000 -0.0000  
## 150 0.0647 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8412 nan 0.1000 0.0908  
## 2 0.7208 nan 0.1000 0.0607  
## 3 0.6290 nan 0.1000 0.0464  
## 4 0.5549 nan 0.1000 0.0363  
## 5 0.4938 nan 0.1000 0.0307  
## 6 0.4424 nan 0.1000 0.0251  
## 7 0.3988 nan 0.1000 0.0217  
## 8 0.3614 nan 0.1000 0.0186  
## 9 0.3287 nan 0.1000 0.0163  
## 10 0.3000 nan 0.1000 0.0141  
## 20 0.1463 nan 0.1000 0.0039  
## 40 0.0781 nan 0.1000 0.0005  
## 60 0.0656 nan 0.1000 0.0002  
## 80 0.0636 nan 0.1000 -0.0042  
## 100 0.0533 nan 0.1000 0.0001  
## 120 0.0498 nan 0.1000 -0.0001  
## 140 0.0458 nan 0.1000 -0.0001  
## 150 0.0471 nan 0.1000 -0.0023  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8400 nan 0.1000 0.0897  
## 2 0.7197 nan 0.1000 0.0597  
## 3 0.6272 nan 0.1000 0.0456  
## 4 0.5530 nan 0.1000 0.0369  
## 5 0.4919 nan 0.1000 0.0311  
## 6 0.4408 nan 0.1000 0.0253  
## 7 0.3969 nan 0.1000 0.0220  
## 8 0.3595 nan 0.1000 0.0186  
## 9 0.3263 nan 0.1000 0.0161  
## 10 0.2969 nan 0.1000 0.0143  
## 20 0.1411 nan 0.1000 0.0039  
## 40 0.0707 nan 0.1000 0.0003  
## 60 0.0586 nan 0.1000 0.0000  
## 80 0.0508 nan 0.1000 -0.0004  
## 100 0.0448 nan 0.1000 0.0000  
## 120 0.0413 nan 0.1000 0.0000  
## 140 0.0380 nan 0.1000 -0.0001  
## 150 0.0369 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8523 nan 0.1000 0.0882  
## 2 0.7309 nan 0.1000 0.0604  
## 3 0.6367 nan 0.1000 0.0461  
## 4 0.5618 nan 0.1000 0.0365  
## 5 0.4997 nan 0.1000 0.0310  
## 6 0.4472 nan 0.1000 0.0264  
## 7 0.4020 nan 0.1000 0.0225  
## 8 0.3640 nan 0.1000 0.0191  
## 9 0.3295 nan 0.1000 0.0174  
## 10 0.2993 nan 0.1000 0.0146  
## 20 0.1392 nan 0.1000 0.0046  
## 40 0.0681 nan 0.1000 0.0005  
## 60 0.0580 nan 0.1000 0.0000  
## 80 0.0557 nan 0.1000 -0.0000  
## 100 0.0543 nan 0.1000 -0.0000  
## 120 0.0531 nan 0.1000 -0.0001  
## 140 0.0519 nan 0.1000 -0.0000  
## 150 0.0517 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8496 nan 0.1000 0.0888  
## 2 0.7267 nan 0.1000 0.0614  
## 3 0.6327 nan 0.1000 0.0466  
## 4 0.5573 nan 0.1000 0.0372  
## 5 0.4951 nan 0.1000 0.0311  
## 6 0.4426 nan 0.1000 0.0261  
## 7 0.3977 nan 0.1000 0.0224  
## 8 0.3589 nan 0.1000 0.0192  
## 9 0.3252 nan 0.1000 0.0165  
## 10 0.2954 nan 0.1000 0.0150  
## 20 0.1360 nan 0.1000 0.0044  
## 40 0.0661 nan 0.1000 0.0007  
## 60 0.0535 nan 0.1000 -0.0001  
## 80 0.0495 nan 0.1000 -0.0000  
## 100 0.0470 nan 0.1000 -0.0000  
## 120 0.0452 nan 0.1000 -0.0000  
## 140 0.0431 nan 0.1000 0.0000  
## 150 0.0422 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8491 nan 0.1000 0.0913  
## 2 0.7259 nan 0.1000 0.0619  
## 3 0.6313 nan 0.1000 0.0477  
## 4 0.5560 nan 0.1000 0.0377  
## 5 0.4934 nan 0.1000 0.0297  
## 6 0.4405 nan 0.1000 0.0269  
## 7 0.3951 nan 0.1000 0.0227  
## 8 0.3563 nan 0.1000 0.0192  
## 9 0.3223 nan 0.1000 0.0167  
## 10 0.2925 nan 0.1000 0.0148  
## 20 0.1323 nan 0.1000 0.0042  
## 40 0.0598 nan 0.1000 0.0003  
## 60 0.0490 nan 0.1000 0.0001  
## 80 0.0430 nan 0.1000 -0.0001  
## 100 0.0393 nan 0.1000 -0.0000  
## 120 0.0371 nan 0.1000 -0.0000  
## 140 0.0345 nan 0.1000 -0.0000  
## 150 0.0336 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8476 nan 0.1000 0.0867  
## 2 0.7282 nan 0.1000 0.0586  
## 3 0.6375 nan 0.1000 0.0449  
## 4 0.5651 nan 0.1000 0.0363  
## 5 0.5058 nan 0.1000 0.0299  
## 6 0.4555 nan 0.1000 0.0245  
## 7 0.4119 nan 0.1000 0.0217  
## 8 0.3742 nan 0.1000 0.0187  
## 9 0.3408 nan 0.1000 0.0166  
## 10 0.3114 nan 0.1000 0.0147  
## 20 0.1561 nan 0.1000 0.0046  
## 40 0.0868 nan 0.1000 0.0005  
## 60 0.0770 nan 0.1000 -0.0001  
## 80 0.0739 nan 0.1000 -0.0001  
## 100 0.0715 nan 0.1000 -0.0000  
## 120 0.0701 nan 0.1000 -0.0000  
## 140 0.0689 nan 0.1000 -0.0001  
## 150 0.0685 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8472 nan 0.1000 0.0873  
## 2 0.7267 nan 0.1000 0.0594  
## 3 0.6347 nan 0.1000 0.0456  
## 4 0.5612 nan 0.1000 0.0363  
## 5 0.4999 nan 0.1000 0.0302  
## 6 0.4480 nan 0.1000 0.0257  
## 7 0.4048 nan 0.1000 0.0212  
## 8 0.3664 nan 0.1000 0.0190  
## 9 0.3330 nan 0.1000 0.0163  
## 10 0.3043 nan 0.1000 0.0143  
## 20 0.1494 nan 0.1000 0.0044  
## 40 0.0823 nan 0.1000 0.0006  
## 60 0.0697 nan 0.1000 0.0000  
## 80 0.0648 nan 0.1000 -0.0000  
## 100 0.0587 nan 0.1000 -0.0001  
## 120 0.0557 nan 0.1000 0.0000  
## 140 0.0524 nan 0.1000 0.0000  
## 150 0.0514 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8471 nan 0.1000 0.0883  
## 2 0.7258 nan 0.1000 0.0612  
## 3 0.6340 nan 0.1000 0.0453  
## 4 0.5594 nan 0.1000 0.0366  
## 5 0.4979 nan 0.1000 0.0308  
## 6 0.4460 nan 0.1000 0.0258  
## 7 0.4025 nan 0.1000 0.0219  
## 8 0.3645 nan 0.1000 0.0185  
## 9 0.3309 nan 0.1000 0.0164  
## 10 0.3021 nan 0.1000 0.0143  
## 20 0.1457 nan 0.1000 0.0038  
## 40 0.0756 nan 0.1000 0.0005  
## 60 0.0628 nan 0.1000 -0.0001  
## 80 0.0564 nan 0.1000 -0.0001  
## 100 0.0494 nan 0.1000 0.0000  
## 120 0.0462 nan 0.1000 -0.0001  
## 140 0.0438 nan 0.1000 -0.0000  
## 150 0.0442 nan 0.1000 0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8513 nan 0.1000 0.0886  
## 2 0.7304 nan 0.1000 0.0616  
## 3 0.6375 nan 0.1000 0.0458  
## 4 0.5641 nan 0.1000 0.0362  
## 5 0.5028 nan 0.1000 0.0312  
## 6 0.4514 nan 0.1000 0.0256  
## 7 0.4082 nan 0.1000 0.0215  
## 8 0.3690 nan 0.1000 0.0190  
## 9 0.3354 nan 0.1000 0.0166  
## 10 0.3065 nan 0.1000 0.0143  
## 20 0.1499 nan 0.1000 0.0041  
## 40 0.0832 nan 0.1000 0.0006  
## 60 0.0738 nan 0.1000 0.0000  
## 80 0.0711 nan 0.1000 -0.0000  
## 100 0.0691 nan 0.1000 0.0000  
## 120 0.0676 nan 0.1000 0.0000  
## 140 0.0668 nan 0.1000 -0.0001  
## 150 0.0662 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8506 nan 0.1000 0.0894  
## 2 0.7283 nan 0.1000 0.0613  
## 3 0.6354 nan 0.1000 0.0462  
## 4 0.5617 nan 0.1000 0.0366  
## 5 0.5001 nan 0.1000 0.0314  
## 6 0.4486 nan 0.1000 0.0257  
## 7 0.4046 nan 0.1000 0.0220  
## 8 0.3659 nan 0.1000 0.0192  
## 9 0.3321 nan 0.1000 0.0167  
## 10 0.3027 nan 0.1000 0.0146  
## 20 0.1468 nan 0.1000 0.0039  
## 40 0.0795 nan 0.1000 0.0007  
## 60 0.0662 nan 0.1000 -0.0001  
## 80 0.0608 nan 0.1000 -0.0001  
## 100 0.0560 nan 0.1000 -0.0001  
## 120 0.0520 nan 0.1000 -0.0001  
## 140 0.0480 nan 0.1000 -0.0000  
## 150 0.0468 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8509 nan 0.1000 0.0884  
## 2 0.7285 nan 0.1000 0.0613  
## 3 0.6355 nan 0.1000 0.0465  
## 4 0.5604 nan 0.1000 0.0378  
## 5 0.4983 nan 0.1000 0.0309  
## 6 0.4461 nan 0.1000 0.0262  
## 7 0.4013 nan 0.1000 0.0222  
## 8 0.3628 nan 0.1000 0.0194  
## 9 0.3297 nan 0.1000 0.0166  
## 10 0.3001 nan 0.1000 0.0144  
## 20 0.1429 nan 0.1000 0.0045  
## 40 0.0738 nan 0.1000 0.0004  
## 60 0.0619 nan 0.1000 -0.0000  
## 80 0.0541 nan 0.1000 0.0003  
## 100 0.0499 nan 0.1000 0.0002  
## 120 0.0438 nan 0.1000 -0.0000  
## 140 0.0393 nan 0.1000 -0.0001  
## 150 0.0377 nan 0.1000 0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8615 nan 0.1000 0.0901  
## 2 0.7393 nan 0.1000 0.0613  
## 3 0.6452 nan 0.1000 0.0469  
## 4 0.5702 nan 0.1000 0.0365  
## 5 0.5076 nan 0.1000 0.0313  
## 6 0.4557 nan 0.1000 0.0260  
## 7 0.4103 nan 0.1000 0.0225  
## 8 0.3716 nan 0.1000 0.0194  
## 9 0.3378 nan 0.1000 0.0168  
## 10 0.3077 nan 0.1000 0.0147  
## 20 0.1481 nan 0.1000 0.0047  
## 40 0.0784 nan 0.1000 0.0004  
## 60 0.0679 nan 0.1000 0.0000  
## 80 0.0660 nan 0.1000 -0.0001  
## 100 0.0643 nan 0.1000 -0.0000  
## 120 0.0634 nan 0.1000 -0.0001  
## 140 0.0623 nan 0.1000 -0.0000  
## 150 0.0617 nan 0.1000 0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8604 nan 0.1000 0.0876  
## 2 0.7375 nan 0.1000 0.0612  
## 3 0.6433 nan 0.1000 0.0471  
## 4 0.5673 nan 0.1000 0.0376  
## 5 0.5041 nan 0.1000 0.0311  
## 6 0.4510 nan 0.1000 0.0261  
## 7 0.4056 nan 0.1000 0.0225  
## 8 0.3665 nan 0.1000 0.0196  
## 9 0.3326 nan 0.1000 0.0167  
## 10 0.3027 nan 0.1000 0.0148  
## 20 0.1427 nan 0.1000 0.0041  
## 40 0.0748 nan 0.1000 0.0004  
## 60 0.0625 nan 0.1000 0.0001  
## 80 0.0552 nan 0.1000 0.0000  
## 100 0.0515 nan 0.1000 -0.0002  
## 120 0.0464 nan 0.1000 -0.0000  
## 140 0.0441 nan 0.1000 -0.0001  
## 150 0.0428 nan 0.1000 -0.0000  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8593 nan 0.1000 0.0893  
## 2 0.7361 nan 0.1000 0.0629  
## 3 0.6418 nan 0.1000 0.0476  
## 4 0.5657 nan 0.1000 0.0381  
## 5 0.5026 nan 0.1000 0.0310  
## 6 0.4494 nan 0.1000 0.0266  
## 7 0.4037 nan 0.1000 0.0221  
## 8 0.3642 nan 0.1000 0.0196  
## 9 0.3302 nan 0.1000 0.0171  
## 10 0.3007 nan 0.1000 0.0147  
## 20 0.1406 nan 0.1000 0.0044  
## 40 0.0696 nan 0.1000 0.0002  
## 60 0.0549 nan 0.1000 -0.0001  
## 80 0.0448 nan 0.1000 -0.0000  
## 100 0.0381 nan 0.1000 0.0001  
## 120 0.0352 nan 0.1000 -0.0000  
## 140 0.0319 nan 0.1000 -0.0001  
## 150 0.0310 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8618 nan 0.1000 0.0913  
## 2 0.7393 nan 0.1000 0.0609  
## 3 0.6449 nan 0.1000 0.0474  
## 4 0.5700 nan 0.1000 0.0371  
## 5 0.5066 nan 0.1000 0.0318  
## 6 0.4536 nan 0.1000 0.0261  
## 7 0.4085 nan 0.1000 0.0223  
## 8 0.3702 nan 0.1000 0.0192  
## 9 0.3360 nan 0.1000 0.0172  
## 10 0.3070 nan 0.1000 0.0145  
## 20 0.1475 nan 0.1000 0.0042  
## 40 0.0783 nan 0.1000 0.0006  
## 60 0.0679 nan 0.1000 0.0000  
## 80 0.0657 nan 0.1000 -0.0001  
## 100 0.0638 nan 0.1000 0.0000  
## 120 0.0632 nan 0.1000 -0.0001  
## 140 0.0623 nan 0.1000 0.0001  
## 150 0.0623 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8611 nan 0.1000 0.0885  
## 2 0.7371 nan 0.1000 0.0629  
## 3 0.6431 nan 0.1000 0.0463  
## 4 0.5669 nan 0.1000 0.0379  
## 5 0.5033 nan 0.1000 0.0314  
## 6 0.4503 nan 0.1000 0.0265  
## 7 0.4052 nan 0.1000 0.0223  
## 8 0.3658 nan 0.1000 0.0198  
## 9 0.3319 nan 0.1000 0.0165  
## 10 0.3021 nan 0.1000 0.0150  
## 20 0.1435 nan 0.1000 0.0044  
## 40 0.0738 nan 0.1000 0.0005  
## 60 0.0603 nan 0.1000 -0.0001  
## 80 0.0561 nan 0.1000 -0.0000  
## 100 0.0534 nan 0.1000 0.0000  
## 120 0.0474 nan 0.1000 -0.0000  
## 140 0.0443 nan 0.1000 -0.0000  
## 150 0.0427 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8615 nan 0.1000 0.0875  
## 2 0.7375 nan 0.1000 0.0617  
## 3 0.6431 nan 0.1000 0.0480  
## 4 0.5660 nan 0.1000 0.0388  
## 5 0.5027 nan 0.1000 0.0320  
## 6 0.4495 nan 0.1000 0.0269  
## 7 0.4038 nan 0.1000 0.0226  
## 8 0.3645 nan 0.1000 0.0197  
## 9 0.3303 nan 0.1000 0.0169  
## 10 0.3005 nan 0.1000 0.0147  
## 20 0.1401 nan 0.1000 0.0042  
## 40 0.0693 nan 0.1000 0.0004  
## 60 0.0571 nan 0.1000 -0.0000  
## 80 0.0482 nan 0.1000 -0.0000  
## 100 0.0444 nan 0.1000 -0.0003  
## 120 0.0408 nan 0.1000 0.0000  
## 140 0.0391 nan 0.1000 -0.0001  
## 150 0.0376 nan 0.1000 -0.0001  
##   
## Iter TrainDeviance ValidDeviance StepSize Improve  
## 1 0.8509 nan 0.1000 0.0896  
## 2 0.7297 nan 0.1000 0.0608  
## 3 0.6374 nan 0.1000 0.0467  
## 4 0.5633 nan 0.1000 0.0360  
## 5 0.5027 nan 0.1000 0.0301  
## 6 0.4518 nan 0.1000 0.0253  
## 7 0.4090 nan 0.1000 0.0219  
## 8 0.3704 nan 0.1000 0.0194  
## 9 0.3378 nan 0.1000 0.0161  
## 10 0.3084 nan 0.1000 0.0147  
## 20 0.1516 nan 0.1000 0.0047  
## 40 0.0812 nan 0.1000 0.0005  
## 50 0.0741 nan 0.1000 0.0002

#Logistic regression as top layer model  
model\_glm<-  
train(trainSet[,predictors\_top],trainSet[,outcomeName],method='glm')  
  
#predict using GBM top layer model  
testSet$gbm\_stacked<-predict(model\_gbm,testSet[,predictors\_top])  
  
#predict using logictic regression top layer model  
testSet$glm\_stacked<-predict(model\_glm,testSet[,predictors\_top])

confusionMatrix(testSet$IsAlert,testSet$gbm\_stacked)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 515 10  
## Yes 9 1965  
##   
## Accuracy : 0.9924   
## 95% CI : (0.9882, 0.9954)  
## No Information Rate : 0.7903   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9771   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9828   
## Specificity : 0.9949   
## Pos Pred Value : 0.9810   
## Neg Pred Value : 0.9954   
## Prevalence : 0.2097   
## Detection Rate : 0.2061   
## Detection Prevalence : 0.2101   
## Balanced Accuracy : 0.9889   
##   
## 'Positive' Class : No   
##

confusionMatrix(testSet$IsAlert,testSet$glm\_stacked)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 514 11  
## Yes 3 1971  
##   
## Accuracy : 0.9944   
## 95% CI : (0.9906, 0.9969)  
## No Information Rate : 0.7931   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.983   
## Mcnemar's Test P-Value : 0.06137   
##   
## Sensitivity : 0.9942   
## Specificity : 0.9945   
## Pos Pred Value : 0.9790   
## Neg Pred Value : 0.9985   
## Prevalence : 0.2069   
## Detection Rate : 0.2057   
## Detection Prevalence : 0.2101   
## Balanced Accuracy : 0.9943   
##   
## 'Positive' Class : No   
##

When GBM model at top layer, the ensemable mdoel has accuracy as 0.9924. When logistic model at top layer, the ensemable model has accuracy as 0.9944. they both are good, but I prefer using logistic model as top layer model.