Aviation

Code

Loading libraries required:

Hide

```
library(DataExplorer)
library(ggplot2)
library(ggpubr)
library(psych)
library(corrplot)
library(pscl)
library(caret)
library(mice)
library(randomForest)
library(ggplotAssist)
library(scales)
library(rpart)
library(rpart.plot)
library(Rtsne)
library(fpc)
library(rattle)
library(cluster)
library(ineq)
library(InformationValue)
library(NbClust)
library(clValid)
library(factoextra)
library(dplyr)
library(pROC)
#install.packages(c("fpc", "rattle", "Rtsne", "sqldf", "zipcode"))
```

Setting up working directory:

Hide

```
setwd('D:/Smitayan/PGP BABI/Capstone/Aviation-Marketing Project files')
```

```
Error in setwd("D:/Smitayan/PGP BABI/Capstone/Aviation-Marketing Project files") :
  cannot change working directory
```

Reading the data files:

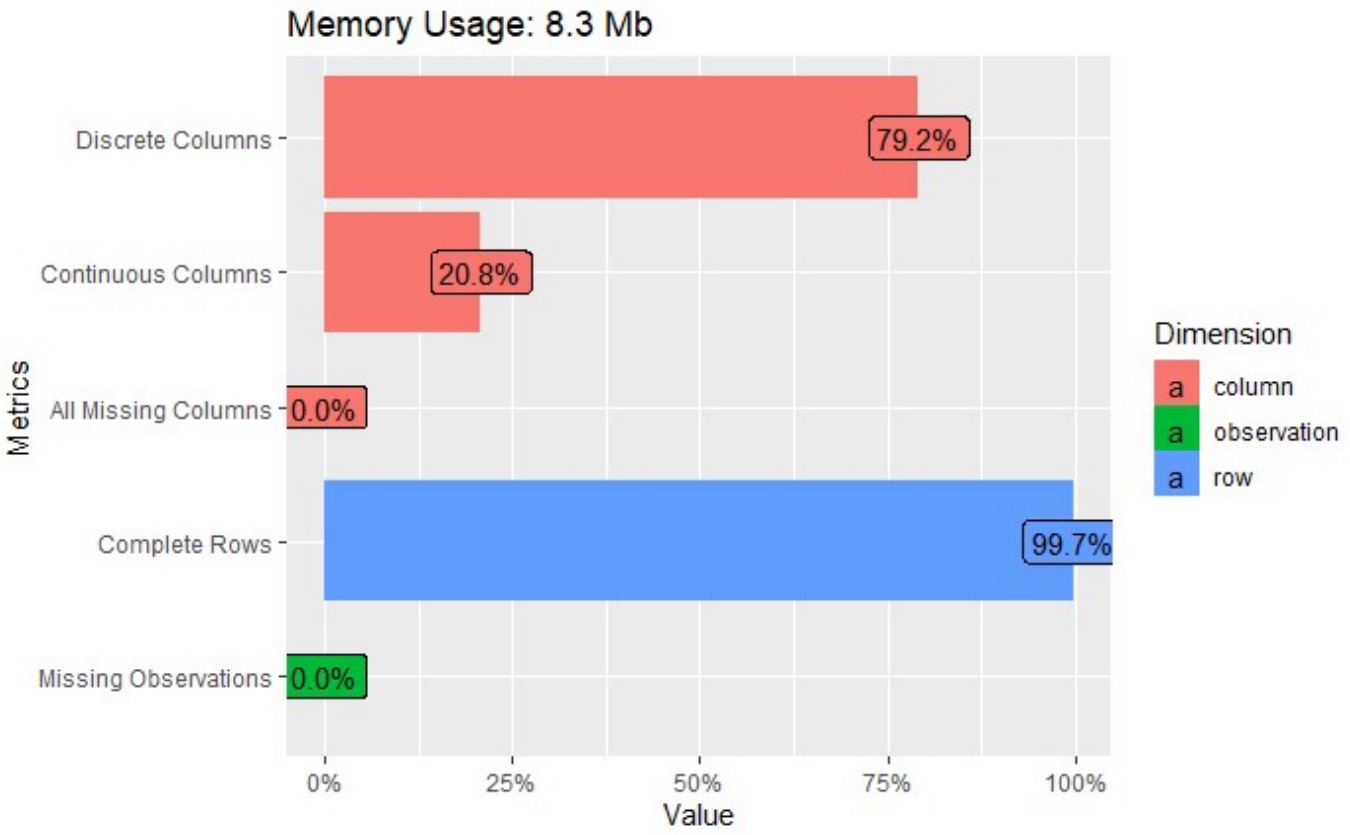
Hide

```
flightdata=read.csv('Marketing Project-Flight data.csv',header = T)
surveydata=read.csv('Marketing Project-Survey data.csv',header = T)
Cleansed.Data=read.csv('Cleansed_Data.csv',header = T)
```

Merging both the files on Customer ID:

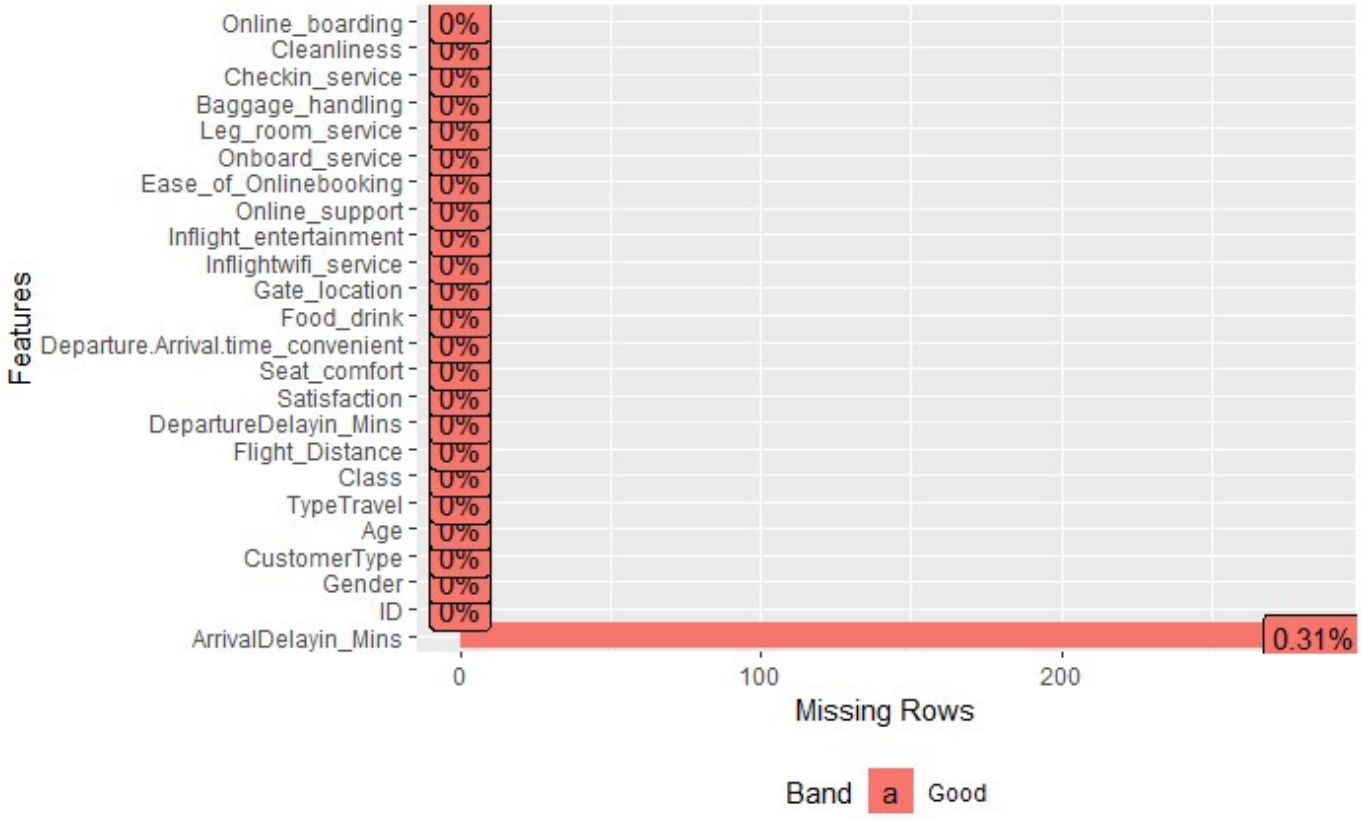
Hide

```
?plot_intro
plot_intro(Aviation)
```



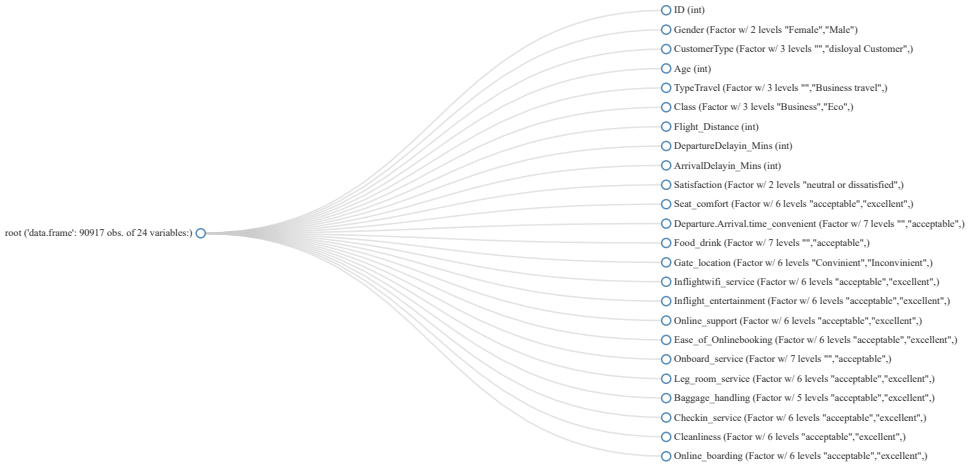
Hide

```
plot_missing(Aviation)
```



Hide

```
?plot_str
??plot_str_args
plot_str(Aviation)
```



Checking for null values:

Hide

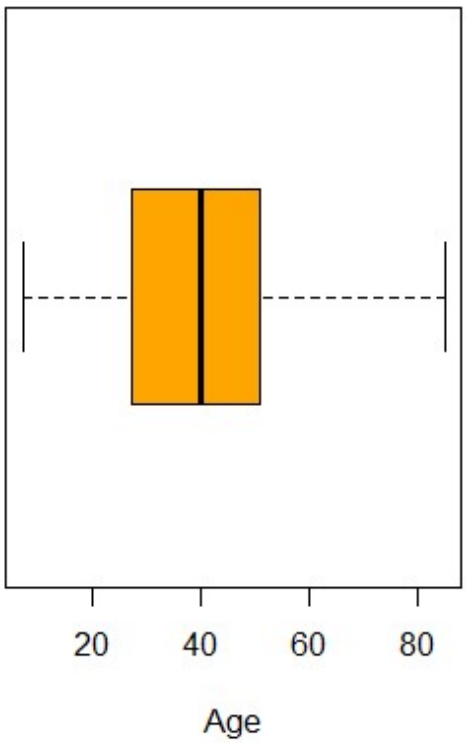
```
sapply(Aviation,function(x) sum(is.na(x)))
```

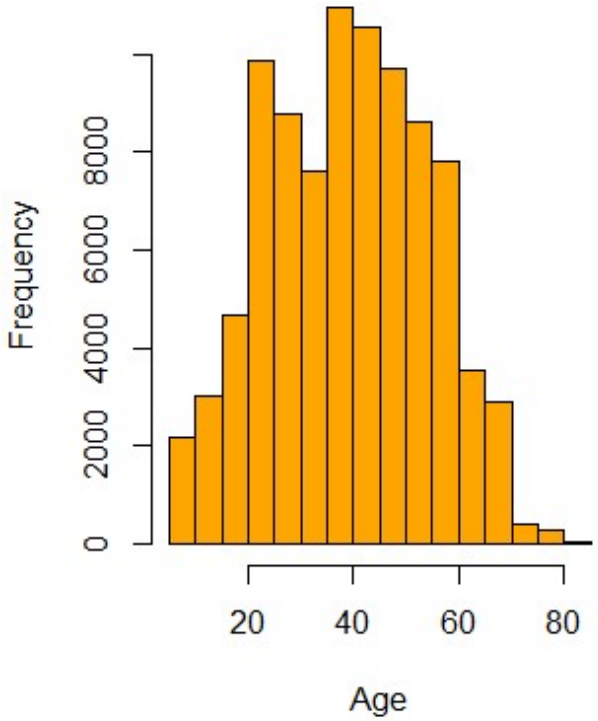
```
                             ID                          Gender                      Cust
omerType
                              0                               0
0
                            Age                      TypeTravel
Class
                              0                               0
0
                Flight_Distance          DepartureDelayin_Mins              ArrivalDela
yin_Mins
                              0                               0
284
                   Satisfaction                  Seat_comfort Departure.Arrival.time_co
nvenient
                              0                               0
0
                     Food_drink                  Gate_location             Inflightwifi
_service
                              0                               0
0
          Inflight_entertainment                Online_support           Ease_of_Onlin
ebooking
                              0                               0
0
                Onboard_service               Leg_room_service                 Baggage_
handling
                              0                               0
0
                Checkin_service                   Cleanliness                 Online_
boarding
                              0                               0
0
```

#Univariate Analysis of continuous variables

Hide

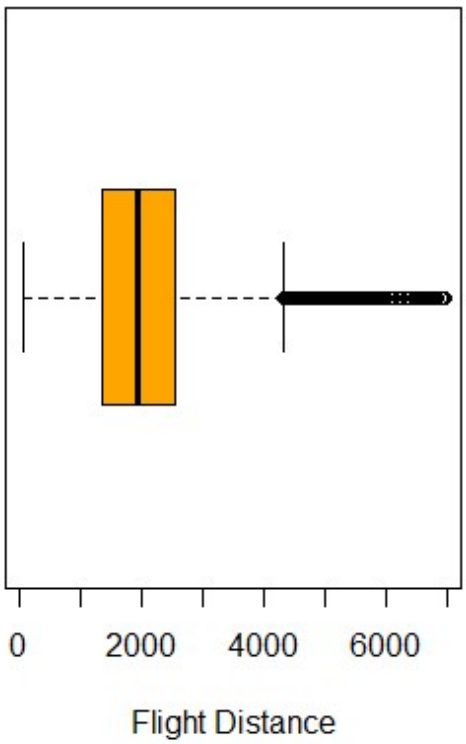
```
par(mfrow=c(1,2))
boxplot(Aviation$Age,xlab='Age',horizontal = T,col='orange')
hist(Aviation$Age,xlab = 'Age',main='',col = 'orange')
```

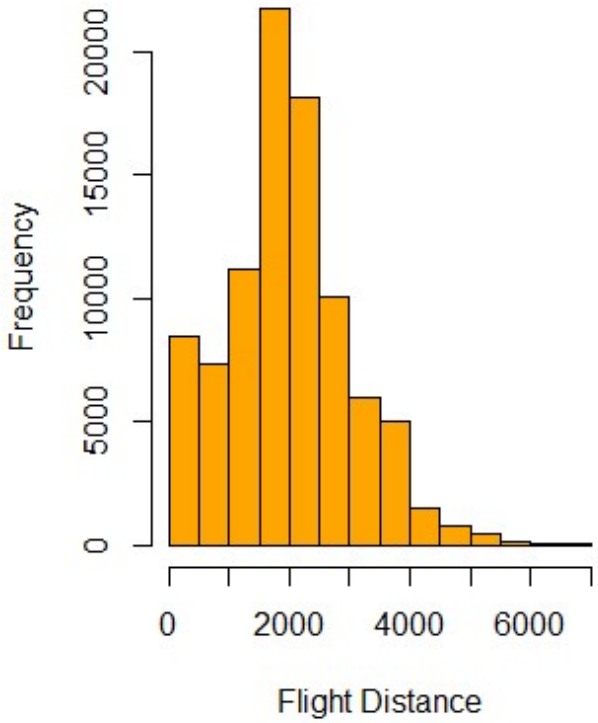




Hide

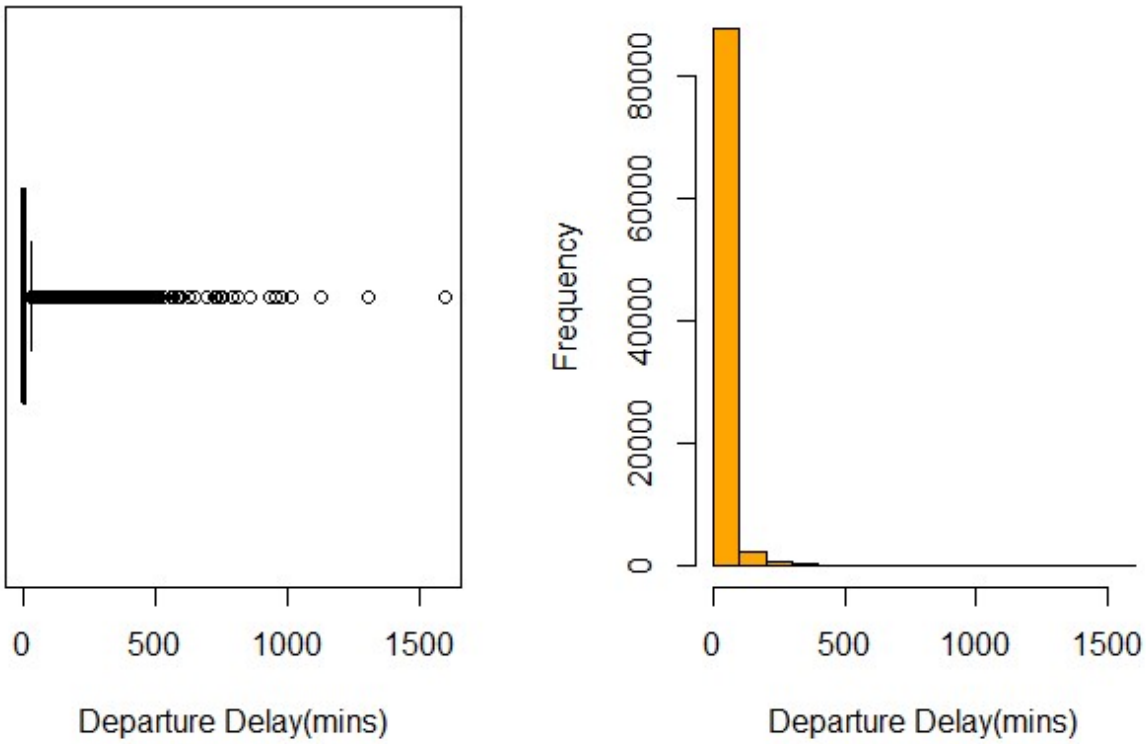
```
boxplot(Aviation$Flight_Distance,xlab='Flight Distance',horizontal = T,col = 'orange')
hist(Aviation$Flight_Distance,xlab = 'Flight Distance',main = '',col = 'orange')
```





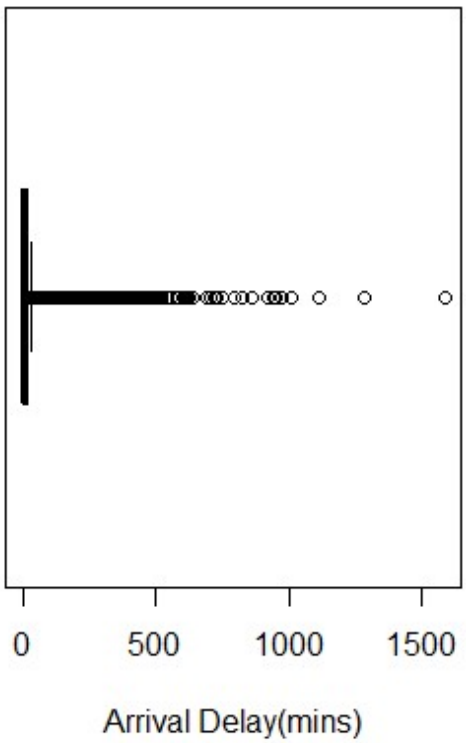
Hide

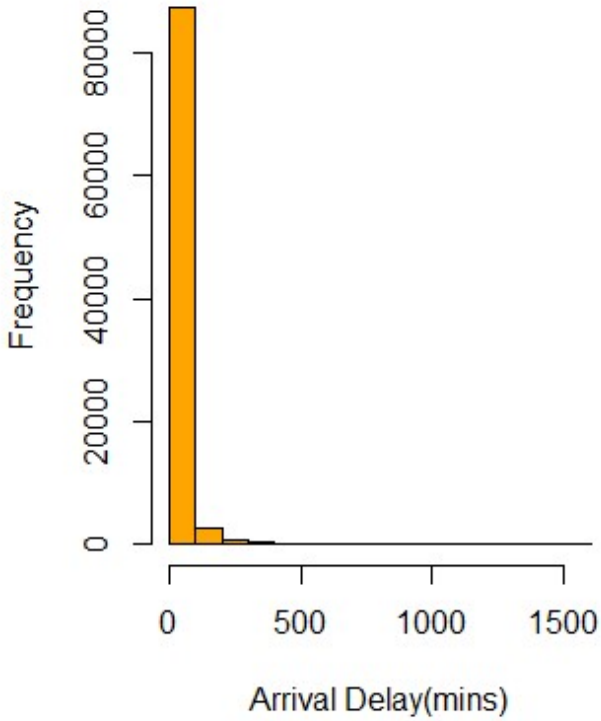
```
boxplot(Aviation$DepartureDelayin_Mins,xlab='Departure Delay(mins)',horizontal = T,col = 'ora
nge')
hist(Aviation$DepartureDelayin_Mins,xlab = 'Departure Delay(mins)',main = '',col = 'orange')
```



Hide

```
boxplot(Aviation$ArrivalDelayin_Mins,xlab='Arrival Delay(mins)',horizontal = T,col = 'orang
e')
hist(Aviation$ArrivalDelayin_Mins,xlab = 'Arrival Delay(mins)',main = '',col = 'orange')
```

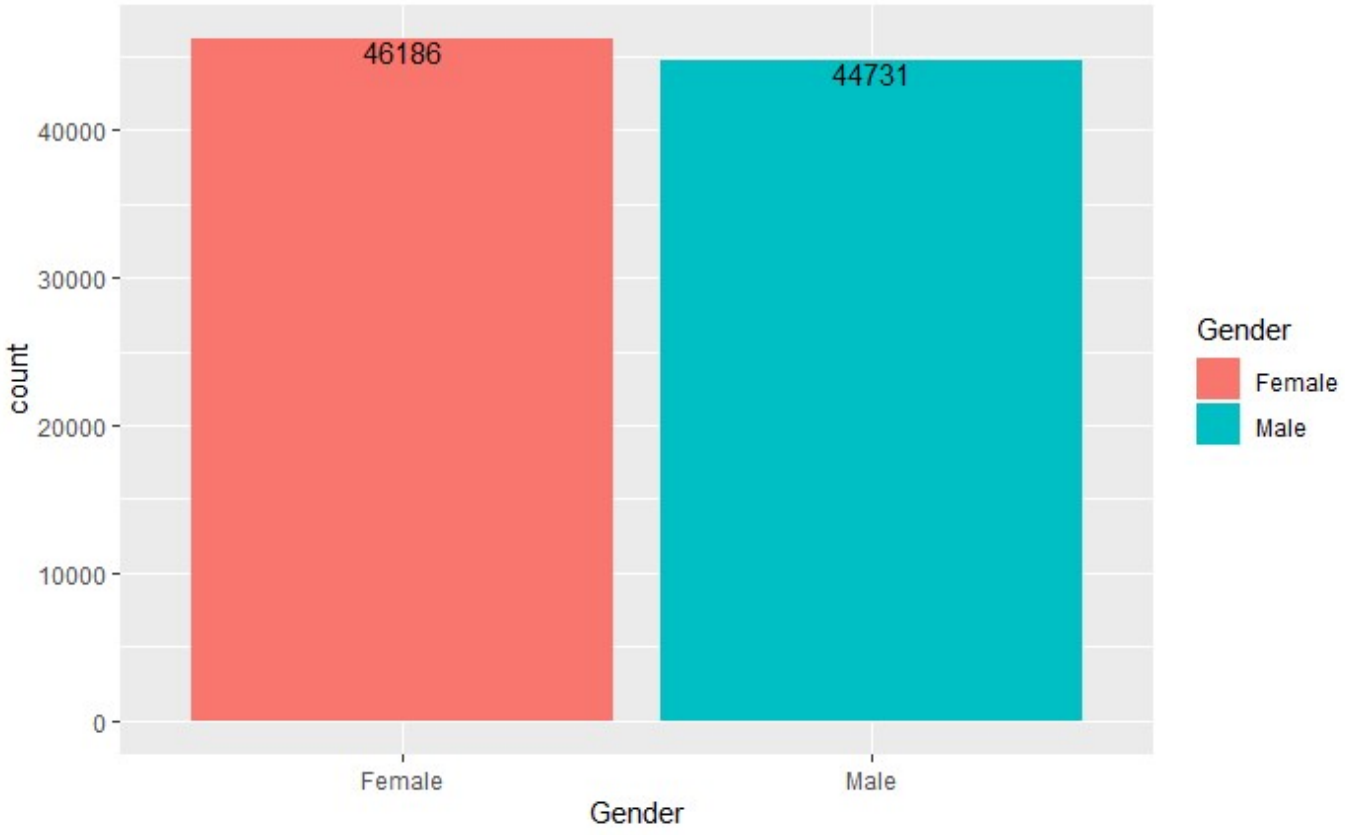




#Univariate analysis of factor variables

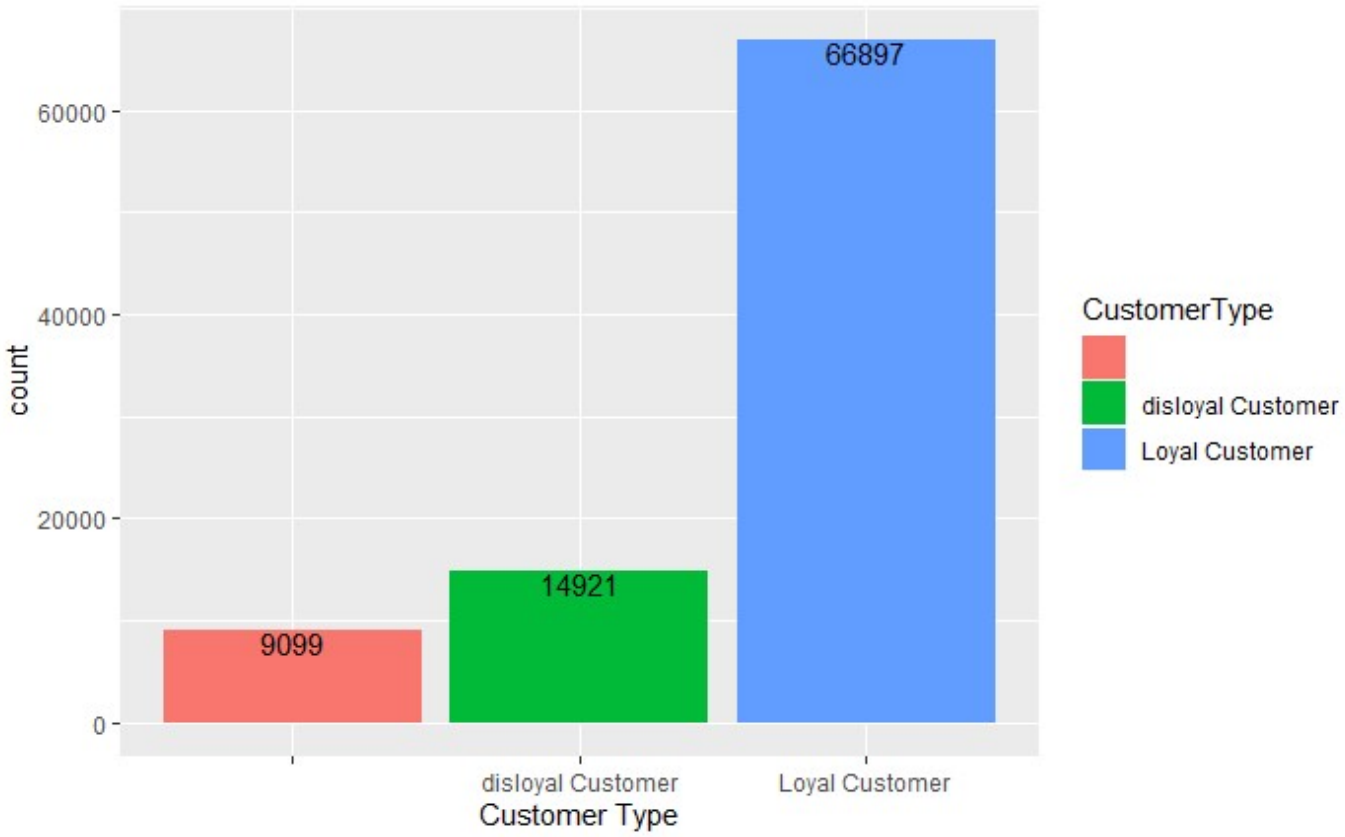
Hide

```
?stat_count
ggplot(data = Aviation)+aes(x=Aviation$Gender,fill=Gender)+geom_bar()+labs(x='Gender')+geom_t
ext(aes(label=..count..),stat="count",vjust=1)
```



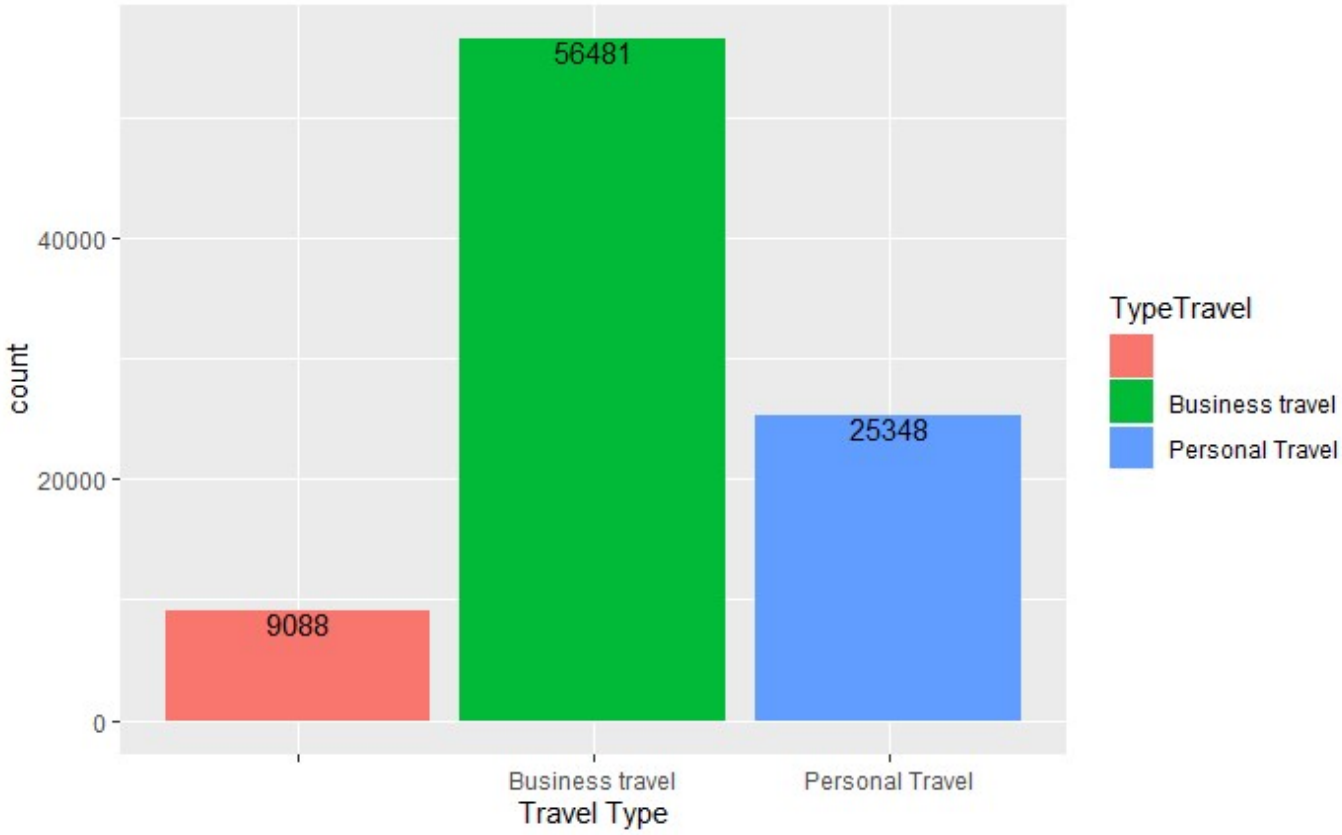
Hide

```
ggplot(data = Aviation)+aes(Aviation$CustomerType,fill=CustomerType)+geom_bar()+labs(x='Custo
mer Type')+geom_text(aes(label=..count..),stat="count",vjust=1)
```



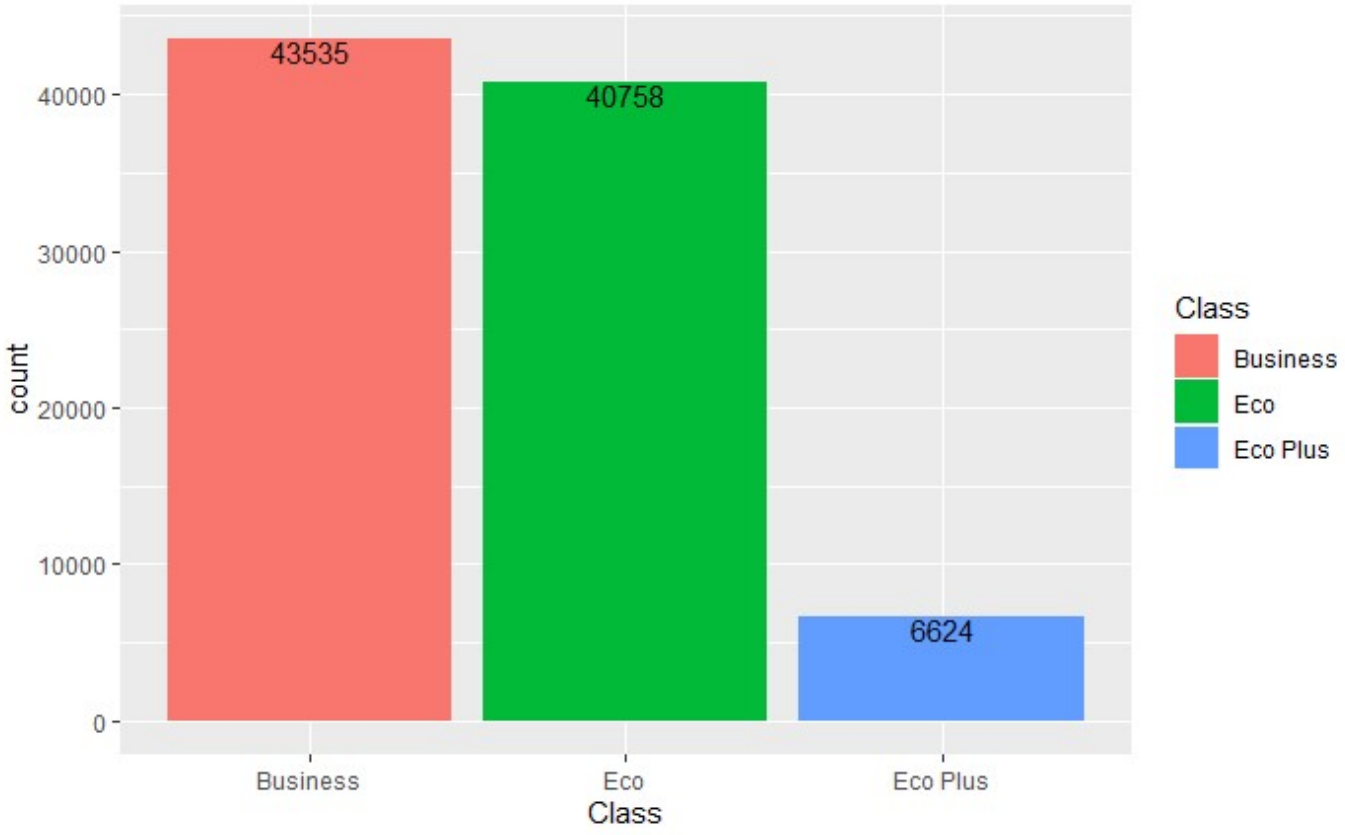
Hide

```
ggplot(data = Aviation)+aes(Aviation$TypeTravel,fill=TypeTravel)+geom_bar()+labs(x='Travel Type')+geom_text(aes(label=..count..),stat="count",vjust=1)
```



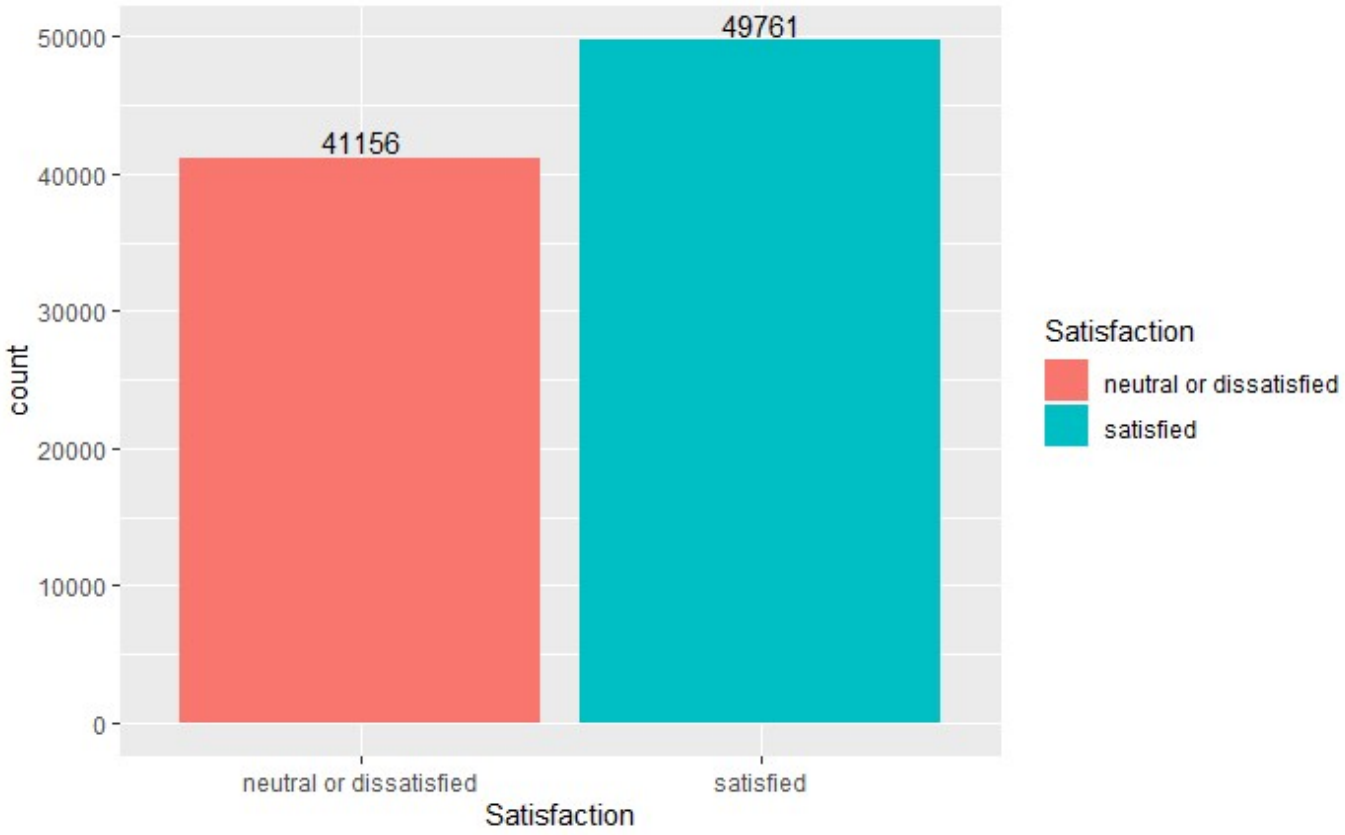
Hide

```
ggplot(data = Aviation)+aes(Aviation$Class,fill=Class)+geom_bar()+labs(x='Class')+geom_text(aes(label=..count..),stat="count",vjust=1)
```



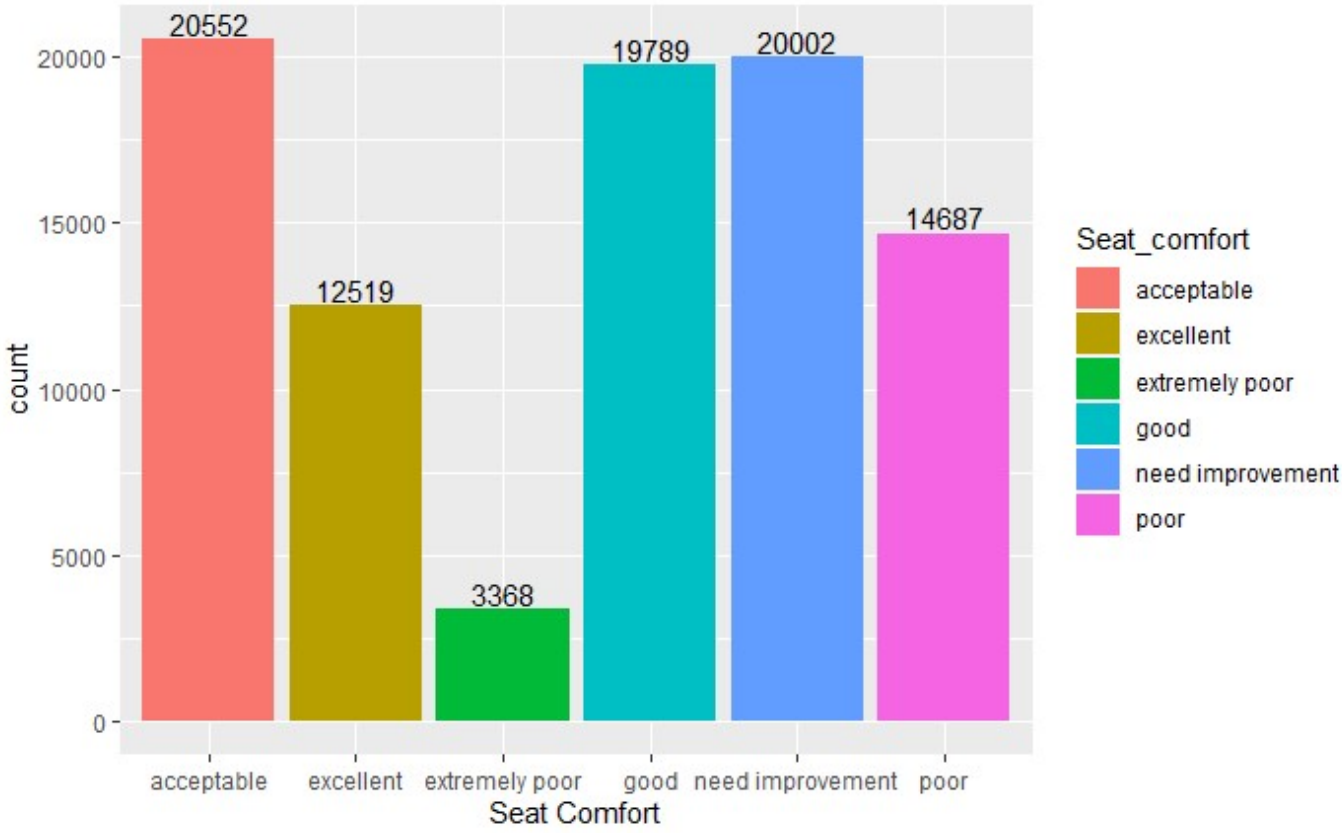
Hide

```
ggplot(Aviation,aes(x=Satisfaction,fill=Satisfaction))+ geom_bar(position='dodge') +geom_text
(aes(label=..count..),stat='count',vjust=-0.2)
```



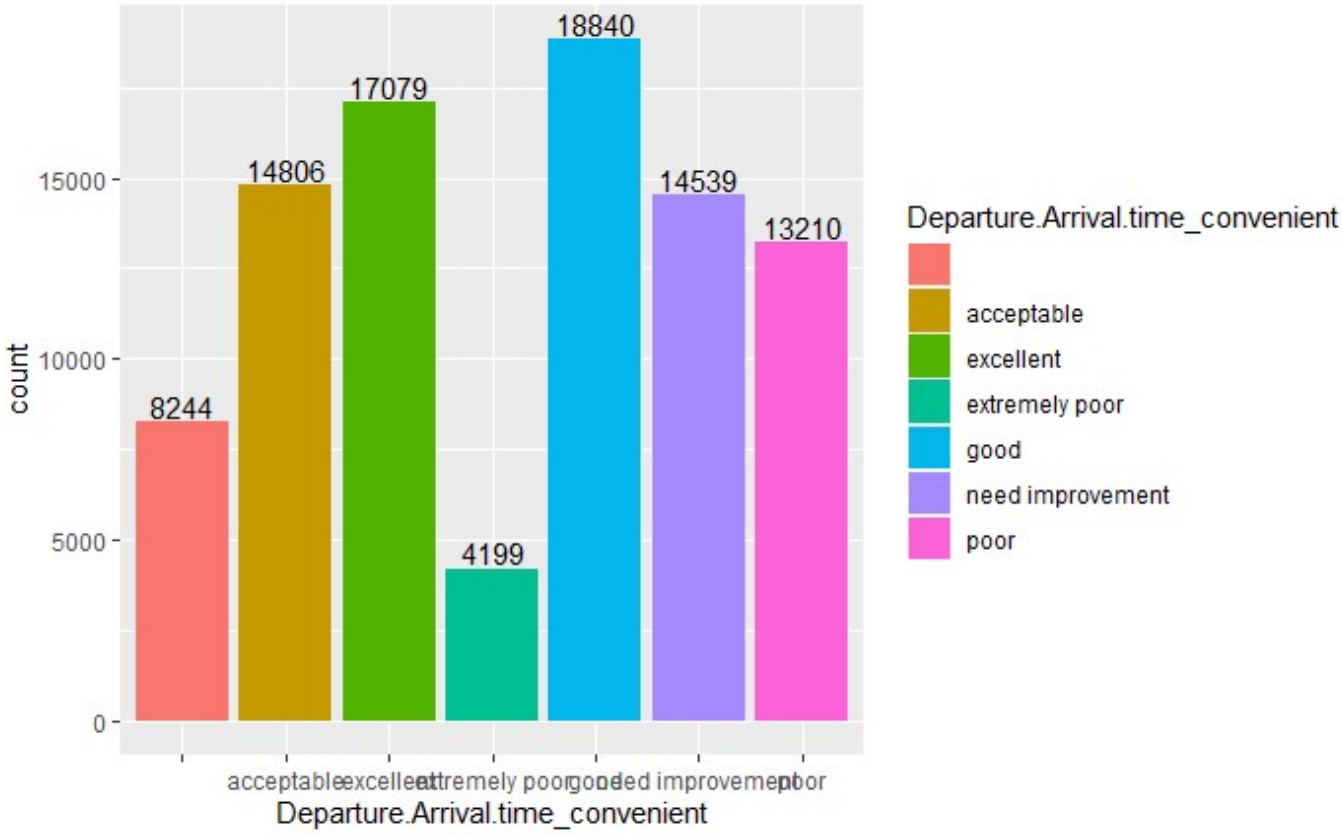
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Seat_comfort,fill=Seat_comfort)+geom_bar(position = 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Seat Comfort')
```



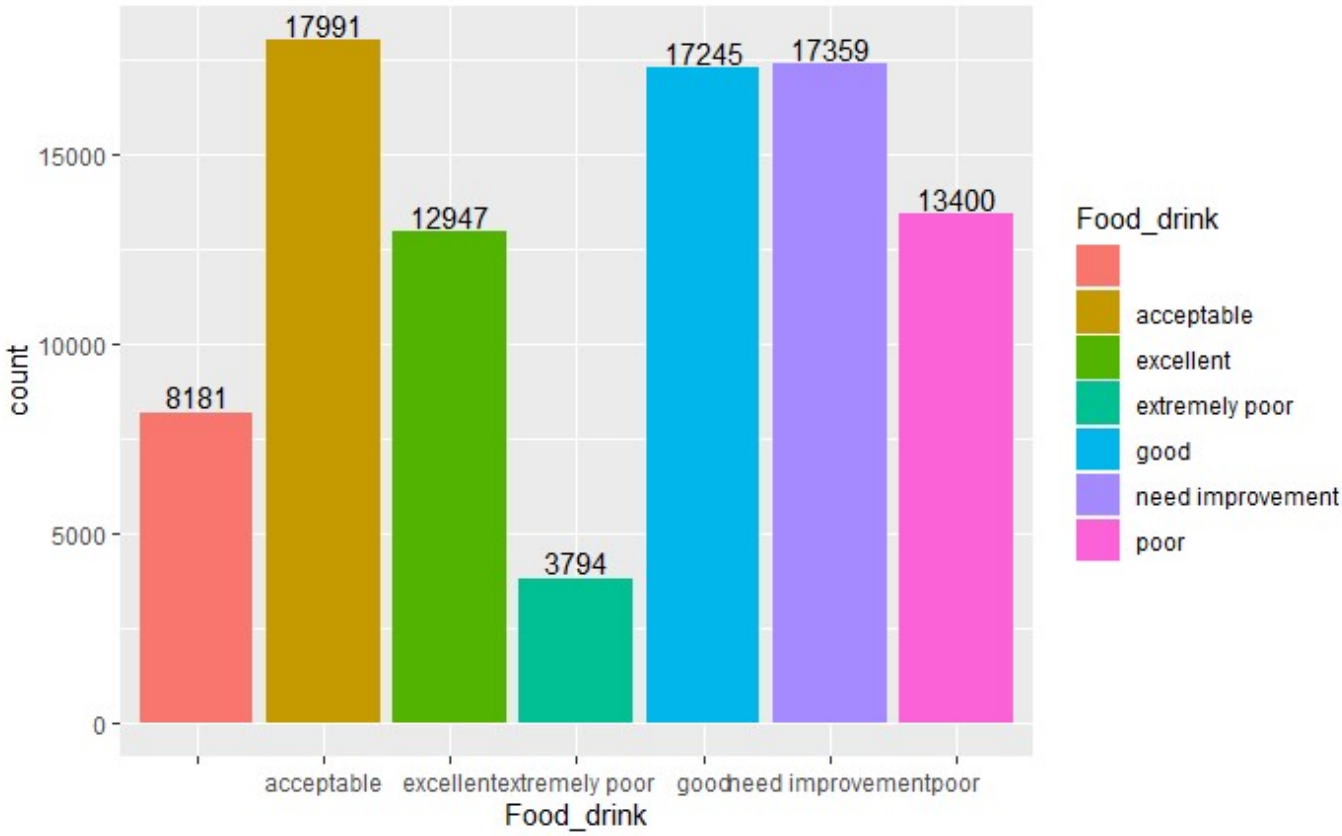
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Departure.Arrival.time_convenient,fill=Departure.Arrival.time_convenient)+geom_bar(position = 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Departure.Arrival.time_convenient')
```



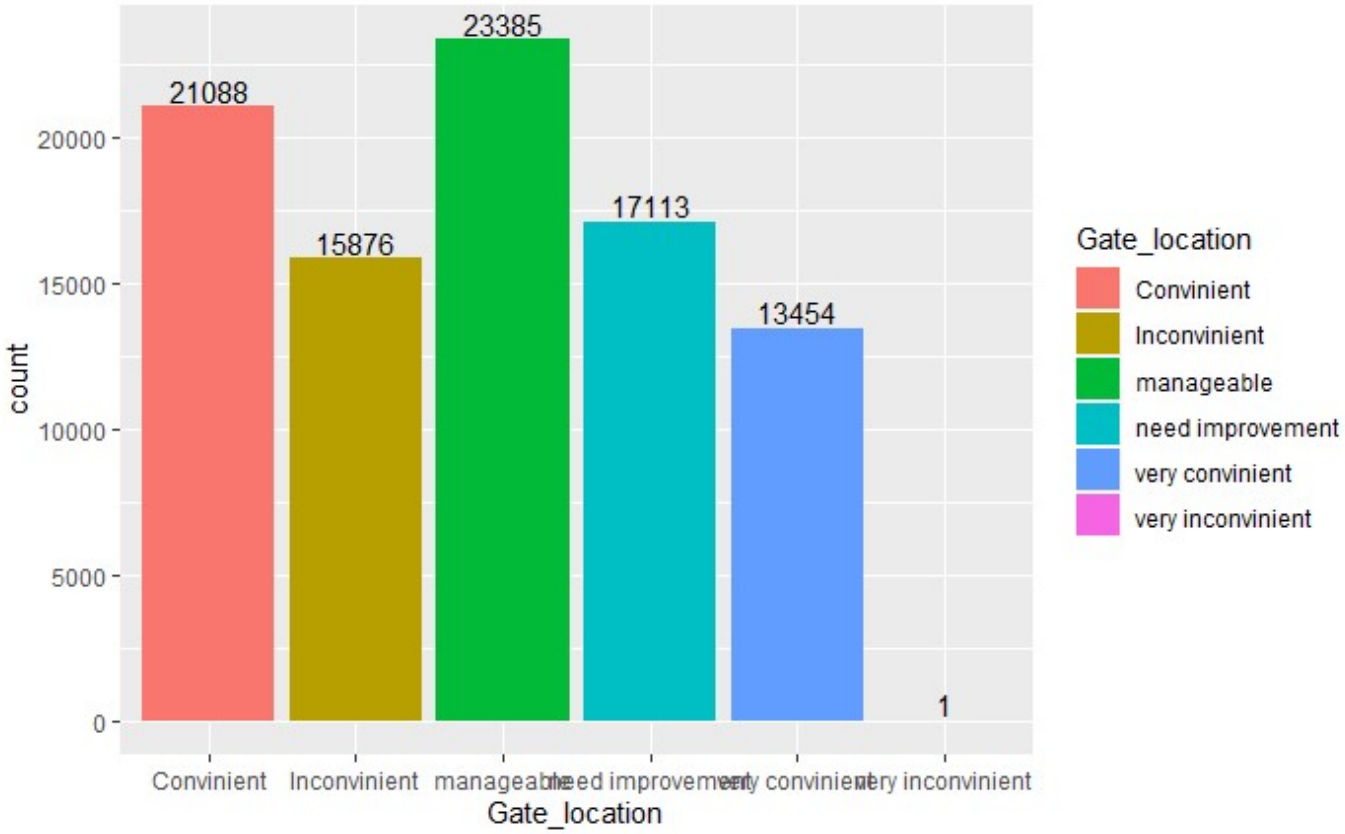
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Food_drink,fill=Food_drink)+geom_bar(position = 'dodg
e')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Food_drink')
```



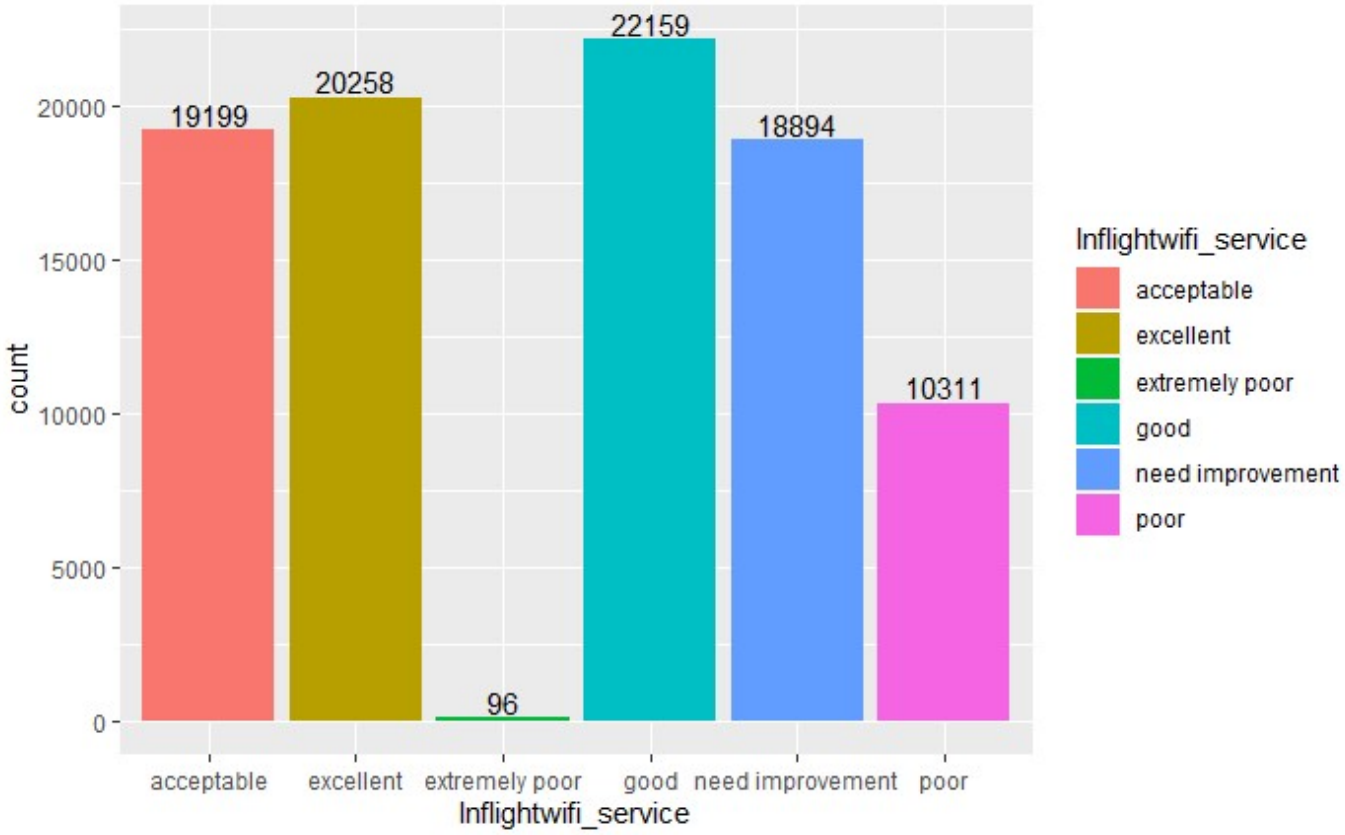
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Gate_location,fill=Gate_location)+geom_bar(position =
'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Gate_location')
```



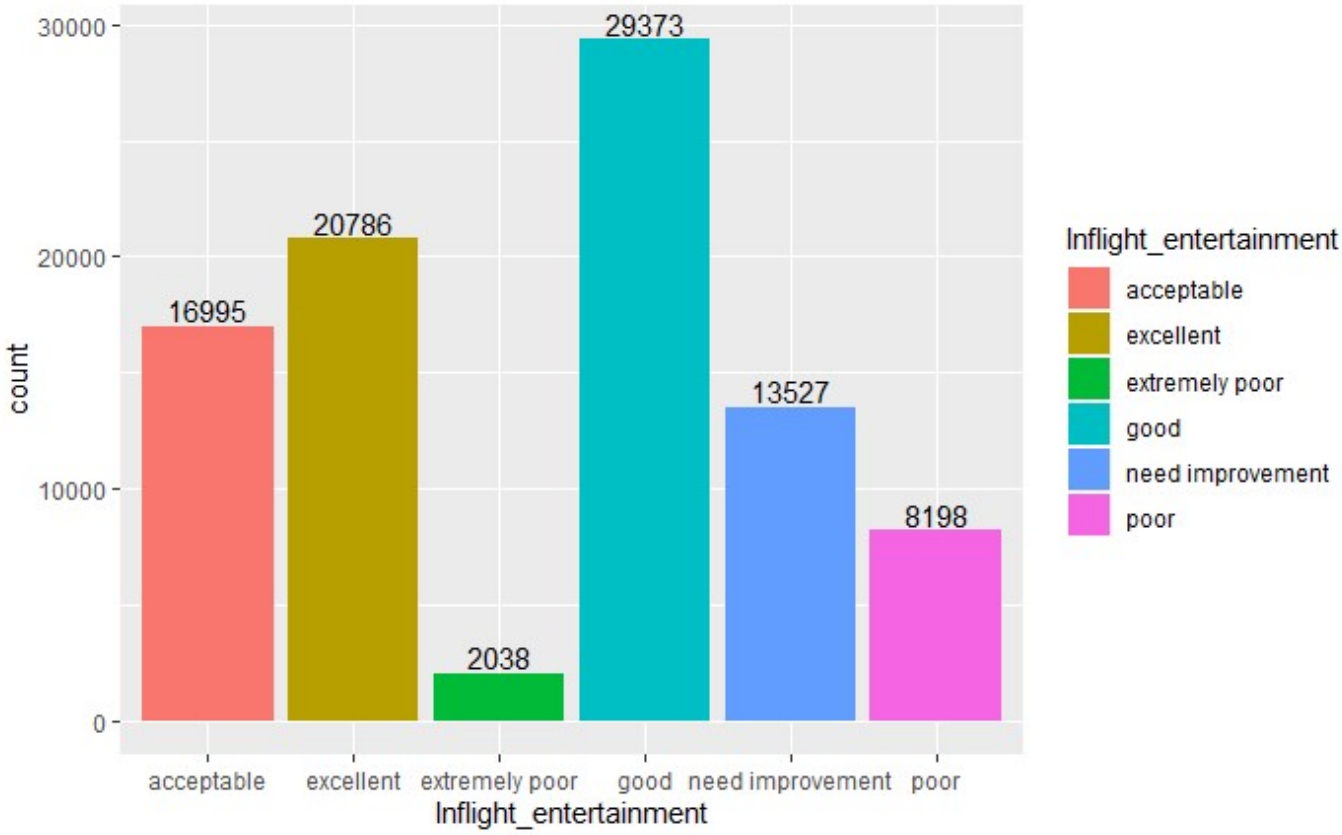
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Inflightwifi_service,fill=Inflightwifi_service)+geom_b
ar(position = 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Infl
ightwifi_service')
```



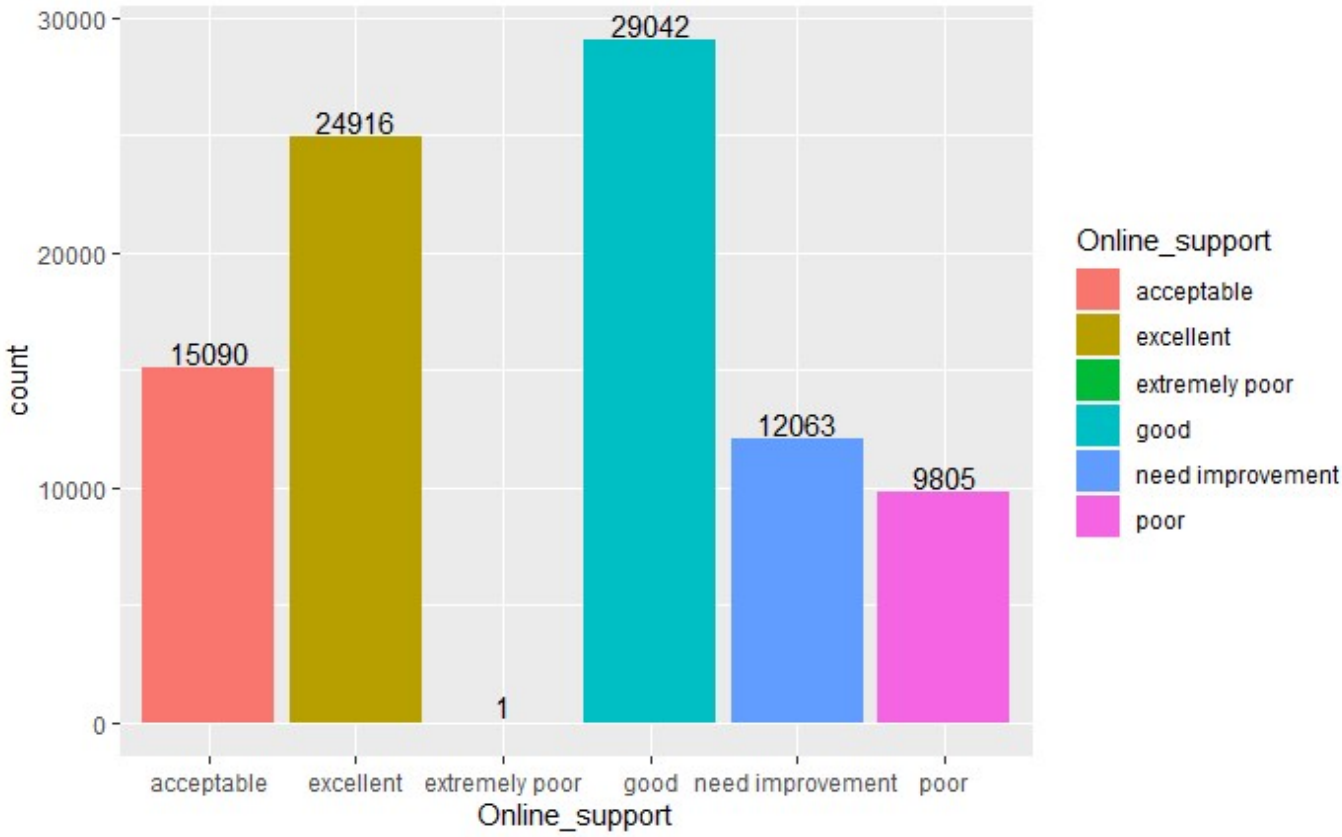
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Inflight_entertainment,fill=Inflight_entertainment)+ge
om_bar(position = 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x
='Inflight_entertainment')
```



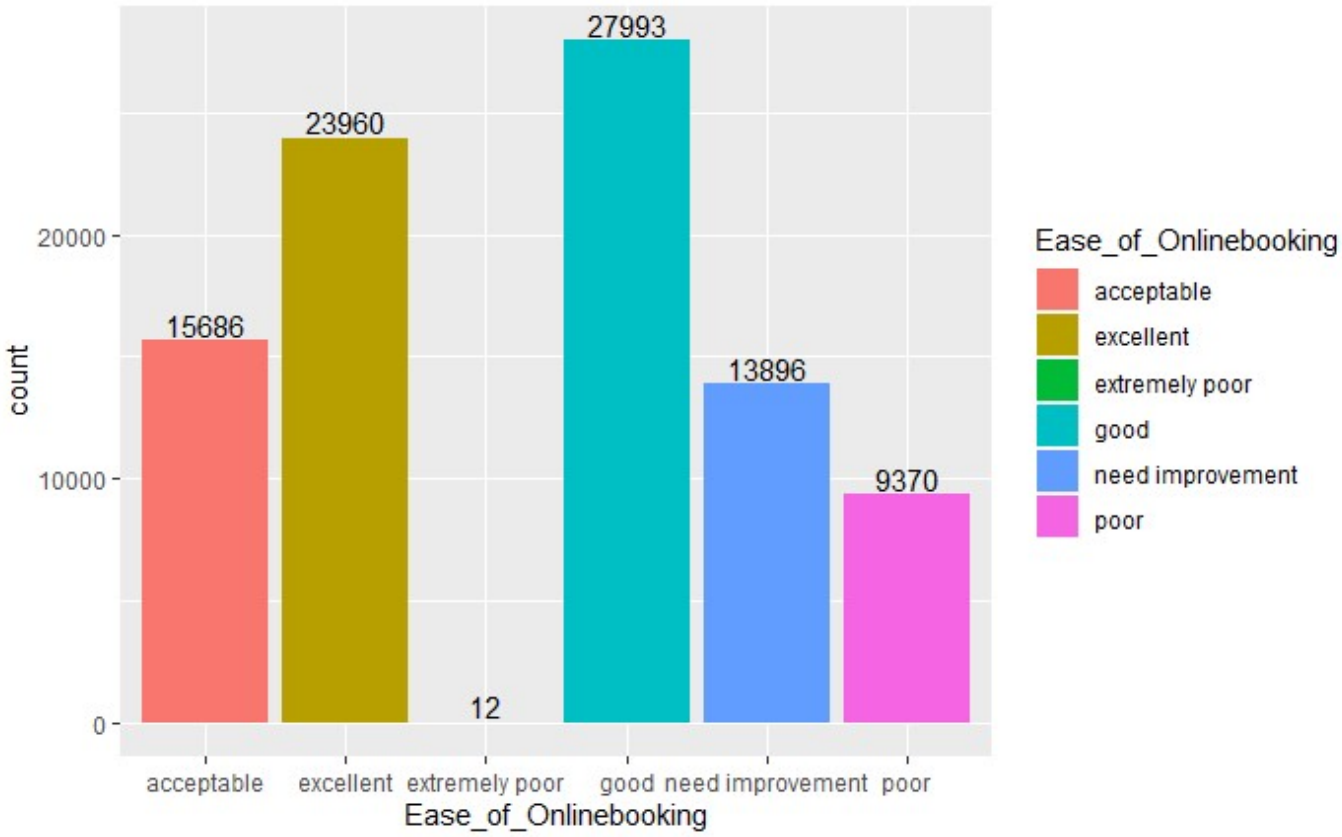
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Online_support,fill=Online_support)+geom_bar(position
= 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Online_support')
```



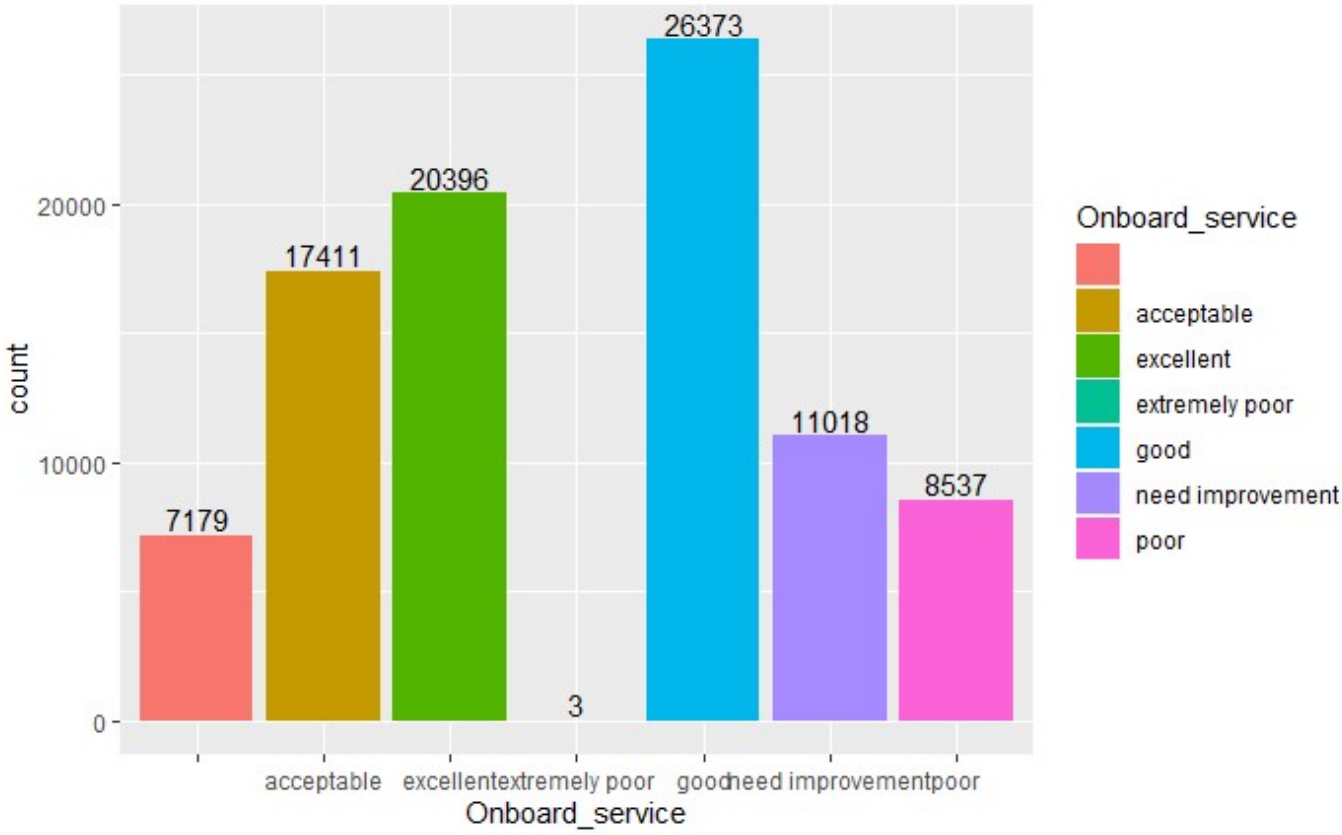
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Ease_of_Onlinebooking,fill=Ease_of_Onlinebooking)+geom
_bar(position = 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Ea
se_of_Onlinebooking')
```



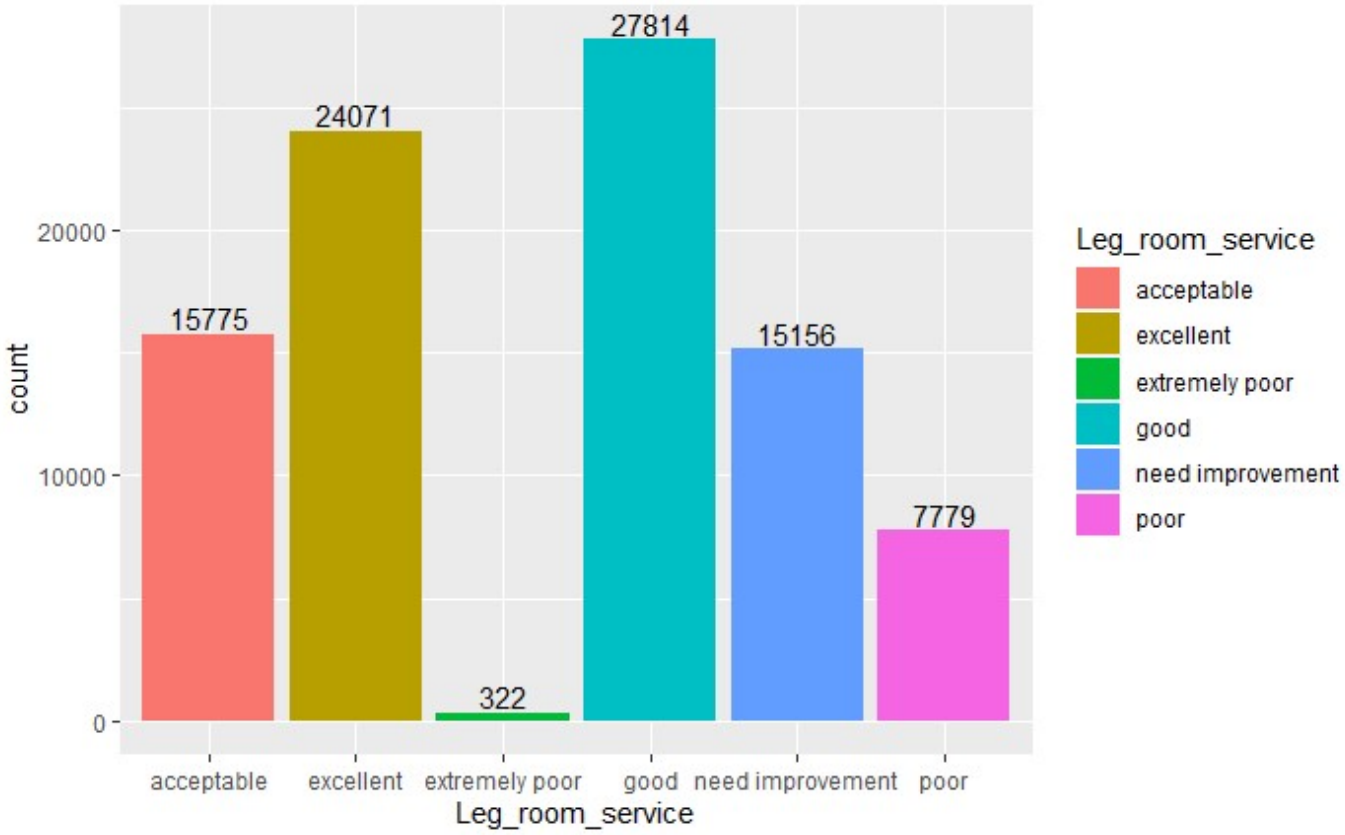
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Onboard_service,fill=Onboard_service)+geom_bar(positio
n = 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Onboard_servic
e')
```



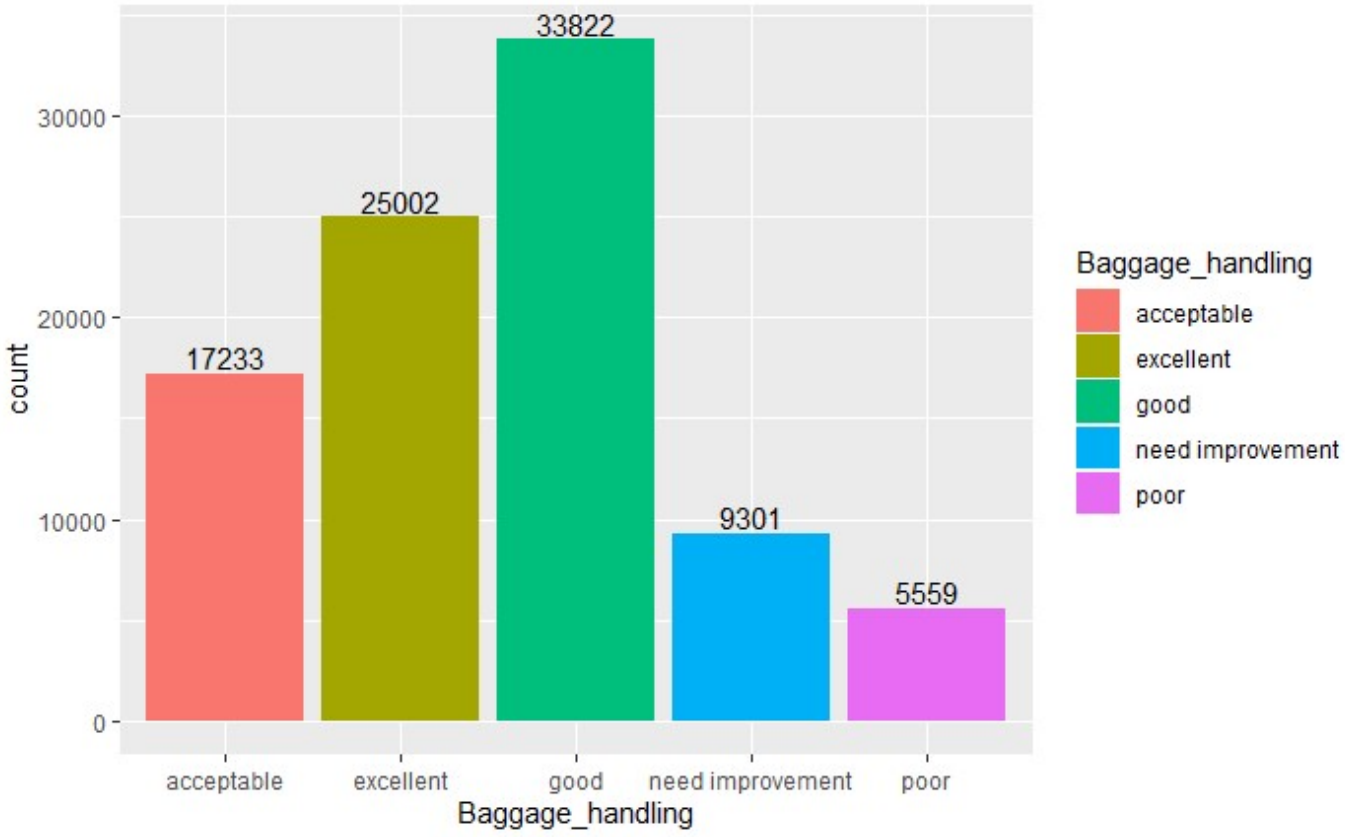
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Leg_room_service,fill=Leg_room_service)+geom_bar(posit
ion = 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Leg_room_ser
vice')
```



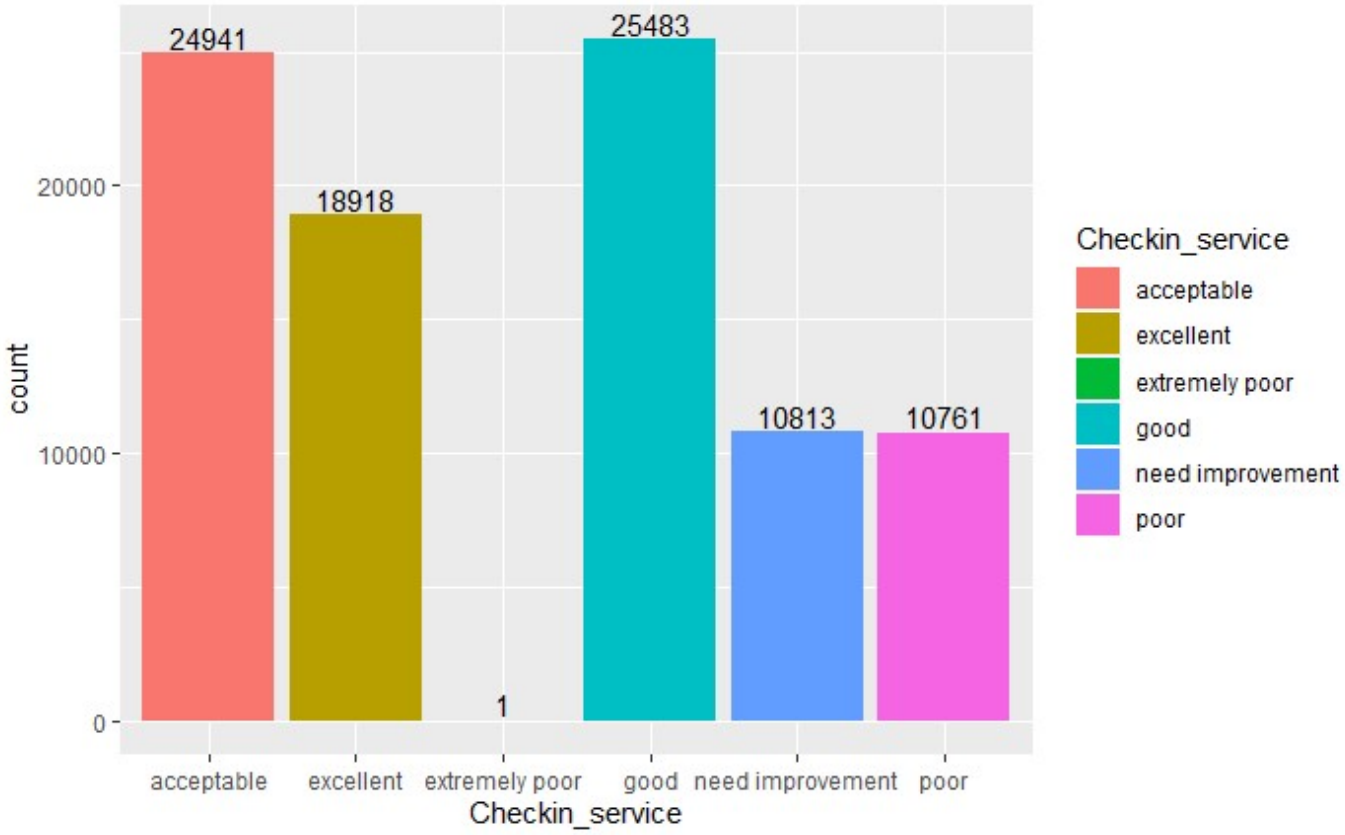
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Baggage_handling,fill=Baggage_handling)+geom_bar(posit
ion = 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Baggage_hand
ling')
```



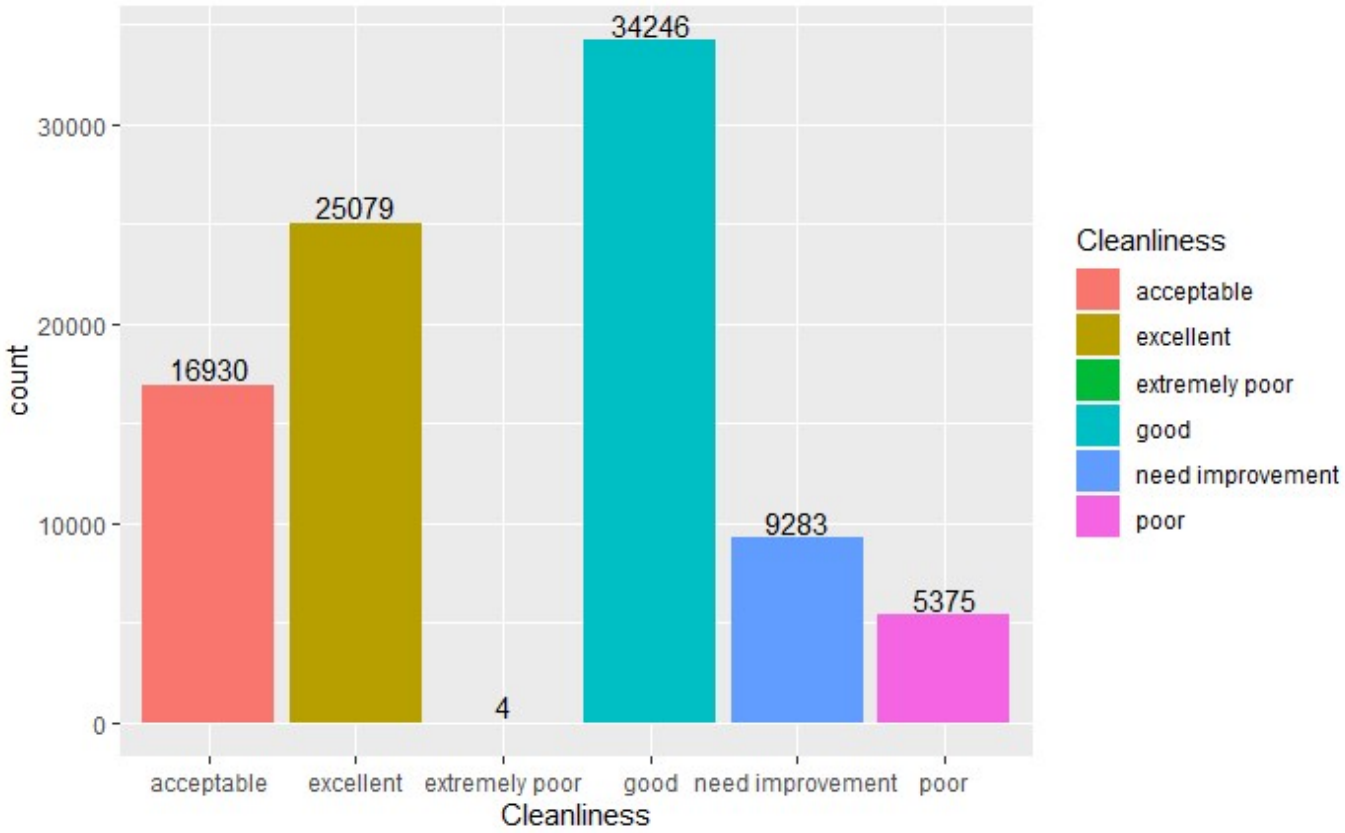
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Checkin_service,fill=Checkin_service)+geom_bar(positio
n = 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Checkin_servic
e')
```



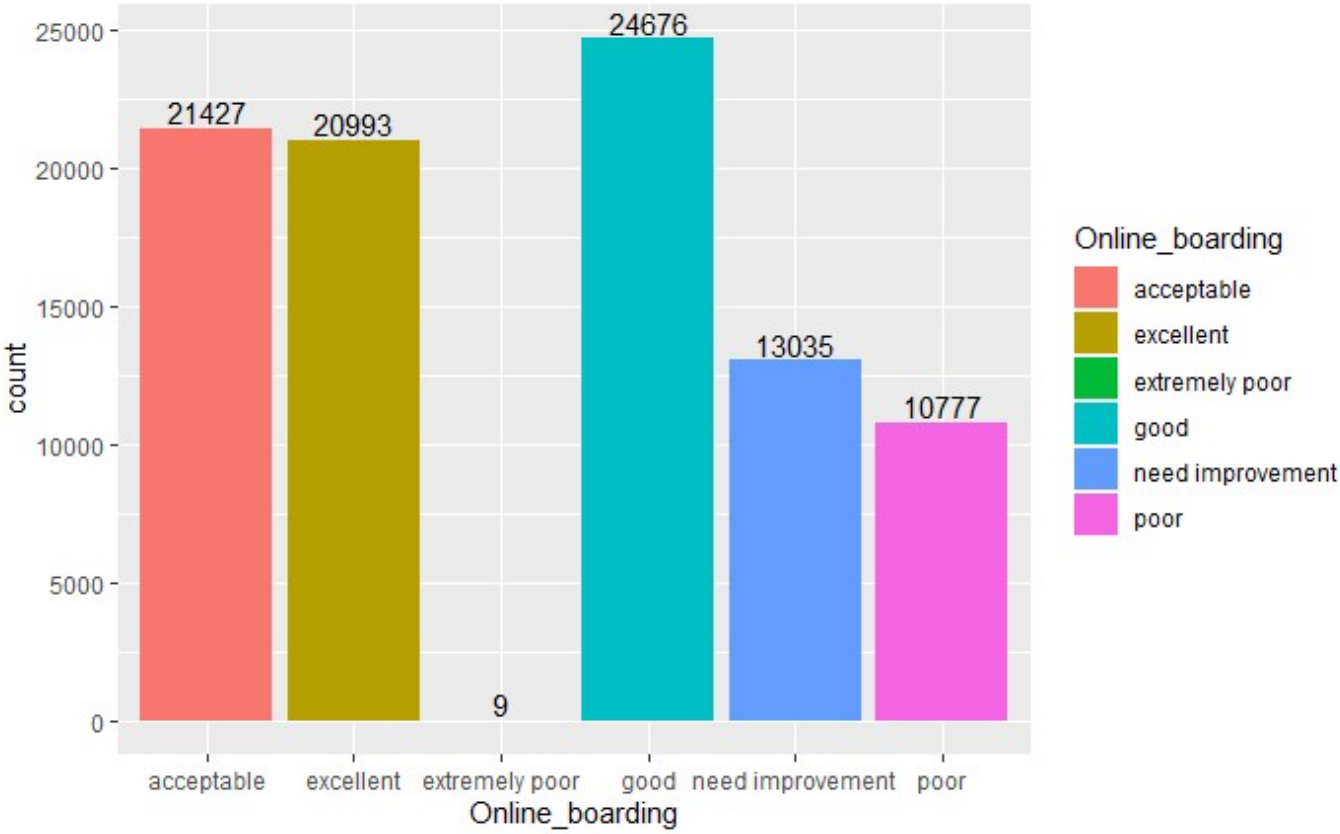
Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Cleanliness,fill=Cleanliness)+geom_bar(position = 'dod
ge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Cleanliness')
```



Hide

```
ggplot(data = Aviation)+aes(x=Aviation$Online_boarding,fill=Online_boarding)+geom_bar(positio
n = 'dodge')+geom_text(aes(label=..count..),stat = 'count',vjust=-0.2)+labs(x='Online_boardin
g')
```



Hide

```
describe(Aviation$Age)
```

	vars <dbl>	n <dbl>	mean <dbl>	sd <dbl>	median <dbl>	trimmed <dbl>	mad <dbl>	min <dbl>	max <dbl>
X1	1	90917	39.45	15.13	40	39.46	17.79	7	85

1 row | 1-10 of 13 columns

Hide

```
describe(Aviation$ArrivalDelayin_Mins)
```

	vars <dbl>	n <dbl>	mean <dbl>	sd <dbl>	median <dbl>	trimmed <dbl>	mad <dbl>	min <dbl>	max <dbl>
X1	1	90633	15.06	39.04	0	6.06	0	0	1584

1 row | 1-10 of 13 columns

Hide

```
describe(Aviation$DepartureDelayin_Mins)
```

	vars <dbl>	n <dbl>	mean <dbl>	sd <dbl>	median <dbl>	trimmed <dbl>	mad <dbl>	min <dbl>	max <dbl>
X1	1	90917	14.69	38.67	0	5.77	0	0	1592

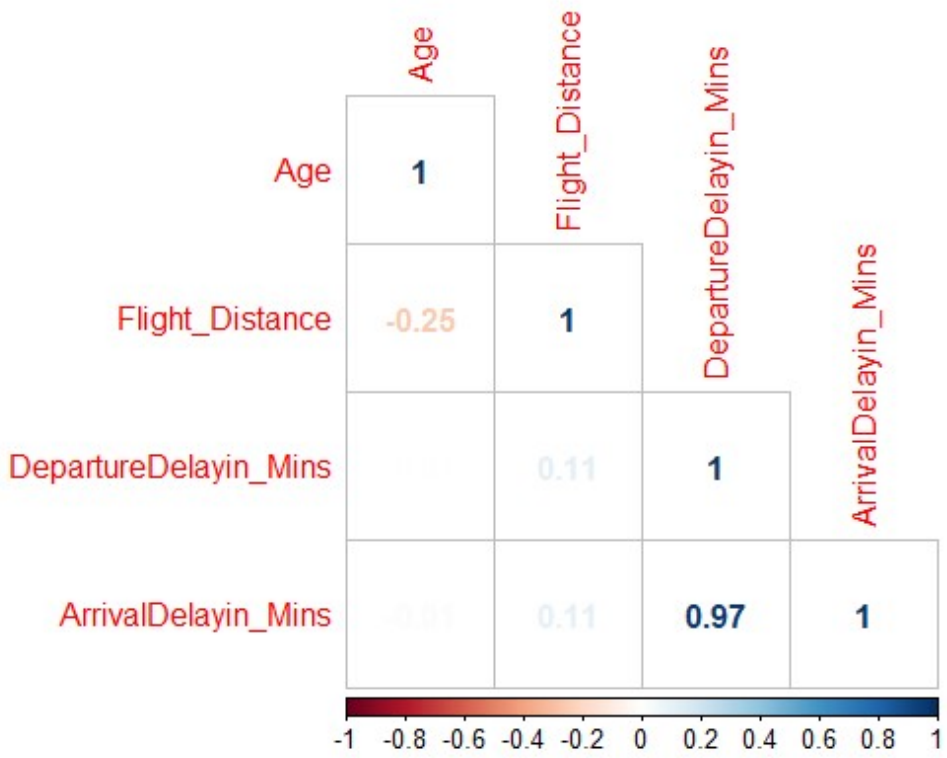
1 row | 1-10 of 13 columns

#Bi-variate Analysis

There is a strong positive correlation between Arrival Delays and Departure Delays, which signifies that flights which have arrived late also depart late for the onward journey. There is a weak negative correlation between Age and Flight distance, which signifies that people who are older prefer shorter distance flights.

Hide

```
corrplot(cor(Aviation[!is.na(Aviation$ArrivalDelayin_Mins),c(4,7,8,9)]),method = 'number',type = 'lower')
```



Since Satisfaction is our dependent variable, we will assess the dependency of other factor variables on Satisfaction. Our null hypothesis is (Ho) = Variables are independent Alternative hypothesis is (Ha) = Variables are not independent The below table illustrates the p-values from chi-square test run on each of the independent variables:

Hide

```
chisq.test(Aviation$Gender,Aviation$Satisfaction)$p.value
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$CustomerType,Aviation$Satisfaction)$p.value
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$TypeTravel,Aviation$Satisfaction)$p.value
```

```
[1] 7.140985e-220
```

Hide

```
chisq.test(Aviation$Class,Aviation$Satisfaction)$p.value
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Seat_comfort,Aviation$Satisfaction)$p.value
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Departure.Arrival.time_convenient,Aviation$Satisfaction)$p.value
```

```
[1] 1.001355e-35
```

Hide

```
chisq.test(Aviation$Food_drink,Aviation$Satisfaction)$p.value
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Gate_location,Aviation$Satisfaction)$p.value
```

```
Chi-squared approximation may be incorrect
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Inflightwifi_service,Aviation$Satisfaction)$p.value
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Inflight_entertainment,Aviation$Satisfaction)$p.value
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Online_support,Aviation$Satisfaction)$p.value
```

```
Chi-squared approximation may be incorrect
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Ease_of_Onlinebooking,Aviation$Satisfaction)$p.value
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Onboard_service,Aviation$Satisfaction)$p.value
```

```
Chi-squared approximation may be incorrect
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Leg_room_service,Aviation$Satisfaction)$p.value
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Baggage_handling,Aviation$Satisfaction)$p.value
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Checkin_service,Aviation$Satisfaction)$p.value
```

```
Chi-squared approximation may be incorrect
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Cleanliness,Aviation$Satisfaction)$p.value
```

```
Chi-squared approximation may be incorrect
```

```
[1] 0
```

Hide

```
chisq.test(Aviation$Online_boarding,Aviation$Satisfaction)$p.value
```

```
Chi-squared approximation may be incorrect
```

```
[1] 0
```

Hide

```
levels(surveydata$Online_boarding)
```

Hide

```
surveydata$Online_boarding=as.character(surveydata$Online_boarding)
```

Hide

```
surveydata$Online_boarding=ifelse(surveydata$Online_boarding=='extremely poor','poor',surveyd
ata$Online_boarding)
```

Hide

```
surveydata$Online_boarding=as.factor(surveydata$Online_boarding)
```

Hide

```
chisq.test(surveydata$Online_boarding,surveydata$Satisfaction)$p.value
```

Hide

```
surveydata$Gate_location=as.character(surveydata$Gate_location)
surveydata$Gate_location=ifelse(surveydata$Gate_location=='very inconvinient','Inconvinient',
surveydata$Gate_location)
surveydata$Gate_location=as.factor(surveydata$Gate_location)
chisq.test(surveydata$Gate_location,surveydata$Satisfaction)$p.value
```

Hide

```
levels(surveydata$Online_support)
surveydata$Online_support=as.character(surveydata$Online_support)
surveydata$Online_support=ifelse(surveydata$Online_support=='extremely poor','poor',surveydat
a$Online_support)
surveydata$Online_support=as.factor(surveydata$Online_support)
chisq.test(surveydata$Online_support,surveydata$Satisfaction)$p.value
```

Hide

```
levels(surveydata$Online_support)
surveydata$Ease_of_Onlinebooking=as.character(surveydata$Ease_of_Onlinebooking)
surveydata$Ease_of_Onlinebooking=ifelse(surveydata$Ease_of_Onlinebooking=='extremely poor','p
oor',surveydata$Ease_of_Onlinebooking)
surveydata$Ease_of_Onlinebooking=as.factor(surveydata$Ease_of_Onlinebooking)
chisq.test(surveydata$Ease_of_Onlinebooking,surveydata$Satisfaction)$p.value
```

Hide

```
levels(surveydata$Checkin_service)
surveydata$Checkin_service=as.character(surveydata$Checkin_service)
surveydata$Checkin_service=ifelse(surveydata$Checkin_service=='extremely poor','poor',surveyd
ata$Checkin_service)
surveydata$Checkin_service=as.factor(surveydata$Checkin_service)
chisq.test(surveydata$Checkin_service,surveydata$Satisfaction)$p.value
```

Hide

```
levels(surveydata$Cleanliness)
surveydata$Cleanliness=as.character(surveydata$Cleanliness)
surveydata$Cleanliness=ifelse(surveydata$Cleanliness=='extremely poor','poor',surveydata$Clea
nliness)
surveydata$Cleanliness=as.factor(surveydata$Cleanliness)
chisq.test(surveydata$Cleanliness,surveydata$Satisfaction)$p.value
```

Hide

```
levels(surveydata$Online_boarding)
surveydata$Online_boarding=as.character(surveydata$Online_boarding)
surveydata$Online_boarding=ifelse(surveydata$Online_boarding=='extremely poor','poor',surveyd
ata$Online_boarding)
surveydata$Online_boarding=as.factor(surveydata$Online_boarding)
chisq.test(surveydata$Online_boarding,surveydata$Satisfaction)$p.value
```

Hide

```
levels(surveydata$Onboard_service)
surveydata$Onboard_service=as.character(surveydata$Onboard_service)
surveydata$Onboard_service=ifelse(surveydata$Onboard_service=='extremely poor','poor',surveyd
ata$Onboard_service)
surveydata$Onboard_service=as.factor(surveydata$Onboard_service)
chisq.test(surveydata$Onboard_service,surveydata$Satisfaction)$p.value
```

Hide

```
levels(surveydata$Inflightwifi_service)
table(surveydata$Inflightwifi_service)
```

Hide

```
ggplot(data = surveydata)+aes(x=Online_boarding,fill=Satisfaction)+geom_bar(position='stac
k')+labs(x='Online_boarding')
```

Results of chi-square test shows that all the factor variables are significant. We will plot the data and try get some more insights. #Bivariate Analysis

Hide

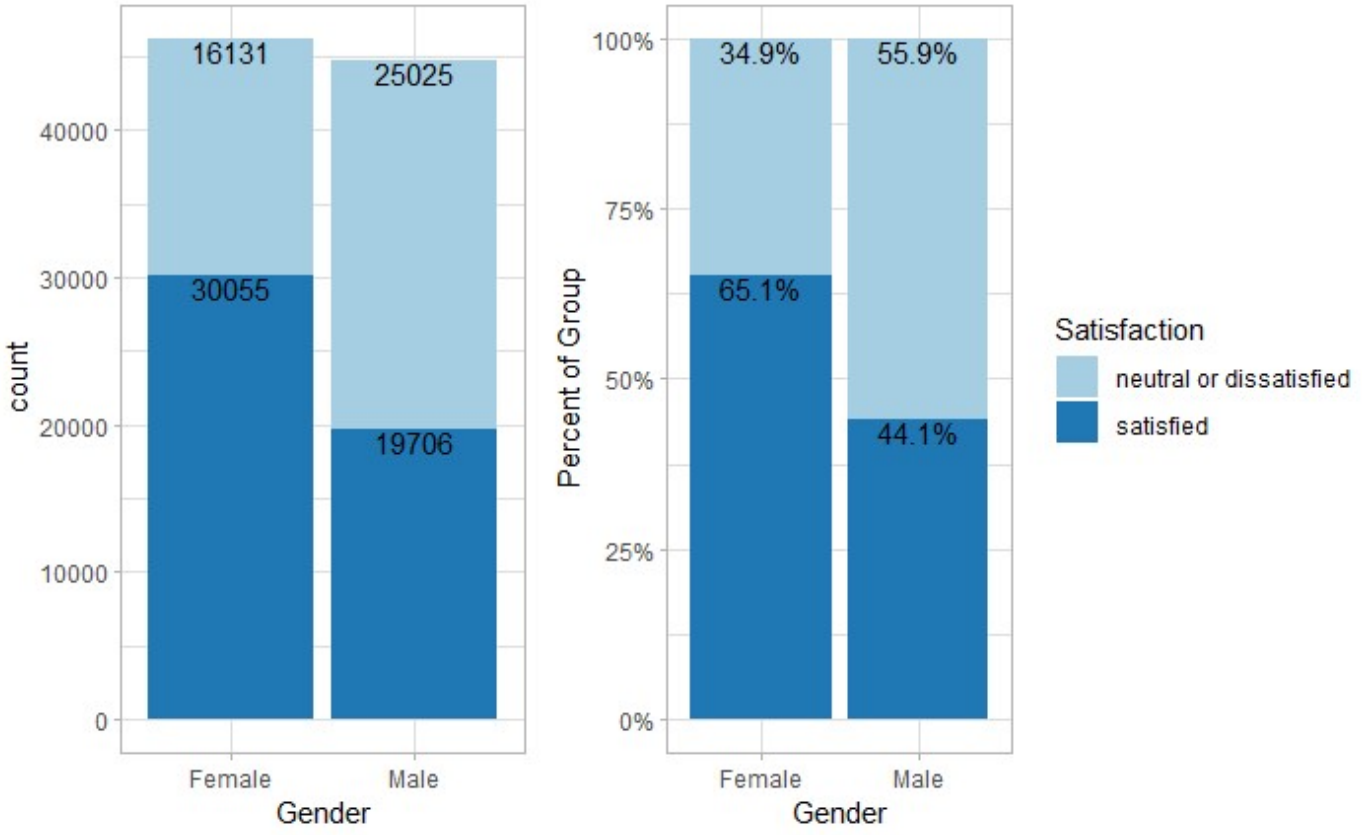
```
a=ggplot(data = Aviation)+aes(x=Gender,fill=Satisfaction)+geom_bar(position='stack')+labs(x
='Gender')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1)+scale_fil
l_brewer(palette="Paired")+theme_light()+theme(legend.position = 'None')
```

Hide

```
b=ggplot(data = Aviation)+aes(x=Gender,fill=Satisfaction,y=..count../tapply(..count..,..x..,s
um)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes(label=perce
nt(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stack', vjust=
1)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paired")+theme_light()+the
me(legend.position = 'right')
```

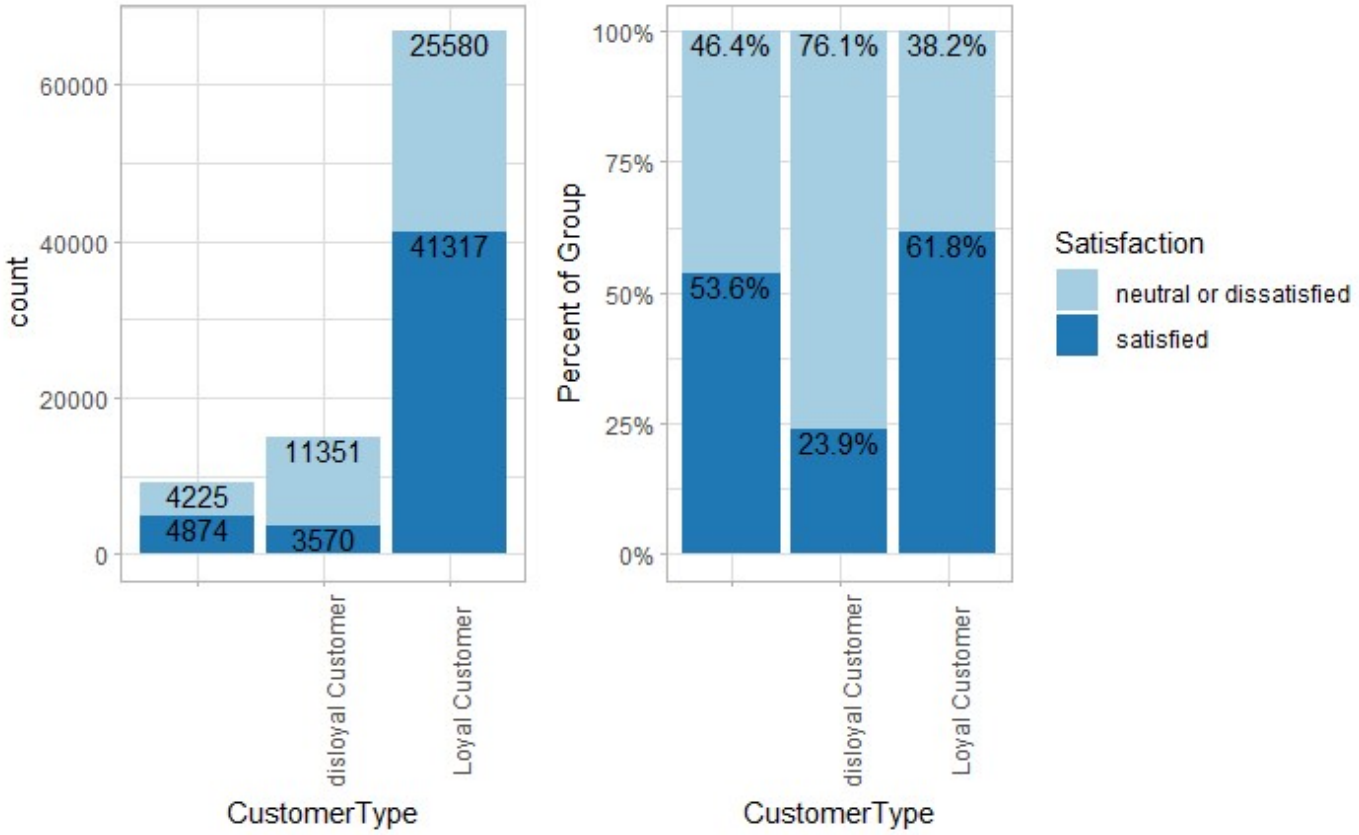
Hide

```
ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(4,6))
```



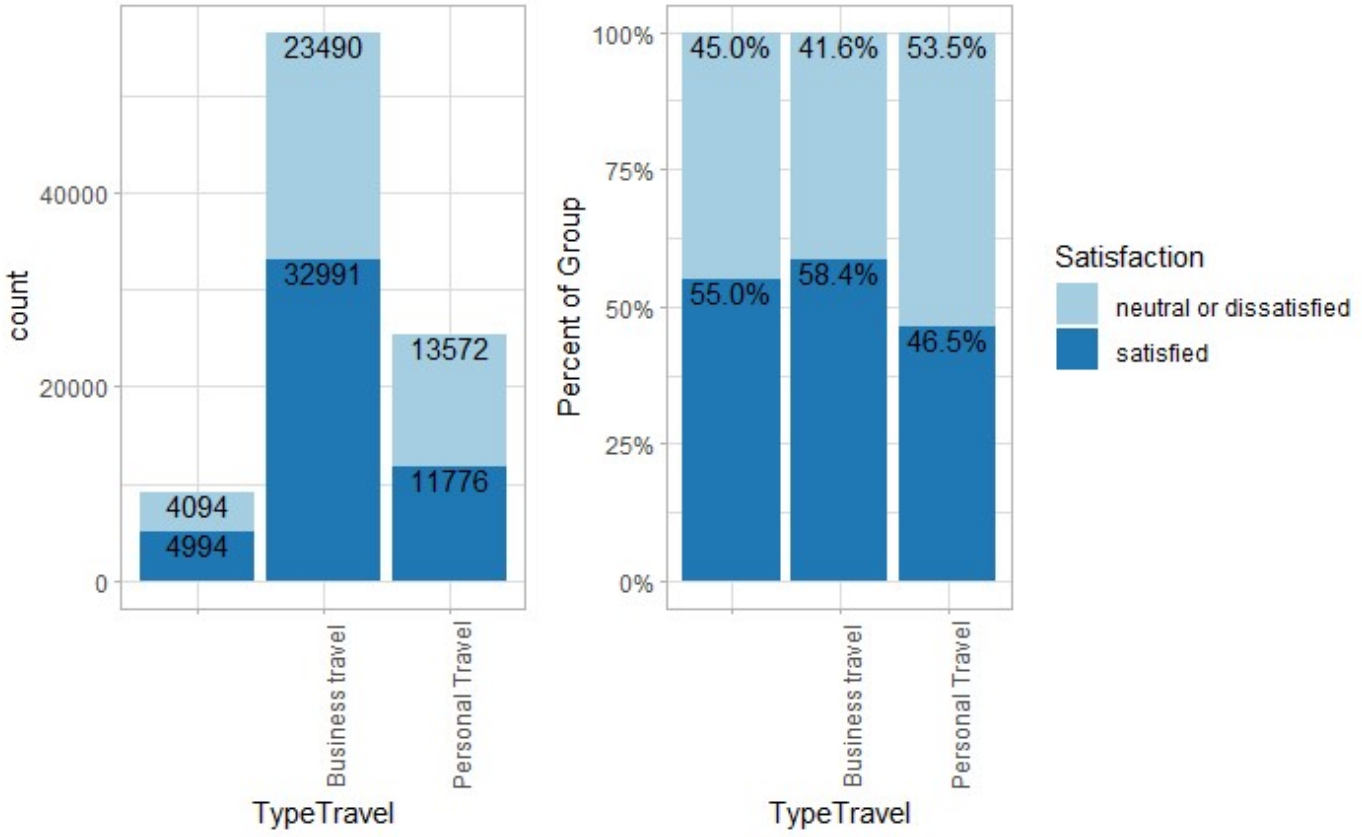
Hide

```
a=ggplot(data = Aviation)+aes(x=CustomerType,fill=Satisfaction)+geom_bar(position='stack')+ge
om_text(aes(label=..count..),stat = 'count',position='stack', vjust=1)+scale_fill_brewer(pale
tte="Paired")+theme_light()+theme(legend.position = 'None')+theme(axis.text.x=element_text(an
gle = 90))

b=ggplot(data = Aviation)+aes(x=CustomerType,fill=Satisfaction,y=..count../tapply(..coun
t..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes
(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stac
k', vjust=1)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paired")+theme_l
ight()+theme(axis.text.x=element_text(angle = 90),legend.position = 'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(4,6))
```



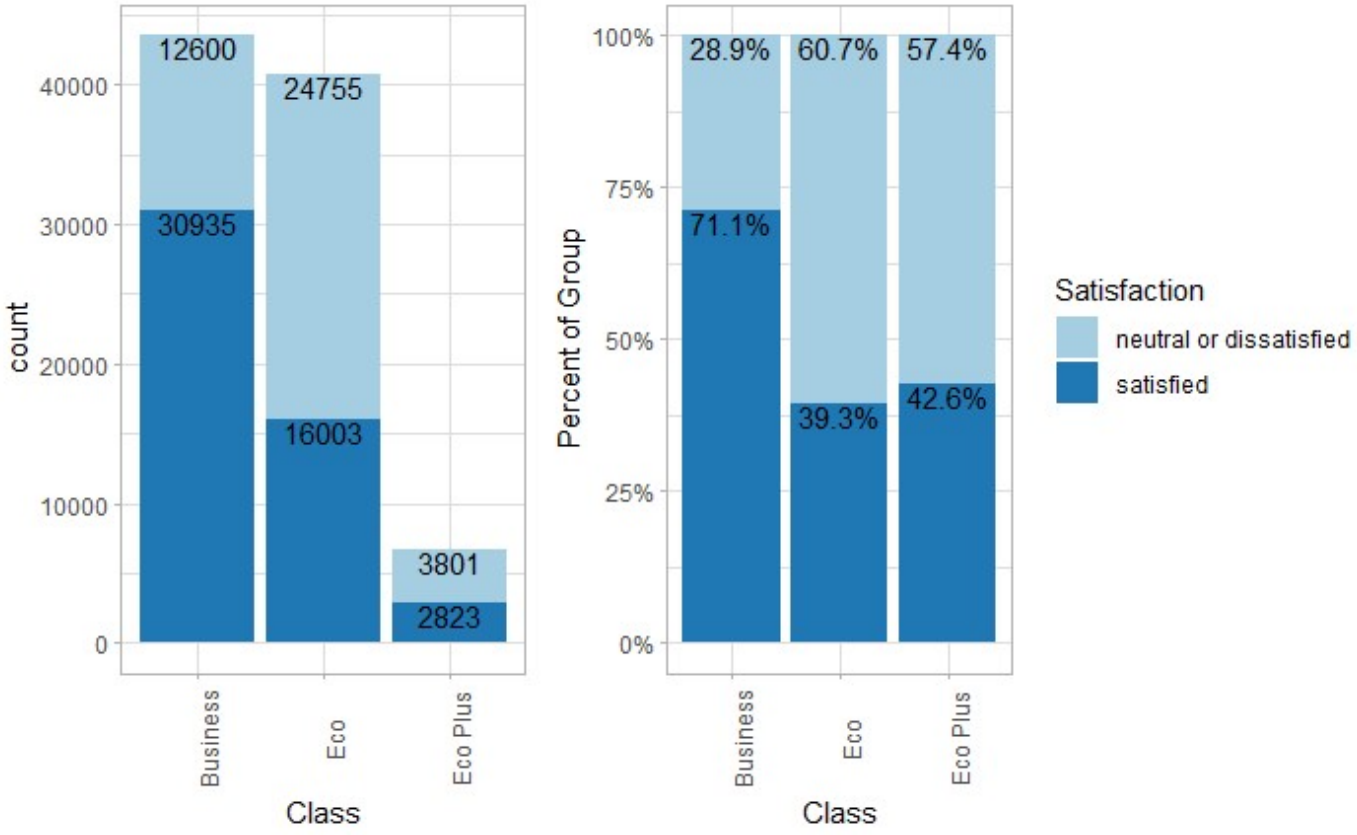
Hide

```
a=ggplot(data = Aviation)+aes(x=TypeTravel,fill=Satisfaction)+geom_bar(position='stack')+geom
_text(aes(label=..count..),stat = 'count',position='stack', vjust=1)+scale_fill_brewer(palett
e="Paired")+theme_light()+theme(legend.position = 'None')+theme(axis.text.x=element_text(angl
e = 90))

b=ggplot(data = Aviation)+aes(x=TypeTravel,fill=Satisfaction,y=..count../tapply(..count..,..
x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes(label=
percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stack', vju
st=1)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paired")+theme_light
()+theme(axis.text.x=element_text(angle = 90),legend.position = 'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(4,6))
```



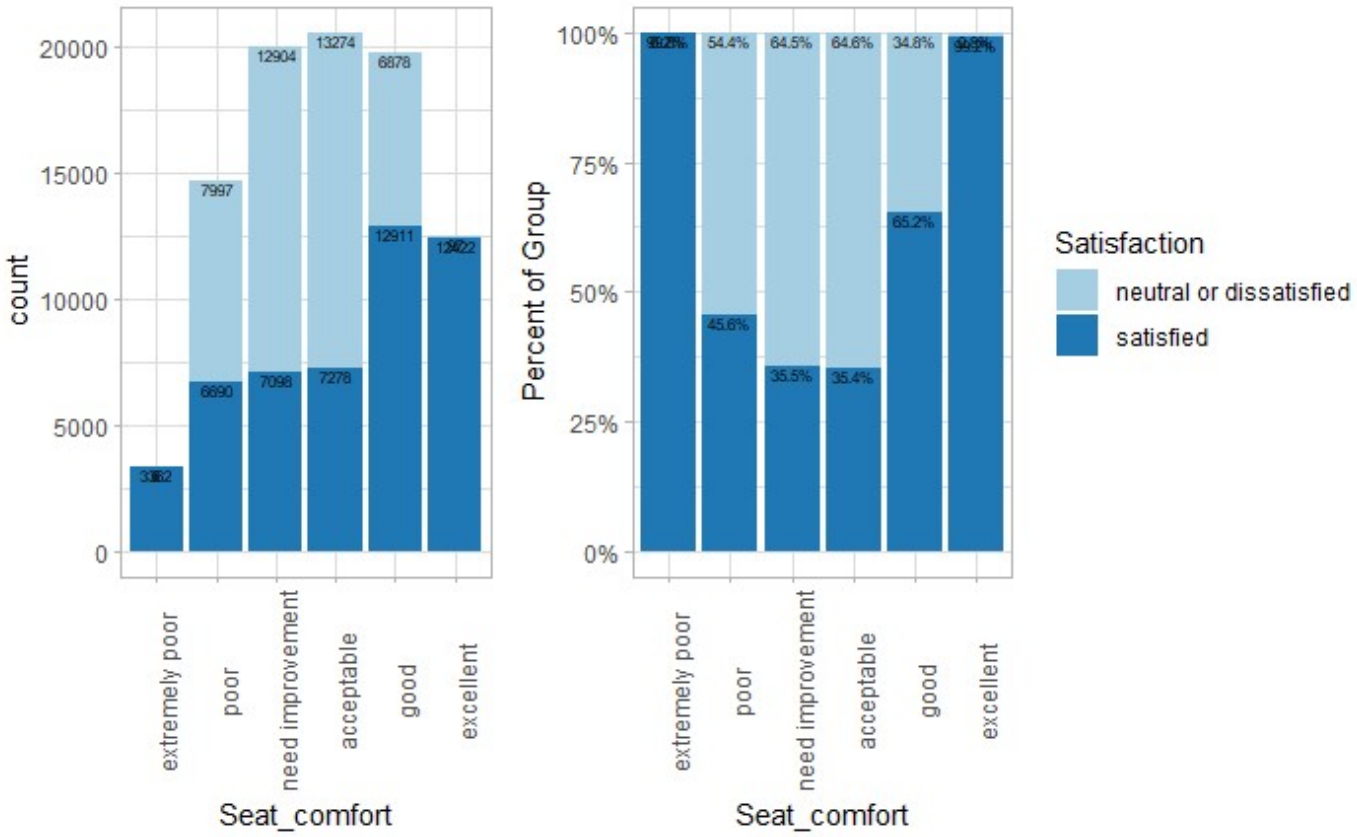
Hide

```
a=ggplot(data = Aviation)+aes(x=Class,fill=Satisfaction)+geom_bar(position='stack')+geom_text
(aes(label=..count..),stat = 'count',position='stack', vjust=1)+scale_fill_brewer(palette="Pa
ired")+theme_light()+theme(legend.position = 'None')+theme(axis.text.x=element_text(angle = 9
0))

b=ggplot(data = Aviation)+aes(x=Class,fill=Satisfaction,y=..count../tapply(..count..,..x..,su
m)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes(label=percen
t(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stack', vjust=
1)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paired")+theme_light()+the
me(axis.text.x=element_text(angle = 90),legend.position = 'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(4,6))
```



Hide

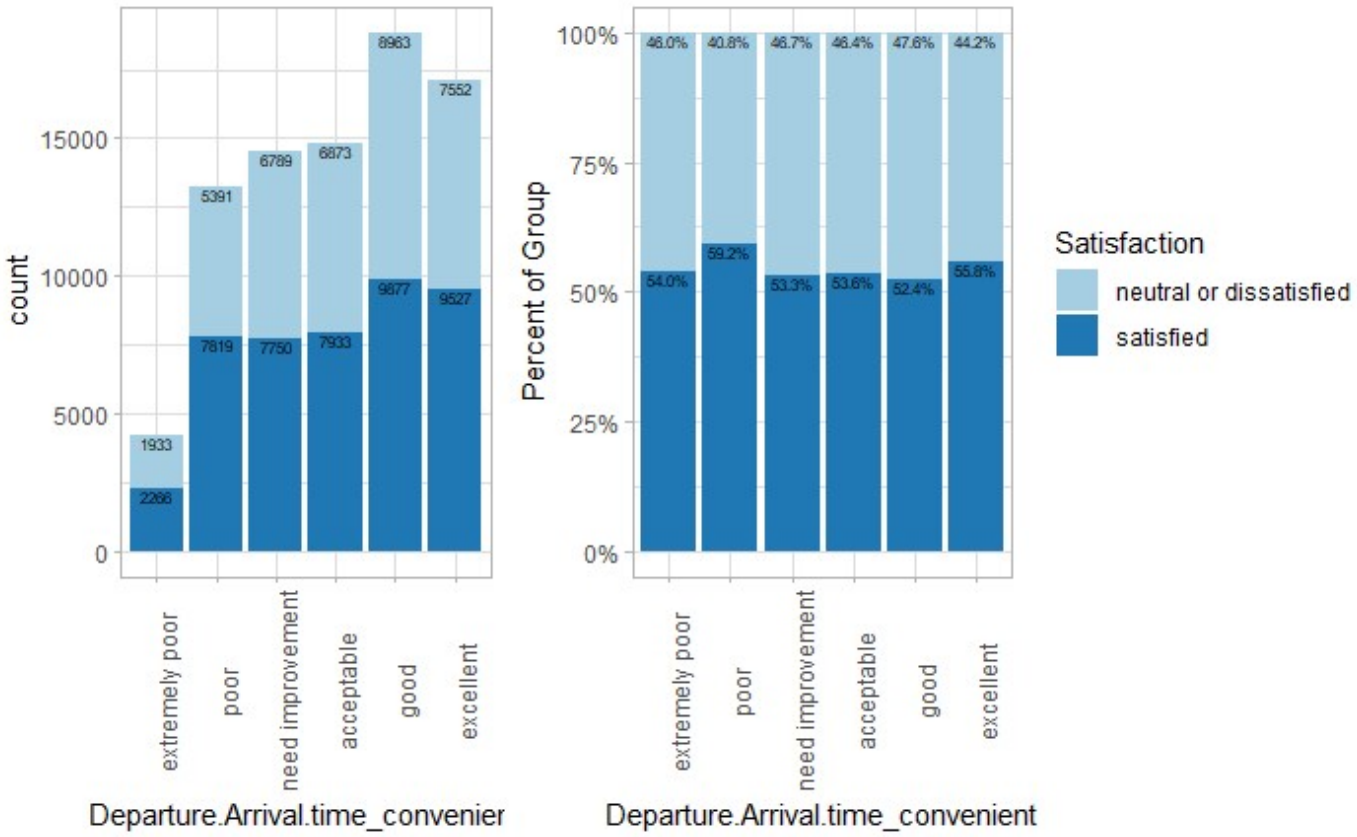
```
a=ggplot(data = Aviation)+aes(x=Seat_comfort,fill=Satisfaction)+geom_bar(position='stack')+ge
om_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_fill_brew
er(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","ac
ceptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(axis.text.
x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Seat_comfort,fill=Satisfaction,y=..count../tapply(..coun
t..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes
(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stac
k', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paire
d")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acceptable","goo
d","excellent"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position =
'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



Hide

```
a=ggplot(data = Aviation)+aes(x=Departure.Arrival.time_convenient,fill=Satisfaction)+geom_bar
(position='stack')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,si
ze=2)+scale_fill_brewer(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poo
r","need improvement","acceptable","good","excellent"))+theme_light()+theme(legend.position =
'None')+theme(axis.text.x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Departure.Arrival.time_convenient,fill=Satisfaction,y=..coun
t../tapply(..count..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Grou
p')+geom_text(aes(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'coun
t',position='stack', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(p
alette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","accept
able","good","excellent"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.po
sition = 'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```

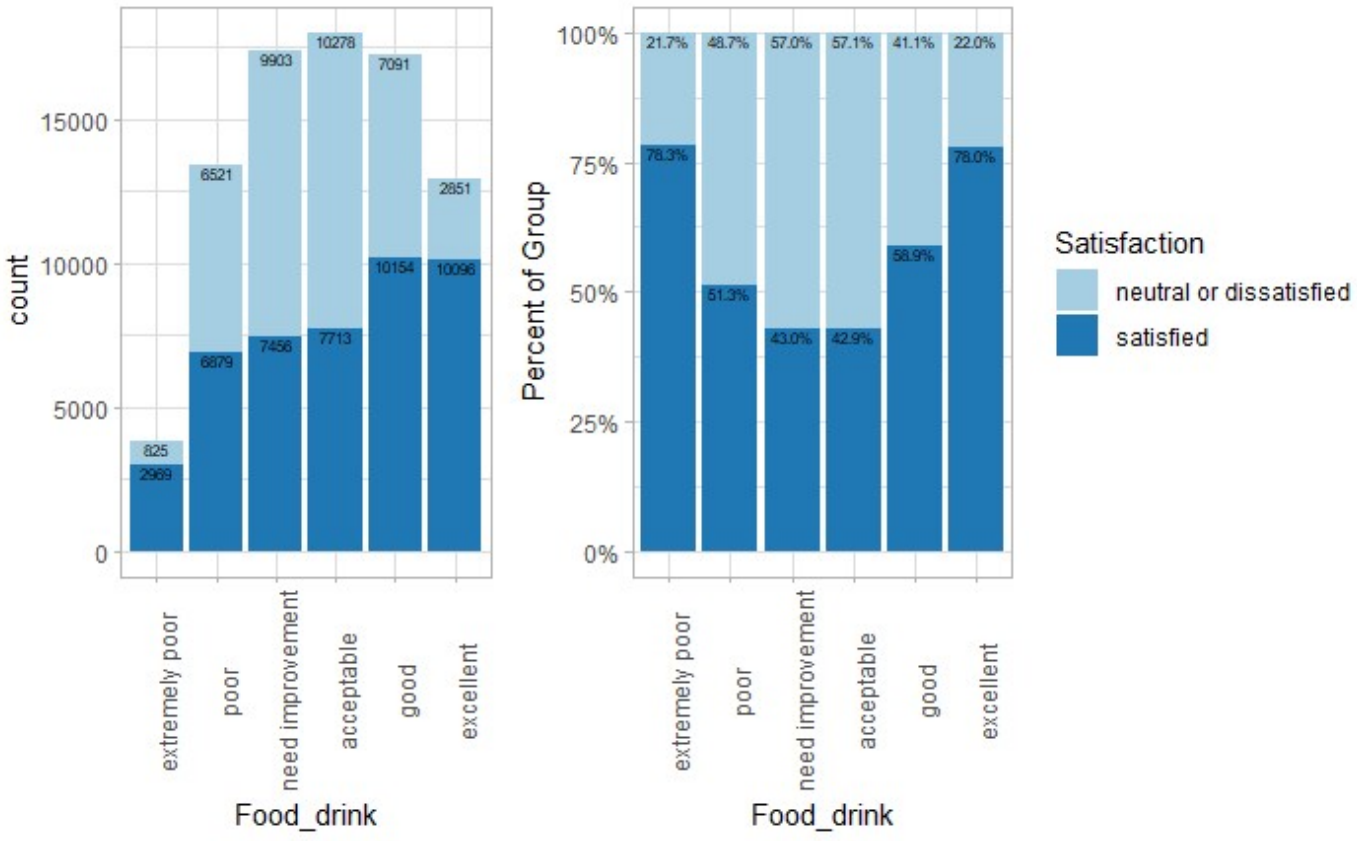
```
Removed 8244 rows containing non-finite values (stat_count).Removed 8244 rows containing non-
finite values (stat_count).Removed 8244 rows containing non-finite values (stat_count).Remove
d 8244 rows containing non-finite values (stat_count).
```



Hide

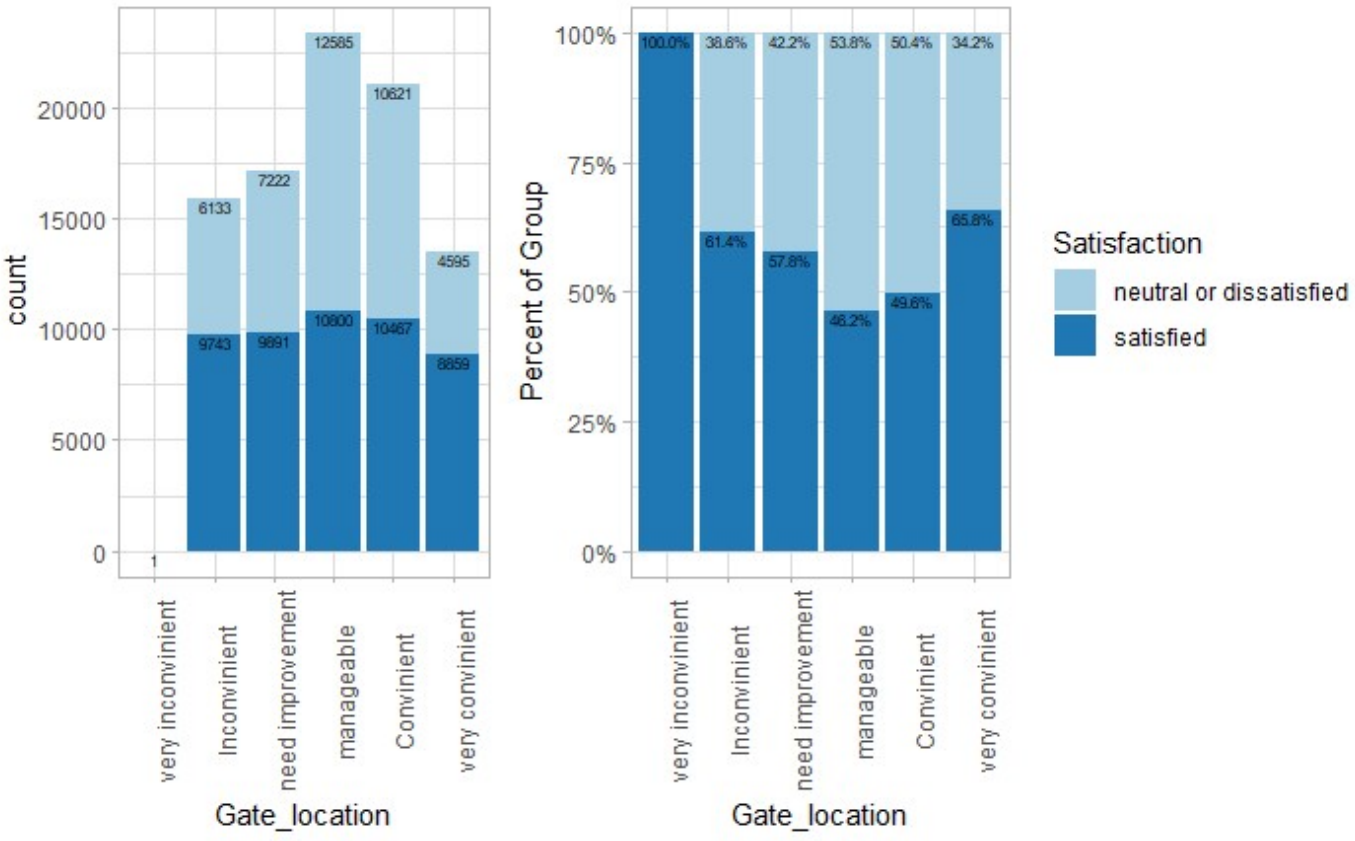
```
a=ggplot(data = Aviation)+aes(x=Food_drink,fill=Satisfaction)+geom_bar(position='stack')+geom
_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_fill_brewer
(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acce
ptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(axis.text.x=
element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Food_drink,fill=Satisfaction,y=..count../tapply(..count..,..
x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes(label=
percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stack', vju
st=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paired")+scale_x
_discrete(limits=c("extremely poor","poor","need improvement","acceptable","good","excellen
t"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position = 'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```

```
Removed 8181 rows containing non-finite values (stat_count).Removed 8181 rows containing non-
finite values (stat_count).Removed 8181 rows containing non-finite values (stat_count).Remove
d 8181 rows containing non-finite values (stat_count).
```



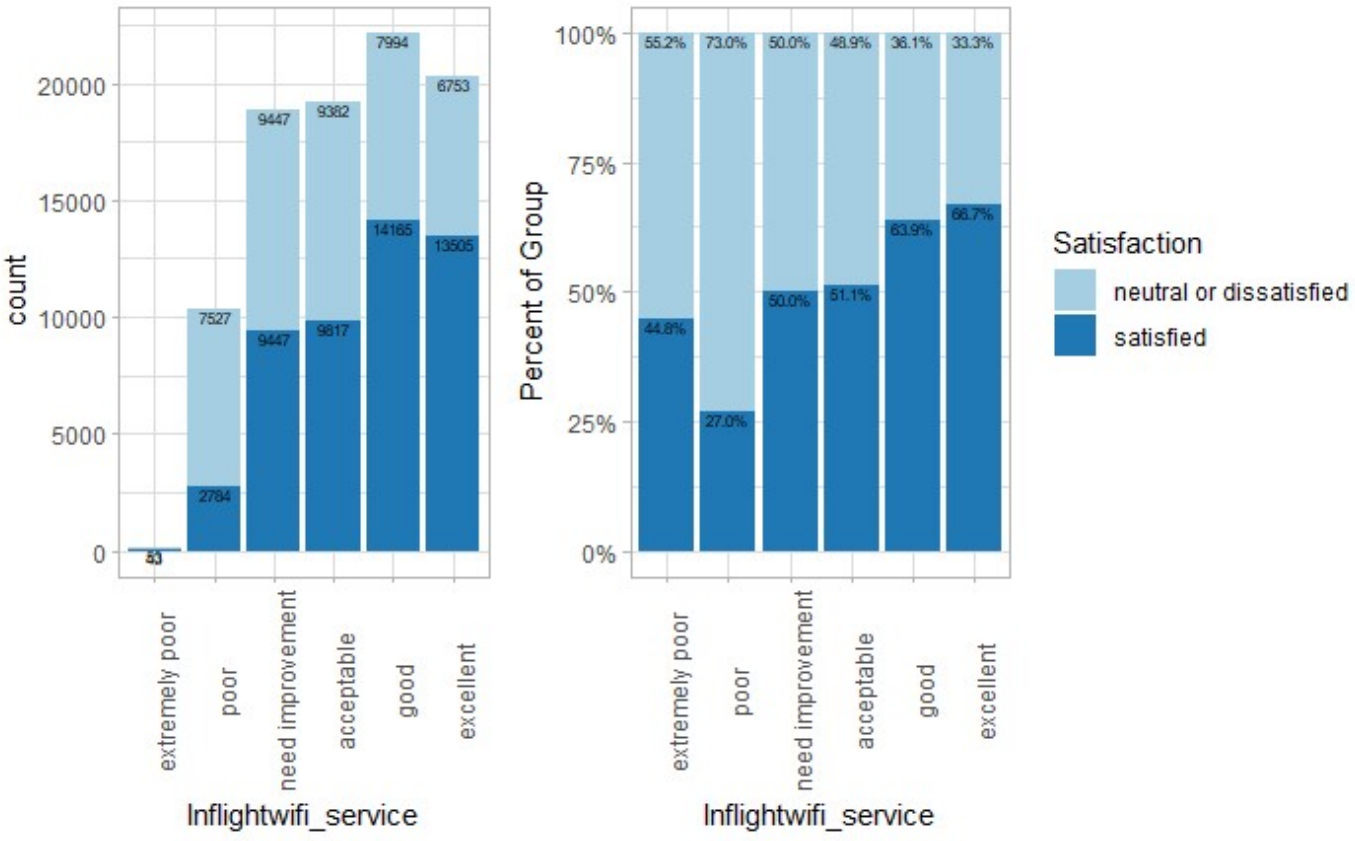
Hide

```
a=ggplot(data = Aviation)+aes(x=Gate_location,fill=Satisfaction)+geom_bar(position='stack')+g
eom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_fill_bre
wer(palette="Paired")+scale_x_discrete(limits=c("very inconvinient","Inconvinient","need impr
ovement","manageable","Convinient","very convinient"))+theme_light()+theme(legend.position =
'None')+theme(axis.text.x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Gate_location,fill=Satisfaction,y=..count../tapply(..coun
t..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes
(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stac
k', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paire
d")+scale_x_discrete(limits=c("very inconvinient","Inconvinient","need improvement","manageab
le","Convinient","very convinient"))+theme_light()+theme(axis.text.x=element_text(angle = 9
0),legend.position = 'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



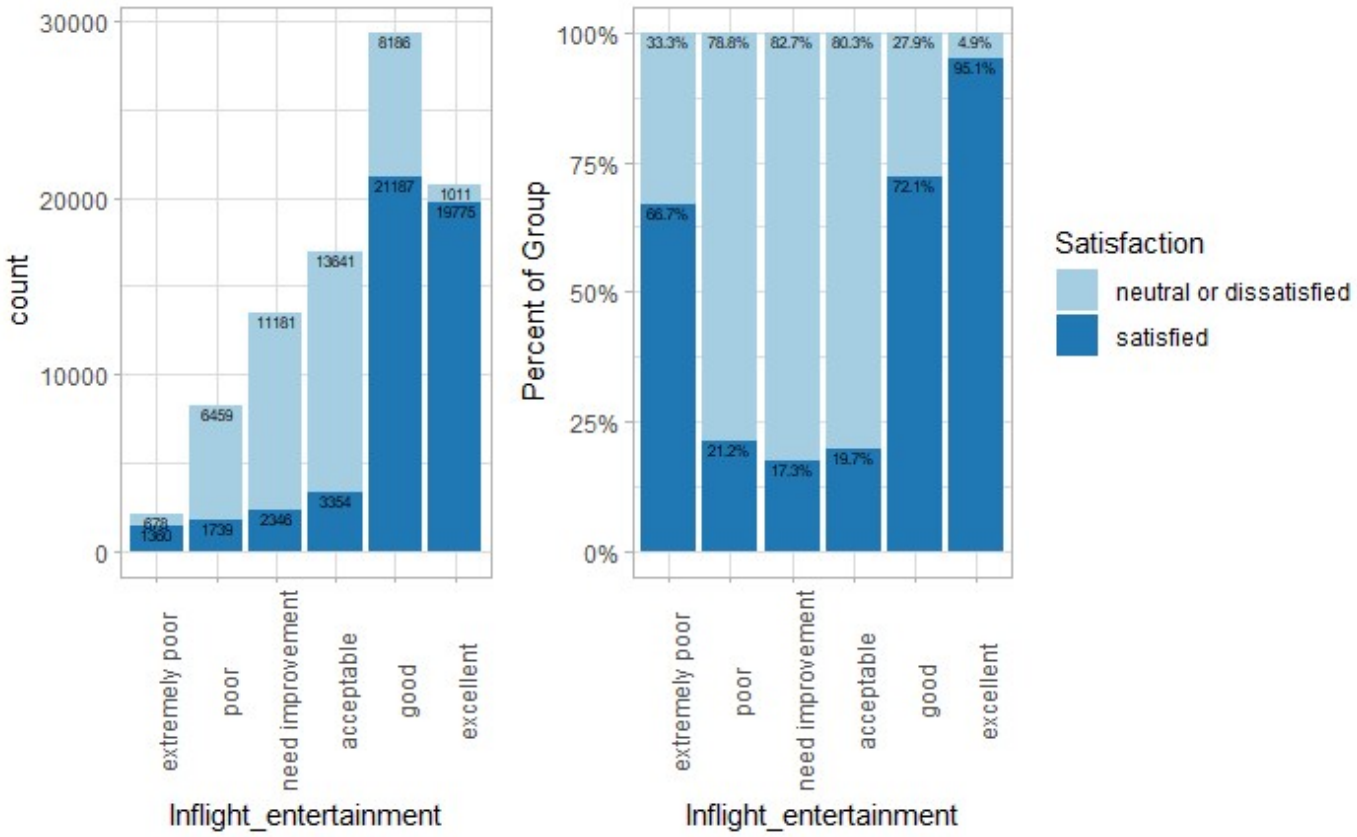
Hide

```
a=ggplot(data = Aviation)+aes(x=Inflightwifi_service,fill=Satisfaction)+geom_bar(position='st
ack')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_f
ill_brewer(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improvem
ent","acceptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(ax
is.text.x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Inflightwifi_service,fill=Satisfaction,y=..count../tapply(..c
ount..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(a
es(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='st
ack', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paire
d")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acceptable","goo
d","excellent"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position =
'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



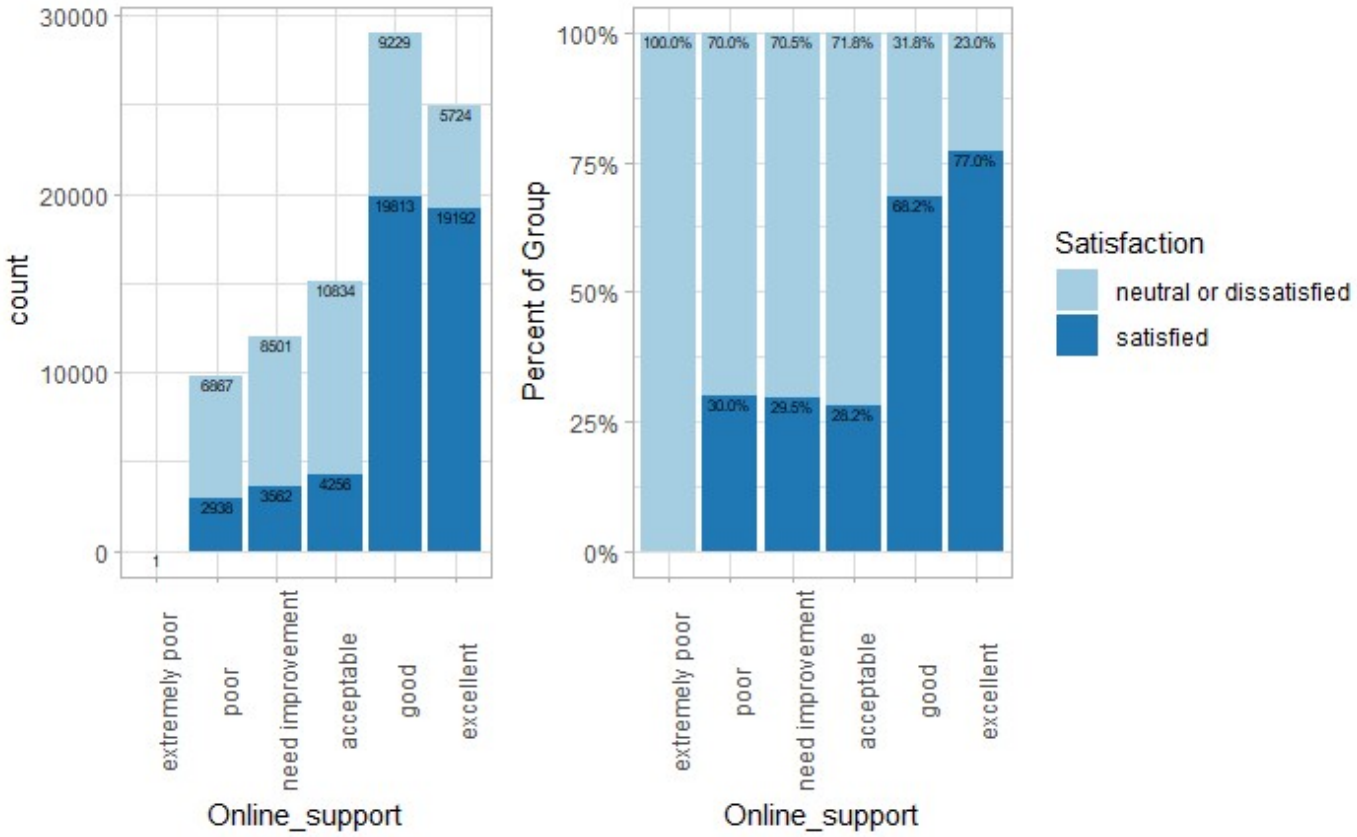
Hide

```
a=ggplot(data = Aviation)+aes(x=Inflight_entertainment,fill=Satisfaction)+geom_bar(position
='stack')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+sca
le_fill_brewer(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need impr
ovement","acceptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+them
e(axis.text.x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Inflight_entertainment,fill=Satisfaction,y=..count../tapply
(..count..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_te
xt(aes(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position
='stack', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Pai
red")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acceptable","goo
d","excellent"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position =
'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



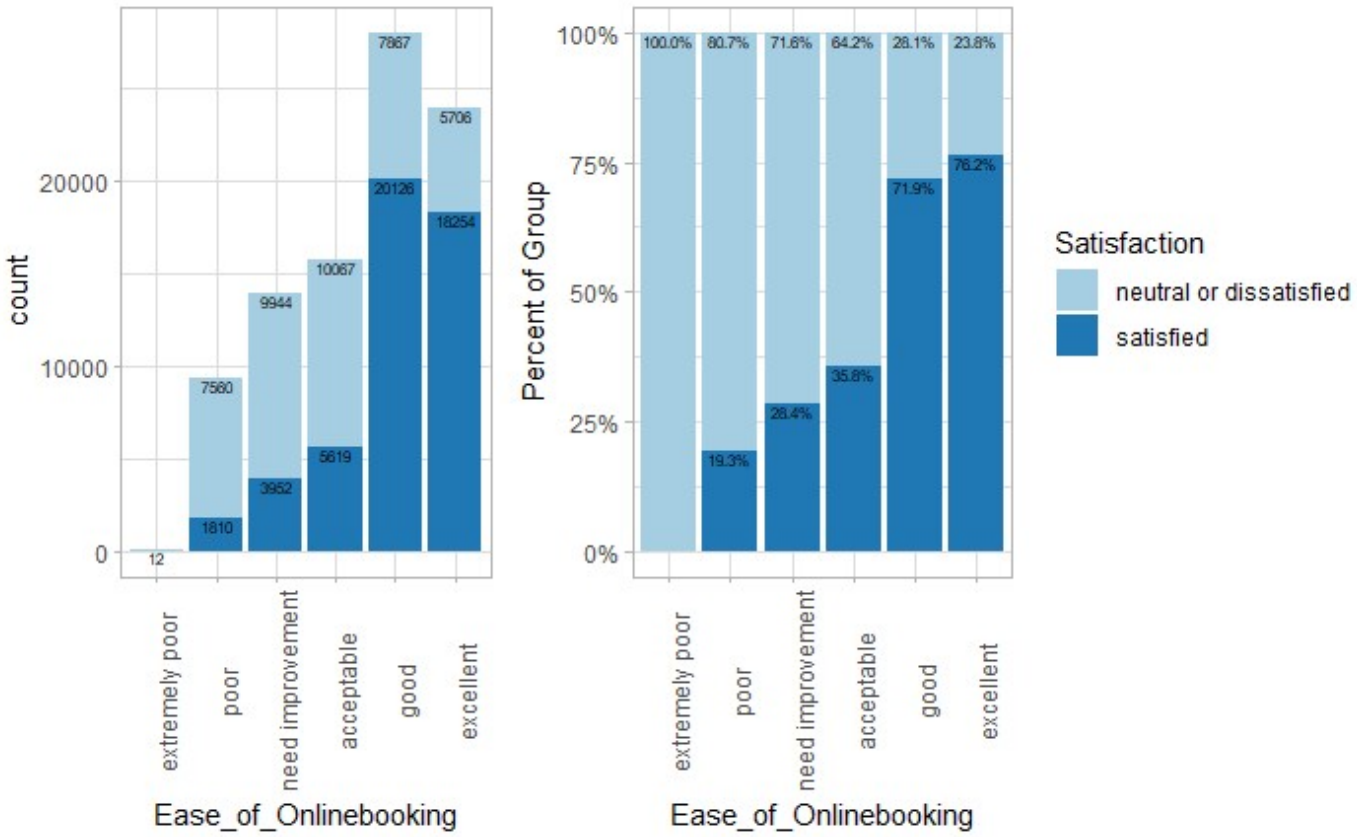
Hide

```
a=ggplot(data = Aviation)+aes(x=Online_support,fill=Satisfaction)+geom_bar(position='stac
k')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_fil
l_brewer(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improvemen
t","acceptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(axi
s.text.x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Online_support,fill=Satisfaction,y=..count../tapply(..coun
t..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes
(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stac
k', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paire
d")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acceptable","goo
d","excellent"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position =
'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



Hide

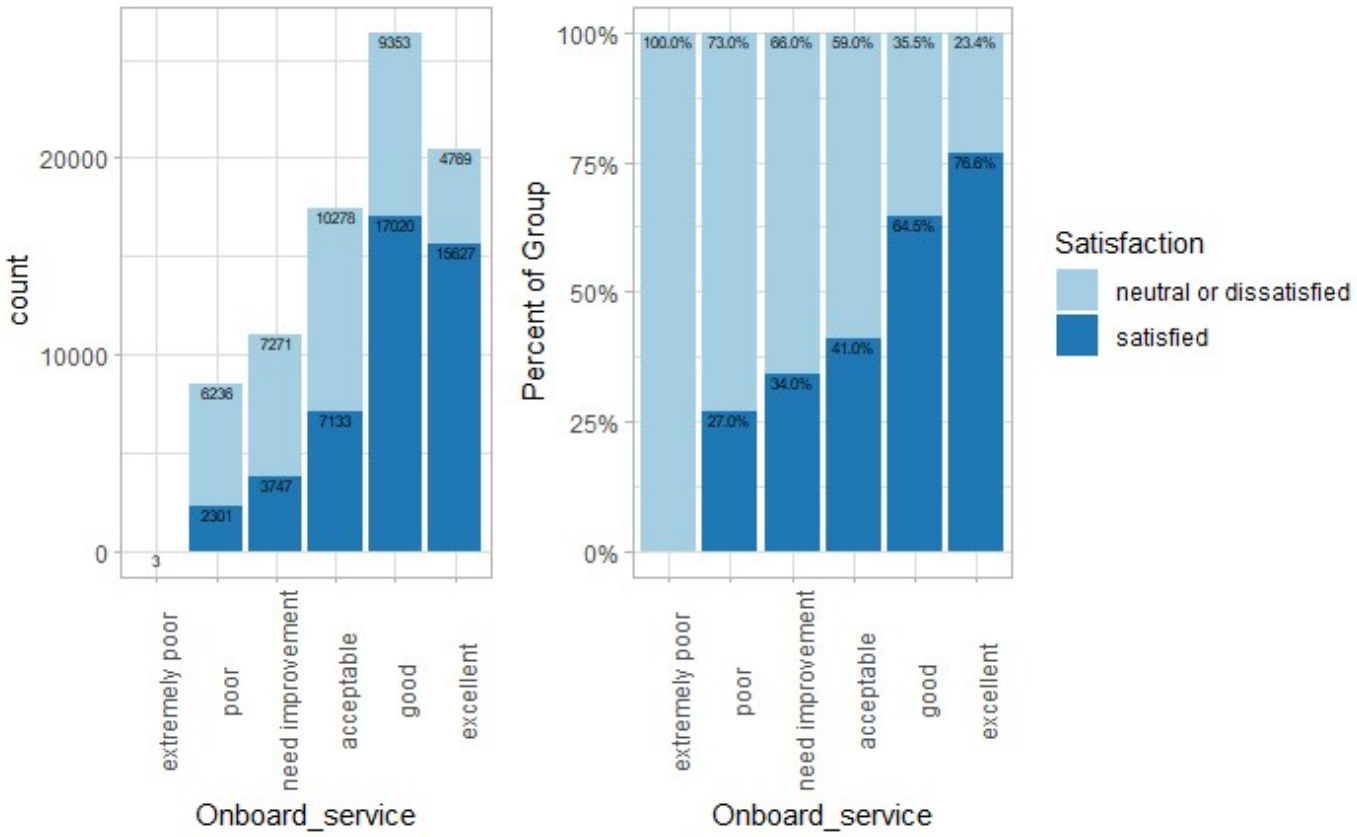
```
a=ggplot(data = Aviation)+aes(x=Ease_of_Onlinebooking,fill=Satisfaction)+geom_bar(position='s
tack')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_
fill_brewer(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improve
ment","acceptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(a
xis.text.x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Ease_of_Onlinebooking,fill=Satisfaction,y=..count../tapply(..
count..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text
(aes(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position
='stack', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Pai
red")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acceptable","goo
d","excellent"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position =
'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



Hide

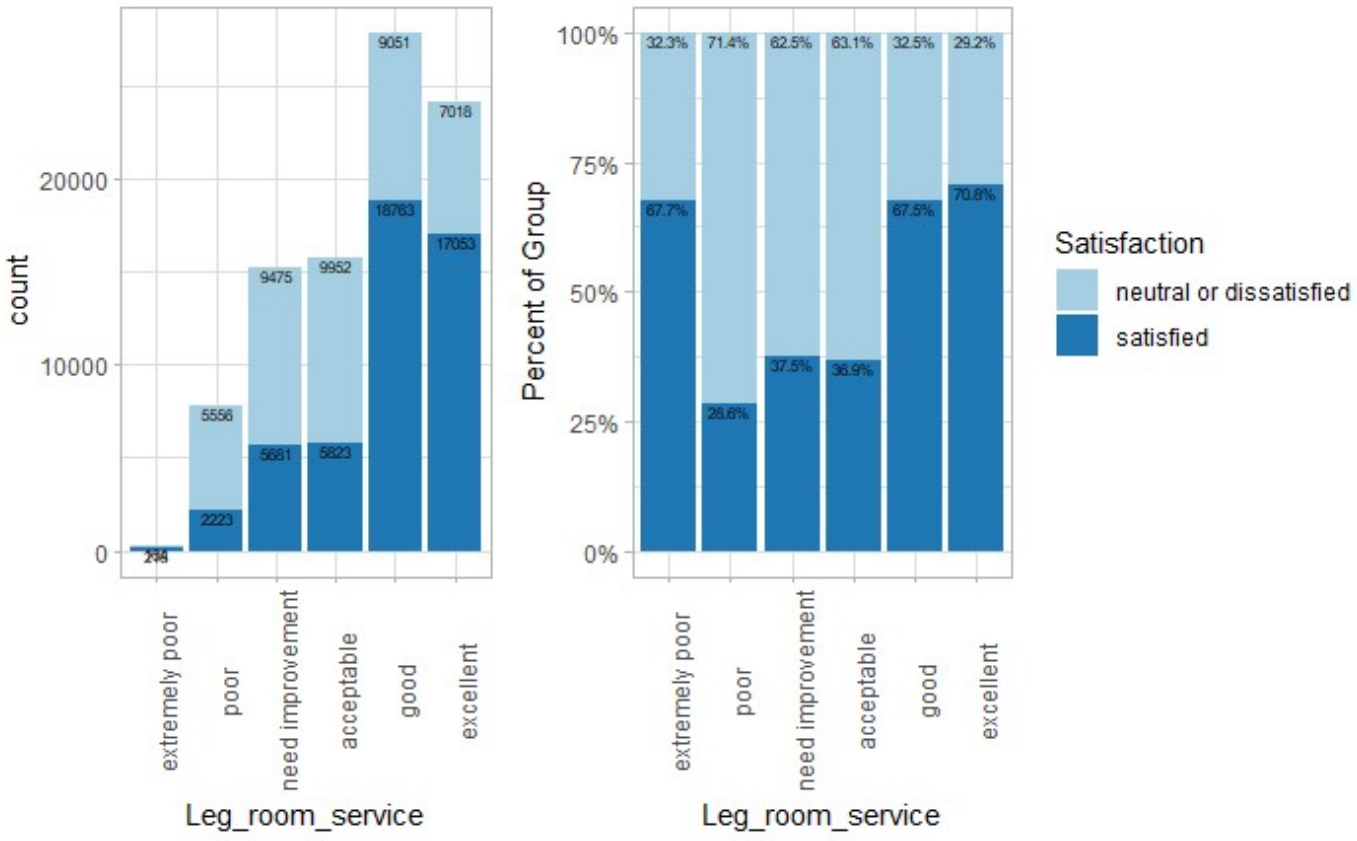
```
a=ggplot(data = Aviation)+aes(x=Onboard_service,fill=Satisfaction)+geom_bar(position='stac
k')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_fil
l_brewer(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improvemen
t","acceptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(axi
s.text.x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Onboard_service,fill=Satisfaction,y=..count../tapply(..coun
t..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes
(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stac
k', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paire
d")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acceptable","goo
d","excellent"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position =
'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```

```
Removed 7179 rows containing non-finite values (stat_count).Removed 7179 rows containing non-
finite values (stat_count).Removed 7179 rows containing non-finite values (stat_count).Remove
d 7179 rows containing non-finite values (stat_count).
```



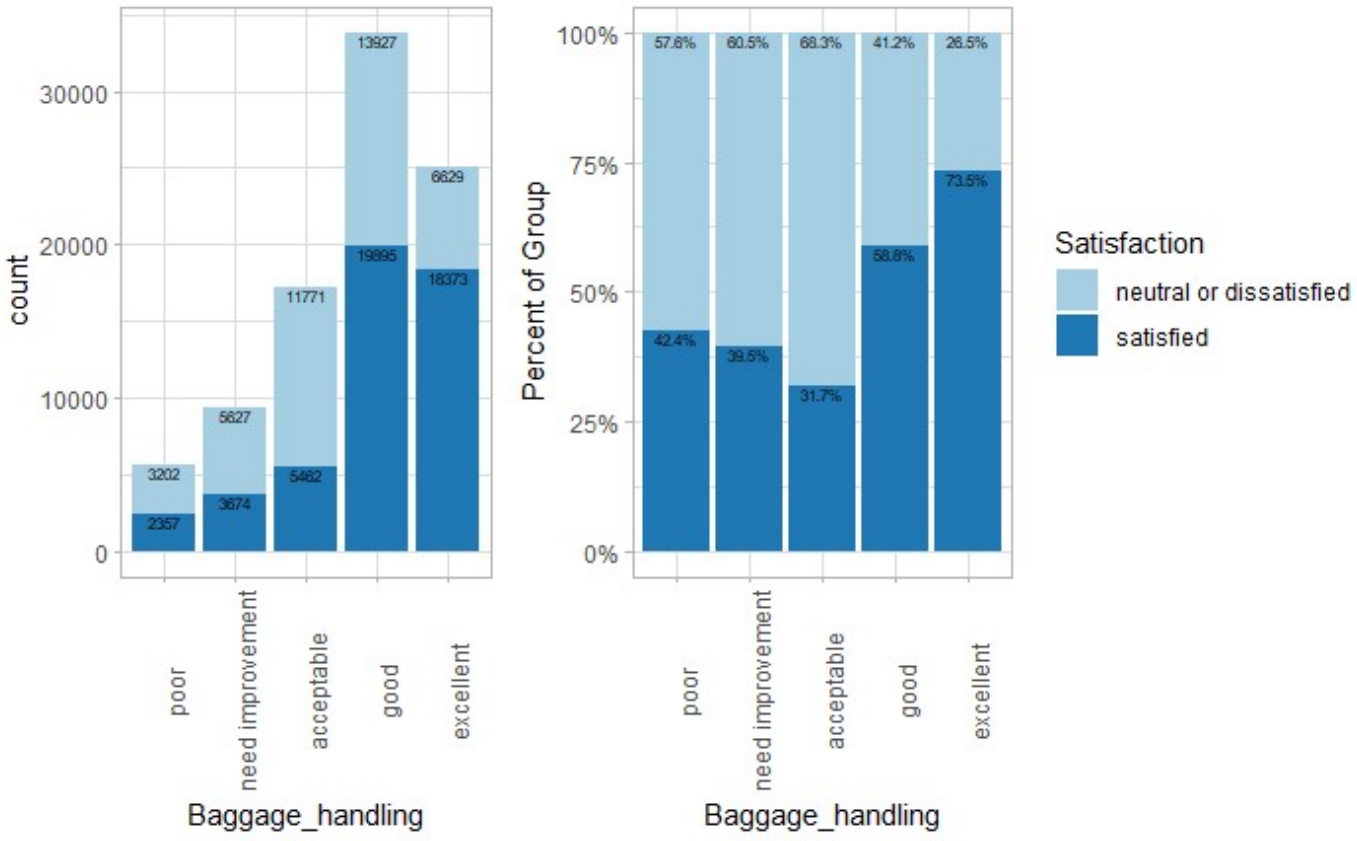
Hide

```
a=ggplot(data = Aviation)+aes(x=Leg_room_service,fill=Satisfaction)+geom_bar(position='stac
k')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_fil
l_brewer(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improvemen
t","acceptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(axi
s.text.x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Leg_room_service,fill=Satisfaction,y=..count../tapply(..coun
t..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes
(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stac
k', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paire
d")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acceptable","goo
d","excellent"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position =
'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



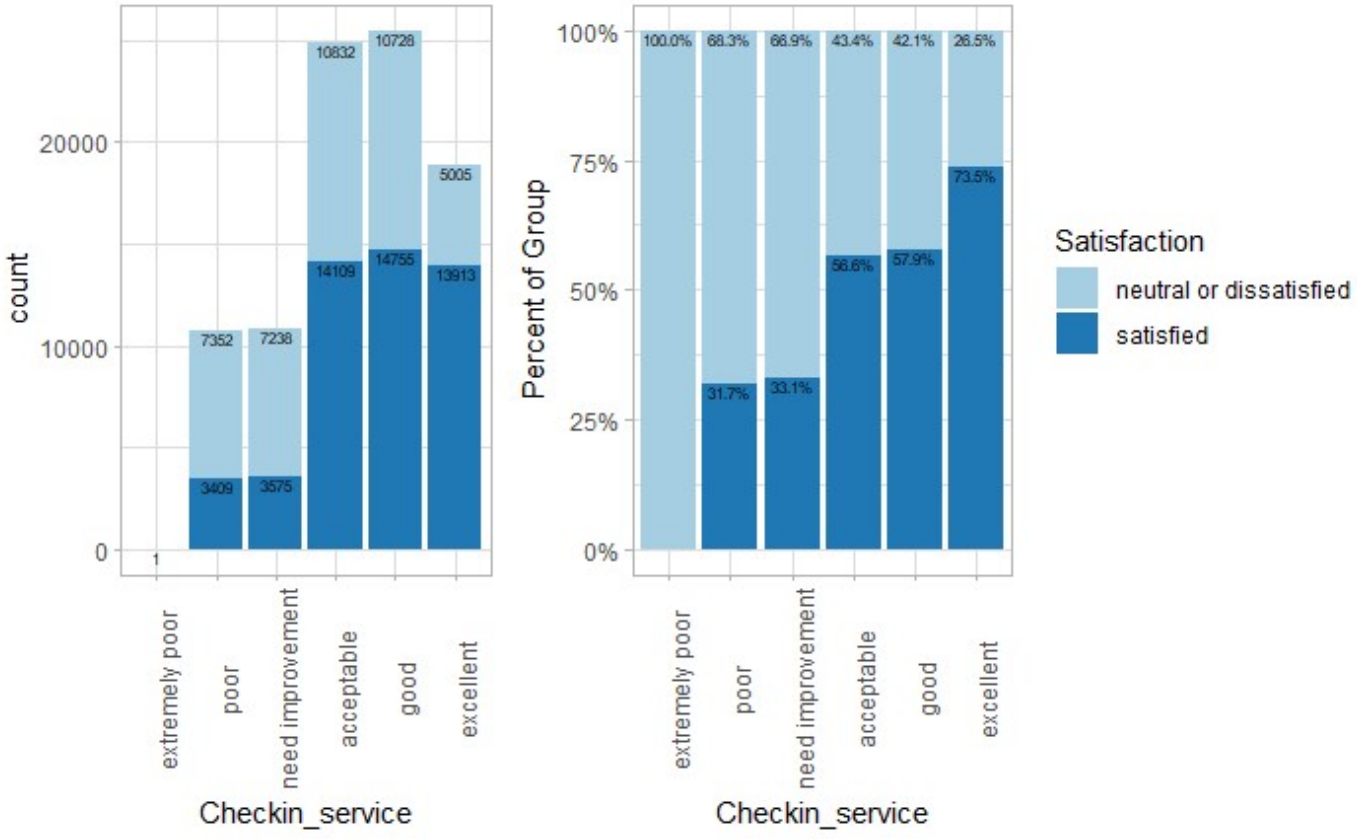
Hide

```
a=ggplot(data = Aviation)+aes(x=Baggage_handling,fill=Satisfaction)+geom_bar(position='stac
k')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_fil
l_brewer(palette="Paired")+scale_x_discrete(limits=c("poor","need improvement","acceptabl
e","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(axis.text.x=eleme
nt_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Baggage_handling,fill=Satisfaction,y=..count../tapply(..coun
t..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes
(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stac
k', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paire
d")+scale_x_discrete(limits=c("poor","need improvement","acceptable","good","excellent"))+the
me_light()+theme(axis.text.x=element_text(angle = 90),legend.position = 'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



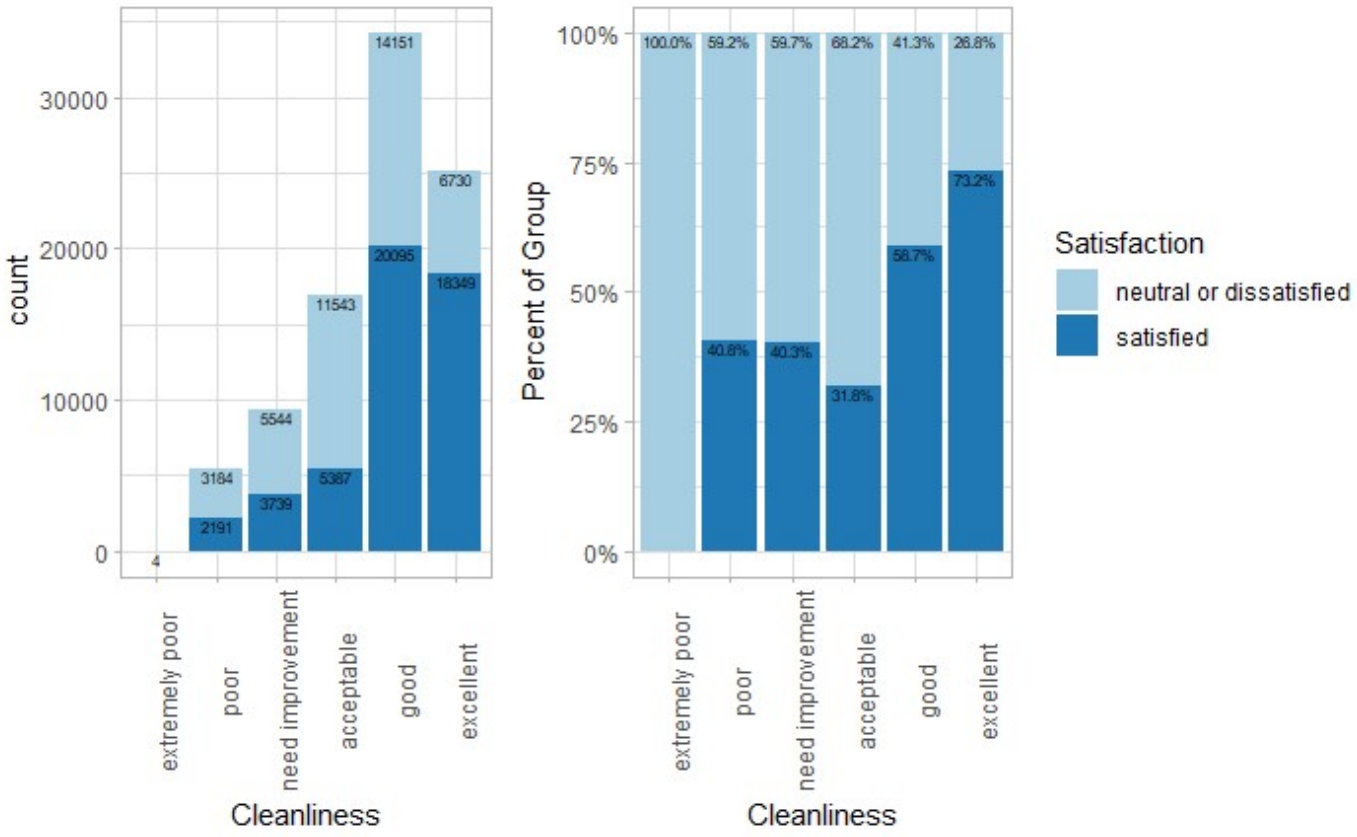
Hide

```
a=ggplot(data = Aviation)+aes(x=Checkin_service,fill=Satisfaction)+geom_bar(position='stac
k')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_fil
l_brewer(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improvemen
t","acceptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(axi
s.text.x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Checkin_service,fill=Satisfaction,y=..count../tapply(..coun
t..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes
(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stac
k', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paire
d")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acceptable","goo
d","excellent"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position =
'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



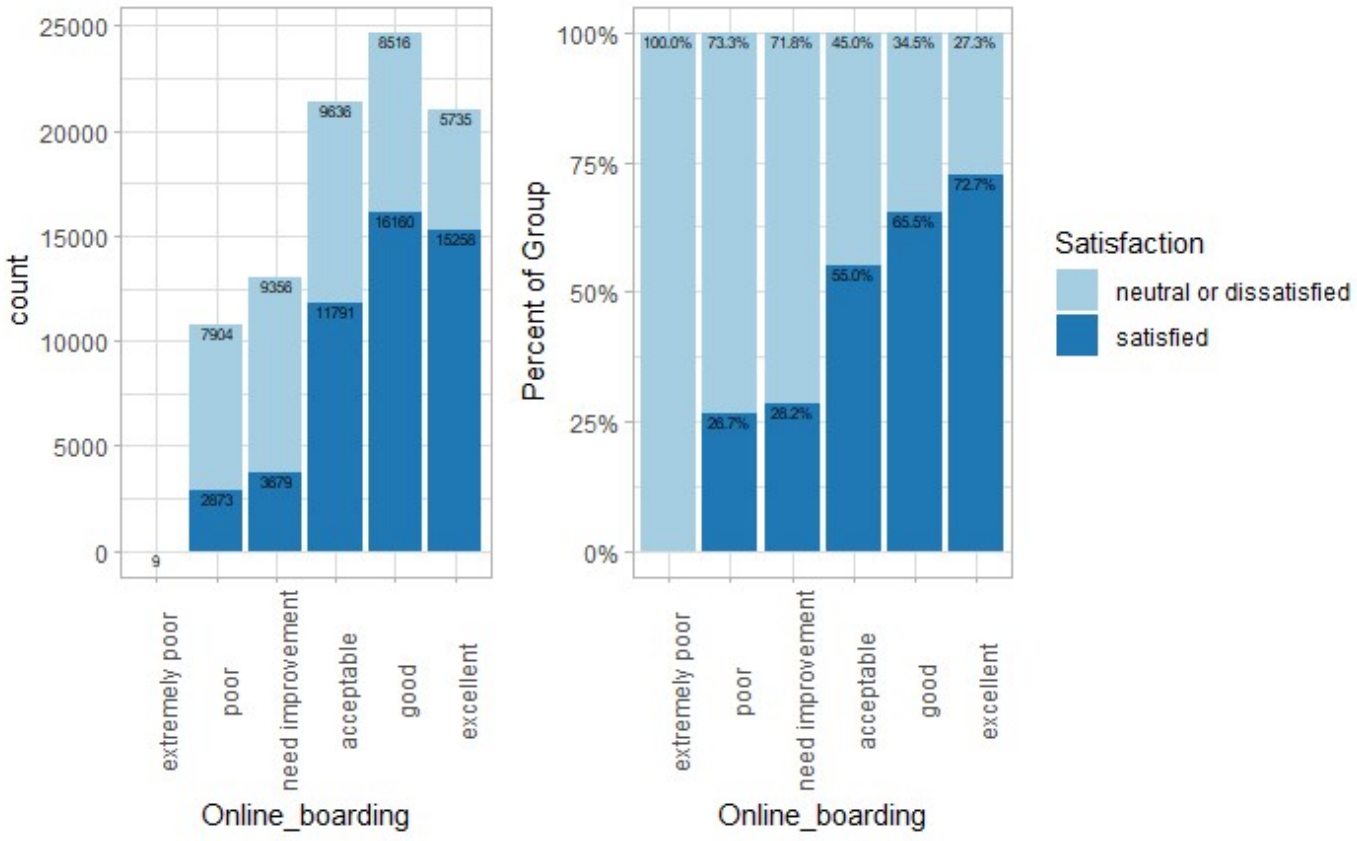
Hide

```
a=ggplot(data = Aviation)+aes(x=Cleanliness,fill=Satisfaction)+geom_bar(position='stack')+geo
m_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_fill_brewe
r(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acc
eptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(axis.text.x
=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Cleanliness,fill=Satisfaction,y=..count../tapply(..count..,..
x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes(label=
percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stack', vju
st=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paired")+scale_x
_discrete(limits=c("extremely poor","poor","need improvement","acceptable","good","excellen
t"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position = 'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



Hide

```
a=ggplot(data = Aviation)+aes(x=Online_boarding,fill=Satisfaction)+geom_bar(position='stac
k')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjust=1,size=2)+scale_fil
l_brewer(palette="Paired")+scale_x_discrete(limits=c("extremely poor","poor","need improvemen
t","acceptable","good","excellent"))+theme_light()+theme(legend.position = 'None')+theme(axi
s.text.x=element_text(angle = 90))

b=ggplot(data = Aviation)+aes(x=Online_boarding,fill=Satisfaction,y=..count../tapply(..coun
t..,..x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes
(label=percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),stat = 'count',position='stac
k', vjust=1,size=2)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paire
d")+scale_x_discrete(limits=c("extremely poor","poor","need improvement","acceptable","goo
d","excellent"))+theme_light()+theme(axis.text.x=element_text(angle = 90),legend.position =
'right')

ggarrange(a,b,ncol = 2,nrow = 1,heights = c(4,4), widths = c(6,10))
```



Relationship between dependent variable and continuous variable through Logistic regression models:

Hide

```
?data.frame
logistic.dataframe = data.frame(Aviation$Satisfaction,Aviation$Age,Aviation$Flight_Distance,A
viation$DepartureDelayin_Mins,Aviation$ArrivalDelayin_Mins)
```

Hide

```
colnames(logistic.dataframe)=c('Satisfaction','Age','Flight_Distance','DepartureDelayin_Min
s','ArrivalDelayin_Mins')
```

Hide

```
logistic.dataframe$Satisfaction=as.factor(logistic.dataframe$Satisfaction)
```

Hide

```
levels(logistic.dataframe$Satisfaction)
```

```
[1] "neutral or dissatisfied" "satisfied"
```

Hide

```
?describe
describe(logistic.dataframe$DepartureDelayin_Mins,quant = c(.10,.25,.50,.75,.90,.95),IQR=T)
```

Hide

```
describe(logistic.dataframe$Flight_Distance,quant = c(.10,.25,.50,.75,.90,.95),IQR=T)
```

Capping outliers:

Hide

```
logistic.dataframe$Flight_Distance=ifelse(logistic.dataframe$Flight_Distance>4315,4315,logist
ic.dataframe$Flight_Distance)
```

Hide

```
logistic.dataframe$DepartureDelayin_Mins=ifelse(logistic.dataframe$DepartureDelayin_Mins>30,3
0,logistic.dataframe$DepartureDelayin_Mins)
logistic.dataframe$ArrivalDelayin_Mins=ifelse(logistic.dataframe$ArrivalDelayin_Mins>30,30,lo
gistic.dataframe$ArrivalDelayin_Mins)
```

Logistic models to check the correlation between continuous independent variable and the dependent variable.

Hide

```
logistic.model.Age=glm(Satisfaction ~ Age,family='binomial',data = Aviation,na.action = na.om
it)
logistic.model.FlightDistance = glm(Satisfaction~Flight_Distance,family = 'binomial',data = A
viation,na.action = na.omit)
logistic.model.DepartureDelayin_Mins = glm(Satisfaction~DepartureDelayin_Mins,family = 'binom
ial',data = Aviation,na.action = na.omit)
logistic.model.ArrivalDelayin_Mins = glm(Satisfaction~ArrivalDelayin_Mins,family = 'binomia
l',data = Aviation,na.action = na.omit)
```

Hide

```
summary(logistic.model.Age)
```

```
Call:
glm(formula = Satisfaction ~ Age, family = "binomial", data = Aviation,
    na.action = na.omit)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5803  -1.2230   0.9735   1.0868   1.3155

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.4292460  0.0187421  -22.90   <2e-16 ***
Age          0.0157623  0.0004474   35.23   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 125222  on 90916  degrees of freedom
Residual deviance: 123961  on 90915  degrees of freedom
AIC: 123965

Number of Fisher Scoring iterations: 4
```

Hide

```
summary(logistic.model.FlightDistance)
```

```
Call:
glm(formula = Satisfaction ~ Flight_Distance, family = "binomial",
    data = Aviation, na.action = na.omit)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.325  -1.257   1.047   1.098   1.262

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      3.448e-01  1.454e-02   23.72   <2e-16 ***
Flight_Distance -7.802e-05  6.495e-06  -12.01   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 125222  on 90916  degrees of freedom
Residual deviance: 125078  on 90915  degrees of freedom
AIC: 125082

Number of Fisher Scoring iterations: 3
```

Hide

```
summary(logistic.model.DepartureDelayin_Mins)
```

```
Call:
glm(formula = Satisfaction ~ DepartureDelayin_Mins, family = "binomial",
    data = Aviation, na.action = na.omit)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.285  -1.284   1.073   1.073   3.217

Coefficients:
                       Estimate Std. Error z value Pr(>|z|)
(Intercept)            0.2502336  0.0072217   34.65   <2e-16 ***
DepartureDelayin_Mins -0.0041538  0.0001943  -21.38   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 125222  on 90916  degrees of freedom
Residual deviance: 124714  on 90915  degrees of freedom
AIC: 124718

Number of Fisher Scoring iterations: 3
```

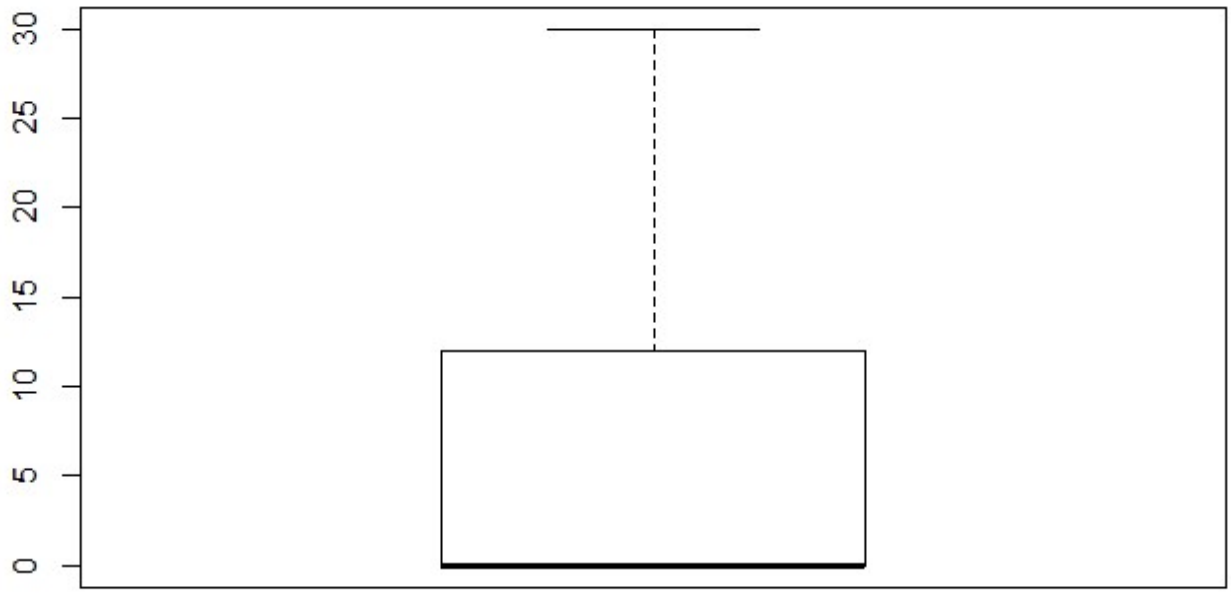
Hide

```
summary(logistic.model.ArrivalDelayin_Mins)
```

```
Call:
glm(formula = Satisfaction ~ ArrivalDelayin_Mins, family = "binomial",
    data = Aviation, na.action = na.omit)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.288  -1.282   1.071   1.071   3.311

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          0.2567529  0.0072574   35.38   <2e-16 ***
ArrivalDelayin_Mins -0.0044789  0.0001948  -22.99   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 124828  on 90632  degrees of freedom
Residual deviance: 124236  on 90631  degrees of freedom
  (284 observations deleted due to missingness)
AIC: 124240

Number of Fisher Scoring iterations: 3
```

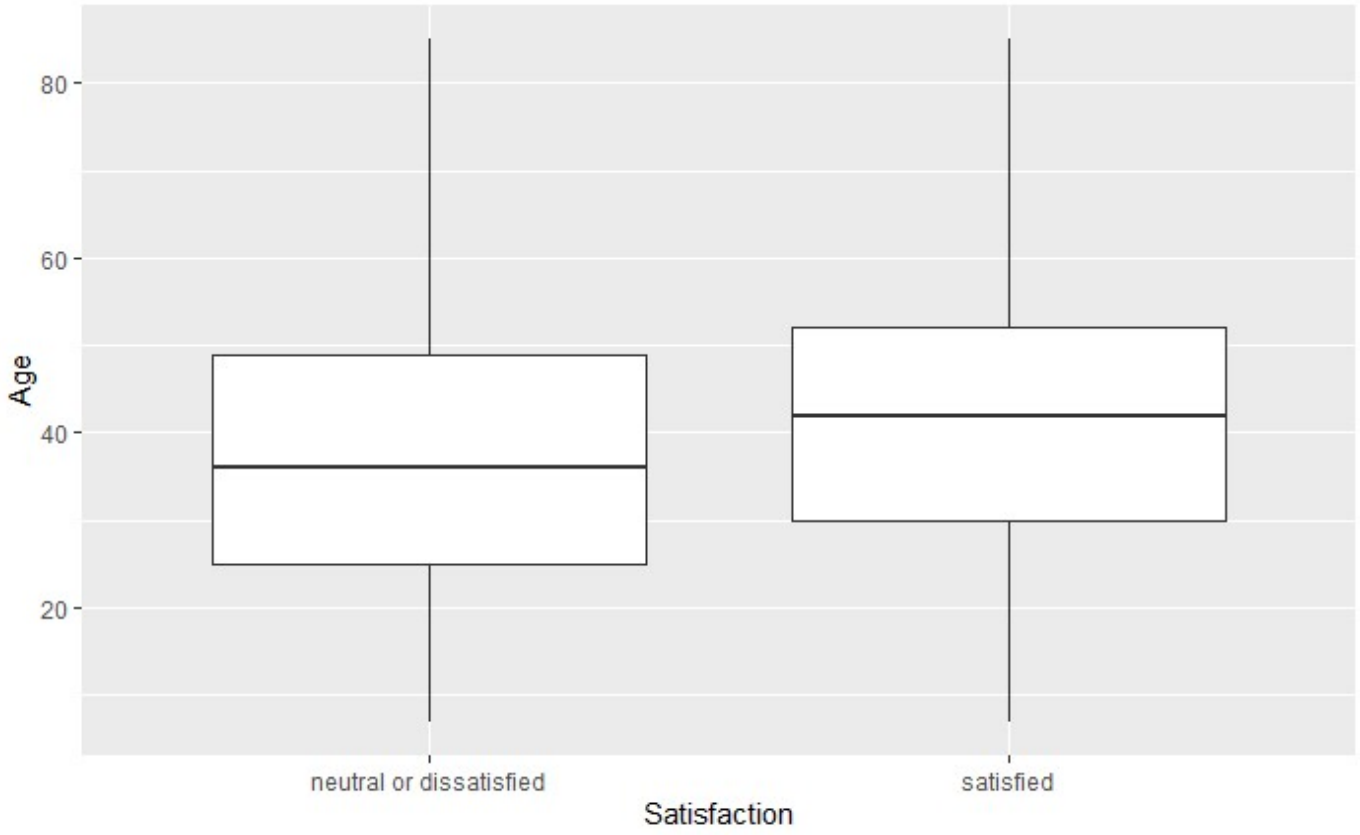
Hide

```
boxplot(logistic.dataframe$DepartureDelayin_Mins)
```



Hide

```
ggplot(data = logistic.dataframe)+aes(x=Satisfaction,y=Age)+geom_boxplot()
```



Hide

```
varImp(logistic.model)
```

Hide

```
pR2(logistic.model.Age)["McFadden"]
```

```
 McFadden
0.0100687
```

Hide

```
pR2(logistic.model.FlightDistance)["McFadden"]
```

```
   McFadden
0.001154021
```

Hide

```
pR2(logistic.model.ArrivalDelayin_Mins)["McFadden"]
```

```
   McFadden
0.007871765
```

Hide

```
pR2(logistic.model.DepartureDelayin_Mins)["McFadden"]
```

```
   McFadden
0.004054502
```

Data Imputation using Mice, all the spaces are replaced by Null in factor variables before applying Mice.

Hide

```
WorkingDF=Aviation
```

Hide

```
WorkingDF$CustomerType=as.character(WorkingDF$CustomerType)
WorkingDF$TypeTravel=as.character(WorkingDF$TypeTravel)
WorkingDF$Departure.Arrival.time_convenient=as.character(WorkingDF$Departure.Arrival.time_con
venient)
WorkingDF$Food_drink=as.character(WorkingDF$Food_drink)
WorkingDF$CustomerType=replace(WorkingDF$CustomerType,WorkingDF$CustomerType=='',NA)
WorkingDF$TypeTravel=replace(WorkingDF$TypeTravel,WorkingDF$TypeTravel=='',NA)
WorkingDF$Departure.Arrival.time_convenient=replace(WorkingDF$Departure.Arrival.time_convenie
nt,WorkingDF$Departure.Arrival.time_convenient=='',NA)
WorkingDF$Food_drink=replace(WorkingDF$Food_drink,WorkingDF$Food_drink=='',NA)
WorkingDF$Onboard_service=replace(WorkingDF$Onboard_service,WorkingDF$Onboard_service=='',NA)
```

Hide

```
WorkingDF$CustomerType=as.factor(WorkingDF$CustomerType)
WorkingDF$TypeTravel=as.factor(WorkingDF$TypeTravel)
WorkingDF$Departure.Arrival.time_convenient=as.factor(WorkingDF$Departure.Arrival.time_conven
ient)
WorkingDF$Food_drink=as.factor(WorkingDF$Food_drink)
WorkingDF$Onboard_service=as.factor(WorkingDF$Onboard_service)
```

Hide

```
summary(Test)
```

Hide

```
imputed.data=mice(WorkingDF[-c(1,9)],m=1,method = 'polyreg',maxit = 25,seed = 500)
```

Hide

```
Imputed.Test=complete(imputed.data,1)
```

Hide

```
imputed.data1=mice(data.frame(WorkingDF[c(9)],Imputed.Test),m=1,method = 'pmm',maxit=25,seed
= 500)
```

Hide

```
Imputed.Test=complete(imputed.data1,1)
```

Hide

```
summary(Imputed.Test)
```

Hide

```
Imputed.Test$Onboard_service=as.character(Imputed.Test$Onboard_service)
Imputed.Test=Imputed.Test[!Imputed.Test$Onboard_service=='',]
Imputed.Test$Onboard_service=as.factor(Imputed.Test$Onboard_service)
#levels(Imputed.Test$Onboard_service)=c("acceptable","excellent","extremely poor","good","nee
d improvement","poor")
```

Hide

```
ggplot(data = Imputed.Test)+aes(x=CustomerType,fill=Satisfaction)+geom_bar(position='stac
k')+labs(x='CustomerType')
```

Hide

```
ggplot(data = Imputed.Test)+aes(x=TypeTravel,fill=Satisfaction)+geom_bar(position='stack')+la
bs(x='TypeTravel')
```

Hide

```
Imputed.Test$Departure.Arrival.time_convenient=ordered(Imputed.Test$Departure.Arrival.time_co
nvenient,levels=c("extremely poor","poor","need improvement","acceptable","good","excellen
t"))
ggplot(data = Imputed.Test)+aes(x=Departure.Arrival.time_convenient,fill=Satisfaction)+geom_b
ar(position='stack')+labs(x='Departure.Arrival.time_convenient')
```

Hide

```
Imputed.Test$Food_drink=ordered(Imputed.Test$Food_drink,levels=c("extremely poor","poor","nee
d improvement","acceptable","good","excellent"))
ggplot(data = Imputed.Test)+aes(x=Food_drink,fill=Satisfaction)+geom_bar(position='stack')+la
bs(x='Food_drink')
```

Hide

```
describe(Imputed.Test$DepartureDelayin_Mins,IQR=T,quant = c(.25,.50,.75,.95,1))
```

Hide

```
table(Imputed.Test[Imputed.Test$DepartureDelayin_Mins>76,]$Satisfaction)
```

Hide

```
Cleansed.Data=data.frame(Aviation[c(1)],Imputed.Test)
```

Hide

```
levels(Cleansed.Data$Gate_location)=c('good','poor','acceptable','need improvement','excellen
t','extremely poor')
levels(Cleansed.Data$Satisfaction)=c('dissatisfied','satisfied')
```

Hide

```
write.csv(Cleansed.Data,'Cleansed_Data.csv',row.names = F)
```

Hide

```
summary(Cleansed.Data)
```

#Binning Age

Hide

```
Cleansed.Data$agebin = cut(Cleansed.Data$Age,breaks = c(0,20,40,60,80,100))
```

Hide

```
Cleansed.Data$Distancebin = cut(Cleansed.Data$Flight_Distance,breaks = c(0,1500,3000,4500,695
0),labels = c('Short','Medium','Long','Very Long'))
```

Hide

```
table(Cleansed.Data$Satisfaction,Cleansed.Data$Distancebin)
```

```
              Short Medium  Long Very Long
  dissatisfied  9908  26284  4478       486
  satisfied    17090  23743  7969       959
```

Hide

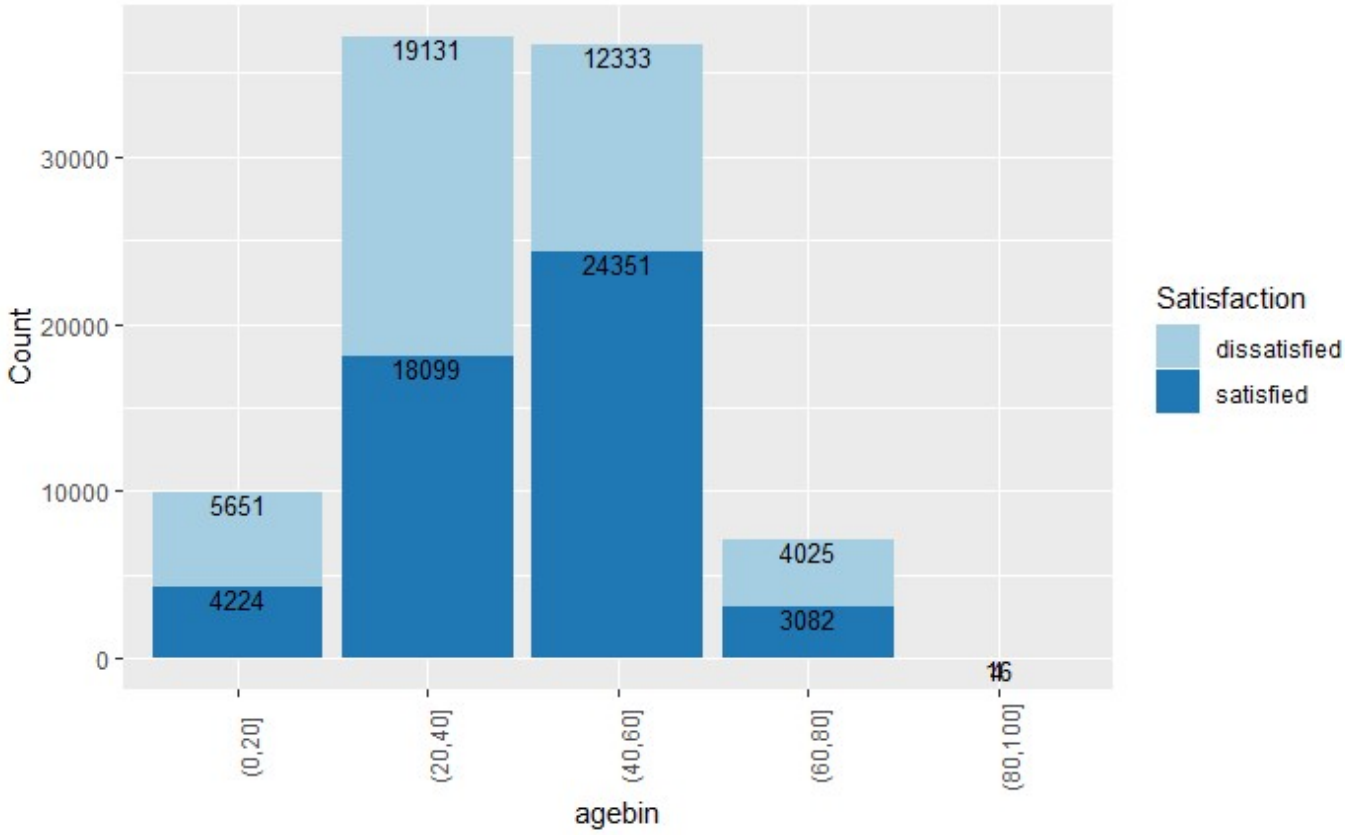
```
levels(Cleansed.Data$agebin)
```

Hide

```
table(Cleansed.Data$agebin)
```

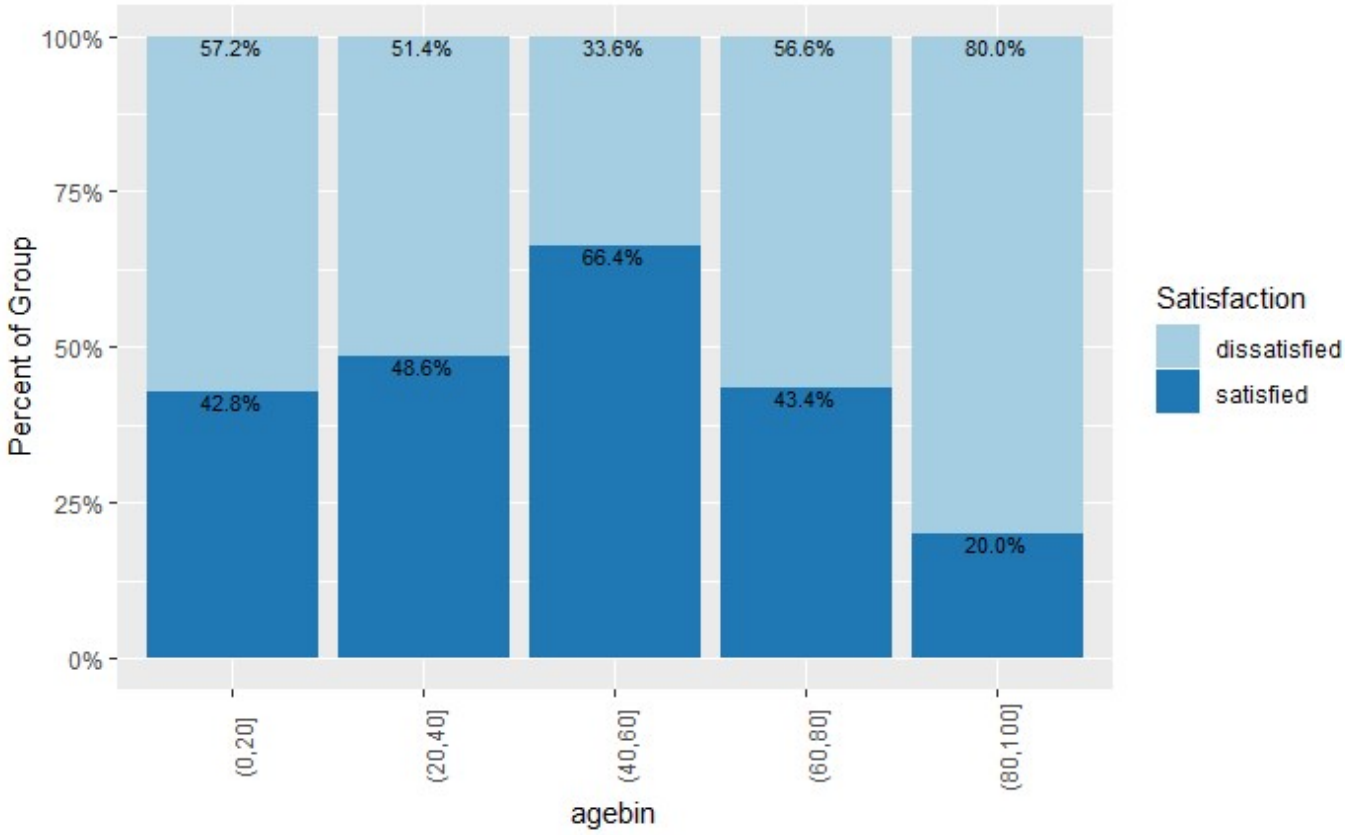
Hide

```
ggplot(data = Cleansed.Data)+aes(x=agebin,fill=Satisfaction,y=..count..)+geom_bar(position =
'stack')+labs(y='Count')+geom_text(aes(label=..count..),stat = 'count',position='stack', vjus
t=1,size=3.5)+scale_fill_brewer(palette="Paired")+theme(axis.text.x=element_text(angle = 90))
```



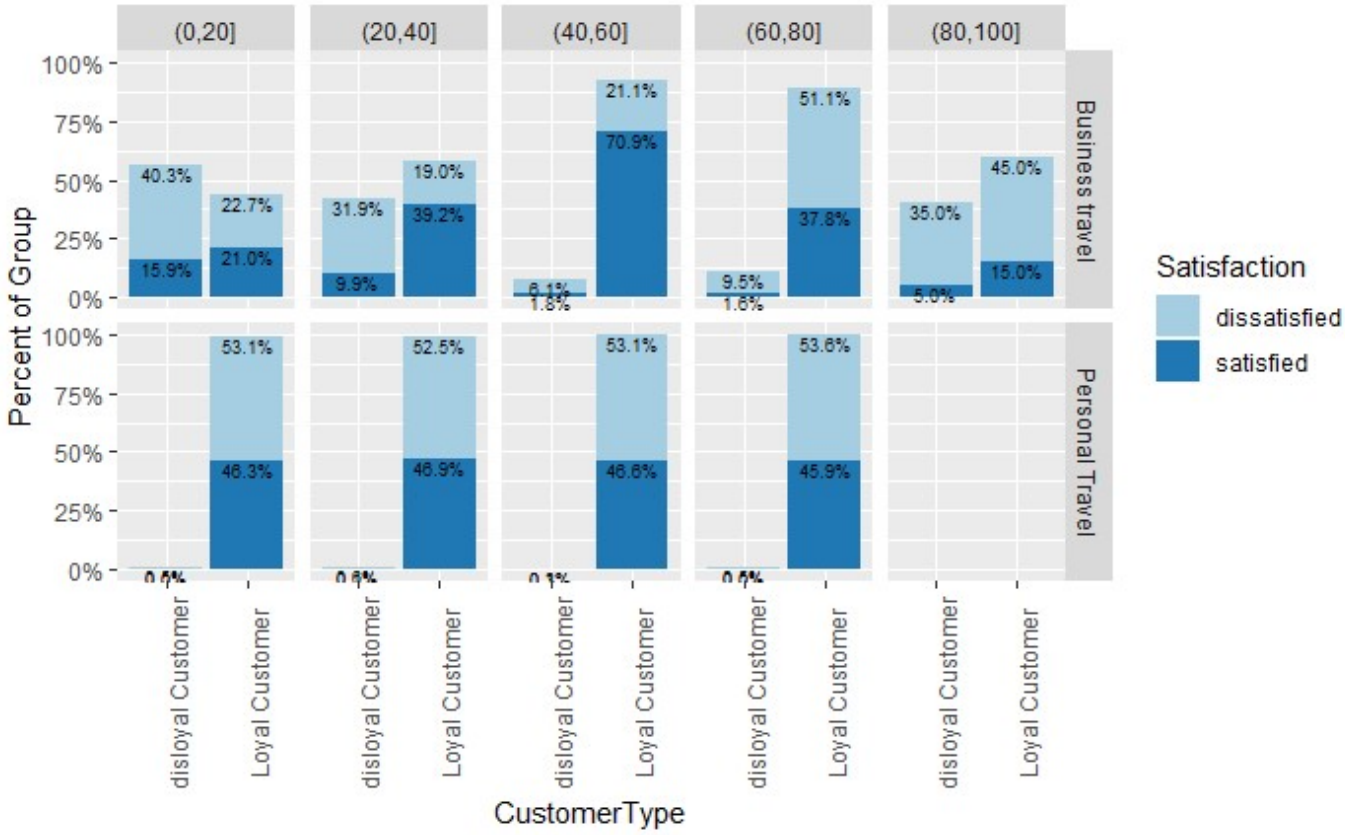
Hide

```
ggplot(data = Cleansed.Data)+aes(x=agebin,fill=Satisfaction,y=..count../tapply(..count..,..
x..,sum)[..x..])+geom_bar(position = 'stack')+labs(y='Percent of Group')+geom_text(aes(label=
percent(..count../tapply(..count..,..x..,sum)[..x..])),stat = 'count',position='stack', vjust
=1,size=3)+scale_y_continuous(labels = percent)+scale_fill_brewer(palette="Paired")+theme(axi
s.text.x=element_text(angle = 90))
```



Hide

```
ggplot(data = Cleansed.Data)+aes(x=CustomerType,fill=Satisfaction,y=..count../tapply(..coun
t..,..PANEL..,sum)[..PANEL..])+geom_bar(position = 'stack')+facet_grid(TypeTravel~agebin)+lab
s(y='Percent of Group')+geom_text(aes(label=percent(..count../tapply(..count..,..PANEL..,sum)
[..PANEL..])),stat = 'count',position='stack', vjust=1,size=2.5)+scale_y_continuous(labels =
percent)+scale_fill_brewer(palette="Paired")+theme(axis.text.x=element_text(angle = 90))
```



#Preparing Data for Clustering/Customer Segmentation

Clustering was done on passenger attributes like Age, Gender, Customer Type, Travel Type and Class. Segmentation might help us understand some pattern of the type of passengers in the survey. Using the segmentation data we can further drill down on how each segment of passenger has rated the amenities and the overall satisfaction levels. Since the data has both numeric and categorical features, gower distance was used to create distance matrix and then clustering.

PAM is a very resource intensive method, hence only 10% of the data is taken for Clustering

Hide

```
Clustidx = sample(nrow(Cleansed.Data),.1*nrow(Cleansed.Data),replace = F)
ClustData = Cleansed.Data[Clustidx,]
```

Hide

```
summary(ClustData)
```

Hide

```
Market.SegmentDF = data.frame(ClustData$Gender,ClustData$CustomerType,ClustData$Age,ClustData
$TypeTravel,ClustData$Class)
colnames(Market.SegmentDF)=c('Gender','CustomerType','Age','TypeTravel','Class')
```

Hide

```
str(Market.SegmentDF)
```

Hide

```
Market.SegmentDF.Numeric = data.matrix(Market.SegmentDF)
```

Hide

```
gc()
gower_dist<-daisy(Market.SegmentDF,metric = "gower")
```

Hide

```
sil_width <- c(NA)

for(i in 2:10){

  pam_fit <- pam(gower_dist,
                 diss = TRUE,
                 k = i)

  sil_width[i] <- pam_fit$silinfo$avg.width

}
```

Hide

```
plot(1:10, sil_width,
     xlab = "Number of clusters",
     ylab = "Silhouette Width")
lines(1:10, sil_width)
```

#Calculate the optimum number of clusters to be created , using the elbow method

Hide

```
fviz_nbclust(Market.SegmentDF.Numeric,pam,method = "wss",diss=gower_dist)+geom_vline(xinterce
pt = 4, linetype = 2)+labs(subtitle = "Elbow method")
```

Hide

```
?fviz_nbclust
fviz_nbclust(Market.SegmentDF.Numeric,pam,method = "silhouette",diss=gower_dist)
```

#4 is the optimumm number of clusters suggested as per this method

Hide

```
pam_fit_4 <- pam(gower_dist, diss = TRUE, k = 4)
```

profiling the clusters

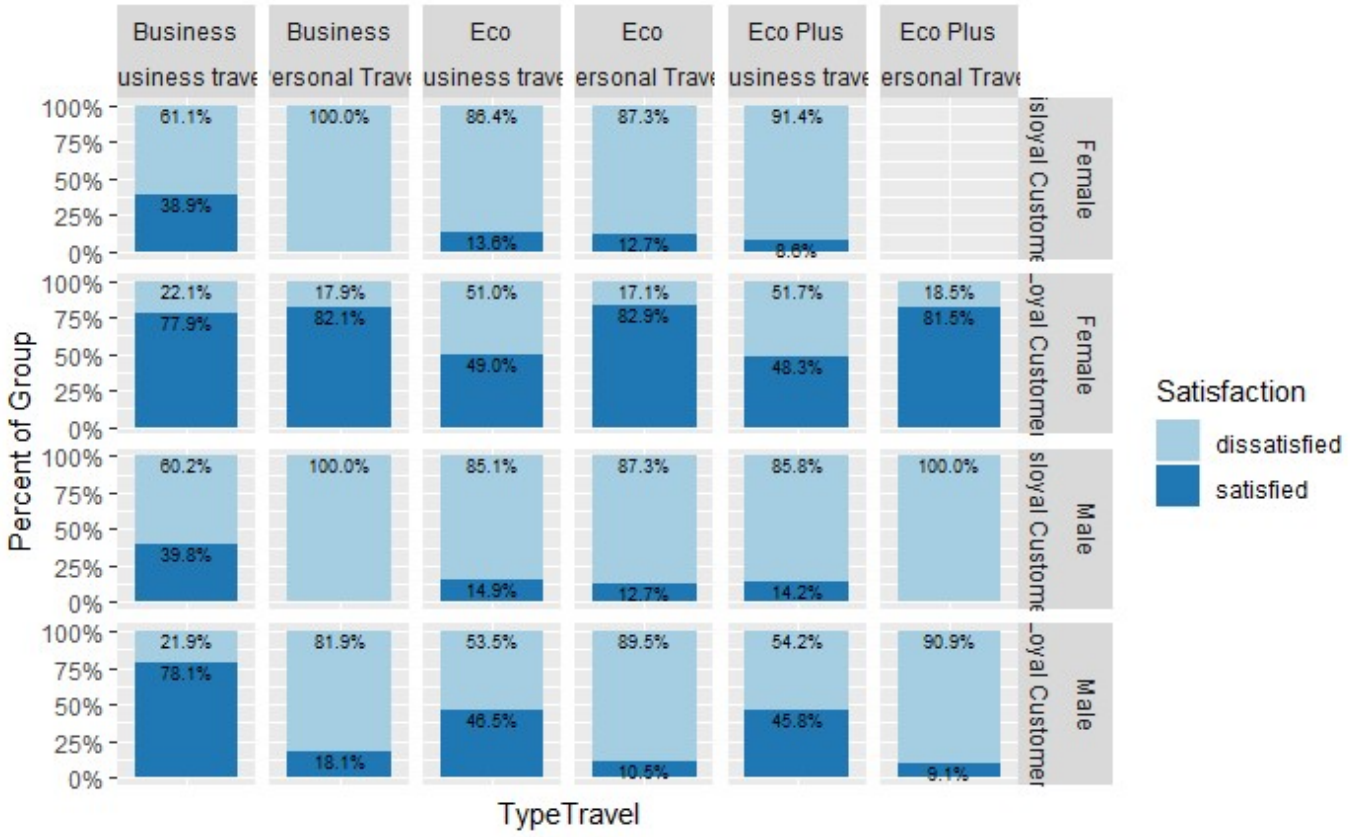
Hide

```
pam_results_4 <- Market.SegmentDF %>%
  mutate(cluster_id = pam_fit_4$clustering) %>%
  group_by(cluster_id) %>%
  do(overall_summary = summary(.))

pam_results_4$overall_summary
```

Description of the above Clusters: Cluster 1: Male, Loyal customers flying Business class for Business trips. Cluster 2: Loyal customers flying Economy class for Personal trips. Cluster 3: Female, Loyal customers flying Business class for Business trips. Cluster 4: Disloyal customers flying Economy class for Business trips.

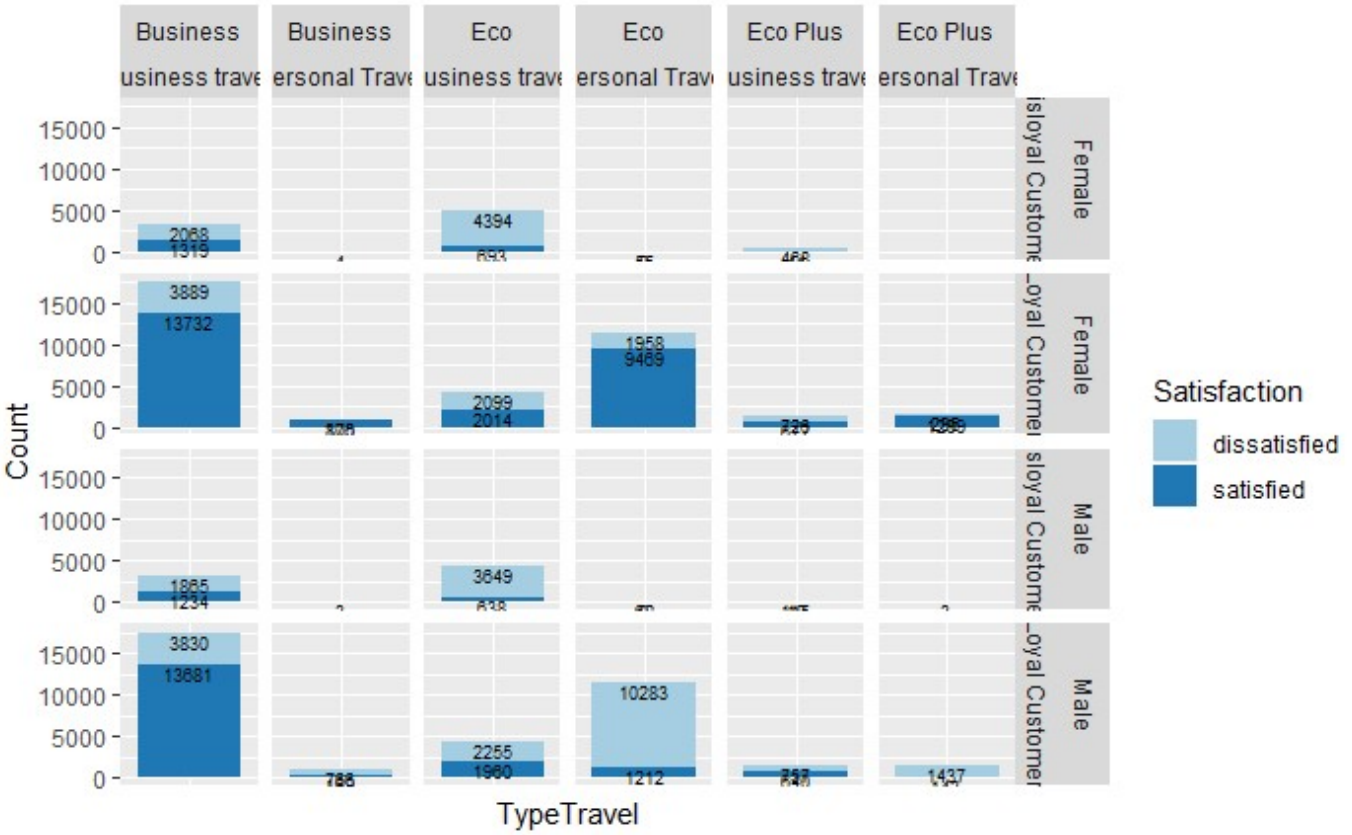
Hide

```
ggplot(data = Cleansed.Data)+aes(x=TypeTravel,fill=Satisfaction,y=..count../tapply(..coun
t..,..PANEL..,sum)[..PANEL..])+geom_bar(position = 'stack')+facet_grid(rows=vars(Gender,Custo
merType),cols=vars(Class,TypeTravel),space = 'free',scales = 'free_x')+labs(y='Percent of Gro
up')+geom_text(aes(label=percent(..count../tapply(..count..,..PANEL..,sum)[..PANEL..])),stat
= 'count',position='stack', vjust=1,size=2.5)+scale_y_continuous(labels = percent)+scale_fill
_brewer(palette="Paired")+theme(axis.ticks.x = element_blank(),axis.text.x = element_blank())
```



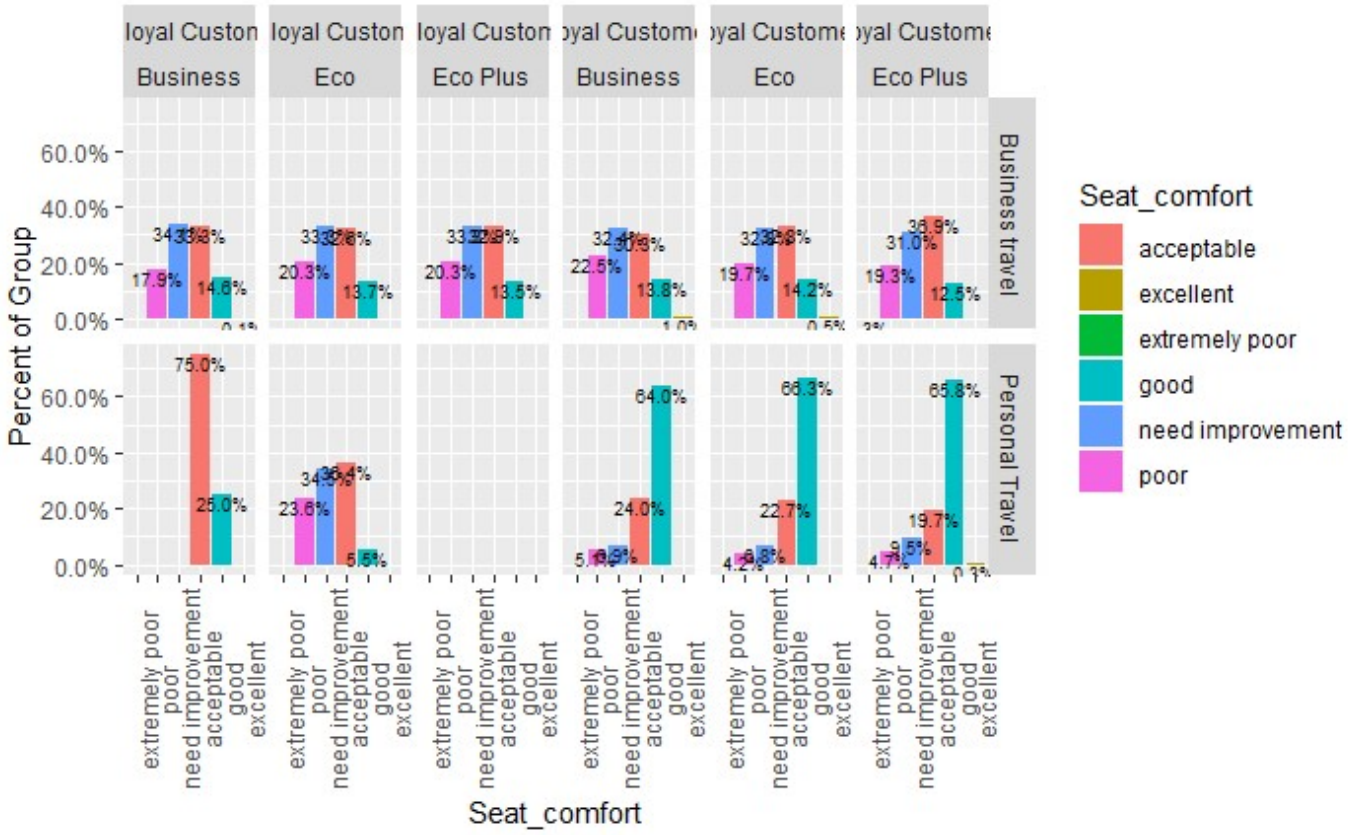
Hide

```
ggplot(data = Cleansed.Data)+aes(x=TypeTravel,fill=Satisfaction,y=..count..)+geom_bar(positio
n = 'stack')+facet_grid(rows=vars(Gender,CustomerType),cols=vars(Class,TypeTravel),space = 'f
ree',scales = 'free_x')+labs(y='Count')+geom_text(aes(label=..count..),stat = 'count',positio
n='stack', vjust=1,size=2.5)+scale_fill_brewer(palette="Paired")+theme(axis.ticks.x = element
_blank(),axis.text.x = element_blank())
```



Hide

```
ggplot(data = Cleansed.Data[Cleansed.Data$Gender=='Female' & Cleansed.Data$Satisfaction=='dis
satisfied',])+aes(x=Seat_comfort, y=..count../tapply(..count..,..PANEL..,sum)[..PANEL..],fill
=Seat_comfort)+geom_bar(position = 'stack')+facet_grid(cols = vars(CustomerType,Class),rows =
vars(TypeTravel))+theme(axis.text.x=element_text(angle = 90))+scale_x_discrete(limits=c("extr
emely poor","poor","need improvement","acceptable","good","excellent"))+geom_text(aes(label=p
ercent(..count../tapply(..count..,..PANEL..,sum)[..PANEL..])),stat = 'count',position='stac
k', vjust=1,size=2.5)+scale_y_continuous(labels = percent)+labs(y='Percent of Group')
```



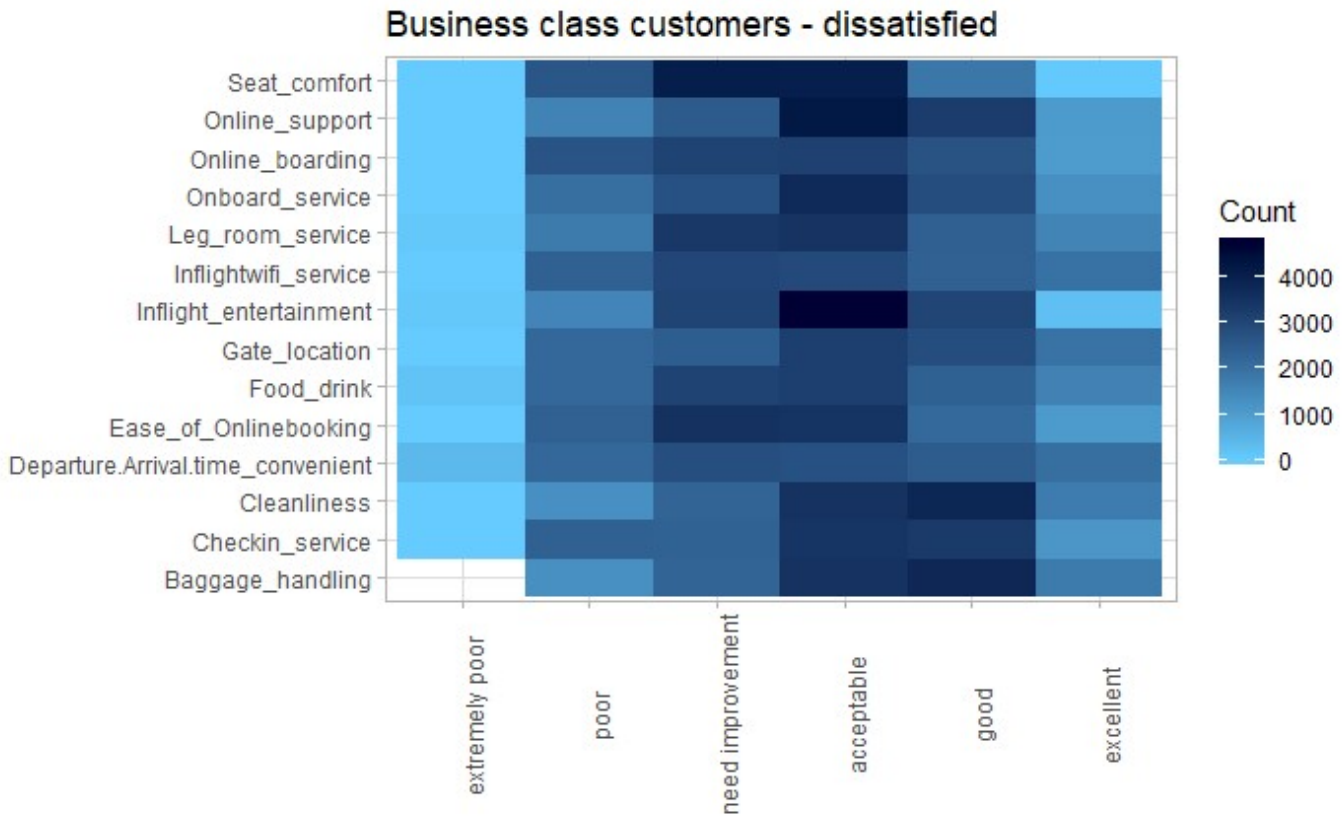
In an effort to understand the trend of ratings received across all flight features by various segment of customers some heatmaps were plotted.

Hide

```
function_heatmap = function(myData) {

factorlist = c('Seat_comfort','Departure.Arrival.time_convenient','Food_drink','Gate_locatio
n','Inflightwifi_service','Inflight_entertainment','Online_support','Ease_of_Onlinebookin
g','Onboard_service','Leg_room_service','Baggage_handling','Checkin_service','Cleanliness','O
nline_boarding')

heatmapDf=data.frame(factor=character(),Ratings=character(),Count=numeric())
colorder=c('factor','Ratings','Count')

 for (i in factorlist) {
    tempdf = data.frame(table(myData[i]))
    colnames(tempdf)=c('Ratings','Count')
    tempdf$factor=i
    tempdf=tempdf[,colorder]
    heatmapDf=rbind(heatmapDf,tempdf)
    rm(tempdf)
 }

a = ggplot(data = heatmapDf)+aes(x=Ratings,y=factor,fill=Count)+geom_tile()+scale_fill_gradie
nt(low='#66CCFF',high = '#000033')+scale_x_discrete(limits=c("extremely poor","poor","need im
provement","acceptable","good","excellent"))+theme_light()+theme(axis.text.x=element_text(ang
le = 90))
return(a)

}
```

Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$Class=='Business' & Cleansed.Data$Satisfaction
=='dissatisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Business class customers - dissatisfied')
```

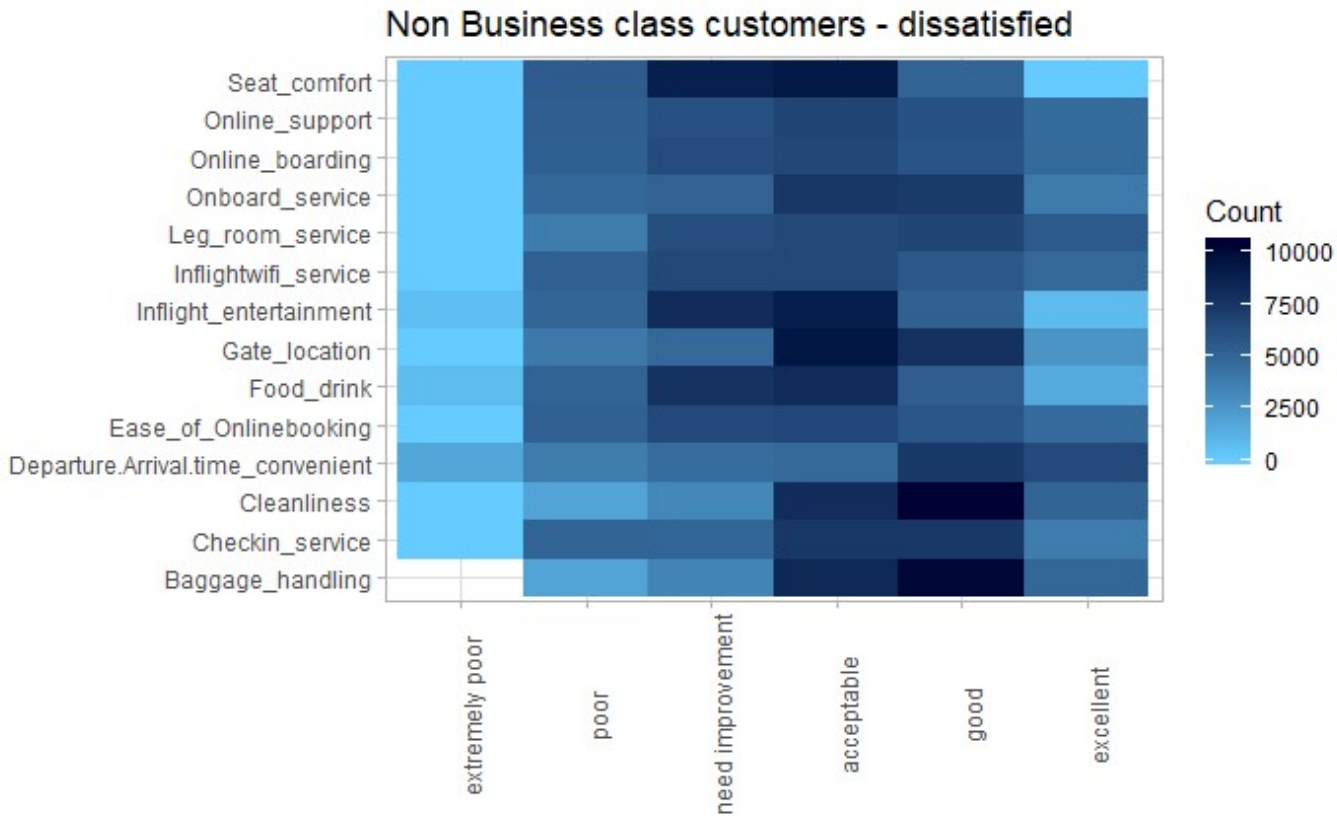


Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$Class=='Business' & Cleansed.Data$Satisfaction
=='satisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Business class customers - satisfied')
```

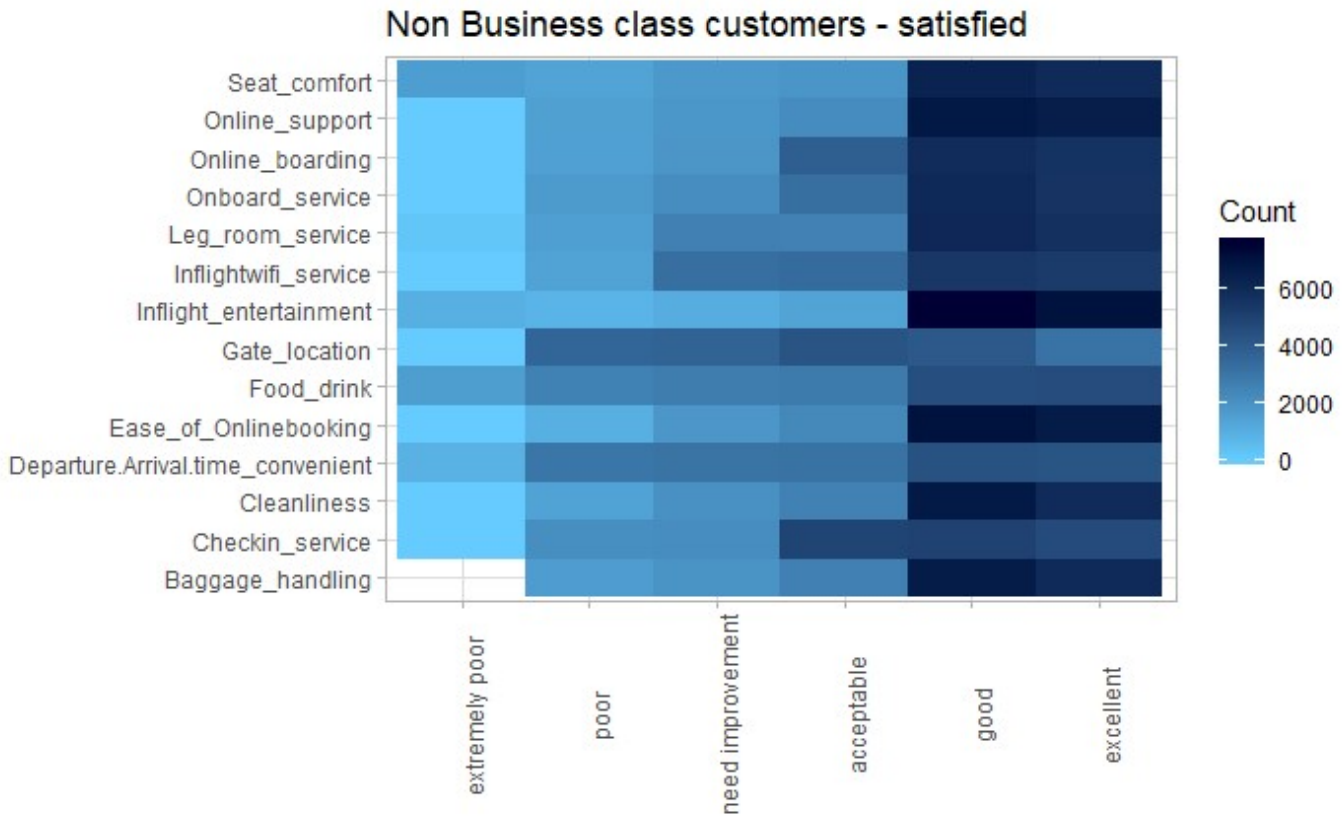
Hide

```
a=function_heatmap(Cleansed.Data[!Cleansed.Data$Class=='Business' & Cleansed.Data$Satisfactio
n=='dissatisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Non Business class customers - dissatisfi
ed')
```



Hide

```
a=function_heatmap(Cleansed.Data[!Cleansed.Data$Class=='Business' & Cleansed.Data$Satisfactio
n=='satisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Non Business class customers - satisfie
d')
```



An alternate way of analyzing the features and their impact on satisfaction is Factor analysis. We can reduce the number of features or condense them into categories to help Falcon airlines assess them as a whole. The feature survey data is ordinal data and are factor variables. Usual Correlation functions work best on continuous or numeric variables. Here we will apply Polychoric correlation as the data is ordinal.

Creating a Test dataframe with only the factors, converting them to numeric values

Hide

```
Test=data.frame(data.matrix(Cleansed.Data[c(11:24)]))
```

Hide

```
str(Test)
```

```
'data.frame':   90917 obs. of  14 variables:
 $ Seat_comfort                     : int  3 3 3 3 3 3 3 3 3 3 ...
 $ Departure.Arrival.time_convenient: int  3 3 2 3 3 3 3 2 3 3 ...
 $ Food_drink                       : int  3 3 3 3 3 3 3 3 3 3 ...
 $ Gate_location                    : int  5 1 1 1 1 1 1 1 1 4 ...
 $ Inflightwifi_service             : int  5 5 1 4 5 5 5 5 1 5 ...
 $ Inflight_entertainment           : int  4 3 4 1 3 2 3 3 1 3 ...
 $ Online_support                   : int  5 5 1 4 5 2 5 5 1 5 ...
 $ Ease_of_Onlinebooking            : int  1 5 6 5 5 2 5 5 1 5 ...
 $ Onboard_service                  : int  1 2 6 5 2 2 1 5 1 1 ...
 $ Leg_room_service                 : int  3 1 3 3 4 3 1 4 3 5 ...
 $ Baggage_handling                 : int  1 3 5 4 2 2 3 2 5 2 ...
 $ Checkin_service                  : int  2 4 4 4 2 2 2 1 5 5 ...
 $ Cleanliness                      : int  1 4 6 5 4 2 4 4 1 2 ...
 $ Online_boarding                  : int  5 5 1 2 5 1 5 5 2 5 ...
```

Hide

```
Test$Seat_comfort=as.factor(Test$Seat_comfort)
Test$Departure.Arrival.time_convenient=as.factor(Test$Departure.Arrival.time_convenient)
Test$Food_drink=as.factor(Test$Food_drink)
Test$Gate_location=as.factor(Test$Gate_location)
Test$Inflightwifi_service=as.factor(Test$Inflightwifi_service)
Test$Inflight_entertainment=as.factor(Test$Inflight_entertainment)
Test$Online_support=as.factor(Test$Online_support)
Test$Ease_of_Onlinebooking=as.factor(Test$Ease_of_Onlinebooking)
Test$Onboard_service=as.factor(Test$Onboard_service)
Test$Leg_room_service=as.factor(Test$Leg_room_service)
Test$Baggage_handling=as.factor(Test$Baggage_handling)
Test$Checkin_service=as.factor(Test$Checkin_service)
Test$Cleanliness=as.factor(Test$Cleanliness)
Test$Online_boarding=as.factor(Test$Online_boarding)
str(Test)
```

```
'data.frame':   90917 obs. of  14 variables:
 $ Seat_comfort                     : Factor w/ 6 levels "1","2","3","4",..: 3 3 3 3 3 3 3 3
3 3 ...
 $ Departure.Arrival.time_convenient: Factor w/ 6 levels "1","2","3","4",..: 3 3 2 3 3 3 3 2
3 3 ...
 $ Food_drink                       : Factor w/ 6 levels "1","2","3","4",..: 3 3 3 3 3 3 3 3
3 3 ...
 $ Gate_location                    : Factor w/ 6 levels "1","2","3","4",..: 5 1 1 1 1 1 1 1
1 4 ...
 $ Inflightwifi_service             : Factor w/ 6 levels "1","2","3","4",..: 5 5 1 4 5 5 5 5
1 5 ...
 $ Inflight_entertainment           : Factor w/ 6 levels "1","2","3","4",..: 4 3 4 1 3 2 3 3
1 3 ...
 $ Online_support                   : Factor w/ 6 levels "1","2","3","4",..: 5 5 1 4 5 2 5 5
1 5 ...
 $ Ease_of_Onlinebooking            : Factor w/ 6 levels "1","2","3","4",..: 1 5 6 5 5 2 5 5
1 5 ...
 $ Onboard_service                  : Factor w/ 6 levels "1","2","3","4",..: 1 2 6 5 2 2 1 5
1 1 ...
 $ Leg_room_service                 : Factor w/ 6 levels "1","2","3","4",..: 3 1 3 3 4 3 1 4
3 5 ...
 $ Baggage_handling                 : Factor w/ 5 levels "1","2","3","4",..: 1 3 5 4 2 2 3 2
5 2 ...
 $ Checkin_service                  : Factor w/ 6 levels "1","2","3","4",..: 2 4 4 4 2 2 2 1
5 5 ...
 $ Cleanliness                      : Factor w/ 6 levels "1","2","3","4",..: 1 4 6 5 4 2 4 4
1 2 ...
 $ Online_boarding                  : Factor w/ 6 levels "1","2","3","4",..: 5 5 1 2 5 1 5 5
2 5 ...
```

Hide

```
poly_values=polychoric(Test)
```

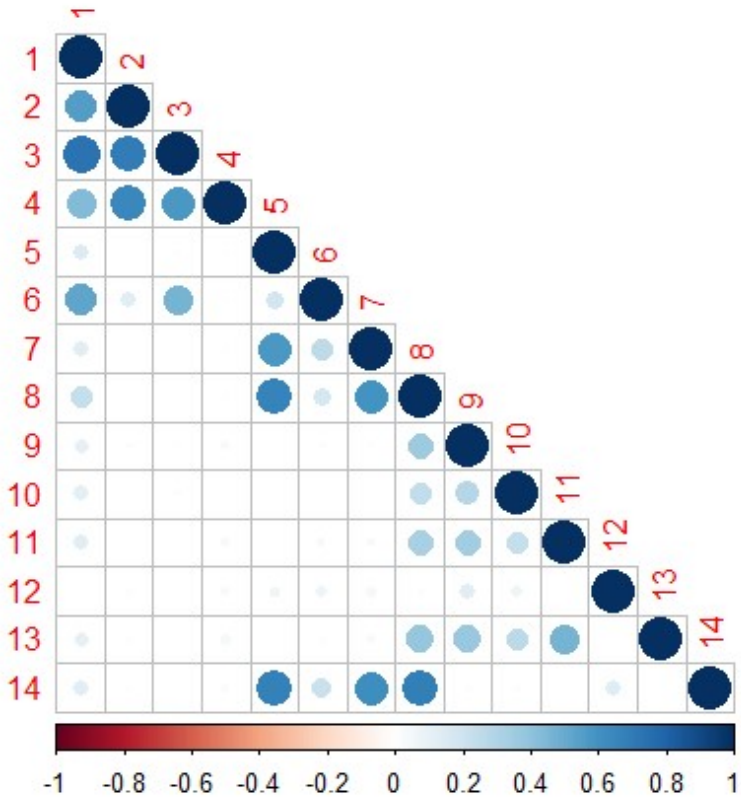
```
Converted non-numeric input to numeric
80 cells were adjusted for 0 values using the correction for continuity. Examine your data ca
refully.
```

Hide

```
items_polychoric = poly_values$rho
#items_polychoric
#poly_values1$rho
```

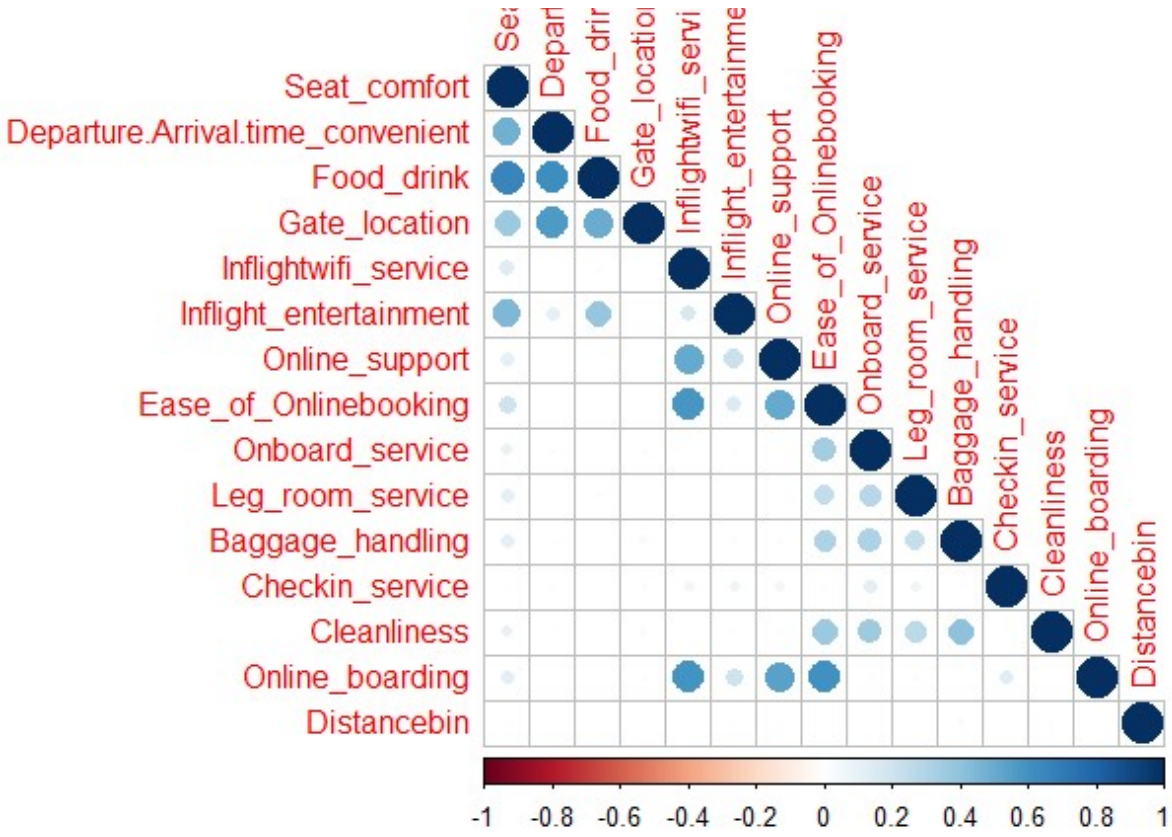
Hide

```
corrplot(items_polychoric,type='lower')
```



Hide

```
#corrplot(poly_values1$rho,type='lower')
corrplot(cor(data.matrix(Cleansed.Data[,c(11:24,27)])),type='lower')
```



Polychoric correlation of factor variables below shows some collinearity in data. Sphericity check and KMO test indicates that dimension reduction is possible for the data.

Hide

```
KMO(items_polychoric)
```

```
Kaiser-Meyer-Olkin factor adequacy
Call: KMO(r = items_polychoric)
Overall MSA =  0.73
MSA for each item =
 [1] 0.80 0.77 0.74 0.74 0.80 0.60 0.83 0.65 0.67 0.80 0.76 0.30 0.66 0.78
```

Hide

```
cortest.bartlett(items_polychoric)
```

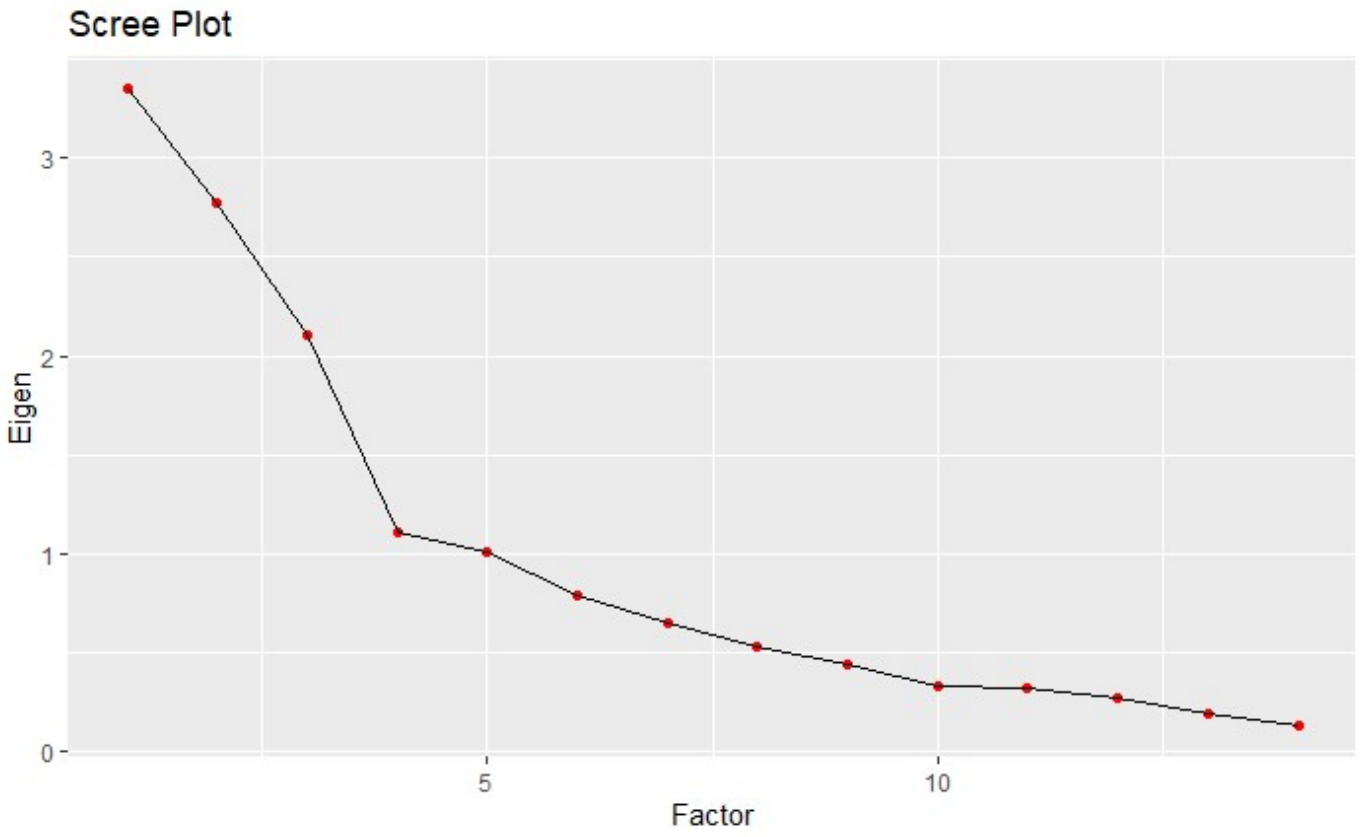
```
n not specified, 100 used
```

```
$chisq
[1] 583.9409

$p.value
[1] 5.344682e-73

$df
[1] 91
```

Scree plot shows that 4 factors could be derived from the data

Hide

```
ev=eigen(items_polychoric)
Eigen = ev$values
Factor = c(1:length(Test))
Scree = data.frame(Factor,Eigen)
ggplot(data = Scree)+aes(x=Factor,y=Eigen)+geom_point(col='red')+geom_line()+labs(title='Scre
e Plot')
```



Hide

```
Unrotate = fa(data.matrix(Cleansed.Data[,c(11:24)]),nfactors = 4,rotate = 'none')
print(Unrotate,digits = 5)
#biplot(Unrotate,row.names(Unrotate$loadings))
Unrotate$loadings
fa.diagram(Unrotate)
```

We will try use Varimax rotation to improve the loadings.

Hide

```
VariRotate = fa(data.matrix(Cleansed.Data[,c(11:24)]),nfactors = 4,rotate = 'varimax')
print(VariRotate,digits = 5)
```

```
Factor Analysis using method =  minres
Call: fa(r = data.matrix(Cleansed.Data[, c(11:24)]), nfactors = 4,
    rotate = "varimax")
Standardized loadings (pattern matrix) based upon correlation matrix
```

	MR1 <S3: Asis>	MR2 <S3: Asis>	MR3 <S3: Asis>	MR4 <S3: Asis>	h2 <dbl>
Seat_comfort	0.11136	0.59972	0.15252	0.47653	0.6224075
Departure.Arrival.time_convenient	-0.01270	0.78447	-0.01048	0.02607	0.6163408

	MR1 <S3: AsIs>	MR2 <S3: AsIs>	MR3 <S3: AsIs>	MR4 <S3: AsIs>	h2 <dbl>
Food_drink	-0.03487	0.77093	-0.00769	0.40332	0.75827222
Gate_location	-0.01430	0.71963	0.00146	-0.13401	0.53603627
Inflightwifi_service	0.74555	0.01962	-0.02232	0.05724	0.55999799
Inflight_entertainment	0.17971	0.12029	-0.03173	0.75237	0.61384010
Online_support	0.67516	-0.00374	-0.05176	0.10573	0.46970790
Ease_of_Onlinebooking	0.78067	0.01654	0.53034	0.04561	0.89306385
Onboard_service	0.03656	0.00051	0.57033	0.04253	0.32841957
Leg_room_service	0.02182	0.01434	0.43097	0.03588	0.18770495

1-10 of 14 rows Previous **1** 2 Next

```
                         MR1     MR2     MR3     MR4
SS loadings           2.28886 2.10323 1.58076 1.00519
Proportion Var        0.16349 0.15023 0.11291 0.07180
Cumulative Var        0.16349 0.31372 0.42663 0.49843
Proportion Explained  0.32801 0.30141 0.22653 0.14405
Cumulative Proportion 0.32801 0.62942 0.85595 1.00000

Mean item complexity =  1.3
Test of the hypothesis that 4 factors are sufficient.

The degrees of freedom for the null model are  91  and the objective function was  4.49198 wi
th Chi Square of  408368.2
The degrees of freedom for the model are 41  and the objective function was  0.08078

The root mean square of the residuals (RMSR) is  0.01724
The df corrected root mean square of the residuals is  0.02569

The harmonic number of observations is  90917 with the empirical chi square  4920.855  with p
rob <  0
The total number of observations was  90917  with Likelihood Chi Square =  7343.168  with pro
b <  0

Tucker Lewis Index of factoring reliability =  0.960302
RMSEA index =  0.044262  and the 90 % confidence intervals are  0.04341 0.045116
BIC =  6875.042
Fit based upon off diagonal values = 0.99486
Measures of factor score adequacy
                                                    MR1     MR2     MR3     MR4
Correlation of (regression) scores with factors   0.93287 0.91224 0.87434 0.82316
Multiple R square of scores with factors          0.87025 0.83218 0.76447 0.67759
Minimum correlation of possible factor scores     0.74049 0.66437 0.52894 0.35518
```

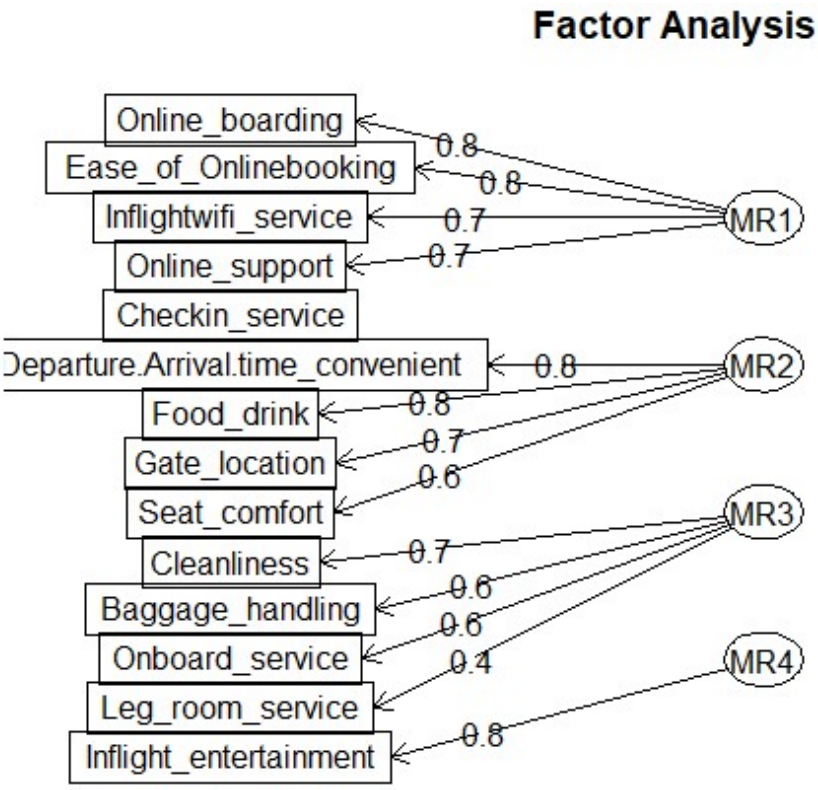
Hide

```
VariRotate$loadings
```

```
Loadings:
                                 MR1    MR2    MR3    MR4
Seat_comfort                      0.111  0.600  0.153  0.477
Departure.Arrival.time_convenient        0.784
Food_drink                               0.771         0.403
Gate_location                            0.720        -0.134
Inflightwifi_service              0.746
Inflight_entertainment            0.180  0.120         0.752
Online_support                    0.675                0.106
Ease_of_Onlinebooking             0.781         0.530
Onboard_service                                 0.570
Leg_room_service                                0.431
Baggage_handling                                0.576
Checkin_service
Cleanliness                                     0.652
Online_boarding                   0.784

                 MR1   MR2   MR3   MR4
SS loadings    2.289 2.103 1.581 1.005
Proportion Var 0.163 0.150 0.113 0.072
Cumulative Var 0.163 0.314 0.427 0.498
```

Hide

```
fa.diagram(VariRotate)
```



Factors derived through varimax rotation: MR1: Passenger digital amenities MR2: Passenger convenience MR3: Passenger onboard services MR4: Inflight entertainment

An alternate approach could have been to assess the factors identified from Factor Analysis and build machine learning models using it; but I will focus on using the data as given so that we can identify features impacting Satisfaction at the most granular level.

#Plot the clusters

Hide

```
tsne_obj_4 <- Rtsne(gower_dist, is_distance = TRUE)
tsne_data_4 <- tsne_obj_4$Y %>%
  data.frame() %>%
  setNames(c("X", "Y")) %>%
  mutate(cluster = factor(pam_fit_4$clustering))
ggplot(aes(x = X, y = Y), data = tsne_data_4) +
  geom_point(aes(color = cluster))
```

We will assign cluster id to all the observations of the data set before splitting into training and test datasets.

Hide

```
library(dplyr)
Cleansed.Data=
Cleansed.Data %>%
  mutate(clusterid = case_when(
    Gender=='Male' & CustomerType=='Loyal Customer' & TypeTravel=='Business travel' & Class=
='Business' ~ '1',
    Gender=='Female' & CustomerType=='Loyal Customer' & TypeTravel=='Business travel' & Class
=='Business' ~ '3',
    CustomerType=='Loyal Customer' & TypeTravel=='Personal Travel' & Class!='Business' ~ '2',
    CustomerType=='disloyal Customer' & TypeTravel=='Business travel' & Class!='Business' ~
'4',
    TRUE ~ 'Other'
  ))
```

Hide

```
Cleansed.Data$clusterid=as.factor(Cleansed.Data$clusterid)
```

Hide

```
table(Cleansed.Data$clusterid)
```

```
    1     2     3     4 Other
17511 26097 17621 10020 19668
```

Around 70K records have been assigned to the 4 partitions we have identified from pam clustering, rest ~20k observations have been labelled as others.

Hide

```
str(Cleansed.Data)
```

```
'data.frame':   90917 obs. of  26 variables:
 $ ID                                : int  149965 149966 149967 149968 149969 149970 149971 1
49972 149973 149974 ...
 $ ArrivalDelayin_Mins               : int  0 0 0 0 0 15 0 26 48 0 ...
 $ Gender                            : Factor w/ 2 levels "Female","Male": 1 1 1 1 2 1 2 2 1 1
...
 $ CustomerType                      : Factor w/ 2 levels "disloyal Customer",..: 2 2 2 2 2 2
2 2 2 2 ...
 $ Age                               : int  65 15 60 70 30 66 10 22 58 34 ...
 $ TypeTravel                        : Factor w/ 2 levels "Business travel",..: 2 2 2 2 2 2 2
2 2 2 ...
 $ Class                             : Factor w/ 3 levels "Business","Eco",..: 2 2 2 2 2 2 2 2
2 2 ...
 $ Flight_Distance                   : int  265 2138 623 354 1894 227 1812 1556 104 3633 ...
 $ DepartureDelayin_Mins             : int  0 0 0 0 0 17 0 30 47 0 ...
 $ Satisfaction                      : Factor w/ 2 levels "dissatisfied",..: 2 2 2 2 2 2 2 2 2
2 ...
 $ Seat_comfort                      : Factor w/ 6 levels "acceptable","excellent",..: 3 3 3 3
3 3 3 3 3 3 ...
 $ Departure.Arrival.time_convenient: Factor w/ 6 levels "acceptable","excellent",..: 3 3 2 3
3 3 3 2 3 3 ...
 $ Food_drink                        : Factor w/ 6 levels "acceptable","excellent",..: 3 3 3 3
3 3 3 3 3 3 ...
 $ Gate_location                     : Factor w/ 6 levels "acceptable","excellent",..: 5 1 1 1
1 1 1 1 1 4 ...
 $ Inflightwifi_service              : Factor w/ 6 levels "acceptable","excellent",..: 5 5 1 4
5 5 5 5 1 5 ...
 $ Inflight_entertainment            : Factor w/ 6 levels "acceptable","excellent",..: 4 3 4 1
3 2 3 3 1 3 ...
 $ Online_support                    : Factor w/ 6 levels "acceptable","excellent",..: 5 5 1 4
5 2 5 5 1 5 ...
 $ Ease_of_Onlinebooking             : Factor w/ 6 levels "acceptable","excellent",..: 1 5 6 5
5 2 5 5 1 5 ...
 $ Onboard_service                   : Factor w/ 6 levels "acceptable","excellent",..: 1 2 6 5
2 2 1 5 1 1 ...
 $ Leg_room_service                  : Factor w/ 6 levels "acceptable","excellent",..: 3 1 3 3
4 3 1 4 3 5 ...
 $ Baggage_handling                  : Factor w/ 5 levels "acceptable","excellent",..: 1 3 5 4
2 2 3 2 5 2 ...
 $ Checkin_service                   : Factor w/ 6 levels "acceptable","excellent",..: 2 4 4 4
2 2 2 1 5 5 ...
 $ Cleanliness                       : Factor w/ 6 levels "acceptable","excellent",..: 1 4 6 5
4 2 4 4 1 2 ...
 $ Online_boarding                   : Factor w/ 6 levels "acceptable","excellent",..: 5 5 1 2
5 1 5 5 2 5 ...
 $ agebin                            : Factor w/ 5 levels "(0,20]","(20,40]",..: 4 1 3 4 2 4 1
2 3 2 ...
 $ clusterid                         : Factor w/ 5 levels "1","2","3","4",..: 2 2 2 2 2 2 2 2
2 2 ...
```

Hide

```
write.csv(Cleansed.Data,'Cleansed_Data.csv',sep=',',row.names = F)
```

Splitting data into Training and Test with 70-30 split. There will be two kinds of split: 1. a 70-30 split for the overall data set 2. a 70-30 split for each of the clusters in the data set

Hide

```
set.seed(100)
Trainingidx = sample(nrow(Cleansed.Data),.7*nrow(Cleansed.Data),replace = F)
TrainingData = Cleansed.Data[Trainingidx,]
TestData = Cleansed.Data[-Trainingidx,]
```

Split for cluster id:1

Hide

```
set.seed(1)
Trainingidx = sample(nrow(Cleansed.Data[Cleansed.Data$clusterid=='1',]),.7*nrow(Cleansed.Data
[Cleansed.Data$clusterid=='1',]),replace = F)
Clust1_TrngData=Cleansed.Data[Cleansed.Data$clusterid=='1',][Trainingidx,]
Clust1_TestData=Cleansed.Data[Cleansed.Data$clusterid=='1',][-Trainingidx,]
```

Split for cluster id:2

Hide

```
set.seed(2)
Trainingidx = sample(nrow(Cleansed.Data[Cleansed.Data$clusterid=='2',]),.7*nrow(Cleansed.Data
[Cleansed.Data$clusterid=='2',]),replace = F)
Clust2_TrngData=Cleansed.Data[Cleansed.Data$clusterid=='2',][Trainingidx,]
Clust2_TestData=Cleansed.Data[Cleansed.Data$clusterid=='2',][-Trainingidx,]
```

Split for cluster id:3

Hide

```
set.seed(3)
Trainingidx = sample(nrow(Cleansed.Data[Cleansed.Data$clusterid=='3',]),.7*nrow(Cleansed.Data
[Cleansed.Data$clusterid=='3',]),replace = F)
Clust3_TrngData=Cleansed.Data[Cleansed.Data$clusterid=='3',][Trainingidx,]
Clust3_TestData=Cleansed.Data[Cleansed.Data$clusterid=='3',][-Trainingidx,]
```

Split for cluster id:4

Hide

```
set.seed(4)
Trainingidx = sample(nrow(Cleansed.Data[Cleansed.Data$clusterid=='4',]),.7*nrow(Cleansed.Data
[Cleansed.Data$clusterid=='4',]),replace = F)
Clust4_TrngData=Cleansed.Data[Cleansed.Data$clusterid=='4',][Trainingidx,]
Clust4_TestData=Cleansed.Data[Cleansed.Data$clusterid=='4',][-Trainingidx,]
```

Split for cluster id:5

Hide

```
levels(Cleansed.Data$clusterid)
```

```
[1] "1"     "2"     "3"     "4"     "Other"
```

Hide

```
set.seed(5)
Trainingidx = sample(nrow(Cleansed.Data[Cleansed.Data$clusterid=='Other',]),.7*nrow(Cleansed.
Data[Cleansed.Data$clusterid=='Other',]),replace = F)
Clust5_TrngData=Cleansed.Data[Cleansed.Data$clusterid=='Other',][Trainingidx,]
Clust5_TestData=Cleansed.Data[Cleansed.Data$clusterid=='Other',][-Trainingidx,]
```

#Decision Tree

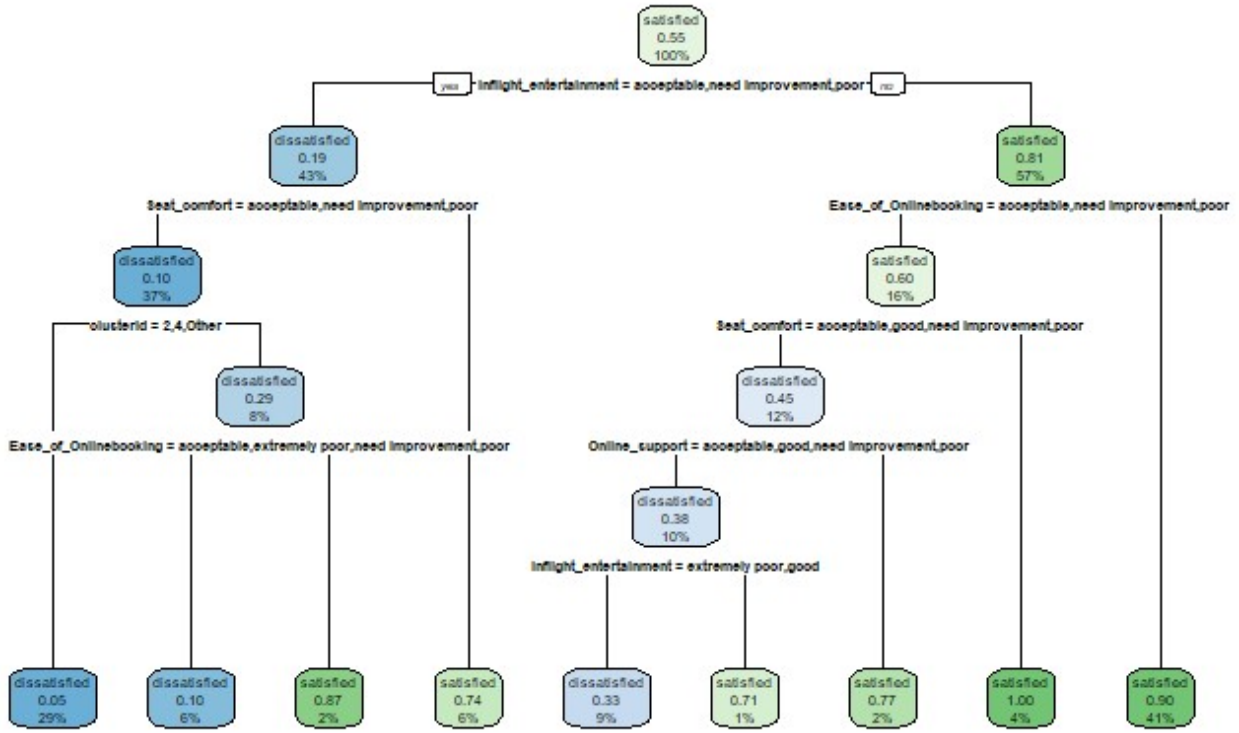
Hide

```
r.control= rpart.control(minbucket = 10,cp=0.009,xval = 10)
DTree = rpart(Satisfaction~.,data = TrainingData[-c(1)],control = r.control,method = 'class')
```

Hide

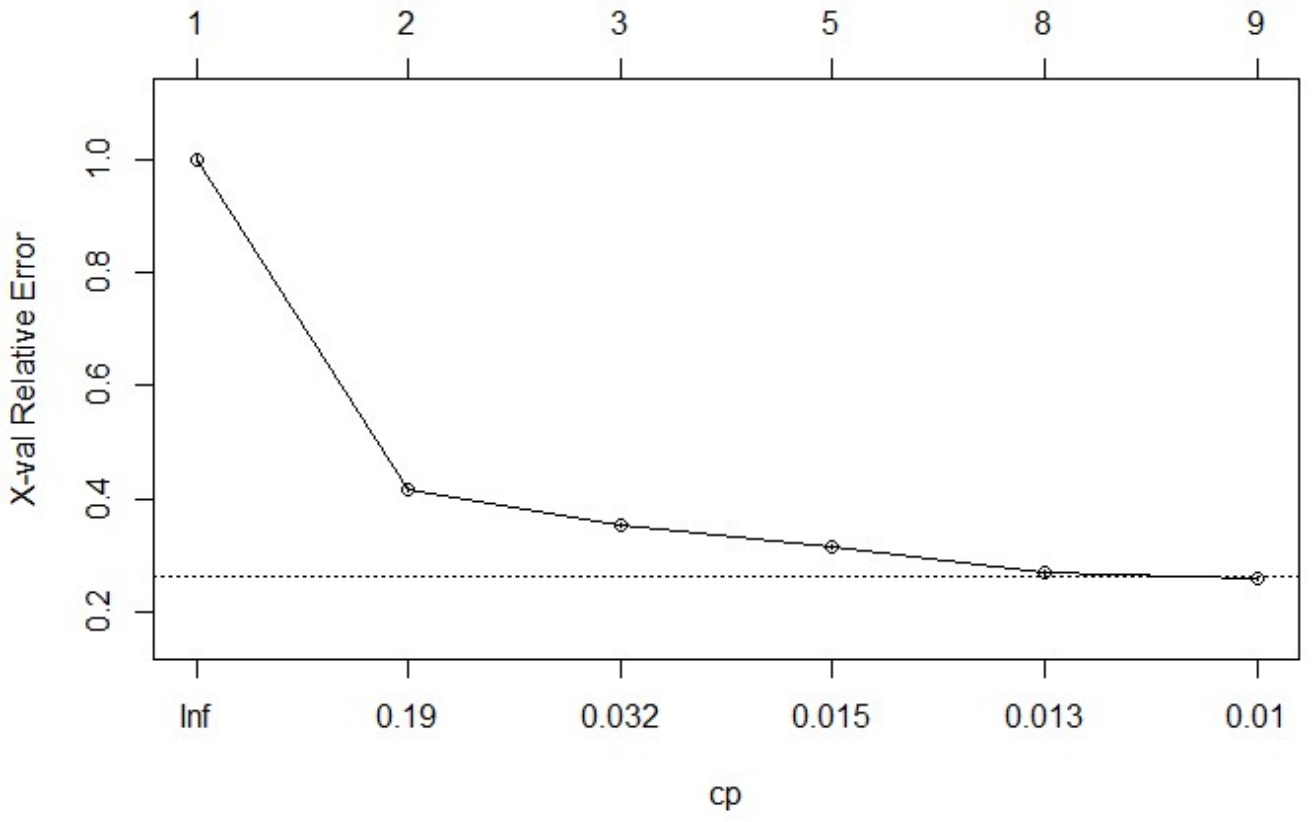
```
rpart.plot(DTree)
```

```
Bad 'data' field in model 'call' (expected a data.frame or a matrix).
To silence this warning:
    Call rpart.plot with roundint=FALSE,
    or rebuild the rpart model with model=TRUE.
```



Hide

```
plotcp(DTree)
```



Hide

```
print(DTree)
```

Hide

```
summary(DTree)
```

```
Call:
rpart(formula = Satisfaction ~ ., data = TrainingData[-c(1)],
    method = "class", control = r.control)
  n= 63641

          CP nsplit rel error    xerror        xstd
1 0.58315687      0 1.0000000 1.0000000 0.004358973
2 0.06477592      1 0.4168431 0.4168431 0.003426356
3 0.01548235      2 0.3520672 0.3520672 0.003205296
4 0.01365987      4 0.3211025 0.3150276 0.003062089
5 0.01190683      7 0.2687888 0.2687541 0.002862609
6 0.00900000      8 0.2568820 0.2584441 0.002814612

Variable importance
Inflight_entertainment           Seat_comfort         Online_support  Ease_of_Onlinebooking
Online_boarding
                    30                     18                     14                     13
9
            Food_drink        Onboard_service            Cleanliness       Baggage_handling
clusterid
                     8                      2                      2                      2
1
      Leg_room_service
                     1

Node number 1: 63641 observations,    complexity param=0.5831569
  predicted class=satisfied     expected loss=0.4526484  P(node) =1
    class counts: 28807 34834
   probabilities: 0.453 0.547
  left son=2 (27113 obs) right son=3 (36528 obs)
  Primary splits:
      Inflight_entertainment splits as  LRRRLL, improve=12050.750, (0 missing)
      Ease_of_Onlinebooking  splits as  LRLRLL, improve= 6278.310, (0 missing)
      Online_support         splits as  LRLRLL, improve= 5816.321, (0 missing)
      Seat_comfort           splits as  LRRRLL, improve= 5399.970, (0 missing)
      clusterid              splits as  RLRLL,  improve= 4417.806, (0 missing)
  Surrogate splits:
      Online_support        splits as  LRRRLL, agree=0.744, adj=0.398, (0 split)
      Seat_comfort          splits as  LRRRLL, agree=0.697, adj=0.288, (0 split)
      Online_boarding       splits as  RRLRLL, agree=0.683, adj=0.256, (0 split)
      Ease_of_Onlinebooking splits as  LRLRLL, agree=0.677, adj=0.243, (0 split)
      Food_drink            splits as  LRRRLL, agree=0.659, adj=0.199, (0 split)

Node number 2: 27113 observations,    complexity param=0.06477592
  predicted class=dissatisfied  expected loss=0.190204  P(node) =0.4260304
    class counts: 21956  5157
   probabilities: 0.810 0.190
  left son=4 (23295 obs) right son=5 (3818 obs)
  Primary splits:
      Seat_comfort     splits as  LRRRLL,     improve=2729.3470, (0 missing)
      clusterid        splits as  RLRLL,      improve= 775.2170, (0 missing)
```

```
      Leg_room_service splits as  LRRRLL,     improve= 703.1380, (0 missing)
      Food_drink       splits as  LRRRLL,     improve= 690.5214, (0 missing)
      Flight_Distance  < 927.5  to the right, improve= 680.0469, (0 missing)
  Surrogate splits:
      Food_drink            splits as  LLRLLL,     agree=0.868, adj=0.063, (0 split)
      Leg_room_service      splits as  LLRLLL,     agree=0.860, adj=0.006, (0 split)
      Flight_Distance       < 126.5  to the right, agree=0.860, adj=0.005, (0 split)
      Online_boarding       splits as  LLRLLL,     agree=0.859, adj=0.001, (0 split)
      Ease_of_Onlinebooking splits as  LLRLLL,     agree=0.859, adj=0.000, (0 split)

Node number 3: 36528 observations,    complexity param=0.01365987
  predicted class=satisfied     expected loss=0.1875548  P(node) =0.5739696
    class counts:  6851 29677
   probabilities: 0.188 0.812
  left son=6 (10319 obs) right son=7 (26209 obs)
  Primary splits:
      Ease_of_Onlinebooking  splits as  LR-RLL, improve=1357.3530, (0 missing)
      Online_support         splits as  LRLRLL, improve=1073.4810, (0 missing)
      Inflight_entertainment splits as  -RLL--, improve= 926.8437, (0 missing)
      Onboard_service        splits as  LRRRLL, improve= 920.9806, (0 missing)
      Online_boarding        splits as  RRLRLL, improve= 901.8905, (0 missing)
  Surrogate splits:
      Online_support   splits as  LRRRLL, agree=0.815, adj=0.346, (0 split)
      Online_boarding  splits as  RRRRLL, agree=0.800, adj=0.291, (0 split)
      Cleanliness      splits as  LR-RLL, agree=0.795, adj=0.275, (0 split)
      Baggage_handling splits as  LRRLL,  agree=0.786, adj=0.242, (0 split)
      Onboard_service  splits as  LRRRLL, agree=0.777, adj=0.210, (0 split)

Node number 4: 23295 observations,    complexity param=0.01548235
  predicted class=dissatisfied  expected loss=0.09937755  P(node) =0.3660376
    class counts: 20980  2315
   probabilities: 0.901 0.099
  left son=8 (18437 obs) right son=9 (4858 obs)
  Primary splits:
      clusterid                         splits as  RLRLL,  improve=453.0192, (0 missing)
      Class                             splits as  RLL,    improve=238.2979, (0 missing)
      Departure.Arrival.time_convenient splits as  RLLLRR, improve=210.9210, (0 missing)
      Onboard_service                   splits as  LRLRLL, improve=191.1825, (0 missing)
      Leg_room_service                  splits as  LRLRLL, improve=178.5581, (0 missing)
  Surrogate splits:
      Class               splits as  RLL,        agree=0.886, adj=0.454, (0 split)
      Flight_Distance     < 3468.5 to the left,  agree=0.812, adj=0.097, (0 split)
      Food_drink          splits as  LRLLLL,     agree=0.793, adj=0.006, (0 split)
      Leg_room_service    splits as  LLRLLL,     agree=0.792, adj=0.005, (0 split)
      ArrivalDelayin_Mins < 711    to the left,  agree=0.792, adj=0.001, (0 split)

Node number 5: 3818 observations
  predicted class=satisfied     expected loss=0.2556312  P(node) =0.05999277
    class counts:   976  2842
   probabilities: 0.256 0.744
```

```
Node number 6: 10319 observations,    complexity param=0.01365987
  predicted class=satisfied     expected loss=0.4047873  P(node) =0.1621439
    class counts:  4177  6142
   probabilities: 0.405 0.595
  left son=12 (7563 obs) right son=13 (2756 obs)
  Primary splits:
      Seat_comfort           splits as  LRRLLL, improve=1227.8570, (0 missing)
      Inflight_entertainment splits as  -RRL--, improve= 636.1387, (0 missing)
      Checkin_service        splits as  RR-RLL, improve= 396.4310, (0 missing)
      Food_drink             splits as  LRRLLL, improve= 386.4067, (0 missing)
      Online_support         splits as  LR-LLL, improve= 275.2373, (0 missing)
  Surrogate splits:
      Food_drink                         splits as  LRRLLL, agree=0.877, adj=0.538, (0 split)
      Departure.Arrival.time_convenient splits as  LLRLLL, agree=0.766, adj=0.122, (0 split)
      Inflight_entertainment             splits as  -LRL--, agree=0.758, adj=0.094, (0 split)
      Gate_location                      splits as  LRLLLL, agree=0.738, adj=0.018, (0 split)
      Leg_room_service                   splits as  LLRLLL, agree=0.737, adj=0.015, (0 split)

Node number 7: 26209 observations
  predicted class=satisfied     expected loss=0.102026  P(node) =0.4118257
    class counts:  2674 23535
   probabilities: 0.102 0.898

Node number 8: 18437 observations
  predicted class=dissatisfied  expected loss=0.04876064  P(node) =0.2897032
    class counts: 17538   899
   probabilities: 0.951 0.049

Node number 9: 4858 observations,    complexity param=0.01548235
  predicted class=dissatisfied  expected loss=0.291478  P(node) =0.07633444
    class counts:  3442  1416
   probabilities: 0.709 0.291
  left son=18 (3648 obs) right son=19 (1210 obs)
  Primary splits:
      Ease_of_Onlinebooking splits as  LRLRLL, improve=1073.3610, (0 missing)
      Onboard_service       splits as  LRLRLL, improve= 831.0414, (0 missing)
      Cleanliness           splits as  LRLRLL, improve= 761.3069, (0 missing)
      Baggage_handling      splits as  LRRLL,  improve= 750.6783, (0 missing)
      Leg_room_service      splits as  LRLRLL, improve= 723.5774, (0 missing)
  Surrogate splits:
      Onboard_service  splits as  LRRRLL, agree=0.864, adj=0.455, (0 split)
      Cleanliness      splits as  LRLLLL, agree=0.842, adj=0.366, (0 split)
      Baggage_handling splits as  LRLLL,  agree=0.838, adj=0.351, (0 split)
      Leg_room_service splits as  LRRRLL, agree=0.819, adj=0.273, (0 split)
      Online_boarding  splits as  LRLLLL, agree=0.797, adj=0.187, (0 split)

Node number 12: 7563 observations,    complexity param=0.01365987
  predicted class=dissatisfied  expected loss=0.4479704  P(node) =0.1188385
    class counts:  4175  3388
   probabilities: 0.552 0.448
  left son=24 (6217 obs) right son=25 (1346 obs)
```

```
  Primary splits:
      Online_support         splits as  LR-LLL, improve=334.2716, (0 missing)
      Online_boarding        splits as  RR-RLL, improve=318.3781, (0 missing)
      Inflight_entertainment splits as  -RLL--, improve=307.2764, (0 missing)
      Checkin_service        splits as  RR-RLL, improve=303.3174, (0 missing)
      Class                  splits as  RLL,    improve=180.9948, (0 missing)
  Surrogate splits:
      Online_boarding       splits as  LR-LLL,     agree=0.837, adj=0.085, (0 split)
      Flight_Distance       < 52.5   to the right, agree=0.822, adj=0.001, (0 split)
      DepartureDelayin_Mins < 453    to the left,  agree=0.822, adj=0.001, (0 split)

Node number 13: 2756 observations
  predicted class=satisfied     expected loss=0.0007256894  P(node) =0.04330542
    class counts:     2  2754
   probabilities: 0.001 0.999

Node number 18: 3648 observations
  predicted class=dissatisfied  expected loss=0.1000548  P(node) =0.05732154
    class counts:  3283   365
   probabilities: 0.900 0.100

Node number 19: 1210 observations
  predicted class=satisfied     expected loss=0.131405  P(node) =0.0190129
    class counts:   159  1051
   probabilities: 0.131 0.869

Node number 24: 6217 observations,    complexity param=0.01190683
  predicted class=dissatisfied  expected loss=0.3788001  P(node) =0.0976886
    class counts:  3862  2355
   probabilities: 0.621 0.379
  left son=48 (5414 obs) right son=49 (803 obs)
  Primary splits:
      Inflight_entertainment splits as  -RLL--, improve=206.68630, (0 missing)
      Checkin_service        splits as  RR-RLL, improve=196.60830, (0 missing)
      Online_boarding        splits as  LR-LLL, improve=178.74110, (0 missing)
      Online_support         splits as  L--RLL, improve= 97.01108, (0 missing)
      Class                  splits as  RLL,    improve= 70.29236, (0 missing)
  Surrogate splits:
      Online_boarding splits as  LR-LLL, agree=0.873, adj=0.019, (0 split)

Node number 25: 1346 observations
  predicted class=satisfied     expected loss=0.2325409  P(node) =0.02114989
    class counts:   313  1033
   probabilities: 0.233 0.767

Node number 48: 5414 observations
  predicted class=dissatisfied  expected loss=0.3291467  P(node) =0.08507094
    class counts:  3632  1782
   probabilities: 0.671 0.329

Node number 49: 803 observations
```

```
  predicted class=satisfied     expected loss=0.2864259  P(node) =0.01261765
    class counts:   230   573
   probabilities: 0.286 0.714
```

Hide

```
path.rpart(DTree,13,print.it = T)
```

```
 node number: 13
   root
   Inflight_entertainment=excellent,extremely poor,good
   Ease_of_Onlinebooking=acceptable,need improvement,poor
   Seat_comfort=excellent,extremely poor
```

Hide

```
predicted.class=predict(DTree,TrainingData[-c(1,10)],'class')
predicted.prob=predict(DTree,TrainingData[,-c(1,10)],'prob')
```

Hide

```
table(TrainingData$Satisfaction,predicted.class)
```

```
              predicted.class
               dissatisfied satisfied
  dissatisfied        24453      4354
  satisfied            3046     31788
```

Hide

```
sum(diag(table(TrainingData$Satisfaction,predicted.class)))/nrow(TrainingData)
```

```
[1] 0.8837228
```

Hide

```
library(pROC)
roc(TrainingData$Satisfaction,predicted.prob[,2])
```

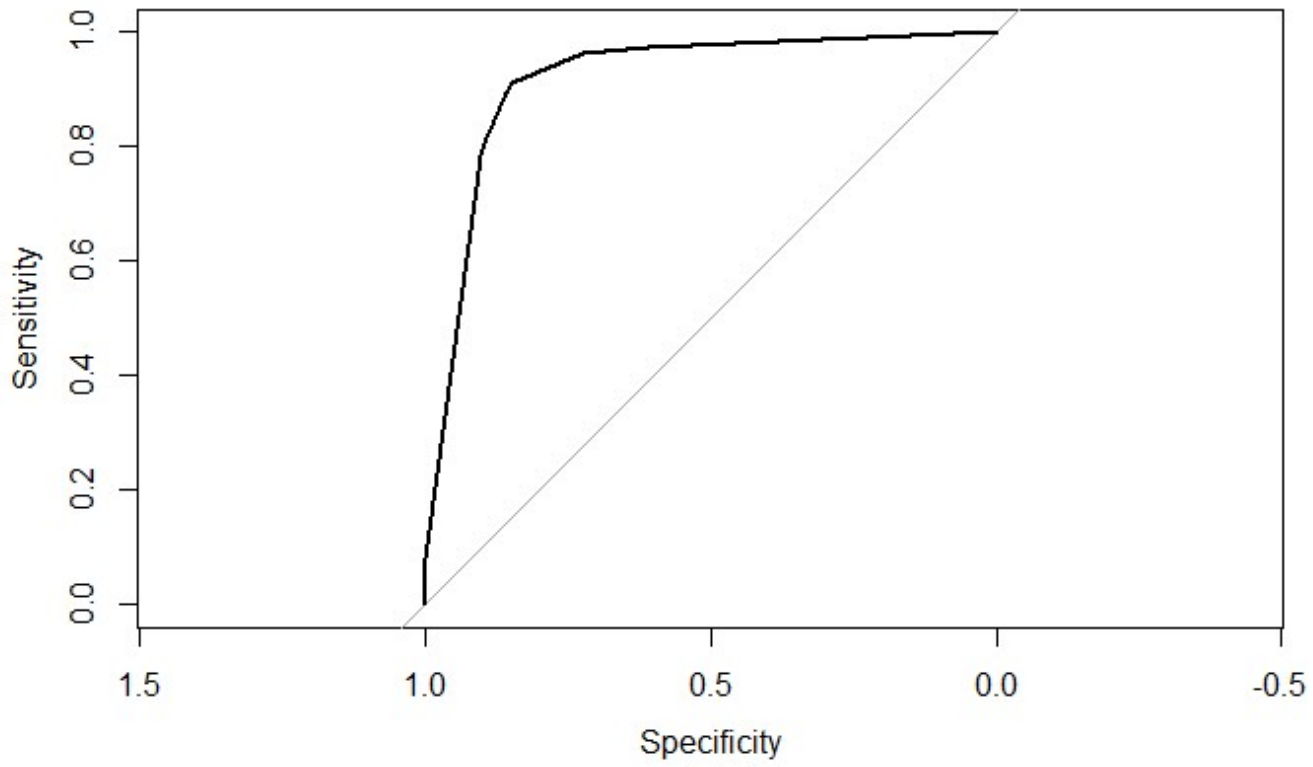
```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```

```
Call:
roc.default(response = TrainingData$Satisfaction, predictor = predicted.prob[,     2])

Data: predicted.prob[, 2] in 28807 controls (TrainingData$Satisfaction dissatisfied) < 34834
cases (TrainingData$Satisfaction satisfied).
Area under the curve: 0.9175
```

Hide

```
plot(roc(TrainingData$Satisfaction,predicted.prob[,2]))
```

```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```



Hide

```
predicted.class=predict(DTree,TestData[-c(1,10)],'class')
predicted.prob=predict(DTree,TestData[,-c(1,10)],'prob')
```

Hide

```
#confusion matrix on Test Data
table(TestData$Satisfaction,predicted.class)
```

```
             predicted.class
              dissatisfied satisfied
  dissatisfied        10445      1904
  satisfied            1352     13575
```

Hide

```
#accuracy on Training Data
sum(diag(table(TestData$Satisfaction,predicted.class)))/nrow(TestData)
```

```
[1] 0.8806277
```

Hide

```
roc(TestData$Satisfaction,predicted.prob[,2])
```

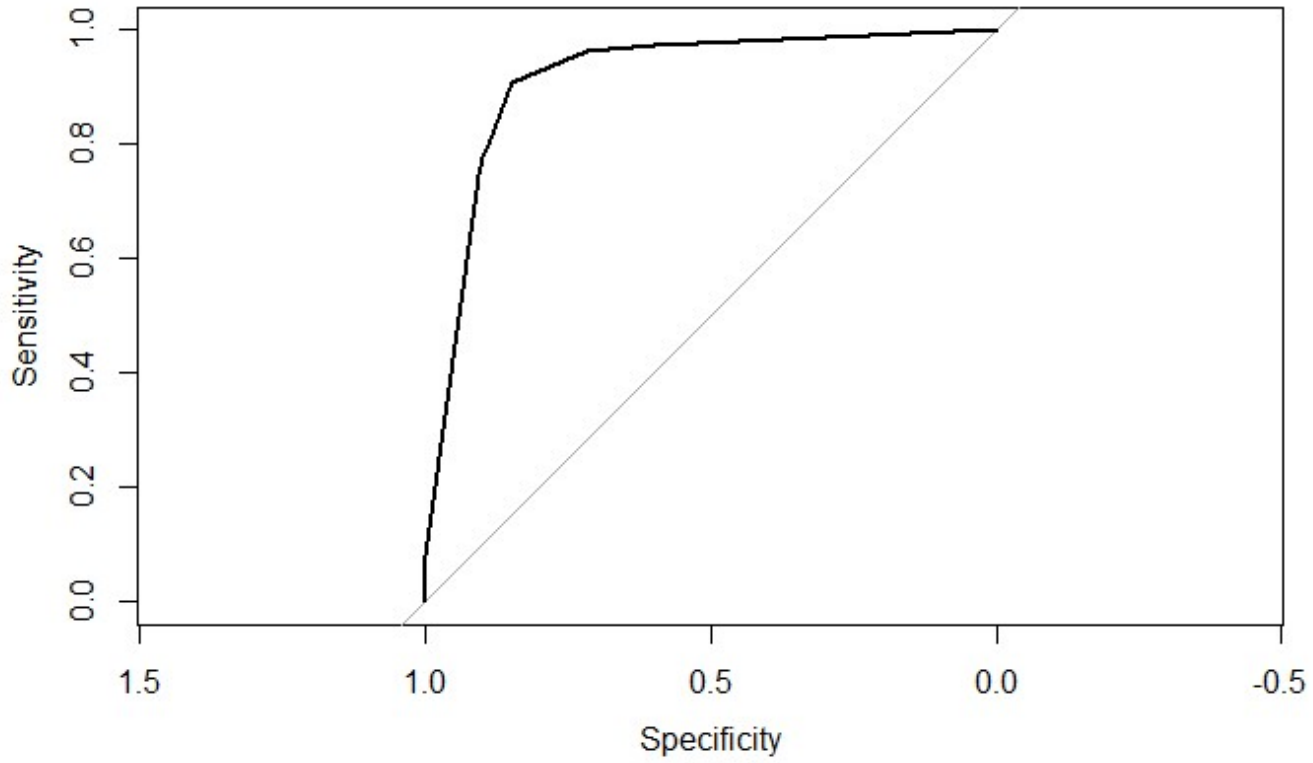
```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```

```

Call:
roc.default(response = TestData$Satisfaction, predictor = predicted.prob[,     2])

Data: predicted.prob[, 2] in 12349 controls (TestData$Satisfaction dissatisfied) < 14927 case
s (TestData$Satisfaction satisfied).
Area under the curve: 0.9152
```

Hide

```
plot(roc(TestData$Satisfaction,predicted.prob[,2]))
```

```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```



#Logistic Regression

Hide

```
aviation.logistic=glm(Satisfaction~.,data = TrainingData[-c(1)],family = 'binomial')
```

```
glm.fit: fitted probabilities numerically 0 or 1 occurred
```

Hide

```
summary(aviation.logistic)
```

```
Call:
glm(formula = Satisfaction ~ ., family = "binomial", data = TrainingData[-c(1)])

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.3041  -0.2809   0.0135   0.2343   3.4447

Coefficients: (2 not defined because of singularities)
                                                Estimate Std. Error z value Pr(>|z|)
(Intercept)                                   -2.141e+00  3.107e-01  -6.889 5.61e-12 ***
ArrivalDelayin_Mins                           -7.126e-03  1.517e-03  -4.697 2.64e-06 ***
GenderMale                                    -1.406e+00  4.076e-02 -34.488  < 2e-16 ***
CustomerTypeLoyal Customer                     1.886e+00  2.871e-01   6.571 5.01e-11 ***
Age                                           -1.469e-02  2.852e-03  -5.152 2.57e-07 ***
TypeTravelPersonal Travel                     -1.114e+00  2.863e-01  -3.891 9.99e-05 ***
ClassEco                                       1.659e-01  2.862e-01   0.580 0.562097
ClassEco Plus                                  2.577e-02  2.910e-01   0.089 0.929426
Flight_Distance                               -5.838e-05  1.679e-05  -3.476 0.000509 ***
DepartureDelayin_Mins                          2.891e-03  1.532e-03   1.887 0.059185 .
Seat_comfortexcellent                          6.020e+00  1.518e-01  39.668  < 2e-16 ***
Seat_comfortextremely poor                     3.447e+01  2.081e+02   0.166 0.868470
Seat_comfortgood                               1.197e+00  5.525e-02  21.672  < 2e-16 ***
Seat_comfortneed improvement                   1.106e-01  6.226e-02   1.776 0.075728 .
Seat_comfortpoor                               4.270e-01  7.265e-02   5.877 4.17e-09 ***
Departure.Arrival.time_convenientexcellent    -2.064e+00  7.248e-02 -28.476  < 2e-16 ***
Departure.Arrival.time_convenientextremely poor -4.327e-01  9.972e-02  -4.340 1.43e-05 ***
Departure.Arrival.time_convenientgood         -1.020e+00  6.347e-02 -16.074  < 2e-16 ***
Departure.Arrival.time_convenientneed improvement  2.709e-02  7.196e-02   0.377 0.706518
Departure.Arrival.time_convenientpoor         -3.619e-02  7.719e-02  -0.469 0.639181
Food_drinkexcellent                            3.972e-01  8.248e-02   4.816 1.46e-06 ***
Food_drinkextremely poor                      -1.614e+00  4.029e-01  -4.005 6.20e-05 ***
Food_drinkgood                                -8.043e-02  6.565e-02  -1.225 0.220503
Food_drinkneed improvement                    -4.308e-01  7.357e-02  -5.855 4.78e-09 ***
Food_drinkpoor                                -3.827e-01  8.255e-02  -4.636 3.55e-06 ***
Gate_locationexcellent                         9.032e-02  7.793e-02   1.159 0.246406
Gate_locationextremely poor                    2.112e+01  1.075e+04   0.002 0.998433
Gate_locationgood                              3.067e-01  5.335e-02   5.749 8.97e-09 ***
Gate_locationneed improvement                  3.600e-01  5.953e-02   6.048 1.47e-09 ***
Gate_locationpoor                              3.436e-01  6.343e-02   5.417 6.06e-08 ***
Inflightwifi_serviceexcellent                 -1.978e-01  6.359e-02  -3.111 0.001863 **
Inflightwifi_serviceextremely poor            -1.226e+01  1.438e+02  -0.085 0.932054
Inflightwifi_servicegood                      -1.062e-02  6.144e-02  -0.173 0.862812
Inflightwifi_serviceneed improvement           2.010e-01  6.445e-02   3.118 0.001818 **
Inflightwifi_servicepoor                      -5.508e-02  7.634e-02  -0.722 0.470595
Inflight_entertainmentexcellent                2.911e+00  6.567e-02  44.330  < 2e-16 ***
Inflight_entertainmentextremely poor           1.217e+00  4.284e-01   2.841 0.004493 **
Inflight_entertainmentgood                     1.797e+00  4.759e-02  37.749  < 2e-16 ***
Inflight_entertainmentneed improvement         1.812e-01  6.270e-02   2.891 0.003845 **
Inflight_entertainmentpoor                     1.165e-01  7.368e-02   1.582 0.113747
Online_supportexcellent                        1.451e+00  6.269e-02  23.153  < 2e-16 ***
```

```
Online_supportextremely poor                    -1.906e+01  1.075e+04  -0.002 0.998586
Online_supportgood                               7.994e-01  5.327e-02  15.006  < 2e-16 ***
Online_supportneed improvement                   1.028e+00  6.797e-02  15.132  < 2e-16 ***
Online_supportpoor                               1.525e+00  7.558e-02  20.178  < 2e-16 ***
Ease_of_Onlinebookingexcellent                  -7.090e-01  8.518e-02  -8.324  < 2e-16 ***
Ease_of_Onlinebookingextremely poor             -3.983e+01  3.891e+03  -0.010 0.991831
Ease_of_Onlinebookinggood                        5.749e-02  7.461e-02   0.771 0.440973
Ease_of_Onlinebookingneed improvement           -9.695e-01  9.184e-02 -10.556  < 2e-16 ***
Ease_of_Onlinebookingpoor                       -1.951e+00  1.064e-01 -18.339  < 2e-16 ***
Onboard_serviceexcellent                         6.610e-01  5.582e-02  11.841  < 2e-16 ***
Onboard_serviceextremely poor                   -4.173e-01  1.724e+00  -0.242 0.808797
Onboard_servicegood                              1.692e-01  5.049e-02   3.352 0.000803 ***
Onboard_serviceneed improvement                 -4.212e-01  6.524e-02  -6.455 1.08e-10 ***
Onboard_servicepoor                             -5.379e-01  6.777e-02  -7.937 2.07e-15 ***
Leg_room_serviceexcellent                        8.378e-01  5.269e-02  15.902  < 2e-16 ***
Leg_room_serviceextremely poor                  -4.173e-01  5.618e-01  -0.743 0.457556
Leg_room_servicegood                             6.517e-01  4.821e-02  13.519  < 2e-16 ***
Leg_room_serviceneed improvement                 1.528e-01  5.354e-02   2.853 0.004326 **
Leg_room_servicepoor                            -5.698e-02  6.981e-02  -0.816 0.414358
Baggage_handlingexcellent                        9.976e-01  6.024e-02  16.559  < 2e-16 ***
Baggage_handlinggood                             4.787e-01  5.253e-02   9.114  < 2e-16 ***
Baggage_handlingneed improvement                 5.212e-01  7.039e-02   7.404 1.32e-13 ***
Baggage_handlingpoor                             5.995e-01  8.149e-02   7.358 1.87e-13 ***
Checkin_serviceexcellent                         5.430e-01  4.778e-02  11.365  < 2e-16 ***
Checkin_serviceextremely poor                           NA         NA      NA       NA
Checkin_servicegood                             -2.634e-02  4.110e-02  -0.641 0.521560
Checkin_serviceneed improvement                 -5.025e-01  5.320e-02  -9.446  < 2e-16 ***
Checkin_servicepoor                             -6.734e-01  5.389e-02 -12.496  < 2e-16 ***
Cleanlinessexcellent                             1.097e+00  6.394e-02  17.152  < 2e-16 ***
Cleanlinessextremely poor                        2.155e+01  1.144e+04   0.002 0.998496
Cleanlinessgood                                  5.542e-01  5.571e-02   9.949  < 2e-16 ***
Cleanlinessneed improvement                      6.336e-01  7.450e-02   8.505  < 2e-16 ***
Cleanlinesspoor                                  8.173e-01  8.587e-02   9.518  < 2e-16 ***
Online_boardingexcellent                         1.738e-01  6.747e-02   2.577 0.009980 **
Online_boardingextremely poor                           NA         NA      NA       NA
Online_boardinggood                             -1.141e-01  5.771e-02  -1.977 0.047989 *
Online_boardingneed improvement                 -4.247e-01  7.107e-02  -5.976 2.29e-09 ***
Online_boardingpoor                             -4.194e-01  7.377e-02  -5.685 1.31e-08 ***
agebin(20,40]                                    2.418e-01  7.105e-02   3.403 0.000666 ***
agebin(40,60]                                    6.204e-01  1.121e-01   5.535 3.11e-08 ***
agebin(60,80]                                    3.555e-01  1.606e-01   2.213 0.026900 *
agebin(80,100]                                  -2.221e+00  1.293e+00  -1.718 0.085805 .
clusterid2                                      -1.624e+00  5.644e-01  -2.877 0.004016 **
clusterid3                                      -1.433e+00  6.602e-02 -21.697  < 2e-16 ***
clusterid4                                      -2.469e+00  5.727e-01  -4.310 1.63e-05 ***
clusteridOther                                  -1.613e+00  2.893e-01  -5.576 2.47e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 87654  on 63640  degrees of freedom
Residual deviance: 28584  on 63556  degrees of freedom
AIC: 28754

Number of Fisher Scoring iterations: 18
```

Inflight entertainment,Seat Comfort,Gender,Departure Arrival Time convenience,Online Support,Seat comfort,Ease of Online booking, Cleanliness,Baggage Handling, Legroom service,Checkin service, Onboard service

Hide

```
?varImp
a=varImp(aviation.logistic)
a=data.frame('Overall'=a$Overall,'Names'=row.names(a))
a[order(-a$Overall),]
```

	Overall <dbl>	Names <fctr>
35	44.330462706	Inflight_entertainmentexcellent
10	39.667617627	Seat_comfortexcellent
37	37.748914592	Inflight_entertainmentgood
2	34.488098411	GenderMale
15	28.475614192	Departure.Arrival.time_convenientexcellent
40	23.153101955	Online_supportexcellent
82	21.697104518	clusterid3
12	21.672392303	Seat_comfortgood
44	20.178498619	Online_supportpoor
49	18.339137082	Ease_of_Onlinebookingpoor

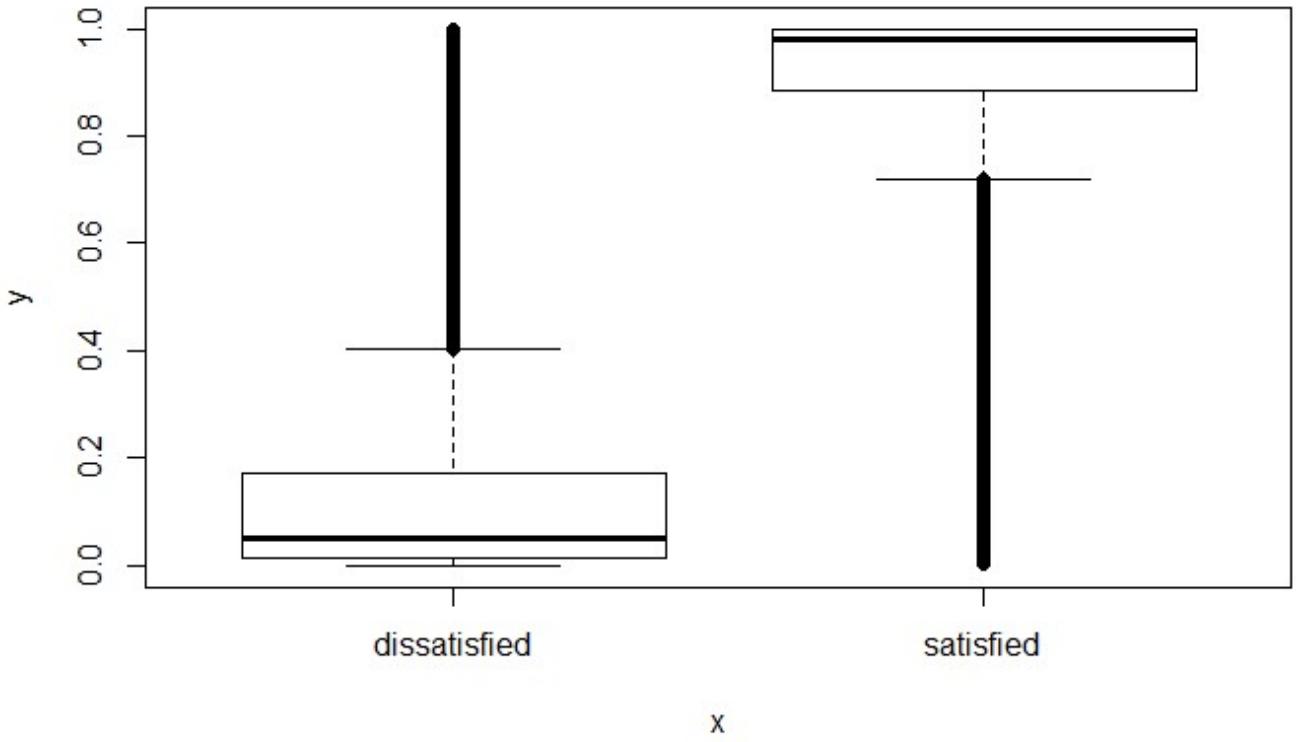
1-10 of 84 rows Previous **1** 2 3 4 5 6 … 9 Next

Hide

```
rm(a)
```

Hide

```
plot(TrainingData$Satisfaction,aviation.logistic$fitted.values)
```



Hide

```
predicted.class=ifelse(aviation.logistic$fitted.values>0.5,'satisfied','dissatisfied')
```

Hide

```
#confusion matrix on Training Data
table(TrainingData$Satisfaction,predicted.class)
```

```
              predicted.class
               dissatisfied satisfied
  dissatisfied        26130      2677
  satisfied            3021     31813
```

Hide

```
#accuracy on Training Data
sum(diag(table(TrainingData$Satisfaction,predicted.class)))/nrow(TrainingData)
```

```
[1] 0.9104665
```

Hide

```
#TNR
table(TrainingData$Satisfaction,predicted.class)[1,1]/sum(table(TrainingData$Satisfaction,pre
dicted.class)[1,])
```

```
[1] 0.9070712
```

Hide

```
#TPR
table(TrainingData$Satisfaction,predicted.class)[2,2]/sum(table(TrainingData$Satisfaction,pre
dicted.class)[2,])
```

```
[1] 0.9132744
```

Hide

```
roc(TrainingData$Satisfaction,aviation.logistic$fitted.values)
```

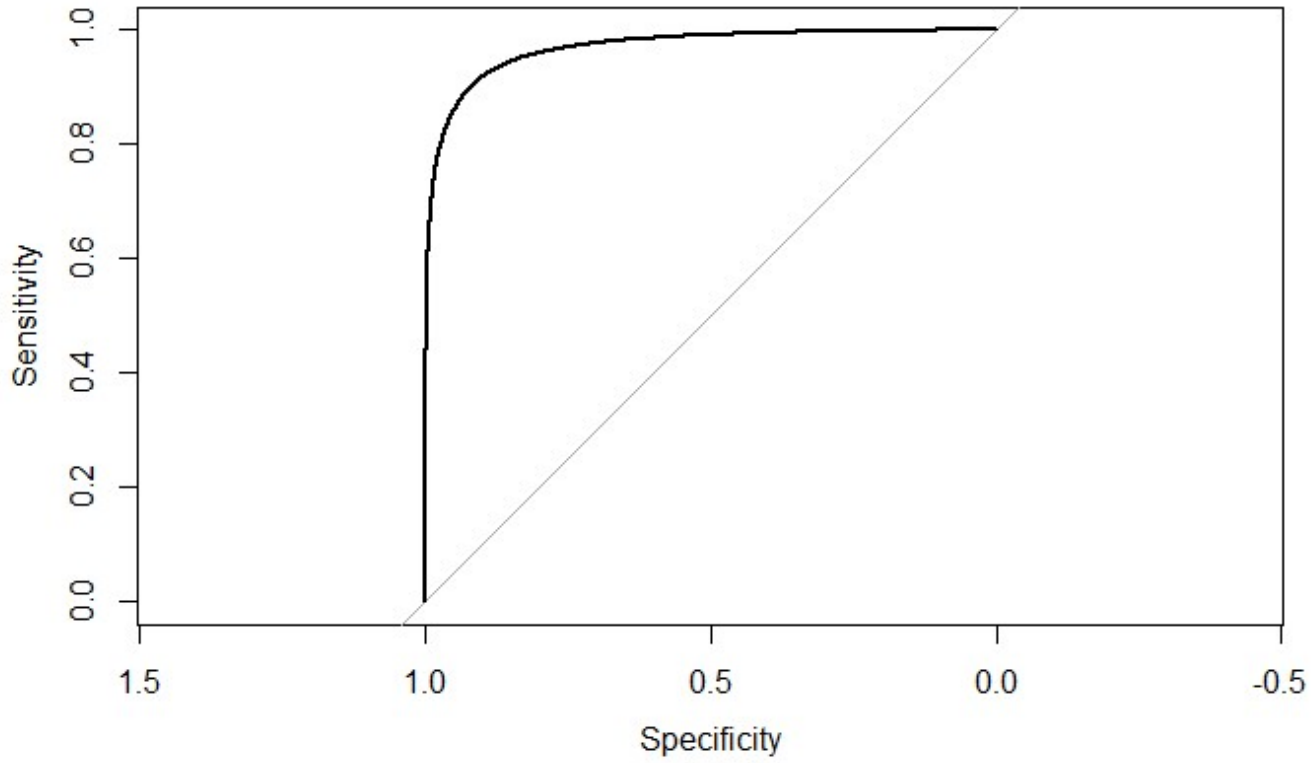
```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```

```
Call:
roc.default(response = TrainingData$Satisfaction, predictor = aviation.logistic$fitted.value
s)

Data: aviation.logistic$fitted.values in 28807 controls (TrainingData$Satisfaction dissatisfi
ed) < 34834 cases (TrainingData$Satisfaction satisfied).
Area under the curve: 0.9687
```

Hide

```
plot(roc(TrainingData$Satisfaction,aviation.logistic$fitted.values))
```

```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```



Hide

```
predicted.prob=predict.glm(aviation.logistic,newdata = TestData[-c(1,10)],type = 'response')
```

```
prediction from a rank-deficient fit may be misleading
```

Hide

```
predicted.class=predicted.class=ifelse(predicted.prob>0.5,'satisfied','dissatisfied')
```

Hide

```
#confusion matrix
table(TestData$Satisfaction,predicted.class)
```

```
             predicted.class
              dissatisfied satisfied
  dissatisfied        11207      1142
  satisfied            1316     13611
```

Hide

```
#accuracy
accuracy = sum(diag(table(TestData$Satisfaction,predicted.class)))/nrow(TestData)
accuracy
```

```
[1] 0.9098841
```

Hide

```
#TPR
table(TestData$Satisfaction,predicted.class)[1,1]/sum(table(TestData$Satisfaction,predicted.class)[1,])
```

```
[1] 0.9075229
```

Hide

```
#TNR
table(TestData$Satisfaction,predicted.class)[2,2]/sum(table(TestData$Satisfaction,predicted.class)[2,])
```

```
[1] 0.9118376
```

Hide

```
#AUC
roc(TestData$Satisfaction,predicted.prob)
```

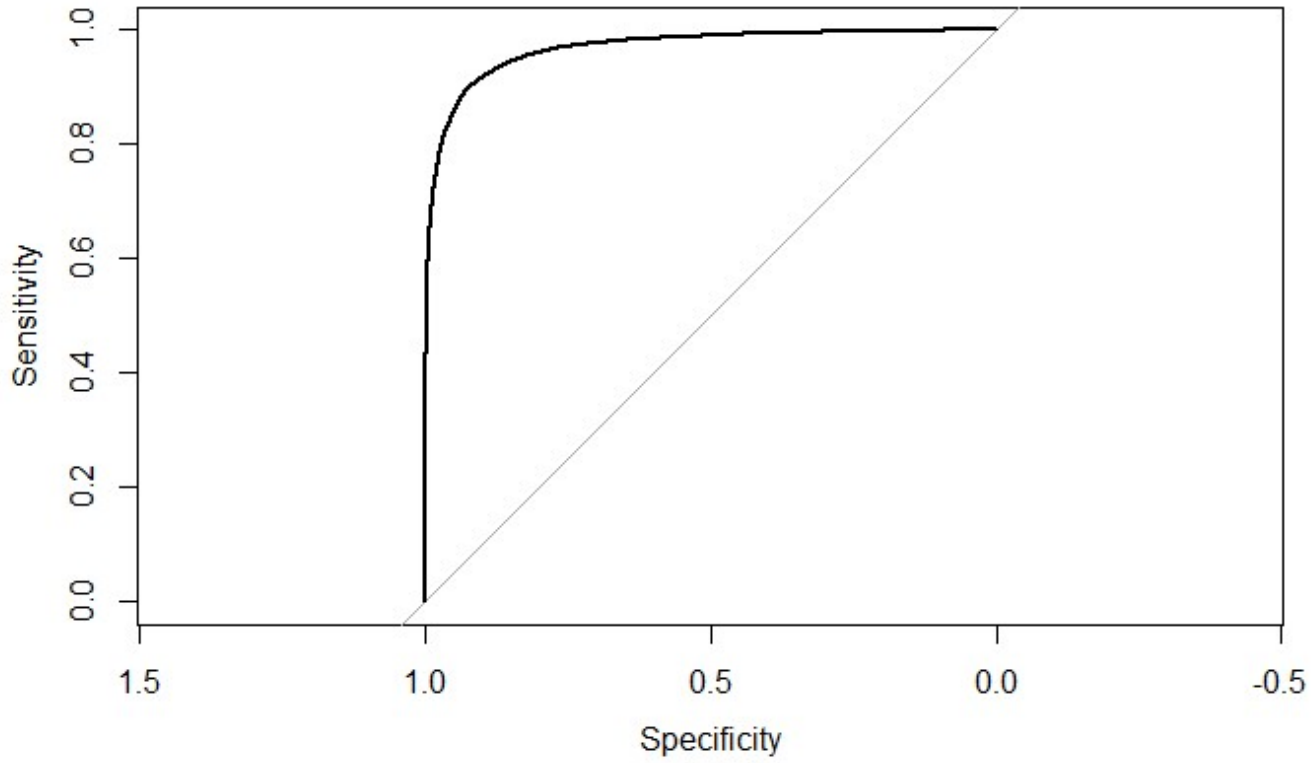
```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```

```

Call:
roc.default(response = TestData$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 12349 controls (TestData$Satisfaction dissatisfied) < 14927 cases (TestData$Satisfaction satisfied).
Area under the curve: 0.968
```

Hide

```
plot(roc(TestData$Satisfaction,predicted.prob))
```

```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```



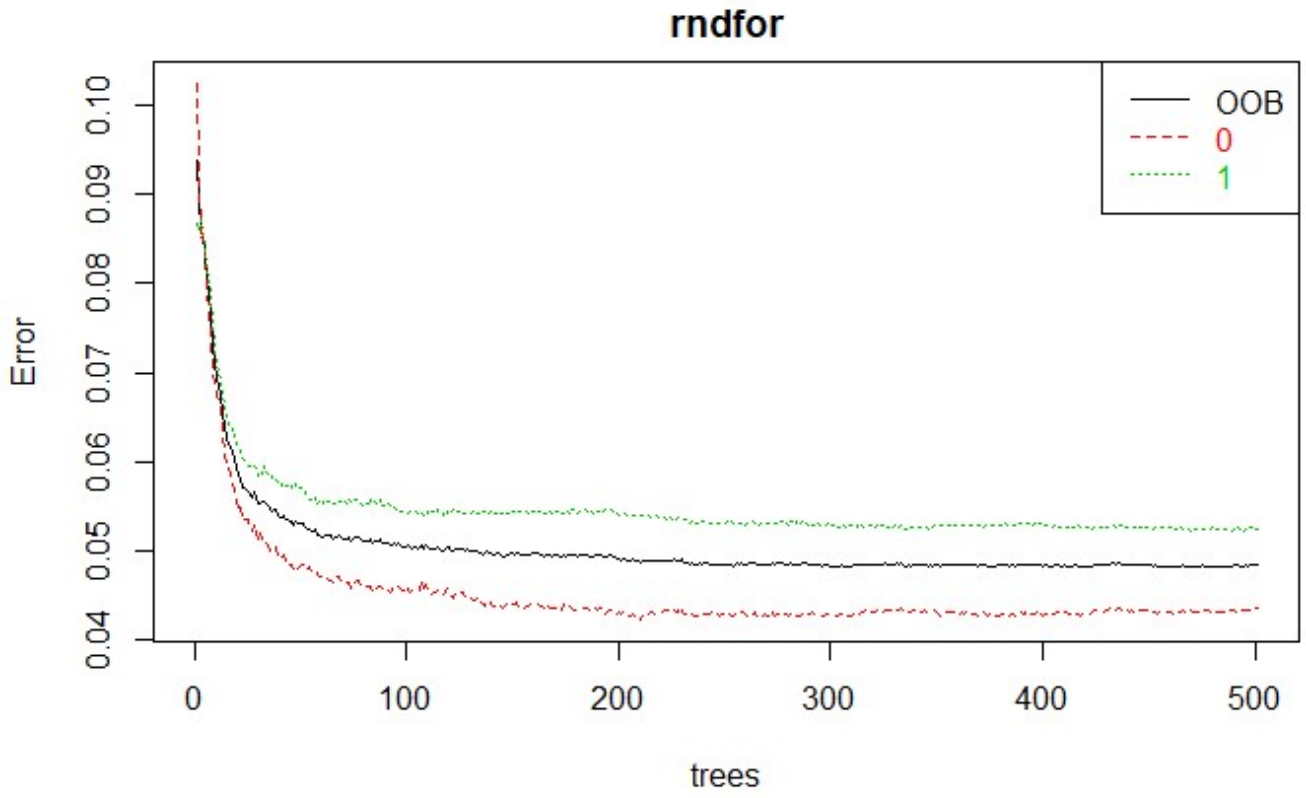
#Random Forest

Hide

```
set.seed(11000)
rndfor = randomForest(Satisfaction~.,data = TrainingData[-c(1)],ntree=501,mtry=4,nodesize=10,
importance=T)
```

Hide

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



Hide

```
print(rndfor)
```

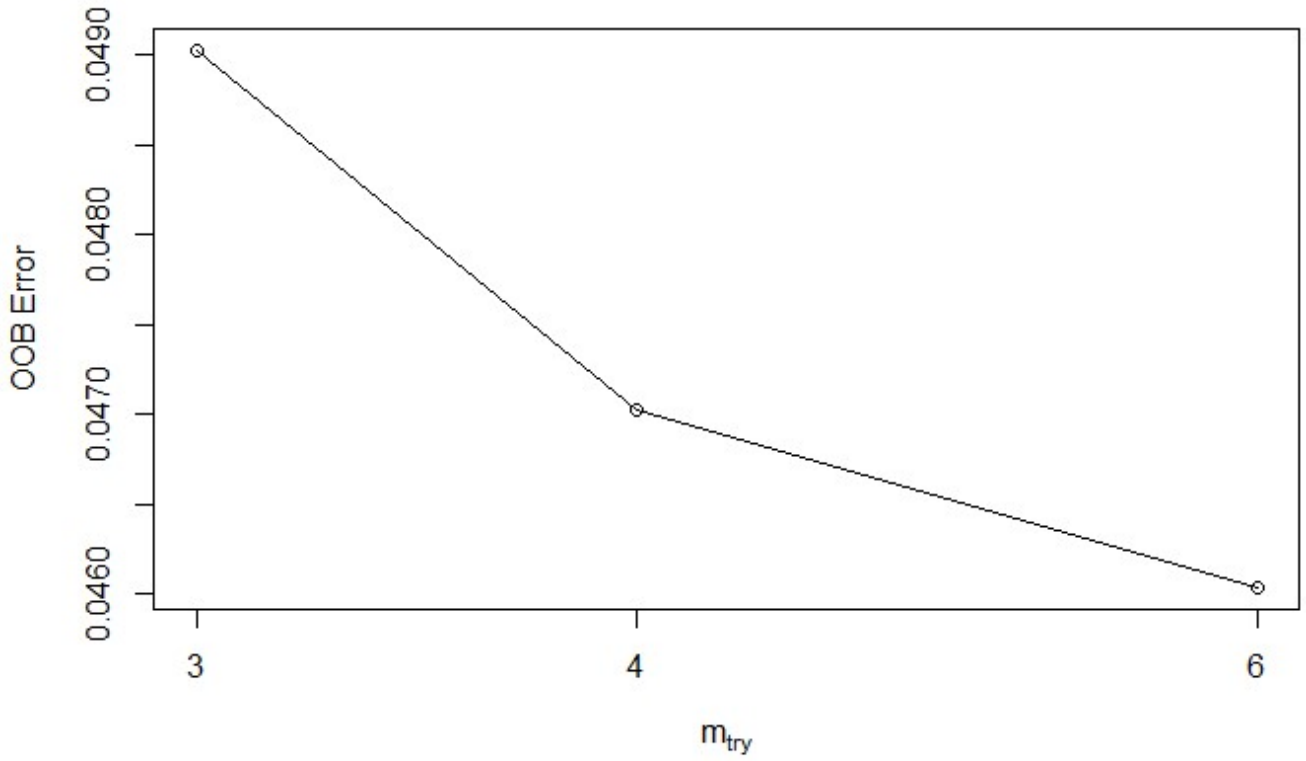
Hide

```
importance(rndfor)
```

Hide

```
trndfor = tuneRF(x=TrainingData[-c(1,10)],
                 y=TrainingData$Satisfaction,
                 mtryStart = 4,
                 stepFactor = 1.5,
                 ntreeTry = 501,
                 trace = T,
                 plot = T,
                 importance=T,
                 doBest = T)
```

```
mtry = 4  OOB error = 4.7%
Searching left ...
mtry = 3    OOB error = 4.9%
-0.04243234 0.05
Searching right ...
mtry = 6    OOB error = 4.6%
0.02104911 0.05
```



Hide

```
finalfor = randomForest(Satisfaction~.,data = TrainingData[-c(1)],ntree=501,mtry=6,nodesize=1
0,importance=T)
```

Hide

```
print(finalfor)
```

```
Call:
 randomForest(formula = Satisfaction ~ ., data = TrainingData[-c(1)],      ntree = 501, mtry
= 6, nodesize = 10, importance = T)
               Type of random forest: classification
                     Number of trees: 501
No. of variables tried at each split: 6

        OOB estimate of  error rate: 4.66%
Confusion matrix:
             dissatisfied satisfied class.error
dissatisfied        27664      1143  0.03967786
satisfied            1821     33013  0.05227651
```

Hide

```
predicted.class = predict(finalfor,TrainingData[-c(1,10)],'class')
predicted.prob = predict(finalfor,TrainingData[-c(1,10)],'prob')
```

Hide

```
#confusion matrix on Training Data
table(TrainingData$Satisfaction,predicted.class)
```

```
              predicted.class
               dissatisfied satisfied
  dissatisfied        28605       202
  satisfied             518     34316
```

Hide

```
#accuracy on Training Data
sum(diag(table(TrainingData$Satisfaction,predicted.class)))/nrow(TrainingData)
```

```
[1] 0.9886865
```

Hide

```
#TNR
table(TrainingData$Satisfaction,predicted.class)[1,1]/sum(table(TrainingData$Satisfaction,pre
dicted.class)[1,])
```

```
[1] 0.9929878
```

Hide

```
#TPR
table(TrainingData$Satisfaction,predicted.class)[2,2]/sum(table(TrainingData$Satisfaction,pre
dicted.class)[2,])
```

```
[1] 0.9851295
```

Hide

```
#AUC
roc(TrainingData$Satisfaction,predicted.prob[,2])
```

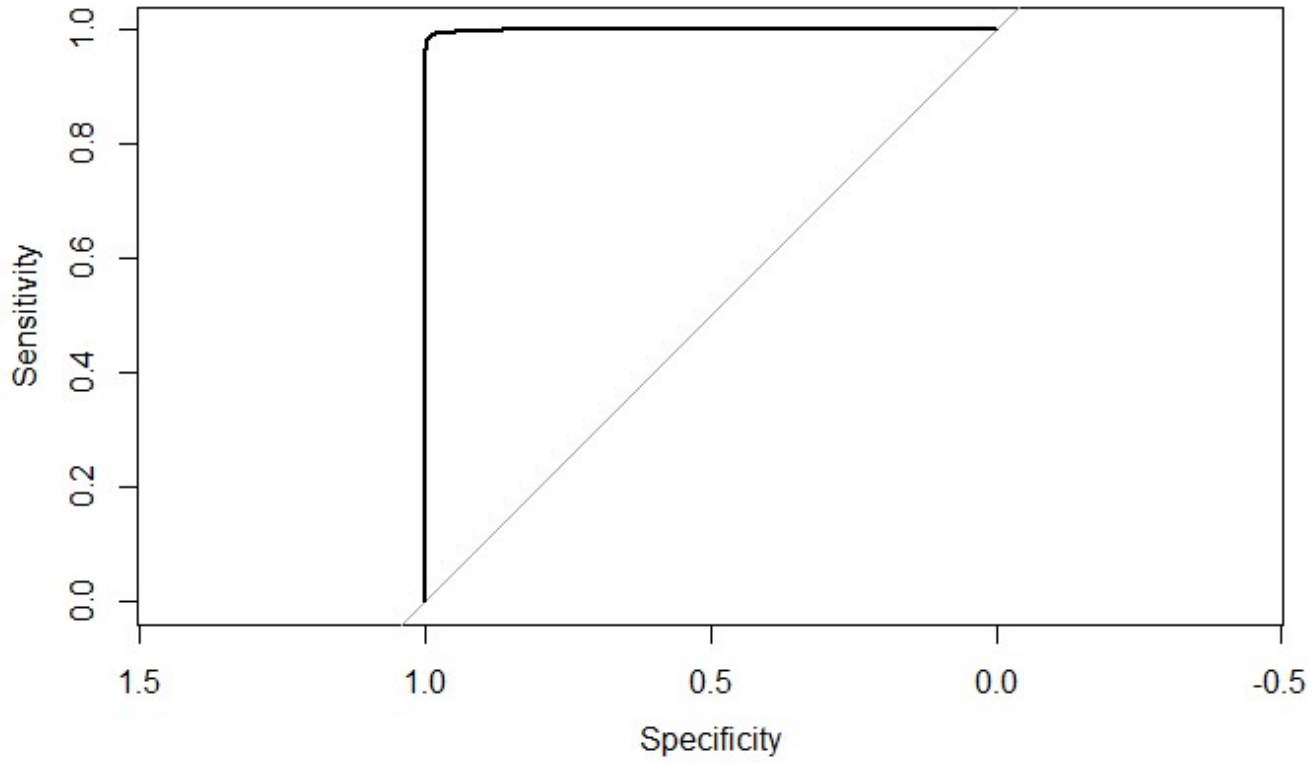
```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```

```

Call:
roc.default(response = TrainingData$Satisfaction, predictor = predicted.prob[,     2])

Data: predicted.prob[, 2] in 28807 controls (TrainingData$Satisfaction dissatisfied) < 34834
cases (TrainingData$Satisfaction satisfied).
Area under the curve: 0.9993
```

Hide

```
plot(roc(TrainingData$Satisfaction,predicted.prob[,2]))
```

```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```



Hide

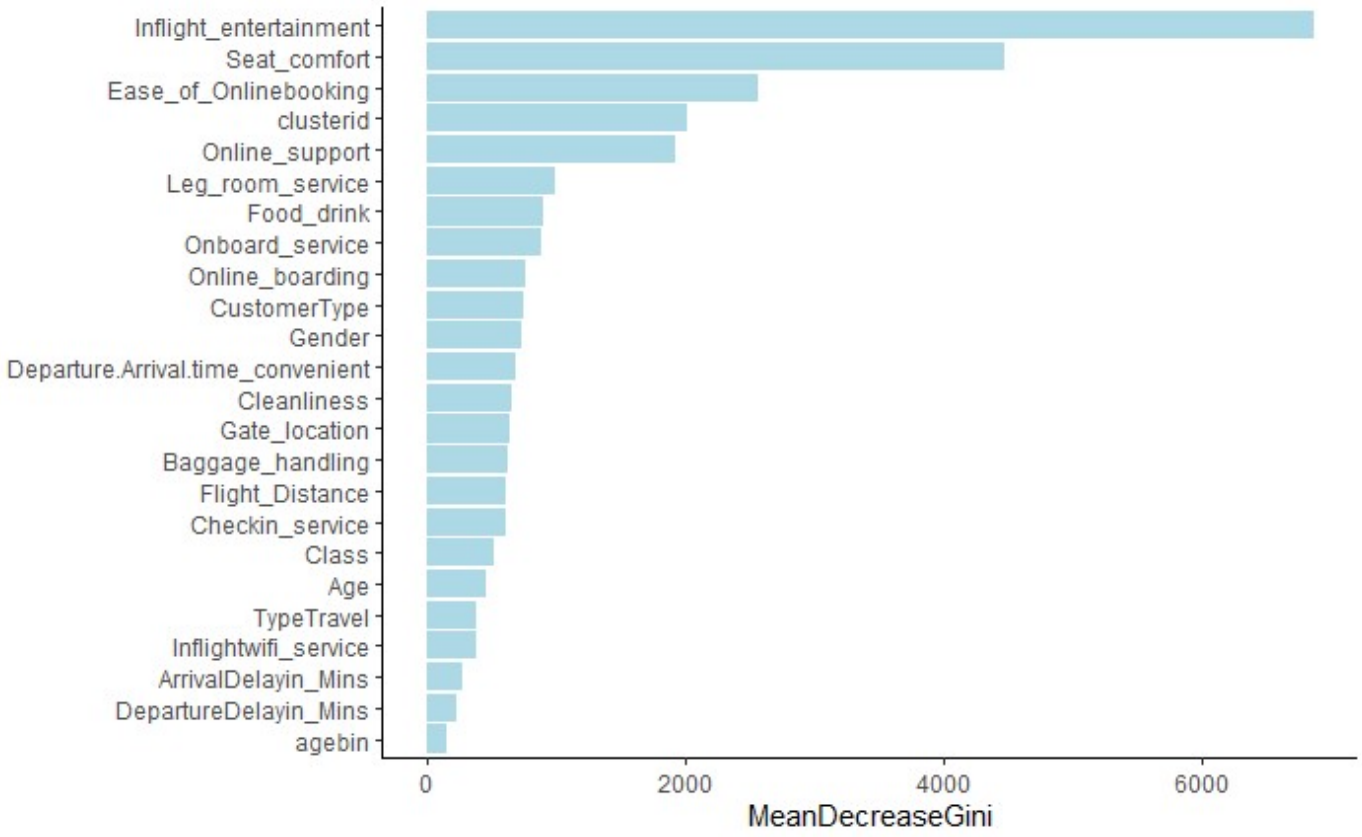
```
a=data.frame(finalfor$importance)
a[order(-a$MeanDecreaseGini),]
```

	dissatisfied <dbl>	satisfied <dbl>	MeanDecreaseAccuracy <dbl>
Inflight_entertainment	0.135097671	0.036742161	0.081270377
Seat_comfort	0.192882393	0.151801625	0.170392372
Ease_of_Onlinebooking	0.054754403	0.044529102	0.049156777
clusterid	0.070091110	0.063869886	0.066684535
Online_support	0.044234598	0.017686783	0.029705434
Leg_room_service	0.017740951	0.037363347	0.028480211
Food_drink	0.030889505	0.011977276	0.020539038
Onboard_service	0.019777756	0.020079932	0.019942897
Online_boarding	0.052995120	0.013682894	0.031484482
CustomerType	0.027689382	0.020981238	0.024019092

1-10 of 24 rows Previous **1** 2 3 Next

Hide

```
a$features=row.names(a)
ggplot(data = a)+aes(x=reorder(features,MeanDecreaseGini),MeanDecreaseGini)+geom_bar(stat =
'identity',fill='light blue')+coord_flip()+theme_classic()+theme(axis.title.y = element_blank
())
```



Hide

```
predicted.class = predict(finalfor,TestData[-c(1,10)],'class')
predicted.prob = predict(finalfor,TestData[-c(1,10)],'prob')
```

Hide

```
#confusion matrix on Training Data
table(TestData$Satisfaction,predicted.class)
```

```
              predicted.class
               dissatisfied satisfied
  dissatisfied        11835       514
  satisfied             740     14187
```

Hide

```
#accuracy on Training Data
sum(diag(table(TestData$Satisfaction,predicted.class)))/nrow(TestData)
```

```
[1] 0.9540255
```

Hide

```
#TNR
table(TestData$Satisfaction,predicted.class)[1,1]/sum(table(TestData$Satisfaction,predicted.class)[1,])
```

```
[1] 0.9583772
```

Hide

```
#TPR
table(TestData$Satisfaction,predicted.class)[2,2]/sum(table(TestData$Satisfaction,predicted.class)[2,])
```

```
[1] 0.9504254
```

Hide

```
#AUC
roc(TestData$Satisfaction,predicted.prob[,2])
```

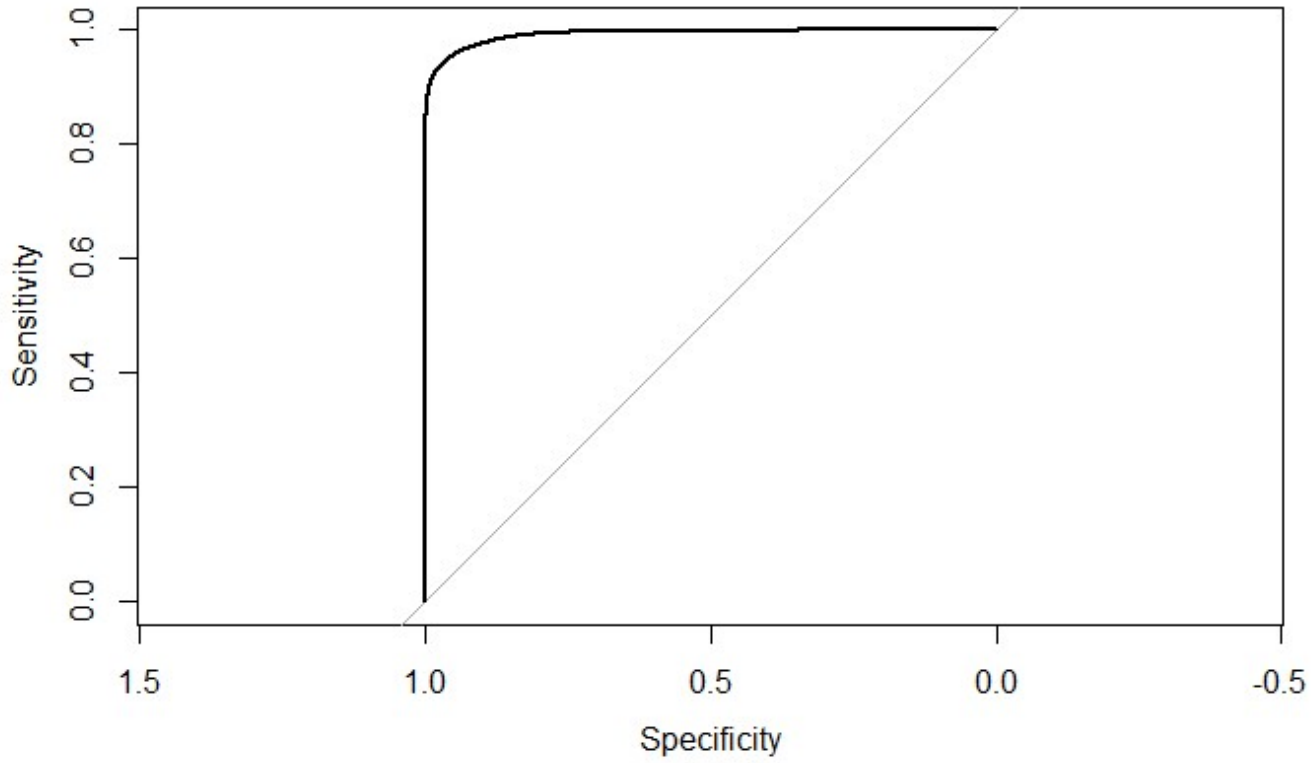
```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```

```

Call:
roc.default(response = TestData$Satisfaction, predictor = predicted.prob[,     2])

Data: predicted.prob[, 2] in 12349 controls (TestData$Satisfaction dissatisfied) < 14927 cases (TestData$Satisfaction satisfied).
Area under the curve: 0.9925
```

Hide

```
plot(roc(TestData$Satisfaction,predicted.prob[,2]))
```

```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```



XGBOOST

Hide

```
library(xgboost)
```

```
package 'xgboost' was built under R version 3.6.1
Attaching package: 'xgboost'

The following object is masked from 'package:dplyr':

    slice

The following object is masked from 'package:rattle':

    xgboost
```

Hide

```
xgbTrain = TrainingData
xgbTest = TestData
```

Hide

```
xgbTrain$Satisfaction=ifelse(xgbTrain$Satisfaction=='satisfied','1','0')
xgbTrain$Satisfaction=as.integer(xgbTrain$Satisfaction)
```

Hide

```
xgbTest$Satisfaction=ifelse(xgbTest$Satisfaction=='satisfied','1','0')
xgbTest$Satisfaction=as.integer(xgbTest$Satisfaction)
```

Trying to determine the appropriate parameter values:

Hide

```
lr = c(.001,.01,0.1,0.3,0.5,0.7,1)
md = c(1,3,5,7,9)
accuracy=c()

for (i in 1:length(md)){
  i
  aviation.xgb.fit = xgboost(
  data = as.matrix(data.matrix(xgbTrain[,-c(1,10,25)])),
  label = as.matrix(data.matrix(xgbTrain[,c(10)])),
  eta = 0.7,#this is like shrinkage in the previous algorithm
  max_depth = md[i],#Larger the depth, more complex the model; higher chances of overfitting.
There is no standard                        value for max_depth. Larger data sets require deep
trees to learn the rules from data.
  min_child_weight = 5,#it blocks the potential feature interactions to prevent overfitting
  nrounds = 501,#controls the maximum number of iterations. For classification, it is similar
to the number of                         trees to grow.
  nfold = 5,
  objective = "binary:logistic",  # for regression models
  verbose = 0,               # silent,
  early_stopping_rounds = 10 # stop if no improvement for 10 consecutive trees
)
predicted.prob=predict(aviation.xgb.fit,as.matrix(data.matrix(xgbTrain[,-c(1,10,25)])))
predicted.class=ifelse(predicted.prob>0.5,'1','0')
accuracy[i]=sum(diag(table(as.factor(xgbTrain$Satisfaction),predicted.class)))/nrow(xgbTrain)
print(accuracy)
}
```

```
[1] 0.9072296
[1] 0.9072296 0.9647083
[1] 0.9072296 0.9647083 0.9990415
[1] 0.9072296 0.9647083 0.9990415 0.9999057
[1] 0.9072296 0.9647083 0.9990415 0.9999057 0.9999529
```

Hide

```
View(accuracy)
```

Final model with correct parameters:

Hide

```
aviation.xgb.fit = xgboost(
  data = as.matrix(data.matrix(xgbTrain[,-c(1,10,25)])),
  label = as.matrix(data.matrix(xgbTrain[,c(10)])),
  eta = 0.1,
  max_depth = 6,
  min_child_weight = 5,
  nrounds = 399,
  nfold = 5,
  objective = "binary:logistic",
  verbose = 0,
  early_stopping_rounds = 10
)
```

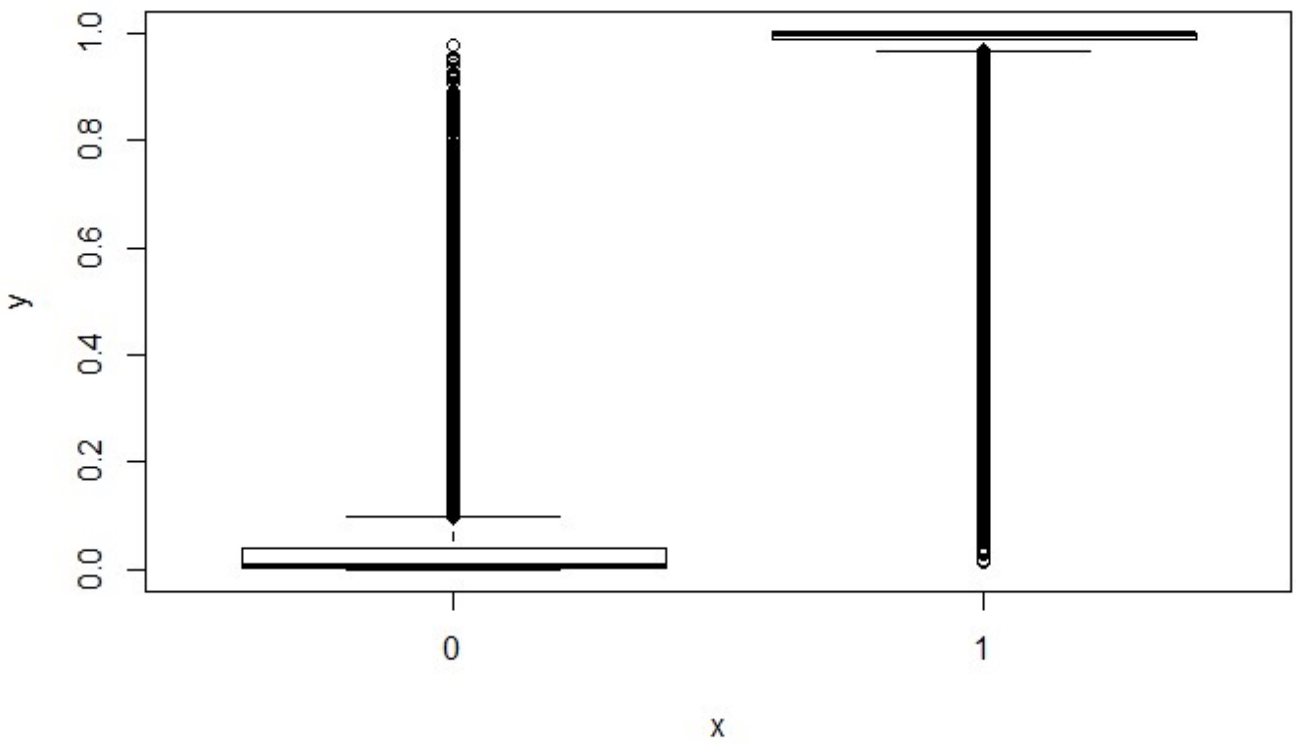
Hide

```
aviation.xgb.fit$best_iteration
```

```
[1] 399
```

Hide

```
predicted.prob=predict(aviation.xgb.fit,as.matrix(data.matrix(xgbTrain[,-c(1,10,25)])))
plot(as.factor(xgbTrain$Satisfaction),predicted.prob)
```



Hide

```
predicted.class=ifelse(predicted.prob>0.5,'1','0')
```

Hide

```
View(predicted.prob)
```

Hide

```
#confusion matrix on Training Data
table(as.factor(xgbTrain$Satisfaction),predicted.class)
```

```
   predicted.class
        0     1
  0 28184   623
  1   888 33946
```

Hide

```
#accuracy on Training Data
sum(diag(table(as.factor(xgbTrain$Satisfaction),predicted.class)))/nrow(xgbTrain)
```

```
[1] 0.9762574
```

Hide

```
#TNR
table(as.factor(xgbTrain$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(xgbTrain$Sat
isfaction),predicted.class)[1,])
```

```
[1] 0.9783733
```

Hide

```
#TPR
table(as.factor(xgbTrain$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(xgbTrain$Sat
isfaction),predicted.class)[2,])
```

```
[1] 0.9745077
```

Hide

```
#AUC
roc(xgbTrain$Satisfaction,predicted.prob)
```

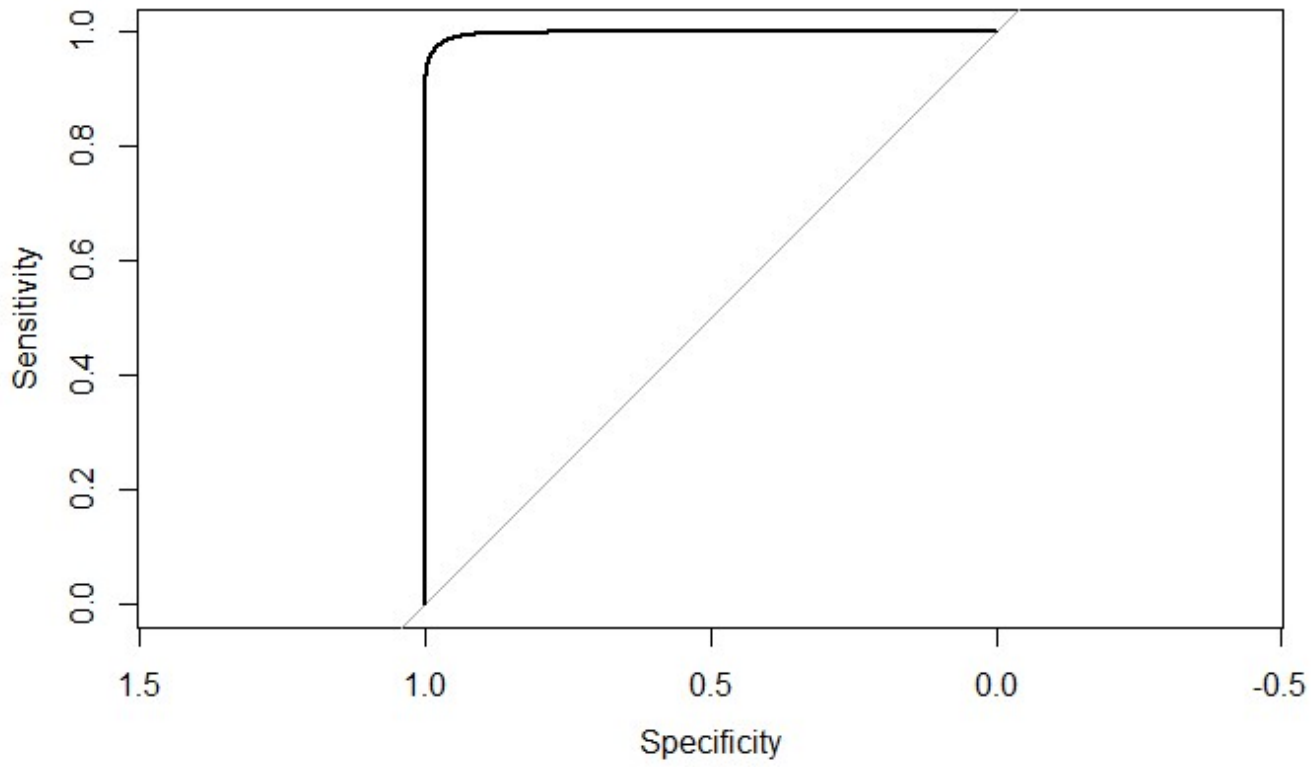
```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

```
Call:
roc.default(response = xgbTrain$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 28807 controls (xgbTrain$Satisfaction 0) < 34834 cases (xgbTrain$Satisfaction 1).
Area under the curve: 0.998
```

Hide

```
plot(roc(xgbTrain$Satisfaction,predicted.prob))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```



Hide

```
predicted.prob=predict(aviation.xgb.fit,as.matrix(data.matrix(xgbTest[,-c(1,10,25)])))
predicted.class=ifelse(predicted.prob>0.5,'1','0')
```

Hide

```
#confusion matrix on Test Data
table(as.factor(xgbTest$Satisfaction),predicted.class)
```

```
   predicted.class
        0     1
  0 11779   570
  1   683 14244
```

Hide

```
#accuracy on Test Data
sum(diag(table(as.factor(xgbTest$Satisfaction),predicted.class)))/nrow(xgbTest)
```

```
[1] 0.9540622
```

Hide

```
#TNR
table(as.factor(xgbTest$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(xgbTest$Satis
faction),predicted.class)[1,])
```

```
[1] 0.9538424
```

Hide

```
#TPR
table(as.factor(xgbTest$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(xgbTest$Satis
faction),predicted.class)[2,])
```

```
[1] 0.954244
```

Hide

```
#AUC
roc(xgbTest$Satisfaction,predicted.prob)
```

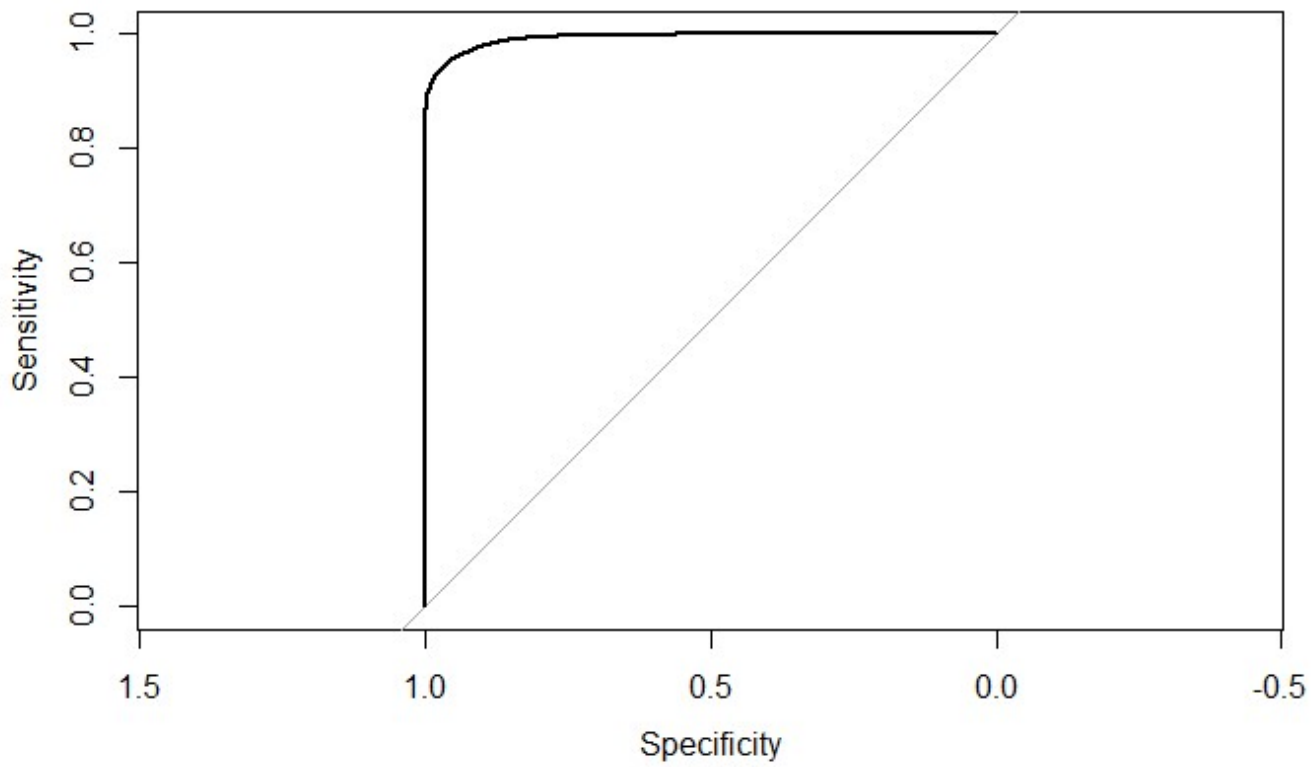
```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

```

Call:
roc.default(response = xgbTest$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 12349 controls (xgbTest$Satisfaction 0) < 14927 cases (xgbTest$Satisf
action 1).
Area under the curve: 0.9931
```

Hide

```
plot(roc(xgbTest$Satisfaction,predicted.prob))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```



Hide

```
xgb.importance(model = aviation.xgb.fit)
```

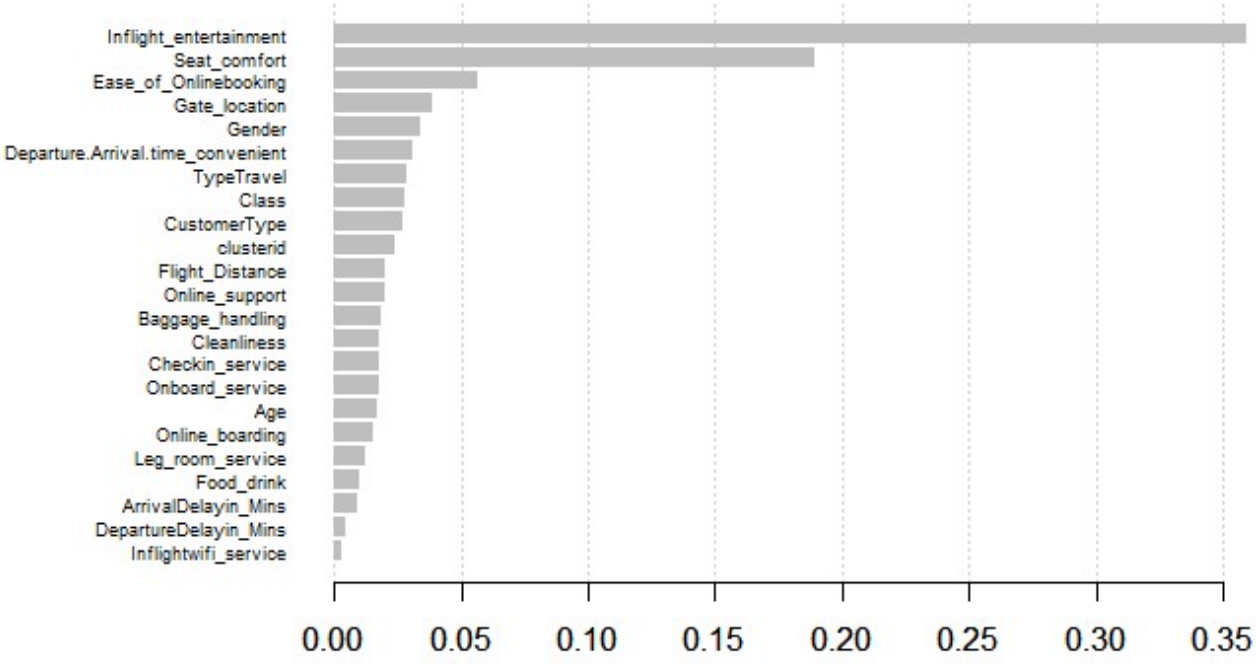
Feature <chr>	Gain <dbl>	Cover <dbl>	Frequency <dbl>
Inflight_entertainment	0.359229579	0.089607344	0.04427638
Seat_comfort	0.189542081	0.096147427	0.08112754
Ease_of_Onlinebooking	0.056713914	0.039243993	0.03932623
Gate_location	0.038463920	0.035107594	0.04743898
Gender	0.034111589	0.015007922	0.02041939
Departure.Arrival.time_convenient	0.031003753	0.033188141	0.04455139
TypeTravel	0.028375594	0.022592340	0.03169474
Class	0.027455681	0.020390422	0.02942592

Feature <chr>	Gain <dbl>	Cover <dbl>	Frequency <dbl>
CustomerType	0.027101400	0.024588190	0.01595050
clusterid	0.024288276	0.023701149	0.03506360

1-10 of 23 rows Previous **1** 2 3 Next

Hide

```
xgb.plot.importance(xgb.importance(model = aviation.xgb.fit))
```



xgboost model for each cluster:

Cluster 1

Hide

```
Clust1_TrngData$Satisfaction=ifelse(Clust1_TrngData$Satisfaction=='satisfied','1','0')
Clust1_TrngData$Satisfaction=as.integer(Clust1_TrngData$Satisfaction)
```

Hide

```
Clust1_TestData$Satisfaction=ifelse(Clust1_TestData$Satisfaction=='satisfied','1','0')
Clust1_TestData$Satisfaction=as.integer(Clust1_TestData$Satisfaction)
```

Hide

```
clust1.xgb.fit = xgboost(
  data = as.matrix(data.matrix(Clust1_TrngData[,-c(1,10,25,26)])),
  label = as.matrix(data.matrix(Clust1_TrngData[,c(10)])),
  eta = 0.3,#this is like shrinkage in the previous algorithm
  max_depth = 5,#Larger the depth, more complex the model; higher chances of overfitting. The
re is no standard                       value for max_depth. Larger data sets require deep tre
es to learn the rules from data.
  min_child_weight = 5,#it blocks the potential feature interactions to prevent overfitting
  nrounds = 154,#controls the maximum number of iterations. For classification, it is similar
to the number of                        trees to grow.
  nfold = 5,
  objective = "binary:logistic",  # for regression models
  verbose = 0,               # silent,
  early_stopping_rounds = 10 # stop if no improvement for 10 consecutive trees
)
```

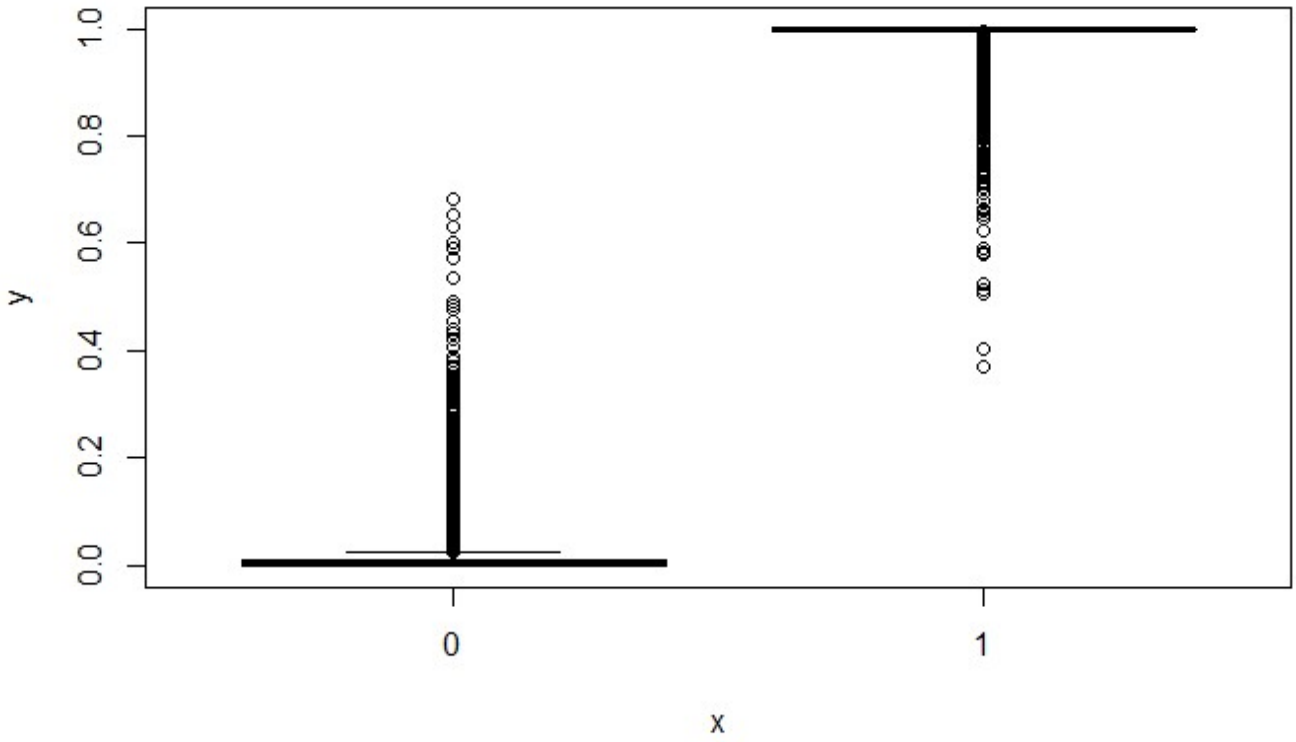
Hide

```
clust1.xgb.fit$best_iteration
```

```
[1] 154
```

Hide

```
predicted.prob=predict(clust1.xgb.fit,as.matrix(data.matrix(Clust1_TrngData[,-c(1,10,25,2
6)])))
plot(as.factor(Clust1_TrngData$Satisfaction),predicted.prob)
```



Hide

```
predicted.class=ifelse(predicted.prob>0.5,'1','0')
```

Hide

```
#confusion matrix on Training Data
table(as.factor(Clust1_TrngData$Satisfaction),predicted.class)
```

```
   predicted.class
       0    1
  0 2674    7
  1    2 9574
```

Hide

```
#accuracy on Training Data
sum(diag(table(as.factor(Clust1_TrngData$Satisfaction),predicted.class)))/nrow(Clust1_TrngDat
a)
```

```
[1] 0.9992657
```

Hide

```
#TNR
table(as.factor(Clust1_TrngData$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(Clust
1_TrngData$Satisfaction),predicted.class)[1,])
```

```
[1] 0.997389
```

Hide

```
#TPR
table(as.factor(Clust1_TrngData$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(Clust
1_TrngData$Satisfaction),predicted.class)[2,])
```

```
[1] 0.9997911
```

Hide

```
#AUC
roc(Clust1_TrngData$Satisfaction,predicted.prob)
```

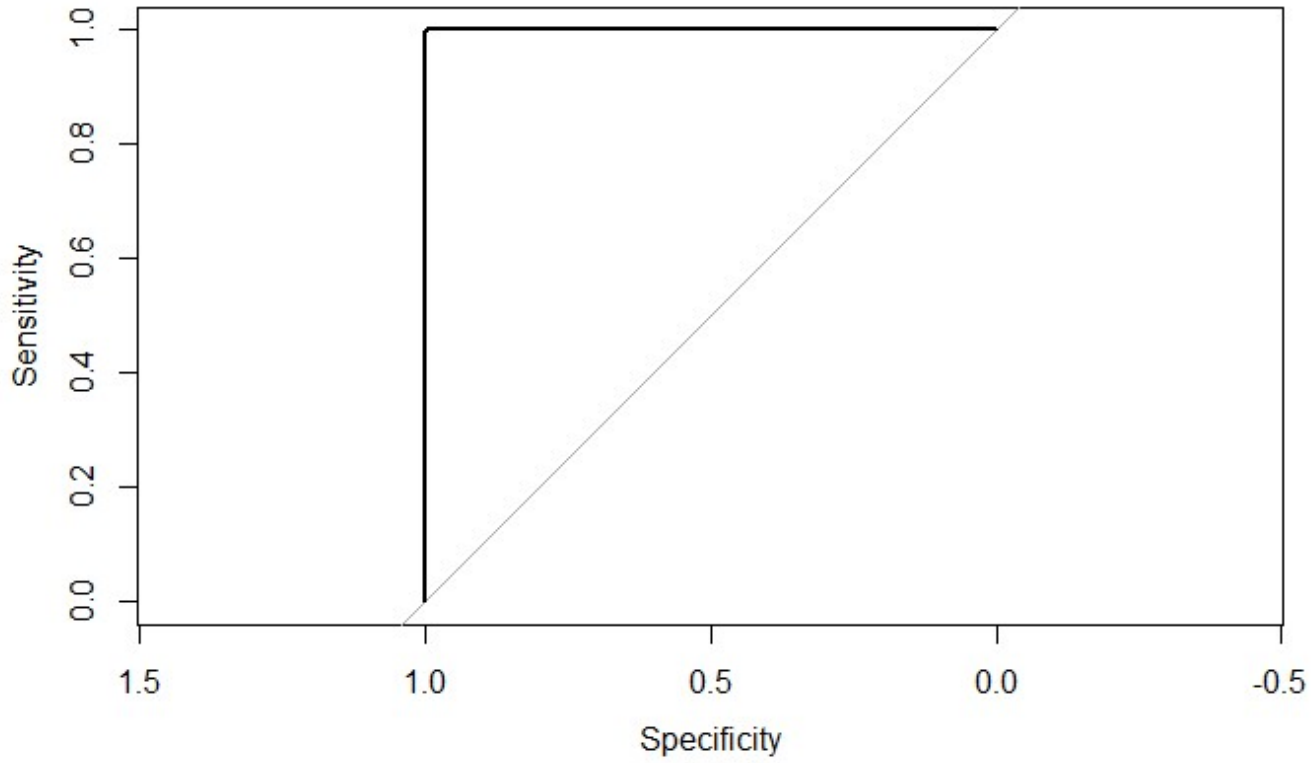
```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

```

Call:
roc.default(response = Clust1_TrngData$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 2681 controls (Clust1_TrngData$Satisfaction 0) < 9576 cases (Clust1_T
rngData$Satisfaction 1).
Area under the curve: 1
```

Hide

```
plot(roc(Clust1_TrngData$Satisfaction,predicted.prob))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```



Hide

```
predicted.prob=predict(clust1.xgb.fit,as.matrix(data.matrix(Clust1_TestData[,-c(1,10,25,2
6)])))
predicted.class=ifelse(predicted.prob>0.5,'1','0')
```

Hide

```
#confusion matrix on Test Data
table(as.factor(Clust1_TestData$Satisfaction),predicted.class)
```

```
   predicted.class
      0    1
  0 1105   44
  1   37 4068
```

Hide

```
#accuracy on Test Data
sum(diag(table(as.factor(Clust1_TestData$Satisfaction),predicted.class)))/nrow(Clust1_TestDat
a)
```

```
[1] 0.9845832
```

Hide

```
#TNR
table(as.factor(Clust1_TestData$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(Clust
1_TestData$Satisfaction),predicted.class)[1,])
```

```
[1] 0.9617058
```

Hide

```
#TPR
table(as.factor(Clust1_TestData$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(Clust
1_TestData$Satisfaction),predicted.class)[2,])
```

```
[1] 0.9909866
```

Hide

```
#AUC
roc(Clust1_TestData$Satisfaction,predicted.prob)
```

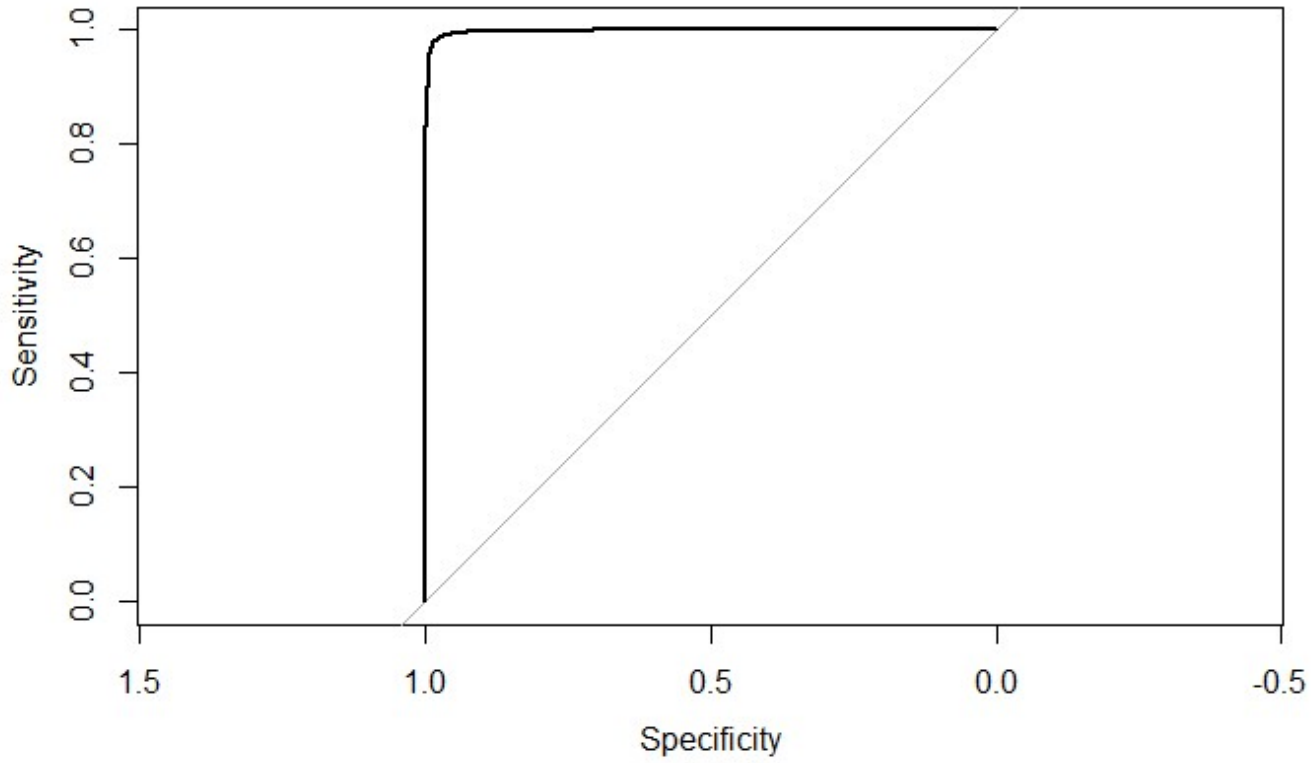
```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

```

Call:
roc.default(response = Clust1_TestData$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 1149 controls (Clust1_TestData$Satisfaction 0) < 4105 cases (Clust1_T
estData$Satisfaction 1).
Area under the curve: 0.9975
```

Hide

```
plot(roc(Clust1_TestData$Satisfaction,predicted.prob))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```



Hide

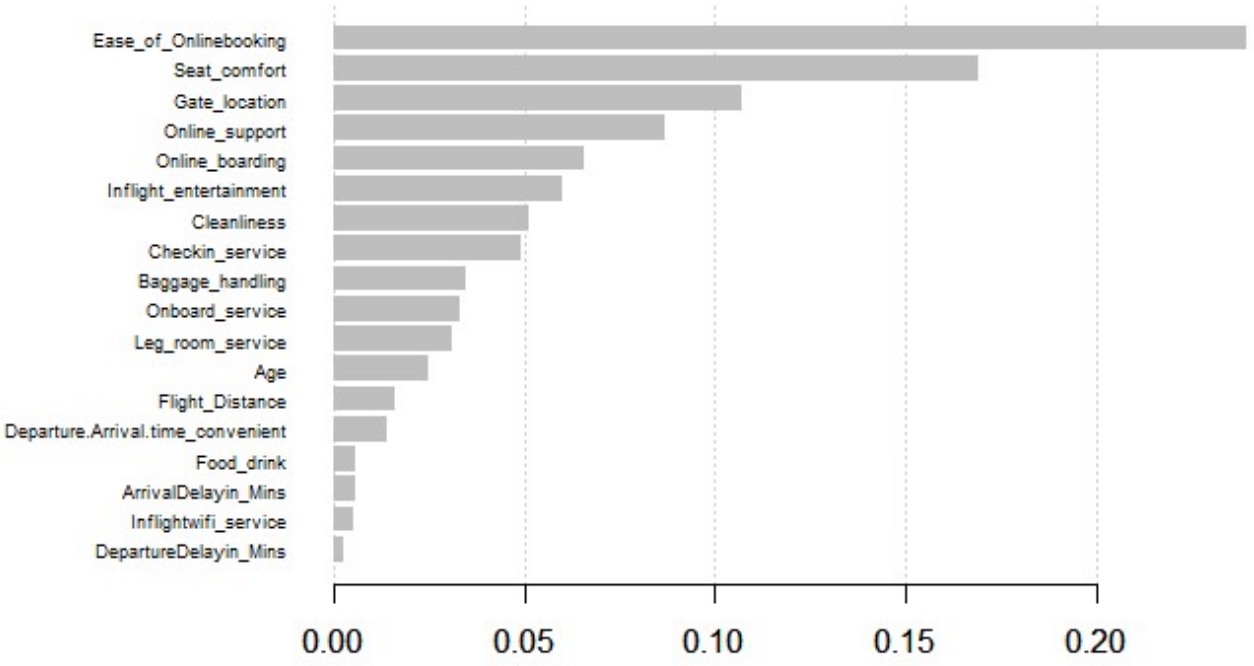
```
xgb.importance(model = clust1.xgb.fit)
```

Feature <chr>	Gain <dbl>	Cover <dbl>	Frequency <dbl>
Ease_of_Onlinebooking	0.239620649	0.08838184	0.04273084
Seat_comfort	0.169364303	0.09381111	0.10609037
Gate_location	0.106936512	0.05091308	0.05402750
Online_support	0.086954793	0.07753154	0.04518664
Online_boarding	0.065466165	0.06041798	0.03880157
Inflight_entertainment	0.059909672	0.12050833	0.05206287
Cleanliness	0.051102087	0.05429465	0.03438114
Checkin_service	0.048852953	0.06791981	0.04076621
Baggage_handling	0.034858927	0.04160836	0.05108055
Onboard_service	0.032913487	0.04296282	0.04027505

1-10 of 18 rows Previous **1** 2 Next

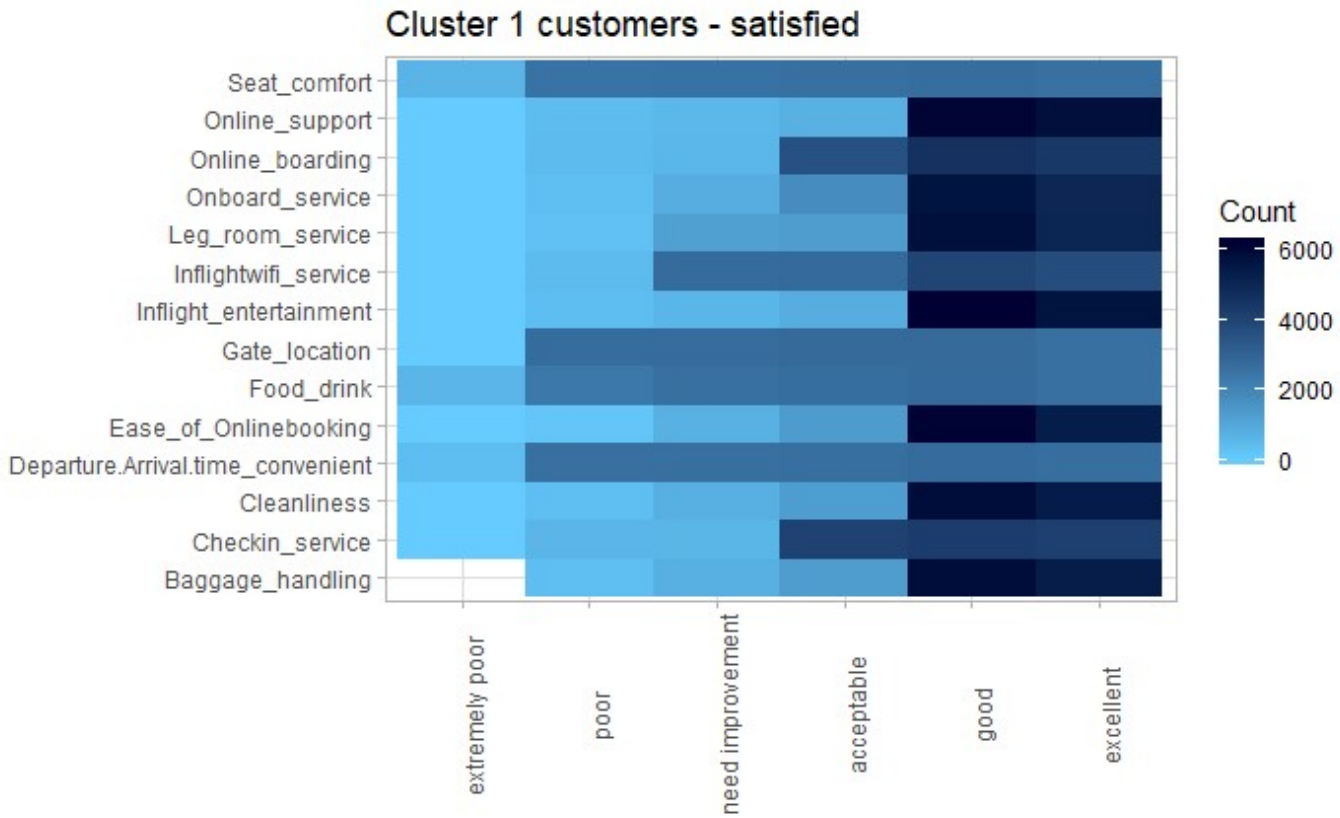
Hide

```
xgb.plot.importance(xgb.importance(model = clust1.xgb.fit))
```



Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$clusterid=='1' & Cleansed.Data$Satisfaction=
='satisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Cluster 1 customers - satisfied')
```



Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$clusterid=='1' & Cleansed.Data$Satisfaction=
='dissatisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Cluster 1 customers - dissatisfied')
```

Random forest model to study the variable importance

Hide

```
Clust1_TrngData$Satisfaction=ifelse(Clust1_TrngData$Satisfaction==1,'satisfied','dissatisfie
d')
Clust1_TrngData$Satisfaction=as.factor(Clust1_TrngData$Satisfaction)
```

Hide

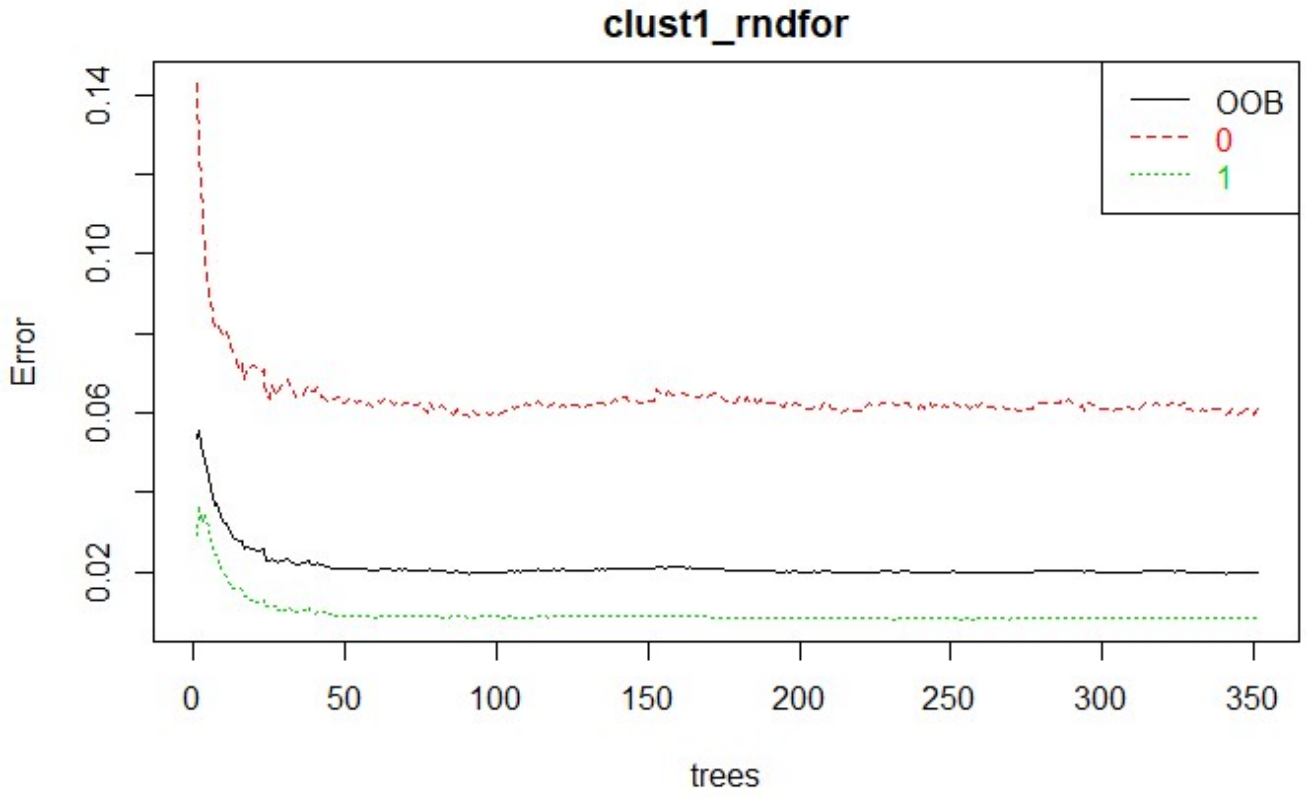
```
Clust1_TestData$Satisfaction=ifelse(Clust1_TestData$Satisfaction==1,'satisfied','dissatisfie
d')
Clust1_TestData$Satisfaction=as.factor(Clust1_TestData$Satisfaction)
```

Hide

```
set.seed(1000)
clust1_rndfor = randomForest(Satisfaction~.,data = Clust1_TrngData[-c(1,25,26)],ntree=99,mtry
=6,nodesize=10,importance=T)
```

Hide

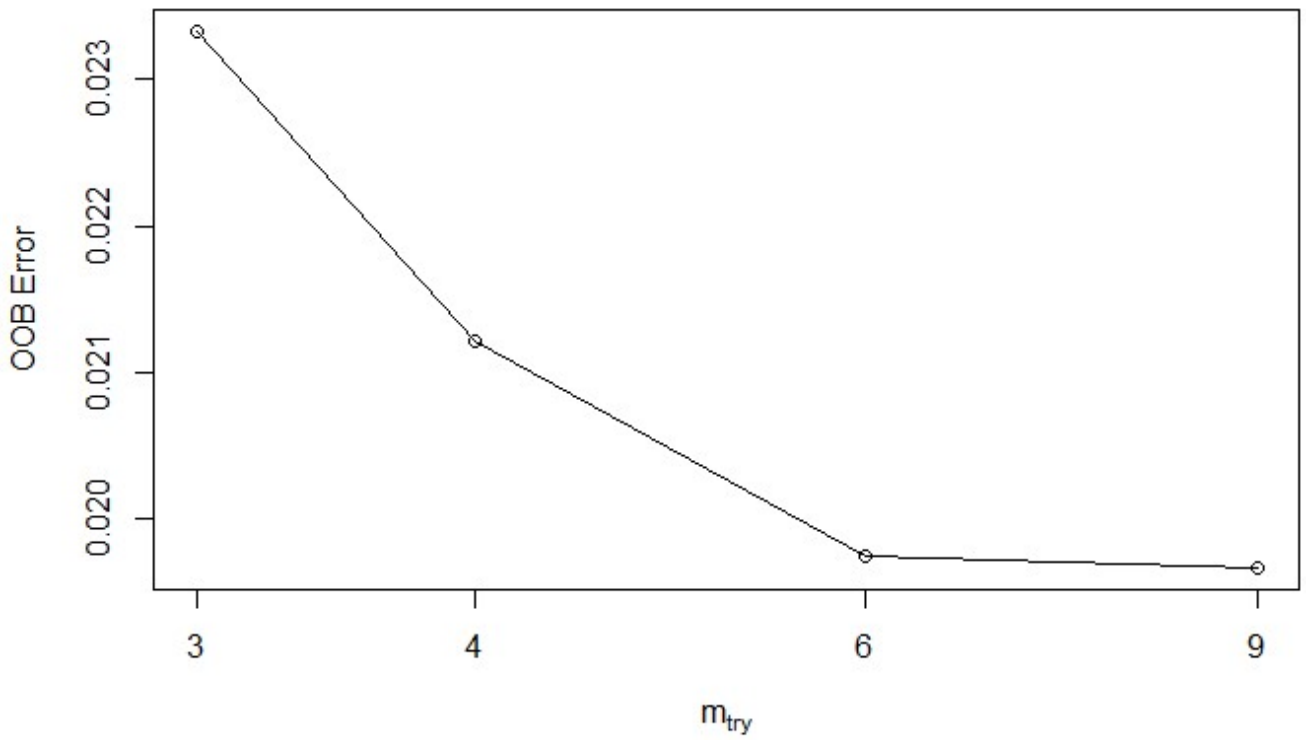
```
plot(clust1_rndfor)
legend("topright", c("OOB", "0", "1"), text.col=1:6, lty=1:3, col=1:3)
```



Hide

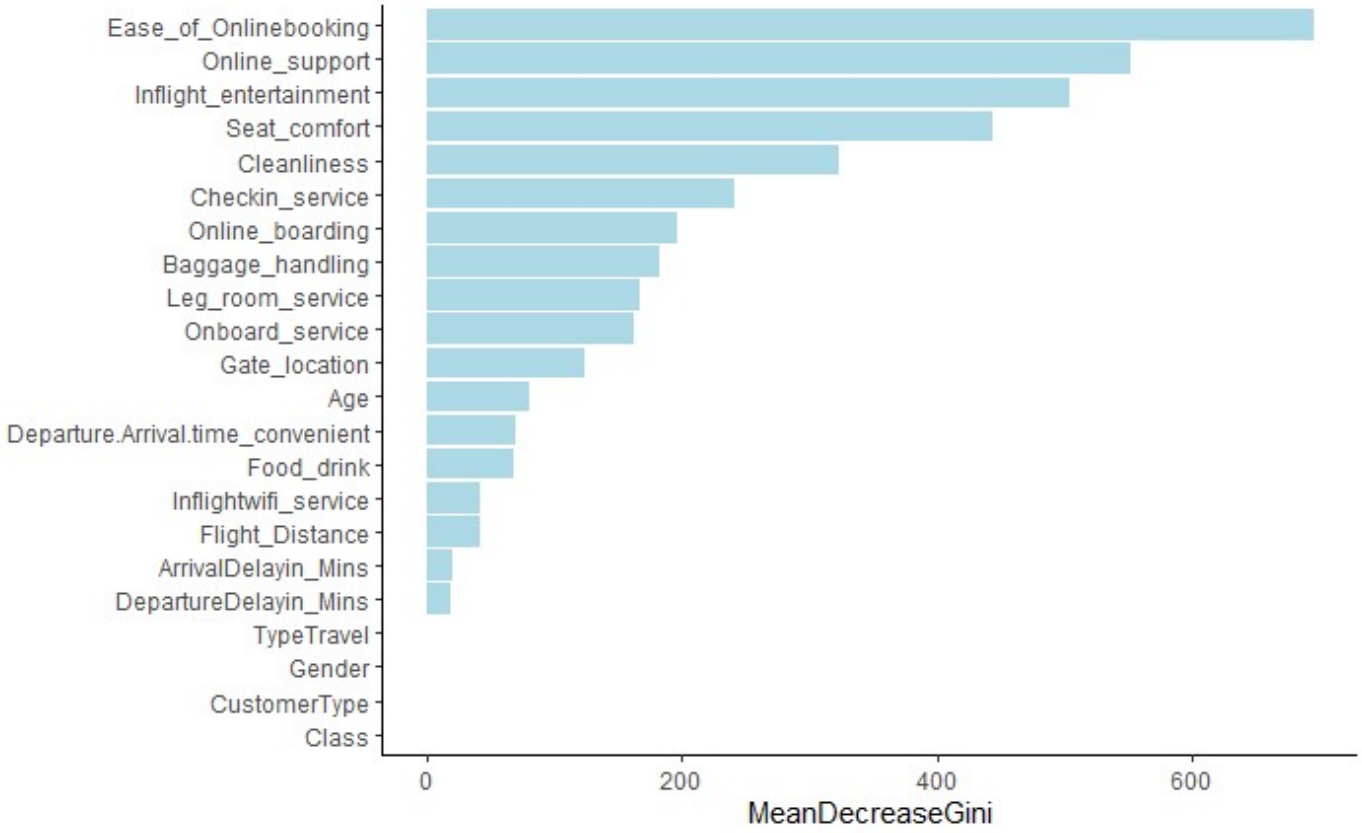
```
trndfor = tuneRF(x=Clust1_TrngData[-c(1,10,25,26)],
                 y=Clust1_TrngData$Satisfaction,
                 mtryStart = 4,
                 stepFactor = 1.5,
                 ntreeTry = 99,
                 trace = T,
                 plot = T,
                 importance=T,
                 doBest = T)
```

```
mtry = 4  OOB error = 2.12%
Searching left ...
mtry = 3    OOB error = 2.33%
-0.1 0.05
Searching right ...
mtry = 6    OOB error = 1.97%
0.06923077 0.05
mtry = 9    OOB error = 1.97%
0.004132231 0.05
```



Hide

```
a=as.data.frame(clust1_rndfor$importance)
a$features=row.names(a)
ggplot(data = a)+aes(x=reorder(features,MeanDecreaseGini),MeanDecreaseGini)+geom_bar(stat =
'identity',fill='light blue')+coord_flip()+theme_classic()+theme(axis.title.y = element_blank
())
```



Hide

```
predicted.class = predict(clust1_rndfor,Clust1_TestData[-c(1,10)],'class')
predicted.prob = predict(clust1_rndfor,Clust1_TestData[-c(1,10)],'prob')
```

Hide

```
#confusion matrix on Training Data
table(Clust1_TestData$Satisfaction,predicted.class)
```

```
              predicted.class
               dissatisfied satisfied
  dissatisfied         1087        62
  satisfied              36      4069
```

Hide

```
#accuracy on Training Data
sum(diag(table(Clust1_TestData$Satisfaction,predicted.class)))/nrow(Clust1_TestData)
```

```
[1] 0.9813475
```

Hide

```
#TNR
table(Clust1_TestData$Satisfaction,predicted.class)[1,1]/sum(table(Clust1_TestData$Satisfacti
on,predicted.class)[1,])
```

```
[1] 0.94604
```

Hide

```
#TPR
table(Clust1_TestData$Satisfaction,predicted.class)[2,2]/sum(table(Clust1_TestData$Satisfacti
on,predicted.class)[2,])
```

```
[1] 0.9912302
```

Hide

```
#AUC
roc(Clust1_TestData$Satisfaction,predicted.prob[,2])
```

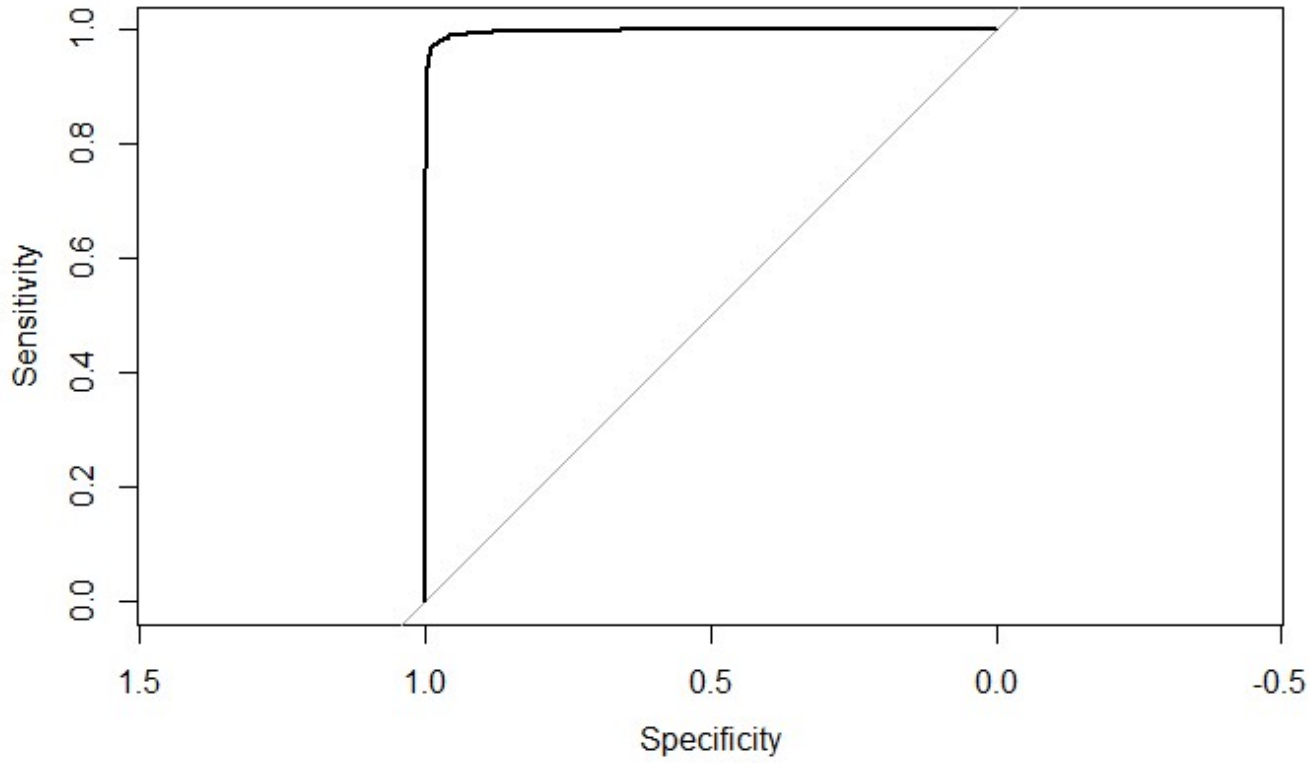
```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```

```

Call:
roc.default(response = Clust1_TestData$Satisfaction, predictor = predicted.prob[,     2])

Data: predicted.prob[, 2] in 1149 controls (Clust1_TestData$Satisfaction dissatisfied) < 4105
cases (Clust1_TestData$Satisfaction satisfied).
Area under the curve: 0.9964
```

Hide

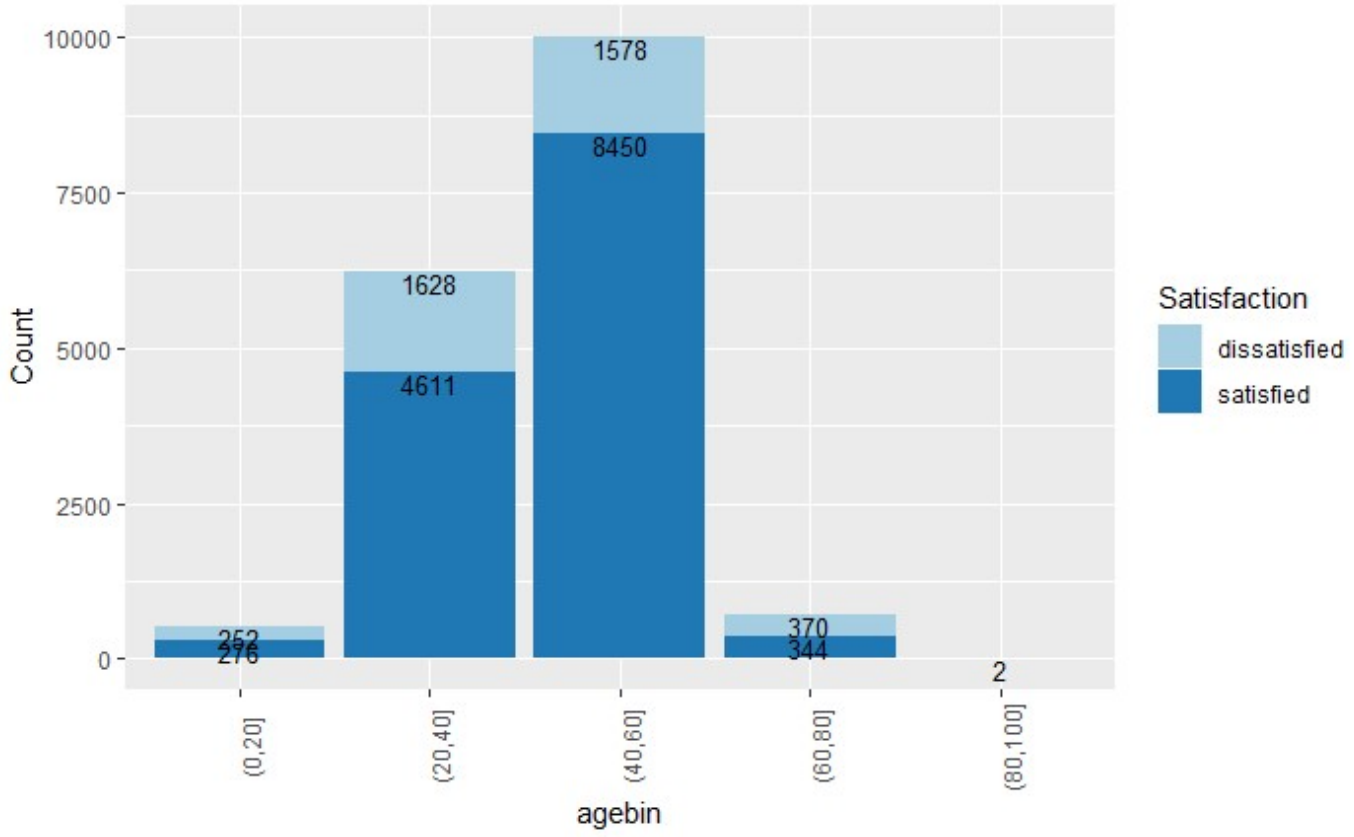
```
plot(roc(Clust1_TestData$Satisfaction,predicted.prob[,2]))
```

```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```



Hide

```
ggplot(data = Cleansed.Data[Cleansed.Data$clusterid=='1',])+aes(x=agebin,fill=Satisfaction,y
=..count..)+geom_bar(position = 'stack')+labs(y='Count')+geom_text(aes(label=..count..),stat
= 'count',position='stack', vjust=1,size=3.5)+scale_fill_brewer(palette="Paired")+theme(axis.
text.x=element_text(angle = 90))
```



Cluster 2

Hide

```
Clust2_TrngData$Satisfaction=ifelse(Clust2_TrngData$Satisfaction=='satisfied','1','0')
Clust2_TrngData$Satisfaction=as.integer(Clust2_TrngData$Satisfaction)
```

Hide

```
Clust2_TestData$Satisfaction=ifelse(Clust2_TestData$Satisfaction=='satisfied','1','0')
Clust2_TestData$Satisfaction=as.integer(Clust2_TestData$Satisfaction)
```

Hide

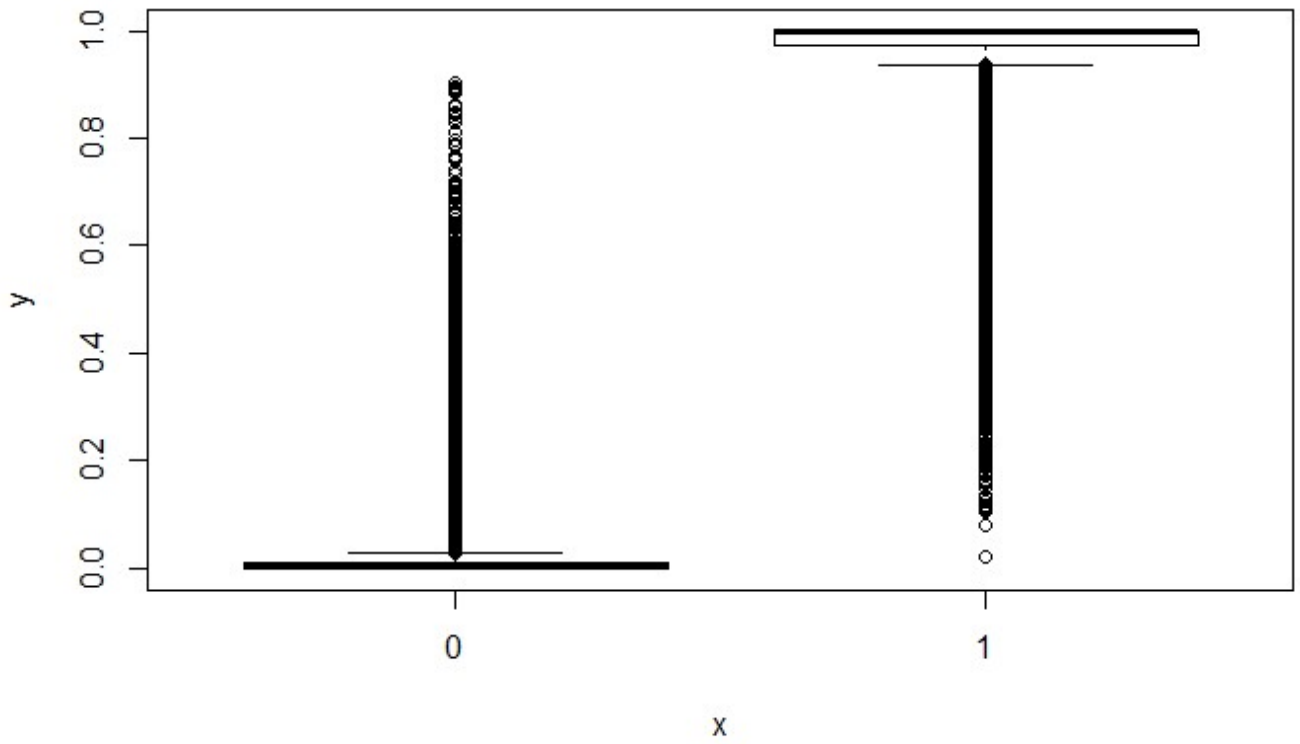
```
clust2.xgb.fit = xgboost(
  data = as.matrix(data.matrix(Clust2_TrngData[,-c(1,10,25,26)])),
  label = as.matrix(data.matrix(Clust2_TrngData[,c(10)])),
  eta = 0.3,#this is like shrinkage in the previous algorithm
  max_depth = 5,#Larger the depth, more complex the model; higher chances of overfitting. The
re is no standard                       value for max_depth. Larger data sets require deep tre
es to learn the rules from data.
  min_child_weight = 5,#it blocks the potential feature interactions to prevent overfitting
  nrounds = 154,#controls the maximum number of iterations. For classification, it is similar
to the number of                        trees to grow.
  nfold = 5,
  objective = "binary:logistic",  # for regression models
  verbose = 0,               # silent,
  early_stopping_rounds = 10 # stop if no improvement for 10 consecutive trees
)
```

Hide

```
clust2.xgb.fit$best_iteration
```

Hide

```
predicted.prob=predict(clust2.xgb.fit,as.matrix(data.matrix(Clust2_TrngData[,-c(1,10,25,2
6)])))
plot(as.factor(Clust2_TrngData$Satisfaction),predicted.prob)
```



Hide

```
predicted.class=ifelse(predicted.prob>0.5,'1','0')
```

Hide

```
#confusion matrix on Training Data
table(as.factor(Clust2_TrngData$Satisfaction),predicted.class)
```

```
   predicted.class
       0    1
  0 9672  131
  1  302 8162
```

Hide

```
#accuracy on Training Data
sum(diag(table(as.factor(Clust2_TrngData$Satisfaction),predicted.class)))/nrow(Clust2_TrngDat
a)
```

```
[1] 0.9762961
```

Hide

```
#TNR
table(as.factor(Clust2_TrngData$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(Clust
2_TrngData$Satisfaction),predicted.class)[1,])
```

```
[1] 0.9866367
```

Hide

```
#TPR
table(as.factor(Clust2_TrngData$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(Clust
2_TrngData$Satisfaction),predicted.class)[2,])
```

```
[1] 0.9643195
```

Hide

```
#AUC
roc(Clust2_TrngData$Satisfaction,predicted.prob)
```

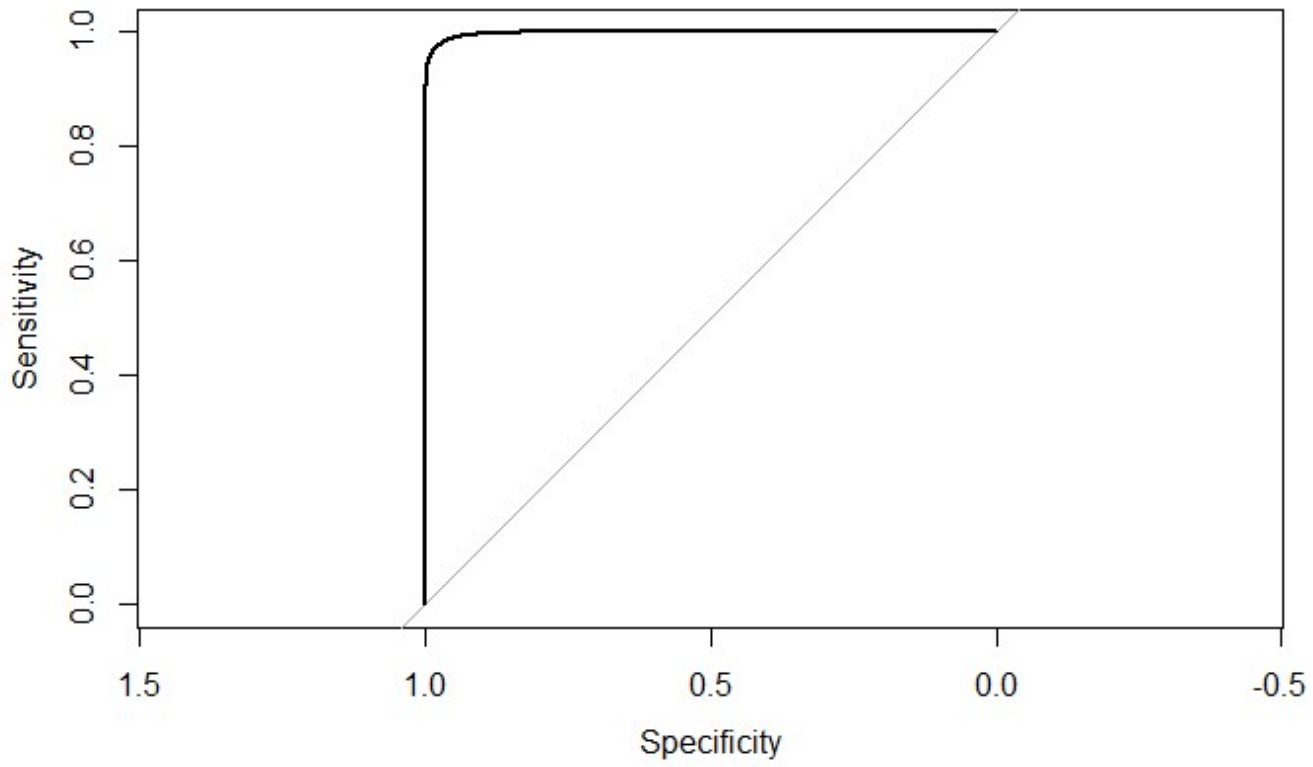
```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

```
Call:
roc.default(response = Clust2_TrngData$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 9803 controls (Clust2_TrngData$Satisfaction 0) < 8464 cases (Clust2_T
rngData$Satisfaction 1).
Area under the curve: 0.998
```

Hide

```
plot(roc(Clust2_TrngData$Satisfaction,predicted.prob))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```



Hide

```
predicted.prob=predict(clust2.xgb.fit,as.matrix(data.matrix(Clust2_TestData[,-c(1,10,25,2
6)])))
predicted.class=ifelse(predicted.prob>0.5,'1','0')
```

Hide

```
#confusion matrix on Test Data
table(as.factor(Clust2_TestData$Satisfaction),predicted.class)
```

```
   predicted.class
       0    1
  0 3959  211
  1  279 3381
```

Hide

```
#accuracy on Test Data
sum(diag(table(as.factor(Clust2_TestData$Satisfaction),predicted.class)))/nrow(Clust2_TestDat
a)
```

```
[1] 0.9374202
```

Hide

```
#TNR
table(as.factor(Clust2_TestData$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(Clust
2_TestData$Satisfaction),predicted.class)[1,])
```

```
[1] 0.9494005
```

Hide

```
#TPR
table(as.factor(Clust2_TestData$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(Clust
2_TestData$Satisfaction),predicted.class)[2,])
```

```
[1] 0.9237705
```

Hide

```
#AUC
roc(Clust2_TestData$Satisfaction,predicted.prob)
```

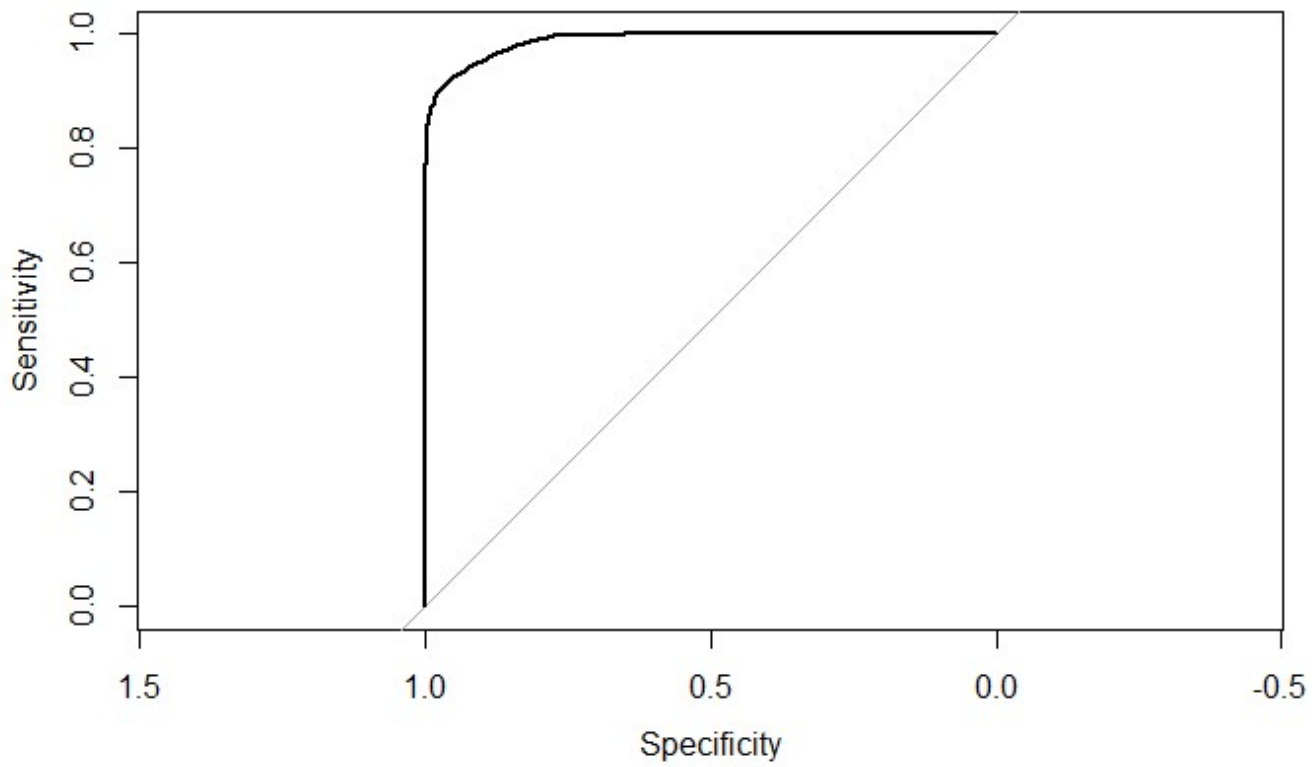
```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

```

Call:
roc.default(response = Clust2_TestData$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 4170 controls (Clust2_TestData$Satisfaction 0) < 3660 cases (Clust2_T
estData$Satisfaction 1).
Area under the curve: 0.9881
```

Hide

```
plot(roc(Clust2_TestData$Satisfaction,predicted.prob))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```



Hide

```
xgb.importance(model = clust2.xgb.fit)
```

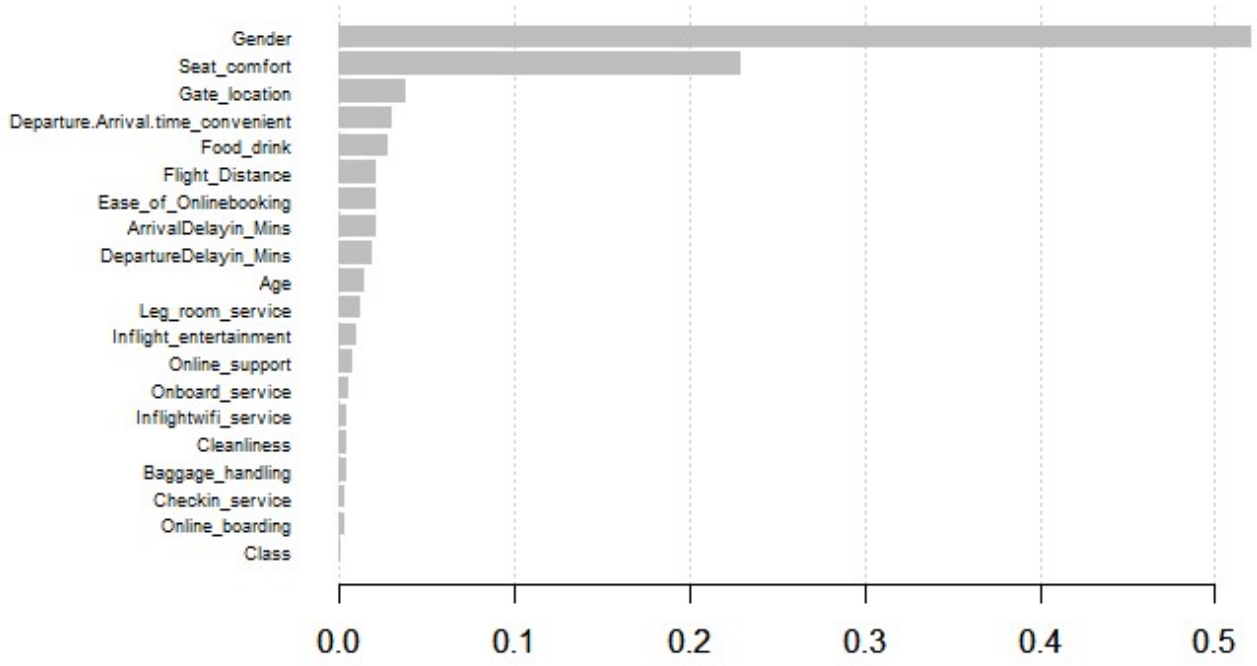
Feature <chr>	Gain <dbl>	Cover <dbl>	Frequency <dbl>
Gender	0.5211753058	0.051714467	0.01951017
Seat_comfort	0.2289885379	0.136036464	0.06890826
Gate_location	0.0377060544	0.036121359	0.05437941
Departure.Arrival.time_convenient	0.0306194743	0.031394241	0.04856787
Food_drink	0.0274356708	0.033229023	0.03569946
Flight_Distance	0.0216577689	0.231675371	0.20838522
Ease_of_Onlinebooking	0.0215954137	0.025261911	0.03279369
ArrivalDelayin_Mins	0.0206808176	0.079405074	0.05770029

Feature <chr>	Gain <dbl>	Cover <dbl>	Frequency <dbl>
DepartureDelayin_Mins	0.0187125189	0.095728026	0.06973848
Age	0.0149275484	0.081836687	0.10668327

1-10 of 20 rows Previous **1** 2 Next

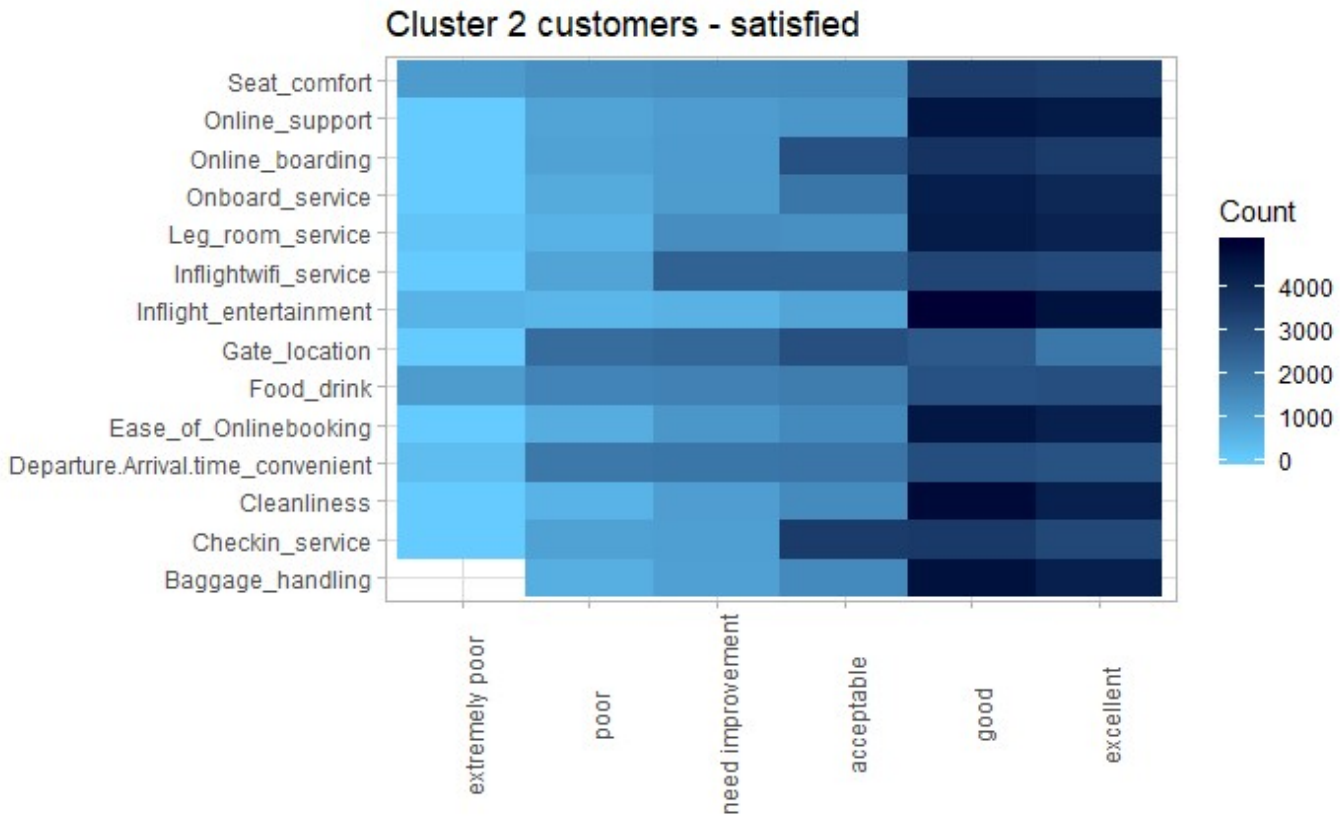
Hide

```
xgb.plot.importance(xgb.importance(model = clust2.xgb.fit))
```



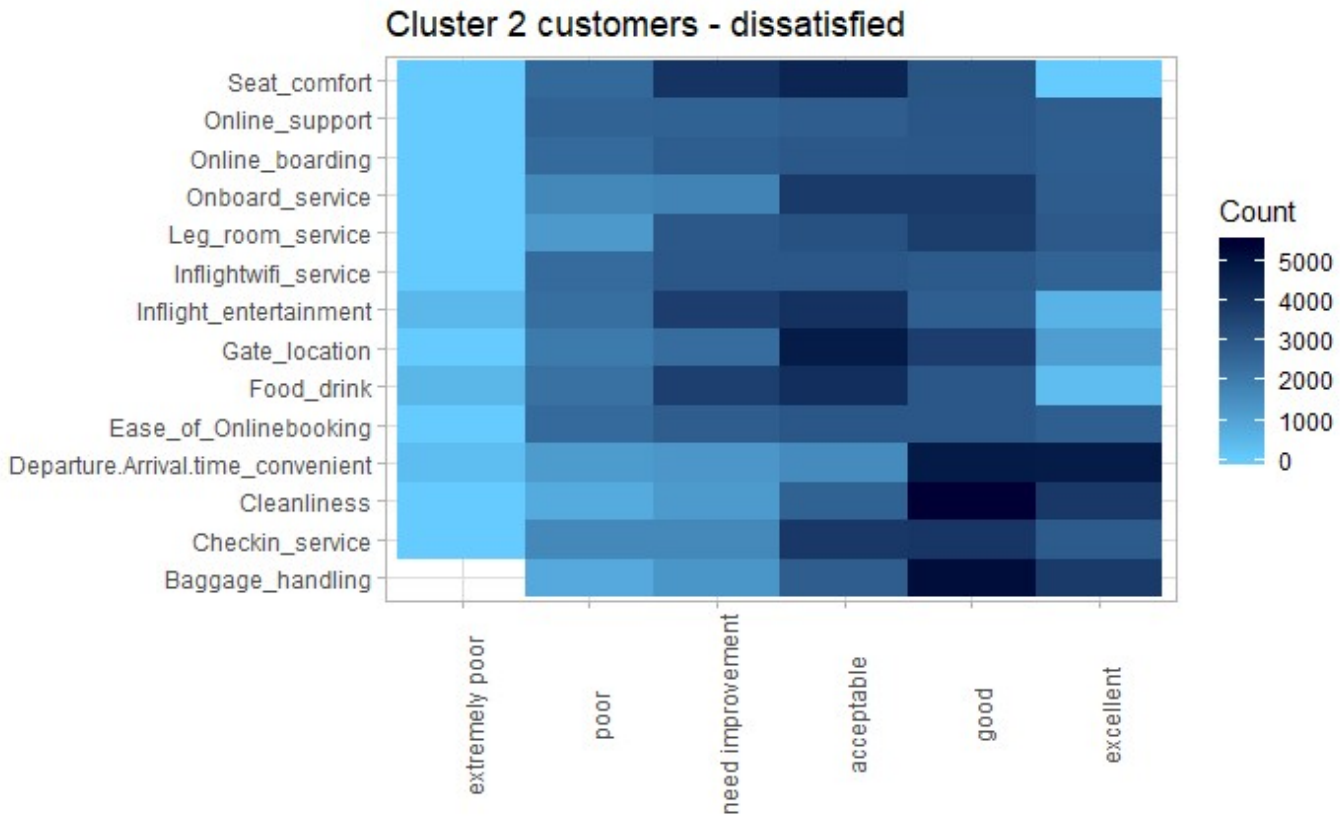
Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$clusterid=='2' & Cleansed.Data$Satisfaction=
='satisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Cluster 2 customers - satisfied')
```



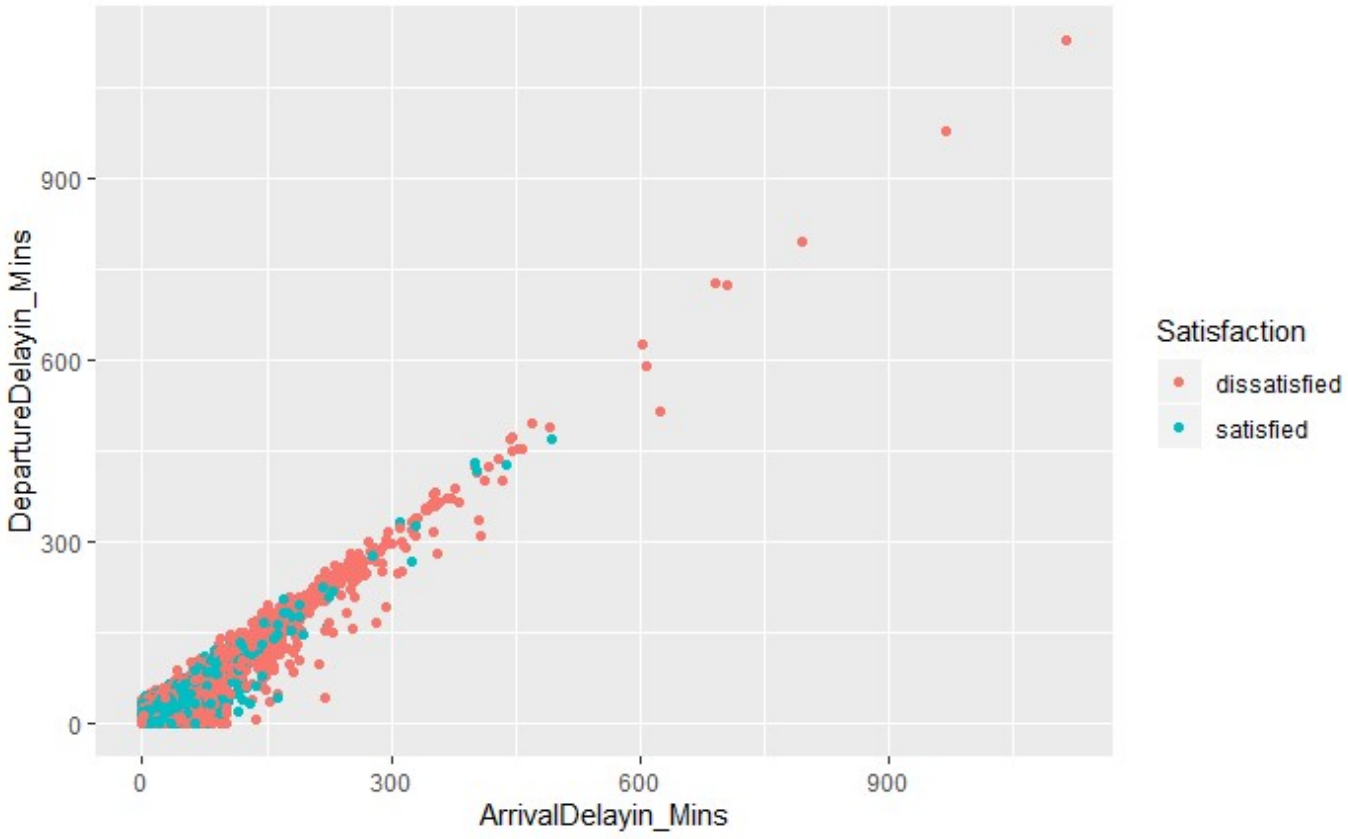
Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$clusterid=='2' & Cleansed.Data$Satisfaction=
='dissatisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Cluster 2 customers - dissatisfied')
```



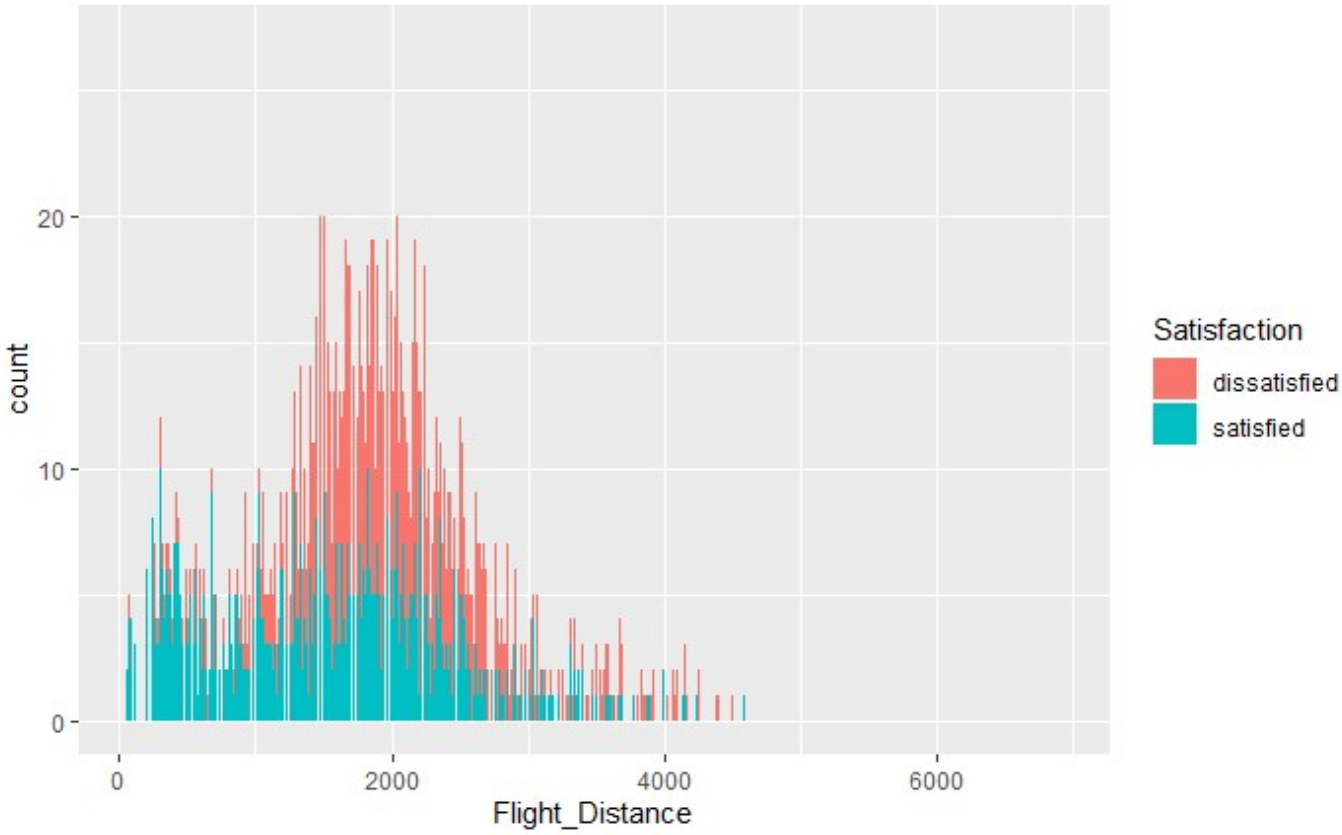
Hide

```
ggplot(data = Cleansed.Data[Cleansed.Data$clusterid=='2',])+aes(x=ArrivalDelayin_Mins,y=Depar
tureDelayin_Mins,color=Satisfaction)+geom_point()
```



Hide

```
ggplot(data = Cleansed.Data[Cleansed.Data$clusterid=='2',])+aes(x=Flight_Distance,fill=Satisfaction)+geom_bar()
```



Random forest model to identify variable importance:

Hide

```
Clust2_TrngData$Satisfaction=ifelse(Clust2_TrngData$Satisfaction==1,'satisfied','dissatisfied')
Clust2_TrngData$Satisfaction=as.factor(Clust2_TrngData$Satisfaction)
```

Hide

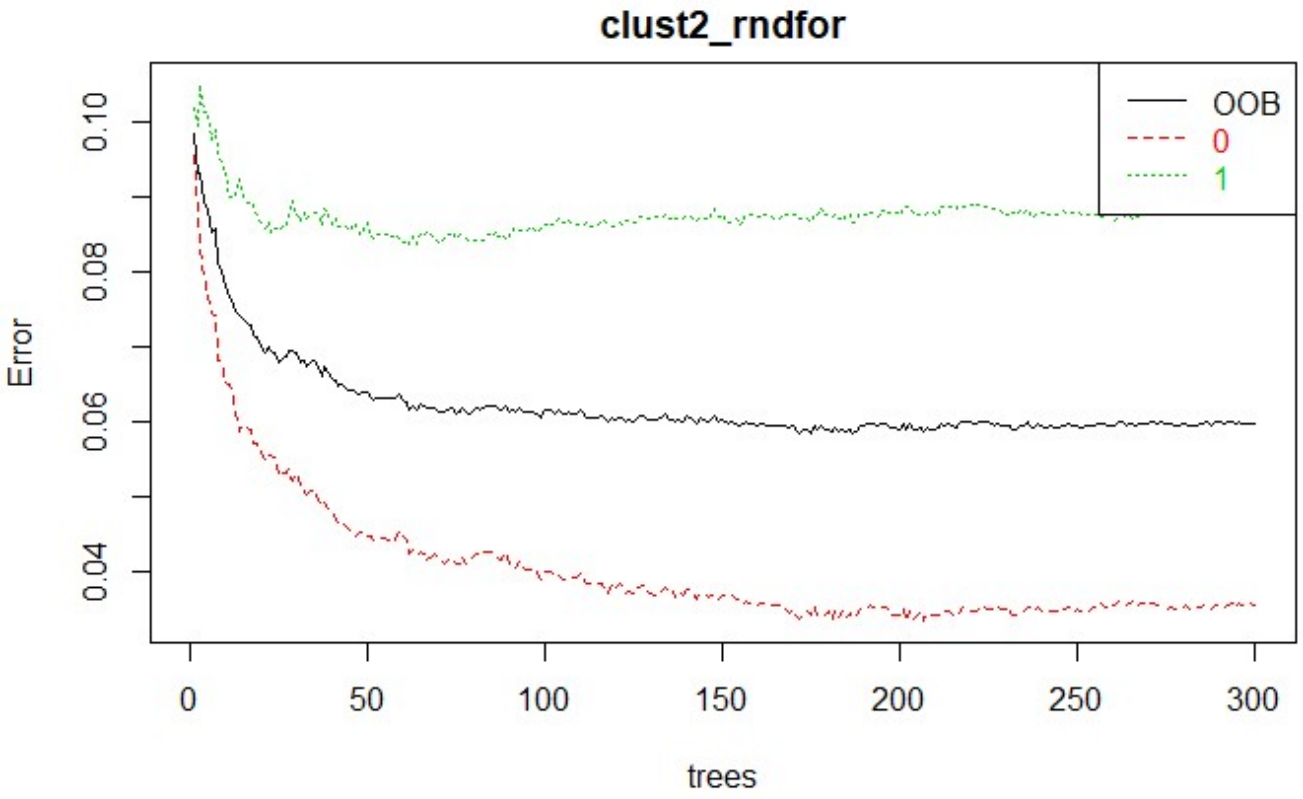
```
Clust2_TestData$Satisfaction=ifelse(Clust2_TestData$Satisfaction==1,'satisfied','dissatisfied')
Clust2_TestData$Satisfaction=as.factor(Clust2_TestData$Satisfaction)
```

Hide

```
set.seed(2000)
clust2_rndfor = randomForest(Satisfaction~.,data = Clust2_TrngData[-c(1,25,26)],ntree=51,mtry=6,nodesize=10,importance=T)
```

Hide

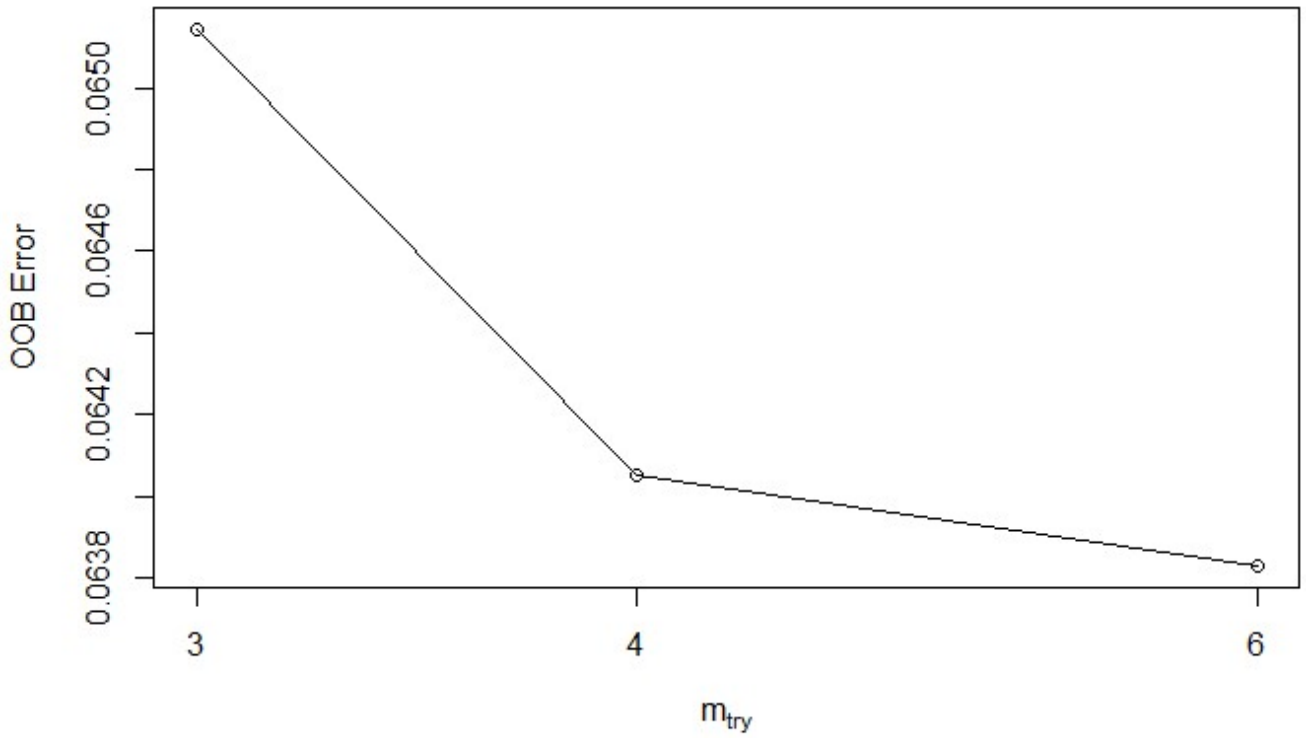
```
plot(clust2_rndfor)
legend("topright", c("OOB", "0", "1"), text.col=1:6, lty=1:3, col=1:3)
```



Hide

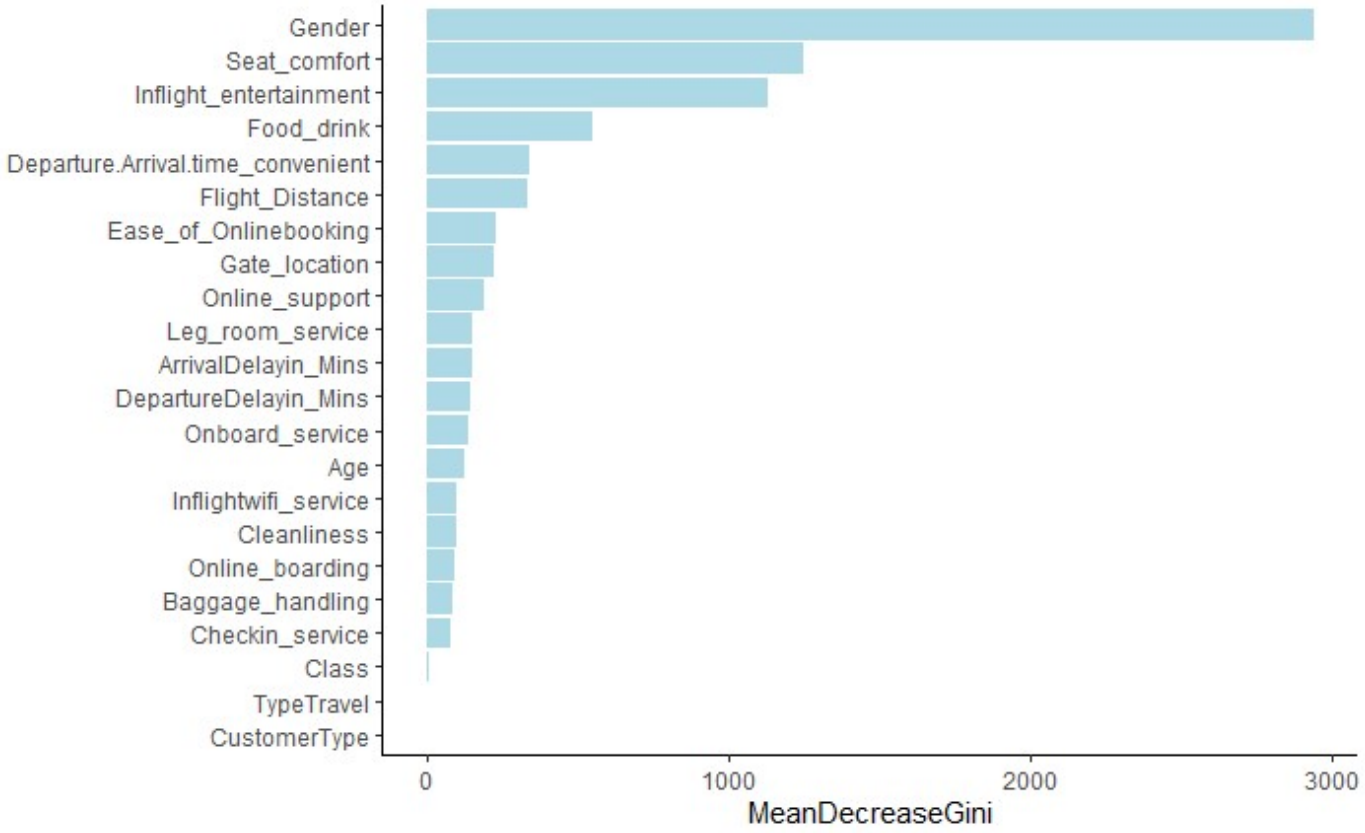
```
trndfor = tuneRF(x=Clust2_TrngData[-c(1,10,25,26)],
                 y=Clust2_TrngData$Satisfaction,
                 mtryStart = 4,
                 stepFactor = 1.5,
                 ntreeTry = 51,
                 trace = T,
                 plot = T,
                 importance=T,
                 doBest = T)
```

```
mtry = 4  OOB error = 6.4%
Searching left ...
mtry = 3    OOB error = 6.51%
-0.01709402 0.05
Searching right ...
mtry = 6    OOB error = 6.38%
0.003418803 0.05
```



Hide

```
a=as.data.frame(clust2_rndfor$importance)
a$features=row.names(a)
ggplot(data = a)+aes(x=reorder(features,MeanDecreaseGini),MeanDecreaseGini)+geom_bar(stat =
'identity',fill='light blue')+coord_flip()+theme_classic()+theme(axis.title.y = element_blank
())
```



Hide

```
predicted.class = predict(clust2_rndfor,Clust2_TestData[-c(1,10)],'class')
predicted.prob = predict(clust2_rndfor,Clust2_TestData[-c(1,10)],'prob')
```

Hide

```
#confusion matrix on Training Data
table(Clust2_TestData$Satisfaction,predicted.class)
```

```
              predicted.class
               dissatisfied satisfied
  dissatisfied         4013       157
  satisfied             291      3369
```

Hide

```
#accuracy on Training Data
sum(diag(table(Clust2_TestData$Satisfaction,predicted.class)))/nrow(Clust2_TestData)
```

```
[1] 0.9427842
```

Hide

```
#TNR
table(Clust2_TestData$Satisfaction,predicted.class)[1,1]/sum(table(Clust2_TestData$Satisfacti
on,predicted.class)[1,])
```

```
[1] 0.9623501
```

Hide

```
#TPR
table(Clust2_TestData$Satisfaction,predicted.class)[2,2]/sum(table(Clust2_TestData$Satisfacti
on,predicted.class)[2,])
```

```
[1] 0.9204918
```

Hide

```
#AUC
roc(Clust2_TestData$Satisfaction,predicted.prob[,2])
```

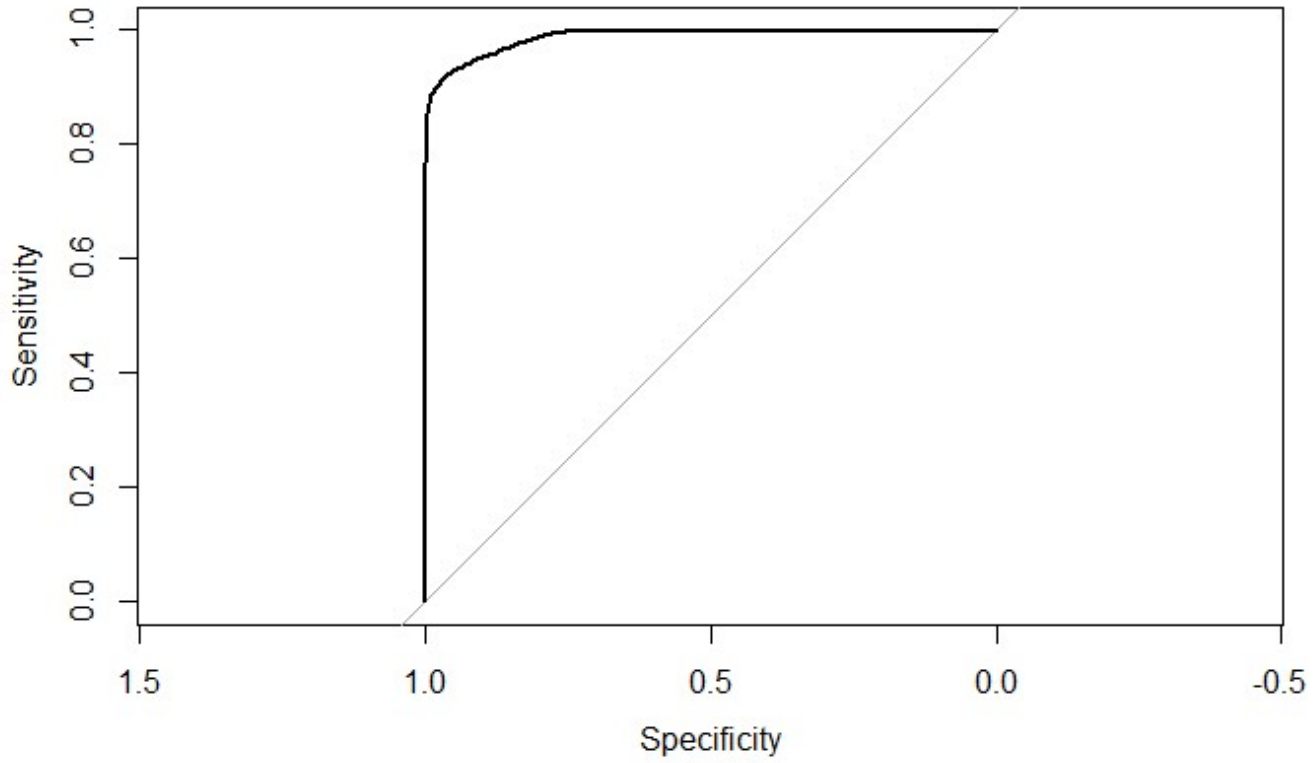
```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```

```

Call:
roc.default(response = Clust2_TestData$Satisfaction, predictor = predicted.prob[,     2])

Data: predicted.prob[, 2] in 4170 controls (Clust2_TestData$Satisfaction dissatisfied) < 3660
cases (Clust2_TestData$Satisfaction satisfied).
Area under the curve: 0.9876
```

Hide

```
plot(roc(Clust2_TestData$Satisfaction,predicted.prob[,2]))
```

```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```



Cluster 3

Hide

```
Clust3_TrngData$Satisfaction=ifelse(Clust3_TrngData$Satisfaction=='satisfied','1','0')
Clust3_TrngData$Satisfaction=as.integer(Clust3_TrngData$Satisfaction)
```

Hide

```
Clust3_TestData$Satisfaction=ifelse(Clust3_TestData$Satisfaction=='satisfied','1','0')
Clust3_TestData$Satisfaction=as.integer(Clust3_TestData$Satisfaction)
```

Hide

```
clust3.xgb.fit = xgboost(
  data = as.matrix(data.matrix(Clust3_TrngData[,-c(1,10,25,26)])),
  label = as.matrix(data.matrix(Clust3_TrngData[,c(10)])),
  eta = 0.3,#this is like shrinkage in the previous algorithm
  max_depth = 5,#Larger the depth, more complex the model; higher chances of overfitting. The
re is no standard                       value for max_depth. Larger data sets require deep tre
es to learn the rules from data.
  min_child_weight = 5,#it blocks the potential feature interactions to prevent overfitting
  nrounds = 150,#controls the maximum number of iterations. For classification, it is similar
to the number of                        trees to grow.
  nfold = 5,
  objective = "binary:logistic",  # for regression models
  verbose = 0,               # silent,
  early_stopping_rounds = 10 # stop if no improvement for 10 consecutive trees
)
```

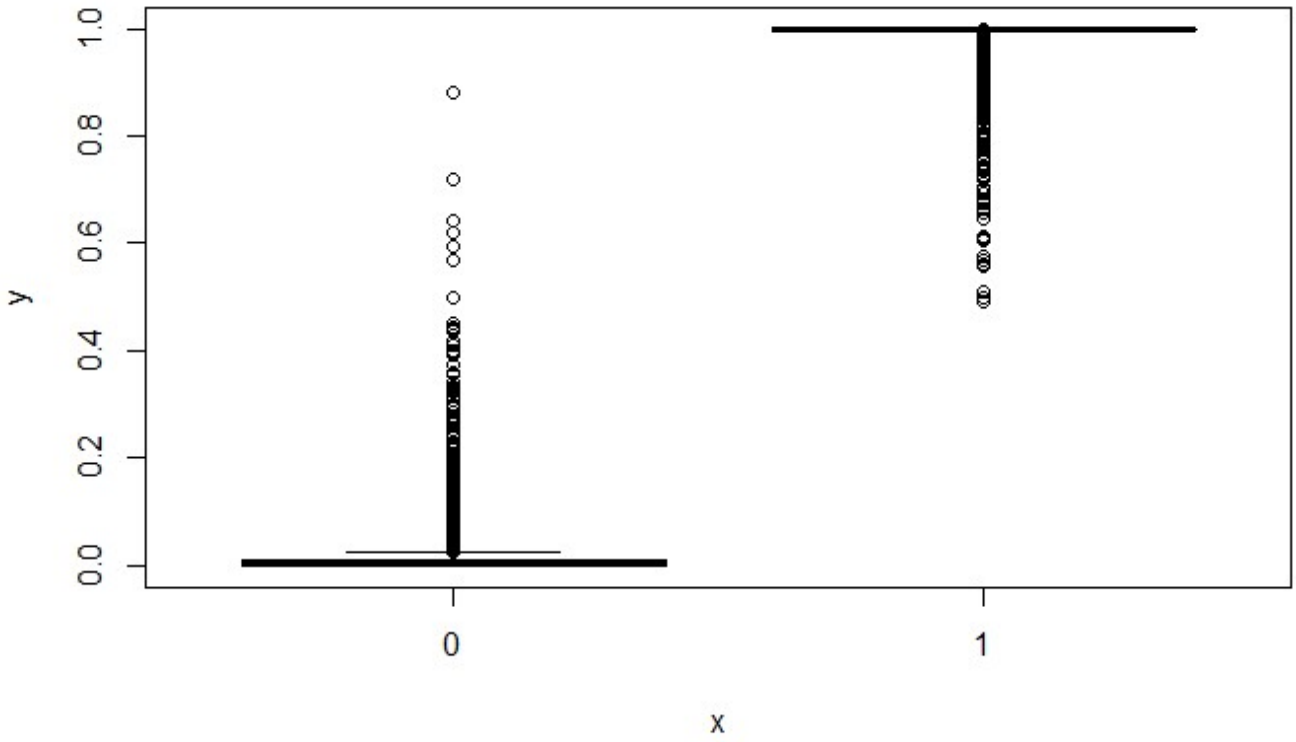
Hide

```
clust3.xgb.fit$best_iteration
```

```
[1] 150
```

Hide

```
predicted.prob=predict(clust3.xgb.fit,as.matrix(data.matrix(Clust3_TrngData[,-c(1,10,25,2
6)])))
plot(as.factor(Clust3_TrngData$Satisfaction),predicted.prob)
```



Hide

```
predicted.class=ifelse(predicted.prob>0.5,'1','0')
```

Hide

```
#confusion matrix on Training Data
table(as.factor(Clust3_TrngData$Satisfaction),predicted.class)
```

```
   predicted.class
       0    1
  0 2749    6
  1    1 9578
```

Hide

```
#accuracy on Training Data
sum(diag(table(as.factor(Clust3_TrngData$Satisfaction),predicted.class)))/nrow(Clust3_TrngDat
a)
```

```
[1] 0.9994325
```

Hide

```
#TNR
table(as.factor(Clust3_TrngData$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(Clust
3_TrngData$Satisfaction),predicted.class)[1,])
```

```
[1] 0.9978221
```

Hide

```
#TPR
table(as.factor(Clust3_TrngData$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(Clust
3_TrngData$Satisfaction),predicted.class)[2,])
```

```
[1] 0.9998956
```

Hide

```
#AUC
roc(Clust3_TrngData$Satisfaction,predicted.prob)
```

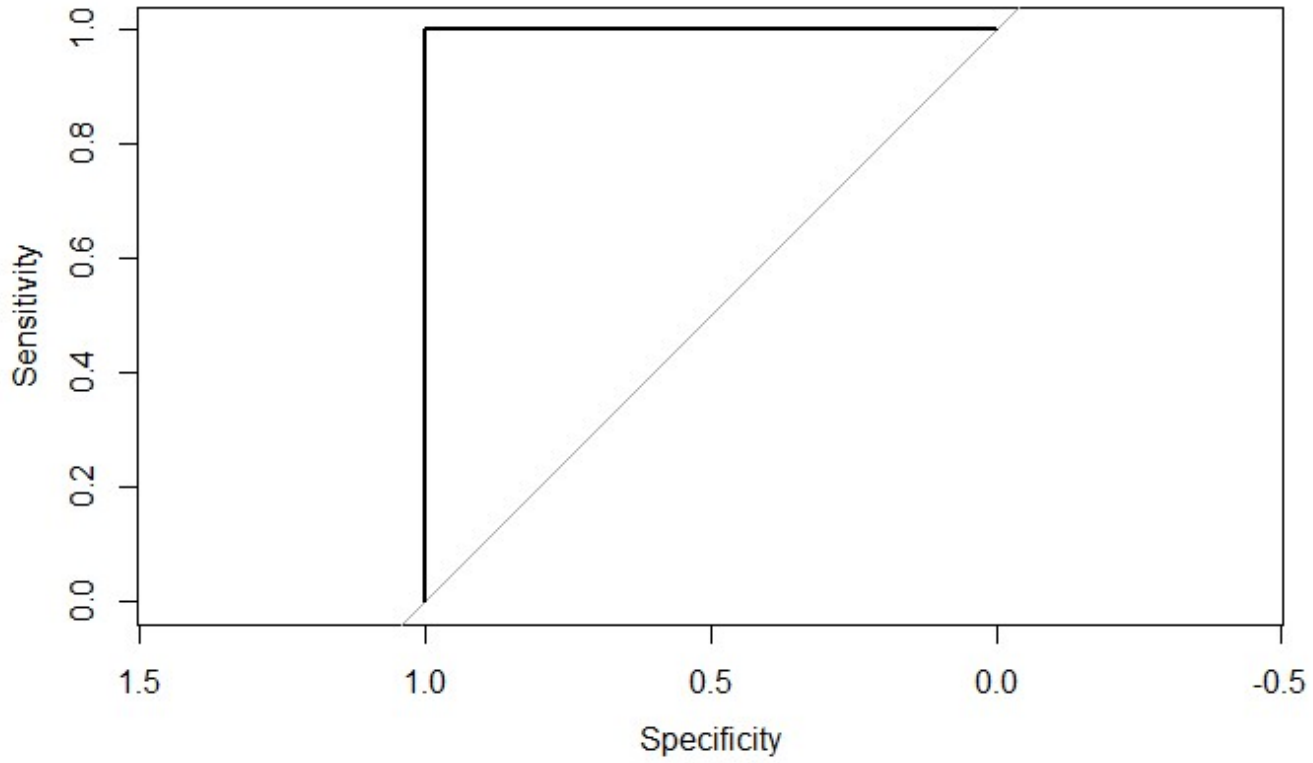
```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

```

Call:
roc.default(response = Clust3_TrngData$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 2755 controls (Clust3_TrngData$Satisfaction 0) < 9579 cases (Clust3_T
rngData$Satisfaction 1).
Area under the curve: 1
```

Hide

```
plot(roc(Clust3_TrngData$Satisfaction,predicted.prob))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```



Hide

```
predicted.prob=predict(clust3.xgb.fit,as.matrix(data.matrix(Clust3_TestData[,-c(1,10,25,2
6)])))
predicted.class=ifelse(predicted.prob>0.5,'1','0')
```

Hide

```
#confusion matrix on Test Data
table(as.factor(Clust3_TestData$Satisfaction),predicted.class)
```

```
   predicted.class
      0    1
  0 1096   35
  1   32 4121
```

Hide

```
#accuracy on Test Data
sum(diag(table(as.factor(Clust3_TestData$Satisfaction),predicted.class)))/nrow(Clust3_TestDat
a)
```

```
[1] 0.9873202
```

Hide

```
#TNR
table(as.factor(Clust3_TestData$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(Clust
3_TestData$Satisfaction),predicted.class)[1,])
```

```
[1] 0.9690539
```

Hide

```
#TPR
table(as.factor(Clust3_TestData$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(Clust
3_TestData$Satisfaction),predicted.class)[2,])
```

```
[1] 0.9922947
```

Hide

```
#AUC
roc(Clust3_TestData$Satisfaction,predicted.prob)
```

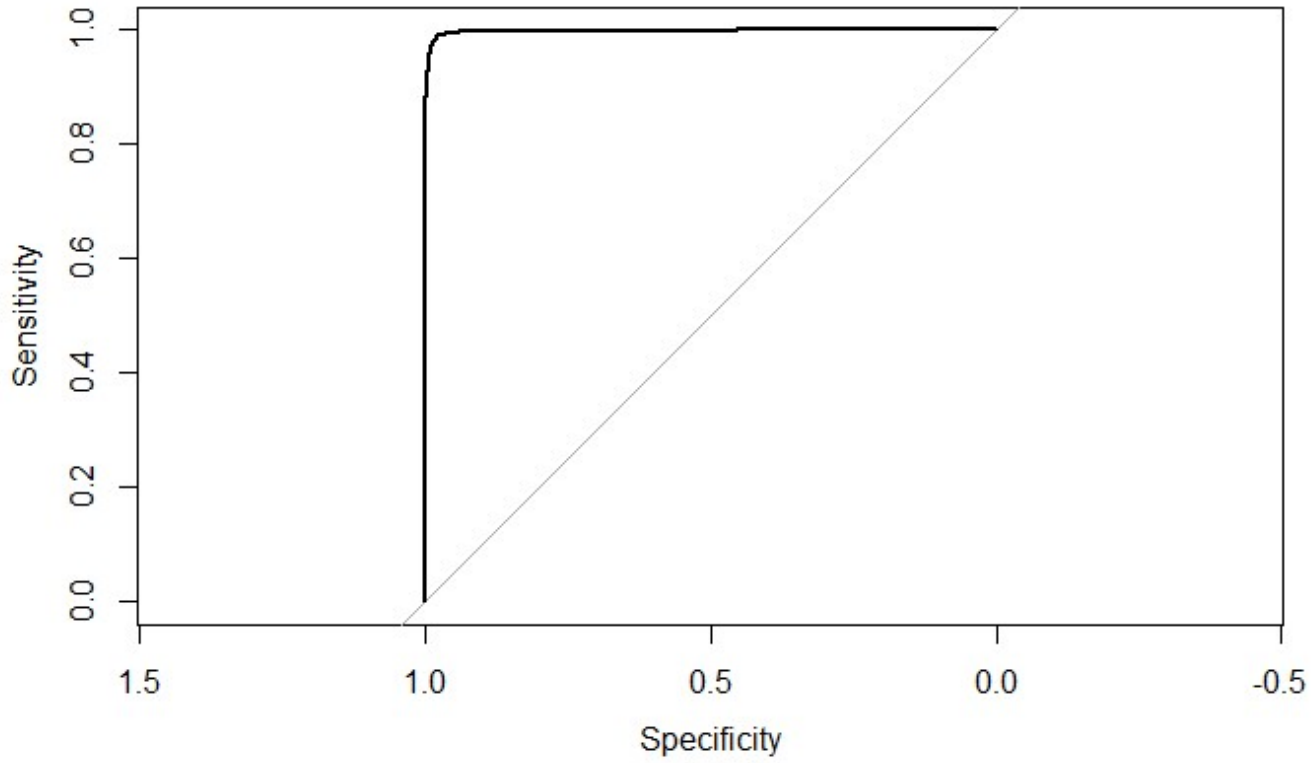
```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

```

Call:
roc.default(response = Clust3_TestData$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 1131 controls (Clust3_TestData$Satisfaction 0) < 4153 cases (Clust3_T
estData$Satisfaction 1).
Area under the curve: 0.998
```

Hide

```
plot(roc(Clust3_TestData$Satisfaction,predicted.prob))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```



Hide

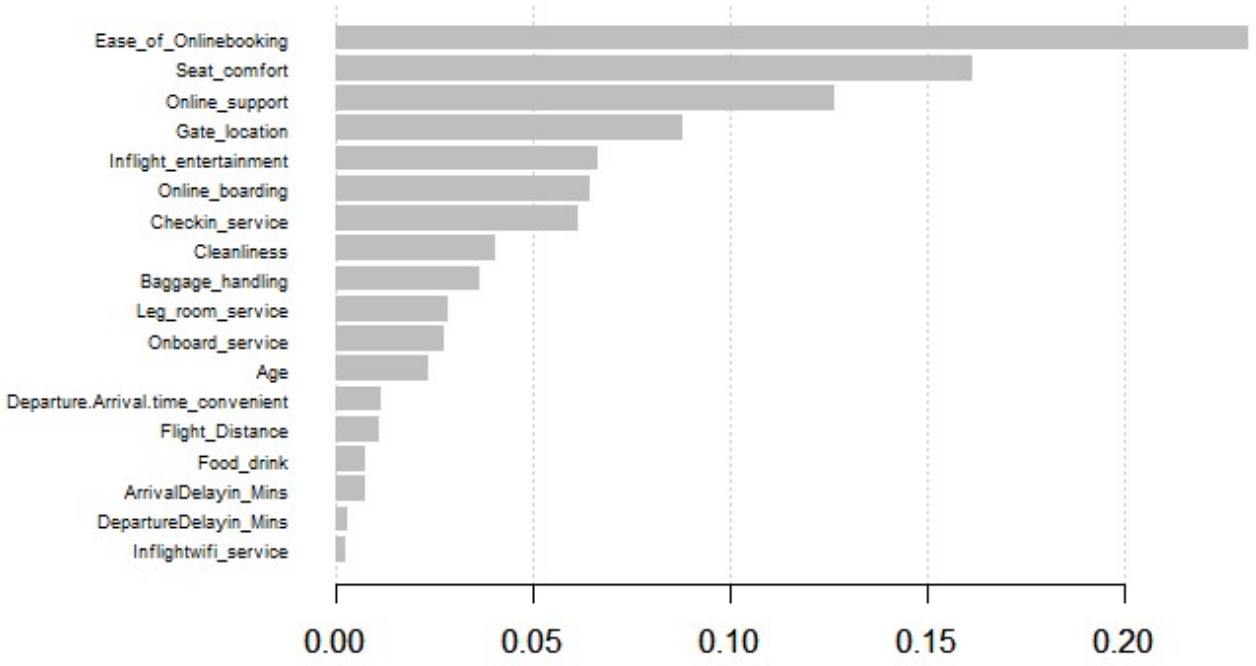
```
xgb.importance(model = clust3.xgb.fit)
```

Feature <chr>	Gain <dbl>	Cover <dbl>	Frequency <dbl>
Ease_of_Onlinebooking	0.231791286	0.086907507	0.04636119
Seat_comfort	0.161442739	0.097430004	0.10943396
Online_support	0.126494160	0.101938555	0.04366577
Gate_location	0.088241800	0.053866254	0.05876011
Inflight_entertainment	0.066625137	0.111743594	0.04905660
Online_boarding	0.064671822	0.067108885	0.04312668
Checkin_service	0.061457253	0.087302762	0.04097035
Cleanliness	0.040533081	0.041091924	0.03234501
Baggage_handling	0.036332338	0.042459695	0.06522911
Leg_room_service	0.028240864	0.045840138	0.03881402

1-10 of 18 rows Previous **1** 2 Next

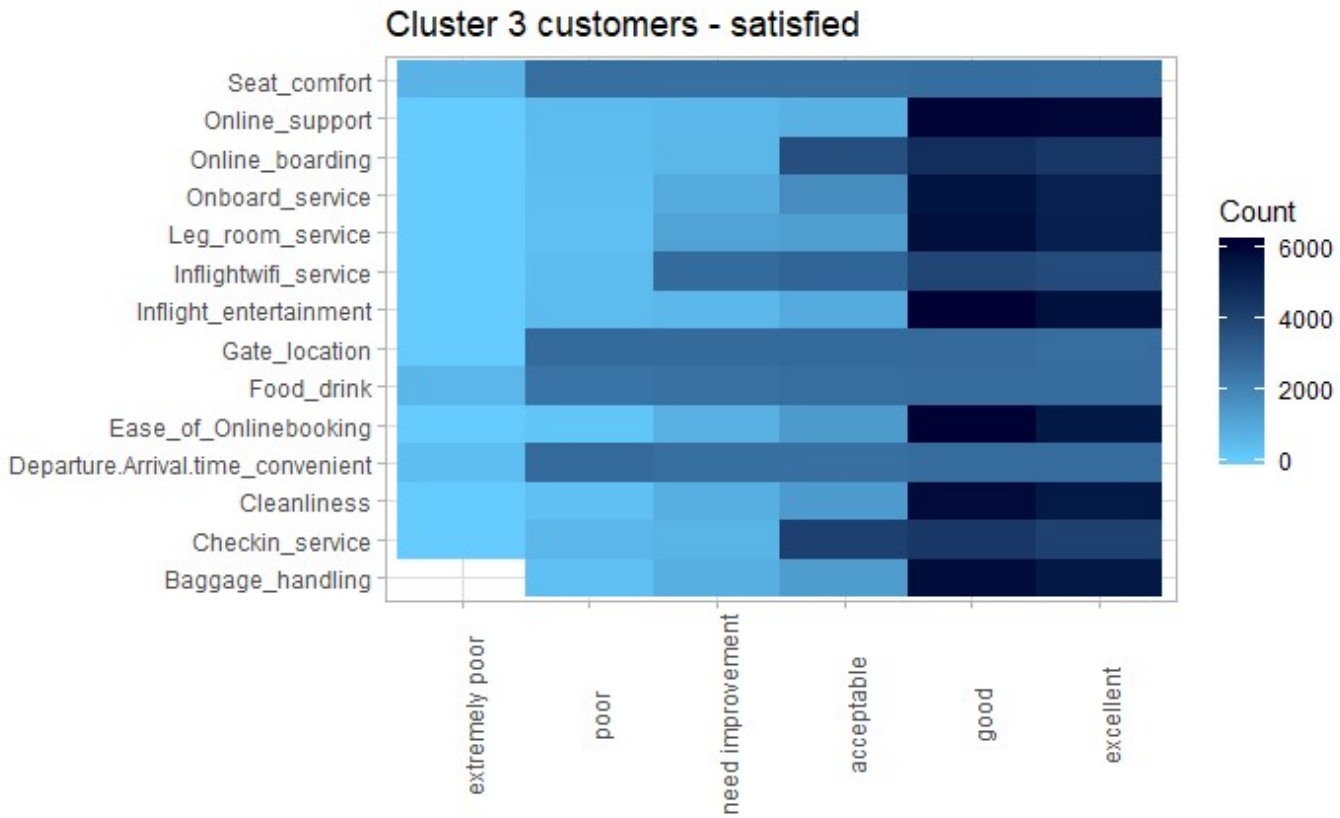
Hide

```
xgb.plot.importance(xgb.importance(model = clust3.xgb.fit))
```



Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$clusterid=='3' & Cleansed.Data$Satisfaction=
='satisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Cluster 3 customers - satisfied')
```



Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$clusterid=='3' & Cleansed.Data$Satisfaction=
='dissatisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Cluster 3 customers - dissatisfied')
```

Random forest model for variable importance:

Hide

```
Clust3_TrngData$Satisfaction=ifelse(Clust3_TrngData$Satisfaction==1,'satisfied','dissatisfie
d')
Clust3_TrngData$Satisfaction=as.factor(Clust3_TrngData$Satisfaction)
```

Hide

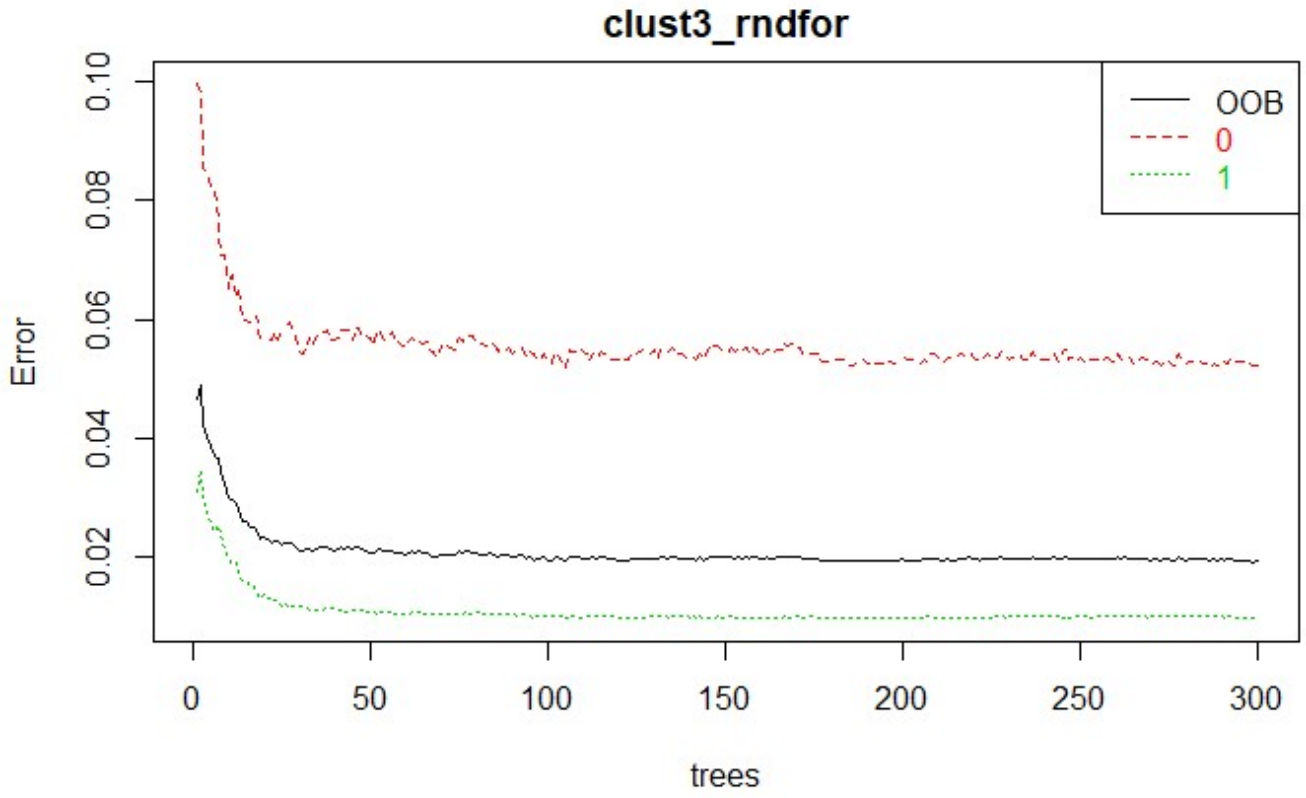
```
Clust3_TestData$Satisfaction=ifelse(Clust3_TestData$Satisfaction==1,'satisfied','dissatisfie
d')
Clust3_TestData$Satisfaction=as.factor(Clust3_TestData$Satisfaction)
```

Hide

```
set.seed(3000)
clust3_rndfor = randomForest(Satisfaction~.,data = Clust3_TrngData[-c(1,25,26)],ntree=199,mtr
y=4,nodesize=10,importance=T)
```

Hide

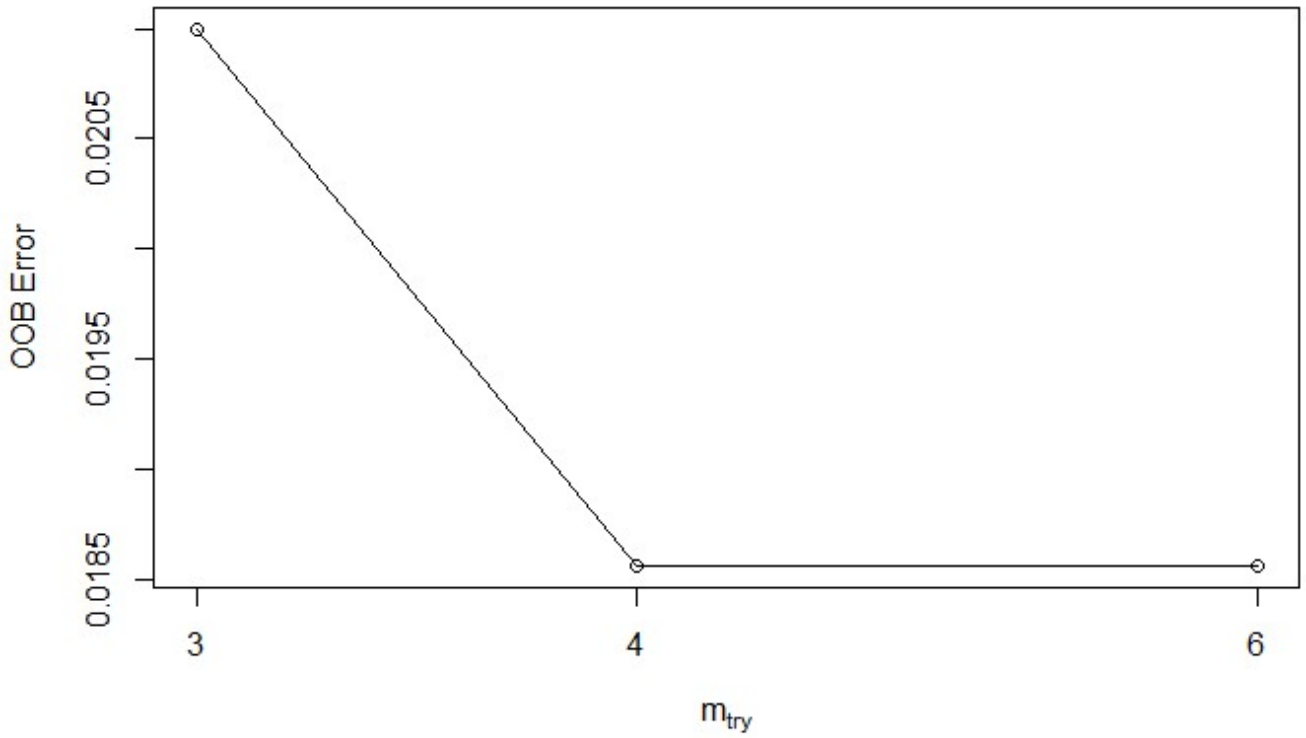
```
plot(clust3_rndfor)
legend("topright", c("OOB", "0", "1"), text.col=1:6, lty=1:3, col=1:3)
```



Hide

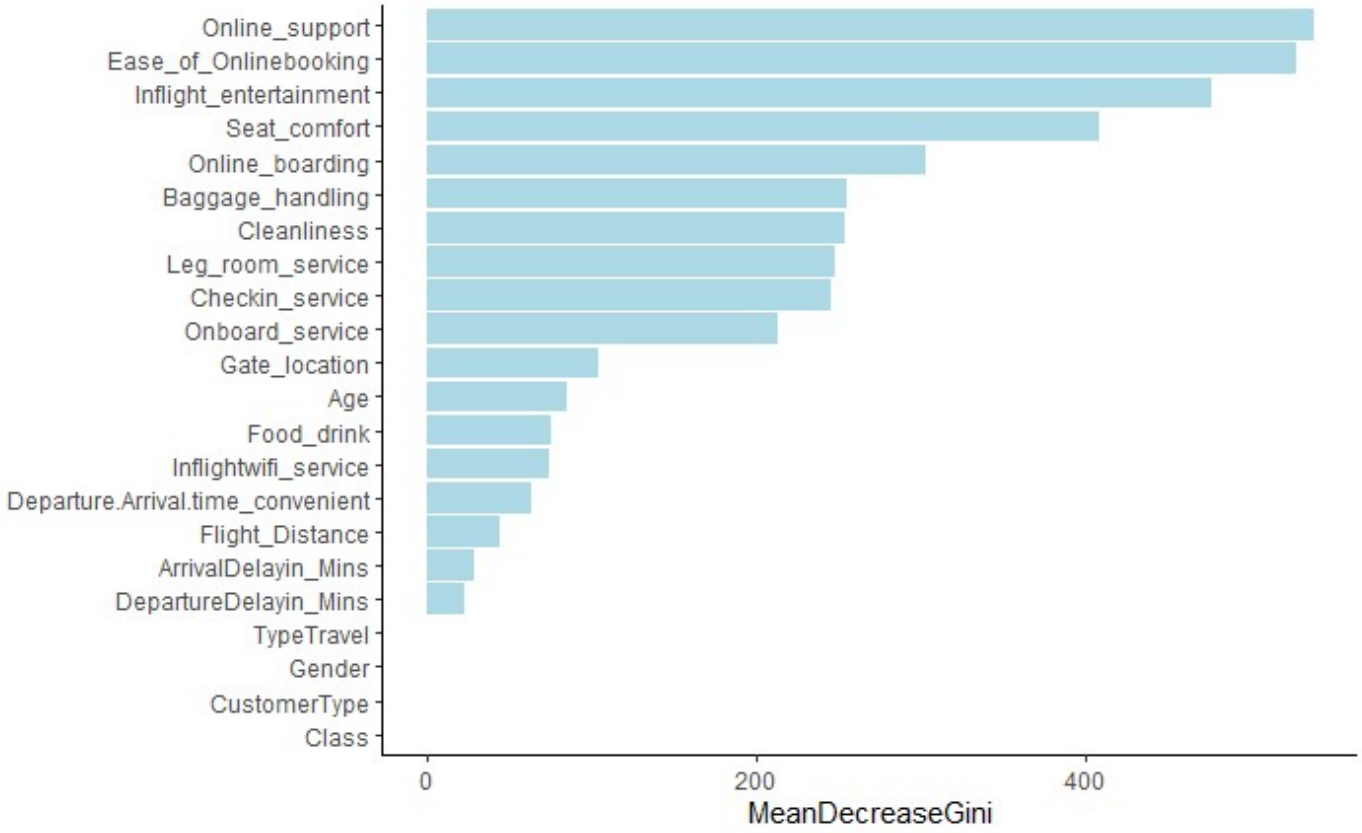
```
trndfor = tuneRF(x=Clust3_TrngData[-c(1,10,25,26)],
                 y=Clust3_TrngData$Satisfaction,
                 mtryStart = 4,
                 stepFactor = 1.5,
                 ntreeTry = 199,
                 trace = T,
                 plot = T,
                 importance=T,
                 doBest = T)
```

```
mtry = 4  OOB error = 1.86%
Searching left ...
mtry = 3    OOB error = 2.1%
-0.1310044 0.05
Searching right ...
mtry = 6    OOB error = 1.86%
0 0.05
```



Hide

```
a=as.data.frame(clust3_rndfor$importance)
a$features=row.names(a)
ggplot(data = a)+aes(x=reorder(features,MeanDecreaseGini),MeanDecreaseGini)+geom_bar(stat =
'identity',fill='light blue')+coord_flip()+theme_classic()+theme(axis.title.y = element_blank
())
```



Hide

```
predicted.class = predict(clust3_rndfor,Clust3_TestData[-c(1,10)],'class')
predicted.prob = predict(clust3_rndfor,Clust3_TestData[-c(1,10)],'prob')
```

Hide

```
#confusion matrix on Training Data
table(Clust3_TestData$Satisfaction,predicted.class)
```

```
              predicted.class
               dissatisfied satisfied
  dissatisfied         1073        58
  satisfied              49      4104
```

Hide

```
#accuracy on Training Data
sum(diag(table(Clust3_TestData$Satisfaction,predicted.class)))/nrow(Clust3_TestData)
```

```
[1] 0.9797502
```

Hide

```
#TNR
table(Clust3_TestData$Satisfaction,predicted.class)[1,1]/sum(table(Clust3_TestData$Satisfacti
on,predicted.class)[1,])
```

```
[1] 0.9487179
```

Hide

```
#TPR
table(Clust3_TestData$Satisfaction,predicted.class)[2,2]/sum(table(Clust3_TestData$Satisfacti
on,predicted.class)[2,])
```

```
[1] 0.9882013
```

Hide

```
#AUC
roc(Clust3_TestData$Satisfaction,predicted.prob[,2])
```

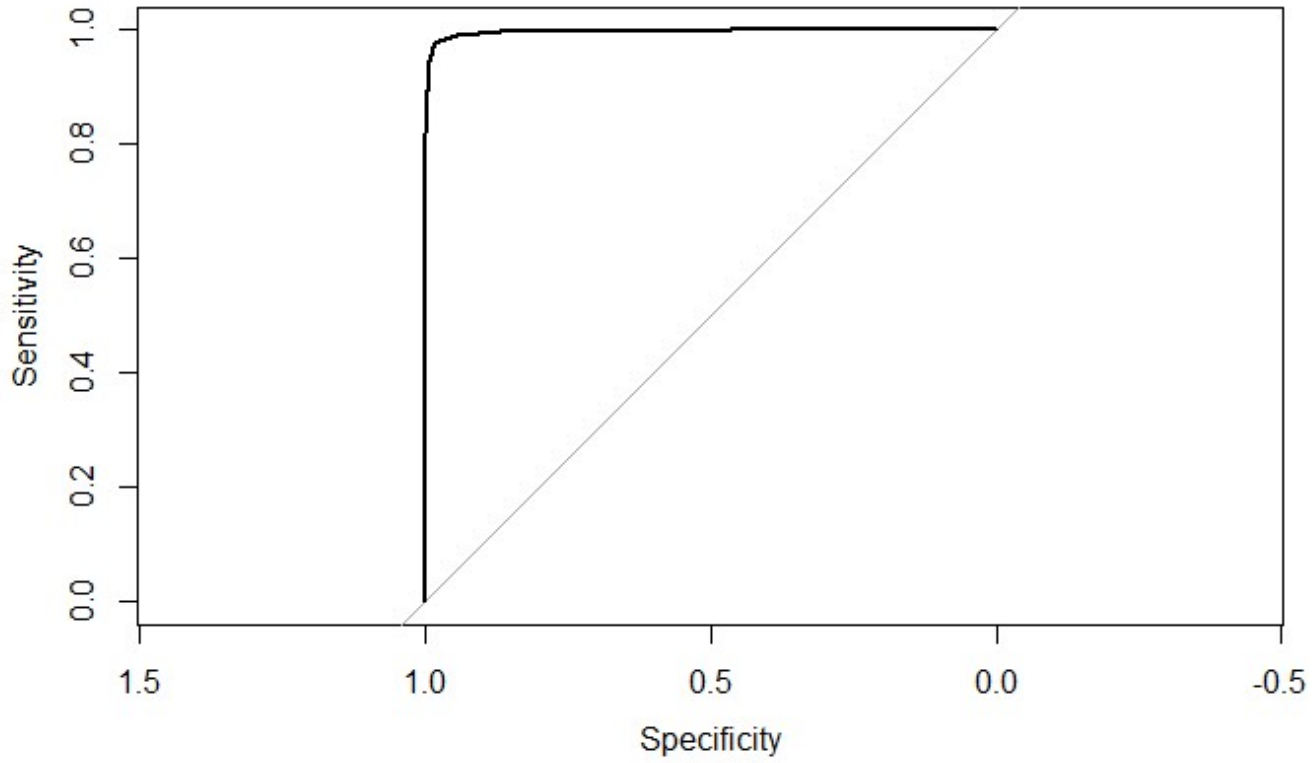
```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```

```

Call:
roc.default(response = Clust3_TestData$Satisfaction, predictor = predicted.prob[,     2])

Data: predicted.prob[, 2] in 1131 controls (Clust3_TestData$Satisfaction dissatisfied) < 4153
cases (Clust3_TestData$Satisfaction satisfied).
Area under the curve: 0.9967
```

Hide

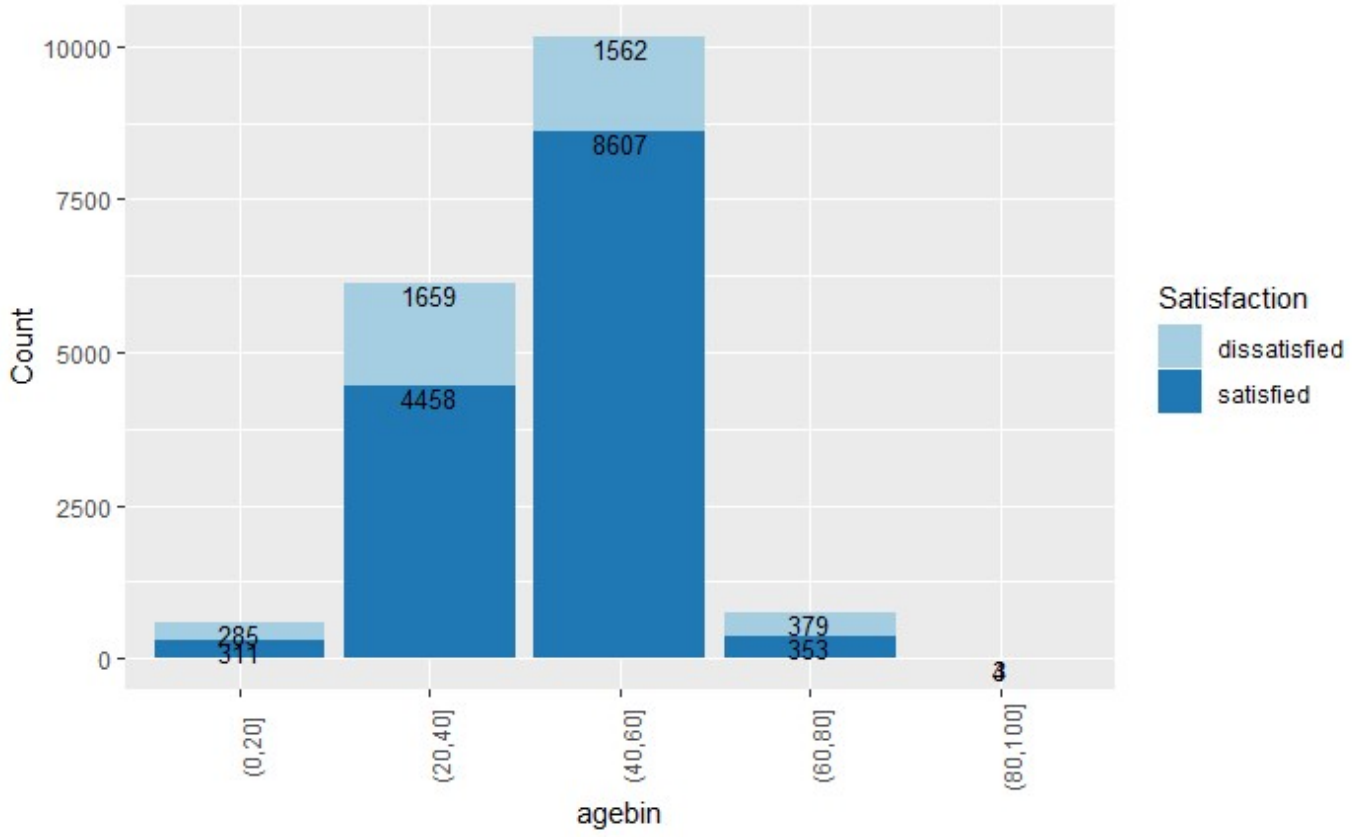
```
plot(roc(Clust3_TestData$Satisfaction,predicted.prob[,2]))
```

```
Setting levels: control = dissatisfied, case = satisfied
Setting direction: controls < cases
```



Hide

```
ggplot(data = Cleansed.Data[Cleansed.Data$clusterid=='3',])+aes(x=agebin,fill=Satisfaction,y
=..count..)+geom_bar(position = 'stack')+labs(y='Count')+geom_text(aes(label=..count..),stat
= 'count',position='stack', vjust=1,size=3.5)+scale_fill_brewer(palette="Paired")+theme(axis.
text.x=element_text(angle = 90))
```



Cluster 4: XGBoost model for prediction:

Hide

```
Clust4_TrngData$Satisfaction=ifelse(Clust4_TrngData$Satisfaction=='satisfied','1','0')
Clust4_TrngData$Satisfaction=as.integer(Clust4_TrngData$Satisfaction)
```

Hide

```
Clust4_TestData$Satisfaction=ifelse(Clust4_TestData$Satisfaction=='satisfied','1','0')
Clust4_TestData$Satisfaction=as.integer(Clust4_TestData$Satisfaction)
```

Hide

```
clust4.xgb.fit = xgboost(
  data = as.matrix(data.matrix(Clust4_TrngData[,-c(1,10,25,26)])),
  label = as.matrix(data.matrix(Clust4_TrngData[,c(10)])),
  eta = 0.1,#this is like shrinkage in the previous algorithm
  max_depth = 5,#Larger the depth, more complex the model; higher chances of overfitting. The
re is no standard                       value for max_depth. Larger data sets require deep tre
es to learn the rules from data.
  min_child_weight = 5,#it blocks the potential feature interactions to prevent overfitting
  nrounds = 200,#controls the maximum number of iterations. For classification, it is similar
to the number of                        trees to grow.
  nfold = 5,
  objective = "binary:logistic",  # for regression models
  verbose = 0,               # silent,
  early_stopping_rounds = 10 # stop if no improvement for 10 consecutive trees
)
```

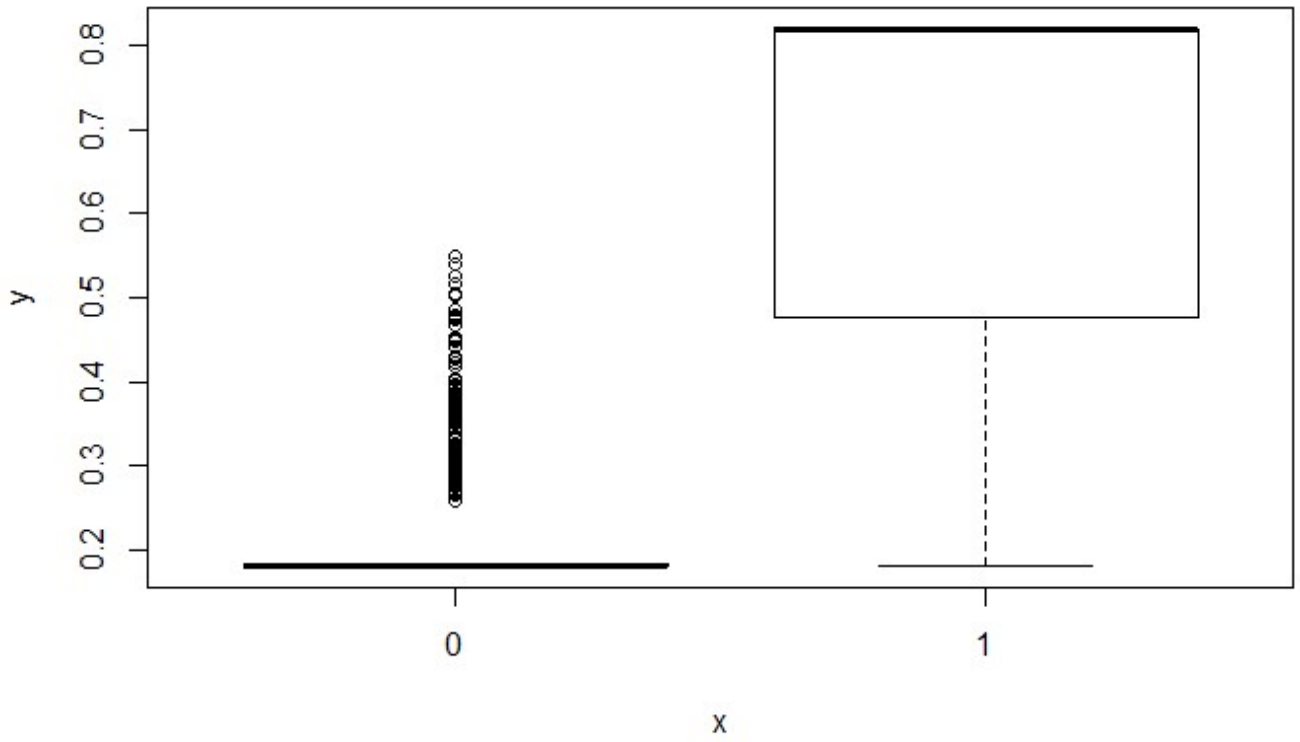
Hide

```
clust4.xgb.fit$best_ntreelimit
```

```
[1] 10
```

Hide

```
predicted.prob=predict(clust4.xgb.fit,as.matrix(data.matrix(Clust4_TrngData[,-c(1,10,25,2
6)])))
plot(as.factor(Clust4_TrngData$Satisfaction),predicted.prob)
```



Hide

```
predicted.class=ifelse(predicted.prob>0.3,'1','0')
```

Hide

```
#confusion matrix on Training Data
table(as.factor(Clust4_TrngData$Satisfaction),predicted.class)
```

```
   predicted.class
       0    1
  0 5567  505
  1   61  881
```

Hide

```
#accuracy on Training Data
sum(diag(table(as.factor(Clust4_TrngData$Satisfaction),predicted.class)))/nrow(Clust4_TrngDat
a)
```

```
[1] 0.9193042
```

Hide

```
#TNR
table(as.factor(Clust4_TrngData$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(Clust
4_TrngData$Satisfaction),predicted.class)[1,])
```

```
[1] 0.9168314
```

Hide

```
#TPR
table(as.factor(Clust4_TrngData$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(Clust
4_TrngData$Satisfaction),predicted.class)[2,])
```

```
[1] 0.9352442
```

Hide

```
#AUC
roc(Clust4_TrngData$Satisfaction,predicted.prob)
```

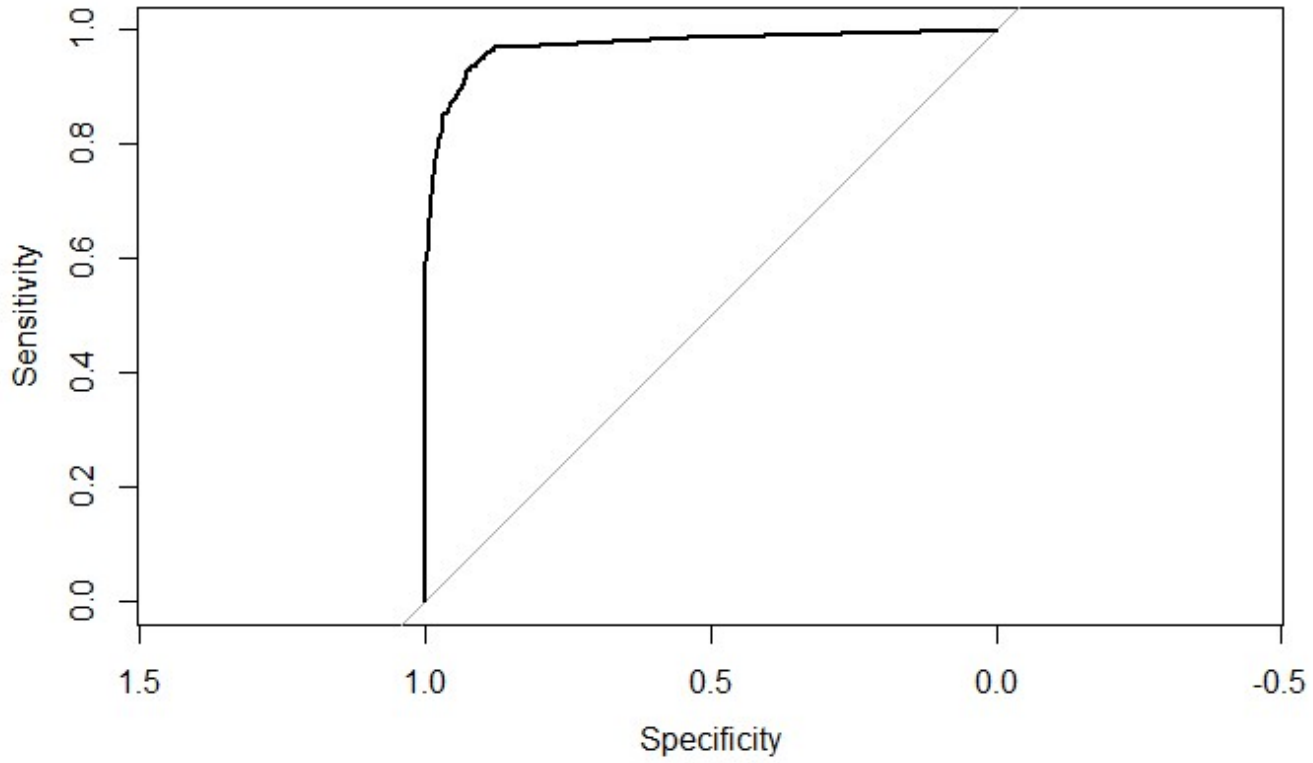
```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

```

Call:
roc.default(response = Clust4_TrngData$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 6072 controls (Clust4_TrngData$Satisfaction 0) < 942 cases (Clust4_Tr
ngData$Satisfaction 1).
Area under the curve: 0.973
```

Hide

```
plot(roc(Clust4_TrngData$Satisfaction,predicted.prob))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```



Hide

```
predicted.prob=predict(clust4.xgb.fit,as.matrix(data.matrix(Clust4_TestData[,-c(1,10,25,2
6)])))
predicted.class=ifelse(predicted.prob>0.3,'1','0')
```

Hide

```
#confusion matrix on Test Data
table(as.factor(Clust4_TestData$Satisfaction),predicted.class)
```

```
   predicted.class
      0    1
  0 2321  233
  1   32  420
```

Hide

```
#accuracy on Test Data
sum(diag(table(as.factor(Clust4_TestData$Satisfaction),predicted.class)))/nrow(Clust4_TestDat
a)
```

```
[1] 0.911843
```

Hide

```
#TNR
table(as.factor(Clust4_TestData$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(Clust
4_TestData$Satisfaction),predicted.class)[1,])
```

```
[1] 0.9087706
```

Hide

```
#TPR
table(as.factor(Clust4_TestData$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(Clust
4_TestData$Satisfaction),predicted.class)[2,])
```

```
[1] 0.9292035
```

Hide

```
#AUC
roc(Clust4_TestData$Satisfaction,predicted.prob)
```

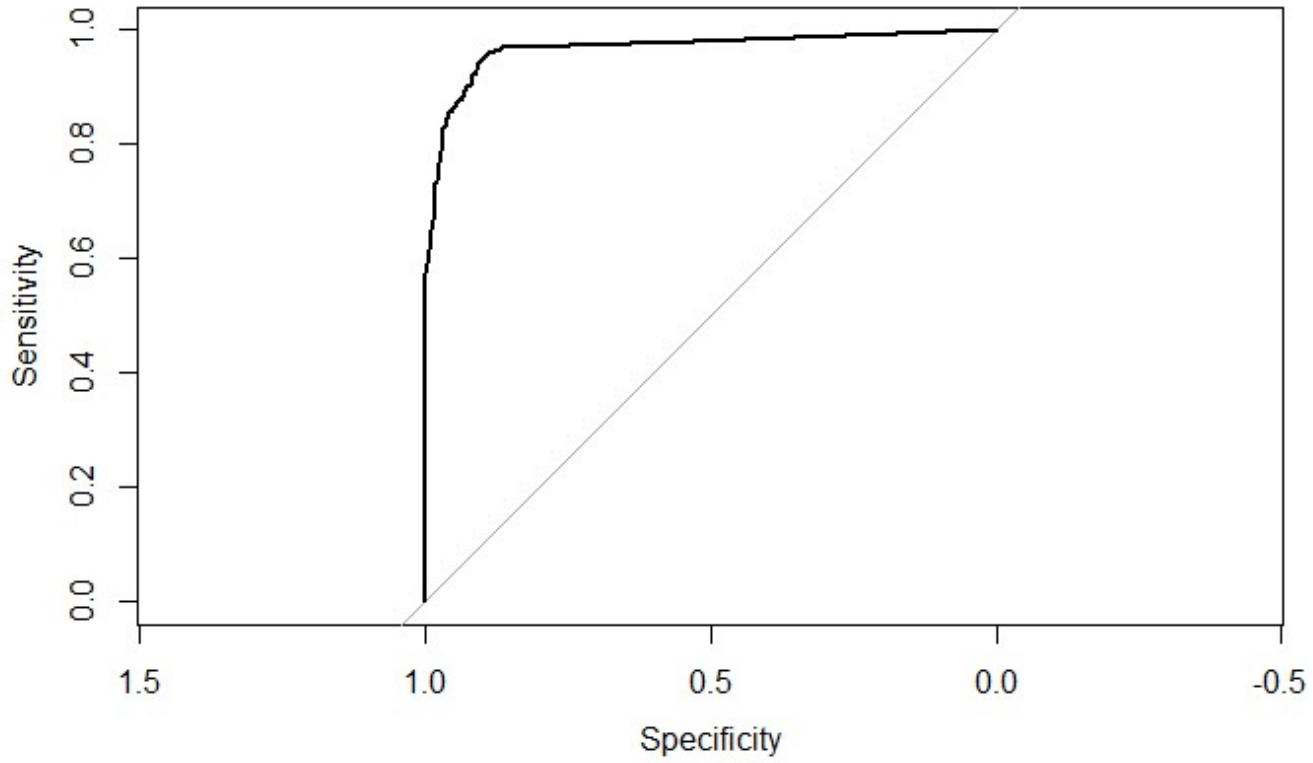
```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```

```

Call:
roc.default(response = Clust4_TestData$Satisfaction, predictor = predicted.prob)

Data: predicted.prob in 2554 controls (Clust4_TestData$Satisfaction 0) < 452 cases (Clust4_Te
stData$Satisfaction 1).
Area under the curve: 0.9683
```

Hide

```
plot(roc(Clust4_TestData$Satisfaction,predicted.prob))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases
```



Hide

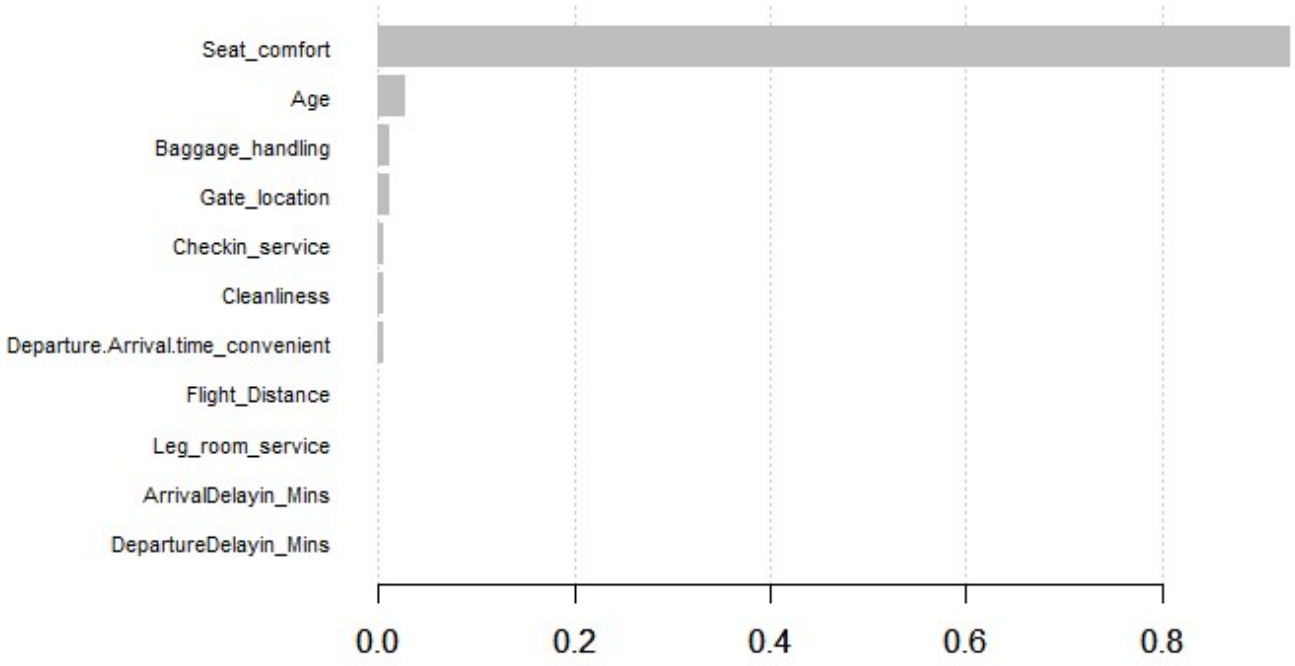
```
xgb.importance(model = clust4.xgb.fit)
```

Feature <chr>	Gain <dbl>	Cover <dbl>	Frequency <dbl>
Seat_comfort	9.313047e-01	0.7726495597	0.425531915
Age	2.791868e-02	0.0615414113	0.134751773
Baggage_handling	1.220259e-02	0.0346868610	0.070921986
Gate_location	1.086167e-02	0.0356864674	0.134751773
Checkin_service	5.696908e-03	0.0255119836	0.049645390
Cleanliness	5.588417e-03	0.0087444629	0.028368794
Departure.Arrival.time_convenient	5.448688e-03	0.0226875927	0.035460993
Flight_Distance	6.136658e-04	0.0355522766	0.092198582
Leg_room_service	2.168998e-04	0.0021485422	0.007092199
ArrivalDelayin_Mins	1.021421e-04	0.0005171278	0.014184397

1-10 of 11 rows Previous **1** 2 Next

Hide

```
xgb.plot.importance(xgb.importance(model = clust4.xgb.fit))
```



Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$clusterid=='4' & Cleansed.Data$Satisfaction=
='satisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Cluster 4 customers - satisfied')
```

Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$clusterid=='4' & Cleansed.Data$Satisfaction=
='dissatisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Cluster 4 customers - dissatisfied')
```

Since the variable importance of XGBoost is not coherent with the findings from the heatmap, lets try create a random forest model to just study the variable importance.

Hide

```
Clust4_TrngData$Satisfaction=ifelse(Clust4_TrngData$Satisfaction==1,'satisfied','dissatisfie
d')
Clust4_TrngData$Satisfaction=as.factor(Clust4_TrngData$Satisfaction)
```

Hide

```
Clust4_TestData$Satisfaction=ifelse(Clust4_TestData$Satisfaction==1,'satisfied','dissatisfie
d')
Clust4_TestData$Satisfaction=as.factor(Clust4_TestData$Satisfaction)
```

Hide

```
set.seed(4000)
clust4_rndfor = randomForest(Satisfaction~.,data = Clust4_TrngData[-c(1,25,26)],ntree=151,mtr
y=6,nodesize=10,importance=T)
```

Hide

```
plot(clust4_rndfor)
legend("topright", c("OOB", "0", "1"), text.col=1:6, lty=1:3, col=1:3)
```

Hide

```
a=as.data.frame(clust4_rndfor$importance)
a$features=row.names(a)
```

Hide

```
ggplot(data = a)+aes(x=reorder(features,MeanDecreaseGini),MeanDecreaseGini)+geom_bar(stat =
'identity',fill='light blue')+coord_flip()+theme_classic()+theme(axis.title.y = element_blank
())
```

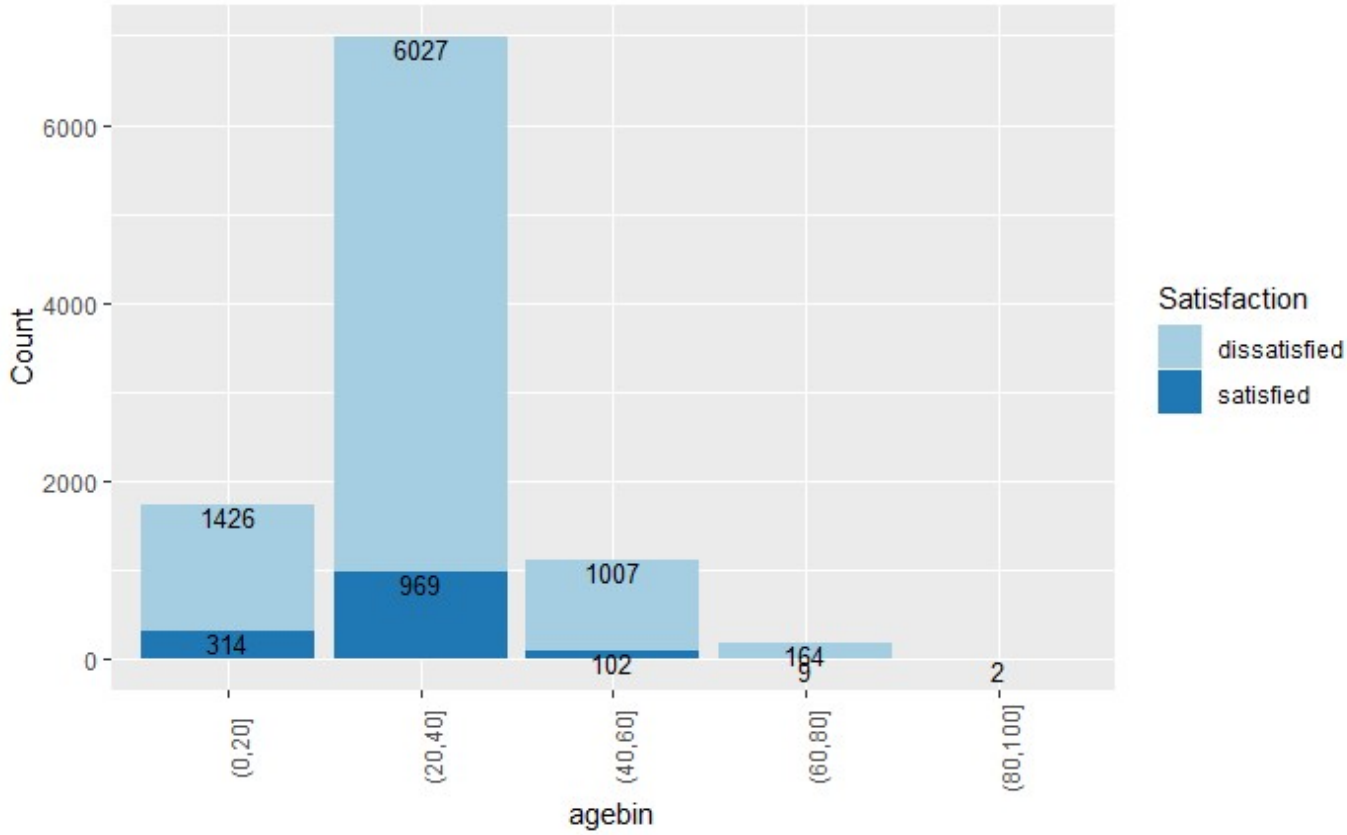
Hide

```
table(Clust4_TrngData$Satisfaction)
```

```
dissatisfied    satisfied
        6072          942
```

Hide

```
ggplot(data = Cleansed.Data[Cleansed.Data$clusterid=='4',])+aes(x=agebin,fill=Satisfaction,y
=..count..)+geom_bar(position = 'stack')+labs(y='Count')+geom_text(aes(label=..count..),stat
= 'count',position='stack', vjust=1,size=3.5)+scale_fill_brewer(palette="Paired")+theme(axis.
text.x=element_text(angle = 90))
```



Hide

```
predicted.class = predict(clust4_rndfor,Clust4_TestData[-c(1,10)],'class')
predicted.prob = predict(clust4_rndfor,Clust4_TestData[-c(1,10)],'prob')
```

Hide

```
#confusion matrix on Training Data
table(Clust4_TrngData$Satisfaction,predicted.class)
#accuracy on Training Data
sum(diag(table(Clust4_TrngData$Satisfaction,predicted.class)))/nrow(Clust4_TrngData)
#TNR
table(Clust4_TrngData$Satisfaction,predicted.class)[1,1]/sum(table(Clust4_TrngData$Satisfacti
on,predicted.class)[1,])
#TPR
table(Clust4_TrngData$Satisfaction,predicted.class)[2,2]/sum(table(Clust4_TrngData$Satisfacti
on,predicted.class)[2,])
#AUC
roc(Clust4_TrngData$Satisfaction,predicted.prob[,2])
plot(roc(Clust4_TrngData$Satisfaction,predicted.prob[,2]))
```

Hide

```
trndfor = tuneRF(x=Clust4_TrngData[-c(1,10,25,26)],
                 y=Clust4_TrngData$Satisfaction,
                 mtryStart = 4,
                 stepFactor = 1.5,
                 ntreeTry = 151,
                 trace = T,
                 plot = T,
                 importance=T,
                 doBest = T)
```

Others:

Hide

```
summary(Clust5_TestData)
```

Hide

```
Clust5_TrngData$Satisfaction=ifelse(Clust5_TrngData$Satisfaction=='satisfied','1','0')
Clust5_TrngData$Satisfaction=as.integer(Clust5_TrngData$Satisfaction)
```

Hide

```
Clust5_TestData$Satisfaction=ifelse(Clust5_TestData$Satisfaction=='satisfied','1','0')
Clust5_TestData$Satisfaction=as.integer(Clust5_TestData$Satisfaction)
```

Hide

```
clust5.xgb.fit = xgboost(
  data = as.matrix(data.matrix(Clust5_TrngData[,-c(1,10,25,26)])),
  label = as.matrix(data.matrix(Clust5_TrngData[,c(10)])),
  eta = 0.1,#this is like shrinkage in the previous algorithm
  max_depth = 5,#Larger the depth, more complex the model; higher chances of overfitting. The
re is no standard                       value for max_depth. Larger data sets require deep tre
es to learn the rules from data.
  min_child_weight = 5,#it blocks the potential feature interactions to prevent overfitting
  nrounds = 149,#controls the maximum number of iterations. For classification, it is similar
to the number of                        trees to grow.
  nfold = 5,
  objective = "binary:logistic",  # for regression models
  verbose = 0,               # silent,
  early_stopping_rounds = 10 # stop if no improvement for 10 consecutive trees
)
```

Hide

```
clust5.xgb.fit$best_iteration
```

Hide

```
predicted.prob=predict(clust5.xgb.fit,as.matrix(data.matrix(Clust5_TrngData[,-c(1,10,25,2
6)])))
plot(as.factor(Clust5_TrngData$Satisfaction),predicted.prob)
```

Hide

```
predicted.class=ifelse(predicted.prob>0.3,'1','0')
```

Hide

```
#confusion matrix on Training Data
table(as.factor(Clust5_TrngData$Satisfaction),predicted.class)
#accuracy on Training Data
sum(diag(table(as.factor(Clust5_TrngData$Satisfaction),predicted.class)))/nrow(Clust5_TrngDat
a)
#TNR
table(as.factor(Clust5_TrngData$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(Clust
5_TrngData$Satisfaction),predicted.class)[1,])
#TPR
table(as.factor(Clust5_TrngData$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(Clust
5_TrngData$Satisfaction),predicted.class)[2,])
#AUC
roc(Clust5_TrngData$Satisfaction,predicted.prob)
plot(roc(Clust5_TrngData$Satisfaction,predicted.prob))
```

Hide

```
predicted.prob=predict(clust5.xgb.fit,as.matrix(data.matrix(Clust5_TestData[,-c(1,10,25,2
6)])))
predicted.class=ifelse(predicted.prob>0.3,'1','0')
```

Hide

```
#confusion matrix on Test Data
table(as.factor(Clust5_TestData$Satisfaction),predicted.class)
#accuracy on Test Data
sum(diag(table(as.factor(Clust5_TestData$Satisfaction),predicted.class)))/nrow(Clust5_TestDat
a)
#TNR
table(as.factor(Clust5_TestData$Satisfaction),predicted.class)[1,1]/sum(table(as.factor(Clust
5_TestData$Satisfaction),predicted.class)[1,])
#TPR
table(as.factor(Clust5_TestData$Satisfaction),predicted.class)[2,2]/sum(table(as.factor(Clust
5_TestData$Satisfaction),predicted.class)[2,])
#AUC
roc(Clust5_TestData$Satisfaction,predicted.prob)
plot(roc(Clust5_TestData$Satisfaction,predicted.prob))
```

Hide

```
xgb.importance(model = clust5.xgb.fit)
xgb.plot.importance(xgb.importance(model = clust5.xgb.fit))
```

Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$clusterid=='Other' & Cleansed.Data$Satisfactio
n=='satisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Other customers - satisfied')
```

Hide

```
a=function_heatmap(Cleansed.Data[Cleansed.Data$clusterid=='Other' & Cleansed.Data$Satisfactio
n=='dissatisfied',])
a+labs(y=element_blank(),x=element_blank(),title = 'Other customers - dissatisfied')
```

Random forest model to study the variable importance:

Hide

```
Clust5_TrngData$Satisfaction=ifelse(Clust5_TrngData$Satisfaction==1,'satisfied','dissatisfie
d')
Clust5_TrngData$Satisfaction=as.factor(Clust5_TrngData$Satisfaction)
```

Hide

```
Clust5_TestData$Satisfaction=ifelse(Clust5_TestData$Satisfaction==1,'satisfied','dissatisfie
d')
Clust5_TestData$Satisfaction=as.factor(Clust5_TestData$Satisfaction)
```

Hide

```
set.seed(5000)
clust5_rndfor = randomForest(Satisfaction~.,data = Clust5_TrngData[-c(1,25,26)],ntree=151,mtr
y=6,nodesize=10,importance=T)
```

Hide

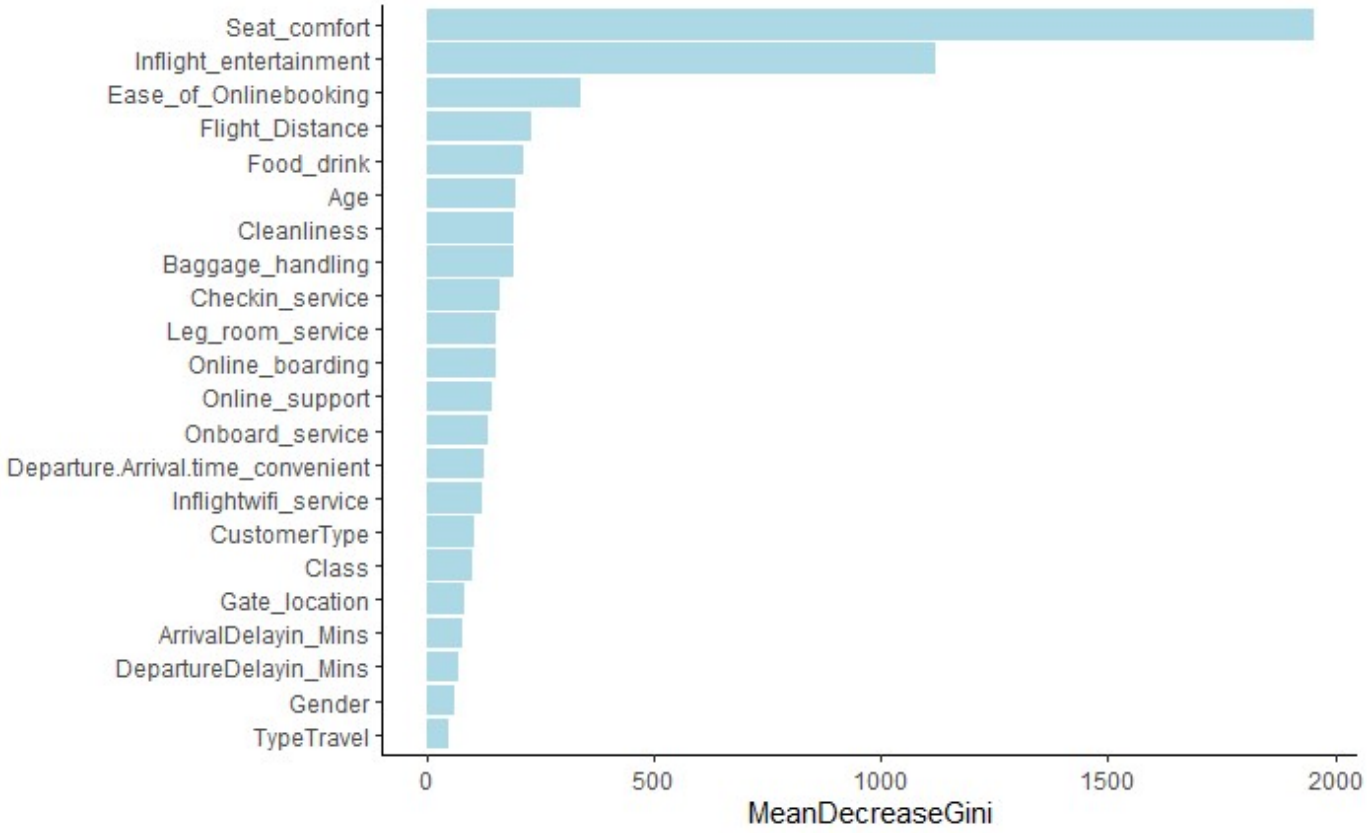
```
plot(clust5_rndfor)
legend("topright", c("OOB", "0", "1"), text.col=1:6, lty=1:3, col=1:3)
```

Hide

```
trndfor = tuneRF(x=Clust5_TrngData[-c(1,10,25,26)],
                 y=Clust5_TrngData$Satisfaction,
                 mtryStart = 4,
                 stepFactor = 1.5,
                 ntreeTry = 151,
                 trace = T,
                 plot = T,
                 importance=T,
                 doBest = T)
```

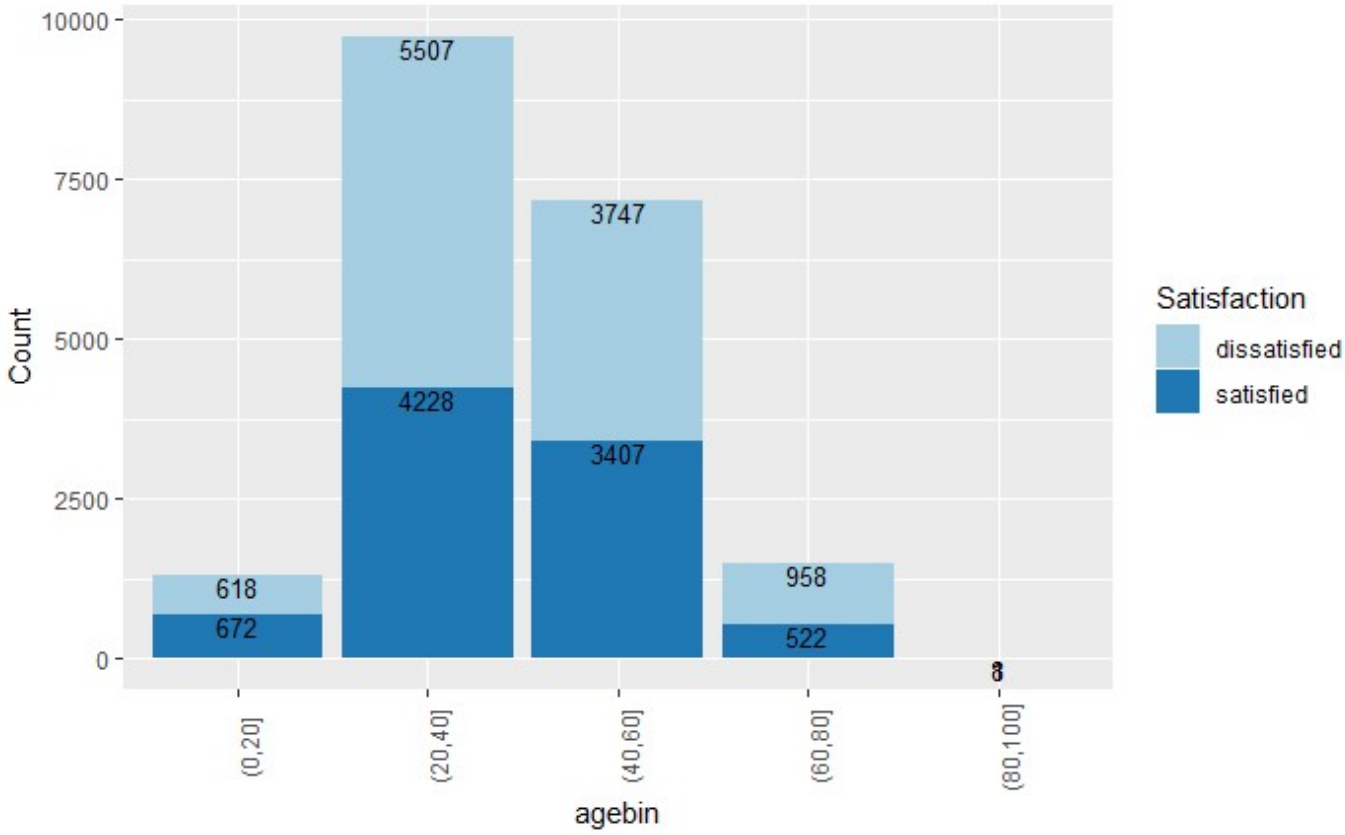
Hide

```
a=as.data.frame(clust5_rndfor$importance)
a$features=row.names(a)
ggplot(data = a)+aes(x=reorder(features,MeanDecreaseGini),MeanDecreaseGini)+geom_bar(stat =
'identity',fill='light blue')+coord_flip()+theme_classic()+theme(axis.title.y = element_blank
())
```



Hide

```
ggplot(data = Cleansed.Data[Cleansed.Data$clusterid=='Other',])+aes(x=agebin,fill=Satisfactio
n,y=..count..)+geom_bar(position = 'stack')+labs(y='Count')+geom_text(aes(label=..count..),st
at = 'count',position='stack', vjust=1,size=3.5)+scale_fill_brewer(palette="Paired")+theme(ax
is.text.x=element_text(angle = 90))
```


10000
7500
5000
2500
0
Count
5507
3747
4228
3407
618
672
958
522
8
(0,20]
(20,40]
(40,60]
(60,80]
(80,100]
agebin
Satisfaction
dissatisfied
satisfied