```
install.packages('tidyverse')
```

```
## Installing tidyverse [1.3.1] ...
##  OK [linked cache]
```

```
install.packages('yardstick')
```

```
## Installing yardstick [0.0.9] ...
##  OK [linked cache]
```

```
install.packages('pROC')
```

```
## Installing pROC [1.18.0] ...
##  OK [linked cache]
```

National Accessibility Case Study

Project Brief

Congratulations, you have landed your first job as a data scientist at National Accessibility! National Accessibility currently installs wheelchair ramps for office buildings and schools. However, the marketing manager wants the company to start installing ramps for event venues as well. According to a new survey, approximately 40% of event venues are not wheelchair accessible. However, it is not easy to know whether a venue already has a ramp installed.

The marketing manager would like to know whether you can develop a model to predict whether an event venue has a wheelchair ramp. To help you with this, he has provided you with a dataset of London venues. This data includes whether the venue has a ramp.

It is a waste of time to contact venues that already have a ramp installed, and it also looks bad for the company. Therefore, it is especially important to exclude locations that already have a ramp. Ideally, at least two-thirds of venues predicted to be without a ramp should not have a ramp.

You will need to present your findings in two formats:

- First, you will need to present your findings to the marketing manager via a 10 minute oral presentation. The owner has no technical data science background.
- You will also need to submit a technical report to your manager, who does have a strong technical data science background.

The data you will use for this analysis can be accessed here: `"data/event_venues.csv"`

1. Data Preparation

The goal of this section is to understand the structure of the variables and find potential correlations for model fitting. This section is especially interested in finding variables that might correlate with the existence of ramps in the events.

1.1 EDA

```
# Clean the environment
#rm(list=ls())

# Load the data
df <- read.csv("data/event_venues.csv")

# Overview of the structure of variables
head(df)
```

```
##              venue_name Loud.music...events Venue.provides.alcohol Wi.Fi
## 1 techspace aldgate east               False                      0  True
## 2      green rooms hotel                True                      1  True
## 3  148 leadenhall street               False                      0  True
## 4            conway hall               False                      0  True
## 5      gridiron building               False                      0  True
## 6 kimpton fitzroy london                True                      1  True
##   supervenue U.Shaped_max max_standing Theatre_max Promoted...ticketed.events
## 1      False     35.04545            0    112.7159                      False
## 2      False     40.00000          120     80.0000                       True
## 3      False     35.04545            0    112.7159                      False
## 4      False     35.04545           60     60.0000                      False
## 5      False     35.04545            0    112.7159                      False
## 6      False      6.00000            0    112.7159                       True
##   Wheelchair.accessible
## 1                 False
## 2                 False
## 3                 False
## 4                 False
## 5                 False
## 6                 False
```

```
str(df)
```

```
## 'data.frame':    3910 obs. of  10 variables:
##  $ venue_name               : chr  "techspace aldgate east" "green rooms hotel" "148 leadenhall str
##  $ Loud.music...events      : chr  "False" "True" "False" "False" ...
##  $ Venue.provides.alcohol   : int  0 1 0 0 0 1 0 1 0 1 ...
##  $ Wi.Fi                    : chr  "True" "True" "True" "True" ...
##  $ supervenue               : chr  "False" "False" "False" "False" ...
##  $ U.Shaped_max             : num  35 40 35 35 35 ...
##  $ max_standing             : int  0 120 0 60 0 0 0 200 0 180 ...
##  $ Theatre_max              : num  113 80 113 60 113 ...
##  $ Promoted...ticketed.events: chr  "False" "True" "False" "False" ...
##  $ Wheelchair.accessible    : chr  "False" "False" "False" "False" ...
```

```
# Format the character type to be logical: help calculating the mean
df$Wheelchair.accessible <- as.logical(df$Wheelchair.accessible)
df$Loud.music...events <- as.logical(df$Loud.music...events)
df$Venue.provides.alcohol <- as.logical(df$Venue.provides.alcohol)
df$Wi.Fi <- as.logical(df$Wi.Fi)
df$supervenue <- as.logical(df$supervenue)
df$Promoted...ticketed.events <- as.logical(df$Promoted...ticketed.events)
```

```r
# Mean of dummy variables
lapply(df, mean)
```

```
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA
```

```
## $venue_name
## [1] NA
##
## $Loud.music...events
## [1] 0.3588235
##
## $Venue.provides.alcohol
## [1] 0.7132992
##
## $Wi.Fi
## [1] 0.9294118
##
## $supervenue
## [1] 0.06240409
##
## $U.Shaped_max
## [1] 35.04545
##
## $max_standing
## [1] 114.2036
##
## $Theatre_max
## [1] 112.7159
##
## $Promoted...ticketed.events
## [1] 0.3790281
##
## $Wheelchair.accessible
## [1] 0.5
```

```r
# Structure of dummy and numeric variables
summary(df)
```

```
##   venue_name        Loud.music...events Venue.provides.alcohol   Wi.Fi
##  Length:3910        Mode :logical       Mode :logical          Mode :logical
##  Class :character   FALSE:2507          FALSE:1121             FALSE:276
##  Mode  :character   TRUE :1403          TRUE :2789             TRUE :3634
##
##
##
##  supervenue       U.Shaped_max      max_standing     Theatre_max
##  Mode :logical   Min.   :   1.00   Min.   :   0.0   Min.   :   1.0
##  FALSE:3666      1st Qu.:  35.05   1st Qu.:   0.0   1st Qu.:  80.0
##  TRUE :244       Median :  35.05   Median :  50.0   Median : 112.7
##                  Mean   :  35.05   Mean   : 114.2   Mean   : 112.7
##                  3rd Qu.:  35.05   3rd Qu.: 120.0   3rd Qu.: 112.7
```

```
##                  Max.   :2520.00   Max.   :7500.0   Max.   :4000.0
##  Promoted...ticketed.events Wheelchair.accessible
##  Mode :logical              Mode :logical
##  FALSE:2428                 FALSE:1955
##  TRUE :1482                 TRUE :1955
##
##
##
```

In the EDA, we found that there are 3910 observations and 10 variables in the dataset. Most of the variables are converted to logical variables to facilitate the understanding of their structures, including the presence of loud music, alcohol, WIFI, super venue, promotion, and wheelchair accessibility. The capacity variables are kept numeric. One advantage of this dataset is that exactly half of the data are wheelchair accessible, so analysis can be easily performed. Other variables are not so well distributed, such as only 7% of the events do not provide WIFI.

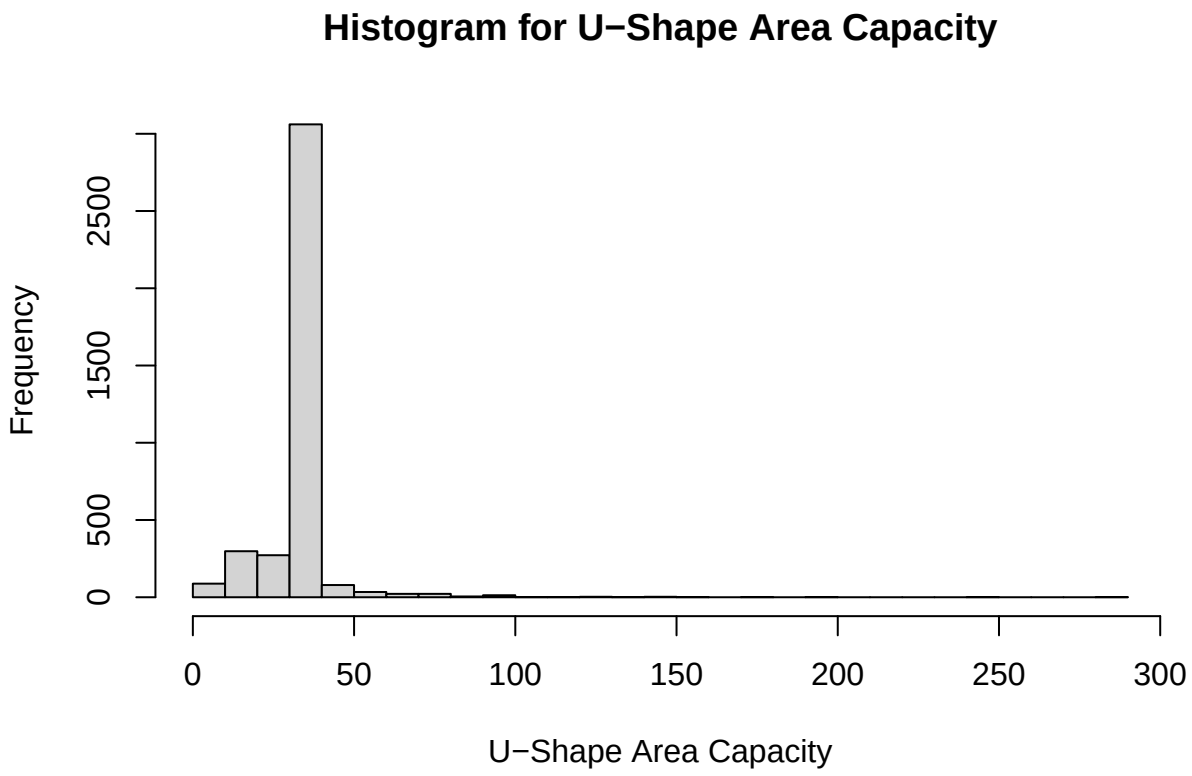
1.2 Visualizations

```
# The relationship between the dependent variable and numeric variables


# Total capacity of U-shaped portion
# Since only 1 events have total capacity of U-shaped portion exceeding 500,
# we treat them as outliers and plot without them
length(df$U.Shaped_max[df$U.Shaped_max > 500])
```

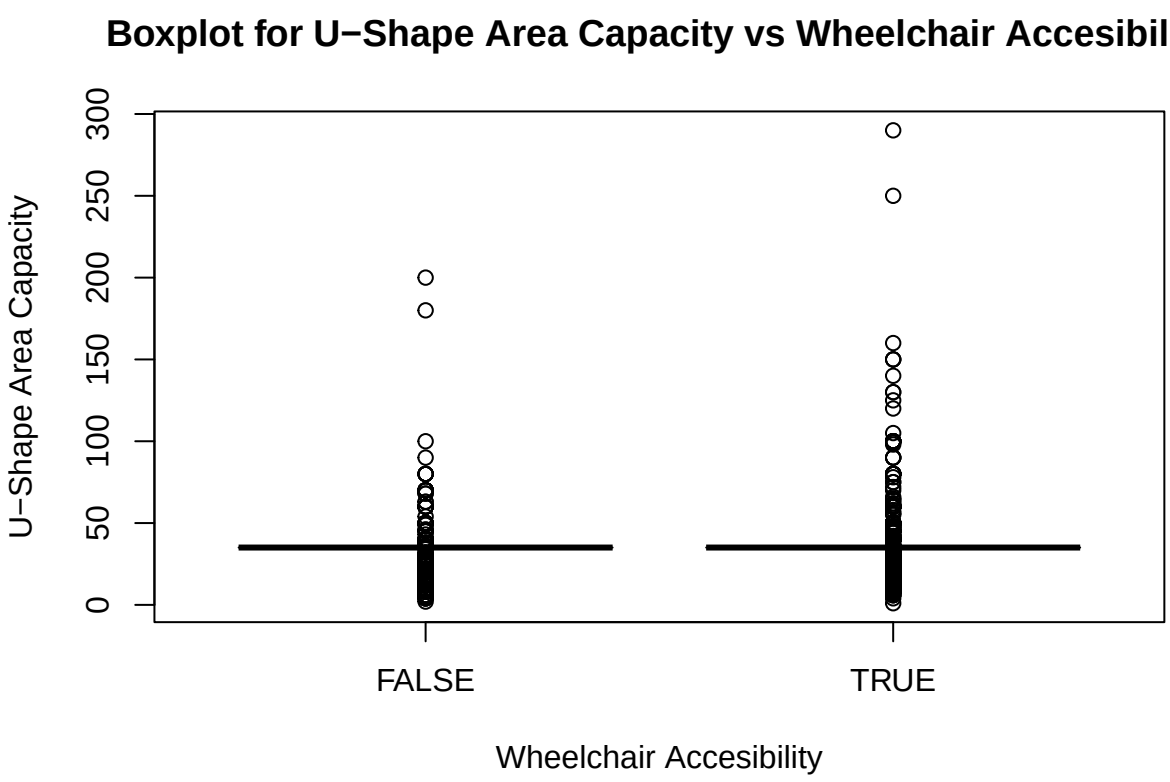
1.2.1 Capacity and wheelchair accessibility

```
## [1] 2
```

```
hist(df$U.Shaped_max[df$U.Shaped_max <= 500],breaks = 30,
     main="Histogram for U-Shape Area Capacity", xlab = "U-Shape Area Capacity")
```




```
boxplot(df$U.Shaped_max[df$U.Shaped_max <= 500] ~
        df$Wheelchair.accessible[df$U.Shaped_max <= 500],
        main="Boxplot for U-Shape Area Capacity vs Wheelchair Accesibility",
        xlab = "Wheelchair Accesibility", ylab =  "U-Shape Area Capacity")
```



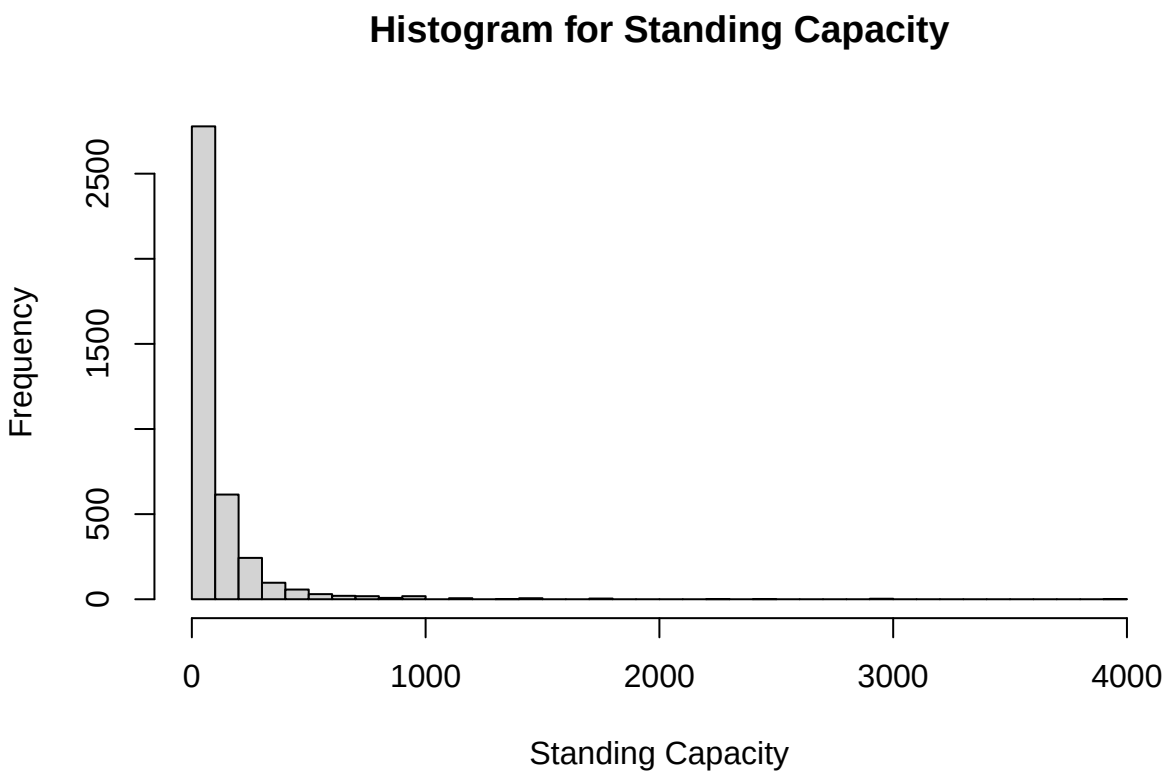

```
length(df$U.Shaped_max[df$U.Shaped_max > 35.001 & df$U.Shaped_max <35.1])
```

```
## [1] 2920
```

```
# In the dataset, one value 35.04545 appears frequently, also as shown in histogram.
# It is not an integer and appears about 3/4 of the time, which is abnormal.
# By using summary statistics above, this is equal to the mean.
# This means original data had some missing values. Need to be careful
# about this variable for prediction.


# Total standing capacity
# Since only 3 events have total standing capacity exceeding 4000,
# we treat them as outliers and plot without them
length(df$max_standing[df$max_standing > 4000])
```

```
## [1] 3
```

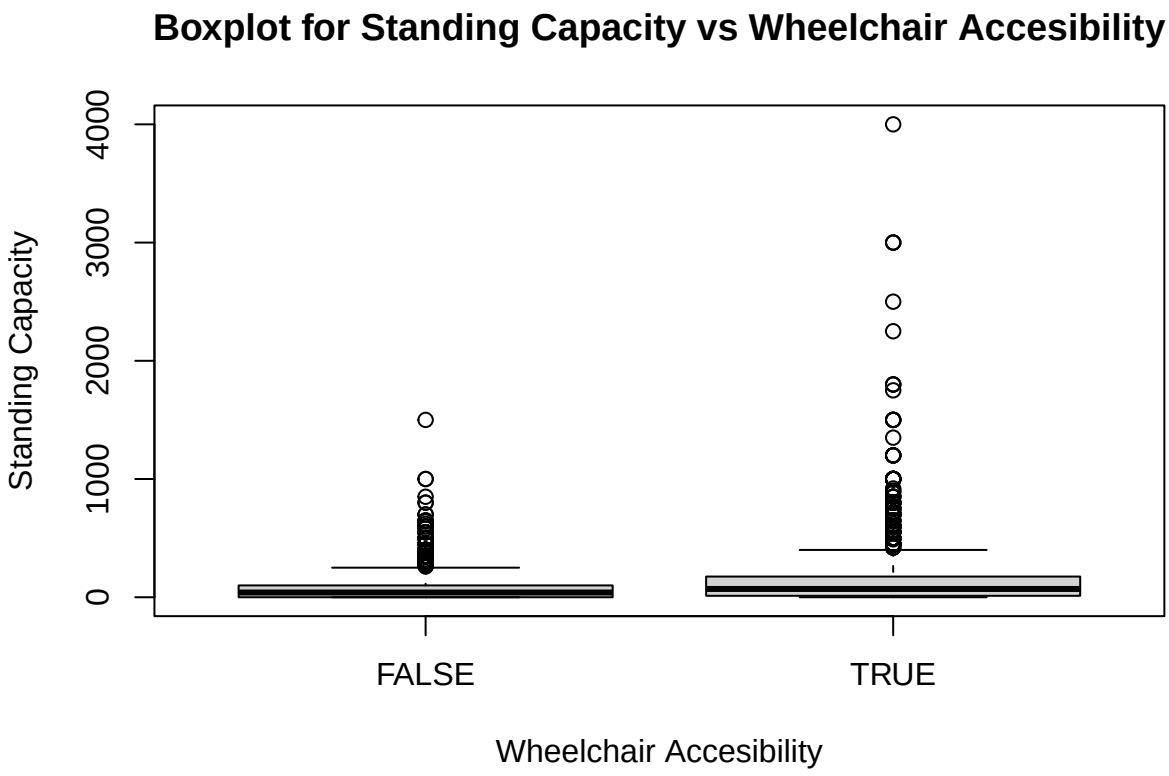
```
hist(df$max_standing[df$max_standing <= 4000],breaks = 30,
     main="Histogram for Standing Capacity", xlab = "Standing Capacity")
```




```
# Zero inflated
length(df$max_standing[df$max_standing == 0])
```

```
## [1] 1000
```

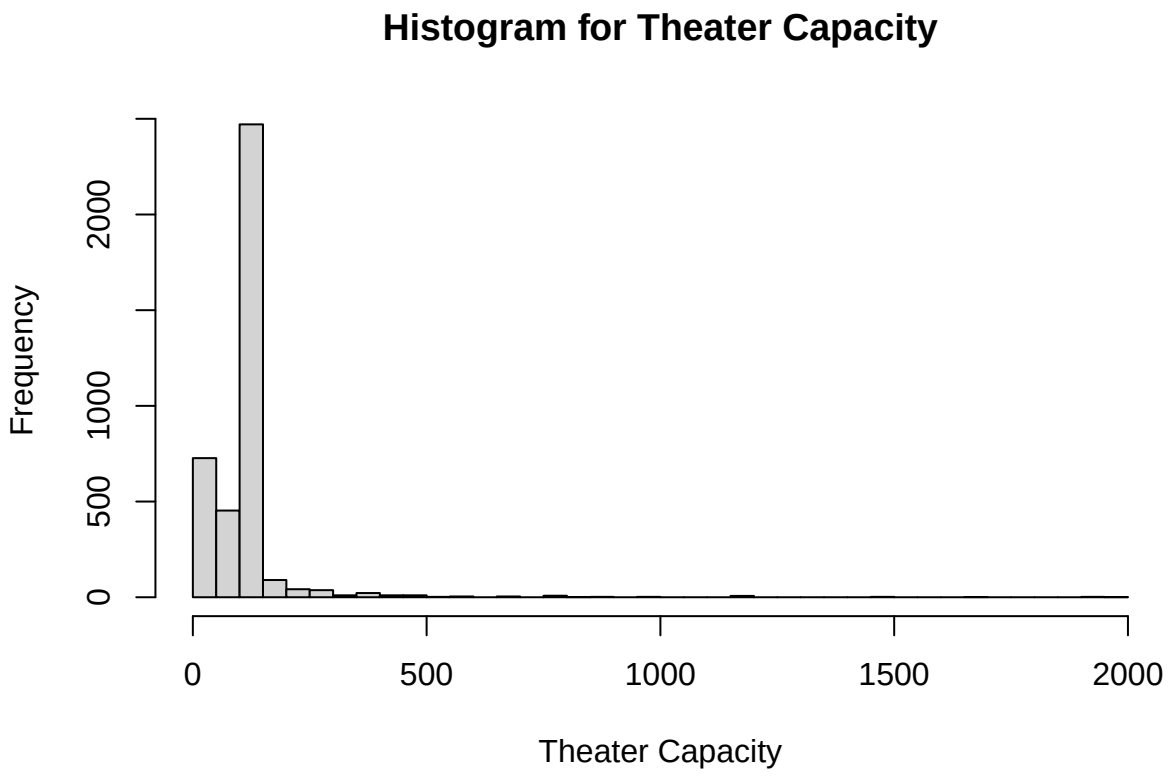
```
boxplot(df$max_standing[df$max_standing <= 4000] ~
        df$ Wheelchair.accessible[df$max_standing <= 4000],
        main="Boxplot for Standing Capacity vs Wheelchair Accesibility",
        xlab = "Wheelchair Accesibility", ylab =  "Standing Capacity")
```



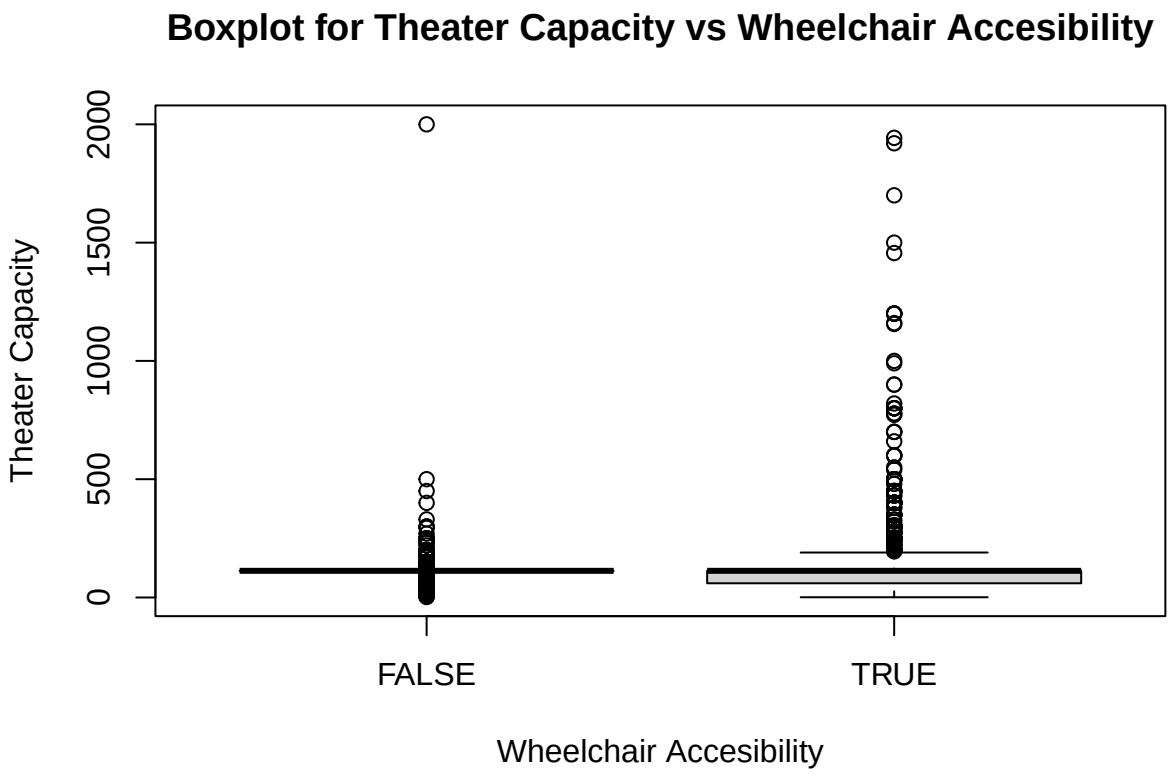

```
# Total capacity for theater

# Since only 2 events have total standing capacity exceeding 2000,
# we treat them as outliers and plot without them
length(df$Theatre_max[df$Theatre_max > 2000])
```

```
## [1] 2
```

```
hist(df$Theatre_max[df$Theatre_max <= 2000],breaks = 30,
       main="Histogram for Theater Capacity", xlab = "Theater Capacity")
```



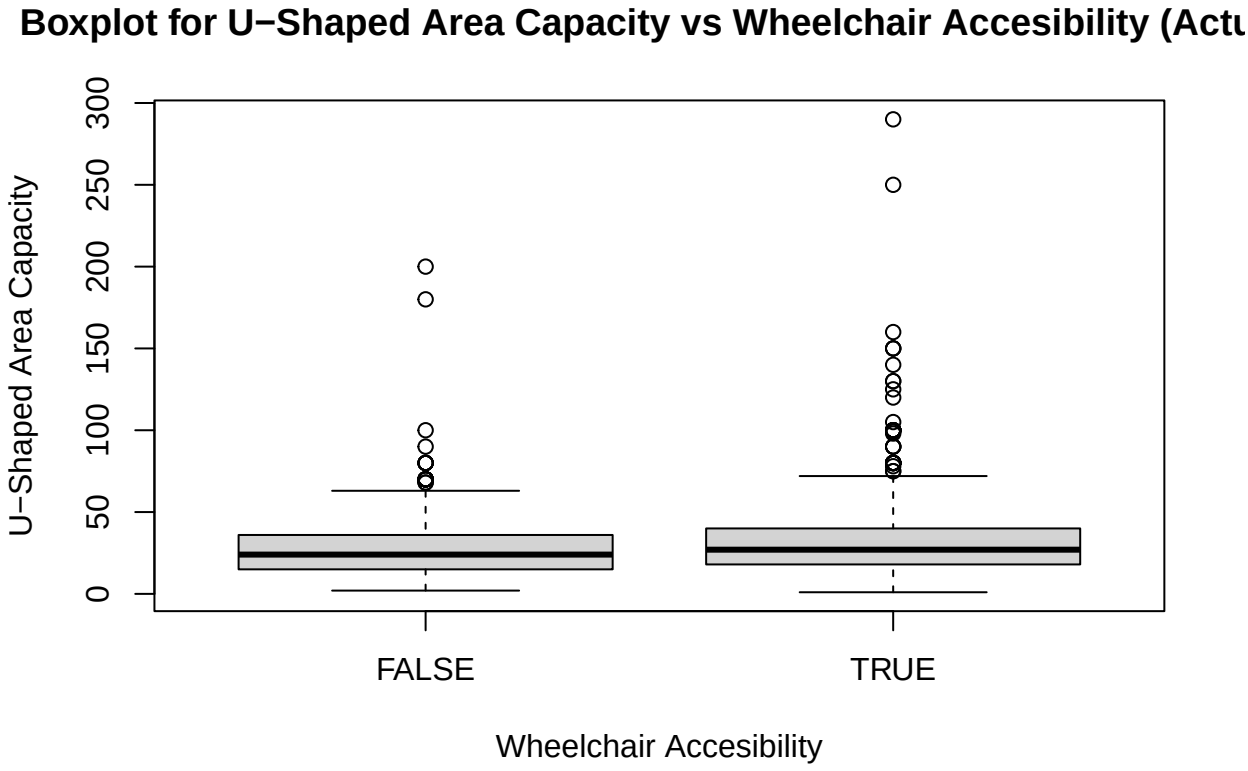

```
boxplot(df$Theatre_max[df$Theatre_max <= 2000] ~
        df$Wheelchair.accessible[df$Theatre_max <= 2000],
        main="Boxplot for Theater Capacity vs Wheelchair Accesibility",
        xlab = "Wheelchair Accesibility", ylab =  "Theater Capacity")
```



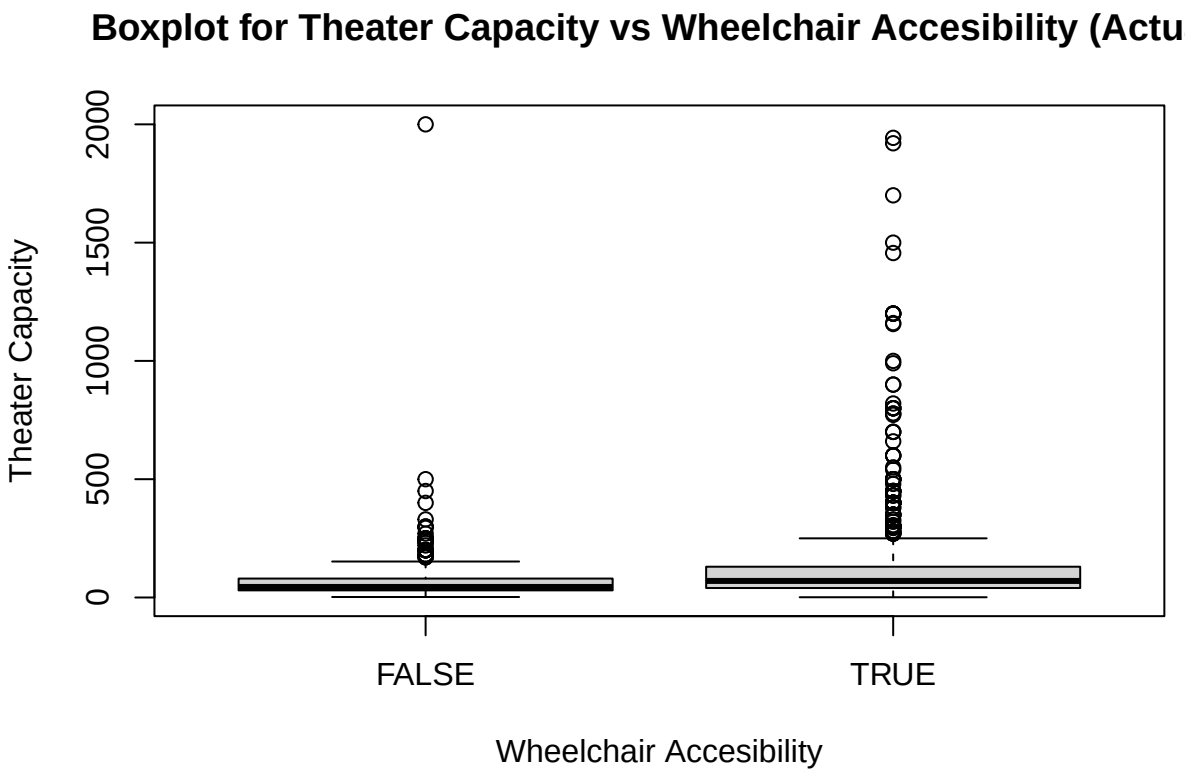

```
length(df$Theatre_max[df$Theatre_max > 112.7 & df$Theatre_max <  112.8])
```

```
## [1] 2284
```

```
# Similar as in total capacity of U-shaped portion, 58% of data are the mean.
# Careful when interpreting


# If we don't consider the ones with missing actual values
boxplot(df$U.Shaped_max[df$U.Shaped_max < 35.001 |
        df$U.Shaped_max > 35.1 & df$U.Shaped_max <= 500] ~
        df$Wheelchair.accessible[df$U.Shaped_max < 35.001 |
        df$U.Shaped_max > 35.1 & df$U.Shaped_max <= 500],
        main="Boxplot for U-Shaped Area Capacity vs Wheelchair Accesibility (Actual)",
        xlab = "Wheelchair Accesibility", ylab =  "U-Shaped Area Capacity")
```



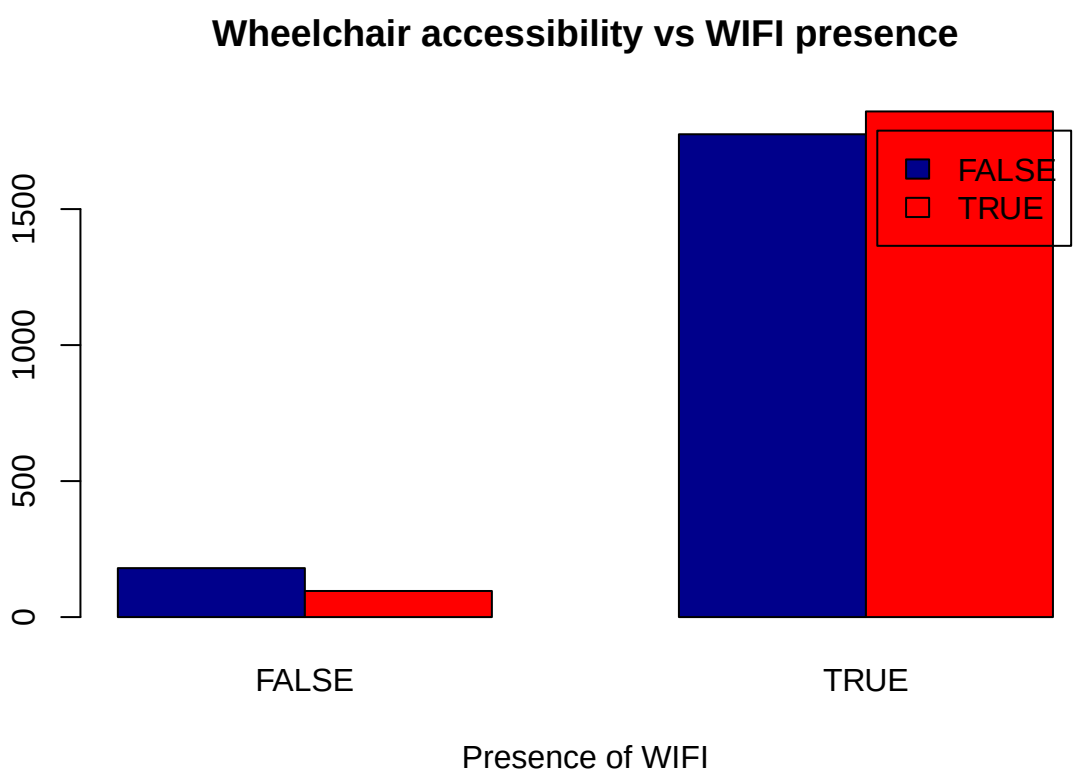

```
boxplot(df$Theatre_max[df$Theatre_max < 112.7 |
        df$Theatre_max > 112.8 & df$Theatre_max <= 2000] ~
        df$Wheelchair.accessible[df$Theatre_max < 112.7 |
        df$Theatre_max > 112.8 & df$Theatre_max <= 2000],
        main="Boxplot for Theater Capacity vs Wheelchair Accesibility (Actual)",
        xlab = "Wheelchair Accesibility", ylab =  "Theater Capacity" )
```



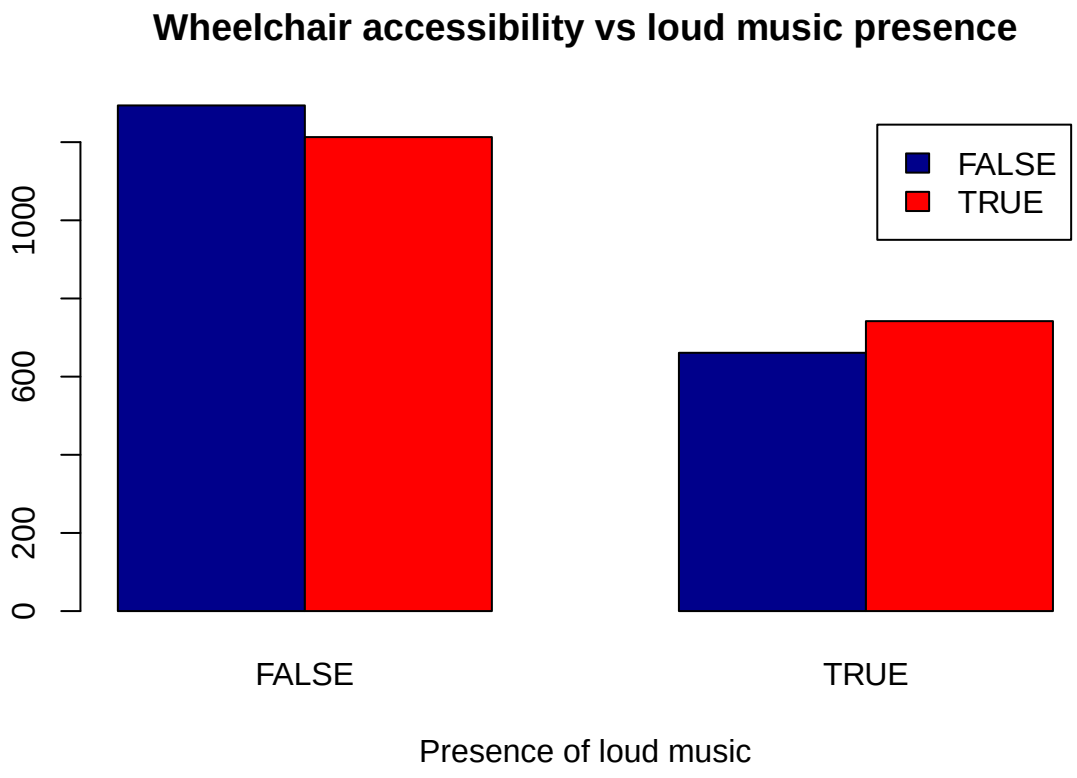
Given the visualizations, only total standing capacity seems to make difference. U-shaped and theater capacities do not exhibit a clear pattern for wheelchair accessibility, due to large amount of missing actual values. If we only consider true values, all three variable show that larger capacity in any regard is associated with wheelchairs accessibility.

1.2.2 Other factors and wheelchair accessibility Now we consider the relationship between the dependent variable and dummy variables.

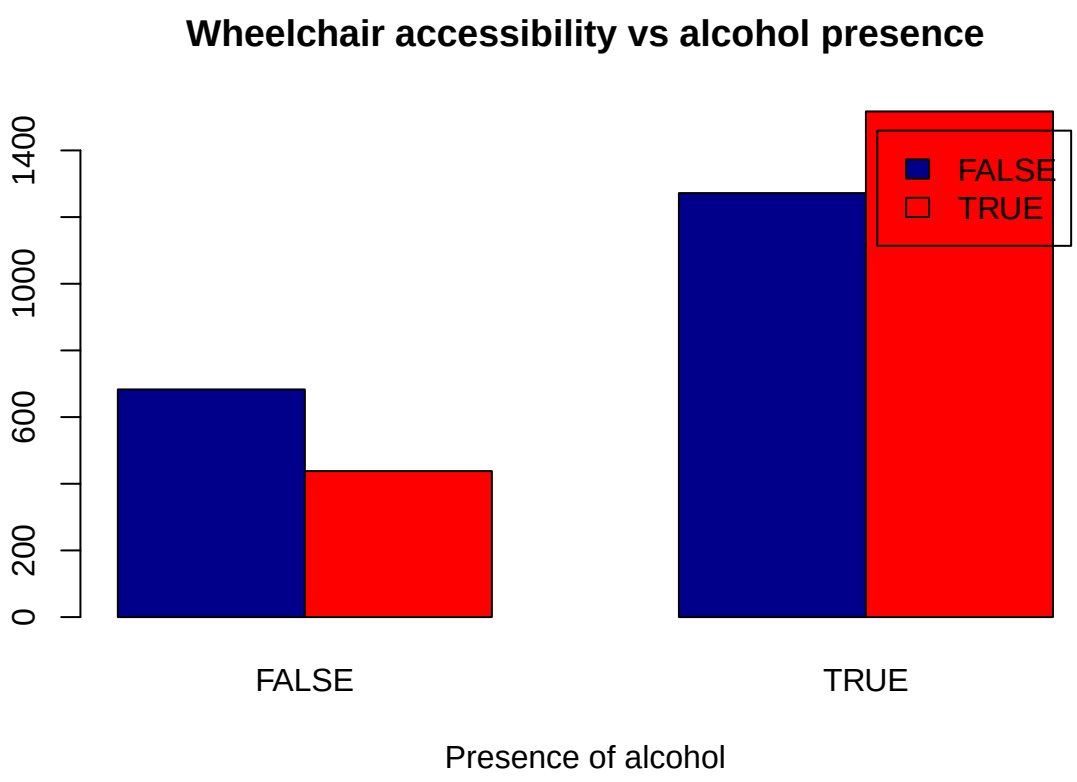
```
Wh_wifi <- table(df$Wheelchair.accessible, df$Wi.Fi)
barplot(Wh_wifi, main="Wheelchair accessibility vs WIFI presence",
  xlab="Presence of WIFI", col=c("darkblue","red"),
  legend = rownames(Wh_wifi), beside=TRUE)
```



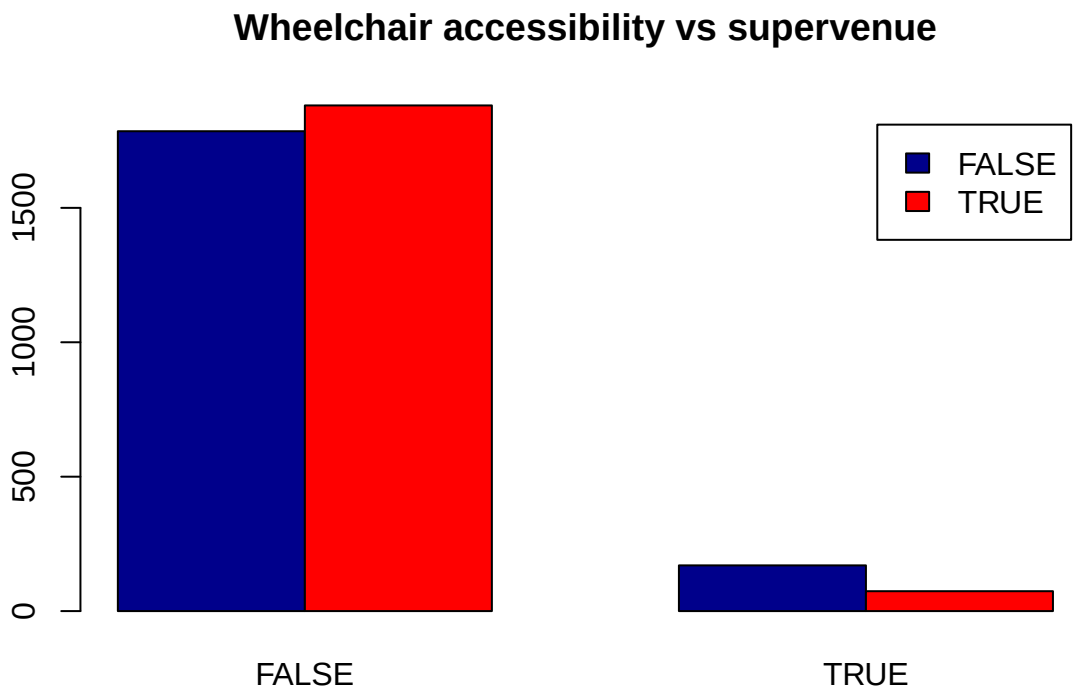

```
Wh_loud <- table(df$Wheelchair.accessible, df$Loud.music...events)
barplot(Wh_loud, main="Wheelchair accessibility vs loud music presence",
  xlab="Presence of loud music", col=c("darkblue","red"),
  legend = rownames(Wh_loud), beside=TRUE)
```



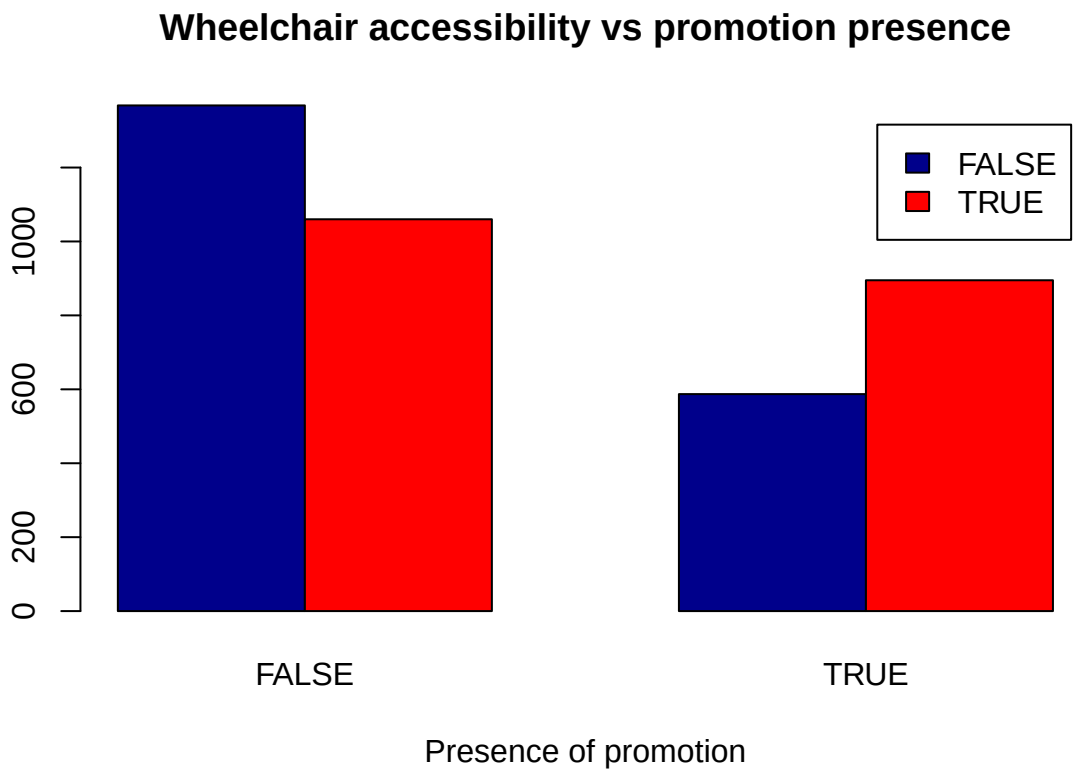

```
Wh_alcohol <- table(df$Wheelchair.accessible, df$Venue.provides.alcohol)
barplot(Wh_alcohol, main="Wheelchair accessibility vs alcohol presence",
  xlab="Presence of alcohol", col=c("darkblue","red"),
  legend = rownames(Wh_alcohol), beside=TRUE)
```




```
Wh_supervenue <- table(df$Wheelchair.accessible, df$supervenue)
barplot(Wh_supervenue, main="Wheelchair accessibility vs supervenue",
  xlab="Whether the venue is supervenue", col=c("darkblue","red"),
  legend = rownames(Wh_supervenue), beside=TRUE)
```




```
Wh_promote <- table(df$Wheelchair.accessible, df$Promoted...ticketed.events)
barplot(Wh_promote, main="Wheelchair accessibility vs promotion presence",
  xlab="Presence of promotion", col=c("darkblue","red"),
  legend = rownames(Wh_promote), beside=TRUE)
```



Based on visualizations, the presence of WIFI, loud music, alcohol, and promotional tickets is associated with better wheelchair accessibility. If the venue is a supervenue, it is less likely to have wheelchair accessibility.

2. Model

The goal of this section is to fit a logistic model on the data to be able to make predictions for ramp availability (wheelchair accessibility) based on other variables. It is particularly concerned that at least two-thirds of venues predicted to be without a ramp should not have a ramp. The model is evaluated accordingly: the focus is on the negative predictive value (npv), while not sacrificing too much accuracy.

2.1 Model Fitting

I fit two models because of the incomplete information in two variables, U.Shaped_max and Theatre_max. The purpose is to be able to make predictions in a more precise manner. If we do have accurate information for those two variables, it is desirable that we use the second model. Otherwise, we can input the mean values for those two variables and make a prediction using the first model.

Model with all values First I fit a model with all values in the dataset. Step-wise regression is used to find the optimal regression model.

```
library(tidyverse)
```

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

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

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

```r
# Specify a null model with no predictors
null_model <- glm(Wheelchair.accessible ~ 1, data = df, family = "binomial")

# Specify the full model using all of the potential predictors
full_model <- glm(Wheelchair.accessible ~ . - venue_name , data = df, family = "binomial")

# Use a forward stepwise algorithm to build a parsimonious model
step_model <- step(null_model, scope = list(lower = null_model, upper = full_model), direction = "forwa
```

```
## Start:  AIC=5422.41
## Wheelchair.accessible ~ 1
##
##                              Df Deviance    AIC
## + max_standing                1   5288.3 5292.3
## + Promoted...ticketed.events  1   5316.8 5320.8
## + Venue.provides.alcohol      1   5344.9 5348.9
## + Theatre_max                 1   5374.9 5378.9
## + supervenue                  1   5379.1 5383.1
## + Wi.Fi                       1   5392.5 5396.5
## + Loud.music...events         1   5413.1 5417.1
## + U.Shaped_max                1   5416.7 5420.7
## <none>                            5420.4 5422.4
##
## Step:  AIC=5292.28
## Wheelchair.accessible ~ max_standing
##
##                              Df Deviance    AIC
## + Promoted...ticketed.events  1   5210.2 5216.2
## + Venue.provides.alcohol      1   5248.6 5254.6
## + Wi.Fi                       1   5251.0 5257.0
## + supervenue                  1   5260.3 5266.3
## + Theatre_max                 1   5286.1 5292.1
## <none>                            5288.3 5292.3
## + U.Shaped_max                1   5287.9 5293.9
## + Loud.music...events         1   5288.1 5294.1
##
## Step:  AIC=5216.22
## Wheelchair.accessible ~ max_standing + Promoted...ticketed.events
##
##                            Df Deviance    AIC
## + Wi.Fi                     1   5174.8 5182.8
## + Venue.provides.alcohol    1   5174.8 5182.8
## + supervenue                1   5177.7 5185.7
## + Loud.music...events       1   5202.1 5210.1
```

```
## <none>                         5210.2 5216.2
## + Theatre_max             1   5208.7 5216.7
## + U.Shaped_max            1   5210.1 5218.1
##
## Step:  AIC=5182.78
## Wheelchair.accessible ~ max_standing + Promoted...ticketed.events +
##     Wi.Fi
##
##                          Df Deviance    AIC
## + supervenue              1   5140.3 5150.3
## + Venue.provides.alcohol  1   5140.6 5150.6
## + Loud.music...events     1   5165.9 5175.9
## + Theatre_max             1   5172.3 5182.3
## <none>                         5174.8 5182.8
## + U.Shaped_max            1   5174.7 5184.7
##
## Step:  AIC=5150.28
## Wheelchair.accessible ~ max_standing + Promoted...ticketed.events +
##     Wi.Fi + supervenue
##
##                          Df Deviance    AIC
## + Venue.provides.alcohol  1   5112.3 5124.3
## + Loud.music...events     1   5131.3 5143.3
## + Theatre_max             1   5137.7 5149.7
## <none>                         5140.3 5150.3
## + U.Shaped_max            1   5140.2 5152.2
##
## Step:  AIC=5124.3
## Wheelchair.accessible ~ max_standing + Promoted...ticketed.events +
##     Wi.Fi + supervenue + Venue.provides.alcohol
##
##                        Df Deviance    AIC
## + Loud.music...events  1   5093.7 5107.7
## + Theatre_max          1   5109.2 5123.2
## <none>                     5112.3 5124.3
## + U.Shaped_max         1   5112.1 5126.1
##
## Step:  AIC=5107.73
## Wheelchair.accessible ~ max_standing + Promoted...ticketed.events +
##     Wi.Fi + supervenue + Venue.provides.alcohol + Loud.music...events
##
##                Df Deviance    AIC
## + Theatre_max   1   5090.9 5106.9
## <none>              5093.7 5107.7
## + U.Shaped_max  1   5093.4 5109.4
##
## Step:  AIC=5106.93
## Wheelchair.accessible ~ max_standing + Promoted...ticketed.events +
##     Wi.Fi + supervenue + Venue.provides.alcohol + Loud.music...events +
##     Theatre_max
##
##                Df Deviance    AIC
## <none>              5090.9 5106.9
## + U.Shaped_max  1   5090.7 5108.7
```

```
summary(step_model)
```

```
##
## Call:
## glm(formula = Wheelchair.accessible ~ max_standing + Promoted...ticketed.events +
##     Wi.Fi + supervenue + Venue.provides.alcohol + Loud.music...events +
##     Theatre_max, family = "binomial", data = df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -5.075  -1.137  -0.227   1.123   2.007
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   -1.4912032  0.1553928  -9.596  < 2e-16 ***
## max_standing                   0.0018704  0.0002836   6.595 4.27e-11 ***
## Promoted...ticketed.eventsTRUE  0.7089227  0.0743478   9.535  < 2e-16 ***
## Wi.FiTRUE                      0.8284301  0.1380878   5.999 1.98e-09 ***
## supervenueTRUE                -0.7499639  0.1478555  -5.072 3.93e-07 ***
## Venue.provides.alcoholTRUE     0.4831852  0.0787495   6.136 8.48e-10 ***
## Loud.music...eventsTRUE       -0.3325564  0.0781483  -4.255 2.09e-05 ***
## Theatre_max                    0.0007479  0.0004655   1.607    0.108
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5420.4  on 3909  degrees of freedom
## Residual deviance: 5090.9  on 3902  degrees of freedom
## AIC: 5106.9
##
## Number of Fisher Scoring iterations: 5
```

Model with only true values Then I fit a model with only true values (in terms of two variables, namely U.Shaped_max and Theatre_max) in the dataset. This model contains roughly 1/4 of the observations in the original dataset. Step-wise regression is used to find the optimal regression model.

```
df1 = df[df$Theatre_max < 112.7 | df$Theatre_max > 112.8,]
df2 = df1[df1$U.Shaped_max < 35.001 | df1$U.Shaped_max > 35.1, ]

# Specify a null model with no predictors
null_model1 <- glm(Wheelchair.accessible ~ 1,
                   data = df2, family = "binomial")

# Specify the full model using all of the potential predictors
full_model1 <- glm(Wheelchair.accessible ~ . - venue_name ,
                   data = df2, family = "binomial")

# Use a forward stepwise algorithm to build a parsimonious model
step_model1 <- step(null_model1,
                    scope = list(lower = null_model1, upper = full_model1),
                    direction = "forward")
```

```
## Start:  AIC=1089.1
## Wheelchair.accessible ~ 1
##
##                              Df Deviance    AIC
## + Theatre_max                 1   1050.8 1054.8
## + Venue.provides.alcohol      1   1065.1 1069.1
## + Promoted...ticketed.events  1   1074.2 1078.2
## + supervenue                  1   1074.2 1078.2
## + max_standing                1   1077.6 1081.6
## + Wi.Fi                       1   1080.2 1084.2
## + U.Shaped_max                1   1080.8 1084.8
## <none>                            1087.1 1089.1
## + Loud.music...events         1   1087.1 1091.1
##
## Step:  AIC=1054.8
## Wheelchair.accessible ~ Theatre_max
##
##                              Df Deviance    AIC
## + Venue.provides.alcohol      1   1036.8 1042.8
## + max_standing                1   1039.5 1045.5
## + Promoted...ticketed.events  1   1039.7 1045.7
## + Wi.Fi                       1   1040.6 1046.6
## + supervenue                  1   1041.0 1047.0
## + U.Shaped_max                1   1044.2 1050.2
## <none>                            1050.8 1054.8
## + Loud.music...events         1   1050.0 1056.0
##
## Step:  AIC=1042.77
## Wheelchair.accessible ~ Theatre_max + Venue.provides.alcohol
##
##                              Df Deviance    AIC
## + max_standing                1   1020.4 1028.4
## + Promoted...ticketed.events  1   1026.9 1034.9
## + Wi.Fi                       1   1027.6 1035.6
## + supervenue                  1   1028.3 1036.3
## + U.Shaped_max                1   1031.2 1039.2
## + Loud.music...events         1   1032.3 1040.3
## <none>                            1036.8 1042.8
##
## Step:  AIC=1028.39
## Wheelchair.accessible ~ Theatre_max + Venue.provides.alcohol +
##     max_standing
##
##                              Df Deviance    AIC
## + Promoted...ticketed.events  1   1009.3 1019.3
## + supervenue                  1   1010.8 1020.8
## + Wi.Fi                       1   1013.2 1023.2
## + U.Shaped_max                1   1018.0 1028.0
## + Loud.music...events         1   1018.0 1028.0
## <none>                            1020.4 1028.4
##
## Step:  AIC=1019.27
## Wheelchair.accessible ~ Theatre_max + Venue.provides.alcohol +
##     max_standing + Promoted...ticketed.events
```

```
##
##                       Df Deviance    AIC
## + Loud.music...events  1   998.35 1010.4
## + supervenue           1   999.22 1011.2
## + Wi.Fi                1  1003.77 1015.8
## + U.Shaped_max         1  1005.53 1017.5
## <none>                    1009.27 1019.3
##
## Step:  AIC=1010.35
## Wheelchair.accessible ~ Theatre_max + Venue.provides.alcohol +
##     max_standing + Promoted...ticketed.events + Loud.music...events
##
##                Df Deviance    AIC
## + supervenue    1   988.76 1002.8
## + Wi.Fi         1   993.43 1007.4
## + U.Shaped_max  1   993.79 1007.8
## <none>              998.35 1010.4
##
## Step:  AIC=1002.76
## Wheelchair.accessible ~ Theatre_max + Venue.provides.alcohol +
##     max_standing + Promoted...ticketed.events + Loud.music...events +
##     supervenue
##
##                Df Deviance    AIC
## + U.Shaped_max  1   984.27 1000.3
## + Wi.Fi         1   984.30 1000.3
## <none>              988.76 1002.8
##
## Step:  AIC=1000.27
## Wheelchair.accessible ~ Theatre_max + Venue.provides.alcohol +
##     max_standing + Promoted...ticketed.events + Loud.music...events +
##     supervenue + U.Shaped_max
##
##         Df Deviance     AIC
## + Wi.Fi  1   979.65  997.65
## <none>       984.27 1000.27
##
## Step:  AIC=997.65
## Wheelchair.accessible ~ Theatre_max + Venue.provides.alcohol +
##     max_standing + Promoted...ticketed.events + Loud.music...events +
##     supervenue + U.Shaped_max + Wi.Fi
```

```
summary(step_model1)
```

```
##
## Call:
## glm(formula = Wheelchair.accessible ~ Theatre_max + Venue.provides.alcohol +
##     max_standing + Promoted...ticketed.events + Loud.music...events +
##     supervenue + U.Shaped_max + Wi.Fi, family = "binomial", data = df2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7320  -1.1462   0.6482   0.8580   2.2630
##
```

```
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    -1.411190   0.556889  -2.534  0.01127 *
## Theatre_max                     0.016375   0.002912   5.623 1.88e-08 ***
## Venue.provides.alcoholTRUE      0.818116   0.182233   4.489 7.14e-06 ***
## max_standing                   -0.004361   0.001515  -2.879  0.00400 **
## Promoted...ticketed.eventsTRUE  0.790858   0.181278   4.363 1.28e-05 ***
## Loud.music...eventsTRUE        -0.629748   0.193627  -3.252  0.00114 **
## supervenueTRUE                 -1.065492   0.356027  -2.993  0.00276 **
## U.Shaped_max                   -0.009970   0.004316  -2.310  0.02088 *
## Wi.FiTRUE                       1.133509   0.536578   2.112  0.03465 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1087.10  on 869  degrees of freedom
## Residual deviance:  979.65  on 861  degrees of freedom
## AIC: 997.65
##
## Number of Fisher Scoring iterations: 5
```

Comparing the two models, we can find that a few variables are reliable in predicting wheelchair accessibility. The presence of alcohol, the event being promoted or ticketed are the presence of WIFI positively correlated with wheelchair accessibility; the venue as supervenue and the presence of loud music are negatively correlated with wheelchair accessibility. The capacity variables have mixing results and depend on the specific model. Therefore, in general, we would look for events where no alcohol, WIFI, or ticketing is present, as well as events where the venue is supervenue and where loud music is played.

2.2 Model Evaluation

The key to evaluate the two models is how good their predictions are when venues are not predicted to be with a ramp. ROC curves are also used to assist model evaluation.

Model with all values Threshold is set to 0.4 to control the False Negative rates, such that the result for negative predictive value will be higher, without sacrificing too much accuracy.

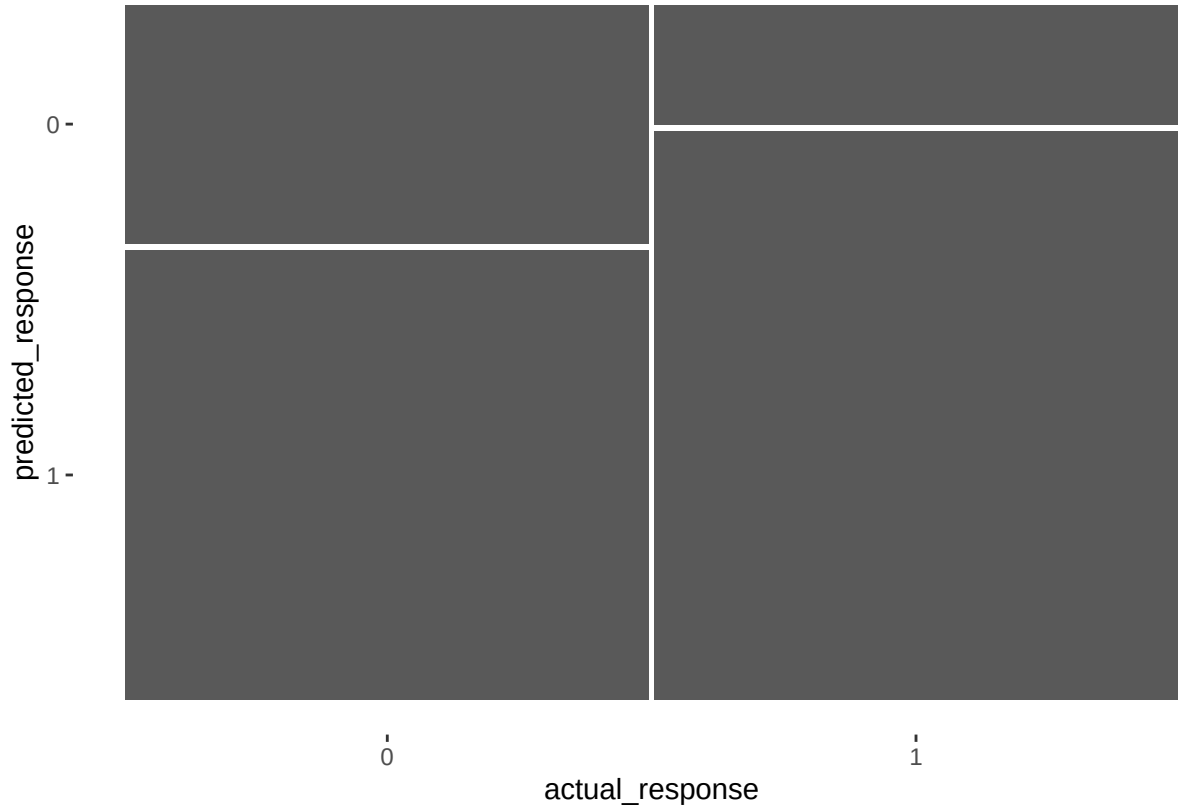
```
library(yardstick)
```

```
## For binary classification, the first factor level is assumed to be the event.
## Use the argument `event_level = "second"` to alter this as needed.
```

```
##
## Attaching package: 'yardstick'
```

```
## The following object is masked from 'package:readr':
##
##     spec
```

```
# From previous step
actual_response <- as.numeric(df$Wheelchair.accessible)
predicted_response <- round(fitted(step_model)+0.1)
outcomes <- table(predicted_response, actual_response)
confusion <- conf_mat(outcomes)

# "Automatically" plot the confusion matrix
autoplot(confusion)
```




```
# Get summary metrics
summary(confusion, event_level = "second")
```

```
## # A tibble: 13 x 3
##    .metric              .estimator .estimate
##    <chr>                <chr>          <dbl>
##  1 accuracy             binary         0.586
##  2 kap                  binary         0.173
##  3 sens                 binary         0.827
##  4 spec                 binary         0.346
##  5 ppv                  binary         0.558
##  6 npv                  binary         0.666
##  7 mcc                  binary         0.197
##  8 j_index              binary         0.173
##  9 bal_accuracy         binary         0.586
## 10 detection_prevalence binary         0.740
```

```
## 11 precision            binary         0.558
## 12 recall               binary         0.827
## 13 f_meas               binary         0.667
```

The NPV is 66.6%, which is desirable for our project.

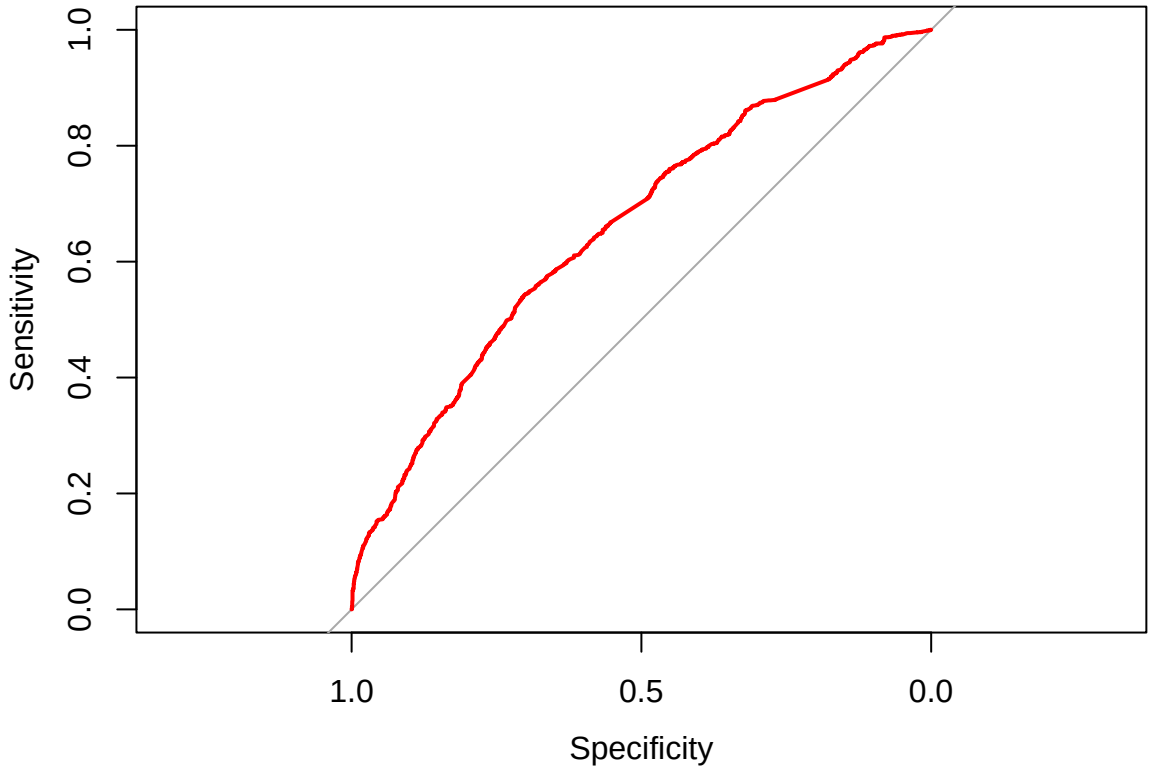
```
# Estimate the stepwise donation probability
step_prob <- predict(step_model, type = "response")

# Plot the ROC of the stepwise model
library(pROC)
```

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

##
## Attaching package: 'pROC'

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

```
ROC <- roc(df$Wheelchair.accessible, step_prob)
```

```
## Setting levels: control = FALSE, case = TRUE

## Setting direction: controls < cases
```

```
plot(ROC, col = "red")
```



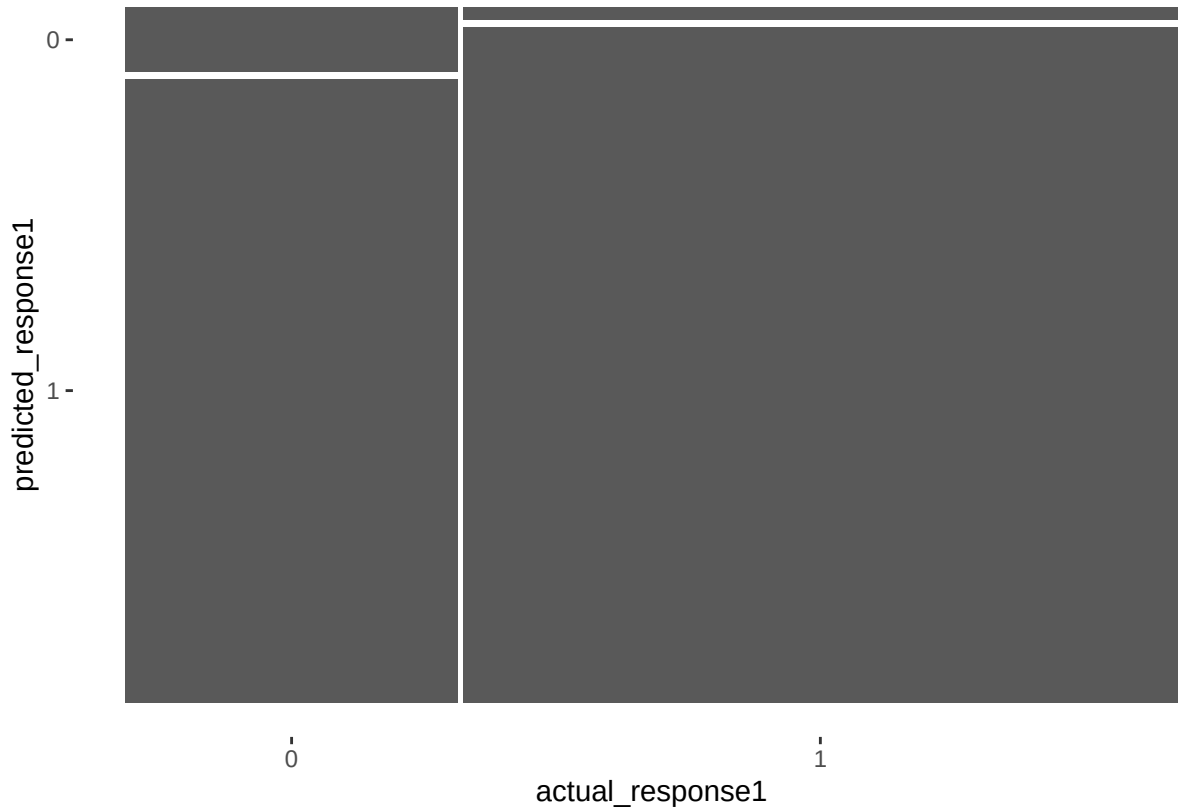

```
auc(ROC)
```

```
## Area under the curve: 0.6594
```

The area under the curve is above 0.6594, so the model performs just fine.

Model with only true values Threshold is set to 0.4 to control the False Negative rates, such that the result for negative predictive value will be higher, without sacrificing too much accuracy.

```
library(yardstick)

# From previous step
actual_response1 <- as.numeric(df2$Wheelchair.accessible)
predicted_response1 <- round(fitted(step_model1) + 0.1)
outcomes1 <- table(predicted_response1, actual_response1)
confusion1 <- conf_mat(outcomes1)

# "Automatically" plot the confusion matrix
autoplot(confusion1)
```




```
# Get summary metrics
summary(confusion1, event_level = "second")
```

```
## # A tibble: 13 x 3
##    .metric              .estimator .estimate
##    <chr>                <chr>          <dbl>
##  1 accuracy             binary        0.7
##  2 kap                  binary        0.0985
##  3 sens                 binary        0.981
##  4 spec                 binary        0.0942
##  5 ppv                  binary        0.700
##  6 npv                  binary        0.703
##  7 mcc                  binary        0.175
##  8 j_index              binary        0.0757
##  9 bal_accuracy         binary        0.538
## 10 detection_prevalence binary        0.957
## 11 precision            binary        0.700
## 12 recall               binary        0.981
## 13 f_meas               binary        0.817
```

The NPV is 70.3%, which is desirable for our project. The accuracy is also good at 70%.

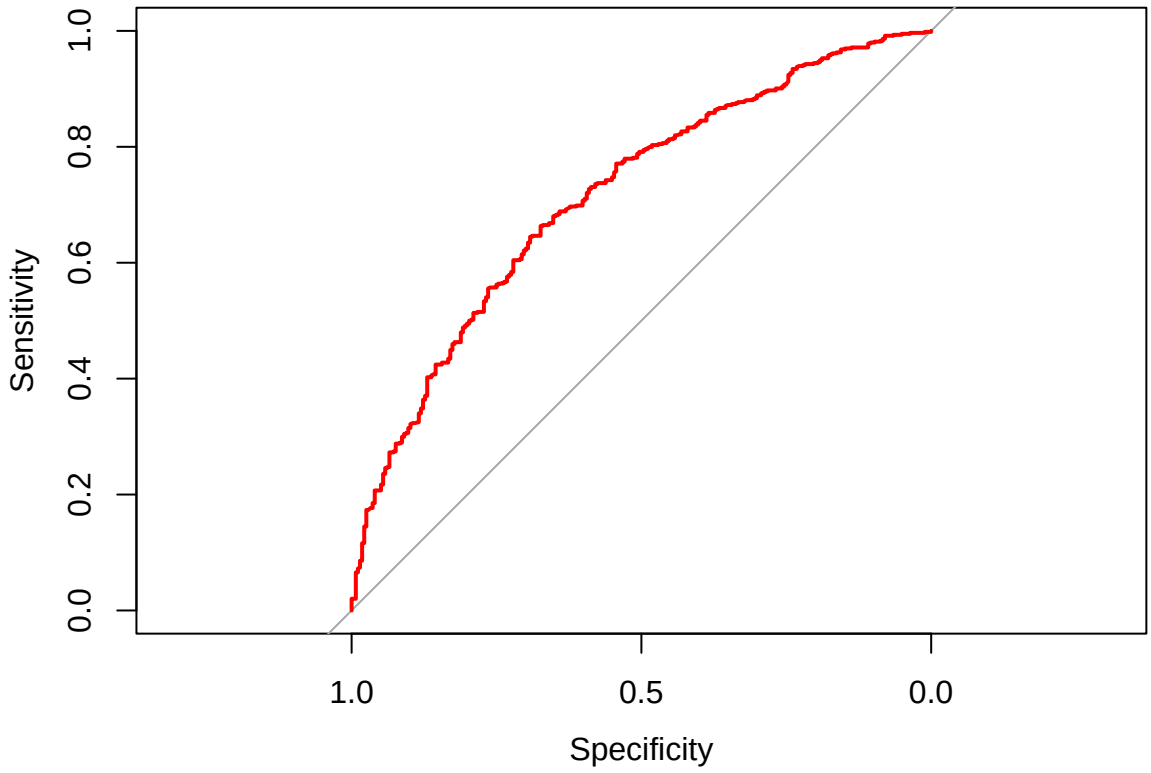
```
# Estimate the stepwise donation probability
step_prob1 <- predict(step_model1, type = "response")
```

```r
# Plot the ROC of the stepwise model
library(pROC)
ROC <- roc(df2$Wheelchair.accessible, step_prob1)
```

```
## Setting levels: control = FALSE, case = TRUE
```

```
## Setting direction: controls < cases
```

```r
plot(ROC, col = "red")
```




```r
auc(ROC)
```

```
## Area under the curve: 0.7153
```

The area under the curve is above 0.7, so the model performs well.

Therefore, the model with the actual values is better at predicting wheelchair accessibility when the predicted value is 0. But we still need the first model to be able to account for the situation where the capacity information is missing.

3. Summary

The goal of this section is to summarize the analysis and discuss its impacts. It's also concerned to provide insights on how future steps might improve the results and how to make best use of the results.

3.1 Main Findings

Two models can be applied depending on whether we have complete information on the capacity information, and either model will be able to provide the right suggestion 2/3 of the time. Also, we should take time prioritizing events without alcohol, WIFI, ticketing or promotion. At the same time, we want to search for super venue events and events with loud music.

3.2 Further Steps

First, we can keep collecting data, preferably data with capacity information because the second model is better. We may also seek to gather new information on the type of host, e.g., NGO or corporate, and the type of the event, e.g., a show or concert. We can improve our models in this manner.

Second, we should prioritize reaching to events with the characteristics we just found. Then we are likely to give help to people who really need the ramps and not to make our company look bad.