practicum_1

2023-05-19

loading the libraries

Understanding the data

1. Use the data dictionary describe each of the variables/features in the CSV in your report.

showing the data dictionary :

```
data <- read.delim(file = "C:/Users/Narges/Documents/practical_1/data_dic.txt" ,  header = TRUE, sep = 

print(data)
```

```
##                                                                X..Data.Dictionary
## 1                                        PatientID: Unique identifier for each patient
## 2                                    AppointmentID: Unique identifier to each appointment
## 3                                      Gender: Patient Gender (limited to Male or Female)
## 4                                ScheduledDate: date on which the appointment was scheduled
## 5                                     AppointmentDate: date of the actual appointment
## 6                                                                  Age: Patient age
## 7                             Neighbourhood: District of Vitória in which the appointment
## 8                   SocialWelfare: Patient is a recipient of Bolsa Família welfare payments
## 9                    Hypertension: Patient previously diagnoised with hypertensio (Boolean)
## 10                          Diabetes: Patient previously diagnosed with diabetes (Boolean)
## 11 AlcoholUseDisorder: Patient previously diagnosed with alcohol use disorder (Boolean)
## 12      Disability: Patient previously diagnosed with a disability (severity rated 0-4)
## 13                SMSReceived: At least 1 reminder text sent before appointment (Boolean)
## 14                 NoShow: Patient did not attend scheduled appointment (Boolean: Yes/No)
```

2. Can you think of 3 hypotheses for why someone may be more likely to miss a medical appointment?

Maybe the patient

3. Can you provide 3 examples of important contextual information that is missing in this data dictionary and dataset that could impact your analyses e.g., what type of medical appointment does each `AppointmentID` refer to?

1.how far is the patient address from the vitoria? 2.if the patient has any body to assist him or her? 3.the perso whom makes the appointment is the patient or some one else?

Data Parsing and Cleaning

4. Modify the following to make it reproducible i.e., downloads the data file directly from version control

we can download data separately and import it from the saved address (line 41) or download it by the command in line 42(reproducibility)

```
raw.data <- readr::read_csv('https://maguire-lab.github.io/health_data_science_research_2023/static_file
```

```
## Rows: 110527 Columns: 14
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): Gender, Neighbourhood, NoShow
## dbl  (9): PatientID, AppointmentID, Age, SocialWelfare, Hypertension, Diabet...
## dttm (2): ScheduledDate, AppointmentDate
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
print(raw.data)
```

```
## # A tibble: 110,527 x 14
##    PatientID AppointmentID Gender ScheduledDate       AppointmentDate       Age
##        <dbl>         <dbl> <chr>  <dttm>              <dttm>              <dbl>
##  1   2.99e13       5642903 F      2016-04-29 18:38:08 2016-04-29 00:00:00    62
##  2   5.59e14       5642503 M      2016-04-29 16:08:27 2016-04-29 00:00:00    56
##  3   4.26e12       5642549 F      2016-04-29 16:19:04 2016-04-29 00:00:00    62
##  4   8.68e11       5642828 F      2016-04-29 17:29:31 2016-04-29 00:00:00     8
##  5   8.84e12       5642494 F      2016-04-29 16:07:23 2016-04-29 00:00:00    56
##  6   9.60e13       5626772 F      2016-04-27 08:36:51 2016-04-29 00:00:00    76
##  7   7.34e14       5630279 F      2016-04-27 15:05:12 2016-04-29 00:00:00    23
##  8   3.45e12       5630575 F      2016-04-27 15:39:58 2016-04-29 00:00:00    39
##  9   5.64e13       5638447 F      2016-04-29 08:02:16 2016-04-29 00:00:00    21
## 10   7.81e13       5629123 F      2016-04-27 12:48:25 2016-04-29 00:00:00    19
## # i 110,517 more rows
## # i 8 more variables: Neighbourhood <chr>, SocialWelfare <dbl>,
## #   Hypertension <dbl>, Diabetes <dbl>, AlcoholUseDisorder <dbl>,
## #   Disability <dbl>, SMSReceived <dbl>, NoShow <chr>
```

checking if anybody is elder than 100 years or no?

```
age_under_110 <- raw.data %>% filter(Age > 110)
print(age_under_110)
```

```
## # A tibble: 5 x 14
##   PatientID AppointmentID Gender ScheduledDate       AppointmentDate       Age
##       <dbl>         <dbl> <chr>  <dttm>              <dttm>              <dbl>
## 1   3.20e13       5700278 F      2016-05-16 09:17:44 2016-05-19 00:00:00   115
## 2   3.20e13       5700279 F      2016-05-16 09:17:44 2016-05-19 00:00:00   115
## 3   3.20e13       5562812 F      2016-04-08 14:29:17 2016-05-16 00:00:00   115
## 4   3.20e13       5744037 F      2016-05-30 09:44:51 2016-05-30 00:00:00   115
## 5   7.48e14       5717451 F      2016-05-19 07:57:56 2016-06-03 00:00:00   115
## # i 8 more variables: Neighbourhood <chr>, SocialWelfare <dbl>,
## #   Hypertension <dbl>, Diabetes <dbl>, AlcoholUseDisorder <dbl>,
## #   Disability <dbl>, SMSReceived <dbl>, NoShow <chr>
```

```
age_under_110 <- age_under_110[!duplicated( age_under_110[c('PatientID')]),]
print(age_under_110)
```

```
## # A tibble: 2 x 14
##   PatientID AppointmentID Gender ScheduledDate       AppointmentDate       Age
##       <dbl>         <dbl> <chr>  <dttm>              <dttm>              <dbl>
## 1   3.20e13       5700278 F      2016-05-16 09:17:44 2016-05-19 00:00:00   115
## 2   7.48e14       5717451 F      2016-05-19 07:57:56 2016-06-03 00:00:00   115
## # i 8 more variables: Neighbourhood <chr>, SocialWelfare <dbl>,
## #   Hypertension <dbl>, Diabetes <dbl>, AlcoholUseDisorder <dbl>,
## #   Disability <dbl>, SMSReceived <dbl>, NoShow <chr>
```

we can see that there are 2 person elder than 110.

Exploratory Data Analysis

5 Are there any individuals with impossible ages? If so we can drop this row using `filter` i.e., `data <- data %>% filter(CRITERIA)`

if the data contains a person with age under 0, so we can say that's impossible.

```
raw.data %>% filter(Age<0)
```

```
## # A tibble: 1 x 14
##   PatientID AppointmentID Gender ScheduledDate       AppointmentDate       Age
##       <dbl>         <dbl> <chr>  <dttm>              <dttm>              <dbl>
## 1   4.66e14       5775010 F      2016-06-06 08:58:13 2016-06-06 00:00:00    -1
## # i 8 more variables: Neighbourhood <chr>, SocialWelfare <dbl>,
## #   Hypertension <dbl>, Diabetes <dbl>, AlcoholUseDisorder <dbl>,
## #   Disability <dbl>, SMSReceived <dbl>, NoShow <chr>
```

as you can see we have one person under 0 age, so we should drop it from our data set. so I'm going to filter people with less than 0 age. in the below cell I'm going to check the data dimension before and after the filtering to see it has affected the data set or not.

```
print(dim(raw.data))
```

```
## [1] 110527     14
```

```
raw.data <- raw.data %>% filter(Age >= 0)
print(dim(raw.data))
```

```
## [1] 110526     14
```

as its obvious just that one person is deleted.

so lets check if we have anybody with age 0 ?

```r
age_0 <- raw.data %>% filter(Age == 0)

age_0 <- age_0[!duplicated( age_0[c('PatientID')]),]
print(dim(age_0))
```

```
## [1] 2082   14
```

so we see that we have 2082 users with age 0 in our data set and i think that may happen for example newborn babies. so I'm not going to drop them.

but we are going to do some Exploratory on it. for example we wouldn't expect any of newborns to be diagnosed with Diabetes, Alcohol Use Disorder, and Hypertension. so we should check it in the data set too.

```r
raw.data %>% filter(Age == 0) %>% select(Hypertension, Diabetes, AlcoholUseDisorder) %>% unique()
```

```
## # A tibble: 1 x 3
##   Hypertension Diabetes AlcoholUseDisorder
##          <dbl>    <dbl>              <dbl>
## 1            0        0                  0
```

We can also explore things like how many different neighborhoods are there and how many appoints are from each?

```r
count(raw.data, Neighbourhood, sort = TRUE)
```

```
## # A tibble: 81 x 2
##    Neighbourhood         n
##    <chr>             <int>
##  1 JARDIM CAMBURI     7717
##  2 MARIA ORTIZ        5805
##  3 RESISTÊNCIA        4431
##  4 JARDIM DA PENHA    3877
##  5 ITARARÉ            3514
##  6 CENTRO             3334
##  7 TABUAZEIRO         3132
##  8 SANTA MARTHA       3131
##  9 JESUS DE NAZARETH  2853
## 10 BONFIM             2773
## # i 71 more rows
```

6 What is the maximum number of appointments from the same patient?

```r
count(raw.data, PatientID, sort = TRUE)
```

```
## # A tibble: 62,298 x 2
##    PatientID     n
##        <dbl> <int>
##  1   8.22e14    88
##  2   9.96e10    84
##  3   2.69e13    70
##  4   3.35e13    65
```

```
##  5   2.58e11    62
##  6   6.26e12    62
##  7   7.58e13    62
##  8   8.71e14    62
##  9   6.68e13    57
## 10   8.72e11    55
## # i 62,288 more rows
```

as you can see the maximum value equals to 88. i faced to a concern. and that's it, as we don't know whats the appointment for, if a person has more than one appointment in a day, it's a kind of duplicated data and i think they must be dropped out. so i would print the maximum appointment of a person also after deleting duplicates.

```
raw.data <- raw.data [!duplicated( raw.data[c('PatientID','ScheduledDate')]),]

print(dim(raw.data))
```

```
## [1] 109192     14
```

```
count(raw.data, PatientID, sort = TRUE)
```

```
## # A tibble: 62,298 x 2
##    PatientID     n
##        <dbl> <int>
##  1   8.22e14    88
##  2   9.96e10    84
##  3   2.69e13    70
##  4   3.35e13    65
##  5   2.58e11    62
##  6   6.26e12    62
##  7   8.71e14    62
##  8   7.58e13    61
##  9   6.68e13    57
## 10   8.72e11    55
## # i 62,288 more rows
```

again the maximum is 88.

Let's explore the correlation between variables: first let's replace non numeric values with some numeric values(because with its real labels the correlation haetmap dosent represnt as expected). so replacements would be as follow:

1.for gender encoding:

```
labs = LabelEncoder.fit(raw.data$Gender)

#convert labels to numeric values
raw.data$Gender = transform(labs, raw.data$Gender)
```

2.for Neighbourhood encoding:

```
labs = LabelEncoder.fit(raw.data$Neighbourhood)

#convert labels to numeric values
raw.data$Neighbourhood = transform(labs, raw.data$Neighbourhood)
```

3.for NoShow encoding:

```
labs = LabelEncoder.fit(raw.data$NoShow)

#convert labels to numeric values
raw.data$NoShow = transform(labs, raw.data$NoShow)
```

and after all checking the changes.

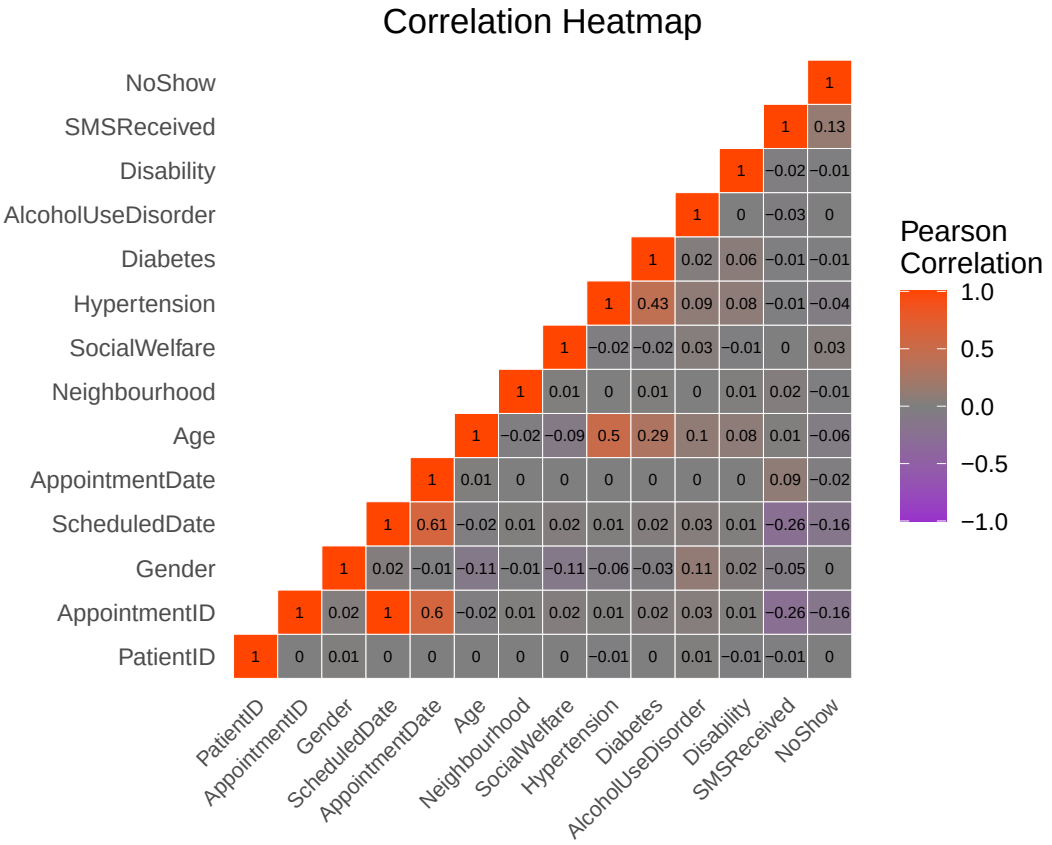
```
print(raw.data)
```

```
## # A tibble: 109,192 x 14
##    PatientID AppointmentID Gender ScheduledDate       AppointmentDate       Age
##        <dbl>         <dbl>  <int> <dttm>              <dttm>              <dbl>
##  1   2.99e13       5642903      1 2016-04-29 18:38:08 2016-04-29 00:00:00    62
##  2   5.59e14       5642503      2 2016-04-29 16:08:27 2016-04-29 00:00:00    56
##  3   4.26e12       5642549      1 2016-04-29 16:19:04 2016-04-29 00:00:00    62
##  4   8.68e11       5642828      1 2016-04-29 17:29:31 2016-04-29 00:00:00     8
##  5   8.84e12       5642494      1 2016-04-29 16:07:23 2016-04-29 00:00:00    56
##  6   9.60e13       5626772      1 2016-04-27 08:36:51 2016-04-29 00:00:00    76
##  7   7.34e14       5630279      1 2016-04-27 15:05:12 2016-04-29 00:00:00    23
##  8   3.45e12       5630575      1 2016-04-27 15:39:58 2016-04-29 00:00:00    39
##  9   5.64e13       5638447      1 2016-04-29 08:02:16 2016-04-29 00:00:00    21
## 10   7.81e13       5629123      1 2016-04-27 12:48:25 2016-04-29 00:00:00    19
## # i 109,182 more rows
## # i 8 more variables: Neighbourhood <int>, SocialWelfare <dbl>,
## #   Hypertension <dbl>, Diabetes <dbl>, AlcoholUseDisorder <dbl>,
## #   Disability <dbl>, SMSReceived <dbl>, NoShow <int>
```

Let's explore the correlation between variables:

```
# let's define a plotting function
corplot = function(df){

  cor_matrix_raw <- round(cor(df),2)
  cor_matrix <- melt(cor_matrix_raw)


  #Get triangle of the correlation matrix
  #Lower Triangle
  get_lower_tri<-function(cor_matrix_raw){
    cor_matrix_raw[upper.tri(cor_matrix_raw)] <- NA
    return(cor_matrix_raw)
  }

  # Upper Triangle
  get_upper_tri <- function(cor_matrix_raw){
```

```
    cor_matrix_raw[lower.tri(cor_matrix_raw)]<- NA
    return(cor_matrix_raw)
  }

  upper_tri <- get_upper_tri(cor_matrix_raw)

  # Melt the correlation matrix
  cor_matrix <- melt(upper_tri, na.rm = TRUE)

  # Heatmap Plot
  cor_graph <- ggplot(data = cor_matrix, aes(Var2, Var1, fill = value))+
    geom_tile(color = "white")+
    scale_fill_gradient2(low = "darkorchid", high = "orangered", mid = "grey50",
                         midpoint = 0, limit = c(-1,1), space = "Lab",
                         name="Pearson\nCorrelation") +
    theme_minimal()+
    theme(axis.text.x = element_text(angle = 45, vjust = 1,
                                     size = 8, hjust = 1))+
    coord_fixed()+ geom_text(aes(Var2, Var1, label = value), color = "black", size = 2) +
    theme(
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      panel.grid.major = element_blank(),
      panel.border = element_blank(),
      panel.background = element_blank(),
      axis.ticks = element_blank())+
      ggtitle("Correlation Heatmap")+
      theme(plot.title = element_text(hjust = 0.5))

  cor_graph
}

numeric.data = mutate_all(raw.data, function(x) as.numeric(x))

# Plot Correlation Heatmap
corplot(numeric.data)
```



7 Which parameters most strongly correlate with missing appointments (`NoShow`)?

column SMSReceived is most related one to NoShow.

8 Are there any other variables which strongly correlate with one another?

i would say the correlation by decending order as follow:

1.PatientID and AppointmentID are most related columns as their value in correlation heat map is equal to 0.65.

2.then AppointmentID and AppointmentDate are most correlated ones. also ScheduledDate and AppointmentDate are correlated with same corelation values(0.61).

3.third rank would be for Hypertension and Age with correlation value 0.5.

9 Do you see any issues with PatientID/AppointmentID being included in this plot?

actually they are not carrying any important feature or information in themselves and they are just 2 ID columns. so their correlation don't make any help at the final predictions. because they are unique and they don't have any valuable information in themselves. Including PatientID and AppointmentID in a correlation plot may not provide meaningful insights as they are unique identifiers and not directly related to the variables being analyzed. It would be more appropriate to focus on numerical variables and relevant categorical variables for correlations.

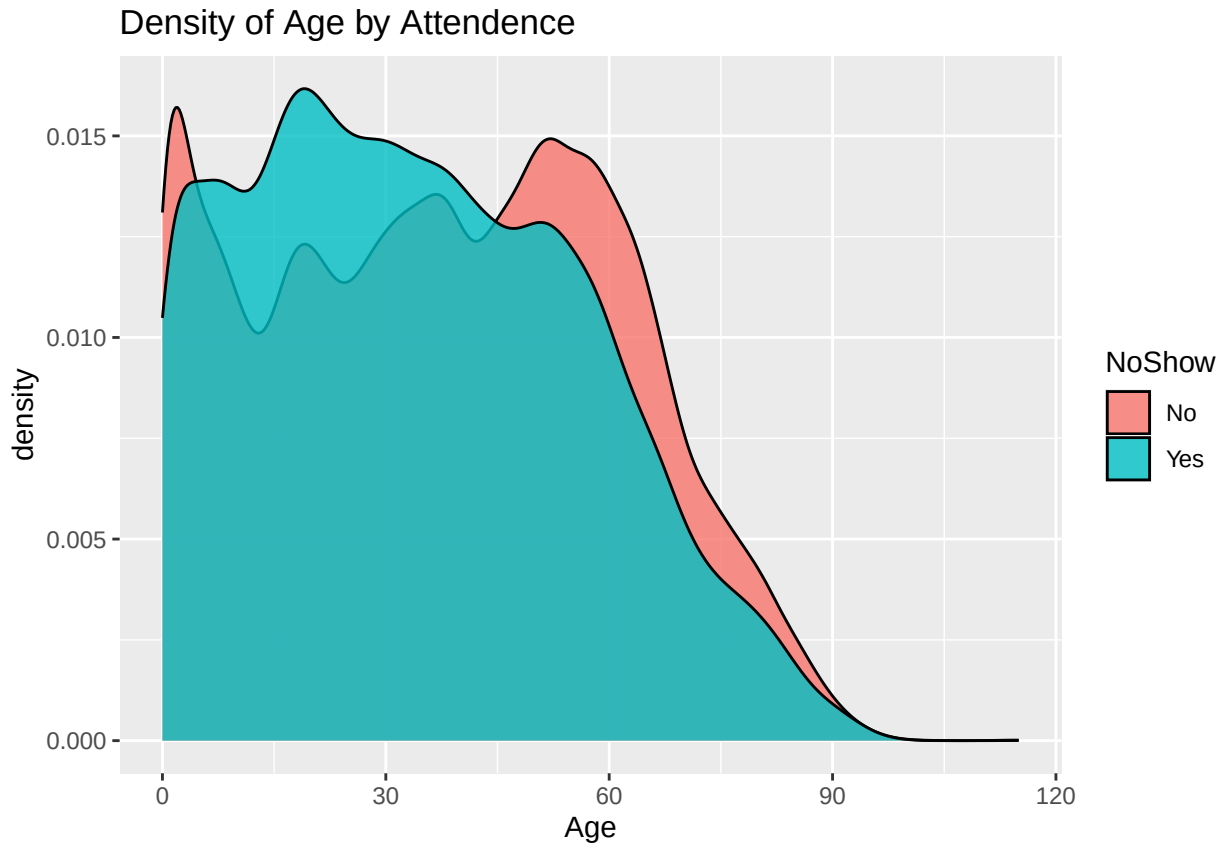
Let's look at some individual variables and their relationship with NoShow.

```
raw.data$NoShow = inverse.transform(labs, raw.data$NoShow)
print(raw.data)
```

```
## # A tibble: 109,192 x 14
```

```
##    PatientID AppointmentID Gender ScheduledDate       AppointmentDate       Age
##        <dbl>         <dbl>  <int> <dttm>              <dttm>              <dbl>
##  1   2.99e13       5642903      1 2016-04-29 18:38:08 2016-04-29 00:00:00    62
##  2   5.59e14       5642503      2 2016-04-29 16:08:27 2016-04-29 00:00:00    56
##  3   4.26e12       5642549      1 2016-04-29 16:19:04 2016-04-29 00:00:00    62
##  4   8.68e11       5642828      1 2016-04-29 17:29:31 2016-04-29 00:00:00     8
##  5   8.84e12       5642494      1 2016-04-29 16:07:23 2016-04-29 00:00:00    56
##  6   9.60e13       5626772      1 2016-04-27 08:36:51 2016-04-29 00:00:00    76
##  7   7.34e14       5630279      1 2016-04-27 15:05:12 2016-04-29 00:00:00    23
##  8   3.45e12       5630575      1 2016-04-27 15:39:58 2016-04-29 00:00:00    39
##  9   5.64e13       5638447      1 2016-04-29 08:02:16 2016-04-29 00:00:00    21
## 10   7.81e13       5629123      1 2016-04-27 12:48:25 2016-04-29 00:00:00    19
## # i 109,182 more rows
## # i 8 more variables: Neighbourhood <int>, SocialWelfare <dbl>,
## #   Hypertension <dbl>, Diabetes <dbl>, AlcoholUseDisorder <dbl>,
## #   Disability <dbl>, SMSReceived <dbl>, NoShow <chr>
```

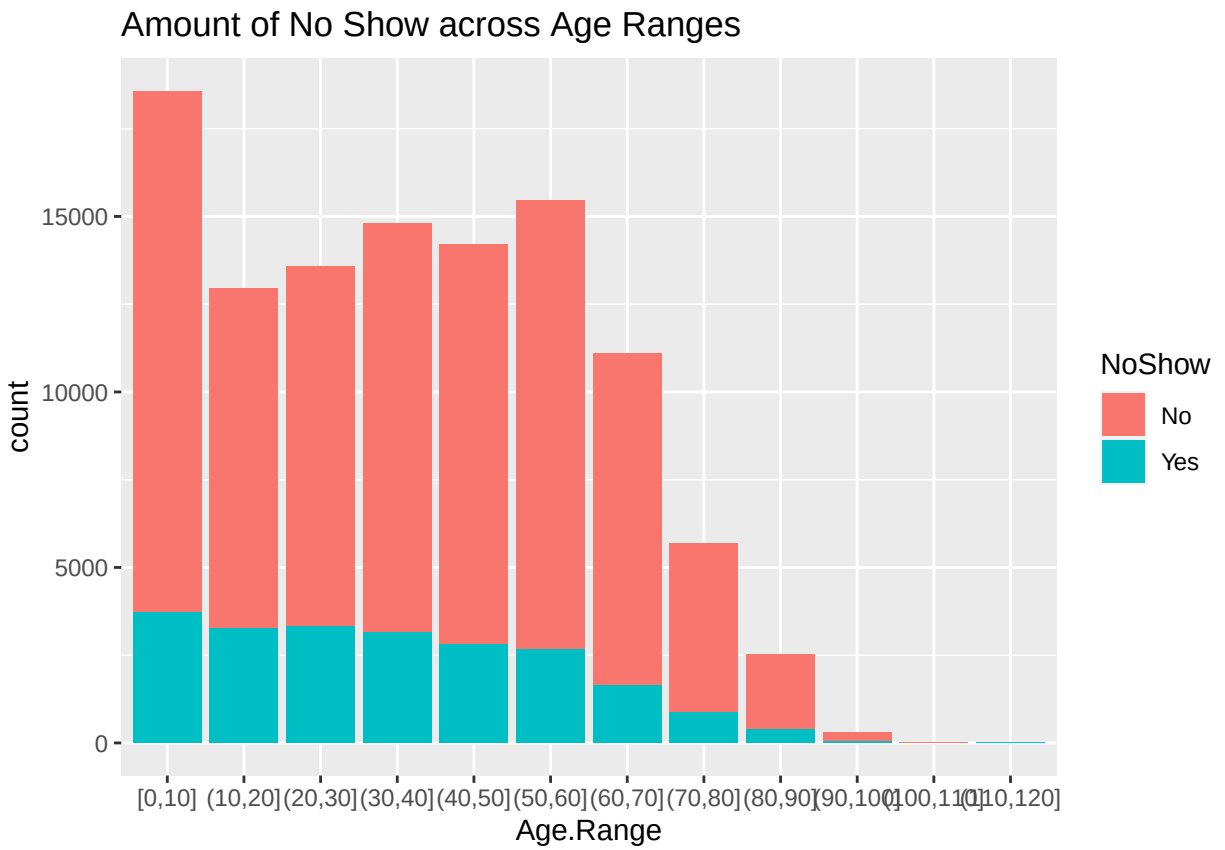
```
ggplot(raw.data) +
  geom_density(aes(x=Age, fill=NoShow), alpha=0.8) +
  ggtitle("Density of Age by Attendence")
```



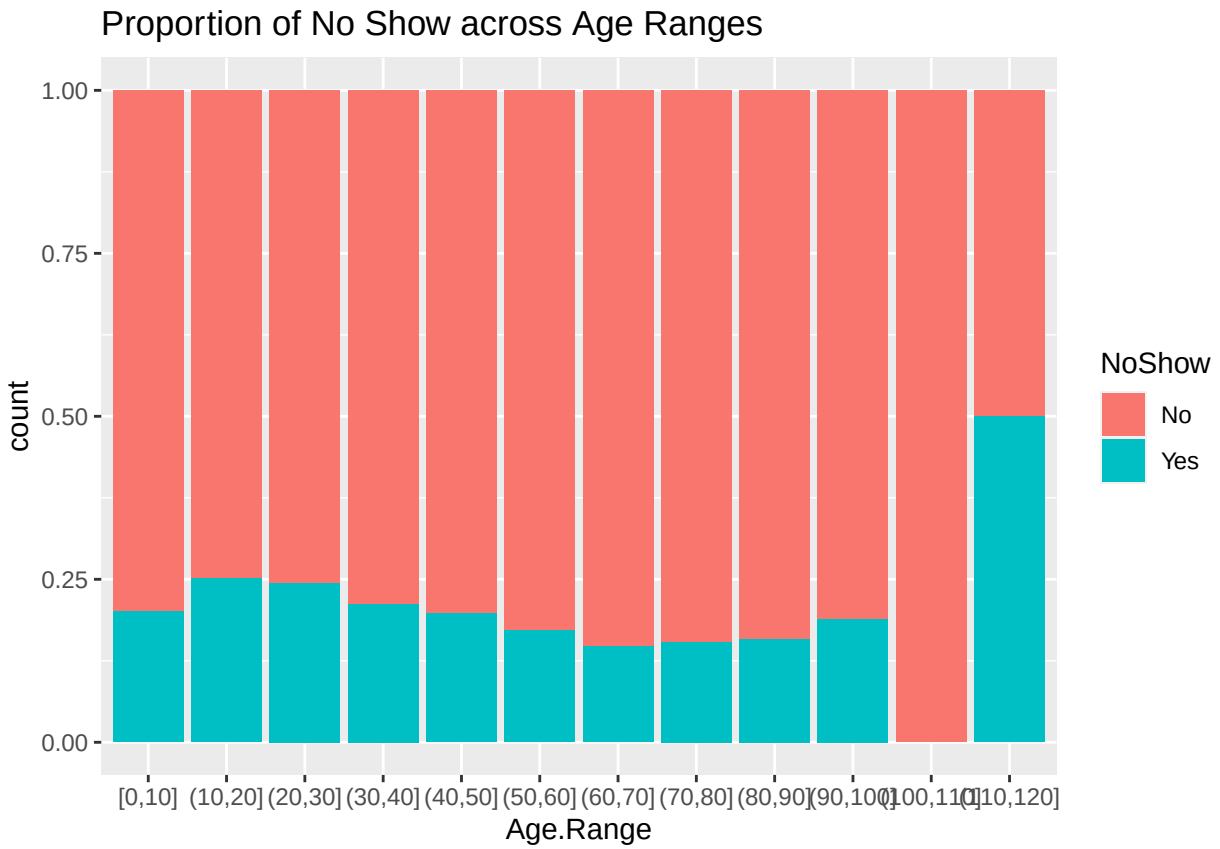
Let's take a closer look at age by breaking it into categories.

```
raw.data <- raw.data %>% mutate(Age.Range=cut_interval(Age, length=10))

ggplot(raw.data) +
```

```
  geom_bar(aes(x=Age.Range, fill=NoShow)) +
  ggtitle("Amount of No Show across Age Ranges")
```




```
ggplot(raw.data) +
  geom_bar(aes(x=Age.Range, fill=NoShow), position='fill') +
  ggtitle("Proportion of No Show across Age Ranges")
```

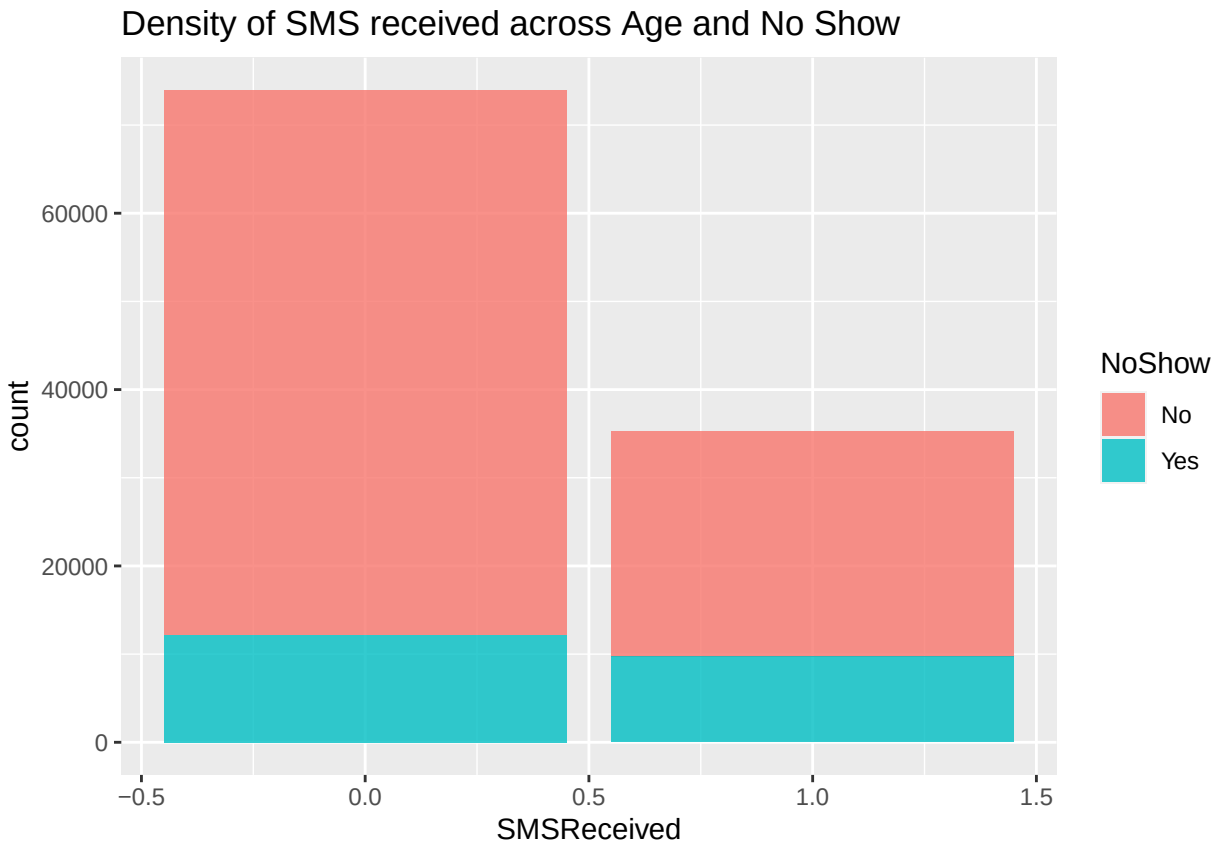


10. How could you be misled if you only plotted 1 of these 2 plots of attendance by age group? The key takeaway from this is that number of individuals > 90 are very few from plot 1 so probably are very small so unlikely to make much of an impact on the overall distributions. However, other patterns do emerge such as 10-20 age group is nearly twice as likely to miss appointments as the 60-70 years old.

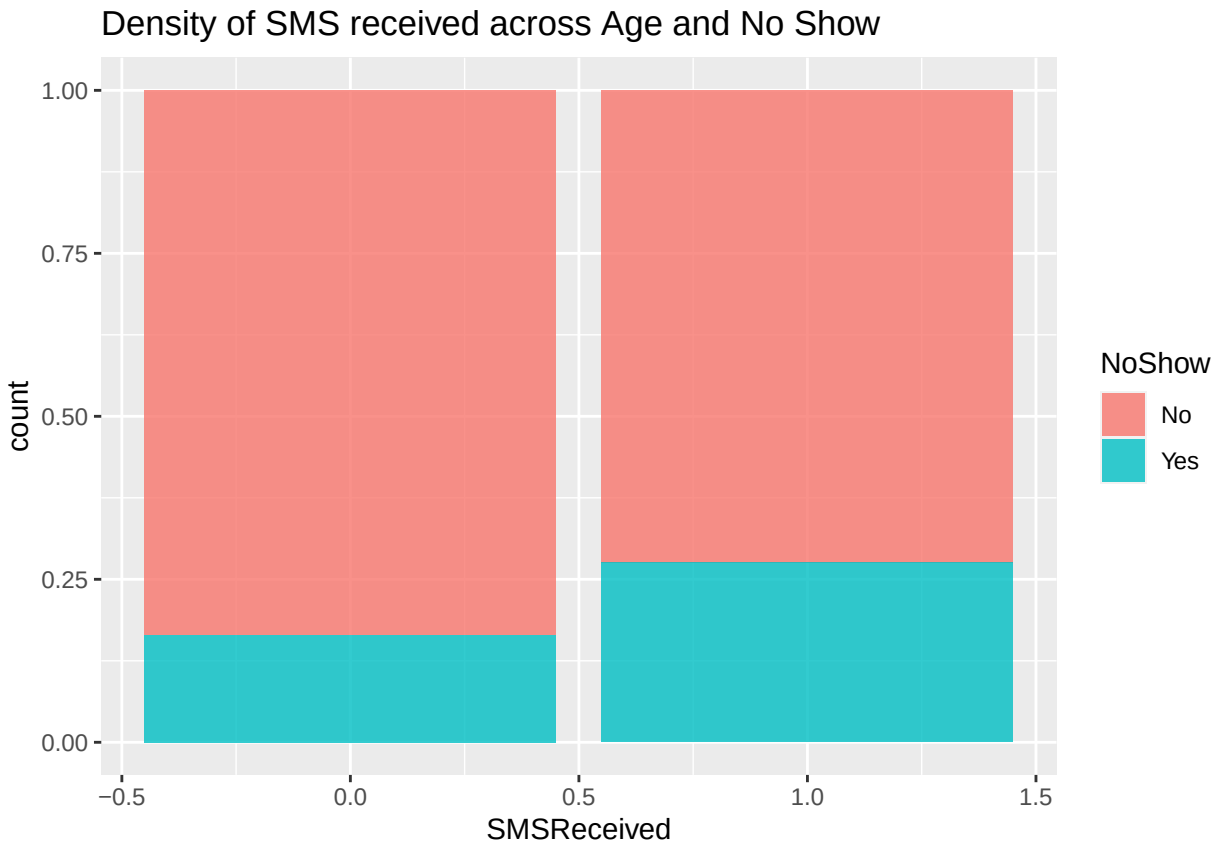
```
raw.data %>% filter(Age == 0) %>% count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  3524
```

```
ggplot(raw.data) +
  geom_bar(aes(x=SMSReceived, fill=NoShow), alpha=0.8) +
  ggtitle("Density of SMS received across Age and No Show")
```




```
ggplot(raw.data) +
  geom_bar(aes(x=SMSReceived, fill=NoShow), position='fill', alpha=0.8) +
  ggtitle("Density of SMS received across Age and No Show")
```

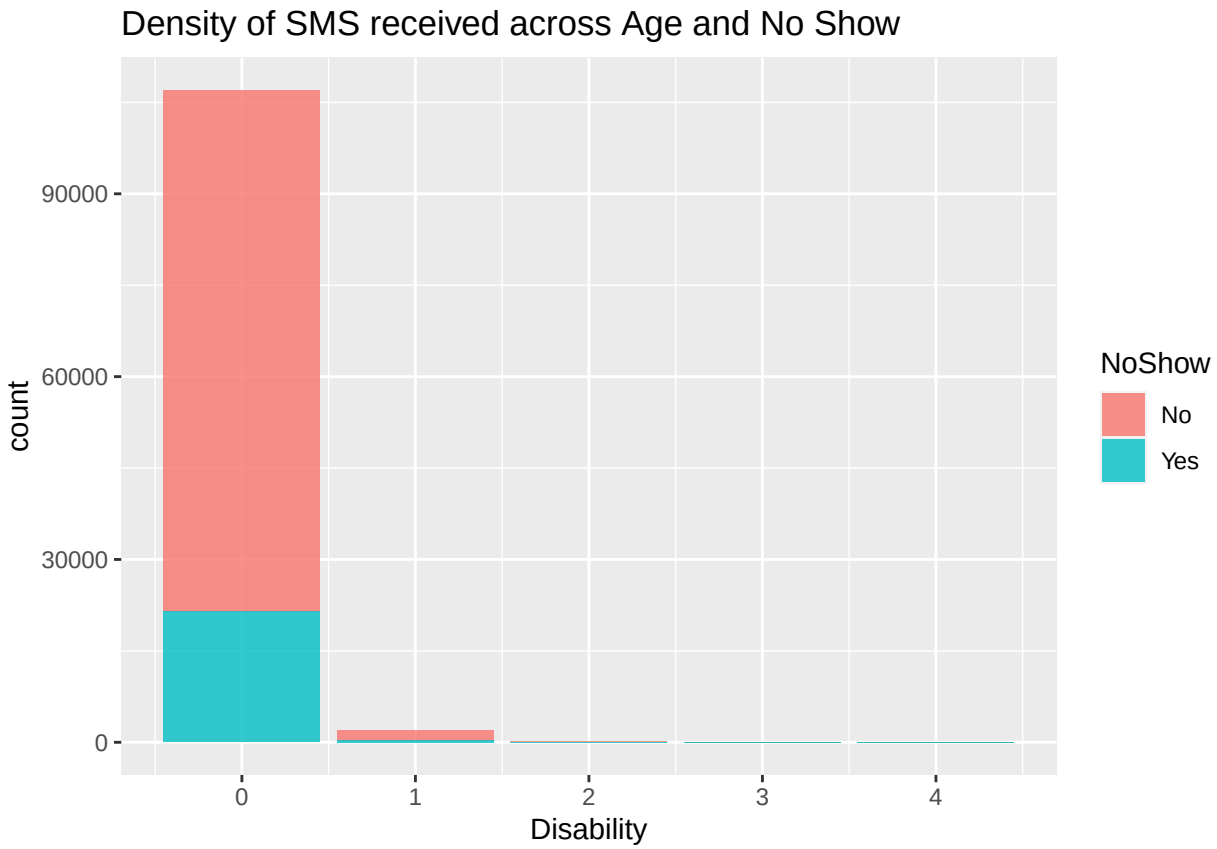


11. From this plot does it look like SMS reminders increase or decrease the chance of someone not attending an appointment? Why might the opposite actually be true (hint: think about biases)?

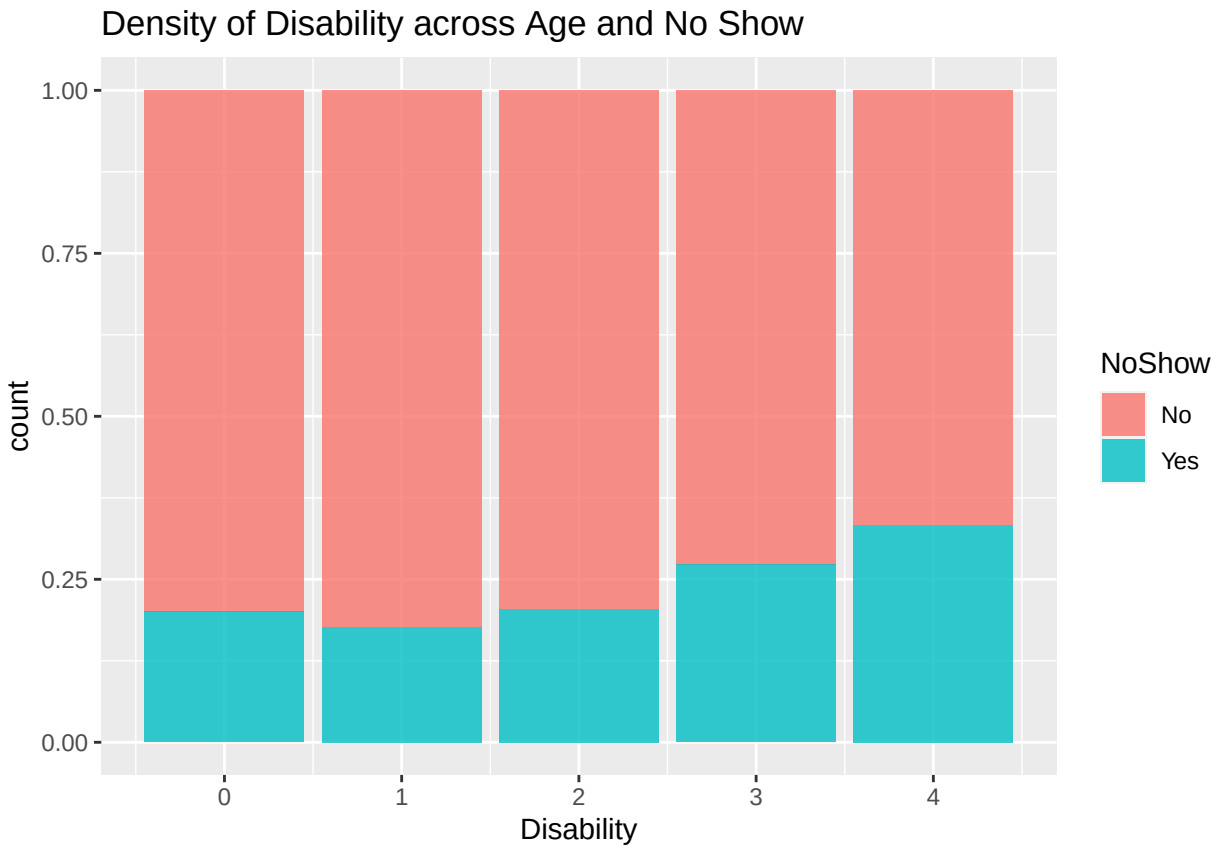
it seems that reviving a sms has increased the chance of some one not attending to the appointment. i think the portion of received messages are lower than not received and it shows in misled way that its in opposite way. but actually by taking a look at the first plot i think it says if they have not recived that sms but they attended. and in the small portion of recived ones in compare to not recived ones, it works opposit.

12. Create a similar plot which compares the the density of `NoShow` across the values of disability

```
ggplot(raw.data) +
  geom_bar(aes(x=Disability, fill=NoShow), alpha=0.8) +
  ggtitle("Density of SMS received across Age and No Show")
```

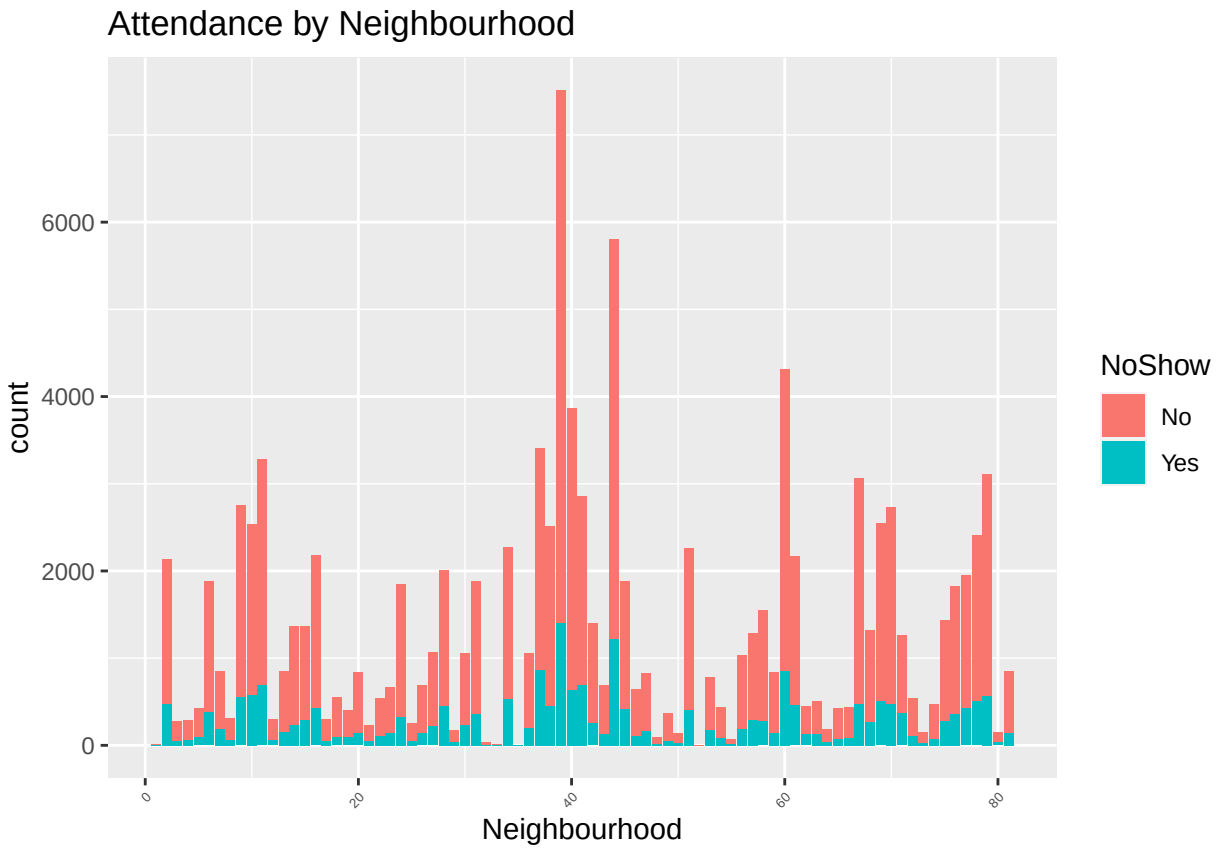



```
ggplot(raw.data) +
  geom_bar(aes(x=Disability, fill=NoShow), position='fill', alpha=0.8) +
  ggtitle("Density of Disability across Age and No Show")
```

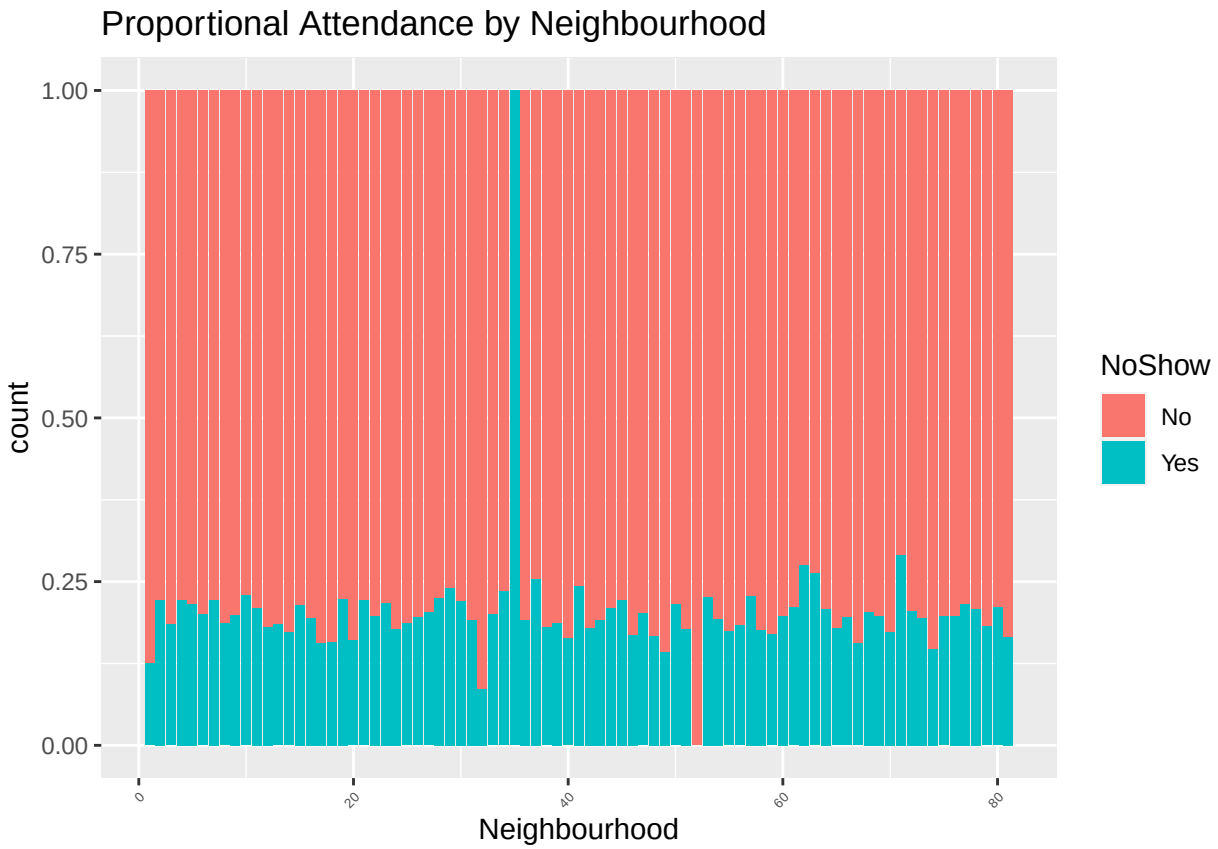


Now let's look at the neighbourhood data as location can correlate highly with many social determinants of health.

```
ggplot(raw.data) +
  geom_bar(aes(x=Neighbourhood, fill=NoShow)) +
  theme(axis.text.x = element_text(angle=45, hjust=1, size=5)) +
  ggtitle('Attendance by Neighbourhood')
```




```
ggplot(raw.data) +
  geom_bar(aes(x=Neighbourhood, fill=NoShow), position='fill') +
  theme(axis.text.x = element_text(angle=45, hjust=1, size=5)) +
  ggtitle('Proportional Attendance by Neighbourhood')
```



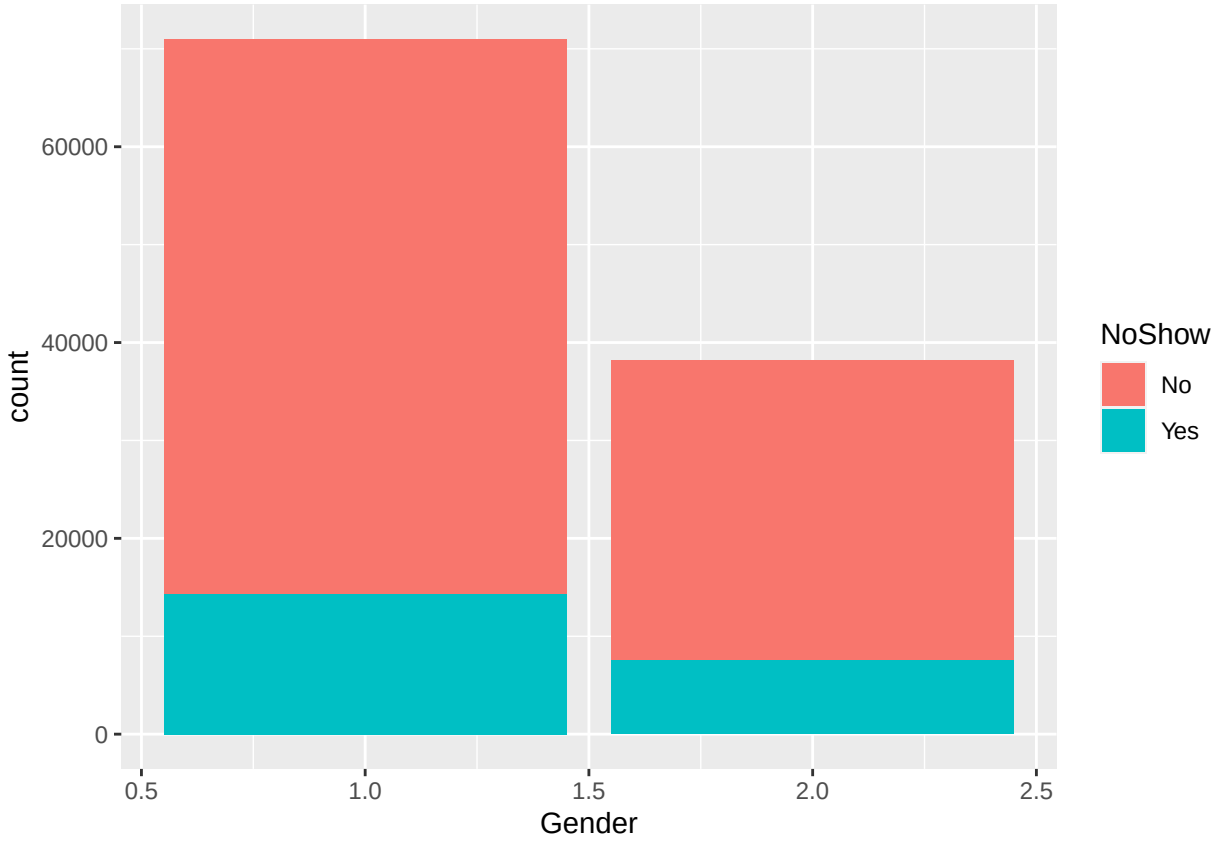
Most neighborhoods have similar proportions of no-show but some have much higher and lower rates.

13 Suggest a reason for differences in attendance rates across neighbourhoods.

it seems some of them are so far and this far distance may cause this noshow.

Now let's explore the relationship between gender and NoShow.

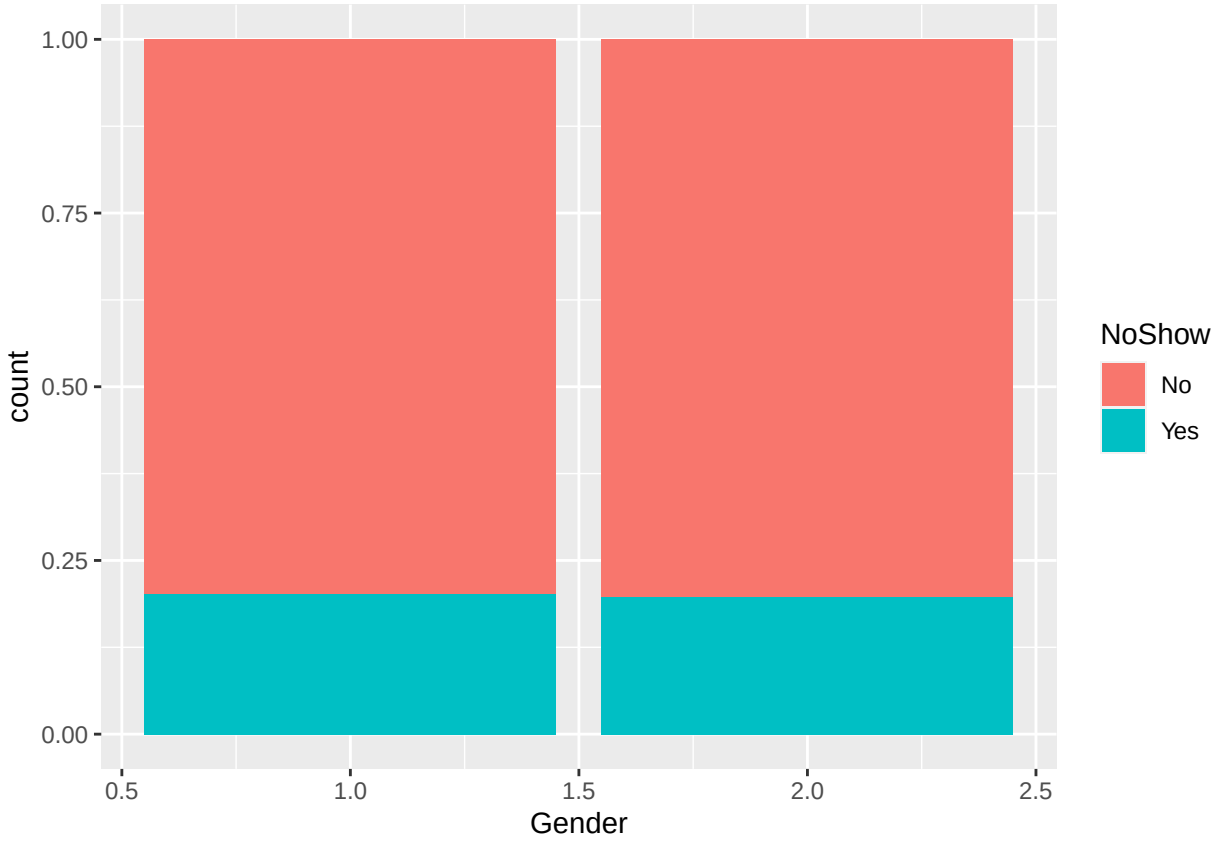
```
ggplot(raw.data) +
  geom_bar(aes(x=Gender, fill=NoShow))
```




```
  ggtitle("Gender by attendance")
```

```
## $title
## [1] "Gender by attendance"
##
## attr(,"class")
## [1] "labels"
```

```
ggplot(raw.data) +
  geom_bar(aes(x=Gender, fill=NoShow), position='fill')
```

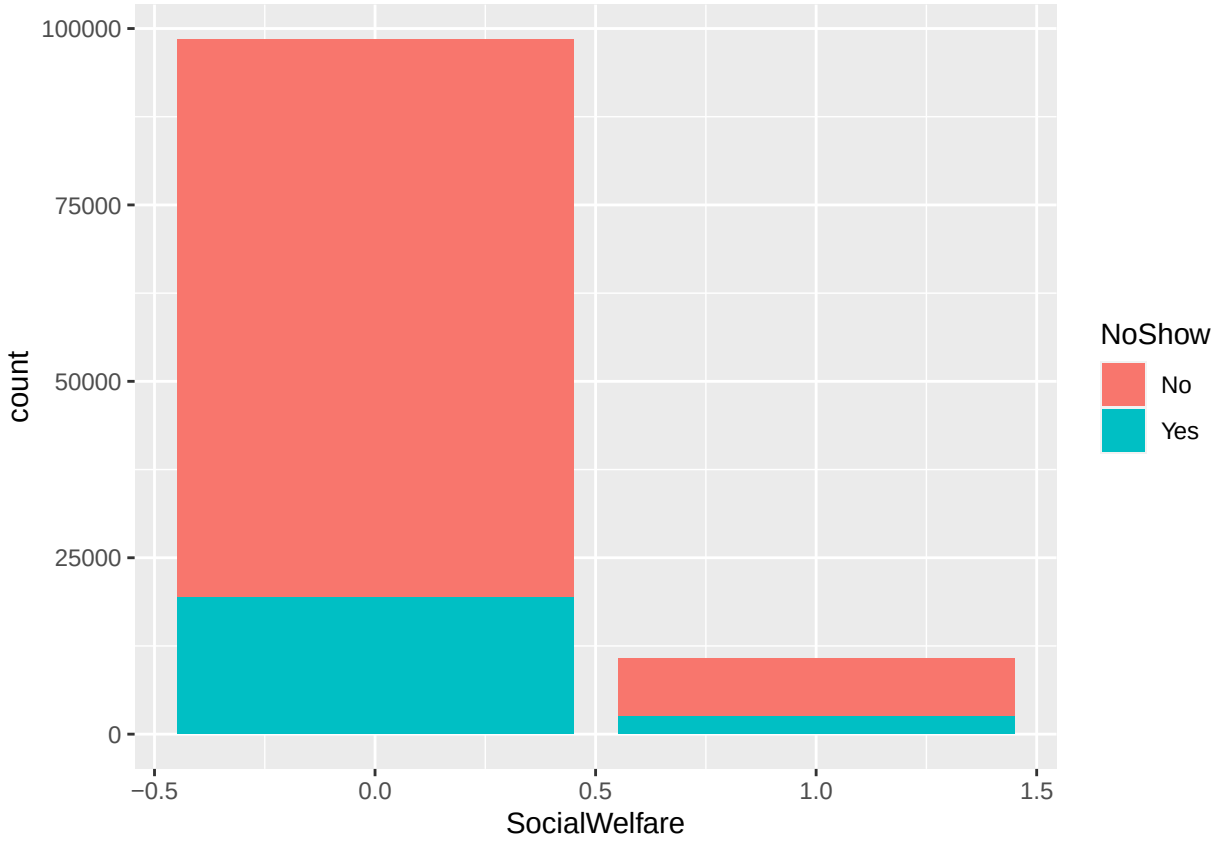



```
  ggtitle("Gender by attendance")
```

```
## $title
## [1] "Gender by attendance"
##
## attr(,"class")
## [1] "labels"
```

14 Create a similar plot using `SocialWelfare`

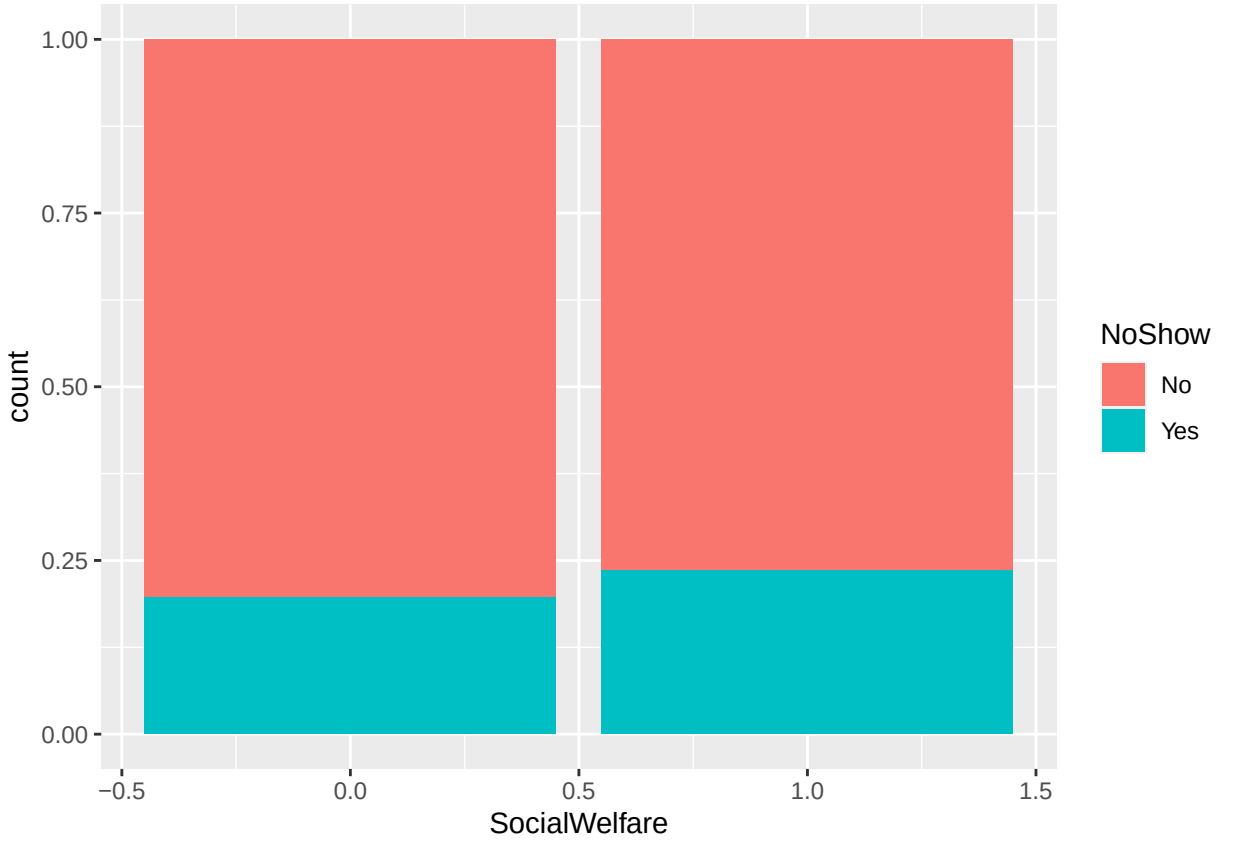
```
ggplot(raw.data) +
  geom_bar(aes(x=SocialWelfare, fill=NoShow))
```




```
  ggtitle("Gender by attendance")
```

```
## $title
## [1] "Gender by attendance"
##
## attr(,"class")
## [1] "labels"
```

```
ggplot(raw.data) +
  geom_bar(aes(x=SocialWelfare, fill=NoShow), position='fill')
```




```
  ggtitle("Gender by attendance")
```

```
## $title
## [1] "Gender by attendance"
##
## attr(,"class")
## [1] "labels"
```

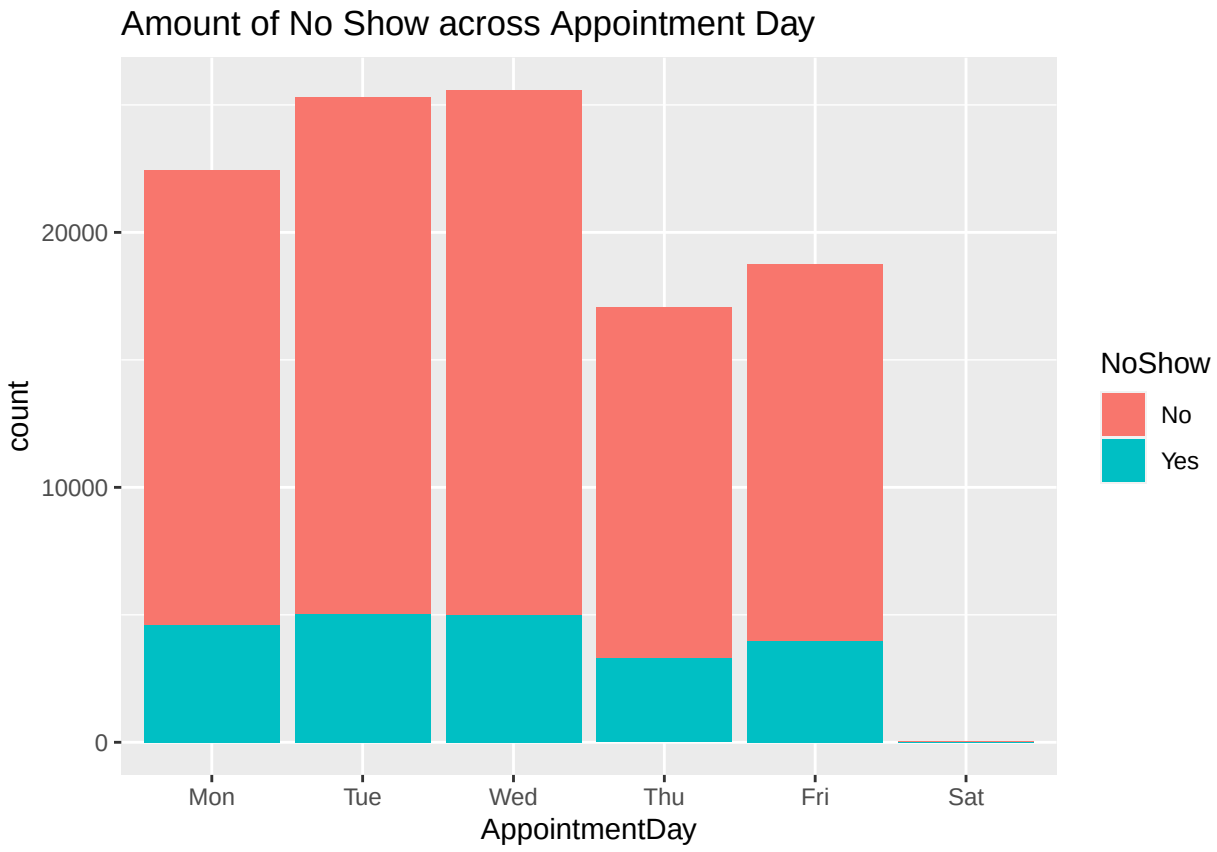
Far more exploration could still be done, including dimensionality reduction approaches but although we have found some patterns there is no major/striking patterns on the data as it currently stands.

However, maybe we can generate some new features/variables that more strongly relate to the `NoShow`.

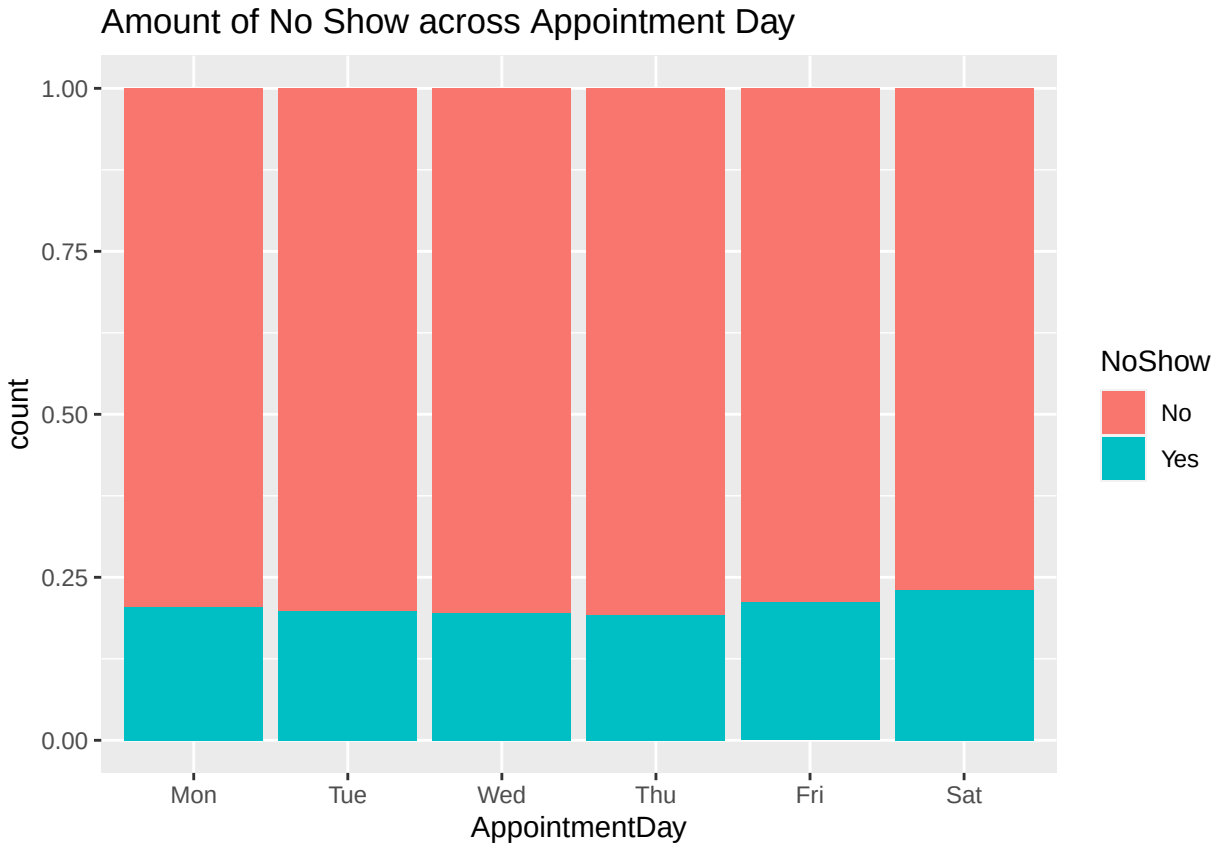
Feature Engineering

Let's begin by seeing if appointments on any day of the week has more no-show's. Fortunately, the `lubridate` library makes this quite easy!

```
raw.data <- raw.data %>% mutate(AppointmentDay = wday(AppointmentDate, label=TRUE, abbr=TRUE),
                                ScheduledDay = wday(ScheduledDate,  label=TRUE, abbr=TRUE))

ggplot(raw.data) +
  geom_bar(aes(x=AppointmentDay, fill=NoShow)) +
  ggtitle("Amount of No Show across Appointment Day")
```



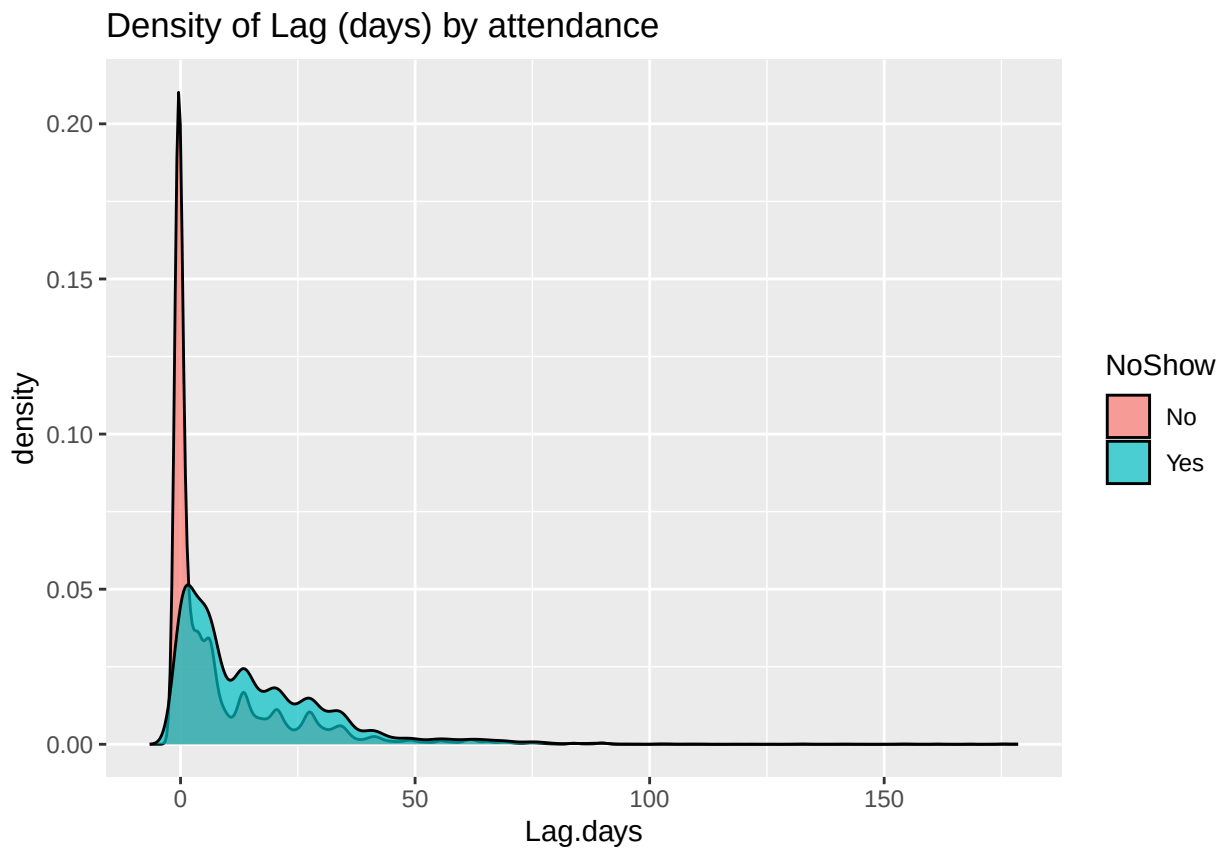

```
ggplot(raw.data) +
  geom_bar(aes(x=AppointmentDay, fill=NoShow), position = 'fill') +
  ggtitle("Amount of No Show across Appointment Day")
```



Let's begin by creating a variable called `Lag`, which is the difference between when an appointment was scheduled and the actual appointment.

```
raw.data <- raw.data %>% mutate(Lag.days=difftime(AppointmentDate, ScheduledDate, units = "days"),
                                Lag.hours=difftime(AppointmentDate, ScheduledDate, units = "hours"))
raw.data <- raw.data %>% filter(Age >= 0)

ggplot(raw.data) +
  geom_density(aes(x=Lag.days, fill=NoShow), alpha=0.7)+
  ggtitle("Density of Lag (days) by attendance")
```

```
## Don't know how to automatically pick scale for object of type <difftime>.
## Defaulting to continuous.
```



15 i thought that if the lag be a longer time, so the patient may have forgotten to attend but we see thay have attended. and i think may be the count of these late schecduled apointments are few.

Predictive Modeling

Let's see how well we can predict NoShow from the data.

We'll start by preparing the data, followed by splitting it into testing and training set, modeling and finally, evaluating our results. For now we will subsample but please run on full dataset for final execution.

```
###reading the data from scratch because i had made label encoding which its better to flash it back:
raw.data <- readr::read_csv('https://maguire-lab.github.io/health_data_science_research_2023/static_file
```

```
## Rows: 110527 Columns: 14
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): Gender, Neighbourhood, NoShow
## dbl  (9): PatientID, AppointmentID, Age, SocialWelfare, Hypertension, Diabet...
## dttm (2): ScheduledDate, AppointmentDate
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
print(raw.data)
```

```
## # A tibble: 110,527 x 14
##    PatientID AppointmentID Gender ScheduledDate       AppointmentDate       Age
##        <dbl>         <dbl> <chr>  <dttm>              <dttm>              <dbl>
##  1   2.99e13       5642903 F      2016-04-29 18:38:08 2016-04-29 00:00:00    62
##  2   5.59e14       5642503 M      2016-04-29 16:08:27 2016-04-29 00:00:00    56
##  3   4.26e12       5642549 F      2016-04-29 16:19:04 2016-04-29 00:00:00    62
##  4   8.68e11       5642828 F      2016-04-29 17:29:31 2016-04-29 00:00:00     8
##  5   8.84e12       5642494 F      2016-04-29 16:07:23 2016-04-29 00:00:00    56
##  6   9.60e13       5626772 F      2016-04-27 08:36:51 2016-04-29 00:00:00    76
##  7   7.34e14       5630279 F      2016-04-27 15:05:12 2016-04-29 00:00:00    23
##  8   3.45e12       5630575 F      2016-04-27 15:39:58 2016-04-29 00:00:00    39
##  9   5.64e13       5638447 F      2016-04-29 08:02:16 2016-04-29 00:00:00    21
## 10   7.81e13       5629123 F      2016-04-27 12:48:25 2016-04-29 00:00:00    19
## # i 110,517 more rows
## # i 8 more variables: Neighbourhood <chr>, SocialWelfare <dbl>,
## #   Hypertension <dbl>, Diabetes <dbl>, AlcoholUseDisorder <dbl>,
## #   Disability <dbl>, SMSReceived <dbl>, NoShow <chr>
```

```
### REMOVE SUBSAMPLING FOR FINAL MODEL
data.prep <- raw.data %>% select(-AppointmentID, -PatientID) #%>% sample_n(10000)

set.seed(42)
data.split <- initial_split(data.prep, prop = 0.7)
train  <- training(data.split)
test <- testing(data.split)
```

Let's now set the cross validation parameters, and add classProbs so we can use AUC as a metric for xgboost.

```
fit.control <- trainControl(method="cv",number=3,
                            classProbs = TRUE, summaryFunction = twoClassSummary)
```

16 Based on the EDA, how well do you think this is going to work?

Now we can train our XGBoost model.

```
xgb.grid <- expand.grid(eta=c(0.05),
                        max_depth=c(4),colsample_bytree=1,
                        subsample=1, nrounds=500, gamma=0, min_child_weight=5)

xgb.model <- train(NoShow ~ .,data=train, method="xgbTree",metric="ROC",
                   tuneGrid=xgb.grid, trControl=fit.control)
```

again for changing the null levels on Neighbourhood to its real ones for both train and test.

```
train$Neighbourhood<- as.factor(train$Neighbourhood)

levels(train$Neighbourhood)
```

```
##  [1] "AEROPORTO"                   "ANDORINHAS"
```

```
##  [3] "ANTÔNIO HONÓRIO"             "ARIOVALDO FAVALESSA"
##  [5] "BARRO VERMELHO"              "BELA VISTA"
##  [7] "BENTO FERREIRA"              "BOA VISTA"
##  [9] "BONFIM"                      "CARATOÍRA"
## [11] "CENTRO"                      "COMDUSA"
## [13] "CONQUISTA"                   "CONSOLAÇÃO"
## [15] "CRUZAMENTO"                  "DA PENHA"
## [17] "DE LOURDES"                  "DO CABRAL"
## [19] "DO MOSCOSO"                  "DO QUADRO"
## [21] "ENSEADA DO SUÁ"              "ESTRELINHA"
## [23] "FONTE GRANDE"                "FORTE SÃO JOÃO"
## [25] "FRADINHOS"                   "GOIABEIRAS"
## [27] "GRANDE VITÓRIA"              "GURIGICA"
## [29] "HORTO"                       "ILHA DAS CAIEIRAS"
## [31] "ILHA DE SANTA MARIA"         "ILHA DO BOI"
## [33] "ILHA DO FRADE"               "ILHA DO PRÍNCIPE"
## [35] "ILHAS OCEÂNICAS DE TRINDADE" "INHANGUETÁ"
## [37] "ITARARÉ"                     "JABOUR"
## [39] "JARDIM CAMBURI"              "JARDIM DA PENHA"
## [41] "JESUS DE NAZARETH"           "JOANA D´ARC"
## [43] "JUCUTUQUARA"                 "MARIA ORTIZ"
## [45] "MÁRIO CYPRESTE"              "MARUÍPE"
## [47] "MATA DA PRAIA"               "MONTE BELO"
## [49] "MORADA DE CAMBURI"           "NAZARETH"
## [51] "NOVA PALESTINA"              "PARQUE MOSCOSO"
## [53] "PIEDADE"                     "PONTAL DE CAMBURI"
## [55] "PRAIA DO CANTO"              "PRAIA DO SUÁ"
## [57] "REDENÇÃO"                    "REPÚBLICA"
## [59] "RESISTÊNCIA"                 "ROMÃO"
## [61] "SANTA CECÍLIA"               "SANTA CLARA"
## [63] "SANTA HELENA"                "SANTA LÚCIA"
## [65] "SANTA LUÍZA"                 "SANTA MARTHA"
## [67] "SANTA TEREZA"                "SANTO ANDRÉ"
## [69] "SANTO ANTÔNIO"               "SANTOS DUMONT"
## [71] "SANTOS REIS"                 "SÃO BENEDITO"
## [73] "SÃO CRISTÓVÃO"               "SÃO JOSÉ"
## [75] "SÃO PEDRO"                   "SEGURANÇA DO LAR"
## [77] "SOLON BORGES"                "TABUAZEIRO"
## [79] "UNIVERSITÁRIO"               "VILA RUBIM"
```

```
test$Neighbourhood<- as.factor(test$Neighbourhood)

levels(test$Neighbourhood)
```

```
##  [1] "AEROPORTO"           "ANDORINHAS"          "ANTÔNIO HONÓRIO"
##  [4] "ARIOVALDO FAVALESSA" "BARRO VERMELHO"      "BELA VISTA"
##  [7] "BENTO FERREIRA"      "BOA VISTA"           "BONFIM"
## [10] "CARATOÍRA"           "CENTRO"              "COMDUSA"
## [13] "CONQUISTA"           "CONSOLAÇÃO"          "CRUZAMENTO"
## [16] "DA PENHA"            "DE LOURDES"          "DO CABRAL"
## [19] "DO MOSCOSO"          "DO QUADRO"           "ENSEADA DO SUÁ"
## [22] "ESTRELINHA"          "FONTE GRANDE"        "FORTE SÃO JOÃO"
## [25] "FRADINHOS"           "GOIABEIRAS"          "GRANDE VITÓRIA"
## [28] "GURIGICA"            "HORTO"               "ILHA DAS CAIEIRAS"
```

```
## [31] "ILHA DE SANTA MARIA" "ILHA DO BOI"         "ILHA DO FRADE"
## [34] "ILHA DO PRÍNCIPE"    "INHANGUETÁ"          "ITARARÉ"
## [37] "JABOUR"              "JARDIM CAMBURI"      "JARDIM DA PENHA"
## [40] "JESUS DE NAZARETH"   "JOANA D´ARC"         "JUCUTUQUARA"
## [43] "MARIA ORTIZ"         "MÁRIO CYPRESTE"      "MARUÍPE"
## [46] "MATA DA PRAIA"       "MONTE BELO"          "MORADA DE CAMBURI"
## [49] "NAZARETH"            "NOVA PALESTINA"      "PARQUE INDUSTRIAL"
## [52] "PARQUE MOSCOSO"      "PIEDADE"             "PONTAL DE CAMBURI"
## [55] "PRAIA DO CANTO"      "PRAIA DO SUÁ"        "REDENÇÃO"
## [58] "REPÚBLICA"           "RESISTÊNCIA"         "ROMÃO"
## [61] "SANTA CECÍLIA"       "SANTA CLARA"         "SANTA HELENA"
## [64] "SANTA LÚCIA"         "SANTA LUÍZA"         "SANTA MARTHA"
## [67] "SANTA TEREZA"        "SANTO ANDRÉ"         "SANTO ANTÔNIO"
## [70] "SANTOS DUMONT"       "SANTOS REIS"         "SÃO BENEDITO"
## [73] "SÃO CRISTÓVÃO"       "SÃO JOSÉ"            "SÃO PEDRO"
## [76] "SEGURANÇA DO LAR"    "SOLON BORGES"        "TABUAZEIRO"
## [79] "UNIVERSITÁRIO"       "VILA RUBIM"
```

as we can see in above the levels of Neighbourhood in both test and train differ from each other. it happens because of splitting non shuffled data. so may be a portion including a special level goes to test portion which has not appeared in train. so we have to delete the difference to see the performance by confusion matrix.

```
test <- subset(test,Neighbourhood !=  "PARQUE INDUSTRIAL")
```

because the levels of test$NoShow is null i have to do the following code at first:

```
test$NoShow<- as.factor(test$NoShow)
```

then lets check the XGBoost model performance:

```
xgb.pred <- predict(xgb.model, newdata=test)
xgb.probs <- predict(xgb.model, newdata=test, type="prob")

test <- test %>% mutate(NoShow.numerical = ifelse(NoShow=="Yes",1,0))
confusionMatrix(xgb.pred, test$NoShow, positive="Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    No   Yes
##        No  26376  6443
##        Yes   150   189
##
##                Accuracy : 0.8012
##                  95% CI : (0.7968, 0.8054)
##     No Information Rate : 0.8
##     P-Value [Acc > NIR] : 0.2989
##
##                   Kappa : 0.0355
##
##  Mcnemar's Test P-Value : <2e-16
```

```
##
##             Sensitivity : 0.02850
##             Specificity : 0.99435
##          Pos Pred Value : 0.55752
##          Neg Pred Value : 0.80368
##              Prevalence : 0.20001
##          Detection Rate : 0.00570
##    Detection Prevalence : 0.01022
##       Balanced Accuracy : 0.51142
##
##        'Positive' Class : Yes
##
```

```
paste("XGBoost Area under ROC Curve: ", round(auc(test$NoShow.numerical, xgb.probs[,2]),3), sep="")
```

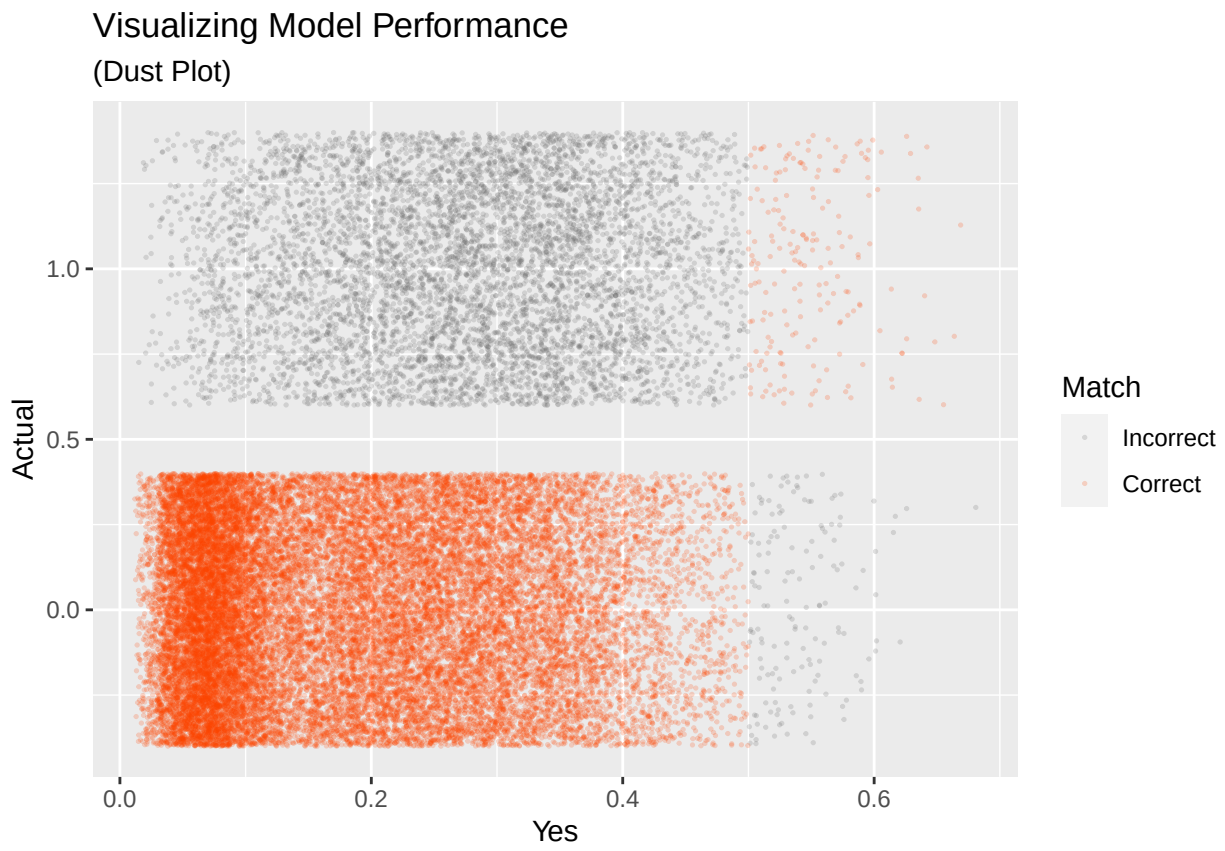
```
## Setting levels: control = 0, case = 1
```

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

```
## [1] "XGBoost Area under ROC Curve: 0.726"
```

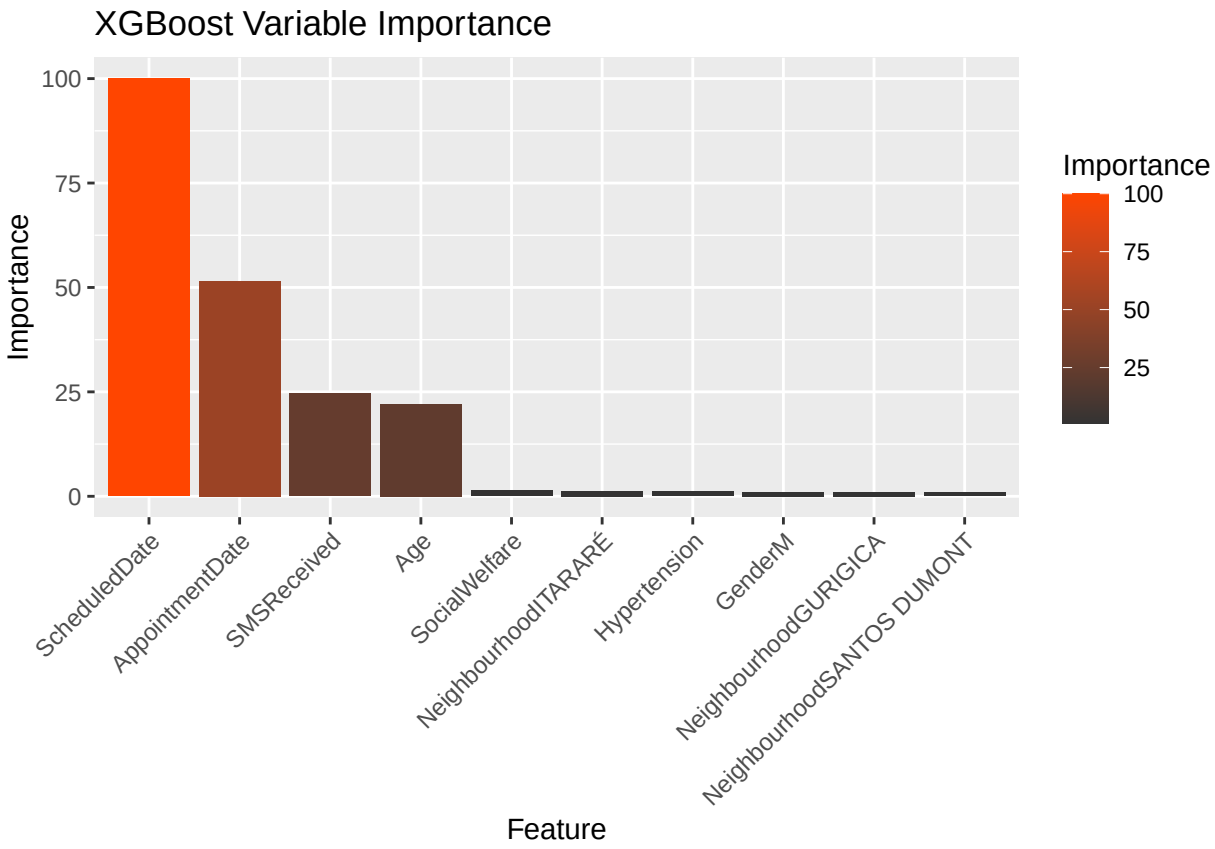
This isn't an unreasonable performance, but let's look a bit more carefully at the correct and incorrect predictions,

```
xgb.probs$Actual = test$NoShow.numerical
xgb.probs$ActualClass = test$NoShow
xgb.probs$PredictedClass = xgb.pred
xgb.probs$Match = ifelse(xgb.probs$ActualClass == xgb.probs$PredictedClass,
                         "Correct","Incorrect")
# [4.8] Plot Accuracy
xgb.probs$Match = factor(xgb.probs$Match,levels=c("Incorrect","Correct"))
ggplot(xgb.probs,aes(x=Yes,y=Actual,color=Match))+
  geom_jitter(alpha=0.2,size=0.25)+
  scale_color_manual(values=c("grey40","orangered"))+
  ggtitle("Visualizing Model Performance", "(Dust Plot)")
```



Finally, let's close it off with the variable importance of our model:

```
results = data.frame(Feature = rownames(varImp(xgb.model)$importance)[1:10],
                     Importance = varImp(xgb.model)$importance[1:10,])

results$Feature = factor(results$Feature,levels=results$Feature)


# [4.10] Plot Variable Importance
ggplot(results, aes(x=Feature, y=Importance,fill=Importance))+
  geom_bar(stat="identity")+
  scale_fill_gradient(low="grey20",high="orangered")+
  ggtitle("XGBoost Variable Importance")+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



17 Using the caret package fit and evaluate 1 other ML model on this data.

Fitting Vanilla Neural Network

```
### REMOVE SUBSAMPLING FOR FINAL MODEL
data.prep <- raw.data %>% select(-AppointmentID, -PatientID) #%>% sample_n(1000)

set.seed(42)
data.split <- initial_split(data.prep, prop = 0.7)
train  <- training(data.split)
test <- testing(data.split)

fit.control <- trainControl(method="cv",number=3,
                            classProbs = TRUE, summaryFunction = twoClassSummary)
```

```
set.seed(123)
neural.network <- train( NoShow ~.,
                         data = train,
                         method  = "nnet",
                         na.action = na.omit,
                         trControl = fit.control,
                         trace = FALSE)
```

```
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was not
```

```
## in the result set. ROC will be used instead.
```

again for changing the null levels on Neighbourhood to its real ones for both train and test.

```
train$Neighbourhood<- as.factor(train$Neighbourhood)

levels(train$Neighbourhood)
```

```
##  [1] "AEROPORTO"                   "ANDORINHAS"
##  [3] "ANTÔNIO HONÓRIO"             "ARIOVALDO FAVALESSA"
##  [5] "BARRO VERMELHO"              "BELA VISTA"
##  [7] "BENTO FERREIRA"              "BOA VISTA"
##  [9] "BONFIM"                      "CARATOÍRA"
## [11] "CENTRO"                      "COMDUSA"
## [13] "CONQUISTA"                   "CONSOLAÇÃO"
## [15] "CRUZAMENTO"                  "DA PENHA"
## [17] "DE LOURDES"                  "DO CABRAL"
## [19] "DO MOSCOSO"                  "DO QUADRO"
## [21] "ENSEADA DO SUÁ"              "ESTRELINHA"
## [23] "FONTE GRANDE"                "FORTE SÃO JOÃO"
## [25] "FRADINHOS"                   "GOIABEIRAS"
## [27] "GRANDE VITÓRIA"              "GURIGICA"
## [29] "HORTO"                       "ILHA DAS CAIEIRAS"
## [31] "ILHA DE SANTA MARIA"         "ILHA DO BOI"
## [33] "ILHA DO FRADE"               "ILHA DO PRÍNCIPE"
## [35] "ILHAS OCEÂNICAS DE TRINDADE" "INHANGUETÁ"
## [37] "ITARARÉ"                     "JABOUR"
## [39] "JARDIM CAMBURI"              "JARDIM DA PENHA"
## [41] "JESUS DE NAZARETH"           "JOANA D´ARC"
## [43] "JUCUTUQUARA"                 "MARIA ORTIZ"
## [45] "MÁRIO CYPRESTE"              "MARUÍPE"
## [47] "MATA DA PRAIA"               "MONTE BELO"
## [49] "MORADA DE CAMBURI"           "NAZARETH"
## [51] "NOVA PALESTINA"              "PARQUE MOSCOSO"
## [53] "PIEDADE"                     "PONTAL DE CAMBURI"
## [55] "PRAIA DO CANTO"              "PRAIA DO SUÁ"
## [57] "REDENÇÃO"                    "REPÚBLICA"
## [59] "RESISTÊNCIA"                 "ROMÃO"
## [61] "SANTA CECÍLIA"               "SANTA CLARA"
## [63] "SANTA HELENA"                "SANTA LÚCIA"
## [65] "SANTA LUÍZA"                 "SANTA MARTHA"
## [67] "SANTA TEREZA"                "SANTO ANDRÉ"
## [69] "SANTO ANTÔNIO"               "SANTOS DUMONT"
## [71] "SANTOS REIS"                 "SÃO BENEDITO"
## [73] "SÃO CRISTÓVÃO"               "SÃO JOSÉ"
## [75] "SÃO PEDRO"                   "SEGURANÇA DO LAR"
## [77] "SOLON BORGES"                "TABUAZEIRO"
## [79] "UNIVERSITÁRIO"               "VILA RUBIM"
```

```
test$Neighbourhood<- as.factor(test$Neighbourhood)

levels(test$Neighbourhood)
```

```
##  [1] "AEROPORTO"           "ANDORINHAS"          "ANTÔNIO HONÓRIO"
##  [4] "ARIOVALDO FAVALESSA" "BARRO VERMELHO"      "BELA VISTA"
##  [7] "BENTO FERREIRA"      "BOA VISTA"           "BONFIM"
## [10] "CARATOÍRA"           "CENTRO"              "COMDUSA"
## [13] "CONQUISTA"           "CONSOLAÇÃO"          "CRUZAMENTO"
## [16] "DA PENHA"            "DE LOURDES"          "DO CABRAL"
## [19] "DO MOSCOSO"          "DO QUADRO"           "ENSEADA DO SUÁ"
## [22] "ESTRELINHA"          "FONTE GRANDE"        "FORTE SÃO JOÃO"
## [25] "FRADINHOS"           "GOIABEIRAS"          "GRANDE VITÓRIA"
## [28] "GURIGICA"            "HORTO"               "ILHA DAS CAIEIRAS"
## [31] "ILHA DE SANTA MARIA" "ILHA DO BOI"         "ILHA DO FRADE"
## [34] "ILHA DO PRÍNCIPE"    "INHANGUETÁ"          "ITARARÉ"
## [37] "JABOUR"              "JARDIM CAMBURI"      "JARDIM DA PENHA"
## [40] "JESUS DE NAZARETH"   "JOANA D´ARC"         "JUCUTUQUARA"
## [43] "MARIA ORTIZ"         "MÁRIO CYPRESTE"      "MARUÍPE"
## [46] "MATA DA PRAIA"       "MONTE BELO"          "MORADA DE CAMBURI"
## [49] "NAZARETH"            "NOVA PALESTINA"      "PARQUE INDUSTRIAL"
## [52] "PARQUE MOSCOSO"      "PIEDADE"             "PONTAL DE CAMBURI"
## [55] "PRAIA DO CANTO"      "PRAIA DO SUÁ"        "REDENÇÃO"
## [58] "REPÚBLICA"           "RESISTÊNCIA"         "ROMÃO"
## [61] "SANTA CECÍLIA"       "SANTA CLARA"         "SANTA HELENA"
## [64] "SANTA LÚCIA"         "SANTA LUÍZA"         "SANTA MARTHA"
## [67] "SANTA TEREZA"        "SANTO ANDRÉ"         "SANTO ANTÔNIO"
## [70] "SANTOS DUMONT"       "SANTOS REIS"         "SÃO BENEDITO"
## [73] "SÃO CRISTÓVÃO"       "SÃO JOSÉ"            "SÃO PEDRO"
## [76] "SEGURANÇA DO LAR"    "SOLON BORGES"        "TABUAZEIRO"
## [79] "UNIVERSITÁRIO"       "VILA RUBIM"
```

as we can see in above the levels of Neighbourhood in both test and train differ from each other. it happens because of splitting non shuffled data. so may be a portion including a special level goes to test portion which has not appeared in train. so we have to delete the difference to see the performance by confusion matrix.

```
test <- subset(test,Neighbourhood !=  "PARQUE INDUSTRIAL")
```

also same as before we have to use the following code to change the levels of test$Noshow from null to its real levels.

```
test$NoShow<- as.factor(test$NoShow)
```

now lets check the outputs and the performance:

```
# Best tuning parameter mtry
neural.network$bestTune
```

```
##   size decay
## 3    1   0.1
```

```
# Make predictions on the test data
predictions <- neural.network %>% predict(test)
head(predictions)
```

```
## [1] No No No No No No
## Levels: No Yes
```

```
head(test$NoShow)
```

```
## [1] No No No No No No
## Levels: No Yes
```

```
nn.pred <- predict(neural.network, newdata=test)
nn.probs <- predict(neural.network, newdata=test, type="prob")

test <- test %>% mutate(NoShow.numerical = ifelse(NoShow=="Yes",1,0))
confusionMatrix(nn.pred, test$NoShow, positive="Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    No   Yes
##        No  26526  6632
##        Yes     0     0
##
##                Accuracy : 0.8
##                  95% CI : (0.7956, 0.8043)
##     No Information Rate : 0.8
##     P-Value [Acc > NIR] : 0.5033
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.0
##             Specificity : 1.0
##          Pos Pred Value : NaN
##          Neg Pred Value : 0.8
##              Prevalence : 0.2
##          Detection Rate : 0.0
##    Detection Prevalence : 0.0
##       Balanced Accuracy : 0.5
##
##        'Positive' Class : Yes
##
```

```
paste("neural network Area under ROC Curve: ", round(auc(test$NoShow.numerical, nn.probs[,2]),3), sep="
```

```
## Setting levels: control = 0, case = 1
```

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

```
## [1] "neural network Area under ROC Curve: 0.5"
```

18 Based on everything, do you think we can trust analyses based on this dataset? Explain your reasoning.

actually the percision is not too high, so we can not trust with high level to this models. The provided models have low accuracy and perform poorly in identifying positive instances. Their sensitivity is extremely low, and in the case of the second model, it fails to identify any positive instances at all. Therefore, we cannot trust the analyses based on these models as their performance is not reliable.

For the first model, the accuracy is 0.8023, which means it correctly classifies approximately 80.23% of the instances. However, when we examine the sensitivity (also known as recall or true positive rate), it is extremely low at 0.030848, indicating that the model has a poor ability to correctly identify positive instances (Yes). The specificity (true negative rate) is relatively high at 0.994462, suggesting a good ability to correctly identify negative instances (No). However, the low sensitivity severely impacts the overall performance of the model.

Similarly, the second model has an accuracy of 0.8006, which is slightly lower than the first model. However, the sensitivity is 0.0000, indicating that the model fails to identify any positive instances correctly. This is further supported by the fact that the positive predictive value is NaN (not a number), which indicates that the model does not make any positive predictions at all. The specificity remains at 1.0000, meaning the model correctly identifies all negative instances.