This practical is based on exploratory data analysis and prediction of a dataset derived from a municipal database of healthcare administrative data. This dataset is derived from Vitoria, the capital city of Espírito Santo, Brazil (population 1.8 million) and was freely shared under a creative commons license.

Generate an rmarkdown report that contains all the necessary code to document and perform: EDA, prediction of no-shows using XGBoost, and an analysis of variable/feature importance using this data set. Ensure your report includes answers to any questions marked in bold. Please submit your report via brightspace as a link to a git repository containing the rmarkdown and compiled/knitted html version of the notebook.

Introduction

The Brazilian public health system, known as SUS for Unified Health System in its acronym in Portuguese, is one of the largest health system in the world, representing government investment of more than 9% of GDP. However, its operation is not homogeneous and there are distinct perceptions of quality from citizens in different regions of the country. Non-attendance of medical appointments contributes a significant additional burden on limited medical resources. This analysis will try and investigate possible factors behind non-attendance using an administrative database of appointment data from Vitoria, Espírito Santo, Brazil.

The data required is available via the course website.

Understanding the data

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

PatientID: Unique identifier for each patient.

AppointmentID: Unique identifier to each appointment .

Gender: Patient Gender (limited to Male or Female).

ScheduledDate: date on which the appointment was scheduled.

AppointmentDate: date of the actual appointment.

Age: Patient age.

Neighbourhood: District of Vitória in which the appointment.

SocialWelfare: Patient is a recipient of Bolsa Família welfare payments.

Hypertension: Patient previously diagnoised with hypertensio (Boolean) . Diabetes: Patient previously diagnosed with diabetes (Boolean).

AlcoholUseDisorder: Patient previously diagnosed with alcohol use disorder (Boolean).

Disability: Patient previously diagnosed with a disability (severity rated 0-4).

SMSReceived: At least 1 reminder text sent before appointment (Boolean).

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?

1. Age: Generally, younger patients are more likely to ignore the disease, leading to absences or being unable to attend appointments due to busy work.
2. Disability: People with physical disabilities are more likely to be absent due to mobility issues.
3. SMSReceived: Failure to receive the confirmation text message may cause people to forget to record the appointment time in the plan, and finally forget to attend the appointment.

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. Is patientID related to age, gender, etc.? Is it randomly generated?
2. Are there any correlation between AppointmentID and date? Are they randomly generated?
3. What kind of disability does Disability refer to specifically? Hands, legs or other parts?

Data Parsing and Cleaning

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

```
# raw.data <- read_csv('2016_05v2_VitoriaAppointmentData.csv',
# col_types='fffTTifllllflf')
url <- paste0("https://maguire-lab.github.io/health_data_science_research_2024/static_files/",
    "practicals/lab1_data/2016_05v2_VitoriaAppointmentData.csv")
raw.data <- readr::read_csv(url)
```

```
## 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.
```

Now we need to check data is valid: because we specified col_types and the data parsed without error most of our data seems to at least be formatted as we expect i.e., ages are integers

```
raw.data %>% filter(Age > 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>
```

We can see there are 2 patient's older than 110 which seems suspicious but we can't actually say if this is impossible.

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

```
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>
```

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

Exploratory Data Analysis

First, we should get an idea if the data meets our expectations, there are newborns in the data (`Age==0`) and we wouldn't expect any of these to be diagnosed with Diabetes, Alcohol Use Disorder, and Hypertension (although in theory it could be possible). We can easily check this:

```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
```

Patient(PatientID: 8.221459e+14) has most times appointments, total 88 times.

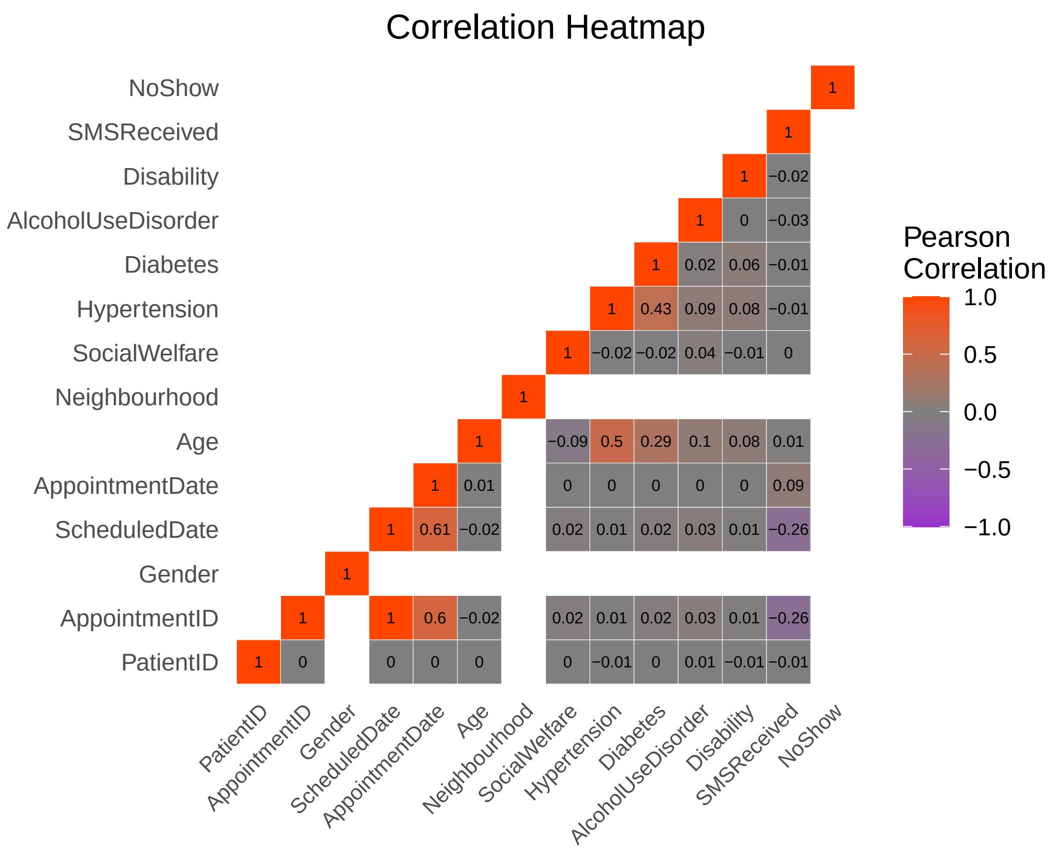
Let's explore the correlation between variables:

```r
# 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))
```

```
## Warning: There were 3 warnings in `mutate()`.
## The first warning was:
## i In argument: `Gender = (function (x) ...`.
## Caused by warning:
```

```
## ! NAs introduced by coercion
## i Run `dplyr::last_dplyr_warnings()` to see the 2 remaining warnings.
```

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



Correlation heatmaps are useful for identifying linear relationships between variables/features. In this case, we are particularly interested in relationships between `NoShow` and any specific variables.

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

ScheduluedDate most strongly correlate with missing appointments with -0.16 correlation.

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

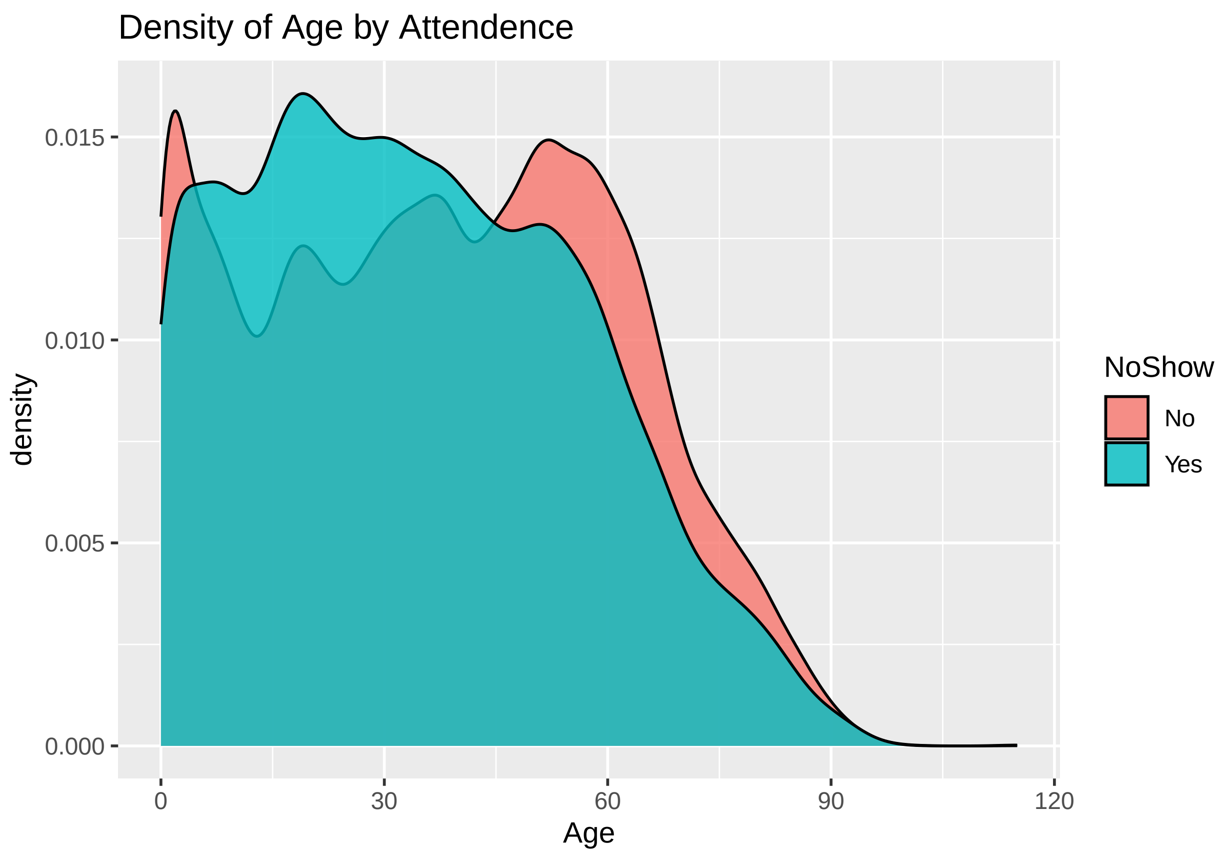
PatientID and AppointmentID strongly correlate with each other with 0.65 correlation ratio.

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

There is no connection between PatientID/AppointmentID and predicted values, and adding them to the heatmap has no meaning.

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

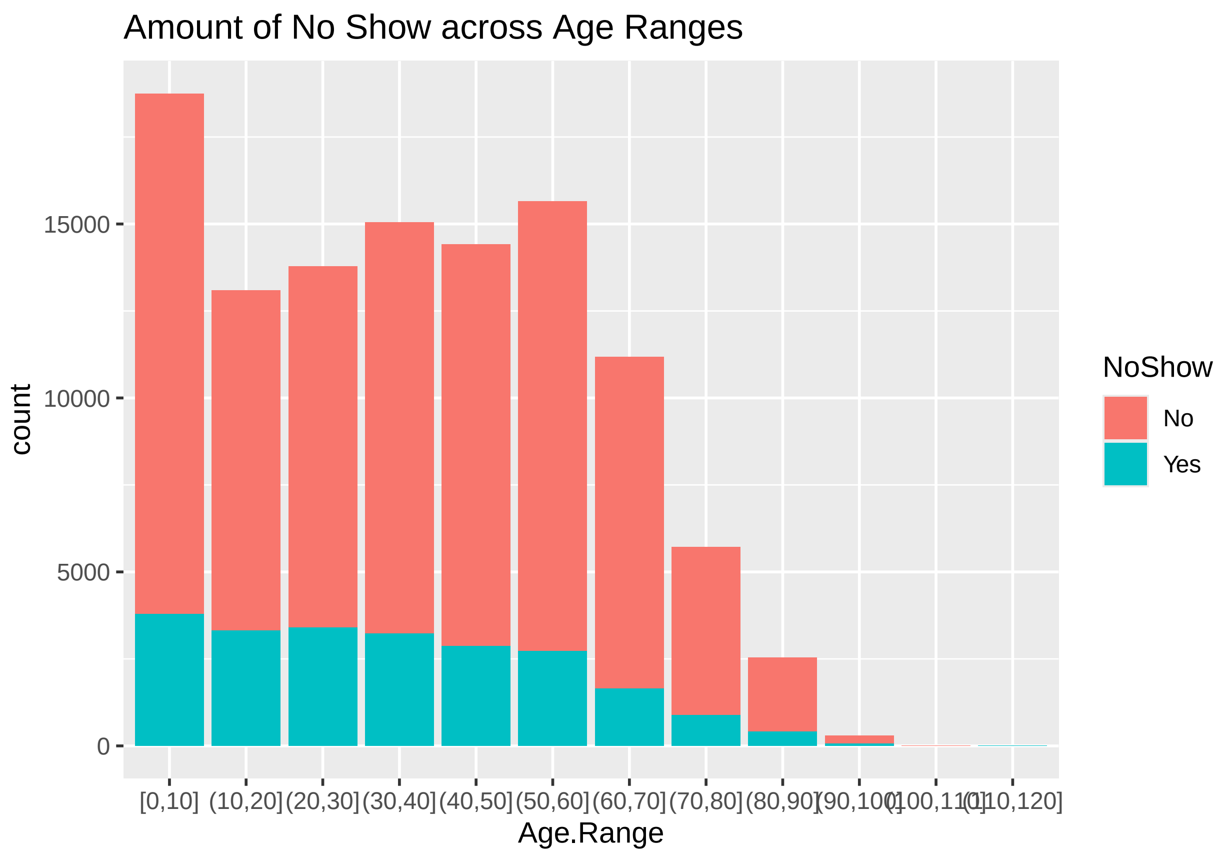
```r
ggplot(raw.data) +
  geom_density(aes(x=Age, fill=NoShow), alpha=0.8) +
  ggtitle("Density of Age by Attendence")
```



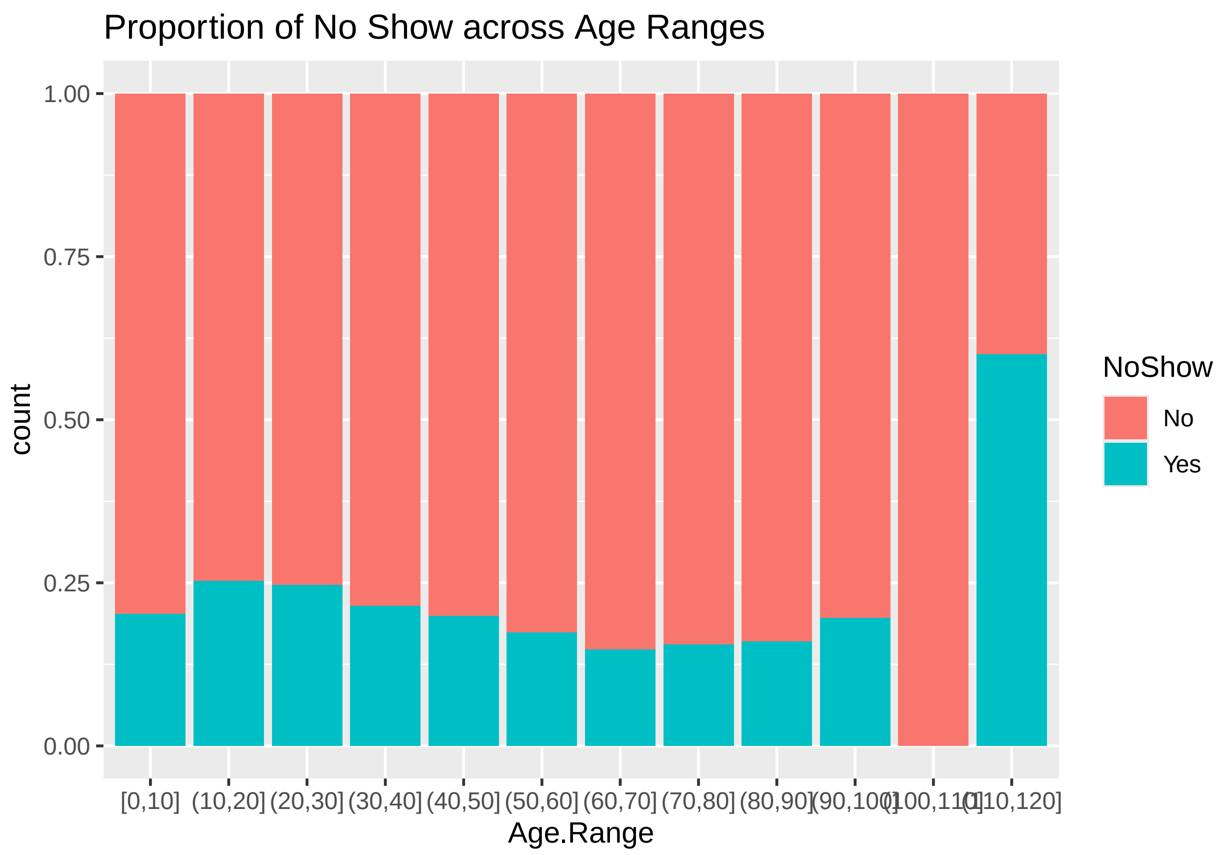
There does seem to be a difference in the distribution of ages of people that miss and don't miss appointments. However, the shape of this distribution means the actual correlation is near 0 in the heatmap above. This highlights the need to look at individual variables.

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")
```

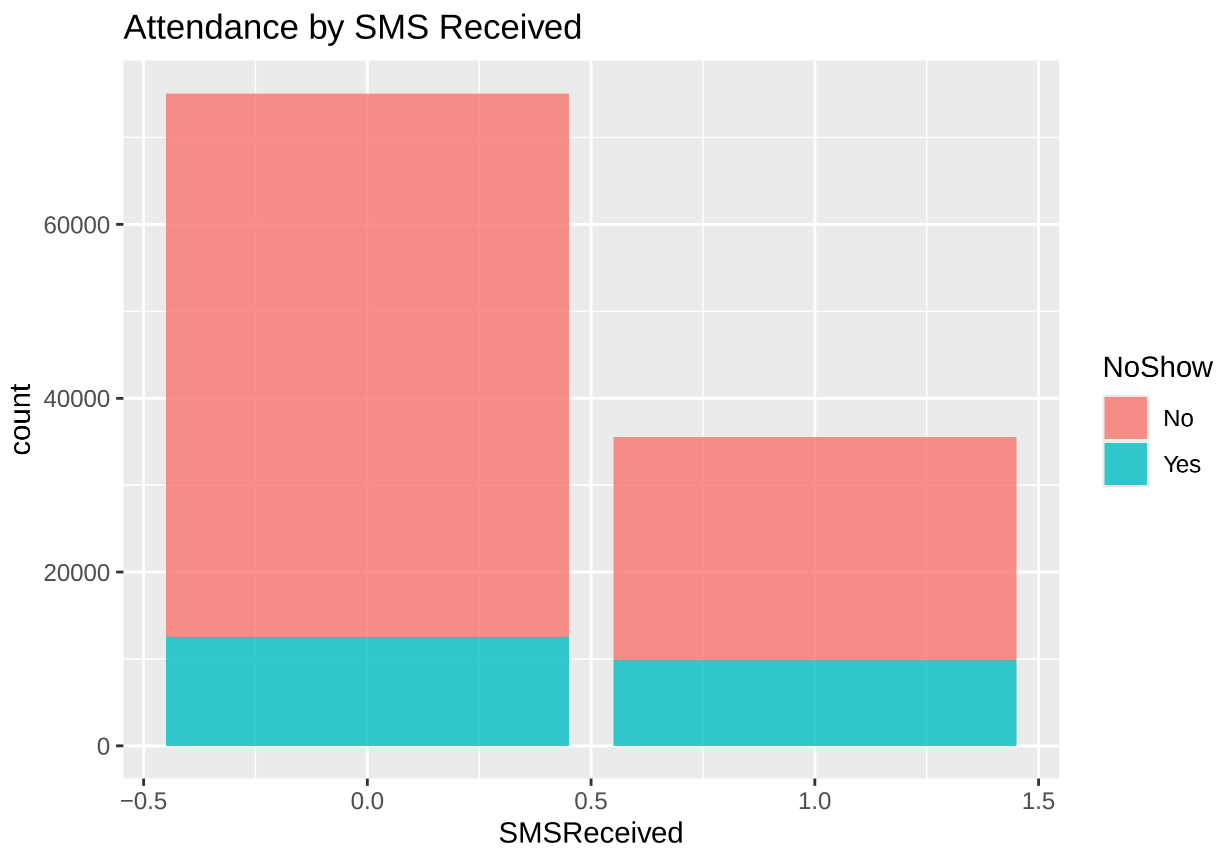


10 How could you be misled if you only plotted 1 of these 2 plots of attendance by age group?

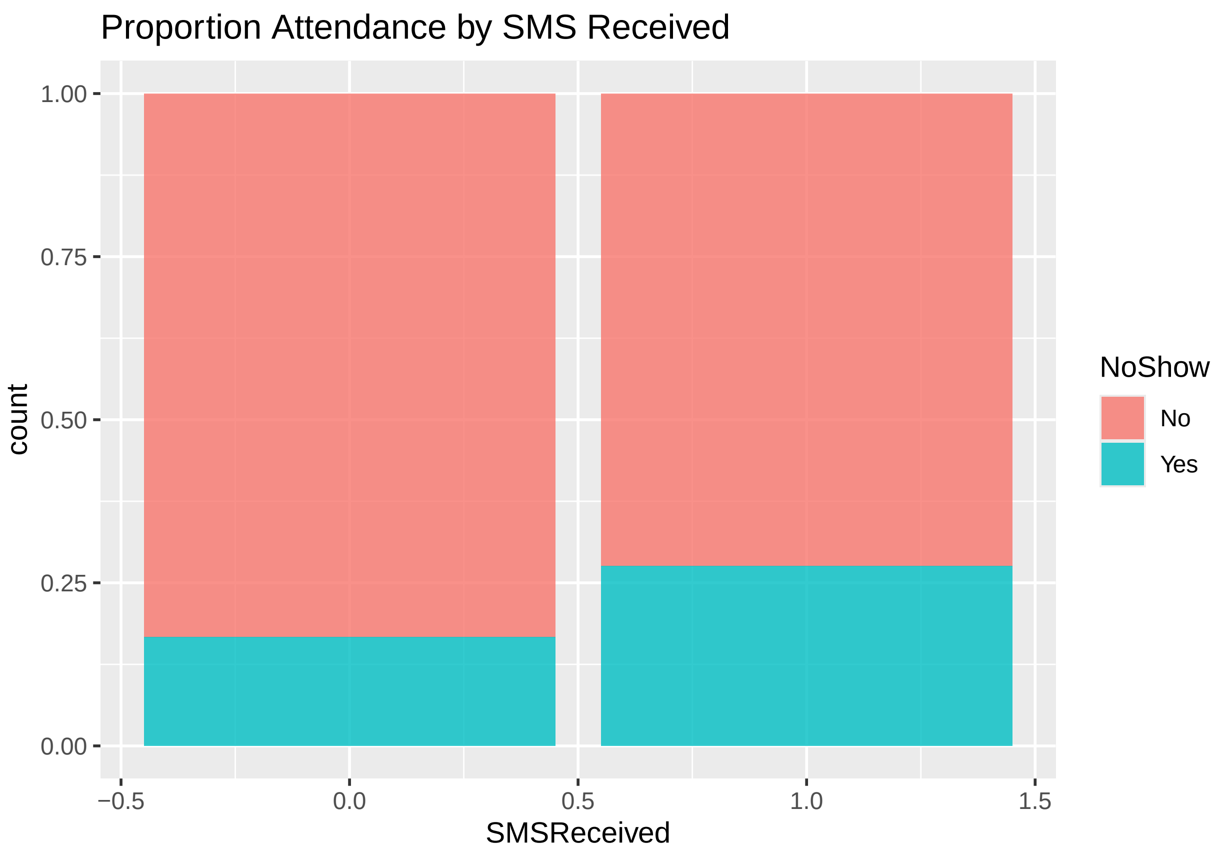
Because the number of people in each age group is different, it is impossible to intuitively see the proportion of no show in each age group from the plot picture and compare these age groups, so the second plot is needed.

Next, we'll have a look at `SMSReceived` variable:

```r
ggplot(raw.data) +
  geom_bar(aes(x=SMSReceived, fill=NoShow), alpha=0.8) +
  ggtitle("Attendance by SMS Received")
```



```r
ggplot(raw.data) +
  geom_bar(aes(x=SMSReceived, fill=NoShow), position='fill', alpha=0.8) +
  ggtitle("Proportion Attendance by SMS Received")
```

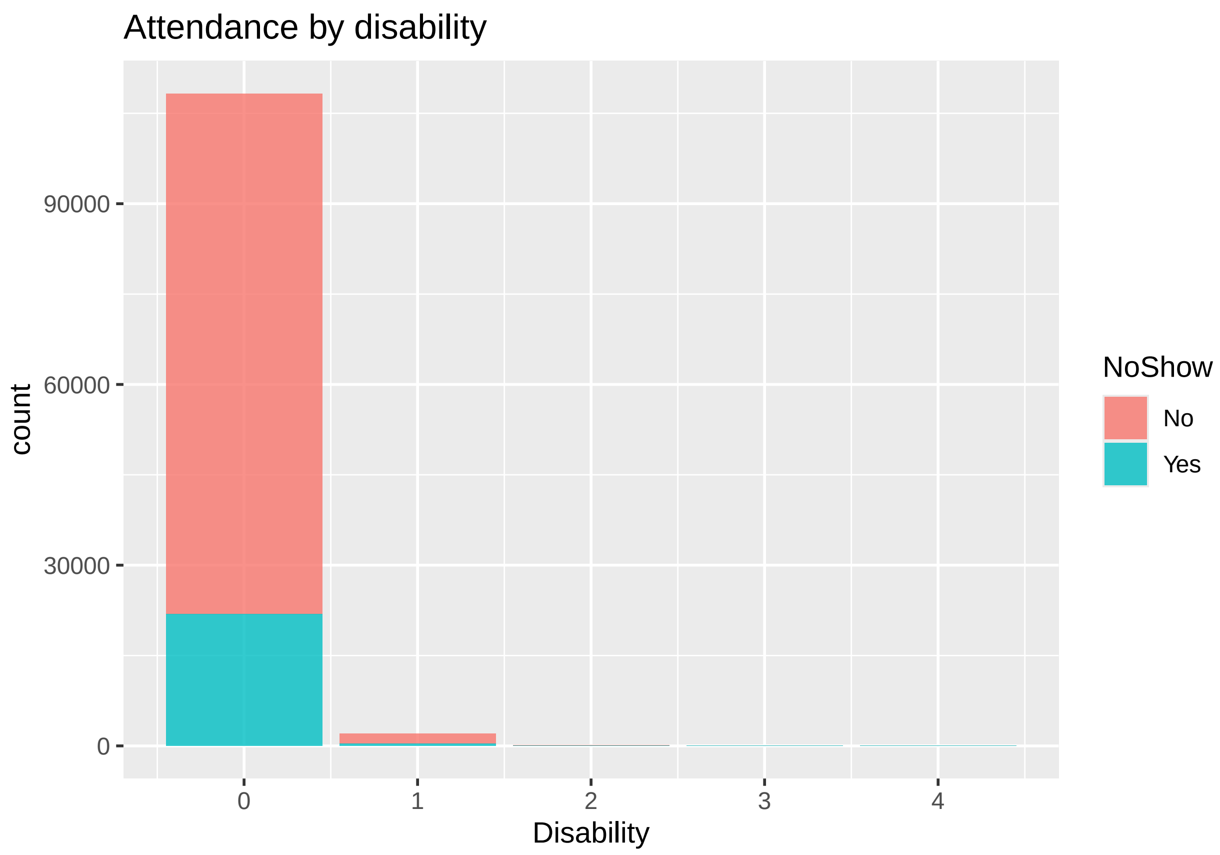


11 From this plot does it look like SMS reminders increase or decrease the chance of someone not attending an appointment? Why might the opposite actually be true (hint: think about biases)?

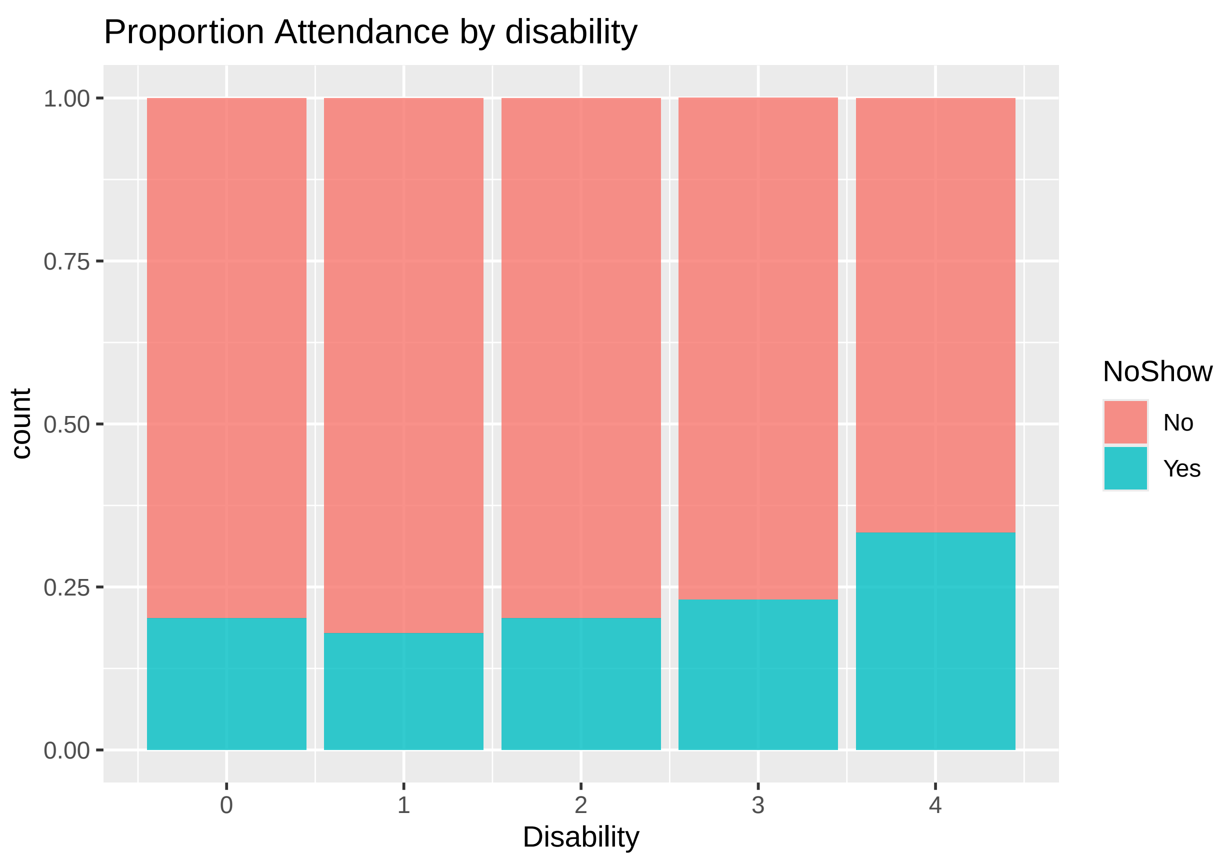
1. SMS reminders may not be randomly assigned. For example, text message reminders may be more likely to be sent to those who historically have missed more appointments. Therefore, people who are already more punctual may not need text reminders at all.
2. Some people may actively choose not to receive text message reminders because they think they do not need these reminders. These people are usually also those with lower absentee rates.

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

```r
ggplot(raw.data) +
  geom_bar(aes(x=Disability, fill=NoShow), alpha=0.8) +
  ggtitle("Attendance by disability")
```

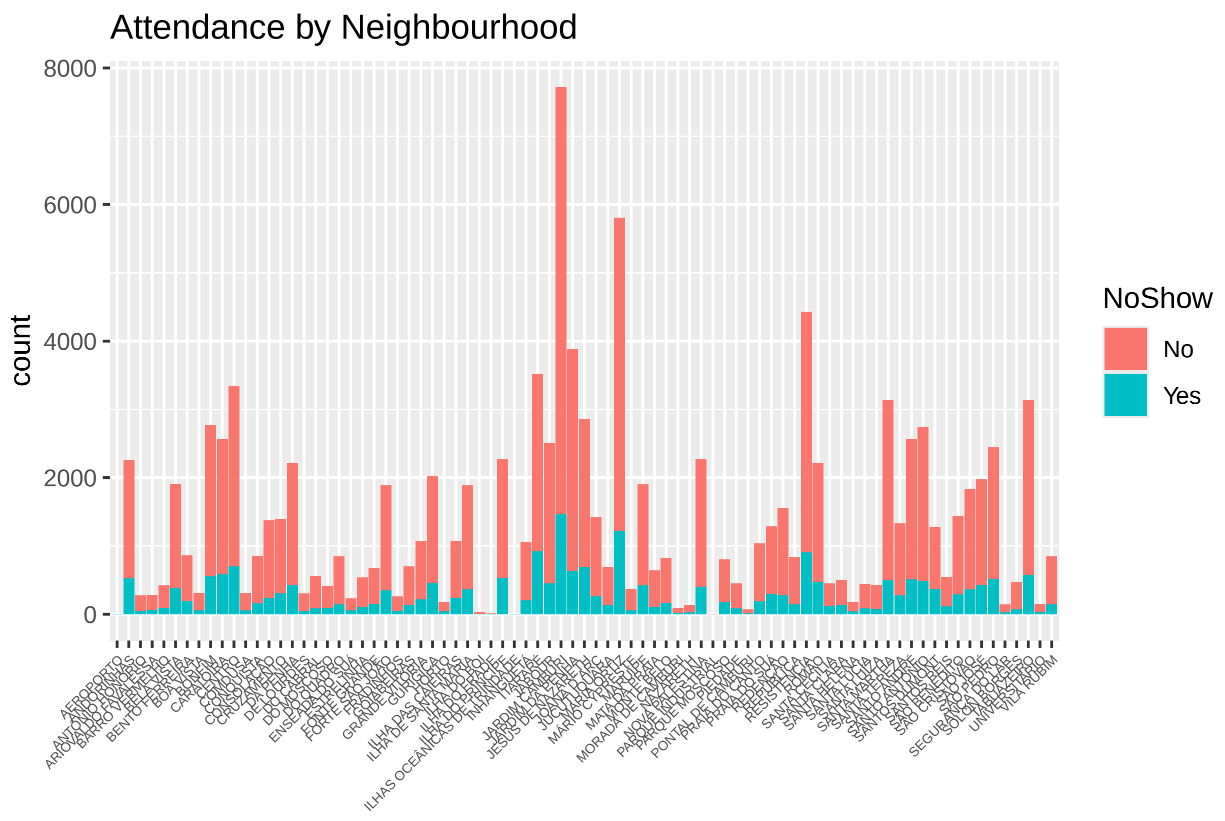


```r
ggplot(raw.data) +
  geom_bar(aes(x=Disability, fill=NoShow), position='fill', alpha=0.8) +
  ggtitle("Proportion Attendance by disability")
```

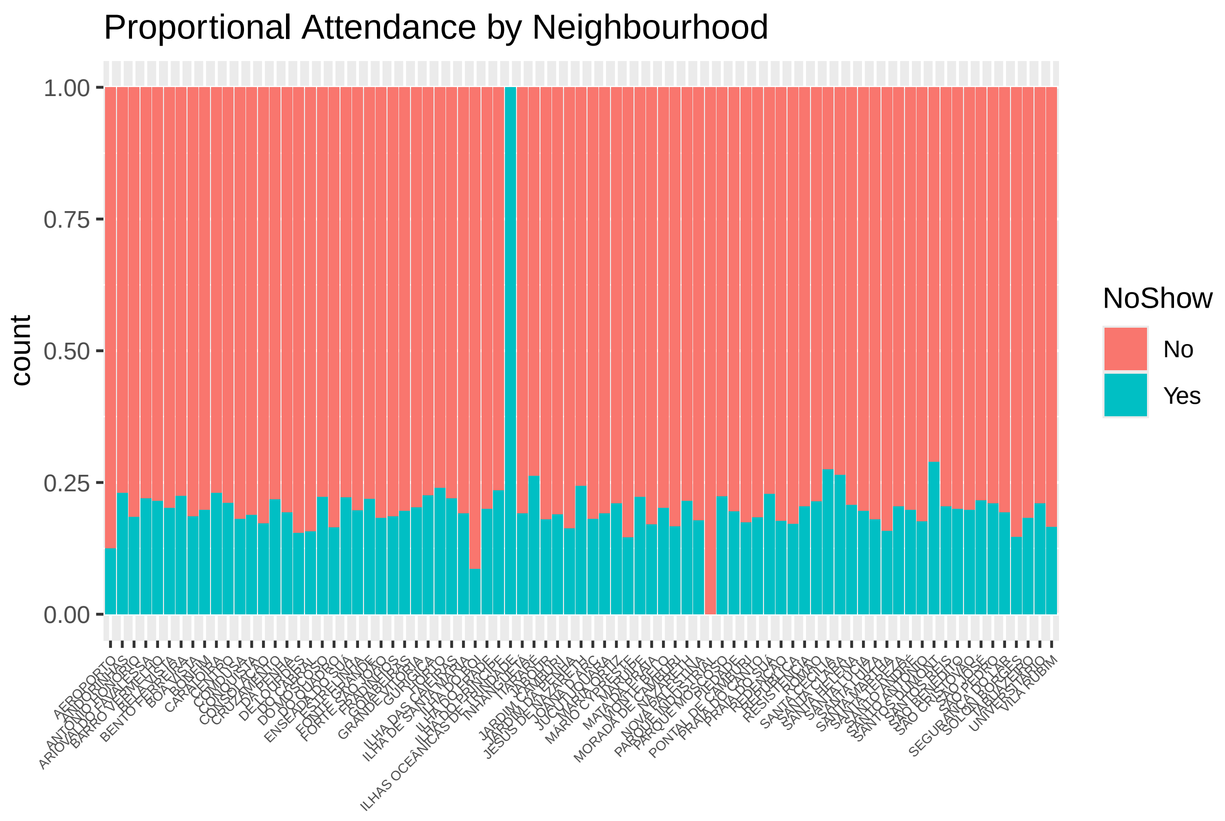


Now let's look at the neighbourhood data as location can correlate highly with many social determinants of health.

```r
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')
```



```r
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')
```



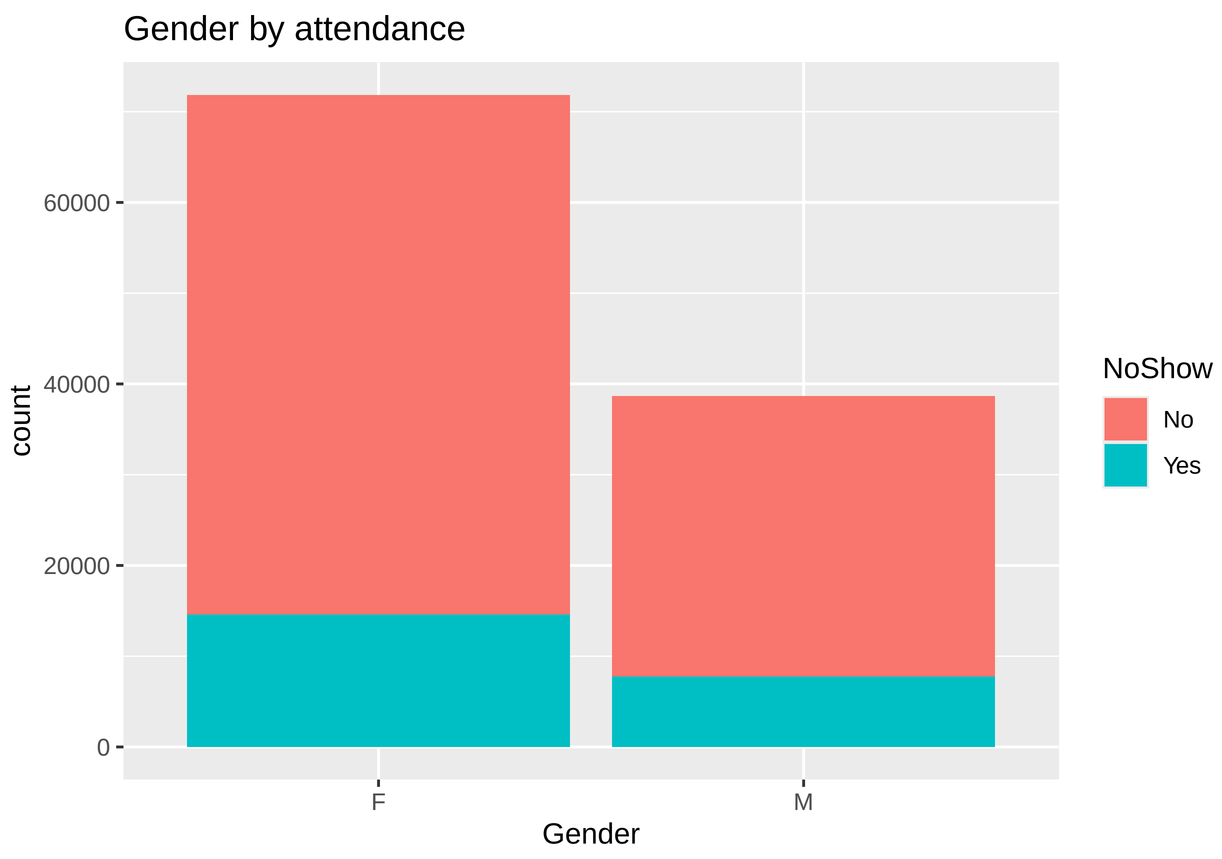
Most neighborhoods have similar proportions of no-show but some have much higher and lower rates.

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

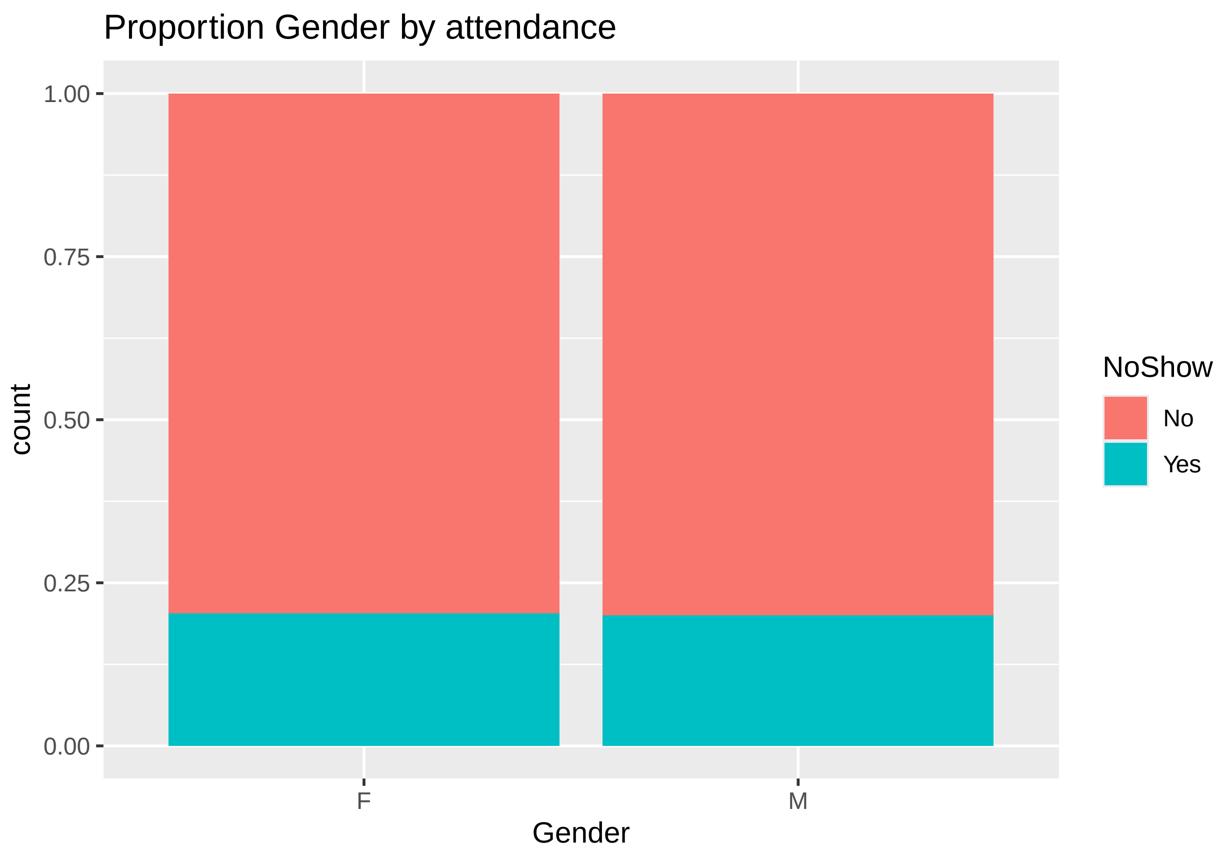
Because the sample size of those two special neighborhoods is too small, the proportion has no reference significance.

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

```r
ggplot(raw.data) +
  geom_bar(aes(x=Gender, fill=NoShow))+
  ggtitle("Gender by attendance")
```

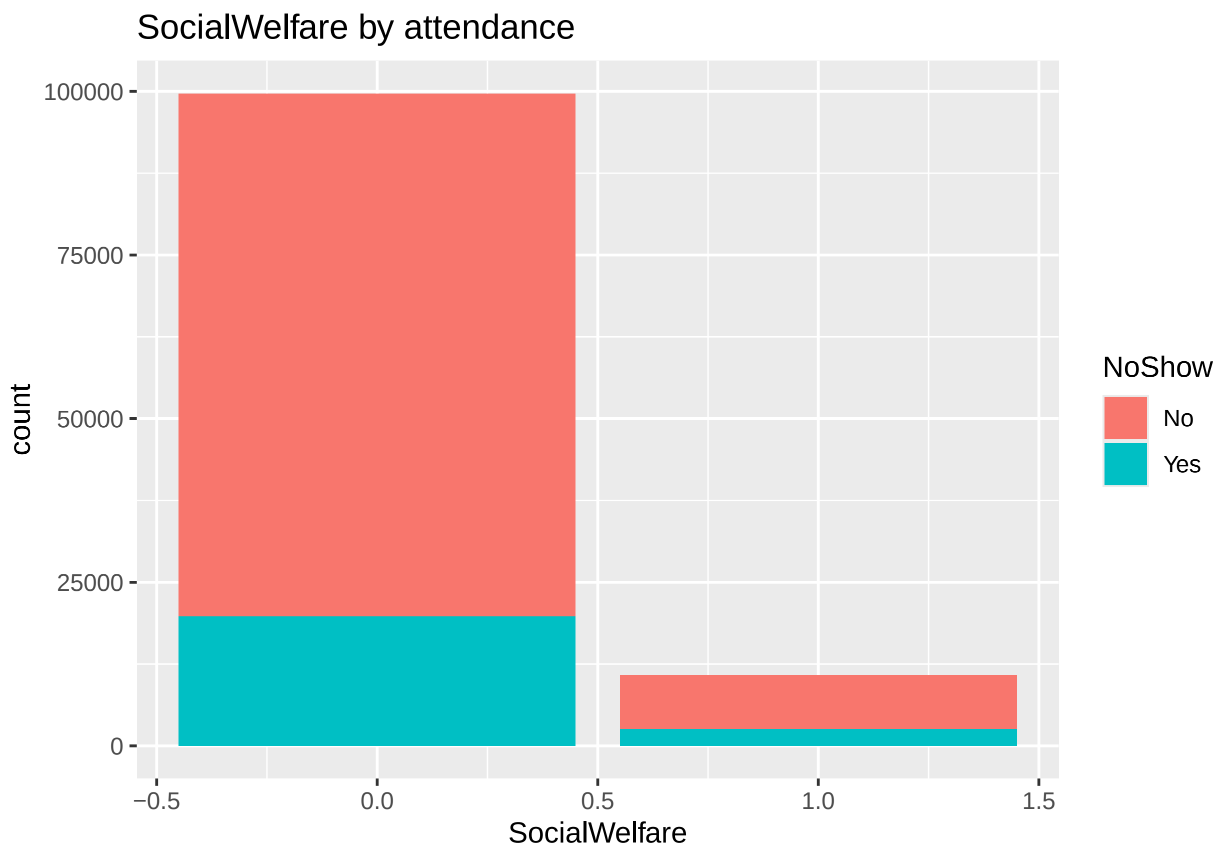


```r
ggplot(raw.data) +
  geom_bar(aes(x=Gender, fill=NoShow), position='fill')+
  ggtitle("Proportion Gender by attendance")
```

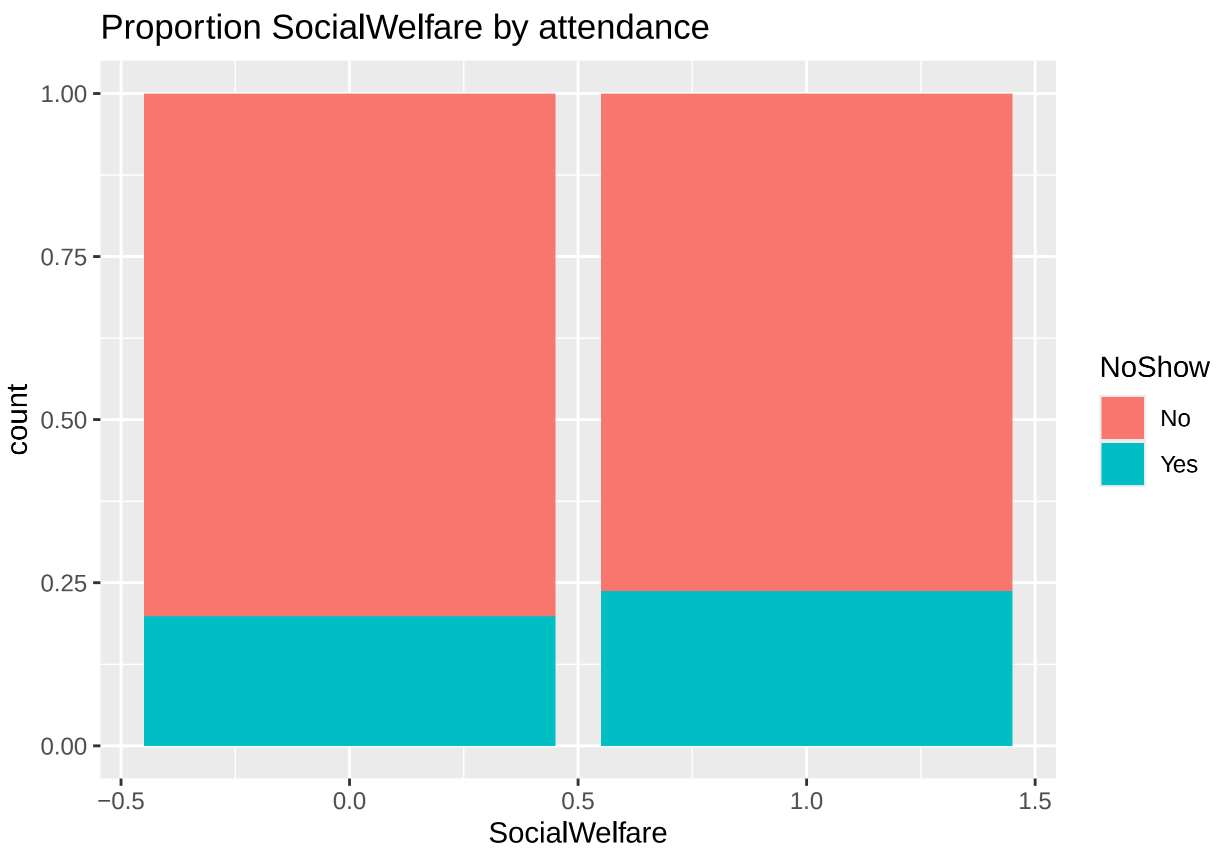


14 Create a similar plot using `SocialWelfare`

```r
ggplot(raw.data) +
  geom_bar(aes(x=SocialWelfare, fill=NoShow))+
  ggtitle("SocialWelfare by attendance")
```



```
ggplot(raw.data) +
  geom_bar(aes(x=SocialWelfare, fill=NoShow), position='fill')+
  ggtitle("Proportion SocialWelfare by attendance")
```



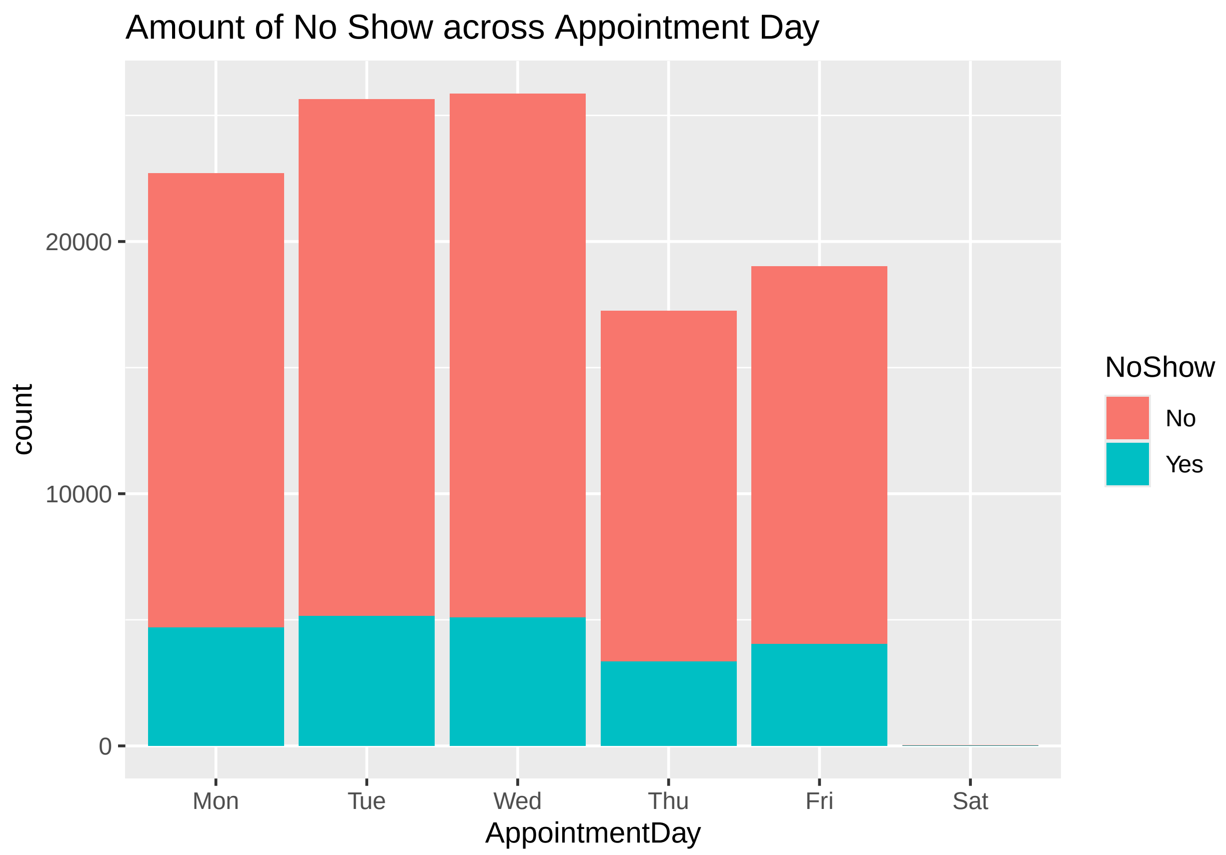
Far more exploration could still be done, including dimensionality reduction approaches but although we have found some patterns there is no major/striking patterns on the data as it currently stands.

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

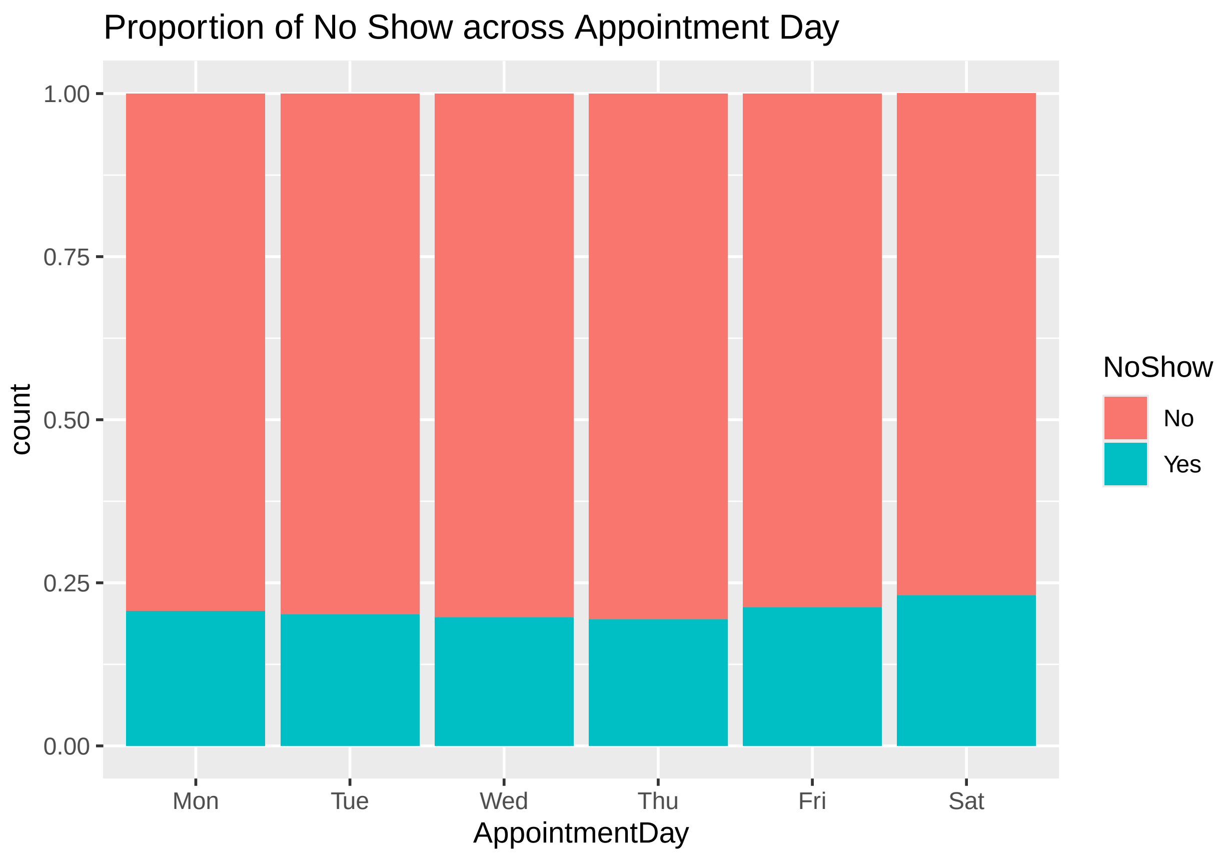
Feature Engineering

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")
```



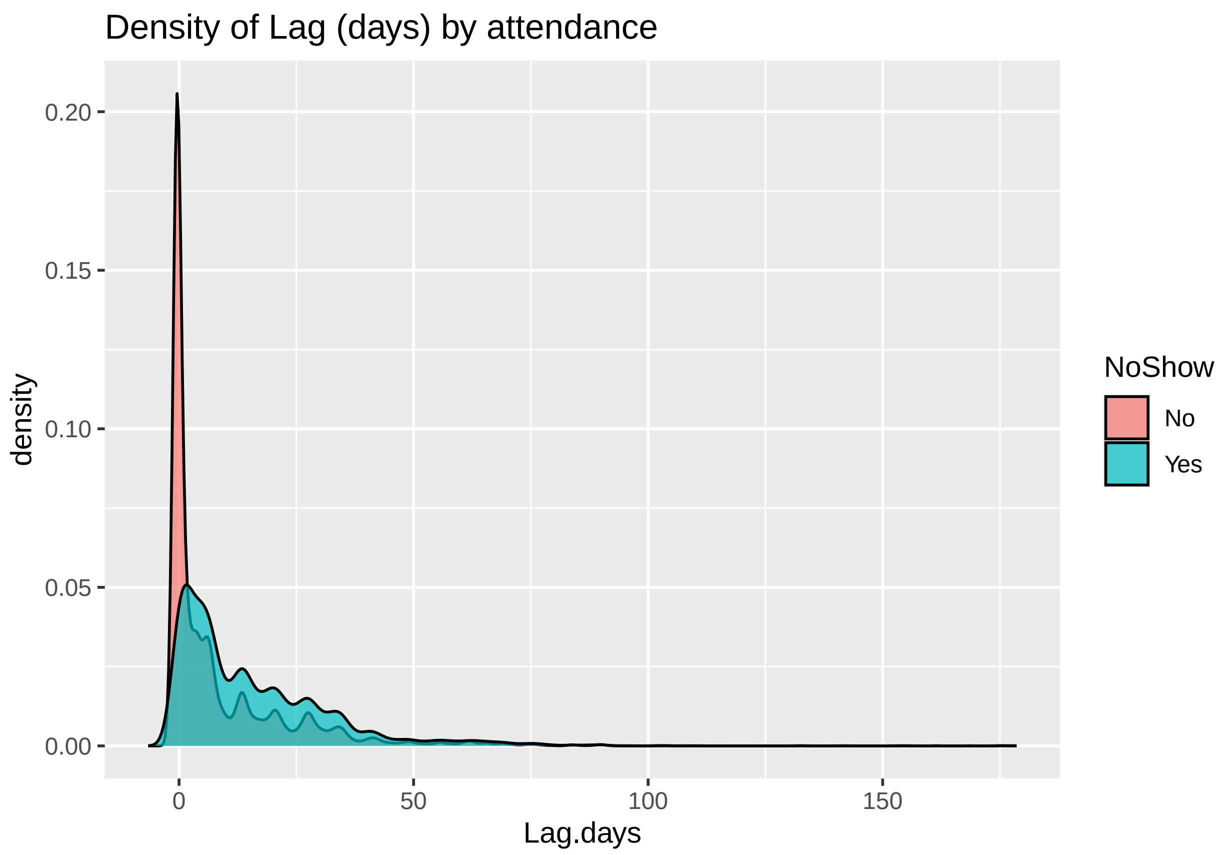
```r
ggplot(raw.data) +
  geom_bar(aes(x=AppointmentDay, fill=NoShow), position = 'fill') +
  ggtitle("Proportion of No Show across Appointment Day")
```



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"))

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 Have a look at the values in lag variable, does anything seem odd?

I can understand that when the lag is low, everyone will remember to make an appointment, so the proportion of no shows is much smaller than that of show. But I can't understand why as the lag increases, the proportion of no show does not increase significantly.

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.

```
### REMOVE SUBSAMPLING FOR FINAL MODEL
data.prep <- raw.data %>% select(-AppointmentID, -PatientID) #%>% sample_n(10000)
data.prep$Neighbourhood <- factor(data.prep$Neighbourhood)
levels(data.prep$Neighbourhood)[table(data.prep$Neighbourhood) < 10] <- "Other"
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?

Based on the EDA, many features seem to be obviously related to no show. Therefore, I think the accuracy of the model should be very high.

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)

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))
xgb.pred <- as.factor(xgb.pred)
test$NoShow <- as.factor(test$NoShow)
levels(xgb.pred) <- levels(test$NoShow)
confusionMatrix(xgb.pred, test$NoShow, positive="Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    No   Yes
##        No  26433  6432
##        Yes   112   181
##
##                Accuracy : 0.8026
##                  95% CI : (0.7983, 0.8069)
##     No Information Rate : 0.8006
##     P-Value [Acc > NIR] : 0.1733
##
##                   Kappa : 0.0361
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.027370
##             Specificity : 0.995781
##          Pos Pred Value : 0.617747
##          Neg Pred Value : 0.804290
##              Prevalence : 0.199439
##          Detection Rate : 0.005459
##    Detection Prevalence : 0.008836
##       Balanced Accuracy : 0.511576
##
##        'Positive' Class : Yes
##
```

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

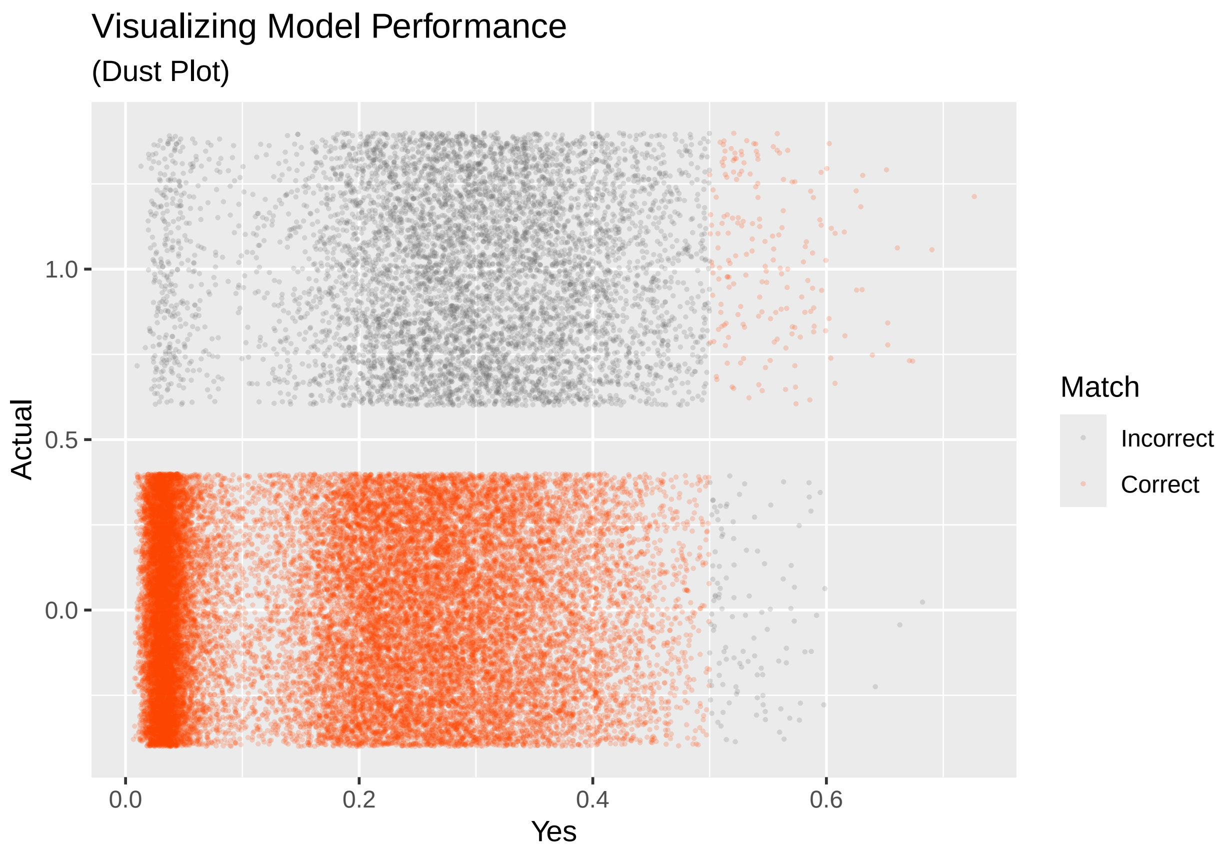
```
## Setting levels: control = 0, case = 1
```

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

```
## [1] "XGBoost Area under ROC Curve: 0.743"
```

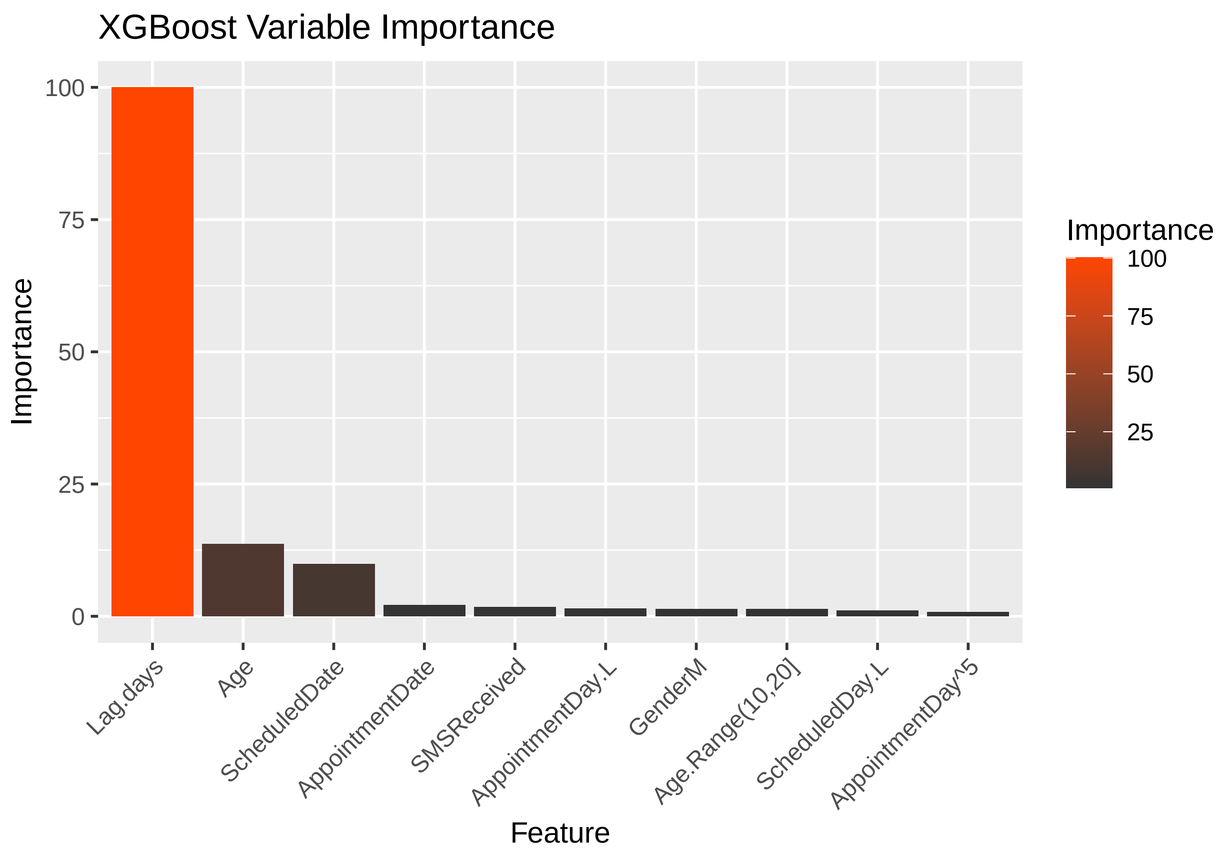
This isn't an unreasonable performance, but let's look a bit more carefully at the correct and incorrect predictions,

```r
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:

```r
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.

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

                              summaryFunction = twoClassSummary)
rf_model <- train(NoShow ~ .,
                  data = train,
                  method = "rf",
                  trControl = fit.control,
                  metric = "ROC",
                  ntree = 100)  # Reduce the number of trees
print(rf_model)
```

```
## Random Forest
##
## 77368 samples
##    16 predictor
##     2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (3 fold)
## Summary of sample sizes: 51578, 51579, 51579
## Resampling results across tuning parameters:
##
##   mtry  ROC        Sens       Spec
##     2   0.6785263  1.0000000  0.0000000
##    57   0.7356683  0.9551912  0.1640135
##   113   0.7315213  0.9537803  0.1677704
```

```
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 57.
```

```
rf_probs <- predict(rf_model, test, type = "prob")
rf_preds <- predict(rf_model, test)

rf_preds <- factor(rf_preds, levels = levels(test$NoShow))

confusion_matrix <- confusionMatrix(rf_preds, test$NoShow)
print(confusion_matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    No   Yes
##        No  25398  5381
##        Yes  1147  1232
##
##                Accuracy : 0.8031
##                  95% CI : (0.7988, 0.8074)
##     No Information Rate : 0.8006
##     P-Value [Acc > NIR] : 0.1227
##
##                   Kappa : 0.1884
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9568
##             Specificity : 0.1863
##          Pos Pred Value : 0.8252
##          Neg Pred Value : 0.5179
##              Prevalence : 0.8006
##          Detection Rate : 0.7660
##    Detection Prevalence : 0.9283
##       Balanced Accuracy : 0.5715
##
##        'Positive' Class : No
##
```

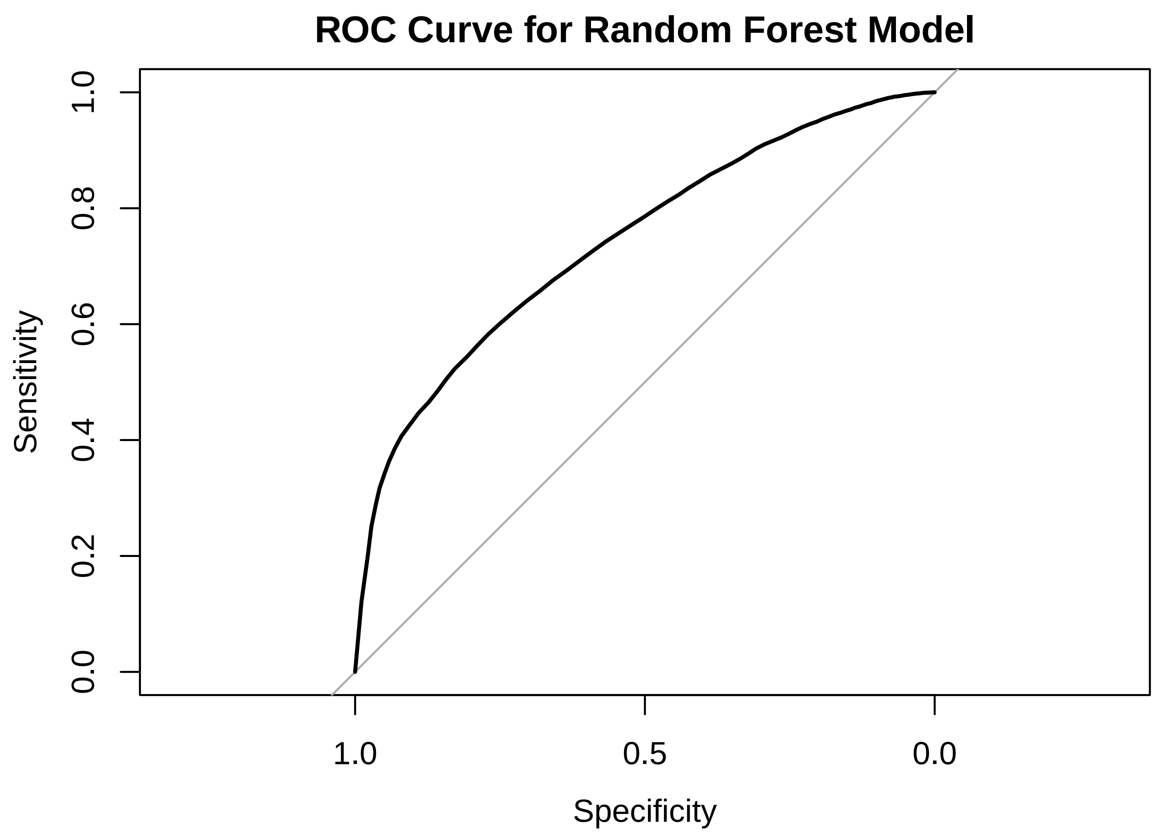
```
# Calculate ROC and AUC
rf_roc <- roc(test$NoShow, rf_probs$Yes, levels = rev(levels(test$NoShow)))
```

```
## Setting direction: controls > cases
```

```
auc_rf <- auc(rf_roc)
print(paste("Random Forest AUC:", round(auc_rf, 3)))
```

```
## [1] "Random Forest AUC: 0.742"
```

```
# Plot ROC curve
plot(rf_roc, main = "ROC Curve for Random Forest Model")
```



```r
rf_var_importance <- varImp(rf_model, scale = FALSE)
print(rf_var_importance)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 113)
##
##                            Overall
## Lag.days                    3723.7
## Lag.hours                   3681.8
## ScheduledDate               3455.1
## Age                         2420.5
## AppointmentDate              816.4
## GenderM                      432.5
## ScheduledDay^5               282.7
## ScheduledDay^6               269.0
## AppointmentDay^5             266.0
## AppointmentDay^6             252.4
## SocialWelfare                245.8
## ScheduledDay.Q               230.8
## ScheduledDay^4               224.0
## AppointmentDay.Q             223.9
## ScheduledDay.L               218.6
## AppointmentDay^4             215.3
## Hypertension                 214.4
## AppointmentDay.L             212.4
## NeighbourhoodJARDIM CAMBURI  200.4
## NeighbourhoodMARIA ORTIZ     199.9
```

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

According to the above indicators, in terms of feature importance, it is very close to our previous ideas and more reasonable, but the model still has many problems: 1.Class Imbalance: A high prevalence of positive classes (No) indicates severe class imbalance, reflected in low specificity and moderate balanced accuracy. 2.Model performance: The model's accuracy is not significantly better than the uninformative rate, and the Kappa coefficient indicates poor agreement in model predictions. 3.High sensitivity, low specificity: Although the model has high sensitivity, low specificity means it has difficulty correctly identifying the negative class (Yes). I think that the model will be more trustworthy after more parameter tuning, oversampling and other methods are performed.

Credits

This notebook was based on a combination of other notebooks e.g., 1, 2, 3