Medical Appointment No-Show Analysis

by Andrew Dettor

Link to Source Code:

Table of Contents

Introduction

What do doctors do when you abruptly cancel an appointment? Do they sit there twiddling their thumbs, waiting for the next person, and killing time while they could be treating patients? It would be helpful if doctors could predict which patients would show up or not; to use the empty time slot more effectively, or to bump up a patient who needs help quickly but can't fit into a packed schedule.

This data set comes from Joni Hoppen at Aquarela Advanced Analytics in Brazil, which makes sense because all the neighborhood names are not English.

I downloaded this dataset from Kaggle.com at the URL https://www.kaggle.com/joniarroba/noshowappointments.

The language used for this project is R. The source code link takes you to the knitted RMarkdown document with all the code and all my comments.

Exploratory Data Analysis

Dataset Size

The dataset has 110527 observations and 14 features (including the response variable, No.show).

Missing Values

Luckily, there are 0 missing values across the dataset, so no imputation is required.

Features and Data Types

The majority of the features are categorical, with exception of Age, ScheduledDay, and AppointmentDay.

```
 $ PatientId     : num  2.99e+13 5.59e+14 4.26e+12 8.68e+11 8.84e+12 ...
 $ AppointmentID : int  5642903 5642503 5642549 5642828 5642494 5626772 5630279 5630575 5638447 5629123 ...
 $ Gender        : chr  "F" "M" "F" "F" ...
 $ ScheduledDay  : chr  "2016-04-29T18:38:08Z" "2016-04-29T16:08:27Z" "2016-04-29T16:19:04Z" "2016-04-
29T17:29:31Z" ...
 $ AppointmentDay: chr  "2016-04-29T00:00:00Z" "2016-04-29T00:00:00Z" "2016-04-29T00:00:00Z" "2016-04-
29T00:00:00Z" ...
 $ Age           : int  62 56 62 8 56 76 23 39 21 19 ...
 $ Neighbourhood : chr  "JARDIM DA PENHA" "JARDIM DA PENHA" "MATA DA PRAIA" "PONTAL DE CAMBURI" ...
 $ Scholarship   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Hipertension  : int  1 0 0 0 1 1 0 0 0 0 ...
 $ Diabetes      : int  0 0 0 0 1 0 0 0 0 0 ...
 $ Alcoholism    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Handcap       : int  0 0 0 0 0 0 0 0 0 0 ...
 $ SMS_received  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ No.show       : chr  "No" "No" "No" "No" ...
```

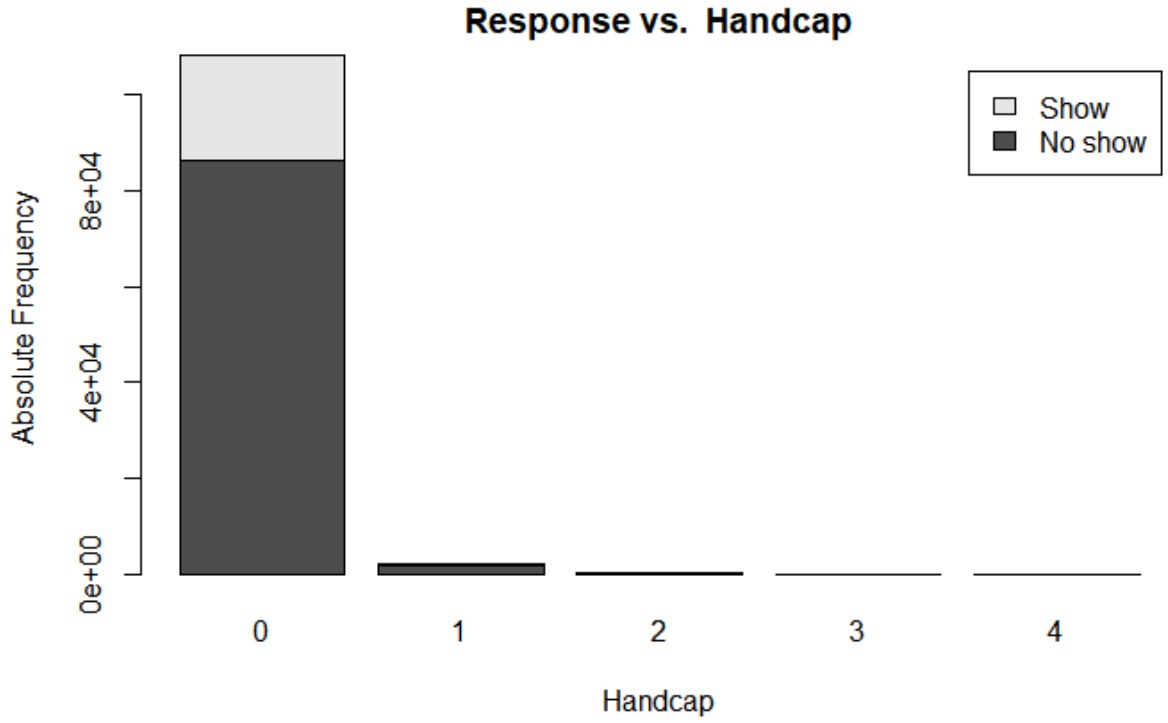
Unique Values

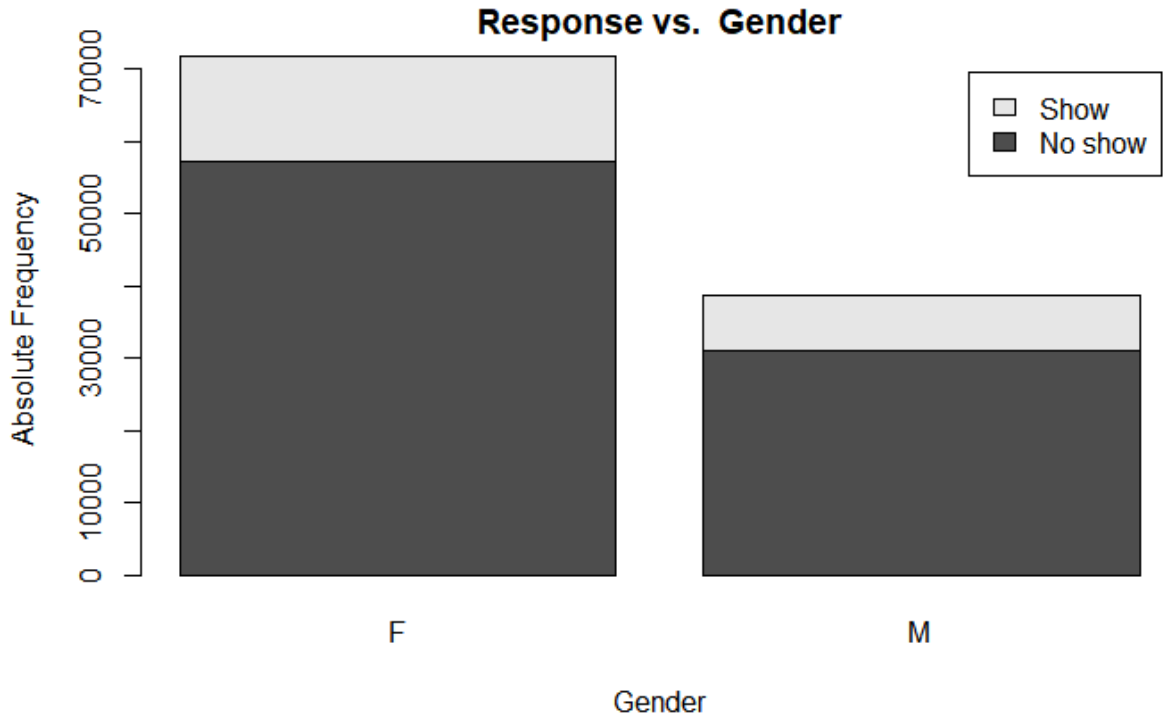
```
AppointmentID   ScheduledDay      PatientId            Age  Neighbourhood AppointmentDay        Handcap
       110527         103549          62299            104             81             27              5
       Gender    Scholarship   Hipertension       Diabetes     Alcoholism   SMS_received        No.show
            2              2              2              2              2              2              2
```

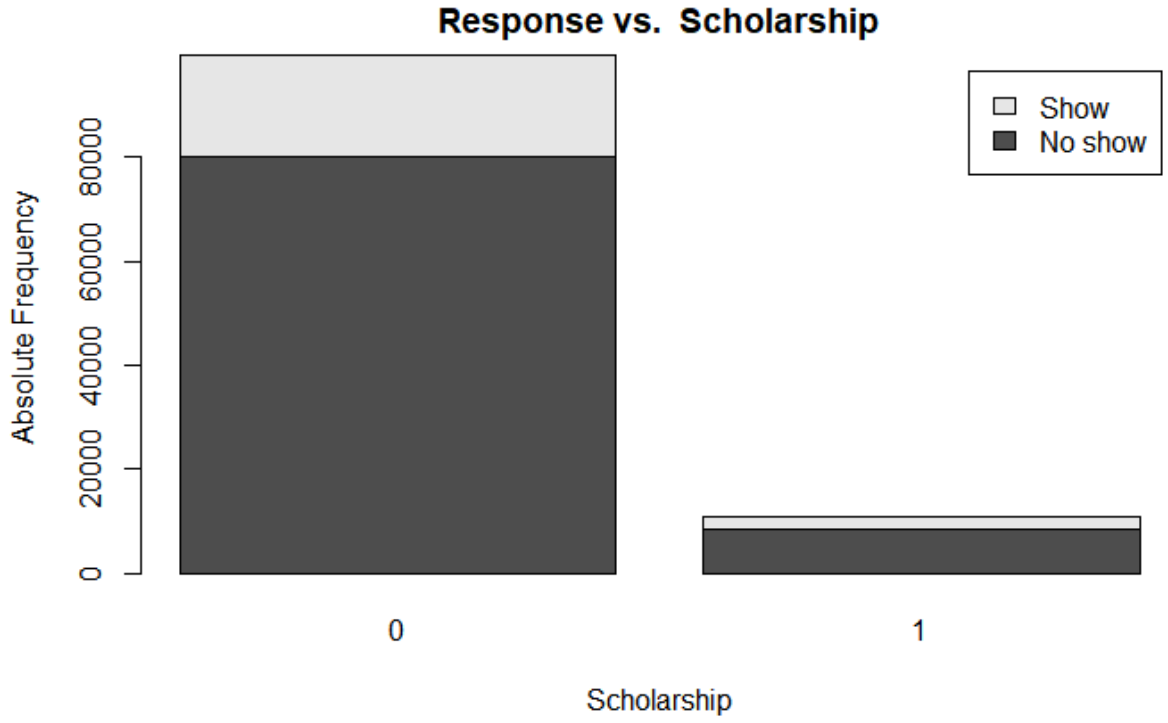
ScheduledDay has 103549 unique values, while AppointmentDay has only 27 unique values. There are 81 unique neighborhoods represented in this dataset. I can't do EDA on ScheduledDay because there are too many unique values. I'll do EDA if ScheduledDay is very important in the final model.

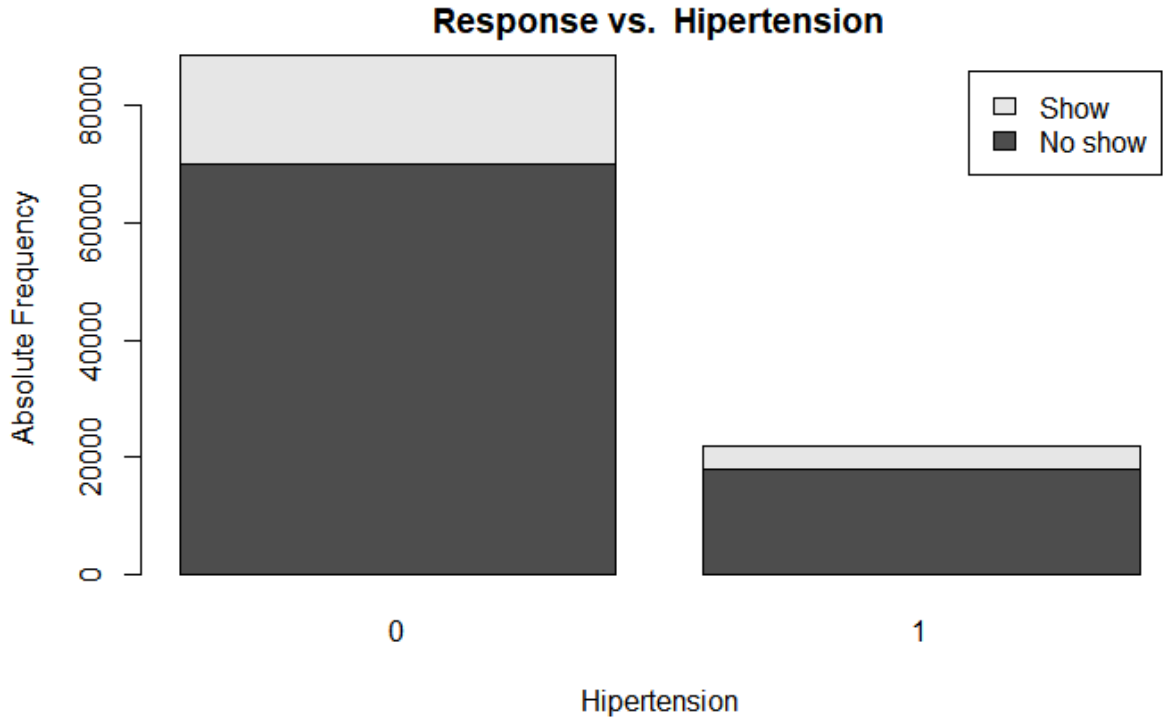
Graphical EDA

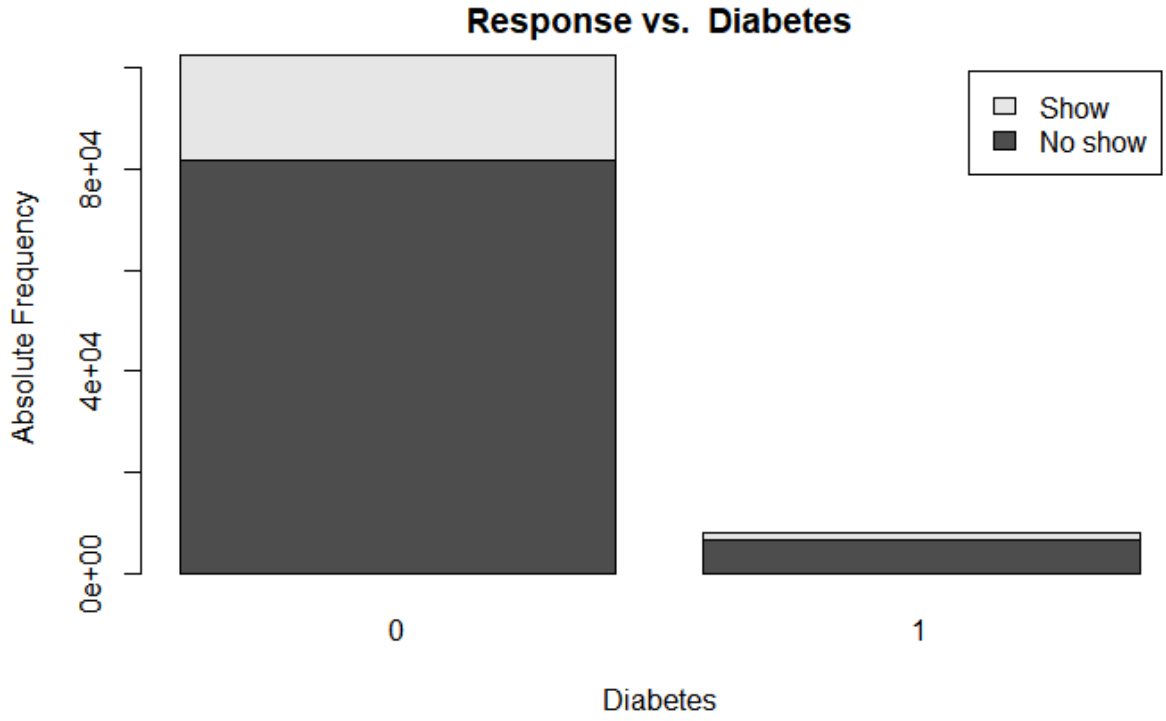
Stacked Barplots of Response vs. Categorical Variables

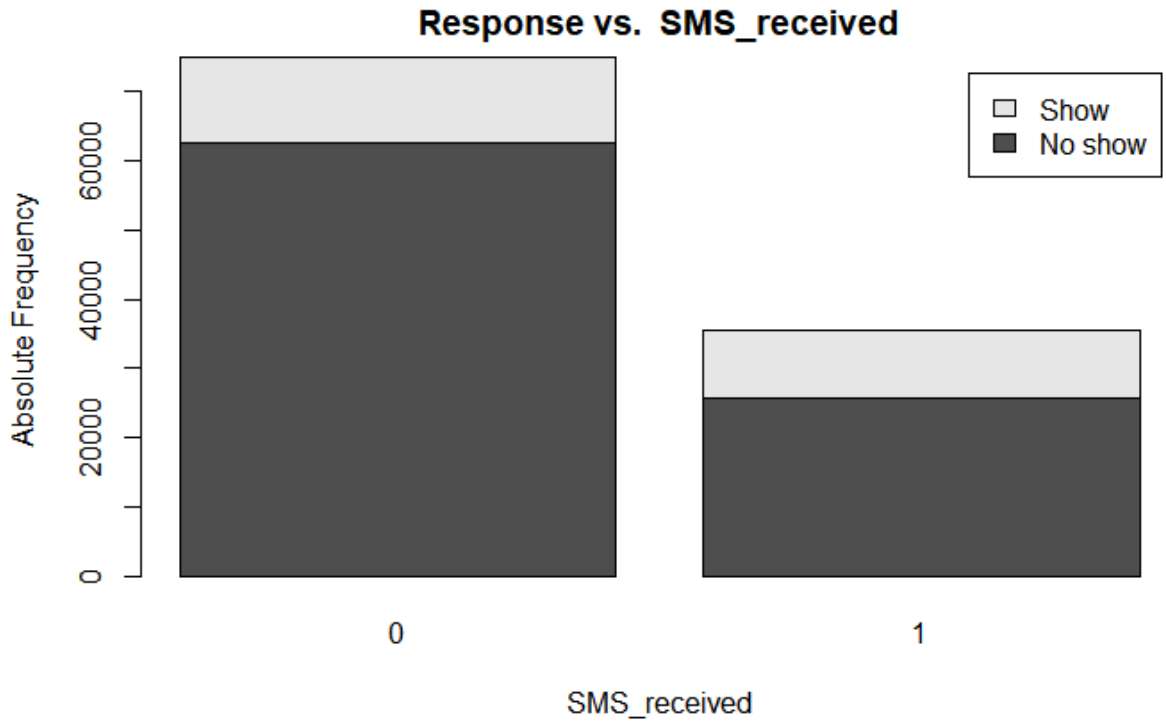


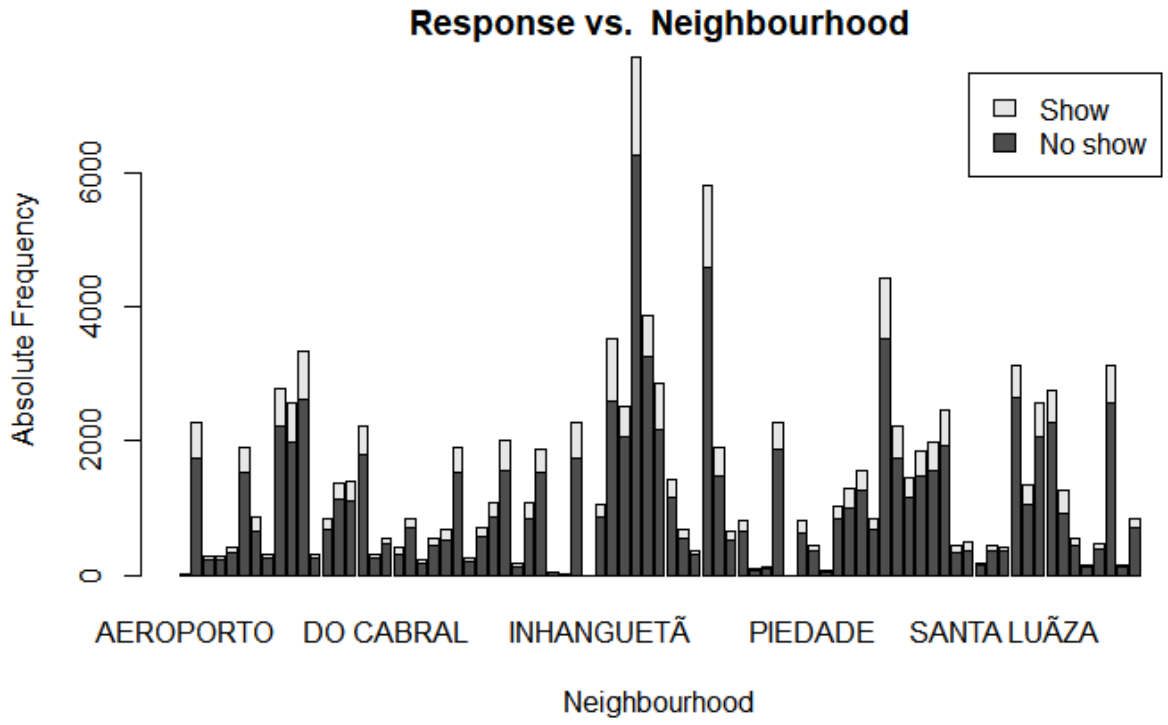


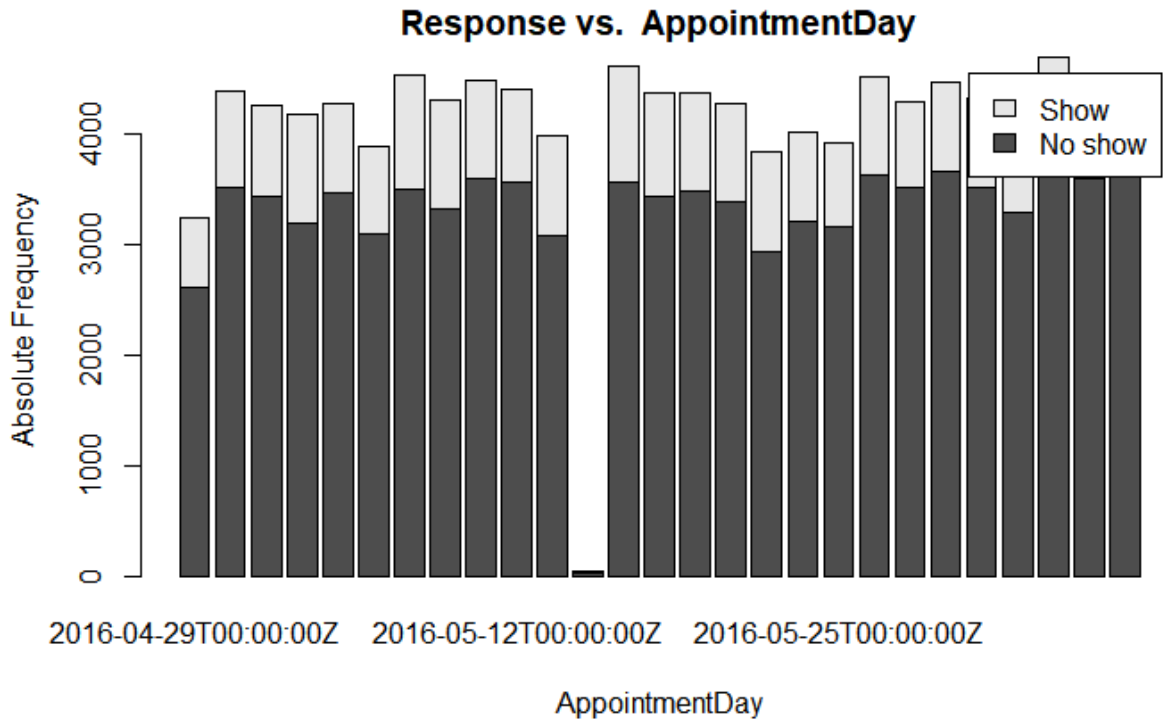

Response vs. Scholarship
Show
No show
Absolute Frequency
0
20000
40000
60000
80000
0
1
Scholarship


Response vs. Hipertension
Show
No show
Absolute Frequency
0
20000
40000
60000
80000
0
1
Hipertension


Response vs. Diabetes
Show
No show
Absolute Frequency
0e+00
4e+04
8e+04
0
1
Diabetes


Response vs. SMS_received
Show
No show
Absolute Frequency
0
20000
40000
60000
0
1
SMS_received

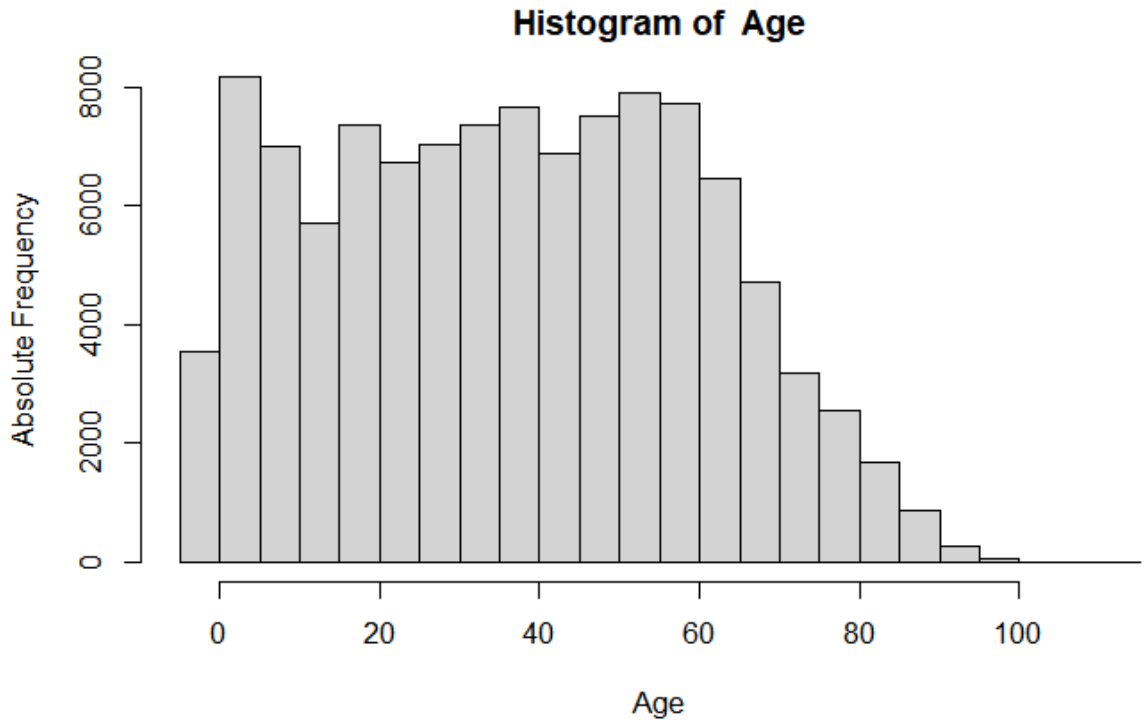




Show vs. No Show is pretty much a tie for each value of each feature. However, among people who received a text message reminder, more people showed up. Also, among the small population of documented alcoholics, people who were alcoholics were less likely to show up. People who had a financial scholarship were more likely to show up, too.

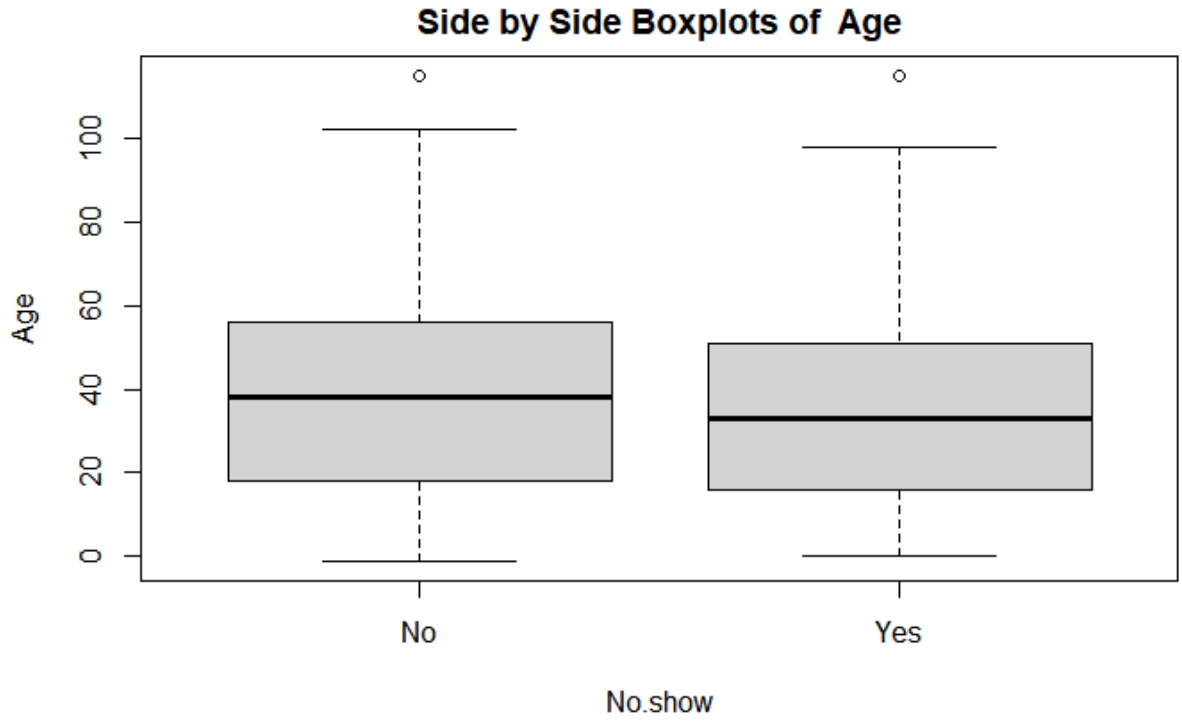
The 27 unique appointment days in the dataset are about evenly spread out, except for one. Some neighborhoods are represented much more than others.

Histograms of Numerical Variable(s)



Ages 5-60 are equally common, and Ages after 60 are much less common.

Side by Side Boxplots of Response vs. Numerical Variables



Seems like younger people were slightly more likely to show up.

Numerical EDA

Pivot Tables

```
Handcap show_rate
      4       0.33
      3       0.23
      0       0.20
      2       0.20
      1       0.18

Gender show_rate
"F"    "0.2"
"M"    "0.2"

Scholarship show_rate
          1       0.24
          0       0.20

Hipertension show_rate
           0       0.21
           1       0.17

Diabetes show_rate
       0       0.20
       1       0.18
```

Neighbourhood	show_rate
"ILHAS OCEÃ,NICAS DE TRINDADE"	"1"
"SANTOS DUMONT"	"0.29"
"SANTA CECÃ\u008dLIA"	"0.27"
"ITARARÃ‰"	"0.26"
"SANTA CLARA"	"0.26"
"JESUS DE NAZARETH"	"0.24"
"HORTO"	"0.24"
"ANDORINHAS"	"0.23"
"PRAIA DO SUÃ\u0081"	"0.23"
"CARATOÃ\u008dRA"	"0.23"
"GURIGICA"	"0.23"
"ILHA DO PRÃ\u008dNCIPE"	"0.23"
"BENTO FERREIRA"	"0.22"
"MARUÃ\u008dPE"	"0.22"
"ILHA DAS CAIEIRAS"	"0.22"
"SÃƒO JOSÃ‰"	"0.22"
"PARQUE MOSCOSO"	"0.22"
"DO MOSCOSO"	"0.22"
"ARIOVALDO FAVALESSA"	"0.22"
"BARRO VERMELHO"	"0.22"
"FONTE GRANDE"	"0.22"
"ENSEADA DO SUÃ\u0081"	"0.22"
"CRUZAMENTO"	"0.22"
"SÃƒO PEDRO"	"0.21"
"MARIA ORTIZ"	"0.21"
"SANTA HELENA"	"0.21"
"UNIVERSITÃ\u0081RIO"	"0.21"
"CENTRO"	"0.21"
"ROMÃƒO"	"0.21"
"NAZARETH"	"0.21"
"GOIABEIRAS"	"0.2"
"SÃƒO CRISTÃ“VÃƒO"	"0.2"
"GRANDE VITÃ“RIA"	"0.2"
"SÃƒO BENEDITO"	"0.2"
"SANTO ANDRÃ‰"	"0.2"
"BONFIM"	"0.2"
"RESISTÃŠNCIA"	"0.2"
"MONTE BELO"	"0.2"
"BELA VISTA"	"0.2"
"ILHA DO FRADE"	"0.2"
"SANTA LÚCIA"	"0.2"
"ESTRELINHA"	"0.2"
"SANTOS REIS"	"0.2"
"SANTA TEREZA"	"0.2"
"CONQUISTA"	"0.19"
"DA PENHA"	"0.19"
"JARDIM CAMBURI"	"0.19"
"ILHA DE SANTA MARIA"	"0.19"
"JUCUTUQUARA"	"0.19"
"INHANGUETÃ\u0081"	"0.19"
"BOA VISTA"	"0.19"
"PIEDADE"	"0.19"
"SEGURANÃ‡A DO LAR"	"0.19"
"FRADINHOS"	"0.19"
"NOVA PALESTINA"	"0.18"
"TABUAZEIRO"	"0.18"
"JABOUR"	"0.18"
"ANTÃ”NIO HONÃ“RIO"	"0.18"
"SANTO ANTÃ”NIO"	"0.18"
"REDENÃ‡ÃƒO"	"0.18"
"JOANA DÂ´ARC"	"0.18"
"PRAIA DO CANTO"	"0.18"
"SANTA LUÃ\u008dZA"	"0.18"
"FORTE SÃƒO JOÃƒO"	"0.18"
"COMDUSA"	"0.18"
"MATA DA PRAIA"	"0.17"
"PONTAL DE CAMBURI"	"0.17"
"REPÚBLICA"	"0.17"
"CONSOLAÇÃƒO"	"0.17"
"MORADA DE CAMBURI"	"0.17"
"VILA RUBIM"	"0.17"
"JARDIM DA PENHA"	"0.16"
"SANTA MARTHA"	"0.16"
"DO QUADRO"	"0.16"
"DO CABRAL"	"0.16"
"SOLON BORGES"	"0.15"
"MÃ\u0081RIO CYPRESTE"	"0.15"
"DE LOURDES"	"0.15"
"AEROPORTO"	"0.12"
"ILHA DO BOI"	"0.09"
"PARQUE INDUSTRIAL"	"0"

AppointmentDay	show_rate
"2016-05-10T00:00:00Z"	"0.23"
"2016-05-16T00:00:00Z"	"0.23"
"2016-05-04T00:00:00Z"	"0.23"
"2016-05-20T00:00:00Z"	"0.23"
"2016-05-13T00:00:00Z"	"0.23"
"2016-05-09T00:00:00Z"	"0.23"
"2016-05-14T00:00:00Z"	"0.23"
"2016-05-17T00:00:00Z"	"0.21"
"2016-05-19T00:00:00Z"	"0.21"
"2016-04-29T00:00:00Z"	"0.2"
"2016-05-03T00:00:00Z"	"0.2"
"2016-05-24T00:00:00Z"	"0.2"
"2016-05-02T00:00:00Z"	"0.2"
"2016-05-30T00:00:00Z"	"0.2"
"2016-05-06T00:00:00Z"	"0.2"
"2016-05-11T00:00:00Z"	"0.2"
"2016-05-18T00:00:00Z"	"0.2"
"2016-06-03T00:00:00Z"	"0.2"
"2016-05-12T00:00:00Z"	"0.19"
"2016-05-05T00:00:00Z"	"0.19"
"2016-05-25T00:00:00Z"	"0.19"
"2016-06-02T00:00:00Z"	"0.19"
"2016-06-06T00:00:00Z"	"0.19"
"2016-05-31T00:00:00Z"	"0.18"
"2016-06-07T00:00:00Z"	"0.18"
"2016-06-01T00:00:00Z"	"0.18"
"2016-06-08T00:00:00Z"	"0.17"

Neighborhoods and appointment days have about even distributions of show rate. However there are outliers; one neighborhood has a show rate of 0% and another has a show rate of 100%.

As seen before, people who received a reminder text were more likely to show up.

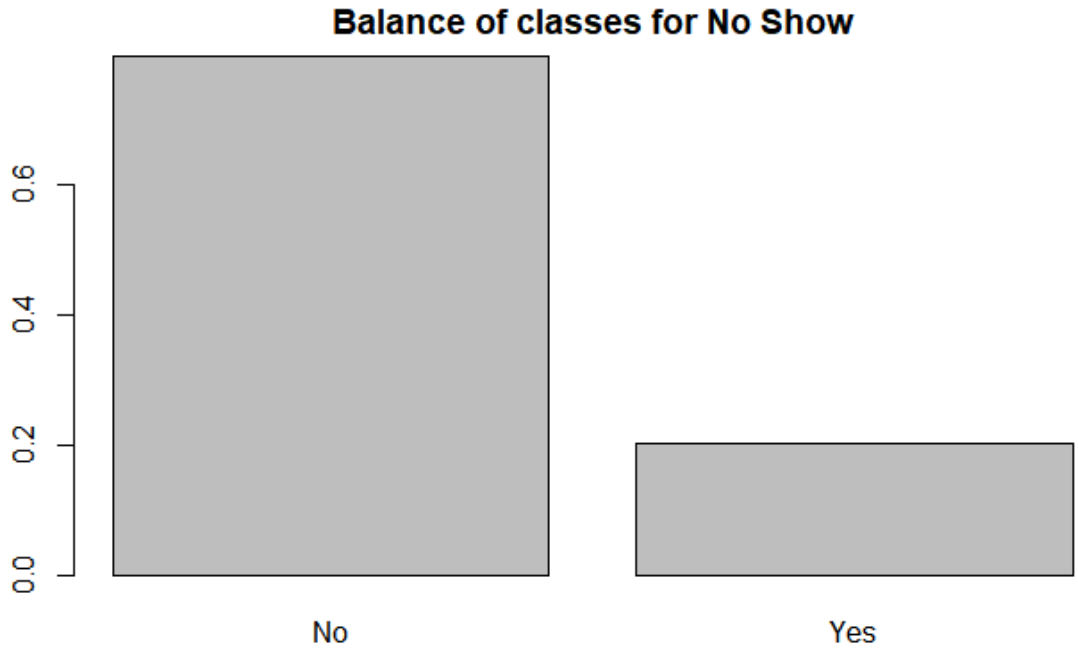
Handicap group 1 was about half as likely to show up as group 4.

People with financial scholarship and those with hypertension were slightly more likely to show up.

```
No.show Average Age
"No"    "37.79"
"Yes"   "34.32"
```

Numerical proof of what I found in the Age vs No.show box plot.

Response Variable EDA



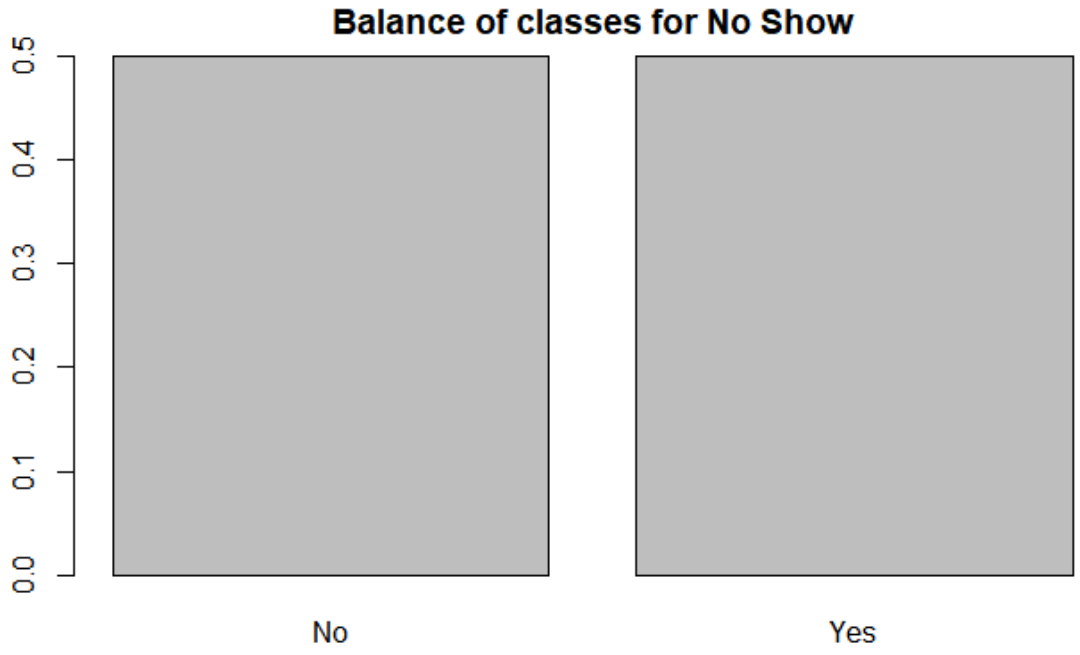

```
       No       Yes
0.7980674 0.2019326
```

Imbalanced classes. The base chance for correctly predicting No is 80%.
Going to try to make the class sizes the same.

Going to randomly downsample (without replacement) the overrepresented class and randomly upsample (with replacement) the underrepresented class so they are represented the same amount.

Across the dataset, there are 22319 no shows and 88208 shows, so I'm going to balance it to 55264 each.

After doing all this...




```
 No Yes
0.5 0.5
```

Data Cleaning / Feature Engineering

Datetime Format Features

Using the lubridate R library to extract features from ScheduledDay and AppointmentDay.

```r
library(lubridate)
scheduledday = ymd_hms(data$ScheduledDay)
appointmentday = ymd_hms(data$AppointmentDay)
```

```r
x = scheduledday
data[,"scheduledWeekDay"] = wday(x)
data[,"scheduledMonth"]= month(x)
data[,"scheduledDayofMonth"] = day(x)
data[,"scheduledHourofDay"] = hour(x)
data[,"scheduledDayofYear"] = yday(x)
data[,"scheduledAM"] = am(x)
data[,"scheduledWeekofYear"] = week(x)
data[,"scheduledQuarter"] = quarter(x)
```

```r
x = appointmentday
data[,"appointmentWeekDay"] = wday(x)
data[,"appointmentMonth"]= month(x)
data[,"appointmentDayofMonth"] = day(x)
data[,"appointmentHourofDay"] = hour(x)
data[,"appointmentDayofYear"] = yday(x)
data[,"appointmentAM"] = am(x)
data[,"appointmentWeekofYear"] = week(x)
data[,"appointmentQuarter"] = quarter(x)
```

I would do EDA on all these variables but it would be quite verbose. Most of them are unimportant anyways. I'll do feature selection later and then do EDA on which ones are most important.

I ended up dropping appointmentHourofDay, appointmentAM, and appointmentQuarter because they all had 0 variance, and thus wouldn't be useful to a learning algorithm.

Train/Test Split

Using a randomly sampled 80%/20% train/test split.
There are 88621 observations in the training set and 21907 observations in the test set.

Neighborhood Target Encoding

Replacing neighborhood with neighborhood_show_rate.
Find the average show rate for each neighborhood in the training set, then set the test set values of neighborhood_show_rate to the averages.

For neighborhoods not in the training set, set the test set values of neighborhood_show_rate to the average show_rate of all neighborhoods.

Only one of the 81 neighborhoods are not in the training set, and upon further inspection, this neighborhood occurs only once. I don't need to do a separate Target Encoding for each individual fold during cross validation because the average show rate for all the other neighborhoods won't really change depending on the fold.

Pictured here are the top 3 and bottom 3 neighborhoods in terms of show rate.

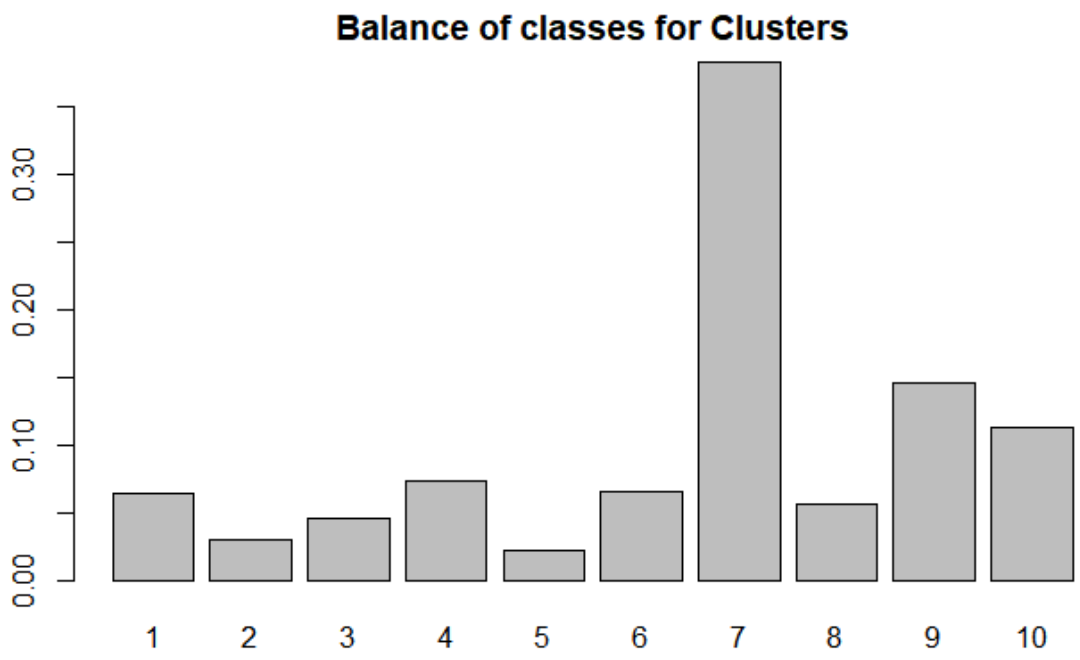
```
Neighborhood      show_rate
"AEROPORTO"       "0"
"ILHA DO BOI"     "0.1765"
"MÃ\u0081RIO CYPRESTE" "0.2851"

 Neighborhood        show_rate
 "PARQUE INDUSTRIAL" "0.500682682434186"
 "SANTA CECÃ\u008dLIA" "0.5059"
 "ILHA DO FRADE"     "0.6667"
```

Neighborhoods with a higher show rate (e.g. Ilha Do Frade) are more likely to show up to appointments while other neighborhoods (e.g. Aeroporto) are much less likely to show up.

Some neighborhoods are very underrepresented, so their show rates might be extreme.

Clustering as a New Feature

Trying to use unsupervised clustering (KMeans with 10 clusters) as a feature for the supervised learning algorithms.
Can use both the train and test sets in the clustering because it's unsupervised.



We'll see later on if this is useful.

Model Evaluation and Selection

5-Fold Cross Validation

Going to try many algorithms with base hyperparameters first, then optimize hyperparameters on the best model.

Using 5 fold cross validation on the training set to get a good idea of model performance.

I'm using the confusion matrices of each fold's performance to calculate the average performance of each algorithm. I'm keeping track of accuracy, precision, recall, and F1 score.

Naive Bayes

Confusion Matrices

```
      Predicted
Actual    0    1
     0 6674 2251
     1 4904 4010
      Predicted
Actual    0    1
     0 6638 2258
     1 4806 4092
      Predicted
Actual    0    1
     0 6497 2221
     1 4882 3913
      Predicted
Actual    0    1
     0 6587 2260
     1 4708 4075
      Predicted
Actual    0    1
     0 6700 2164
     1 5120 3861
```

Averaged Performance

Metrics	Values
"Accuracy"	"59.858"
"Recall"	"44.972"
"Precision"	"64.138"
"F1"	"52.862"

Logistic Regression

Confusion Matrices

```
      Predicted
Actual    0    1
     0 7621 1304
     1 6190 2724
      Predicted
Actual    0    1
     0 7591 1305
     1 6153 2745
      Predicted
Actual    0    1
     0 7510 1208
     1 6102 2693
      Predicted
Actual    0    1
     0 7577 1270
     1 6000 2783
      Predicted
Actual    0    1
     0 7601 1263
     1 6332 2649
```

Averaged Performance

Metrics	Values
"Accuracy"	"58.108"
"Recall"	"30.644"
"Precision"	"68.164"
"F1"	"42.274"

Decision Tree

Confusion Matrices

```
      Predicted
Actual    0    1
     0 5662 3263
     1 3436 5478
      Predicted
Actual    0    1
     0 5645 3251
     1 3412 5486
      Predicted
Actual    0    1
     0 5551 3167
     1 3457 5338
      Predicted
Actual    0    1
     0 5503 3344
     1 3414 5369
      Predicted
Actual    0    1
     0 5627 3237
     1 3596 5385
```

Averaged Performance

Metrics	Values
"Accuracy"	"62.112"
"Recall"	"60.976"
"Precision"	"62.46"
"F1"	"61.708"

Random Forest

Confusion Matrices

```
      Predicted
Actual    0    1
     0 6339 2586
     1 1237 7677
      Predicted
Actual    0    1
     0 6381 2515
     1 1209 7689
      Predicted
Actual    0    1
     0 6150 2568
     1 1287 7508
      Predicted
Actual    0    1
     0 6287 2560
     1 1164 7619
      Predicted
Actual    0    1
     0 6358 2506
     1 1273 7708
```

Averaged Performance

Metrics	Values
"Accuracy"	"78.666"
"Recall"	"86.096"
"Precision"	"74.996"
"F1"	"80.16"

Gradient Boosted Random Forest

Confusion Matrices

```
Distribution not specified, assuming bernoulli ...
      Predicted
Actual    0    1
     0 7535 1390
     1 5836 3078
Distribution not specified, assuming bernoulli ...
      Predicted
Actual    0    1
     0 7474 1422
     1 5828 3070
Distribution not specified, assuming bernoulli ...
      Predicted
Actual    0    1
     0 7347 1371
     1 5715 3080
Distribution not specified, assuming bernoulli ...
      Predicted
Actual    0    1
     0 7424 1423
     1 5653 3130
Distribution not specified, assuming bernoulli ...
      Predicted
Actual    0    1
     0 7447 1417
     1 5909 3072
```

Averaged Performance

Metrics	Values
"Accuracy"	"59.42"
"Recall"	"34.78"
"Precision"	"68.722"
"F1"	"46.182"

K Nearest Neighbors

Confusion Matrices

```
      Predicted
Actual    0    1
     0 5447 3478
     1 2194 6720
      Predicted
Actual    0    1
     0 5402 3494
     1 2067 6831
      Predicted
Actual    0    1
     0 5323 3395
     1 2182 6613
      Predicted
Actual    0    1
     0 5224 3623
     1 2120 6663
      Predicted
Actual    0    1
     0 5451 3413
     1 2245 6736
```

Averaged Performance

Metrics	Values
"Accuracy"	"68.164"
"Recall"	"75.642"
"Precision"	"65.858"
"F1"	"70.406"

Overall Performance

5 Fold CV Model Performance

	Naive Bayes	Logistic Regression	Decision Tree	Random Forest	Gradient Boosted Random Forest	K Nearest Neighbors
Accuracy	59.858	58.108	62.112	78.666	59.42	68.164
Recall	44.972	30.644	60.976	86.096	34.78	75.642
Precision	64.138	68.164	62.46	74.996	68.722	65.858
F1	52.862	42.274	61.708	80.16	46.182	70.406

Random Forests were the best performing model, with the highest Precision (75%) and the highest Accuracy (75%). Going to use them moving forward.

Precision is important here, because when we predict someone will not show up and they actually do show up, their appointment spot will likely be taken already.

Hyperparameter Optimization

The goal here was to use 5-fold-cross-validated grid-search on Random Forest to find the best mtry value (number of features to use in each tree split) and the best number of trees to use in the forest.

```
mtry_grid = c(floor(sqrt(ncol(data))),
              floor(log(ncol(data))),
              floor(sqrt(ncol(data))*2)) # default sqrt(ncol(data))
ntree_grid = c(200, 400, 500, 600) # default 500
```

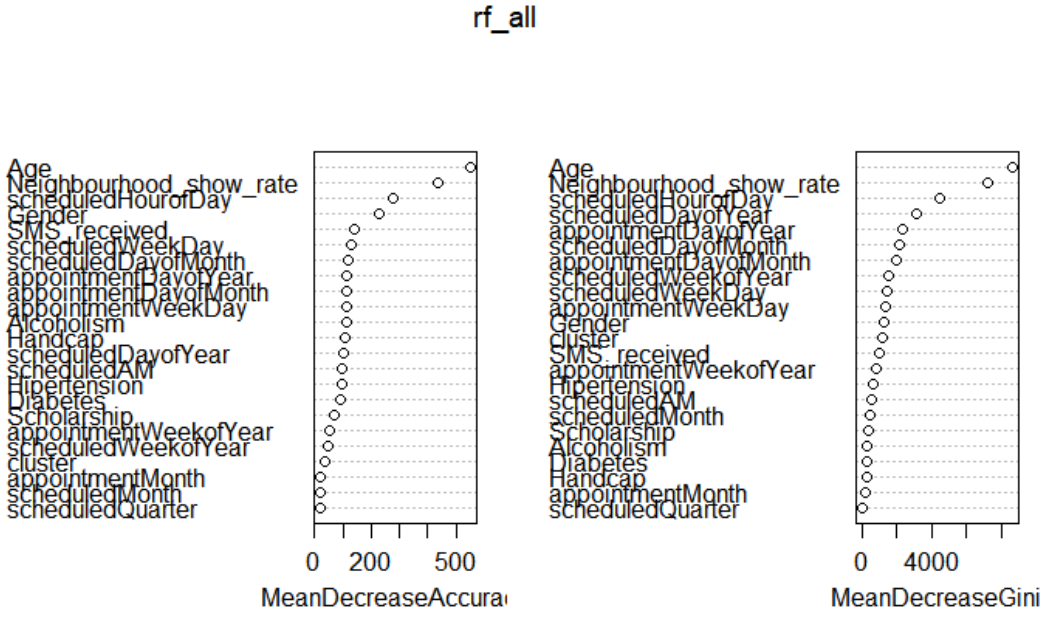
For each fold, 9 was the best number of variables to try at each split (which is floor(sqrt(ncol(data))*2).
The best number of trees to use was 500, which happens to be the default.

Feature Importance and Selection

Test Performance With All Features and Entire Training Set

```
       Predicted
Actual     0     1
     0  8584  2430
     1   824 10069
     Metrics     Values
[1,] "Accuracy"  "85.15"
[2,] "Recall"    "92.44"
[3,] "Precision" "80.56"
[4,] "F1"        "86.09"
```

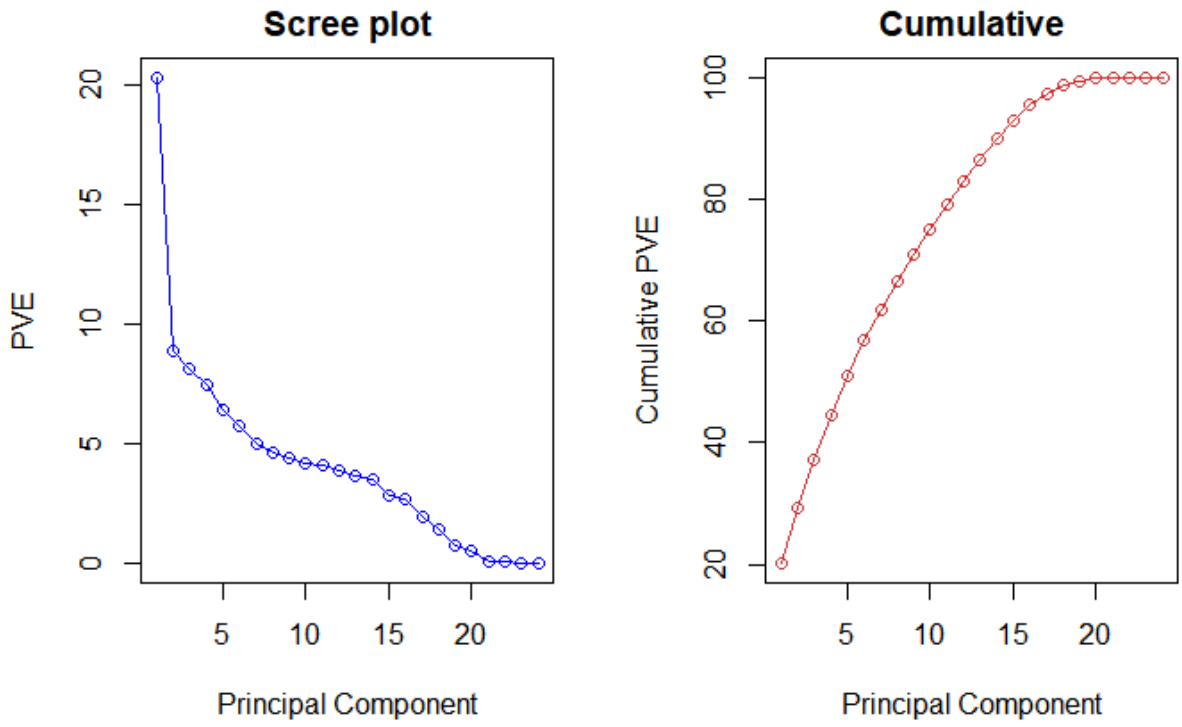
Random Forest Feature Importance



According to random forest's feature importance algorithm based on accuracy, the top 5 features are age, show rate of the person's neighborhood, gender, hour of the appointment, and day of the week of the appointment.

Surprisingly, the cluster feature I created using unsupervised learning was not very important for either feature importance metric.

Analysis of PCA Loadings



It takes 12 principal components to explain 80% of the variation in the data. This means the true rank of this dataset is quite high (i.e. it takes many features to explain the phenomenon well). The first two principal components explain about 29% of the variation in the data.

The first principal component explains about 20% of the variation in the data. The cumulative proportion of variation explained goes up logarithmically with the number of principal components.

Principal Component 1

```
   appointmentMonth appointmentWeekofYear  appointmentDayofYear        scheduledMonth
          0.3258295             0.3490378             0.3544376             0.4045466
scheduledWeekofYear    scheduledDayofYear
          0.4132359             0.4142458

             Gender                Handcap Neighbourhood_show_rate
        0.002742409            0.010935412            0.019554163
         Alcoholism            scheduledAM       scheduledWeekDay
        0.020418191            0.021088678            0.022342839
```

The features that have the highest loadings in the first (i.e. most impactful) principal component all have to do with the timing of scheduling and timing of the appointment itself. It seems that the month/week/day of both are important.

The features with the lowest loadings in this PC are more about the patient, like gender, alcoholism, and being handicapped. Also having the appointment in the morning or night does not matter that much, and neither does the day of the week.

Principal Component 2

```
appointmentWeekofYear  appointmentDayofYear             cluster                 Age
            0.1562930             0.1585991           0.3804287           0.4279203
        Hipertension              Diabetes
            0.5052436             0.5105040

        scheduledMonth   appointmentDayofMonth       scheduledWeekDay
             0.0004055065           0.0021023746          0.0081146611
Neighbourhood_show_rate       scheduledQuarter     scheduledWeekofYear
             0.0122822413           0.0163233388          0.0231890050
```

The second principal component is a little more interesting; it focuses more on the patient themself. Their conditions and age matter, as well as the cluster they were put in when I ran KMeans on the dataset.

The least impactful features in this PC are features that were most impactful in the first PC.

Hypothesis Tests For Response Variable Relationship

Categorical/Categorical - Chi Square Test
Quantitative/Categorical - Logistic Regression t-test

Which features are quantitative and which are categorical? I'm going to say if the cardinality of the feature is above 15, it's quantitative.

Here are the number of unique values for each feature.

```
 scheduledDayofYear                 Age Neighbourhood_show_rate     scheduledDayofMonth    appointmentDayofYear
                107                 104                      78                      31                      27
scheduledWeekofYear  appointmentDayofMonth      scheduledHourofDay                 cluster         scheduledMonth
                 26                    24                      16                      10                       8
   scheduledWeekDay      appointmentWeekDay  appointmentWeekofYear                 Handcap       scheduledQuarter
                  6                       6                       6                       5                       3
   appointmentMonth                  Gender             Scholarship           Hipertension                Diabetes
                  3                       2                       2                       2                       2
         Alcoholism           SMS_received                No.show           scheduledAM
                  2                       2                       2                       2
```

All p-values after running each feature's respective hypothesis test.

```
              Gender                 Age         Scholarship
                0.00                0.00                0.00
        Hipertension            Diabetes       SMS_received
                0.00                0.00                0.00
     scheduledWeekDay       scheduledMonth  scheduledDayofMonth
                0.00                0.00                0.00
   scheduledHourofDay    scheduledDayofYear         scheduledAM
                0.00                0.00                0.00
  scheduledWeekofYear      scheduledQuarter   appointmentWeekDay
                0.00                0.00                0.00
     appointmentMonth  appointmentDayofYear appointmentWeekofYear
                0.00                0.00                0.00
Neighbourhood_show_rate             cluster appointmentDayofMonth
                0.00                0.00                0.06
               Handcap           Alcoholism
                0.11                0.62
```

These hypothesis tests are not really informative. They say that there is no association between No.show and 3 features (Handicap, Alcoholism, and appointmentDayofMonth). All other features have an association with No.show.

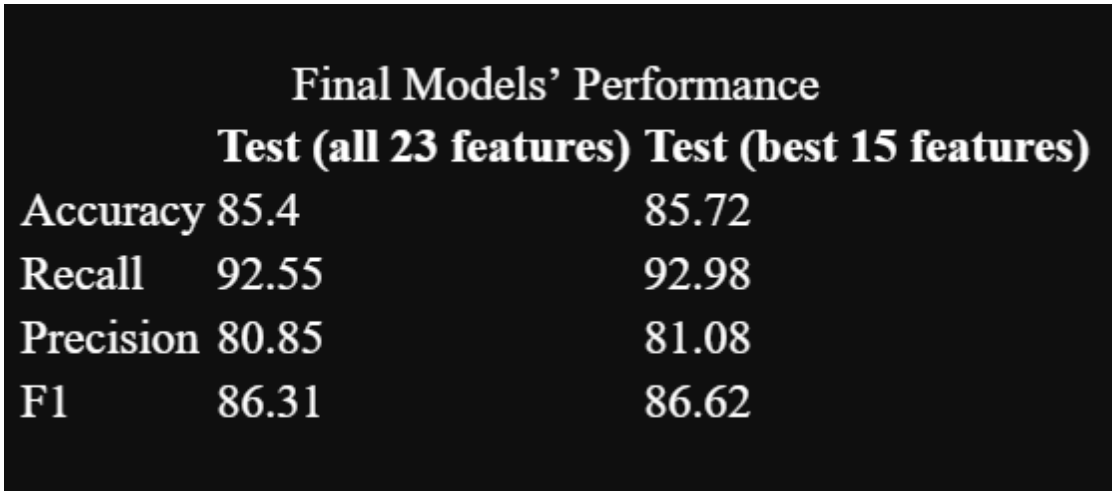
Test Performance With Most Important Features

Removing Alcoholism, Handicap, scheduledMonth, appointmentMonth, scheduledQuarter, Scholarship, scheduledAM, Diabetes due to what I've found. Now there are 15 features instead of 23.

Rerunning the random forest with the same hyperparameters (note that mtry is 7 this time instead of 9, because floor(sqrt(23))*2) = 9 while floor(sqrt(15))*2) = 7.

```
      Predicted
Actual     0     1
     0  8651  2363
     1   765 10128
     Metrics     Values
[1,] "Accuracy"  "85.72"
[2,] "Recall"    "92.98"
[3,] "Precision" "81.08"
[4,] "F1"        "86.62"
```

Final Results



Final Models' Performance		
	Test (all 23 features)	**Test (best 15 features)**
Accuracy	85.4	85.72
Recall	92.55	92.98
Precision	80.85	81.08
F1	86.31	86.62

The model with 1/3 fewer features had slightly better performance! Around .32% higher accuracy and .23% higher precision. This shows how removing unnecessary features can reduce variance, boosting a model's performance metrics.