DATA606 Project Report - Hotel Booking Demand and Cancellation Analysis

Group 4 - Harold Lee, Radhika Joshi, Alex Papparis, David St. George, Daniel Zhou

February 20, 2022

Introduction & Background

In 2019, tourism and travel brought in 42.8 billion USD to Portugal [1], contributing to 19.8% of GDP [2]. The rich culture, sunny beaches, and traditional food, all offered at affordable prices, explain the high amount of money coming into the country. Following the 2008 financial crisis, Portugal was hit especially hard. Considering that tourism is such an important part of the economy, being able to maximize efficiency of hospitality management would be beneficial.

Consistency is an important factor in the hospitality industry. The behavior of clients can be chaotic, leading to cancellations, shortened stays, a desire for more expensive accommodations, and many other factors. The rise of data science has enabled businesses to utilize historical client data to create profiles in attempts to maximize profits and minimize liabilities.

While guests make bookings in advance for hotels, often they cancel their reservation prior to their stay. Being able to predict a guest's behavior in advance of their stay is an asset for the management team. Creating a profile for customers based on historical client data has enabled businesses to create profiles to maximize profits while minimizing liabilities.

Dataset Description

The dataset used for this report is the 'Hotel Booking Demand Dataset' acquired from Kaggle [3], the dataset originated from ScienceDirect [4], authored by Nuno Antonio, Ana de Almeida, Luis Nunes. It consists of hotel demand data from two hotels in Portugal from July 2015 to August 2017, with one hotel located in an urban region (Lisbon) classed in the dataset as 'City Hotel', and another located in a resort region (Algarve) classed 'Resort Hotel'.

The dataset includes 119,390 data points, each data point representing a single hotel room booking. There are 32 variables total, containing both qualitative (binary and nominal) and quantitative variables (discrete and continuous). The details of each variable are listed in the table below in alphabetical order

Analysis and Results

Research Questions

There are 3 research questions analyzed in this report:

1. Using contingency tables, can it be determined if there is independence between categorical variables?
2. Which model is the best at predicting hotel booking cancellation in Portugal between Logistic Regression, Classification Tree, Random Forest Analysis, and Linear/Quadratic Discriminant Analysis.
3. Using regression tree, which variables are important in predicting average daily rate of a hotel stay in Portugal?

In this part of the report, the research questions will be explored, showing the full steps taken in R in order to conduct the analyses, along with presenting detailed results.

Data Import and Wrangling

Initially, the required packages were imported:

```
### import packages
library('sampling') # sampling package
library('survey') # sample mean and sample SD functions
library('MASS') # package for lda and qda functions
library('car') # package for for VIF
library('klaR') # package for partimat (partition plot)
library('ggplot2') #using ggplot2 for data visualization
library('dplyr') # used for comparisons and filters
library('tree') # tree functions
library('caret') # train / createFolds function for cross validation
library('tidyverse')
library('GGally')
library('randomForest') # for RandomForest classification algorithm

# packages specific to research question 1
library(questionr)
library(fmsb)

# packages specific to research question 3
library('olsrr') # for stepwise
library('lmtest') # for Breusch-Pagan homoscedasticity test
library('nortest') # for Anderson-Darling normality test
```

The next step is to import the dataset into R from the raw CSV file from Kaggle

```
### import dataset
df=read.csv("hotel_bookings.csv",header=TRUE)

# check initial dataset dimensions
dim(df)
```

```
## [1] 119390     32
```

```r
# check column / variables. names
names(df)
```

```
##  [1] "hotel"                          "is_canceled"
##  [3] "lead_time"                      "arrival_date_year"
##  [5] "arrival_date_month"             "arrival_date_week_number"
##  [7] "arrival_date_day_of_month"      "stays_in_weekend_nights"
##  [9] "stays_in_week_nights"           "adults"
## [11] "children"                       "babies"
## [13] "meal"                           "country"
## [15] "market_segment"                 "distribution_channel"
## [17] "is_repeated_guest"              "previous_cancellations"
## [19] "previous_bookings_not_canceled" "reserved_room_type"
## [21] "assigned_room_type"             "booking_changes"
## [23] "deposit_type"                   "agent"
## [25] "company"                        "days_in_waiting_list"
## [27] "customer_type"                  "adr"
## [29] "required_car_parking_spaces"    "total_of_special_requests"
## [31] "reservation_status"             "reservation_status_date"
```

Next, several steps are taken to properly wrangle the data, specifically:

1. replace 'NULL' in 'agent' column with 'None' (as NULL in this case means no agent was used)
2. replace 'NULL' in 'company' column with 'None' (as NULL in this case means no booking company was used)
3. replace 'Undefined' in 'meal' column with 'SC' (as they mean the same thing which is that no meal package was booked)
4. remove rows where 'children' column has 'NA' value (as they are unclean data)
5. remove rows where 'country' column has 'NULL' value (as they are unclean data)
6. remove rows where 'market_segment' column has 'Undefined' value (as they are unclean data)
7. remove rows where 'distribution_channel' column has 'Undefined' value (as they are unclean data)
8. factor our response variable "is_canceled" (as this makes the variable easier to work with later on in our analysis)
9. remove "stays_in_week_nights" and "stays_in_weekend_nights" = 0 (as they are unclean data)
10. removed data where "adult" and "children" = 0
11. added "consistent_room_type" column to the data to be used later in analysis
12. added "arrival_date_season" column to the data to be used later in analysis
13. duplicate the wrangled dataset for each group member, so that they can do their own problem specific wrangling without affecting other member's data

- There was an attempt to preserve as much data as possible in the wrangling
- Starting with 119,390 datapoints, 118,087 were left after cleaning/wrangling (1,303 datapoints removed in the process).

```r
### data cleaning
df1=df
# replace 'NULL' in 'agent' column with 'None'
df1$agent[df1$agent == 'NULL'] = 'None'

# replace 'NULL' in 'company' column with 'None'
df1$company[df1$company == 'NULL'] = 'None'
```

```r
# replace 'Undefined' in 'meal' column with 'SC'
# data source states they are the same thing:
# https://www.sciencedirect.com/science/article/pii/S2352340918315191#tbl1fna
df1$meal[df1$meal == 'Undefined'] = 'SC'

# remove rows where 'children' column has 'NA' value
df1 = df1[!is.na(df1$children),]

# remove rows where 'country' column has 'NULL' value
df1 = df1[df1$country != 'NULL',]

# remove rows where 'market_segment' column has 'Undefined' value
df1 = df1[df1$market_segment != 'Undefined',]

# remove rows where 'distribution_channel' column has 'Undefined' value
df1 = df1[df1$distribution_channel != 'Undefined',]

# remove rows where 'market_segment' column has 'Undefined' value
df1 = df1[df1$market_segment != 'Undefined',]

# remove rows where "stays_in_week_nights" and "stays_in_weekend_nights" = 0
df1 = df1[df1$stays_in_week_nights > 0 | df1$stays_in_weekend_nights > 0,]

# removed rows where "adult" and "children" = 0
df1 = df1[df1$adults > 0 | df1$children > 0,]

# check cleaned dataset dimensions
dim(df1)
```

```
## [1] 118087     32
```

```r
# factor our response variable "is_canceled"
df1$is_canceled = factor(df1$is_canceled)

# check how many classes there are (the variable "Diabetes")
unique(df1$is_canceled)
```

```
## [1] 0 1
## Levels: 0 1
```

```r
# check the class sizes.
table(df1$is_canceled)
```

```
##
##     0     1
## 73972 44115
```

```r
# Check how R dummifies the "is_canceled" variable
contrasts(df1$is_canceled)
```

```
##   1
## 0 0
## 1 1
```

```r
# add consistent_room_type column to the data
df1$consistent_room_type = ifelse(df1$assigned_room_type == df1$reserved_room_type,
                                  "Yes", "No")
# df1 = df1 %>%
#  dplyr::select(-c(assigned_room_type, reserved_room_type))

# add arrival_date_season column to the data
df1$arrival_date_season = ifelse(df1$arrival_date_month %in% c("December",
                                                               "January",
                                                               "February"),
                                 "Winter",
                                 ifelse(df1$arrival_date_month %in% c("March",
                                                                      "April"),
                                        "Spring",
                                 ifelse(df1$arrival_date_month %in% c("May",
                                                                      "June",
                                                                      "July",
                                                                      "August"),
                                        "Summer",
                                 "Fall")))

# reduce factors for agent
df1$agent_yn = ifelse(df1$agent=="None","No","Yes")

# reduce factors for company
df1$company_yn = ifelse(df1$company=="None","No","Yes")

# duplicate to specific datasets for each group member (so no accidents occur)
# Alex's dataset
dfAP = df1
# Radhika's dataset
dfRJ = df1
# Harold's dataset
dfHL = df1
# David's dataset
dfDSG = df1
# Daniel's dataset
dfDZ = df1
```

Research Question 1: Contingency tables- independence between categorical variables?

Creating contingency tables between categorical variables is a method to determine whether the response variable is independent from input variables. In this dataset, there are a number of categorical variables that can be compared to whether a booking is canceled. These include: "arrival_date_month","hotel","meal","country","market_segment","is_repeated_guest","previous_cancellations",and "deposit_type". First, contingency tables are outputted for each pair of variables, as well as residual tables and a Chi-squared test output for significance.

```
cont_vars=c("arrival_date_month","hotel","meal","country","market_segment","is_repeated_guest","previous
library(questionr)

# Create loop to make contingency tables for applicable variables


for (i in cont_vars){
  output= sprintf("Contingency table between cancellation and %s",i)
  print(output)
  tab<-table(dfAP$is_canceled,dfAP[[i]])
  #prop.table(tab)
  print(tab)
  print(chisq.test(tab))
  print(chisq.residuals(tab,std=TRUE))
}
```

```
## [1] "Contingency table between cancellation and arrival_date_month"
##
##     April August December February January July June March  May November
##   0  6498   8555     4280     5258    4014 7822 6348  6517 7017     4553
##   1  4507   5232     2362     2686    1802 4730 4531  3145 4675     2120
##
##     October September
##   0    6772      6338
##   1    4227      4098
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 542.5, df = 11, p-value < 2.2e-16
##
##
##      April August December February January   July   June  March    May
##   0  -8.19  -1.53     3.12     6.77   10.31  -0.80  -9.71  10.20  -6.19
##   1   8.19   1.53    -3.12    -6.77  -10.31   0.80   9.71 -10.20   6.19
##
##     November October September
##   0     9.71   -2.44     -4.22
##   1    -9.71    2.44      4.22
## [1] "Contingency table between cancellation and hotel"
##
##     City Hotel Resort Hotel
##   0      45831        28141
##   1      33048        11067
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  tab
## X-squared = 2091, df = 1, p-value < 2.2e-16
##
##
##     City Hotel Resort Hotel
##   0     -45.73        45.73
##   1      45.73       -45.73
## [1] "Contingency table between cancellation and meal"
##
##        BB    FB    HB    SC
##   0 56851   319  9374  7428
##   1 34428   478  4980  4229
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 228.58, df = 3, p-value < 2.2e-16
##
##
##         BB     FB     HB     SC
##   0  -4.71 -13.24   7.04   2.54
##   1   4.71  13.24  -7.04  -2.54
## [1] "Contingency table between cancellation and country"
##
##       ABW   AGO   AIA   ALB   AND   ARE   ARG   ARM   ASM   ATA   ATF   AUS
##   0     2   156     1    10     2     7   160     6     1     2     1   319
##   1     0   205     0     2     5    43    53     2     0     0     0   107
##
##       AUT   AZE   BDI   BEL   BEN   BFA   BGD   BGR   BHR   BHS   BIH   BLR
##   0  1033     8     1  1865     0     1     3    63     1     1    10    17
##   1   229     9     0   473     3     0     9    12     4     0     3     8
##
##       BOL   BRA   BRB   BWA   CAF   CHE   CHL   CHN   CIV   CMR    CN   COL
##   0    10  1388     4     1     5  1296    49   536     4    10  1024    48
##   1     0   829     0     0     0   428    16   462     2     0   254    23
##
##       COM   CPV   CRI   CUB   CYM   CYP   CZE   DEU   DJI   DMA   DNK   DOM
##   0     2    12    18     8     1    40   134  6067     1     1   326     6
##   1     0    12     1     0     0    11    37  1218     0     0   109     8
##
##       DZA   ECU   EGY   ESP   EST   ETH   FIN   FJI   FRA   FRO   GAB   GBR
##   0    82    19    21  6368    65     2   377     0  8457     1     2  9644
##   1    21     8    11  2176    18     1    69     1  1932     4     2  2451
##
##       GEO   GGY   GHA   GIB   GLP   GNB   GRC   GTM   GUY   HKG   HND   HRV
##   0     7     0     2     7     0     8    93     4     1     3     0    75
##   1    15     3     2    11     2     1    35     0     0    26     1    25
##
##       HUN   IDN   IMN   IND   IRL   IRN   IRQ   ISL   ISR   ITA   JAM   JEY
##   0   153    11     0   116  2541    59    14    52   500  2424     6     0
##   1    77    24     2    35   832    23     0     4   169  1332     0     8
```

```
##
##       JOR   JPN   KAZ   KEN   KHM   KIR   KNA   KOR   KWT   LAO   LBN   LBY
##   0    18   169    14     4     0     1     2    78    10     2    22     8
##   1     3    28     5     2     2     0     0    55     6     0     9     0
##
##       LCA   LIE   LKA   LTU   LUX   LVA   MAC   MAR   MCO   MDG   MDV   MEX
##   0     1     2     7    74   176    46     1   149     3     1     3    74
##   1     0     1     0     7   109     9    15   109     1     0     9    10
##
##       MKD   MLI   MLT   MMR   MNE   MOZ   MRT   MUS   MWI   MYS   MYT   NAM
##   0     8     1    13     1     3    47     1     6     2    25     0     1
##   1     2     0     5     0     2    19     0     1     0     3     2     0
##
##       NCL   NGA   NIC   NLD   NOR   NPL   NZL   OMN   PAK   PAN   PER   PHL
##   0     1    13     0  1714   425     1    68    14     5     9    23    15
##   1     0    21     1   387   181     0     6     4     9     0     6    25
##
##       PLW   POL   PRI   PRT   PRY   PYF   QAT   ROU   RUS   RWA   SAU   SDN
##   0     1   702    10 20448     4     1     3   366   388     2    15     1
##   1     0   213     2 27490     0     0    11   134   239     0    33     0
##
##       SEN   SGP   SLE   SLV   SMR   SRB   STP   SUR   SVK   SVN   SWE   SYC
##   0     3    22     1     2     1    98     2     5    41    40   792     1
##   1     8    16     0     0     0     3     0     0    24    15   227     1
##
##       SYR   TGO   THA   TJK   TMP   TUN   TUR   TWN   TZA   UGA   UKR   UMI
##   0     3     2    41     1     2    19   146    37     1     2    48     0
##   1     0     0    18     8     1    19   102    14     3     0    20     1
##
##       URY   USA   UZB   VEN   VGB   VNM   ZAF   ZMB   ZWE
##   0    23  1588     2    14     0     6    49     1     2
##   1     9   501     2    12     1     2    31     1     2

## Warning in chisq.test(tab): Chi-squared approximation may be incorrect

##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 15838, df = 176, p-value < 2.2e-16

## Warning in stats::chisq.test(tab): Chi-squared approximation may be incorrect

##
##         ABW     AGO     AIA     ALB     AND     ARE     ARG     ARM     ASM
##   0    1.09   -7.64    0.77    1.48   -1.86   -7.11    3.77    0.72    0.77
##   1   -1.09    7.64   -0.77   -1.48    1.86    7.11   -3.77   -0.72   -0.77
##
##         ATA     ATF     AUS     AUT     AZE     BDI     BEL     BEN     BFA
##   0    1.09    0.77    5.23   14.18   -1.33    0.77   17.29   -2.24    0.77
##   1   -1.09   -0.77   -5.23  -14.18    1.33   -0.77  -17.29    2.24   -0.77
##
##         BGD     BGR     BHR     BHS     BIH     BLR     BOL     BRA     BRB
```

```
##   0   -2.70    3.82   -1.97    0.77    1.06    0.55    2.44   -0.03    1.54
##   1    2.70   -3.82    1.97   -0.77   -1.06   -0.55   -2.44    0.03   -1.54
##
##         BWA     CAF     CHE     CHL     CHN     CIV     CMR      CN     COL
##   0    0.77    1.73   10.84    2.12   -5.86    0.20    2.44   12.99    0.86
##   1   -0.77   -1.73  -10.84   -2.12    5.86   -0.20   -2.44  -12.99   -0.86
##
##         COM     CPV     CRI     CUB     CYM     CYP     CZE     DEU     DJI
##   0    1.09   -1.28    2.89    2.18    0.77    2.33    4.25   37.59    0.77
##   1   -1.09    1.28   -2.89   -2.18   -0.77   -2.33   -4.25  -37.59   -0.77
##
##         DMA     DNK     DOM     DZA     ECU     EGY     ESP     EST     ETH
##   0    0.77    5.31   -1.53    3.56    0.83    0.35   23.59    2.95    0.14
##   1   -0.77   -5.31    1.53   -3.56   -0.83   -0.35  -23.59   -2.95   -0.14
##
##         FIN     FJI     FRA     FRO     GAB     GBR     GEO     GGY     GHA
##   0    9.57   -1.29   41.39   -1.97   -0.52   41.02   -2.99   -2.24   -0.52
##   1   -9.57    1.29  -41.39    1.97    0.52  -41.02    2.99    2.24    0.52
##
##         GIB     GLP     GNB     GRC     GTM     GUY     HKG     HND     HRV
##   0   -2.08   -1.83    1.63    2.34    1.54    0.77   -5.82   -1.29    2.56
##   1    2.08    1.83   -1.63   -2.34   -1.54   -0.77    5.82    1.29   -2.56
##
##         HUN     IDN     IMN     IND     IRL     IRN     IRQ     ISL     ISR
##   0    1.22   -3.82   -1.83    3.60   15.46    1.74    2.89    4.68    6.49
##   1   -1.22    3.82    1.83   -3.60  -15.46   -1.74   -2.89   -4.68   -6.49
##
##         ITA     JAM     JEY     JOR     JPN     KAZ     KEN     KHM     KIR
##   0    2.44    1.89   -3.66    2.19    6.72    1.00    0.20   -1.83    0.77
##   1   -2.44   -1.89    3.66   -2.19   -6.72   -1.00   -0.20    1.83   -0.77
##
##         KNA     KOR     KWT     LAO     LBN     LBY     LCA     LIE     LKA
##   0    1.09   -0.95   -0.01    1.09    0.96    2.18    0.77    0.14    2.04
##   1   -1.09    0.95    0.01   -1.09   -0.96   -2.18   -0.77   -0.14   -2.04
##
##         LTU     LUX     LVA     MAC     MAR     MCO     MDG     MDV     MEX
##   0    5.34   -0.31    3.22   -4.66   -1.63    0.51    0.77   -2.70    4.82
##   1   -5.34    0.31   -3.22    4.66    1.63   -0.51   -0.77    2.70   -4.82
##
##         MKD     MLI     MLT     MMR     MNE     MOZ     MRT     MUS     MWI
##   0    1.13    0.77    0.84    0.77   -0.12    1.44    0.77    1.26    1.09
##   1   -1.13   -0.77   -0.84   -0.77    0.12   -1.44   -0.77   -1.26   -1.09
##
##         MYS     MYT     NAM     NCL     NGA     NIC     NLD     NOR     NPL
##   0    2.91   -1.83    0.77    0.77   -2.94   -1.29   18.11    3.82    0.77
##   1   -2.91    1.83   -0.77   -0.77    2.94    1.29  -18.11   -3.82   -0.77
##
##         NZL     OMN     PAK     PAN     PER     PHL     PLW     POL     PRI
##   0    5.20    1.33   -2.08    2.32    1.86   -3.29    0.77    8.84    1.48
##   1   -5.20   -1.33    2.08   -2.32   -1.86    3.29   -0.77   -8.84   -1.48
##
##         PRT     PRY     PYF     QAT     ROU     RUS     RWA     SAU     SDN
##   0 -117.37    1.54    0.77   -3.19    4.89   -0.39    1.09   -4.50    0.77
##   1  117.37   -1.54   -0.77    3.19   -4.89    0.39   -1.09    4.50   -0.77
```

```
##
##         SEN     SGP     SLE     SLV     SMR     SRB     STP     SUR     SVK
##   0   -2.43   -0.61    0.77    1.09    0.77    7.15    1.09    1.73    0.07
##   1    2.43    0.61   -0.77   -1.09   -0.77   -7.15   -1.09   -1.73   -0.07
##
##         SVN     SWE     SYC     SYR     TGO     THA     TJK     TMP     TUN
##   0    1.55   10.00   -0.37    1.34    1.09    1.09   -3.20    0.14   -1.61
##   1   -1.55  -10.00    0.37   -1.34   -1.09   -1.09    3.20   -0.14    1.61
##
##         TUR     TWN     TZA     UGA     UKR     UMI     URY     USA     UZB
##   0   -1.23    1.46   -1.56    1.09    1.35   -1.29    1.08   12.75   -0.52
##   1    1.23   -1.46    1.56   -1.09   -1.35    1.29   -1.08  -12.75    0.52
##
##         VEN     VGB     VNM     ZAF     ZMB     ZWE
##   0   -0.93   -1.29    0.72   -0.26   -0.37   -0.52
##   1    0.93    1.29   -0.72    0.26    0.37    0.52
## [1] "Contingency table between cancellation and market_segment"
##
##     Aviation Complementary Corporate Direct Groups Offline TA/TO Online TA
##   0      180           626      4076  10374   7661         15722     35333
##   1       51            87       976   1916  12093          8270     20722
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 8301.6, df = 6, p-value < 2.2e-16
##
##
##     Aviation Complementary Corporate Direct Groups Offline TA/TO Online TA
##   0     4.81         13.93     27.09  52.70 -75.97         10.36      2.64
##   1    -4.81        -13.93    -27.09 -52.70  75.97        -10.36     -2.64
## [1] "Contingency table between cancellation and is_repeated_guest"
##
##         0     1
##   0 71025  2947
##   1 43566   549
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  tab
## X-squared = 720.93, df = 1, p-value < 2.2e-16
##
##
##          0      1
##   0 -26.87  26.87
##   1  26.87 -26.87
## [1] "Contingency table between cancellation and previous_cancellations"
##
##         0     1     2     3     4     5     6    11    13    14    19    21
##   0 73450   321    74    45    24    17    15    25     1     0     0     0
##   1 38199  5689    37    20     7     2     7    10    11    14    19     1
##
##        24    25    26
##   0     0     0     0
```

```
##   1    48    25    26

## Warning in chisq.test(tab): Chi-squared approximation may be incorrect

##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 9154.4, df = 14, p-value < 2.2e-16

## Warning in stats::chisq.test(tab): Chi-squared approximation may be incorrect

##
##          0      1      2      3      4      5      6     11     13     14
##   0  93.02 -94.26   0.88   1.10   1.70   2.42   0.54   1.07  -3.89  -4.85
##   1 -93.02  94.26  -0.88  -1.10  -1.70  -2.42  -0.54  -1.07   3.89   4.85
##
##         19     21     24     25     26
##   0  -5.64  -1.29  -8.97  -6.48  -6.60
##   1   5.64   1.29   8.97   6.48   6.60
## [1] "Contingency table between cancellation and deposit_type"
##
##     No Deposit Non Refund Refundable
##   0      73753         93        126
##   1      29599      14480         36
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 27314, df = 2, p-value < 2.2e-16
##
##
##     No Deposit Non Refund Refundable
##   0     164.03    -165.26       3.99
##   1    -164.03     165.26      -3.99
```

All of the categorical variables are dependent on the response variable, "is_canceled".

Tests of 2×2-contingency table

As "is_repeated_guest" and "hotel" are binomial variables, they can be compared to "is_canceled" to create 2x2-contingency tables.

```
tab_hotel<-table(dfAP$is_canceled,dfAP$hotel)

no1=tab_hotel[1,1]
no2=tab_hotel[2,1]

tot1=no1+tab_hotel[1,2]
tot2=no2+tab_hotel[2,2]

riskdifference(no1,no2,tot1,tot2, conf.level = 0.95)
```

```
##                  Cases People at risk         Risk
## Exposed   4.583100e+04   7.397200e+04 6.195723e-01
## Unexposed 3.304800e+04   4.411500e+04 7.491329e-01
## Total     7.887900e+04   1.180870e+05 6.679736e-01

##
##  Risk difference and its significance probability (H0: The difference
##  equals to zero)
##
## data:  no1 no2 tot1 tot2
## p-value < 2.2e-16
## 95 percent confidence interval:
##  -0.1349091 -0.1242123
## sample estimates:
## [1] -0.1295607
```

```
riskratio(no1,no2,tot1,tot2)
```

```
##            Disease Nondisease Total
## Exposed      45831      28141 73972
## Nonexposed   33048      11067 44115

## Warning in n1 * m1: NAs produced by integer overflow

## Warning in n1 * n2: NAs produced by integer overflow

##
##  Risk ratio estimate and its significance probability
##
## data:  no1 no2 tot1 tot2
## p-value = NA
## 95 percent confidence interval:
##  0.8206157 0.8335398
## sample estimates:
## [1] 0.8270525
```

```
oddsratio(as.matrix(tab_hotel), conf.level = 0.95, p.calc.by.independence = TRUE)
```

```
##            Disease Nondisease  Total
## Exposed      45831      28141  73972
## Nonexposed   33048      11067  44115
## Total        78879      39208 118087

## Warning in N1 * M1: NAs produced by integer overflow

## Warning in N1 * N0: NAs produced by integer overflow

##
##  Odds ratio estimate and its significance probability
##
## data:  as.matrix(tab_hotel)
```

```
## p-value = NA
## 95 percent confidence interval:
##  0.5313112 0.5598351
## sample estimates:
## [1] 0.5453867
```

```
tab_rep<-table(dfAP$is_canceled,dfAP$is_repeated_guest)

no1=tab_rep[1,1]
no2=tab_rep[2,1]

tot1=no1+tab_rep[1,2]
tot2=no2+tab_rep[2,2]

riskdifference(no1,no2,tot1,tot2, conf.level = 0.95)
```

```
##                  Cases People at risk         Risk
## Exposed   7.102500e+04   7.397200e+04 9.601606e-01
## Unexposed 4.356600e+04   4.411500e+04 9.875553e-01
## Total     1.145910e+05   1.180870e+05 9.703947e-01

##
##  Risk difference and its significance probability (H0: The difference
##  equals to zero)
##
## data:  no1 no2 tot1 tot2
## p-value < 2.2e-16
## 95 percent confidence interval:
##  -0.02914298 -0.02564632
## sample estimates:
## [1] -0.02739465
```

```
riskratio(no1,no2,tot1,tot2)
```

```
##            Disease Nondisease Total
## Exposed      71025       2947 73972
## Nonexposed   43566        549 44115

## Warning in n1 * m1: NAs produced by integer overflow

## Warning in n1 * n2 * m1: NAs produced by integer overflow

##
##  Risk ratio estimate and its significance probability
##
## data:  no1 no2 tot1 tot2
## p-value = NA
## 95 percent confidence interval:
##  0.9705084 0.9740150
## sample estimates:
## [1] 0.9722601
```

```
oddsratio(as.matrix(tab_rep), conf.level = 0.95, p.calc.by.independence = TRUE)
```

```
##            Disease Nondisease  Total
## Exposed      71025       2947  73972
## Nonexposed   43566        549  44115
## Total       114591       3496 118087
```

```
## Warning in N1 * M1: NAs produced by integer overflow
```

```
## Warning in N1 * N0: NAs produced by integer overflow
```

```
##
##  Odds ratio estimate and its significance probability
##
## data:  as.matrix(tab_rep)
## p-value = NA
## 95 percent confidence interval:
##  0.2770450 0.3329363
## sample estimates:
## [1] 0.3037077
```

Performing the 2x2 contingency tests once again proves these variables to be dependent on the response variable.

Research Question 2: Which model is the best at predicting hotel booking cancellation in Portugal between Logistic Regression, Classification Tree, Random Forest Analysis, and Linear/Quadratic Discriminant Analysis.

Logistic Regression

```
dfRJ = df %>%
  dplyr::select(-c(reservation_status_date, reservation_status, company, previous_bookings_not_canceled
            country))

# Assuming that the "NULL" represents bookings made without an agent,
# we convert the variable 'agent' to a binary categorical variable
# with 0 representing "NULL" and 1 representing valid agents.
dfRJ$agent = ifelse(dfRJ$agent == "NULL", "yes", "no")

# Create a feature for consistent_room_type and drop assigned_room_type and reserved_room_type
dfRJ$consistent_room_type = ifelse(dfRJ$assigned_room_type == dfRJ$reserved_room_type, "yes", "no")
dfRJ = dfRJ %>%
  dplyr::select(-c(assigned_room_type, reserved_room_type))

## Create arrival_date_season from month data and drop months
dfRJ$arrival_date_season = ifelse(dfRJ$arrival_date_month %in% c("December", "January", "February"), "W
                                  ifelse(dfRJ$arrival_date_month %in% c("March", "April"), "Spring",
                                  ifelse(dfRJ$arrival_date_month %in% c("May", "June", "July", "August")
                                  "Fall")))

dfRJ = dfRJ %>% dplyr::select(-c(arrival_date_month))

# Convert year to factor
dfRJ$arrival_date_year = as.factor(dfRJ$arrival_date_year)

#convert all categorical variables to factors
dfRJ = dfRJ %>% mutate_if(is.character, as.factor)

# Check the observations in the levels of the market segment variable
# Drop levels if there are very few observations available.
table(dfRJ$market_segment)
```

Data wrangling for Logistic Regression

```
##
##      Aviation Complementary     Corporate        Direct        Groups
##           237           743          5295         12606         19811
## Offline TA/TO     Online TA     Undefined
##         24219         56477             2
```

```
# Drop the rows with undefined market segment
dfRJ = dfRJ[!dfRJ$market_segment == 'Undefined', ]

# Drop the rows with undefined distribution channel
dfRJ = dfRJ[!dfRJ$distribution_channel == 'Undefined', ]
```

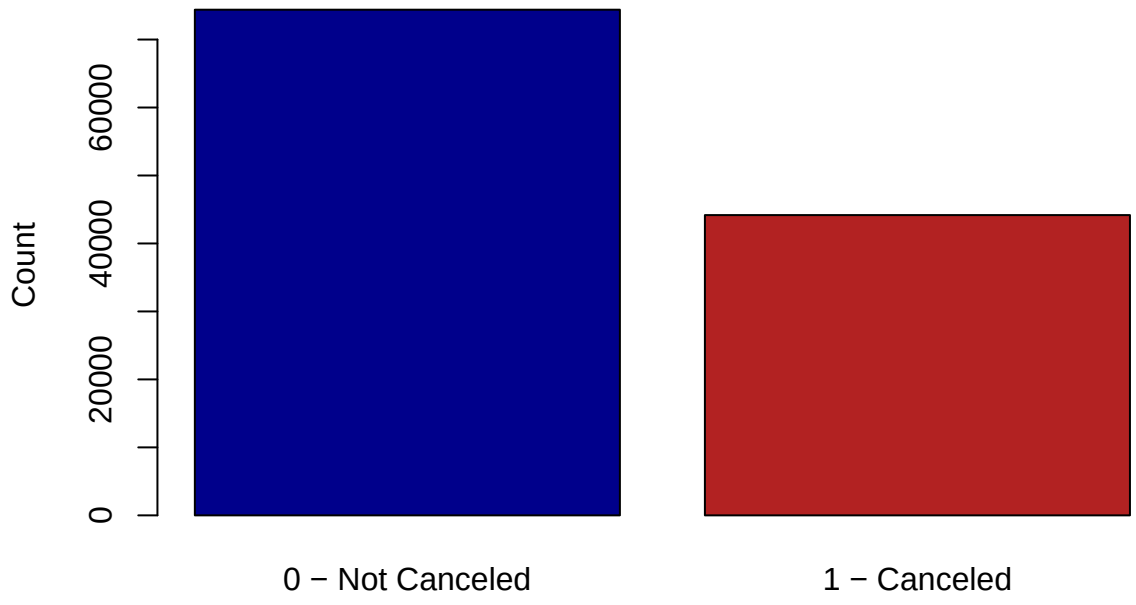
```r
# replace 'Undefined' in 'meal' column with 'SC'
# data source states they are the same thing:
# https://www.sciencedirect.com/science/article/pii/S2352340918315191#tbl1fna
dfRJ$meal[dfRJ$meal == 'Undefined'] = 'SC'

# remove rows where "stays_in_week_nights" and "stays_in_weekend_nights" = 0
dfRJ = dfRJ[dfRJ$stays_in_week_nights > 0 | dfRJ$stays_in_weekend_nights > 0,]

# removed rows where "adult" and "children" = 0
dfRJ = dfRJ[dfRJ$adults > 0 | dfRJ$children > 0,]
```

```r
unique(dfRJ$is_canceled)
```

Check the distribution of the response variable. Convert it to a factor.

```
## [1] 0 1
```

```r
#png(file="./Images/plot1.png")
barplot(table(dfRJ$is_canceled), ylab= "Count",
        names.arg = c("0 - Not Canceled", "1 - Canceled"),
        col = c("darkblue", "firebrick"))
```



```
#dev.off()
```

The imbalance suggests that stratification ius required for creating train and test sets.

```
set.seed(10)
NC = nrow(dfRJ[dfRJ$is_canceled == "0", ]); NC
```

```
## [1] 74388
```

```
C = nrow(dfRJ[dfRJ$is_canceled == "1", ]); C
```

```
## [1] 44172
```

```
Nh = c(NC, C)
N = nrow(dfRJ)
n = round(0.75*N)

idx = sampling:::strata(dfRJ, stratanames=c("is_canceled"), size=round(n*round(Nh/N, 1)), method="srswo
train = dfRJ[idx$ID_unit, ]
test = dfRJ[-idx$ID_unit, ]
```

Examine the categorical variables for the maximum number of levels.

```
train_cat = train %>% keep(is.factor)
str(train_cat)
```

```
## 'data.frame':    88920 obs. of  10 variables:
##  $ hotel               : Factor w/ 2 levels "City Hotel","Resort Hotel": 2 2 2 2 2 2 2 2 2 2 ...
##  $ arrival_date_year   : Factor w/ 3 levels "2015","2016",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ meal                : Factor w/ 5 levels "BB","FB","HB",..: 1 1 1 1 1 3 3 1 1 1 ...
##  $ market_segment      : Factor w/ 8 levels "Aviation","Complementary",..: 4 3 7 7 4 7 7 7 7 6 ...
##  $ distribution_channel: Factor w/ 5 levels "Corporate","Direct",..: 2 1 4 4 2 4 4 4 4 4 ...
##  $ deposit_type        : Factor w/ 3 levels "No Deposit","Non Refund",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ agent               : Factor w/ 2 levels "no","yes": 2 1 1 1 2 1 1 1 1 1 ...
##  $ customer_type       : Factor w/ 4 levels "Contract","Group",..: 3 3 3 3 3 3 3 3 3 1 ...
##  $ consistent_room_type: Factor w/ 2 levels "no","yes": 1 2 2 2 2 2 2 2 1 2 ...
##  $ arrival_date_season : Factor w/ 4 levels "Fall","Spring",..: 3 3 3 3 3 3 3 3 3 3 ...
```

```
logit_fit = glm(is_canceled~., family = binomial, data = train)
```

Apply the logistic regression model

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

```
summary(logit_fit)
```

```
##
## Call:
## glm(formula = is_canceled ~ ., family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.0354  -0.7713  -0.2812   0.7377   3.4529
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   -3.836e+00  2.162e-01 -17.746  < 2e-16 ***
## hotelResort Hotel              1.390e-01  2.142e-02   6.488 8.68e-11 ***
## lead_time                      3.991e-03  1.177e-04  33.915  < 2e-16 ***
## arrival_date_year2016         -8.379e-02  3.047e-02  -2.750 0.005963 **
## arrival_date_year2017         -7.647e-02  3.679e-02  -2.079 0.037649 *
## arrival_date_week_number       1.349e-03  1.029e-03   1.311 0.189770
## arrival_date_day_of_month     -9.045e-05  1.013e-03  -0.089 0.928884
## stays_in_weekend_nights        2.509e-02  1.019e-02   2.463 0.013795 *
## stays_in_week_nights           3.059e-02  5.404e-03   5.661 1.50e-08 ***
## adults                         1.387e-01  1.891e-02   7.331 2.29e-13 ***
## children                       1.158e-01  2.224e-02   5.204 1.95e-07 ***
## babies                         2.233e-01  9.720e-02   2.298 0.021587 *
## mealFB                         6.979e-01  1.234e-01   5.655 1.55e-08 ***
## mealHB                        -1.130e-01  3.060e-02  -3.692 0.000222 ***
## mealSC                         4.592e-02  2.780e-02   1.652 0.098518 .
## market_segmentComplementary    3.187e-01  2.650e-01   1.203 0.229103
## market_segmentCorporate       -3.561e-01  2.004e-01  -1.777 0.075554 .
## market_segmentDirect          -1.546e-02  2.229e-01  -0.069 0.944694
## market_segmentGroups          -1.112e-02  2.108e-01  -0.053 0.957930
## market_segmentOffline TA/TO   -6.789e-01  2.115e-01  -3.210 0.001329 **
## market_segmentOnline TA        6.180e-01  2.108e-01   2.931 0.003375 **
## distribution_channelDirect    -5.332e-01  1.081e-01  -4.933 8.09e-07 ***
## distribution_channelGDS       -1.298e+00  2.477e-01  -5.240 1.61e-07 ***
## distribution_channelTA/TO     -7.537e-02  8.679e-02  -0.868 0.385166
## is_repeated_guest             -1.130e+00  8.661e-02 -13.043  < 2e-16 ***
## previous_cancellations         1.757e+00  5.507e-02  31.905  < 2e-16 ***
## booking_changes               -3.484e-01  1.727e-02 -20.180  < 2e-16 ***
## deposit_typeNon Refund         5.217e+00  1.185e-01  44.014  < 2e-16 ***
## deposit_typeRefundable         9.576e-02  2.450e-01   0.391 0.695843
## agentyes                      -1.251e-01  4.484e-02  -2.790 0.005263 **
## days_in_waiting_list          -9.609e-04  5.587e-04  -1.720 0.085455 .
## customer_typeGroup            -4.839e-01  2.036e-01  -2.376 0.017492 *
## customer_typeTransient         7.629e-01  5.843e-02  13.057  < 2e-16 ***
## customer_typeTransient-Party   3.030e-01  6.194e-02   4.892 9.98e-07 ***
## adr                            3.746e-03  2.686e-04  13.946  < 2e-16 ***
## required_car_parking_spaces   -2.678e+01  5.447e+01  -0.492 0.622897
## total_of_special_requests     -7.168e-01  1.311e-02 -54.672  < 2e-16 ***
## consistent_room_typeyes        1.817e+00  4.558e-02  39.866  < 2e-16 ***
## arrival_date_seasonSpring      3.845e-02  3.917e-02   0.982 0.326246
## arrival_date_seasonSummer     -5.946e-02  2.902e-02  -2.049 0.040460 *
## arrival_date_seasonWinter      1.727e-01  3.620e-02   4.770 1.84e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 119688  on 88919  degrees of freedom
## Residual deviance:  76913  on 88879  degrees of freedom
## AIC: 76995
##
## Number of Fisher Scoring iterations: 13
```

Check for multicollinearity in the logistic model coeficients.

```
library(car)
vif(logit_fit)
```

```
##                                  GVIF Df GVIF^(1/(2*Df))
## hotel                        1.279260  1        1.131044
## lead_time                    1.628320  1        1.276056
## arrival_date_year            2.032618  2        1.194026
## arrival_date_week_number     2.393695  1        1.547157
## arrival_date_day_of_month    1.030816  1        1.015291
## stays_in_weekend_nights      1.389387  1        1.178723
## stays_in_week_nights         1.492084  1        1.221509
## adults                       1.203477  1        1.097031
## children                     1.213391  1        1.101540
## babies                       1.016433  1        1.008183
## meal                         1.458719  3        1.064949
## market_segment              73.889237  6        1.431249
## distribution_channel        29.223068  3        1.755042
## is_repeated_guest            1.283969  1        1.133124
## previous_cancellations       1.353074  1        1.163217
## booking_changes              1.045504  1        1.022499
## deposit_type                 1.099848  2        1.024078
## agent                        2.030432  1        1.424932
## days_in_waiting_list         1.084664  1        1.041472
## customer_type                2.377988  3        1.155318
## adr                          2.112733  1        1.453524
## required_car_parking_spaces  1.000113  1        1.000057
## total_of_special_requests    1.211106  1        1.100502
## consistent_room_type         1.023546  1        1.011705
## arrival_date_season          2.841901  3        1.190149
```

The VIF values are very high for market_segment and distribution_channel. One of the variables can be discarded. market_segment will be used in our model.

The features whose coefficients are *not* significantly different from zero will also be removed. These include-

- arrival_date_week_number
- arrival_date_day_of_month
- required_car_parking_spaces

The p-value for the coefficient of the feature-"days_in_waiting_list" is not significant at $\alpha = 0.05$, but very close to the significance level. Therefore, it will be kept in the model.

```
train2 = train %>% dplyr::select(-c(arrival_date_day_of_month, arrival_date_week_number, required_car_pa
```

```
logit_fit2 = glm(is_canceled~., family = binomial, data = train2)
```

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

```
summary(logit_fit2)
```

```
##
## Call:
## glm(formula = is_canceled ~ ., family = binomial, data = train2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -5.8871  -0.7828  -0.3639   0.7770   3.5859
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  -3.6690130  0.2108135 -17.404  < 2e-16 ***
## hotelResort Hotel            -0.0601073  0.0206245  -2.914 0.003564 **
## lead_time                     0.0041178  0.0001138  36.180  < 2e-16 ***
## arrival_date_year2016        -0.0911337  0.0288019  -3.164 0.001555 **
## arrival_date_year2017        -0.0453998  0.0328867  -1.380 0.167435
## stays_in_weekend_nights       0.0368039  0.0099852   3.686 0.000228 ***
## stays_in_week_nights          0.0401540  0.0052985   7.578 3.50e-14 ***
## adults                        0.1275353  0.0183053   6.967 3.23e-12 ***
## children                      0.0949795  0.0212356   4.473 7.73e-06 ***
## babies                        0.2072572  0.0941196   2.202 0.027661 *
## mealFB                        0.6904671  0.1177451   5.864 4.52e-09 ***
## mealHB                       -0.0991673  0.0297563  -3.333 0.000860 ***
## mealSC                        0.0560222  0.0274593   2.040 0.041332 *
## market_segmentComplementary  -0.1970922  0.2469602  -0.798 0.424828
## market_segmentCorporate      -0.4083888  0.1997638  -2.044 0.040918 *
## market_segmentDirect         -0.6513276  0.1965503  -3.314 0.000920 ***
## market_segmentGroups         -0.1201295  0.1989166  -0.604 0.545898
## market_segmentOffline TA/TO  -0.7719477  0.1980244  -3.898 9.69e-05 ***
## market_segmentOnline TA       0.4763817  0.1969009   2.419 0.015546 *
## is_repeated_guest            -1.1638997  0.0853148 -13.642  < 2e-16 ***
## previous_cancellations        1.6845719  0.0526076  32.021  < 2e-16 ***
## booking_changes              -0.3752231  0.0170573 -21.998  < 2e-16 ***
## deposit_typeNon Refund        5.2277159  0.1181685  44.240  < 2e-16 ***
## deposit_typeRefundable       -0.1107728  0.2349722  -0.471 0.637334
## agentyes                     -0.1668355  0.0412433  -4.045 5.23e-05 ***
## days_in_waiting_list         -0.0011764  0.0005574  -2.111 0.034807 *
## customer_typeGroup           -0.4384524  0.2011574  -2.180 0.029284 *
## customer_typeTransient        0.7228913  0.0577772  12.512  < 2e-16 ***
## customer_typeTransient-Party  0.2809010  0.0613848   4.576 4.74e-06 ***
## adr                           0.0029338  0.0002547  11.520  < 2e-16 ***
## total_of_special_requests    -0.7138709  0.0128639 -55.494  < 2e-16 ***
## consistent_room_typeyes       1.8223269  0.0452136  40.305  < 2e-16 ***
## arrival_date_seasonSpring    -0.0044890  0.0316051  -0.142 0.887054
## arrival_date_seasonSummer    -0.0759435  0.0259050  -2.932 0.003372 **
```

```
## arrival_date_seasonWinter     0.1257436  0.0317726   3.958 7.57e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 119688  on 88919  degrees of freedom
## Residual deviance:  80356  on 88885  degrees of freedom
## AIC: 80426
##
## Number of Fisher Scoring iterations: 7
```

Checking for multicollinearity again.

```
vif(logit_fit2)
```

```
##                               GVIF Df GVIF^(1/(2*Df))
## hotel                     1.261865  1        1.123327
## lead_time                 1.579291  1        1.256698
## arrival_date_year         1.651764  2        1.133671
## stays_in_weekend_nights   1.383520  1        1.176231
## stays_in_week_nights      1.485362  1        1.218754
## adults                    1.195947  1        1.093594
## children                  1.229225  1        1.108704
## babies                    1.017619  1        1.008771
## meal                      1.438753  3        1.062505
## market_segment            5.204176  6        1.147350
## is_repeated_guest         1.288781  1        1.135245
## previous_cancellations    1.348475  1        1.161238
## booking_changes           1.044792  1        1.022151
## deposit_type              1.065449  2        1.015975
## agent                     1.777970  1        1.333405
## days_in_waiting_list      1.081347  1        1.039878
## customer_type             2.350805  3        1.153107
## adr                       2.056688  1        1.434116
## total_of_special_requests 1.197837  1        1.094458
## consistent_room_type      1.022379  1        1.011128
## arrival_date_season       1.882290  3        1.111171
```

The VIF values are satisfactorily low for the all coefficients suggesting the absence of multicollinearity between the features in the model.

Make predictions on the test set

```
prob_predict = predict(logit_fit2, test, type="response")
cancel_predict = rep("0", nrow(test))
cancel_predict[prob_predict >= 0.5]= "1"
```

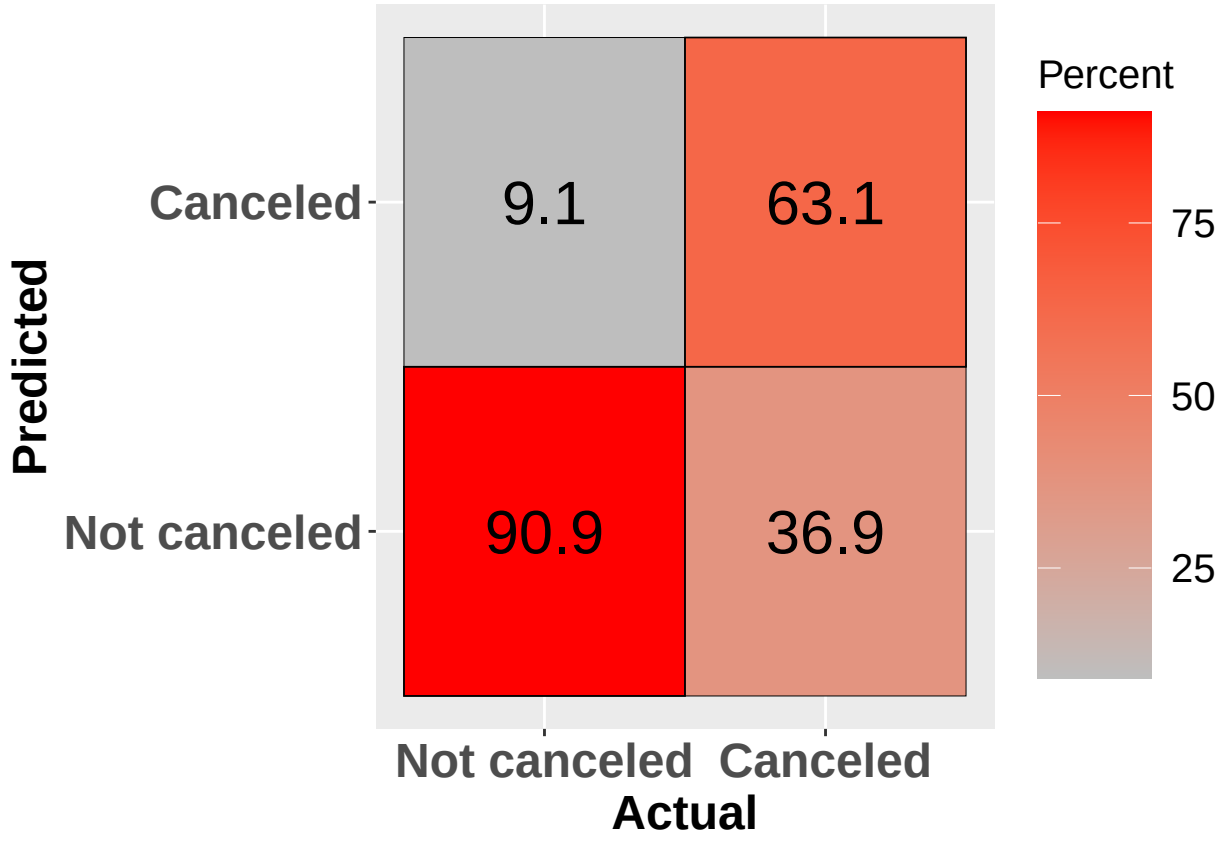
Create confusion matrix for the model performance

```
trueValues = test$is_canceled
confusion_mat = table(cancel_predict, trueValues)
cm = as.matrix(table(cancel_predict, trueValues)); cm
```

```
##                trueValues
## cancel_predict     0     1
##              0 19130  3177
##              1  1906  5427
```

Heatmap for Confusion Matrix

```
# Create dataframe for the confusion matrix
actual = as.data.frame(table(test$is_canceled))
names(actual) = c("Actual","ActualFreq")

# Build confusion matrix
hm_cm = as.data.frame(table(test$is_canceled, cancel_predict))
names(hm_cm) = c("Actual", "Predicted", "Freq")

#calculate percentage of test cases based on actual frequency
hm_cm = merge(hm_cm, actual, by=c("Actual"))
hm_cm$Percent = hm_cm$Freq/hm_cm$ActualFreq*100

#png(file="./Images/plot2.png", width=800, height=600)
tile <- ggplot() +
geom_tile(aes(x=Actual, y=Predicted,fill=Percent),data=hm_cm, color="black",size=0.1) +
labs(x="Actual",y="Predicted") +
  scale_x_discrete(labels=c("0" = "Not canceled", "1" = "Canceled")) +
   scale_y_discrete(labels=c("0" = "Not canceled", "1" = "Canceled")) +
  theme(axis.text.y = element_text(face = "bold", size=18)) +
  theme(axis.text.x = element_text(face = "bold", size=18)) +
  theme(axis.title.x = element_text(face = "bold", size=18)) +
  theme(axis.title.y = element_text(face = "bold", size=18),
        legend.key.size = unit(1.5, 'cm'),
        legend.title = element_text(size=15), #change legend title font size
        legend.text = element_text(size=15))

tile = tile +
geom_text(aes(x=Actual,y=Predicted, label=sprintf("%.1f", Percent)),data=hm_cm, size=8, colour="black") +
scale_fill_gradient(low="grey",high="red")

tile = tile +
geom_tile(aes(x=Actual,y=Predicted),data=subset(hm_cm, as.character(Actual)==as.character(Predicted)), c

#render
tile
```




```
#dev.off()
```

Metrics for Model Performance Next, we calculate various metrics for validating the performnce of the logistic model.

```
n = sum(cm) # number of instances
nc = nrow(cm) # number of classes
diag = diag(cm) # number of correctly classified instances per class
rowsums = apply(cm, 1, sum) # number of instances per class
colsums = apply(cm, 2, sum) # number of predictions per class
p = rowsums / n # distribution of instances over the actual classes
q = colsums / n # distribution of instances over the predicted classes
```

1. Sensitivity

```
sens = cm[4]/sum(cm[2], cm[4])

print(paste0("Sensitivity = ", sens))
```

```
## [1] "Sensitivity = 0.740079094504296"
```

2. Accuracy

```
accuracy=sum(diag)/n;
print(paste0("Accuracy = ", accuracy))
```

```
## [1] "Accuracy = 0.828508771929825"
```

3. Per-class Precision, Recall, and F-1

```
precision = diag / colsums
recall = diag / rowsums
f1 = 2 * precision * recall / (precision + recall)

data.frame(precision, recall, f1)[2, ]
```

```
##   precision    recall        f1
## 1 0.6307531 0.7400791 0.6810567
```

4. Misclassification rate calculation

```
#sum(confusion_mat[2:3])/nrow(test)
misclasRate = 1 - accuracy;
print(paste0("Misclassification Rate = ", misclasRate))
```

```
## [1] "Misclassification Rate = 0.171491228070175"
```

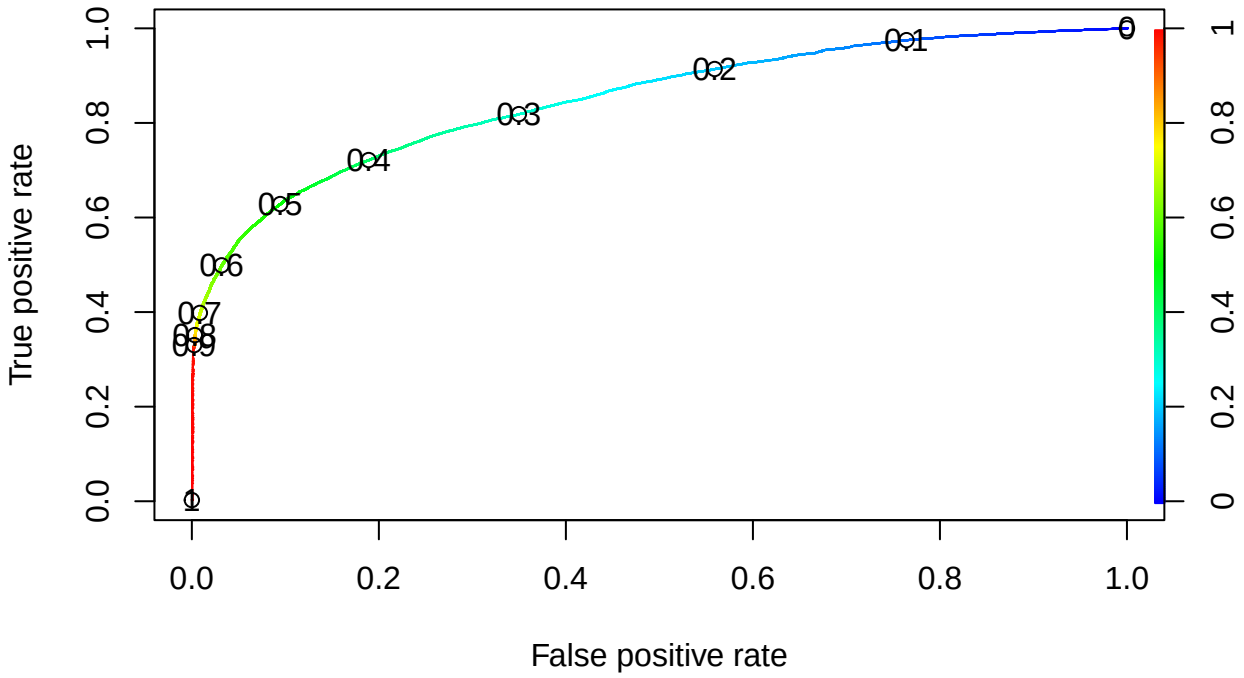
```
library(ROCR)
```

Optimizing the threshold value using the ROC Curve Trying to find a threshold for the binary decision.

```
predicted <- predict(logit_fit2, newdata = train2, type="response")
```

ROC Curve

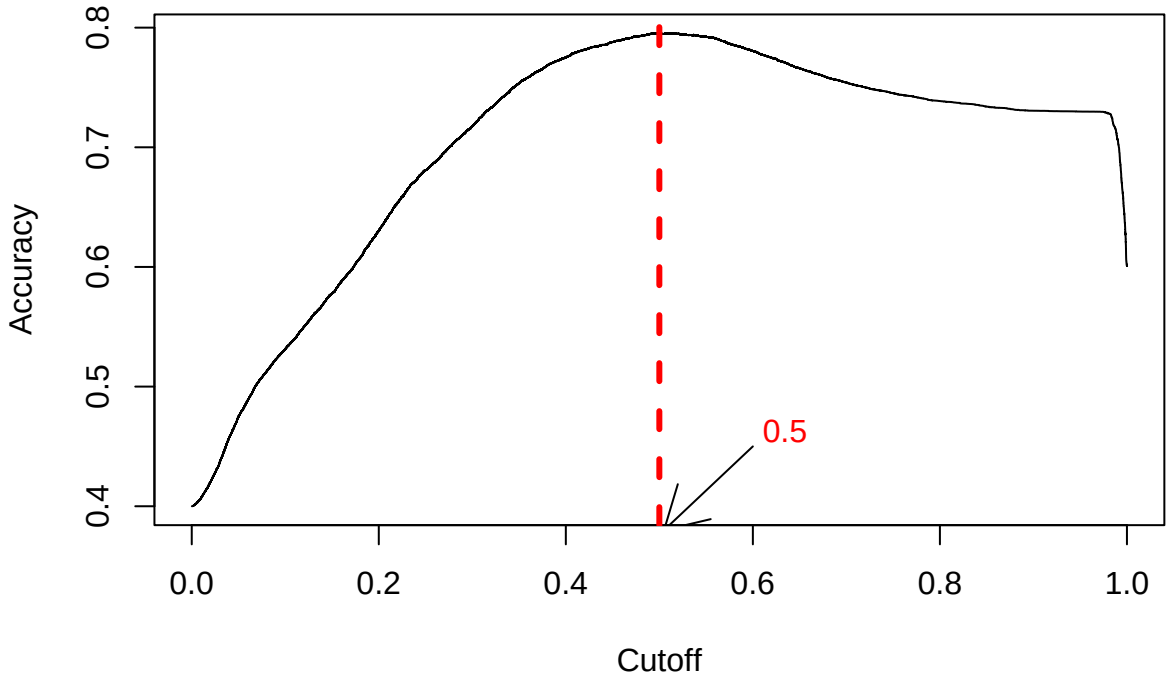
```
pred <- prediction(predicted, train2$is_canceled)
perf <- performance(pred, measure = "tpr", x.measure = "fpr")
#png(file="./Images/roc.png", width=400, height=300)
plot(perf, colorize=T, print.cutoffs.at=seq(0,1,by=0.1))
```




```
# dev.off()
```

Plot the accuracy of the model as a function of the threshold value.

```
acc.perf = performance(pred, measure = "acc")
#png(file="./Images/roc2.png", width=400, height=300)
plot(acc.perf)
abline(v = 0.5, col="red", lwd=3, lty=2)
text(x=0.59, y=0.46, labels="0.5", pos=4, col="red")
arrows(x0=0.6, y0=0.45, x1=0.505, y1=0.38, col="black", lwd=1)
```




```
#dev.off()
```

The threshold of 0.5 gives the best accuracy in the model and shows a good tradeoff between the true and false positive rates. Therefore, the analyses continue with the selected threshold of 0.5.

```
library(caret)
```

Cross Validation To perform stratified partition, the function "createFolds" is used from the package "caret".

```
set.seed(10)
folds<-createFolds(factor(dfRJ$is_canceled), k=10)
```

```
df_fold1 = dfRJ[folds$Fold01, ]
print(table(df_fold1$is_canceled))
```

```
##
##    0    1
## 7438 4418
```

Function to calculate the misclassification rate.

```
library(MASS)

misclassification<-function(idx){
  Train<-dfRJ[-idx,]
  Test<-dfRJ[idx,]
  fit<-glm(is_canceled~., family = binomial, data = Train)
  prob_pred<-predict(fit,Test, type="response")
  cancel_predict = rep("0", nrow(Test))
  cancel_predict[prob_pred >= 0.5]= "1"
  return(1-mean(cancel_predict==Test$is_canceled))
}
```

Next, apply the function "lapply" to pass along a vector of indices to the function "misclassification".

```
mis_rate=lapply(folds,misclassification)
mis_rate
```

```
## $Fold01
## [1] 0.1864879
##
## $Fold02
## [1] 0.1866565
##
## $Fold03
## [1] 0.1890866
##
## $Fold04
## [1] 0.1865036
##
## $Fold05
## [1] 0.1871626
##
## $Fold06
## [1] 0.1933198
##
## $Fold07
## [1] 0.1940789
##
## $Fold08
## [1] 0.192392
##
## $Fold09
## [1] 0.1950067
##
## $Fold10
## [1] 0.1883435
```

Calculate the average misclassification error.

```
mean(as.numeric(mis_rate))
```

```
## [1] 0.1899038
```

Cross Validation for Feature Selection Inspect the feature names

```
names(train2)
```

```
##  [1] "hotel"                     "is_canceled"
##  [3] "lead_time"                 "arrival_date_year"
##  [5] "stays_in_weekend_nights"   "stays_in_week_nights"
##  [7] "adults"                    "children"
##  [9] "babies"                    "meal"
## [11] "market_segment"            "is_repeated_guest"
## [13] "previous_cancellations"    "booking_changes"
## [15] "deposit_type"              "agent"
## [17] "days_in_waiting_list"      "customer_type"
## [19] "adr"                       "total_of_special_requests"
## [21] "consistent_room_type"      "arrival_date_season"
```

Apply 10-fold cross-validation to measure the accuracy rate for models created with incremental addition of features to the base model with just 1 feature.

```
model_1 = train(factor(is_canceled)~hotel, data=dfRJ, trControl = trainControl(method = "CV", number=10)
```

```
model_2 = train(factor(is_canceled)~hotel+lead_time, data=dfRJ, trControl = trainControl(method = "CV",
```

```
model_3 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year, data=dfRJ, trControl = trainConti
```

```
model_4 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights, data=dfR
```

```
model_5 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_w
```

```
model_6 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_w
                +adults, data=dfRJ, trControl = trainControl(method = "CV", number=10), method='glm', fa
```

```
model_7 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_w
                +adults+children, data=dfRJ, trControl = trainControl(method = "CV", number=10), method=
```

```
model_8 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_w
                +adults+children+babies, data=dfRJ, trControl = trainControl(method = "CV", number=10),
```

```
model_9 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_w
                +adults+children+babies+meal, data=dfRJ, trControl = trainControl(method = "CV", number=
```

```
model_10 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_
                +adults+children+babies+meal+market_segment, data=dfRJ, trControl = trainControl(method
```

```
model_11 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_
                +adults+children+babies+meal+market_segment+is_repeated_guest, data=dfRJ, trControl = tr
```

```
model_12 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in
               +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations, da

model_13 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in
               +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                 booking_changes, data=dfRJ, trControl = trainControl(method = "CV", number=10), metho

model_14 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in
               +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                 booking_changes+deposit_type, data=dfRJ, trControl = trainControl(method = "CV", numb

model_15 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in
               +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                 booking_changes+deposit_type+agent, data=dfRJ, trControl = trainControl(method = "CV"

model_16 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in
               +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                 booking_changes+deposit_type+agent+days_in_waiting_list, data=dfRJ, trControl = train

model_17 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in
               +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                 booking_changes+deposit_type+agent+days_in_waiting_list+customer_type, data=dfRJ, trC

model_18 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in
               +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                 booking_changes+deposit_type+agent+days_in_waiting_list+customer_type+adr, data=dfRJ,

model_19 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in
               +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                 booking_changes+deposit_type+agent+days_in_waiting_list+customer_type+adr+total_of_sp

model_20 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in
               +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                 booking_changes+deposit_type+agent+days_in_waiting_list+customer_type+adr+total_of_sp
                 consistent_room_type, data=dfRJ, trControl = trainControl(method = "CV", number=10),

model_21 = train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in
               +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                 booking_changes+deposit_type+agent+days_in_waiting_list+customer_type+adr+total_of_sp
                 consistent_room_type+arrival_date_season, data=dfRJ, trControl = trainControl(method
```

Calculate accuracy with all models

```
accuracy = c(model_1$results[2],
             model_2$results[2],
             model_3$results[2],
             model_4$results[2],
             model_5$results[2],
             model_6$results[2],
```

```
            model_7$results[2],
            model_8$results[2],
            model_9$results[2],
            model_10$results[2],
            model_11$results[2],
            model_12$results[2],
            model_13$results[2],
            model_14$results[2],
            model_15$results[2],
            model_16$results[2],
            model_17$results[2],
            model_18$results[2],
            model_19$results[2],
            model_20$results[2],
            model_21$results[2])

accuracy
```

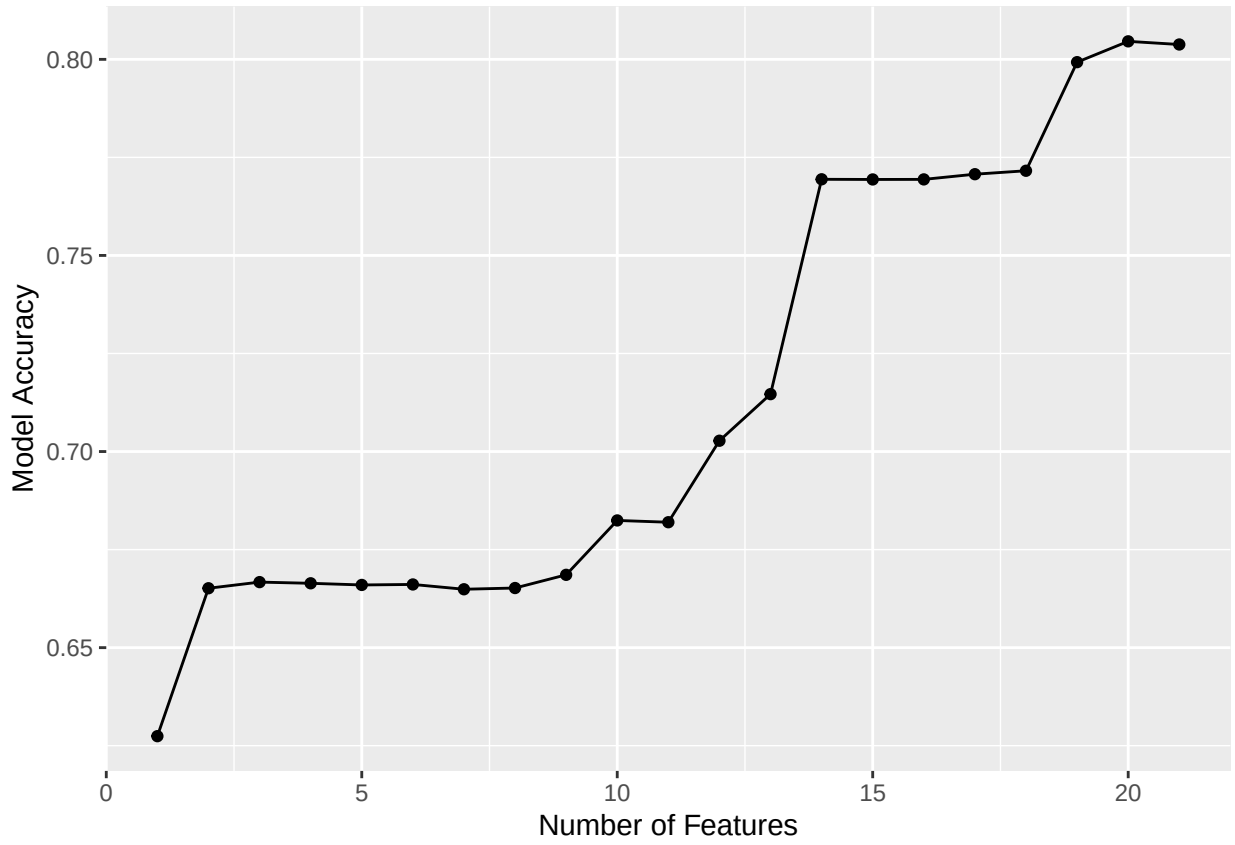
```
## $Accuracy
## [1] 0.6274291
##
## $Accuracy
## [1] 0.6651399
##
## $Accuracy
## [1] 0.6667087
##
## $Accuracy
## [1] 0.6664052
##
## $Accuracy
## [1] 0.6659834
##
## $Accuracy
## [1] 0.66611
##
## $Accuracy
## [1] 0.664887
##
## $Accuracy
## [1] 0.6651991
##
## $Accuracy
## [1] 0.6685644
##
## $Accuracy
## [1] 0.6824308
##
## $Accuracy
## [1] 0.6819753
##
## $Accuracy
## [1] 0.7027581
```

```
##
## $Accuracy
## [1] 0.7146339
##
## $Accuracy
## [1] 0.7694163
##
## $Accuracy
## [1] 0.7693742
##
## $Accuracy
## [1] 0.769391
##
## $Accuracy
## [1] 0.7707153
##
## $Accuracy
## [1] 0.7715925
##
## $Accuracy
## [1] 0.799283
##
## $Accuracy
## [1] 0.8045883
##
## $Accuracy
## [1] 0.8037871
```

Plot the accuracy rate against the number of features in the model.

```
accuracy_df = data.frame(matrix(unlist(accuracy), nrow=length(accuracy), byrow=TRUE), seq(1,21, 1))
names(accuracy_df) = c("cv_accuracy", "features")
```

```
ggplot(data=accuracy_df, aes(features, cv_accuracy)) + geom_point() + geom_line() +
  labs(x="Number of Features",
         y="Model Accuracy")
```




```
#ggsave("./Images/cvPlot.png")
```

The plot shows that the accuracy rate does not improve beyond the first 19 features in the model. Therefore, removing consistent_room_type and arrival_date_season from our final logistic regression model is optimal.

Final Model

```
final_model = glm(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_i
                +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                  booking_changes+deposit_type+agent+days_in_waiting_list+customer_type+adr+total_of_sp
                data=train2,
                family = binomial)
```

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

```
summary(final_model)
```

```
##
## Call:
## glm(formula = factor(is_canceled) ~ hotel + lead_time + arrival_date_year +
##     stays_in_weekend_nights + stays_in_week_nights + adults +
##     children + babies + meal + market_segment + is_repeated_guest +
```

```
##     previous_cancellations + booking_changes + deposit_type +
##     agent + days_in_waiting_list + customer_type + adr + total_of_special_requests,
##     family = binomial, data = train2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.0473  -0.7739  -0.4819   0.7940   3.2128
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  -2.0333548  0.2048336  -9.927  < 2e-16 ***
## hotelResort Hotel            -0.1655044  0.0201330  -8.221  < 2e-16 ***
## lead_time                     0.0041191  0.0001047  39.332  < 2e-16 ***
## arrival_date_year2016        -0.0254208  0.0271482  -0.936  0.34908
## arrival_date_year2017         0.0613055  0.0289582   2.117  0.03426 *
## stays_in_weekend_nights       0.0582096  0.0098594   5.904 3.55e-09 ***
## stays_in_week_nights          0.0562150  0.0052232  10.763  < 2e-16 ***
## adults                        0.1240969  0.0176138   7.045 1.85e-12 ***
## children                      0.1017910  0.0207320   4.910 9.11e-07 ***
## babies                        0.1906787  0.0883463   2.158  0.03090 *
## mealFB                        0.8228966  0.1152693   7.139 9.41e-13 ***
## mealHB                       -0.0756588  0.0292055  -2.591  0.00958 **
## mealSC                        0.0817018  0.0271890   3.005  0.00266 **
## market_segmentComplementary  -0.3648996  0.2465955  -1.480  0.13894
## market_segmentCorporate      -0.4963361  0.1982769  -2.503  0.01231 *
## market_segmentDirect         -0.6152723  0.1950369  -3.155  0.00161 **
## market_segmentGroups         -0.1630063  0.1974664  -0.825  0.40909
## market_segmentOffline TA/TO  -0.8365807  0.1965876  -4.256 2.09e-05 ***
## market_segmentOnline TA       0.5128072  0.1954395   2.624  0.00869 **
## is_repeated_guest            -1.1889298  0.0844140 -14.085  < 2e-16 ***
## previous_cancellations        1.7955262  0.0530162  33.867  < 2e-16 ***
## booking_changes              -0.4201032  0.0170856 -24.588  < 2e-16 ***
## deposit_typeNon Refund        5.3606531  0.1174622  45.637  < 2e-16 ***
## deposit_typeRefundable       -0.0718079  0.2306181  -0.311  0.75552
## agentyes                     -0.1918901  0.0406970  -4.715 2.42e-06 ***
## days_in_waiting_list         -0.0012067  0.0005490  -2.198  0.02795 *
## customer_typeGroup           -0.4274344  0.1987788  -2.150  0.03153 *
## customer_typeTransient        0.6618134  0.0572890  11.552  < 2e-16 ***
## customer_typeTransient-Party  0.2483199  0.0607937   4.085 4.41e-05 ***
## adr                           0.0028929  0.0002212  13.075  < 2e-16 ***
## total_of_special_requests    -0.7262606  0.0127209 -57.092  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 119688  on 88919  degrees of freedom
## Residual deviance:  82842  on 88889  degrees of freedom
## AIC: 82904
##
## Number of Fisher Scoring iterations: 7
```

Make predictions on the test set

```
prob_predict = predict(final_model, test, type="response")
cancel_predict = rep("0", nrow(test))
cancel_predict[prob_predict >= 0.5]= "1"
logit_misrate = mean(test$is_canceled!=cancel_predict)*100
logit_misrate
```

```
## [1] 17.30094
```

```
misclassification<-function(idx){
  Train<-dfRJ[-idx,]
  Test<-dfRJ[idx,]
  fit<-glm(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_week_
                +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                  booking_changes+deposit_type+agent+days_in_waiting_list+customer_type+adr+total_of_sp
                data=Train,
                family = binomial)
  prob_pred<-predict(fit,Test, type="response")
  cancel_predict = rep("0", nrow(Test))
  cancel_predict[prob_pred >= 0.5]= "1"
  return(1-mean(cancel_predict==Test$is_canceled))
}
```

Next, apply the function "lapply" to pass along a vector of indices to the function "misclassification".

```
mis_rate=lapply(folds,misclassification)
mis_rate
```

```
## $Fold01
## [1] 0.1962719
##
## $Fold02
## [1] 0.196778
##
## $Fold03
## [1] 0.1992072
##
## $Fold04
## [1] 0.1976381
##
## $Fold05
## [1] 0.1990553
##
## $Fold06
## [1] 0.2028509
##
## $Fold07
## [1] 0.2069838
##
## $Fold08
## [1] 0.2010796
##
## $Fold09
```

```
## [1] 0.2080803
##
## $Fold10
## [1] 0.197031
```

```
logit_cvmisrate=mean(as.numeric(mis_rate))*100
logit_cvmisrate
```

```
## [1] 20.04976
```

Final Model Interpretation

1. Features with a positive influence on predicting hotel booking cancellations:

- City Hotel
- Lead Time
- Arrival in the year 2017 compared with years 2015 and 2016
- No meal package
- Bookings made online
- Previous cancellations by the customer
- Transient Customer Type
- Average daily Rate

These features make it more likely for cancellations to occur.

2. Features with a negative influence:

- Resort Hotel
- Half-board meal package (breakfast and dinner)
- Complementary and corporate bookings; Group bookings; Bookings through travel agents and tour operators
- Repeated guests
- Bookings Changes
- Special Requests

These features make it less likely for cancellations to occur.

Classification Tree

First convert variables into factors:

```
cols=c("is_canceled","arrival_date_year","is_repeated_guest","hotel","meal","reserved_room_type","assig

dfHL[,cols]=lapply(dfHL[,cols], factor)
str(dfHL)
```

```
## 'data.frame':    118087 obs. of  36 variables:
##  $ hotel                         : Factor w/ 2 levels "City Hotel","Resort Hotel": 2 2 2 2 2 2 2 2 2
##  $ is_canceled                   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 2 2 1 ...
##  $ lead_time                     : int  7 13 14 14 0 9 85 75 23 35 ...
##  $ arrival_date_year             : Factor w/ 3 levels "2015","2016",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ arrival_date_month            : Factor w/ 12 levels "April","August",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ arrival_date_week_number      : int  27 27 27 27 27 27 27 27 27 27 ...
##  $ arrival_date_day_of_month     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ stays_in_weekend_nights       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ stays_in_week_nights          : int  1 1 2 2 2 2 3 3 4 4 ...
##  $ adults                        : int  1 1 2 2 2 2 2 2 2 2 ...
##  $ children                      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ babies                        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ meal                          : Factor w/ 4 levels "BB","FB","HB",..: 1 1 1 1 1 2 1 3 1 3 ...
##  $ country                       : chr  "GBR" "GBR" "GBR" "GBR" ...
##  $ market_segment                : Factor w/ 7 levels "Aviation","Complementary",..: 4 3 7 7 4 4 7 6
##  $ distribution_channel          : Factor w/ 4 levels "Corporate","Direct",..: 2 1 4 4 2 2 4 4 4 4 .
##  $ is_repeated_guest             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ previous_cancellations        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ previous_bookings_not_canceled: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ reserved_room_type            : Factor w/ 9 levels "A","B","C","D",..: 1 1 1 1 3 3 1 4 5 4 ...
##  $ assigned_room_type            : Factor w/ 11 levels "A","B","C","D",..: 3 1 1 1 3 3 1 4 5 4 ...
##  $ booking_changes               : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ deposit_type                  : Factor w/ 3 levels "No Deposit","Non Refund",..: 1 1 1 1 1 1 1 1
##  $ agent                         : chr  "None" "304" "240" "240" ...
##  $ company                       : chr  "None" "None" "None" "None" ...
##  $ days_in_waiting_list          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ customer_type                 : Factor w/ 4 levels "Contract","Group",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ adr                           : num  75 75 98 98 107 ...
##  $ required_car_parking_spaces   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ total_of_special_requests     : int  0 0 1 1 0 1 1 0 0 0 ...
##  $ reservation_status            : Factor w/ 3 levels "Canceled","Check-Out",..: 2 2 2 2 2 2 1 1 1 2
##  $ reservation_status_date       : chr  "2015-07-02" "2015-07-02" "2015-07-03" "2015-07-03" ...
##  $ consistent_room_type          : Factor w/ 2 levels "No","Yes": 1 2 2 2 2 2 2 2 2 2 ...
##  $ arrival_date_season           : Factor w/ 4 levels "Fall","Spring",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ agent_yn                      : chr  "No" "Yes" "Yes" "Yes" ...
##  $ company_yn                    : chr  "No" "No" "No" "No" ...
```

Train test splits of the data are created for all the models:

```
set.seed(10)
NC = nrow(dfHL[dfHL$is_canceled == "0", ]); NC
```

```
## [1] 73972
```

```
C = nrow(dfHL[dfHL$is_canceled == "1", ]); C
```

```
## [1] 44115
```

```
Nh = c(NC, C)
N = nrow(dfHL)
n = round(0.75*N)

idx = sampling:::strata(dfHL, stratanames=c("is_canceled"), size=round(n*round(Nh/N, 1)), method="srswo
trainHL=dfHL[idx$ID_unit, ]
testHL=dfHL[-idx$ID_unit, ]
```

10 fold cross validation for all models:

```
folds<-createFolds(factor(dfHL$is_canceled), k=10)
```

Fit the data into a classification tree.

reservation_status is not included because it is too similar to the response variable.

Other variables not fitted: reservation_status_date -is captured in arrival month and season Country, agent, company -too many factors for R to handle agent and company were simplified to Y/N

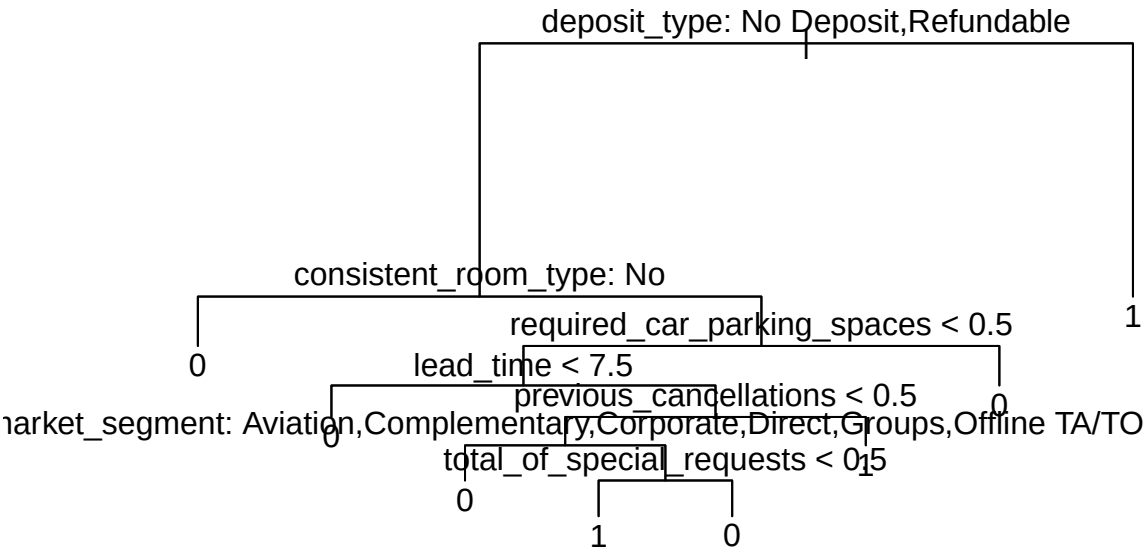
```
hotel_ctree=tree(factor(is_canceled)~.,trainHL[,-c(31)])
```

```
## Warning in tree(factor(is_canceled) ~ ., trainHL[, -c(31)]): NAs introduced by
## coercion
```

```
summary(hotel_ctree)
```

```
##
## Classification tree:
## tree(formula = factor(is_canceled) ~ ., data = trainHL[, -c(31)])
## Variables actually used in tree construction:
## [1] "deposit_type"                "consistent_room_type"
## [3] "required_car_parking_spaces" "lead_time"
## [5] "previous_cancellations"      "market_segment"
## [7] "total_of_special_requests"
## Number of terminal nodes:  8
## Residual mean deviance:  0.8543 = 75660 / 88560
## Misclassification error rate: 0.1981 = 17545 / 88565
```

```
plot(hotel_ctree)
text(hotel_ctree, pretty=0)
```



Non-refund deposit type appears to be the most important factor.

```
summary(dfHL[dfHL$deposit_type=="Non Refund",]$is_canceled)
```

```
##     0     1
##    93 14480
```

However almost all bookings that are "Non Refund" were canceled. The result of the classification tree may be due to over fitting.

```
summary(dfHL$is_canceled)
```

```
##     0     1
## 73972 44115
```

```
14480/44115
```

```
## [1] 0.328233
```

Cancellations by Non Refund deposit type make up about 30% of all cancellations.

According to the data source (https://www.sciencedirect.com/science/article/pii/S2352340918315191) "Non Refund" is described as "a deposit was made in the value of the total stay cost" and "If the payment was equal or exceeded the total cost of stay, the value is set as 'Non Refund'."

It does not explicitly specify that a deposit is not refundable in the conventional sense -only that a deposit of amount equal to or greater than the cost of stay was made.

If this fitted tree were pruned:

```
set.seed(10)
cv.tree=cv.tree(hotel_ctree, FUN = prune.misclass)
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

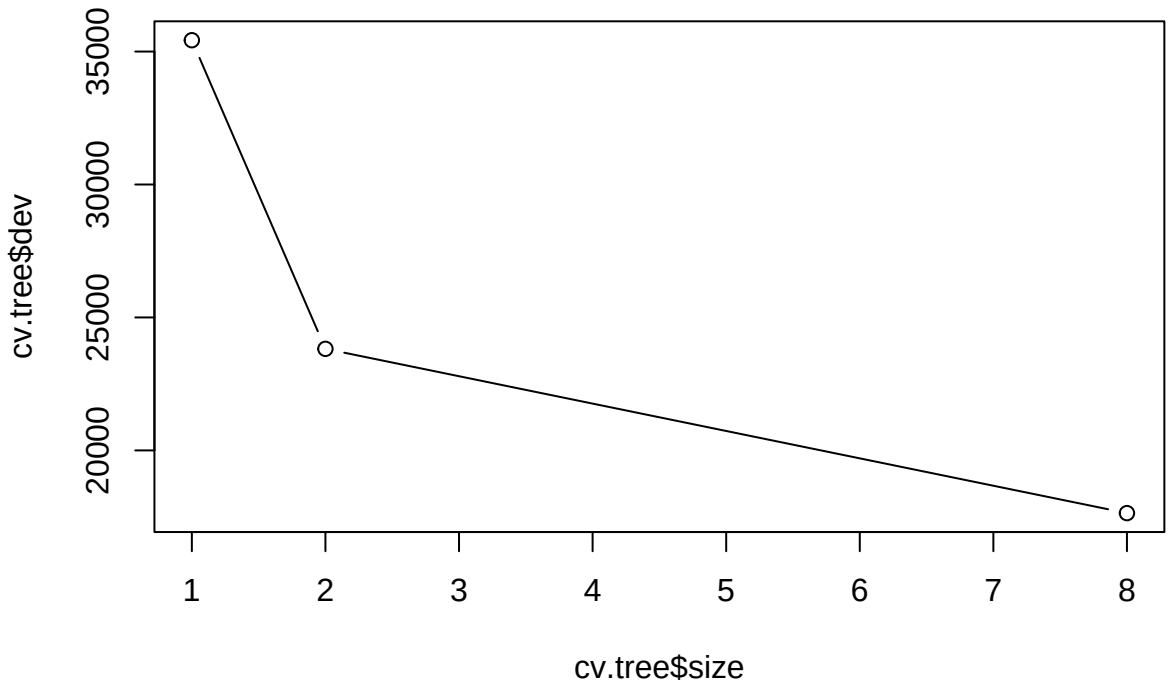
```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion
```

```
## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
plot(cv.tree$size,cv.tree$dev,type='b')
```



Two nodes result in the steepest decrease in misclassification rate.

```
prune.tree=prune.tree(hotel_ctree,best=2)
plot(prune.tree)
text(prune.tree,pretty=0)
```

Resulting in deposit type as the only important factor in classifying whether a hotel booking gets canceled.

```
summary(prune.tree)
```

```
##
## Classification tree:
## snip.tree(tree = hotel_ctree, nodes = 2L)
## Variables actually used in tree construction:
## [1] "deposit_type"
## Number of terminal nodes:  2
## Residual mean deviance:  1.083 = 95890 / 88560
## Misclassification error rate: 0.2689 = 23819 / 88565
```

```
set.seed(10)
tree.pred<-predict(prune.tree,testHL,type = "class")
```

```
## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion
```

```
pred=tree.pred
actual=testHL$is_canceled
table(pred,actual)
```

```
##      actual
## pred     0     1
##    0 20813  5889
##    1    20  2800
```

```
mean(pred!=actual)
```

```
## [1] 0.2001558
```

Deposit type removed If deposit type causes over-fitting to the data, what does a fitted tree look like with deposit type removed?

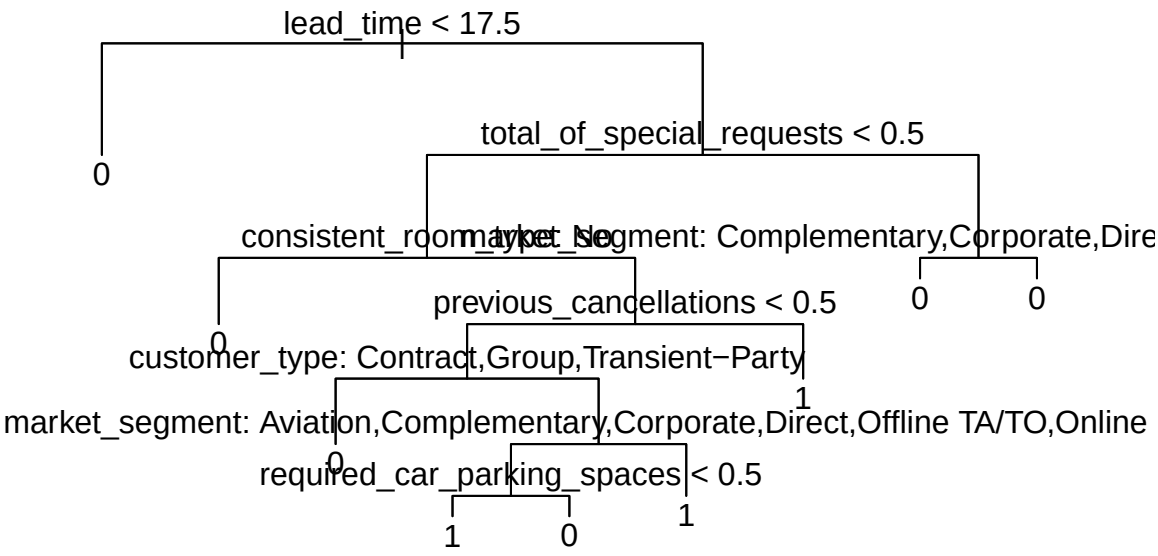
```
hotel_ctree=tree(factor(is_canceled)~.,trainHL[,-c(31,23)])
```

```
## Warning in tree(factor(is_canceled) ~ ., trainHL[, -c(31, 23)]): NAs introduced
## by coercion
```

```
summary(hotel_ctree)
```

```
##
## Classification tree:
## tree(formula = factor(is_canceled) ~ ., data = trainHL[, -c(31,
##     23)])
## Variables actually used in tree construction:
## [1] "lead_time"                   "total_of_special_requests"
## [3] "consistent_room_type"        "previous_cancellations"
## [5] "customer_type"               "market_segment"
## [7] "required_car_parking_spaces"
## Number of terminal nodes:  9
## Residual mean deviance:  0.9787 = 86670 / 88560
## Misclassification error rate: 0.2301 = 20381 / 88565
```

```
plot(hotel_ctree)
text(hotel_ctree, pretty=0)
```



Lead time (days in advance a booking was made) becomes the first split.

```
set.seed(10)
tree.pred<-predict(hotel_ctree,testHL,type = "class")
```

```
## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion
```

```
pred=tree.pred
actual=testHL$is_canceled
table(pred,actual)
```

```
##     actual
## pred     0     1
##    0 18322  3484
##    1  2511  5205
```

```
mean(pred!=actual)
```

```
## [1] 0.2030689
```

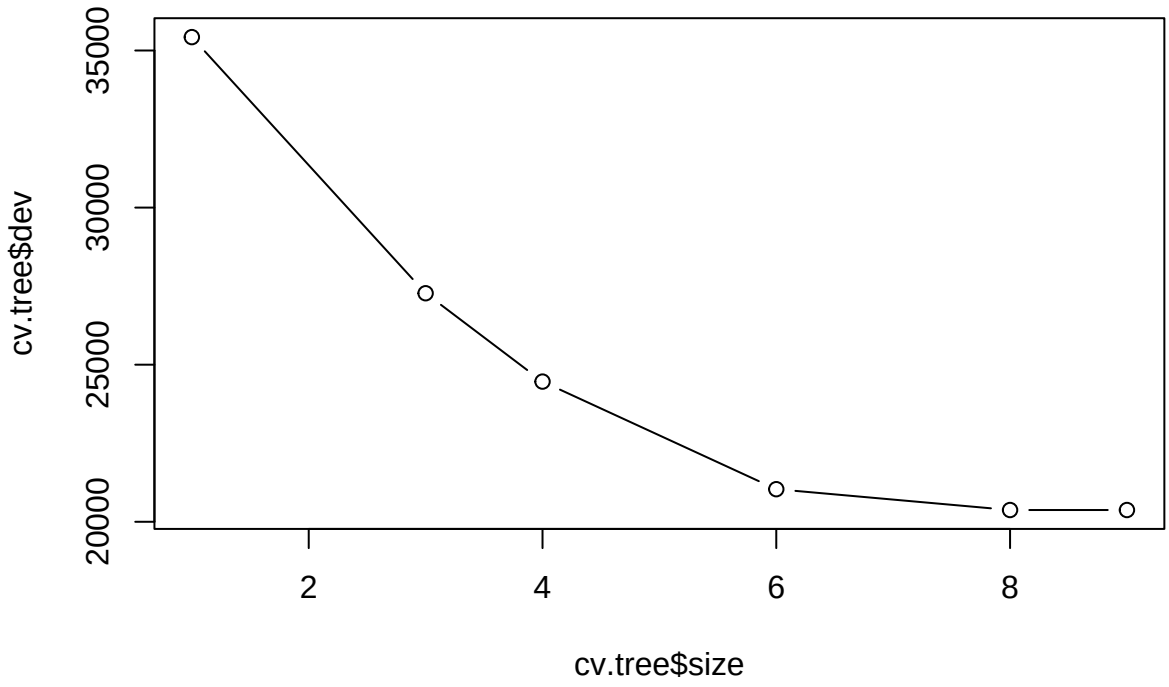
A validation misclassification rate of 0.2030689 is outputted.

Now we prune the tree:

```r
set.seed(10)
cv.tree=cv.tree(hotel_ctree, FUN = prune.misclass)
```

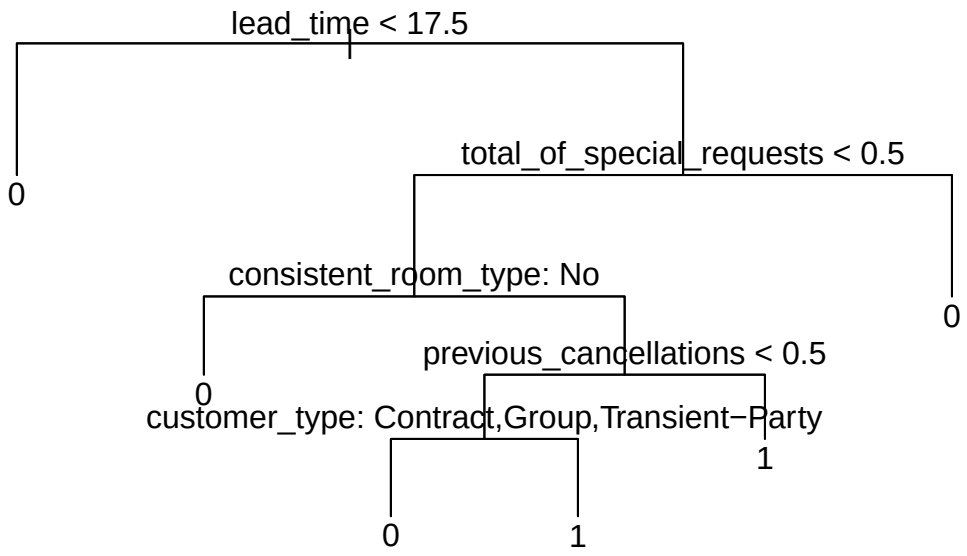
```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
plot(cv.tree$size,cv.tree$dev,type='b')
```



Nodes beyond 6 give a marginal return to lowering misclassification rate.

```
prune.tree=prune.tree(hotel_ctree,best=6)
plot(prune.tree)
text(prune.tree,pretty=0)
```



The above is the resulting pruned tree. The classification decision is as follows: 1. If lead_time (days in advance booking was made) is less than 17.5 days: not canceled; else 2. More than one special request: not canceled; else 3. Guest got an inconsistent room type from their booking: not canceled; else 4. They've had at least one previous cancellation: canceled; else 5: The customer_type is Contract, Group, or Transient-Party: not canceled, else canceled.

```
summary(prune.tree)
```

```
##
## Classification tree:
## snip.tree(tree = hotel_ctree, nodes = c(7L, 53L))
## Variables actually used in tree construction:
## [1] "lead_time"                 "total_of_special_requests"
## [3] "consistent_room_type"      "previous_cancellations"
## [5] "customer_type"
## Number of terminal nodes:  6
## Residual mean deviance:  1.048 = 92780 / 88560
## Misclassification error rate: 0.2374 = 21028 / 88565
```

```
set.seed(10)
tree.pred<-predict(prune.tree,testHL,type = "class")
```

```
## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion
```

```
pred=tree.pred
actual=testHL$is_canceled
table(pred,actual)
```

```
##     actual
## pred     0     1
##    0 18061  3484
##    1  2772  5205
```

```
mean(pred!=actual)
```

```
## [1] 0.2119098
```

```
classificationTreePrunedResult=mean(pred!=actual)
```

The pruned tree has a validation misclassification rate of 0.2119098

```
mcrCTree<-function(idx){
  Train<-dfHL[-idx,-c(31,23)]
  Test<-dfHL[idx,-c(31,23)]
  fit<-tree(factor(is_canceled)~., Train)
  pred<-predict(fit,Test,type="class")
  return(1-mean(pred==Test$is_canceled))
}

mis_rate=lapply(folds,mcrCTree)
```

Stratified 10 fold cross validation misclassification rate:

```
## Warning in tree(factor(is_canceled) ~ ., Train): NAs introduced by coercion

## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion

## Warning in tree(factor(is_canceled) ~ ., Train): NAs introduced by coercion

## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion

## Warning in tree(factor(is_canceled) ~ ., Train): NAs introduced by coercion

## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion

## Warning in tree(factor(is_canceled) ~ ., Train): NAs introduced by coercion

## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion

## Warning in tree(factor(is_canceled) ~ ., Train): NAs introduced by coercion
```

```
## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion

## Warning in tree(factor(is_canceled) ~ ., Train): NAs introduced by coercion

## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion

## Warning in tree(factor(is_canceled) ~ ., Train): NAs introduced by coercion

## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion

## Warning in tree(factor(is_canceled) ~ ., Train): NAs introduced by coercion

## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion

## Warning in tree(factor(is_canceled) ~ ., Train): NAs introduced by coercion

## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion

## Warning in tree(factor(is_canceled) ~ ., Train): NAs introduced by coercion

## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion
```

```
classificationTreeCVResult=mean(as.numeric(mis_rate))
```

The stratified 10 fold cross validation misclassification rate is 0.2232253.

Random Forest

Given the number of categorical variables and non-normally distributed numerical variables in the dataset, an attempt at Random Forest learning as a method to solve the classification problem is approached.

```
dfDSG <- dfDSG[c('is_canceled', 'hotel', 'lead_time', 'arrival_date_year',
                 'stays_in_weekend_nights', 'stays_in_week_nights', 'adults',
                 'children', 'babies', 'meal', 'market_segment', 'is_repeated_guest',
                 'previous_cancellations', 'booking_changes', 'deposit_type',
                 'agent_yn', 'days_in_waiting_list', 'customer_type', 'adr',
                 'total_of_special_requests')]
```

```
set.seed(10)
rf <- randomForest(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_
                +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                  booking_changes+deposit_type+agent_yn+days_in_waiting_list+customer_type+adr+total_of_
                 data=dfDSG,
                 ntree=200,
                 proximity=FALSE)
rf
```

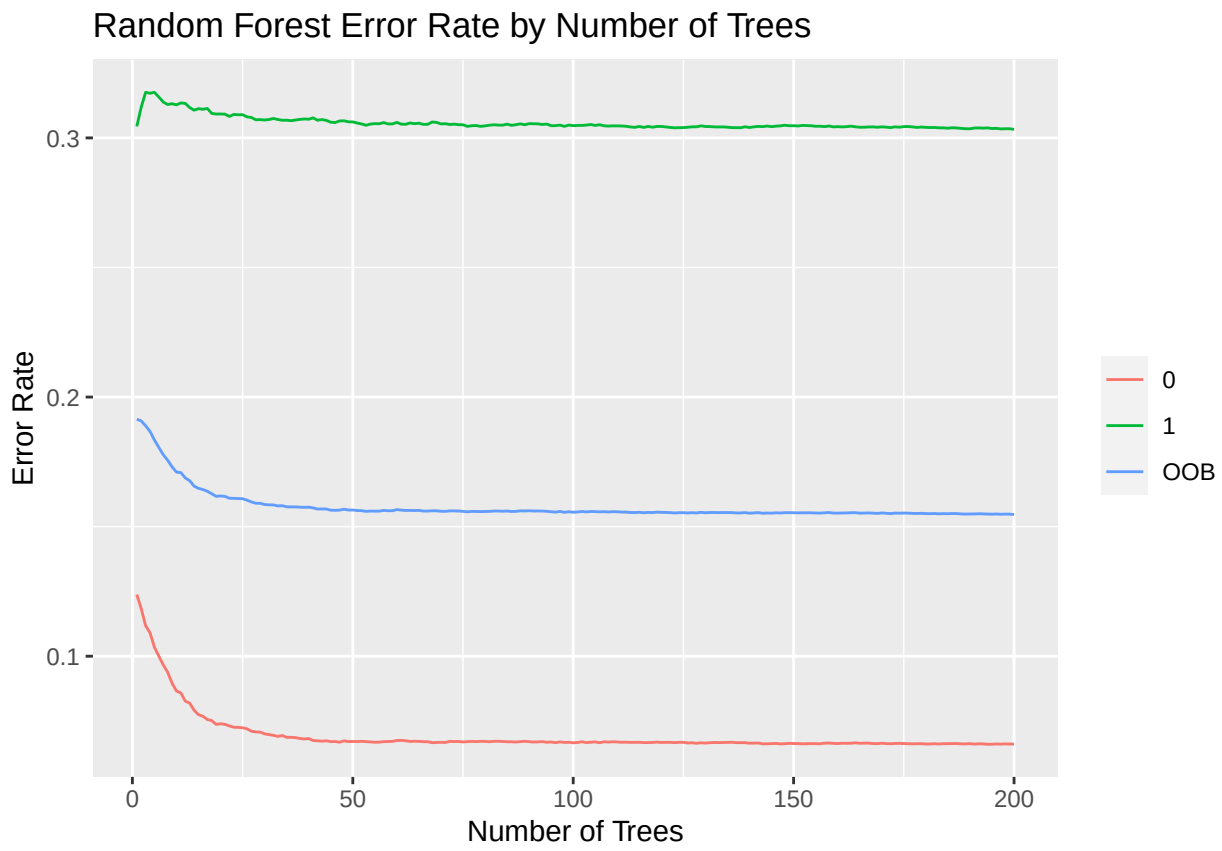
```
##
## Call:
##  randomForest(formula = factor(is_canceled) ~ hotel + lead_time +      arrival_date_year + stays_in_
##                Type of random forest: classification
##                      Number of trees: 200
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 15.47%
## Confusion matrix:
##       0     1 class.error
## 0 69085  4887  0.06606554
## 1 13380 30735  0.30329820
```

```
# Misclassification Rate
rf_va_misclss = 100*(rf$confusion[2,1]+rf$confusion[1,2])/(sum(rf$confusion[,2])+sum(rf$confusion[,1]))
rf_va_misclss
```

```
## [1] 15.4691
```

The misclassification rate for Random Forest through the standard approach is 15.47%.

```
# Plot Random Forest Error Rate
oob.error.data <- data.frame(
     num_trees=rep(1:nrow(rf$err.rate), times=3),
     type=rep(c("OOB", "0", "1"), each=nrow(rf$err.rate)),
     error=c(rf$err.rate[,"OOB"],
             rf$err.rate[,"0"],
             rf$err.rate[,"1"]))

ggplot(data=oob.error.data, aes(x=num_trees, y=error)) +
  geom_line(aes(color=type)) +
  ggtitle("Random Forest Error Rate by Number of Trees")+xlab("Number of Trees") +
  ylab("Error Rate") + theme(legend.title = element_blank())
```



It can be seen from the above plot that the error rate levels off at about n=50 trees.
There is also a significant gap between the error rate when predicting a cancellation compared to predicting a non-cancellation. There is some laziness in this model where it tends to have bias towards choosing non-cancellation.

```
set.seed(10)
n = 10

folds <- createFolds(factor(dfDSG$is_canceled), k=n)
trCtrl <- trainControl(index = folds, method = "cv", number = n)

rf_fit <- train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_
                +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                  booking_changes+deposit_type+agent_yn+days_in_waiting_list+customer_type+adr+total_of
                 data=dfDSG,
                 trControl=trCtrl,
                 method="rf", metric='Accuracy')

rf_fit
```

```
## Random Forest
##
## 118087 samples
##     19 predictor
##      2 classes: '0', '1'
```

```
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11808, 11809, 11808, 11809, 11809, 11809, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.7904219  0.5024965
##   15    0.8236150  0.6104951
##   29    0.8187786  0.6011158
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 15.
```

```
# Misclassification Rate
rf_cv_misclss = 100*(1-max(rf_fit$results[,2]))
rf_cv_misclss
```

```
## [1] 17.6385
```

The misclassification rate for Random Forest k-Fold Cross Validation at k=10 is 17.64%.

Linear Discriminant Analysis

To maintain consistency while analyzing models, the same variables used in the Logistic Regression Model are retained and the results are compared.

However, there is a problem as many of the variables in the dataset are categorical and therefore will not possess a normal distribution. Below is a breakdown of the variables and their type:

Independent Variable	Type
hotel	Categorical, Binary
lead_time	int
arrival_date_year	int (three 'classes')
stays_in_weekend_nights	int
stays_in_week_nights	int
adults	int
children	int
babies	int
meal	Categorical
market_segment	Categorical
is_repeated_guest	Categorical, Binary
previous_cancellations	int
booking_changes	int
deposit_type	Categorical
agent_yn	Categorical, Binary
days_in_waiting_list	int
customer_type	Categorical
adr	float
total_of_special_requests	int

The distributions for those numerical variables are explored and tested for their normality.

Unfortunately multivariate normality tests in R will not work with the size of the dataset. Each of the eligible individual variables will be tested for normality at both classes of 'is_cancelled'. The sample size is too large for the Shapiro-Wilk test, so the Anderson-Darling test is used instead.

H_o : The variable demonstrates a normal distribution.
H_a : The variable does not demonstrate a normal distribution.

```
numeric_col_vector <- colnames(dplyr::select_if(dfDSG, is.numeric))

for (i in numeric_col_vector){
  print(paste("Anderson Darling Test for Uncancelled Class, Variable:", i))
  print(ad.test(dfDSG[dfDSG$is_canceled == "0", i]))

  print(paste("Anderson Darling Test for Cancelled Class, Variable: ", i))
  print(ad.test(dfDSG[dfDSG$is_canceled == "1", i]))
}
```

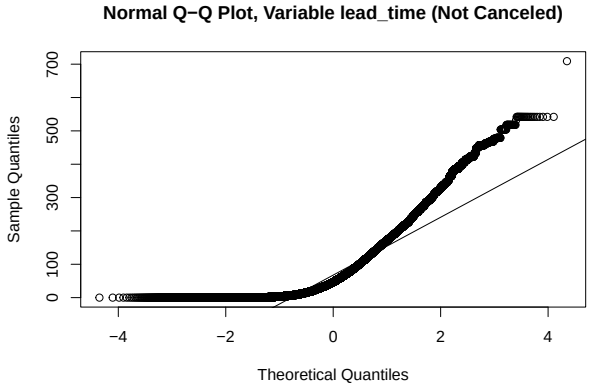
```
## [1] "Anderson Darling Test for Uncancelled Class, Variable: lead_time"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 3986.3, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  lead_time"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 1077.6, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: arrival_date_year"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 6323.8, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  arrival_date_year"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 3747.5, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: stays_in_weekend_nights"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
```

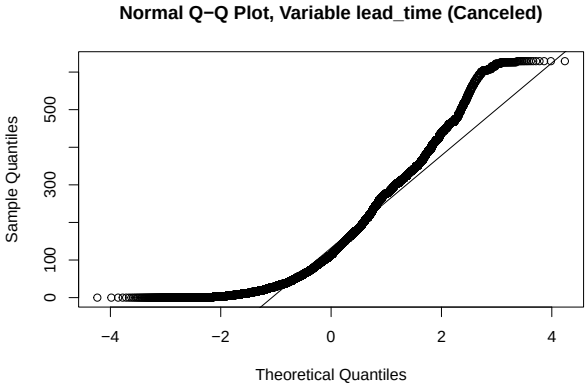
```
## A = 5356, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  stays_in_weekend_nights"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 3285.5, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: stays_in_week_nights"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 2728.1, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  stays_in_week_nights"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 2100.7, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: adults"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 12825, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  adults"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 8493.5, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: children"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 24429, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  children"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 14718, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: babies"
##
##  Anderson-Darling normality test
```

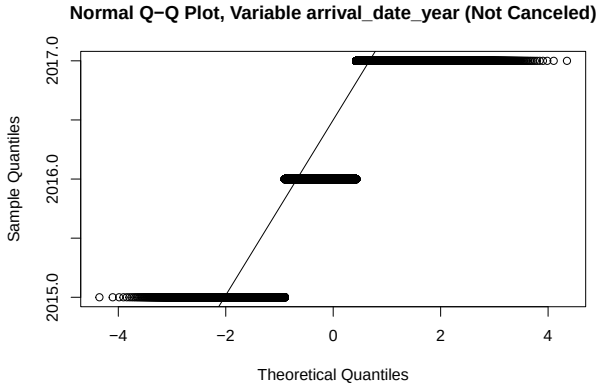
```
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 28141, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  babies"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 16981, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: is_repeated_guest"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 27157, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  is_repeated_guest"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 16825, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: previous_cancellations"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 28023, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  previous_cancellations"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 12666, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: booking_changes"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 15903, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  booking_changes"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 14425, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: days_in_waiting_list"
```

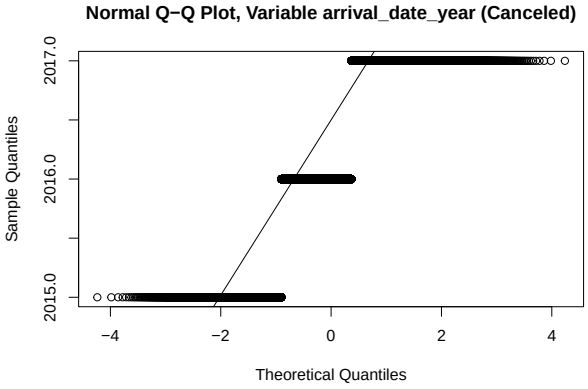
```
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 27406, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  days_in_waiting_list"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 14754, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: adr"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 923.82, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  adr"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 1081.2, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Uncancelled Class, Variable: total_of_special_requests"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "0", i]
## A = 6553.9, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Cancelled Class, Variable:  total_of_special_requests"
##
##  Anderson-Darling normality test
##
## data:  dfDSG[dfDSG$is_canceled == "1", i]
## A = 8919.2, p-value < 2.2e-16
```

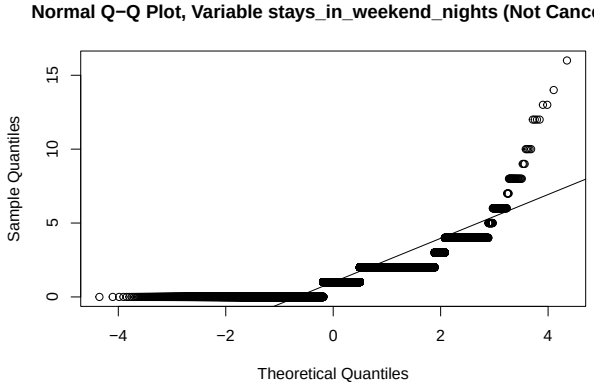
```r
for (i in numeric_col_vector){
  qqnorm(dfDSG[dfDSG$is_canceled == "0", i], main=paste("Normal Q-Q Plot, Variable", i, "(Not Canceled)"))
  qqline(dfDSG[dfDSG$is_canceled == "0", i])
  qqnorm(dfDSG[dfDSG$is_canceled == "1", i], main=paste("Normal Q-Q Plot, Variable", i, "(Canceled)"))
  qqline(dfDSG[dfDSG$is_canceled == "1", i])
}
```

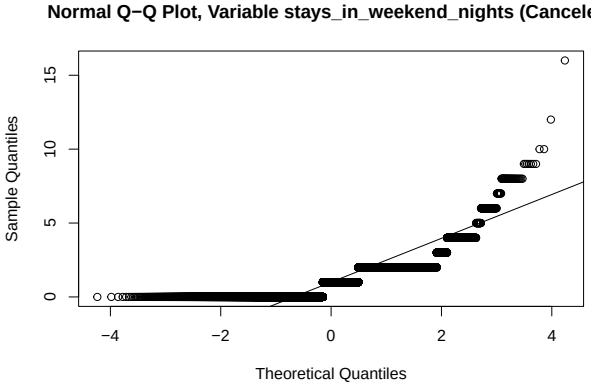

Normal Q−Q Plot, Variable lead_time (Not Canceled)
Sample Quantiles
Theoretical Quantiles
0
100
300
500
700
−4
−2
0
2
4

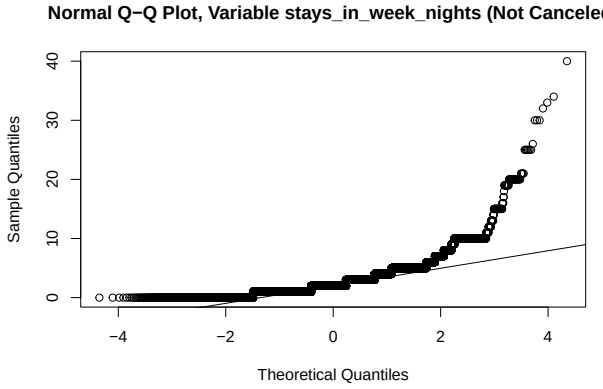

Normal Q−Q Plot, Variable lead_time (Canceled)
Sample Quantiles
Theoretical Quantiles
0
100
300
500
−4
−2
0
2
4

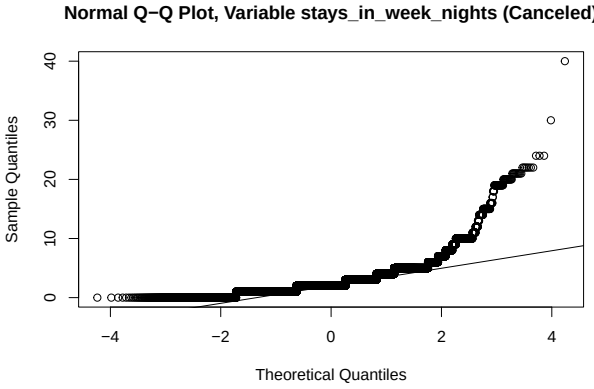

Normal Q−Q Plot, Variable arrival_date_year (Not Canceled)
Sample Quantiles
Theoretical Quantiles
2015.0
2016.0
2017.0
−4
−2
0
2
4

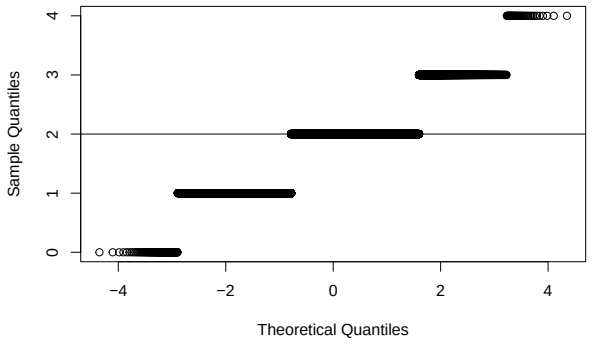

Normal Q−Q Plot, Variable arrival_date_year (Canceled)
Sample Quantiles
Theoretical Quantiles
2015.0
2016.0
2017.0
−4
−2
0
2
4

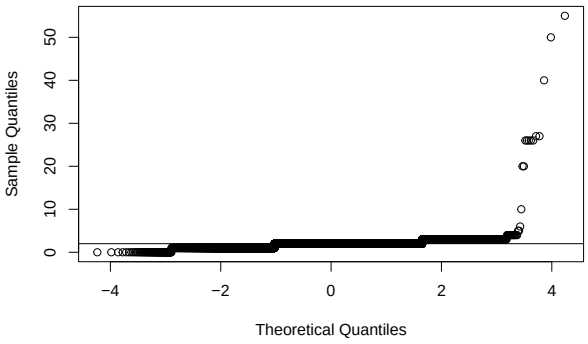

Normal Q−Q Plot, Variable stays_in_weekend_nights (Not Canceled
Sample Quantiles
Theoretical Quantiles
0
5
10
15
−4
−2
0
2
4

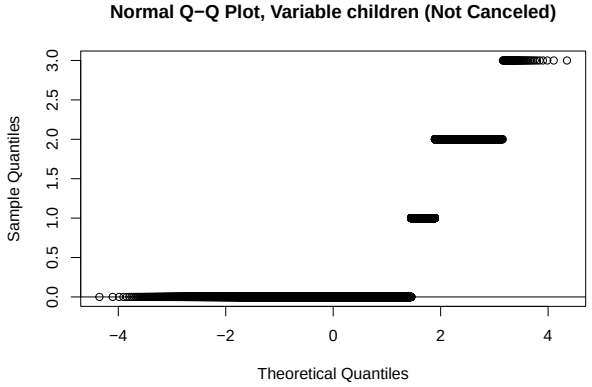

Normal Q−Q Plot, Variable stays_in_weekend_nights (Canceled)
Sample Quantiles
Theoretical Quantiles
0
5
10
15
−4
−2
0
2
4

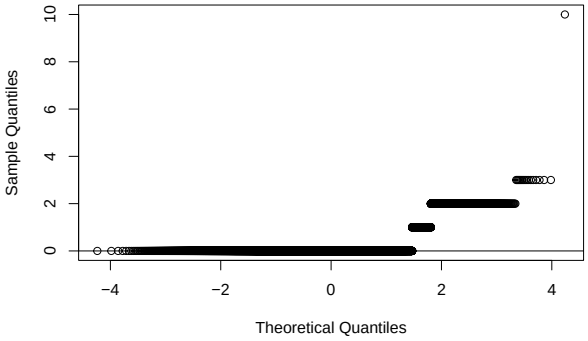

Normal Q−Q Plot, Variable stays_in_week_nights (Not Canceled)
Sample Quantiles
Theoretical Quantiles
0
10
20
30
40
−4
−2
0
2
4

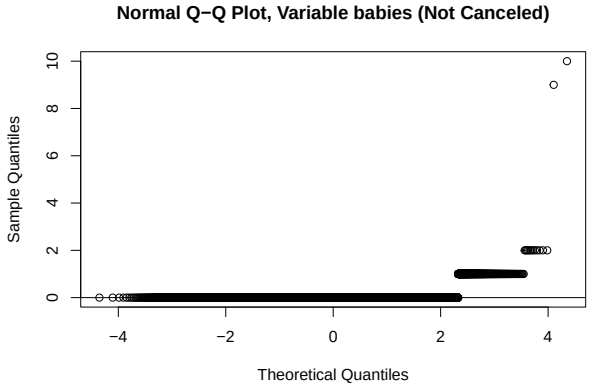

Normal Q−Q Plot, Variable stays_in_week_nights (Canceled)
Sample Quantiles
Theoretical Quantiles
0
10
20
30
40
−4
−2
0
2
4

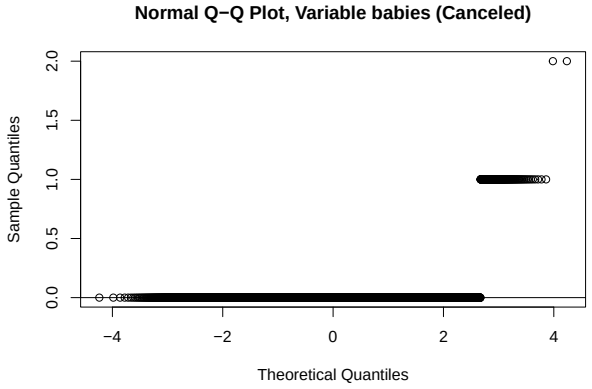

Normal Q−Q Plot, Variable adults (Not Canceled)
Sample Quantiles
Theoretical Quantiles
0
1
2
3
4
−4
−2
0
2
4

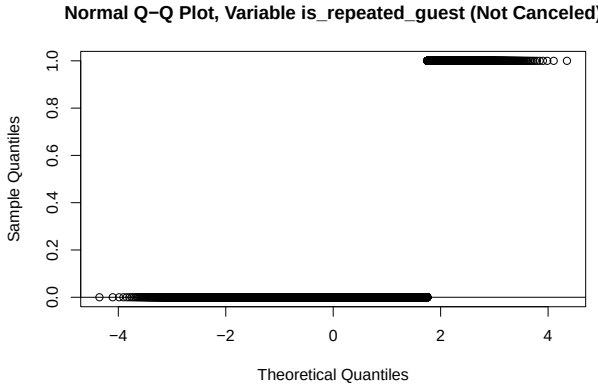

Normal Q−Q Plot, Variable adults (Canceled)
Sample Quantiles
Theoretical Quantiles
0
10
20
30
40
50
−4
−2
0
2
4

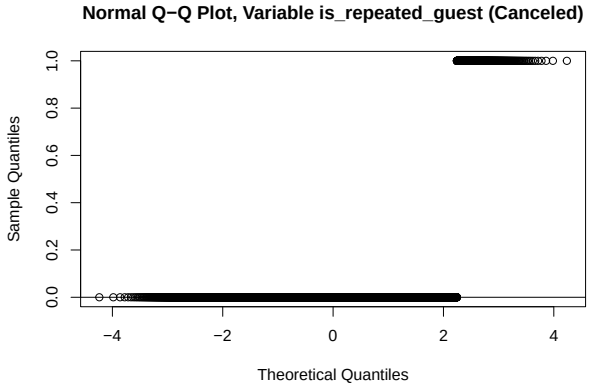

Normal Q−Q Plot, Variable children (Not Canceled)
Sample Quantiles
Theoretical Quantiles
0.0
0.5
1.0
1.5
2.0
2.5
3.0
−4
−2
0
2
4

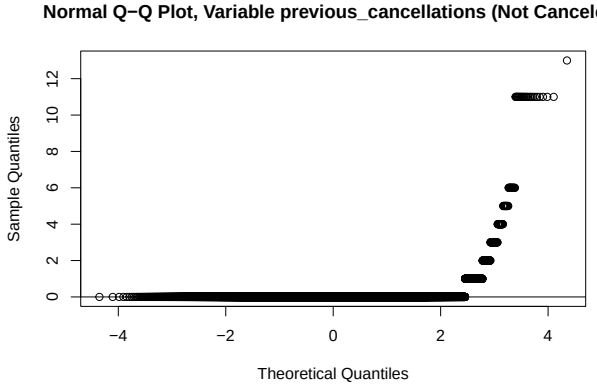

Normal Q−Q Plot, Variable children (Canceled)
Sample Quantiles
Theoretical Quantiles
0
2
4
6
8
10
−4
−2
0
2
4

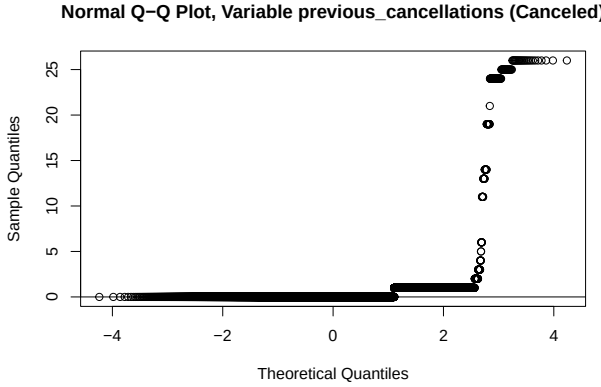

Normal Q−Q Plot, Variable babies (Not Canceled)
Sample Quantiles
Theoretical Quantiles
0
2
4
6
8
10
−4
−2
0
2
4

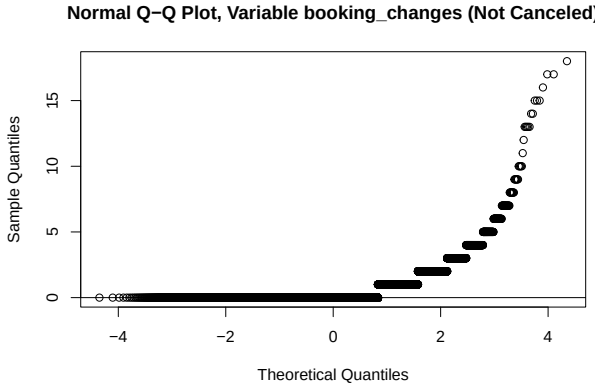

Normal Q−Q Plot, Variable babies (Canceled)
Sample Quantiles
Theoretical Quantiles
0.0
0.5
1.0
1.5
2.0
−4
−2
0
2
4

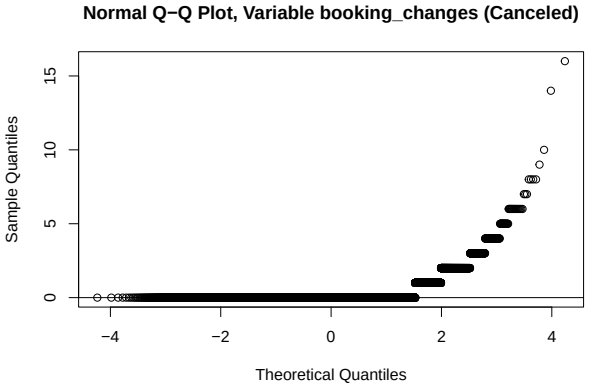

Normal Q−Q Plot, Variable is_repeated_guest (Not Canceled)
Sample Quantiles
Theoretical Quantiles
0.0
0.2
0.4
0.6
0.8
1.0
−4
−2
0
2
4

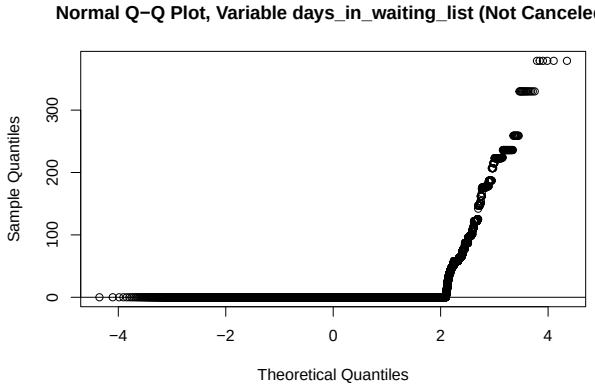

Normal Q−Q Plot, Variable is_repeated_guest (Canceled)
Sample Quantiles
Theoretical Quantiles
0.0
0.2
0.4
0.6
0.8
1.0
−4
−2
0
2
4

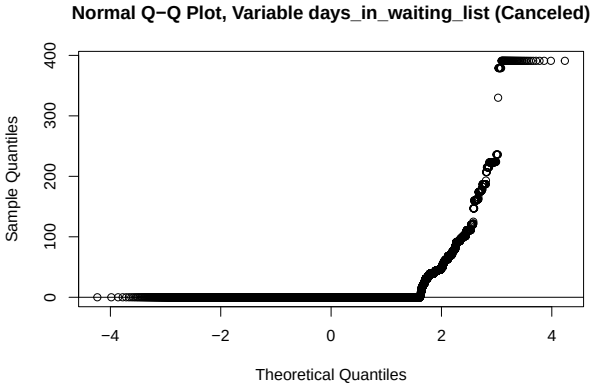

Normal Q−Q Plot, Variable previous_cancellations (Not Canceled)
Sample Quantiles
Theoretical Quantiles
0
2
4
6
8
10
12
−4
−2
0
2
4

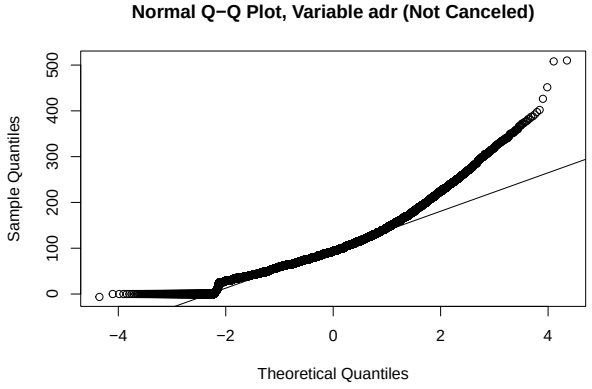

Normal Q−Q Plot, Variable previous_cancellations (Canceled)
Sample Quantiles
Theoretical Quantiles
0
5
10
15
20
25
−4
−2
0
2
4

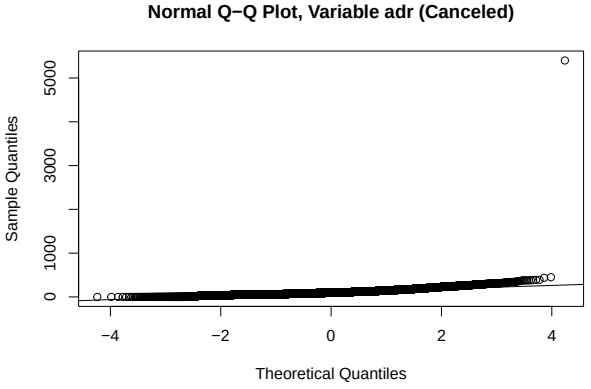

Normal Q−Q Plot, Variable booking_changes (Not Canceled)
Sample Quantiles
Theoretical Quantiles
0
5
10
15
−4
−2
0
2
4

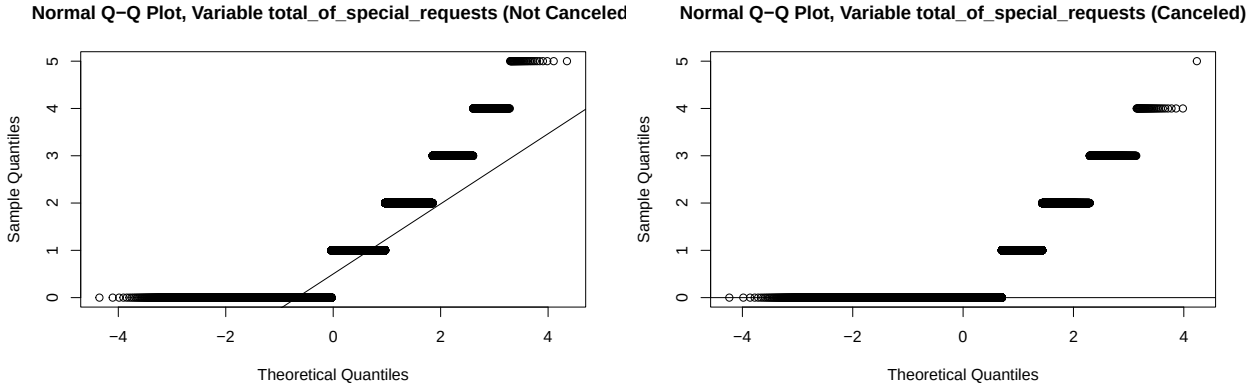

Normal Q−Q Plot, Variable booking_changes (Canceled)
Sample Quantiles
Theoretical Quantiles
0
5
10
15
−4
−2
0
2
4


Normal Q−Q Plot, Variable days_in_waiting_list (Not Canceled)
Sample Quantiles
Theoretical Quantiles
0
100
200
300
−4
−2
0
2
4


Normal Q−Q Plot, Variable days_in_waiting_list (Canceled)
Sample Quantiles
Theoretical Quantiles
0
100
200
300
400
−4
−2
0
2
4


Normal Q−Q Plot, Variable adr (Not Canceled)
Sample Quantiles
Theoretical Quantiles
0
100
200
300
400
500
−4
−2
0
2
4


Normal Q−Q Plot, Variable adr (Canceled)
Sample Quantiles
Theoretical Quantiles
0
1000
3000
5000
−4
−2
0
2
4



Based on the results of the Anderson-Darling test, the Null Hypothesis is rejected for each numerical variable in the logistic regression model. None of the QQPlots resemble a normal distribution.A log-transform is attempted where possible to see if that will generate a normal distribution.

```
numeric_col_vector <- colnames(dplyr::select_if(dfDSG, is.numeric))

for (i in numeric_col_vector){
  print(i)
  print(paste("% of Zeros in Variable: ", i, " For Uncancelled Class"))
  print(100*nrow(dfDSG[(dfDSG$is_canceled == "0") & (dfDSG[[i]] == 0), ])/nrow(dfDSG[(dfDSG$is_canceled

  print(paste("% of Zeros in Variable: ", i, " For Cancelled Class"))
  print(100*nrow(dfDSG[(dfDSG$is_canceled == "1") & (dfDSG[[i]] == 0), ])/nrow(dfDSG[(dfDSG$is_canceled

  print("")
}
```

```
## [1] "lead_time"
## [1] "% of Zeros in Variable:  lead_time  For Uncancelled Class"
## [1] 7.421727
## [1] "% of Zeros in Variable:  lead_time  For Cancelled Class"
## [1] 0.9248555
## [1] ""
## [1] "arrival_date_year"
## [1] "% of Zeros in Variable:  arrival_date_year  For Uncancelled Class"
## [1] 0
## [1] "% of Zeros in Variable:  arrival_date_year  For Cancelled Class"
## [1] 0
## [1] ""
## [1] "stays_in_weekend_nights"
## [1] "% of Zeros in Variable:  stays_in_weekend_nights  For Uncancelled Class"
## [1] 42.60396
## [1] "% of Zeros in Variable:  stays_in_weekend_nights  For Cancelled Class"
## [1] 44.04624
## [1] ""
## [1] "stays_in_week_nights"
## [1] "% of Zeros in Variable:  stays_in_week_nights  For Uncancelled Class"
## [1] 6.780944
## [1] "% of Zeros in Variable:  stays_in_week_nights  For Cancelled Class"
```

```
## [1] 4.245721
## [1] ""
## [1] "adults"
## [1] "% of Zeros in Variable:  adults  For Uncancelled Class"
## [1] 0.1879089
## [1] "% of Zeros in Variable:  adults  For Cancelled Class"
## [1] 0.1904114
## [1] ""
## [1] "children"
## [1] "% of Zeros in Variable:  children  For Uncancelled Class"
## [1] 92.67561
## [1] "% of Zeros in Variable:  children  For Cancelled Class"
## [1] 92.91397
## [1] ""
## [1] "babies"
## [1] "% of Zeros in Variable:  babies  For Uncancelled Class"
## [1] 98.99692
## [1] "% of Zeros in Variable:  babies  For Cancelled Class"
## [1] 99.62144
## [1] ""
## [1] "is_repeated_guest"
## [1] "% of Zeros in Variable:  is_repeated_guest  For Uncancelled Class"
## [1] 96.01606
## [1] "% of Zeros in Variable:  is_repeated_guest  For Cancelled Class"
## [1] 98.75553
## [1] ""
## [1] "previous_cancellations"
## [1] "% of Zeros in Variable:  previous_cancellations  For Uncancelled Class"
## [1] 99.29433
## [1] "% of Zeros in Variable:  previous_cancellations  For Cancelled Class"
## [1] 86.5896
## [1] ""
## [1] "booking_changes"
## [1] "% of Zeros in Variable:  booking_changes  For Uncancelled Class"
## [1] 79.78559
## [1] "% of Zeros in Variable:  booking_changes  For Cancelled Class"
## [1] 93.61215
## [1] ""
## [1] "days_in_waiting_list"
## [1] "% of Zeros in Variable:  days_in_waiting_list  For Uncancelled Class"
## [1] 98.19797
## [1] "% of Zeros in Variable:  days_in_waiting_list  For Cancelled Class"
## [1] 94.65261
## [1] ""
## [1] "adr"
## [1] "% of Zeros in Variable:  adr  For Uncancelled Class"
## [1] 1.343752
## [1] "% of Zeros in Variable:  adr  For Cancelled Class"
## [1] 0.3740224
## [1] ""
## [1] "total_of_special_requests"
## [1] "% of Zeros in Variable:  total_of_special_requests  For Uncancelled Class"
## [1] 48.68329
## [1] "% of Zeros in Variable:  total_of_special_requests  For Cancelled Class"
```

```
## [1] 75.88349
## [1] ""
```

The output above shows that there is a significant ($> 5\%$) number of zeros present in 9 of the 11 variables. A check for normality of the log transform of the two other variables, adr and adults is attempted.

H_o : The variable demonstrates a normal distribution.
H_a : The variable does not demonstrate a normal distribution.

```
for (i in c("adults","adr")){
  print(paste("Anderson Darling Test for Log-Transform Uncancelled Class, Variable:", i))
  print(ad.test(log(dfDSG[(dfDSG$is_canceled == "0")&(dfDSG[[i]]>0), i])))

  print(paste("Anderson Darling Test for Log-Transform Cancelled Class, Variable: ", i))
  print(ad.test(log(dfDSG[(dfDSG$is_canceled == "1")&(dfDSG[[i]]>0), i])))
  print(" ")
}
```

```
## [1] "Anderson Darling Test for Log-Transform Uncancelled Class, Variable: adults"
##
##  Anderson-Darling normality test
##
## data:  log(dfDSG[(dfDSG$is_canceled == "0") & (dfDSG[[i]] > 0), i])
## A = 14457, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Log-Transform Cancelled Class, Variable:  adults"
##
##  Anderson-Darling normality test
##
## data:  log(dfDSG[(dfDSG$is_canceled == "1") & (dfDSG[[i]] > 0), i])
## A = 10105, p-value < 2.2e-16
##
## [1] " "
## [1] "Anderson Darling Test for Log-Transform Uncancelled Class, Variable: adr"
##
##  Anderson-Darling normality test
##
## data:  log(dfDSG[(dfDSG$is_canceled == "0") & (dfDSG[[i]] > 0), i])
## A = 155.49, p-value < 2.2e-16
##
## [1] "Anderson Darling Test for Log-Transform Cancelled Class, Variable:  adr"
##
##  Anderson-Darling normality test
##
## data:  log(dfDSG[(dfDSG$is_canceled == "1") & (dfDSG[[i]] > 0), i])
## A = 113.23, p-value < 2.2e-16
##
## [1] " "
```

Given the result of above, none of the individual variables come from a normal distribution and therefore the assumption of multivariate normality fails.

To test the equality of variance, the Levene's Test is used.

H_o : The variable demonstrates equal variance across classes.
H_a : The variable does not demonstrate equal variance across classes.

```
for (i in numeric_col_vector){
  print(paste("Levene's Test for Variable:", i))
  print(leveneTest(dfDSG[,i]~is_canceled, dfDSG))
  print(" ")
}
```

```
## [1] "Levene's Test for Variable: lead_time"
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     1  3522.7 < 2.2e-16 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: arrival_date_year"
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     1  56.321 6.197e-14 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: stays_in_weekend_nights"
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     1  22.448 2.162e-06 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: stays_in_week_nights"
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     1  196.75 < 2.2e-16 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: adults"
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     1  373.12 < 2.2e-16 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: children"
## Levene's Test for Homogeneity of Variance (center = median)
```

```
##           Df F value Pr(>F)
## group      1  2.2657 0.1323
##       118085
## [1] " "
## [1] "Levene's Test for Variable: babies"
## Levene's Test for Homogeneity of Variance (center = median)
##           Df F value    Pr(>F)
## group      1  126.46 < 2.2e-16 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: is_repeated_guest"
## Levene's Test for Homogeneity of Variance (center = median)
##           Df F value    Pr(>F)
## group      1  726.31 < 2.2e-16 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: previous_cancellations"
## Levene's Test for Homogeneity of Variance (center = median)
##           Df F value    Pr(>F)
## group      1  1439.5 < 2.2e-16 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: booking_changes"
## Levene's Test for Homogeneity of Variance (center = median)
##           Df F value    Pr(>F)
## group      1  2553.8 < 2.2e-16 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: days_in_waiting_list"
## Levene's Test for Homogeneity of Variance (center = median)
##           Df F value    Pr(>F)
## group      1  342.16 < 2.2e-16 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: adr"
## Levene's Test for Homogeneity of Variance (center = median)
##           Df F value    Pr(>F)
## group      1   58.84 1.724e-14 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
## [1] "Levene's Test for Variable: total_of_special_requests"
## Levene's Test for Homogeneity of Variance (center = median)
```

```
##           Df F value    Pr(>F)
## group      1   10544 < 2.2e-16 ***
##       118085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] " "
```

From the results of the Levene Test it can be seen that only the variable 'children' satisfies the equal variance condition as it has a p-value greater than 0.05, where the null hypothesis fails to be rejected. The remaining variables do not demonstrate equal variance across classes given that their p-values are « 0.05, and the null hypothesis is rejected.

The Fligner-Killeen test is put into place to see if any different results are achieved.

H_o : The variable demonstrates equal variance across classes.
H_a : The variable does not demonstrate equal variance across classes.

```
for (i in numeric_col_vector){
  print(paste("Fligner-Killen Test for Variable:", i))
  print(fligner.test(dfDSG[,i]~is_canceled, dfDSG))
  print(" ")
}
```

```
## [1] "Fligner-Killen Test for Variable: lead_time"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 5114.3, df = 1, p-value < 2.2e-16
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: arrival_date_year"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 56.295, df = 1, p-value = 6.237e-14
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: stays_in_weekend_nights"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 11.888, df = 1, p-value = 0.000565
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: stays_in_week_nights"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
```

```
## Fligner-Killeen:med chi-squared = 417.71, df = 1, p-value < 2.2e-16
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: adults"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 667.6, df = 1, p-value < 2.2e-16
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: children"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 0.084404, df = 1, p-value = 0.7714
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: babies"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 141.33, df = 1, p-value < 2.2e-16
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: is_repeated_guest"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 721.87, df = 1, p-value < 2.2e-16
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: previous_cancellations"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 8016.4, df = 1, p-value < 2.2e-16
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: booking_changes"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 3604.8, df = 1, p-value < 2.2e-16
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: days_in_waiting_list"
##
##  Fligner-Killeen test of homogeneity of variances
```

```
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 976.95, df = 1, p-value < 2.2e-16
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: adr"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 89.594, df = 1, p-value < 2.2e-16
##
## [1] " "
## [1] "Fligner-Killen Test for Variable: total_of_special_requests"
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  dfDSG[, i] by is_canceled
## Fligner-Killeen:med chi-squared = 7358, df = 1, p-value < 2.2e-16
##
## [1] " "
```

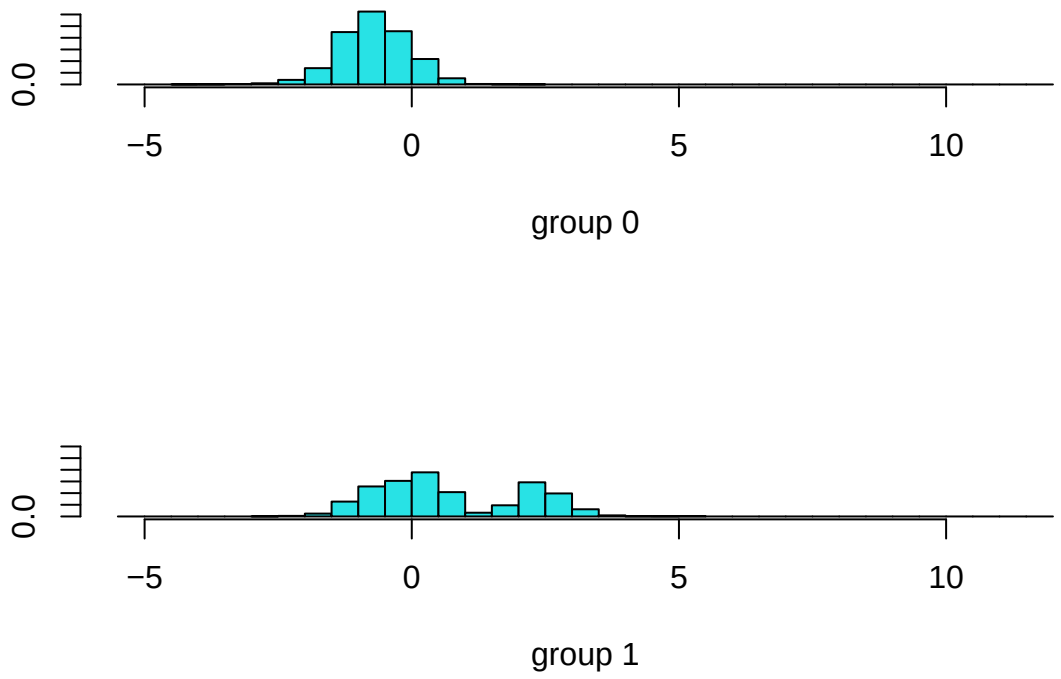
Again, it can be seen that the variable 'children' is the only numerical variable to have a p-value greater than 0.05, and thus a failure to reject the null hypothesis. This forces the rejection of null hypothesis with the other variables.

Given that the variable 'children' also did not pass the test previous for normality, there are no other variables from the Logistic Regression Model which are suitable to use with a Linear Discriminant Analysis model.

In any event, the creation and evaluation of the model to compare with the other results in the knowledge that both assumptions have failed and the inferences which can be made from both the LDA and QDA models are at best flawed.

```
set.seed(10)
NC = nrow(dfDSG[dfDSG$is_canceled == "0", ]);
C = nrow(dfDSG[dfDSG$is_canceled == "1", ]);

Nh = c(NC, C)
N = nrow(dfDSG)
n = round(0.75*N)

idx = sampling::strata(dfDSG, stratanames=c("is_canceled"), size=round(n*round(Nh/N, 1)), method="srsw
trainDSG = dfDSG[idx$ID_unit, ]
testDSG = dfDSG[-idx$ID_unit, ]

lda.fit <- lda(factor(is_canceled)~hotel+lead_time+arrival_date_year+
                 stays_in_weekend_nights+stays_in_week_nights+adults+children+
                 babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                 booking_changes+deposit_type+agent_yn+days_in_waiting_list+customer_type+
                 adr+total_of_special_requests,
               data=trainDSG)
```

```
plot(lda.fit)
```




```
print(lda.fit)
```

```
## Call:
## lda(factor(is_canceled) ~ hotel + lead_time + arrival_date_year +
##     stays_in_weekend_nights + stays_in_week_nights + adults +
##     children + babies + meal + market_segment + is_repeated_guest +
##     previous_cancellations + booking_changes + deposit_type +
##     agent_yn + days_in_waiting_list + customer_type + adr + total_of_special_requests,
##     data = trainDSG)
##
## Prior probabilities of groups:
##   0   1
## 0.6 0.4
##
## Group means:
##   hotelResort Hotel lead_time arrival_date_year stays_in_weekend_nights
## 0         0.3806620  80.49583          2016.151               0.9343420
## 1         0.2500988 144.84822          2016.172               0.9230509
##   stays_in_week_nights   adults  children      babies      mealFB    mealHB
## 0             2.481981 1.836354 0.1029376 0.010199665 0.004196541 0.1260280
## 1             2.554847 1.903404 0.1057698 0.004121267 0.010783041 0.1123186
##       mealSC market_segmentComplementary market_segmentCorporate
```

```
## 0 0.10032180                0.008223715              0.0558347
## 1 0.09690623                0.001863038              0.0217919
##   market_segmentDirect market_segmentGroups market_segmentOffline TA/TO
## 0           0.14070645            0.1030693                  0.2118783
## 1           0.04293457            0.2755039                  0.1882798
##   market_segmentOnline TA is_repeated_guest previous_cancellations
## 0               0.4779729        0.03989537             0.01620279
## 1               0.4683848        0.01222266             0.20561170
##   booking_changes deposit_typeNon Refund deposit_typeRefundable agent_ynYes
## 0       0.2924970            0.001373756           0.0016560342   0.8398163
## 1       0.0984023            0.329701349           0.0007339242   0.9099531
##   days_in_waiting_list customer_typeGroup customer_typeTransient
## 0             1.630535        0.006906415              0.7037204
## 1             3.539999        0.001326709              0.8269068
##   customer_typeTransient-Party      adr total_of_special_requests
## 0                    0.2516984 101.2779                 0.7187565
## 1                    0.1434822 104.9969                 0.3279230
##
## Coefficients of linear discriminants:
##                                       LD1
## hotelResort Hotel            -0.178302573
## lead_time                     0.002767396
## arrival_date_year             0.012555857
## stays_in_weekend_nights       0.054171632
## stays_in_week_nights          0.049355663
## adults                        0.092609370
## children                      0.117837747
## babies                        0.271160373
## mealFB                        0.335048583
## mealHB                       -0.032661365
## mealSC                        0.056098314
## market_segmentComplementary   0.154462503
## market_segmentCorporate      -0.162789846
## market_segmentDirect         -0.272107796
## market_segmentGroups         -0.060047035
## market_segmentOffline TA/TO  -0.389372007
## market_segmentOnline TA       0.643209091
## is_repeated_guest            -0.006834758
## previous_cancellations        0.105522254
## booking_changes              -0.262883089
## deposit_typeNon Refund        2.611078781
## deposit_typeRefundable       -0.180072176
## agent_ynYes                   0.070602105
## days_in_waiting_list         -0.000909805
## customer_typeGroup           -0.150782494
## customer_typeTransient        0.353659668
## customer_typeTransient-Party  0.134742611
## adr                           0.001900326
## total_of_special_requests    -0.558664020
```

Interpretation of results:

1) According to the training data, 60% of bookings ended up being canceled.

2) For each co-efficient used, the prediction tendency based on the training data is noted below:

Note that the categorical variables have not been included due to their difficulty in being interpreted given that their base factor is combined in the constant.

Independent Variable	Tendency
lead_time	As $\to \infty$, more likely to be *cancelled*
stays_in_weekend_nights	As $\to \infty$, more likely to be *not be cancelled*
stays_in_week_nights	As $\to \infty$, more likely to be *cancelled*
adults	As $\to \infty$, more likely to be *cancelled*
children	As $\to \infty$, more likely to be *cancelled*
babies	As $\to \infty$, more likely to be *not be cancelled*
is_repeated_guest	More likely to *not be cancelled*
previous_cancellations	As $\to \infty$, more likely to be *cancelled*
booking_changes	As $\to \infty$, more likely to be *not be cancelled*
days_in_waiting_list	As $\to \infty$, more likely to be *cancelled*
adr	As $\to \infty$, more likely to be *cancelled*
total_of_special_requests	As $\to \infty$, more likely to be *not be cancelled*

```
lda.pred = predict(lda.fit, testDSG)
caret::confusionMatrix(lda.pred$class,testDSG$is_canceled)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 19575  4069
##          1  1258  4620
##
##                Accuracy : 0.8196
##                  95% CI : (0.8151, 0.8239)
##     No Information Rate : 0.7057
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5204
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9396
##             Specificity : 0.5317
##          Pos Pred Value : 0.8279
##          Neg Pred Value : 0.7860
##              Prevalence : 0.7057
##          Detection Rate : 0.6631
##    Detection Prevalence : 0.8009
##       Balanced Accuracy : 0.7357
##
##        'Positive' Class : 0
##
```

```r
# Misclassification Rate
lda_va_misclss = 100*(1-as.numeric(caret::confusionMatrix(lda.pred$class,testDSG$is_canceled)$overall[1]
lda_va_misclss
```

```
## [1] 18.04417
```

The misclassification rate for LDA using the validation approach with a 75/25 split is found to be 18.04%.

```r
n = 10

folds <- createFolds(factor(dfDSG$is_canceled), k=n)
trCtrl <- trainControl(index = folds, method = "cv", number = n)

set.seed(10)
lda_cv <- train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_w
                +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                  booking_changes+deposit_type+agent_yn+days_in_waiting_list+customer_type+adr+total_of
                data=dfDSG,
                trControl=trCtrl,
                method="lda")

lda_cv
```

```
## Linear Discriminant Analysis
##
## 118087 samples
##     19 predictor
##      2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11809, 11809, 11810, 11809, 11809, 11808, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7814916  0.4877399
```

```r
# Misclassification Rate
lda_cv_misclss = 100*(1-as.numeric(lda_cv$results[2]))
```

The misclassification rate under k-fold Cross Validation at (k=10) for the Linear Discriminant Analysis is 21.85%.

Quadratic Discriminant Analysis

```r
qda.fit <- qda(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_w
                +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                  booking_changes+deposit_type+agent_yn+days_in_waiting_list+customer_type+adr+total_of
```

```
              data=trainDSG)

qda.class<-predict(qda.fit, testDSG)$class
caret::confusionMatrix(qda.class,
                       testDSG$is_canceled)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 17473  3142
##          1  3360  5547
##
##                Accuracy : 0.7798
##                  95% CI : (0.775, 0.7845)
##     No Information Rate : 0.7057
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4736
##
##  Mcnemar's Test P-Value : 0.007121
##
##             Sensitivity : 0.8387
##             Specificity : 0.6384
##          Pos Pred Value : 0.8476
##          Neg Pred Value : 0.6228
##              Prevalence : 0.7057
##          Detection Rate : 0.5919
##    Detection Prevalence : 0.6983
##       Balanced Accuracy : 0.7386
##
##        'Positive' Class : 0
##
```

```
# Misclassification Rate
qda_va_misclss = 100*(1-as.numeric(caret::confusionMatrix(qda.class,testDSG$is_canceled)$overall[1]))
qda_va_misclss
```

```
## [1] 22.02425
```

The misclassification rate for QDA using the validation approach with a 75/25 split is found to be 22.02%.

```
# Use folds from previous
set.seed(10)

qda_cv <- train(factor(is_canceled)~hotel+lead_time+arrival_date_year+stays_in_weekend_nights+stays_in_
                +adults+children+babies+meal+market_segment+is_repeated_guest+previous_cancellations+
                  booking_changes+deposit_type+agent_yn+days_in_waiting_list+customer_type+adr+total_of
              data=dfDSG,
              trControl=trCtrl,
              method="qda")
```

```
## Warning: model fit failed for Fold06: parameter=none Error in qda.default(x, grouping, ...) : rank d

## Warning: model fit failed for Fold10: parameter=none Error in qda.default(x, grouping, ...) : rank d

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```
qda_cv
```

```
## Quadratic Discriminant Analysis
##
## 118087 samples
##     19 predictor
##      2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11809, 11809, 11810, 11809, 11809, 11808, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7649178  0.4796207
```

```
# Misclassification Rate
qda_cv_misclss = 100*(1-as.numeric(qda_cv$results[2]))
qda_cv_misclss
```

```
## [1] 23.50822
```

The misclassification rate under k-fold Cross Validation at (k=10) for the Quadratic Discriminant Analysis is 23.51%.

Research Question 3: Using regression tree, which variables are important in predicting average daily rate of a hotel stay in Portugal?

The intent of this question is to determine which variables are important in predicting average daily rate of a hotel stay in Portugal. Specifically, which variables affect the price paid per day to stay at a hotel. Since the response variable, "average daily rate", is a continuous variable, the learning of regression tree is applied here, and compared to the results of both the pruned and unpruned tree. The learning of validation set approach is applied for cross validation, planning to use 75% of data for training, 25% for testing.

Starting off, some additional wrangling on variables is performed to be used in this analysis.

```
# convert date values from string to date type
dfDZ$reservation_status_date = as.Date(dfDZ$reservation_status_date, "%Y-%m-%d")

# factor all vategorical variables
dfDZ$hotel = factor(dfDZ$hotel)
dfDZ$arrival_date_month = factor(dfDZ$arrival_date_month)
dfDZ$meal = factor(dfDZ$meal)
dfDZ$country = factor(dfDZ$country)
dfDZ$market_segment = factor(dfDZ$market_segment)
dfDZ$distribution_channel = factor(dfDZ$distribution_channel)
dfDZ$is_repeated_guest = factor(dfDZ$is_repeated_guest)
dfDZ$reserved_room_type = factor(dfDZ$reserved_room_type)
dfDZ$assigned_room_type = factor(dfDZ$assigned_room_type)
dfDZ$deposit_type = factor(dfDZ$deposit_type)
dfDZ$agent = factor(dfDZ$agent)
dfDZ$company = factor(dfDZ$company)
dfDZ$customer_type = factor(dfDZ$customer_type)
dfDZ$reservation_status = factor(dfDZ$reservation_status)

dfDZ$consistent_room_type = factor(dfDZ$consistent_room_type)
dfDZ$arrival_date_season = factor(dfDZ$arrival_date_season)
```

The sample function is used to split out sets to be used for cross validation (75% training, 25% testing).

A regression tree is used using the tree() function with "average daily rate" as the response. The predictors are all other variable that are either quantitative or categorical that has less than 32 levels (due to constraints with tree() function, categorical variables with more than 32 levels are unable to be used).

```
# using regression tree to predict ADR
set.seed(10)
# split dataset: 75% training, 25% testing
idx=sample(1:nrow(dfDZ),3/4*nrow(dfDZ))
train=dfDZ[idx,]
test=dfDZ[-idx,]
# tree with every variable that has a factor of less than 32
# because we cant have more than 32 levels for factor for the tree function
# so the variables taken out are: country, agent, company, reservation status date
tree.adr<-tree(adr~hotel+is_canceled+lead_time+arrival_date_year+arrival_date_month
               +arrival_date_week_number+arrival_date_day_of_month
               +stays_in_weekend_nights+stays_in_week_nights+adults+children+babies
               +meal+market_segment+distribution_channel+is_repeated_guest
               +previous_cancellations+previous_bookings_not_canceled
               +reserved_room_type+assigned_room_type+booking_changes+deposit_type
```

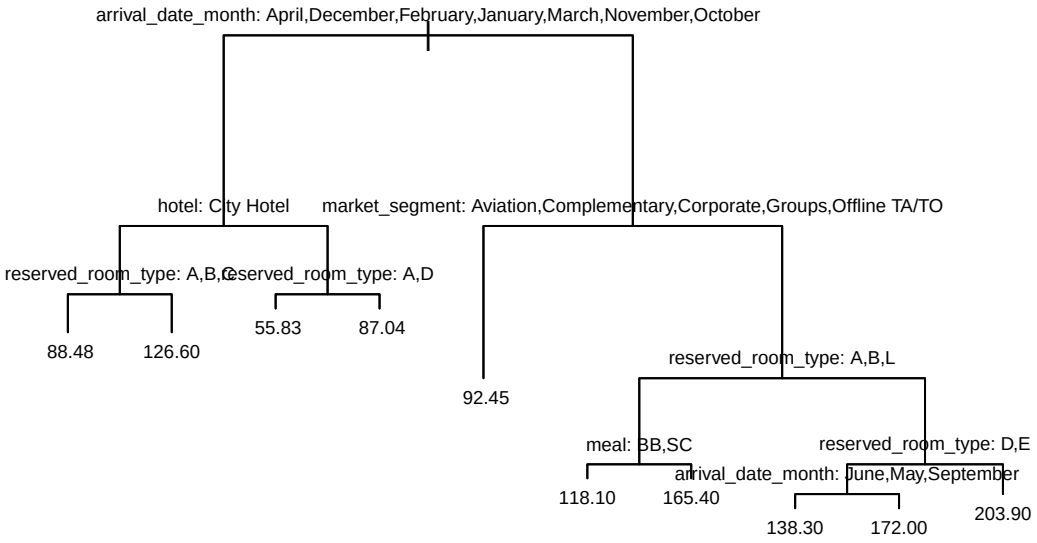
```
               +days_in_waiting_list+customer_type
               +required_car_parking_spaces+total_of_special_requests
               +reservation_status+consistent_room_type+arrival_date_season
               , train)
summary(tree.adr)
```

```
##
## Regression tree:
## tree(formula = adr ~ hotel + is_canceled + lead_time + arrival_date_year +
##     arrival_date_month + arrival_date_week_number + arrival_date_day_of_month +
##     stays_in_weekend_nights + stays_in_week_nights + adults +
##     children + babies + meal + market_segment + distribution_channel +
##     is_repeated_guest + previous_cancellations + previous_bookings_not_canceled +
##     reserved_room_type + assigned_room_type + booking_changes +
##     deposit_type + days_in_waiting_list + customer_type + required_car_parking_spaces +
##     total_of_special_requests + reservation_status + consistent_room_type +
##     arrival_date_season, data = train)
## Variables actually used in tree construction:
## [1] "arrival_date_month" "hotel"              "reserved_room_type"
## [4] "market_segment"     "meal"
## Number of terminal nodes:  10
## Residual mean deviance:  1448 = 128200000 / 88560
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -203.900  -20.480   -2.484    0.000   17.550 5312.000
```

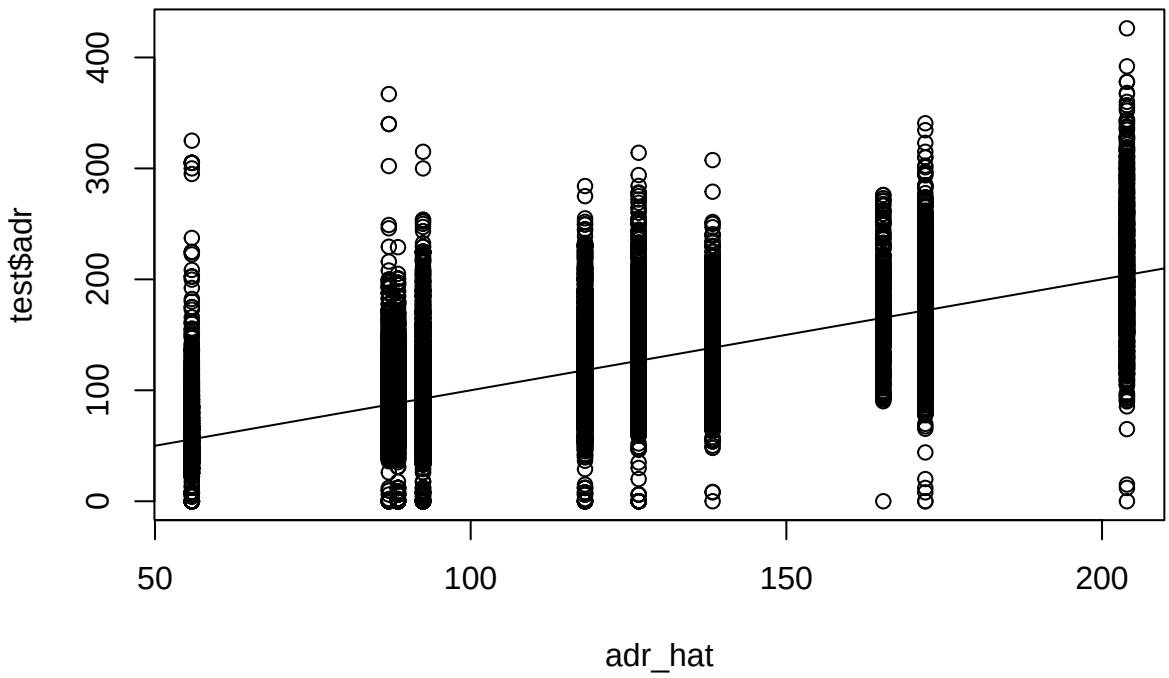
[1] “hotel” “is_canceled” “lead_time” “arrival_date_year”
[5] “arrival_date_month” “arrival_date_week_number” “arrival_date_day_of_month” “stays_in_weekend_nights”
[9] “stays_in_week_nights” “adults” “children” “babies”
[13] “meal” “country” “market_segment” “distribution_channel”
[17] “is_repeated_guest” “previous_cancellations” “previous_bookings_not_canceled” “reserved_room_type”
[21] “assigned_room_type” “booking_changes” “deposit_type” “agent”
[25] “company” “days_in_waiting_list” “customer_type” “adr”
[29] “required_car_parking_spaces” “total_of_special_requests” “reservation_status” “reservation_status_date”

Plot the tree and check the cross-validation error.

```
# plot tree
plot(tree.adr)
text(tree.adr, cex = 0.55, pretty=0)
```




```
# Apply the tree to the test set, calculate RMSE
adr_hat<-predict(tree.adr,test)
plot(adr_hat,test$adr)
abline(0,1)
```




```
sqrt(mean((adr_hat-test$adr)^2)) #the mean squared error
```

```
## [1] 33.89481
```

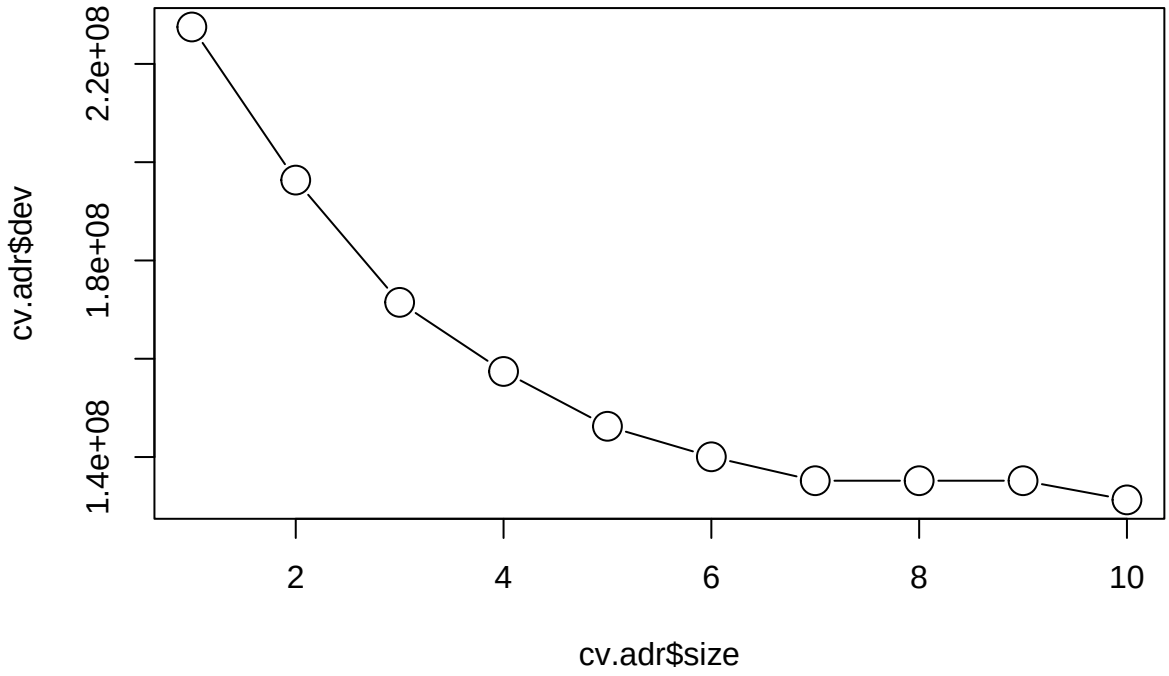
The un-pruned tree cross-validation error is: RMSE = €33.89, This is interpreted as the standard deviation of the unexplained variance by the model.

To answer the research question of which variables are important in predicting average daily rate of a hotel stay in Portugal, the tree shows that these are the 4 variables identified in our tree:

1. Arrival Month - Warmer months (May to September, especially July to August) lead to higher average daily rate.
2. Hotel - Hotel in resort region is cheaper than hotel in the city in colder months.
3. Market Segment - Market segment pays a major role in pricing, with Direct Booking and Online Travel Agents resulting in more expensive rates as compared to corporate and groups.
4. Meal Plan - Meal plans contributes to higher prices, with no meal/breakfast cheaper than Half/Full Board
5. Reserved Room Type - This variable is coded for privacy reasons, so we can not make any conclusions here, but we can conclude that certain room types are cheaper than others. A possible educated guess is that the smoking rooms / standard rooms are cheaper comparatively, but we have no evidence to back that up due to the data censoring in the original dataset.

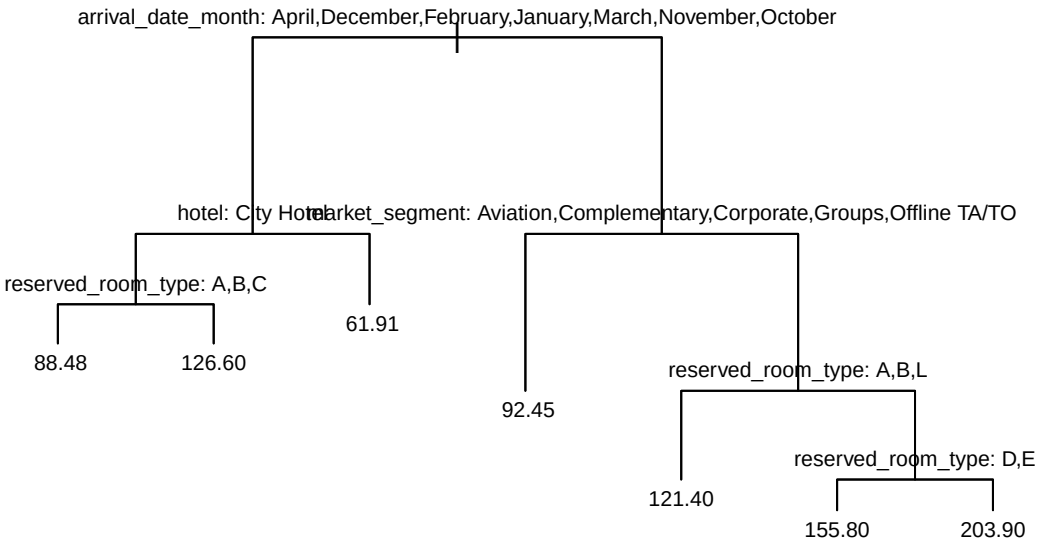
Next, the pruning of the tree is explored. first CV error vs size of tree is plotted.

```
# tree pruning based on cross validation error

# check cross-valiation error and the size of the tree,
# select the "best" number of terminal nodes
cv.adr=cv.tree(tree.adr)
plot(cv.adr$size,cv.adr$dev,type='b', cex = 2)
```

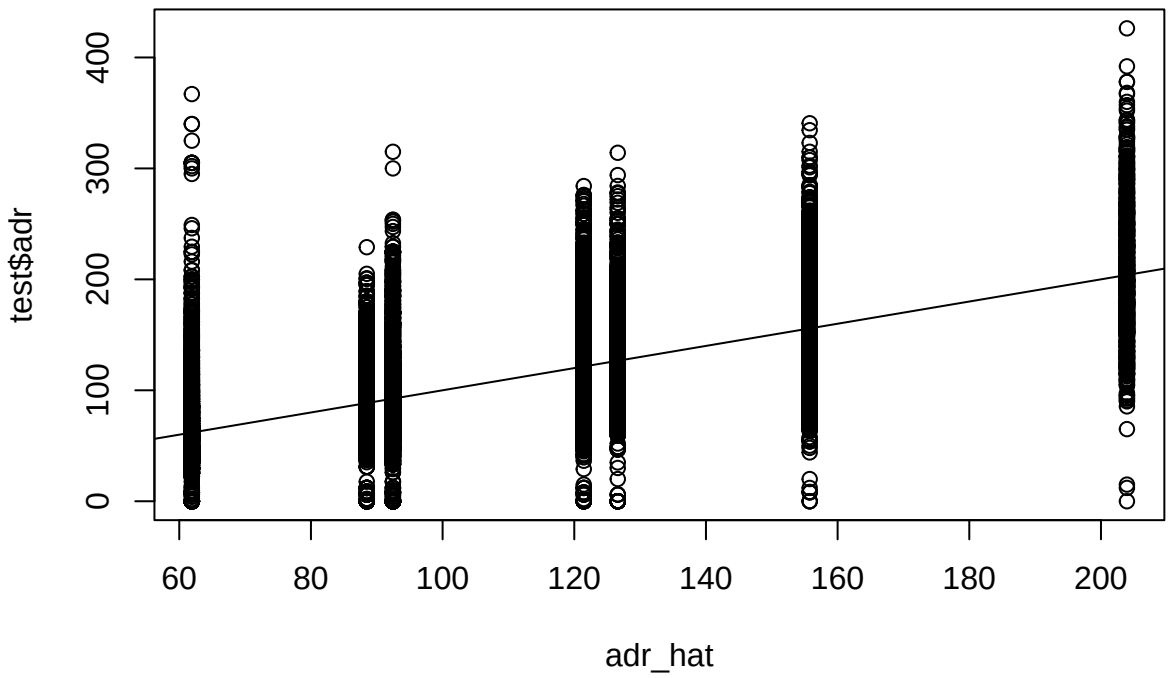


It can be seen that the cross-validation error is smallest at 10 terminal nodes, but improvement seemed to plateau at 7 terminal nodes. Thus, the tree will be pruned to 7 terminal nodes for a simpler model.

```
# pick 7 terminal nodes, go ahead and prunned tree
prune.adr=prune.tree(tree.adr,best=7)
plot(prune.adr)
text(prune.adr, cex = 0.7, pretty=0)
```




```
# Apply the pruned tree to the test set, calculate RMSE
adr_hat<-predict(prune.adr,test)
plot(adr_hat,test$adr)
abline(0,1)
```




```
sqrt(mean((adr_hat-test$adr)^2))
```

```
## [1] 34.96272
```

The pruned tree cross-validation error is: RMSE = €34.96 (vs. the €33.89 of the un-pruned tree), slightly higher than the un-pruned tree

By comparing the variables and branches to the un-pruned tree's, it can be seen that the pruned tree does not make use of the 'meal' (aka. meal plan) feature.

For a comparison, a simple first order multiple linear regression model is built using methods learned in DATA603, using a validation set approach for cross validation.

Here are the steps taken to build this simple first order multiple linear regression model:

1. Start with a full model with all variables, do ANOVA / f-test to check that at least one variable is significant.
2. Remove Aliased variables, in this case the variables we take out are 'arrival_date_season, which is an alias of 'arrival_date_month', and 'is_canceled', which is an alias of 'reservation_status'. Do ANOVA / f-test to check no significant variables are removed.
3. Use Stepwise regression (P_ent = 0.05, P_rem = 0.1) to reduce the full model, do ANOVA / f-test to check no significant variables are removed.
4. Check for multicollinearity and finalize model.

5. Check model assumptions, in our case they are linearity (using Residual vs Fitted), homoscedasticity (using Breusch-Pagan test), and normality (using Anderson-Darling test). Please see code comments below for our full write up of our Null Hypotheses and Alternative Hypotheses we used for these assumption checks.

Please note that many of the summary() outputs as well as some other functions such as alias() are commented out in order to cut down on the pages. The outputs are several pages long each due to the high number of factored variables in our model.

```
# buiild linear regression model

# full model
fullmodel<-lm(adr~hotel+is_canceled+lead_time+arrival_date_year+arrival_date_month
        +arrival_date_week_number+arrival_date_day_of_month
        +stays_in_weekend_nights+stays_in_week_nights+adults+children+babies
        +meal+market_segment+distribution_channel+is_repeated_guest
        +previous_cancellations+previous_bookings_not_canceled
        +reserved_room_type+assigned_room_type+booking_changes+deposit_type
        +days_in_waiting_list+customer_type
        +required_car_parking_spaces+total_of_special_requests
        +reservation_status+consistent_room_type+arrival_date_season
        , data=train)
nomodel<-lm(adr~1, data = train)
# summary(fullmodel)

# check to see if at least one variable is significant
anova(nomodel, fullmodel)
```

```
## Analysis of Variance Table
##
## Model 1: adr ~ 1
## Model 2: adr ~ hotel + is_canceled + lead_time + arrival_date_year + arrival_date_month +
##     arrival_date_week_number + arrival_date_day_of_month + stays_in_weekend_nights +
##     stays_in_week_nights + adults + children + babies + meal +
##     market_segment + distribution_channel + is_repeated_guest +
##     previous_cancellations + previous_bookings_not_canceled +
##     reserved_room_type + assigned_room_type + booking_changes +
##     deposit_type + days_in_waiting_list + customer_type + required_car_parking_spaces +
##     total_of_special_requests + reservation_status + consistent_room_type +
##     arrival_date_season
##   Res.Df       RSS Df Sum of Sq      F    Pr(>F)
## 1  88564 227509474
## 2  88498 106389245 66 121120228 1526.5 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# check for aliases
# alias(fullmodel)

# results not shown to save pages, as there were a lot of factored variables,
# but we reduce model based on alias result below:
# removed aliased variables "arrival_date_season"(an alias of arrival_date_month)
# removed aliased variables "is_canceled"(an alias of reservation_status)
```

```
reducedmodel<-lm(adr~hotel+lead_time+arrival_date_year+arrival_date_month
        +arrival_date_week_number+arrival_date_day_of_month
        +stays_in_weekend_nights+stays_in_week_nights+adults+children+babies
        +meal+market_segment+distribution_channel+is_repeated_guest
        +previous_cancellations+previous_bookings_not_canceled
        +reserved_room_type+assigned_room_type+booking_changes+deposit_type
        +days_in_waiting_list+customer_type
        +required_car_parking_spaces+total_of_special_requests
        +reservation_status+consistent_room_type
        , data=train)
# summary(reducedmodel)

# check to make sure no significant variables are removed
anova(reducedmodel, fullmodel)
```

```
## Analysis of Variance Table
##
## Model 1: adr ~ hotel + lead_time + arrival_date_year + arrival_date_month +
##     arrival_date_week_number + arrival_date_day_of_month + stays_in_weekend_nights +
##     stays_in_week_nights + adults + children + babies + meal +
##     market_segment + distribution_channel + is_repeated_guest +
##     previous_cancellations + previous_bookings_not_canceled +
##     reserved_room_type + assigned_room_type + booking_changes +
##     deposit_type + days_in_waiting_list + customer_type + required_car_parking_spaces +
##     total_of_special_requests + reservation_status + consistent_room_type
## Model 2: adr ~ hotel + is_canceled + lead_time + arrival_date_year + arrival_date_month +
##     arrival_date_week_number + arrival_date_day_of_month + stays_in_weekend_nights +
##     stays_in_week_nights + adults + children + babies + meal +
##     market_segment + distribution_channel + is_repeated_guest +
##     previous_cancellations + previous_bookings_not_canceled +
##     reserved_room_type + assigned_room_type + booking_changes +
##     deposit_type + days_in_waiting_list + customer_type + required_car_parking_spaces +
##     total_of_special_requests + reservation_status + consistent_room_type +
##     arrival_date_season
##   Res.Df       RSS Df  Sum of Sq F Pr(>F)
## 1  88498 106389245
## 2  88498 106389245  0 7.4506e-08
```

```
# use stepwise regression to get a model
stepw=ols_step_both_p(reducedmodel,pent = 0.05, prem = 0.1, details=FALSE)
```

```
## Warning in if (pvals[minp] <= pent) {: the condition has length > 1 and only the
## first element will be used
```

```
summary(stepw$model)
```

```
##
## Call:
## lm(formula = paste(response, "~", paste(preds, collapse = " + ")),
##     data = l)
##
## Residuals:
```

```
##    Min     1Q Median     3Q    Max
## -393.9  -18.4   -2.2   14.9 5314.5
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -3.432e+04  4.454e+02 -77.060  < 2e-16 ***
## arrival_date_year               1.706e+01  2.209e-01  77.209  < 2e-16 ***
## arrival_date_monthAugust        4.459e+01  5.297e-01  84.166  < 2e-16 ***
## arrival_date_monthDecember     -1.114e-01  6.555e-01  -0.170 0.865007
## arrival_date_monthFebruary     -2.505e+01  5.942e-01 -42.163  < 2e-16 ***
## arrival_date_monthJanuary      -2.907e+01  6.582e-01 -44.161  < 2e-16 ***
## arrival_date_monthJuly          3.281e+01  5.376e-01  61.033  < 2e-16 ***
## arrival_date_monthJune          2.043e+01  5.452e-01  37.469  < 2e-16 ***
## arrival_date_monthMarch        -1.649e+01  5.613e-01 -29.376  < 2e-16 ***
## arrival_date_monthMay           1.316e+01  5.336e-01  24.664  < 2e-16 ***
## arrival_date_monthNovember     -2.714e+00  6.528e-01  -4.158 3.21e-05 ***
## arrival_date_monthOctober       1.277e+01  5.807e-01  21.996  < 2e-16 ***
## arrival_date_monthSeptember     2.942e+01  5.914e-01  49.747  < 2e-16 ***
## adults                          6.452e+00  2.224e-01  29.013  < 2e-16 ***
## children                        1.321e+01  3.938e-01  33.552  < 2e-16 ***
## mealFB                          4.105e+01  1.423e+00  28.838  < 2e-16 ***
## mealHB                          2.591e+01  3.821e-01  67.808  < 2e-16 ***
## mealSC                         -8.358e+00  4.293e-01 -19.470  < 2e-16 ***
## market_segmentComplementary    -9.706e+01  3.237e+00 -29.980  < 2e-16 ***
## market_segmentCorporate        -4.787e+00  2.768e+00  -1.729 0.083802 .
## market_segmentDirect            9.824e+00  2.949e+00   3.331 0.000865 ***
## market_segmentGroups           -6.870e+00  2.859e+00  -2.403 0.016285 *
## market_segmentOffline TA/TO    -4.316e+00  2.859e+00  -1.510 0.131173
## market_segmentOnline TA         1.543e+01  2.855e+00   5.403 6.56e-08 ***
## distribution_channelDirect      4.358e+00  1.168e+00   3.731 0.000191 ***
## distribution_channelGDS         2.355e+01  3.077e+00   7.654 1.97e-14 ***
## distribution_channelTA/TO       2.734e-01  9.329e-01   0.293 0.769439
## reserved_room_typeB            -1.674e+01  1.677e+00  -9.984  < 2e-16 ***
## reserved_room_typeC             2.664e+01  1.877e+00  14.192  < 2e-16 ***
## reserved_room_typeD             1.268e+01  7.895e-01  16.063  < 2e-16 ***
## reserved_room_typeE             1.704e+01  1.259e+00  13.536  < 2e-16 ***
## reserved_room_typeF             3.006e+01  1.773e+00  16.949  < 2e-16 ***
## reserved_room_typeG             4.055e+01  2.365e+00  17.147  < 2e-16 ***
## reserved_room_typeH             6.439e+01  3.793e+00  16.976  < 2e-16 ***
## reserved_room_typeL             3.804e+01  1.737e+01   2.190 0.028533 *
## assigned_room_typeB             5.423e+00  1.320e+00   4.110 3.96e-05 ***
## assigned_room_typeC             1.033e+01  1.317e+00   7.848 4.27e-15 ***
## assigned_room_typeD             4.554e+00  7.896e-01   5.768 8.07e-09 ***
## assigned_room_typeE             1.229e+01  1.226e+00  10.018  < 2e-16 ***
## assigned_room_typeF             1.881e+01  1.645e+00  11.435  < 2e-16 ***
## assigned_room_typeG             2.089e+01  2.188e+00   9.548  < 2e-16 ***
## assigned_room_typeH             7.111e+00  3.498e+00   2.033 0.042060 *
## assigned_room_typeI            -1.485e+01  2.882e+00  -5.152 2.58e-07 ***
## assigned_room_typeK             1.694e+00  3.530e+00   0.480 0.631224
## assigned_room_typeL            -1.250e+02  3.878e+01  -3.225 0.001262 **
## customer_typeGroup             -7.149e+00  1.799e+00  -3.974 7.07e-05 ***
## customer_typeTransient          3.444e+00  6.774e-01   5.084 3.69e-07 ***
## customer_typeTransient-Party    6.776e+00  7.142e-01   9.488  < 2e-16 ***
## total_of_special_requests       2.240e+00  1.704e-01  13.144  < 2e-16 ***
```

```
## hotelResort Hotel              -2.224e+01  3.002e-01 -74.101  < 2e-16 ***
## lead_time                      -1.053e-01  1.414e-03 -74.468  < 2e-16 ***
## deposit_typeNon Refund          1.772e+01  5.547e-01  31.941  < 2e-16 ***
## deposit_typeRefundable          1.439e+01  3.168e+00   4.544 5.52e-06 ***
## arrival_date_day_of_month       1.630e-01  1.332e-02  12.245  < 2e-16 ***
## is_repeated_guest1             -7.547e+00  8.087e-01  -9.331  < 2e-16 ***
## reservation_statusCheck-Out    -4.352e+00  3.059e-01 -14.227  < 2e-16 ***
## reservation_statusNo-Show      -2.525e+00  1.184e+00  -2.134 0.032882 *
## days_in_waiting_list            7.612e-02  6.940e-03  10.968  < 2e-16 ***
## required_car_parking_spaces     5.658e+00  5.099e-01  11.096  < 2e-16 ***
## previous_cancellations         -9.825e-01  1.445e-01  -6.798 1.07e-11 ***
## consistent_room_typeYes         5.825e+00  7.641e-01   7.624 2.49e-14 ***
## booking_changes                 1.331e+00  1.890e-01   7.041 1.92e-12 ***
## previous_bookings_not_canceled -5.350e-01  8.943e-02  -5.982 2.21e-09 ***
## stays_in_weekend_nights        -1.047e+00  1.373e-01  -7.627 2.43e-14 ***
## stays_in_week_nights            5.499e-01  7.473e-02   7.358 1.88e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34.67 on 88500 degrees of freedom
## Multiple R-squared:  0.5324, Adjusted R-squared:  0.532
## F-statistic:  1574 on 64 and 88500 DF,  p-value: < 2.2e-16
```

```
anova(stepw$model, fullmodel)
```

```
## Analysis of Variance Table
##
## Model 1: adr ~ arrival_date_year + arrival_date_month + adults + children +
##     meal + market_segment + distribution_channel + reserved_room_type +
##     assigned_room_type + customer_type + total_of_special_requests +
##     hotel + lead_time + deposit_type + arrival_date_day_of_month +
##     is_repeated_guest + reservation_status + days_in_waiting_list +
##     required_car_parking_spaces + previous_cancellations + consistent_room_type +
##     booking_changes + previous_bookings_not_canceled + stays_in_weekend_nights +
##     stays_in_week_nights
## Model 2: adr ~ hotel + is_canceled + lead_time + arrival_date_year + arrival_date_month +
##     arrival_date_week_number + arrival_date_day_of_month + stays_in_weekend_nights +
##     stays_in_week_nights + adults + children + babies + meal +
##     market_segment + distribution_channel + is_repeated_guest +
##     previous_cancellations + previous_bookings_not_canceled +
##     reserved_room_type + assigned_room_type + booking_changes +
##     deposit_type + days_in_waiting_list + customer_type + required_car_parking_spaces +
##     total_of_special_requests + reservation_status + consistent_room_type +
##     arrival_date_season
##   Res.Df       RSS Df Sum of Sq      F Pr(>F)
## 1  88500 106393622
## 2  88498 106389245  2    4377.3 1.8206 0.1619
```

```
# check for multicolinearity
vif(stepw$model)
```

```
##                                     GVIF Df GVIF^(1/(2*Df))
## arrival_date_year                1.803725  1        1.343028
```

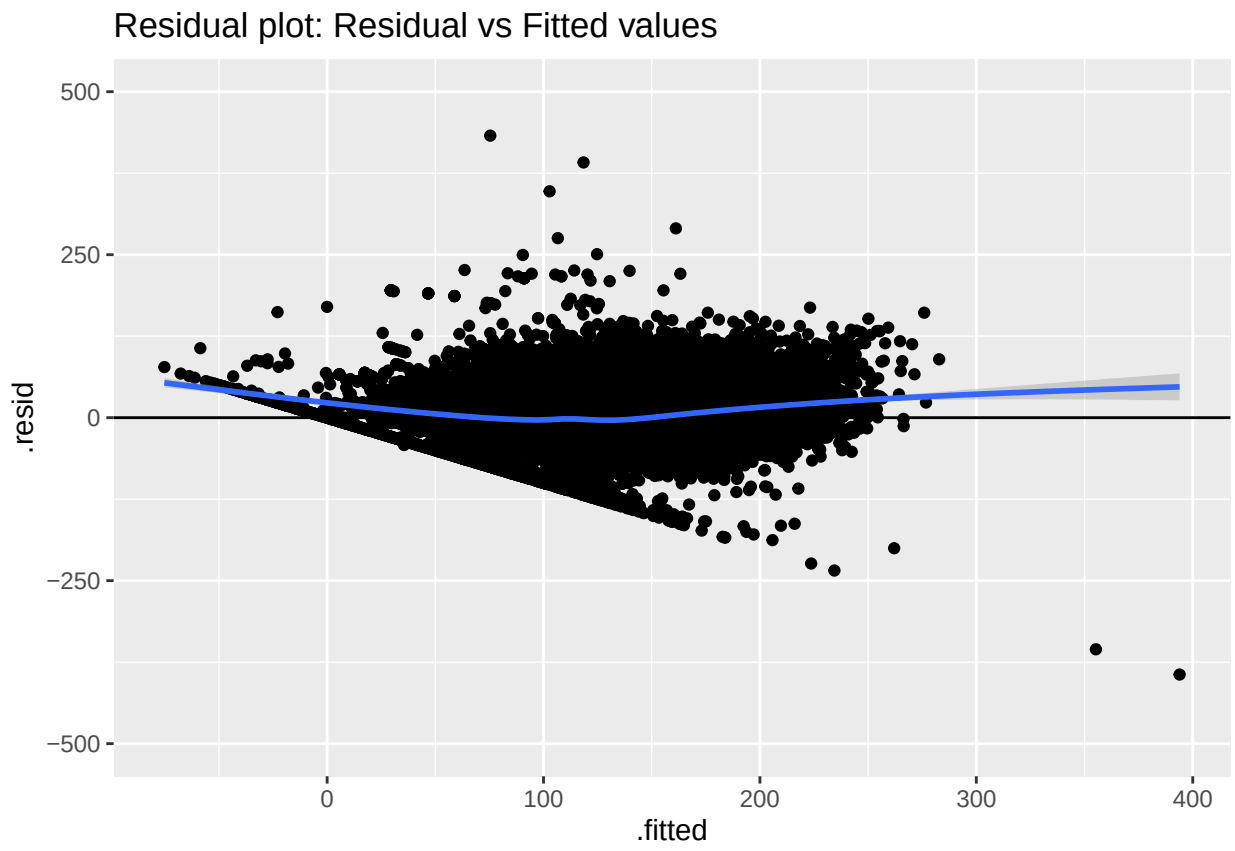
```
## arrival_date_month                 2.091116 11        1.034100
## adults                             1.204635  1        1.097559
## children                           1.803717  1        1.343025
## meal                               1.433324  3        1.061836
## market_segment                    64.067472  6        1.414338
## distribution_channel              20.922628  3        1.659979
## reserved_room_type              8774.895438  8        1.763813
## assigned_room_type             10052.162686 10        1.585306
## customer_type                      2.032692  3        1.125499
## total_of_special_requests          1.347400  1        1.160776
## hotel                              1.471886  1        1.213213
## lead_time                          1.681913  1        1.296886
## deposit_type                       2.524874  2        1.260550
## arrival_date_day_of_month          1.008833  1        1.004407
## is_repeated_guest                  1.395562  1        1.181339
## reservation_status                 1.602956  2        1.125202
## days_in_waiting_list               1.068125  1        1.033502
## required_car_parking_spaces        1.129776  1        1.062909
## previous_cancellations             1.099864  1        1.048744
## consistent_room_type               4.522935  1        2.126719
## booking_changes                    1.081884  1        1.040137
## previous_bookings_not_canceled     1.322202  1        1.149871
## stays_in_weekend_nights            1.371217  1        1.170990
## stays_in_week_nights               1.472320  1        1.213392
```

```r
# we check for the GVIF corrected by the degrees of freedom: GVIF^(1/(2*Df))
# everything is close to or below 2
# so we conclude that there is a little multicollinearity but not significant

# set our final model = stepwise model
final_simple_lm<-lm(adr~arrival_date_year+arrival_date_month+adults+children
                  +meal+market_segment+distribution_channel
                  +reserved_room_type+assigned_room_type+customer_type
                  +total_of_special_requests+hotel+lead_time+deposit_type
                  +arrival_date_day_of_month+is_repeated_guest
                  +reservation_status+days_in_waiting_list
                  +required_car_parking_spaces+previous_cancellations
                  +consistent_room_type+booking_changes
                  +previous_bookings_not_canceled+stays_in_weekend_nights
                  +stays_in_week_nights
                  , data=train)
# summary(final_simple_lm)

# technically,we can do more to build a more complete model
# such as adding in interactions and higher order terms
# however, seeing as that's covered in data 603 and we already did that there,
# we will stop here to keep this report on topic and down to a reasonable length.

# Apply linear regression model to the test set, calculate RMSE
adr_hat<-predict(final_simple_lm,test)
rmse = sqrt(mean((adr_hat-test$adr)^2)) #the mean squared error
rmse
```

```
## [1] 29.90732
```

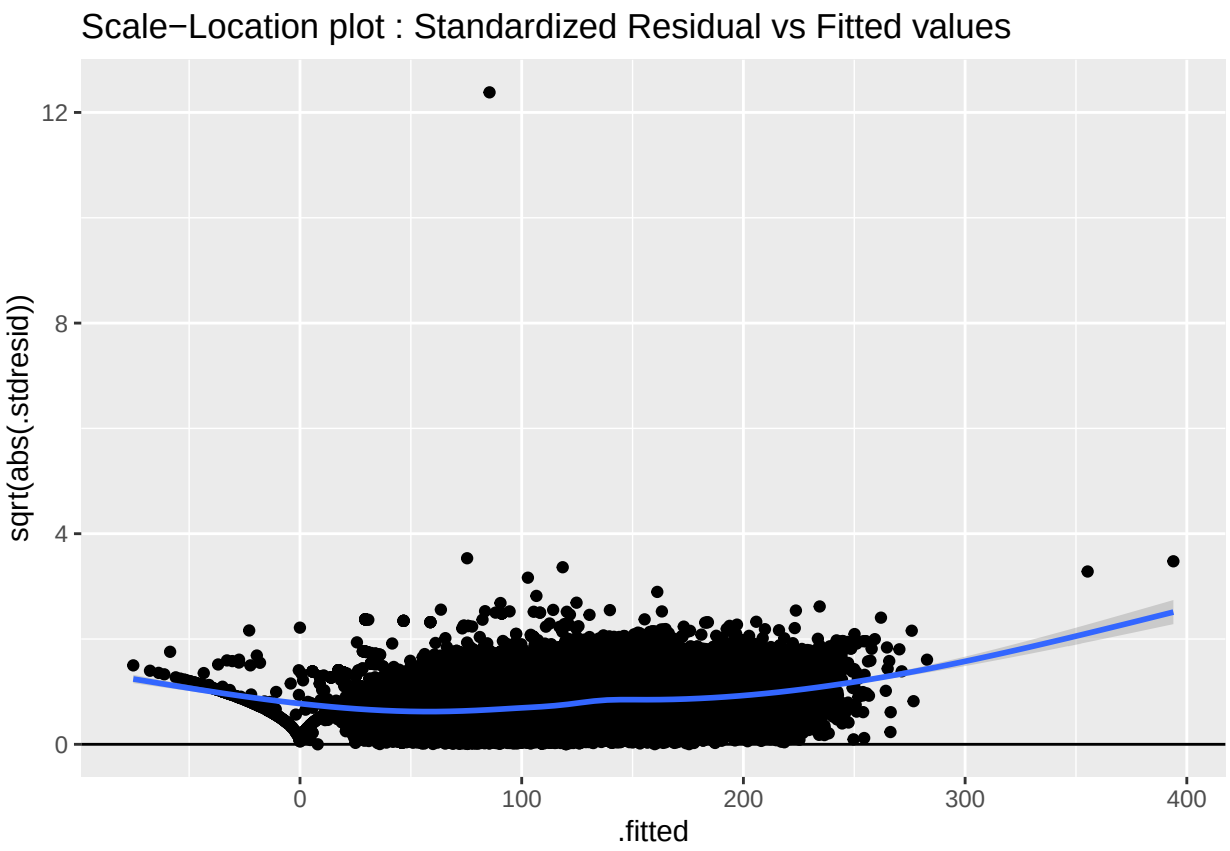
```
# Check our assumptions
# Linearity Assumption check
ggplot(final_simple_lm, aes(x=.fitted, y=.resid)) +
geom_point() +
geom_hline(yintercept = 0) +
geom_smooth()+
ylim(-500, 500)+
ggtitle("Residual plot: Residual vs Fitted values")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'

## Warning: Removed 1 rows containing non-finite values (stat_smooth).

## Warning: Removed 1 rows containing missing values (geom_point).
```




```
# Linearity Assumption Conclusion:
# The Residual vs. Fitted plot shows a curved line,
# indicating that the linearity assumption is not met.

# Equal Variance Assumption Check
ggplot(final_simple_lm, aes(x=.fitted, y=sqrt(abs(.stdresid)))) +
geom_point() +
geom_hline(yintercept = 0) +
geom_smooth()+
ggtitle("Scale-Location plot : Standardized Residual vs Fitted values")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
# Breusch-Pagan test (for equal variance)
# Ho: heteroscedasticity is not present (homoscedasticity)
# Ha: heteroscedasticity is present
bptest(final_simple_lm)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  final_simple_lm
## BP = 39.312, df = 64, p-value = 0.9936
```

```
# Equal Variance Assumption Conclusion:
# The Residual vs. Fitted and Scale-Location plot both don't show any
# evidence of a pattern, hinting that the Equal Variance assumption is met.
# The Breusch-Pagan test returns a p-value = 0.9936 > alpha = 0.05,
# strongly concluding that the Equal Variance assumption is met.

# Check Normality Assumption - Anderson-Darling
# note that we would have used Shapiro wilk if possible,
# but the sample size is >5000, so we can use that

#test normality with Anderson-Darling normality test for larger sample sizes
#Ho: the sample data are significantly normally distributed
```

```
#Ha: the sample data are not significantly normally distributed
ad.test(residuals(final_simple_lm))
```

```
##
##  Anderson-Darling normality test
##
## data:  residuals(final_simple_lm)
## A = 1747.1, p-value < 2.2e-16
```

```
# The Anderson-Darling test returns a p-value < alpha = 0.05,
# strongly concluding that the Equal Variance assumption is not met.
```

Simple first order linear regression model cross-validation error is: RMSE = €29.91, which is smaller than €34.96 (pruned tree) and €33.89 (un-pruned tree). The linear regression model also contains many more significant variables compared to the trees (25 total in the final linear model, vs. 5 in the un-pruned tree, and 4 in the pruned tree)

However, at $\alpha = 0.05$, homoscedasticity assumption was met, but linearity and normality assumptions were not met. This suggests that a linear model may not be a good choice.

Conclusion

The conclusions for each research question is as per below.

Research Question 1: Using contingency tables – the response variable *is_canceled* is dependent on all categorical tested features.

Research Question 2: Random forest appears to be the best model for predicting booking cancellations with the lowest misclassification rate of 15.47% against the test data, and 17.64% in cross-validation, below is a full table comparing the different models misclassification rates.

Classification Method	Misclassification Rate using Validation Set Approach	Misclassification Rate using 10-fold Cross Validation
Logistic Regression	17.3 %	20.05 %
Classification Tree	21.19 %	22.32 %
Random Forest	15.47%	17.64%
LDA*	18.04%	21.85%
QDA*	22.02%	23.51%

Research Question 3: Using regression tree, the variables shown to be important in predicting average daily rate of a hotel stay in Portugal are: Arrival Month, Hotel Type, Market Segment, and Meal Plan.

The report concludes that a relatively accurate model can be produced to a profile for a prospective client at a hotel in Portugal. Using historical client data, a hospitality management team may be able to predict whether a customer will cancel their stay with up to _____ % accuracy. This can be useful for a management team, as well as for future hospitality groups to formulate business models and customer planning.

References

[1] - **Total contribution of travel and tourism to the gross domestic product in Portugal in 2019 and 2020** (Statista, 2021). Retrieved from https://www.statista.com/statistics/770057/travel-and-tourism-s-total-contribution-to-gdp-in-portugal/

[2] - **Portugal - Contribution of travel and tourism to GDP in constant prices of 2011** (knoema, 2022). Retrieved February 16, 2022 from https://knoema.com/atlas/Portugal/topics/Tourism/Travel-and-Tourism-Total-Contribution-to-GDP/Real-contribution-of-travel-and-tourism-to-GDP

[3] - **Hotel booking demand** (Kaggle, 2020). Retrieved February 18, 2022 from https://www.kaggle.com/jessemostipak/hotel-booking-demand

[4] - **Hotel booking demand datasets** (ScienceDirect, 2018). Retrieved February 18, 2022 from https://www.sciencedirect.com/science/article/pii/S2352340918315191#bib5