Logistic Regression

Naomi Zilber

18 February 2023

Overview

Linear models for classifications, where the target variable is qualitative, create decision boundaries to separate the observations into regions in which most observations are of the same class, and each of these decision boundaries is a linear combination of X parameters. Some strengths of these linear models are that they have low variance, are fairly easy to implement and interpret, and some work better for smaller or larger data sets so you are able to choose a specific type of model that will work better given your data set. Some weaknesses are that they have high bias (which causes underfitting) due to the assumption that data follows a linear trend.

The data set used in this notebook is from this link. (https://www.kaggle.com/datasets/ahsan81/hotel-reservations-classification-dataset)

Load data

Read in the data of hotel reservations

```
df <- read.csv("Hotel_Reservations.csv", header=TRUE)
str(df)
```

```
## 'data.frame':    36275 obs. of  19 variables:
##  $ Booking_ID                          : chr  "INN00001" "INN00002" "INN0000
3" "INN00004" ...
##  $ no_of_adults                        : int  2 2 1 2 2 2 2 2 3 2 ...
##  $ no_of_children                      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ no_of_weekend_nights                : int  1 2 2 0 1 0 1 1 0 0 ...
##  $ no_of_week_nights                   : int  2 3 1 2 1 2 3 3 4 5 ...
##  $ type_of_meal_plan                   : chr  "Meal Plan 1" "Not Selected"
"Meal Plan 1" "Meal Plan 1" ...
##  $ required_car_parking_space          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ room_type_reserved                  : chr  "Room_Type 1" "Room_Type 1" "R
oom_Type 1" "Room_Type 1" ...
##  $ lead_time                           : int  224 5 1 211 48 346 34 83 121 4
4 ...
##  $ arrival_year                        : int  2017 2018 2018 2018 2018 2018
2017 2018 2018 2018 ...
##  $ arrival_month                       : int  10 11 2 5 4 9 10 12 7 10 ...
##  $ arrival_date                        : int  2 6 28 20 11 13 15 26 6 18 ...
##  $ market_segment_type                 : chr  "Offline" "Online" "Online" "O
nline" ...
##  $ repeated_guest                      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ no_of_previous_cancellations        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ no_of_previous_bookings_not_canceled: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ avg_price_per_room                  : num  65 106.7 60 100 94.5 ...
##  $ no_of_special_requests              : int  0 1 0 0 0 1 1 1 1 3 ...
##  $ booking_status                      : chr  "Not_Canceled" "Not_Canceled"
"Canceled" "Canceled" ...
```

Data cleaning

Got rid of features that I don’t think will affect the target value (booking status), and converted room_type_reserved, booking_status, and repeated_guest into factors.

```
df <- df[,c(-1,-6,-7,-10,-11,-12,-13)]
df$room_type_reserved <- as.factor(df$room_type_reserved)
df$booking_status <- as.factor(df$booking_status)
df$repeated_guest <- as.factor(df$repeated_guest)
str(df)
```

```
## 'data.frame':	36275 obs. of  12 variables:
##  $ no_of_adults                        : int  2 2 1 2 2 2 2 2 3 2 ...
##  $ no_of_children                      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ no_of_weekend_nights                : int  1 2 2 0 1 0 1 1 0 0 ...
##  $ no_of_week_nights                   : int  2 3 1 2 1 2 3 3 4 5 ...
##  $ room_type_reserved                  : Factor w/ 7 levels "Room_Type 
1",..: 1 1 1 1 1 1 1 4 1 4 ...
##  $ lead_time                           : int  224 5 1 211 48 346 34 83 121 4
4 ...
##  $ repeated_guest                      : Factor w/ 2 levels "0","1": 1 1 1
1 1 1 1 1 1 1 ...
##  $ no_of_previous_cancellations        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ no_of_previous_bookings_not_canceled: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ avg_price_per_room                  : num  65 106.7 60 100 94.5 ...
##  $ no_of_special_requests              : int  0 1 0 0 0 1 1 1 1 3 ...
##  $ booking_status                      : Factor w/ 2 levels "Canceled","Not_
Canceled": 2 2 1 1 1 1 2 2 2 2 ...
```

Handle missing values

There are no NAs to handle in this data set

```
sapply(df, function(x) sum(is.na(x)==TRUE))
```

```
##                         no_of_adults                       no_of_children 
##                                    0                                    0 
##                 no_of_weekend_nights                    no_of_week_nights 
##                                    0                                    0 
##                   room_type_reserved                            lead_time 
##                                    0                                    0 
##                       repeated_guest         no_of_previous_cancellations 
##                                    0                                    0 
## no_of_previous_bookings_not_canceled                   avg_price_per_room 
##                                    0                                    0 
##               no_of_special_requests                       booking_status 
##                                    0                                    0 
```

Divide into train and test data

Divide the data to 80% train data and 20% test data

```
set.seed(1234)
i <- sample(1:nrow(df), 0.8*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

Data exploration

I looked at the first 4 rows using head() to get a peek into what my data looks like, and then used summary() to get an even better idea of the distribution of the data and to get more detailed statistics about it. I also found the average lead time to see how ahead of time people reserved rooms on average, and found the range of the room prices.

```
head(train, n=4)
```

```
summary(train)
```

```
##   no_of_adults   no_of_children   no_of_weekend_nights no_of_week_nights
##  Min.   :0.000   Min.   :0.0000   Min.   :0.0000       Min.   : 0.000   
##  1st Qu.:2.000   1st Qu.:0.0000   1st Qu.:0.0000       1st Qu.: 1.000   
##  Median :2.000   Median :0.0000   Median :1.0000       Median : 2.000   
##  Mean   :1.845   Mean   :0.1063   Mean   :0.8106       Mean   : 2.206   
##  3rd Qu.:2.000   3rd Qu.:0.0000   3rd Qu.:2.0000       3rd Qu.: 3.000   
##  Max.   :4.000   Max.   :9.0000   Max.   :7.0000       Max.   :17.000   
##                                                                         
##    room_type_reserved   lead_time      repeated_guest
##  Room_Type 1:22541    Min.   :  0.00   0:28276       
##  Room_Type 2:  548    1st Qu.: 17.00   1:  744       
##  Room_Type 3:    6    Median : 57.00                 
##  Room_Type 4: 4814    Mean   : 85.08                 
##  Room_Type 5:  214    3rd Qu.:126.00                 
##  Room_Type 6:  772    Max.   :443.00                 
##  Room_Type 7:  125                                   
##  no_of_previous_cancellations no_of_previous_bookings_not_canceled
##  Min.   : 0.00000             Min.   : 0.0000                     
##  1st Qu.: 0.00000             1st Qu.: 0.0000                     
##  Median : 0.00000             Median : 0.0000                     
##  Mean   : 0.02123             Mean   : 0.1537                     
##  3rd Qu.: 0.00000             3rd Qu.: 0.0000                     
##  Max.   :13.00000             Max.   :58.0000                     
##                                                                   
##  avg_price_per_room no_of_special_requests      booking_status 
##  Min.   :  0.00     Min.   :0.0000         Canceled    : 9507  
##  1st Qu.: 80.30     1st Qu.:0.0000         Not_Canceled:19513  
##  Median : 99.45     Median :0.0000                             
##  Mean   :103.40     Mean   :0.6167                             
##  3rd Qu.:120.00     3rd Qu.:1.0000                             
##  Max.   :540.00     Max.   :5.0000                             
## 
```

```
mean(train$lead_time)
```

```
## [1] 85.08115
```

```
range(train$avg_price_per_room)
```

```
## [1]   0 540
```

Generate the covariance and correlations for the quantitative data

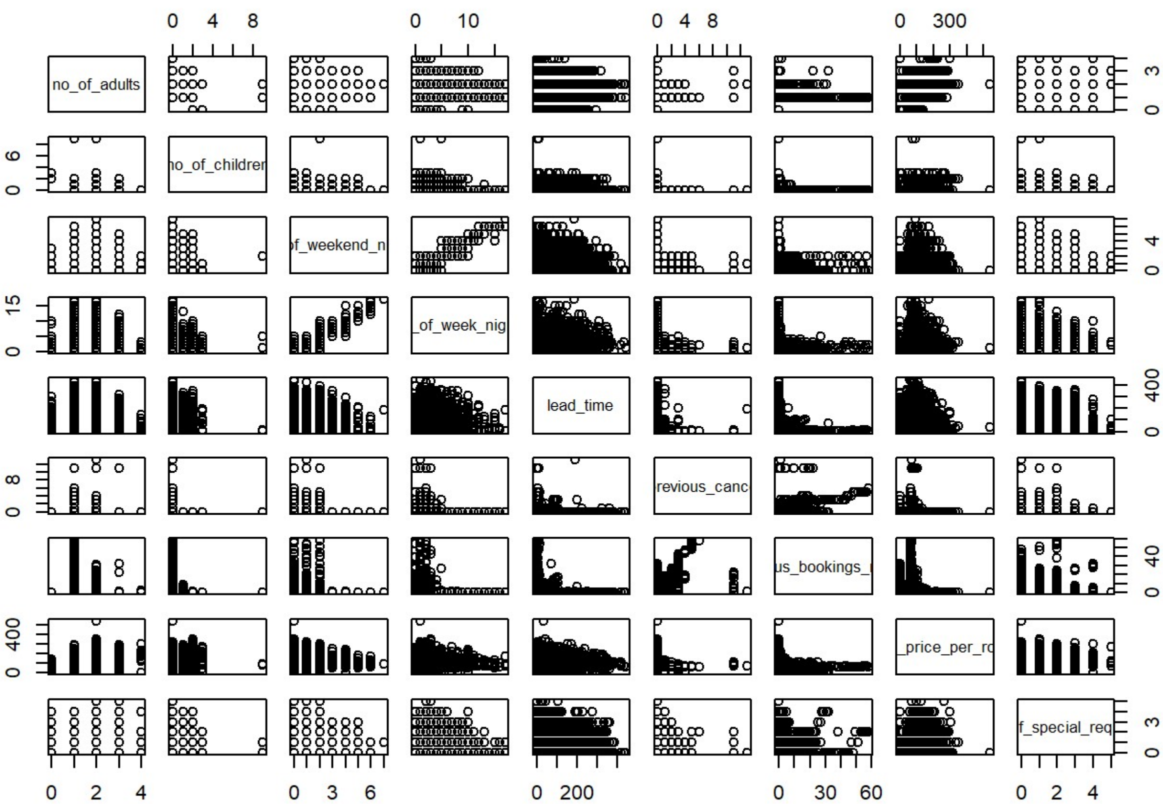
```
train_sub <- train[,c(-5,-7,-12)]

cor(train_sub)
```

```
##                                        no_of_adults no_of_children
## no_of_adults                            1.00000000    -0.02032591
## no_of_children                         -0.02032591     1.00000000
## no_of_weekend_nights                    0.10492641     0.03466046
## no_of_week_nights                       0.10884900     0.02149379
## lead_time                               0.09873086    -0.04891591
## no_of_previous_cancellations           -0.04980791    -0.01678060
## no_of_previous_bookings_not_canceled  -0.11979237    -0.02149803
## avg_price_per_room                      0.29590760     0.33706605
## no_of_special_requests                  0.19109435     0.12303927
##                                        no_of_weekend_nights no_of_week_nights
## no_of_adults                                   0.1049264092        0.10884900
## no_of_children                                 0.0346604576        0.02149379
## no_of_weekend_nights                           1.0000000000        0.18232922
## no_of_week_nights                              0.1823292239        1.00000000
## lead_time                                      0.0468496233        0.14646854
## no_of_previous_cancellations                  -0.0207180993       -0.03384258
## no_of_previous_bookings_not_canceled          -0.0322797398       -0.05583437
## avg_price_per_room                             0.0003565024        0.02305004
## no_of_special_requests                         0.0565876458        0.04499998
##                                          lead_time no_of_previous_cancellation
s
## no_of_adults                            0.09873086                -0.049807914
2
## no_of_children                         -0.04891591                -0.016780598
4
## no_of_weekend_nights                    0.04684962                -0.020718099
3
## no_of_week_nights                       0.14646854                -0.033842577
0
## lead_time                               1.00000000                -0.046720175
5
## no_of_previous_cancellations           -0.04672018                 1.000000000
0
## no_of_previous_bookings_not_canceled -0.07780787                 0.496453936
6
## avg_price_per_room                     -0.06008949                -0.064261801
8
## no_of_special_requests                 -0.09977715                -0.000644492
9
##                                        no_of_previous_bookings_not_canceled
## no_of_adults                                                    -0.11979237
## no_of_children                                                  -0.02149803
## no_of_weekend_nights                                            -0.03227974
## no_of_week_nights                                               -0.05583437
## lead_time                                                       -0.07780787
## no_of_previous_cancellations                                     0.49645394
## no_of_previous_bookings_not_canceled                             1.00000000
```

```
## avg_price_per_room                                                  -0.11337049
## no_of_special_requests                                               0.02898359
##                                        avg_price_per_room no_of_special_reques
ts
## no_of_adults                                 0.2959076049           0.19109435
02
## no_of_children                               0.3370660456           0.12303926
95
## no_of_weekend_nights                         0.0003565024           0.05658764
58
## no_of_week_nights                            0.0230500358           0.04499997
75
## lead_time                                   -0.0600894851          -0.09977714
66
## no_of_previous_cancellations                -0.0642618018          -0.00064449
29
## no_of_previous_bookings_not_canceled        -0.1133704917           0.02898359
02
## avg_price_per_room                           1.0000000000           0.18494416
12
## no_of_special_requests                       0.1849441612           1.00000000
00
```

```
pairs(train_sub)
```




```
cov(train_sub)
```

```
##                                        no_of_adults no_of_children
## no_of_adults                             0.268696483   -0.004233116
## no_of_children                          -0.004233116    0.161420400
## no_of_weekend_nights                     0.047399956    0.012136002
## no_of_week_nights                        0.080127486    0.012263611
## lead_time                                4.394271604   -1.687455442
## no_of_previous_cancellations            -0.008638890   -0.002255876
## no_of_previous_bookings_not_canceled -0.108725304   -0.015123360
## avg_price_per_room                       5.369643289    4.740809819
## no_of_special_requests                   0.077817787    0.038834971
##                                        no_of_weekend_nights no_of_week_nights
## no_of_adults                                     0.047399956        0.08012749
## no_of_children                                   0.012136002        0.01226361
## no_of_weekend_nights                             0.759493858        0.22565477
## no_of_week_nights                                0.225654765        2.01675056
## lead_time                                        3.505671686       17.85967030
## no_of_previous_cancellations                    -0.006041442       -0.01608119
## no_of_previous_bookings_not_canceled            -0.049256396       -0.13883474
## avg_price_per_room                               0.010876345        1.14592453
## no_of_special_requests                           0.038742157        0.05020401
##                                          lead_time no_of_previous_cancellation
s
## no_of_adults                              4.394272                 -0.00863889
0
## no_of_children                           -1.687455                 -0.00225587
6
## no_of_weekend_nights                      3.505672                 -0.00604144
2
## no_of_week_nights                        17.859670                 -0.01608118
7
## lead_time                              7372.351077                 -1.34225814
0
## no_of_previous_cancellations             -1.342258                  0.11195852
1
## no_of_previous_bookings_not_canceled  -11.697607                  0.29085599
0
## avg_price_per_room                     -180.617491                 -0.75273098
5
## no_of_special_requests                   -6.730295                 -0.00016941
3
##                                        no_of_previous_bookings_not_canceled
## no_of_adults                                                    -0.10872530
## no_of_children                                                  -0.01512336
## no_of_weekend_nights                                            -0.04925640
## no_of_week_nights                                               -0.13883474
## lead_time                                                      -11.69760681
## no_of_previous_cancellations                                     0.29085599
## no_of_previous_bookings_not_canceled                             3.06577976
```

```
## avg_price_per_room                                                    -6.94910421
## no_of_special_requests                                                 0.03986784
##                                     avg_price_per_room no_of_special_reques
ts
## no_of_adults                                5.36964329           0.0778177
87
## no_of_children                              4.74080982           0.0388349
71
## no_of_weekend_nights                        0.01087634           0.0387421
57
## no_of_week_nights                           1.14592453           0.0502040
12
## lead_time                                -180.61749101          -6.7302953
01
## no_of_previous_cancellations               -0.75273099          -0.0001694
13
## no_of_previous_bookings_not_canceled       -6.94910421           0.0398678
39
## avg_price_per_room                       1225.50937147           5.0862649
45
## no_of_special_requests                      5.08626494           0.6171632
70
```

From the cor(), I found that most of the correlations are quite weak. The strongest correlations in this data set seem to be between:

- no_of_previous_bookings_not_canceled and no_of_previous_cancellations
- no_of_children and avg_price_per_room
- no_of_adults and avg_price_per_room

The pairs() plots all of these relationships which helps visualize and see the correlations

From the cov(), I found that the strongest covariance values tell me that:

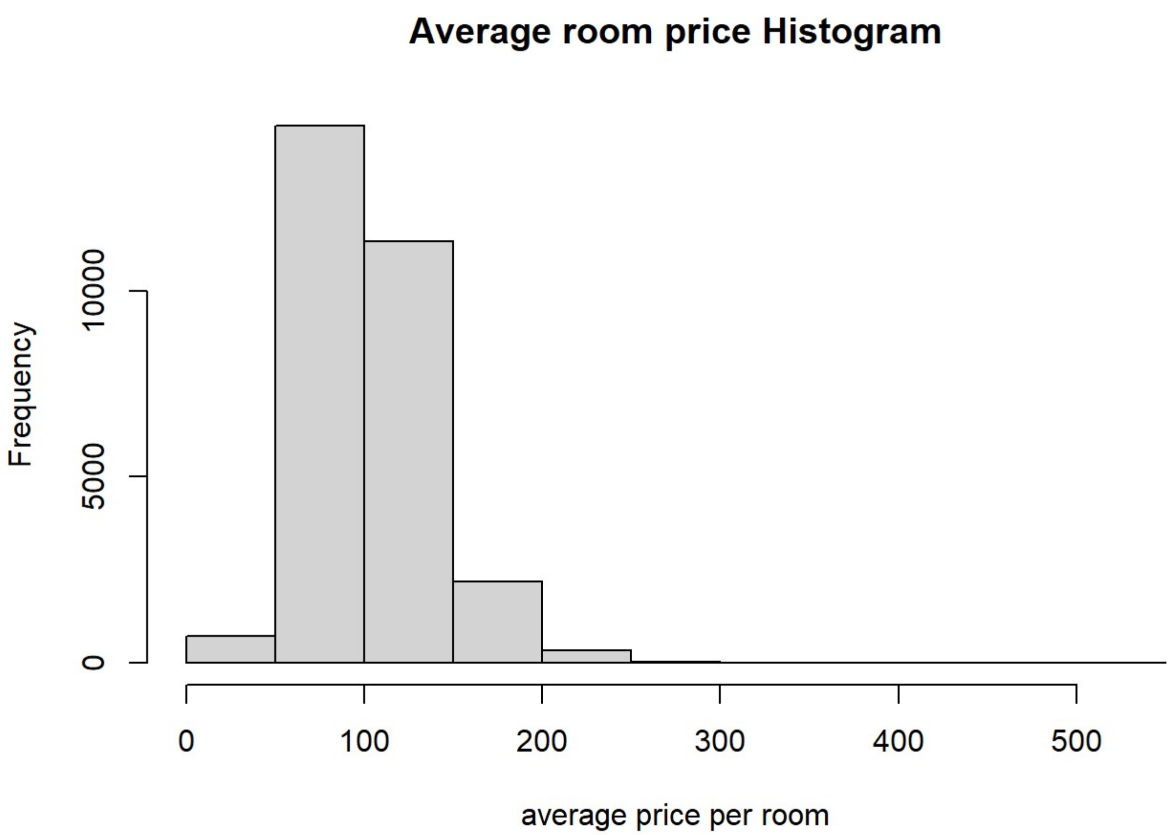
- lead_time and no_of_week_nights are positively and relatively strongly related
- lead_time and avg_price_per_room are negatively and very strongly related
- lead_time and no_of_previous_bookings_not_canceled are negatively and relatively strongly related

Therefore, it seems that lead_time might be an important predictor

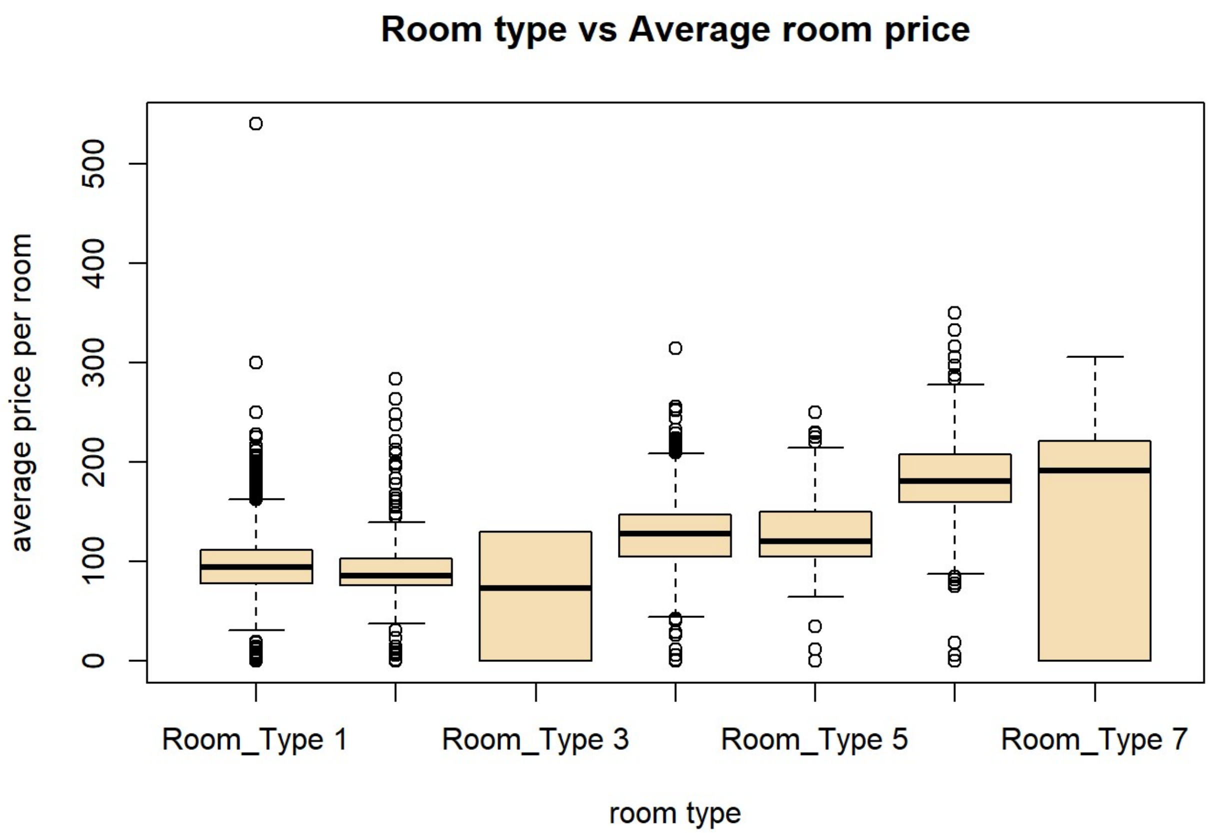
Plots and graphs

The histogram shows that most rooms cost between $50 to $150, and there are many more rooms whose price is above that range than below. The plot built boxplots for every room type based on their average price, which shows that room type 7 has the biggest price range and also the highest median price out of all the room types. Some additional information from this plot is that room types 1 and 2 have close median prices, and room types 4 and 5 have similar median prices as well.

```
hist(train$avg_price_per_room, main="Average room price Histogram", xlab="avera
ge price per room")
```

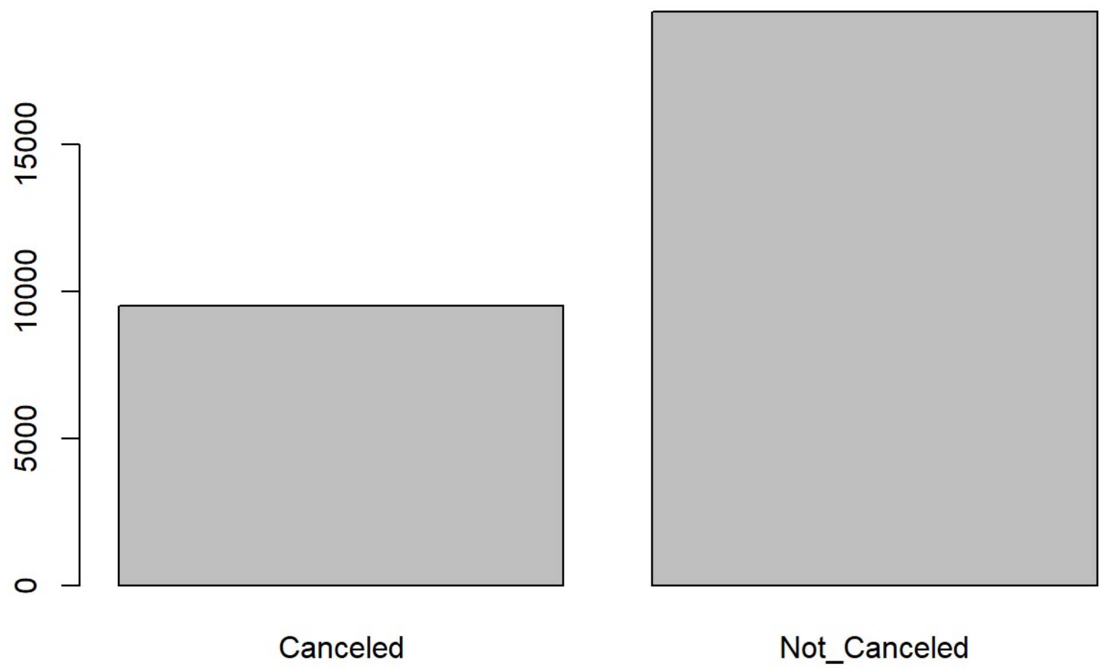



```
plot(train$room_type_reserved, train$avg_price_per_room, col="wheat", main="Roo
m type vs Average room price", xlab="room type", ylab="average price per room")
```

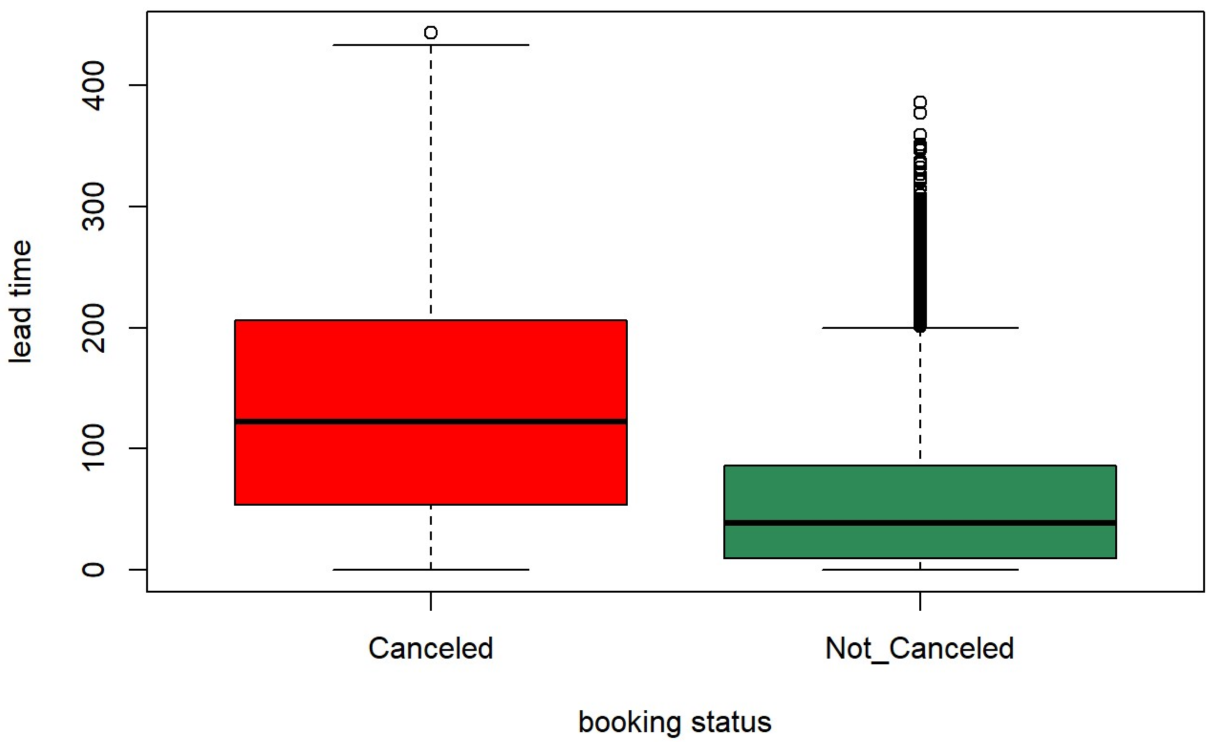


The booking status plot shows that the data is imbalanced since there are almost twice as many “not canceled” cases than there are “canceled” cases, which could mess up to models. The boxplot shows that generally, at larger lead times there are more canceled booking while when the lead time is smaller there are less canceled bookings. This means that there are more cancellations ahead of time rather than last minute. The conditional plot shows that there are less cancellations when a larger number of special requests is made.

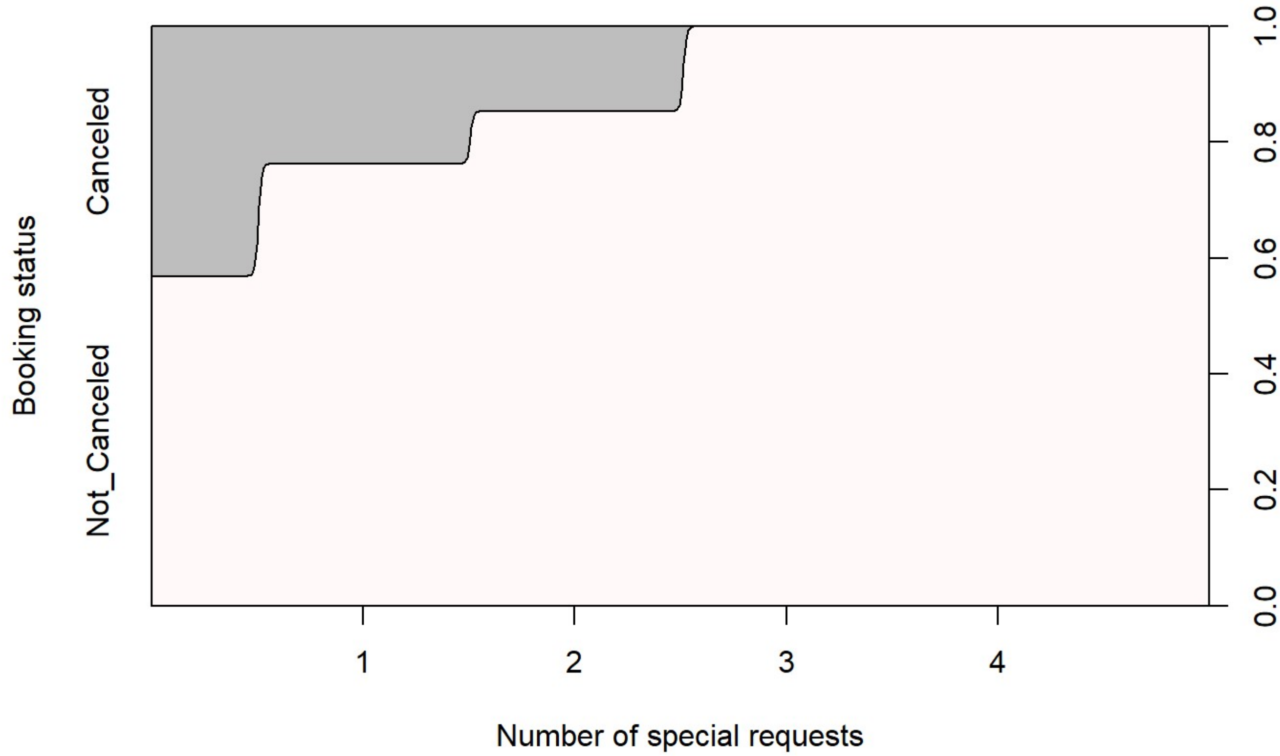
```
plot(train$booking_status)
```




```
boxplot(train$lead_time~train$booking_status, col=c("red", "seagreen"), xlab="b
ooking status", ylab="lead time")
```



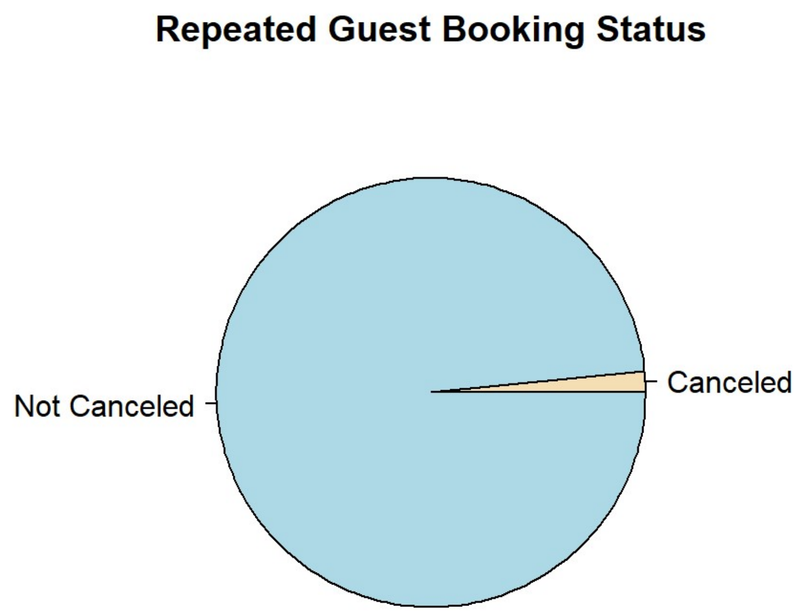

```
cdplot(train$no_of_special_requests, train$booking_status, col=c("snow", "gra
y"), xlab="Number of special requests", ylab="Booking status")
```



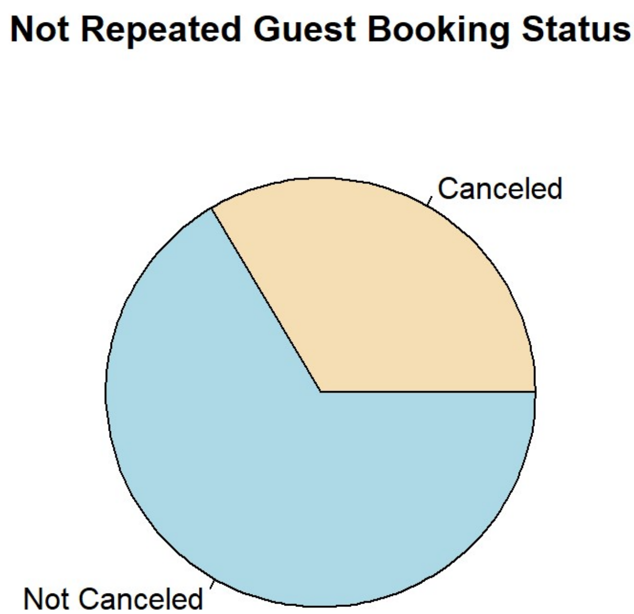
The two pie charts show that there are less cancellations being made by repeated guests rather than not repeated guests. This should be further investigated because the data could be imbalanced in regards to the amount of repeated and not repeated guests.

```
rep_guest <- train$booking_status[train$repeated_guest==1]
not_rep_guest <- train$booking_status[train$repeated_guest==0]

lbls <- c("Canceled", "Not Canceled")
pie(c(sum(rep_guest=="Canceled"), sum(rep_guest=="Not_Canceled")), labels=lbl
s, main="Repeated Guest Booking Status", col=c("wheat", "lightblue"))
```




```
pie(c(sum(not_rep_guest=="Canceled"), sum(not_rep_guest=="Not_Canceled")), labe
ls=lbls, main="Not Repeated Guest Booking Status", col=c("wheat", "lightblue"))
```



Make the logistic regression model

Build a logistic regression model using all predictors. I got rid of the room_type_reserved because it didn't seem like it would have much effect on the booking status.

The summary shows a few things:

- The deviance residuals statistics give an idea of the loss function and a given point's contribution to the overall likelihood
- The coefficient estimates quantify the difference in the log odds of the target value (booking status). It seems most coefficient are good except no_of_children and no_of_previous_bookings_not_canceled.
- The null deviance measures the lack of fit of the model while considering only the intercept
- The residual deviance measures the lack of fit of the model while considering the entire model
- In this model, the residual deviance is much lower than the null deviance, which is what we want to see
- The AIC doesn't tell us much since it is mostly useful when comparing models

```
train <- train[,-5]

glm1 <- glm(booking_status~., data=train, family="binomial")
summary(glm1)
```

```
##
## Call:
## glm(formula = booking_status ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1250  -0.7168   0.4306   0.7286   2.6879
##
## Coefficients:
##                                       Estimate Std. Error z value Pr(>|z
|)
## (Intercept)                          3.8052355  0.0782318  48.641  < 2e-16
***
## no_of_adults                        -0.1147161  0.0315257  -3.639 0.000274
***
## no_of_children                      -0.0045090  0.0399469  -0.113 0.91012
9
## no_of_weekend_nights                -0.2004585  0.0173159 -11.577  < 2e-16
***
## no_of_week_nights                   -0.0591248  0.0106394  -5.557 2.74e-08
***
## lead_time                           -0.0125424  0.0001990 -63.039  < 2e-16
***
## repeated_guest1                      2.3303866  0.4360791   5.344 9.09e-08
***
## no_of_previous_cancellations        -0.2653784  0.0960262  -2.764 0.005717
**
## no_of_previous_bookings_not_canceled  0.1842477  0.1528003   1.206 0.22789
2
## avg_price_per_room                  -0.0187851  0.0005417 -34.677  < 2e-16
***
## no_of_special_requests               1.0951172  0.0243687  44.939  < 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: 36708  on 29019  degrees of freedom
## Residual deviance: 27212  on 29009  degrees of freedom
## AIC: 27234
##
## Number of Fisher Scoring iterations: 9
```

Build a naive Bayes model

The output of the naive Bayes model shows the prior and likelihoods of the data.

- The prior for booking_status (probability of booking_status) is 0.328 canceled and 0.672 not canceled
- Since most of the data is continuous, the mean and standard deviation are outputted for the two classes (canceled/not canceled)
- For repeated_guest, a discrete data type, a breakdown by canceled/not canceled for each possible value of the attribute is outputted. The probabilities of canceled are 99.87% for a not repeated guest and 0.13% for a repeated guest
- The no_of_adults is continuous, so the mean for canceling is 1.9 with standard deviation of 0.48, and the mean for not surviving is 1.8 with standard deviation of 0.53. The means are very close, which means that the number of adults alone doesn't tell us much.

```
library(e1071)
nb1 <- naiveBayes(booking_status~., data=train)
nb1
```

```
## 
## Naive Bayes Classifier for Discrete Predictors
## 
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
## 
## A-priori probabilities:
## Y
##     Canceled Not_Canceled 
##    0.3276017    0.6723983 
## 
## Conditional probabilities:
##               no_of_adults
## Y                  [,1]      [,2]
##   Canceled     1.904597 0.4839600
##   Not_Canceled 1.816379 0.5319384
## 
##               no_of_children
## Y                    [,1]      [,2]
##   Canceled     0.12643315 0.4461335
##   Not_Canceled 0.09644852 0.3779004
## 
##               no_of_weekend_nights
## Y                   [,1]      [,2]
##   Canceled     0.8920795 0.9271414
##   Not_Canceled 0.7708707 0.8402103
## 
##               no_of_week_nights
## Y                  [,1]     [,2]
##   Canceled     2.394972 1.611547
##   Not_Canceled 2.113360 1.306959
## 
##               lead_time
## Y                   [,1]     [,2]
##   Canceled     138.96634 99.02558
##   Not_Canceled  58.82755 63.89847
## 
##               repeated_guest
## Y                        0           1
##   Canceled     0.998737772 0.001262228
##   Not_Canceled 0.962486547 0.037513453
## 
##               no_of_previous_cancellations
## Y                     [,1]      [,2]
##   Canceled     0.003786683 0.1912959
##   Not_Canceled 0.029723774 0.3853057
## 
##               no_of_previous_bookings_not_canceled
```

```
## Y                      [,1]       [,2]
##   Canceled     0.001262228 0.07535934
##   Not_Canceled 0.227899349 2.13071617
##
##               avg_price_per_room
## Y                   [,1]     [,2]
##   Canceled     110.60689 32.19462
##   Not_Canceled  99.88407 35.77694
##
##               no_of_special_requests
## Y                   [,1]      [,2]
##   Canceled     0.3331230 0.5745496
##   Not_Canceled 0.7549326 0.8359151
```

Predict and evaluate results

For the naive Bayes, an accuracy and confusion matrix are generated. It seems that the nb1 model is accurate about 43% of the time, and the confusion matrix shows the following:

- TP - true positive: 2369 bookings were canceled and were predicted as canceled
- FP - false positive: 4122 bookings were not canceled but were predicted as canceled
- FN - false negative: 9 booking were canceled but were predicted as not canceled
- TN - true negative: 755 bookings were not canceled and were predicted as not canceled

```
p1 <- predict(nb1, newdata=test, type="class")
accnb <- mean(p1==test$booking_status)
print(paste("naive Bayes accuracy = ", accnb))
```

```
## [1] "naive Bayes accuracy =  0.430599586492074"
```

```
confus_nb <- table(p1, test$booking_status)
confus_nb
```

```
##
## p1             Canceled Not_Canceled
##   Canceled         2369         4122
##   Not_Canceled        9          755
```

For the logistic regression model, the model is accurate 22% of the time. The confusion matrix shows that:

- TP - true positive: 1129 bookings were canceled and were predicted as canceled
- FP - false positive: 4380 bookings were not canceled but were predicted as canceled
- FN - false negative: 1249 booking were canceled but were predicted as not canceled
- TN - true negative: 497 bookings were not canceled and were predicted as not canceled

```
probs <- predict(glm1, newdata=test, type="response")
pred <- ifelse(probs>0.5, 1, 2)
acc <- mean(pred==as.integer(test$booking_status))
print(paste("glm accuracy = ", acc))
```

```
## [1] "glm accuracy =  0.224121295658167"
```

```
confus_glm <- table(pred, as.integer(test$booking_status))
confus_glm
```

```
##
## pred    1    2
##    1 1129 4380
##    2 1249  497
```

Compare results

Using confusionMatrix() I get more metrics about each model. When comparing the models, it looks like:

- The naive Bayes model is twice as accurate and twice as sensitive (its true positive rate is twice as high) compared to the logistic regression model
- Both models have similar specificity (true negative rate), with the specificity of the nb model being slightly larger than that of the glm
- The Kappa metric is terrible for both models, with the np model having kappa=0.1047 which means there is poor agreement and the classification did a bit better than random values, while the glm model having kappa=-0.3165 which is very poor agreement and classification is worse than random

This supports the results of the confusion matrices, which show that the naive Bayes model did a better job predicting the data.

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
# naive Bayes
confusionMatrix(as.factor(as.integer(p1)), reference=as.factor(as.integer(test
$booking_status)))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    2
##          1 2369 4122
##          2    9  755
##
##                Accuracy : 0.4306
##                  95% CI : (0.4192, 0.4421)
##     No Information Rate : 0.6722
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1047
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9962
##             Specificity : 0.1548
##          Pos Pred Value : 0.3650
##          Neg Pred Value : 0.9882
##              Prevalence : 0.3278
##          Detection Rate : 0.3265
##    Detection Prevalence : 0.8947
##       Balanced Accuracy : 0.5755
##
##        'Positive' Class : 1
##
```

```
# glm
confusionMatrix(as.factor(pred), reference=as.factor(as.integer(test$booking_st
atus)))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    2
##          1 1129 4380
##          2 1249  497
##
##                Accuracy : 0.2241
##                  95% CI : (0.2146, 0.2339)
##     No Information Rate : 0.6722
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.3165
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.4748
##             Specificity : 0.1019
##          Pos Pred Value : 0.2049
##          Neg Pred Value : 0.2847
##              Prevalence : 0.3278
##          Detection Rate : 0.1556
##    Detection Prevalence : 0.7593
##       Balanced Accuracy : 0.2883
##
##        'Positive' Class : 1
##
```

The MCC metric, which accounts for differences in class distribution unlike the accuracy metric, of each model show that:

- There is weak agreement between the predictions and actual values in the naive Bayes model
- There some disagreement between the predictions and actual values in the logistic regression model

```
library(mltools)
```

```
##
## Attaching package: 'mltools'
```

```
## The following object is masked from 'package:e1071':
##
##     skewness
```

```
print(paste("nb mcc = ", mcc(as.integer(p1), as.integer(test$booking_status))))
```

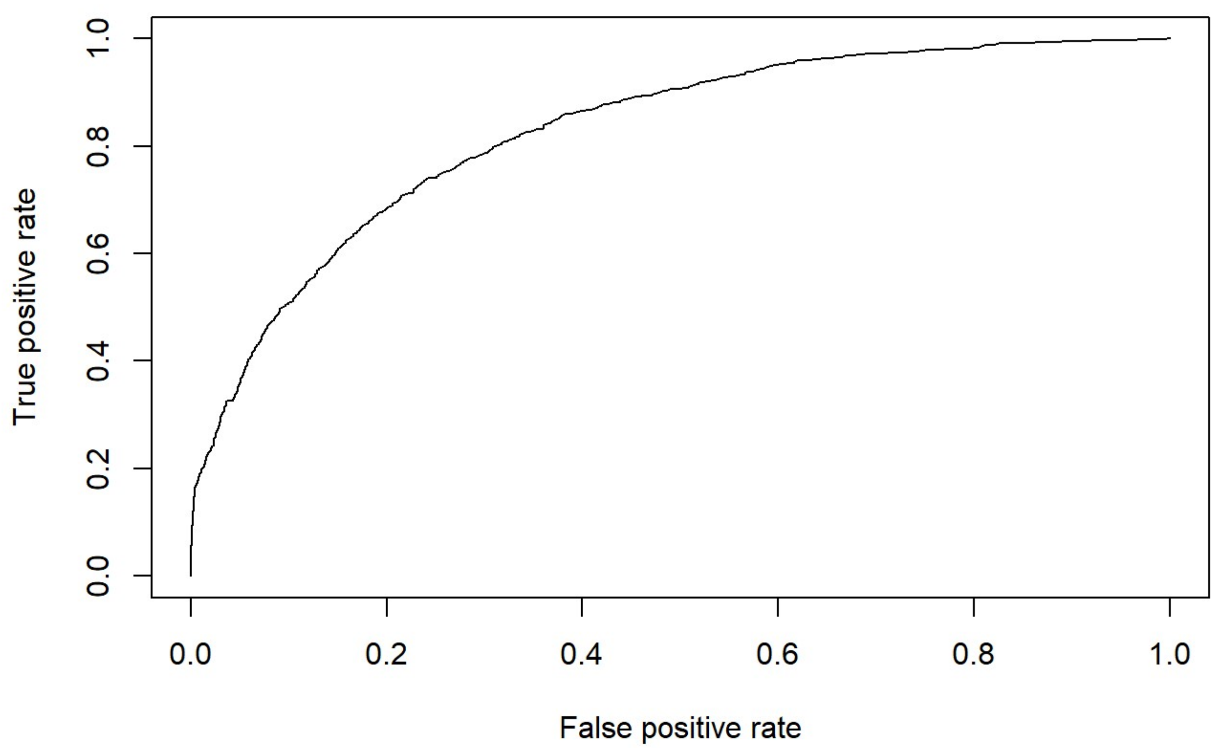
```
## [1] "nb mcc =  0.230953533406315"
```

```
print(paste("glm mcc = ", mcc(pred, as.integer(test$booking_status))))
```

```
## [1] "glm mcc =  -0.464833130928096"
```

The ROC curve shows the trade-off between predicting true positives while avoiding false positives by plotting the TPR against the FPR. Here, the ROC curve show up much too quickly. The AUC metric is the area under the curve, where 0.5 means the classifier has no predictive value while 1 means the classifier is perfect. Here, the AUC value is 0.83, which is relatively good.

```
library(ROCR)
p3 <- predict(glm1, newdata=test, type="response")
pr <- prediction(p3, test$booking_status)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")
plot(prf)
```




```
auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
print(paste("auc = ", auc))
```

```
## [1] "auc =  0.827232596387537"
```

The naive Bayes did much better then the logistic regression model based on all of the above metrics. However, since the data is not balanced, that most likely affected the results in some way. A reason why the naive Bayes model performed better than the logistic regression model could be due to the naive assumption that naive Bayes makes that all of the predictors are independent and because of the size of the training data set.

Logistic regression vs naive Bayes

Logistic Regression: Despite its misleading name, logistic regression is used for classification, not regression. Logistic regression is a still considered a linear model because it is linear in the parameters; the sigmoid function then shapes the output to be in the range [0,1] for probabilities. Some strengths of logistic regression are:

- It separates classes well given that the classes are linearly separable
- It is computationally inexpensive
- It has a nice probabilistic output

Some weaknesses are that it is prone to underfitting due to high bias and because it is not flexible enough to capture complex non-linear decision boundaries.

Naive Bayes: it is a dependable classifier often used as a baseline for more sophisticated algorithms that are expected to outperform it. Some strengths of naive Bayes are:

- It works best with small data sets
- It is easy to implement and interpret
- It can handle high dimensions well

Some weaknesses are:

- It will likely get outperformed by other classifiers for larger data sets
- Assumes that predictors are independent
- It has high bias and therefore is prone to underfitting
- Makes guesses for test data values that didn’t occur in the training data

Classification metrics

Accuracy: the most common metric to evaluate results in classification. Tells how accurate the model is in the range [0,1], with values closer to 1 being better.

- accuracy = (number of correct predictions) / (total number of test observations)
- Benefit - gives a quick glance into the accuracy of the model
- Drawback - doesn’t account for differences in class distribution or predictions by chance

Sensitivity: measures the true positive rate, and range [0,1] with values closer to 1 being better

- Benefit - help quantify the extent to which a given class was misclassified
- Drawback - more likely to be affected by imbalanced data sets, can be affected by thresholds

Specificity: measures the true negative rate, and range [0,1] with values closer to 1 being better

- Benefit - help quantify the extent to which a given class was misclassified, less likely to be affected by imbalanced data sets
- Drawback - can be affected by thresholds

Kappa: attempts to adjust accuracy by accounting for the possibility of a correct prediction by chance alone. It is often used to quantify agreement between two annotators of data.

- Benefit - takes into account imbalance in class distribution, takes chance into consideration
- Drawback - more complex to interpret and the same model will give different kappa value depending on how balanced the test data is.

ROC Curve: shows the trade-off between predicting true positives while avoiding false positives.

- Benefit - shows sensitivity vs specificity at all possible thresholds
- Drawback - dependent on the order of probabilities, can't be used to compare models to one another

AUC: the area under the ROC curve. Its values range [0.5, 1] (for a classifier with no predictive value to a prefect classifier).

- Benefit - can be used to compare different models
- Drawback - ignores the predictive probability values and goodness-of-fit of the model

MCC: accounts for differences in class distribution unlike accuracy; ranges [-1,1].

- Benefit - takes class distribution differences into account, useful when classes are imbalanced
- Drawback - for binary classification only

References:

Mazidi, Karen. *Machine Learning Handbook Using R and Python*. 2nd ed., 2020.