Hotel Booking Analysis

Sezgi Çobanbaş (212144), Zeynep Sıla Kaya (2106877), Emel Dar Omar Safarini (207500)

2024/06/21

Project Description

In this project, we tackle a binary classification problem related to hotel bookings. Customers' reservations can either be "Canceled" or "Not Canceled," based on various parameters represented in the dataset. These parameters include details such as lead time, number of guests, and stay duration, which influence the likelihood of a booking being canceled. This analysis aims to predict whether a reservation will be canceled and to identify the key characteristics that contribute to this outcome using regression methods.

Data Set

We used the hotel_booking dataset from Kaggle for our project which contains 119,390 observations and 36 features. Each observation represents a hotel booking between July 1, 2015, and August 31, 2017, for two different types of hotels which are City Hotel and Resort Hotels. Our dataset includes both completed and canceled reservations, represented by the "is_canceled" feature. Below, you can see the libraries we used in our project.

```r
library(magrittr)
library(dplyr)
library(xtable)
library(readr)
library(plotrix)
library(corrplot)
library(rsample)
library(pROC)
library(olsrr)
library(MASS)
library(glmnet)
library(stepAIC)
```

Data Pre-processing

```r
booking <- read.table(file="C:\\Users\\User\\Desktop\\stat\\hotel_booking.csv", header=TRUE,
sep=',')
```

Variable	**Description**
hotel	The dataset contains the booking information of two hotels. One is a resort hotel and the other is a city hotel.
is_cancelled (target)	The value indicating if the booking was canceled (1) or not (0).
lead_time	Number of days that elapsed between the entering date of the booking into the PMS and the arrival date.
arrival_date_year	Year of the arrival date.
arrival_date_month	The month of the arrival date with 12 categories: “January” to “December”.
arrival_date_week_number	Week number of the arrival date
arrival_date_day_of_month	Day of the month of the arrival date.
stays_in_weekend_nights	The number of weekend nights (Saturday or Sunday) the guest stayed or booked to stay at the hotel.
stays_in_week_nights	The number of weeknights (Monday to Friday) the guest stayed or booked to stay at the hotel.
adults	Number of adults included in the booking.
children	Number of children included in the booking.
babies	Number of babies included in the booking.
meal	Type of meal booked. Categories include "BB – Bed & Breakfast".
country	Country of origin of the guest.
market_segment	Market segment designation. Categories include "TA" for Travel Agents and "TO" for Tour Operators.
distribution_channel	Booking distribution channel. Categories include "TA" for Travel Agents and "TO" for Tour Operators.

is_repeated_guest	The value indicating if the booking name was from a repeated guest (1) or not (0).
previous_cancellations	Number of previous bookings that were canceled by the customer before the current booking.
previous_bookings_not_canceled	Number of previous bookings not canceled by the customer before the current booking.
reserved_room_type	Code of room type reserved. Code is presented instead of designation for anonymity reasons.
assigned_room_type	Code for the type of room assigned to the booking. Sometimes differs from reserved room type due to hotel operation reasons.
booking_changes	Number of changes/amendments made to the booking from booking entry to check-in or cancellation.
deposit_type	Type of deposit made. Categories include "No Deposit", "Non Refund", "Refundable".
agent	ID of the travel agency that made the booking.
company	ID of the company/entity that made the booking or responsible for paying the booking
days_in_waiting_list	Number of days the booking was on the waiting list before it was confirmed to the customer.
customer_type	Type of booking. Categories include "Group", "Transient", and "Transient-party".
adr	Average Daily Rate, calculated by dividing the sum of all lodging transactions by the total number of staying nights.
required_car_parking_spaces	Number of car parking spaces required by the customer.
total_of_special_requests	The number of special requests made by the customer (e.g., twin bed or high floor).

reservation_status	Current reservation status. Categories include "Check-Out" and "No-Show".
reservation_status_date	The date at which the last status was set
name	Name of the guest (Not Real).
email	Email of the guest (Not Real).
phone-number	Phone number of the guest (Not Real).
credit_card	Credit Card Number of the guest (Not Real).

We have 36 features in our dataset, and not all of them may contribute to the model. Some variables may contain null or NA values, adversely affecting the model's outcomes. Therefore, we proceeded step by step with data cleaning to address these issues. First, we removed variables containing personal information such as name, email, phone number, and credit card. Additionally, we deleted the adr variable because it was not relevant to the customer; rather, it was a metric that was primarily used on the hotel's side. We also deleted the reservation_status_date variable because information about when the last status change occurred in the customer's reservation was not particularly useful for our analysis.

```
booking <- booking[, -which(names(booking) %in%
c("adr","arrival_date_day_of_month","arrival_date_week_number",
                                        "credit_card","email","name","phone.number",
"reservation_status_date"))]
```

Then we examined the reservation_status variable. In addition to the is_canceled variable, it also included "No-Show," indicating customers who did not cancel their reservation but did not show up at the hotel. We decided to delete this variable to avoid redundancy, as it contained duplicate information with is_canceled.

```
is_cancelled_counts <- table(booking$is_canceled)
is_cancelled_counts

##     0     1
## 75166 44224

reservation_status_counts <- table(booking$reservation_status)
reservation_status_counts

##  Canceled Check-Out   No-Show
##     43017     75166      1207
```

```
booking <- booking[, -which(names(booking) %in% c("reservation_status"))]
```

To check the NA values in the data set, a set named booking_na was created and the blank data was assigned as NA.

```
booking_na <- booking
booking_na[booking_na == ""] <- NA
anyNA(booking_na)

## [1] TRUE
```

By looking at the number of null values, it can be seen that the "Company" and "Agent" columns contain a large number of null values. In addition, when there is no "Company" column, the "Agent" column does not affect the research because they are related to each other, so these variables are also removed from our dataset.

```
colSums(is.na(booking_na))

##                          hotel                    is_canceled
##                              0                              0
##                      lead_time              arrival_date_year
##                              0                              0
##             arrival_date_month        stays_in_weekend_nights
##                              0                              0
##           stays_in_week_nights                         adults
##                              0                              0
##                       children                         babies
##                              4                              0
##                           meal                        country
##                              0                            488
##                 market_segment           distribution_channel
##                              0                              0
##              is_repeated_guest         previous_cancellations
##                              0                              0
## previous_bookings_not_canceled             reserved_room_type
##                              0                              0
##             assigned_room_type                booking_changes
##                              0                              0
##                   deposit_type                          agent
##                              0                          16340
##                        company           days_in_waiting_list
##                         112593                              0
##                  customer_type    required_car_parking_spaces
##                          0                          0
##      total_of_special_requests
##                          0
```

Using the na.omit function, rows containing NA were deleted and the remaining values were assigned as booking_clean.

```
booking_na<- booking_na[, -which(names(booking) %in% c("agent","company"))]

booking_clean <- na.omit(booking_na)
anyNA(booking_clean)

## [1] FALSE
```

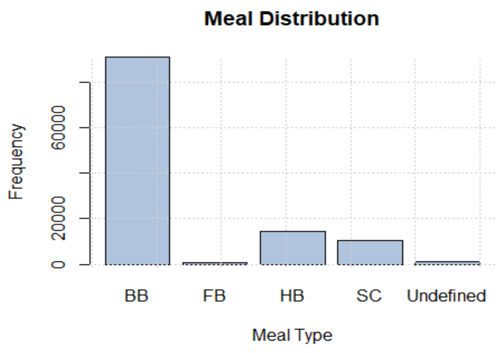
Also, there were contradictory cases in some rows. For instance, bookings with zero adults or zero nights stayed. To address these inconsistencies in the dataset, we removed rows where both weekends and weekdays were zero, where the number of adults was zero, and where adults, children, and babies were all zero simultaneously.

```
booking_clean$staysin <- rowSums(booking_clean[, c("stays_in_weekend_nights",
"stays_in_week_nights")])
booking_clean <- booking_clean[!(booking_clean$staysin == 0), ]

booking_clean <- booking_clean[, -which(names(booking_clean) %in% c("is_repeated_guest",
"previous_cancellations", "previous_bookings_not_canceled", "total_of_special_requests"))]

booking_clean <- booking_clean[!(booking_clean$adults == 0),]
booking_clean <- booking_clean[!(booking_clean$adults == 0 & booking_clean$children == 0
& booking_clean$babies == 0), ]

attach(booking_clean)
```

We also removed the variable 'total_of_special_requests' from our dataset because only 36 entries had special requests which is very low prevalence across the entire dataset. And finally, due to discrepancies in the counts among the variables 'is_repeated_guest', 'previous_cancellations', and 'previous_booking_not_canceled', we decided to remove these variables from our dataset. When we examined the dataset, we observed cases where the 'previous_booking_not_canceled' value was greater than zero for customers defined as never having arrived before, or where the number of previously arrived customers was less than those with 'previous_booking_not_canceled' values greater than zero. Due to significant discrepancies among these three variables, we decided to remove them from the dataset. After completing all these cleaning stages, our dataset now contains 117,865 rows and 21 columns.

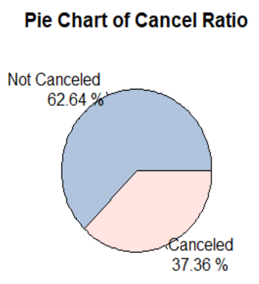
Data Visualization

```
bar_plot <- table(booking_clean$meal)
percentages <- round((bar_plot / sum(bar_plot)) * 100, 1)
barplot_heights <- barplot(bar_plot,
                        main = "Meal Distribution",
                        xlab = "Meal Type",
                        ylab = "Frequency",
                        col = "lightsteelblue",
                        font.main = 2)
grid()
```



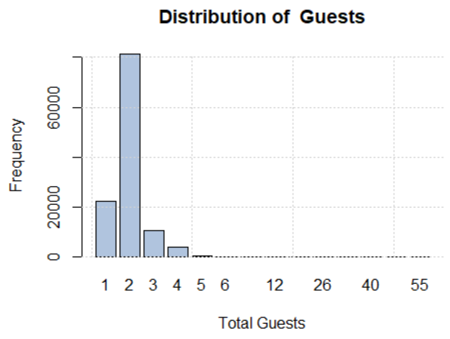
As seen in the chart, the majority of people include breakfast in their hotel reservations with the Bed&Breakfast option among the options given during reservation.

```
freq.tb.iscanceled <- table(is_canceled)
prop.tb.iscanceled <- prop.table(freq.tb.iscanceled)
prop.tb.iscanceled <- round(prop.tb.iscanceled * 100,
2)
labelpie <- c("Not Canceled", "Canceled")
labels <- paste(labelpie, "\n", format(prop.tb.iscanceled, nsmall = 2), "%")
a<-prop.table(freq.tb.iscanceled)
pie(a, labels = labels, main = "Pie Chart of Cancel Ratio", col = c("lightsteelblue",
"mistyrose"),cex = 1,
   font.main = 2)
```



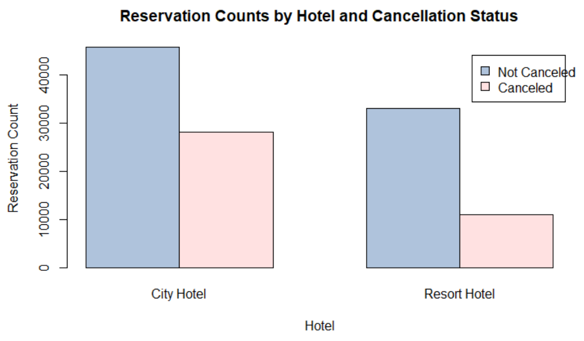
It was determined that there was a 37.36% rate of cancellation of reservations across Resort Hotels and City Hotels.

```
                                    booking_clean$total_guests <- rowSums(booking_clean[,
c("adults", "children", "babies")])
freq.total_guests <- table(booking_clean$total_guests)
barplot_heights <-barplot(freq.total_guests,
                      main = "Distribution of  Guests",
                      xlab = "Total Guests",
                      ylab = "Frequency",
                      col = "lightsteelblue",
                      ylim = c(0, max(freq.total_guests)))
```



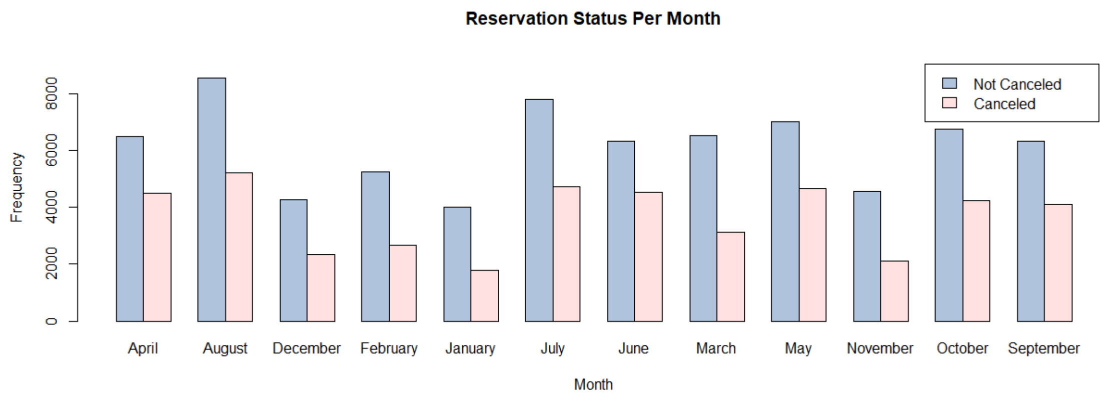
As can be seen on the chart, the majority (68.7%) are reservations for 2 people.

```
hotel_names <- c("City Hotel" , "Resort Hotel")
booking_counts <- table(booking_clean$hotel, booking_clean$is_canceled)
barplot(booking_counts, beside = TRUE,
      col = c("lightsteelblue", "mistyrose"),
      main = "Reservation Counts by Hotel and Cancellation Status",
      xlab = "Hotel", ylab = "Reservation Count",
      legend = c("Not Canceled", "Canceled"),
      names.arg = hotel_names)
```



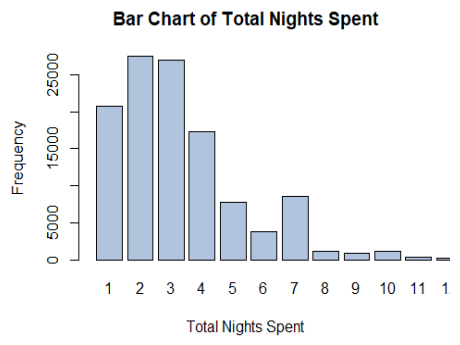
In the above visualization, we can observe the cancellation rates for two separate hotels. Of all reservations, 66.9% were made at the City Hotel, and 33.1% at the Resort Hotel. Specifically, at the City Hotel, the cancellation rate is 41.7%, whereas at the Resort Hotel, it is 28.2%.

```r
reservation_summary <- table(booking_clean$is_canceled, booking_clean$arrival_date_month)
barplot(reservation_summary, beside = TRUE, col = c("lightsteelblue", "mistyrose"),
        legend = c("Not Canceled", "Canceled"),
        main = "Reservation Status Per Month",
        xlab = "Month",
        ylab = "Frequency",
        ylim = c(0, max(reservation_summary) * 1.1))
```



We examined the reservation status by month and we observed that hotel bookings significantly increase during summer months compared to winter.

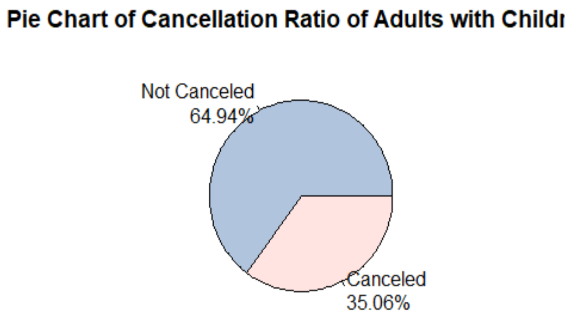
```r
freq.total_staysin <- table(booking_clean$staysin)
barplot(freq.total_staysin,
        main = "Bar Chart of Total Nights Spent",
        xlab = "Total Nights Spent",
        ylab = "Frequency",
        col = "lightsteelblue",
        ylim = c(0, max(freq.total_staysin)),
        xlim = c(0,12))
```



We can observe that people mostly make reservations spanning between 1 and 7 days.

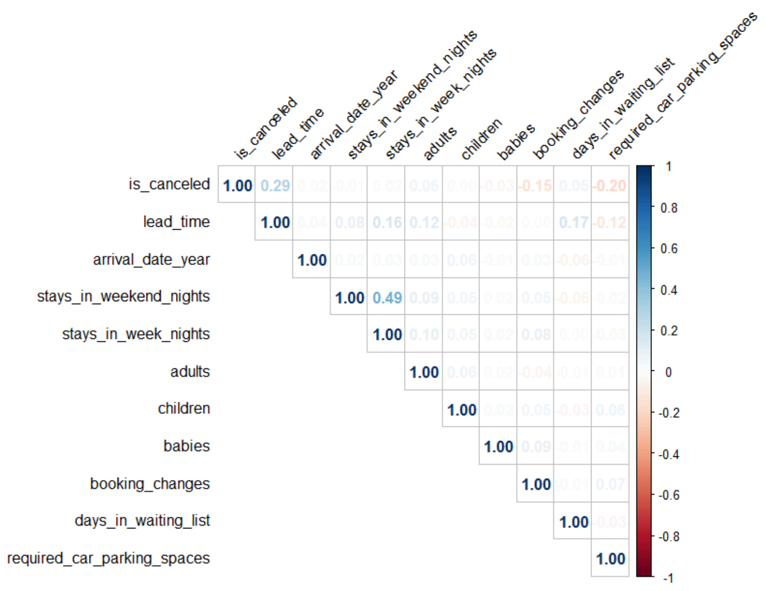
```r
booking_clean <- booking_clean[, -which(names(booking_clean) %in% c("staysin"))]
```

```
families <- booking_clean$children > 0 | booking_clean$babies > 0
true_families_count <- sum(families)
families_with_children_or_babies <- subset(booking_clean, children > 0 | babies > 0)

freq.canceled <- table(families_with_children_or_babies$is_canceled)
prop.canceled <- prop.table(freq.canceled)
labelpie <- c("Not Canceled", "Canceled")
percentages <- round(prop.canceled * 100, 2)
labels <- paste(labelpie, "\n", percentages, "%", sep="")
colors <- c("lightsteelblue", "mistyrose")
pie(prop.canceled, labels = labels, main = "Pie Chart of Cancellation Ratio of Adults with
Children", col = colors)
```

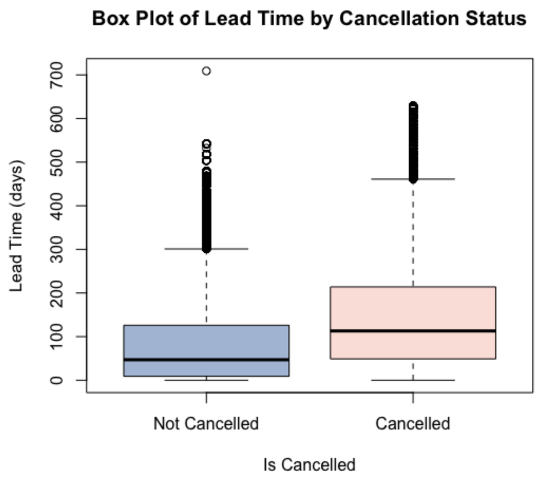


The pie chart reveals that among adults accompanied by children or babies, 35.06% canceled their reservations, while 64.94% did not cancel and stayed at the hotel.

```
numeric <- booking_clean[, sapply(booking_clean, is.numeric)]
numeric<-na.omit(numeric)

korelasyon_matris <- cor(numeric)

corrplot(korelasyon_matris, method = "number", type = "upper",
        tl.col = "black", tl.srt = 45,
        tl.pos = "lt"
)
```

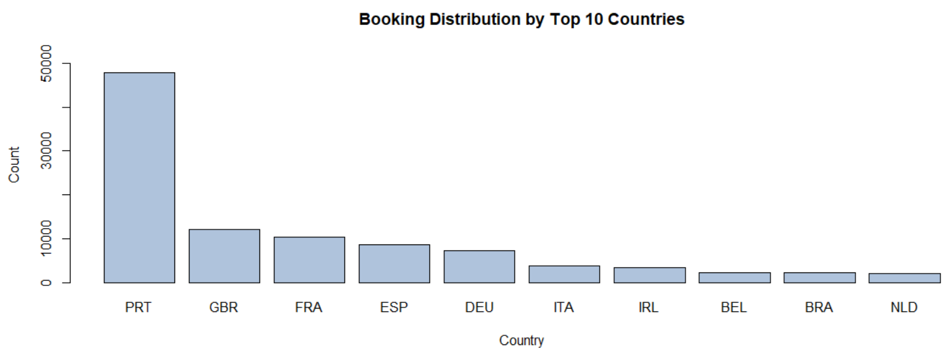


Upon reviewing the correlation matrix, a moderate relationship of 0.29 is observed between "is_canceled" and "lead_time". This suggests that as the duration between reservation and arrival increases, the probability of cancellation also rises. Additionally, it is noticeable that customers who canceled their reservations had notably longer lead times compared to those who did not cancel.



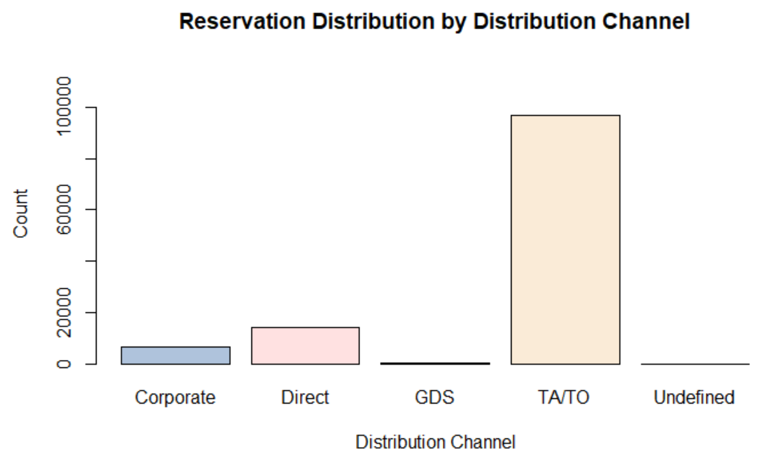
A correlation of -0.20 between “required-car_parking_space” and “is_canceled” suggests a weak negative relationship between these two variables. This implies that as the number of required car parking spaces increases, the likelihood of cancellation decreases slightly. However, it is important to note that the strength of this relationship is weak, so the impact of total special requests on cancellation is not substantial.

```
country_counts <- table(booking_clean$country)
top_10_countries <- head(sort(country_counts, decreasing = TRUE), 10)
barplot(top_10_countries, main = "Booking Distribution by Top 10 Countries", xlab =
"Country",
      ylab = " Count", col = "lightsteelblue", ylim = c(0, max(top_10_countries) * 1.1))
```



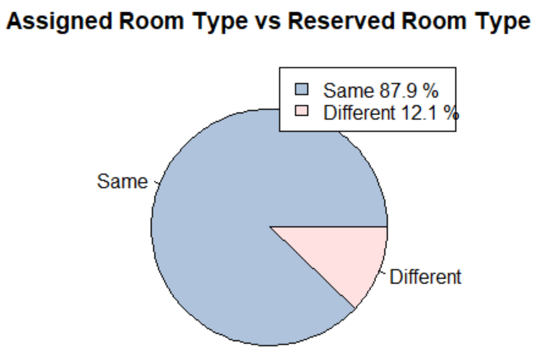
When we look at the countries of the people making hotel reservations, the majority of people staying in Resort hotels and City Hotels are Portuguese, while the second and third countries are the United Kingdom and France, respectively.

```
bar_colors <- c("lightsteelblue", "mistyrose", "thistle", "antiquewhite", "rosybrown")

channel_counts <- table(booking_clean$distribution_channel)

barplot(channel_counts, main = "Reservation Distribution by Distribution Channel",ylab =
"Count", xlab = "Distribution Channel", col = bar_colors,  ylim = c(0, max(channel_counts) *
1.2), beside = TRUE)

options(scipen = 999)
```



In the above graph, we can observe that 82% of customers make their bookings through travel agencies or tour operators (TA/TO).

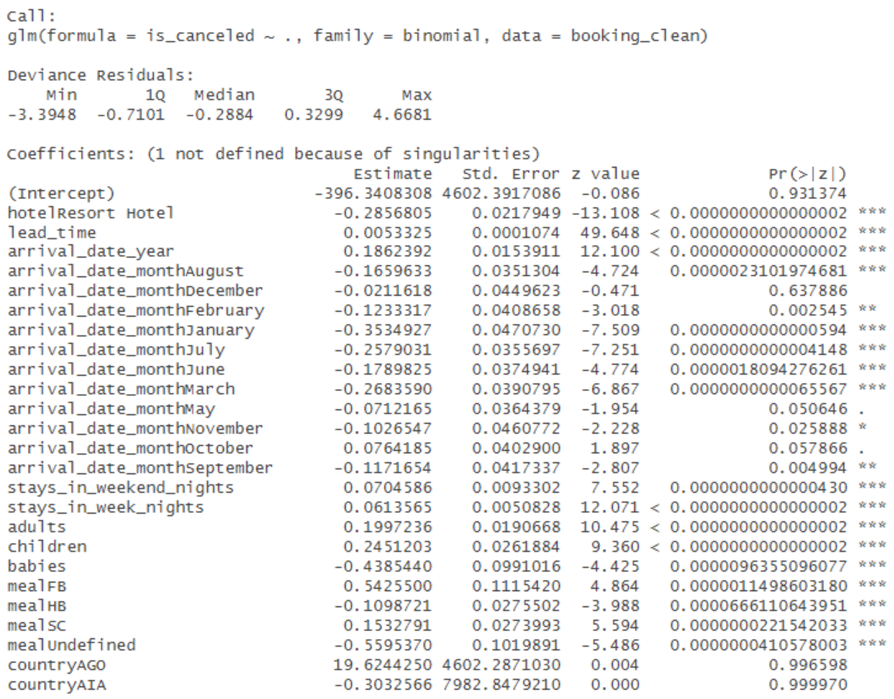
```
same_count <- sum(booking_clean$assigned_room_type ==
booking_clean$reserved_room_type)
different_count <- sum(booking_clean$assigned_room_type !=
booking_clean$reserved_room_type)
total <- nrow(booking_clean)

same_ratio <- same_count / total * 100
different_ratio <- different_count / total * 100

labels <- c("Same", "Different")
ratios <- c(same_ratio, different_ratio)
colors <- c("lightsteelblue", "mistyrose")
pie(ratios, labels = labels, col = colors, main = "Assigned Room Type vs Reserved Room Type")
legend_labels <- paste(labels, round(ratios, 1), "%")
legend("topright", legend = legend_labels, fill = colors)
```



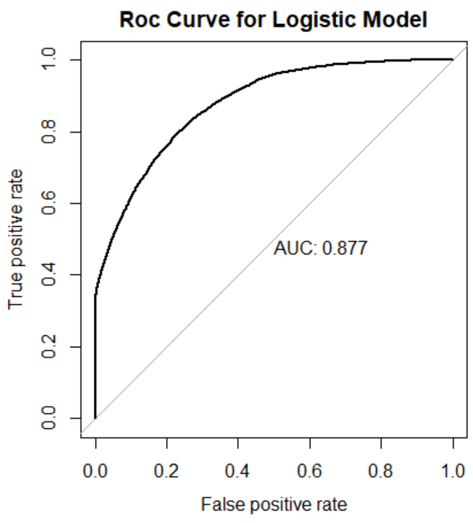
From the above pie chart, we can observe that 12.1% of the customers are assigned a room different from their request, and 5% of these customers have canceled their reservations.

Logistic Regression

```r
logistic_model <- glm(is_canceled~ . , data = booking_clean, family = binomial)

summary(logistic_model)

glm.prob_lm <- predict(logistic_model, type="response")
roc.out_lm <- roc(booking_clean$is_canceled, glm.prob_lm)

plot(roc.out_lm, print.auc=TRUE, legacy.axes=TRUE, xlab="False positive rate", ylab="True positive rate",main="Roc Curve for Logistic Model")
```



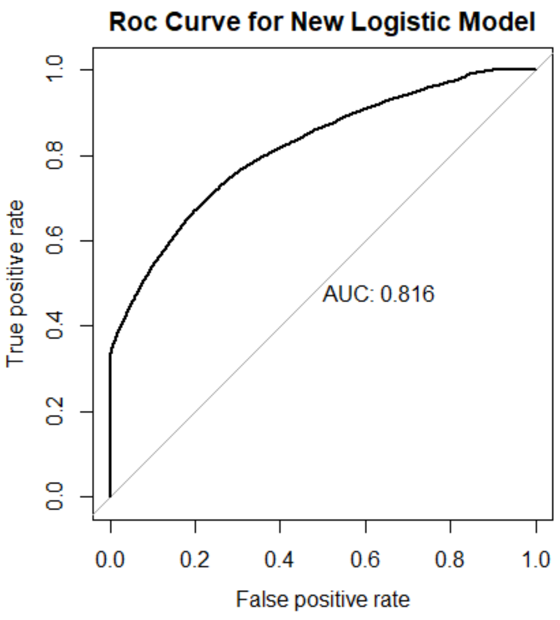
```
Call:
glm(formula = is_canceled ~ ., family = binomial, data = booking_clean)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.3948  -0.7101  -0.2884   0.3299   4.6681

Coefficients: (1 not defined because of singularities)
                                Estimate   Std. Error z value             Pr(>|z|)
(Intercept)                 -396.3408308 4602.3917086  -0.086             0.931374
hotelResort Hotel             -0.2856805    0.0217949 -13.108 < 0.0000000000000002 ***
lead_time                      0.0053325    0.0001074  49.648 < 0.0000000000000002 ***
arrival_date_year              0.1862392    0.0153911  12.100 < 0.0000000000000002 ***
arrival_date_monthAugust      -0.1659633    0.0351304  -4.724   0.0000023101974681 ***
arrival_date_monthDecember    -0.0211618    0.0449623  -0.471             0.637886
arrival_date_monthFebruary    -0.1233317    0.0408658  -3.018             0.002545 **
arrival_date_monthJanuary     -0.3534927    0.0470730  -7.509   0.0000000000000594 ***
arrival_date_monthJuly        -0.2579031    0.0355697  -7.251   0.0000000000004148 ***
arrival_date_monthJune        -0.1789825    0.0374941  -4.774   0.0000018094276261 ***
arrival_date_monthMarch       -0.2683590    0.0390795  -6.867   0.0000000000065567 ***
arrival_date_monthMay         -0.0712165    0.0364379  -1.954             0.050646 .
arrival_date_monthNovember    -0.1026547    0.0460772  -2.228             0.025888 *
arrival_date_monthOctober      0.0764185    0.0402900   1.897             0.057866 .
arrival_date_monthSeptember   -0.1171654    0.0417337  -2.807             0.004994 **
stays_in_weekend_nights        0.0704586    0.0093302   7.552   0.0000000000000430 ***
stays_in_week_nights           0.0613565    0.0050828  12.071 < 0.0000000000000002 ***
adults                         0.1997236    0.0190668  10.475 < 0.0000000000000002 ***
children                       0.2451203    0.0261884   9.360 < 0.0000000000000002 ***
babies                        -0.4385440    0.0991016  -4.425   0.0000096355096077 ***
mealFB                         0.5425500    0.1115420   4.864   0.0000011498603180 ***
mealHB                        -0.1098721    0.0275502  -3.988   0.0000666110643951 ***
mealSC                         0.1532791    0.0273993   5.594   0.0000000221542033 ***
mealUndefined                 -0.5595370    0.1019891  -5.486   0.0000000410578003 ***
countryAGO                    19.6244250 4602.2871030   0.004             0.996598
countryAIA                    -0.3032566 7982.8479210   0.000             0.999970
```



According to the logistic model, several factors influence the “is_canceled” variable. “Hotel”, “lead_time”, “arrival_date_year”, “adults”, “previous_cancellations”, and “booking_changes” are among the observed factors affecting reservation cancellation. But, according to the model “country” variable does not have an impact as we can see it Pr(>|z|) value is greater than 0.5. The model demonstrates high performance in predicting reservation cancellations using the specified independent variables. Furthermore, it was concluded that all variables except 'country' play a decisive role in reservation cancellation.

Logistic Regression without Country Variable

```
model <- glm(formula = is_canceled ~ hotel + lead_time + arrival_date_year +
             arrival_date_month + stays_in_weekend_nights + stays_in_week_nights + adults +
children + babies + meal + market_segment +
             distribution_channel + reserved_room_type +
             booking_changes + deposit_type + days_in_waiting_list + customer_type +
required_car_parking_spaces,
           family = binomial, data = booking_clean)

summary(model)

glm.prob <- predict(model, type="response")
roc.out <- roc(booking_clean$is_canceled, glm.prob)

plot(roc.out, print.auc=TRUE, legacy.axes=TRUE, xlab="False positive rate", ylab="True
positive rate", main= "Roc Curve for New Logistic Model")
```



A logistic model was created that included all variables except the "country" variable. Our goal is to determine whether the "country" variable, which does not appear to affect the model, does affect it.

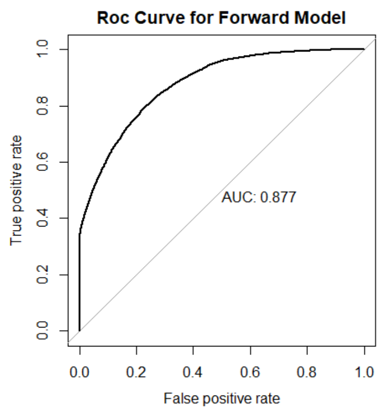
The effect of the variable on the model is examined by comparing the AIC values. When we compared the AIC values of the models, we observed that the AIC increased from 95765 to 109987. This means that the model has a worse fit and explains less information than the first model.

In addition, the Roc Curve graph shows an AUC value of 0.816. This means that the model is more effective in testing cancellations when it shows country variability.

```
Call:
glm(formula = is_canceled ~ hotel + lead_time + arrival_date_year +
    arrival_date_month + stays_in_weekend_nights + stays_in_week_nights +
    adults + children + babies + meal + market_segment + distribution_channel +
    reserved_room_type + booking_changes + deposit_type + days_in_waiting_list +
    customer_type + required_car_parking_spaces, family = binomial,
    data = booking_clean)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.3993  -0.8343  -0.5119   0.7238   3.7190

Coefficients:
                               Estimate Std. Error z value Pr(>|z|)
(Intercept)                   1.549e+02  2.782e+01   5.569 2.57e-08 ***
hotelResort Hotel            -1.241e-01  1.855e-02  -6.691 2.22e-11 ***
lead_time                     3.975e-03  9.448e-05  42.076  < 2e-16 ***
arrival_date_year            -7.780e-02  1.380e-02  -5.638 1.72e-08 ***
arrival_date_monthAugust      3.680e-02  3.212e-02   1.146 0.251993
arrival_date_monthDecember    2.441e-02  4.121e-02   0.592 0.553723
arrival_date_monthFebruary    5.170e-02  3.761e-02   1.374 0.169291
arrival_date_monthJanuary    -8.476e-02  4.288e-02  -1.976 0.048101 *
arrival_date_monthJuly       -2.243e-02  3.248e-02  -0.690 0.489894
arrival_date_monthJune       -6.722e-02  3.447e-02  -1.950 0.051158 .
arrival_date_monthMarch      -1.898e-01  3.590e-02  -5.287 1.24e-07 ***
arrival_date_monthMay        -6.029e-02  3.345e-02  -1.803 0.071458 .
arrival_date_monthNovember   -1.863e-01  4.215e-02  -4.420 9.89e-06 ***
arrival_date_monthOctober    -9.720e-02  3.721e-02  -2.612 0.008993 **
arrival_date_monthSeptember  -1.958e-01  3.809e-02  -5.140 2.75e-07 ***
stays_in_weekend_nights       3.711e-02  8.492e-03   4.370 1.24e-05 ***
stays_in_week_nights          4.035e-02  4.536e-03   8.896  < 2e-16 ***
adults                        1.060e-01  1.518e-02   6.983 2.89e-12 ***
children                      1.668e-01  2.383e-02   7.000 2.55e-12 ***
babies                       -1.850e-01  8.872e-02  -2.085 0.037041 *
mealFB                        8.744e-01  1.012e-01   8.640  < 2e-16 ***
```

Forward Logistic Regression

```
forward_model <- stepAIC(logistic_model, direction = "forward")

summary(forward_model)

glm.prob_f <- predict(forward_model, type="response")
roc.out_f <- roc(booking_clean$is_canceled, glm.prob_f)

plot(roc.out_f, print.auc=TRUE, legacy.axes=TRUE, xlab="False positive rate", ylab="True positive rate",main="Roc Curve for Forward Model")
```



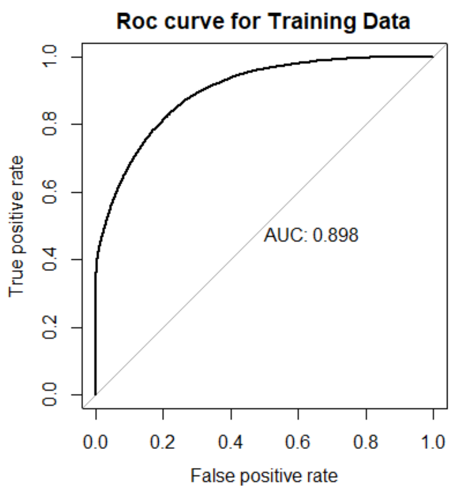
```
Start:  AIC=95984.39
is_canceled ~ hotel + lead_time + arrival_date_year + arrival_date_month +
    stays_in_weekend_nights + stays_in_week_nights + adults +
    children + babies + meal + country + market_segment + distribution_channel +
    reserved_room_type + assigned_room_type + booking_changes +
    deposit_type + days_in_waiting_list + customer_type + required_car_parking_spaces
```

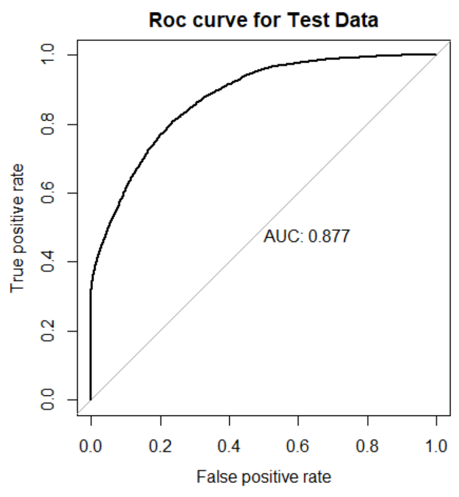
In the Forward Regression model, we achieved results similar to our logistic regression model that included all variables and AUC value was similar to the logistic regression model we first created.

Splitting Data into Training and Test Set

```
set.seed(123)
n <- nrow(booking_clean)
train_indices <- sample(1:n, size = 0.8*n, replace = FALSE)
train_data <- booking_clean[train_indices, ]
test_data <- booking_clean[-train_indices, ]

train_data$country <- factor(train_data$country)
test_data$country <- factor(test_data$country, levels = levels(train_data$country))

train_data$country <- droplevels(train_data$country)
test_data$country <- droplevels(test_data$country)

model2 <- glm(train_data$is_canceled ~ ., data = train_data, family = binomial)

glm.prob_train <- predict(model2, newdata = train_data, type = "response")

roc.out2 <- roc(train_data$is_canceled, glm.prob_train)

plot(roc.out2, print.auc=TRUE, legacy.axes=TRUE, xlab="False positive rate", ylab="True
positive rate", main= "Roc curve for Training Data")
```

```
glm.prob_test <- predict(model2, newdata = test_data, type = "response")

roc.out3 <- roc(test_data$is_canceled, glm.prob_test)

plot(roc.out3, print.auc=TRUE, legacy.axes=TRUE, xlab="False positive rate", ylab="True
positive rate", main= "Roc curve for Test Data")
```





For further tests, we divided the dataset into training and test sets, with 80% allocated to training data and 20% to test data. A logistic regression model was built using the training data, with "is_canceled" as the dependent variable and all independent variables included in the model. ROC curves were utilized to assess the model's fit and performance on both training and test datasets.

Upon evaluating these steps, the model's performance on both datasets was visualized and assessed using ROC curves and AUC values. These results indicate that the model performs well and effectively predicts hotel reservations, demonstrating good accuracy and generalization ability.

```
# Confusion matrix
conf_matrix_test <- table(test_data$is_canceled, glm.prob_test > 0.5)
# True Positives (TP)
TP_Test <- conf_matrix_test[2, 2]
# True Negatives (TN)
TN_Test <- conf_matrix_test[1, 1]
# False Positives (FP)
FP_Test <- conf_matrix_test[1, 2]
# False Negatives (FN)
FN_Test <- conf_matrix_test[2, 1]
# Accuracy
accuracy_Test <- (TP_Test + TN_Test) / sum(conf_matrix_test)
# Precision
precision_Test <- TP_Test / (TP_Test + FP_Test)
# Recall
recall_Test <- TP_Test / (TP_Test + FN_Test)
# F1-Score
f1_score_Test <- 2 * (precision_Test * recall_Test) / (precision_Test + recall_Test)

# Print the metrics
print(paste("Accuracy for Test :", accuracy_Test))

## [1] "Accuracy for Test : 0.792829868476877"

print(paste("Precision Test:", precision_Test))

## [1] "Precision Test: 0.783744131455399"

print(paste("Recall for Test:", recall_Test))

## [1] "Recall for Test: 0.610444520626214"

print(paste("F1-Score for Test:", f1_score_Test))
```

```
## [1] "F1-Score for Test: 0.686323633326909"
```

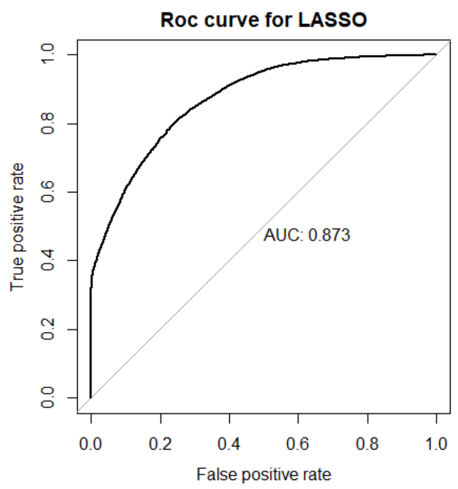
The logistic regression model's performance in predicting canceled reservations within the hotel dataset was evaluated using training and testing data. Performance metrics on test data are explained in detail below:

- Accuracy: The model achieved an accuracy of 79.28%, indicating the proportion of correctly predicted samples out of the total.
- Precision: The precision was 78.37%, meaning 78.37% of predicted cancellations were correct.
- Sensitivity (Recall): The model achieved a sensitivity of 61.04%, correctly identifying 61.04% of actual cancellations.
- F1-Score: The F1-Score was 68.63%, providing a balanced measure of the model's accuracy and recall.

LASSO Regression

```
X <- booking_clean[, !names(booking_clean) %in% "is_canceled"]
y <- booking_clean$is_canceled
X$hotel <- as.factor(X$hotel)
X$meal <- as.factor(X$meal)
X$country <- as.factor(X$country)
lasso_model <- glmnet(X, y, alpha = 1)

X1 <- model.matrix(~ . - 1, data = X)
lasso_model1 <- glmnet(X1, y, alpha = 1)
# Cross-validation choose lambda
cv_fit <- cv.glmnet(X1, y, alpha = 1)
best_lambda <- cv_fit$lambda.min
lasso_model_best <- glmnet(X1, y, alpha = 1, lambda = best_lambda)

predictions <- predict(lasso_model_best, newx = X1, s = best_lambda, type = "response")
roc.outp <- roc(y, predictions)

plot(roc.outp, print.auc=TRUE, legacy.axes=TRUE, xlab="False positive rate", ylab="True
positive rate", main= "Roc curve for LASSO")

# Confusion matrix
conf_matrix_ <- table(y, predictions > 0.5)
# True Positives (TP)
TP_ <- conf_matrix_[2, 2]
# True Negatives (TN)
TN_ <- conf_matrix_[1, 1]
# False Positives (FP)
FP_ <- conf_matrix_[1, 2]
# False Negatives (FN)
FN_ <- conf_matrix_[2, 1]
```

```
# Accuracy
accuracy_ <- (TP_ + TN_) / sum(conf_matrix_)
# Precision
precision_ <- TP_ / (TP_ + FP_)
# Recall
recall_ <- TP_ / (TP_ + FN_)
# F1-Score
f1_score_ <- 2 * (precision_ * recall_) / (precision_ + recall_)

# Print the metrics
print(paste("Accuracy for Lasso :", accuracy_))

## [1] "Accuracy for Lasso : 0.785398549 18763"

print(paste("Precision for Lasso:", precision_))

## [1] "Precision for Lasso: 0.819798600443762"

print(paste("Recall for Lasso:", recall_))

## [1] "Recall for Lasso: 0.545433898843996"

print(paste("F1-Score for Lasso:", f1_score_))

## [1] "F1-Score for Lasso: 0.65504732291411"
```



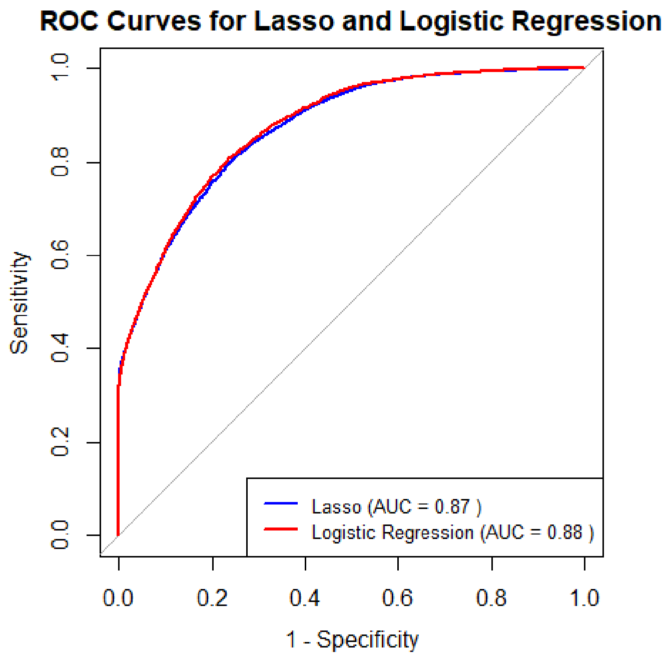
Finally, we applied the LASSO regression model to our dataset. During model creation, categorical variables were appropriately transformed and included in the model matrix. The optimal lambda value was determined using cross-validation.

The model's performance was assessed using various metrics. We achieved an accuracy of 78.54%, a precision of 81.98%, a recall of 54.54%, and an F1-Score of 65.50%. The ROC curve analysis yielded an AUC value of 0.873, indicating excellent classification performance.

Overall, these results demonstrate that the model performs well, particularly in accurately predicting positive outcomes, as evidenced by high precision and AUC values.

Conclusion

```r
predictions_lasso <- predict(lasso_model_best, newx = X1, s = best_lambda, type = "response")
predictions_logit <- predict(model2, newdata = test_data, type = "response")

roc_lasso <- roc(y, predictions_lasso)

roc_logit <- roc(test_data$is_canceled, predictions_logit)

plot(roc_lasso, col = "blue", legacy.axes = TRUE, print.auc = FALSE,
    main = "ROC Curves for Lasso and Logistic Regression", lwd = 2)
plot(roc_logit, col = "red", legacy.axes = TRUE, print.auc = FALSE, add = TRUE, lwd = 2)
legend("bottomright", legend = c(paste("Lasso (AUC =", round(auc(roc_lasso), 2), ")"),
                        paste("Logistic Regression (AUC =", round(auc(roc_logit), 2), ")")),
      col = c("blue", "red"), lwd = 2, cex = 0.8)
```



In this study, we employed two models, namely the Logistic Regression Model and the LASSO Regression Model, to predict hotel reservation cancellations. Our evaluation encompassed key metrics such as Accuracy, Precision, Recall, and F1-Score. The findings indicate that the Logistic Regression model outperformed the LASSO model with a slightly higher accuracy rate. Specifically, the Logistic Regression model demonstrated superior Recall values, showcasing its effectiveness in correctly identifying canceled reservations. Moreover, the higher F1-Score achieved by the Logistic Regression model highlights its better balance between Precision and Recall, underscoring its overall stronger performance in this predictive task.

	Logistic Regression	Lasso
Accuracy	0.79282	0.7854
Precision	0.78374	0.8198
Recall	0.61044	0.5454
F1-Score	0.68632	0.6550