Predict Hotel Bookings Cancellations

Bharatwaj Majji
UB PERSON ID: 50442312
Jayanth Puthineedi
UB PERSON ID: 50442725
Vishnu Bhadramraju
UB PERSON ID: 50441735

2022-12-09

1. "Visualize And Understand the dataset"

```r
set.seed(1) # seed for any random generation
df = read.csv('hotel_bookings.csv')
head(df)
```

```
##          hotel is_canceled lead_time arrival_date_year arrival_date_month
## 1 Resort Hotel           0       342              2015               July
## 2 Resort Hotel           0       737              2015               July
## 3 Resort Hotel           0         7              2015               July
## 4 Resort Hotel           0        13              2015               July
## 5 Resort Hotel           0        14              2015               July
## 6 Resort Hotel           0        14              2015               July
##   arrival_date_week_number arrival_date_day_of_month stays_in_weekend_nights
## 1                       27                         1                       0
## 2                       27                         1                       0
## 3                       27                         1                       0
## 4                       27                         1                       0
## 5                       27                         1                       0
## 6                       27                         1                       0
##   stays_in_week_nights adults children babies meal country market_segment
## 1                    0      2        0      0   BB     PRT         Direct
## 2                    0      2        0      0   BB     PRT         Direct
## 3                    1      1        0      0   BB     GBR         Direct
## 4                    1      1        0      0   BB     GBR      Corporate
## 5                    2      2        0      0   BB     GBR      Online TA
## 6                    2      2        0      0   BB     GBR      Online TA
##   distribution_channel is_repeated_guest previous_cancellations
## 1               Direct                 0                      0
## 2               Direct                 0                      0
## 3               Direct                 0                      0
## 4            Corporate                 0                      0
## 5                TA/TO                 0                      0
## 6                TA/TO                 0                      0
```

```
##   previous_bookings_not_canceled reserved_room_type assigned_room_type
## 1                              0                  C                  C
## 2                              0                  C                  C
## 3                              0                  A                  C
## 4                              0                  A                  A
## 5                              0                  A                  A
## 6                              0                  A                  A
##   booking_changes deposit_type agent company days_in_waiting_list customer_type
## 1               3   No Deposit  NULL    NULL                    0     Transient
## 2               4   No Deposit  NULL    NULL                    0     Transient
## 3               0   No Deposit  NULL    NULL                    0     Transient
## 4               0   No Deposit   304    NULL                    0     Transient
## 5               0   No Deposit   240    NULL                    0     Transient
## 6               0   No Deposit   240    NULL                    0     Transient
##   adr required_car_parking_spaces total_of_special_requests reservation_status
## 1   0                           0                         0          Check-Out
## 2   0                           0                         0          Check-Out
## 3  75                           0                         0          Check-Out
## 4  75                           0                         0          Check-Out
## 5  98                           0                         1          Check-Out
## 6  98                           0                         1          Check-Out
##   reservation_status_date
## 1              2015-07-01
## 2              2015-07-01
## 3              2015-07-02
## 4              2015-07-02
## 5              2015-07-03
## 6              2015-07-03
```

```r
dim(df)
```

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

Columns with MissingValues

```r
cat("Columns with NA values - ", names(which(sapply(df, function(x) any(is.na(x))))), "\n")
```

```
## Columns with NA values -  children
```

```r
cat("Columns with NULL values - ", names(which(sapply(df, function(x) any(x=='NULL')))), "\n")
```

```
## Columns with NULL values -  country agent company
```

Handle Missing Values columns

```r
df$children = ifelse(is.na(df$children), 0, df$children)
df$country = ifelse(df$country == 'NULL', 'Unknown', df$country)
df$agent = ifelse(df$agent == 'NULL', 0, df$agent)
df$company = ifelse(df$company == 'NULL', 0, df$company)
df$guests_stayed = df$adults + df$children + df$babies
df$nights_stayed = df$stays_in_week_nights + df$stays_in_weekend_nights
df <- subset(df, select = -c(adults, children, babies, stays_in_week_nights, stays_in_weekend_nights))
```

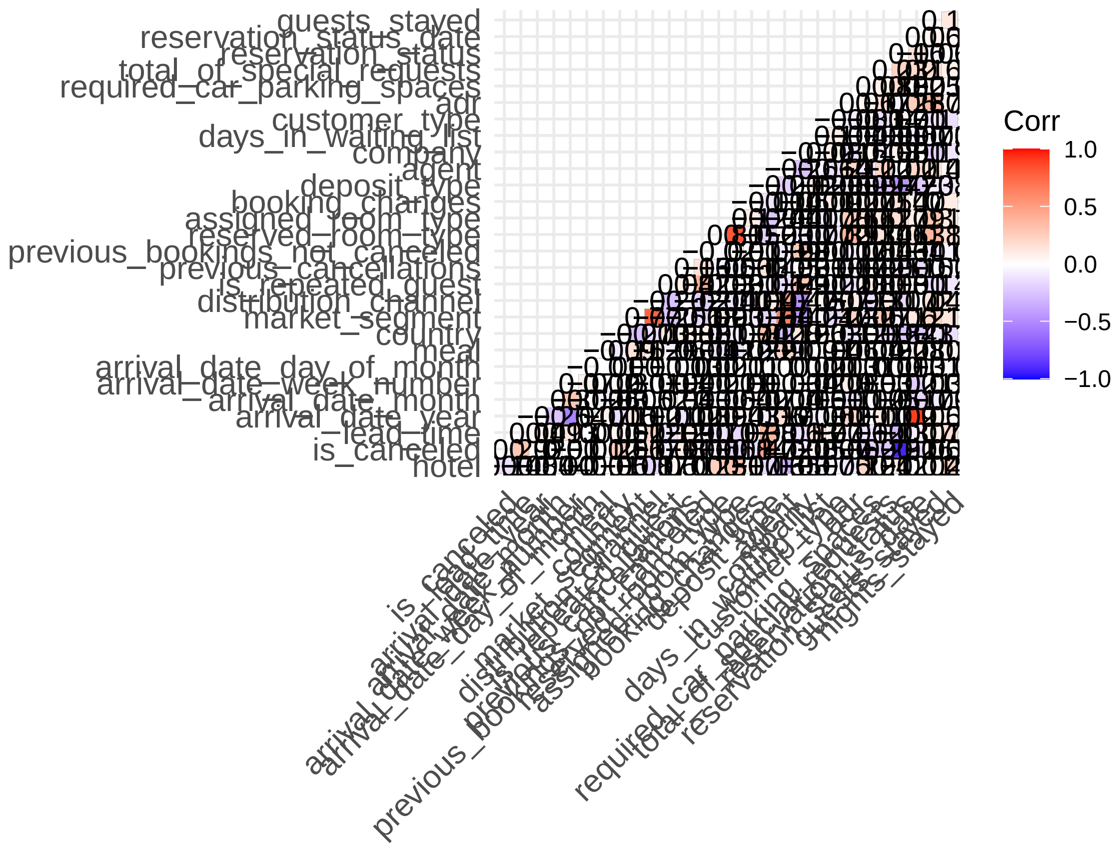
Correlation b/w numerical variables

```
res <- cor(df[sapply(df,is.numeric)])
res[order(res[,1],decreasing=TRUE), ncol=1]
```

```
##                    is_canceled                       lead_time
##                    1.000000000                     0.293123356
##         previous_cancellations            days_in_waiting_list
##                    0.110132808                     0.054185824
##                            adr                    guests_stayed
##                    0.047556598                     0.046521756
##                  nights_stayed               arrival_date_year
##                    0.017779269                     0.016659860
##       arrival_date_week_number       arrival_date_day_of_month
##                    0.008148065                    -0.006130079
## previous_bookings_not_canceled               is_repeated_guest
##                   -0.057357723                    -0.084793418
##                booking_changes     required_car_parking_spaces
##                   -0.144380991                    -0.195497817
##      total_of_special_requests
##                   -0.234657774
```

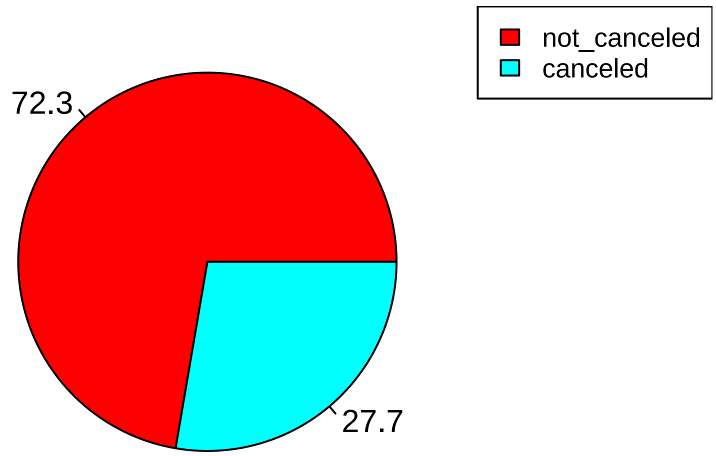
From, this we can understand that lead_time, previous_cancellations has strongest correlations > 0.1 while total_of_special_requests, required_car_parking_spaces, booking_changes has least correlations.

```
cor<- cor(data.matrix(df))
ggcorrplot(cor, lab=TRUE, type='lower')
```

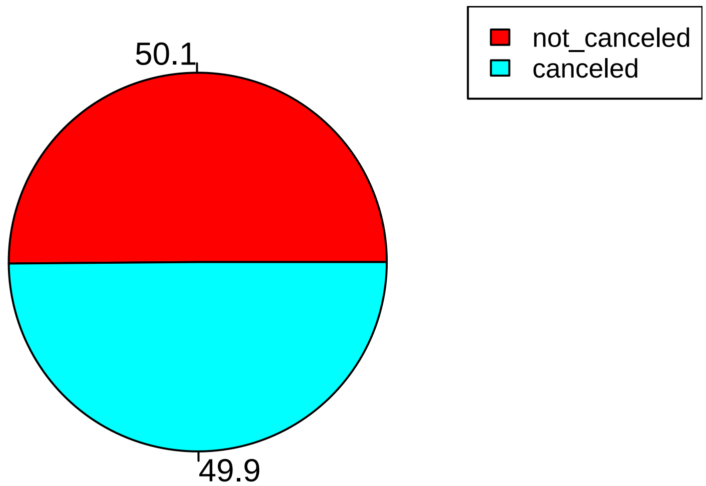


Lead Time vs Cancellations

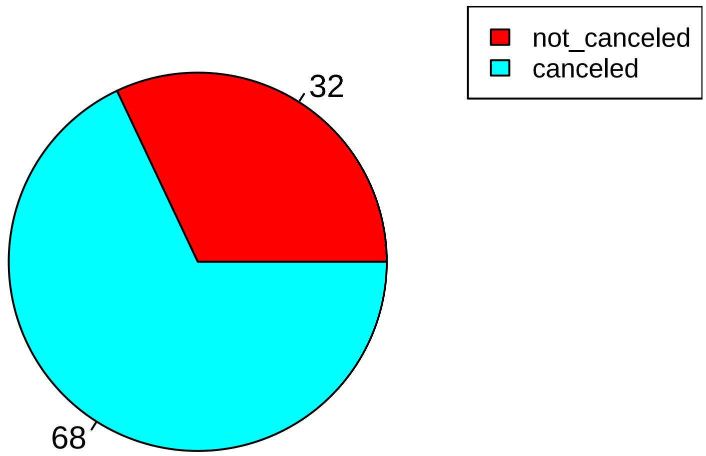
```
lead_100 = ddply(filter(df, lead_time<100), .(is_canceled), nrow)
piepercent<- round(100*lead_100$V1/sum(lead_100$V1), 1)
pie(x=lead_100$V1, labels=piepercent, col=rainbow(length(lead_100$V1)))
legend('topright', c('not_canceled', 'canceled'), cex = 0.8, fill=rainbow(length(lead_100$V1)))
```




```
lead_365 = ddply(filter(df, lead_time>=100 & lead_time < 365), .(is_canceled), nrow) #less than an year
piepercent<- round(100*lead_365$V1/sum(lead_365$V1), 1)
pie(x=lead_365$V1, labels=piepercent, col=rainbow(length(lead_365$V1)))
legend('topright', c('not_canceled', 'canceled'), cex = 0.8, fill=rainbow(length(lead_365$V1)))
```




```
lead_365_gt = ddply(filter(df, lead_time >= 365), .(is_canceled), nrow)
piepercent<- round(100*lead_365_gt$V1/sum(lead_365_gt$V1), 1)
pie(x=lead_365_gt$V1, labels=piepercent, col=rainbow(length(lead_365_gt$V1)))
legend('topright', c('not_canceled', 'canceled'), cex = 0.8, fill=rainbow(length(lead_365_gt$V1)))
```



From this we can understand that as lead_time increases the chances of booking cancellation as well increases.

Previous Cancelations vs Cancelations

```
cat("Never previously cancelled -", mean(filter(df, previous_cancellations==0)$is_canceled), "%", "\n")
```

```
## Never previously cancelled - 0.3390608 %
```

```
cat("Previously cancelled once -", mean(filter(df, previous_cancellations==1)$is_canceled), "%", "\n")
```

```
## Previously cancelled once - 0.9443067 %
```

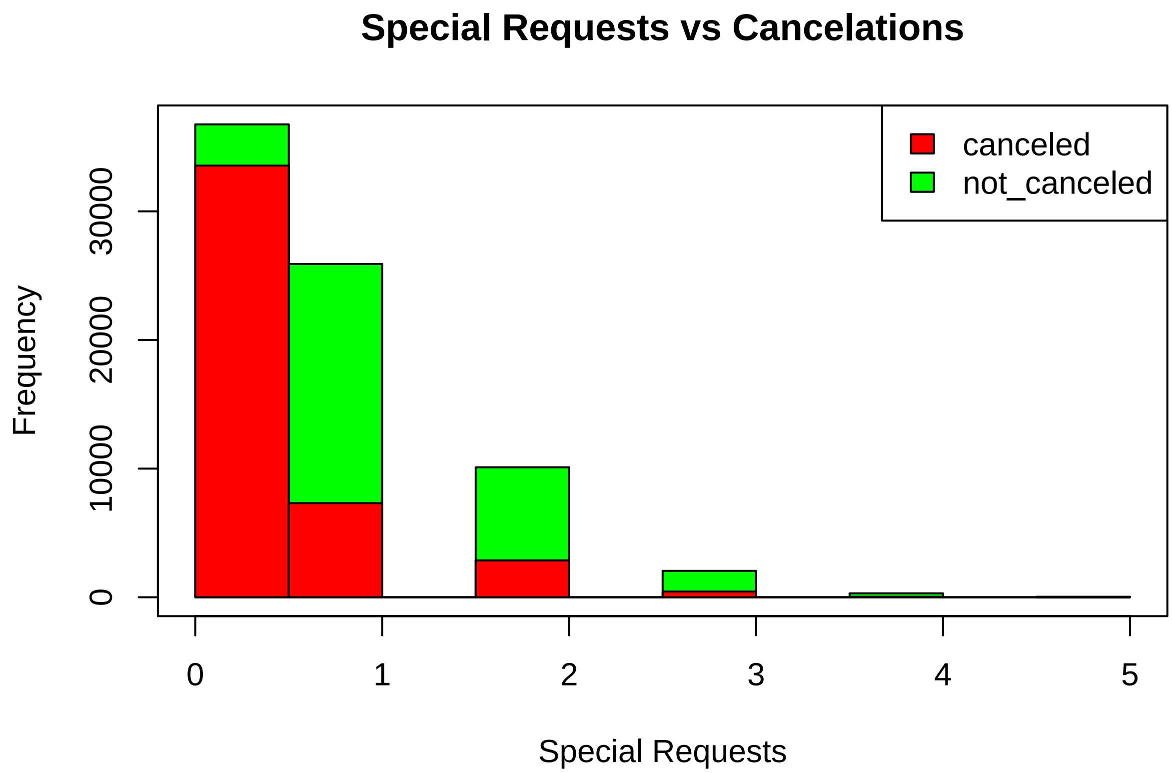
```
cat("Previously cancelled more than 11 -", mean(filter(df, previous_cancellations>11)$is_canceled), "%"
```

```
## Previously cancelled more than 11 - 0.9931034 %
```

As the number of previous cancelations increases the chances of booking cancelations as well increases.

Special Requests vs Cancelations

```
hist(main='Special Requests vs Cancelations', xlab='Special Requests', filter(df, is_canceled==0)$total_
hist(filter(df, is_canceled==1)$total_of_special_requests,col="red",pch=20,cex=4,breaks=15,add=TRUE)
legend("topright", c("canceled", "not_canceled"), fill=c("red", "green"))
box()
```



From the above graph we can understand that as the number of special requests increases the booking cancelation percentage decreases.

Parking Spaces vs Cancelations

```
print("Parking spaces for not canceled bookings - ")
```

```
## [1] "Parking spaces for not canceled bookings - "
```

```
ddply(filter(df, is_canceled==0), .(required_car_parking_spaces), nrow)
```

```
##   required_car_parking_spaces    V1
## 1                           0 67750
## 2                           1  7383
## 3                           2    28
## 4                           3     3
## 5                           8     2
```

```
print("Parking spaces for canceled bookings - ")
```

```
## [1] "Parking spaces for canceled bookings - "
```

```
ddply(filter(df, is_canceled==1), .(required_car_parking_spaces), nrow)
```

```
##   required_car_parking_spaces    V1
## 1                           0 44224
```

From this we can understand the model can tune in such a way that if the number of required spaces is zero the booking can be canceled which is not the case ideally. So, we can ignore this feature while modeling.

Categorical variables

Hotel vs Cancelations

```
ordered_months <- c("January", "February", "March", "April", "May", "June",
          "July", "August", "September", "October", "November", "December")

city_0 <- ddply(filter(df, is_canceled==0 & hotel=='City Hotel'), .(arrival_date_month), nrow)
city_1 <- ddply(filter(df, is_canceled==1 & hotel=='City Hotel'), .(arrival_date_month), nrow)

resort_0 <- ddply(filter(df, is_canceled==0 & hotel=='Resort Hotel'), .(arrival_date_month), nrow)
resort_1 <- ddply(filter(df, is_canceled==1 & hotel=='Resort Hotel'), .(arrival_date_month), nrow)

resort_cancel <- rep()
city_cancel <- rep()

for (month in ordered_months) {
  resort_0_mon <- filter(resort_0, arrival_date_month==month)
  resort_1_mon <- filter(resort_1, arrival_date_month==month)
  resort_cancel <- append(resort_cancel, resort_1_mon[1, "V1"]/(resort_1_mon[1, "V1"]+resort_0_mon[1, "V

  city_0_mon <- filter(city_0, arrival_date_month==month)
  city_1_mon <- filter(city_1, arrival_date_month==month)
  city_cancel <- append(city_cancel, city_1_mon[1, "V1"]/(city_1_mon[1, "V1"]+city_0_mon[1, "V1"]))
}
result <- data.frame(resort_cancel=resort_cancel, city_cancel=city_cancel, row.names=ordered_months)
result
```

```
##           resort_cancel city_cancel
## January       0.1481988   0.3966809
## February      0.2562037   0.3828802
## March         0.2287170   0.3694642
## April         0.2934331   0.4632353
## May           0.2877213   0.4437561
## June          0.3307061   0.4469217
## July          0.3140171   0.4087537
## August        0.3344912   0.4009796
## September     0.3236808   0.4202703
## October       0.2751055   0.4297173
## November      0.1891670   0.3812256
## December      0.2382931   0.4211036
```

From the above stats we can understand that wrt month city hotels has more booking cancelations compared to resort hotels according to arrival months.

Meal vs Cancelations

```
cancelled_meal <- ddply(filter(df, is_canceled==1), .(meal), nrow)
uncancelled_meal <- ddply(filter(df, is_canceled==0), .(meal), nrow)
percent <- rep()
for (val in cancelled_meal$meal) {
  cancel_val <- filter(cancelled_meal, meal==val)[1, "V1"]
```

```r
  uncancel_val <- filter(uncancelled_meal, meal==val)[1, "V1"]
  percent <- append(percent, cancel_val/(cancel_val+uncancel_val))
}
result <- data.frame(meal=cancelled_meal$meal, percent_cancellations=percent)
result
```

```
##        meal percent_cancellations
## 1        BB             0.3738490
## 2        FB             0.5989975
## 3        HB             0.3446035
## 4        SC             0.3723944
## 5 Undefined             0.2446536
```

From this we can understand that FB meal is the most frequently canceled booking. And meal Undefined can relate to SC no-meal.

MarketSegment vs Cancelations

```r
cancelled_market <- ddply(filter(df, is_canceled==1), .(market_segment), nrow)
uncancelled_market <- ddply(filter(df, is_canceled==0), .(market_segment), nrow)
percent <- rep()
for (val in cancelled_market$market_segment) {
  cancel_val <- filter(cancelled_market, market_segment==val)[1, "V1"]
  uncancel_val <- filter(uncancelled_market, market_segment==val)[1, "V1"]
  percent <- append(percent, cancel_val/(cancel_val+uncancel_val))
}
result <- data.frame(market_segment=cancelled_market$market_segment, percent_cancellations=percent)
result
```

```
##   market_segment percent_cancellations
## 1       Aviation             0.2194093
## 2  Complementary             0.1305518
## 3      Corporate             0.1873466
## 4         Direct             0.1534190
## 5         Groups             0.6106204
## 6  Offline TA/TO             0.3431603
## 7      Online TA             0.3672114
## 8      Undefined                    NA
```

From the above stats we can understand that cancellations are higher for Groups, Offline and Online TA/TO travel and tour operator bookings

DistributionChannel vs Cancelations

```r
cancelled_channel <- ddply(filter(df, is_canceled==1), .(distribution_channel), nrow)
uncancelled_channel <- ddply(filter(df, is_canceled==0), .(distribution_channel), nrow)
percent <- rep()
for (val in cancelled_channel$distribution_channel) {
  cancel_val <- filter(cancelled_channel, distribution_channel==val)[1, "V1"]
  uncancel_val <- filter(uncancelled_channel, distribution_channel==val)[1, "V1"]
  percent <- append(percent, cancel_val/(cancel_val+uncancel_val))
}
result <- data.frame(distribution_channel=cancelled_channel$distribution_channel, percent_cancellations=
result
```

```
##   distribution_channel percent_cancellations
## 1            Corporate             0.2207578
## 2               Direct             0.1745988
## 3                  GDS             0.1917098
## 4                TA/TO             0.4102585
## 5            Undefined             0.8000000
```

From the above stats we can understand that cancellations are higher for TA/TO travel and tour operator bookings .

CustomerType vs Cancelations

```r
cancelled_cust <- ddply(filter(df, is_canceled==1), .(customer_type), nrow)
uncancelled_cust <- ddply(filter(df, is_canceled==0), .(customer_type), nrow)
percent <- rep()
for (val in cancelled_cust$customer_type) {
  cancel_val <- filter(cancelled_cust, customer_type==val)[1, "V1"]
  uncancel_val <- filter(uncancelled_cust, customer_type==val)[1, "V1"]
  percent <- append(percent, cancel_val/(cancel_val+uncancel_val))
}
result <- data.frame(customer_type=cancelled_cust$customer_type, percent_cancellations=percent)
result
```

```
##     customer_type percent_cancellations
## 1        Contract             0.3096173
## 2           Group             0.1022530
## 3       Transient             0.4074632
## 4 Transient-Party             0.2542987
```

From the above stats we can understand that cancellations are higher for Transient customer_type bookings.

DepositType vs Cancelations

```r
cancelled_deposit <- ddply(filter(df, is_canceled==1), .(deposit_type), nrow)
uncancelled_deposit <- ddply(filter(df, is_canceled==0), .(deposit_type), nrow)
percent <- rep()
for (val in cancelled_deposit$deposit_type) {
  cancel_val <- filter(cancelled_deposit, deposit_type==val)[1, "V1"]
  uncancel_val <- filter(uncancelled_deposit, deposit_type==val)[1, "V1"]
  percent <- append(percent, cancel_val/(cancel_val+uncancel_val))
}
result <- data.frame(deposit_type=cancelled_deposit$deposit_type, percent_cancellations=percent)
result
```

```
##   deposit_type percent_cancellations
## 1   No Deposit             0.2837702
## 2   Non Refund             0.9936245
## 3   Refundable             0.2222222
```

From the above we can see that non-refund bookings has 99 percent cancelations which is weird since ideally non-refund transactions tend to have lower cancelations. Looks like the values of cancelled and not-cancelled must have swapped up for non-refund transactions. Let us check this while modeling.

2. "Data Cleaning"

Remove rows with zero guests

```
df <- filter(df, guests_stayed>0)
```

Drop irrelevant columns

```
df <- subset(df, select = -c(agent, company, booking_changes, arrival_date_day_of_month, arrival_date_ye
df <- subset(df, select = -c(reservation_status, reservation_status_date, assigned_room_type, country) )
```

Numerical Columns: - agent & company => These columns are uninformative since they contain discrete codes for the agents and company using which the booking is made. - booking_changes => Could constantly change over time and has no much effect on the predictor. - arrival_date_day_of_month & arrival_date_year => Prevents the model from generalizing, since we have arrival_week information that would be sufficient.

Categorical Columns: - reservation_status => It has values Check-Out, Cancelled and No-Show which means not-canceled and canceled considering this feature can cause the model to overfit. - reservation_status_date => Date when the reservation_status is last changed this is not relevant. - assigned_room_type => This is irrelevant and more over reserved_room_type makes more sense since the booking can be canceled only before checking-in which means room is assigned. - country => There are many countries and not uniformly distributed so there are higher chances that this model can prevent the model from generalising.

Replace value of column

```
df["meal"][df["meal"] == "Undefined"] <- "SC"
```

Encode categorical data

```
df$hotel <- as.numeric(as.factor(df$hotel))  # Convert categories to numbers
df$arrival_date_month <- as.numeric(as.factor(df$arrival_date_month))
df$meal <- as.numeric(as.factor(df$meal))
df$market_segment <- as.numeric(as.factor(df$market_segment))
df$distribution_channel <- as.numeric(as.factor(df$distribution_channel))
df$reserved_room_type <- as.numeric(as.factor(df$reserved_room_type))
df$deposit_type <- as.numeric(as.factor(df$deposit_type))
df$customer_type <- as.numeric(as.factor(df$customer_type))
```

Scale the dataset

```
df$lead_time <- scale(df$lead_time)
df$adr <- scale(df$adr)
```

Divide the dataset into test and train

```
head(df)
```

```
##   hotel is_canceled  lead_time arrival_date_month arrival_date_week_number meal
## 1     2           0  2.2258692                  6                       27    1
## 2     2           0  5.9217601                  6                       27    1
```

```
## 3      2           0 -0.9086205                  6                       27    1
## 4      2           0 -0.8524804                  6                       27    1
## 5      2           0 -0.8431237                  6                       27    1
## 6      2           0 -0.8431237                  6                       27    1
##   market_segment distribution_channel is_repeated_guest previous_cancellations
## 1              4                    2                 0                      0
## 2              4                    2                 0                      0
## 3              4                    2                 0                      0
## 4              3                    1                 0                      0
## 5              7                    4                 0                      0
## 6              7                    4                 0                      0
##   previous_bookings_not_canceled reserved_room_type deposit_type
## 1                              0                  3            1
## 2                              0                  3            1
## 3                              0                  1            1
## 4                              0                  1            1
## 5                              0                  1            1
## 6                              0                  1            1
##   days_in_waiting_list customer_type         adr required_car_parking_spaces
## 1                    0             3 -2.02183206                           0
## 2                    0             3 -2.02183206                           0
## 3                    0             3 -0.53474022                           0
## 4                    0             3 -0.53474022                           0
## 5                    0             3 -0.07869872                           0
## 6                    0             3 -0.07869872                           0
##   total_of_special_requests guests_stayed nights_stayed
## 1                         0             2             0
## 2                         0             2             0
## 3                         0             1             1
## 4                         0             1             1
## 5                         1             2             2
## 6                         1             2             2
```

```
idx <- sample(nrow(df), nrow(df)*0.3)
test <- df[idx,]
train <-df[-idx,]
```

3. "Data Modeling"

Logistic Regression

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

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

```
summary(log_classifier)
```

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

```
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -8.4904  -0.7972  -0.4836   0.3275   5.7784
##
## Coefficients:
##                                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    -7.382e+00  1.401e-01 -52.693  < 2e-16 ***
## hotel                          -8.204e-02  2.117e-02  -3.876 0.000106 ***
## lead_time                       2.991e-01  1.083e-02  27.620  < 2e-16 ***
## arrival_date_month             -1.812e-02  2.690e-03  -6.738 1.61e-11 ***
## arrival_date_week_number       -2.947e-03  7.209e-04  -4.089 4.34e-05 ***
## meal                            3.749e-04  8.226e-03   0.046 0.963651
## market_segment                  5.763e-01  1.471e-02  39.191  < 2e-16 ***
## distribution_channel           -3.162e-01  1.937e-02 -16.320  < 2e-16 ***
## is_repeated_guest              -5.649e-01  9.796e-02  -5.767 8.07e-09 ***
## previous_cancellations          2.899e+00  7.170e-02  40.431  < 2e-16 ***
## previous_bookings_not_canceled -4.848e-01  2.965e-02 -16.349  < 2e-16 ***
## reserved_room_type             -2.307e-03  6.427e-03  -0.359 0.719623
## deposit_type                    4.633e+00  7.676e-02  60.352  < 2e-16 ***
## days_in_waiting_list           -1.129e-03  5.600e-04  -2.017 0.043725 *
## customer_type                  -7.467e-02  1.673e-02  -4.462 8.11e-06 ***
## adr                             2.461e-01  1.163e-02  21.166  < 2e-16 ***
## required_car_parking_spaces    -6.266e+02  9.159e+05  -0.001 0.999454
## total_of_special_requests      -6.297e-01  1.297e-02 -48.530  < 2e-16 ***
## guests_stayed                   1.328e-01  1.559e-02   8.516  < 2e-16 ***
## nights_stayed                   3.433e-02  3.617e-03   9.493  < 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: 110021  on 83446  degrees of freedom
## Residual deviance:  75913  on 83427  degrees of freedom
## AIC: 75953
##
## Number of Fisher Scoring iterations: 12
```

```
prob_pred = predict(log_classifier, train, type='response')
y_pred = ifelse(prob_pred > 0.5, 1, 0)
cm=table(y_pred, train$is_canceled)
cat("Prediction vs Actual table for Train Logistic Regression below -", "\n")
```

```
## Prediction vs Actual table for Train Logistic Regression below -
```

```
print(cm)
```

```
##
## y_pred     0     1
##      0 50135 14717
##      1  2394 16201
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train error rate for KNN -", mean(y_pred != train$is_canceled), "\n")
```

```
## Train error rate for KNN - 0.2050523
```

```r
cat("Train Accuracy for Logistic Regression -", accuracy, "\n")
```

```
## Train Accuracy for Logistic Regression - 0.7949477
```

```r
cat("----------------------------------------------------------------", "\n")
```

```
## ----------------------------------------------------------------
```

```r
prob_pred = predict(log_classifier, test, type='response')
y_pred = ifelse(prob_pred > 0.5, 1, 0)

# Making the Confusion Matrix
cat("Prediction vs Actual table for Test Logistic Regression below -", "\n")
```

```
## Prediction vs Actual table for Test Logistic Regression below -
```

```r
cm=table(y_pred, test$is_canceled)
print(cm)
```

```
##
## y_pred     0     1
##      0 21417  6331
##      1  1065  6950
```

```r
cat("Test error rate for Logistic Regression -", mean(y_pred != test$is_canceled), "\n")
```

```
## Test error rate for Logistic Regression - 0.2068059
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for Logistic Regression -", accuracy, "\n")
```

```
## Test Accuracy for Logistic Regression - 0.7931941
```

Cross Validation for Logistic Regression

```r
knitr::opts_chunk$set(warning = FALSE, message = FALSE)
folds = createFolds(train$is_canceled, k = 10)
cv = lapply(folds, function(x) {
  training_fold = train[-x, ]
  test_fold = train[x, ]
  log_classifier = glm(formula=is_canceled ~ ., family=binomial, data=training_fold)
  prob_pred = predict(log_classifier, test_fold, type='response')
  y_pred = ifelse(prob_pred > 0.5, 1, 0)
  cm=table(y_pred, test_fold$is_canceled)
  accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
  return(accuracy)
})
```

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

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

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

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

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

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

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

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

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

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

```
accuracy = mean(as.numeric(cv))
cat("Test Accuracy for Logistic Regression CV -", accuracy)
```

```
## Test Accuracy for Logistic Regression CV - 0.7949596
```

KNN

```
y_pred = knn(train=subset(train, select = -c(is_canceled)),
             test=subset(train, select = -c(is_canceled)),
             cl=train$is_canceled,
             k = 5,
             prob = TRUE)
cm <- table(train$is_canceled, y_pred)
cat("Prediction vs Actual table for Train KNN below -", "\n")
```

```
## Prediction vs Actual table for Train KNN below -
```

```
print(cm)
```

```
##    y_pred
##         0     1
##   0 48072  4457
##   1  7293 23625
```

```
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train Accuracy for KNN -", accuracy, "\n")
```

```
## Train Accuracy for KNN - 0.8591921
```

```r
cat("Train error rate for KNN -", mean(y_pred != train$is_canceled), "\n")
```

```
## Train error rate for KNN - 0.1408079
```

```r
cat("-------------------------------------------------------", "\n")
```

```
## -------------------------------------------------------
```

```r
y_pred = knn(train=subset(train, select = -c(is_canceled)),
             test=subset(test, select = -c(is_canceled)),
             cl=train$is_canceled,
             k = 5,
             prob = TRUE)
cm <- table(test$is_canceled, y_pred)
cat("Prediction vs Actual table for Test KNN below -", "\n")
```

```
## Prediction vs Actual table for Test KNN below -
```

```r
cm
```

```
##    y_pred
##         0     1
##   0 19467  3015
##   1  4237  9044
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for KNN -", accuracy, "\n")
```

```
## Test Accuracy for KNN - 0.7972206
```

```r
cat("Test error rate for KNN -", mean(y_pred != test$is_canceled))
```

```
## Test error rate for KNN - 0.2027794
```

Cross Validation for KNN

```r
knitr::opts_chunk$set(warning = FALSE, message = FALSE)
folds = createFolds(train$is_canceled, k = 10)
cv = lapply(folds, function(x) {
  training_fold = train[-x, ]
  test_fold = train[x, ]
  y_pred = knn(train=subset(training_fold, select = -c(is_canceled)),
             test=subset(test_fold, select = -c(is_canceled)),
             cl=training_fold$is_canceled,
             k = 5,
             prob = TRUE)
  cm=table(y_pred, test_fold$is_canceled)
  accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
  return(accuracy)
})
accuracy = mean(as.numeric(cv))
cat("Test Accuracy for KNN CVV -", accuracy, "\n")
```

```
## Test Accuracy for KNN CVV - 0.7966254
```

Decision Tree

```
y_train = train$is_canceled
y_test = test$is_canceled
tree.fit = rpart(is_canceled ~ ., data=train, method='class')
tree.pred.train <- predict(tree.fit, train, type='class')
cat("Confusion Matrix for trees - \n")
```

```
## Confusion Matrix for trees -
```

```
cm <- table(tree.pred.train, y_train)
cat("Train error for trees -", mean(tree.pred.train != y_train))
```

```
## Train error for trees - 0.1936798
```

```
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train Accuracy for Decision Tree -", accuracy, "\n")
```

```
## Train Accuracy for Decision Tree - 0.8063202
```

```
tree.pred.test <- predict(tree.fit, test, type='class')
cat("Confusion Matrix for trees - \n")
```

```
## Confusion Matrix for trees -
```

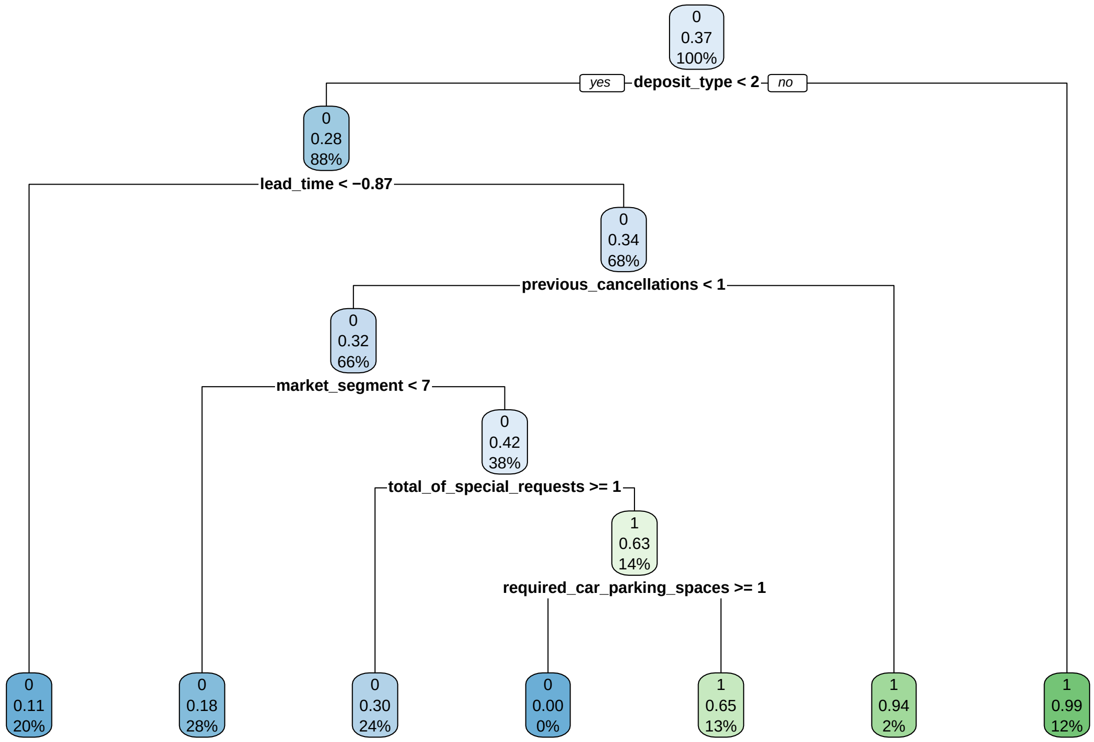
```
cm <- table(tree.pred.test, y_test)
cat("Test error for trees -", mean(tree.pred.test != y_test))
```

```
## Test error for trees - 0.1953975
```

```
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for Decision Tree -", accuracy, "\n")
```

```
## Test Accuracy for Decision Tree - 0.8046025
```

```
rpart.plot(tree.fit)
```



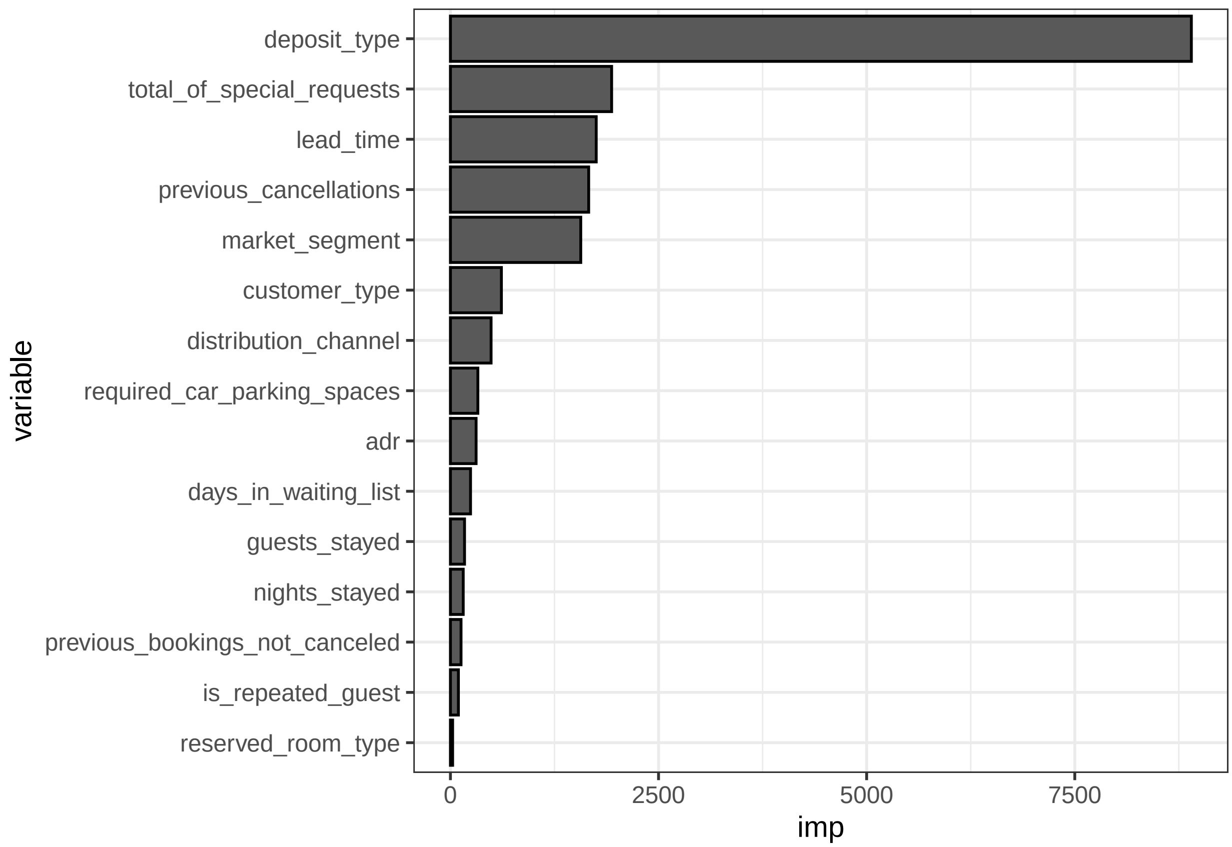
Cross Validation for Decision Tree

```
folds = createFolds(train$is_canceled, k = 10)
cv = lapply(folds, function(x) {
  training_fold = train[-x, ]
  test_fold = train[x, ]
  tree.fit = rpart(is_canceled ~ ., data=training_fold, method='class')
  tree.pred.test <- predict(tree.fit, test_fold, type='class')
  cm=table(tree.pred.test, test_fold$is_canceled)
  accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
  return(accuracy)
})
accuracy = mean(as.numeric(cv))
cat("Test Accuracy for Decision Tree CV -", accuracy, "\n")
```

```
## Test Accuracy for Decision Tree CV - 0.8057089
```

```
imp <- data.frame(imp = tree.fit$variable.importance)
df2 <- imp %>%
  tibble::rownames_to_column() %>%
  dplyr::rename("variable" = rowname) %>%
  dplyr::arrange(imp) %>%
  dplyr::mutate(variable = forcats::fct_inorder(variable))
ggplot2::ggplot(df2) +
  geom_col(aes(x = variable, y = imp),
```

```
           col = "black", show.legend = F) +
  coord_flip() +
  scale_fill_grey() +
  theme_bw()
```



Random Forest

```
train_rf = train
train_rf$is_canceled = as.factor(train_rf$is_canceled)
rf <- randomForest(is_canceled~., data = train_rf)
pred_train_rf <- predict(rf, train_rf)
cm <- table(pred_train_rf, train_rf$is_canceled)
cat("Confusion Matrix for RandomForest - \n")
```

```
## Confusion Matrix for RandomForest -
```

```
print(cm)
```

```
##
## pred_train_rf     0     1
##             0 50075  5357
##             1  2454 25561
```

```
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train Accuracy for RandomForest -", accuracy, "\n")
```

```
## Train Accuracy for RandomForest - 0.9063957
```

```
cat("---------------------------------------", "\n")
```

```
## ---------------------------------------
```

```
test_rf = test
test_rf$is_canceled = as.factor(test_rf$is_canceled)
pred_test_rf <- predict(rf, test_rf)
cat("Confusion Matrix for RandomForest - \n")
```

```
## Confusion Matrix for RandomForest -
```

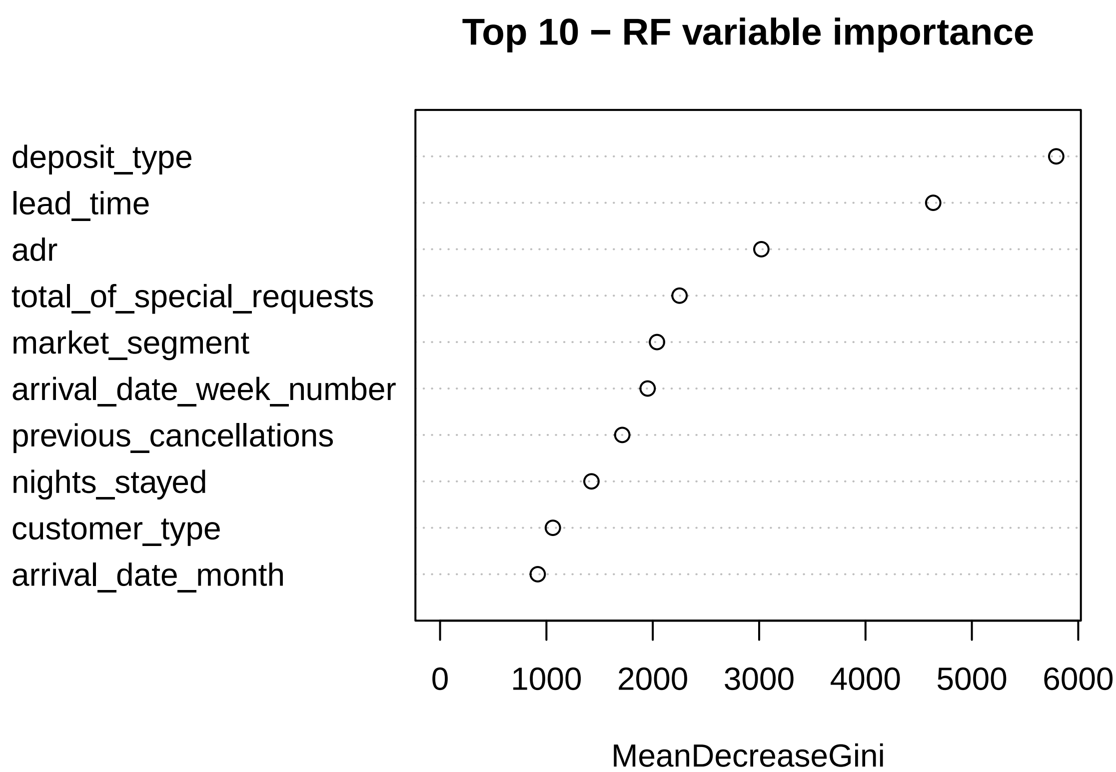
```
cm <- table(pred_test_rf, test_rf$is_canceled)
print(cm)
```

```
##
## pred_test_rf     0     1
##            0 20887  3709
##            1  1595  9572
```

```
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for RandomForest -", accuracy, "\n")
```

```
## Test Accuracy for RandomForest - 0.8516903
```

```
varImpPlot(rf,
           sort = T,
           n.var = 10,
           main = "Top 10 - RF variable importance")
```




```r
importance(rf)
```

```
##                                MeanDecreaseGini
## hotel                                 413.47373
## lead_time                            4636.43320
## arrival_date_month                    918.16558
## arrival_date_week_number             1951.41784
## meal                                  437.41691
## market_segment                       2039.48285
## distribution_channel                  427.63873
## is_repeated_guest                      72.34775
## previous_cancellations               1713.04314
## previous_bookings_not_canceled        183.79982
## reserved_room_type                    569.16997
## deposit_type                         5793.12857
## days_in_waiting_list                  119.81027
## customer_type                        1060.62602
## adr                                  3020.81933
## required_car_parking_spaces           903.30151
## total_of_special_requests            2251.46568
## guests_stayed                         633.76796
## nights_stayed                        1423.53708
```

```r
varUsed(rf)
```

```
##  [1]  79798 540172 276993 477728 125962  78679  46732  15173  14307  23453
## [11] 196225   6008  12239  69129 549934  18084 116897 183232 359017
```

AdaBoost

```r
adaboost <- gbm(is_canceled ~ ., data=train,
  distribution = "adaboost",
  n.trees = 500
)
adaboost.pred <- predict(adaboost, train, type='response') %>% round()
cm <- table(adaboost.pred, train$is_canceled)
cat("Confusion Matrix for AdaBoost - \n")
```

```
## Confusion Matrix for AdaBoost -
```

```r
print(cm)
```

```
##
## adaboost.pred     0     1
##             0 49052 12057
##             1  3477 18861
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train Accuracy for AdaBoost -", accuracy, "\n")
```

```
## Train Accuracy for AdaBoost - 0.8138459
```

```r
adaboost.pred <- predict(adaboost, test, type='response') %>% round()
cm <- table(adaboost.pred, test$is_canceled)
cat("Confusion Matrix for AdaBoost - \n")
```

```
## Confusion Matrix for AdaBoost -
```

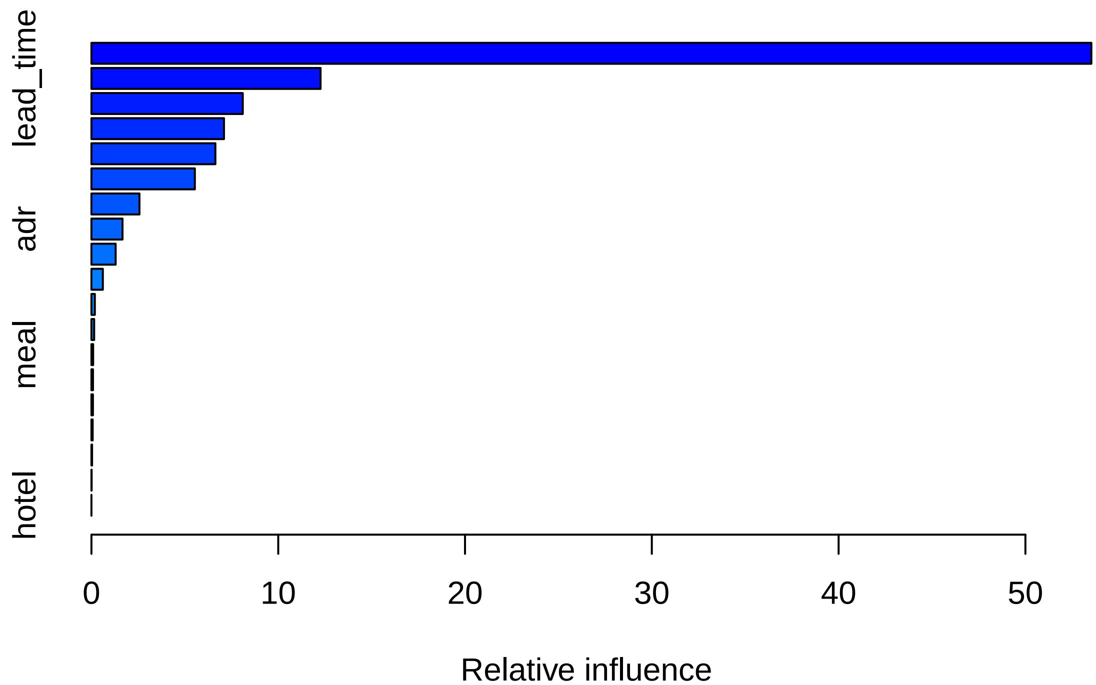
```r
print(cm)
```

```
##
## adaboost.pred     0     1
##             0 20947  5219
##             1  1535  8062
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for AdaBoost -", accuracy, "\n")
```

```
## Test Accuracy for AdaBoost - 0.8111456
```

```r
summary(adaboost)
```




```
##                                                               var     rel.inf
## deposit_type                                         deposit_type 53.53102156
## lead_time                                               lead_time 12.26655797
## total_of_special_requests               total_of_special_requests  8.10110147
## market_segment                                     market_segment  7.09569673
## previous_cancellations                     previous_cancellations  6.63608055
## required_car_parking_spaces           required_car_parking_spaces  5.53595331
## customer_type                                       customer_type  2.56971698
## adr                                                           adr  1.66361322
## previous_bookings_not_canceled previous_bookings_not_canceled  1.29683003
## nights_stayed                                       nights_stayed  0.61048989
## arrival_date_month                             arrival_date_month  0.18250997
## arrival_date_week_number                 arrival_date_week_number  0.14502837
## meal                                                         meal  0.09094234
## guests_stayed                                       guests_stayed  0.08340799
## days_in_waiting_list                         days_in_waiting_list  0.08004776
## reserved_room_type                             reserved_room_type  0.06530153
## distribution_channel                         distribution_channel  0.03439573
## is_repeated_guest                               is_repeated_guest  0.01130460
## hotel                                                       hotel  0.00000000
```

4. "Data Modeling with Important Features"

Create Dataframe with important 5 features

```r
df_features <- df
df_features <- subset(df_features, select=c(deposit_type, adr, total_of_special_requests, market_segment
head(df_features)
```

```
##   deposit_type         adr total_of_special_requests market_segment  lead_time
## 1            1 -2.02183206                         0              4  2.2258692
## 2            1 -2.02183206                         0              4  5.9217601
## 3            1 -0.53474022                         0              4 -0.9086205
## 4            1 -0.53474022                         0              3 -0.8524804
## 5            1 -0.07869872                         1              7 -0.8431237
## 6            1 -0.07869872                         1              7 -0.8431237
##   is_canceled
## 1           0
## 2           0
## 3           0
## 4           0
## 5           0
## 6           0
```

Split into test and train dataset

```r
idx <- sample(nrow(df_features), nrow(df_features)*0.3)
test_features <- df_features[idx,]
train_features <- df_features[-idx,]
```

Logistic Regression with features

```r
log_classifier = glm(formula=is_canceled ~ ., family=binomial, data=train_features)
summary(log_classifier)
```

```
##
## Call:
## glm(formula = is_canceled ~ ., family = binomial, data = train_features)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.1003  -0.8169  -0.5518   0.7186   3.1733
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -7.434707   0.089268  -83.28   <2e-16 ***
## deposit_type               4.340116   0.064713   67.07   <2e-16 ***
## adr                        0.244822   0.009047   27.06   <2e-16 ***
## total_of_special_requests -0.636219   0.012465  -51.04   <2e-16 ***
## market_segment             0.417823   0.008503   49.14   <2e-16 ***
## lead_time                  0.379298   0.009343   40.60   <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: 110214  on 83446  degrees of freedom
```

```
## Residual deviance:  82638  on 83441  degrees of freedom
## AIC: 82650
##
## Number of Fisher Scoring iterations: 6
```

```
prob_pred = predict(log_classifier, train_features, type='response')
y_pred = ifelse(prob_pred > 0.5, 1, 0)
cm=table(y_pred, train_features$is_canceled)
cat("Prediction vs Actual table for Train Logistic Regression below -", "\n")
```

```
## Prediction vs Actual table for Train Logistic Regression below -
```

```
print(cm)
```

```
##
## y_pred     0     1
##      0 50086 17199
##      1  2259 13903
```

```
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train error rate for KNN -", mean(y_pred != train_features$is_canceled), "\n")
```

```
## Train error rate for KNN - 0.2331779
```

```
cat("Train Accuracy for Logistic Regression -", accuracy, "\n")
```

```
## Train Accuracy for Logistic Regression - 0.7668221
```

```
cat("---------------------------------------------------------------", "\n")
```

```
## ---------------------------------------------------------------
```

```
prob_pred = predict(log_classifier, test_features, type='response')
y_pred = ifelse(prob_pred > 0.5, 1, 0)

# Making the Confusion Matrix
cat("Prediction vs Actual table for Test Logistic Regression below -", "\n")
```

```
## Prediction vs Actual table for Test Logistic Regression below -
```

```
cm=table(y_pred, test_features$is_canceled)
print(cm)
```

```
##
## y_pred     0     1
##      0 21715  7228
##      1   951  5869
```

```
cat("Test error rate for Logistic Regression -", mean(y_pred != test_features$is_canceled), "\n")
```

```
## Test error rate for Logistic Regression - 0.2287001
```

```
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for Logistic Regression -", accuracy, "\n")
```

```
## Test Accuracy for Logistic Regression - 0.7712999
```

Cross Validation for Logistic Regression With Imp Features

```
knitr::opts_chunk$set(warning = FALSE, message = FALSE)
folds = createFolds(train_features$is_canceled, k = 10)
cv = lapply(folds, function(x) {
  training_fold = train_features[-x, ]
  test_fold = train_features[x, ]
  log_classifier = glm(formula=is_canceled ~ ., family=binomial, data=training_fold)
  prob_pred = predict(log_classifier, test_fold, type='response')
  y_pred = ifelse(prob_pred > 0.5, 1, 0)
  cm=table(y_pred, test_fold$is_canceled)
  accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
  return(accuracy)
})
accuracy = mean(as.numeric(cv))
cat("Test Accuracy for Logistic Regression CV -", accuracy)
```

```
## Test Accuracy for Logistic Regression CV - 0.766738
```

KNN With Imp Features

```
y_pred = knn(train=subset(train_features, select = -c(is_canceled)),
             test=subset(train_features, select = -c(is_canceled)),
             cl=train_features$is_canceled,
             k = 5,
             prob = TRUE)
cm <- table(train_features$is_canceled, y_pred)
cat("Prediction vs Actual table for Train KNN below -", "\n")
```

```
## Prediction vs Actual table for Train KNN below -
```

```
print(cm)
```

```
##    y_pred
##         0     1
##   0 47709  4636
##   1  8060 23042
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train Accuracy for KNN -", accuracy, "\n")
```

```
## Train Accuracy for KNN - 0.8478555
```

```r
cat("Train error rate for KNN -", mean(y_pred != train_features$is_canceled), "\n")
```

```
## Train error rate for KNN - 0.1521445
```

```r
cat("-------------------------------------------------------", "\n")
```

```
## -------------------------------------------------------
```

```r
y_pred = knn(train=subset(train_features, select = -c(is_canceled)),
             test=subset(test_features, select = -c(is_canceled)),
             cl=train_features$is_canceled,
             k = 5,
             prob = TRUE)
cm <- table(test_features$is_canceled, y_pred)
cat("Prediction vs Actual table for Test KNN below -", "\n")
```

```
## Prediction vs Actual table for Test KNN below -
```

```r
cm
```

```
##    y_pred
##         0     1
##   0 19796  2870
##   1  4346  8751
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for KNN -", accuracy, "\n")
```

```
## Test Accuracy for KNN - 0.7982272
```

```r
cat("Test error rate for KNN -", mean(y_pred != test_features$is_canceled))
```

```
## Test error rate for KNN - 0.2017728
```

Cross Validation for KNN

```r
knitr::opts_chunk$set(warning = FALSE, message = FALSE)
folds = createFolds(train_features$is_canceled, k = 10)
cv = lapply(folds, function(x) {
  training_fold = train_features[-x, ]
  test_fold = train_features[x, ]
  y_pred = knn(train=subset(training_fold, select = -c(is_canceled)),
             test=subset(test_fold, select = -c(is_canceled)),
```

```
             cl=training_fold$is_canceled,
             k = 5,
             prob = TRUE)
  cm=table(y_pred, test_fold$is_canceled)
  accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
  return(accuracy)
})
accuracy = mean(as.numeric(cv))
cat("Test Accuracy for KNN CVV -", accuracy, "\n")
```

```
## Test Accuracy for KNN CVV - 0.7931861
```

Decision Tree With Imp Features

```
y_train = train_features$is_canceled
y_test = test_features$is_canceled
tree.fit = rpart(is_canceled ~ ., data=train_features, method='class')
tree.pred.train <- predict(tree.fit, train_features, type='class')
cat("Confusion Matrix for trees - \n")
```

```
## Confusion Matrix for trees -
```

```
cm <- table(tree.pred.train, y_train)
cat("Train error for trees -", mean(tree.pred.train != y_train))
```

```
## Train error for trees - 0.2149388
```

```
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train Accuracy for Decision Tree -", accuracy, "\n")
```

```
## Train Accuracy for Decision Tree - 0.7850612
```

```
tree.pred.test <- predict(tree.fit, test_features, type='class')
cat("Confusion Matrix for trees - \n")
```

```
## Confusion Matrix for trees -
```

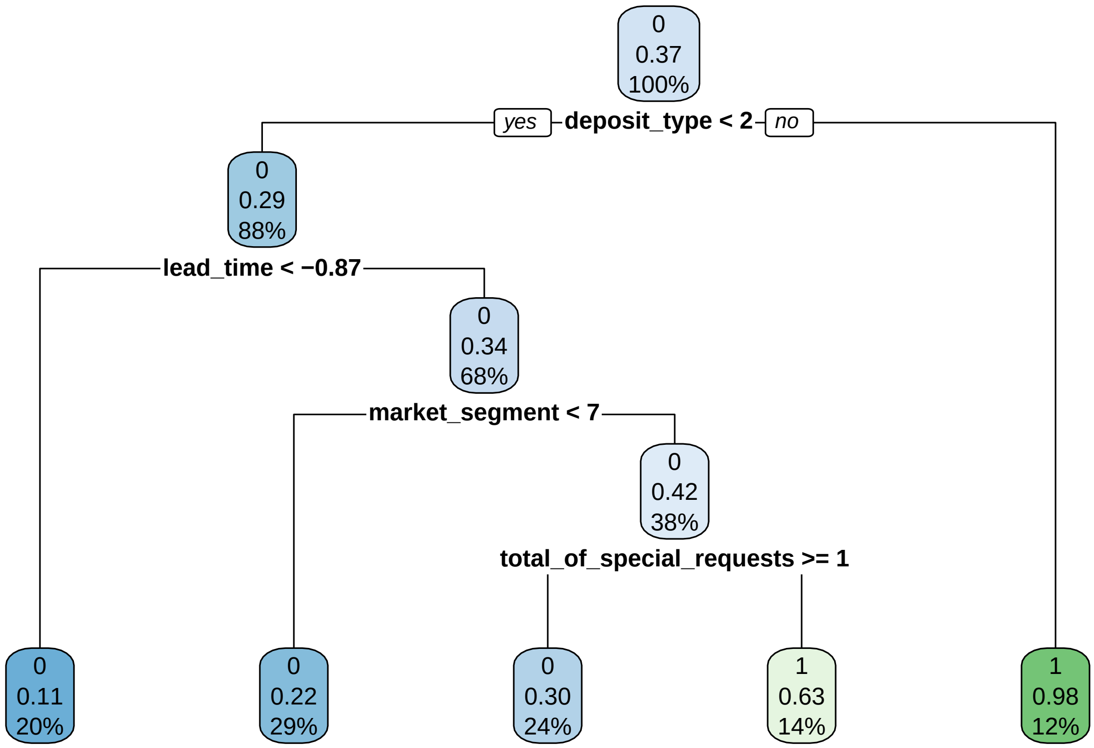
```
cm <- table(tree.pred.test, y_test)
cat("Test error for trees -", mean(tree.pred.test != y_test))
```

```
## Test error for trees - 0.2127898
```

```
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for Decision Tree -", accuracy, "\n")
```

```
## Test Accuracy for Decision Tree - 0.7872102
```

```r
rpart.plot(tree.fit)
```



Cross Validation for Decision Tree

```r
folds = createFolds(train_features$is_canceled, k = 10)
cv = lapply(folds, function(x) {
  training_fold = train_features[-x, ]
  test_fold = train_features[x, ]
  tree.fit = rpart(is_canceled ~ ., data=training_fold, method='class')
  tree.pred.test <- predict(tree.fit, test_fold, type='class')
  cm=table(tree.pred.test, test_fold$is_canceled)
  accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
  return(accuracy)
})
accuracy = mean(as.numeric(cv))
cat("Test Accuracy for Decision Tree CV -", accuracy, "\n")
```

```
## Test Accuracy for Decision Tree CV - 0.7849535
```

Random Forest With Imp Features

```r
train_rf = train_features
train_rf$is_canceled = as.factor(train_rf$is_canceled)
rf <- randomForest(is_canceled~., data = train_rf)
pred_train_rf <- predict(rf, train_rf)
```

```r
cm <- table(pred_train_rf, train_rf$is_canceled)
cat("Confusion Matrix for RandomForest - \n")
```

```
## Confusion Matrix for RandomForest -
```

```r
print(cm)
```

```
##
## pred_train_rf     0     1
##             0 48146 12480
##             1  4199 18622
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train Accuracy for RandomForest -", accuracy, "\n")
```

```
## Train Accuracy for RandomForest - 0.8001246
```

```r
cat("---------------------------------------", "\n")
```

```
## ---------------------------------------
```

```r
test_rf = test_features
test_rf$is_canceled = as.factor(test_rf$is_canceled)
pred_test_rf <- predict(rf, test_rf)
cat("Confusion Matrix for RandomForest - \n")
```

```
## Confusion Matrix for RandomForest -
```

```r
cm <- table(pred_test_rf, test_rf$is_canceled)
print(cm)
```

```
##
## pred_test_rf     0     1
##            0 20811  5336
##            1  1855  7761
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for RandomForest -", accuracy, "\n")
```

```
## Test Accuracy for RandomForest - 0.7989263
```

AdaBoost With Imp Features

```r
adaboost <- gbm(is_canceled ~ ., data=train_features,
  distribution = "adaboost",
  n.trees = 500
)
adaboost.pred <- predict(adaboost, train_features, type='response') %>% round()
cm <- table(adaboost.pred, train_features$is_canceled)
cat("Confusion Matrix for AdaBoost - \n")
```

```
## Confusion Matrix for AdaBoost -
```

```r
print(cm)
```

```
##
## adaboost.pred     0     1
##             0 48258 13162
##             1  4087 17940
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train Accuracy for AdaBoost -", accuracy, "\n")
```

```
## Train Accuracy for AdaBoost - 0.7932939
```

```r
adaboost.pred <- predict(adaboost, test_features, type='response') %>% round()
cm <- table(adaboost.pred, test_features$is_canceled)
cat("Confusion Matrix for AdaBoost - \n")
```

```
## Confusion Matrix for AdaBoost -
```

```r
print(cm)
```

```
##
## adaboost.pred     0     1
##             0 20896  5546
##             1  1770  7551
```

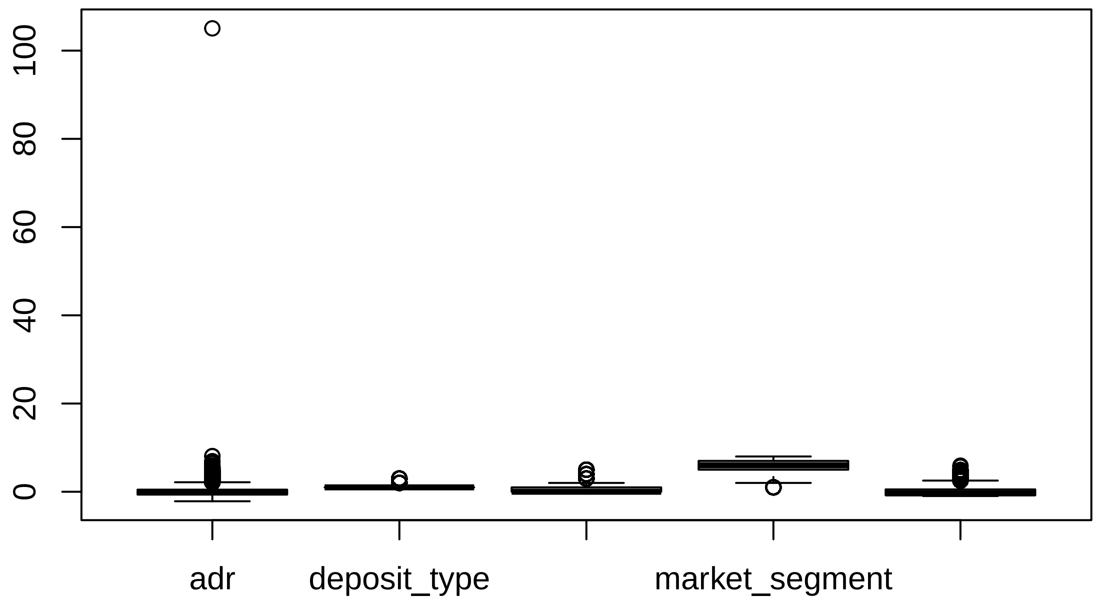
```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for AdaBoost -", accuracy, "\n")
```

```
## Test Accuracy for AdaBoost - 0.795431
```

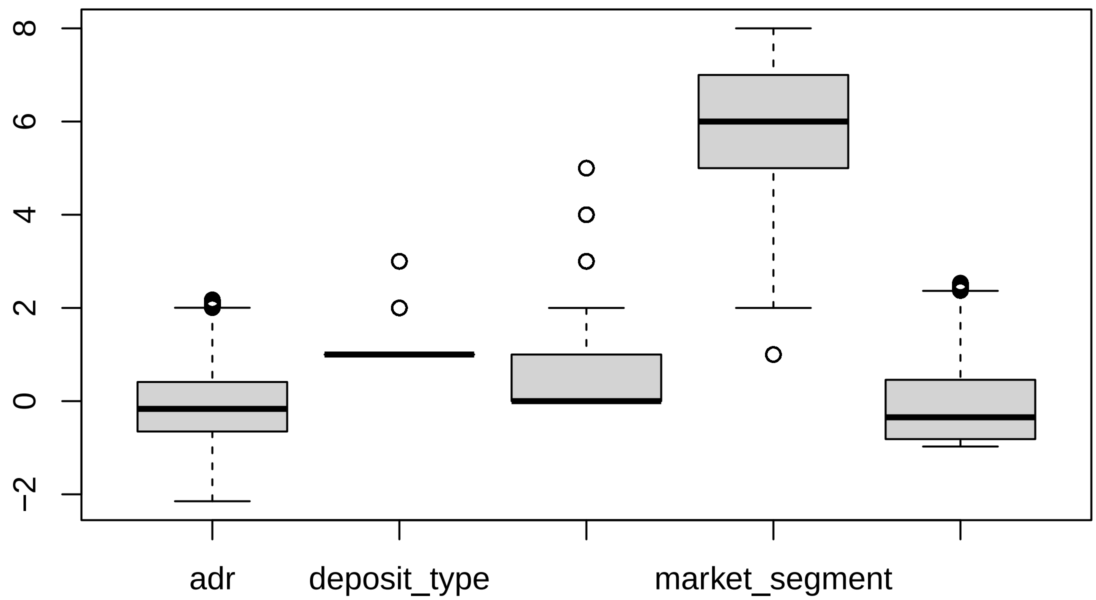
5. "Outliers with Important Features"

BoxPlots

```r
outlier_df <- df_features

box_plot <- data.frame(adr = outlier_df$adr,
                   deposit_type = outlier_df$deposit_type,
                   total_spl_requests = outlier_df$total_of_special_requests,
                   market_segment=outlier_df$market_segment,
                   lead_time=outlier_df$lead_time)
boxplot(box_plot)
```




```
outliers = boxplot(outlier_df$lead_time, plot=FALSE)$out
outlier_df = outlier_df[-which(outlier_df$lead_time %in% outliers),]

outliers = boxplot(outlier_df$adr, plot=FALSE)$out
outlier_df = outlier_df[-which(outlier_df$adr %in% outliers),]

box_plot <- data.frame(adr = outlier_df$adr,
                   deposit_type = outlier_df$deposit_type,
                   total_spl_requests = outlier_df$total_of_special_requests,
                   market_segment=outlier_df$market_segment,
                   lead_time=outlier_df$lead_time)
boxplot(box_plot)
```




```
idx <- sample(nrow(outlier_df), nrow(outlier_df)*0.3)
outlier_test <- outlier_df[idx,]
outlier_train <-outlier_df[-idx,]
```

RandomForest With Imp Features and no outliers

```
outlier_train_rf = outlier_train
outlier_train_rf$is_canceled = as.factor(outlier_train_rf$is_canceled)
outlier_rf <- randomForest(is_canceled~., data = outlier_train_rf)
pred_train_rf <- predict(outlier_rf, outlier_train_rf)
cm <- table(pred_train_rf, outlier_train_rf$is_canceled)
cat("Confusion Matrix for RandomForest No Outliers and Imp Features - \n")
```

```
## Confusion Matrix for RandomForest No Outliers and Imp Features -
```

```
print(cm)
```

```
##
## pred_train_rf     0     1
##             0 46369 11770
##             1  3918 16668
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Train Accuracy for RandomForest Without Outliers -", accuracy, "\n")
```

```
## Train Accuracy for RandomForest Without Outliers - 0.800724
```

```r
cat("-----------------------------------------------------------", "\n")
```

```
## -----------------------------------------------------------
```

```r
outlier_test_rf = outlier_test
outlier_test_rf$is_canceled = as.factor(outlier_test_rf$is_canceled)
outlier_p2_rf <- predict(outlier_rf, outlier_test_rf)
cm <- table(outlier_p2_rf, outlier_test_rf$is_canceled)
cat("Confusion Matrix for RandomForest No Outliers and Imp Features - \n")
```

```
## Confusion Matrix for RandomForest No Outliers and Imp Features -
```

```r
print(cm)
```

```
##
## outlier_p2_rf     0     1
##             0 19742  5258
##             1  1681  7057
```

```r
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for RandomForest Without Outliers -", accuracy, "\n")
```

```
## Test Accuracy for RandomForest Without Outliers - 0.7943269
```