Predict the Cancellation of a City Hotel and a Resort Hotel

Leiteng Huang, Jie Tang

California University Los Angeles Extension Program

Class374815: Machine Learning Using R

Michael Chang

September 2, 2020

Table of Contents

Introduction

Nowadays machine learning technique is utilized in many different filed. As long as people is intended in it, machine learning can offer improvement for user to understand their customer interested places. For our final project report, our group found out a hotel booking data set very interested us on Kaggle.com. This data set contains booking information for a city hotel and a resort hotel, meanwhile includes information such as when the booking was made, length of stay, the number of adults, children, and/or babies, and the number of available parking spaces, among other things. But most importantly, it contains one property called “is_canceled”. This property represents that whether a customer who booked this hotel cancel this booking at the end and this is what our machine learning project want to focus on: Do those attributes of hotel influence customers’ decision of cancelling the booking of hotel? If so, which attribute has the most affect on this decision. We are going to create precise model by machine learning algorithms we were taught in class and data set we got from Kaggle. This model should help us predict whether a customer is going to cancel the hotel booking when we have the other information from this customer. It is important for hotel managers to solve this problem because once they solve it, they will know which property of hotel should focus on most. And it can help them reduce the cancellation from customers.

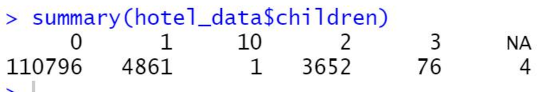
Apparently, this problem is a binary classification problem. The prediction has only two possibility: cancel or not cancel. To enhance the quality of our machine learning model, because confusion matrix describes the complete performance of the model and it is clear for us to see detailed result of our tests, we can apply confusion matrix evaluation metrics to be our evaluation metrics.

Actually, problem about hotel booking have done previously by Booking.com. At website https://booking.ai/, it introduces that during the last five years, Machine Learning became a standard tool for Product Development in Booking.com and it presents its Machine Learning Product ionization System which supports a large variety of Machine Learning approaches. Our problem should be counted as another branch of machine learning problem for hotel booking because we focus on the cancellation of booking from customers.

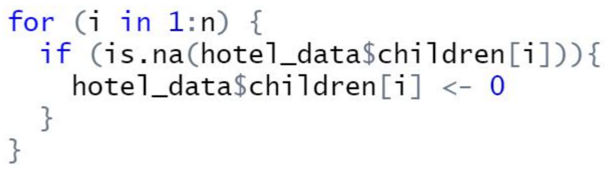
Data source and Data preprocessing

As we mentioned above, our data set was extracted from Kaggle.com. Kaggle offers a public, convenient and clean source for users to obtain the dataset they want. To clean our dataset, first thing we did is checking whether it contains NULL value. And we found out that there are many NULL values exist in some variables such as children, agent and company. At this point, we noticed that actually they are not really all empty value inside this table. Most of them is filled with "NULL" in it. We cannot directly use is.na() to check whether we need to fix it. At the same time, all "NULL" value inside the table can reasonably change to zero. Null value in company and agent means customer did not book this hotel by company or agent which can be represent by number zero. To fix "NULL" value we simply used replace() method to replace all

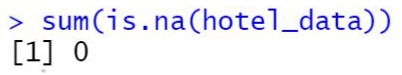
"NULL" to zero. For the NA value part, take children variables as example, we used a for loop to find out the NA values and replace it by 0. After we cleaning the dataset, there should be no NULL and NA values in our dataset.



```
> summary(hotel_data$children)
     0      1     10      2      3     NA 
110796   4861      1   3652     76      4 
```



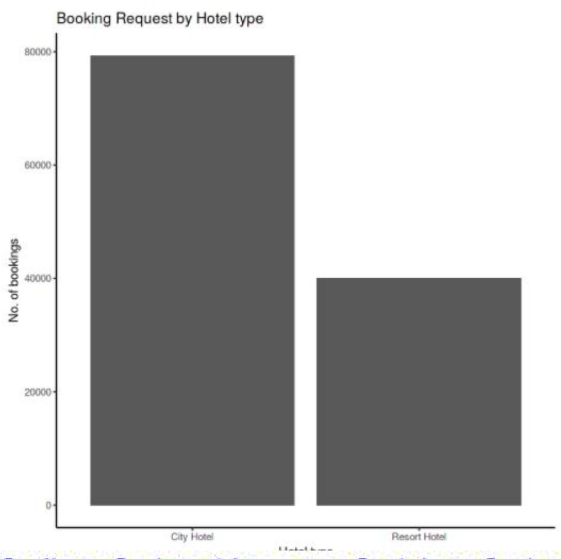
```
for (i in 1:n) {
  if (is.na(hotel_data$children[i])){
    hotel_data$children[i] <- 0
  }
}
```



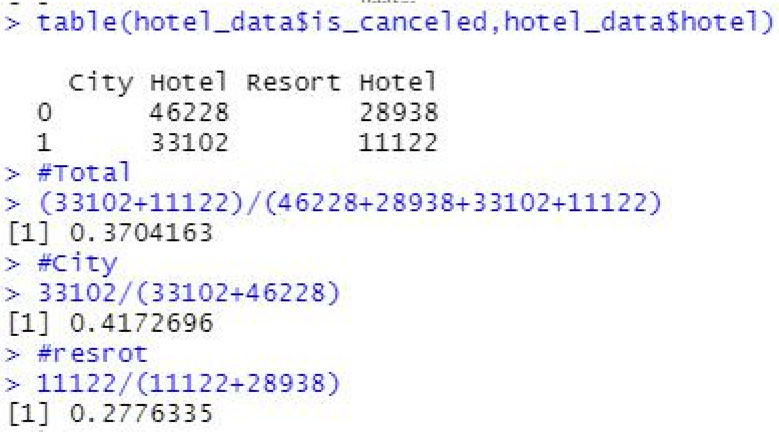
```
> sum(is.na(hotel_data))
[1] 0
```

EDA

- Which month has the highest number of visitors?
- What is the monthly average daily rate per person over the year?
- Which country has the most number of hotel visitors?
- Which customer type contributes to the most hotel booking cancellations?

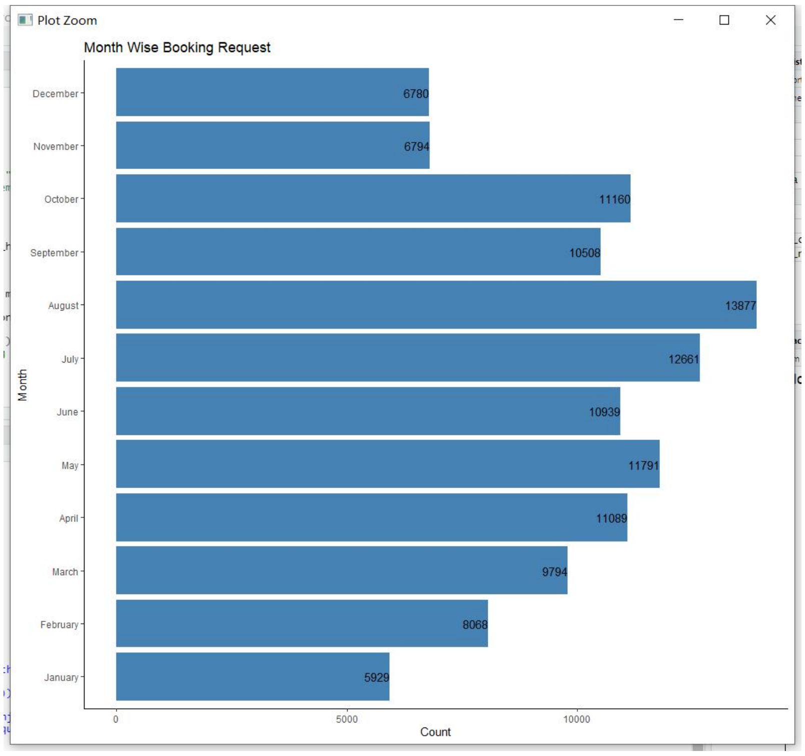


First, we check the booking request by hotel type. We can see on the left; the city hotel has more demand than the resort hotel. This does not affect the process of training, we probably will train the model with two hotels data first, then try to use the separate data to train again.

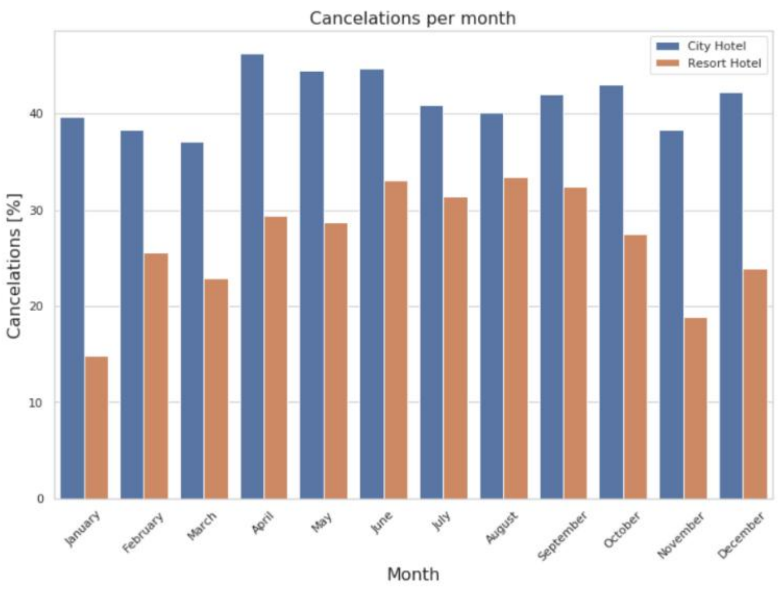


```
> table(hotel_data$is_canceled,hotel_data$hotel)

    City Hotel Resort Hotel
  0      46228        28938
  1      33102        11122
> #Total
> (33102+11122)/(46228+28938+33102+11122)
[1] 0.3704163
> #City
> 33102/(33102+46228)
[1] 0.4172696
> #resrot
> 11122/(11122+28938)
[1] 0.2776335
```

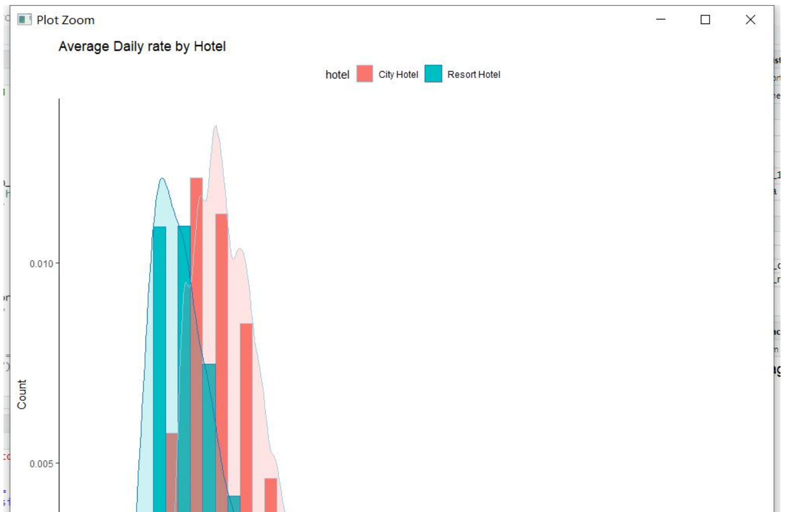
We then checked the cancelation rate in our data. We can see that for the total, we have a 37.0% cancelation rate, and the city hotel has a higher ratio of 41.7% and the Resort hotel has a lower ratio of 27.8% cancelation



We first check the booking request in different months. We can see that during the summer of the year. Hotels have higher booking requests than during wintertime. August has twice much demand than January. With a higher booking request, we guess the cancelation rate will be higher in summer too.



This is the graph of cancelations per month by percentage. We can see that the City hotel has a stable ratio. The Resort hotel has a high cancelations rate during summer, from April to October. So, the different month in the year may have a high correlation with



This is the graph about the ADR attribute by the hotel. ADR is the average daily rate by the hotel. We do see that the average price of the room is around 70 to 100 euros. We think the room price could also be part of the reason for the cancelation. If guests found another hotel is cheaper. They may cancel the reservation.

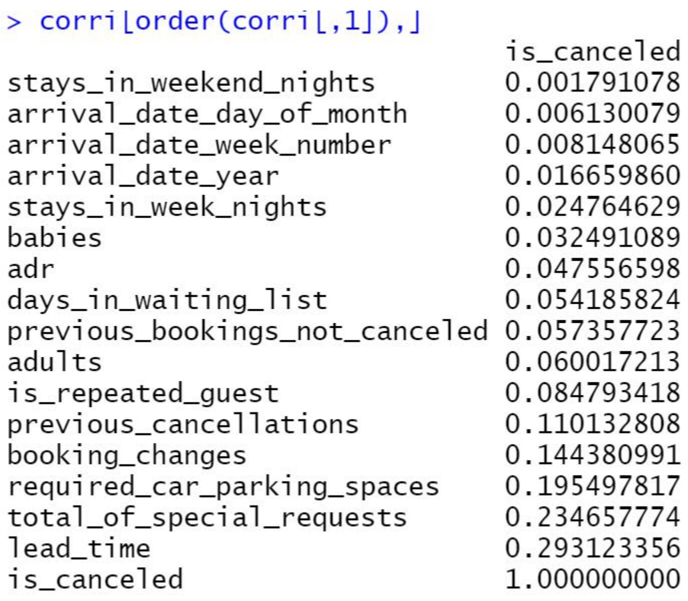
False positive and False negative

Because we are facing a binary classification problem, we have to consider the situation that our prediction result is wrong. If we say a customer will not cancel the hotel booking is null hypothesis. The false positive (type I error) will be the case that prediction is not cancel but customer actually cancel the booking. The false negative (type II error) will be the case that prediction is customer will cancel the booking but actually not. From the perspective of manager of hotel, we should focus more on false positive for the reason that Type I error causes the reduction of customers and profits.

To avoid Type I error, it is important to run our tests for long enough. Based on the fact that we have 11k plus data size, I think we can avoid a decent number of Type I error.

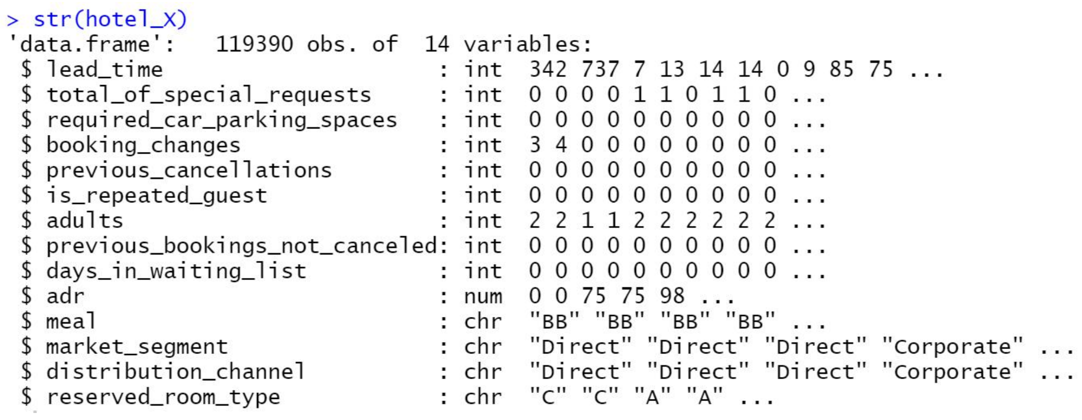
Feature Engineering and Selection

For variables we are going to use, we decided to test correlation of each numeric variable at first. Then we picked those variables that have a decent correlation with “is_canceled” variable. From the correlation list, we picked top ten numeric variables which has high correlation with “is_canceled”. Those are lead_time, total_of_special_requests, required_car_parking_spaces, booking_changes, previous_cancellations, is_repeated_guest, agent, adults, previous_bookings_not_canceled, days_in_waiting_list and adr.

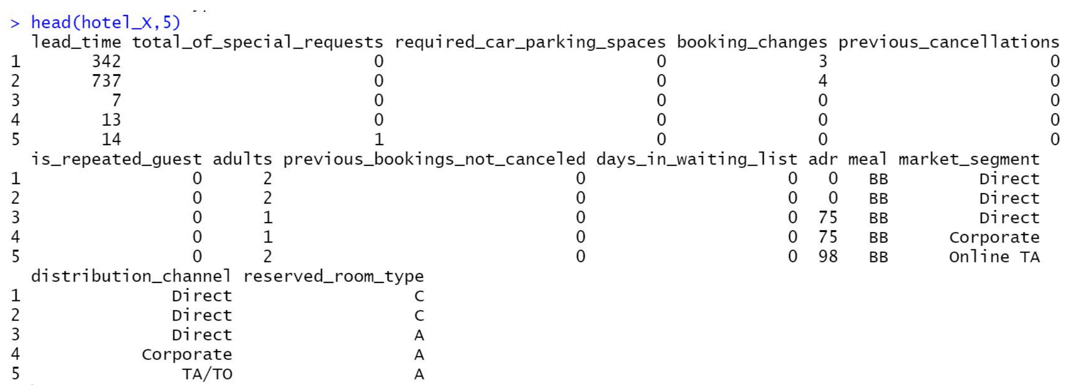


```
> corri[order(corri[,1]),]
                               is_canceled
stays_in_weekend_nights        0.001791078
arrival_date_day_of_month      0.006130079
arrival_date_week_number       0.008148065
arrival_date_year              0.016659860
stays_in_week_nights           0.024764629
babies                         0.032491089
adr                            0.047556598
days_in_waiting_list           0.054185824
previous_bookings_not_canceled 0.057357723
adults                         0.060017213
is_repeated_guest              0.084793418
previous_cancellations         0.110132808
booking_changes                0.144380991
required_car_parking_spaces    0.195497817
total_of_special_requests      0.234657774
lead_time                      0.293123356
is_canceled                    1.000000000
```

It is not ideal to only pick numeric variables. For those category variables, we discussed their relationship with “is_canceled” and picked four category variables that we considered they are the most fitted variables for our machine learning model which are meal, market_segment, distribution_channel, reserved_room_type. Finally, we merge numeric variables and category variables together to be our final dataset.



```
> str(hotel_X)
'data.frame':	119390 obs. of  14 variables:
 $ lead_time                     : int  342 737 7 13 14 14 0 9 85 75 ...
 $ total_of_special_requests     : int  0 0 0 0 1 1 0 1 1 0 ...
 $ required_car_parking_spaces   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ booking_changes               : int  3 4 0 0 0 0 0 0 0 0 ...
 $ previous_cancellations        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ is_repeated_guest             : int  0 0 0 0 0 0 0 0 0 0 ...
 $ adults                        : int  2 2 1 1 2 2 2 2 2 2 ...
 $ previous_bookings_not_canceled: int  0 0 0 0 0 0 0 0 0 0 ...
 $ days_in_waiting_list          : int  0 0 0 0 0 0 0 0 0 0 ...
 $ adr                           : num  0 0 75 75 98 ...
 $ meal                          : chr  "BB" "BB" "BB" "BB" ...
 $ market_segment                : chr  "Direct" "Direct" "Direct" "Corporate" ...
 $ distribution_channel          : chr  "Direct" "Direct" "Direct" "Corporate" ...
 $ reserved_room_type            : chr  "C" "C" "A" "A" ...
```



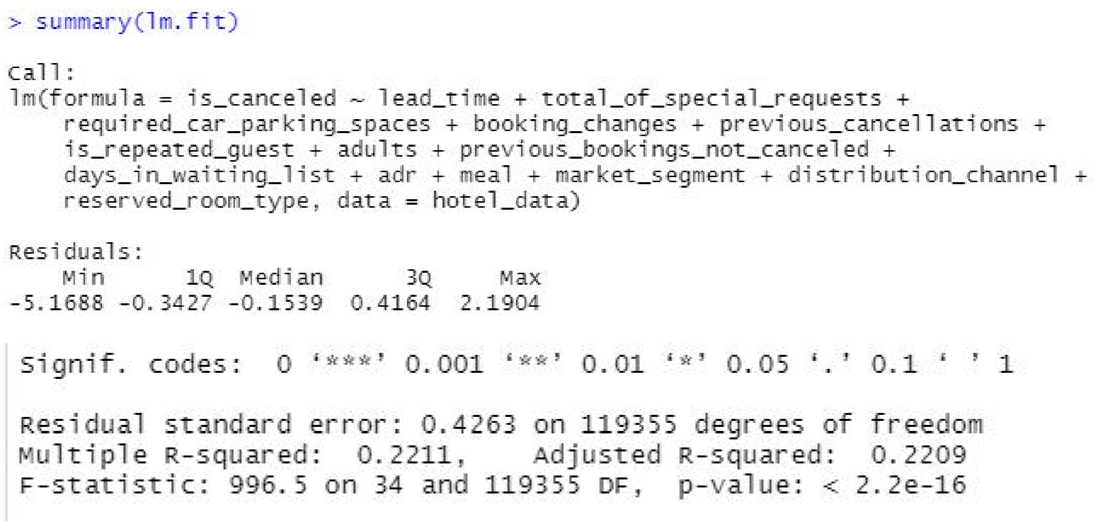
```
> head(hotel_X,5)
  lead_time total_of_special_requests required_car_parking_spaces booking_changes previous_cancellations
1       342                         0                           0               3                      0
2       737                         0                           0               4                      0
3         7                         0                           0               0                      0
4        13                         0                           0               0                      0
5        14                         1                           0               0                      0
  is_repeated_guest adults previous_bookings_not_canceled days_in_waiting_list adr meal market_segment
1                 0      2                              0                    0   0   BB         Direct
2                 0      2                              0                    0   0   BB         Direct
3                 0      1                              0                    0  75   BB         Direct
4                 0      1                              0                    0  75   BB      Corporate
5                 0      2                              0                    0  98   BB      Online TA
  distribution_channel reserved_room_type
1               Direct                  C
2               Direct                  C
3               Direct                  A
4            Corporate                  A
5                TA/TO                  A
```

We did not consider add complicated feature transformations in our dataset because we consider that there are enough variables for us to design our machine learning models. Only transformations we are going to add is only a new variable called prediction to store the model prediction results. Also, no need to scaling or normalizing our features. For interaction, our opinion is that we already tested all numeric variables correlation with “is_canceled” variable and variables we picked have positive relationship with “is_canceled” and we do not need to consider feature interaction. And all variables shown an independent relationship between each other. Our feature selection should pertain to all our models because we only focus on classification model and our dataset is fitted for them.

Modeling

1. Multiple Linear Regression

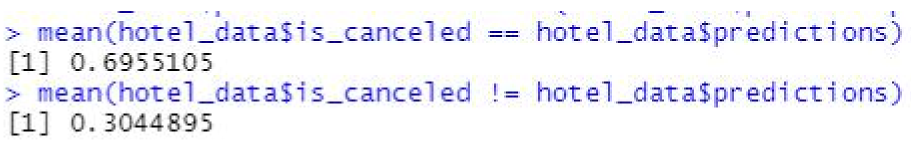
The first model, also the simplest model we want use is the Multiple Linear Regression. It is easy to use in the R, we just use the lm() function. The syntax lm(y~x1+x2+x3…)is used to fit model. And then we use summary() to print everything.



```
> summary(lm.fit)

Call:
lm(formula = is_canceled ~ lead_time + total_of_special_requests +
    required_car_parking_spaces + booking_changes + previous_cancellations +
    is_repeated_guest + adults + previous_bookings_not_canceled +
    days_in_waiting_list + adr + meal + market_segment + distribution_channel +
    reserved_room_type, data = hotel_data)

Residuals:
    Min      1Q  Median      3Q     Max
-5.1688 -0.3427 -0.1539  0.4164  2.1904
```

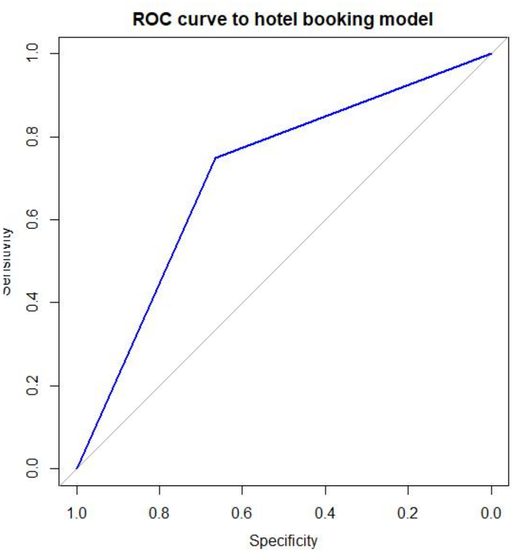
```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4263 on 119355 degrees of freedom
Multiple R-squared:  0.2211,    Adjusted R-squared:  0.2209
F-statistic: 996.5 on 34 and 119355 DF,  p-value: < 2.2e-16
```

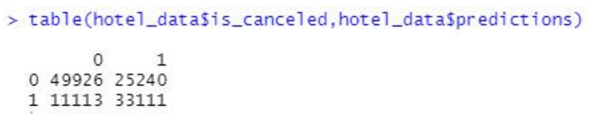
Then we calculated actually percentage of cancelation rate, it is 37.04%.



```
> mean(hotel_data$is_canceled == hotel_data$predictions)
[1] 0.6955105
> mean(hotel_data$is_canceled != hotel_data$predictions)
[1] 0.3044895
```

From the code result, we can get an accuracy is around 69.55 percentage and error rate are around 30.45 percentage. We used both ROC curve and confusion matrix to show the train and test evaluation metrics.



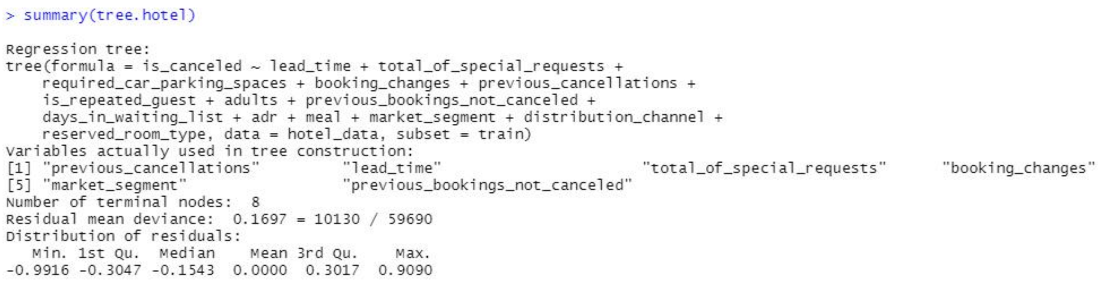


```
> table(hotel_data$is_canceled,hotel_data$predictions)

        0     1
  0 49926 25240
  1 11113 33111
```

By these evaluation metrics, we can have a conclusion that even though the overall accuracy is at acceptable level 71.7 percentage accuracy, the Type I error rate is much higher than Type II error rate. Type I error occurs mean that we predict this customer will not cancel the booking, but he cancels it. In this case, Type I error is the one that really affect the profit of hotel. Therefore, this is not a perfect result we wish to get.

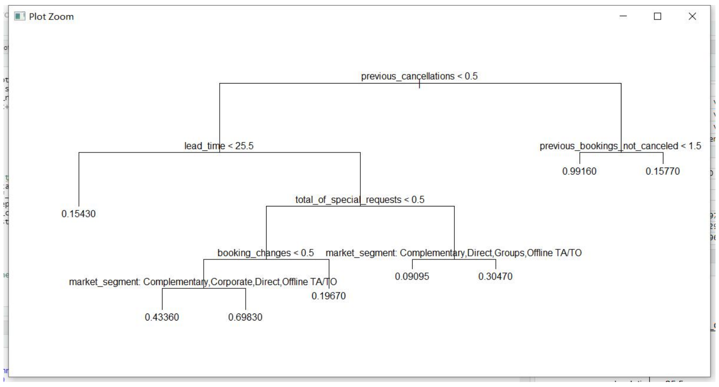
2. Tree Regression.

The first model we want to try is the tree regression. First, we have created a training set, and fit the tree to the training data.

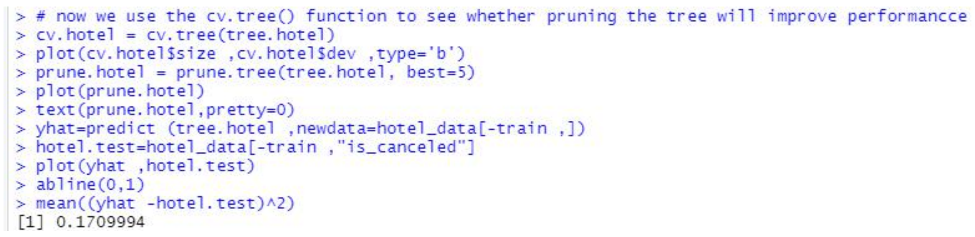


```
> summary(tree.hotel)

Regression tree:
tree(formula = is_canceled ~ lead_time + total_of_special_requests +
    required_car_parking_spaces + booking_changes + previous_cancellations +
    is_repeated_guest + adults + previous_bookings_not_canceled +
    days_in_waiting_list + adr + meal + market_segment + distribution_channel +
    reserved_room_type, data = hotel_data, subset = train)
Variables actually used in tree construction:
[1] "previous_cancellations"         "lead_time"                      "total_of_special_requests"      "booking_changes"
[5] "market_segment"                 "previous_bookings_not_canceled"
Number of terminal nodes:  8
Residual mean deviance:  0.1697 = 10130 / 59690
Distribution of residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.9916 -0.3047 -0.1543  0.0000  0.3017  0.9090
```

We now can see that the variables actually used in tree construction is “previous_cancellation”, “lead_time”, “total_of_special_requests”, “booking_changes”, “market_segment” and “previous_bookings_not_canceled”. We now want to plot the tree.


Plot Zoom
previous_cancellations < 0.5
lead_time < 25.5
previous_bookings_not_canceled < 1.5
0.99160
0.15770
0.15430
total_of_special_requests < 0.5
booking_changes < 0.5
market_segment: Complementary,Direct,Groups,Offline TA/TO
market_segment: Complementary,Corporate,Direct,Offline TA/TO
0.09095
0.30470
0.19670
0.43360
0.69830

The tree indicates that lower values of previous cancelation corresponding to higher cancelation ratio. Which means if the customer had cancelation before seems like to have higher chance to cancel the booking. After this we used cv.tree() function to see whether pruning the tree will improve performance. We then use the prune.tree() function to prune thee tree. We keeping with the cross-validation results, we use the unpruned tree to make predictions on the test set.

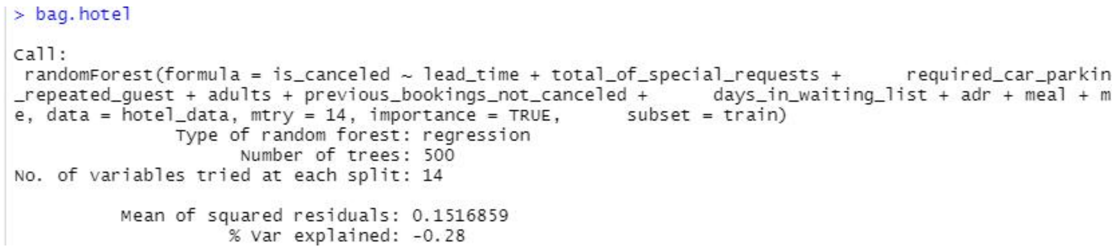


```
> # now we use the cv.tree() function to see whether pruning the tree will improve performancce
> cv.hotel = cv.tree(tree.hotel)
> plot(cv.hotel$size ,cv.hotel$dev ,type='b')
> prune.hotel = prune.tree(tree.hotel, best=5)
> plot(prune.hotel)
> text(prune.hotel,pretty=0)
> yhat=predict (tree.hotel ,newdata=hotel_data[-train ,])
> hotel.test=hotel_data[-train ,"is_canceled"]
> plot(yhat ,hotel.test)
> abline(0,1)
> mean((yhat -hotel.test)^2)
[1] 0.1709994
```

The MSE associated with the regression tree is 0.171. The square root of the MSE is therefore around 0.414. This indicating that this model leads to test predictions that are within around 41% of customer cancelation rate.

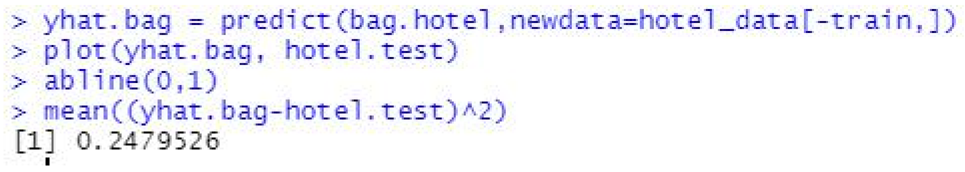
3. Random Forests.

Then we want to try the random forest and bagging. We used randomForest library. The bagging is simply a special case of a random forest with m=p. Therefore, the randomForest() function can be used to perform both random forests and bagging. We first tried to preform bagging.



```
> bag.hotel

Call:
 randomForest(formula = is_canceled ~ lead_time + total_of_special_requests +      required_car_parkin
_repeated_guest + adults + previous_bookings_not_canceled +      days_in_waiting_list + adr + meal + m
e, data = hotel_data, mtry = 14, importance = TRUE,      subset = train)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 14

          Mean of squared residuals: 0.1516859
                    % Var explained: -0.28
```

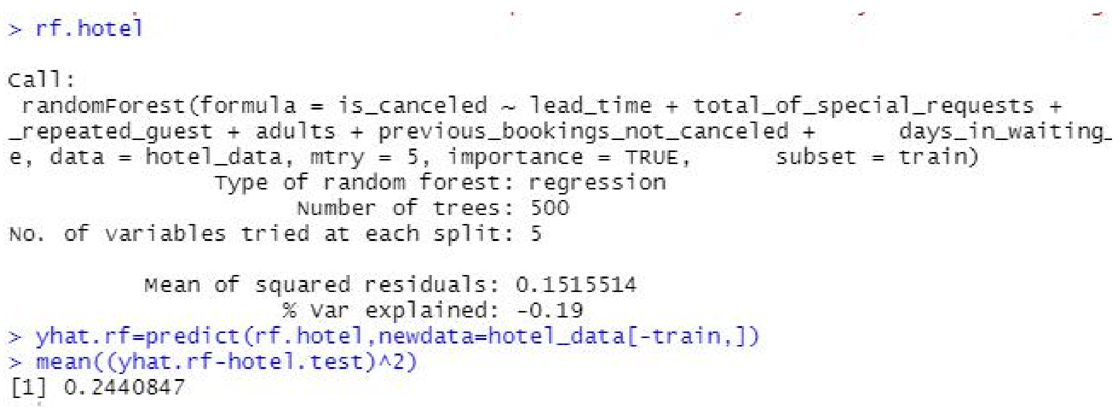
We set the mtry=14. This indicates that all 14 predictors should be considered for each split of the tree, so the bagging should be done. So, we want to check how well the bagged model perform on the test set.



```
> yhat.bag = predict(bag.hotel,newdata=hotel_data[-train,])
> plot(yhat.bag, hotel.test)
> abline(0,1)
> mean((yhat.bag-hotel.test)^2)
[1] 0.2479526
```

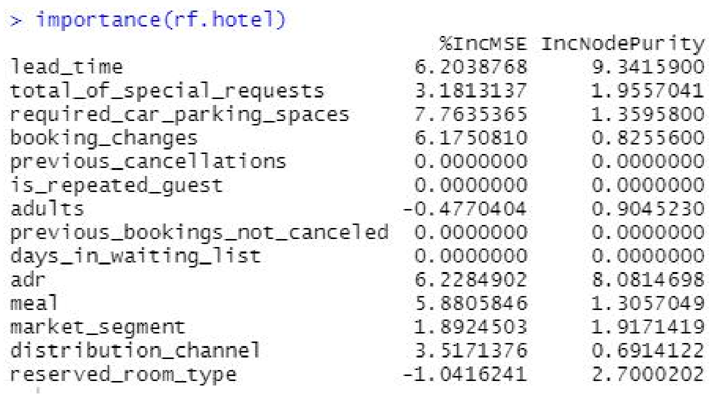
Here we have our MSE that is associated with the bagged regression tree is 0.248. It is better than we thought. If we take a square root of MSE, we get 49.8% cancelation rate.

Let’s try the random forest.



```
> rf.hotel

Call:
 randomForest(formula = is_canceled ~ lead_time + total_of_special_requests +
_repeated_guest + adults + previous_bookings_not_canceled +      days_in_waiting_
e, data = hotel_data, mtry = 5, importance = TRUE,      subset = train)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 5

          Mean of squared residuals: 0.1515514
                    % Var explained: -0.19
> yhat.rf=predict(rf.hotel,newdata=hotel_data[-train,])
> mean((yhat.rf-hotel.test)^2)
[1] 0.2440847
```

We used a smaller value of the mtry here which is 5. We used 14 when bagging, we want use p/3 variables when building a random forest model. We then get an MSE with 0.244, it is slightly smaller than bagging MSE. So, the performance has slightly improved in this case. Then we use importance() function to view the importance of each variable.



```
> importance(rf.hotel)
                                  %IncMSE IncNodePurity
lead_time                       6.2038768     9.3415900
total_of_special_requests       3.1813137     1.9557041
required_car_parking_spaces     7.7635365     1.3595800
booking_changes                 6.1750810     0.8255600
previous_cancellations          0.0000000     0.0000000
is_repeated_guest               0.0000000     0.0000000
adults                         -0.4770404     0.9045230
previous_bookings_not_canceled  0.0000000     0.0000000
days_in_waiting_list            0.0000000     0.0000000
adr                             6.2284902     8.0814698
meal                            5.8805846     1.3057049
market_segment                  1.8924503     1.9171419
distribution_channel            3.5171376     0.6914122
reserved_room_type             -1.0416241     2.7000202
```

We can see that this is really close to our correlation test above.

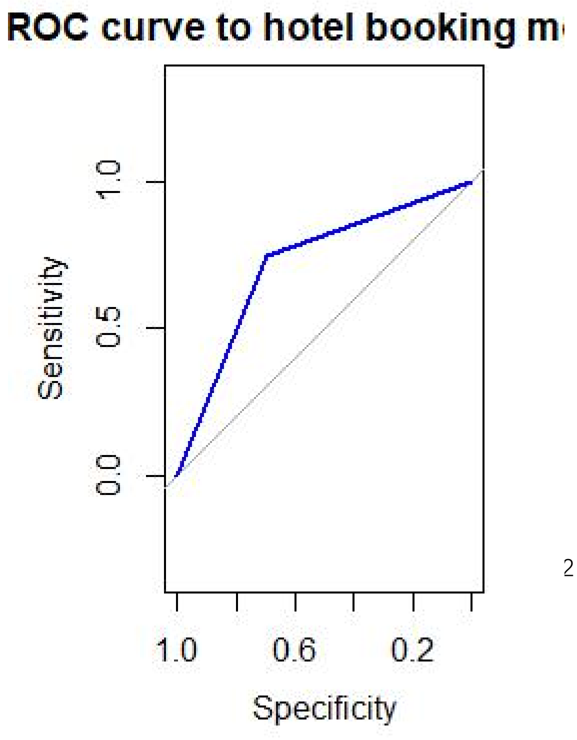
4. Logistic Regression

Next is the Logistic regression model. After we fed in the data in model, we can see those features that effect more on model creation like lead_time, total_of_special_requests and previous_cancellations.

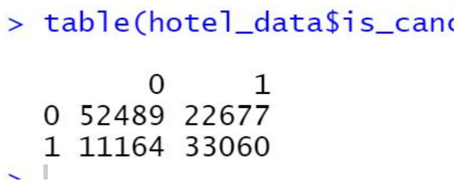
```
> summary(logistic_model)

Call:
glm(formula = is_canceled ~ lead_time + total_of_special_requests + 
    required_car_parking_spaces + booking_changes + previous_cancellations + 
    is_repeated_guest + adults + previous_bookings_not_canceled + 
    days_in_waiting_list + adr + meal + market_segment + distribution_channel + 
    reserved_room_type, family = "binomial", data = hotel_data)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-8.4904  -0.8401  -0.4384   0.9115   5.8769  

Coefficients:
                                 Estimate Std. Error z value Pr(>|z|)    
(Intercept)                    -1.681e+00  1.705e-01  -9.859  < 2e-16 ***
lead_time                       5.002e-03  7.757e-05  64.475  < 2e-16 ***
total_of_special_requests      -8.623e-01  1.160e-02 -74.358  < 2e-16 ***
required_car_parking_spaces    -3.662e+03  7.673e+05  -0.005 0.996192    
booking_changes                -6.114e-01  1.591e-02 -38.424  < 2e-16 ***
previous_cancellations          2.965e+00  5.671e-02  52.279  < 2e-16 ***
is_repeated_guest              -9.266e-01  8.753e-02 -10.586  < 2e-16 ***
adults                          7.856e-02  1.447e-02   5.428 5.69e-08 ***
previous_bookings_not_canceled -5.228e-01  2.547e-02 -20.527  < 2e-16 ***
days_in_waiting_list           -8.181e-04  3.787e-04  -2.160 0.030750 *  
adr                             7.418e-03  1.865e-04  39.766  < 2e-16 ***
mealFB                          6.528e-01  8.837e-02   7.387 1.51e-13 ***
mealHB                         -4.729e-01  2.315e-02 -20.431  < 2e-16 ***
mealSC                          9.878e-03  2.528e-02   0.391 0.695996    
mealUndefined                  -1.027e+00  7.991e-02 -12.853  < 2e-16 ***
```

At this point, we added a new column called "prediction" to store our prediction results in order to convenient compare the prediction and real result. Then we calculated the "p", probability of canceled the booking in dataset and fix the prediction to zero and one by whether it is bigger than p value.

```
> mean(hotel_data$is_canceled == hotel_data$pre
[1] 0.7165508
> mean(hotel_data$is_canceled != hotel_data$pre
[1] 0.2834492
```



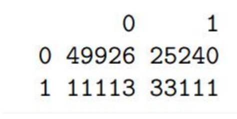
From the code result, we can get an accuracy is around 71.7 percentage and error rate is around 28.3 percentage. We used both ROC curve and confusion matrix to show the train and test evaluation metrics.



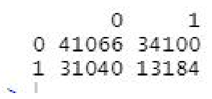
```
> table(hotel_data$is_canc

      0     1
0 52489 22677
1 11164 33060
```

By these evaluation metrics, we can have a conclusion that even though the overall accuracy is at acceptable level 71.7 percentage accuracy, the Type I error rate is much higher than Type II error rate. Type I error occurs mean that we predict this customer will not cancel the booking but he actually cancels it. In this case, Type I error is the one that really affect the profit of hotel. Therefore, it is not a perfect result we wish to get.

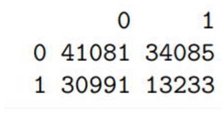
Conclusion



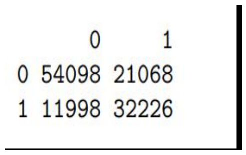
```
      0     1
0 49926 25240
1 11113 33111
```



```
      0     1
0 41066 34100
1 31040 13184
```



```
      0     1
0 41081 34085
1 30991 13233
```



```
      0     1
0 54098 21068
1 11998 32226
```

Above four tables are four confusion matrixes of our final test result for every four models. And the best performance in these four is the last one which is the Logistic regression model. Confusion matrix from Logistic regression model have a 72-percentage accuracy at the end. So apparently, in our project the logistic regression is our best recommendation. And we can see all of these shown a higher Type I error in it, we need to consider more on how to avoid this happen at next time.

This project pushes us to understand those machine learning algorithms and models we were taught in classes. To finish our final project, everything has to be done by ourselves and no more sample code like lab code for us to take as examples. Each step we move forward needs to consider many details and errors behind it. At the same time, because of much practical experience of real coding on machine learning project. We both improve not only the algorithms definition, but also the coding skill like data preprocessing, picking and designing model, and testing prediction results.

However, I have to say we did not do a perfect job on this final project. Because the lack of understanding and practicing those complicated machine learning algorithms, we did not challenge those hard algorithms. At the same time, we did not optimize our model to the best by parameter fixing. These are points we realized when we were doing this final project. Thus, if we have change to make any progress on this project. First, we wish to challenge the other harder algorithms to create model. And then, try more parameters when we are designing our models and utilize more tools like cross-validation to make our test result better.