DSEairline passengers satisfaction

Sofia Gervasoni

July 13, 2022

Abstract

The aim of this paper is to guide the DSEairline company in determining which are the factors that influence the passenger satisfaction, and classify the passengers as *Satisfied* or *Not satisfied*. To reach this goal, three supervised learning classification techniques have been applied: logistic regression, decision tree and random forest. To improve the results of logistic regression, as there was a variable that resulted not statistically significant, I implemented the Lasso penalized logistic regression,improving the results of my model. Regarding the decision tree learning algorithm, the pruning and partitioning techniques have been applied to improve the performance of the algorithm. Lastly, the random forest algorithm has been applied, but before doing so hyper-parameter tuning was implemented. To analyse and compare the performances of the different algorithms, I used their accuracy and their confusion matrices. From all the different techniques applied, *Entertainment* resulted the most influential feature and the method that provided the best performance is the *Random Forest*, even if it is the method with the highest running time.

Contents

1 Goal of the analysis and a brief introduction

Starting from the dataset Airline Customer Satisfaction (available on Kaggle), the goal of this project is to guide the DSEairline company in determining which are the factors that influences the passenger satisfaction. Customer satisfaction plays a major role in affecting the business of a company therefore analysing and improving the factors that are closely related to customer satisfaction is important for the growth and reputation of a company. Moreover, using different supervised learning techniques we will be able to classify customers in two categories: "Satisfied" and "Unsatisfied".

After some data manipulation, cleaning of the dataset and imputation of training and test set, I started my analysis using the Logistic Regression technique (as we have a binary response variable). Fortunately, I did not met multicollinarity issues, but I decided to implement a Lasso penalized logistic regression as the coefficient related to the variable *Departure Delay* resulted not statistically significant. Actually, Lasso penalized logistic regression suggested the removal of the variable *Departure Delay* (as it set its coefficient equal to zero), so I decided to refit my model (on the training set) without this variable. So, I used this model to make predictions (on the test set).

The second classification method used is the Decision Tree. I trained the Decision tree learning algorithm on the training set previously defined and I used it to make prediction on the test set. As I was not satisfied by the performance of the decision tree obtained, I applied pruning and partitioning techniques. Pruning techniques did not implemented the performance of my decision tree predictor, wheres the partitioning techniques improved my results.

Lastly, I decided to apply the Random Forest technique, but before I tuned the hyperparameters to find the ones for which the random forest return the best results.

In all the cases I used the accuracy to evaluate the performance of the different learning algorithms and I summarized the performances of the classification algorithms using the confusion matrix.

2 Top 4 findings

- According to logistic regression, the factors that have the greatest influence on the satisfaction of the passengers are the **loyalty** of the costumer (if the customer is loyal it has an higher probability of being satisfied) and the **entertainment** offered by the company (if the customer gave an higher rate to the entertainment, is more likely to be satisfied);
- According to the decision tree predictor (after partitioning), the most important variables to determine if a client is satisfied or not are: **entertainment**, **seat comfort**, **online booking** and **online support**;
- According to the random forest algorithm (considering both accuracy-based importance and Gini-based importance), the most important features are **entertainment**, **seat comfort** and **online support**, whereas the less important are **inflight Wifi** and **gate location**;
- The best classification performance is obtained with Random Forest (highest accuracy), even if it takes the highest amount of time compared with the other learning algorithms.

3 Data

As previously said, the dataset used is referred to the satisfaction of the passengers of the DSEairline company (for privacy reasons the name of the company is not specified, so I decided to call it *DSEairline*), collected through surveys.
The dataset contains 129.880 observations and 23 variables. The response variable is *Satisfied*, it is a binary categorical variable, which takes value 1 if the passenger is "satisfied" (by the service offered by the company) or 0 if the passenger is "neutral or not satisfied". There are 4 qualitative variables and 18 quantitative variables.

The qualitative variables are: gender (Male or Female), customer type (Loyal customer or Disloyal customer), type of travel (Business or Personal travel) and class (Eco, EcoPlus or Business).

Between the quantitative variables there are: age, flight distance,departure delay, arrival delay and the evaluations to some serviced offered by the airline company (rated from 0 to 5). The evaluated services are: seat comfort, time convenient, food and drink, gate location, in-flight WiFi (0 if not available), entertainment, online support, online booking, onboard service, leg-room service, baggage handling, check-in, cleanliness, and online boarding. Note that all the variables that refer to the scores for the different services offered by the company has been treated as numeric. The main reason for keeping them numeric is traceable to the fact that we could be interested in the average values assumed by the different scores, this could be useful to understand which services have the greatest (or lowest) scores.

More in detail, we can describe the 23 variables as:

- **Satisfied**. it is the response variable and takes values {0,1}, where 0 represent an unsatisfied customer and 1 represent a satisfied customer. I used it as a factor variable;
- **Gender**. it represent the gender of the passenger and takes values {Female, Male}. It has been considered as a factor variable, where Female is the baseline;
- **Customer type**. it gives information about the loyalty of the customer and takes values {Loyal customer, Disloyal customer}. It has been considered as a factor variable and take Disloyal customer as baseline;
- **Age**. it gives information about the age of the passenger and it is a numeric variable. The ages ranges between 7 and 85, with a mean between 39 and 40;
- **Type of travel**. it gives information about the type of travel and takes values {Personal travel, Business travel}. It has been considered as a factor variable where Business Travel is the baseline;
- **Class**. it gives information about the class in which the passenger travelled and it takes values {Eco, EcoPlus, Business}. It has been considered as a factor variable where Business is the baseline;
- **Flight Distance**. it gives information about the length (in miles) of the flight. It is a numeric variable and takes values that range between 50 and 6951, with an average fight distance of 1981 miles;
- **Seat Comfort**. it represent the satisfaction of the customer with respect to the comfort of the seats. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 2.839, and I decided to keep this variable as numeric;
- **Time convenient**. it represent the satisfaction of the customer with respect to the time convenient (both for departure and arrival). It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 2.991 , and I decided to keep this variable as numeric;
- **Food and Drink**. it represent the satisfaction of the customer with respect to Food and Drink. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 2.852, and I decided to keep this variable as numeric;
- **Gate Location**. it represent the satisfaction of the customer with respect to location of the departure gate. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 2.99, and I decided to keep this variable as numeric;
- **In-flight WiFi**. it represent the satisfaction of the customer with respect to the in-flight WiFi. It takes values that ranges between 0 (In-flight WiFi not available) and 5 (highest score) with a mean of 3.249, and I decided to keep this variable as numeric;
- **Entertainment**. it represent the satisfaction of the customer with respect to the in-flight entertainment offered by the company. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 3.383, and I decided to keep this variable as numeric;

- **Online Support**. it represent the satisfaction of the customer with respect to the online support. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 3.52, and I decided to keep this variable as numeric;
- **Online Booking**. it represent the satisfaction of the customer with respect to the online support. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 3.472, and I decided to keep this variable as numeric;
- **Onboard Service**. it represent the satisfaction of the customer with respect to the service onboard. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 3.465, and I decided to keep this variable as numeric;
- **Leg room Service**. it represent the satisfaction of the customer with respect to the Leg Room service. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 3.486, and I decided to keep this variable as numeric;
- **Baggage handling**. it represent the satisfaction of the customer with respect to the Baggage Handling. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 3.696, and I decided to keep this variable as numeric;
- **Check-in**. it represent the satisfaction of the customer with respect to the check-in. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 3.341, and I decided to keep this variable as numeric;
- **Cleanliness**. it represent the satisfaction of the customer with respect to the cleanliness of the plane. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 3.706, and I decided to keep this variable as numeric;
- **Online boarding**. it represent the satisfaction of the customer with respect to the boarding online. It takes values that ranges between 0 (lowest score) and 5 (highest score) with a mean of 3.353, and I decided to keep this variable as numeric;
- **Departure delay**. it represent the departure delay of the flight on which the passenger is travelling (in minutes). It takes values that ranges between 0 and 1592 with a mean of 14.71, and I decided to keep this variable as numeric;
- **Arrival delay**. it represent the arrival delay of the flight on which the passenger is travelling (in minutes). It takes values that ranges between 0 and 1584 with a mean of 15.09, and I decided to keep this variable as numeric. This variable presented 393 NAs, and I will show later how to treat them.

```
satisfied    gender              customer_type        age              type_of_travel       class       flight_distance  seat_comfort
0:58793   Female:65899   disloyal Customer: 23780   Min.   : 7.00   Business travel:89693   Business:62160   Min.   :  50    Min.   :0.000
1:71087   Male  :63981   Loyal Customer   :106100   1st Qu.:27.00   Personal Travel:40187   Eco     :58309   1st Qu.:1359    1st Qu.:2.000
                                                     Median :40.00                           Eco Plus: 9411   Median :1925    Median :3.000
                                                     Mean   :39.43                                            Mean   :1981    Mean   :2.839
                                                     3rd Qu.:51.00                                            3rd Qu.:2544    3rd Qu.:4.000
                                                     Max.   :85.00                                            Max.   :6951    Max.   :5.000

time_convenient   food_drink    gate_location       Wifi       entrateinment   online_support online_booking  onboard_service legroom_service
Min.   :0.000   Min.   :0.000   Min.   :0.00   Min.   :0.000   Min.   :0.000   Min.   :0.00   Min.   :0.000   Min.   :0.000   Min.   :0.000
1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.00   1st Qu.:2.000   1st Qu.:2.000   1st Qu.:3.00   1st Qu.:2.000   1st Qu.:3.000   1st Qu.:2.000
Median :3.000   Median :3.000   Median :3.00   Median :3.000   Median :4.000   Median :4.00   Median :4.000   Median :4.000   Median :4.000
Mean   :2.991   Mean   :2.852   Mean   :2.99   Mean   :3.249   Mean   :3.383   Mean   :3.52   Mean   :3.472   Mean   :3.465   Mean   :3.486
3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:4.00   3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:5.00   3rd Qu.:5.000   3rd Qu.:4.000   3rd Qu.:5.000
Max.   :5.000   Max.   :5.000   Max.   :5.00   Max.   :5.000   Max.   :5.000   Max.   :5.00   Max.   :5.000   Max.   :5.000   Max.   :5.000

baggage_handling    checkin       cleanliness    online_boarding departure_delay   arrival_delay
Min.   :1.000    Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :   0.00   Min.   :   0.00
1st Qu.:3.000    1st Qu.:3.000   1st Qu.:3.000   1st Qu.:2.000   1st Qu.:   0.00   1st Qu.:   0.00
Median :4.000    Median :3.000   Median :4.000   Median :4.000   Median :   0.00   Median :   0.00
Mean   :3.696    Mean   :3.341   Mean   :3.706   Mean   :3.353   Mean   :  14.71   Mean   :  15.09
3rd Qu.:5.000    3rd Qu.:4.000   3rd Qu.:5.000   3rd Qu.:4.000   3rd Qu.:  12.00   3rd Qu.:  13.00
Max.   :5.000    Max.   :5.000   Max.   :5.000   Max.   :5.000   Max.   :1592.00   Max.   :1584.00
                                                                                  NA's   :393
```

Figure 1: Variables summary

3.1 Data manipulation

Before starting the analysis it is important to clean the dataset and manipulate a little bit the data. In particular, in the initial dataset the variable *Satisfied* was codified as ["Satisfied", "Neutral or not Satisfied"], so I encoded it with the One-hot-Encoding technique (0 for "Neutral or not Satisfied" and 1 for "Satisfied"). Moreover, I transformed in factors all the character variables and I dropped the variable *Id* as it is useless for my analysis.

3.2 NAs

In the dataset there are 393 (0.3%) missing values (NAs) concentrated in the variable *Arrival Delay* (as showed in Figure 16). To deal with NAs without losing observations I decided to predict the missing values using the Decision Tree technique to approximate its values.

3.3 Data visualization

The dataset results to be balanced as in the response variable 45% are "Not-Satisfied" (0) while the 55% are "Satisfied" (1). In particular there are 58793 "Satisfied" and 71087 "Not-Satisfied". This is showed in Figure 17. The rule of thumb which followed to determine if a dataset is balanced or not state that, when the minority class in a binary classification is not less than 20%, class imbalance would not impact the model performance much.

With respect to the variable *Age*, calculating the frequencies and plotting them in a barplot, it is possible to observe that people which age range between 7 and 38 or between 61 and 80 are mostly unsatisfied, whereas people aged between 40 and 60 result mostly satisfied (see Figure 18, where 0 are "Unsatisfied" and 1 are "Satisfied"). Always referring to Age, from the boxplot in Figure 19, it is possible to notice that satisfied people are older than the satisfied (on average), as the median of the first distribution is lower than the second one. This variable does not seem to present outliers.

As it possible to observe from Figure 20, there is no evident relation between flight distance and satisfaction.

Another variable present in the dataset is Gender that takes values ["Female","Male"]. As it is possible to notice from Figure 21, females result mostly satisfied, while males result mostly unsatisfied.

Classifying costumer by their loyalty, we can notice that "Loyal Customers" result mostly satisfied, whereas "Disloyal Customers" result mostly unsatisfied (see Figure 22).

People travelling for business, result more satisfied than people travelling for personal travels. Actually, most of the people traveling for business result satisfied, whereas most of the people travelling for personal reasons result unsatisfied. This has been showed in Figure 23.

Classifying customers by class of travel, we can notice that most of the people travelling in business class result satisfied whereas most of the people travelling in Eco and Ecoplus classes resulted unsatisfied. This results have been plotted in Figure 24.

3.4 Outliers

To detect the presence of the outliers I used the interquartile rule and the related visual tool (the box plot). Following this rule it is possible to define as extreme outliers the values that are below the lower limit and the values that are above the upper limit. I considered extreme outliers the observations which are more than 3 times the interquartile range below the first quartile or above the third quartile.

$$\text{lower limit} = Q1 - (3 \times IQR)$$

$$\text{upper limit} = Q3 + (3 \times IQR)$$

Where $Q1$ represents the first quartile, $Q3$ represents the third quartile and IQR represents the interquartile range ($Q3 - Q1$).

Using this technique, I found out:

- 61 extreme outliers for the variable Flight Distance;
- 11.608 extreme outliers for the variable Departure delay;
- 10.825 extreme outliers for the variable Arrival delay.

As I have many observations, I decided to drop all the observations containing extreme outliers, remaining with 116.933 observations.

4 Correlation

4.1 Qualitative variables

In order to calculate the correlation between the qualitative variables we can use the *gamma coefficient* (also called the *gamma statistic*, or *Goodman and Kruskal's gamma*). Gamma tests for an association between points and tells us the strength of association.
The gamma coefficient ranges between -1 and 1. It takes value 1 when there is perfect positive correlation (if one value goes up, so does the other), -1 when there is perfect inverse correlation (as one value goes up, the other goes down) and 0 when there is no association between the variables. So, the closer you get to a 1 (or -1), the stronger the relationship.

In this specific case, the gamma coefficient is almost 0 in most of the cases. Even if the association from Class to Type of Travel is *(x,y)=0,31*, as indicated by the ellipse in the (4,3)-element of this plot array. In contrast, the opposite association - from Type of Travel to Class has a value of only 0,25. As noted, this result means that Type of Travel is highly predictable from Class, but Type of Travel provides less information about Class. Figure 2 provides a visual evidence of what said up to now.

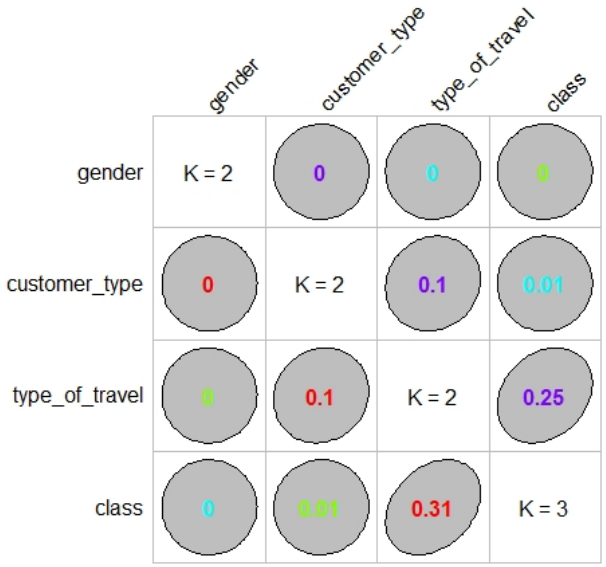

Figure 2: Goodman and Kruskal's gamma (qualitative variables)

4.2 Quantitative variables

For the quantitative variables Pearson correlation has been used. Also in this case the values of this metrics range between -1 (perfect negative correlation) and 1 (perfect positive correlation), and take value 0 in case of absence of correlation.

As it is possible to notice looking at the corrplot in Figure 3, between most of the variables there is very low correlation. Nevertheless, there is a strong positive correlation (0.74) between *Arrival Delay* and *Departure Delay*. In a regression problem this could lead to problems of multicollinearity, but we will see later how to eventually treat this issue.
Also between *Food & Drink* and *Seat comfort* there is a high positive correlation (0.72). The variable *Online Boarding* is quite strong correlated with *Inflight Wifi* (0.63), *Online support* (0.66) and *Online booking* (0.7).
There is a negative correlation between the flight distance and the age, this could be due to the fact that older people may not want to face too long flights.

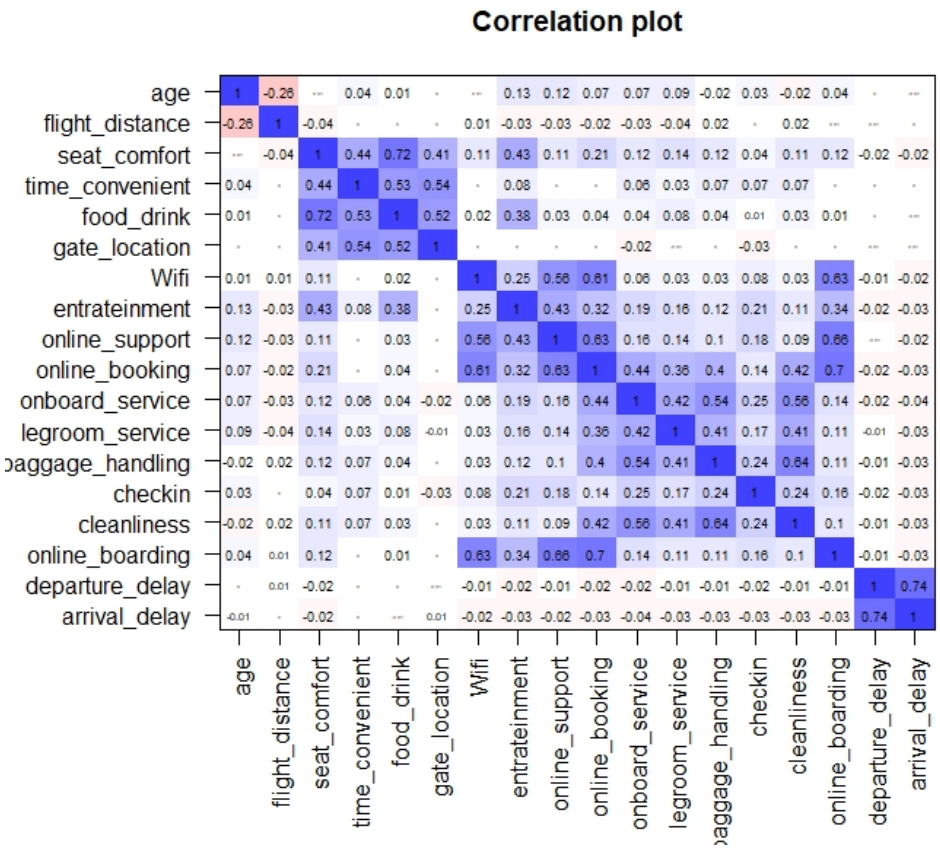

Figure 3: Pearson correlation (quantitative variables)

5 Train set and Test set imputation

Before starting the implementation of the supervised learning techniques, we split our dataset into train and test set. In this case, the training set contains the 80% of the observation, whereas the test set contains the remaining 20%. From Figure 4 it is possible to notice that the two sets remain balanced, as the difference between the two classes (0 and 1) still be more or less 20%.

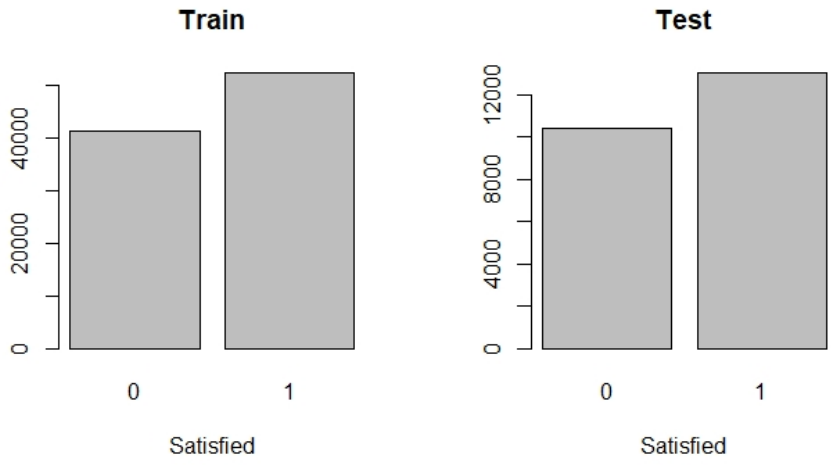

Figure 4: Satisfied (Y) distribution in train and test set

6 Logistic regression

The first classification method implemented is the logistic regression. Logistic regression is used to predict the class (or category) of individuals based on one or multiple predictor variables (x). In this case I decided to use logistic regression because I have a binary outcome (0 or 1, "Non-satisfied" or "Satisfied").
Logistic regression belongs to a family, named Generalized Linear Model (GLM), developed for extending the linear regression model to other situations. Logistic regression does not return directly the class of observations. It allows us to estimate the probability (p) of class membership. The probability will range between 0 and 1. You need to decide the threshold probability at which the category flips from one to the other. By default, this is set to p=0.5, but in reality it should be settled based on the analysis purpose (I decided to set p=0.5).

I first created a model introducing all the variables, and they all their coefficients resulted to be statistically different from zero, except Departure delay that does not result statistically significant. Moreover, calculating the McFadden's pseudo R2, it resulted equal to 0.4438084. Values ranging from 0.2 to 0.4 indicates very good model fit, our McFadden's pseudo R2 very close to 0.4 so we have a good fit.

The Variance Inflation Factor (VIF), is a metric to understand if the model is affected by multicollinearity. VIF shows that how much the variance of the coefficient estimate is being inflated by multicollinearity. It is calculate as the inverse of the *tollerance*.

$$VIF = 1/(1 - R^2)$$

A rule of thumb for interpreting the variance inflation factor (VIF) says that if VIF=1 there is no correlation between the variable and the others, if VIF is between 1 and 5 than the variable is moderately correlated with the others wheres if it is greater than 5 the variable is highly correlated with the others.

Variable	**VIF**
Food&Drink	2.76
Online booking	3.52
Departure delay	2.18
Arrival delay	2.19

In the table just above I summarized the highest VIF values. As it is possible to notice, all these values are between 1 and 5, so there is a moderate correlation. At this point we can say that there is no multicollinearity.

Even if we do not have problems of multicollinearity, we have a variable that resulted not statistically significant. To deal with this variable, I introduced the Lasso penalized logistic regression.

6.1 Lasso penalized logistic regression

To deal with the problem stated in the previous section, we can use regularization, that means to keep all the features but reducing the magnitude of the coefficients of the model. This is a good solution when each predictor contributes to predict the dependent variable, as in our case. There are two main regularization methods, LASSO Regression and Ridge Regression, that only differs for their penalty function. Both can be used in logistic regression. I decided to apply the LASSO regression.

First I used the CV on the training set to choose the best value of labda (*lambda.min*), which is the one that minimizes the prediction error. In this case lambda takes value 0.0005113412. Once the value of lambda is determined, we could fit the model using the training set with the command *glmnet()* and specifying as lamba parameter the value obtained from cross-validation (hyperparameter tuning). This model is then used to make predictions on the test set. As previously said, logistic regression does not return directly the class of observations, but it allows us to estimate the probability (p) of class membership. So, I set as trash-old p=0.5, classifyng as 1 ("Satisfied") predictions

with probability higher than p, and 0 ("Unsatisfied") otherwise. This model presented an accuracy of 0.836013. The lasso penalized logistic regression set to zero the variable Departure delay that was the one which resulted not statistically significant from the previous model.

Generally, the purpose of regularization is to balance accuracy and simplicity. This means, a model with the smallest number of predictors that also gives a good accuracy. To this end, the function cv.glmnet() finds also the value of lambda that gives the simplest model but also lies within one standard error of the optimal value of lambda. This value is called lambda.1se. So, I retrained the model on the training set, but this time I used as parameter *lambda.1se*. Again, I used this model to make prediction on the test set and I used as threshold p=0.5. This time I obtained an accuracy equal to 0.8354571 (a little bit lower than before), and even in this case the only variable dropped from the model is Departure delay, which is set to zero. Even with lambda.1se, the obtained accuracy remains good enough.

6.2 Logistic regression (without Departure Delay)

In the previous section, we have seen that using both *lambda.min* and *lambda.1se* the coefficient settled to zero is the one related with Departure delay. So, I tried to refit the model dropping this variable.

Once again we fit the model on the training set and we evaluate the model using the McFadden's Pseudo R2. In this case it resulted equal to 0.4437999: a little bit higher than before but always close to 0.4, to the model improved. Calculating the VIF, it is also possible to notice that again we do not have any problems of multicollineairity (see Figure 5).

	GVIF	Df	GVIF^(1/(2*Df))
gender	1.068171	1	1.033524
customer_type	1.553565	1	1.246421
age	1.245366	1	1.115960
type_of_travel	1.991285	1	1.411129
class	1.670070	2	1.136799
flight_distance	1.192599	1	1.092062
seat_comfort	2.345967	1	1.531655
time_convenient	1.708123	1	1.306952
food_drink	2.758950	1	1.661009
gate_location	1.537375	1	1.239909
Wifi	2.205446	1	1.485075
entrateinment	1.570169	1	1.253064
online_support	1.994132	1	1.412137
online_booking	3.520366	1	1.876264
onboard_service	1.626736	1	1.275436
legroom_service	1.229873	1	1.108996
baggage_handling	1.844335	1	1.358063
checkin	1.170885	1	1.082074
cleanliness	1.991690	1	1.411272
online_boarding	2.537192	1	1.592856
arrival_delay	1.006698	1	1.003343

Figure 5: VIF

Even if there are no problems of multicollinearity, we need to check the other conditions for the logistic model. In particular, there aren't problems of linearity (actually there is a linear relationship between the logit of the outcome and each predictor variables), but there are influential values, so not all the assumptions of the logistic regression hold. To solve this problem I dropped the rows influential values

and I refitted a new model. At this point all the assumption under the logistic regression hold.
Also in this case we use the test set for making predictions and we set p=0.5. Once again we calculate the accuracy of this learning algorithm that in this case is equal to 0.8364406. Moreover we plot the confusion matrix to have a visual representation of the observations that have been correctly classified. As it is possible to notice from Figure 6, there are: 9106 true negative, 10455 true positive, 2552 false negative and 1273 false positive.

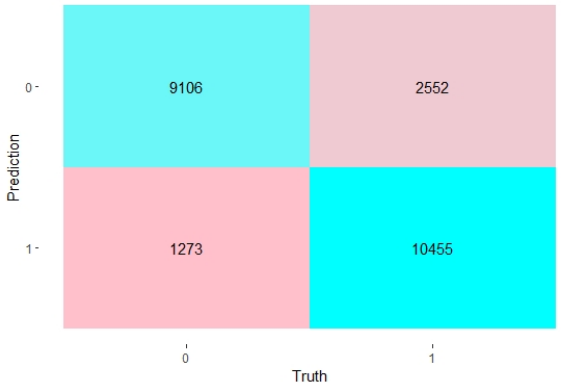

Figure 6: Confusion matrix (Logistic Regression)

To obtain an aggregate measure of performance across all possible classification thresholds we use the AUC (Area under the ROC Curve). It is s the probability that a classifier will be more confident that a randomly chosen positive example is actually positive than that a randomly chosen negative example is positive. AUC-ROC curve helps us visualize how well our learning classifier is performing. When AUC between 0.5 and 1, there is a high chance that the classifier will be able to distinguish the positive class values from the negative class values. This is so because the classifier is able to detect more numbers of True positives and True negatives than False negatives and False positives. In our case AUC=0.908, so we are really close to 1, meaning that we have a good classifier (see Figure 7).

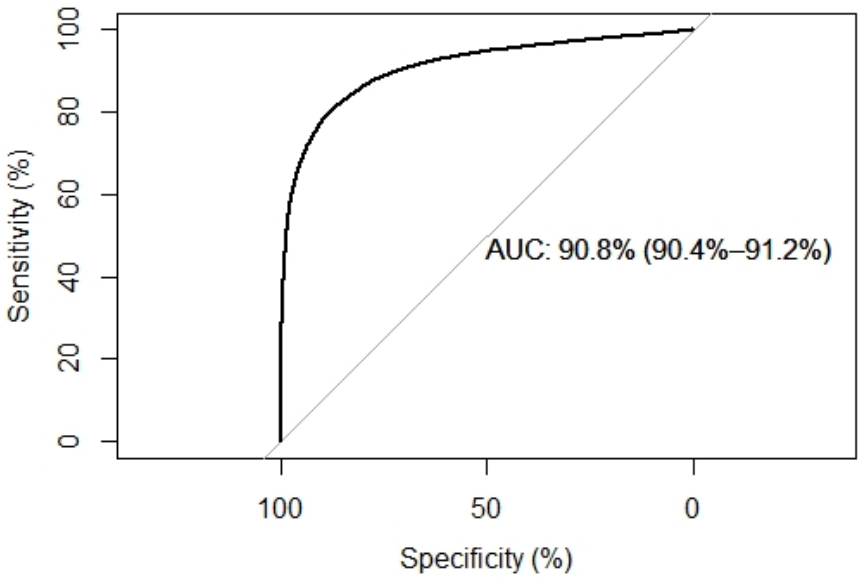

Figure 7: AUC-ROC curve

6.2.1 Interpretation of the coefficients

To understand which are the key factors on which DSEairline should work to improve the satisfaction of its costumers, we can study the coefficients of the model built with logistic regression. As it is possible to notice from Figure 8, the coefficients are all statistically significant (associated with a p-value lower than 0.05), so our model suggests that all these factors in fact influence the satisfaction of DSEairline passengers.

Which are the factors that have the greatest influence on customer satisfaction? How do these factors influence the odds of being satisfied or not?

The logistic regression coefficient *beta* associated with a predictor X is the expected change in log-odds of having the outcome per unit change in X. So increasing the predictor by 1 unit (or going from 0 to 1) multiplies the odds of having the outcome by *exp(beta)*.

As it is possible to notice from Figure 8, the highest coefficient is the one associate with the **Type of Costumer**: passing from being a disloyal costumer to a loyal one, increase 7.65 times the odds of the costumer being satisfied rather than not. The **Entertainment** service also have a great influence on the satisfaction of the customer, actually, increasing by one unit the satisfaction of the customer about entertainment, increase almost 2 times the odds of the costumer being satisfied rather than not. **Gender** is the variable that has the greatest negative influence on the probability of the passengers being satisfied. Actually, passing from being a female to being a male the probability of being satisfied decrease by 63.1%. Also the **Class** influences the satisfaction of the customer: passing from Business to Eco (EcoPlus) the probability of the customer being satisfied decrease by 52% (55%). Also the **Type of travel** influences the odds of the passengers being satisfied: passing from business to personal travel decrease the probability of being satisfied by 54 %. **Age, flight distance and arrival delay** do not have a great influence on the satisfaction of the customers.

	Coefficients (exp)	Coefficients (%)
(Intercept)	0.00	-99.89
genderMale	0.37	-63.10
customer_typeLoyal Customer	7.65	665.33
age	0.99	-0.90
type_of_travelPersonal Travel	0.46	-54.05
classEco	0.48	-52.02
classEco Plus	0.44	-55.57
flight_distance	1.00	-0.01
seat_comfort	1.36	35.61
time_convenient	0.82	-18.01
food_drink	0.79	-21.42
gate_location	1.13	12.79
Wifi	0.90	-10.36
entrateinment	2.06	105.55
online_support	1.10	10.37
online_booking	1.25	25.41
onboard_service	1.38	38.34
legroom_service	1.26	26.25
baggage_handling	1.12	11.61
checkin	1.36	36.05
cleanliness	1.06	5.86
online_boarding	1.21	21.22
arrival_delay	0.98	-1.82

Figure 8: Coefficients Logistic Regression

7 Decision tree

Decision trees are supervised learning algorithms used to perform both regression and classification tasks. Each decision tree has nodes and branches, where the tests on each attribute are represented at the nodes, the outcome of this procedure is represented at the branches and the class labels are

represented at the leaf nodes. As our response variable is categorical, we have a Categorical Variable Decision Tree which refers to the decision trees whose target variables have limited value and belong to a particular group.

I build my tree predictor using the command *tree()*. My response variable is always Satisfied, and I decided to introduce all the variables. I fit my model on the training set and I used this model to make prediction on the test set. Then, I plotted the confusion matrix (Figure 9) and I calculated the accuracy of this predictor. It resulted an accuracy of 0.8756521, with 1457 false positive, 1451 false negative, 11556 true positive and 8922 true negative. Even if the accuracy of the model is higher than the one referred to the logistic regression (in section 6.2), the number of false positive is increasing. This is riskier because we are classifying more people as satisfied than in reality.

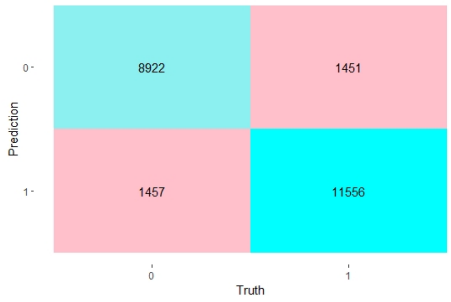

Figure 9: Confusion matrix (decision tree)

7.1 Pruning

Pruning refers to the process wherein the branch nodes are turned into leaf nodes which results in the shortening of the branches of the tree. The essence behind this idea is that over-fitting is avoided by simpler trees as most complex classification trees may fit the training data well but do an underwhelming job in classifying new values.
We can use cross-validation to select a good pruning of the tree. This could be done using the command *cv.tree()* that runs a K-fold (k=10) cross-validation experiment to find the deviance or number of misclassifications as a function of the cost-complexity parameter k. In Figure 10, we can observe that the tree size that minimize the CV-RMSE is 10, we will prune to a size of 10.

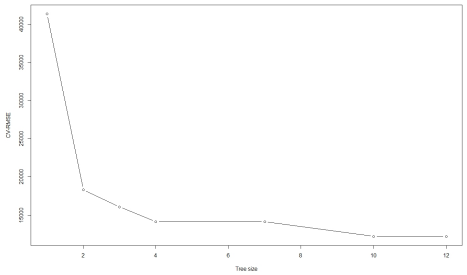

Figure 10: Cross validation

Unfortunately this not improve our performance as we get the same confusion matrix (Figure 9) . So, I will try to use partitioning, and see if this improve the performance of my decision tree.

7.2 Partitioning

Partitioning consist in splitting the data set into subsets. The decision of making strategic splits greatly affects the accuracy of the tree. Many algorithms are used by the tree to split a node into

sub-nodes which results in an overall increase in the clarity of the node with respect to the target variable. To implement the partitioning in R, I used the command rpart(). Once again I trained the model on my training set and I used it to make prediction on my test set. With this technique I was able to improve the performance of the learning algorithm that in this case present an accuracy of 0.8808652, even if the number of false positive increased. In particular, I obtained 1537 false positive, 1261 false negative, 11746 true positive and 8942 true negative (see Figure 11).

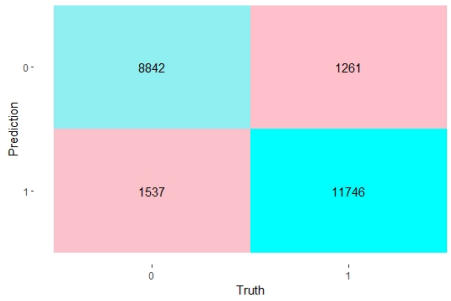

Figure 11: Confusion matrix (decision tree with partitiong)

7.2.1 Interpretation of the tree obtained with Partitioning

In Figure 12 it is plotted the tree obtained with partitioning. Looking at this tree we could say that the features that determine the satisfaction of the client in this case are: entertainment, seat comfort, online booking and online support. Once again **Entertainment** is between the most important features and in particular is the variable that determine the first split. Other variables like **seat comfort**, **online booking**, **online support**, **class** and **type of customer** are relevant in this case.

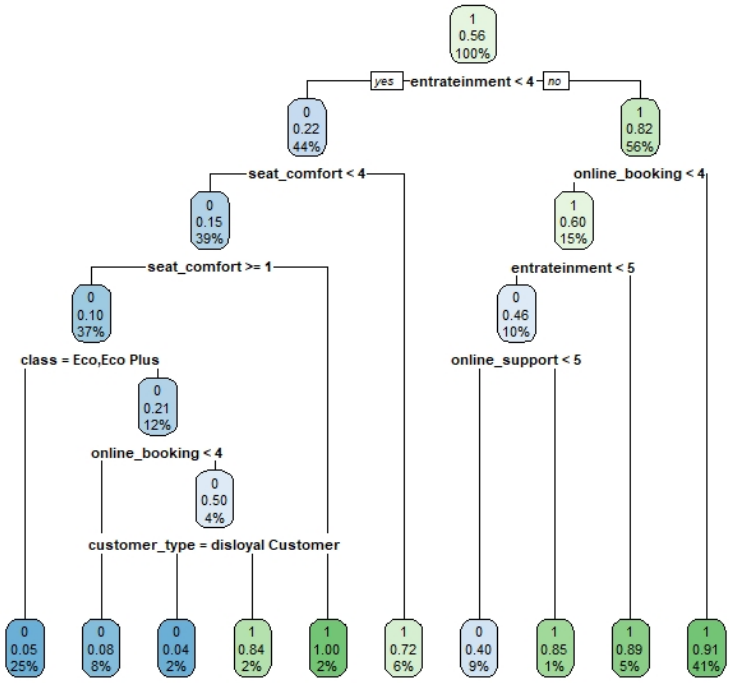

Figure 12: Tree obtained with partitioning

8 Random Forest

Random forest is a machine learning algorithm that uses a collection of decision trees providing more flexibility, accuracy, and ease of access in the output. It is a supervised nonlinear classification and regression algorithm. In R programming, *randomForest()* function of randomForest package is used to create and analyze the random forest. They are called random because they choose predictors randomly at a time of training. They are called forest because they take the output of multiple trees to make a decision. Random forest outperforms decision trees as a large number of uncorrelated trees(models) operating as a committee will always outperform the individual constituent models.

As it choose predictors randomly at a time of training, I set the seed to initialize a pseudorandom number generator. I than apply the *randomForest()* to my training set, keeping all the variables in ma model. Using the model obtained in the previous step, I made prediction on the test set, and to evaluate them I used the confusion matrix (Figure 14).

Before running the command *randomForest()* it is necessary to do some hyper-parameters tuning. in particular, we have to choose mtry (the number of variables randomly sampled as candidates at each split) and *ntree* (the number of trees to grow). I set the ntrees=500 and I tuned the parameter mtry using the command *tuneRF()*. With this command I obtained that the ideal number of variables randomly sampled as candidates at each split (mtry) is 9, as it minimizes the OOB error (Figure 13).

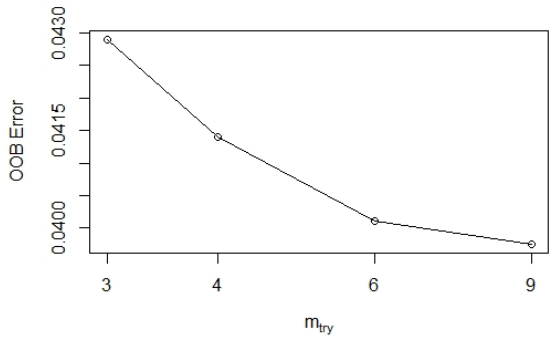

Figure 13: Hyperparamenter tuning (mtry)

As we pointed out that the value of mtry that minimize the OBB error is 9 (so that maximizes the accuracy), it is possible to run the *randomForest()* command.

With this algorithm I obtained an accuracy of 0.9610023, with 10.037 true negative, 12.437 true positive, 570 false negative and 342 false positive (see Figure 14). Looking at this results we could say that with random forest we obtain the best results (compared to the previous methods used), even if this method is the one that takes the highest amount of time.

8.1 Variable importance

After training a random forest, it is natural to ask which variables have the most predictive power. Variables with high importance are drivers of the outcome and their values have a significant impact on the outcome values. By contrast, variables with low importance might be omitted from a model, making it simpler and faster to fit and predict.

There are two measures of importance given for each variable in the random forest: **accuracy-based importance** (based on how much the accuracy decreases when the variable is excluded, the more the accuracy suffers, the more important the variable is for the successful classification) and **Gini-based importance** (based on the decrease of Gini impurity when a variable is chosen to split a node, Gini

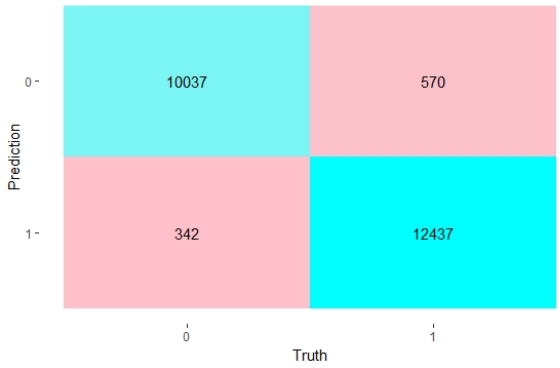

Figure 14: Confusion matrix (Random Forest)

coefficient is a measure of how each variable contributes to the homogeneity of the nodes and leaves in the resulting random forest).

Looking at Figure 15, we could notice that also in this case **entertainment** is between the most important features with **seat comfort** and **online support**. Within the less important features we find **inflight Wifi**, **gate location**, **departure** and **arrival delay**. It is interesting to notice the variable **check-in**, actually, it is the most important variable in terms of Mean Decreasing Accuracy (so dropping this variable the accuracy decreases a lot), but it has a low importance in terms of Mean Decreasing Gini; this means that the variable *check-in* is the most usefull for classfication (in this case) but it gives a low contribute to the homogeneity of the nodes and leaves in the resulting random forest.

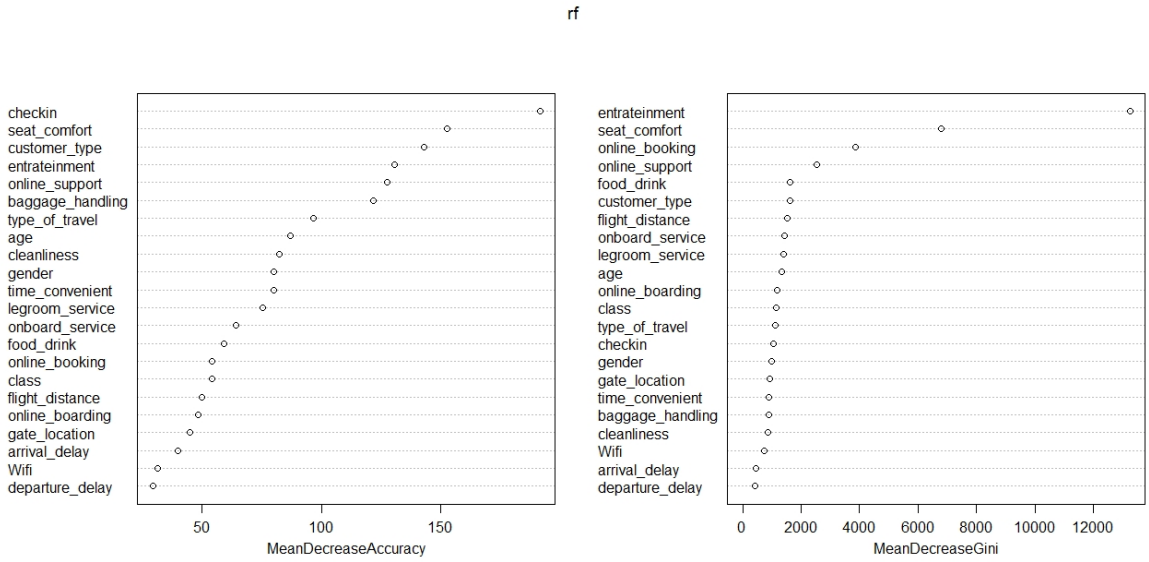

Figure 15: Variable importance (random forest)

9 Conclusions

To summarize, the main key-factor on which the DSEairline should work is *entertainment*. We saw that if the passenger give an high rating to this service, he/she is also more likely to be satisfied.

From Logistic regression, we pointed out that all the variables are statistically significant in explaining the response variable, except for the variable *Departure Delay* that resulted not statistically significant. Moreover, we saw that the *type of customer* strongly influences the classification in satisfied or not. In particular, loyal customers are more likely to be satisfied rather than the disloyal costumers. This is logic, because a customer is less likely to be loyal if it is not satisfied by the services offered by the company. So, the company should work for gaining loyal customers (maybe reserving them special promotions).Always in logistic regression we have seen that *age*, *flight distance* and *arrival delay* do not have a great influence on the satisfaction of the costumers, otherwise, the *gender* has a great influence on the satisfaction. If the costumer is a male, he/she is more likely to be unsatisfied, wheres females result mostly satisfied. Also the *class* and the *type of travel* have a great influence on the satisfaction of the customer.

From the decision tree we have seen that also the seat comfort is a factor that influences the satisfaction of a passenger: if the passenger is satisfied with the comfort of the seat, he/she is also more likely to be satisfied in general. So, if the company want its customer to be satisfied, should work on the comfort of its seats. Other factors on which the company should invest are the *online support* and the *online booking*.

From the random forest, we saw that *entertainment* is between the most important features with *seat comfort* and *online support*, whereas within the less important features we find *in-flight WiFi* and *gate location*.

Comparing the performances of the different classification algorithms we can say that the Random Forest return the best results as its accuracy is the highest (0.9610023) and it also have the lowest number of false positive (passengers classified as satisfied even if unsatisfied). The second best performance (in terms of accuracy) is the one obtained with the decision tree on which we applied the partitioning; in this case we obtain an accuracy of 0.880865, even if the number of false positive is the highest between the method used. Also the decision tree returned a good performance with an accuracy of 0.875652. The worst performance in terms of accuracy is obtained with the Logistic Regression, nevertheless this method return the lowest number of false positive wit respect to the decision tree and the decision tree with partitioning. Even if Random Forest is the best in terms of accuracy is the one that took the highest amount of time compared with the other methods.

The following table summarize the results obtained with the different learning algorithms used:

Learning algorithm	**Accuracy**	**False positive**
Logistic Regression	0.8364406	1273
Decision Tree	0.875652	1457
Decision Tree (Partitioning)	0.880865	1537
Random Forest	0.9610023	342

10 R Code

```
#Dataset
library(readxl)
sat=read_excel(file.choose()) #satisfaction
sat=data.frame(sat)
sat%>% glimpse() #129,880 and 24 columns

sat %>% skimr::skim() #393 NA in Arrival Delay in Minutes
#sat=sat[-which(is.na(sat$Arrival.Delay.in.Minutes)),]
#sat %>% skimr::skim() #drop the missing values in Arrival Delay in Minutes
#(1-(129487/129880))*100 -> lose 0.302587% of of observations

#Data manipulation

sat=sat %>%
  mutate(
    satisfaction_v2=ifelse(satisfaction_v2=="satisfied", 1, 0)
  ) %>%
  mutate(across(where(is.character), as.factor)) %>%
  select( # remove some "useless" variables
    -c(id)
  )

colnames(sat)=c("satisfied","gender", "customer_type", "age", "type_of_travel",
                "class", "flight_distance","seat_comfort","time_convenient",
                "food_drink","gate_location", "Wifi","entrateinment",
                "online_support","online_booking","onboard_service",
                "legroom_service", "baggage_handling","checkin",
                "cleanliness","online_boarding",
                "departure_delay", "arrival_delay")
summary(sat)
#Data Visualization

barplot(table(sat$satisfied)/length(sat$satisfied),
        xlab = "Satisfied (Y)", ylab="Frequences")
(1-(table(sat$satisfied)[1]/table(sat$satisfied)[2]))*100
#balanced dataset


(length(which(sat$satisfied==1))/length(which(sat$satisfied==0)))-1 #21%
#more than 20% but close to it-> more or less BALANCED DATASET
#The rule of thumb which I follow is that when the minority class
#in a binary classification is not less than 20%,
#class imbalance would not impact the model performance much.

ggplot(sat, aes(x=as.factor(satisfied),y=age))+
  geom_boxplot(fill= "darkred", alpha= 0.7)

barplot(table(sat$satisfied, sat$age), beside=T, cex.names=0.7,
        legend.text=T, xlab = "Age", ylab="Frequency")
#as the age increase the satisfaction seems to increase

ggplot(sat, aes(x=as.factor(satisfied),y=flight_distance))+
  geom_boxplot(fill= "darkred", alpha= 0.7)
#no evident relation between flight distance and satisfaction
```

```
#ggplot(sat, aes(x=as.factor(satisfied),y=arrival_delay))+
#  geom_boxplot(fill= "darkred", alpha= 0.7)

barplot(table(sat$gender)) #balanced
g=table( sat$satisfied ,sat$gender)
barplot(g, beside=T, legend.text=T)
#female seems to be more satisfied than male

barplot(table(sat$customer_type)) #unbalanced
t=table(sat$satisfied ,sat$customer_type)
barplot(t, beside=T, legend.text=T)
#loyal customers seems to be more satisfied

barplot(table(sat$type_of_travel)) #unbalanced
tt=table(sat$satisfied ,sat$type_of_travel)
barplot(tt, beside=T, legend.text=T)
#who travel for business seems to be more satisfied

barplot(table(sat$class)) #unbalanced
c=table(sat$satisfied ,sat$class)
barplot(c, beside=T, legend.text=T)
#people that travel in business class seems to be mostly satisfied
#people that travel in Eco class seems to be mostly satisfied
#people that travel in Eco plus class seems to be mostly unsatisfied

#distribution of the variables about satisfaction
x11()
par(mfrow=c(3,5))
for(i in 8:21){
  barplot(table(sat[,i])/length(sat), xlab=colnames(sat)[i], ylab="")
}
par(mfrow=c(1,1))

# Deal with NAs ------------------------------------------------------------
sum(is.na(sat)) #393
library(VIM)
aggr(sat, col = c("white", "grey", "black"), cex.axis=.5)


sat = as_tibble(sat)
sum(is.na(sat$arrival_delay)) #393
(393/129880)*100 #0.3%
missing_index = which(is.na(sat$arrival_delay))
X= sat[missing_index,]
train_v = sat[-c(missing_index),]

library("caret")

tree = caret::train(arrival_delay ~ .,
                    data=train_v,
                    method="rpart")

arrival_delay_pred = predict(tree, newdata = X)

sat[missing_index,"arrival_delay"]=arrival_delay_pred
```

```
sum(is.na(sat$arrival_delay)) #no more NAs
sum(is.na(sat))


# clean Glob_Env
rm(train_v,X,tree, arrival_delay_pred,missing_index)

#Outliers ----------------------------------------------------------------------
library(rstatix)
sum(is_extreme(sat$age)) #0
sum(is_extreme(sat$flight_distance)) #61
sum(is_extreme(sat$departure_delay)) #11608
sum(is_extreme(sat$arrival_delay)) #10825
sat=sat[-unique(c(which(is_extreme(sat$flight_distance)),
                  which(is_extreme(sat$departure_delay)),
        which(is_extreme(sat$arrival_delay)))),]
nrow(sat)
#all the extreme values has been dropped

boxplot(sat$flight_distance)
boxplot(sat$departure_delay)
boxplot(sat$arrival_delay)
#I am left with some outliers but no more extreme outliers
#116933 observation left

#Qualitative Correlation -------------------------------------------------------

dt.gk=sat[,c(2,3,5,6)]
library("GoodmanKruskal")

plot(GKtauDataframe(dt.gk))
#the highest correlation is between class and type of travel

#Quantitative Correlation ------------------------------------------------------
library(corrplot)
correl=cor(sat[,-c(2,3,5,6)])
corrplot(correl)
#correl[which(cor(sat[,-c(2,3,5,6)])>0.75&cor(sat[,-c(2,3,5,6)])<1)]

x11()
corPlot(sat[,-c(1,2,3,5,6)], cex = 0.4, xlas=2,show.legend=F)
#strong positive correlation between arrival delay and departure delay (0.96)
#strong positive correlation between online boarding and wifi (0.63),
#online support (0.67) and online booking (0.68)

#problems of collinearity between departure and arrival delay

#Train and Test ----------------------------------------------------------------

sat=as.data.frame(sat)
set.seed(123)
split_train_test=createDataPartition(y = sat$satisfied, p=0.8, list = F)
#80% training 20% test
train=sat[split_train_test,]
test=sat[-split_train_test,]
```

```
par(mfrow=c(1,2))
barplot(table(train$satisfied), xlab="Satisfied", main="Train")
tabtrain=table(train$satisfied)
(1-(tabtrain[1]/tabtrain[2]))*100 #20.85% (BALANCED)

barplot(table(test$satisfied), xlab="Satisfied", main="Test")
tabtest=table(test$satisfied)
(1-(tabtest[1]/tabtest[2]))*100 #20.20% (BALANCED)
par(mfrow=c(1,1))

#Logistic Regression----------------------------------------------------------

logit=glm(as.factor(satisfied)~., data=train, family = binomial(link = 'logit'))
summary(logit)

library("DescTools")
PseudoR2(logit, which = NULL) #0.4438084
#McFadden's pseudo R2 ranging from 0.2 to 0.4 indicates very good model fit
#our McFadden's pseudo R2 very close to 0.4 -> good fit

#plot(logit)

library("regclass")
round(VIF(logit),2)


#Lasso------------------------------------------------------------------------

library(glmnet)
x=model.matrix(satisfied~.,data=train)[,-1]
y=train$satisfied
fit.lass=glmnet(x, y, family = "binomial", alpha = 1, lambda = NULL)
summary(fit.lass)
#alpha=0 -> Ridge regression
#alpha=1 -> Lasso regression

#fit the lasso penalized model
#library(glmnet)
# Find the best lambda using cross-validation
set.seed(123)
cv.lasso = cv.glmnet(x, y, alpha = 1, family = "binomial")
# Fit the final model on the training data
model = glmnet(x, y, alpha = 1, family = "binomial",
               lambda = cv.lasso$lambda.min)
# Display regression coefficients
coef(model)

# Make predictions on the test data
x.test = model.matrix(satisfied ~., data=test)[,-1]
probabilities = model %>% predict(newx = x.test)
predicted.classes = ifelse(probabilities > 0.5, 1, 0)
# Model accuracy
observed.classes = test$satisfied
mean(predicted.classes == observed.classes) #accuracy: 0.836013
```

```
#library(glmnet)
set.seed(123)
cv.lasso = cv.glmnet(x, y, alpha = 1, family = "binomial")
plot(cv.lasso) #log lambda more or less -7
cv.lasso$lambda.min
cv.lasso$lambda.1se

coef(cv.lasso, cv.lasso$lambda.min)
coef(cv.lasso, cv.lasso$lambda.1se)

### Final model with lambda.min
lasso.model = glmnet(x, y, alpha = 1, family = "binomial",
                     lambda = cv.lasso$lambda.min)
summary(lasso.model)
# Make prediction on test data
x.test = model.matrix(satisfied ~., data=test)[,-1]
probabilities = lasso.model %>% predict(newx = x.test)
predicted.classes = ifelse(probabilities > 0.5, 1, 0)
# Model accuracy
observed.classes = test$satisfied
mean(predicted.classes == observed.classes) #0.836013


### Final model with lambda.1se
lasso.model = glmnet(x, y, alpha = 1, family = "binomial",
                     lambda = cv.lasso$lambda.1se)
coefficients(lasso.model)
summary(lasso.model)
# Make prediction on test data
x.test = model.matrix(satisfied ~., data=test)[,-1]
probabilities = lasso.model %>% predict(newx = x.test)
predicted.classes = ifelse(probabilities > 0.5, 1,0)
# Model accuracy rate
observed.classes <- test$satisfied
mean(predicted.classes == observed.classes) #0.8354571

#keep the penalized lasso model with lambda.1se (Drop Departure delay)

#Logistic Regression (no Departure delay)-----------------------------------------

#Lasso set Departure delay to zero,
#try to refit the logistic regression without this variable
logit_noMulty=glm(as.factor(satisfied)~.-departure_delay, data=train,
                  family = binomial(link = 'logit'))
summary(logit_noMulty)

#library("DescTools")
PseudoR2(logit_noMulty, which = NULL) #0.4437999
#McFadden's pseudo R2 improved a little (closer to 0.4)

#plot(logit)
#library("regclass")

#Assumptions check-----------------
#1. Multicollinearity
VIF(logit_noMulty) #no problems of collinearity
```

```
#2. linearity

# Select only numeric predictors
probabilities <- predict(logit_noMulty, type = "response")
mydata <- train %>%
  dplyr::select_if(is.numeric)
predictors <- colnames(mydata)
# Bind the logit and tidying the data for plot
mydata <- mydata %>%
  mutate(logit = log(probabilities/(1-probabilities))) %>%
  gather(key = "predictors", value = "predictor.value", -logit)

ggplot(mydata, aes(logit, predictor.value))+
  geom_point(size = 0.5, alpha = 0.5) +
  geom_smooth(method = "loess") +
  theme_bw() +
  facet_wrap(~predictors, scales = "free_y")

#linearity holds

#3. Influential values
plot(logit_noMulty, which = 4, id.n = 3)
model.data <- augment(logit_noMulty) %>%
  mutate(index = 1:n())

model.data %>% top_n(3, .cooksd)

ggplot(model.data, aes(index, .std.resid)) +
  geom_point(aes(color = train$satisfied), alpha = .5) +
  theme_bw()

model.data %>%
  filter(abs(.std.resid) > 3) #there are influential points

#remove influential values
new.train=train[-which(abs(model.data$.std.resid)>3),]
barplot(table(as.factor(new.train$satisfied))) #balanced


logit_noMulty_noout=glm(as.factor(satisfied)~.-departure_delay, data=new.train,
                        family = binomial(link = 'logit'))
probabilities <- predict(logit_noMulty_noout, type = "response")

plot(logit_noMulty_noout, which = 4, id.n = 3)
model.data <- augment(logit_noMulty_noout) %>%
  mutate(index = 1:n())

model.data %>% top_n(3, .cooksd)

ggplot(model.data, aes(index, .std.resid)) +
  geom_point(aes(color = satisfied), alpha = .5) +
  theme_bw()

model.data %>%
  filter(abs(.std.resid) > 3) #there are few influential points (problem solved)
```

```
lr_prob1 <- predict(logit_noMulty_noout, newdata = test)

lr_preds_test <- rep(0, 12)
i<-1
for (thresh in seq(0.25,0.75,0.05)){
  lr_pred <- ifelse(lr_prob1 > thresh,1,0)
  cm <- table(
    as.factor(lr_pred),
    as.factor(test$satisfied)
  )[2:1, 2:1]
  lr_preds_test[i] <- F_meas(cm) # f1 score
  i<-i+1
}
names(lr_preds_test) <- seq(0.25,0.75,0.05)
lr_preds_test
lr_pred <- as.numeric(ifelse(lr_prob1 > 0.5,"1","0"))
tb <- table(Predicted = lr_pred, Actual = test$satisfied)[2:1, 2:1]
tb
(tb[1:1,1:1] + tb[2:2, 2:2])/(tb[1:1,2:2] + tb[2:2, 1:1] + tb[1:1,1:1] + tb[2:2, 2:2])
#Accuracy
F_meas(tb) # F1
recall(tb)  # Recall
precision(tb) # Precision


library(yardstick)
library(ggplot2)


truth_predicted <- data.frame(
  obs = test$satisfied,
  pred = lr_pred
)
truth_predicted$obs <- as.factor(truth_predicted$obs)
truth_predicted$pred <- as.factor(truth_predicted$pred)

cm <- conf_mat(truth_predicted, obs, pred)

autoplot(cm, type = "heatmap") +
  scale_fill_gradient(low = "pink", high = "cyan")
#9106 true negative
#10455 true positive
#2552 false negative
#1273 false positive
(9106+10455)/(9106+10455+2552+1273) # 0.8364406 correctly predicted

par(mfrow=c(1,1))
library("pROC")
test_roc=roc(as.numeric(test$satisfied)~lr_prob1 , plot = TRUE,
             print.auc = TRUE, percent=TRUE, ci=TRUE)
#very good predictive ability of the model

cbind("Coefficients_(exp)"=round(exp(coefficients(logit_noMulty_noout)),2),
```

```
    "Coefficients (%)"=round((exp(coefficients(logit_noMulty_noout))-1)*100, 2))

#check linearity assumption

# Decision tree ----------------------------------------------------------------

library(datasets)
library(caTools)
library("party")
library(dplyr)
library(magrittr)
library("tree")
library("ISLR")
library(yardstick)
library(ggplot2)

tree.sat=tree(as.factor(satisfied)~.,train)
tree.pred=predict(tree.sat,test,type="class")

truth_predicted <- data.frame(
  obs = as.factor(test$satisfied),
  pred = tree.pred
)
truth_predicted$obs <- as.factor(truth_predicted$obs)
truth_predicted$pred <- as.factor(truth_predicted$pred)

cm <- conf_mat(truth_predicted, obs, pred)

autoplot(cm, type = "heatmap") +
  scale_fill_gradient(low = "pink", high = "cyan")
#1457 false positive
#1451 false negative
#11556 true positive
#8922 true negative
(11556+8922)/(11556+8922+1457+1451) #0.8756521 (Accurancy)


#Pruning
cv.sat=cv.tree(tree.sat,FUN=prune.misclass)
names(cv.sat)
par(mfrow=c(1,2))
plot(cv.sat$size,cv.sat$dev,type="b",ylab="CV-RMSE", xlab="Tree size")
plot(cv.sat$k,cv.sat$dev,type="b")
prune.sat=prune.misclass(tree.sat,best=10)

par(mfrow=c(1,1))
plot(prune.sat)
text(prune.sat,pretty=0)
tree.pred=predict(prune.sat,test,type="class")
table(tree.pred,test$satisfied)

truth_predicted <- data.frame(
  obs = as.factor(test$satisfied),
  pred = tree.pred
)
truth_predicted$obs <- as.factor(truth_predicted$obs)
```

```
truth_predicted$pred <- as.factor(truth_predicted$pred)

cm <- conf_mat(truth_predicted, obs, pred)

autoplot(cm, type = "heatmap") +
  scale_fill_gradient(low = "pink", high = "cyan")
#same results as before

#Partitioning

library("rpart")
library("rpart.plot")
library("partykit")
library("party")

tree3=part(as.factor(satisfied)~.,sat)
tree.pred.part=predict(tree3,test,type="class")
table(tree.pred.part,test$satisfied)

truth_predicted=data.frame(
  obs = as.factor(test$satisfied),
  pred = tree.pred.part
)
truth_predicted$obs=as.factor(truth_predicted$obs)
truth_predicted$pred=as.factor(truth_predicted$pred)

cm=conf_mat(truth_predicted, obs, pred)

autoplot(cm, type = "heatmap") +
  scale_fill_gradient(low = "pink", high = "cyan")
#worse results
#1537 false positive
#1261 false negative
#11746 true positive
#8942 true negative
(8942+11746)/(8942+11746+1537+1261) #0.8808652

printcp(tree3)
rpart.plot(tree3)

#Random Forest----------------------------------------------------------------
library(randomForest)
#library(caret)
#hyperparameter tuning
mtry=tuneRF(sat[-1],as.factor(sat$satisfied), ntreeTry=500,
                stepFactor=1.5,improve=0.01, trace=TRUE, plot=TRUE)
best.m=mtry[mtry[, 2] == min(mtry[, 2]), 1]
print(mtry)
print(best.m) #mtry=9

set.seed(123)
rf=randomForest(as.factor(satisfied)~.,data=train, mtry=best.m, importance=TRUE,
                ntree=500)
print(rf)
#Evaluate variable importance
importance(rf)
```

```
#x11()
varImpPlot(rf)

yhat = predict(rf,newdata=test)

truth_predicted = data.frame(
  obs = as.factor(test$satisfied),
  pred = yhat
)
truth_predicted$obs=as.factor(truth_predicted$obs)
truth_predicted$pred=as.factor(truth_predicted$pred)

cm=conf_mat(truth_predicted, obs, pred)

autoplot(cm, type = "heatmap") +
  scale_fill_gradient(low = "pink", high = "cyan")

#10037 true negative
#12437 true positive
#570 false negative
#342 false positive
(12437+10037)/(12437+10037+570+342) # 0.9610023 (accuracy)
```

11 Images appendix

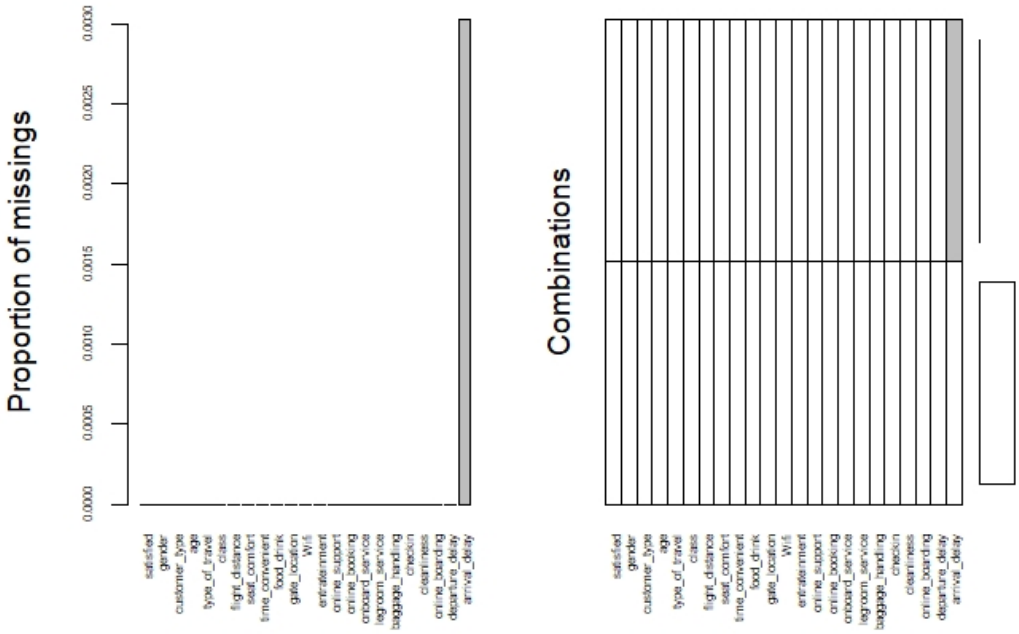

Figure 16: NAs distribution

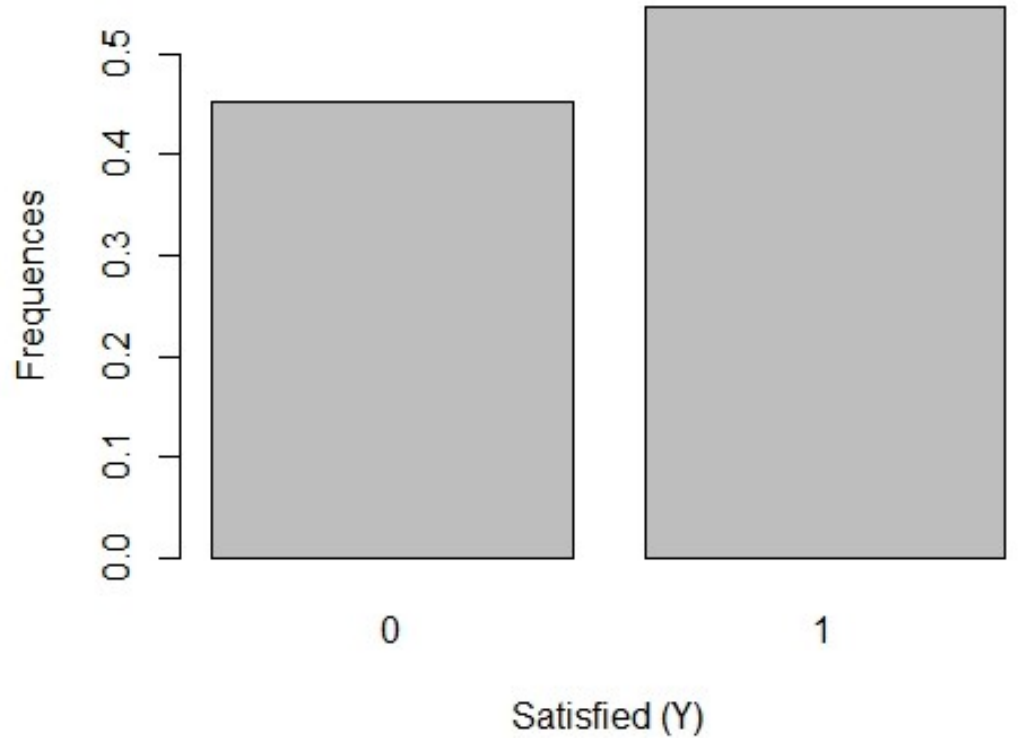

Figure 17: Response variable (Satisfied) distribution

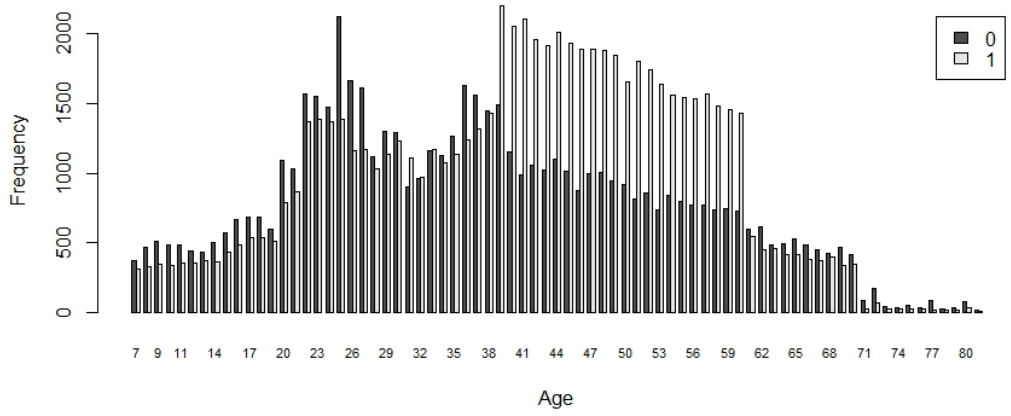

Figure 18: Satisfaction by age

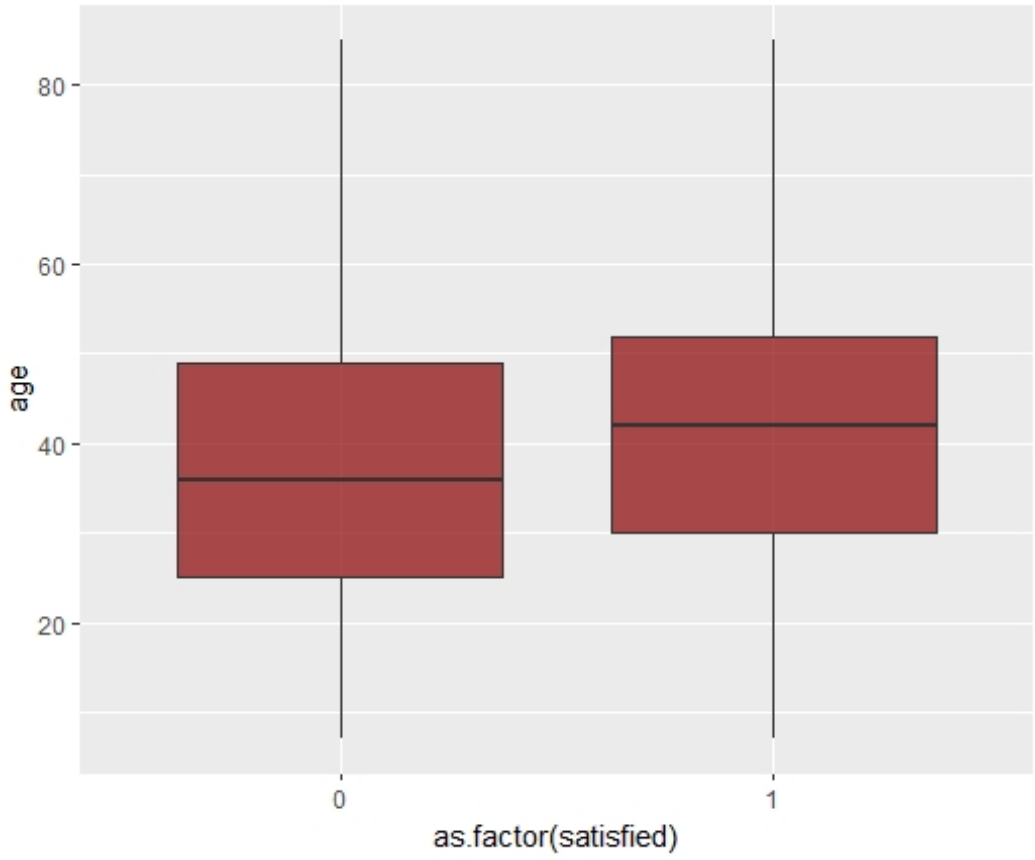

Figure 19: Satisfaction by age (boxplot)

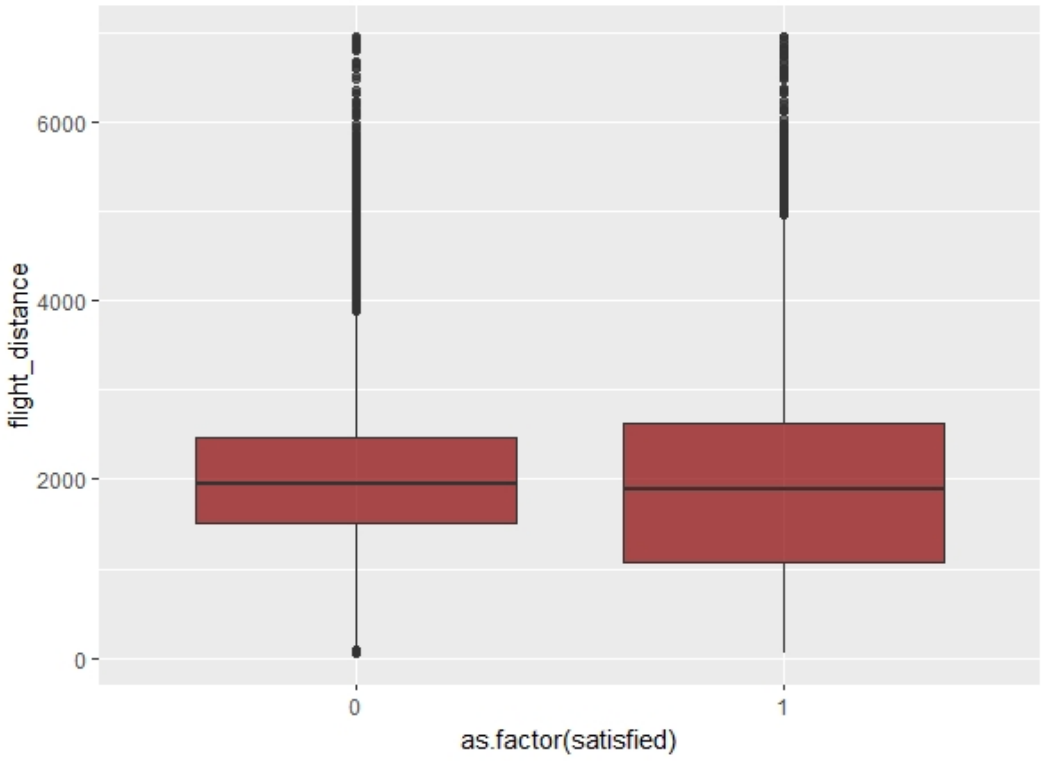

Figure 20: Satisfaction by flight distance (boxplot)

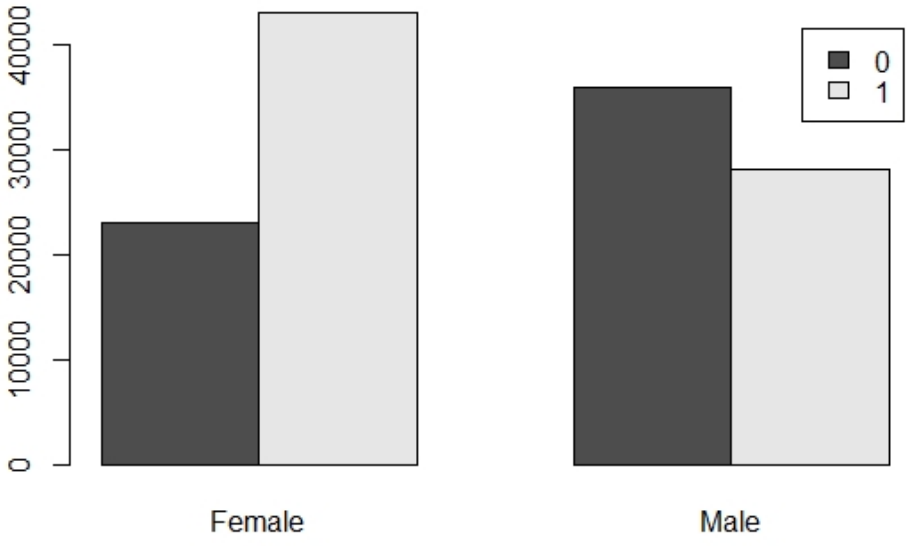

Figure 21: Satisfaction by gender

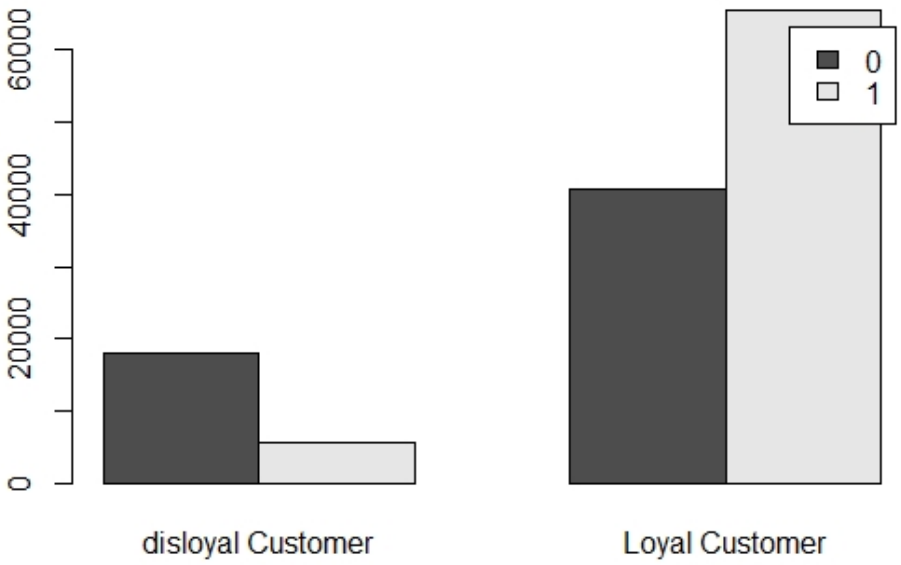

Figure 22: Satisfaction by Customer Type

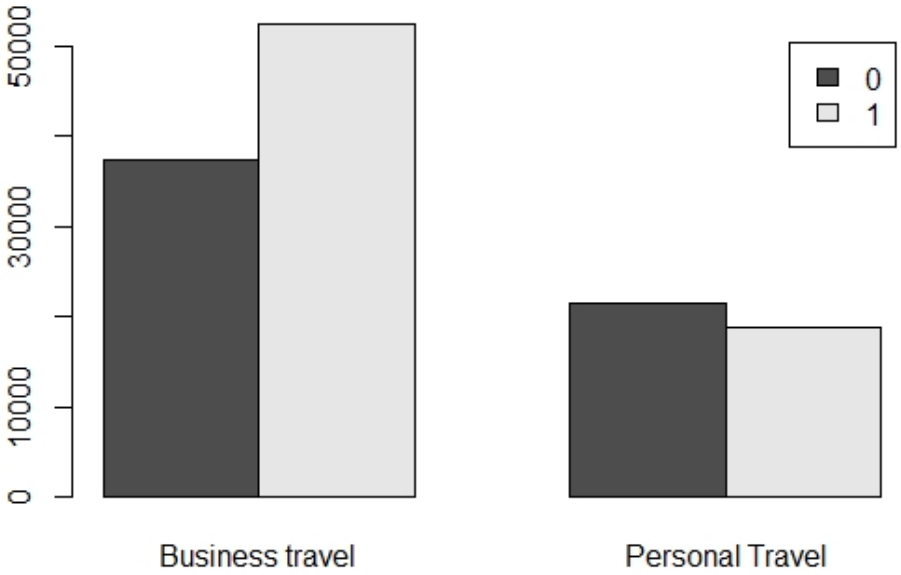

Figure 23: Satisfaction by Type of Travel

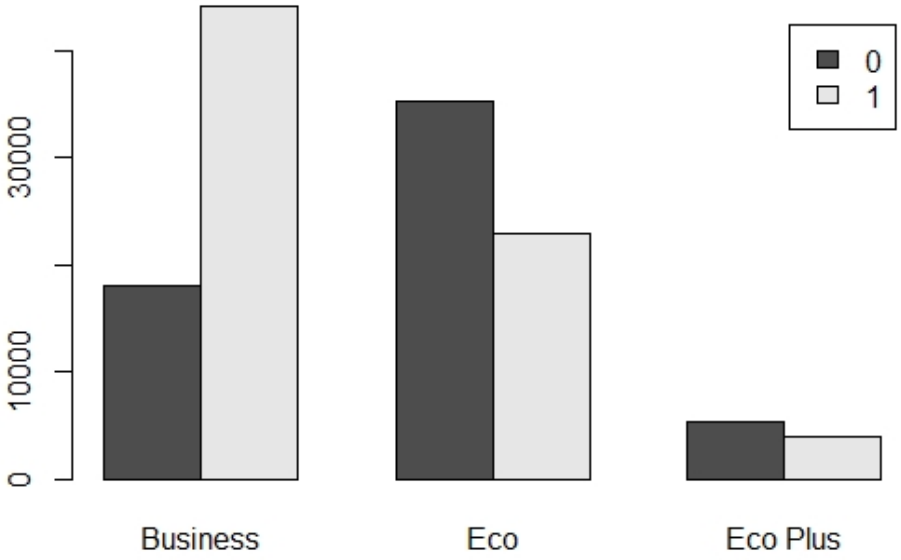

Figure 24: Satisfaction by Class

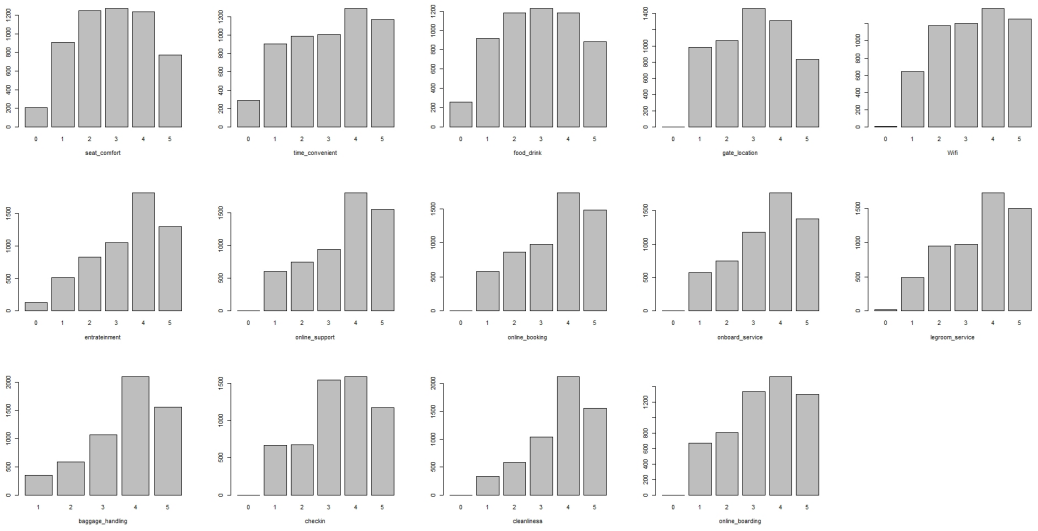

Figure 25: Services histograms