Data Science - Capstone / Choose Your Own!

Emmanuel Dupuis

August 3rd , 2022

Contents

1 Overview **1**
1.1 Description of the Prediction Challenge . . . 2
1.2 Data Field Descriptions . . . 2
1.3 Data files . . . 2

2 Data Preprocessing **3**
2.1 Changing the columns . . . 3
2.2 Verification of the data type . . . 4
2.3 Management of empty cells . . . 5
2.4 An additional column . . . 6
2.5 train set and test set . . . 7

3 A first graphical study of the variables **7**
3.1 Distribution of the population according to variables . . . 7
3.2 Relationship between the variable Transported to explane and the explanatory variables . . . 11

4 Evaluation metrics used **14**

5 Implementation of several learning machine **14**
5.1 Two basic models . . . 14
5.1.1 A basic model based only on the variable *CryoSleep* . . . 14
5.1.2 A basic model based only on the variable *total* . . . 15
5.2 A logistic regression model . . . 15
5.3 The kNN model . . . 16
5.3.1 The kNN model with three explicative variables . . . 16
5.3.2 The kNN model with more explicative variables . . . 16
5.4 The classification tree model . . . 16
5.5 The random forest model . . . 17

6 Results **18**

7 Conclusion **18**

1 Overview

For this project, we are asked to choose a publicly available dataset to solve the problem of our choice by applying some machine learning techniques that we have learned in the the 9 courses program "Data Science". Our course Instructor, Rafael Irizarry, suggested that we work with databases from **https://archive.ics.uci.edu** or **https://www.kaggle.com**

Kaggle is a web platform for data science competitions. On this platform, companies propose data science problems and offer a prize to the data scientists with the best performance.

Currently, there is a competition for get familiar with the basics of machine learning and Kaggle competitions. It is a question of predict which passengers are transported to an alternate dimension when a Spaceship collided with a spacetime anomaly hidden within a dust cloud. This is a fictional example. But the data has been constructed in such a way that it should be possible to build a learning machine to predict the outcome. About 2500 teams are participating in the competition. The best team has so far managed to predict the outcome with only 14% error.

1.1 Description of the Prediction Challenge

In the year 2912, we've received a transmission from four lightyears away and things aren't looking good... Almost 13,000 passengers, on board on a interstellar spaceship, have begun a voyage to three newly habitable exoplanets in orbit around nearby stars. While rounding Alpha Centauri en route to its first destination the Spaceship collided with a spacetime anomaly hidden within a dust cloud. Though the ship stayed intact, almost half of the passengers were transported to an alternate dimension! To help rescue crews and retrieve the lost passengers, we are challenged to predict which passengers were transported by the anomaly using records recovered from the spaceship's damaged computer system.

1.2 Data Field Descriptions

To help us make these predictions, you're given a set of personal records recovered from the ship's damaged computer system.

- PassengerId : A unique Id for each passenger. Each Id takes the form gggg_pp where gggg indicates a group the passenger is travelling with and pp is their number within the group. People in a group are often family members, but not always.
- HomePlanet : The planet the passenger departed from, typically their planet of permanent residence.
- CryoSleep : Indicates whether the passenger elected to be put into suspended animation for the duration of the voyage. Passengers in cryosleep are confined to their cabins.
- Cabin : The cabin number where the passenger is staying. Takes the form deck/num/side, where side can be either P for Port or S for Starboard.
- Destination : The planet the passenger will be debarking to.
- Age : The age of the passenger.
- VIP : Whether the passenger has paid for special VIP service during the voyage.
- RoomService, FoodCourt, ShoppingMall, Spa, VRDeck : Amount the passenger has billed at each of the Spaceship many luxury amenities.
- Name : The first and last names of the passenger.
- Transported : Whether the passenger was transported to another dimension. This is the target, the column you are trying to predict.

Our objective is then to predict the *Transported* data from all the other data.

1.3 Data files

On the website **https://www.kaggle.com**, data are given in the form of 3 files **(here)**: a file to train a learning machine (*train.csv*), a file to test it (*test.csv*) and a file indicating the format in which the results will be returned (*sample_submission.csv*). The difficulty is that in this second file the column *Transported* is missing, not allowing us to test our model ourselves, the evaluation is done directly by **https://www.kaggle.com**.

This is why we will only work from the first file, witch we will decompose ourselves into a file to train our machine and a file to test it. We have registered it on our GitHub page **(here)**, to avoid any problem of registration to **https://www.kaggle.com**. and we have renamed it : **spaceship.csv**

2 Data Preprocessing

First, let's download the data file, we are going to work from, and save it as **spaceship.csv** file. As we have said, to avoid any problems connecting to the site **www.kaggle.com**, a version of this file is saved on our GitHub page.

```
spaceship <-  read.table(
"https://raw.githubusercontent.com/Emmanuel-Dupuis-GitHub/SpaceShip/main/spaceship.csv"
, sep = ",", dec = ".", header = TRUE)
```

This is a 14-column file containing the data described in the paragraph *Data Field Descriptions*.

```
head(spaceship)
```

```
##   PassengerId HomePlanet CryoSleep Cabin   Destination Age   VIP RoomService
## 1     0001_01     Europa     False B/0/P   TRAPPIST-1e  39 False           0
## 2     0002_01      Earth     False F/0/S   TRAPPIST-1e  24 False         109
## 3     0003_01     Europa     False A/0/S   TRAPPIST-1e  58  True          43
## 4     0003_02     Europa     False A/0/S   TRAPPIST-1e  33 False           0
## 5     0004_01      Earth     False F/1/S   TRAPPIST-1e  16 False         303
## 6     0005_01      Earth     False F/0/P PSO J318.5-22  44 False           0
##   FoodCourt ShoppingMall  Spa VRDeck              Name Transported
## 1         0            0    0      0   Maham Ofracculy       False
## 2         9           25  549     44      Juanna Vines        True
## 3      3576            0 6715     49     Altark Susent       False
## 4      1283          371 3329    193      Solam Susent       False
## 5        70          151  565      2 Willy Santantines        True
## 6       483            0  291      0 Sandie Hinetthews        True
```

2.1 Changing the columns

PassengerId is unique Id for each passenger. Each Id takes the form gggg_pp where gggg indicates a group the passenger is travelling with and pp is their number within the group. People in a group are often family members, but not always. We will create a column containing the group of each passenger

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
library(tidyverse)
temp = spaceship$PassengerId %>% str_split(pattern="_", simplify=TRUE)
spaceship$Group = temp[, 1] %>% as.factor()
```

Cabin is number where the passenger is staying. Takes the form deck/num/side, where side can be either P for Port or S for Starboard. We will create 3 columns containing the deck, the number, and the side of each cabin. And we are going to delete the *cabin* column that will no longer be useful to us.

```
temp = spaceship$Cabin %>% str_split(pattern="/", simplify=TRUE)
spaceship$CabinDeck = temp[, 1] %>% as.factor()
spaceship$CabinNum = temp[, 2] %>% as.integer()
spaceship$CabinSide = temp[, 3] %>% as.factor()
spaceship$Cabin <- NULL
```

Name is the first and last names of the passenger. We will create 2 columns containing first nane and the last name of each passenger. And we are going to delete the *Name* column that will no longer be useful to us.

```
temp = spaceship$Name %>% str_split(pattern=" ", simplify=TRUE)
spaceship$FirsName = temp[, 1] %>% as.character()
spaceship$LastName = temp[, 2] %>% as.character()
spaceship$Name <- NULL
```

Now our data file looks like this:

```
head(spaceship)
```

```
##   PassengerId HomePlanet CryoSleep   Destination Age   VIP RoomService
## 1     0001_01     Europa     False   TRAPPIST-1e  39 False           0
## 2     0002_01      Earth     False   TRAPPIST-1e  24 False         109
## 3     0003_01     Europa     False   TRAPPIST-1e  58  True          43
## 4     0003_02     Europa     False   TRAPPIST-1e  33 False           0
## 5     0004_01      Earth     False   TRAPPIST-1e  16 False         303
## 6     0005_01      Earth     False PSO J318.5-22  44 False           0
##   FoodCourt ShoppingMall  Spa VRDeck Transported Group CabinDeck CabinNum
## 1         0            0    0      0       False  0001         B        0
## 2         9           25  549     44        True  0002         F        0
## 3      3576            0 6715     49       False  0003         A        0
## 4      1283          371 3329    193       False  0003         A        0
## 5        70          151  565      2        True  0004         F        1
## 6       483            0  291      0        True  0005         F        0
##   CabinSide FirsName    LastName
## 1         P    Maham   Ofracculy
## 2         S   Juanna       Vines
## 3         S   Altark      Susent
## 4         S    Solam      Susent
## 5         S    Willy Santantines
## 6         P   Sandie  Hinetthews
```

2.2 Verification of the data type

It is then natural to consider the next variables of type factor :

- *PassengerId* : A unique Id for each passenger
- *HomePlanet* : The planet the passenger departed from ("Europa", "Earth" or "Mars")
- *Destination* : The planet the passenger will be debarking to ("55 Cancri e","PSO J318.5-22" or "TRAPPIST-1e")
- *CabinDeck* : "A", "B", "C", "D", "E", "F", "G" and "T"
- *CabinSide* : "P" or "S"
- *Group* : indicates a group the passenger is travelling with

To consider the next variables of type logical :

- *CryoSleep* : Indicates whether the passenger elected to be put into suspended animation for the duration of the voyage. (TRUE or FALSE)
- *VIP* : Whether the passenger has paid for special VIP service during the voyage (TRUE or FALSE)
- *Transported* : Whether the passenger was transported to another dimension. This is the target, the column you are trying to predict (TRUE or FALSE)

To consider the next variables of type integer :

- *Age*
- *CabinNum*

To consider the next variables of type character :

- *FirsName*
- *LastName*

To consider the next variables of type numeric : Amount the passenger has billed at each of the Spaceship many luxury amenities.

- *RoomService*
- *FoodCourt*
- *ShoppingMall*
- *Spa*
- *VRDeck*

To make sure that these different formats are taken into account, we use the following code

```
spaceship$PassengerId = as.factor(spaceship$PassengerId)
spaceship$HomePlanet = as.factor(spaceship$HomePlanet)
spaceship$Destination = as.factor(spaceship$Destination)
spaceship$CryoSleep = as.logical(spaceship$CryoSleep)
spaceship$VIP = as.logical(spaceship$VIP)
spaceship$Age = as.integer(spaceship$Age)
spaceship$RoomService = as.numeric(spaceship$RoomService)
spaceship$FoodCourt = as.numeric(spaceship$FoodCourt)
spaceship$ShoppingMall = as.numeric(spaceship$ShoppingMall)
spaceship$Spa = as.numeric(spaceship$Spa)
spaceship$VRDeck = as.numeric(spaceship$VRDeck)
spaceship$FirsName = as.character(spaceship$FirsName)
spaceship$LastName = as.character(spaceship$LastName)
spaceship$CabinDeck = as.factor(spaceship$CabinDeck)
spaceship$CabinSide = as.factor(spaceship$CabinSide)
spaceship$CabinNum = as.integer(spaceship$CabinNum)
spaceship$Group = as.factor(spaceship$Group)
spaceship$Transported=as.logical(spaceship$Transported)
```

2.3 Management of empty cells

If some cells are empty, we place the value NA in them.

```
spaceship[spaceship==""] <- NA
```

We can now examine the distribution of values in each of the columns.

```
summary(spaceship)
```

```
##   PassengerId    HomePlanet   CryoSleep              Destination
##  0001_01:   1          :   0   Mode :logical                 :   0
##  0002_01:   1   Earth :4602   FALSE:5439      55 Cancri e  :1800
##  0003_01:   1   Europa:2131   TRUE :3037      PSO J318.5-22: 796
##  0003_02:   1   Mars  :1759   NA's :217       TRAPPIST-1e  :5915
##  0004_01:   1   NA's  : 201                   NA's         : 182
##  0005_01:   1
##  (Other):8687
##       Age            VIP           RoomService        FoodCourt
##  Min.   : 0.00   Mode :logical   Min.   :    0.0   Min.   :    0.0
##  1st Qu.:19.00   FALSE:8291      1st Qu.:    0.0   1st Qu.:    0.0
##  Median :27.00   TRUE :199       Median :    0.0   Median :    0.0
##  Mean   :28.83   NA's :203       Mean   :  224.7   Mean   :  458.1
##  3rd Qu.:38.00                   3rd Qu.:   47.0   3rd Qu.:   76.0
##  Max.   :79.00                   Max.   :14327.0   Max.   :29813.0
##  NA's   :179                     NA's   :181       NA's   :183
##   ShoppingMall          Spa               VRDeck        Transported
##  Min.   :    0.0   Min.   :    0.0   Min.   :    0.0   Mode :logical
##  1st Qu.:    0.0   1st Qu.:    0.0   1st Qu.:    0.0   FALSE:4315
##  Median :    0.0   Median :    0.0   Median :    0.0   TRUE :4378
##  Mean   :  173.7   Mean   :  311.1   Mean   :  304.9
##  3rd Qu.:   27.0   3rd Qu.:   59.0   3rd Qu.:   46.0
##  Max.   :23492.0   Max.   :22408.0   Max.   :24133.0
##  NA's   :208       NA's   :183       NA's   :188
##      Group        CabinDeck       CabinNum      CabinSide     FirsName
##  0984   :   8   F      :2794   Min.   :   0.0       :   0   Length:8693
##  4005   :   8   G      :2559   1st Qu.: 167.2   P   :4206   Class :character
##  4256   :   8   E      : 876   Median : 427.0   S   :4288   Mode  :character
##  4498   :   8   B      : 779   Mean   : 600.4   NA's: 199
##  5133   :   8   C      : 747   3rd Qu.: 999.0
##  5756   :   8   (Other): 739   Max.   :1894.0
##  (Other):8645   NA's   : 199   NA's   :199
##    LastName
##  Length:8693
##  Class :character
##  Mode  :character
##
##
##
##
```

We see that there are still NA values in each of the columns. But no more than 208 for the 8693 rows. That is less than 2.5%. We can then decide to delete all the rows containing at least one NA value. But if these are randomly distributed, we risk deleting many rows.

In this case, because there are few NA values per column, it is recommended to replace the NA values of the factor or character columns by the mode (the most common values) and to replace the NA values of the numerical columns by the median.

```
spaceship[is.na(spaceship$HomePlanet),"HomePlanet"]<- "Earth"
spaceship[is.na(spaceship$CryoSleep),"CryoSleep"]<- FALSE
spaceship[is.na(spaceship$Destination),"Destination"]<- "TRAPPIST-1e"
spaceship[is.na(spaceship$Age),"Age"]<- 27
spaceship[is.na(spaceship$VIP),"VIP"]<- FALSE
spaceship[is.na(spaceship$RoomService),"RoomService"]<- 0
spaceship[is.na(spaceship$FoodCourt),"FoodCourt"]<- 0
spaceship[is.na(spaceship$ShoppingMall),"ShoppingMall"]<- 0
spaceship[is.na(spaceship$Spa),"Spa"]<- 0
spaceship[is.na(spaceship$VRDeck),"VRDeck"]<- 0
spaceship[is.na(spaceship$CabinDeck),"CabinDeck"]<- "F"
spaceship[is.na(spaceship$CabinNum),"CabinNum"]<- 427
spaceship[is.na(spaceship$CabinSide),"CabinSide"]<- "S"
```

2.4 An additional column

Now that none of the cells in the columns containing the travellers' expenses are empty, it seems interesting to add a last column containing the travellers' total expenses.

```
spaceship$total=
spaceship$RoomService+spaceship$FoodCourt+spaceship$ShoppingMall+spaceship$Spa+spaceship$VRDeck
```

Now our data file looks like this:

```
head(spaceship)
```

```
##   PassengerId HomePlanet CryoSleep   Destination Age   VIP RoomService
## 1     0001_01     Europa     FALSE   TRAPPIST-1e  39 FALSE           0
## 2     0002_01      Earth     FALSE   TRAPPIST-1e  24 FALSE         109
## 3     0003_01     Europa     FALSE   TRAPPIST-1e  58  TRUE          43
## 4     0003_02     Europa     FALSE   TRAPPIST-1e  33 FALSE           0
## 5     0004_01      Earth     FALSE   TRAPPIST-1e  16 FALSE         303
## 6     0005_01      Earth     FALSE PSO J318.5-22  44 FALSE           0
##   FoodCourt ShoppingMall  Spa VRDeck Transported Group CabinDeck CabinNum
## 1         0            0    0      0       FALSE  0001         B        0
## 2         9           25  549     44        TRUE  0002         F        0
## 3      3576            0 6715     49       FALSE  0003         A        0
## 4      1283          371 3329    193       FALSE  0003         A        0
## 5        70          151  565      2        TRUE  0004         F        1
## 6       483            0  291      0        TRUE  0005         F        0
##   CabinSide FirsName    LastName total
## 1         P    Maham   Ofracculy     0
## 2         S   Juanna       Vines   736
## 3         S   Altark      Susent 10383
## 4         S    Solam      Susent  5176
## 5         S    Willy Santantines  1091
## 6         P   Sandie  Hinetthews   774
```

2.5 train set and test set

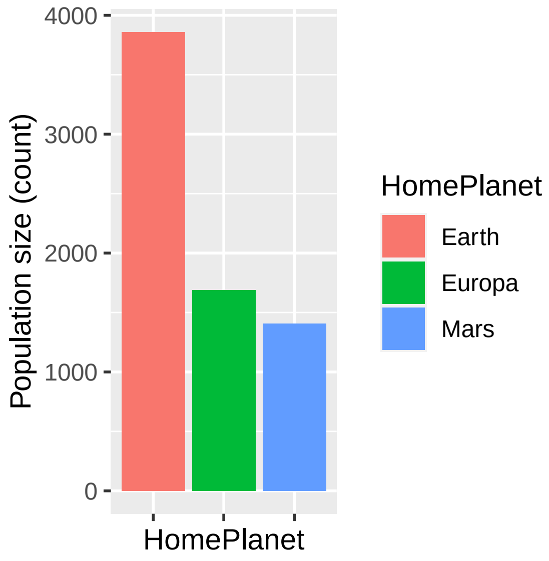
Now, to mimic a evaluation process, we randomly split our data into two — a training set and a test set — and act as if we don't know the outcome of the test set. We develop algorithms using only the training set. the test set is used only for evaluation. We randomly extract 80% to form the train set, **train_set**, the other 20% forming the test set, **test_set**.

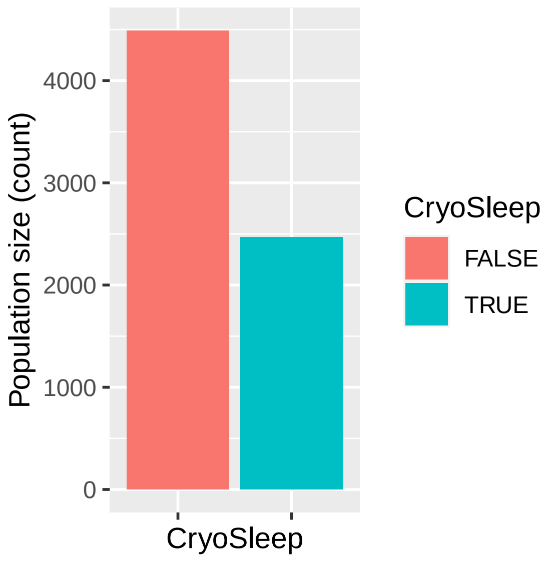
```
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
library(caret)
set.seed(2, sample.kind = "Rounding") #if using R 3.5 or earlier, remove the sample.kind argument
test_index <- createDataPartition(spaceship$Transported, times = 1, p = 0.2, list = FALSE)
test_set <- spaceship[test_index, ]
train_set <- spaceship[-test_index, ]
```

3 A first graphical study of the variables

3.1 Distribution of the population according to variables

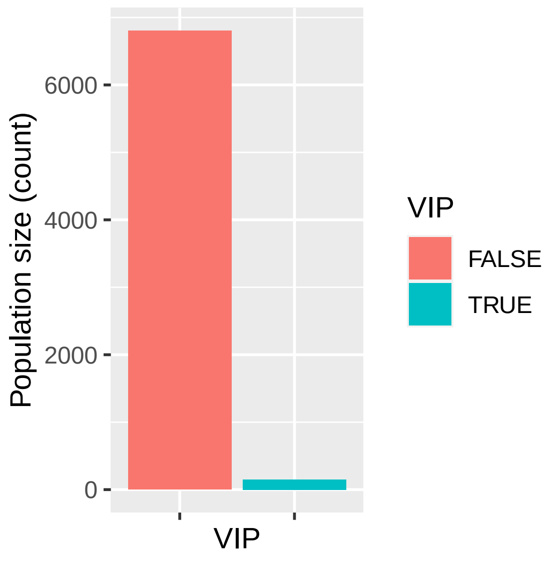
Let's look at the size of the population according to the values of the main variables

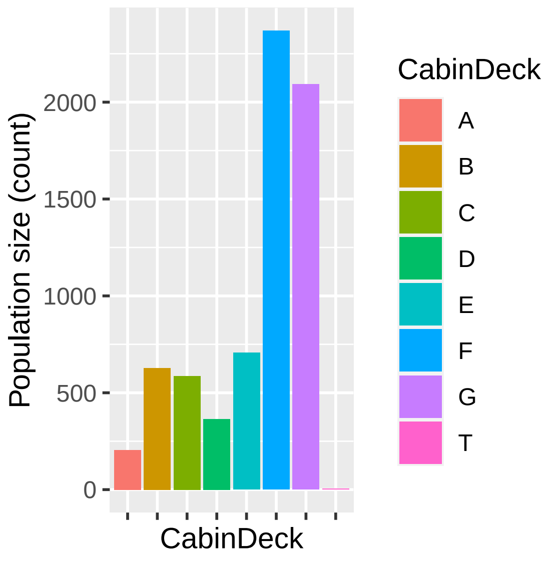




For the HomePlanet variable, we see that almost twice as many passengers are from Earth than Europa/Mars.

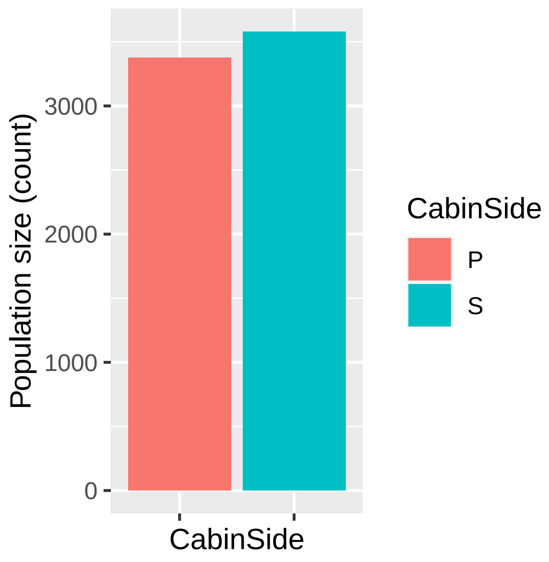
For the CryoSleep variable, we see that the passengers who choose CryoSleep (passengers are confined to their pods) were half of ones who didn't.

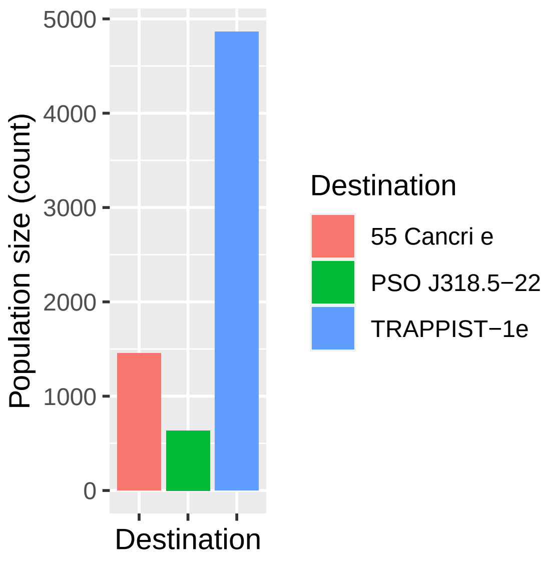




For the HomePlanet variable, we see that almost all passengers are not VIPs.

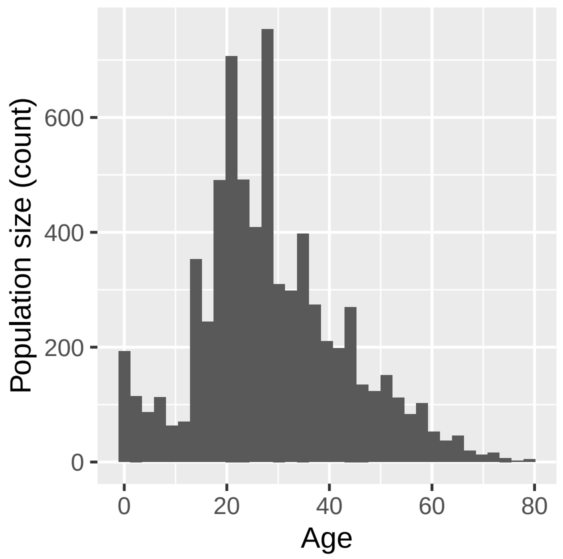
For the CabinDeck variable, we see that Cabin deck F, G contains the highest passenges

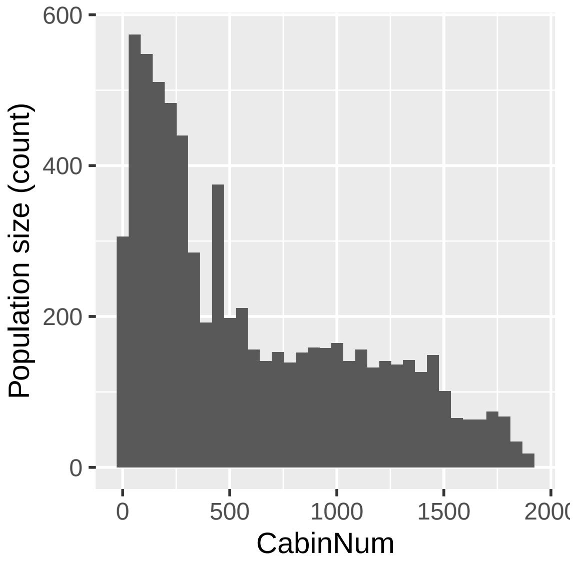




As many passengers are on on port and starboard side

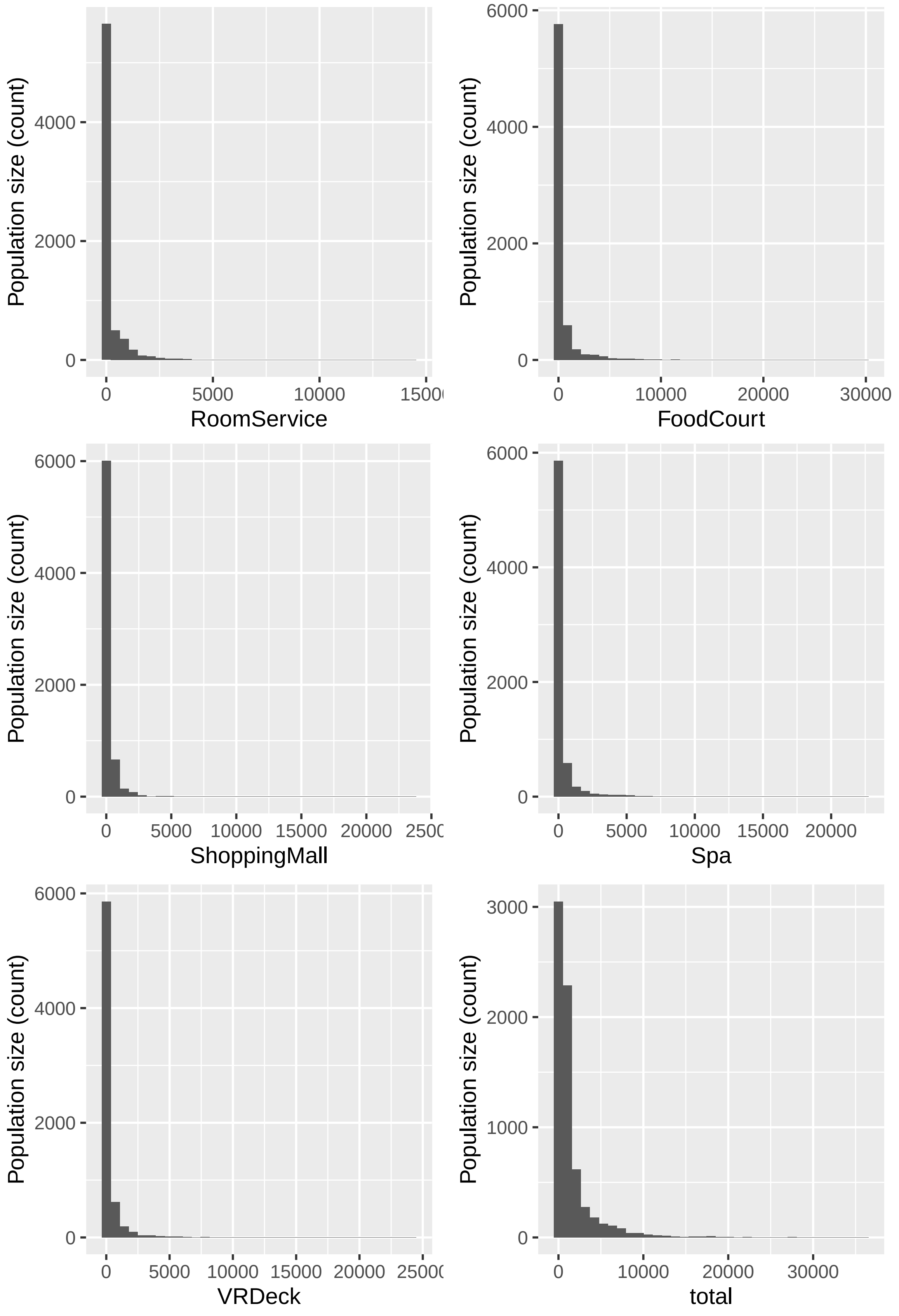
And the majority of passengers go to the planet TRAPPIST 1e





his shows the age distribution of the passengers.

We notice that passengers tend to be accommodated in cabins with numbers below 500.

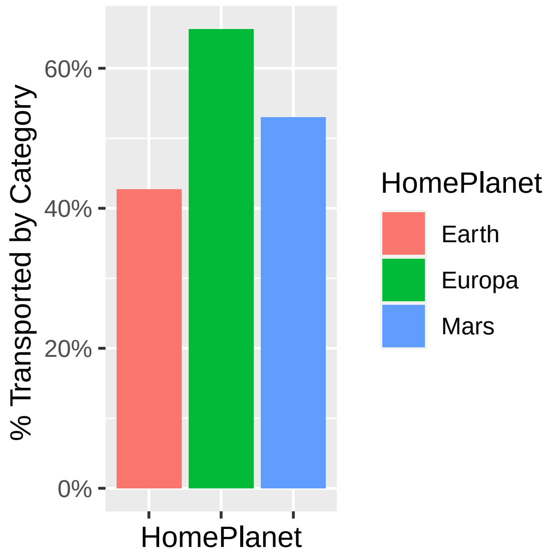


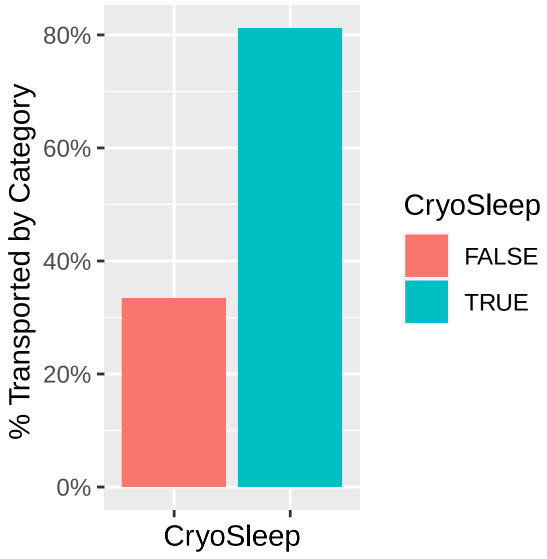
The latter graphs show that the majority of passengers do not make any additional expenditure.

3.2 Relationship between the variable Transported to explane and the explanatory variables

But let's not forget that the aim of this project is to try to predict the variable **Transported** as well as possible according to the other variables. Once again we will try to detect links between these variables and the variable **Transported** by starting with graphs. This, before implementing more precise learning machines.

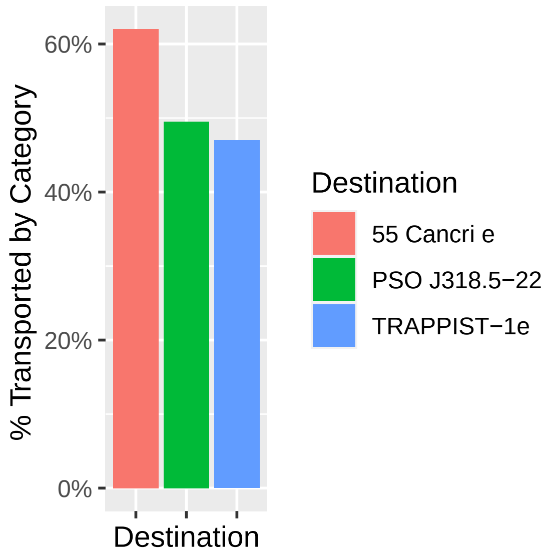
For each of the different values taken by a variable, we will look at the percentage of people transported.

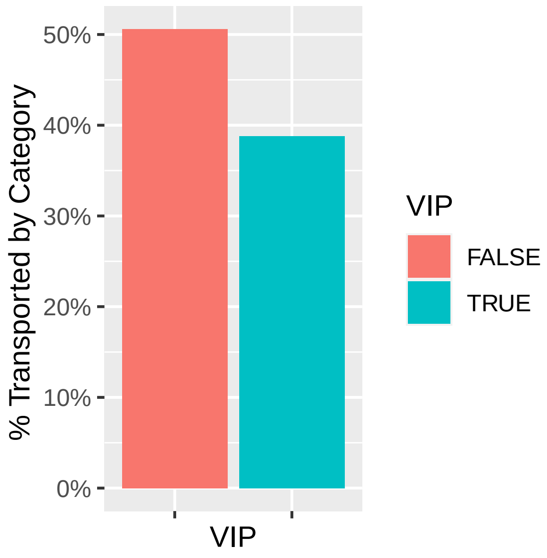




HomePlanet : Europa passengers have higher tendency to be transported (more than 60% of them)

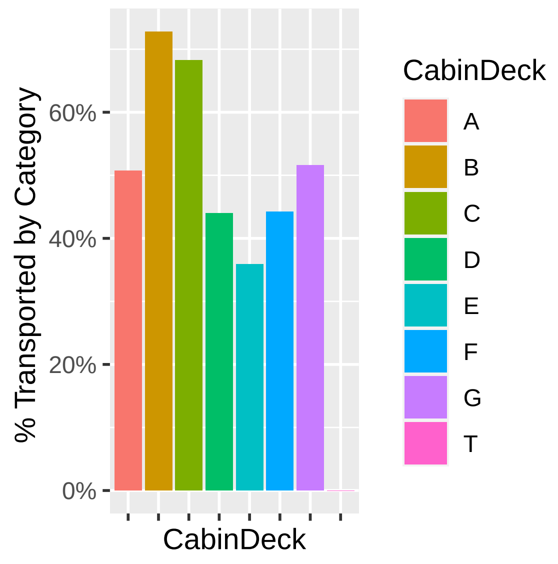
CryoSleep : Here it is obvious that the variable *CryoSleep* is strongly correlated with the variable *Transported*: one can imagine that travellers in cryogenic sleep do not have time to run away to avoid being transported.

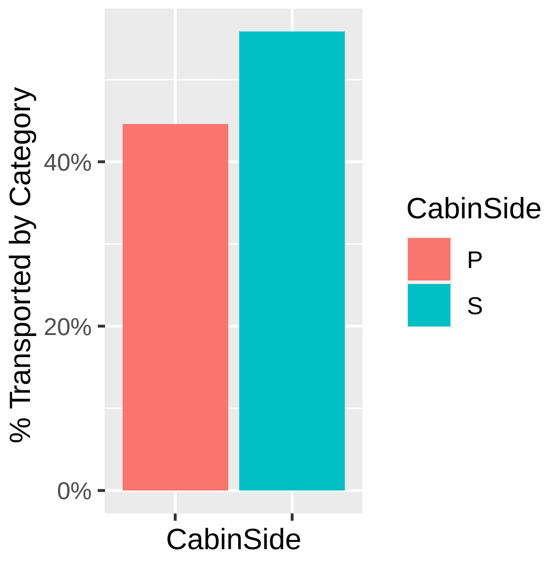




Destination : Passengers to 55 Cancri e have higher tendency to be transported (more than 60% of them)

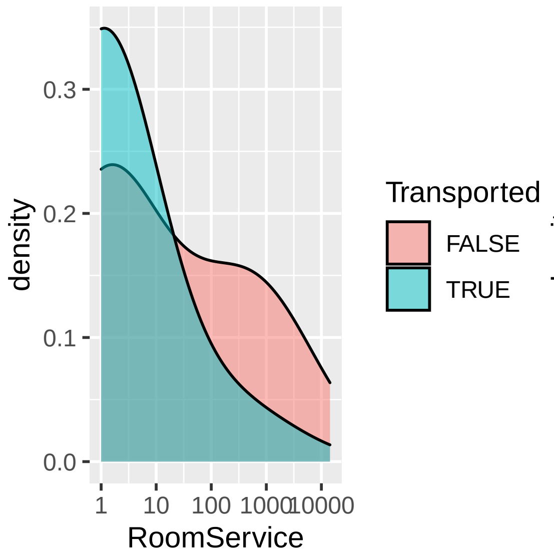
VIP : VIP Passengers are less transported (less than 40% of them)

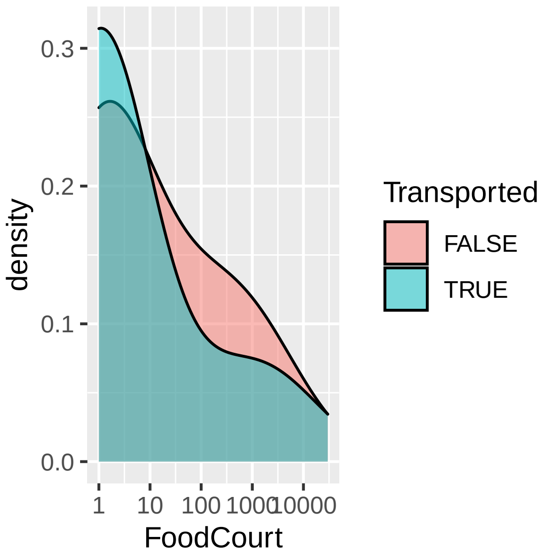


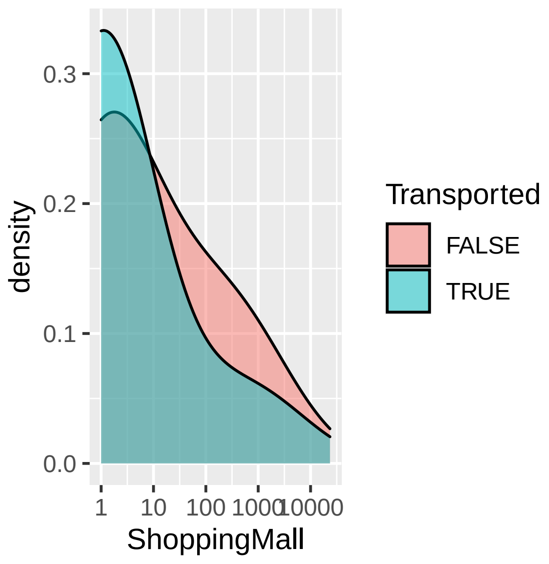


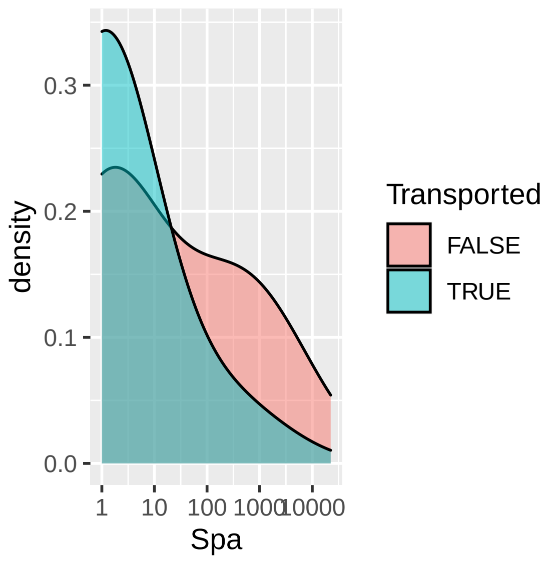
CabinDeck : Passengers Cabine Deck B and C have higher tendency to be transported

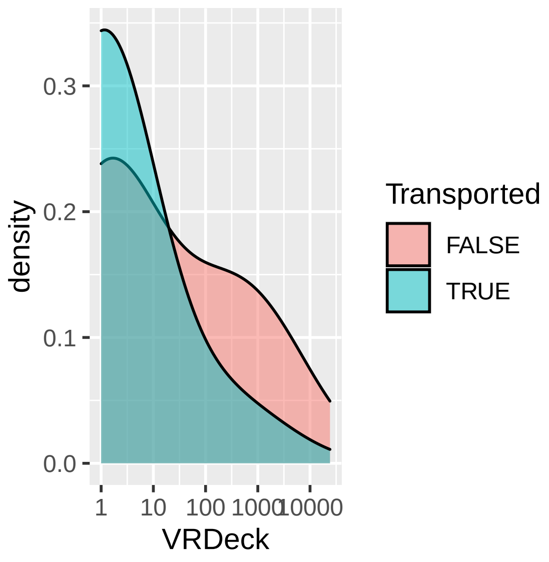
CabinSide : There is no clear trend here.

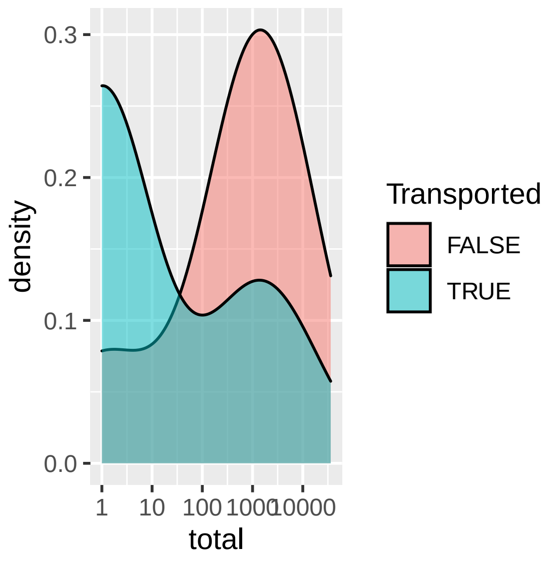




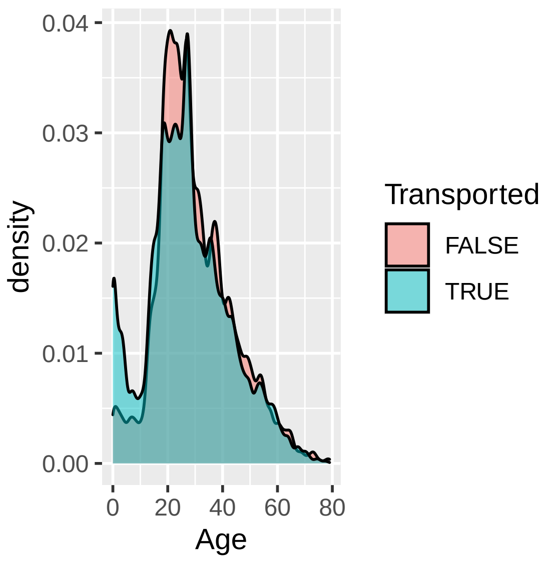








Generally speaking, the more a passenger spends, the less likely they are to be transported. This becomes obvious with the total variable, which is the sum of the expenses.



The graph shows that passenger under 10 years of age are more likely to be transported

In the light of these initial graphical analyses, we can begin to construct explanatory models for the variable **Transported**

4 Evaluation metrics used

Before we start describing approaches to optimize the way we build algorithms, we first define what we mean when we say one approach is better than another.

The simplest evaluation metric for categorical or logical outcomes like **Transported** is overall accuracy: the proportion of cases that were correctly predicted in the test set. It is, in fact, this metric that is used to separate the candidates participating in the different competitions on the Kaggle web platform.

Let's write a function that computes the accuracy for vectors of ratings and their corresponding predictors:

```
accuracy <- function(true_vector, predicted_vector)
        { mean(true_vector == predicted_vector)
        }
```

5 Implementation of several learning machine

5.1 Two basic models

Recall that the purpose of this project is to try to predict as well as possible the variable **Transported** as a function of the other.

Let's start with two rudimentary models, to see if more sophisticated models can do better…

5.1.1 A basic model based only on the variable *CryoSleep*

By examining the graphs of the preceding paragraphs, it appears that passenger who were in cryo sleep were in majority transported. Perhaps these passenger did not have time to save themselves… We can then build a first rudimentary model which says that a traveler will be transported only if he was in cryo sleep.

```
prediction <- ifelse(test_set$CryoSleep == TRUE, TRUE, FALSE)
```

And calculate the accuracy:

```
model_accuracy = accuracy(test_set$Transported , prediction)
model_accuracy
```

```
## [1] 0.7211041
```

5.1.2 A basic model based only on the variable *total*

By examining the graphs of the preceding paragraphs, it also appears that passengers who had a total expenditure lower than about 50 were transported while if they had a higher total expenditure they were not transported We can then build a second rudimentary model which says that a traveler will be transported only if variable **total** is less than 50

```
prediction <- ifelse(test_set$total > 50, FALSE,TRUE)
```

And calculate the accuracy:

```
model_accuracy = accuracy(test_set$Transported , prediction)
model_accuracy
```

```
## [1] 0.7441058
```

5.2 A logistic regression model

In this section, we will use a linear regression model. This model will be of course more elaborate than our two rudimentary models, but will they perform better?

The variable **Transported** can be considered as a qualitative variable (TRUE / FALSE). The R learning machine function then requires that the variable to be explained be of the type factor.

```
train_set$Transported=as.factor(train_set$Transported)
test_set$Transported=as.factor(test_set$Transported)
```

Similarly, when the variable to be explained is qualitative, it is preferable to use a GLM logistic regression, which assures us that the calculated probabilities are between 0 and 1. This is what we will do.

We have seen that the graphs in the previous paragraphs seem to indicate that there is a strong correlation between the variable **Transported** and the two variables **total**and **CryoSleep**. We also saw that passengers under 10 years of age also tended to be transported.

Let's start with a regression with these three explanatory variables : **total**, **CryoSleep** and **Age**

To train a GLM model on the training set, we will use the **caret** train function:

```
set.seed(1, sample.kind = "Rounding")
train_glm <- train(Transported ~ total + CryoSleep + Age, method = "glm", data = train_set)
prediction <- predict(train_glm, test_set)
```

And calculate the accuracy:

```
model_accuracy = accuracy(test_set$Transported , prediction)
model_accuracy
```

```
## [1] 0.7211041
```

This regression does not improve the results of our two rudimentary models.

We will not add other explanatory variables at this time. Let us first see if other models can improve our results.

5.3 The kNN model

We have seen in the course "Data Science: Machine Learning" that the deficiency of linear or logistic regressions is that they cannot capture the nature of true conditional probabilities when it is non-linear. This is the reason why, we have been presented in the course the kNN model which allows to solve this problem.

5.3.1 The kNN model with three explicative variables

Let's apply this model still trying to predict the variable **Transported** according to the same three variables as before: **total**, **CryoSleep** and **Age**.

To train a kNN model on the training set, we will use the **caret** train function:

```r
set.seed(6, sample.kind = "Rounding")
train_knn <- train(Transported ~ Age + total + CryoSleep,
                   method = "knn",
                   data = train_set)
prediction <- predict(train_knn, test_set)
```

And calculate the accuracy:

```r
model_accuracy = accuracy(test_set$Transported , prediction)
model_accuracy
```

```
## [1] 0.7308798
```

It is better than GLM regression. But this kNN model does not improve the results of our two rudimentary models.

5.3.2 The kNN model with more explicative variables

We know that the variable **total** is the sum of the different extra expenses of each passenger (**RoomService**, **FoodCourt**, **ShoppingMall**, **Spa** and **VRDeck**). Can we improve our results by now including each of these expenses instead of **total**?

```r
set.seed(6, sample.kind = "Rounding")
train_knn <- train(Transported ~ Age + CryoSleep + RoomService + FoodCourt + ShoppingMall
                   + Spa + VRDeck,
                   method = "knn",
                   data = train_set)
prediction <- predict(train_knn, test_set)
```

And calculate the accuracy:

```r
model_accuracy = accuracy(test_set$Transported , prediction)
model_accuracy
```

```
## [1] 0.7970098
```

Here, the result is better than with our two rudimentary models.

5.4 The classification tree model

Let's see if the classification tree model can improve our results? Always using the variables : **Age**, **CryoSleep**, **RoomService**, **FoodCourt**, **ShoppingMall**, **Spa** and **VRDeck**, to make a comparison with the last result.

To train it on the training set, we will still use the rpart method

```r
if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org")
library(rpart)
```

```r
repart_fit <- rpart(Transported ~ Age + CryoSleep + RoomService + FoodCourt + ShoppingMall
                    + Spa + VRDeck, data = train_set)
prediction <- predict(repart_fit, test_set, type = "class")
```

And calculate the accuracy:

```r
model_accuracy = accuracy(test_set$Transported , prediction)
model_accuracy
```

```
## [1] 0.7889592
```

The result did not improve. The kNN method is still the most efficient.

5.5 The random forest model

Let us now examine the random forest model. Always using the variables : **Age**, **CryoSleep**, **RoomService**, **FoodCourt**, **ShoppingMall**, **Spa** and **VRDeck**, to make a comparison with the last results.

To train it on the training set, we will still use the rpart method

```r
if(!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
library(randomForest)
Forest_fit <- randomForest(Transported ~ Age + CryoSleep + RoomService + FoodCourt
                           + ShoppingMall + Spa + VRDeck , data = train_set)
prediction <- predict(Forest_fit, test_set, type = "class")
```

And calculate the accuracy:

```r
model_accuracy = accuracy(test_set$Transported , prediction)
model_accuracy
```

```
## [1] 0.8090857
```

We arrive at our best result. The random forest model is the most efficient. We will be able to finish our analysis by completing the explanatory variables

```r
Forest_fit <- randomForest(Transported ~ HomePlanet + Destination + Age + VIP
                           + CryoSleep + RoomService + FoodCourt + ShoppingMall
                           + Spa + VRDeck + CabinDeck + CabinSide + CabinNum, data = train_set)
Forest_model <- predict(Forest_fit, test_set, type = "class")
prediction <- predict(Forest_fit, test_set, type = "class")
```

We added all available variables except **Group**, **CabinNum**, **FirstName** and **LastName**, which seemed less coherent. We calculate the new accuracy:

```r
model_accuracy = accuracy(test_set$Transported , prediction)
model_accuracy
```

```
## [1] 0.8142611
```

This does not improve our result much. This confirms a posteriori that the choice of the variables, **Age**, **CryoSleep**, **RoomService**, **FoodCourt**, **ShoppingMall**, **Spa** and **VRDeck**, was the right one. Our graphical analysis of the dependencies between the variables, even if it could appear to be too intuitive, was therefore relevant.

This result is not so bad in the sense that it allows us to be in the top 3% of the **Kaggle** competition presented at the beginning of this work **(competetion results can be found here)**. The best competitor obtaining an identical Accuracy around 0.87000…

6 Results

As we have already mentioned, Our course Instructor, Rafael Irizarry, suggested that we work with databases from **https://www.kaggle.com**.

We started with an fictional example where we had to predict which passengers are transported to an alternate dimension when a Spaceship collided with a with a spacetime anomaly hidden within a dust cloud.

The first difficulty was to clean the data : we had to manage the empty cells using the usual procedures (use of medians or modes of variables) ; we had to create new variables to extract information from other variables.

Then, we could have calculated a correlation matrix between the variables. But, we preferred a graphical approach allowing us to well visualize the data and understand their interactions. It then appeared to us that the variable **Transported** must to be explained by **Age**, **CryoSleep**, **RoomService**, **FoodCourt**, **ShoppingMall**, **Spa** and **VRDeck**. This was confirmed by the fact that the addition of the other variables does not significantly improve the relevance of the models.

We then limited ourselves to apply the main learning machines seen in the course. It is the random forest model which appeared then as the most powerful.

As we said, our result (accuracy equal to 0.8142611) not so bad in the sense that it allows us to be in the top 3% of the **Kaggle** competition presented at the beginning of this work…

7 Conclusion

We can certainly improve these results since the best competitor on **Kaggle** gets an accuracy around 0.87000…

One way would be to optimize the parameters of the different models we used: for example the **k** parameter for the kNN model, the **cp** parameter for the classification tree model, or the **mtry** parameter for random forest model. Our results being already significant, we did not do it to avoid increase the calculation time.