UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Paris 1 Panthéon-Sorbonne
Ecole d'Economie de la Sorbonne
Master 2: Mosef

Prédiction du coût de traitement médical

EL BRAIJ Maryem
ZARICINII Xenia
Amoussou Paolie

Prefessor : M.Laurent
MAGON

Paris 2021

La prédiction de coût de traitement de patient

Introduction

Cette étude comme son nom l'indique porte sur la prédiction de la cout de traitement d'un patient à partir d'un certain nombre de caractéristiques cliniques relatives à son suivi médical ainsi qu'à son traitement durant toute la période d'admission à l'hôpital.

Nous avons eu comme support pour répondre à notre problématique 4 différentes bases de données. Ces dernières ont été conçu sur la base d'informations relatives à des patients admis dans un centre hospitalier localisé à Singapour.

Ainsi nous avons les tables:

- bill_id relative à la date d'admission du patient pour chacune de ses factures, et qui contient les variables: bill_id (identifiant unique de la facture), patient_id (identifiant unique du patient), date_of_admission (date d'admission du patient)
- bill_amount: relative au montant de chaque facture, et qui contient en plus de la variables bill_id, la variable amount (Montant facturé sur la facture) de type Numérique
- demographic: qui concerne des informations personnelles du patient à savoir: le sexe (gender), la race (race), le statut de résident (resident_status) qui sont toutes des variables catégorielles. Nous avons également dans cette table, la date de naissance du patient (date_of_birth) qui est de type date.
- et enfin clinical_data: qui donne le détail sur l'état de santé et le suivi médical du patient. Ce jeu de données contient en plus de d'ID et de la date d'admission du patient, la date de sortie de celui-ci(date_of_discharge) de type date. Aussi nous avons les 7 histoires médicales (medical_history_1...7) possibles du patient, les 6 traitements préopératoires (preop_medication_1...6), et les 5 symptômes (symptom_1....5) qui sont des booléens prenant respectivement les valeurs 1 ou 0 selon que la condition est rempli ou pas. De plus, on retrouve également dans cette table des variables numériques que sont le poids (weight), la taille (height) et les résultats de 3 différents examens de laboratoire (lab_result_1, lab_result_2, lab_result_3) dont l'échelle varie selon le test.

Après avoir effectué les jointures de tables nécessaires et un nettoyage complet des données pour avoir un dataset exploitable, nous avons procédé à une analyse exploratoire des données afin de les découvrir et de voir les potentielles tendances. Ensuite nous faisons de l'apprentissage non supervisé d'une part et supervisé d'autre part pour pouvoir arriver à prédire le montant de la facture médicale d'un patient résident de Singapour.

EDA et Mise en forme des données.

Partie 1. Analyse des bases de données et obtention de la base globale.

Chargement des libraries nécessaires La version de R 3.6.3

```
library(dplyr)
library(mice)
library(corrplot)
library(randomForest)
library(ggplot2)
```

Chargement des données

```
clinicalDATA<-read.csv("clinical_data.csv")
billID<-read.csv("bill_id.csv")
billAmount<-read.csv("bill_amount.csv")
demographics<-read.csv("demographics.csv")
```

Premier audit de la base clinical_data.csv

```
dim(clinicalDATA) #afficher la dimension de la base
summary(clinicalDATA) #afficher les statistiques descriptives de la base de donnée
sapply(clinicalDATA, class) #afficher les types de chaque variable
colSums(is.na(clinicalDATA)) #verifier le nombre des valeurs manquantes
```

Nous analysons premièrement la base clinical_data qui comporte les informations sur le traitement des patients. Nous observons que la base contient 3400 observations et 26 variables.

Nous observons que la variable id est un identifiant unique du patient de type factor et la variable est représentée par un mélange de chiffres et de lettres. En regardant le summary nous constatons qu'il y a un patient présent 4 fois dans la base de données et d'autres présent 3 fois. Nous pouvons avoir plusieurs fois le même patient, parce qu'il a eu probablemnt plusieurs traitements.

En observant les deux variables : date de début de traitement et de fin de traitement, nous constatons qu'elles n'ont pas de valeurs manquantes et elles sont de type factor et pas de type date. Pour cette raison, il y a la nécessité de convertir ces 2 variables en dates.

Les 7 variables suivantes sont les variables binaires qui comportent les informations sur la présence ou non de l'histoire médicale spécifique. Les variables medical_history_1, medical_history_4, medical_history_6 et medical_history_7 sont de type integer (nombres entiers). Elles n'ont pas de valeurs manquantes. En observant les moyennes de ces variables nous constatons qu'elles ne sont pas trop élevées et qu'il n'y a pas d'histoire médicale qui soit présent dans plus de 30% des cas pour ces 4 variables. Les variables medical_history_2 et medical_history_5 sont de type numérique et comportent des valeurs manquantes. Plus précisément, la variable Medical_history_2 contient 233 valeurs manquantes et medical_history_5 contient 304 valeurs manquantes. La variable medical_history_3 n'a pas des valeurs manquantes et elle est de type factor dû à la présence des modalités *YES* et *No*.

Les 6 variables suivantes sont binaires et comportent les informations sur la présence ou non du traitement pré opérationnel spécifique. Toutes les 6 variables sont de type integer et ne contiennent pas des valeurs manquantes. En observant les moyennes nous constatons que dans plus de 50% des cas pour chaque variable il y avait le traitement pré opérationnel spécifié.

Les 5 variables suivantes sont les variables binaires qui comportent les informations sur la présence ou non de symptôme spécifique. Toutes les 5 variables sont de type integer et ne contiennent pas des valeurs

manquantes. En observant les moyennes nous constatons que dans plus de 50% des cas pour chaque variable il y avait ce symptôme spécifié.

Les 3 variables suivantes comportent les informations sur différents résultats laboratoires. Elles sont de type numérique et n'ont pas des valeurs manquantes. Les observations de lab_results_1 sont dans l'intervalle de 9.10 à 20.30 avec une moyenne de 14.47. Pour la variable lab_result_2 ,les valeurs sont dans l'intervalle de 19.70 à 35.10 avec une moyenne de 27.43. La troisième variable lab_result_3 se trouve dans l'intervalle de 52 à 150 avec une moyenne de 99.49.

La variable "weight" représente le poids des patients. Cette variable est de type numérique sans valeurs manquantes. Les observations se trouvent dans l'intervalle de 48 kg à 121 kg avec une moyenne de 78.75 kg.

La variable height représente la taille des patients. Elle est de type numérique sans valeurs manquantes. Les observations se trouvent dans l'intervalle de 151 cm à 186 cm avec la taille moyenne de 165.1 cm.

Premier audit de la base bill_id.csv

```
dim(billID) #afficher la dimension de la base
summary(billID) #afficher les statistiques descriptives de la base de donnée
sapply(billID, class) #afficher les types de chaque variable
colSums(is.na(billID)) #verifier le nombre des valeurs manquantes
```

La table billID contient 13600 observations et 3 colonnes avec les informations sur les id des factures de traitement, les informations sur les id des patients et leur date d'admission. Selon le dictionnaire des données, les id des patients et les dates d'admission sont les mêmes que dans la table avec les informations sur le traitement des patients. Cette table nous permettra de joindre les informations sur les factures des traitements avec les informations sur le traitement des patients.

Cette table ne contient pas de valeurs manquantes. Nous observons les mêmes id et dates d'admission des patients que dans la table clinicalData, mais la fréquence est 4 fois plus élevée dans la table billID. Nous supposons qu'il y a environ 4 factures pour chaque période de traitement par patient.

Premier audit de la base bill_amount.csv

```
dim(billAmount) #afficher la dimension de la base
summary(billAmount) #afficher les statistiques descriptives de la base de donnée
sapply(billAmount, class) #afficher les types de chaque variable
colSums(is.na(billAmount)) #verifier le nombre des valeurs manquantes
```

La table billAmount contient 13600 observations et 2 variables dont l'id des factures qui nous permettra de joindre cette table avec la table billID. Cette table ne contient pas des valeurs manquantes. En observant la variable amount, nous pouvons constater que la montant minimal des factures est 79.5 dollars de Singapour. Le montant maximal des factures est 81849.8 dollars Singapouriens. En moyenne le montant de la facture est de 5464.8 dollars Singapouriens. En observant la moyenne et le quantile 75% ,nous constatons qu'il y a peu de factures avec un montant très élevé . 25% des factures ont un montant inférieur à 951 de dollars de Singapour et encore 50% des factures avec un montant entre 951 et 7307 dollars Singapouriens. Encore 25% des factures se trouvent dans l'intervalle 7307 et 81850 dollars de Singapour. Nous constatons qu'il y a une forte chance d'avoir des outliers sur des valeurs maximales des factures. **Premier audit de la base demographics.csv**

```
dim(demographics) #afficher la dimension de la base
summary(demographics) #afficher les statistiques descriptives de la base de donnée
sapply(demographics, class) #afficher les types de chaque variable
colSums(is.na(demographics)) #verifier le nombre des valeurs manquantes
```

La table demographics contient 3000 observations et 5 variables parmi lesquels le id des patients qui est le même que dans la table clinicalData qui permettra de faire la jointure de la table avec la table clinicalData et les factures.

Ensuite, il y a la variable gender de type factor qui détermine si le patient est une femme ou un homme. Nous constatons aussi que pour les femmes et pour les hommes il y a 2 notations. Female et f pour les femmes et Male et m pour les hommes. Il y a finalement 1497 femmes(1396+101) et 1503 hommes(1333+170). Donc, nous avons une répartition égalitaire entre hommes et femmes dans notre base de données. Pour une utilisation ultérieure de la variable, il est nécessaire de remplacer les modalités et distinguer 2 classes femmes et hommes sans modalités différentes pour chaque classe.

La variable suivante est ethnicité des patients (Chinois, Indien, Malais et autres). La variable est de type factor. De la même manière que la variable gender, la variable race nécessite un nettoyage dû au fait que certaines ethnicités sont les mêmes, mais ils sont écrits différement.

La variable resident_status est de type factor et nous dit si le patient est un résident permanent(RP), Singapourien (Singapore citizen, Singaporean) ou étranger(Foreigner). La majorité des patients sont des Singapouriens (1782+610). Une classe plus petite est celle des résidents permanents avec 465 observations. La classe monoritaire est celle des étrangers avec 143 observations.

La variable date_of_birth est la date de naissance des patients mais elle est de type factor. Pour une utilisation ultérieure il est nécessaire de la convertir en type date.

Dans cette table ,il n'y a pas des valeurs manquantes.

Jointure des tables avec information sur les factures des patients Afin d'obtenir la table complète sur les factures , nous faisons la jointure des tables billID et billAmount par la clé de jointure "bill_id" commune pour les 2 tables.

Ensuite nous faisons le range pour avoir l'ordre des patients qu'il y aura dans les autres tables pour voir ensuite plus facilement si les jointures sont biens faites.

```
#jointure des tables bill_id et bill_amount
billmerged<-merge(billID,billAmount,by="bill_id")
#le triage par variable patien_id
billmerged<-arrange(billmerged,patient_id)
```

Calcul de variable target

La variable target que nous cherchons à expliquer est le coût de traitement. Pour calculer la target, il est nécessaire de sommer les valeurs des factures pour le traitement de patient.

Nous allons calculer le nombre des patients uniques dans la base de données et le nombre des patients avec la date d'admission unique pour comprendre si nous avons finalement un seul traitement par patient ou nous avons plusieurs traitements sur périodes différents pour un même patient.

```
 # Nombre de patients distincts
select(billmerged,patient_id) %>%distinct %>% count
```

```
##      n
## 1 3000
```

```
#Nombre de valeurs distincts par patient et par date d'admission

select(billmerged,patient_id,date_of_admission) %>%distinct %>% count
```

```
##      n
## 1 3400
```

D'une part ,nous constatons que nous avons 3000 patients distinct dans cette base.D'autre part ,en regroupant la base par patient_id et par date_of_admission, nous constatons qu'il y a 3400 valeurs uniques. Nous avons bien pour certains patients plusieurs traitements à des date différentes. Pour calculer dans ce cas la variable cible à prédire il faut sommer les valeurs des factures par le patient_id et par date_of_admission. On appellera cette nouvelle variable sumAmount.

```
billmerged<- billmerged %>% group_by(patient_id,date_of_admission) %>%
  mutate(sumAmount=sum(amount)) %>% ungroup
```

Après avoir calculés la variables cible nous vérifions si nous avons biens 3400 valeurs distinctes pour le coût de traitement.

```
select(billmerged,sumAmount) %>%distinct %>% count #3400 observations
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  3400
```

Nous confirmons que nous avons 3400 valeurs distinctes pour notre variable cible: une observation par patient et par date de début de traitement.

```
dim(billmerged)
```

```
## [1] 13600     5
```

```
#suppression des variables bill_id et amount que nous n'avons plus besoin
bill<-select(billmerged,-bill_id, -amount)
#La table finale des factures sans doublons de l'informationt des montants totales de traitement
bill_by<-bill %>% distinct
dim(bill_by)
```

```
## [1] 3400    3
```

```
sapply(bill_by, class) # Les types des variables dans la nouvelle base
```

```
##        patient_id date_of_admission         sumAmount
##          "factor"          "factor"         "numeric"
```

Ensuite, nous regardons combien d'observations nous avons dans notre nouvelle base avec variable cible. Il y a au total 13600 observations et 4 variables. Sachant que nous devons maintenant avoir 3400 observations au lieu de 13600, nous supprimons ensuite les doublons ainsi que les variables bill_id et amount. Dans cette nouvelle base, nous avons finalement 3400 observations et 3 colonnes dont sumAmount notre variable cible de type numérique et 2 variables patient_id et date_of_admission de type factor qui sont nos clés de jointure avec la table clinicalData.

Creation de la table globale

Nous faisons le tri de la base par id des patients pour observer plus facilement si la jointure a été bien faite. Ensuite,nous changeons le nom de la variable id par patient_id pour faire référence à l'identifiant du patient et donc de soulever toute sorte d'ambiguïté.

```r
clinicalDATA<-arrange(clinicalDATA,id) #triage par id de patient
#changement du nom de la variable
names(clinicalDATA)[names(clinicalDATA) == "id"] <- "patient_id"
```

Nous allons calculer le nombre des patients uniques dans cette table et le nombre des patients avec la date d'admission unique pour comprendre si nous avons finalement un seul traitement par patient où nous avons plusieurs traitements sur des périodes différentes pour le même patient. Nous calculons exactement la même chose qu'avec la table billmerged.

Nous avons déjà vu que dans la base clinicalData il y avait plusieurs fois le même id patient dans le summary. Il faut vérifier alors, si les lignes ayants le même id patient ne sont pas des doublons impurs.

```r
#nombre de differents patients dans la base clinicalData
select(clinicalDATA,patient_id) %>%distinct %>% count
```

```
##      n
## 1 3000
```

```r
#nombre de differents patients avec differente date d'admission dans la base clinicalData
select(clinicalDATA,patient_id,date_of_admission) %>%distinct %>% count
```

```
##      n
## 1 3400
```

Nous avons donc les mêmes résulats qu'avec la table des factures.À savoir, 3000 patients distincts et 3400 observations en regroupant par patient et date d'admission.

```r
#exemple de patient avec 2 differentes dates d'admission
subset(clinicalDATA,patient_id %in%
         c("4e46fddfa404b306809c350aecbf0f6a","0eacfb2daed1f3ba2adf32e293bc05a6"))
```

```
##                           patient_id date_of_admission date_of_discharge
## 201  0eacfb2daed1f3ba2adf32e293bc05a6        2012-07-12        2012-07-26
## 202  0eacfb2daed1f3ba2adf32e293bc05a6        2014-01-28        2014-02-09
## 203  0eacfb2daed1f3ba2adf32e293bc05a6        2014-03-25        2014-04-02
## 1025 4e46fddfa404b306809c350aecbf0f6a        2011-11-23        2011-12-05
## 1026 4e46fddfa404b306809c350aecbf0f6a        2013-01-06        2013-01-17
## 1027 4e46fddfa404b306809c350aecbf0f6a        2013-01-23        2013-02-02
## 1028 4e46fddfa404b306809c350aecbf0f6a        2013-09-16        2013-09-27
##      medical_history_1 medical_history_2 medical_history_3 medical_history_4
## 201                  0                 1                No                 0
## 202                  0                NA                No                 0
## 203                  0                 0                 0                 0
## 1025                 1                 1                 0                 0
## 1026                 0                 0                 0                 0
## 1027                 0                 0                 0                 0
## 1028                 1                 1                No                 0
##      medical_history_5 medical_history_6 medical_history_7 preop_medication_1
## 201                  1                 1                 0                  1
## 202                 NA                 1                 0                  0
## 203                  0                 0                 0                  1
## 1025                 0                 0                 0                  1
```

```
## 1026                0                 0                 0                  0
## 1027                0                 0                 1                  0
## 1028                0                 1                 0                  0
##      preop_medication_2 preop_medication_3 preop_medication_4
## 201                   1                  1                  1
## 202                   1                  1                  0
## 203                   1                  1                  1
## 1025                  1                  1                  0
## 1026                  1                  1                  1
## 1027                  0                  1                  1
## 1028                  1                  1                  0
##      preop_medication_5 preop_medication_6 symptom_1 symptom_2 symptom_3
## 201                   1                  0         0         1         0
## 202                   1                  1         1         1         0
## 203                   0                  1         1         1         1
## 1025                  0                  1         1         0         1
## 1026                  1                  1         1         0         1
## 1027                  0                  1         0         0         0
## 1028                  1                  0         1         1         0
##      symptom_4 symptom_5 lab_result_1 lab_result_2 lab_result_3 weight height
## 201          1         0         15.4         23.9           87   76.8    168
## 202          1         1         16.4         22.6          109   76.8    168
## 203          1         0         15.4         23.2          107   78.8    168
## 1025         1         0         16.1         25.5          111   71.4    166
## 1026         0         0         14.1         24.1          103   71.4    166
## 1027         0         0         13.0         27.0           91   70.4    166
## 1028         0         1         11.7         22.6           84   70.4    166
```

```r
#Jointure de clinicalDATA et bill_by par patient_id et par date_admission
billClinic<-merge(clinicalDATA,bill_by,by=c("patient_id","date_of_admission"))
 #jointure des tables billClinic et demographics
data<-merge(billClinic,demographics,by="patient_id")
```

Nous avons ensuite décidé de récupérer les observations pour les 2 patients ayant les fréquences la plus élevées dans notre table pour comparer les observations et voir s'il s'agit de doublons ou non. En comparant les observations pour chaque id nous constatons que ce sont biens des observations différentes et pas les doublons impurs .Toutes les variables pour le même patient sont différentes à chaque fois pour chaque nouvel traitement.

Ensuite nous faisons la jointure de la base clinicalDAta avec la base bill_by par les clés de jointure patient_id et date_of_admission et nous obtenons la table billClinic avec les informations sur les factures et le traitement des patients. Il reste à rajouter les informations personnelles des patients. Pour cette raison nous faisons la jointure de la table obtenu billClinic avec la table demographics par la clé de jointure patient_id. Comme nous avons vu précédemment, dans la base demographics il y a 3000 observations, les mêmes patients que dans tous les autres tables, parce que le nombre des id uniques est toujours le même 3000 patients dans chaque table. Il suffit donc d'utiliser la clé de jointure patient_id.

Nous obtenons finalement la base complète que nous allons utiliser ultérieurement pour prédire le coût de traitement. Cependant, certaines variables nécessitent un traitement supplémentaire, soit la conversion de type de variable, comme dans le cas des dates et mettre en forme d'autres. De plus, nous allons créer des nouvelles variables qui pourrons nous aider à prédire le coût de traitement.

Partie 2. Mise en forme de la base de données

Nettoyage des données

Classes des variables

Avant de commencer à travailler avec les variables nous allons regarder de nouveau quelles variables n'ont pas de bon format et nécessitent d'être retravaillées.

```
sapply(data,class)
```

```
##         patient_id  date_of_admission  date_of_discharge  medical_history_1
##           "factor"           "factor"           "factor"          "integer"
##  medical_history_2  medical_history_3  medical_history_4  medical_history_5
##          "numeric"           "factor"          "integer"          "numeric"
##  medical_history_6  medical_history_7 preop_medication_1 preop_medication_2
##          "integer"          "integer"          "integer"          "integer"
## preop_medication_3 preop_medication_4 preop_medication_5 preop_medication_6
##          "integer"          "integer"          "integer"          "integer"
##          symptom_1          symptom_2          symptom_3          symptom_4
##          "integer"          "integer"          "integer"          "integer"
##          symptom_5       lab_result_1       lab_result_2       lab_result_3
##          "integer"          "numeric"          "numeric"          "numeric"
##             weight             height          sumAmount             gender
##          "numeric"          "numeric"          "numeric"           "factor"
##               race    resident_status      date_of_birth
##           "factor"           "factor"           "factor"
```

Il est nécessaire de convertir tous les variables dates en format date. De plus, medical_history_3 nécessite un nettoyage pour convertir son format factor en integer, parce que nous avons vu précédemment que cette variable contient du texte.

Conversion des variables dates qui sont en facteur dans les vrais dates

Sur cette étape nous convertissons les 3 variables date_of_admission, date_of_discharge et date_of_birth en format date.

```
data$date_of_admission<-as.Date(data$date_of_admission)
data$date_of_discharge<-as.Date(data$date_of_discharge)
data$date_of_birth<-as.Date(data$date_of_birth)
```

Ensuite nous vérifions bien que le format des variables est de type date ce qui est bien notre cas.

```
class(data$date_of_admission)
```

```
## [1] "Date"
```

```
class(data$date_of_discharge)
```

```
## [1] "Date"
```

```r
class(data$date_of_birth)
```

```
## [1] "Date"
```

Nettoyage de la variable medical_history_3

Pour remplacer les modalités de la variables medical_history_3 nous convertissons cette variable de format factor en format character. Nous vérifions aussi combiens de modalités prends cette variable et quels sont ces modalités.

```r
data$medical_history_3<-as.character(data$medical_history_3)
unique(data[,"medical_history_3"])
```

```
## [1] "0"   "No"  "1"   "Yes"
```

La variable contienne 4 modalités: 0, 1, *No* qui doit prendre la modalité 0 et *Yes* qui doit prendre la modalité 1. En effet, nous remalacerons *No* par 0 et *Yes* par 1.

```r
data$medical_history_3[data$medical_history_3=="No"]<-"0"
data$medical_history_3[data$medical_history_3=="Yes"]<-"1"
data$medical_history_3<-as.integer(data$medical_history_3)
class(data$medical_history_3)
```

```
## [1] "integer"
```

Après avoir remplacé les modalités nous convertissons la variable en format integer et nous vérifions si le format est bon ce qui est notre cas.

Nettoyage de la variable gender

Pour remplacer les modalités de la variables gender nous convertissons cette variable de format factor en format character. Nous vérifions aussi combiens de modalités prends cette variable et quels sont ces modalités.

```r
data$gender<-as.character(data$gender)
unique(data$gender)
```

```
## [1] "Female" "Male"   "m"      "f"
```

La variable contienne 4 modalités: “Female” et “f” pour dire si c’est une femme et “Male” et “m” pour dire si c’est un homme. Nous remplaçons alors les modalités concernant les femmes par 1 et les modalités pour les hommes par 0.

```r
data$gender[data$gender %in% c("f","Female")]<-"1"
data$gender[data$gender %in% c("m","Male")]<-"0"
data$gender<-as.integer(data$gender)
class(data$gender)
```

```
## [1] "integer"
```

Après avoir remplacé les modalités nous convertissons la variable en format integer et nous vérifions si le format est bon ce qui est notre cas.

Preparation de la variable resident_status

Nous avons décidé de retravailler la variable resident_status. Pour retravailler les modalités de la variables resident_status nous convertissons cette variable de format factor en format character. Nous vérifions aussi combiens de modalités prends cette variable et quels sont ces modalités.

```
data$resident_status<-as.character(data$resident_status)
unique(data$resident_status)
```

```
## [1] "Singaporean"       "Singapore citizen" "Foreigner"
## [4] "PR"
```

La variable contienne 4 modalités: “Singaporean” et “Singapore citizen” pour dire si le patient est Singapourien. “PR” est pour dire si le patient est en résident permanent de Singapour et “Foreigner” pour dire si le patient est un étranger. Nous avons décidé de regrouper les catégories “Singaporean”,“Singapore citizen” et “PR” et dire que tous les 3 modalités vont représenter la catégorie 1. Nous avons regroupé ces 3 catégories, parce que les résidents permanents et les Singapouriens doivent avoir une assurance et donc le traitement doit être moins cher. Cependant, les étrangers n’ont pas toujours de l’assurance et leur traitement doit coûter plus cher et donc le montant des factures sera plus élevé par rapport aux résidents de Singapour. Nous remplaçons la catégorie “Foreigner” par 0 pour obtenir une variable binaire.

```
data$resident_status[data$resident_status %in%
                       c("Singaporean","Singapore citizen","PR")]<-"1"
data$resident_status[data$resident_status=="Foreigner"]<-"0"
data$resident_status<-as.integer(data$resident_status)
class(data$resident_status)
```

```
## [1] "integer"
```

Après avoir remplacé les modalités nous convertissons la variable en format integer et nous vérifions si le format est bon ce qui est notre cas.

Creation des nouvelles variables

Creation de la variable qui represente la periode de traitement de patient

Nous avons décidé de calculer une nouvelle variable qui représente la durée de traitement. Cette variable est calculée comme la différence des dates date_of_discharge et date_of_admission. Nous attendons que cette variable aura une importance sur le coût de traitement, parce que si le traitement est plus long, il y plus de médicaments, injections ce qui influence sur le montant de la facture.

```
data<-data %>% mutate(treatPeriode=difftime(date_of_discharge,date_of_admission))
data$treatPeriode<-as.numeric(data$treatPeriode)
class(data$treatPeriode)
```

```
## [1] "numeric"
```

Après avoir calculé la variable nous convertissons la variable en format numeric et nous vérifions si le format est bon ce qui est notre cas. La variable a eu besoin de conversion parce que avant le type de la variable étais le difftime ce qui ne permet pas l’utilisation ultérieure dans la modélisation.

Creation de la variable qui represente l'age de patient

Ensuite, nous avons décidé de calculer l'âge des patients au moment de début de traitement, parce que probablement l'âgé pourrait jouer un rôle sur le coût de traitement. Les personnes plus âgés souvent ont des maladies plus graves et donc le traitement doit être plus coûteux.

L'âge est calculé en jours. Pour représenter en années nous avons devisé par 365 et nous avons récupérés la partie integer pour savoir le nombre d'années complets au moment de traitement.

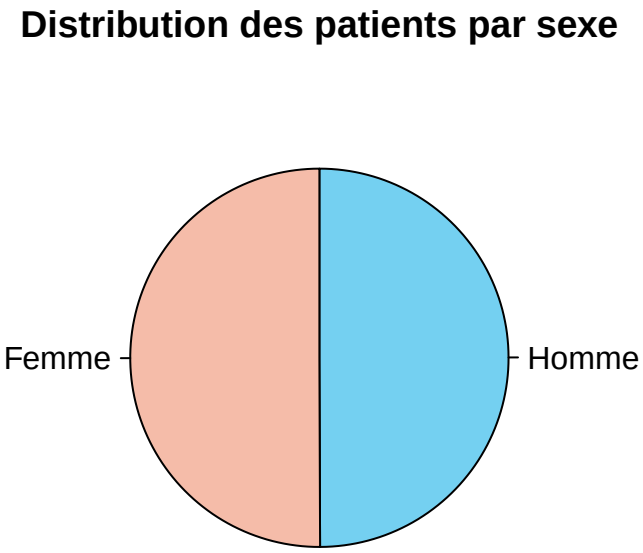
```
data<-data %>% mutate(age=difftime(date_of_admission,date_of_birth,unit="days"))
data$age<-as.numeric(data$age)
data$age<-data$age/365
data$age<-floor(data$age)#garder la partie integer pour avoir le nombre d'annees complets
```

Exploration plus fine des données

```
#création de la table pour réaliser le graphique
data2<-data
data2$gender<-as.integer(data2$gender)

count<-table(data2$gender)
count
```

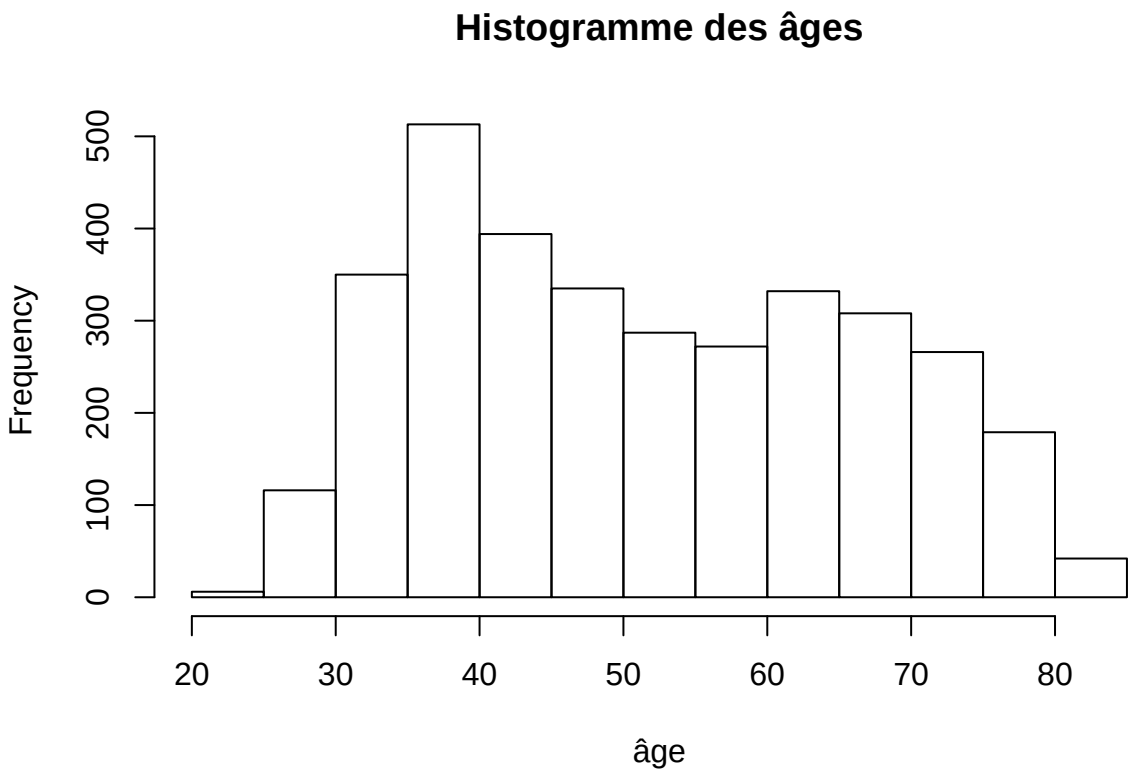
```
##
##    0    1
## 1698 1702
```

```
#création du graphique secteur
pie(count,main = "Distribution des patients par sexe",
    col=c("#74D0F1","#F5BCA9"),labels =c("Homme","Femme"),clockwise = T)
```



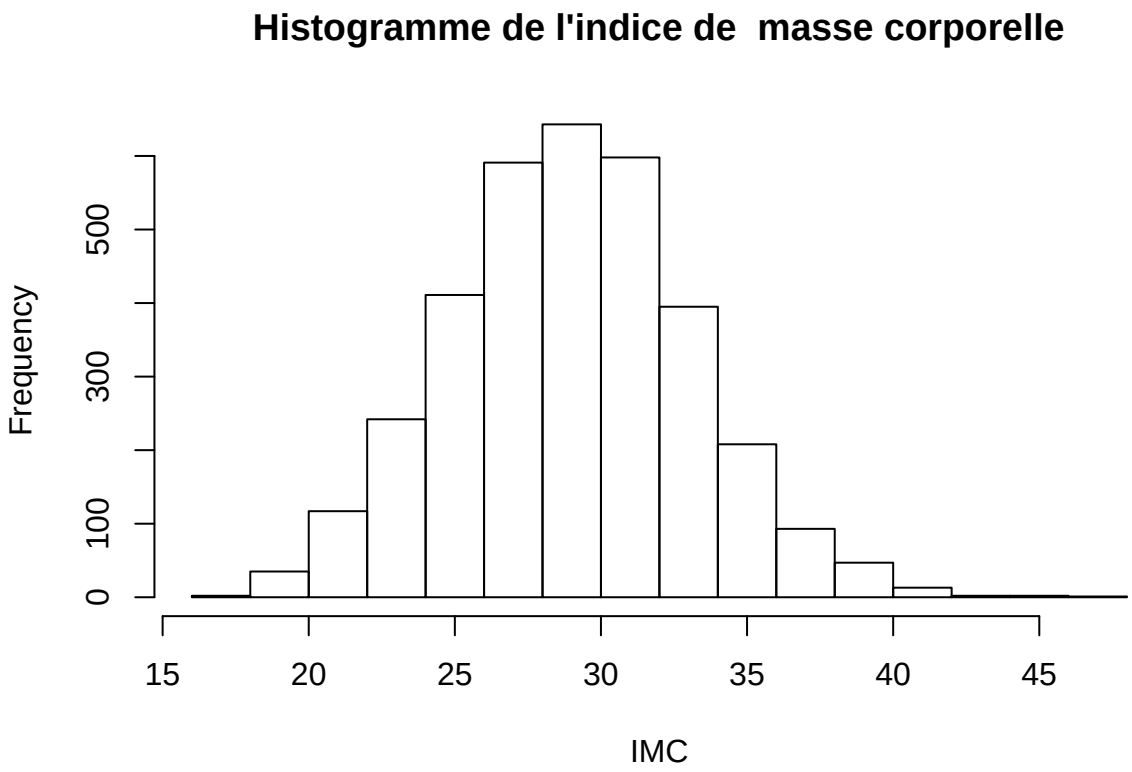
Le diagramme en secteurs montre que notre jeu de données est plutôt équilibré entre les deux sexes.

```
hist(data$age,main="Histogramme des âges",xlab="âge")
```



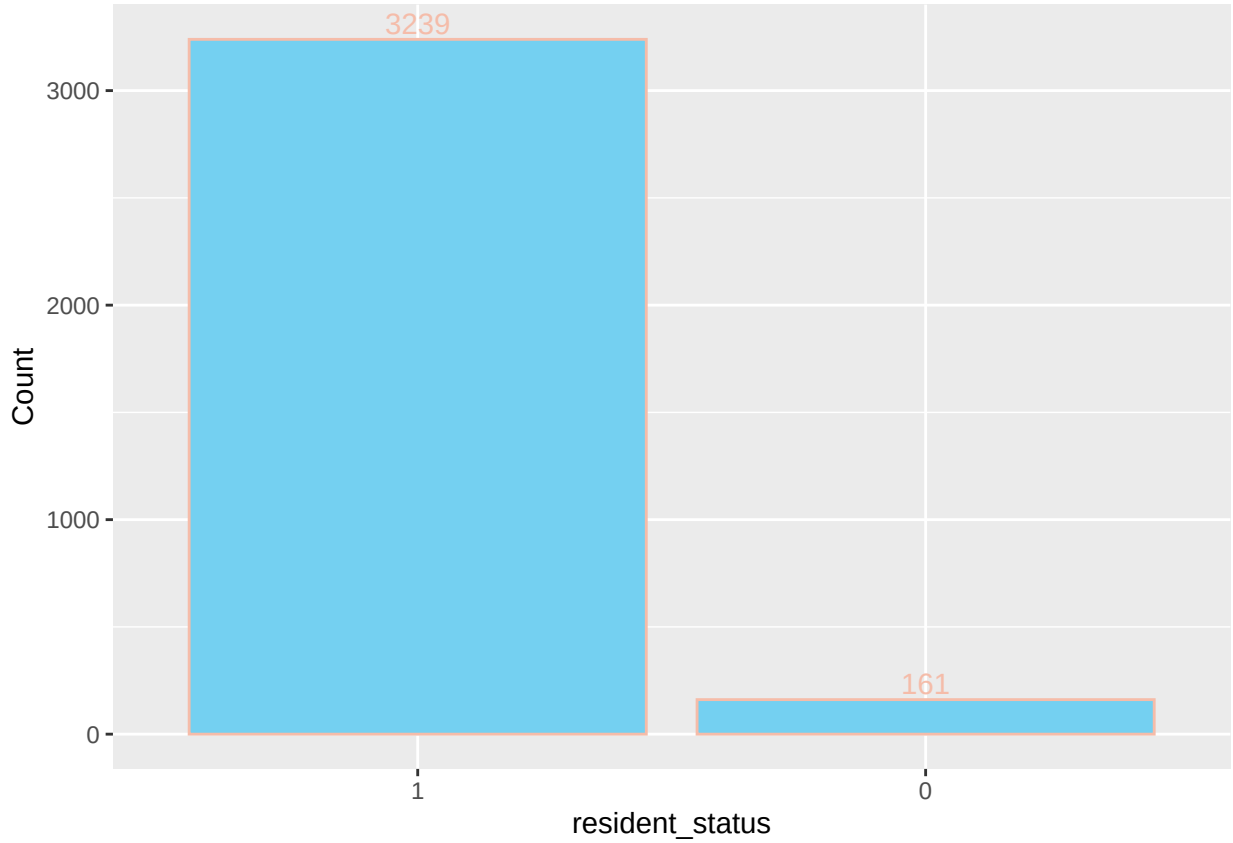
La figure ci-dessus présente la distribution de l'âge au sein de la population étudiée, on a plutôt des patients adultes de tous les âges avec une concentration entre 35 et 40 ans et une minorité entre 20 et 25 ans.

```
datag<-data
datag<-datag %>% mutate(IMC=weight/(height*0.01)^2)
hist(datag$IMC,main="Histogramme de l'indice de  masse corporelle",xlab="IMC")
```



L'histogramme de l'indice de la masse corporelle est sous forme de cloche, les valeurs les plus fréquentes d'IMC sont situées entre 25 et 35 kg/M , pas beaucoup de cas d'obésité ni d'anorexie.

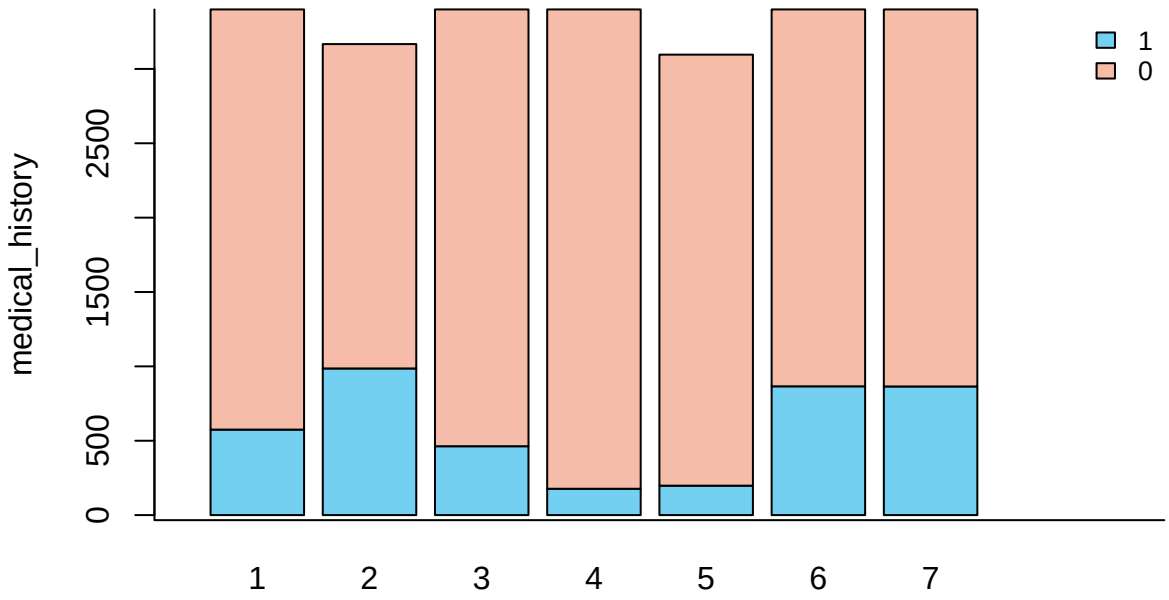
```
count_resident_status <- data %>%
  count(resident_status)

ggplot(count_resident_status, aes(x = reorder(resident_status, -n),y=n))+
  geom_bar(stat="identity", fill="#74D0F1", color = "#F5BCA9")+
  geom_text(aes(label = n),vjust =-0.25,color = "#F5BCA9")+
  labs(x="resident_status", y="Count")
```



On constate que les étrangers non titulaire d'une résidence permanente représente une minorité au sein de la population étudiée.

```
l1 <- c("medical_history_1","medical_history_2","medical_history_3",
        "medical_history_4","medical_history_5","medical_history_6",
        'medical_history_7')

for (i in 1:length(l1))
{
  print(table(data[l1[i]]))
}
```

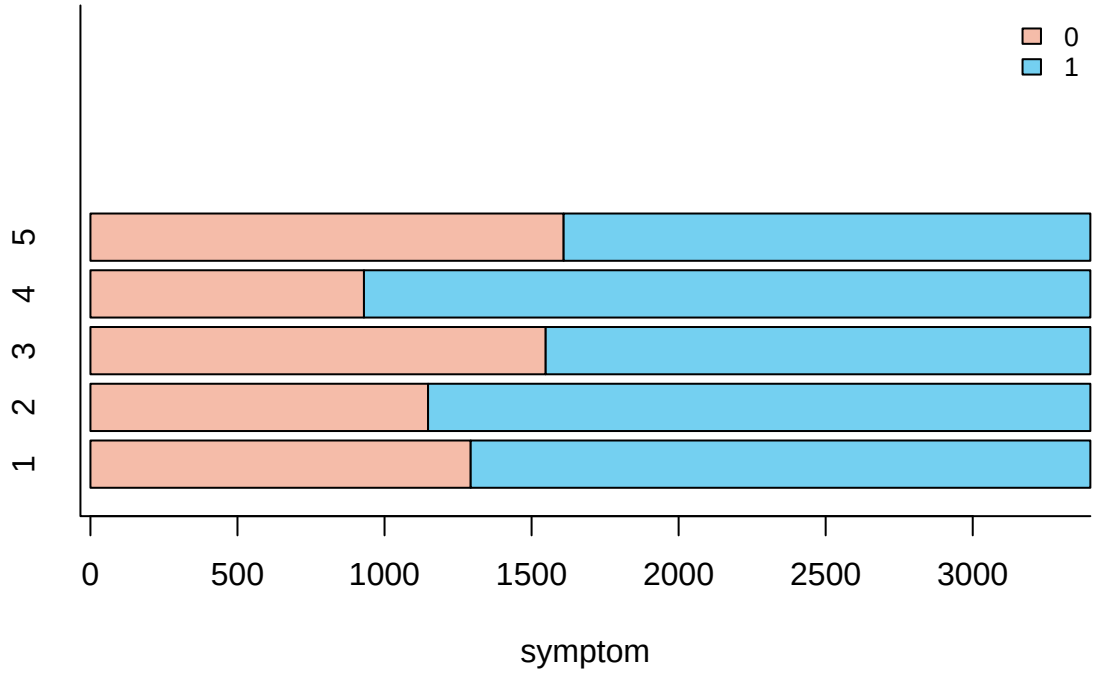
```
par(bty="l")
medical_history=c(seq(1, 7, by=1))
MOD1=c(575 ,986, 463,177,198,866,865  )
MOD0=c(2825,2181,2937,3223,2898,2534,2535)

dataMOD<-cbind(MOD1,MOD0)
barplot(t(dataMOD),beside=F,col=c("#74D0F1","#F5BCA9"),
        ylab="medical_history",names=medical_history,horiz=F,space=0.2,
        xlim=c(0,10))

legend(x="topright", legend=c("1","0"), cex=0.8,fill=c("#74D0F1","#F5BCA9"),bty="n")

box()
```



On constate à partir du diagramme ci-dessus que les histoires médicales les plus commun chez les patients sont les histoire 2,6 et 7.

```
l2 <- c("symptom_1","symptom_2","symptom_3","symptom_4","symptom_5")

for (i in 1:length(l2))
{
  print(table(data[l2[i]]))
}
```

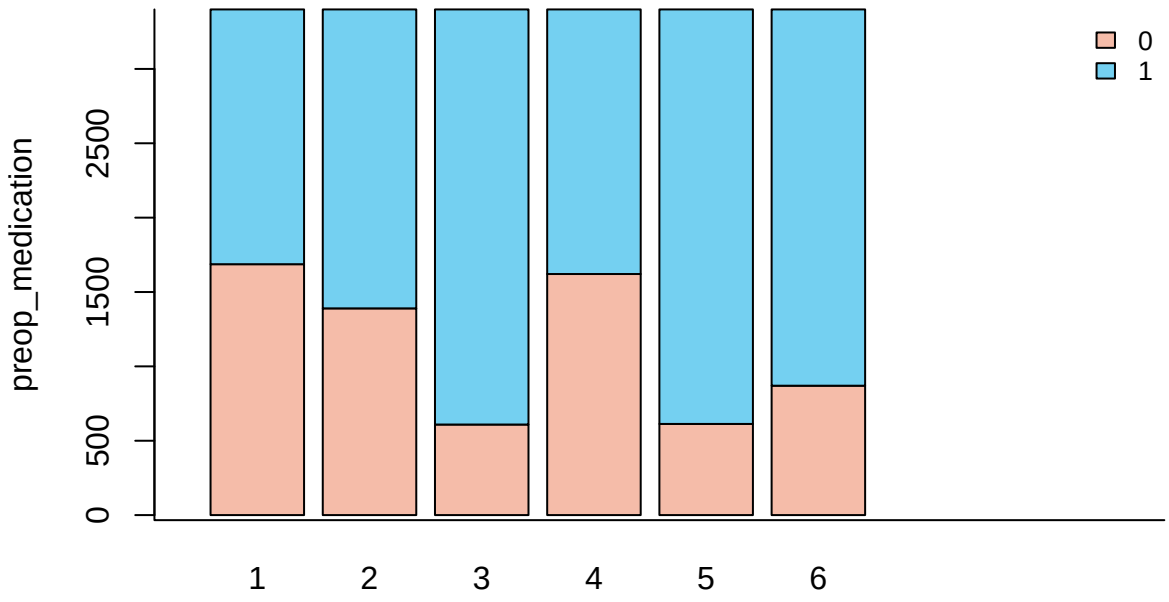
```
par(bty="l")
symptom=c(seq(1, 5, by=1))

MOD0=c(1293 ,1148 ,1548 ,930 ,1609 )
MOD1=c(2107 ,2252 , 1852,2470,1791 )

dataMOD<-cbind(MOD0,MOD1)
barplot(t(dataMOD),beside=F,col=c("#F5BCA9","#74D0F1"),
        xlab="symptom",names=symptom,horiz=T,space=0.2,
        ylim=c(0,10))

legend(x="topright", legend=c("0","1"), cex=0.8,fill=c("#F5BCA9","#74D0F1"),bty="n")

box()
```



La figure ci-dessus révélé que la majorité des patients de la base de données souffrent d'au moins un des 5 symptômes.

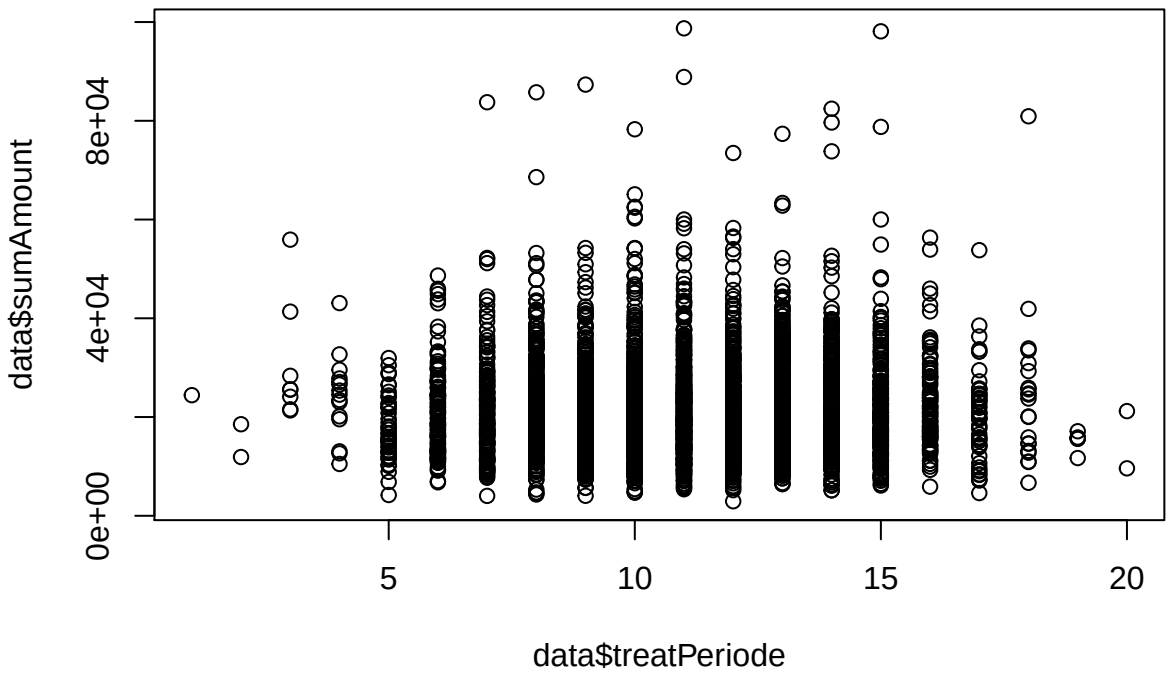
```
l3 <- c("preop_medication_1","preop_medication_2","preop_medication_3",
        "preop_medication_4","preop_medication_5","preop_medication_6")

for (i in 1:length(l3))
{
  print(table(data[l3[i]]))
}
```

```
par(bty="l")
preop_medication=c(seq(1, 6, by=1))
MOD0=c(1687  ,1390 , 609 ,1621 ,613 ,870   )
MOD1=c(1713 ,2010 ,2791 ,1779 ,2787 ,2530)

dataMOD<-cbind(MOD0,MOD1)
barplot(t(dataMOD),beside=F,col=c("#F5BCA9","#74D0F1"),
        ylab="preop_medication",names=preop_medication,horiz=F,space=0.2,
        xlim=c(0,10))

legend(x="topright", legend=c("0","1"), cex=0.8,fill=c("#F5BCA9","#74D0F1"),bty="n")

box()
```



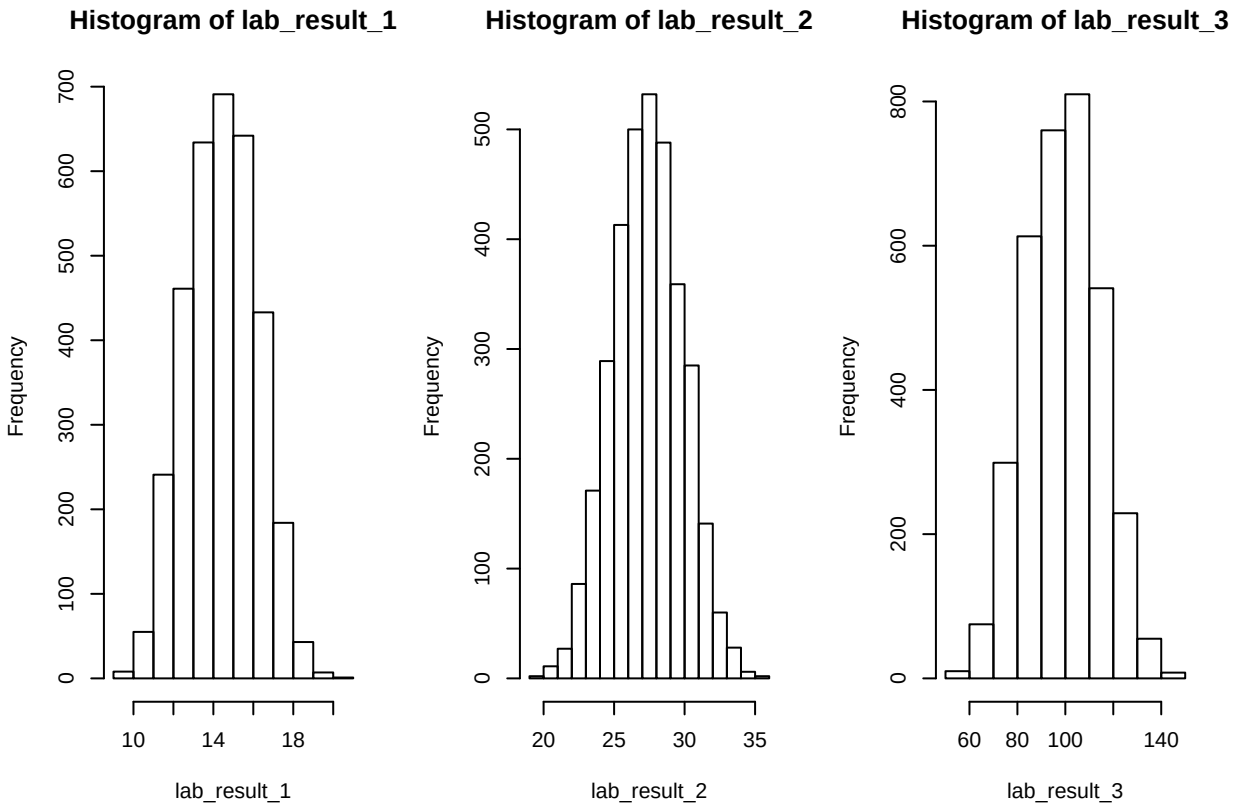
On remarque que dans la majorité des cas , les patient prennent des médicaments préopératoire.

```
plot(data$treatPeriode,data$sumAmount)
```



Le montant de la facture n'évolue pas linéairement en fonction de la durée d'hospitalisation comme on a tendance à le penser.

```
lab_result_1 <- data$lab_result_1
lab_result_2 <- data$lab_result_2
lab_result_3 <- data$lab_result_3
par(mfrow = c(1,3))
hist(lab_result_1)
hist(lab_result_2)
hist(lab_result_3)
```

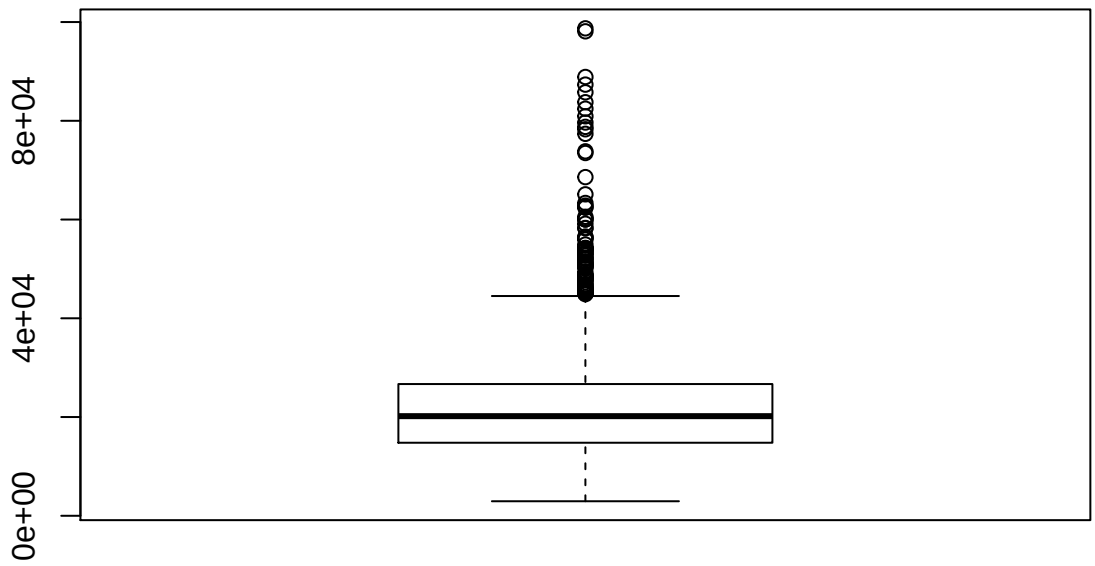


L'allure de l'histogramme de lab_result_1 et celui de lab_result_3 sont quasiment identiques, alors que pour lab_result_3, il y a plus de bâtons.

```
summary(data$sumAmount)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2946   14793   20180   21859   26680   98724
```

```
boxplot (data$sumAmount, main="Boîte à moustache de la variable cible")
```



Le montant de facture le plus bas ne dépasse 2900 dollars singapouriens, tandis que la valeur maximale vaut 98724, la moyenne est autour 20100 dollars de Singapour.

Vu le boxplot 75% des patients ont payé pour sont traitement un montant total inférieur à 26680, mais il existe plusieurs valeurs extrêmes avec un coût de traitement très élevé, parfois allant jusqu'à 80 000 dollars de Singapour et même plus.

Imputatation des valeurs manquantes

Le pourcentages de valeurs manqyantes par colonne et par observation

Comme nous avons vu précédemment, nous avons 2 variables avec valeurs manquantes. Pour comprendre si nous pouvons supprimer les observations pour garder seulement les informations complètes ou il est nécessaire de procéder par imputations des valeurs manquantes.

Nous calculons alors le pourcentage des valeurs manquantes par variable.

```
varna<-colSums(is.na(data))/nrow(data) #valeurs manquants par variable
varna
```

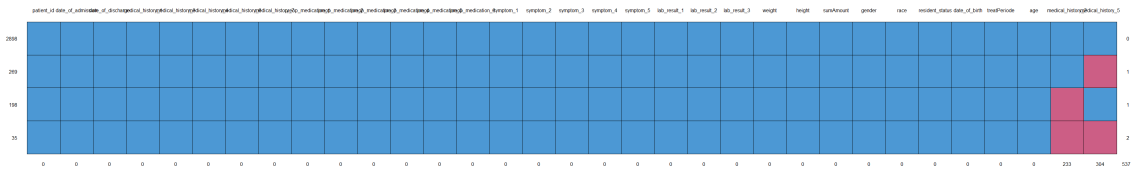
```
##         patient_id  date_of_admission  date_of_discharge  medical_history_1
##         0.00000000         0.00000000         0.00000000         0.00000000
##  medical_history_2  medical_history_3  medical_history_4  medical_history_5
##         0.06852941         0.00000000         0.00000000         0.08941176
##  medical_history_6  medical_history_7 preop_medication_1 preop_medication_2
##         0.00000000         0.00000000         0.00000000         0.00000000
## preop_medication_3 preop_medication_4 preop_medication_5 preop_medication_6
```

```
##         0.00000000          0.00000000          0.00000000          0.00000000
##          symptom_1           symptom_2           symptom_3           symptom_4
##         0.00000000          0.00000000          0.00000000          0.00000000
##          symptom_5        lab_result_1        lab_result_2        lab_result_3
##         0.00000000          0.00000000          0.00000000          0.00000000
##             weight              height           sumAmount              gender
##         0.00000000          0.00000000          0.00000000          0.00000000
##               race     resident_status       date_of_birth       treatPeriode
##         0.00000000          0.00000000          0.00000000          0.00000000
##                age
##         0.00000000
```

Nous observons que pour medical_history_2 il manque 6.8% des observations et 8.9% des observations manquent pour la variable medical_history_5. Le pourcentage des valeurs manquantes est un peu dessus des 5% des valeurs manquantes. Donc, nous ne pouvons pas les supprimer. Il faut alors procéder par imputations des valeurs manquantes. De plus, nous ne connaissons pas si les valeurs manquantes ne manquent pas pour les mêmes observations. Il est probable que les valeurs manquantes sont des 2 variables absentes complètement pour les observations différentes. Dans ce cas la suppression des observations incomplets peut mener à une suppression jusqu'à 15.7% des observations ce qui est inacceptable pour nous. Ensuite nous allons étudier le pattern des valeurs manquantes pour comprendre la nature des valeurs manquantes et choisir la méthode d'imputation.

Pattern des valeurs manquantes

```
md.pattern(data)
```



En observant le pattern des valeurs manquantes nous constatons qu'il y a 269 observations pour lesquelles il manque seulement medical_history_5, 198 observations pour lesquelles il manque seulement les valeurs medical_history_2 et seulement 35 observations avec valeurs manquantes pour les 2 variables. Ça veut dire que l'absence d'une variable ne dépends pas de l'absence d'une autre variable. Selon le pattern nous avons la situations MAR (Missing at Random). Il n'y a pas donc de dépendance entre les variables avec valeurs manquantes.

Imputation des variables manquantes par regression logistique

Nous allons ensuite imputer les valeurs manquantes avec bibliothèque mice en utilisant régression logistique qui permet prédire les variables catégorielles binaires pour 2 et plus variables avec valeurs manquantes. Pour appliquer cette méthode nous avons besoin de convertir les variables à prédire de numérique en factor.

```
data$medical_history_2<-as.factor(data$medical_history_2)
data$medical_history_5<-as.factor(data$medical_history_5)
```

```
data2<-select(data,-patient_id,-date_of_admission,-date_of_discharge,
              -date_of_birth,-race,-sumAmount)
imputed_Data <- mice(data2, m=5, maxit = 20, method = 'logreg', seed = 500)
completeData <- complete(imputed_Data,3)
```

```
completeData$medical_history_2<-as.integer(as.character(completeData$medical_history_2))
completeData$medical_history_5<-as.integer(as.character(completeData$medical_history_5))
table(data2$medical_history_2)
```

```
##
##    0    1
## 2181  986
```

```
table(data2$medical_history_5)
```

```
##
##    0    1
## 2898  198
```

```
table(completeData$medical_history_2)
```

```
##
##    0    1
## 2342 1058
```

```
table(completeData$medical_history_5)
```

```
##
##    0    1
## 3179  221
```

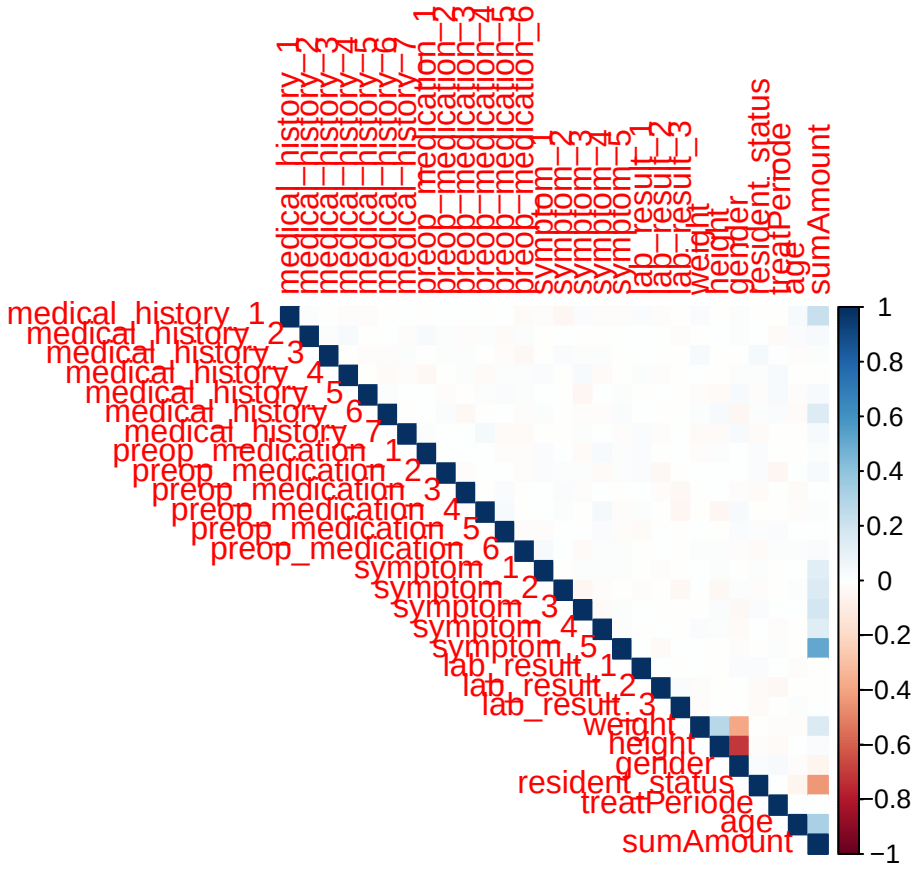
En observant les fréquences des valeurs avant et après imputation nous constatons que les valeurs manquantes ont été bien remplacés.

Ensuite nous préparons notre base pour modélisations. Il faut à cette table concaténer la variable target avec la base des variables explicatives. Il suffit d'utiliser la fonction cbind qui permet simplement concaténer 2 tables, parce que lors de l'imputation des observations n'a pas été changé. Ensuite nous allons changer le nom de la variable cible à nom utilisé précédemment. Et finalement nous calculons le logarithme de sumAmount pour la raison des outliers observés lors du summary et du boxplot vus précédemment. Le logarithme permet de réduire la variabilité de sumAmount et doit nous permettre améliorer les résultats de la modélisation. Cette technique est utile est permet de ne pas supprimer les valeurs aberrantes pour garder au plus de l'information sur la population.

```
dataModel<-cbind(completeData,data$sumAmount)
#changement du nom de la variable
names(dataModel)[names(dataModel) == "data$sumAmount"] <- "sumAmount"
dataModelLog<-dataModel
dataModelLog$sumAmount<-log(dataModelLog$sumAmount)
```

correlation entre les variables numériques

```
CM<-cor(dataModel, use = "complete.obs")
corrplot(CM, method ="color",type="upper")
```



Cette matrice de corrélation montre qu'il y a très peu de variables dans notre jeu de données qui sont fortement corrélés. En effet, les corrélations sont très faibles de manière générale.

Toutefois la variable sumAmount est positivement corrélé aux variables medical_history_1, medical_history_7,symptom_1,symptom_2,symptom_3,symptom_4,symptom_5,weight, age et est négativement corrélée à la variable resident_status. Le genre (gender) est négativement corrélé au poids (weight) et à la taille (height); et ces deux derniers sont positivement corrélées entre eux.

Nous nous proposons donc de faire une matrice de corrélation uniquement pour ces variables qui semblent relativement fortement corrélées afin de voir numériquement l'ampleur de chaque corrélation pour comprendre si nous avons le problème de multi colinéarité. Notamment, nous calculons la corrélation entre height et gender qui ont une forte corrélation négative selon la carte de chaleur (rouge foncé).

```
f<-select(dataModel,gender,height)
matx<-cor(f)
mat<-round(matx,2)
mat
```

```
##        gender height
## gender   1.00  -0.71
## height  -0.71   1.00
```

Dans cette matrice on remarque que même les corrélations les plus fortes entre les variables ne dépassent pas le seuil de 80% positif ou négatif. Notamment, la corrélation -71% n'est pas trop élevée pour exiger un traitement supplémentaire. Il n'y a donc pas lieu de supprimer des variables pour cause de forte corrélation.

Nous pouvons donc utiliser tous ces variables pour faire de la prédiction ultérieurement. Nous nous attendons donc à ce que les variables significativement corrélées avec la variable cible soient les variables les plus importantes dans la prédiction du coût de traitement.

Partie 3. Modélisation

Modélisation non supervisée

K-means

Nous avons décidé d'effectuer sur les variables de cette étude un algorithme de machine learning non supervisé très populaire: celui des K-means.Il s'agit d'une technique de classification dont le but est de regrouper des ensembles de données similaires en K clusters(groupes) par exemple.

L'un de ses nombreux avantages est que les résultats en plus d'être robustes, sont très facile à interpréter. Le K-means fonctionne par apprentissage automatique en affectant à chaque observation le cluster ayant la plus petite distance par rapport au centre de classe jusqu'à ce que l'algorithme se stabilise au sein de chaque cluster. Toutefois il faut noter que le K-means à la particularité d'imposer à l'utilisateur de spécifier initialement le nombre de cluster à prendre en compte. Nous avons donc décidé dans notre cas d'application d'effectuer préalablement une détection du nombre optimal de classes à partir d'une évaluation de la proportion d'inertie expliquée.

Le graphique d'inertie ci-dessous nous permet donc d'identifier 3 clusters que nous pourrons utiliser par la suite pour définir la valeur de K dans notre algorithme de K-means. il s'agit en effet, d'une recommandation du meilleur découpage possible...

Ensuite nous effectuons les K-means (K=3) sur une base de données ne comportant pas toutes les variables de cette étude mais plutôt certaines que nous avons ciblé afin de respecter cette contrainte qu'à la méthode de ne s'exécuter que sur des variables quantitatives et pour des raisons de pertinence.

Nous utilisons donc entre autres, par exemple les variables catégorielles gender et resident_status transformées en binaires dans cette analyse de K-means.

```
inertie.expl<-rep(0,times=15)
for(i in 2:15){
  km<-kmeans(dataModel, centers=i,iter.max=100, nstart=10)
  inertie.expl[i]<-km$betweenss/km$totss
}
plot(1:15, inertie.expl, type = "b", xlab = "Nombre de groupes",
     ylab = "pourcantage inertie expliquée", lwd=1.5)


km<-kmeans(dataModel,3,iter.max=100,nstart=10)
km
```

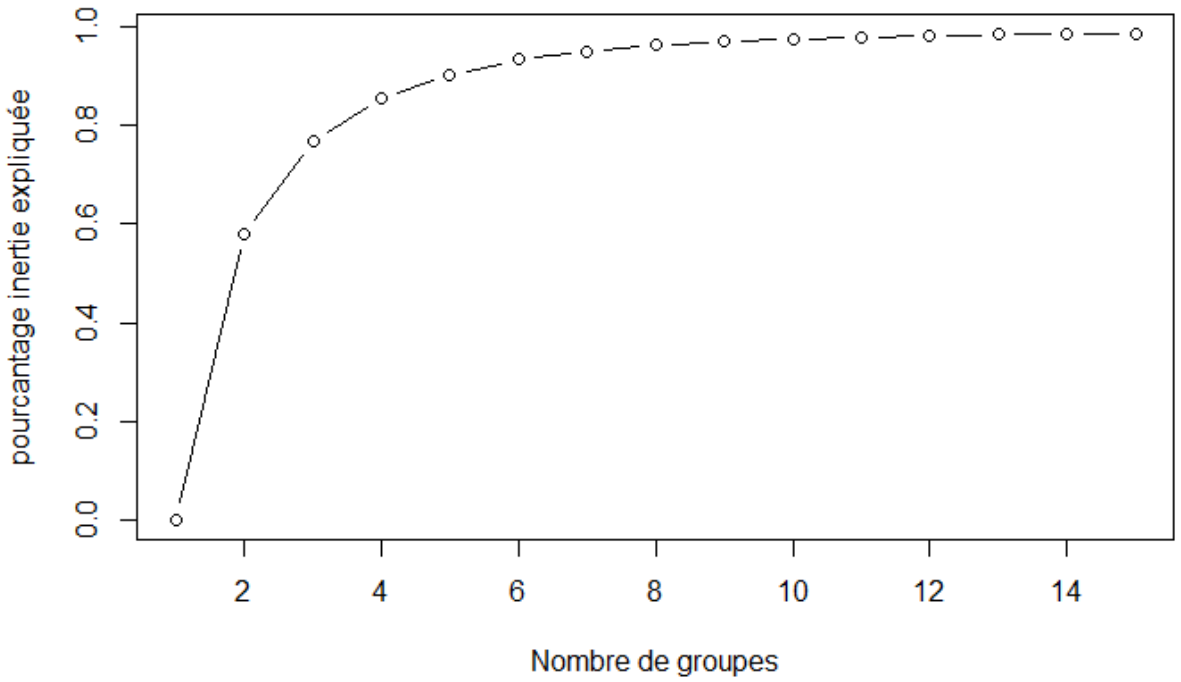


Figure 1: Choix nombre groupes

les résultats du K-means que nous avons obtenu K-means clustering with 3 clusters of sizes 1397, 267, 1736

Cluster means: medical_history_1 medical_history_2 medical_history_3 medical_history_4 medical_history_5 1 0.22906228 0.3099499 0.1467430 0.04581246 0.06012885 2 0.32584270 0.3520599 0.1310861 0.05617978 0.10861423 3 0.09677419 0.3058756 0.1284562 0.05645161 0.06221198 medical_history_6 medical_history_7 preop_medication_1 preop_medication_2 preop_medication_3 1 0.2956335 0.2534001 0.5146743 0.6041518 0.8246242 2 0.3820225 0.3183521 0.4943820 0.5955056 0.8164794 3 0.2021889 0.2453917 0.4965438 0.5800691 0.8185484 preop_medication_4 preop_medication_5 preop_medication_6 symptom_1 symptom_2 symptom_3 symptom_4 1 0.5218325 0.8017180 0.7494631 0.6657122 0.7086614 0.6241947 0.7809592 2 0.5280899 0.8576779 0.7565543 0.7340824 0.7977528 0.6629213 0.7940075 3 0.5236175 0.8283410 0.7379032 0.5650922 0.6042627 0.4625576 0.6722350 symptom_5 lab_result_1 lab_result_2 lab_result_3 weight height gender resident_status 1 0.7752326 14.46063 27.45719 99.44452 79.94503 165.2598 0.4681460 0.9563350 2 0.8913858 14.47154 27.39738 99.29213 82.76517 165.5281 0.4419476 0.6666667 3 0.2707373 14.47707 27.41613 99.55645 77.16774 164.8750 0.5357143 0.9936636 treatPeriode age sumAmount 1 11.03579 55.59127 26401.71 2 11.11985 60.67041 45450.29 3 11.05300 47.93433 14575.13

Clustering vector: [1] 3 1 3 3 2 1 1 1 1 3 1 1 1 3 3 3 1 3 2 3 1 1 2 1 1 3 1 1 1 3 1 1 1 1 2 2 3 1 3 1 1 3 1 3 3 1 [47] 3 1 3 1 1 1 3 3 1 3 1 1 1 3 1 1 2 1 1 2 3 1 3 1 3 3 3 3 3 1 3 1 1 1 3 3 1 1 3 3 2 2 3 3 1 3 [93] 1 3 1 1 1 1 3 3 3 1 3 3 1 3 3 3 3 3 3 3 1 3 3 1 3 3 1 1 2 3 1 1 3 1 1 1 1 1 3 1 1 3 3 3 3 1 [139] 2 3 1 1 3 1 3 3 1 2 3 1 1 3 3 3 3 3 1 3 3 1 1 1 1 3 3 3 [185] 3 3 3 1 3 3 3 3 3 1 1 2 2 1 1 3 3 3 3 1 3 3 1 1 2 1 3 3 1 3 1 1 3 3 1 1 3 1 3 1 2 2 3 3 3 3 [231] 2 1 3 1 2 1 3 1 3 3 1 3 3 1 3 3 3 2 1 1 3 3 1 1 3 3 3 1 3 3 1 3 3 1 3 3 1 3 1 3 1 1 3 3 1 3 [277] 3 3 3 1 3 1 1 3 1 1 3 1 1 3 3 2 3 1 1 3 2 1 3 3 3 1 3 1 1 3 3 1 1 3 2 3 1 3 1 1 1 2 3 1 3 3 [323] 3 3 3 3 3 1 1 3 1 3 1 1 1 3 3 3 1 3 1 3 3 3 3 1 3 3 2 3 3 2 1 1 3 3 3 1 2 1 3 1 1 1 3 1 1 1 [369] 3 3 3 3 1 1 3 1 3 3 3 3 3 1 3 1 3 3 1 3 3 1 1 1 3 3 3 1 3 3 3 3 3 2 1 2 1 3 3 3 3 2 1 3 1 1 [415] 3 3 1 3 3 1 1 1 1 3 3 3 1 3 1 2 1 1 3 3 1 3 1 3 1 3 1 3 1 2 3 3 3 3 1 1 1 1 1 1 1 3 1 3 3 3 [461] 3 3 3 1 1 3 1 3 3 1 3 1 3 3 3 3 1 3 1 2 1 3 1 3 2 3 3 1 1 3 3 3 1 2 1 3 1 3 1 1 3 1 3 3 1 3 [507] 3 3 3 1 1 3 1 3 3 3 1 1 3 1 1 3 1 3 3 1 1 3 3 1 3 1 1 1 3 3 3 1 3 3

1 1 3 1 1 3 1 3 1 3 2 1 3 3 3 3 [553] 1 3 3 3 3 3 3 2 3 1 1 2 2 1 3 3 3 2 3 1 3 3 3 1 1 3 3 3 1 1 3 1 3 3 3 1 1 1 3 3 3 3 1 1 1 1 [599] 1 3 2 3 1 3 1 1 1 3 1 3 3 2 2 2 3 3 3 2 1 1 3 3 2 1 1 1 1 3 1 1 1 1 3 3 1 3 3 1 1 1 1 1 3 1 [645] 1 1 1 1 3 3 1 1 3 3 1 3 3 2 3 3 1 1 1 1 3 3 3 1 3 1 3 1 1 1 3 3 3 3 1 3 3 3 1 1 3 3 2 3 1 3 1 [691] 3 1 1 3 3 1 1 3 1 1 3 3 1 3 1 1 1 3 3 3 1 1 2 3 3 3 3 3 3 3 1 1 3 3 3 3 3 3 2 3 2 3 1 3 2 1 3 [737] 1 3 3 1 3 3 1 3 3 3 3 3 3 3 2 2 3 2 3 3 1 2 2 3 1 1 1 3 3 1 1 3 3 3 3 1 3 3 3 2 2 1 1 3 3 3 3 [783] 1 3 3 1 1 1 3 1 1 1 3 3 1 3 1 3 1 3 3 3 1 2 3 1 1 3 3 3 2 2 3 1 1 1 3 3 3 3 3 2 2 1 2 3 3 3 1 [829] 3 3 3 3 1 3 1 1 3 1 3 2 1 2 3 1 1 3 3 1 1 1 3 2 1 3 1 1 3 1 3 2 1 1 1 3 1 3 1 3 3 3 3 3 2 [875] 3 3 3 3 2 2 3 3 3 3 3 3 1 1 3 1 1 3 3 3 3 1 1 3 3 1 1 3 3 1 1 1 1 3 1 1 1 1 3 1 1 3 3 3 3 3 [921] 2 1 2 3 3 1 3 3 1 1 3 2 1 3 3 1 1 3 1 1 2 1 1 1 1 1 1 3 1 3 3 3 3 1 1 3 3 3 3 3 3 3 3 1 3 3 [967] 2 3 1 3 1 3 1 1 1 3 3 3 1 1 3 1 3 3 1 1 1 1 3 1 3 3 1 1 1 1 1 1 3 1 [reached getOption("max.print") – omitted 2400 entries]

Within cluster sum of squares by cluster: [1] 23543236583 33548981022 23921242076 (between_SS / total_SS = 76.9 %)

Available components:

[1] "cluster" "centers" "totss" "withinss" "tot.withinss" "betweenss"
[7] "size" "iter" "ifault"

Il ressort de ces résultats de K-means que le groupe 1 qui contient 1397 observations représente celui des patients ayant un montant total de facture (variable sumAmout) relativement moyen. Les individus de ce cluster ont particulièrement une preop_medication 1,2 et 3 élevées par rapport aux autres clusters. Le groupe 2 bien qu'il contient moins d'observations (267) par rapport aux autres groupes, est celui qui regroupe les montants de factures les plus élevés.

Aussi pour la majorité des variables considérées (exemple: medical_history_1, symptom_5, 82.76517, age, treatperiode), c'est dans ce groupe qu'on observe les plus grandes valeurs prises. Le cluster 3 qui contient quant à lui 1736 observations est celui qui correspond aux plus faibles montant de facture par rapport aux autres clusters. Dans ce groupe, les variables tel que lab_result_3, lab_result_1, medical_history_4 ont plus de poids en terme de valeurs prises.

Finalement, les variables qui ont contribué le plus dans les 3 classes avec les moyennes assez éloignées entre les classes et permettent de les bien distenguer sont medical_history 1 et 6, les symptômes 1-5, resident_status, âge et le coût de traitement (sumAmount). ## Modélisation supervisée

Régression lineaire

```
reg_lin=lm(sumAmount~.,data = dataModelLog)
```

```
# Resultats de la regression
print(summary(reg_lin))
```

```
##
## Call:
## lm(formula = sumAmount ~ ., data = dataModelLog)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.72323 -0.13820 -0.07286  0.10508  0.58456
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          8.6959665  0.1383430  62.858  < 2e-16 ***
## medical_history_1    0.2756801  0.0092988  29.647  < 2e-16 ***
## medical_history_2    0.0207398  0.0075261   2.756  0.00589 **
```

```
## medical_history_3   0.0460305  0.0101560   4.532 6.04e-06 ***
## medical_history_4   0.0003008  0.0157051   0.019  0.98472
## medical_history_5   0.0562049  0.0141373   3.976 7.17e-05 ***
## medical_history_6   0.1733725  0.0080014  21.668  < 2e-16 ***
## medical_history_7   0.0395378  0.0080006   4.942 8.11e-07 ***
## preop_medication_1  0.0128040  0.0069703   1.837  0.06631 .
## preop_medication_2  0.0224117  0.0070931   3.160  0.00159 **
## preop_medication_3  0.0237311  0.0090853   2.612  0.00904 **
## preop_medication_4  0.0138731  0.0069870   1.986  0.04716 *
## preop_medication_5  0.0128832  0.0090687   1.421  0.15552
## preop_medication_6  0.0256421  0.0079830   3.212  0.00133 **
## symptom_1           0.1323343  0.0071705  18.455  < 2e-16 ***
## symptom_2           0.1890644  0.0073784  25.624  < 2e-16 ***
## symptom_3           0.2036772  0.0070103  29.054  < 2e-16 ***
## symptom_4           0.1747027  0.0078182  22.346  < 2e-16 ***
## symptom_5           0.5116291  0.0069684  73.422  < 2e-16 ***
## lab_result_1       -0.0001746  0.0019927  -0.088  0.93020
## lab_result_2        0.0018799  0.0014135   1.330  0.18362
## lab_result_3        0.0001234  0.0002284   0.540  0.58923
## weight              0.0066392  0.0003433  19.339  < 2e-16 ***
## height             -0.0011503  0.0007090  -1.622  0.10481
## gender              0.0106767  0.0103261   1.034  0.30124
## resident_status    -0.6550165  0.0164446 -39.832  < 2e-16 ***
## treatPeriode        0.0002431  0.0012241   0.199  0.84261
## age                 0.0098318  0.0002371  41.464  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2024 on 3372 degrees of freedom
## Multiple R-squared:  0.7963, Adjusted R-squared:  0.7947
## F-statistic: 488.4 on 27 and 3372 DF,  p-value: < 2.2e-16
```

```
lm <- predict(reg_lin,newdata = dataModelLog)
```

```
rmse<-function(pred,actual){
  sqrt(mean((pred-actual)^2))}
rmse(lm,dataModelLog$sumAmount)
```

```
## [1] 0.2015808
```

Nous avons décidé dans un premier temps de créer un modèle de régression linéaire multiple en utilisant toutes les variables explicatives à notre disposition, le coefficient de corrélation obtenu est assez bon ,R-squared vaut 0.7963, ce qui signifie que le modèle explique 79% de la variabilité des données , d'autre part l'erreur quadratique moyenne vaut 0.2015808 ce qui signifie qu'on peut prédire le montant de la facture avec une erreur de plus ou moins 0.2 unités de logarithme de coût de traitement de patient.

Les coefficients les plus significatifs du modèle sont l'âge, le statut de résidence, le poids, les 5 symptômes , les traitements peropératoires 2,3 et 6 et enfin les histoires médicales 1,3 ,5 ,6 et 7.

```
reg_lin2=lm(sumAmount~medical_history_1+medical_history_7+
              symptom_1+symptom_2+symptom_3+symptom_4+symptom_5+
              weight+age+resident_status,data = dataModelLog)
```

```r
# Resultats de la regression
print(summary(reg_lin2))
```

```
##
## Call:
## lm(formula = sumAmount ~ medical_history_1 + medical_history_7 +
##     symptom_1 + symptom_2 + symptom_3 + symptom_4 + symptom_5 +
##     weight + age + resident_status, data = dataModelLog)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.78479 -0.16363 -0.04545  0.14830  0.69859
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        8.7357443  0.0367834 237.492  < 2e-16 ***
## medical_history_1  0.2740597  0.0100004  27.405  < 2e-16 ***
## medical_history_7  0.0406638  0.0085922   4.733 2.31e-06 ***
## symptom_1          0.1356329  0.0077075  17.598  < 2e-16 ***
## symptom_2          0.1899268  0.0079255  23.964  < 2e-16 ***
## symptom_3          0.1992049  0.0075272  26.465  < 2e-16 ***
## symptom_4          0.1730848  0.0083980  20.610  < 2e-16 ***
## symptom_5          0.5155209  0.0074933  68.798  < 2e-16 ***
## weight             0.0063647  0.0003411  18.661  < 2e-16 ***
## age                0.0096516  0.0002549  37.865  < 2e-16 ***
## resident_status   -0.6484694  0.0176486 -36.743  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.218 on 3389 degrees of freedom
## Multiple R-squared:  0.7625, Adjusted R-squared:  0.7618
## F-statistic:  1088 on 10 and 3389 DF,  p-value: < 2.2e-16
```

```r
lm <- predict(reg_lin2,newdata = dataModelLog)
```

```r
rmse<-function(pred,actual){
  sqrt(mean((pred-actual)^2))}
rmse(lm,dataModelLog$sumAmount)
```

```
## [1] 0.2176813
```

Par la suite à l'aide du graphique des corrélations nous avons choisis les variables fortement corrélées avec notre variable cible à qui nous avons appliquées le modèle de la régression linéaire multiple, le R-carré (76,25%) est proche à celui obtenu dans le premier modèle mais il est moins élevé, de même pour que le RMSE qui est légèrement plus grand (RMSE=0.2176813).Tous les coefficients choisis sont bien significatifs.

Random Forest

Nous avons choisi l'algorithme Random Forest pour prédire le coût de traitement pour plusieurs raisons. En effet, Le Random Forest est un cas particulier du Bagging appliqué aux arbres de décision. Le Random Forest construit de nombreux petits arbres sur une fraction aléatoire de données. De manière générale, cette

méthode évite l'overfitting grâce aux échantillons "Out of bag". Pour chaque arbre à construire le Random Forest sélectionne un sous-échantillon par bootstrap d'individus et à chaque étape, la construction d'un noeud de l'arbre se fait sur un sous-ensemble tiré aléatoirement. Cette méthode est aussi efficace sur les données de grande dimension.

Pour tous ces raisons nous avons choisis le Random Forest. Ensuite nous allons utiliser cette méthode pour faire de la régression et prédire une variable continue qui est log(sumAmount) donc logarithme de la variable cible le coût de traitement. Et nous allons aussi faire du Random Forest Classifications en divisant le coût de traitement en 4 classes de 4 quantiles.

Random Forest Regression

Nous divisons notre base contenant le logarithme de la variable à prédire en train avec 70% des observations et test avec 30% des observations et faisons l'entrainement du modèle en construisant 30 arbres de décision.

```
trainData=sample(1:nrow(dataModelLog),0.7*nrow(dataModelLog))
train=dataModelLog[trainData,]
test=dataModelLog[-trainData,]
set.seed(42)
model.rf <- randomForest(sumAmount ~ ., data = train, ntree = 30)
model.rf
```

Les résultats lors de notre exécutions sont les suivantes: Call: randomForest(formula = sumAmount ~ ., data = train, ntree = 30) Type of random forest: regression Number of trees: 30 No. of variables tried at each split: 9

```
      Mean of squared residuals: 0.05401157
                % Var explained: 72.73
```

```
#prédiction
p.rf <- predict(model.rf,newdata = test)
```

Metrics

```
rmse<-function(pred,actual){
  sqrt(mean((pred-actual)^2))
}
rmse(p.rf,test$sumAmount)
```

RMSE = 0.227127

Nous constatons que le modèle a une variance expliqué à 72.73% ce qui est un assez bon résultat avec le RMSE (Mean Squared Error) qui est de 0.227127. Les R2 et le RMSE observée avec Random Forest sont moins performantes qu'avec les 2 modèles de la régression linéaire exécutés précédemment. Finalement, parmi les modèles de régression choisis la régression linéaire multiple qui utilise toutes les variables disponibles est un favoris.

Affichons ensuite les variables importantes du modèle.

```
#liste des variables importantes
varImpPlot(model.rf)
```

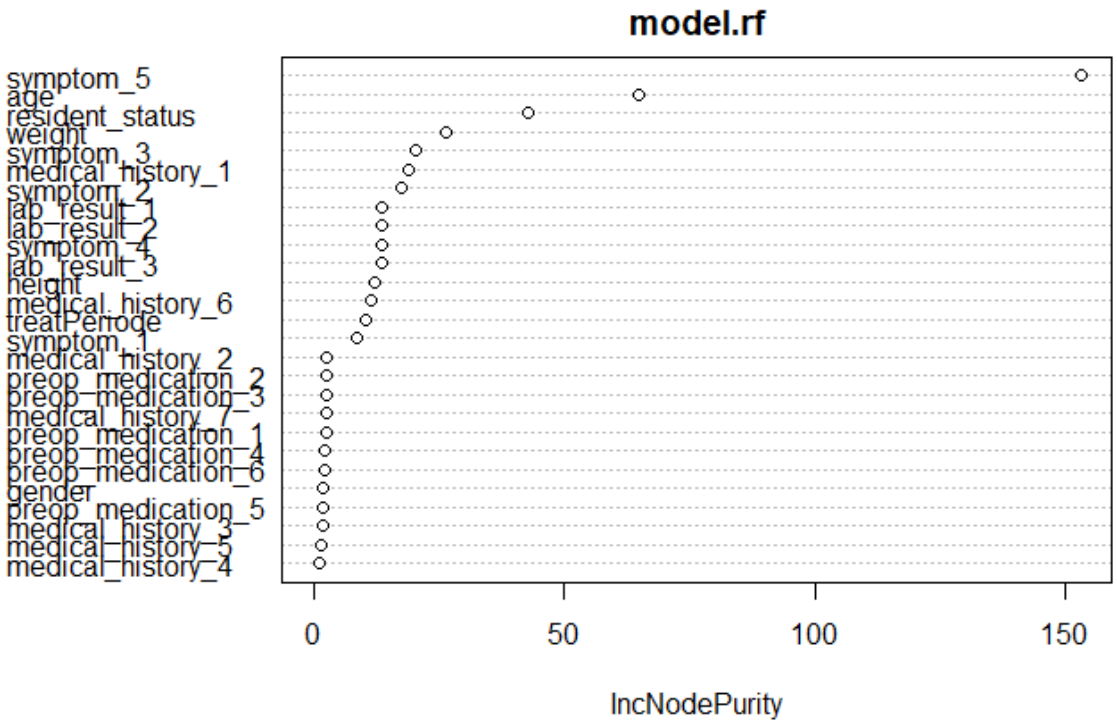


Figure 2: Les variables importantes

Le graphique ci-dessus montre les variables les plus importants dans la prédiction de la variable sumAmount. Nous constatons que la variable la plus important est le symptôme 5. Nous ne connaissons pas quel symptôme est caché sous le nombre 5, mais le fait devoir ce symptôme va influencer significativement le coût de traitement. Probablement, ce symptôme nécessite un traitement plus complexe et plus côuteux ce qui pourrait expliquer l'importance de la variable. Une variable un peu moins significative que la précédente est notre variable calculée indiquant l'âge du patient. De manière générale les personnes plus âgées ont des maladies plus souvent et plus graves et comme conséquence leur traitement devrait être plus cher.

Les variables qui sont aussi importantes mais un peu moins que les 2 précédentes sont resident_status, weight, medical_history_1, symptom_3, lab_result_2, symptom_2, lab_result_3, height, lab_result_1, symptom_4, medical_history_6, symptom_1 et une autre variable calculée treatPEriode. Donc, les variables les plus importantes sont tous les symptômes, certains résultats laboratoires, 2 histoires médicales, et certaines caractéristiques personnelles des patients.

Random Forest. Classification

Nous affichons le summary pour déterminer les bornes des classes pour ensuite diviser notre target en 4 classes bornés par les quantiles.

```
print(summary(dataModel$sumAmount))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2946   14793   20180   21859   26680   98724
```

```
dataModelC<-dataModel
dataModelC$target=cut(dataModel$sumAmount, c(0,14793, 20180, 26680, Inf), labels = FALSE)
dataModelC$sumAmount=NULL
dataModelC$target=as.factor(dataModelC$target)
```

Nous avons déterminé donc 4 classes : * classe 1 [0,14793] le coût de traitement bas * classe 2 [14793,20180] le coût de traitement faible * classe 3 [20180,26680] le coût de traitement élevé * classe 4 [26680,Inf] le coût de traitement très élevé

```
trainData=sample(1:nrow(dataModelC),0.7*nrow(dataModelC))
train=dataModelC[trainData,]
test=dataModelC[-trainData,]
set.seed(42)
model.rf <- randomForest(target ~ ., data = train, ntree = 30)
```

Affichons ensuite les variables importantes du modèle.

```
#liste des variables importantes
varImpPlot(model.rf)
```

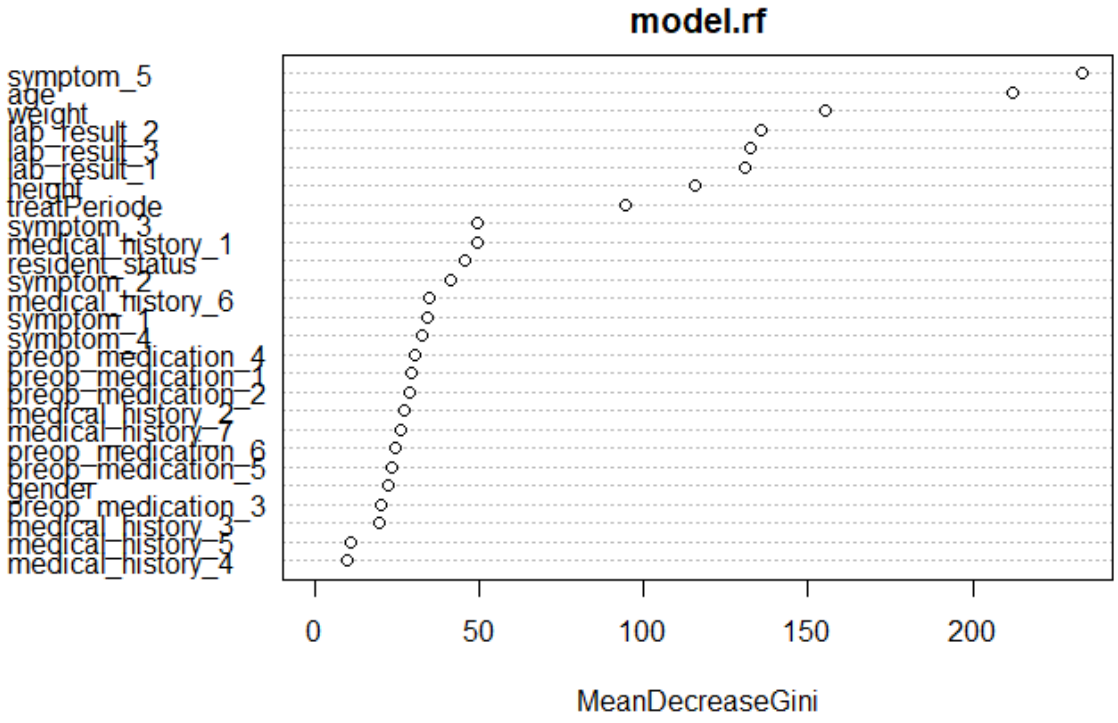


Figure 3: Les variables importantes

Dans le cas de la Classification, l'ordre d'importance des variables a changé, mais les variables importantes sont toujours les mêmes que dans le cas de la régression. A cette étape la variable la plus importante est le symptôme 5 comme dans le cas précédant. Ensuite l'âge. Ensuite, les plus importantes sont lab_result_2, weight, lab_result_1 et 3, height et treatPeriode. Les autres variables sont beaucoup moins importantes.

Ensuite, nous faisons de la prédiction pour afficher la matrice de confusion pour analyser les résultats du modèle.

```
#prédiction
p.rf <- predict(model.rf,newdata = test,type="class")
summary(p.rf)
#matrice de confusion
caret::confusionMatrix(p.rf,test$target)
```

Les résultats obtenus lors de notre execution de code 1 2 3 4 293 235 259 233 Confusion Matrix and Statistics

```
      Reference
```

Prediction 1 2 3 4 1 207 70 14 2 2 41 113 67 14 3 8 77 92 82 4 3 12 66 152

Overall Statistics

```
           Accuracy : 0.5529
             95% CI : (0.5218, 0.5838)
No Information Rate : 0.2667
P-Value [Acc > NIR] : < 2e-16

              Kappa : 0.404
```

Mcnemar's Test P-Value : 0.06217

Statistics by Class:

```
                     Class: 1 Class: 2 Class: 3 Class: 4
```

Sensitivity 0.7992 0.4154 0.3849 0.6080 Specificity 0.8870 0.8369 0.7862 0.8948 Pos Pred Value 0.7065 0.4809 0.3552 0.6524 Neg Pred Value 0.9285 0.7975 0.8068 0.8755 Prevalence 0.2539 0.2667 0.2343 0.2451 Detection Rate 0.2029 0.1108 0.0902 0.1490 Detection Prevalence 0.2873 0.2304 0.2539 0.2284 Balanced Accuracy 0.8431 0.6262 0.5856 0.7514

Nous constatons que le modèle a attribué 293 observations dans la classe 1 du coût de traitement bas, 235 dans le class 2 avec coût faible, 259 dans la classe 3 avec le coût élevé et 233 dans le classe 4 avec le coût très élevé. En observant la matrice de confusion, les résultats de la prédiction ne sont pas bons. L'accuracy est seulement de 55,29%, ce qui rend ce modèle le moins performant entre tous les modèles choisis et assez moins performant par rapport à son équivalent Random Forest Regression.

Sensitivity est un taux des vrais positives (rappels de la classe positive). La sensibilité est calculée comme le nombre de prédictions positives correctes divisé par le nombre total de positifs. La meilleure sensibilité est de 1,0, tandis que la pire est de 0,0. Dans notre cas seules les classes 1 et 4 ont des valeurs de sensibilité assez élevés. Notre modèle réussi assez bien à prédire ces 2 classes. Le fait de la mauvaise reconnaissance des classes 3 et 4 pourrait être dû à une répartition inégalitaire des classes dans le jeu de données test. Cependant en affichant le summary de la target de train nous constatons que les classes sont égalitaires. Donc, ce n'est pas le souci.

La spécificité (Specificity) est calculée comme le nombre de prédictions négatives correctes divisé par le nombre total de négatifs. Il est également appelé taux des vrais négatifs. La meilleure spécificité est de 1,0, tandis que la pire est de 0,0. Le taux est assez élevé dans le cas de chacune des 4 classes. Ici le modèle prédit assez bien, les négatifs. Autrement dit, il a une bonne capacité à reconnaitre qu'un patient n'appartient pas à une catégorie.

Cependant, la valeur obtenu pour l'accuracy montre que le modèle n'est pas performant. Pour améliorer les résultats le rééchantillonage d'une autre manière pourrait aider. Nous considérons que les variables

considérées pour cette analyse ne sont pas suffisantes pour bien discriminer les différentes classes de montant de facture qui peuvent ressortir de la population étudiée. Il aurait fallut des variables qui réflètent des caractéristiques plus spécifiques et plus précises permettant ainsi de bien distinguer les observations.

```
summary(train$target)
```

```
##   1   2   3   4
## 592 600 574 614
```

Conclusion

Après avoir étudié les données qui proviennent du Singapour et qui concernent le traitement, les caractéristiques personnelles des patients et le montant de leurs factures nous avons procédé par une modélisation non supervisée à l'aide du modèle K-means pour essayer déterminer s'il y a des classes spécifiques dans notre dataset des patients de Singapour. Nous avons constaté que les 3 classes déterminées n'ont pas des caractéristiques trop spécifiques. Les moyennes des classes sont assez proches entre eux. Par contre il y a certaines variables qui ont contribué le plus dans les 3 classes avec les moyennes assez éloignées entre les classes. Ce sont medical_history 1 et 6, les symptômes 1-5, resident_status, âge et le coût de traitement (sumAmount).

Ensuite nous avons fait de la modélisation supervisée. Premièrement, nous avons fait la régression multiple sur tous les variables disponibles dans notre data set et une autre régression linéaire avec seulement les variables que sont significativement corrélés avec notre variable cible selon notre carte de chaleur. Nous avons décidé de procéder par un logarithme de la variable cible à cause du fait de la présence de plusieurs valeurs aberrantes et pour ne pas les supprimer. Pour estimer l'amélioration de la performance de la régression multiple nous avons estimé le modèle sur le coût de traitement avant de passer en logarithme. Nous avons constaté qu'avec logarithme nous avons pu accroître la performance de la régression complète en passant de R2=71% à R2=79,56%. Le RMSE obtenu finalement est de 0.2015808. Nous pouvons donc prédire avec certitude log(sumAmount) + - 0.2015808. La régression multiple avec un nombre des variables réduit a aussi une bonne performance mais moindre avec R2 de 76% et RMSE 0.2176813. De plus, comme il était attendu, nous avons constaté que les variables corrélées significativement avec variable cible sont bien tous significativement importantes dans les modèles.

Le modèle suivant choisi est le Random Forest Regression qui a démontré aussi une bonne performance avec la variance expliquée de 72.73 % et avec RMSE de 0.227127. Les résultats sont moins performants qu'avec la régression multiple, mais ils ne sont pas trop éloignés. Finalement, le leader entre les modèles de régression est la régression multiple complète.

Comme un dernier modèle nous avons décidé d'essayer une classification avec Random Forest. Cependant, les résultats n'ont pas réussi à répondre aux attentes. L'accuracy était seulement de 55.29%. Ce manque de performance n'est pas lié aux inégalités des tailles des classes dans le train set. Nous avons bien vu que les observations sont réparties d'une manière égalitaire. Le problème pourrait être dû à un manque d'informations, probablement à un manque des variables plus spécifiques qui pourraient mieux distinguer les classes. Il est difficile de déterminer les variables nécessaires à rajouter pour améliorer la performance pas seulement de la classification, mais aussi de la régression (ces résultats ne sont pas les meilleurs possibles) à cause du fait des variables cachés dans notre base de données. En effet, nous ne pouvons pas comprendre qu'est-ce que c'est une histoire médicale ou quels symptômes ont eu les patients ou quelles données nous possédons exactement. Mais, il pourrait être utile d'avoir les informations si le symptôme était dans sa forme grave ou légère. Si la forme est grave, alors le traitement de symptôme devrait être plus complexe et comme conséquence plus coûteux. Cependant, nous possédons comme seulement information si le patient a eu un symptôme particulier ou non.

De plus, nous ne connaissons pas si les données sont issues d'un seul hôpital ou de plusieurs. Dans différents hôpitaux les prix des procédures de traitements sont différentes ce qui peux biaiser les résultats si la variation des prix est significative.

Finalement, nous avons pu constater que les modèles de régression conviennent plus dans notre cas afin de prédire le coût de traitement.

Cependant, il est nécessaire de rajouter des informations supplémentaires afin de faire une meilleure prédiction.

Annexe

Premier audit de la base clinicalDATA.csv

```
## [1] 3400   26
```

```
##                              patient_id    date_of_admission  date_of_discharge
##  4e46fddfa404b306809c350aecbf0f6a:   4   2014-05-24:   9    2011-10-23:   9
##  0eacfb2daed1f3ba2adf32e293bc05a6:   3   2011-01-18:   7    2011-08-05:   7
##  258807316af4b45fda1b05668d557d06:   3   2012-06-26:   7    2011-10-09:   7
##  5e9e8508e8098fc220a12db23c698ec6:   3   2012-08-19:   7    2012-02-27:   7
##  64a531972193ccf232cc47597ddb85ed:   3   2012-10-14:   7    2012-03-11:   7
##  6a6991ecb85a2f82f77b09eb0eeb747b:   3   2012-10-22:   7    2012-06-04:   7
##  (Other)                          :3381   (Other)   :3356    (Other)   :3356
##  medical_history_1 medical_history_2 medical_history_3 medical_history_4
##  Min.   :0.0000    Min.   :0.0000    0  :2176          Min.   :0.00000
##  1st Qu.:0.0000    1st Qu.:0.0000    1  : 348          1st Qu.:0.00000
##  Median :0.0000    Median :0.0000    No : 761          Median :0.00000
##  Mean   :0.1691    Mean   :0.3113    Yes: 115          Mean   :0.05206
##  3rd Qu.:0.0000    3rd Qu.:1.0000                      3rd Qu.:0.00000
##  Max.   :1.0000    Max.   :1.0000                      Max.   :1.00000
##                    NA's   :233
##  medical_history_5 medical_history_6 medical_history_7 preop_medication_1
##  Min.   :0.00000   Min.   :0.0000    Min.   :0.0000    Min.   :0.0000
##  1st Qu.:0.00000   1st Qu.:0.0000    1st Qu.:0.0000    1st Qu.:0.0000
##  Median :0.00000   Median :0.0000    Median :0.0000    Median :1.0000
##  Mean   :0.06395   Mean   :0.2547    Mean   :0.2544    Mean   :0.5038
##  3rd Qu.:0.00000   3rd Qu.:1.0000    3rd Qu.:1.0000    3rd Qu.:1.0000
##  Max.   :1.00000   Max.   :1.0000    Max.   :1.0000    Max.   :1.0000
##  NA's   :304
##  preop_medication_2 preop_medication_3 preop_medication_4 preop_medication_5
##  Min.   :0.0000     Min.   :0.0000     Min.   :0.0000     Min.   :0.0000
##  1st Qu.:0.0000     1st Qu.:1.0000     1st Qu.:0.0000     1st Qu.:1.0000
##  Median :1.0000     Median :1.0000     Median :1.0000     Median :1.0000
##  Mean   :0.5912     Mean   :0.8209     Mean   :0.5232     Mean   :0.8197
##  3rd Qu.:1.0000     3rd Qu.:1.0000     3rd Qu.:1.0000     3rd Qu.:1.0000
##  Max.   :1.0000     Max.   :1.0000     Max.   :1.0000     Max.   :1.0000
##
##  preop_medication_6   symptom_1        symptom_2        symptom_3
##  Min.   :0.0000     Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000     Median :1.0000   Median :1.0000   Median :1.0000
##  Mean   :0.7441     Mean   :0.6197   Mean   :0.6624   Mean   :0.5447
##  3rd Qu.:1.0000     3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000     Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##
##    symptom_4        symptom_5       lab_result_1    lab_result_2
##  Min.   :0.0000   Min.   :0.0000   Min.   : 9.10   Min.   :19.70
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:13.20   1st Qu.:25.80
##  Median :1.0000   Median :1.0000   Median :14.50   Median :27.40
##  Mean   :0.7265   Mean   :0.5268   Mean   :14.47   Mean   :27.43
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:15.80   3rd Qu.:29.10
##  Max.   :1.0000   Max.   :1.0000   Max.   :20.30   Max.   :35.10
##
```

```
##   lab_result_3        weight           height
##  Min.   : 52.00   Min.   : 48.00   Min.   :151.0
##  1st Qu.: 88.00   1st Qu.: 71.20   1st Qu.:160.0
##  Median :100.00   Median : 78.90   Median :165.0
##  Mean   : 99.49   Mean   : 78.75   Mean   :165.1
##  3rd Qu.:110.00   3rd Qu.: 86.30   3rd Qu.:170.0
##  Max.   :150.00   Max.   :121.00   Max.   :186.0
##
```

```
##         patient_id  date_of_admission  date_of_discharge  medical_history_1
##           "factor"           "factor"           "factor"          "integer"
##  medical_history_2  medical_history_3  medical_history_4  medical_history_5
##          "numeric"           "factor"          "integer"          "numeric"
##  medical_history_6  medical_history_7 preop_medication_1 preop_medication_2
##          "integer"          "integer"          "integer"          "integer"
## preop_medication_3 preop_medication_4 preop_medication_5 preop_medication_6
##          "integer"          "integer"          "integer"          "integer"
##          symptom_1          symptom_2          symptom_3          symptom_4
##          "integer"          "integer"          "integer"          "integer"
##          symptom_5       lab_result_1       lab_result_2       lab_result_3
##          "integer"          "numeric"          "numeric"          "numeric"
##             weight             height
##          "numeric"          "numeric"
```

```
##         patient_id  date_of_admission  date_of_discharge  medical_history_1
##                  0                  0                  0                  0
##  medical_history_2  medical_history_3  medical_history_4  medical_history_5
##                233                  0                  0                304
##  medical_history_6  medical_history_7 preop_medication_1 preop_medication_2
##                  0                  0                  0                  0
## preop_medication_3 preop_medication_4 preop_medication_5 preop_medication_6
##                  0                  0                  0                  0
##          symptom_1          symptom_2          symptom_3          symptom_4
##                  0                  0                  0                  0
##          symptom_5       lab_result_1       lab_result_2       lab_result_3
##                  0                  0                  0                  0
##             weight             height
##                  0                  0
```

Premier audit de la base bill_id.csv

```
dim(billID) #afficher la dimension de la base
```

```
## [1] 13600     3
```

```
summary(billID) #afficher les statistiques descriptives de la base de donnée
```

```
##     bill_id                                 patient_id
##  Min.   :5.584e+05   4e46fddfa404b306809c350aecbf0f6a:   16
##  1st Qu.:2.486e+09   0eacfb2daed1f3ba2adf32e293bc05a6:   12
##  Median :4.993e+09   258807316af4b45fda1b05668d557d06:   12
##  Mean   :5.007e+09   5e9e8508e8098fc220a12db23c698ec6:   12
```

```
##  3rd Qu.:7.525e+09   64a531972193ccf232cc47597ddb85ed:   12
##  Max.   :1.000e+10   6a6991ecb85a2f82f77b09eb0eeb747b:   12
##                      (Other)                          :13524
##   date_of_admission
##  2014-05-24:   36
##  2011-01-18:   28
##  2012-06-26:   28
##  2012-08-19:   28
##  2012-10-14:   28
##  2012-10-22:   28
##  (Other)   :13424
```

```
sapply(billID, class) #afficher les types de chaque variable
```

```
##           bill_id        patient_id date_of_admission
##         "numeric"          "factor"          "factor"
```

```
colSums(is.na(billID)) #verifier le nombre des valeurs manquantes
```

```
##           bill_id        patient_id date_of_admission
##                 0                 0                 0
```

Premier audit de la base bill_amount.csv

```
dim(billAmount) #afficher la dimension de la base
```

```
## [1] 13600     2
```

```
summary(billAmount) #afficher les statistiques descriptives de la base de donnée
```

```
##     bill_id              amount
##  Min.   :5.584e+05   Min.   :   79.5
##  1st Qu.:2.486e+09   1st Qu.:  950.7
##  Median :4.993e+09   Median : 1517.0
##  Mean   :5.007e+09   Mean   : 5464.8
##  3rd Qu.:7.525e+09   3rd Qu.: 7307.1
##  Max.   :1.000e+10   Max.   :81849.8
```

```
sapply(billAmount, class) #afficher les types de chaque variable
```

```
##   bill_id    amount
## "numeric" "numeric"
```

```
colSums(is.na(billAmount)) #verifier le nombre des valeurs manquantes
```

```
## bill_id  amount
##       0       0
```

Premier audit de la base demographics.csv

```r
dim(demographics) #afficher la dimension de la base
```

```
## [1] 3000    5
```

```r
summary(demographics) #afficher les statistiques descriptives de la base de donnée
```

```
##                              patient_id      gender          race
##  00225710a878eff524a1d13be817e8e2:   1   f     : 101   chinese: 307
##  0029d90eb654699c18001c17efb0f129:   1   Female:1396   Chinese:1608
##  0040333abd68527ecb53e1db9073f52e:   1   m     : 170   India  : 100
##  00473b58e3dc8ae37b3cb34069705083:   1   Male  :1333   Indian : 195
##  0078662d1d983dde68ea057c42d5b5cf:   1                 Malay  : 629
##  0088bbd94c90bbc9158e13465441ebb6:   1                 Others : 161
##  (Other)                         :2994
##           resident_status    date_of_birth
##  Foreigner        : 143   1975-08-20:   4
##  PR               : 465   1945-10-11:   3
##  Singapore citizen: 610   1956-04-02:   3
##  Singaporean      :1782   1961-12-23:   3
##                           1962-11-03:   3
##                           1967-10-20:   3
##                           (Other)   :2981
```

```r
sapply(demographics, class) #afficher les types de chaque variable
```

```
##      patient_id          gender            race resident_status   date_of_birth
##        "factor"        "factor"        "factor"        "factor"        "factor"
```

```r
colSums(is.na(demographics)) #verifier le nombre des valeurs manquantes
```

```
##      patient_id          gender            race resident_status   date_of_birth
##               0               0               0               0               0
```