**Apprentissage statistiques TP3 : Prédiction de la souscription à un dépôt à termes**

**Karim BENKIRANE**


## Abstract

Les données utilisées pour le TP sont tirées d'UCI machine learning repositery. Il s'agit des données "Bank Marketing Data Set" publiées en 2012. Nous avons été confronté à un déséquilibre de notre échantilon que nous avons dû corrigé; aussi, une stratégie de discrétisation des variables a été mise en place avant de lancer un Random Forest et une SVM.


# Introduction

Le jeu de données dont nous disposons résulte d' une action de marketing direct menée par une banque portugaise.Les prises de contacts avec les propspects se font par téléphone et requiert parfois plus d'un appel avec la même personne. Le but de cette campagne est de promouvoir la souscription à un dépôt à termes. Elle a permis de collecter un certains nombres de données sur les prospects avec pour résultat la souscription ou non au produit bancaire.

Nous tâchons donc dans le rapport qui suit de prédire la souscription à ce dernier.

Dans un premier temps nous procédons à une brève description des données (caractérisée notamment par un dsiquilibre), avant de passer au retraitement des données (discrétisation notamment) pour finisr par la construction des modèles Random Forest, SVM) et le résultat des prédictions.

# Preprocessing

## 1. Description générale de la table

La table contient 41188 observations représentatnt un prospect et 21 variables dont 11 catégorielles et 10 quantitatives. La variable à prédire étant y (souscription à un dépôt à termes ou non)

Pour en citer certaines

- Catégorielles : le type d'emploi, mois,le jour...

- Discrètes : âge, nombre de fois contacté au cours de la campagne...

- Continues : indice de confiance du consommateur...

Par ailleurs la table ne contient aucune observation avec au moins une donnée manquante.

Intéressons-nous à la répartition du label y (accepter de souscrire à un dépôt à termes ou non)

| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. |
|---|---|---|---|---|---|
| 0.0 | 102.0 | 180.0 | 258.3 | 319.0 | 4918.0 |

Statistiques descriptives de la durée du contact téléphonique.

Comme nous pouvons le constater nos données sont très déséquilibrées entre les souscripteurs et non souscripteurs. Il apparaît donc évident qu'il faut traiter cet aspect pour ne pas affecter nos modèles de prédiction. Nous reviendrons sur ce sujet plus bas.

## 2. Discrétisation des features quantitatives

En partant du constat suivant : nos variables quantitatives peuvent présenter des distributions asymétriques, des observations aberrantes pouvant affecter la qualité de nos modèles de prédiction (voir figures ci-dessous).

| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. |
|---|---|---|---|---|---|
| 0.0 | 102.0 | 180.0 | 258.3 | 319.0 | 4918.0 |

Statistiques descriptives de la durée du contact téléphonique.

| Value | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 | 20 | 22 | 24 | 26 | 28 | 1000 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Frequency | 41 | 61 | 603 | 412 | 142 | 52 | 122 | 20 | 43 | 7 | 6 | 3 | 1 | 1 | 1 | 39673 |
| Proportion | 0.001 | 0.001 | 0.015 | 0.010 | 0.003 | 0.001 | 0.003 | 0.000 | 0.001 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.963 |

Fréquence du nombre de jours écoulés après le dernier contact téléphonique du prospect lors d'une précédente campagne.

Nous optons pour cette approche qui transforme nos données quantitatives en qualitatives. De plus, sachant que nous allons recourir à un algorithme Random Forest et SVM , la discrétisation est recommandée.

Deux approches de transformations ont été utilisées :

- Pour les entiers, type données sur l'âge, le nombre de jours ; nous sommes partie de l'échantillon total et avons créer des intervalles à petits écarts pour les fortes distributions afin d'améliorer la précision et injecté les valeurs aberrantes dans de larges intervalles.

- Pour les valeurs réelles, nous avons partitionné notre table en 2 tables distinctes (une ne contenant que y = oui et l'autre y= non, le label), et avons créer des intervalles par quartiles. Le partitionnement a été fait dans le but d'avoir la meilleure représentation du comportement des valeurs continues pour chaque classe.

Les tables ont ensuite été empilées pour n'en former qu'une

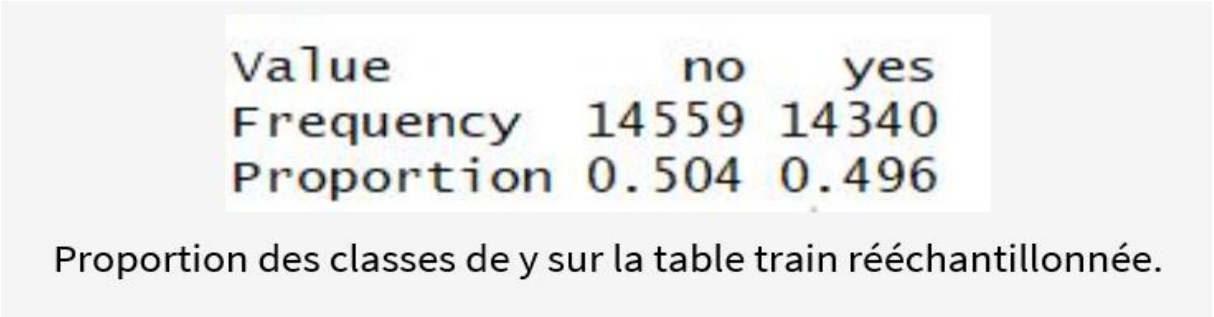
## 3. Rééchantillonnage

Comme vu plus haut, notre jeu de données est très déséquilibré ente les souscripteurs et les non souscripteurs, ceci s'avère très risqué pour faire de la prédiction. En effet, étant donnée la sur représentation des non, les algorithmes qu'on entraînera ne saura prédire que les non, bien qu'il aura un bon score de bonnes prédictions (autour de 88%, la proportion des non dans l'échantillon) il sera mauvais.

Nous optons donc pour la méthode du bootstrap que nous appliquons sur notre échantillon d'entraînement (qui représente 70% de notre échantillon total) et laissant l'échantillon test tel qu'il était à l'origine, déséquilibré (nous reviendrons sur ce choix plus bas).

La méthode du Bootsrap a été choisie car elle est d'une aide conséquente dans la tâche de classification binaire de classes rares ; dans le sens où elle procède à l'échantillonnage en estimant synthétiquement la densité conditionnelle des classes.

Suite à l’application du Bootsrap sur notre échantillon d’entraînement, voici la nouvelle répartition



| Value | no | yes |
|---|---|---|
| Frequency | 14559 | 14340 |
| Proportion | 0.504 | 0.496 |

Proportion des classes de y sur la table train rééchantillonnée.

Nous pouvons constater que nos classes sont équilibrées.

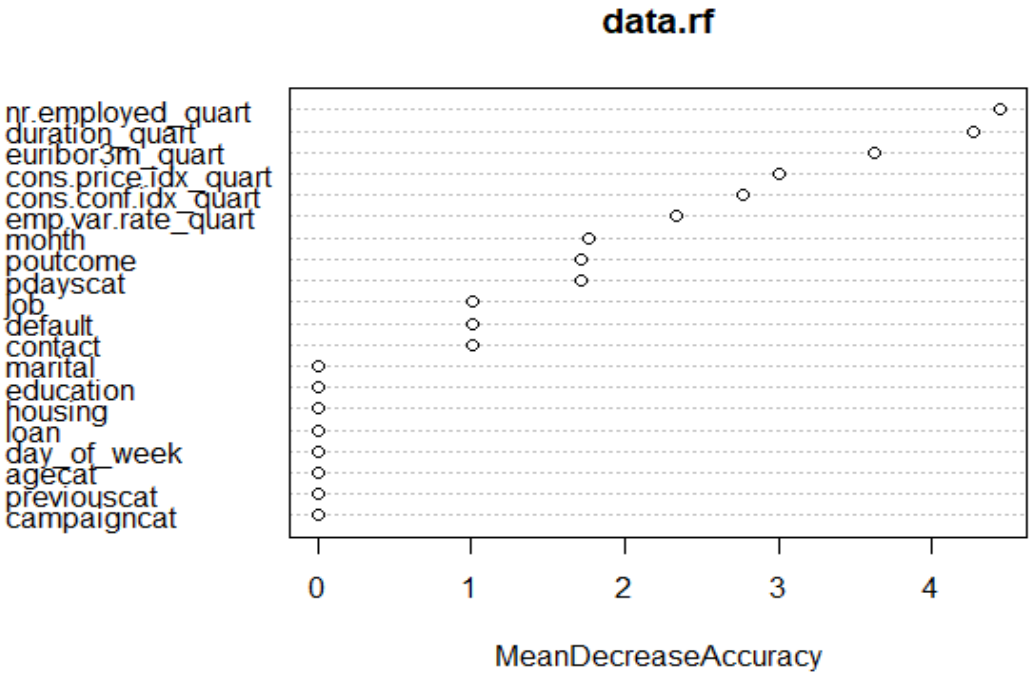
# Prédiction

Comme signalé plus haut, à ce stade nous avons déjà partitionner nos tables en échantillon train et tes (70%,30%). La table train a été rééchantillonnée dans le but de permettre l'apprentissage de nos modèles, le test, lui n'a pas été modifié car notre but est d'évaluer la qualité de nos algorithmes sur la vraie donnée déséquilibrée. L'enjeu central, étant que le modèle puisse prédire le mieux possible les souscripteurs à un dépôt à termes.

Pour l'étude, 2 modèles ont été mis en concurrence, le Random Forest (modèle de bagging) et le SVM.Ces deux algorithmes sont adaptés aux grandes dimensions.

## 1. Sélection des variables

Pour la sélection des variables la plus optimales, nous avons eu recours au Random Forest (tourne sur l'échantillon d'entrainement) pour produire le graphique de variables d'importances suivant, qui classe les features par la perte en précision qu'engendrerait leur retrait.




Il est intéressant de remarquer que la durée de l'appel téléphonique (duration_quart) est très importante pour expliquer la prédiction, ce qui parait intuitif. Cela étant dit il n'est pas réaliste de garder ce feature car la décision de souscription se fait juste après l'appel. Nous décidons donc de le retirer et de garder seulement les 11 plus importants.

## 2. Random Forest VS SVM

Nous faisons tourner les 2 algorithmes sur l'échantillon train en ne retenant que les 11 variables précédentes. Nous procédons aussi en parallèle au tunning des paramètres du SVM (paramètres gamma, T ) par cross validation sur 10 Kfolds pour améliorer au mieux notre modèles.

Ci-dessous les résultats obtenues (à gauche la matrice de confusion du RF, à droite celle du SVM)

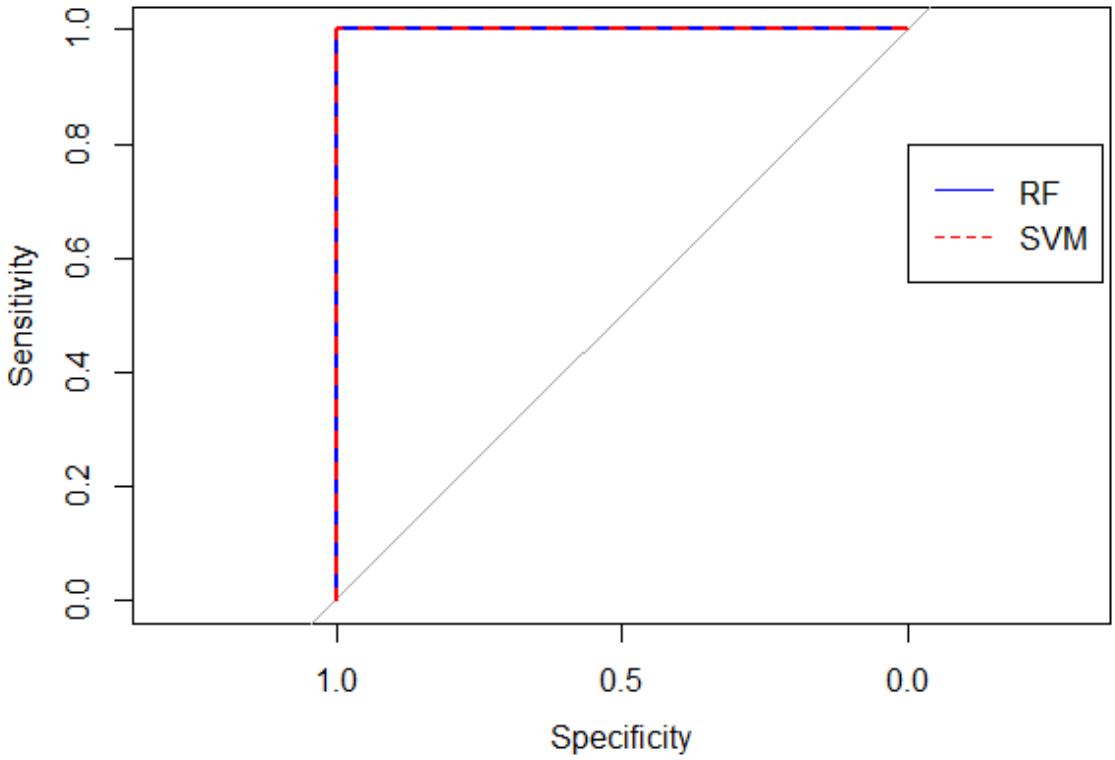
| rf_prediction | no | yes |
|---|---|---|
| no | 10969 | 0 |
| yes | 0 | 1320 |

| SVM_prediction | no | yes |
|---|---|---|
| no | 10969 | 0 |
| yes | 0 | 1320 |

Matrices de confusions

Nous pouvons noter que les deux ont un taux d'erreur de 0% et classent parfaitement les souscripteurs et non souscripteurs.

Il nous paraît assez évident que les 2 algorithmes ont des courbes ROC quasi parfaite, et équivalente. Confortons cette intuition en les affichant



C’est effectivement le cas, nos deux algorithmes ont une courbe ROC en forme d’angle droit. Ceci confirme leur qualité, qui est équivalente.

# Conclusion

La discrétisation des données couplée au rééchantillonnage et à la sélection des variables d’importance a préparé le terrain pour les modèles.  Le fait que le Random Forest et le SVM, soient deux algorithmes puissants de par la facilité à gérer les espaces de grandes dimensions vient parfaire la stratégie.

# ANNEXE- Script R

```
##########################################################
#          TP3 APPRENTISSAGE STATISTIQUES         #
#               Karim BENKIRANE                #
##########################################################


### Nous fixons le chemin où se trouve nos données et vérifions que le répertoire est le bon ###

setwd("C:/Users/MKB/Desktop/bank-additional/bank-additional")

getwd()

install.packages("Hmisc")

install.packages("DMwR")

library(DMwR)

library(Hmisc)

describe(data.rose$y)

### Import de la table et association à un objet ###

Table<-read.csv2(file="bank-additional-full.csv",header=TRUE,sep=";",dec=".",fill=TRUE)
```

```
install.packages("DMwR")
Library(DMwR)

   ### ANALYSE EXPLORATOIRE ###

# Descriptif général de la table

str(Table) # la table contient 41188 observations et 21 variables
table(sapply(Table, class)) # 11 données catégorielles,10 données numériques dont 5 discrètes et 5 continues
missing = apply(Table,1,anyNA) # Comptage du nombre d'observations contenant au moins
sum(missing)/length(missing)            # une donnée manquante
# Aucune donnée manquante
describe(Table$y) # proportion des modalités des variables à prédire, sous représentation des yes


# Production de quelques visualisations de données

# Fréquence des autres variables discrètes
describe(Table$pdays)
describe(Table$campaign)
describe(Table$duration)
summary(Table$campaign)
summary(Table$duration)
describe(Table$age)


### Retraitement des données ###

## Intervalle de valeurs pour les variables discrètes
```

```
# Création d'une nouvelle variable âge par intervalle

Table$agecat[Table$age < 17] <- "- de 17 ans"
Table$agecat[Table$age >= 17 & Table$age <= 25] <- "Entre 17 et 25 ans"
Table$agecat[Table$age > 25 & Table$age <= 30] <- "Entre 26 et 30 ans"
Table$agecat[Table$age > 30 & Table$age <= 35] <- "Entre 31 et 35 ans"
Table$agecat[Table$age > 35 & Table$age <= 40] <- "Entre 36 et 40 ans"
Table$agecat[Table$age > 40 & Table$age <= 50] <- "Entre 41 et 50 ans"
Table$agecat[Table$age > 50 & Table$age <= 60] <- "Entre 51 et 60 ans"
Table$agecat[Table$age > 60] <- "+ 60 ans"
Table$agecat<-as.factor(Table$agecat)


# Création d'une nouvelle variable Previouscat par intervalle

Table$previouscat[Table$previous <= 0] <- "0"
Table$previouscat[Table$previous >= 1 & Table$previous <= 2 ] <- "entre 1 et 2"
Table$previouscat[Table$previous > 2] <- "+ de 2"
Table$previouscat<-as.factor(Table$previouscat)

# Création d'une nouvelle variable Pdayscat par intervalle

Table$pdayscat[Table$pdays >= 0 & Table$pdays <= 2] <- "entre 0 et 2"
Table$pdayscat[Table$pdays > 2 & Table$pdays <= 8] <- "entre 3 et 8"
Table$pdayscat[Table$pdays > 8 & Table$pdays <= 12] <- "entre 9 et 12 "
Table$pdayscat[Table$pdays > 12 & Table$pdays < 999] <- "+ de 12"
Table$pdayscat[Table$pdays >= 999] <- "Pas de contact"
Table$pdayscat<-as.factor(Table$pdayscat)
```

```
# Création d'une nouvelle variable campaign par intervalle
Table$campaigncat[Table$campaign > 0 & Table$campaign < 2 ] <- "1"
Table$campaigncat[Table$campaign > 1 & Table$campaign <= 3] <- "entre 2 et 3"
Table$campaigncat[Table$campaign > 3 ] <- "+ de 3 "
Table$campaigncat<-as.factor(Table$campaigncat)


## Intervalle de valeurs pour les variables continues

# Partition de la table par observations avec label yes et label no

Table_no<- Table[Table$y == "no", ]
Table_yes<- Table[Table$y == "yes", ]

## Discrétisation par quantiles des variables continues pour chaque table

# Table_no
Table_no <- within(Table_no,cons.conf.idx_quart <- (cut(cons.conf.idx,
                                        unique(quantile(cons.conf.idx, probs=0:4/4)),
                                        include.lowest=TRUE)))
Table_no <- within(Table_no,cons.price.idx_quart <- (cut(cons.price.idx,
                                            unique(quantile(cons.price.idx, probs=0:4/4)),
                                            include.lowest=TRUE)))
Table_no <- within(Table_no,emp.var.rate_quart <- (cut(emp.var.rate,
                                            unique(quantile(emp.var.rate, probs=0:4/4)),
                                            include.lowest=TRUE)))
Table_no <- within(Table_no,nr.employed_quart <- (cut(nr.employed,
                                            unique(quantile(nr.employed, probs=0:4/4)),
                                            include.lowest=TRUE)))
Table_no <- within(Table_no,euribor3m_quart <- (cut(euribor3m,
```

```
                                        unique(quantile(euribor3m, probs=0:4/4)),
                                      include.lowest=TRUE)))
Table_no <- within(Table_no,duration_quart <- (cut(duration,
                                        unique(quantile(duration, probs=0:4/4)),
                                        include.lowest=TRUE)))
describe(data.rose$y)
# Table_yes

Table_yes <- within(Table_yes,cons.conf.idx_quart <- (cut(cons.conf.idx,
                                         unique(quantile(cons.conf.idx, probs=0:4/4)),
                                         include.lowest=TRUE)))
Table_yes <- within(Table_yes,cons.price.idx_quart <- (cut(cons.price.idx,
                                          unique(quantile(cons.price.idx, probs=0:4/4)),
                                          include.lowest=TRUE)))
Table_yes <- within(Table_yes,emp.var.rate_quart <- (cut(emp.var.rate,
                                        unique(quantile(emp.var.rate, probs=0:4/4)),
                                        include.lowest=TRUE)))
Table_yes <- within(Table_yes,nr.employed_quart <- (cut(nr.employed,
                                        unique(quantile(nr.employed, probs=0:4/4)),
                                        include.lowest=TRUE)))
Table_yes <- within(Table_yes,euribor3m_quart <- (cut(euribor3m,
                                      unique(quantile(euribor3m, probs=0:4/4)),
                                      include.lowest=TRUE)))
Table_yes <- within(Table_yes,duration_quart <- (cut(duration,
                                    unique(quantile(duration, probs=0:4/4)),
                                    include.lowest=TRUE)))



## Empilement des 2 tables
```

```
Table_full_discret <- rbind(Table_no, Table_yes)

## On ne garde que les colonnes de type factor

myvars <- names(Table_full_discret) %in% c("age", "duration",
"campaign","pdays","previous",

                    "emp.var.rate","cons.price.idx","cons.conf.idx","euribor3m",

                                "nr.employed","agecate")

Table_full_dis <- Table_full_discret[!myvars]



#### Création d'un échantillon d'apprentissage et échantillon test (70/30)


set.seed(7)

ss <- sample(1:2,size=nrow(Table_full_dis),replace=TRUE,prob=c(0.7,0.3))

Table_full_dis_train <- Table_full_dis[ss==1,]

Table_full_dis_test <- Table_full_dis[ss==2,]


##### Resampling de la donnée par la méthode ROSE (bootstrap)

install.packages("ROSE")

library(ROSE)

data.rose <- ROSE(y ~ ., data = Table_full_dis_train, seed = 1)$data

nrow(data.rose)

describe(data.rose$y)


        ########## Prédiction #########



#### Random Forest (nous permet de sélectionner les varibles d'importances pour
les algorithmes en plus

#### de faire de la prédiction)
```

```
# Installation du package random Forest
install.packages("randomForest")
library(randomForest)

# Spécification du modèle

set.seed(1234)
data.rf <- randomForest(y ~ ., data=data.rose, importance=TRUE,ntree=50)
print(data.rf)
varImpPlot(data.rf, sort=TRUE,type=1)

# Nous gardons donc les 12 variables les plus importantes

datanew.rf <- randomForest(y ~ cons.price.idx_quart + euribor3m_quart +
nr.employed_quart +
                           cons.conf.idx_quart + month + emp.var.rate_quart +
                            + pdayscat + job + default + contact + poutcome
                          , data=data.rose, importance=TRUE,ntree=50)

# Appliquons le modèle à l'échantillon test

rf_prediction= predict(datanew.rf, Table_full_dis_test, type="response")
str(rf_prediction)

table(rf_prediction, Table_full_dis_test$y)

# taux de bonne prédiction de 100%

## Création d'une nouvelle table data.rose_new pour la suite (elle
##ne contient que les 12 variables retenues)
```

```
myvars2 <- names(data.rose) %in% c("y","cons.price.idx_quart", "euribor3m_quart",
"nr.employed_quart",
                                   "cons.conf.idx_quart", "month",
                                   "emp.var.rate_quart","pdayscat", "job", "default",
                                    "contact", "poutcome" )
data.rose_new<-data.rose[myvars2]



#### SVM

install.packages("e1071")
library(e1071)

## Faisons tourner le modèle avec une cross validation sur 10 Kfolds
svm.model <- svm(y~ .,data=data.rose_new,cross=10)
SVM_prediction <- predict(svm.model, Table_full_dis_test)
table(SVM_prediction, Table_full_dis_test$y)

# 100% de bonne réponse

######## Courbe ROC des deux modèles
install.packages("pROC")
library(pROC)

rf_prediction1<- as.numeric(rf_prediction)
roc_rf<-roc(Table_full_dis_test$y, rf_prediction1)
SVM_prediction1 <- as.numeric(SVM_prediction)
roc_SVM<-roc(Table_full_dis_test$y, SVM_prediction1)

plot(roc_rf,col="blue", lty=1 )
```

```
lines(roc_SVM,col="red",lty=2)
legend(0,0.8,legend=c("RF","SVM"),lty=c(1,2), col=c("blue","red"))
```