Pràctica 2: Neteja i validació de les dades

Carlos Pérez Martín i Oscar Fernandez Castro

18 de mayo de 2019

1. Descripció del dataset. Perquè és important i quina pregunta/problema pretén respondre?

Els datasets obtingut de <https://www.kaggle.com>, contenen informació sobre els passatgers del titanic. Des de dades demogràfiques dels passatgers fins a si van sobreviure o no al viatge.

En total hi han 3 datatsets:

* train: 12 atributs amb 891 files.
* test: 11 atributs amb 418 files.
* gender\_submission: 2 atributs amb 418 files.

Els tres datasets formen un conjunt de prova i d’entrenament.

Els atributs que tenim en els diferents datasets són:

* Survival: Supervivent (0 = No, 1 = Si)
* Pclass: Classe del passatger (1= 1era, 2 = 2ona, 3 = 3ra)
* Name: Nom
* Sex: Sexe (female, male)
* Age: edad
* Sibsp: Nombre de familiars
* Parch: Nombre de pares/fills embarcats
* Ticket: Número d’entrada
* Fare: Tarifa
* Cabin: cabina
* Embarked: Embarcat (C = Cherbourg, Q = Queenstown, S = Southampton)

La pregunta que volem respondre, és quins són els factors més determinants que influeixen en la taxa de supervivència del passatge. Quins atributs tenen un impacte més elevat? (sexe, classe, edat, etc). Per això s’utilitzaran mètodes de regressió per avaluar les relacions entre atributs.

A continuació es mostra un resum de les dades a analitzar.

summary(data\_train)

## PassengerId Survived Pclass   
## Min. : 1.0 Min. :0.0000 Min. :1.000   
## 1st Qu.:223.5 1st Qu.:0.0000 1st Qu.:2.000   
## Median :446.0 Median :0.0000 Median :3.000   
## Mean :446.0 Mean :0.3838 Mean :2.309   
## 3rd Qu.:668.5 3rd Qu.:1.0000 3rd Qu.:3.000   
## Max. :891.0 Max. :1.0000 Max. :3.000   
##   
## Name Sex Age   
## Abbing, Mr. Anthony : 1 female:314 Min. : 0.42   
## Abbott, Mr. Rossmore Edward : 1 male :577 1st Qu.:20.12   
## Abbott, Mrs. Stanton (Rosa Hunt) : 1 Median :28.00   
## Abelson, Mr. Samuel : 1 Mean :29.70   
## Abelson, Mrs. Samuel (Hannah Wizosky): 1 3rd Qu.:38.00   
## Adahl, Mr. Mauritz Nils Martin : 1 Max. :80.00   
## (Other) :885 NA's :177   
## SibSp Parch Ticket Fare   
## Min. :0.000 Min. :0.0000 1601 : 7 Min. : 0.00   
## 1st Qu.:0.000 1st Qu.:0.0000 347082 : 7 1st Qu.: 7.91   
## Median :0.000 Median :0.0000 CA. 2343: 7 Median : 14.45   
## Mean :0.523 Mean :0.3816 3101295 : 6 Mean : 32.20   
## 3rd Qu.:1.000 3rd Qu.:0.0000 347088 : 6 3rd Qu.: 31.00   
## Max. :8.000 Max. :6.0000 CA 2144 : 6 Max. :512.33   
## (Other) :852   
## Cabin Embarked  
## :687 : 2   
## B96 B98 : 4 C:168   
## C23 C25 C27: 4 Q: 77   
## G6 : 4 S:644   
## C22 C26 : 3   
## D : 3   
## (Other) :186

1. Integració i selecció de les dades d’interès a analitzar.

Per començar unificarem els tres datasets en un per fer la neteja de les dades. Després en cas necessari ja se separaran de nou en conjunts de prova i d’entrenament.

data\_test<- read.csv('C:/Users/vodsk/Desktop/UOC/Topologia i cicle de vida de les dades/PRACTICA2/kaggle-titanic-master/input/test.csv',header=T,sep=",", encoding = "UTF-8")  
data\_sub<-read.csv('C:/Users/vodsk/Desktop/UOC/Topologia i cicle de vida de les dades/PRACTICA2/kaggle-titanic-master/input/gender\_submission.csv',header=T,sep=",", encoding = "UTF-8")  
#unifiquem el dataset de test amb els seus resultats  
data\_test <- merge(data\_test,data\_sub, by="PassengerId")  
#juntem les files dels datasets de train i test  
data<-rbind(data\_train, data\_test)

Ara ja tenim el dataset complet amb 1309 registres i 12 atributs.

Volem analitzar el nombre més gran d’atributs possibles, però a simple vista, ja podem dir que els atributs Name i PassangerId, no ens aporten informació, així que els eliminarem.

Agafarem l’atribut survived com a referència i utilitzant diferents mètodes estadístics, veurem quin impacte tenen els altres atributs sobre aquest.

1. Neteja de les dades.

Al fer una revisió del tipus de dades, veiem que l’atribut Pclass és de tipus enter, quan hauria de ser de tipus factor. En el cas de l’atribut Survived també és de tipus enter quan hauria de ser binari. L’atribut survived es podría convertir a un atribut de tipus lògic (true/false)

Tipus de dades abans de la conversió

sapply(data,class)

## PassengerId Survived Pclass Name Sex Age   
## "integer" "integer" "integer" "factor" "factor" "numeric"   
## SibSp Parch Ticket Fare Cabin Embarked   
## "integer" "integer" "factor" "numeric" "factor" "factor"

Tipus de dades desprès de la conversió

data$Pclass<-as.factor(data$Pclass)  
data$Survived<-as.logical(data$Survived)  
sapply(data,class)

## PassengerId Survived Pclass Name Sex Age   
## "integer" "logical" "factor" "factor" "factor" "numeric"   
## SibSp Parch Ticket Fare Cabin Embarked   
## "integer" "integer" "factor" "numeric" "factor" "factor"

3.1. Les dades contenen zeros o elements buits? Com gestionaries aquests casos?

Busquem en tots els atributs els valors NA i buits.

Valors NA

sapply(data, function(x) sum(is.na(x)))

## PassengerId Survived Pclass Name Sex Age   
## 0 0 0 0 0 263   
## SibSp Parch Ticket Fare Cabin Embarked   
## 0 0 0 1 0 0

Valors buits

sapply(data, function(x) sum(x==""))

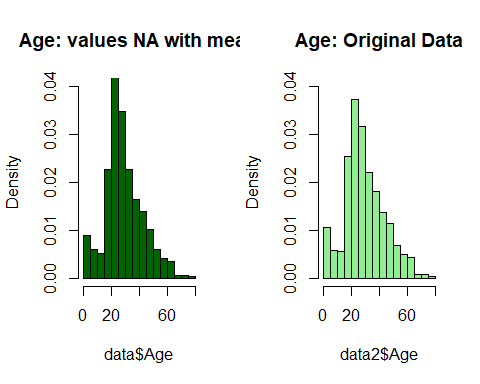
## PassengerId Survived Pclass Name Sex Age   
## 0 0 0 0 0 NA   
## SibSp Parch Ticket Fare Cabin Embarked   
## 0 0 0 NA 1014 2

L’atribut Age té 263 casos amb valor NA, i l’atribut Embarked 2 casos de valor buit. En cada cas utilitzarem una tècnica diferent, per l’atribut age, substituirem els valors NA segons k-Nearest Neighbors (kNN), i en el cas de l’atribut embarked els modificarem per l’atribut majoritari.

data$Age <- kNN(data)$Age  
data$Embarked[data$Embarked==""]<- "S"  
#data$Age[is.na(data$Age)]<-28 #mediana = 28

Un cop modificat el valor d’Age, volem veure si aquesta modificació ha tingut un gran impacte. Creem un dataframe nou, amb les dades sense modificar, y es comparen les dades modificades amb la mitjana amb les dades originals.

data2<-rbind(data\_train, data\_test)  
par(mfrow=c(1,2))  
hist(data$Age, freq=F, main="Age: values NA with mean",col='darkgreen', ylim=c(0,0.04))  
hist(data2$Age, freq=F, main="Age: Original Data",col='lightgreen', ylim=c(0,0.04))

 Es pot observar que el fet de modificar els valors NA d’Age seguint el mètode de k-Nearest Neighbors no afecta gaire a la distribució dels valors, per tant es pot concluir que es una bona aproximació dels valors.

En el cas de l’atribut cabina (cabin) tenim 1014 registres buits dels 1309 del dataset. A més els valors que tenim estan molt fragmentats. En aquestes circumstancies l’atibut cabina no ens aportarà informació rellevant o fiable, així que l’eliminarem en la part de selecció d’atributs (apartat 4.1).

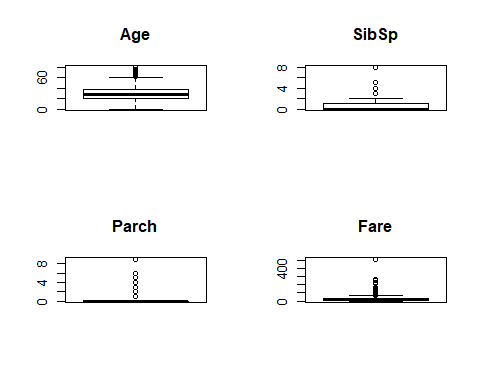
L’atribut tarifa (fare) conté un valor NA. En aquest cas el modifiquem i li posem la mitjana. Per a no pasarli un valor fixe, modifiquem la forma de assignar la mitjana al valor nul. Calculo la mitjana de la variable Fare, excloent al càlcul els valors NA.

data$Fare[is.na(data$Fare)] <- mean(data$Fare, na.rm=TRUE)

3.2. Identificació i tractament de valors extrems.

Busquem valors extrems en els atributs númerics. Comencem amb un mètode molt visualt, el blotpox.

par(mfrow=c(2,2))  
for(i in 2:ncol(data)) #ignorem PassengerId  
{  
 if ((is.numeric(data[,i]))||(is.integer(data[,i])))  
 {  
 boxplot(data[,i], main = colnames(data)[i], width=100)   
 }  
   
}



boxplot.stats(data$Age)$out

## [1] 66.0 65.0 71.0 70.5 62.0 63.0 65.0 64.0 65.0 63.0 71.0 64.0 62.0 62.0  
## [15] 80.0 70.0 70.0 62.0 74.0 62.0 63.0 67.0 76.0 63.0 64.0 64.0 64.0

boxplot.stats(data$SibSp)$out

## [1] 3 4 3 3 4 5 3 4 5 3 3 4 8 4 4 3 8 4 8 3 4 4 4 4 8 3 3 5 3 5 3 4 4 3 3  
## [36] 5 4 3 4 8 4 3 4 8 4 8 3 4 5 3 4 8 4 8 4 3 3

boxplot.stats(data$Parch)$out

## [1] 1 2 1 5 1 1 5 2 2 1 1 2 2 2 1 2 2 2 3 2 2 1 1 1 1 2 1 1 2 2 1 2 2 2 1  
## [36] 2 1 1 2 1 4 1 1 1 1 2 2 1 2 1 1 1 2 1 1 2 2 2 1 1 2 2 1 2 1 1 1 1 1 1  
## [71] 1 2 1 2 2 1 1 2 1 1 2 1 1 1 1 2 1 1 1 4 1 1 2 2 2 2 2 1 1 1 2 2 1 1 2  
## [106] 2 3 4 1 2 1 1 2 1 2 1 2 1 1 2 2 1 1 1 1 2 2 2 2 2 2 1 1 2 1 4 1 1 2 1  
## [141] 2 1 1 2 5 2 1 1 1 2 1 5 2 1 1 1 2 1 6 1 2 1 2 1 1 1 1 1 1 1 3 2 1 1 1  
## [176] 1 2 1 2 3 1 2 1 2 2 1 1 2 1 2 1 2 1 1 1 2 1 1 2 1 2 1 1 1 1 3 2 1 1 1  
## [211] 1 5 2 1 1 1 1 3 1 2 2 1 2 1 2 1 2 4 1 1 2 1 1 1 4 6 2 3 1 1 2 2 2 1 1  
## [246] 2 5 2 3 2 1 1 1 2 1 2 2 2 1 2 1 1 2 1 2 1 2 1 2 2 1 1 1 1 1 2 1 1 2 1  
## [281] 1 1 2 1 2 9 1 1 1 2 2 2 1 9 1 1 2 2 1 1 2 1 1 1 1 1 1

boxplot.stats(data$Fare)$out

## [1] 71.2833 263.0000 146.5208 82.1708 76.7292 80.0000 83.4750  
## [8] 73.5000 263.0000 77.2875 247.5208 73.5000 77.2875 79.2000  
## [15] 66.6000 69.5500 69.5500 146.5208 69.5500 113.2750 76.2917  
## [22] 90.0000 83.4750 90.0000 79.2000 86.5000 512.3292 79.6500  
## [29] 153.4625 135.6333 77.9583 78.8500 91.0792 151.5500 247.5208  
## [36] 151.5500 110.8833 108.9000 83.1583 262.3750 164.8667 134.5000  
## [43] 69.5500 135.6333 153.4625 133.6500 66.6000 134.5000 263.0000  
## [50] 75.2500 69.3000 135.6333 82.1708 211.5000 227.5250 73.5000  
## [57] 120.0000 113.2750 90.0000 120.0000 263.0000 81.8583 89.1042  
## [64] 91.0792 90.0000 78.2667 151.5500 86.5000 108.9000 93.5000  
## [71] 221.7792 106.4250 71.0000 106.4250 110.8833 227.5250 79.6500  
## [78] 110.8833 79.6500 79.2000 78.2667 153.4625 77.9583 69.3000  
## [85] 76.7292 73.5000 113.2750 133.6500 73.5000 512.3292 76.7292  
## [92] 211.3375 110.8833 227.5250 151.5500 227.5250 211.3375 512.3292  
## [99] 78.8500 262.3750 71.0000 86.5000 120.0000 77.9583 211.3375  
## [106] 79.2000 69.5500 120.0000 93.5000 80.0000 83.1583 69.5500  
## [113] 89.1042 164.8667 69.5500 83.1583 82.2667 262.3750 76.2917  
## [120] 263.0000 262.3750 262.3750 263.0000 211.5000 211.5000 221.7792  
## [127] 78.8500 221.7792 75.2417 151.5500 262.3750 83.1583 221.7792  
## [134] 83.1583 83.1583 247.5208 69.5500 134.5000 227.5250 73.5000  
## [141] 164.8667 211.5000 71.2833 75.2500 106.4250 134.5000 136.7792  
## [148] 75.2417 136.7792 82.2667 81.8583 151.5500 93.5000 135.6333  
## [155] 146.5208 211.3375 79.2000 69.5500 512.3292 73.5000 69.5500  
## [162] 69.5500 134.5000 81.8583 262.3750 93.5000 79.2000 164.8667  
## [169] 211.5000 90.0000 108.9000

Els valors dels outliers mostrar pel boxplot de l’atribut edat són valors coherents, no els considerem outliers, ja que es mostren forà del gràfic per la concentració de valors entre els 20 i 30 anys.

El mateix es pot dir pels atributs SibSp i Parch, la majoria de regsitres tenen valor 0, i fa que el boxplot estigui molt comprimit en aquest interval. Tenir 8 fills és un valor elevat, pero no es pot considerar irreal. Per tant, observant els valors de les variables, no es troba necessari realitzar cap altre anàlisi de valors extrems ja que, pels valors que es donen, no es pot considerar cap un valor extrem.

En el cas de l’atribut Fare revisarem a continuació amb més detall els seus valors depenent de l’atribut Pclass, que clarament està relacionat.

par(mfrow=c(2,2))  
boxplot(data$Fare[data$Pclass==1], main = "Fare - Pclass=1", width=100)   
boxplot(data$Fare[data$Pclass==2], main = "Fare - Pclass=2", width=100)   
boxplot(data$Fare[data$Pclass==3], main = "Fare - Pclass=3", width=100)   
summary(data$Fare[data$Pclass==1])

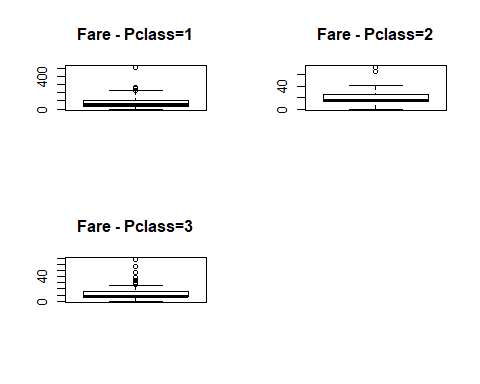
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00 30.70 60.00 87.51 107.66 512.33

summary(data$Fare[data$Pclass==2])

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00 13.00 15.05 21.18 26.00 73.50

summary(data$Fare[data$Pclass==3])

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00 7.75 8.05 13.33 15.25 69.55



Un cop analitzat la tarifa dels bitllets de les diferents classes, observant els valors mitjans, es pot observar una diferència raonable en els preus dels bitllets entre les diferents classes, sent més alta la tarifa a primera classe i més baixa a tercera. Tot i això, sobretot a tercera classe, s’observen uns preus extrems, que podríem considerar erronis, però degut a que no disposem de més informació, decidirem no tractar amb aquests valors, degut a que podría ser correcta aquesta informació.

1. Anàlisi de les dades.

* 4.1. Selecció dels grups de dades que es volen analitzar/comparar (planificació dels anàlisis a aplicar).

Com ja hem comentat anteriorment, tenim atributs que no aporten cap informació útil (PassengerId, Name), i d’altres que tenen masses registres buits perquè puguin aportar informació valida (Cabin). Els eliminem.

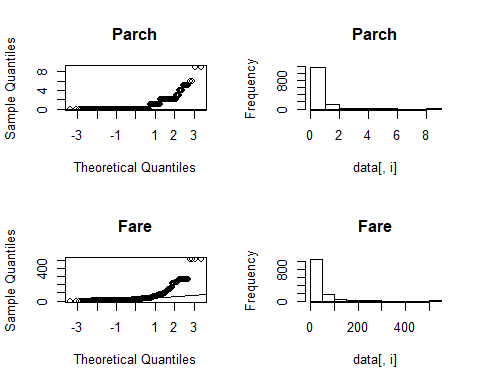
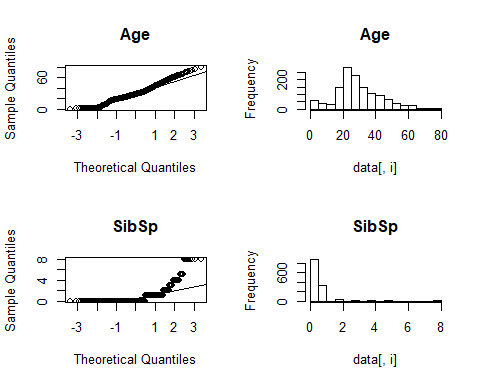
data<-data[ , -which(names(data) %in% c("PassengerId","Name","Cabin"))]

Es vol comparar quins pesos tenen els diferents atributs sobre la supervivenvia o no d’un individu, quins són els factors que més influeixen.

4.2. Comprovació de la normalitat i homogeneïtat de la variància.

Per comprobar la normalitat de les dades fem servir un anàlisi visual (qqplot i histograma) i el test shapiro-wilk.

par(mfrow=c(2,2))  
for(i in 1:ncol(data))  
{  
 if ((is.numeric(data[,i]))|(is.integer(data[,i])))  
 {  
 qqnorm(data[,i], main = colnames(data)[i]);qqline(data[,i])  
 hist(data[,i], main = colnames(data)[i])  
 }  
}



Amb una primera ullada, es veu clarament que els atributs SibSp, Parch i Fare no tenen una distribució normal.

Apliquem l’algoritme de shapiro-wilk per l’atribut edat i la prova no paramètrica de wilcox per els atributs Sibsp, Parch i Fare.

shapiro.test(data$Age)

##   
## Shapiro-Wilk normality test  
##   
## data: data$Age  
## W = 0.97428, p-value = 1.608e-14

wilcox.test(data$SibSp)

##   
## Wilcoxon signed rank test with continuity correction  
##   
## data: data$SibSp  
## V = 87571, p-value < 2.2e-16  
## alternative hypothesis: true location is not equal to 0

wilcox.test(data$Parch)

##   
## Wilcoxon signed rank test with continuity correction  
##   
## data: data$Parch  
## V = 47278, p-value < 2.2e-16  
## alternative hypothesis: true location is not equal to 0

wilcox.test(data$Fare)

##   
## Wilcoxon signed rank test with continuity correction  
##   
## data: data$Fare  
## V = 835280, p-value < 2.2e-16  
## alternative hypothesis: true location is not equal to 0

Els test de shapiro-wilk i el de wilcox ens retornen un p-valor inferior a 0.05 per tant no es compleix la hipòtesi nul·la i per tant les dades no segueixen una distribució normal.

Tant els gràfics com el test de shapiro-wilk, ens indiquen que les dades no segueixen una distribució normal. Però sabem pel que diu el teorema del límit central, que si tenim més de 30 distribucions, es pot aproximar a una distribució normal de mitja 0 i desviació estàndard 1.

Ara revisem l’homogeneïtat de la variància. Com no es compleix la condició de normalitat d’algunes mostres, utilitzem la prova de Fligner-Killen. En aquesta prova la hipòtesis nul·la diu que les variàncies són iguals i que els diferents valors no afecten.

Compararem els atributs en qüestió respecte a l’objectiu de la pràctica que és mesurar la influencia sobre l’atribut survived.

fligner.test(data$Survived ~ data$Age, data=data)

##   
## Fligner-Killeen test of homogeneity of variances  
##   
## data: data$Survived by data$Age  
## Fligner-Killeen:med chi-squared = 76.576, df = 97, p-value =  
## 0.9377

fligner.test(data$Survived ~ data$SibSp, data=data)

##   
## Fligner-Killeen test of homogeneity of variances  
##   
## data: data$Survived by data$SibSp  
## Fligner-Killeen:med chi-squared = 30.756, df = 6, p-value =  
## 2.822e-05

fligner.test(data$Survived ~ data$Parch, data=data)

##   
## Fligner-Killeen test of homogeneity of variances  
##   
## data: data$Survived by data$Parch  
## Fligner-Killeen:med chi-squared = 18.229, df = 7, p-value =  
## 0.01098

fligner.test(data$Survived ~ data$Fare, data=data)

##   
## Fligner-Killeen test of homogeneity of variances  
##   
## data: data$Survived by data$Fare  
## Fligner-Killeen:med chi-squared = 293.05, df = 281, p-value =  
## 0.2984

Els resultats ens mostren que la variància de la superviencia és similar pels atributs age i fare, pero que varia pels diferents valors de SibSp i Parch.

4.3. Aplicació de proves estadístiques per comparar els grups de dades. En funció de les dades i de l’objectiu de l’estudi, aplicar proves de contrast d’hipòtesis, correlacions, regressions, etc. Aplicar almenys tres mètodes d’anàlisi diferents.

Model d’arbre de classificació

El primer model creat, per a predir si un passatger sobreviu o no, serà un model d’arbre de classificació. El primer pas es generar el conjunt d’entrenament, amb el qual es crearà el model, i el conjunt de test, amb el qual es comprobarà com de bo es el model.

train<-data[1:891,]  
test<-data[892:1309,]  
library(C50)  
vars<-c("Pclass","Sex","Age","SibSp","Parch","Fare")  
str(data[c(vars,"Survived")])

## 'data.frame': 1309 obs. of 7 variables:  
## $ Pclass : Factor w/ 3 levels "1","2","3": 3 1 3 1 3 3 1 3 3 2 ...  
## $ Sex : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...  
## $ Age : num 22 38 26 35 35 20 54 2 27 14 ...  
## $ SibSp : int 1 1 0 1 0 0 0 3 0 1 ...  
## $ Parch : int 0 0 0 0 0 0 0 1 2 0 ...  
## $ Fare : num 7.25 71.28 7.92 53.1 8.05 ...  
## $ Survived: logi FALSE TRUE TRUE TRUE FALSE FALSE ...

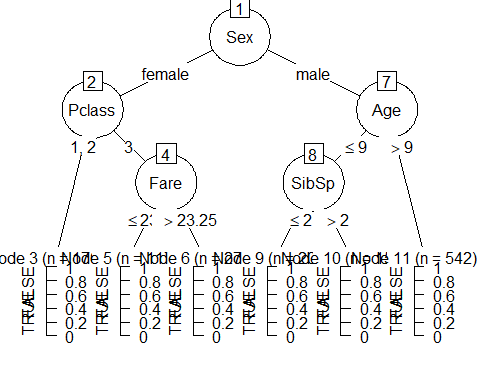
train$Survived<-as.factor(train$Survived)  
tree\_mod <-C5.0(x=train[,vars],y=train$Survived)

A continuació es mostra el model d’arbre de classificació generat.

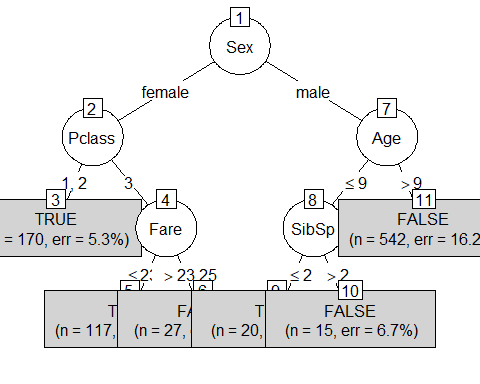
summary(tree\_mod)

##   
## Call:  
## C5.0.default(x = train[, vars], y = train$Survived)  
##   
##   
## C5.0 [Release 2.07 GPL Edition] Sun Jun 02 18:11:01 2019  
## -------------------------------  
##   
## Class specified by attribute `outcome'  
##   
## Read 891 cases (7 attributes) from undefined.data  
##   
## Decision tree:  
##   
## Sex = female:  
## :...Pclass in {1,2}: TRUE (170/9)  
## : Pclass = 3:  
## : :...Fare <= 23.25: TRUE (117/48)  
## : Fare > 23.25: FALSE (27/3)  
## Sex = male:  
## :...Age > 9: FALSE (542/88)  
## Age <= 9:  
## :...SibSp <= 2: TRUE (20)  
## SibSp > 2: FALSE (15/1)  
##   
##   
## Evaluation on training data (891 cases):  
##   
## Decision Tree   
## ----------------   
## Size Errors   
##   
## 6 149(16.7%) <<  
##   
##   
## (a) (b) <-classified as  
## ---- ----  
## 492 57 (a): class FALSE  
## 92 250 (b): class TRUE  
##   
##   
## Attribute usage:  
##   
## 100.00% Sex  
## 64.76% Age  
## 35.24% Pclass  
## 16.16% Fare  
## 3.93% SibSp  
##   
##   
## Time: 0.0 secs

plot(tree\_mod)



plot(tree\_mod,type="simple")



Un cop generat el model, es vol comprobar que el model funciona correctament. Per això, utilitzem el conjunt de test per a comprobar com de bo es el model.

p1<-predict(tree\_mod,test)  
summary(p1)

## FALSE TRUE   
## 260 158

table(test$Survived,Predicted=p1)

## Predicted  
## FALSE TRUE  
## FALSE 254 12  
## TRUE 6 146

Com es pot observar, mitjançant aquest model, dels passatgers que no van sobreviure, classifica 254 casos com a que no sobreviuen, y 12 que sí, mentre que dels que sí van sobreviure, classifica 146 com a que sobreviuen y nomès 12 que no. Per tant, el model de classificació generat es bastant acurat i permet predir de forma bastant exacta si un passatger va sobreviure o no.

Contrast d’hipòtesis

Apliquem el contrast d’hipòtesi als principals atributs per veure quin d’ells és significatiu respecte l’atribut survived.

H0 : µ1 − µ2 = 0 H1 : µ1 − µ2 > 0

Com a hipòtesi nul·la tenim que el nivell de supervivència de les dades del primer paràmetre és el mateix que les dades del segon paràmetre. Com a hipòtesi alternativa, tenim que les dades del primer fan augmentar el nivell de supervivènvia respecte al segon.

Aplicarem la mateixa hipòtesi als diferents atributs. (amb l’interval de confiança per defecte 0.95)

#wilcex necesita el 1er param númeric  
data$Survived\_num <-as.numeric(data$Survived)  
  
wilcox.test(data$Survived\_num[data$Sex == "female"], data$Survived\_num[data$Sex == "male"], alternative = "greater")

##   
## Wilcoxon rank sum test with continuity correction  
##   
## data: data$Survived\_num[data$Sex == "female"] and data$Survived\_num[data$Sex == "male"]  
## W = 333300, p-value < 2.2e-16  
## alternative hypothesis: true location shift is greater than 0

wilcox.test(data$Survived\_num[data$Pclass == "1"], data$Survived\_num[data$Pclass == "3"], alternative = "greater")

##   
## Wilcoxon rank sum test with continuity correction  
##   
## data: data$Survived\_num[data$Pclass == "1"] and data$Survived\_num[data$Pclass == "3"]  
## W = 149590, p-value < 2.2e-16  
## alternative hypothesis: true location shift is greater than 0

wilcox.test(data$Survived\_num, data$SibSp, alternative = "greater")

##   
## Wilcoxon rank sum test with continuity correction  
##   
## data: data$Survived\_num and data$SibSp  
## W = 882030, p-value = 0.05815  
## alternative hypothesis: true location shift is greater than 0

wilcox.test(data$Survived\_num, data$Parch, alternative = "greater")

##   
## Wilcoxon rank sum test with continuity correction  
##   
## data: data$Survived\_num and data$Parch  
## W = 945290, p-value = 6.625e-09  
## alternative hypothesis: true location shift is greater than 0

wilcox.test(data$Survived\_num, data$Fare, alternative = "greater")

##   
## Wilcoxon rank sum test with continuity correction  
##   
## data: data$Survived\_num and data$Fare  
## W = 15326, p-value = 1  
## alternative hypothesis: true location shift is greater than 0

Com els resultats ens mostren, els atributs rellevants que afecten a la supervivencia són Age, Pclass i Fare. Ja que tenim p-valors propers a 0, per tant en aquests casos hem de descartar la hipòtesi nul·la.

Model de regressió múltiple

Ara utilitzarem mètodes de regressió múltiple per esbrinar quins són els atributs que tenen un pes més significatiu en la supervivencia d’una persona.

#train<-data[1:891,]  
#test<-data[892:1309,]  
modelA<-lm(data$Survived ~ data$Sex + data$Age)  
modelB<-lm(data$Survived ~ data$Sex + data$Age + data$Pclass)  
modelC<-lm(data$Survived ~ data$Sex + data$Age + data$Pclass + data$SibSp)  
modelD<-lm(data$Survived ~ data$Sex + data$Age + data$Pclass + data$SibSp + data$Embarked)  
modelE<-lm(data$Survived ~ data$Sex + data$Age + data$Pclass + data$SibSp + data$Embarked + data$Fare)  
modelF<-lm(data$Survived ~ data$Sex + data$Age + data$Pclass + data$SibSp + data$Embarked + data$Fare + data$Parch)  
taulaReg<-matrix(c('A',summary(modelA)$r.squared,  
 'B',summary(modelB)$r.squared,  
 'C',summary(modelC)$r.squared,  
 'D',summary(modelD)$r.squared,  
 'E',summary(modelE)$r.squared,  
 'F',summary(modelF)$r.squared), ncol=2, byrow = TRUE)  
colnames(taulaReg)<-c("Model","R^2")  
taulaReg

## Model R^2   
## [1,] "A" "0.474094399990495"  
## [2,] "B" "0.511724878814509"  
## [3,] "C" "0.51837487026375"   
## [4,] "D" "0.519476548157216"  
## [5,] "E" "0.519800159890929"  
## [6,] "F" "0.519978379885903"

Els resultats obtinguts no són especialment bons, el millor dels models el F, només explica el 52% dels casos totals, pero la diferencia del model f amb el C és infima, donat que els nous atributs introduits no són significatius. De totes maneres ens es útil per veure com influeixen els atributs en la supervivencia.

summary(modelF)

##   
## Call:  
## lm(formula = data$Survived ~ data$Sex + data$Age + data$Pclass +   
## data$SibSp + data$Embarked + data$Fare + data$Parch)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.05726 -0.14010 -0.05733 0.14778 1.00113   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.1047254 0.0463862 23.816 < 2e-16 \*\*\*  
## data$Sexmale -0.6676618 0.0205352 -32.513 < 2e-16 \*\*\*  
## data$Age -0.0037789 0.0008020 -4.712 2.72e-06 \*\*\*  
## data$Pclass2 -0.1252356 0.0335836 -3.729 0.000200 \*\*\*  
## data$Pclass3 -0.2426470 0.0323425 -7.502 1.16e-13 \*\*\*  
## data$SibSp -0.0378592 0.0100595 -3.764 0.000175 \*\*\*  
## data$EmbarkedQ 0.0225065 0.0392975 0.573 0.566933   
## data$EmbarkedS -0.0238933 0.0249661 -0.957 0.338729   
## data$Fare 0.0002557 0.0002414 1.059 0.289699   
## data$Parch -0.0084539 0.0121732 -0.694 0.487513   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3371 on 1299 degrees of freedom  
## Multiple R-squared: 0.52, Adjusted R-squared: 0.5167   
## F-statistic: 156.3 on 9 and 1299 DF, p-value: < 2.2e-16

Podem veure com els atributs més significatius són el sex, age, Pclass i SibSp. El mòdel ens indica que si el sexe = home les probabilitats de supervivencia cauen de manera significativa, al igual que si esten en segona classe o especialment en la tercera.

En canvi els atributs Embarked, Fare i Parch són poc significatius a l’hora d’explicar la supervivencia.

pred<-predict(modelF)  
table(data$Survived,pred>0.5)

##   
## FALSE TRUE  
## FALSE 737 78  
## TRUE 110 384

Depenent del umbral de tall obtindrem més falsos positius o falsos negatius, de totes maneres amb el model obtingut, el nivell d’errors él bastant alt.

Model de classificació Bayesià

Per últim es decideix generar un segon model de classificació, però aquest cop seguint un classificador Bayesià ingenu, mitjançant NaiveBayes. NaiveBayes es un algoritme d’aprenentatge automàtic basat en el teorema de Bayes.

El primer pas es establir quines variables s’utilitzaràn per a generar el model. Desprès de diverses probes, s’ha vist que, el·liminant les variables Parch, Ticket, Fare y Embarked, s’aconsegueix el model més exacte. Per tant, el primer punt es el·liminar aquestes variables. Això permet assegurar que els resultats del model de regressió múltiple anterior son correctes, ja que amb les variables més rellevants que s’indiquen al model anterior, es quan s’aconsegueix el millor model de predicció.

dataNaiveBayes <- data  
dataNaiveBayes <- within(dataNaiveBayes,{  
 Parch <- NULL  
 Ticket <- NULL  
 Fare <- NULL  
 Embarked <- NULL  
 })  
trainNaiveBayes<-dataNaiveBayes[1:891,]  
testNaiveBayes<-dataNaiveBayes[892:1309,]

El model generat es el següent.

model <- naiveBayes(Survived ~ ., data = trainNaiveBayes)  
model

##   
## Naive Bayes Classifier for Discrete Predictors  
##   
## Call:  
## naiveBayes.default(x = X, y = Y, laplace = laplace)  
##   
## A-priori probabilities:  
## Y  
## FALSE TRUE   
## 0.6161616 0.3838384   
##   
## Conditional probabilities:  
## Pclass  
## Y 1 2 3  
## FALSE 0.1457195 0.1766849 0.6775956  
## TRUE 0.3976608 0.2543860 0.3479532  
##   
## Sex  
## Y female male  
## FALSE 0.1475410 0.8524590  
## TRUE 0.6812865 0.3187135  
##   
## Age  
## Y [,1] [,2]  
## FALSE 29.89435 13.30605  
## TRUE 28.73003 14.34411  
##   
## SibSp  
## Y [,1] [,2]  
## FALSE 0.5537341 1.2883991  
## TRUE 0.4736842 0.7086875  
##   
## Survived\_num  
## Y [,1] [,2]  
## FALSE 0 0  
## TRUE 1 0

Un cop generat el model, es proba el model amb el conjunt de test.

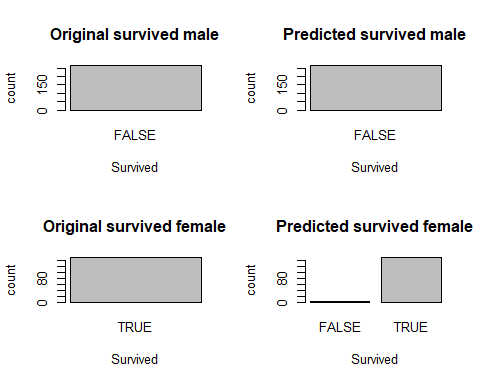
pred <- predict(model,testNaiveBayes)  
tab <- table(testNaiveBayes$Survived, pred, dnn=c("Actual", "Predita"))  
confusionMatrix(tab)

## Confusion Matrix and Statistics  
##   
## Predita  
## Actual FALSE TRUE  
## FALSE 266 0  
## TRUE 2 150  
##   
## Accuracy : 0.9952   
## 95% CI : (0.9828, 0.9994)  
## No Information Rate : 0.6411   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9896   
##   
## Mcnemar's Test P-Value : 0.4795   
##   
## Sensitivity : 0.9925   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.9868   
## Prevalence : 0.6411   
## Detection Rate : 0.6364   
## Detection Prevalence : 0.6364   
## Balanced Accuracy : 0.9963   
##   
## 'Positive' Class : FALSE   
##

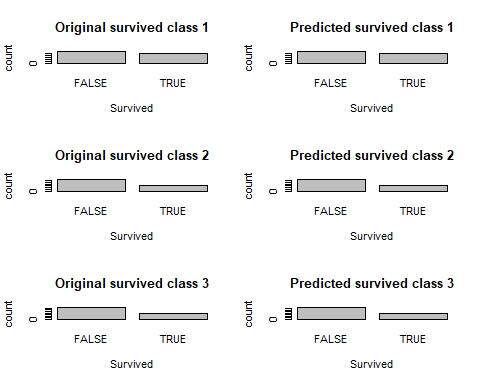
Un cop probat el model amb el conjunt de test, es pot observar que aquest es més acurat que el model d’arbre de classificació anterior a l’hora de predir si un passatger sobreviu o no.

Finalment es mostren les 4 variables utilitzades, comparant els resultats de les dades originals i les dades predites pel model, per comprobar visualment la qualitat del model.

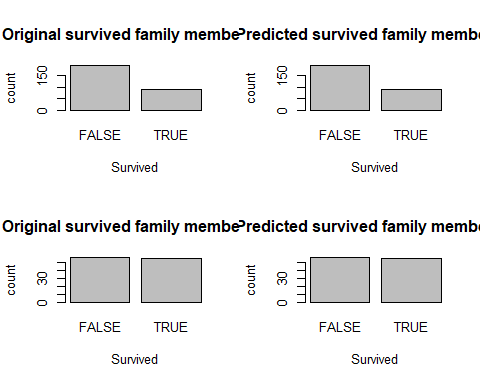
testNaiveBayes$Prediccion <- pred  
par(mfrow=c(2,2))  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$Sex == "male"]), main="Original survived male",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$Sex == "male"]), main="Predicted survived male",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$Sex == "female"]), main="Original survived female",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$Sex == "female"]), main="Predicted survived female",xlab="Survived",ylab="count")



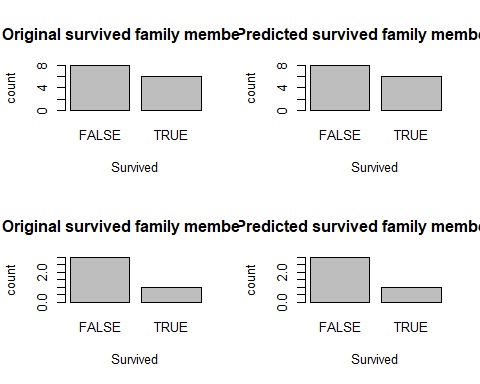
par(mfrow=c(3,2))  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$Pclass == "1"]), main="Original survived class 1",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$Pclass == "1"]), main="Predicted survived class 1",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$Pclass == "2"]), main="Original survived class 2",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$Pclass == "2"]), main="Predicted survived class 2",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$Pclass == "3"]), main="Original survived class 3",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$Pclass == "3"]), main="Predicted survived class 3",xlab="Survived",ylab="count")



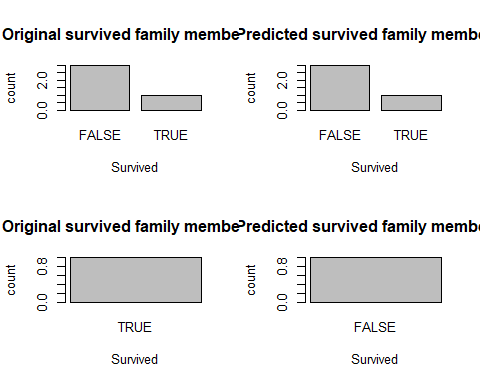
par(mfrow=c(2,2))  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$SibSp == "0"]), main="Original survived family members 0",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$SibSp == "0"]), main="Predicted survived family members 0",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$SibSp == "1"]), main="Original survived family members 1",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$SibSp == "1"]), main="Predicted survived family members 1",xlab="Survived",ylab="count")



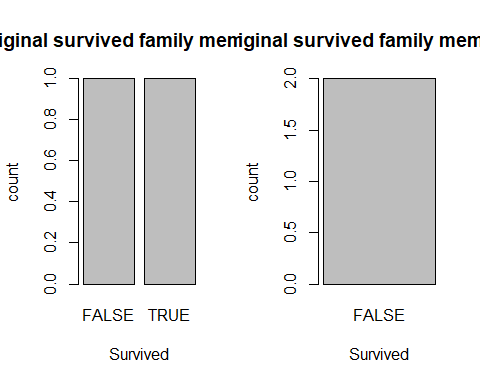
par(mfrow=c(2,2))  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$SibSp == "2"]), main="Original survived family members 2",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$SibSp == "2"]), main="Predicted survived family members 2",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$SibSp == "3"]), main="Original survived family members 3",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$SibSp == "3"]), main="Predicted survived family members 3",xlab="Survived",ylab="count")



par(mfrow=c(2,2))  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$SibSp == "4"]), main="Original survived family members 4",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$SibSp == "4"]), main="Predicted survived family members 4",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$SibSp == "5"]), main="Original survived family members 5",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$SibSp == "5"]), main="Predicted survived family members 5",xlab="Survived",ylab="count")

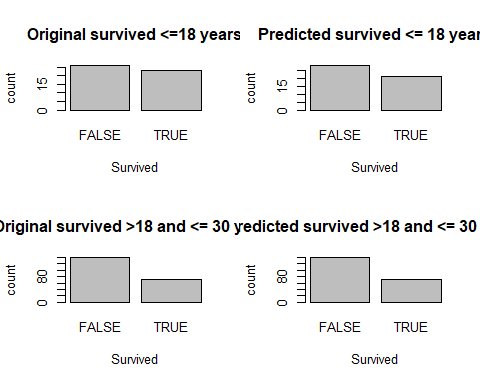


par(mfrow=c(1,2))  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$SibSp == "8"]), main="Original survived family members 8",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$SibSp == "8"]), main="Original survived family members 8",xlab="Survived",ylab="count")

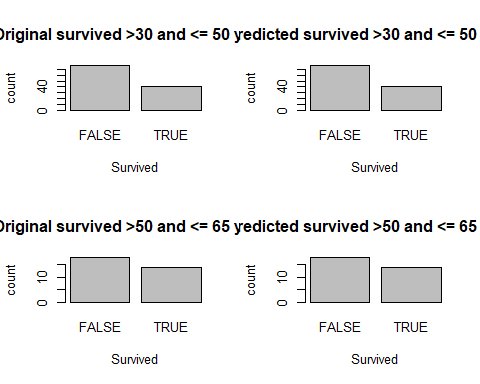


testNaiveBayes$Age[testNaiveBayes$Age<"19"]<- "<=18"  
testNaiveBayes$Age[testNaiveBayes$Age>"65"]<- ">65"  
testNaiveBayes$Age[(testNaiveBayes$Age>"18")&(testNaiveBayes$Age<"31")]<- ">18 and <=30"  
testNaiveBayes$Age[(testNaiveBayes$Age>"30")&(testNaiveBayes$Age<"51")]<- ">30 and <=50"  
testNaiveBayes$Age[(testNaiveBayes$Age>"50")&(testNaiveBayes$Age<"66")]<- ">50 and <=65"

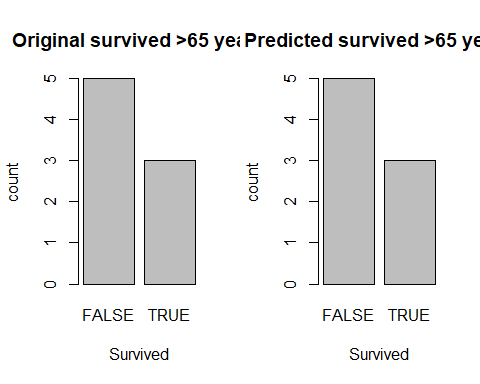
par(mfrow=c(2,2))  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$Age == "<=18"]), main="Original survived <=18 years",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$Age == "<=18"]), main="Predicted survived <= 18 years",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$Age == ">18 and <=30"]), main="Original survived >18 and <= 30 years",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$Age == ">18 and <=30"]), main="Predicted survived >18 and <= 30 years",xlab="Survived",ylab="count")



par(mfrow=c(2,2))  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$Age == ">30 and <=50"]), main="Original survived >30 and <= 50 years",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$Age == ">30 and <=50"]), main="Predicted survived >30 and <= 50 years",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$Age == ">50 and <=65"]), main="Original survived >50 and <= 65 years",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$Age == ">50 and <=65"]), main="Predicted survived >50 and <= 65 years",xlab="Survived",ylab="count")



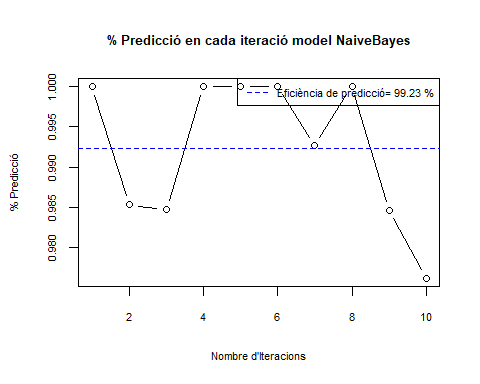
par(mfrow=c(1,2))  
barplot(table(testNaiveBayes$Survived[testNaiveBayes$Age == ">65"]), main="Original survived >65 years",xlab="Survived",ylab="count")  
barplot(table(testNaiveBayes$Prediccio[testNaiveBayes$Age == ">65"]), main="Predicted survived >65 years",xlab="Survived",ylab="count")



Validació creuada

Un cop realitzat l’anàlisi de NaiveBayes, es pot pensar que el conjunt d’entrenament i test no era el més adequat per a realitzar el model. Per tant, es realitzarà un anàlisi de validació creuada per a obtenir un model més acurat a totes les dades existents. Un cop realitzada la validació creuada amb el mètode de NaiveBayes, s’obtenen els següents resultats, obtenint un model amb una predicció del 85.86%.

set.seed(1989)  
folds <- 10  
dataNaiveBayes <- data  
dataNaiveBayes$kfold <- sample(1:folds, nrow(dataNaiveBayes),replace=T)  
Iter <- data.frame(iteracion=NULL,aciertos=NULL)  
for (i in 1:folds)  
{  
 testFold <- subset(dataNaiveBayes, kfold==i)  
 trainingFold <- subset(dataNaiveBayes, !kfold == i)  
 mod <- naiveBayes(Survived ~ ., data = trainingFold)  
 pre <- predict(mod, newdata = testFold)  
 cm <- table(testFold$Survived, pre)  
 precision <- (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] +cm[1,2] + cm[2,1])  
 Iter <- rbind(Iter,data.frame(Iter=i,acierto=precision))  
}  
prom <- format(mean(Iter$acierto, na.rm=TRUE)\*100,digits=4)  
plot(Iter, type = "b", main="% Predicció en cada iteració model NaiveBayes",cex.axis=.7,  
 cex.lab=.7, cex.main=.8,xlab="Nombre d'Iteracions",ylab="% Predicció")  
abline(h=mean(Iter$acierto),col="blue",lty=2)  
legend("topright",legend=paste("Eficiència de predicció=",prom,"%"),  
 col="blue",lty=2,lwd=1,cex=.7, bg=NULL)



Un altre mètode per a realitzar la validació creuada pel mètode NaiveBayes, que dona un valor igual es:

set.seed(1989)  
folds <- createFolds(dataNaiveBayes$Survived, k = 10)  
cvNaiveBayes <- lapply(folds, function(x){  
 training\_fold <- dataNaiveBayes[-x, ]  
 test\_fold <- dataNaiveBayes[x, ]  
 mod <- naiveBayes(Survived ~ ., data = training\_fold)  
 pre <- predict(mod, newdata = test\_fold)  
 cm <- table(test\_fold$Survived, pre)  
 precision <- (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] +cm[1,2] + cm[2,1])  
 return(precision)  
})  
precisionNaiveBayes <- mean(as.numeric(cvNaiveBayes))  
precisionNaiveBayes

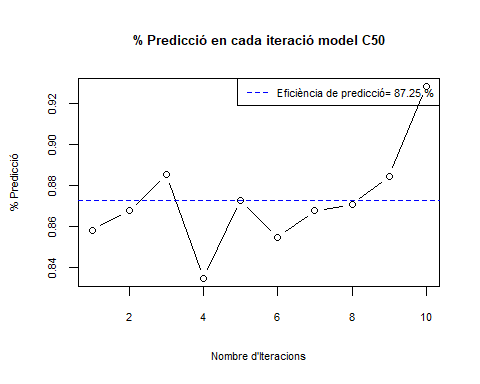
## [1] 0.9931413

Un cop realitzada la validació creuada pel model NaiveBayes, es realitza pel model d’arbre de classificació C50. El resultat obtingut es el següent:

set.seed(1989)  
folds <- 10  
dataC50 <- data  
dataC50$kfold <- sample(1:folds, nrow(dataC50),replace=T)  
vars<-c("Pclass","Sex","Age","SibSp","Parch","Fare")  
str(dataC50[c(vars,"Survived")])

## 'data.frame': 1309 obs. of 7 variables:  
## $ Pclass : Factor w/ 3 levels "1","2","3": 3 1 3 1 3 3 1 3 3 2 ...  
## $ Sex : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...  
## $ Age : num 22 38 26 35 35 20 54 2 27 14 ...  
## $ SibSp : int 1 1 0 1 0 0 0 3 0 1 ...  
## $ Parch : int 0 0 0 0 0 0 0 1 2 0 ...  
## $ Fare : num 7.25 71.28 7.92 53.1 8.05 ...  
## $ Survived: logi FALSE TRUE TRUE TRUE FALSE FALSE ...

dataC50$Survived<-as.factor(dataC50$Survived)  
Iter <- data.frame(iteracion=NULL,aciertos=NULL)  
for (i in 1:folds)  
{  
 testFold <- subset(dataC50, kfold==i)  
 trainingFold <- subset(dataC50, !kfold == i)  
 mod <- C5.0(x=train[,vars],y=train$Survived)  
 pre <- predict(mod, newdata = testFold)  
 cm <- table(testFold$Survived, pre)  
 precision <- (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] +cm[1,2] + cm[2,1])  
 Iter <- rbind(Iter,data.frame(Iter=i,acierto=precision))  
}  
prom <- format(mean(Iter$acierto, na.rm=TRUE)\*100,digits=4)  
plot(Iter, type = "b", main="% Predicció en cada iteració model C50",cex.axis=.7,  
 cex.lab=.7, cex.main=.8,xlab="Nombre d'Iteracions",ylab="% Predicció")  
abline(h=mean(Iter$acierto),col="blue",lty=2)  
legend("topright",legend=paste("Eficiència de predicció=",prom,"%"),  
 col="blue",lty=2,lwd=1,cex=.7, bg=NULL)



Com es pot observar, l’eficiència del model d’arbre de classificació es major que la del classificador bayesià. Per tant, concluïm que, per al nostre conjunt de dades, el model més òptim es el model d’arbre de classificació.

1. Representació dels resultats a partir de taules i gràfiques.

* Durant la resolució dels diferents apartats es van generant les taules i gràfiques necessàries per a la presentació dels resultats obtinguts.

1. Resolució del problema.A partir dels resultats obtinguts, quines són les conclusions? Els resultats permeten respondre al problema?

Un cop realitzat tot l’anàlisi, es pot assegurar que les variables necessàries per a predir si un passatger sobreviu o no són: Sex, Age, Pclass i SibSp. Es pot intuïr que efectivament té sentit que aquestes dades siguin les necessàries, ja que depenent de la edat tindrà més o menys possibilitats de sobreviure. Per una altra banda, si es va donar prioritat a les dones y nens, té sentit que el sexe sigui un factor determinant per a sobreviure. També, depenent de la classe la supervivència varia, ja que es dona proiritat a una classe o a una altra, al igual que al tamany de la família pot afectar. Per contra, variables com on va embarcar o la tarifa del bitllet es pot intuir que no tenen quasi rellevància en la supervivència de la persona.

Generant un model amb aquestes dades som capaços de predir, amb una exactitud del 87%, si un passatger sobreviu o no, utilitzant únicament les dades esmentades.

Per tant, com a conclusió, podem dir que s’ha aconseguit respondre a les preguntes inicials, que era estudiar quines variables tenen més impacte en la taxa de supervivència dels passatgers i, mitjançant aquestes, generar un model adequat per a predir qualsevol altre cas possible.

1. Codi

* Es pot trobar el codi a la carpeta Code del github (<https://github.com/Hakhaz/NetejaValidacioDades>)

CONTRIBUCIONS SIGNATURA  
Investigació prèvia CPM, OFC  
Redacció de les respostes CPM, OFC  
Desenvolupament codi CPM, OFC