Pràctica 2: Com realitzar la neteja i l'anàlisi de dades?

Tipologia i cicle de vida de les dades.

Alumnes: Marc Clupés Peró i Paula Miralles Simó

Data: 13/01/2023

Índex.

1. Descripció del dataset.

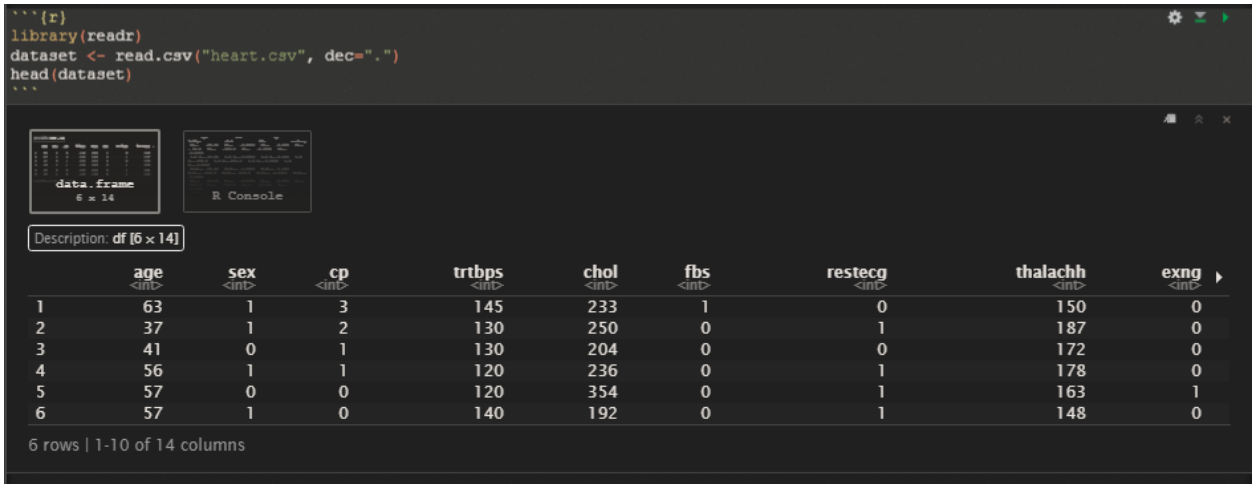
El dataset "Heart Attack Analysis & Prediction" conté informació sobre diferents individus i les seves característiques de salut relacionades amb l'atac de cor. Aquesta informació inclou edat, gènere, pressió arterial, colesterol, hàbits tabàquics i d'exercici, entre altres dades rellevants.

A partir d'aquestes dades, es pot analitzar i predir el risc d'un individu de partir un atac de cor en el futur.

A continuació, expondrem les variables del dataset i el significat que tenen dins del contexte dels atacs al cor:

- ➔ *age* : Edat del pacient
- ➔ *sex*: Sexe del pacient
- ➔ *cp* : tipus de dolor de pit tipus de dolor de pit
  - ◆ Valor 1: angina típica
  - ◆ Valor 2: angina atípica
  - ◆ Valor 3: dolor no anginós
  - ◆ Valor 4: asimptomàtic
- ➔ *trtbps*: pressió arterial en repòs (en mm Hg)
- ➔ *chol* : colesterol en mg/dl obtingut mitjançant el sensor IMC
- ➔ *fbs* : (sucre en sang en dejú > 120 mg/dl) (1 = cert; 0 = fals)
- ➔ *restecg* : resultats electrocardiogràfics en repòs
  - ◆ Valor 0: normal
  - ◆ Valor 1: amb anormalitat de l'ona ST-T (inversions de l'ona T i/o elevació o depressió ST > 0,05 mV)
  - ◆ Valor 2: mostra una hipertròfia ventricular esquerra probable o definitiva segons els criteris d'Estes
- ➔ *thalachh*: freqüència cardíaca màxima aconseguida
- ➔ *exng*: angina induïda per l'exercici (1 = sí; 0 = no)
- ➔ *oldpeak*
- ➔ *slp*
- ➔ *caa*: nombre de vasos cardíacs principals
- ➔ *thall*
- ➔ *output*: 0= menys probabilitat d'infart 1= més probabilitat d'atac cardíac

```r
library(readr)
dataset <- read.csv("heart.csv", dec=".")
head(dataset)
```



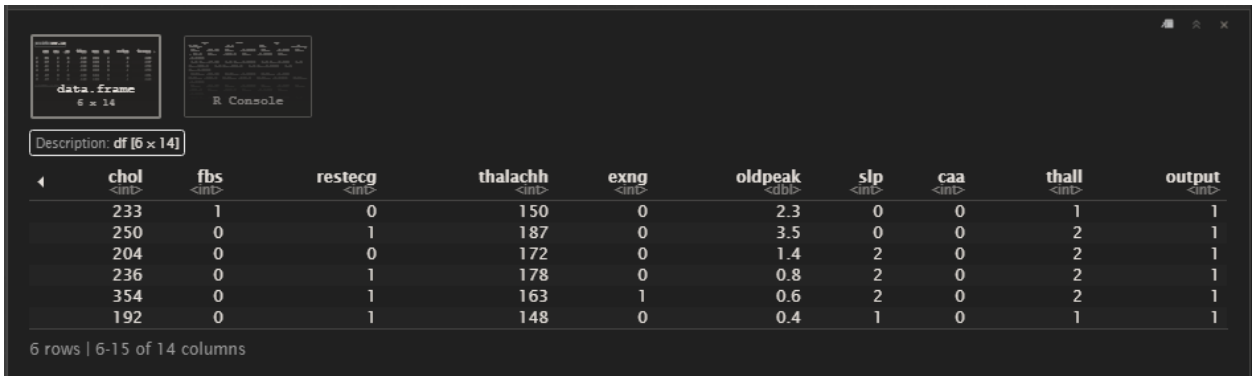
data.frame
6 x 14

R Console

Description: df [6 x 14]

	age <int>	sex <int>	cp <int>	trtbps <int>	chol <int>	fbs <int>	restecg <int>	thalachh <int>	exng <int>
1	63	1	3	145	233	1	0	150	0
2	37	1	2	130	250	0	1	187	0
3	41	0	1	130	204	0	0	172	0
4	56	1	1	120	236	0	1	178	0
5	57	0	0	120	354	0	1	163	1
6	57	1	0	140	192	0	1	148	0

6 rows | 1-10 of 14 columns



data.frame
6 x 14

R Console

Description: df [6 x 14]

	chol <int>	fbs <int>	restecg <int>	thalachh <int>	exng <int>	oldpeak <dbl>	slp <int>	caa <int>	thall <int>	output <int>
	233	1	0	150	0	2.3	0	0	1	1
	250	0	1	187	0	3.5	0	0	2	1
	204	0	0	172	0	1.4	2	0	2	1
	236	0	1	178	0	0.8	2	0	2	1
	354	0	1	163	1	0.6	2	0	2	1
	192	0	1	148	0	0.4	1	0	1	1

6 rows | 6-15 of 14 columns

2. Integració i selecció

Hem decidit crear una subselecció de les dades originals i eliminar les variables "*oldpeak*", "*slp*", "*caa*" i "*thall*" del dataset "Heart Attack Analysis & Prediction" perquè hem considerat que aquestes variables no són necessàries per al nostre estudi, per no ser rellevants per a l'objectiu d'aquest i perquè no disposem de la informació suficient per analitzar-les de manera significativa, és a dir, a la pàgina del dataset, no s'informa el suficient sobre la naturalesa d'aquestes variables.
Aleshores, les variables que podem trobar al dataset actualment són:

```r
eliminar <- c("oldpeak", "slp", "caa", "thall", "output")
dataset_seleccio<- dataset[,!(names(dataset)%in% eliminar)]
head(dataset_seleccio)
```

Description: df [6 × 9]

	age <int>	sex <int>	cp <int>	trtbps <int>	chol <int>	fbs <int>	restecg <int>	thalachh <int>	exng <int>
1	63	1	3	145	233	1	0	150	0
2	37	1	2	130	250	0	1	187	0
3	41	0	1	130	204	0	0	172	0
4	56	1	1	120	236	0	1	178	0
5	57	0	0	120	354	0	1	163	1
6	57	1	0	140	192	0	1	148	0

6 rows

3. Neteja de les dades

3.1. Les dades contenen zeros o elements buits? Gestiona cadascun d'aquests casos.

Utilitzant la funció *samply,* hem si cada una de les variables del dataset té valors nuls:

```r
#Contenen elements nuls
sapply(dataset_seleccio, function(x) any(is.na(x)))
```

age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng
FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE

No hem trobat valors nuls. Això vol dir que totes les dades del dataset estan completes. Això és important perquè els elements nuls poden afectar negativament la qualitat de les dades i les conclusions que es puguin extreure de les mateixes.

En aquest cas, la absència d'elements nuls en el dataset ens permetrà realitzar anàlisis sense haver de preocupar-vos per la possibilitat que hi hagi dades mancants que puguin distorsionar els resultats.

Per als zeros, hem de tenir en compte que moltes de les variables permeten el valor 0, per això, hem de realitzar un estudi sobre quines són aquelles variables que no podrien tindre valors nuls.

En aquest cas i tenint en compte la descripció de cada variable, sabem que aquelles variables que no deurien prendre valors nuls són: *trtbps, chol* i

thalachh (asumirem que una persona té 0 anys si té mesos de vida). Busquem si alguna d'aquestes conté valors zero:

```{r}
#Contenen zeros
variables_sense_zeros <- c("trtbps", "chol", "thalachh")
sapply(dataset_seleccio, function(x) any(x == 0))[(names(dataset_seleccio)%in%
variables_sense_zeros)]
```

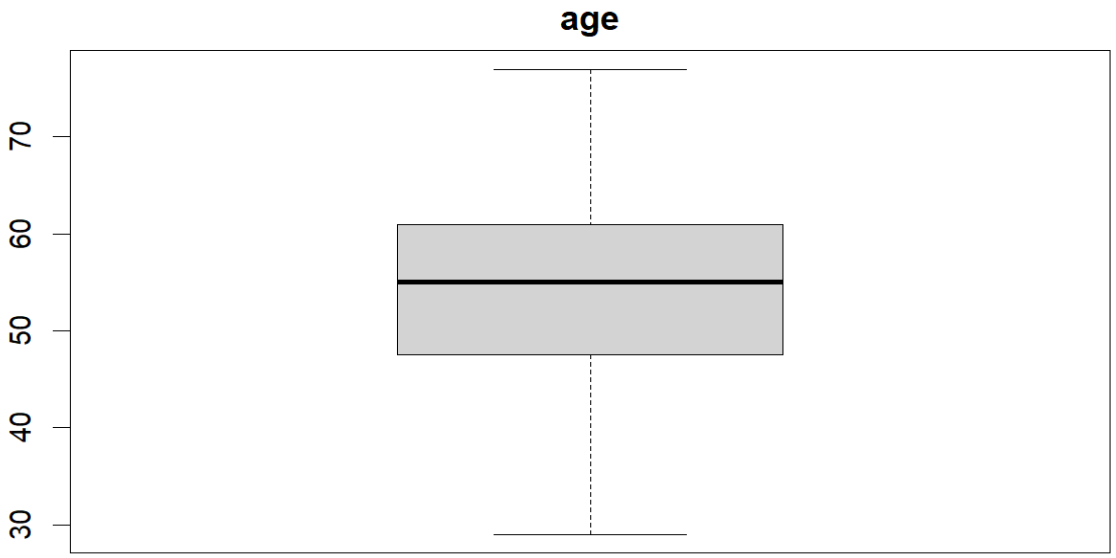
```
  trtbps     chol thalachh
   FALSE    FALSE    FALSE
```

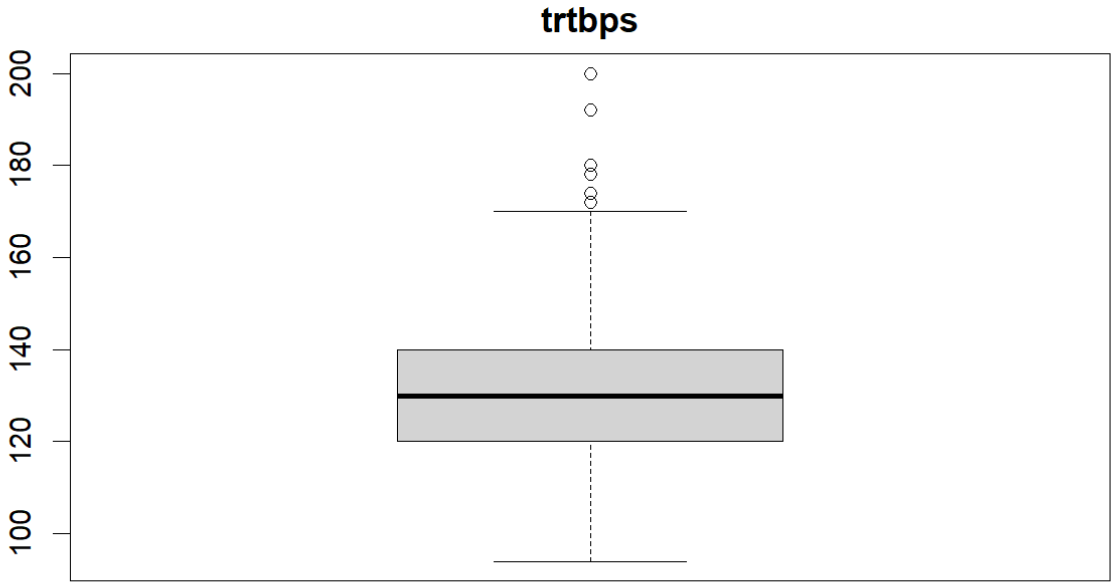
No hem trobat valors zero. Al iguals que ens valors nuls o buits, això és important perquè els elements zeros no previstos poden afectar negativament la qualitat de les dades i les conclusions que es puguin extreure de les mateixes.

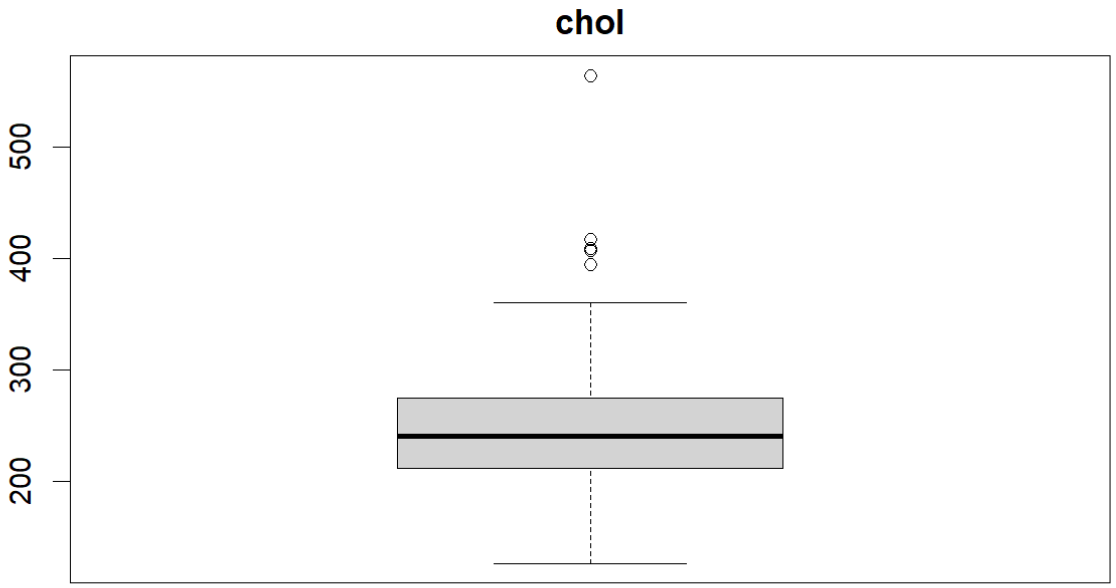
En aquest cas, la absència d'elements zeros no esperats al dataset ens permetrà realitzar anàlisis sense haver de preocupar-vos per la possibilitat que hi hagi dades que puguin distorsionar els resultats.

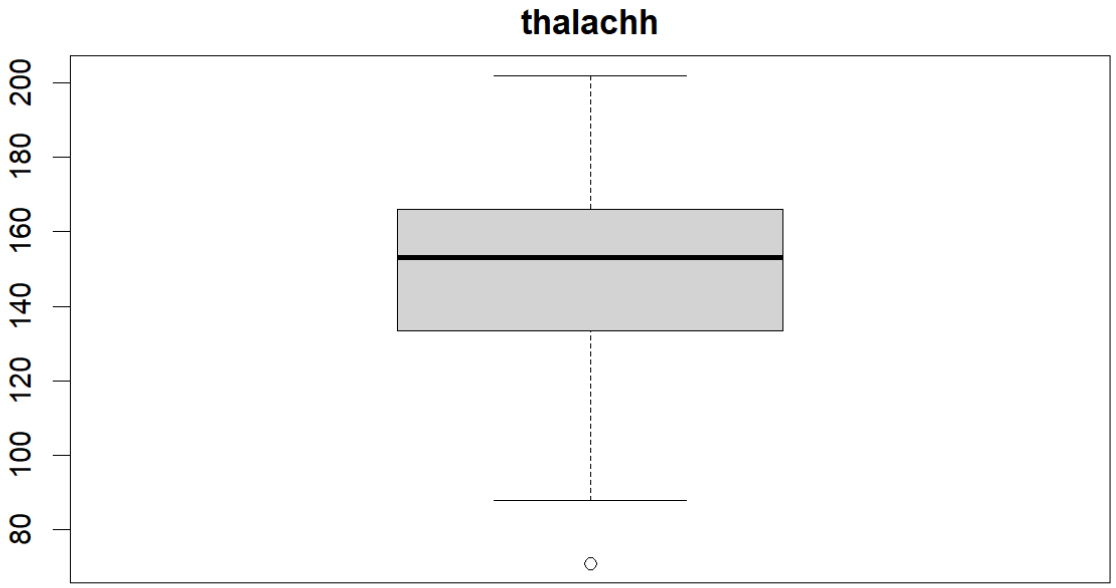
3.2. Identifica i gestiona els valors extrems.

Anem a utilitzar la detecció de valors extrems mitjançant la representació de les variables no categòriques (age, trtbps, chol, thalachh) en un diagrama de caixa:

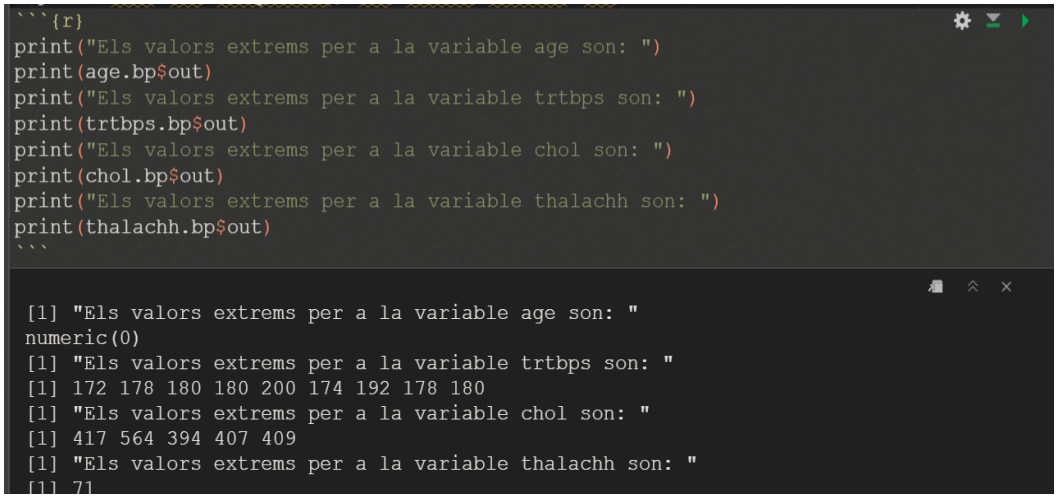

age
30
40
50
60
70


trtbps
100
120
140
160
180
200





I mostrem per pantalla quins valors corresponen a estos outliers:



```r
print("Els valors extrems per a la variable age son: ")
print(age.bp$out)
print("Els valors extrems per a la variable trtbps son: ")
print(trtbps.bp$out)
print("Els valors extrems per a la variable chol son: ")
print(chol.bp$out)
print("Els valors extrems per a la variable thalachh son: ")
print(thalachh.bp$out)
```

```
[1] "Els valors extrems per a la variable age son: "
numeric(0)
[1] "Els valors extrems per a la variable trtbps son: "
[1] 172 178 180 180 200 174 192 178 180
[1] "Els valors extrems per a la variable chol son: "
[1] 417 564 394 407 409
[1] "Els valors extrems per a la variable thalachh son: "
[1] 71
```

En tots aquests casos, aquestes dades són legítimes i formen part de la mostra, de manera que no s'haurà de modificar el conjunt de dades i es contemplaran els outliers a l'anàlisi.

4. Anàlisi de les dades

4.1. Selecció dels grups de dades que es volen analitzar/comparar.

Com hem comentat, el que volem és predir si una persona podria patir o no un atac de cor segons els valors de altres variables. Algunes de les variables interessants a estudiar podrien ser:

- *age* : Edat del pacient
- *sex*: Sexe del pacient
- *cp* : tipus de dolor de pit tipus de dolor de pit
- *trtbps*: pressió arterial en repòs (en mm Hg)
- *chol* : colesterol en mg/dl obtingut mitjançant el sensor IMC
- *fbs* : (sucre en sang en dejú > 120 mg/dl) (1 = cert; 0 = fals)
- *restecg* : resultats electrocardiogràfics en repòs
- *thalachh*: freqüència cardíaca màxima aconseguida
- *exng*: angina induïda per l'exercici (1 = sí; 0 = no)

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

Comprovem la normalitat per a les variables no categòriques:

```
```{r}

print("Test Saphiro per a la variable age son: ")
shapiro.test(dataset_seleccio$age)
print("Test Saphiro per a la variable trtbps son: ")
shapiro.test(dataset_seleccio$trtbps)
print("Test Saphiro per a la variable chol son: ")
shapiro.test(dataset_seleccio$chol)
print("Test Saphiro per a la variable thalachh: ")
shapiro.test(dataset_seleccio$thalachh)
```
```

Executant aquest codi trobem que el p-value és menor de 0.05 per a totes les variables, així que cap d'aquests camps compta amb una distribució normal.

4.3. Aplicació de proves estadístiques per comparar els grups de dades.

Per a la futura avaluació del model, és necessari dividir el conjunt de dades en un conjunt d'entrenament i un conjunt de prova. El conjunt d'entrenament és el subconjunt del conjunt original de dades utilitzat per a construir un primer model; i el conjunt de prova, el subconjunt del conjunt original de dades utilitzat per a avaluar la qualitat del model.

El més correcte serà utilitzar un conjunt de dades diferent del que utilitzem per a construir l'arbre, és a dir, un conjunt diferent del d'entrenament. No hi ha cap proporció fixada respecte al nombre relatiu de components de cada subconjunt, però la més utilitzada acostuma a ser 2/3 per al conjunt d'entrenament i 1/3, per al conjunt de prova.

La variable per la qual classificarem és el camp output, que està en la decena columna. D'aquesta forma, tindrem un conjunt de dades per a l'entrenament i un per a la validació.

```r
dataset_seleccio[ , c(1, 2, 3, 4, 5, 6, 7 , 8, 9 ,10)]
set.seed(1)
df3 <- dataset_seleccio[sample(1:nrow(dataset_seleccio)),]
Y <- df3[,10]
X <- df3[,1:9]
```

Definim un paràmetre que controla el split de manera dinàmica en el test.

```r
split_prop <- 3
indexes = sample(1:nrow(dataset_seleccio),
size=floor(((split_prop-1)/split_prop)*nrow(dataset_seleccio)))
trainX<-X[indexes,]
trainy<-Y[indexes]
testX<-X[-indexes,]
testy<-Y[-indexes]
```

Després d'una extracció aleatòria de casos efectuem una anàlisi de dades mínim per a assegurar-nos de no obtenir classificadors esbiaixats pels valors que conté cada mostra:

```r
summary(trainX);
summary(trainy)
summary(testX)
summary(testy)
```

Seguidament creem i executem l'arbre de decisió usant les dades d'entrenament:

```r
trainy = as.factor(trainy)
model <- C50::C5.0(trainX, trainy,rules=TRUE )
summary(model)
```

Errors mostra el número i percentatge de casos mal classificats en el subconjunt d'entrenament. L'arbre obtingut classifica erròniament 27 dels 202 casos donats, una taxa d'error del 13.4%.

Observant els usos de variables per a les normes creades en el model, podem veure que el camp “cp” (tipus de dolor en el pit) és el que està més relacionat amb el risc d’atac de cor, seguit del gènere i del sucre en sang en dejú:

```
Evaluation on training data (202 cases):

        Rules
  ----------------
    No      Errors

     7   27(13.4%)   <<


    (a)   (b)    <-classified as
   ----  ----
     83     9    (a): class 0
     18    92    (b): class 1


Attribute usage:

100.00% cp
 40.10% sex
 26.73% fbs
 20.30% exng
 19.80% thalachh
 18.32% trtbps
  5.45% age
  4.95% restecg
```

Finalment, utilitzarem aquest model per a predir l'output de les dades de test:

```r
predict_model <- predict( model, testX, type="class" )
print(sprintf("La precisió del model és: %.4f %%",100*sum(predict_model == testy) / length(predict_model)))
```

```
[1] "La precisió del model és: 78.2178 %"
```

Veiem que aquest model té un percentatge d'encert al voltant del 78% en les dades de test.

5. Resolució del problema.

En aquesta pràctica hem analitzat i netejat les dades que provenien del dataset "Heart Attack Analysis & Prediction" contenint informació sobre diferents individus i les seves característiques de salut relacionades amb l'atac de cor. Hem netejat aquestes dades i a partir d'elles hem fet un model per a predir si una persona tindrà risc d'atac de cor basant-nos en les variables del dataset.

6. Signatures

Investigació Prèvia	Paula Miralles Simó, Marc Clupés Peró
Redacció de les respostes	Paula Miralles Simó, Marc Clupés Peró
Desenvolupament del codi	Paula Miralles Simó, Marc Clupés Peró
Participació al vídeo	Paula Miralles Simó, Marc Clupés Peró

7. Codi en R

```
# Dataset

```{r}
library(readr)
dataset <- read.csv("heart.csv", dec=".")
head(dataset)
```

Integració i selecció de les dades

```{r}
eliminar <- c("oldpeak", "slp", "caa", "thall")
dataset_seleccio<- dataset[,!(names(dataset)%in% eliminar)]
head(dataset_seleccio)
```


Neteja de les dades.
Les dades contenen zeros o elements buits?

```{r}
#Contenen elements nuls
sapply(dataset_seleccio, function(x) any(is.na(x)))
```

```{r}
#Contenen zeros
variables_sense_zeros <- c("trtbps", "chol", "thalachh")
sapply(dataset_seleccio, function(x) any(x ==
0))[(names(dataset_seleccio)%in% variables_sense_zeros)]
```

Identifica i gestiona els valors extrems.

Recordem que hem determinat que les variables no categòriques són:
age, trtbps, chol, thalachh
```{r}
age.bp<-boxplot(dataset_seleccio$age ,main="age")
trtbps.bp<-boxplot(dataset_seleccio$trtbps ,main="trtbps")
chol.bp<-boxplot(dataset_seleccio$chol ,main="chol")
thalachh.bp<-boxplot(dataset_seleccio$thalachh ,main="thalachh")
```
```

I per a cada una d'aquestes, els valors extrems són:
```{r}
print("Els valors extrems per a la variable age son: ")
print(age.bp$out)
print("Els valors extrems per a la variable trtbps son: ")
print(trtbps.bp$out)
print("Els valors extrems per a la variable chol son: ")
print(chol.bp$out)
print("Els valors extrems per a la variable thalachh son: ")
print(thalachh.bp$out)
```
#Comprovació de la normalitat i homogeneïtat de la variància.

```{r}

print("Test Saphiro per a la variable age son: ")
shapiro.test(dataset_seleccio$age)
print("Test Saphiro per a la variable trtbps son: ")
shapiro.test(dataset_seleccio$trtbps)
print("Test Saphiro per a la variable chol son: ")
shapiro.test(dataset_seleccio$chol)
print("Test Saphiro per a la variable thalachh: ")
shapiro.test(dataset_seleccio$thalachh)
```
Veiem que cap d'aquests camps compta amb una distribució normal, ja que la p-value per a tots ells
és menor de 0.05

Per a la homogeneïtat de la variància:

```{r}

print("Test Fligner per a la variables age i trtbps: ")
fligner.test(age ~ trtbps,  data = dataset_seleccio)
print("Test Fligner per a la variables age i chol: ")
fligner.test(age ~ chol,  data = dataset_seleccio)
print("Test Fligner per a la variables age i thalachh: ")
fligner.test(age ~ thalachh,  data = dataset_seleccio)
print("Test Fligner per a la variables age i trtbps: ")
fligner.test(age ~ trtbps,  data = dataset_seleccio)
```

Preparació de les dades per al model

```{r}
```

```
dataset_seleccio[ , c(1, 2, 3, 4, 5, 6, 7 , 8, 9 ,10)]
set.seed(1)
df3 <- dataset_seleccio[sample(1:nrow(dataset_seleccio)),]
Y <- df3[,10]
X <- df3[,1:9]
X
Y
```

```{r}
split_prop <- 3
indexes = sample(1:nrow(dataset_seleccio),
size=floor(((split_prop-1)/split_prop)*nrow(dataset_seleccio)))
trainX<-X[indexes,]
trainy<-Y[indexes]
testX<-X[-indexes,]
testy<-Y[-indexes]
```

```{r}
summary(trainX);
summary(trainy)
summary(testX)
summary(testy)
```

Creació del model, qualitat del model i extracció de regles

```{r}
install.packages("inum")
trainy = as.factor(trainy)
model <- C50::C5.0(trainX, trainy,rules=TRUE )
summary(model)
```

```{r}
predict_model <- predict( model, testX, type="class" )
print(sprintf("La precisió del model és: %.4f
%%",100*sum(predict_model == testy) / length(predict_model)))
```

8. Vídeo

https://drive.google.com/file/d/1Lw4Z1cgSRCpY92fmBjd9jyezPybqJ3bA/view?usp=sharing