Apprentissage statistique TP1
Proposition de corrigé

1 Diagnostiquer le cancer

Dans cet exercice, on s'intéresse au diagnostic du cancer (présence ou absence) à partir de la concentration de l'hormone PSA (Prostate Specific Antigen) dans le sang (concentration à valeur réelle). On est donc amené à chercher une fonction $f : \mathbb{R} \longrightarrow \{0, 1\}$.

1.1 Choix d'une fonction de prédiction

Données d'apprentissage

Code chunk 1: «cancer.data»

```
psa, cancer
2.0, 0
4.4, 0
12.1, 1
1.4, 0
16.6, 1
0.4, 0
15.8, 1
4.0, 0
13.9, 0
13.1, 0
```

On choisit la fonction f définie par $f(x) = \mathbb{1}_{\{x>10\}}$.

Code chunk 2: «fonction»

```
f = function (x) {
  x > 10
}
```

1.2 Analyse de la fonction choisie

Le problème considéré est un problème de classification, on choisit naturellement d'évaluer la performance de la fonction f à l'aide d'un nombre d'erreurs ou d'une probabilité d'erreur.

Chargement des données

Code chunk 3: «risque»

```
data = read.csv("cancer.data")
print.data.frame(data,row.names = F)
```

Interpret with R

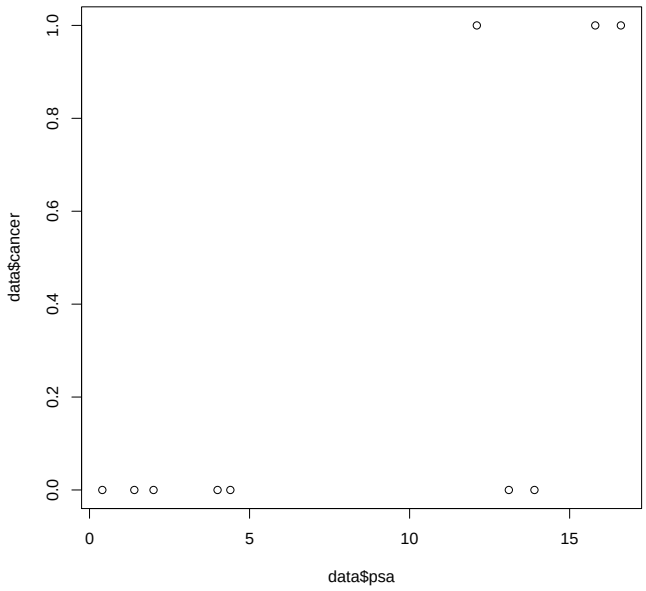
```
  psa cancer
  2.0      0
  4.4      0
 12.1      1
  1.4      0
 16.6      1
  0.4      0
 15.8      1
  4.0      0
 13.9      0
 13.1      0
```

Représentation graphique

Code chunk 4: «risque (part 2)»

```
pdf(file = "cancer_fig1.pdf")
plot(data$psa,data$cancer)
i = dev.off()
```

Interpret with R



Calcul du risque empirique (sur les données d'apprentissage)

Code chunk 5: «risque (part 3)»

```
risque = function(data)
{
  sum (f(data$psa) != data$cancer)
}

cat("Predictions de f")
f(data$psa)
cat("Nombre d'erreurs = ", risque(data),"\n")
```

Interpret with R

```
Predictions de f [1] FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE  TRUE
Nombre d'erreurs =  2
```

Données de test

Code chunk 6: «cancer.test»

```
psa, cancer
15.2, 1
8,1
10.6, 1
9.4, 1
5.5, 0
19.8, 1
13.8, 1
5.3, 0
19.2, 1
11.3, 1
```

Calcul du risque sur les données de test

Code chunk 7: «risque (part 4)»

```
test = read.csv("cancer.test")
cat("Nombre d'erreurs = ", risque(test),"\n")
```

Interpret with R

```
Nombre d'erreurs =  2
```

1.3 Machine apprenante

Pour cet exemple, on a choisi une règle de décision simple, correspondant aux fonctions de la forme $f(x) = \mathbb{1}_{\{x>\alpha\}}$ où α est un seuil. La machine apprenante fournit un algorithme ou une méthode permettant de choisir f étant fourni un jeu de données. En l'occurence, nous testons toutes les fonctions possibles, et nous choisissons celle qui minimise le risque empirique.

On commence par définir une fonction qui calcule le risque en fonction de la valeur du seuil choisie.

Code chunk 8: «machine»

```
risque = function(data,seuil)
{
  prediction = data$psa > seuil
  sum (prediction != data$cancer)
}
```

Interpret with R

On calcule maintenant ce risque pour chacun des seuils possibles.

Code chunk 9: «machine (part 2)»

```
temp = sort(data$psa)
seuils_possibles = (c(0,temp) + c(temp,20))/2
cat(seuils_possibles,"\n")

risques = NULL
for (i in seq(along = seuils_possibles)) {
  risques[i] = risque(data,seuils_possibles[i])
}
cat(risques,"\n")

i = which.min(risques)
cat("Le minimum est obtenu pour",risques[i],"erreurs, avec un seuil de",seuils_possibles[i],"\n")
```

Interpret with R

```
0.2 0.9 1.7 3 4.2 8.25 12.6 13.5 14.85 16.2 18.3
7 6 5 4 3 2 3 2 1 2 3
Le minimum est obtenu pour 1 erreurs, avec un seuil de 14.85
```

Malheureusement, cette fonction fait beaucoup d'erreurs sur les données de test.

Code chunk 10: «machine (part 3)»

```
cat("Nombre d'erreurs = ", risque(test,seuils_possibles[i]),"\n")
```

Interpret with R

```
Nombre d'erreurs =  5
```

1.4 Plus proche voisin

L'algorithme des plus proches voisins peut être implémenté en R de la façon suivante :

Code chunk 11: «ppv»

```
ppv0 = function(data, z) {
  i = which.min(abs(data$psa-z))
  data$cancer[i]
}
ppv = Vectorize(ppv0,vectorize.args="z")
```

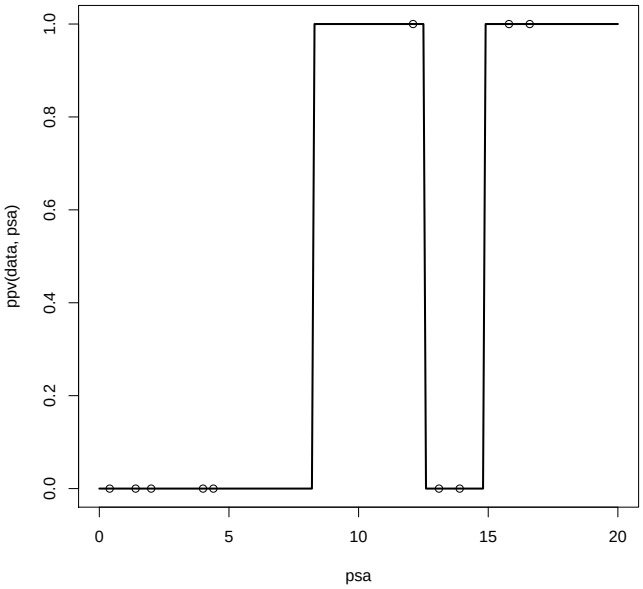
Interpret with R

La fonction de prédiction est une fonction $f : \mathbb{R} \longrightarrow \{0, 1\}$, que l'on peut représenter sur la figure suivante :

Code chunk 12: «ppv (part 2)»

```
data = read.csv("cancer.data")
psa = seq(0,20,by=0.1)
pdf(file = "cancer_fig2.pdf")
plot(psa,ppv(data,psa),typ="l",lwd=2)
points(data$psa,data$cancer)
i = dev.off()
```

Interpret with R



2 Ville de résidence

2.1 Données

Code chunk 13: «ville.data»

```
age,taille,ville
25, 178, bobigny
24, 168, bobigny
32, 163, bobigny
30, 174, bobigny
19, 161, bobigny
32, 176, versailles
6, 177, versailles
45, 167, versailles
22, 166, versailles
33, 183, versailles
40, 170, paris
28, 145, paris
75, 178, paris
53, 146, paris
66, 151, paris
```

2.2 Plus proche voisin

L'algorithme des plus proches voisins peut être implémenté en R de la façon suivante :

Code chunk 14: «ppv (part 3)»

```
ppv0 = function(age, taille,data) {
  i = which.min(sqrt((data$age-age)^2 + (data$taille-taille)^2))
  data$ville[i]
}
ppv = Vectorize(ppv0,vectorize.args=c("age","taille"))
```

Interpret with R

La fonction de prédiction est une fonction $f : \mathbb{R}^2 \longrightarrow \{\text{Bobigny}, \text{Versailles}, \text{Paris}\}$, que l'on peut représenter sur la figure suivante :

Code chunk 15: «ppv (part 4)»

```
data = read.csv("ville.data")
ll = levels(data$ville)
age = seq(0,80)
taille = seq(130,190)
ville = outer(age,taille,ppv,data)
ville0 = matrix(as.numeric(ville),nrow=length(age),ncol=length(taille))
pdf(file = "ville_fig2.pdf")
contour(age,taille,ville0,levels=c(1.5,2.5),xlab="age",ylab="taille")
masque = which(data$ville==ll[1])
points(data$age[masque],data$taille[masque],col="red")
masque = which(data$ville==ll[2])
points(data$age[masque],data$taille[masque],col="green")
masque = which(data$ville==ll[3])
points(data$age[masque],data$taille[masque],col="blue")
i = dev.off()
```

Interpret with R