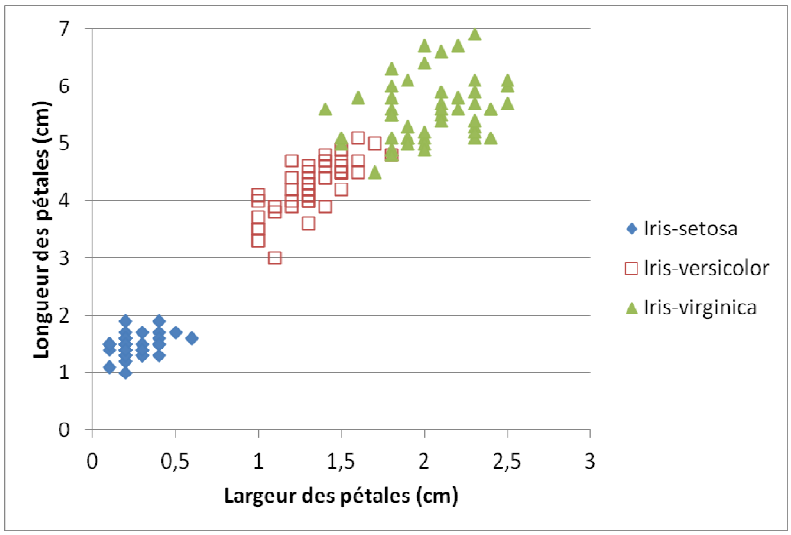
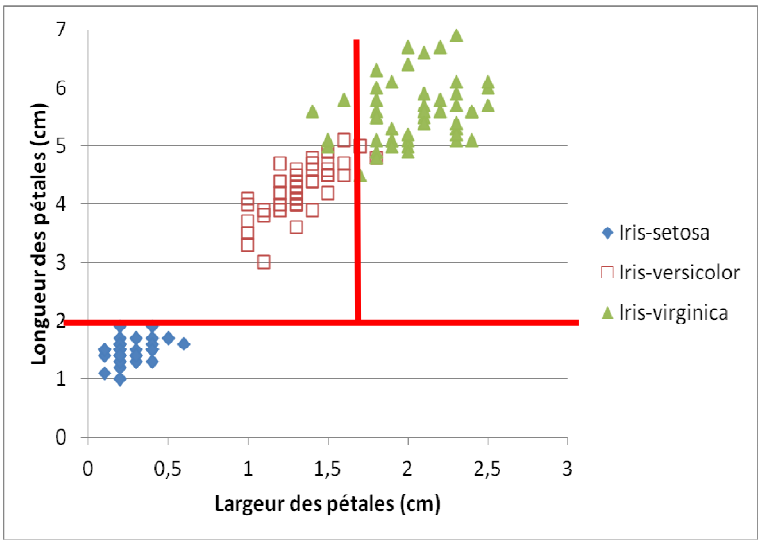
CY TECH	**ING2-GI : MACHINE LEARNING** **ARBRE DE DECISION**
Rédigé par : Astrid Jourdan	A l'intention de : Elèves d'ING2-GI
Durée : 5h	Dernière modification : 17/10/2022

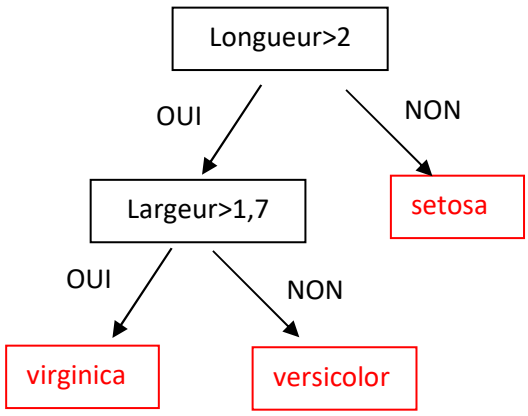
Exercice 1

Considérons le fameux jeu de données « Iris » (http://archive.ics.uci.edu/ml/datasets/Iris) représenté ci-dessous. Il s'agit de trois types d'Iris caractérisés entre autre par la longueur et la largeur des pétales.

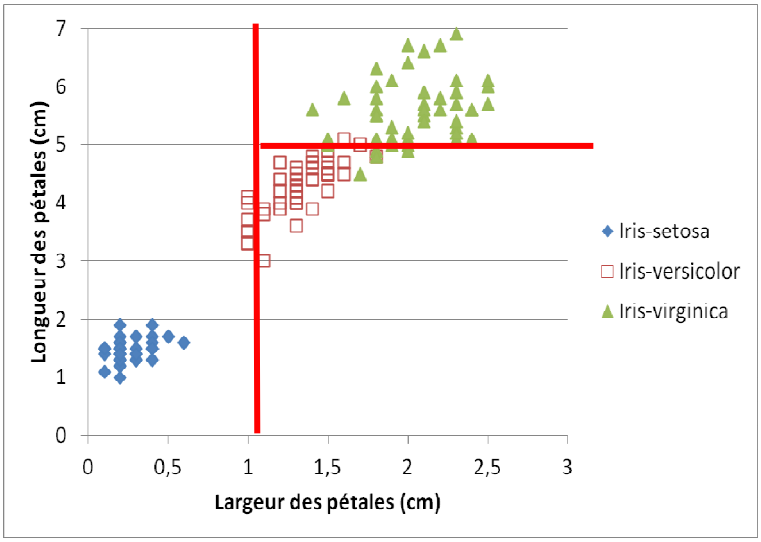


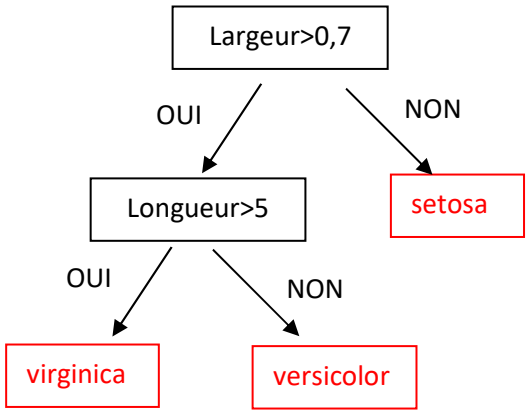
1) Représentez sur le graphique la règle : *Si la longueur des pétales est inférieure à 2 alors l'iris est de type Setosa*.
2) Dans le cas où la longueur des pétales est supérieure à 2, ajoutez sur le graphique la règle : *Si la largeur des pétales est supérieure à 1,7 alors l'iris est de type Virginia, sinon l'iris est de type Versicolor*.
3) Représenter l'arbre correspondant à ces règles.





4) Refaire la même chose en inversant l'ordre de longueur et largeur.





Exercice 2

Considérons l'exemple très simple sur le football.

X_1=Match à domicile ?	X_2=Balance positive ?	X_3=Mauvaises conditions climatiques ?	X_4=Match précédent gagné ?	Y=Match gagné
V	V	F	F	V
F	F	V	V	V
V	V	V	F	V
V	V	F	V	V
F	V	V	V	F
F	F	V	F	F
V	F	F	V	F
V	F	V	F	F

Quelle variable sera choisie à la racine de l'arbre si l'on souhaite maximiser le gain à l'aide de l'entropie ?

- Ent(E)=-1/2log$_2$(1/2) -1/2log$_2$(1/2)=1 (valeur maximale car distribution uniforme)
  G(X_1)=Ent(E)-P(X_1=V)*Ent(X_1=V)-P(X_1=F)*Ent(X_1=F)
  Ent(X_1=V)=-3/5log$_2$(3/5)-2/5log$_2$(2/5)=0,97
  Ent(X_1=F)=-1/3log$_2$(1/3)-2/3log$_2$(2/3)=0,92
  G(X_1)=1-(5/8)*0,971-(3/8)*0,92=0,05

- G(X_2)=Ent(E)-P(X_2=V)*Ent(X_2=V)-P(X_2=F)*Ent(X_2=F)
  Ent(X_2=V)=-3/4log$_2$(3/4)-1/4log$_2$(1/4)=0,811
  Ent(X_2=F)=-1/4log$_2$(1/4)-3/4log$_2$(3/4)=0,811
  G(X_2)=1-(1/2)*0,811-(1/2)*0,811=0,189

Entre ces deux variables, on choisit X_2 qui apporte plus d'information.
Faire les mêmes calculs avec les autres variables pour choisir la meilleure.

Exercice 2

On considère des images en noir et blanc codées sur 4 pixels. Chaque image est donc codée par un élément $(x_1,x_2,x_3,x_4)\in\{0,1\}^4$, où les pixels noirs sont notés 1 et les pixels blancs sont notés 0 et sont numérotés dans l'ordre

x_1	x_2
x_3	x_4

On dispose de la base d'apprentissage ci-dessous pour laquelle les images ont été réparties selon deux classes.

Classe 1 ***Classe 2***

On souhaite construire un arbre de décision sur cet échantillon avec comme variables explicatives (attributs), x_1, x_2, x_3 et x_4.

1) Ecrire la base d'apprentissage sous la forme d'un tableau

Image	X1	X2	X3	X4	Classe
1	1	0	1	1	C1
2	1	0	0	1	C1
3	0	0	0	1	C1
4	1	0	1	0	C1
5	1	0	0	1	C1
6	1	0	0	1	C1
7	0	1	1	0	C2
8	0	0	1	0	C2
9	0	1	1	0	C2
10	1	1	1	0	C2
11	0	1	1	0	C2
12	1	1	0	0	C2

Base d'apprentissage

2) Quelle variable sera choisie à la racine de l'arbre si l'on souhaite maximiser le gain à l'aide de l'indice de Gini ?

On a

$$Gini(E)=(1/2)(1-1/2)+ (1/2)(1-1/2)=1/2$$

Donc,

$$\forall i\in\{1,\dots,4\},\ G(X_i)=(1/2)-P(X_i=1)*Gini(X_i=1)- P(X_i=0)*Gini(X_i=0)$$

où

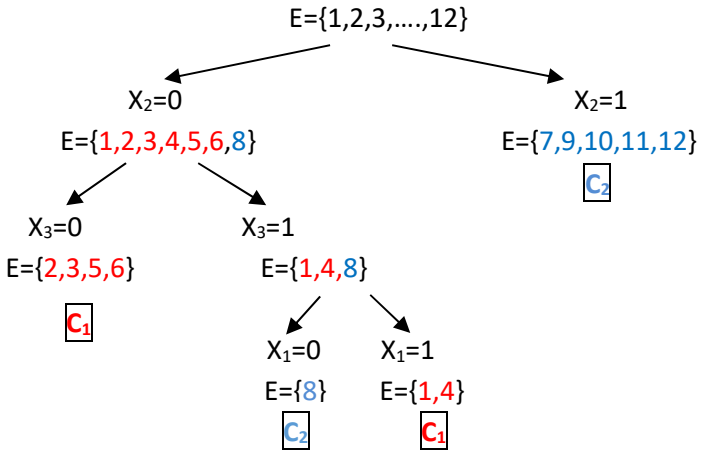
$$Gini(X_i=k)=P(Y=C_1|X_i=k)*[1-P(Y=C_1|X_i=k)]+ P(Y=C_2|X_i=k)*[1-P(Y=C_2|X_i=k)]\ ,\ k\in\{0,1\}$$

On obtient, $G(X_1)=0.129$, $G(X_2)=0.36$, $G(X_3)=0.13$, $G(X_4)=0.36$.

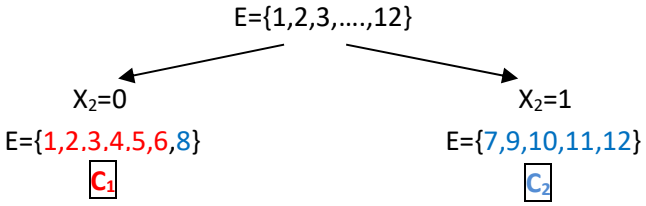
On commence donc l'arbre avec la variable X_2 ou la variable X_4.

3) Représenter ensuite l'arbre T avec un partage des nœuds suivant les variables dans l'ordre suivant : X_2, X_3, X_1 .

On commence avec X_2

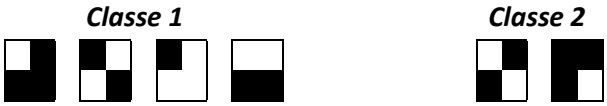


4) Proposer un arbre élagué. Calculer l'erreur d'ajustement pour chacun des arbres.



L'arbre complet a une erreur d'ajustement nulle alors que l'arbre élagué à une erreur d'ajustement de 1/12.

5) On considère l'ensemble de test suivant



Calculer l'erreur sur la base test pour l'arbre complet et l'arbre élagué.

Image	X1	X2	X3	X4	Classe	Prévision arbre complet	Prévision avec arbre élagué
1	0	1	1	1	C1	C2	C2
2	1	0	0	1	C1	C1	C1
3	1	0	0	0	C1	C1	C1
4	0	0	1	1	C1	C2	C1
5	0	1	1	0	C2	C2	C2
6	1	1	1	0	C2	C2	C2
					Erreur	2	1

Base de test

L'arbre entier et l'arbre avec une seule variable explicative donne le meilleur résultat. On préférera l'arbre avec une seule variable explicative car les feuilles ont un effectif plus important. C'est embêtant d'avoir une feuille avec une seule observation.

Exercice 3

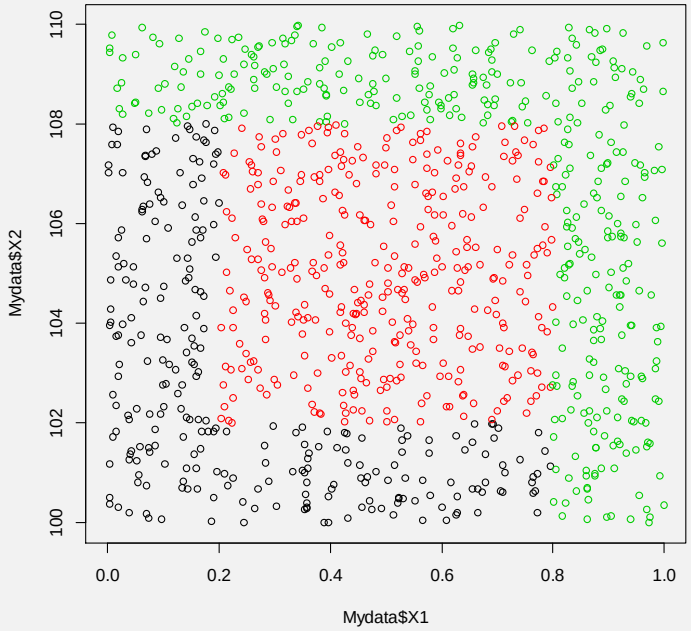
L'objectif du tutoriel http://apiacoa.org/blog/2014/02/initiation-a-rpart.fr.html est de dérouler un arbre de décision avec R.

1) A la suite de ce tutoriel, construire un arbre de décision avec R sur le jeu de données « Iris ».

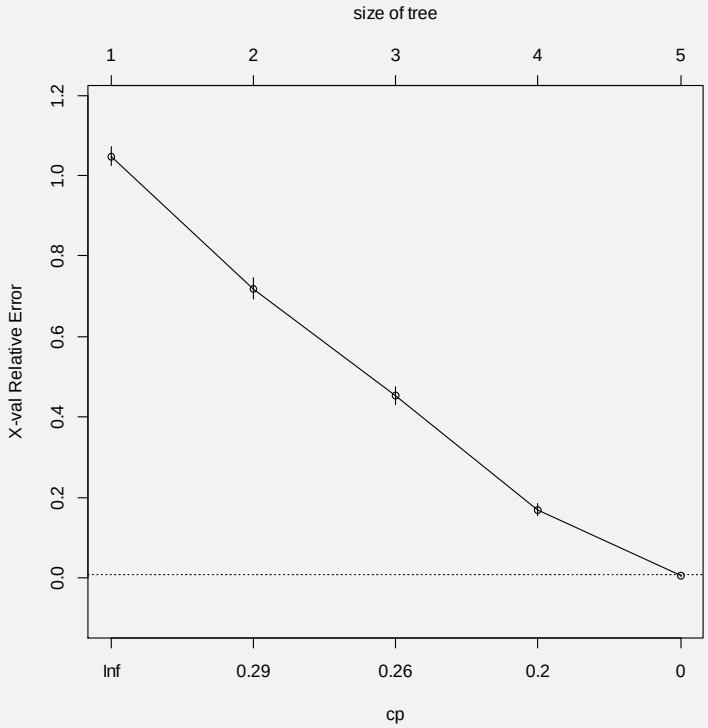
- Construire une base d'apprentissage et une base test.
- Ajuster l'arbre de décision le plus optimal.
- Calculer l'erreur d'apprentissage et l'erreur de prévision.

2) Soient les deux jeux de données simulées : Test_Classif_dpt.txt et Test_Classif_Correl.txt
   - Représenter le nuage de points avec la couleur des classes pour les deux jeux. Un arbre de décision peut-il séparer ces classes ?
   - Comparer la complexité des arbres construits dans chacun des cas.

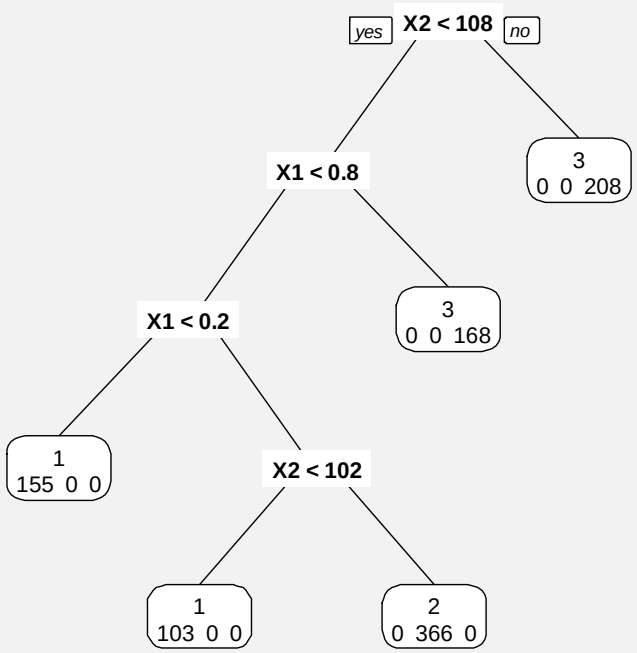
```
Mydata=read.table("Test_Classif_dpt.txt",header=T)
plot(Mydata$X1,Mydata$X2, col=Mydata$classes)
```



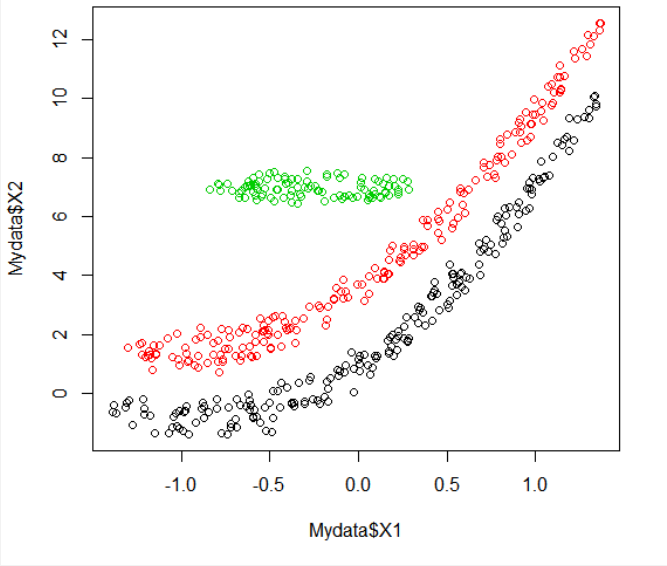

```
# Transformation de la variable cible en variable qualitative
Mydata$classes=as.factor(Mydata$classes)

# Ajustement de l'arbre de décision
library("rpart")
library("rpart.plot")
arbre=rpart(classes~X1+X2, data=Mydata,control=rpart.control(minsplit=5,cp=0))
plotcp(arbre)
```

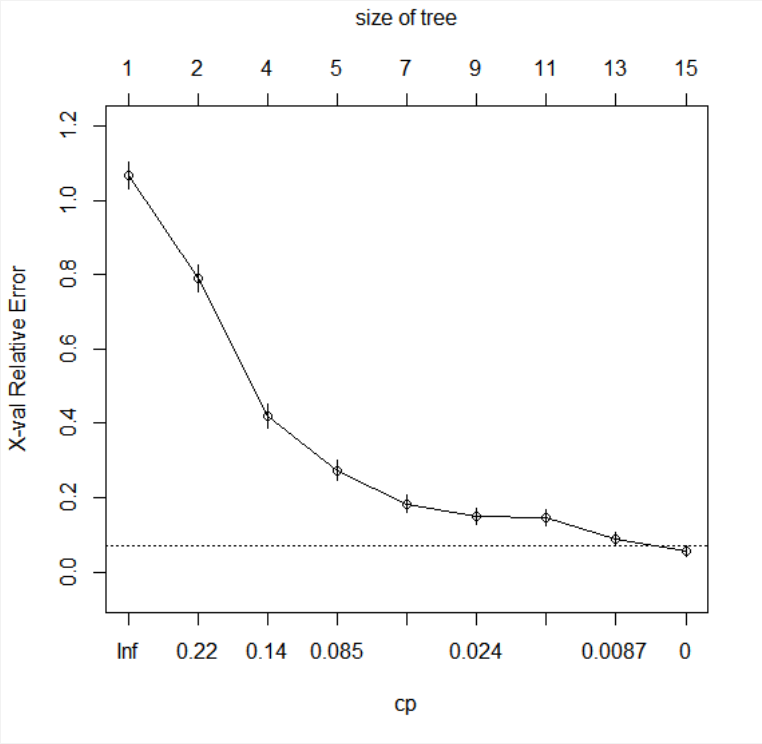



```
# Pas besoin d'élaguer l'arbre
# Affichage de l'arbre
prp(arbre,extra=1)
```

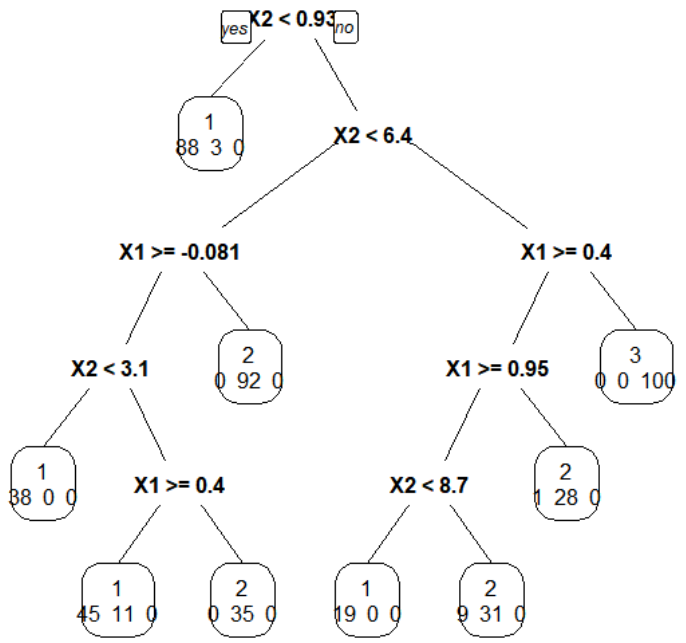



```
# On constate qu'il n'y a aucune erreur et on obtient un arbre assez simple.

Mydata=read.table("Test_Classif_Correl.txt",header=T)
plot(Mydata$X1,Mydata$X2, col=Mydata$classes)
```






```
# On élage l'arbre pour un indice cp=0.024
arbre2=prune(arbre,cp=0.024)
prp(arbre2,extra=1)
```



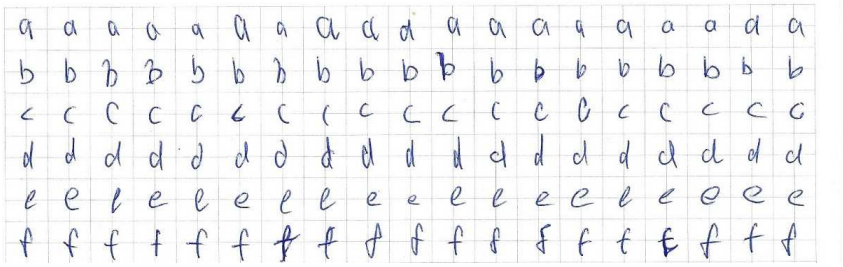

```
# On obtient un arbre plus complexe et pourtant il n'y a pas sur-ajustement. Etant donné qu'un arbre crée des frontières horizontales et verticales, il faut beaucoup de nœuds pour arriver à faire une frontière oblique comme cela est le cas entre la classe 1 (noir) et 2 (rouge)
```

Exercice 5

Tutoriel

Exercice 6

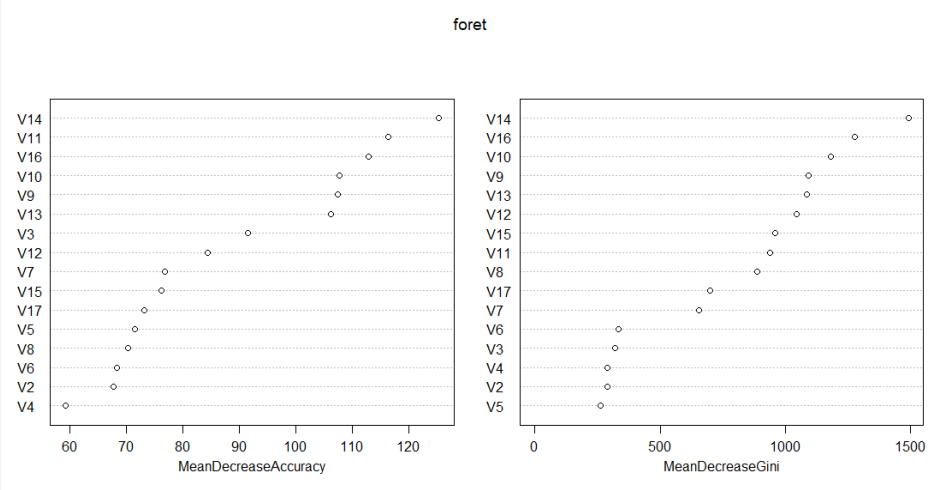
Utiliser la méthode des forêts aléatoires pour faire de la reconnaissance de caractères avec le jeu de données : http://archive.ics.uci.edu/ml/datasets/Letter+Recognition



(a) Créer une base d'apprentissage (2/3) et une base test (1/3).
(b) Construire le modèle sur la base d'apprentissage.
(c) Quelles sont les variables discriminantes ou les variables ayant peu d'impact ?
(d) Calculer l'erreur de prévision et la comparer avec l'erreur oob.
(e) Faire varier les paramètres de l'arbre (ntree, mtry,…). Comparer les erreurs de prévision et en déduire quels paramètres à une influence sur la qualité du modèle.

```
Mydata=read.table("letter-recognition.data",sep=",",header=F)

# a. Construction base d'apprentissage (2/3) + test (1/3)
n=nrow(Mydata) # Taille des données
```

```
ntrain=floor(2*n/3) # taille base d'apprentissage
ntest=n-ntrain # taille base test

index=sample(1:n,ntrain,replace=F) # tirage alétoire sans remise de ntrain éléments parmi n
train=Mydata[index,] # base d'apprentissage
test=Mydata[-index,] # base test

# b. Construction de la Foret
foret=randomForest(V1~.,train,importance=T) # Construction de la forêt aléatoire avec le
paramétrage par défaut de R
show(foret)

# c. Importance des variables
varImpPlot(foret)
```



On note que la variable la plus discriminante est V14 (x-ege mean edge count left to right) alors que les variables liées à la box entourant le caractère (V2 à V8) n'ont pas beaucoup d'impact sur la classification de la variable cible.

```
# d. Base test
prev=predict(foret,test) # vecteur des prévisions du modèle pour les points de la base test
MatConf=table(test$V1,prev) # Matrice de confusion
tb=100*sum(diag(MatConf))/sum(MatConf) # Taux de bien classés
```

On note que les erreurs oob et de prévision sont cohérentes, donc pas de sur-ajustement.

```
# e. hyperparamétrage
```