

Module 3 - Analyse statistique avec R - Séance 2 DUBii 2019

Leslie REGAD

2019-02-26

Plan

Méthodes multivariées

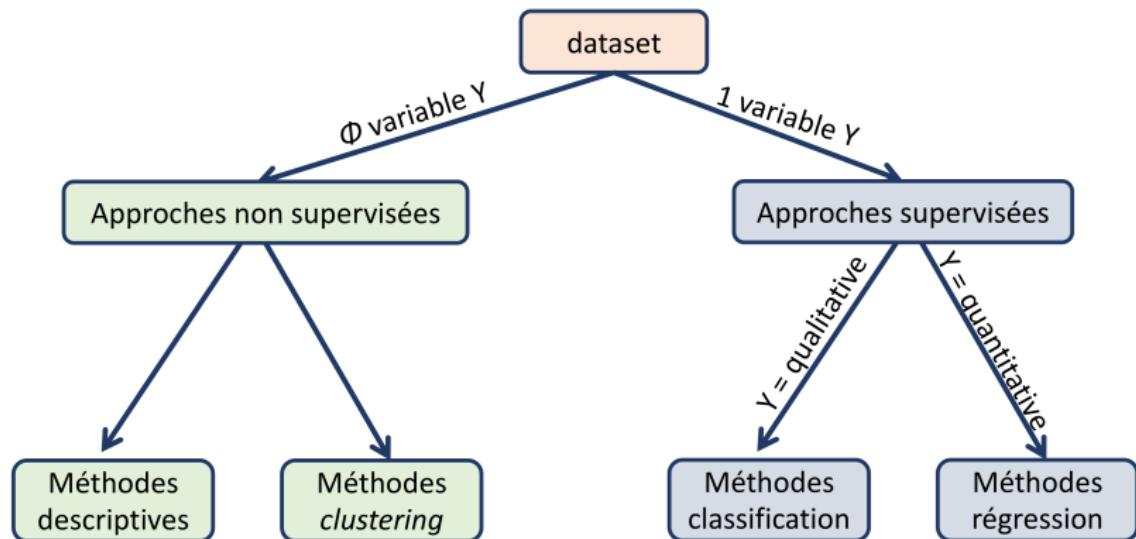


Figure 1:

Prédire la variété des iris en fonction de leur description

- ▶ Travaille avec les données d'iris

```
str(iris)
```

```
'data.frame': 150 obs. of 5 variables:  
 $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...  
 $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...  
 $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.3 ...  
 $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...  
 $ Species      : Factor w/ 3 levels "setosa","versicolor",...
```

- ▶ Variable à prédire (Y) : Species
- ▶ Variables descriptives (X_i) : Sepal.Length, Sepal.Width, Petal.Length et Petal.Width

Construction du modèle de prédition

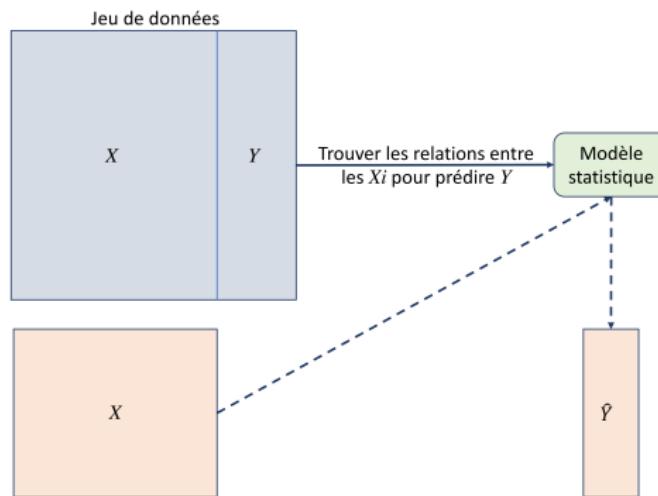


Figure 2:

Propriétés d'un bon modèle de prédiction

- ▶ Bonnes performances
- ▶ Reproductible sur de nouvelles données
- ▶ Simple
- ▶ Facile à interpréter

Protocole

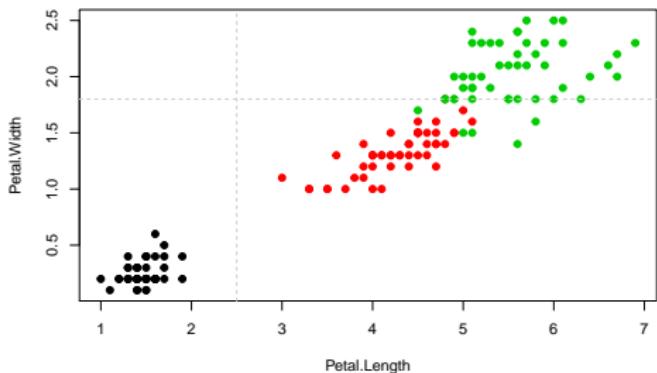
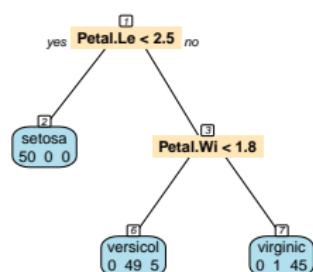
Méthodes

- ▶ Arbres de décision : CART (Classification and Regression Trees)
- ▶ Forêts aléatoires
- ▶ *Support Vector Machines* (SVM)
- ▶ Réseaux de neurones

Qu'est qu'un arbre de décision

Classification par une série de tests

- ▶ Diviser récursivement les individus à l'aide de tests définis à partir des variables jusqu'à ce qu'on obtienne des sous-ensembles d'individus n'appartenant qu'à une seule classe
- ▶ partitionnement de l'espace des données en sous-régions homogènes en termes de classes



Mise en place du modèle de prédiction

Création des échantillons d'apprentissage et de test

- ▶ Echantillon d'apprentissage : 2/3 des individus choisis aléatoirement

```
ind.app <- sample(1:nrow(iris), size = nrow(iris)*2/3)
mat.app <- iris[ind.app,]
summary(mat.app)
```

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
Min. :4.300	Min. :2.000	Min. :1.100	Min. :0.
1st Qu.:5.100	1st Qu.:2.800	1st Qu.:1.600	1st Qu.:0.
Median :5.800	Median :3.000	Median :4.500	Median :1.
Mean :5.904	Mean :3.056	Mean :3.873	Mean :1.
3rd Qu.:6.500	3rd Qu.:3.325	3rd Qu.:5.225	3rd Qu.:1.
Max. :7.900	Max. :4.200	Max. :6.900	Max. :2.

Création des échantillons d'apprentissage et de test

- ▶ Echantillon test : le 1/3 des individus restant

```
mat.test <- iris[-ind.app,]  
summary(mat.test)
```

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
Min. :4.400	Min. :2.20	Min. :1.000	Min. :0.1
1st Qu.:5.100	1st Qu.:2.80	1st Qu.:1.525	1st Qu.:0.2
Median :5.700	Median :3.00	Median :4.050	Median :1.3
Mean :5.722	Mean :3.06	Mean :3.528	Mean :1.1
3rd Qu.:6.300	3rd Qu.:3.30	3rd Qu.:4.850	3rd Qu.:1.6
Max. :7.700	Max. :4.40	Max. :6.700	Max. :2.5

Validation des échantillons d'apprentissage et test

- ▶ Création d'un vecteur couleur :
 - ▶ en rouge les individus appartenant au jeu d'apprentissage
 - ▶ en vert les individus appartenant au jeu test

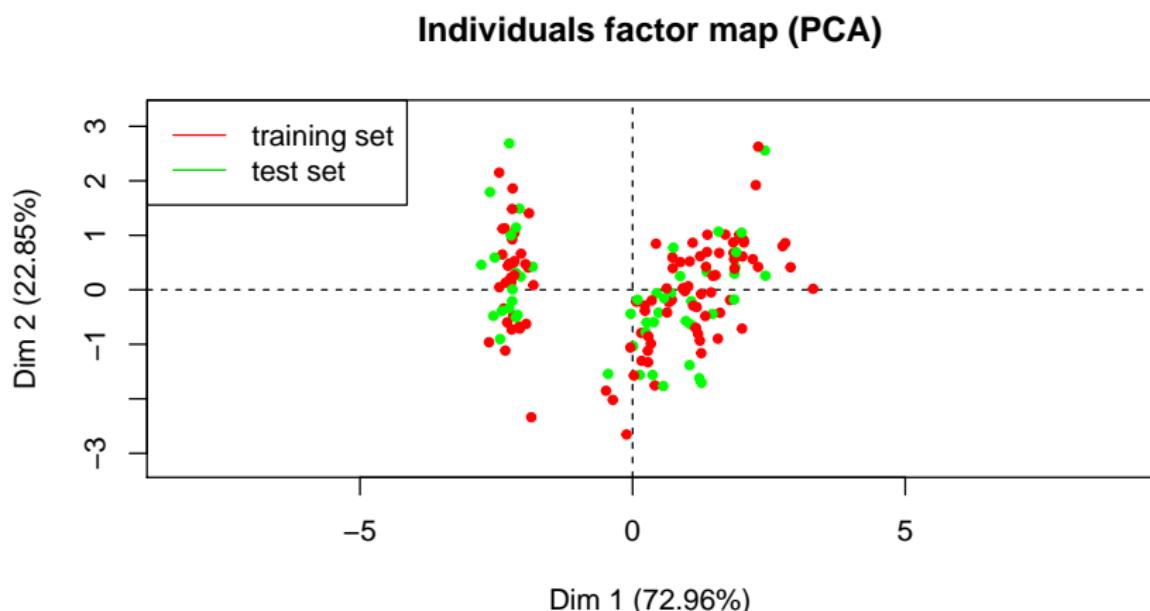
```
vcol.set <- rep("green", length <- nrow(iris))  
vcol.set[ind.app] <- "red"
```

- ▶ calcul de l'ACP en utilisant les 4 descripteurs

```
pca.res <- PCA(iris[,-5], graph = FALSE)
```

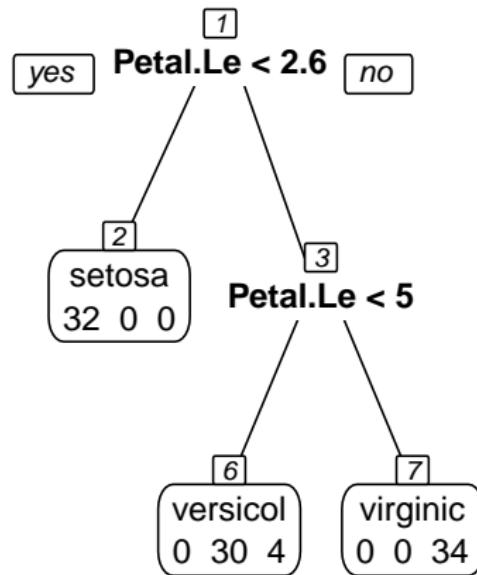
Validation des échantillons d'apprentissage et test

```
plot(pca.res, choix="ind", col.ind=vcoll.set, label="none")  
legend("topleft", c("training set", "test set"), col=c(2,3), l
```



Apprentissage du modèle sur l'échantillon d'apprentissage

```
rpart.fit<-rpart(Species ~ . , data = mat.app)
prp(rpart.fit, type=0, extra=1, nn=TRUE)
```



Evaluation du modèle sur le jeu d'apprentissage

- ▶ Table de confusion

```
pred.app<-predict(rpart.fit,newdata=mat.app,type="class")
(tc <- table(mat.app[, "Species"],pred.app))
```

		pred.app		
		setosa	versicolor	virginica
setosa	setosa	32	0	0
	versicolor	0	30	0
	virginica	0	4	34

- ▶ Taux de bien prédit

```
(TBP <- sum(diag(tc))/sum(tc))
```

[1] 0.96

Evaluation du modèle sur le jeu test

- ▶ Table de confusion

```
pred.test<-predict(rpart.fit,newdata=mat.test,type="class")  
(tc<-table(mat.test[, "Species"], pred.test))
```

		pred.test		
		setosa	versicolor	virginica
setosa	18	0	0	
versicolor	0	18	2	
virginica	0	2	10	

- ▶ Taux de bien prédit

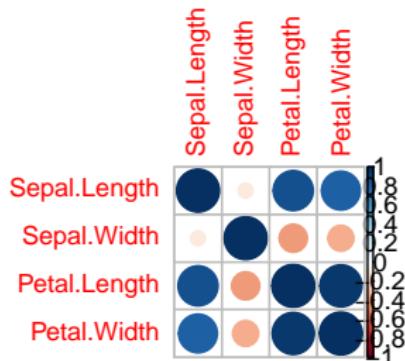
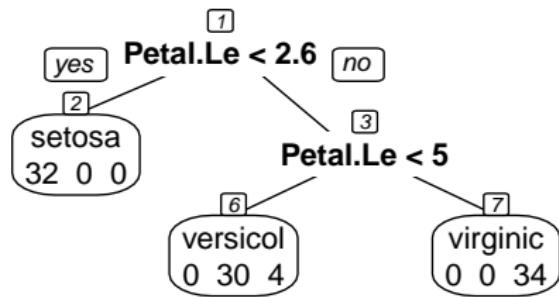
```
(TBP <- sum(diag(tc))/sum(tc))
```

[1] 0.92

Etude de l'importance des descripteurs

- Deux descripteurs Petal.Length et Petal.Width sont nécessaires pour prédire l'espèce.

```
nf<-layout(matrix(c(1,2), 1, 2))
prp(rpart.fit,type=0,extra=1,nn=TRUE)
corrplot(cor(iris[,-5]),method="circle",tl.cex=0.8,cl.cex=0.8)
```



Différents algorithmes d'arbres de décision

- ▶ ID3 (Inductive Decision Tree, Quinlan 1979)
 - ▶ arbres « de discrimination » (variables uniquement qualitatives)
 - ▶ critère d'hétérogénéité = entropie
- ▶ C4.5 (Quinlan 1993)
 - ▶ amélioration des ID3
 - ▶ gestion des valeurs manquantes
- ▶ CART (Classification And Regression Tree, Breiman et al. 1984)
 - ▶ critère d'hétérogénéité = Gini

Avantages et inconvénients des arbres de décision

- ▶ Avantages :

- ▶ Pas besoin de normaliser les données
- ▶ La variable à prédire peut être multi-classe
- ▶ Facilité d'interprétation
- ▶ Sélection de variables

- ▶ Inconvénients :

- ▶ Sensibles au bruit et points abérrants
- ▶ Sélection des variables drastique
- ▶ Sur-apprentissage

Forêts aléatoires : Principe

1 arbre



1 forêt = un grand nombre d'arbres



- ▶ Apprendre un grand nombre d'arbres permettant de voter pour la classe la plus populaire
- ▶ Double randomisation sur les variables et sur les individus

Construction d'une forêt aléatoire

- ▶ But : prédire l'espèce des iris
- ▶ Les données : échantillon d'apprentissage :
 - ▶ $p =$ nombre de variables ($p = 4$)
 - ▶ $n =$ nombre de fleurs ($n = 100$)

Construction du $k^{\text{ème}}$ arbre - randomisation sur les individus

Double randomisation sur les individus et sur les variables

- ▶ Création d'un **échantillon bootstrap** contenant n fleurs = tirage aléatoire avec remise de n fleurs dans le jeu d'apprentissage

```
ech.B1<-sample(1:nrow(mat.app),size=nrow(mat.app),replace=T)
sort(ech.B1)
```

```
[1] 2 3 3 4 6 6 6 9 9 10 12 14 15
```

Construction du $k^{\text{ème}}$ arbre - randomisation sur les individus

- ▶ Génération de deux échantillons différents à partir de la matrice d'apprentissage :
 - ▶ échantillon de bootstrap → Construction d'un arbre
 - ▶ échantillon OOB (Out-Of-Bag) : contient les individus qui ne sont pas présents dans l'échantillon bootstrap → Calcul de l'erreur du modèle

```
ech.OOB1 <- setdiff(1:nrow(mat.app), ech.B1)  
ech.OOB1
```

```
[1] 1 5 7 8 11 13 18 20 21 32 35 43 44 45 46 48 50 51
```

- ▶ Chaque arbre de la forêt va être construit sur un jeu bootstrap différent

Construction du $k^{\text{ème}}$ arbre - randomisation sur les variables

Double randomisation sur les individus et sur les variables

- ▶ Pour la construction d'un noeud de l'arbre : Sélectionne aléatoirement q variables parmi les p

```
nbr.var <- 2  
list.var <- colnames(iris)[-5]  
var.sel <- sample(list.var, size = nbr.var)  
var.sel
```

```
[1] "Sepal.Width"  "Sepal.Length"
```

Construction du $k^{\text{ème}}$ arbre - randomisation sur les variables

- ▶ Le 1er noeud du $k^{\text{ème}}$ arbre se construit avec le jeu suivant :

```
random.data1 <- mat.app[ech.B1, var.sel]  
head(random.data1)
```

	Sepal.Width	Sepal.Length
44	3.5	5.0
7	3.4	4.6
97	2.9	5.7
66	3.1	6.7
25	3.4	4.8
140	3.1	6.9

- ▶ pour les noeud suivant : sélectionne à chaque fois q variable
- ▶ nouvel arbre : création d'un nouvel échantillon bootstrap

Deux paramètres à définir

- ▶ **Nombre d'arbres** dans la forêt (*ntree*)
 - ▶ assez grand pour que chaque observation soit prédite suffisamment de fois
- ▶ **Nombre de variables tirées aléatoirement** (*mtry*) :
 - ▶ permet de gérer la corrélation et la force de la forêt :
 - ▶ **Corrélation** : ressemblance entre les arbres
 - ▶ **Force** : capacité de chaque arbre à ne pas faire d'erreur
 - ▶ idéal : Faible corrélation et une grande force
 - ▶ quand *mtry* augmente : la corrélation et la force augmentent → Trouver un juste milieu.

Performance du modèle

- ▶ Calcul de la performance non biaisée du modèle
→ : utilisation des données des échantillons OOB

Prédiction de l'espèce d'iris - Construction du modèle

```
library(randomForest)
rf.fit<-randomForest(Species~.,data=mat.app,mtry=500,
                      mtry=2,importance=TRUE)
rf.fit
```

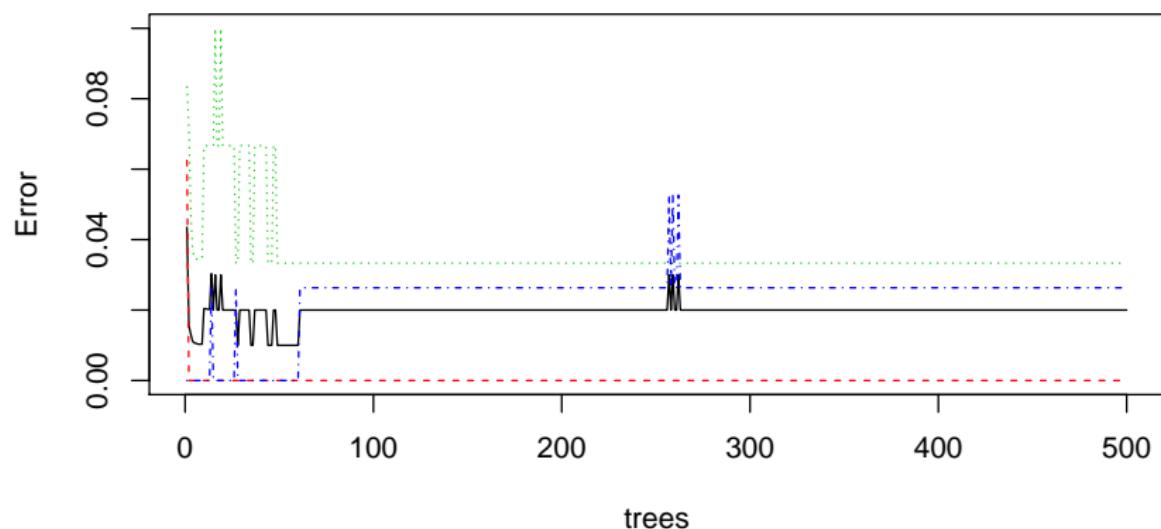
Call:

```
randomForest(formula = Species ~ ., data = mat.app, mtry = 2)
              Type of random forest: classification
                      Number of trees: 500
No. of variables tried at each split: 2

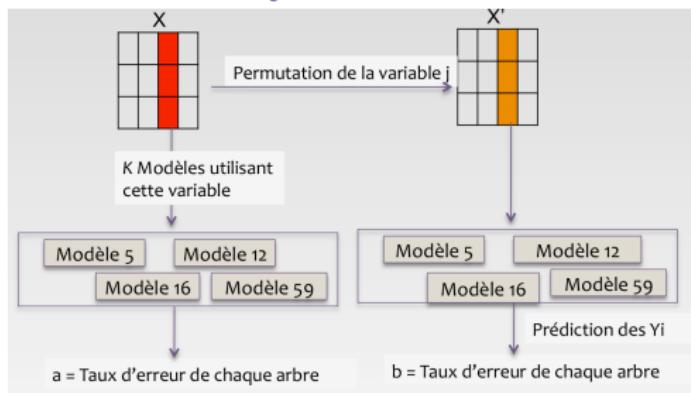
OOB estimate of  error rate: 2%
Confusion matrix:
```

Prédiction de l'espèce d'iris - Estimation des performances du modèle

```
plot(rf.fit, main="")
```



Etude de l'importance des variables

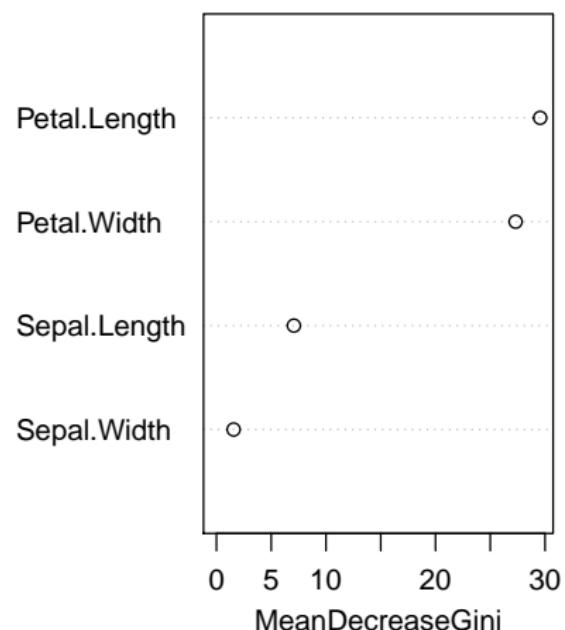
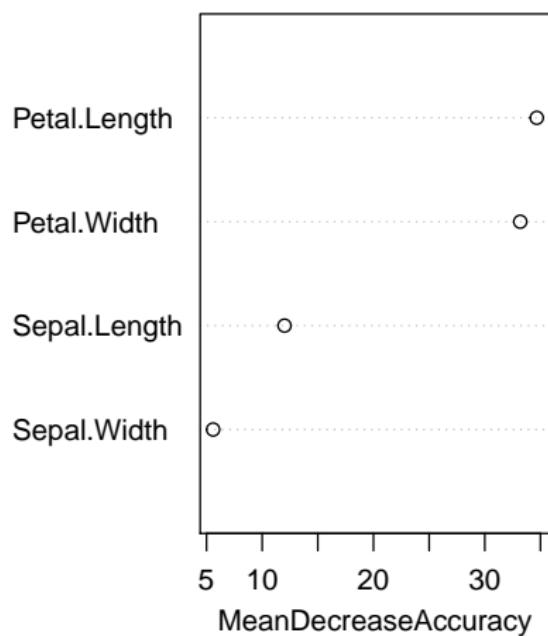


- ▶ Importance de la variable j dans l'arbre i : $Imp(j, i) = b - a$
- ▶ Importance globale de la variable j = moyenne des importances sur les k arbres impliquant la variable j

$$Imp(j) = \sum_{i=1}^k Imp(j, i)$$

Etude de l'importance des variables

```
varImpPlot(rf.fit, main="")
```



Merci de votre attention !!!

- ▶ Prédiction du type de tumeur sur les données
 - ▶ apprentissage d'un modèle de CART
 - ▶ apprentissage d'une forêt aléatoire