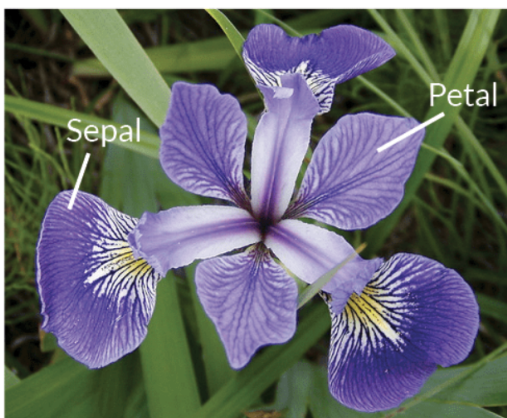


# Classification automatique sous R

## TP – École thématique Big Data Speech 2018

Pour débiter ce TP nous allons utiliser une base de données très populaire pour expérimenter les outils de classification automatique, il s'agit de la base de données Iris de Fisher.

Afin de comprendre ce que représente ce jeu de données, il est conseillé de lire la page Wikipédia suivant : [https://fr.wikipedia.org/wiki/Iris\\_de\\_Fisher](https://fr.wikipedia.org/wiki/Iris_de_Fisher)



**Iris Versicolor**



**Iris Setosa**



**Iris Virginica**

### Environnement R Studio

R Studio est un IDE (Environnement de Développement Intégré) pour R. R Studio n'est pas nécessaire pour utiliser R, mais rend son utilisation bien plus agréable et efficace.

### Chargement des données

```
# Lecture des données iris, récupérées directement sur le web
iris <- read.csv(url("http://archive.ics.uci.edu/ml/machine-
learning-databases/iris/iris.data"),
                header = FALSE)
```

→ Cette commande lit un fichier de type .csv (Comma Separated Value) qui peut être lu à partir d'un disque dur, mais aussi directement à partir d'internet

```
# Affiche les premières lignes
head(iris)
```

→ on remarque que les noms de colonnes ne sont pas explicites

```
# Change le nom des colonnes
names(iris) <- c("Sepal.Length", "Sepal.Width", "Petal.Length",
"Petal.Width", "Species")
```

```
# Vérifions le résultat
head(iris)
```

```
#Affichage de toutes les données :
iris
```

Mieux connaître les données : outils de visualisation

```
library(ggvis)
```

→ on charge la bibliothèque ggvis qui offre des outils de visualisation

```
iris %>% ggvis(~Sepal.Length, ~Sepal.Width, fill = ~Species) %>%
layer_points()
```

→ Notez l'utilisation de %>% qui fonctionne comme le pipe | sous linux

→ Que nous dit la figure ?

Après avoir analysé le lien entre caractères des sépales et espèce d'iris, concentrons sur les pétales :

```
iris %>% ggvis(~Petal.Length, ~Petal.Width, fill = ~Species) %>%
layer_points()
```

→ Que nous dit la figure ?

Pour mieux comprendre les données, il est souvent utile d'analyser les corrélations existantes entre les valeurs des différents caractéristiques (=colonnes).

R offre une commande pour cela :

```
# Corrélation entre `Petal.Length` and `Petal.Width` sur l'ensemble des données
cor(iris$Petal.Length, iris$Petal.Width)
```

Nous pouvons aussi procéder par espèce :

```
# Récupère les valeurs des étiquettes (levels) de la base `iris`
x=levels(iris$Species)
```

```
# Affiche la matrice de corrélation de l'espèce Setosa
print(x[1])
cor(iris[iris$Species==x[1],1:4])
```

→ faites de même pour l'espèce Versicolor, puis l'espèce Virginica

→ que pouvez-vous en déduire ?

Testez ces commandes pour connaître la répartition des étiquettes d'espèces dans les données :

```
# Division of `Species`
table(iris$Species)
```

```
# Percentual division of `Species`
round(prop.table(table(iris$Species)) * 100, digits = 1)
```

**Découpage des données : nous souhaitons découper les données en 2 : 2/3 des données pour l'apprentissage automatique, 1/3 pour le test.**

```
#Initialisation du générateur de nombres aléatoires  
set.seed(1234)
```

Pour que le découpage ne soit pas biaisé par l'ordre de stockage des informations (par exemple, regardez la valeur de l'étiquette en fonction du numéro de ligne), nous allons utiliser la commande `sample` :

```
ind <- sample(2, nrow(iris), replace=TRUE, prob=c(0.67, 0.33))
```

→ cette commande affecte la valeur 1 ou la valeur 2 dans le tableau `ind` pour chaque ligne existant dans `iris`. La probabilité que la valeur 1 soit affectée est de 0,67 (et donc 0,33 pour la valeur 2)

#À partir de `ind` et de `iris`, nous pouvons construire notre corpus d'apprentissage :

```
iris.training <- iris[ind==1, 1:4]
```

→ construisez le corpus `iris.test` :

```
iris.test <- iris[??????????]
```

Regardez le contenu de `iris.training` et `iris.test`

→ vous remarquerez que les étiquettes (=espèces) ne sont pas dans les corpus créés. Il nous faut maintenant les gérer dans deux tableaux (=vecteurs) dédiés.

```
# Construis les étiquettes d'apprentissage `iris`  
iris.trainLabels <- iris[ind==1,5]
```

```
# Affiche le résultat  
print(iris.trainLabels)
```

→ Construisez les étiquettes de test et affichez-les

**Apprentissage automatique et prédiction (utilisation de l'algorithme des k plus proche voisins) :**

```
# Construit le modèle et l'applique sur le test  
iris_pred <- knn(train = iris.training, test = iris.test, cl =  
iris.trainLabels, k=3)
```

```
# Lire les résultats `iris_pred`  
iris_pred
```

### Évaluation du modèle

Nous allons d'abord construire un tableau qui contient, pour chaque échantillon du test, son étiquette prédite par le modèle, et son étiquette réelle

```
# Put `iris.testLabels` in a data frame  
irisTestLabels <- data.frame(iris.testLabels)
```

```
# Merge `iris_pred` and `iris.testLabels`  
merge <- data.frame(iris_pred, iris.testLabels)  
  
# Specify column names for `merge`  
names(merge) <- c("Predicted Species", "Observed Species")  
  
# Inspect `merge`  
merge
```

→ à partir de là, il est possible de faire de visualiser les différences

Pour aller plus loin dans l'analyse, nous pouvons utiliser des outils dédiés, par exemple CrossTable qui affiche la matrice de confusion

```
library(gmodels)  
CrossTable(x = iris.testLabels, y = iris_pred, prop.chisq=FALSE)
```

→ Analysez ces résultats

Appliquez ce que vous venez de voir sur les données 'parole' extraites par Nicolas Audibert et disponibles sur le site de l'école d'été (contactez un intervenant pour plus d'informations)