

Atelier Analyse de données	
Atelier 5 : Machine Learning avec R	Classe : IOT3.1
Enseignante : A.khelifa	Durée : 3 Heures

Important

Avant de commencer l'implémentation des activités de l'atelier, il faut installer les packages *ggplot2*, *caret*, *kernlab* et *randomForest*.

Activité1

L'objectif est d'appliquer des exemples de Machine Learning en utilisant le langage R en utilisant le dataset 'IRIS'.

Travail demandé

En suivant les étapes de 1 à 6, donnez le résultat de chaque étape tout en interprétant les résultats.

Etape1 : Chargement du dataset

Pour charger le dataset 'IRIS', il suffit d'écrire les lignes de code suivantes :

```
data(iris)
dataset <- iris
```

Etape2 : Décomposition en Training/Validation

```
# 80% Training and 20% Validation
index <- createDataPartition(dataset$Species, p=0.80, list=FALSE)
validation <- dataset[-index,]
training <- dataset[index,]
```

Etape3 : Description et synthèse des données

- Dimension des données

```
# dimensions of dataset
dim(dataset)
dim(validation)
dim(training)
```

- **Types des attributs**

```
# list types for each attribute  
sapply(dataset, class)
```

- **Affichage des données**

```
# take a peek at the first 5 rows of the data  
head(dataset)  
View(dataset)
```

- **Modalités de la classe**

```
# list the levels for the class  
levels(dataset$Species)
```

- **Distribution des données par classe**

```
# summarize the class distribution  
percentage <- prop.table(table(dataset$Species)) * 100  
cbind(freq=table(dataset$Species), percentage=percentage)
```

- **Analyse statistique**

```
# summarize attribute distributions  
summary(dataset)
```

Etape3 : Visualisation des données

- **Analyse univariée** : L'objectif est de visualiser chaque attribut à part pour comprendre ses caractéristiques

```
# split input and output  
x <- dataset[,1:4]  
y <- dataset[,5]  
# boxplot for each attribute on one image  
par(mfrow=c(1,4))  
for(i in 1:4) {  
  boxplot(x[,i], main=names(iris)[i])  
}
```

- **Analyse multi-variée** : L'objectif est d'examiner les corrélations entre les variables

Graphique 1

```
# scatterplot matrix  
featurePlot(x=x, y=y, plot="ellipse")
```

Graphique 2

```
# box and whisker plots for each attribute  
featurePlot(x=x, y=y, plot="box")
```

Graphique 3

```
# density plots for each attribute by class value
scales <- list(x=list(relation="free"),
              y=list(relation="free"))
featurePlot(x=x, y=y, plot="density", scales=scales)
```

Etape 4 : Appliquer les modèles de Machine Learning

- Méthode de Cross-validation

```
# Run algorithms using 10-fold cross validation
control <- trainControl(method="cv", number=10)
metric <- "Accuracy"
```

- Modèle 1 : Linear Discriminant Analysis (LDA)

```
#linear algorithms
set.seed(7)
fit.lda <- train(Species~., data=training, method="lda",
metric=metric, trControl=control)
```

- Modèle 2 : Classification and Regression Trees (CART)

```
# nonlinear algorithms
# CART
set.seed(7)
fit.cart <- train(Species~., data=training, method="rpart",
metric=metric, trControl=control)
```

- Modèle 3 : k-Nearest Neighbors (kNN)

```
# kNN
set.seed(7)
fit.knn <- train(Species~., data=training, method="knn",
metric=metric, trControl=control)
```

- Modèle 4: Support Vector Machines (SVM) with a linear kernel

```
# advanced algorithms
# SVM
set.seed(7)
fit.svm <- train(Species~., data=training, method="svmRadial",
metric=metric, trControl=control)
```

- **Modèle 5 : Random Forest (RF)**

```
# Random Forest
set.seed(7)
fit.rf <- train(Species~., data=training, method="rf",
metric=metric, trControl=control)
```

Etape 5 : Evaluer les modèles

- **Afficher les resultats**

```
# summarize accuracy of models
results <- resamples(list(lda=fit.lda, cart=fit.cart,
knn=fit.knn, svm=fit.svm, rf=fit.rf))
summary(results)
```

- **Comparer les résultats**

```
# compare accuracy of models
dotplot(results)
```

- **Afficher le résultat de chaque modèle**

```
print(fit.lda)
print(fit.cart)
print(fit.knn)
print(fit.svm)
print(fit.rf)
```

Etape 6 : Faire des prédictions

```
# estimate skill of LDA on the validation dataset
predictions <- predict(fit.lda, validation)
confusionMatrix(predictions, validation$Species)
```

Activité2

Appliquez les étapes de l'activité1 sur le dataset 'breastCancer.csv'

Remarques :

1. Vous devez transformer la variable diagnosis en Factor (diagnosis constitue l'output)
2. Dans votre analyse, vous devez considérer comme input les variables de 3 à 32.
3. Dans la visualisation, vous pouvez se contenter de visualiser quelques variables de l'input.