Filière ingénieure - ING1-GM

Fouille de Données T.P. N^o 3

Arbres de décision

29 janvier 2023

Etude pratique avec R

Le but de cette section est d'appliquer les algorithmes CART et Random Forest sur le dataset IRIS :

1. **Exploration de l'ensemble des données IRIS** :
2. **Caractéristiques** :

```
dim(iris)
names(iris)
str(iris)
attributes(iris)
```

3. **Devision en train et test**

```
set.seed(1234)
ind <- sample(2, nrow(iris), replace=TRUE, prob=c(0.7, 0.3))
trainData <- iris[ind==1,]
testData <- iris[ind==2,]
trainData
testData
```

4. **Construction d'un arbre de décision**

```
myFormula <- Species ~ Sepal.Length + Sepal.Width +
                       Petal.Length + Petal.Width
library(rpart)
iris_rpart <- rpart(myFormula, data=trainData,
                    control = rpart.control(minsplit = 5))
attributes(iris_rpart)
print(iris_rpart)
```

5. **Plot**

```
library(rpart.plot)
prp(iris_rpart,extra=1)
```

6. **Prédictions et matrices de confusion**

```
trainPred <- predict(iris_rpart, newdata = trainData, type="class")
table(trainPred, trainData$Species)

testPred <- predict(iris_rpart, newdata = testData, type="class")
table(testPred, testData$Species)
```

Exercice 1 Il est possible d'ajuster l'arbre en réglant un paramètre de complexité : afficher le paramètre de complexité avec la commande

```
iris_rpart$control$cp
```

Représenter graphique l'erreur sur l'ensemble d'apprentissage en fonction du paramètre *cp* avec la commande

```
plotcp(iris_rpart)
```

A-t-on besoin d'ajuster mieux ce paramètre ? Effectuer la modification suivante du paramètre *cp* :

```
iris_rpart <- rpart(myFormula, data=trainData,
                    control = rpart.control(minsplit = 5, cp=0.2))
```

Exercice 2 Appliquer la fonction evaluator (voir le TP numéro 2 sur les modèles bayésiens) sur le modèle $iris\_rpart$ et l'ensemble de test.

1. **Construction d'une forêt aléatoire**

```
library(randomForest)
rf <- randomForest(Species ~ ., data=trainData, ntree=100)

print(rf)
attributes(rf)
```

2. **Importance des variables**

```
importance(rf)
varImpPlot(rf)
```

Exercice 3 Faire varier les paramètres suivants : **ntree** qui correspond au nombre d'arbres (500 par défaut) et **mtry** qui correspond au nombre de variables à prendre en compte pour chaque nœud d'arbre. Comparer les résultats sur l'ensemble du test avec la fonction evaluator.

Exercice 4 Refaire le même travail décrit dans cette partie pratique sur le dataset *PimaIndiansDiabetes* de la libraire *mlbench*