المدرسة العليا للتكنولوجيا
École Supérieure de Technologie
ⵜⴰⵖⵍⵍⴰⵜ ⵏ ⵜⵓⵙⵙⵔⵉⵜ ⴰ ⵜⵓⵎⵓⵔⵜ ⴰⵣ ⵙⵍⴰⵎ

PRÉDICTION DE LA POTABILITÉ DE L'EAU

ARBRE DE DÉCISION

RÉALISÉ PAR

Mohammed JABRI
Ayman MAKHOUKHI

ENCADRÉ PAR

Mr Mohammed BADAOUI

Année Universitaire : 2020/2021

REMERCIEMENTS

Au terme de ce travail, nous -étudiants de filière LP-ID- tenons à exprimer nos profonds remerciements à notre professeur et encadrant Monsieur Mohammed BADAOUI pour son suivi et son soutien tout au long de la période de projet.

TABLE DE MATIERES

INTRODUCTION

Dans le domaine d'apprentissage automatique (Machine Learning), on distingue entre deux principaux types d'apprentissages : supervisées et non supervisées.

- L'apprentissage supervisé :

Nous avons une connaissance préalable de ce que devraient être les valeurs de sortie de nos échantillons. Par conséquent, l'objectif de l'apprentissage supervisé est d'apprendre une fonction qui, à partir d'un échantillon de données et des résultats souhaités, se rapproche le mieux de la relation entre entrée et sortie observable dans les données.

- L'apprentissage non supervisé :

On n'a pas de résultats étiquetés. Son objectif est donc de déduire la structure naturelle présente dans un ensemble de points de données.

L'exploration des données peut se regrouper en deux catégories principales :

- Méthodes Descriptives :

Consistent à regrouper des objets en classes (clusters) de sorte que les objets d'un même groupe se ressemblent le plus possible.

- Méthodes Prédictives :

Consistent à estimer la valeur d'une variable cible en fonction de la valeur d'un certain nombre d'autres variables explicatives.

Pour ce type de méthodes on peut distinguer également entre :

- Régression :

Les algorithmes de régression sont utilisés pour prédire une valeur numérique continue (prix, salaire…)

- Classement :

Les algorithmes de classification sont utilisés pour prédire une valeur discrète (sexe, vrai ou faux …)

Pendant cette étude on va opter pour le classement, ou on va prédire la possibilité que l'eau est ou pas potable en utilisant une « arbre de décision ».

MODELE ETUDIE

- **Utilité :**

Le modèle choisi pour élaborer cette classification c'est « Arbre de décision ».
Un arbre de décision est un schéma représentant les résultats possibles d'une série de choix interconnectés. Il permet à une personne ou une organisation d'évaluer différentes actions possibles en fonction de leur coût, leur probabilité et leurs bénéfices. Il peut être utilisé pour alimenter une discussion informelle ou pour générer un algorithme qui détermine le meilleur choix de façon mathématique.
Il s'agit de plus d'une représentation calculable automatiquement par des algorithmes d'apprentissage supervisé, qu'il a l'avantage d'être lisible et rapide à exécuter.

- **Exemple d'utilisation :**

❖ Sécurité et fouille de données
❖ Médecine :
  - Prédire la maladie du patient.
  - Prédire la sensibilité des médicaments.
❖ Finance :
  - Prévoir les résultats futurs et attribuer des probabilités à ces derniers.
  - Prédictions binomiales des prix et analyse des options réelles.
  - Approbation du prêt.

- **Principe et méthodologie :**

Un arbre de décision commence généralement par un nœud d'où découlent plusieurs résultats possibles. Chacun de ces résultats mène à d'autres nœuds, d'où émanent d'autres possibilités. Le schéma ainsi obtenu rappelle la forme d'un arbre.

Il existe trois types de nœuds différents : des nœuds de hasard, des nœuds de décision et des nœuds terminaux. Un nœud de hasard, représenté par un cercle, montre les probabilités de certains résultats. Un nœud de décision, représenté par un carré, illustre une décision à prendre, et un nœud terminal le résultat final d'un chemin de décision.

- **Avantages :**

- Ils sont faciles à comprendre.
- Ils peuvent être utiles avec ou sans données concrètes, et les données quelles qu'elles soient, nécessitent une préparation minimale.
- De nouvelles options peuvent être ajoutées aux arbres existants.
- Ils permettent de sélectionner l'option la plus appropriée parmi plusieurs.
- Il est facile de les associer à d'autres outils de prise de décision.
- Le coût d'utilisation de l'arbre pour prédire des données diminue à chaque point de donnée supplémentaire.
- Ils fonctionnent aussi bien pour les données de catégorie que numériques.
- La modélisation des problèmes est possible avec plusieurs données de sortie.
- Ils utilisent un modèle de boîte blanche, ce qui rend les résultats faciles à expliquer.
- La fiabilité d'un arbre peut être testée et quantifiée.
- Ils tendent à être précis, même si les hypothèses des données source ne sont pas respectées.

- **Inconvénients :**

- Lors de la gestion de données de catégorie comportant plusieurs niveaux, le gain d'information est biaisé en faveur des attributs disposant du plus de niveaux.
- Les calculs peuvent devenir compliqués lorsqu'une certaine incertitude est de mise et que de nombreux résultats sont liés entre eux.
- Les conjonctions entre les nœuds sont limitées à l'opérateur « ET », alors que les graphiques décisionnels permettent de connecter des nœuds avec l'opérateur «OU».

DESCRIPTION DES DONNEES

- **Objectif :**

Prédire la possibilité si une eau donnée est potable ou non, selon des facteurs de leur qualité.

- **Description :**

La base de données utilisé pendant cette étude est nommée « water_potability.csv », et contient 10 attributs.

L'attribut « *Potability* » c'est la **variable cible**, elle désigne la potabilité de l'eau.
Cet attribut regroupe de catégories de valeur :

- 0 : L'eau donné n'est pas potable.
- 1 : L'eau donné est potable.

Cette variable cible est estimée en fonction de plusieurs autres **variables explicatives** qui constitues les 9 attributs restants :

Attribut	**Description**
PH	La valeur du pH (0 => 14)
Hardness	Capaciter de l'eau à précipiter le savon
Solids	Solides dissous totaux en ppm
Chloramines	Quantité de chloramines en ppm
Sulfates	Quantité de sulfates dissous en mg/L
Conductivity	Conductivité électrique de l'eau en µS/cm
Organic_carbon	Quantité de carbone organique en ppm
Trihalomethanes	Quantité de Trihalomethanes en µg/L
Turbidity	Mesure de la propriété d'émission de lumière de l'eau en NTU

APPLICATION & INTERPRETATION

Avant d'appliquer les fonctions qui correspondent au modèle, on prépare d'abord l'environnement en téléchargeant les packages et en important les bibliothèques nécessaires.

L'installation des packages nécessaires

```{r}
install.packages("FSelector")
install.packages("party")
install.packages("rpart.plot")
install.packages("data.tree")
install.packages("ggthemes")
```

Le chargement des librairies nécessaire

```{r}
library(FSelector)
library(rpart)
library(caret)
library(rpart.plot)
library(data.tree)
library(dplyr)

library(randomForest)
library(psych)
library(pROC)
library(Amelia)

library(ggplot2)
library(plotly)
library(ggthemes)

library(rattle)
```

Considérons l'environnement est déjà préparé, ainsi que les packages sont installés et les librairies nécessaires sont importés, on commence par l'étude des données ainsi que sa structure et description, après on vérifie s'il y'a des valeurs manquantes dans les observations comme indiqué ci-dessous :

Importation et étude des données

```{r}
# Importer les donnée qui ont dans le fichier water_potability.csv qui est dans la même répertoire
data <- read.csv("./water_potability.csv")

# les premières observations des données
head(data)

# les dernières observations des données
tail(data)

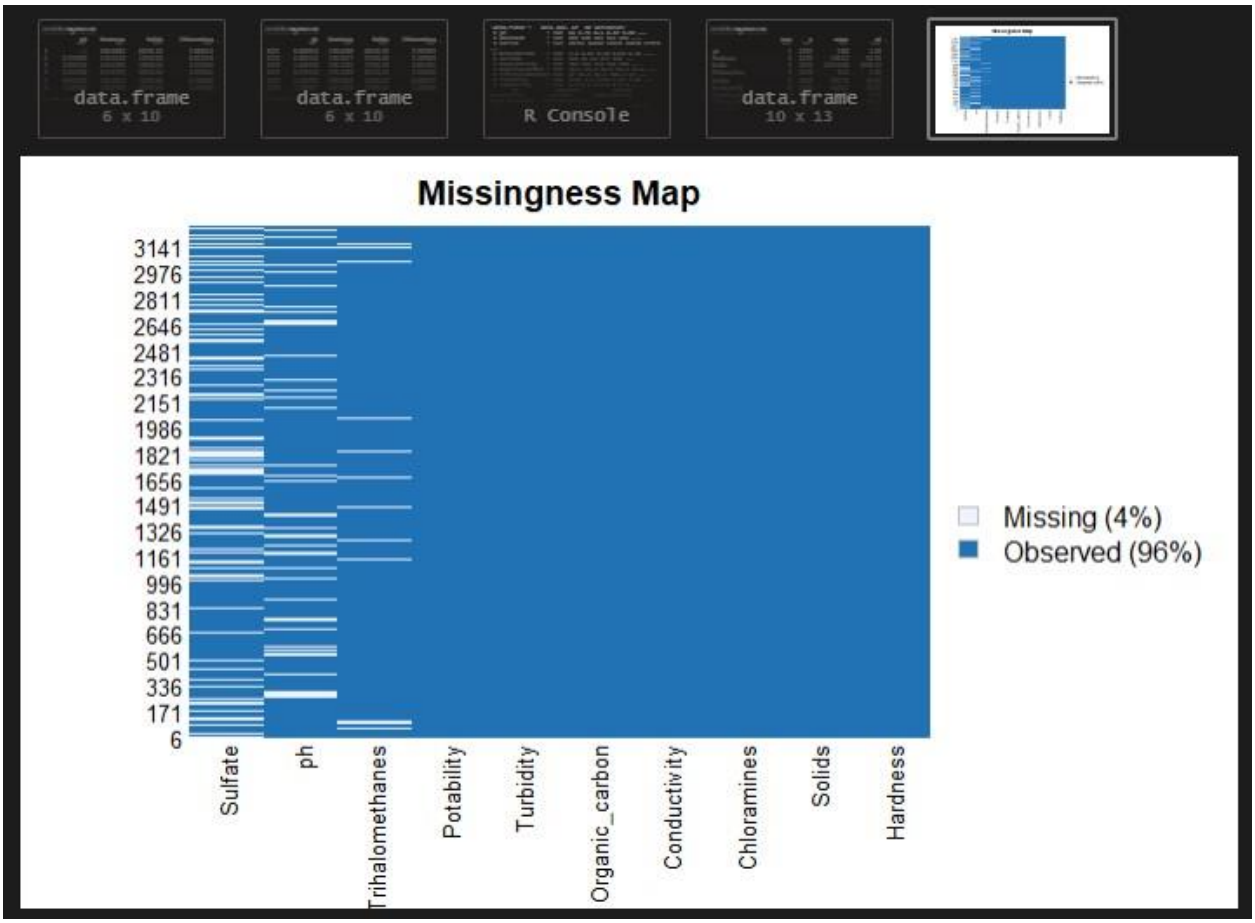
# La structure des attributs
str(data)

# Statiques de bases sur l'ensemble des attributs
summary(data)
describe(data)

# Les valeurs non-observés pour chaque attribut
missmap(data)

table(data$Potability)
```

- Résultat :



Le plot de la fonction missmap() nous montre qu'il y a souvent beaucoup de valeurs manquants dans plusieurs observations de la base de données, surtout lequel de l'attribut « Sulfate » et « pH ».

Tout d'abord on va convertir la variable cible de la forme numérique à la forme catégorielle pour éviter les problèmes avec la classification, et ensuite on va supprimer toutes les observations qui ont des valeurs manquantes :

```
Pre-Processing, correction et nettoyage des données

```{r}

# generer une liste des indexes aléatoires
shuffle_index <- sample(1:nrow(data))

# On va utiliser ses indexes pour mélanger les donnée
data <- data[shuffle_index, ]

# Conversion de la variable cible d'une forme numérique au forme catégorielle
data <- mutate(data, Potability = factor(Potability, levels = c(0, 1), labels = c('No', 'Yes')))

# Supprimer tout les observation avec des valeurs manquantes
data <- na.omit(data)
glimpse(data)

```
```

Après avoir étudier les données, ainsi que les nettoyer, on va diviser la base de données en base d'apprentissage « training data », et une base de test « testing data » :

```
Divise les données entre données d'apprentissage et données de test

```{r}
# Les données sont divisées : 80% d'apprentissage, et 20% de test
set.seed(123)
sample = sample.split(data$Potability, SplitRatio = .80)

# Données d'apprentissage
train_data = subset(data, sample==TRUE)

# Données de test
test_data = subset(data, sample==FALSE)

# Pourcentage de potabilité dans les données d'apprentissage et les données de test
prop.table(table(train_data$Potability))
prop.table(table(test_data$Potability))
```
```

L'objectif principal de cette division est de vérifier les performances du modèle sur des données invisibles. Ou, en d'autres termes d'entrainer le modèle sur l'ensemble d'apprentissage et vérifier ses performances sur l'ensemble de test :

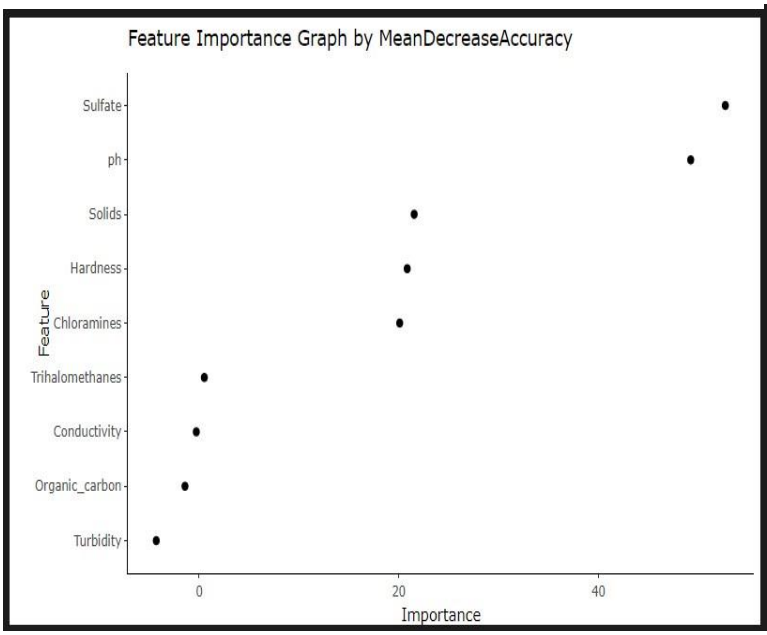
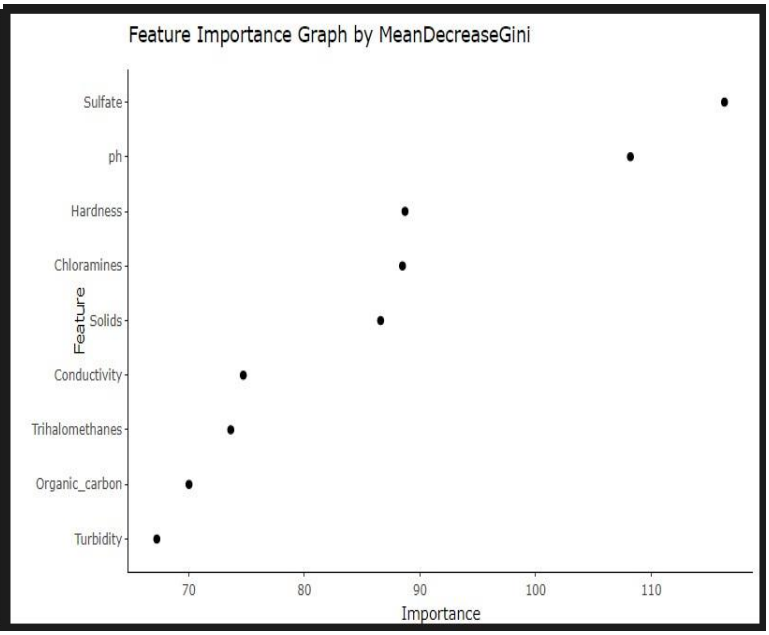
```
# Pourcentage de potabilité dans les données d'apprentissage et les données de test
prop.table(table(train_data$Potability))
prop.table(table(test_data$Potability))
```

```
        0         1
0.6099237 0.3900763

        0         1
0.6097561 0.3902439
```

On divise les données en 80% pour l'apprentissage et 20% pour le test, et après ont vérifié les pourcentages de potabilité dans chaque base.

Etude de l'importance chaque variable selon « Accuracy » avec le plot « MeanDecreaseAccuracy », et aussi « Gini » avec le plot « MeanDecreaseGini » :

```r
```{r}
# Importance des attributs
# L'utilisation des forêts aléatoires juste pour visualiser l'importance de chaque variable

rf_tmp <- randomForest(Potability ~ .,
                       data=train_data, ntree=1000,
                       keep.forest=FALSE,
                       importance=TRUE)

# varImpPlot(rf_tmp, main = "Importance des variables")
# importance(rf_tmp)

# GGploot Plots

feat_imp_df <- importance(rf_tmp) %>%
    data.frame() %>%
    mutate(feature = row.names(.))

# Feature Importance Graph | MeanDecreaseAccuracy

importanceAccuracyGraph <- ggplot(feat_imp_df, aes(x = reorder(feature, MeanDecreaseAccuracy), y = MeanDecreaseAccuracy)) +
    geom_point() +
    coord_flip() +
    theme_classic() +
    labs(
      x     = "Feature",
      y     = "Importance",
      title = "Feature Importance Graph by MeanDecreaseAccuracy",
      color="Feature"
    )

ggplotly(importanceAccuracyGraph)


# Feature Importance Graph | MeanDecreaseGini

importanceGiniGraph <- ggplot(feat_imp_df, aes(x = reorder(feature, MeanDecreaseGini), y = MeanDecreaseGini)) +
    geom_point() +
    coord_flip() +
    theme_classic() +
    labs(
      x     = "Feature",
      y     = "Importance",
      title = "Feature Importance Graph by MeanDecreaseGini",
      color="Feature"
    )

ggplotly(importanceGiniGraph)
```
```

- Résultat :





MeanDecreaseAccuracy : Estimer à quel point l'absence d'une variable affecte la performance.

MeanDecreaseGini : Estimer la pureté des nœuds par division.

Selon le résultat ci-dessus on peut constater que les variables : « Sulfate » et « ph » ont une importance élevée.

Ensuite on applique le classifieur « arbre de décision » avec la fonction rpart () :

```
Création du modèle d'arbre de décision avec les données d'apprentissage

```{r}
# Créattion du classifieur sur les donnée d'apprentissage
tree <- rpart(Potability ~.,
              data = train_data,
              method="class")

```
```

Et finalement on effectue la prédiction sur la base de test « testing data » :

```
Prédiction avec l'arbre de décision sur les donnée de test

```{r}

# Prédiction sur les données de test
tree.Potability.predicted <- predict(tree, test_data, type='class')

# Calculer l'erreur de la prédiction sur les données de test
tab <- table(tree.Potability.predicted, test_data$Potability)
paste("Erreur sur le test_data :", round(1 - sum(diag(tab)) / sum(tab), digits = 2), "%")
cat("\n")

# Générer la courbe ROC
roc(test_data$Potability,
    as.numeric(tree.Potability.predicted),
    plot=TRUE, legacy.axes=TRUE, percent=TRUE, print.auc=TRUE)

# Evaluer le modèle avec la matrice de confusion
confusionMatrix(tree.Potability.predicted,test_data$Potability)
```
```

- Résultat de calcul de l'erreur, et l'évaluation du modèle avec la matrice de confusion :

```
[1] "Erreur sur le test_data : 0.35 %"

Confusion Matrix and Statistics

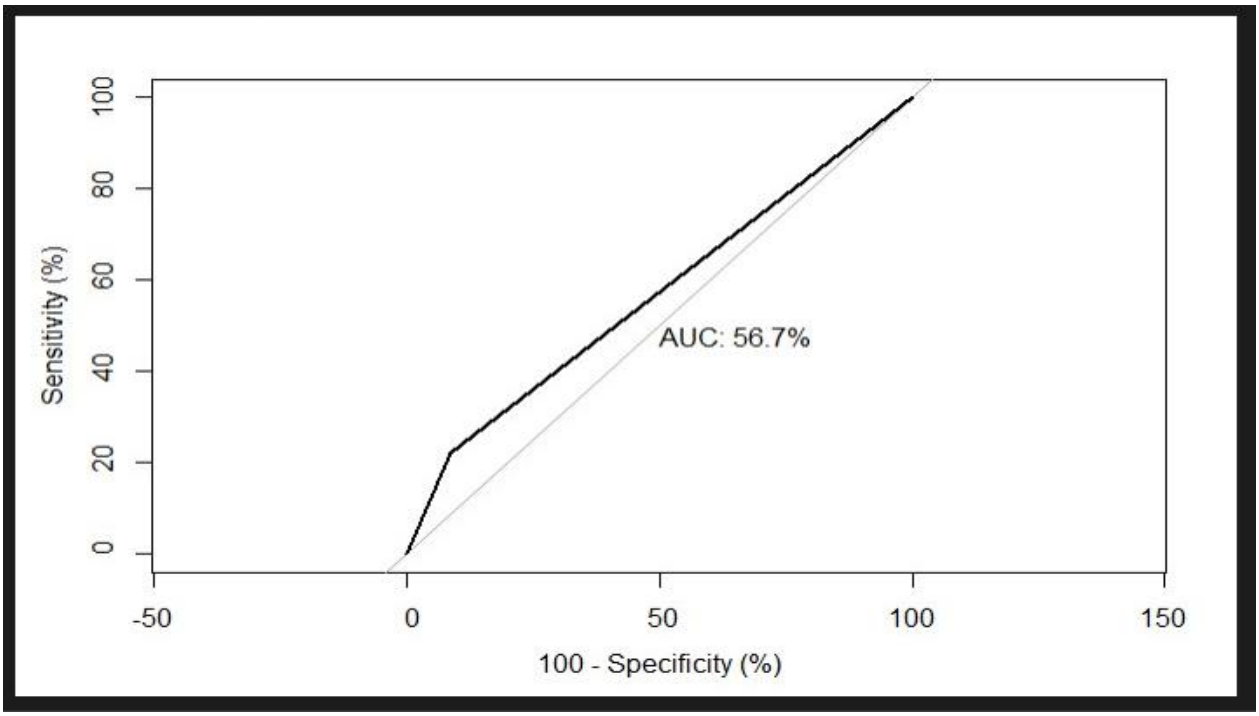
          Reference
Prediction  No Yes
       No  212 112
       Yes  28  50

               Accuracy : 0.6517
                 95% CI : (0.6029, 0.6983)
    No Information Rate : 0.597
    P-Value [Acc > NIR] : 0.01388

                  Kappa : 0.2096

 Mcnemar's Test P-Value : 2.303e-12

            Sensitivity : 0.8833
            Specificity : 0.3086
         Pos Pred Value : 0.6543
         Neg Pred Value : 0.6410
             Prevalence : 0.5970
         Detection Rate : 0.5274
   Detection Prevalence : 0.8060
      Balanced Accuracy : 0.5960

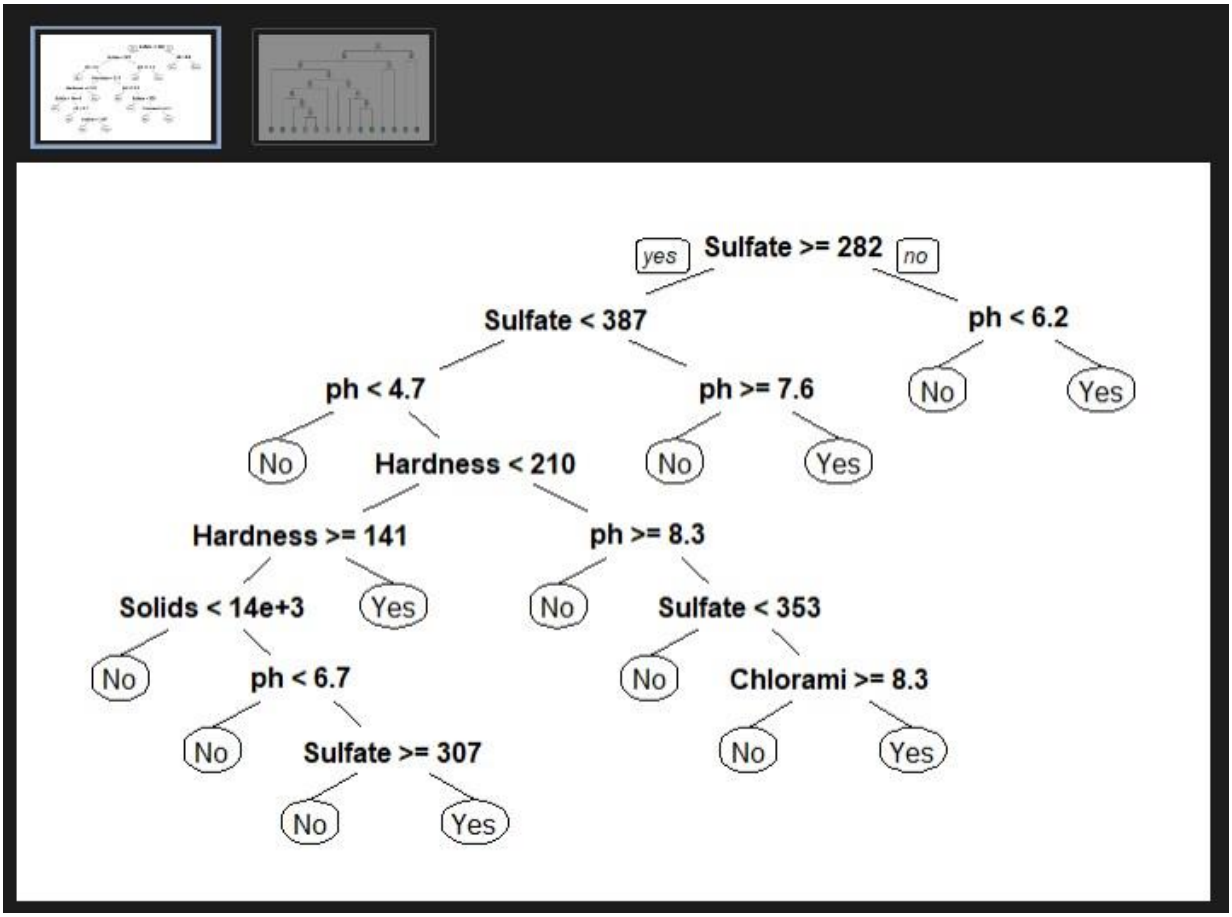
       'Positive' Class : No
```

La précision de ce modèle est 65%, l'erreur c'est 35%, et le P-Value = 0,014.

« 219 échantillons sont bien classés comme potable par contre 21 mal classés sur la colonne 1 ».

« 36 échantillons sont bien classés comme pas potable par contre 126 mal classés sur la colonne 2 ».

- La courbe ROC :



Ce dernier représente les performances du modèle, en traçant le taux des valeurs Faux Positives en fonction de valeurs Vrais Positives.

- Visualisation de l'arbre de décision graphiquement :

```
Visualisation de l'arbre de décision graphiquement

```{r}

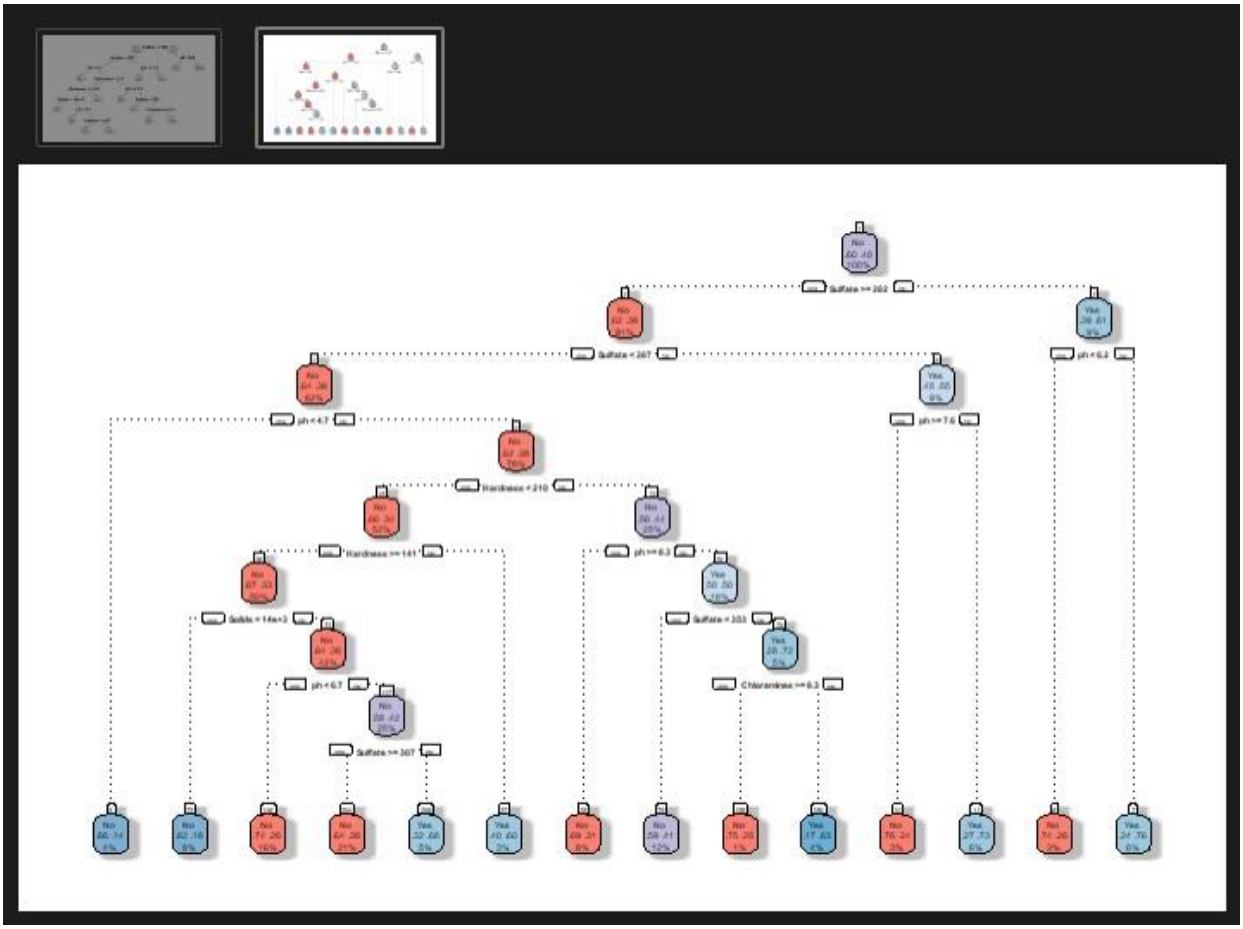
# Visualisation de l'arbre de décision
prp(tree)
rpart.plot(tree)

#print(tree2)
#plot(tree2)
```
```

- Résultat : Affichage 1



- **Résultat : Affichage 2**



CONCLUSION

Les arbres de décision sont largement utilisés pour aider à faire de bons choix dans de nombreuses disciplines différentes, y compris le diagnostic médical, le science cognitive, l'intelligence artificielle, la théorie du programme, l'ingénierie et l'exploration de données.

Cependant les nombreux avantages qui a, et qui le rend un modèle très simple à implémenter et étudier, il peut se rencontrer avec quelques problèmes, on cite parmi eux, par exemple la précision des arbres de décision est généralement environ 60%, ceci est considéré comme une bonne précision, mais il existe plein de situation, ou une précision très élevée est voulu, ce qui exige à choisir un autre modèle qui donne une précision élevée, par exemple « Naïve Bayes », « Random Forest » ...

Si on prend « Random Forest » par exemple, cette dernière peut éviter des problèmes que l'arbre de décision ne peut pas, l'un d'eux c'est le sur-apprentissage « Over-Fitting », le modèle peut tomber sur ce problème lorsque les données d'apprentissage et de test ne couvre pas toutes les combinaisons possibles des données, les forêts aléatoires avoir la capacité d'éviter ce dernier efficacement surtout en utilisant l'approche « Bagging » pour diminuer la variance le maximum possible.

Et finalement il n'existe pas un modèle parfait pour toutes les situations, chaque situation, chaque ensemble de données et chaque objectif souhaité exige un choix spécifique d'un modèle.

WEBOGRAPHIE

https://www.kaggle.com/adityakadiwal/water-potability

https://www.guru99.com/

https://www.lucidchart.com/pages/fr/arbre-de-decision

https://en.wikipedia.org/wiki/Decision_tree

https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

https://www.rdocumentation.org/