Práctica 5
Árboles de decisión

Mario Emilio Jiménez Vizcaíno
A01173359@itesm.mx
Tecnológico de Monterrey
Ingeniería en Tecnologías Computacionales
Monterrey, N.L., México

Jesus Abraham Haros Madrid
A01252642@itesm.mx
Tecnológico de Monterrey
Ingeniería en Tecnologías Computacionales
Monterrey, N.L., México

ABSTRACT

TODO

1 INTRODUCCIÓN

TODO

2 CONCEPTOS PREVIOS

- Programación básica en los lenguajes R y Python
- Conocimiento de las librerías *scikit-learn*, *pandas* y *numpy*
- **TODO**

3 METODOLOGÍA

3.1 Dataset Iris

TODO

3.2 Dataset Wine

TODO

3.3 Dataset Breast Cancer

TODO

3.4 Modelo de árbol de decisión

TODO

El código que ejecutamos para realizar el análisis del dataset se encuentra en el apéndice A.

4 RESULTADOS

TODO

4.1 Dataset Iris

TODO

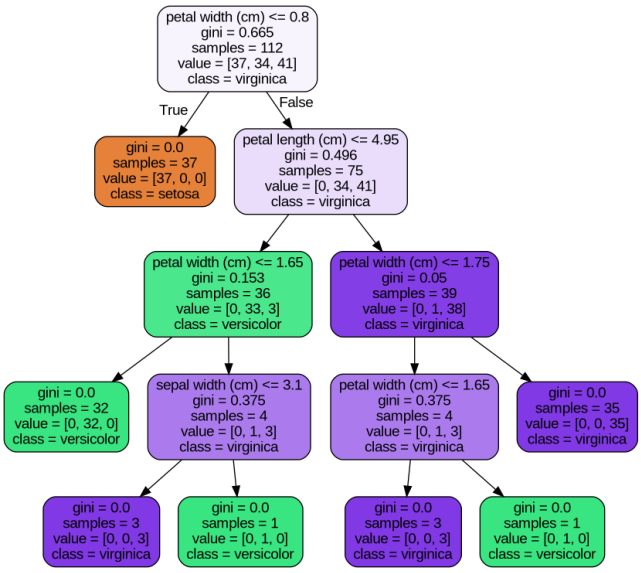

Figure 1: Modelo de árbol de decisión generado para el dataset Iris

4.2 Dataset Wine

TODO

4.3 Dataset Breast Cancer

TODO

5 CONCLUSIONES Y REFLEXIONES

TODO

5.1 Refrexión de Abraham

TODO

5.2 Reflexión de Mario

TODO

REFERENCES

A CÓDIGO PARA LA GENERACIÓN DEL ÁRBOL DE DECISIÓN

```
import os

from graphviz import Source
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, export_graphviz

mpl.rc('axes', labelsize=14)
mpl.rc('xtick', labelsize=12)
mpl.rc('ytick', labelsize=12)

# Setup folders
PROJECT_ROOT_DIR = "."
CHAPTER_ID = "decision_trees"
IMAGES_PATH = os.path.join(PROJECT_ROOT_DIR, "images", CHAPTER_ID)
os.makedirs(IMAGES_PATH, exist_ok=True)


def save_fig(fig_id, tight_layout=True, fig_extension="png", resolution=300):
    path = os.path.join(IMAGES_PATH, fig_id + "." + fig_extension)
    print("Saving figure", fig_id)
    if tight_layout:
        plt.tight_layout()
    plt.savefig(path, format=fig_extension, dpi=resolution)


# Load and split Iris dataset
iris = load_iris()
xTrain, xTest, yTrain, yTest = train_test_split(
    iris.data, iris.target, random_state=0)

# Create and train the decision tree model
tree_clf = DecisionTreeClassifier(random_state=0)
tree_clf.fit(xTrain, yTrain)

# Export an image of the tree
dot_src = export_graphviz(tree_clf, feature_names=iris.feature_names,
                          class_names=iris.target_names, rounded=True,
                          filled=True)
image_filename = os.path.join(IMAGES_PATH, "iris_tree")
Source(dot_src).render(image_filename, format="png", cleanup=True)

# Test the decision tree
accuracy = tree_clf.score(xTest, yTest)
print("Accuracy:", accuracy)
```