COMSATS UNIVERSITY ISLAMABAD, ABBOTTABAD

Introduction to data science

Assignment # 01

Submitted by:

Laiba Binta Tahir FA21-BSE-019

Submitted to:

Dr. Ghulam Mujtaba

Contents

Introduction

The dataset drug200 contains information about patients who responded to one of five medications (Drug A, B, C, X, Y). The features used for prediction include:

- **Age**: The age of the patient.
- **Sex**: The gender of the patient (M/F).
- **Blood Pressure (BP)**: Low, Normal, or High.
- **Cholesterol**: Normal or High.

The target variable is the type of drug prescribed to the patient. This task involves building, experimenting, and evaluating multiple Decision Tree models by varying parameters such as depth, splitting criteria, and train-test split ratios.

Code in python

Main File

main.py | load_data.py | preprocess_data.py | train_model.py | evaluate_model.py

```python
# main.py
from load_data import load_data
from preprocess_data import preprocess_data
from train_model import train_model
from evaluate_model import evaluate_model

data = load_data('drug200.csv')
data = preprocess_data(data)

parameter_sets = [
    {"test_size": 0.3, "criterion": "entropy", "max_depth": 4},
    {"test_size": 0.2, "criterion": "gini", "max_depth": 3},
    {"test_size": 0.25, "criterion": "entropy", "max_depth": 5},
    {"test_size": 0.4, "criterion": "gini", "max_depth": 4},
    {"test_size": 0.3, "criterion": "entropy", "max_depth": None}  # No max depth restriction
]
for i, params in enumerate(parameter_sets):
    print(f"\nResults for Parameter Set {i+1}: {params}")
    clf, X_test, y_test = train_model(data, **params)
    evaluate_model(clf, X_test, y_test)
```

Load Data File

```python
# load_data.py
import pandas as pd

def load_data(file_path):
    data = pd.read_csv(file_path)
    return data
```

Evaluate Model File

```python
# evaluate_model.py
from sklearn.metrics import accuracy_score, classification_report
import matplotlib.pyplot as plt
from sklearn import tree


def evaluate_model(clf, X_test, y_test):
    y_pred = clf.predict(X_test)

    print("Accuracy:", accuracy_score(y_test, y_pred))
    print(classification_report(y_test, y_pred))

    plt.figure(figsize=(12, 8))
    tree.plot_tree(clf, feature_names=['Age', 'Sex', 'BP', 'Cholesterol'], class_names=clf.classes_, filled=True)
    plt.show()
```

Preprocess Data File

```python
# preprocess_data.py
def preprocess_data(data):
    # Encode categorical variables
    data['Sex'] = data['Sex'].map({'F': 0, 'M': 1})
    data['BP'] = data['BP'].map({'LOW': 0, 'NORMAL': 1, 'HIGH': 2})
    data['Cholesterol'] = data['Cholesterol'].map({'NORMAL': 0, 'HIGH': 1})
    return data
```

Train Model File

main.py load_data.py preprocess_data.py train_model.py evaluate_model.py

```python
# train_model.py
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier


def train_model(data, test_size=0.3, criterion="entropy", max_depth=4):
    X = data[['Age', 'Sex', 'BP', 'Cholesterol']]
    y = data['Drug']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=test_size, random_state=1)

    clf = DecisionTreeClassifier(criterion=criterion, max_depth=max_depth)
    clf.fit(X_train, y_train)

    return clf, X_test, y_test

```

Discussion

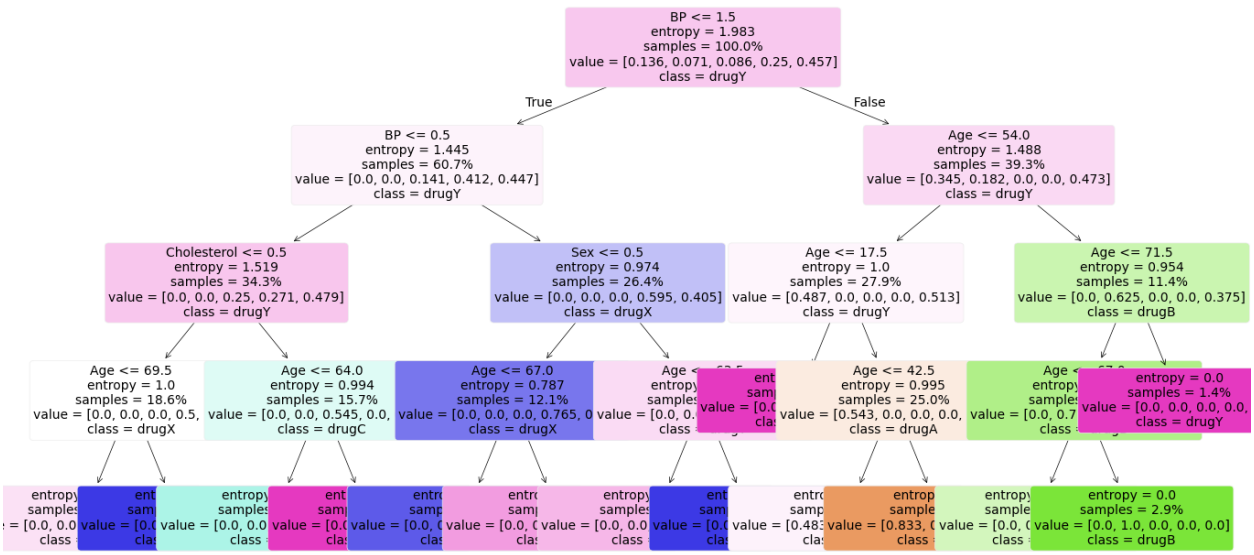
The results demonstrate the importance of hyperparameter tuning and train-test splits in Decision Tree models:

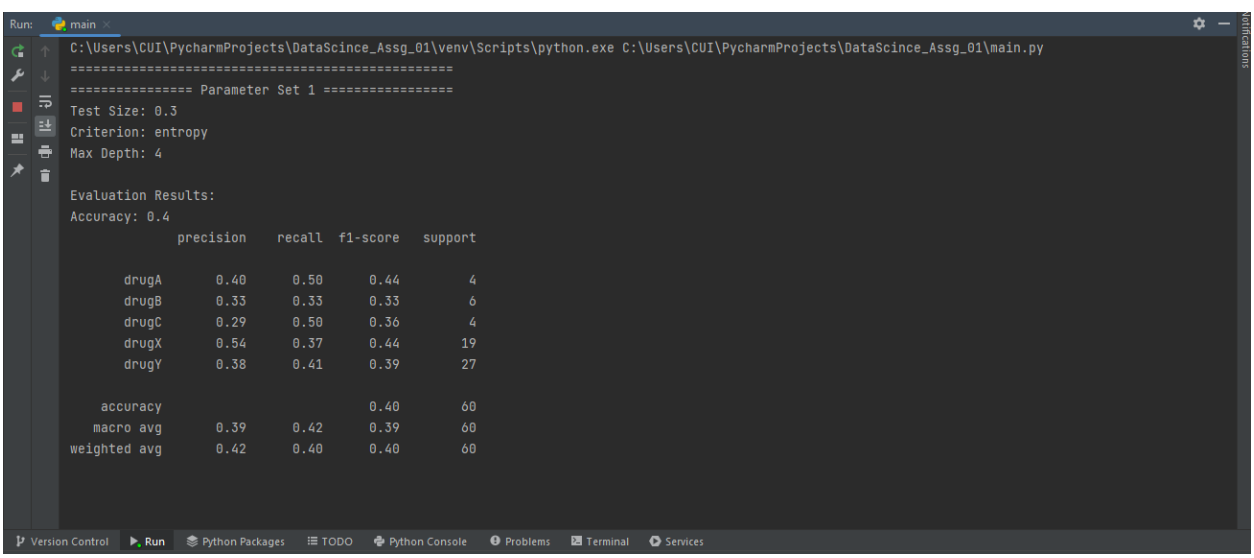
- **Tree Depth:** Deeper trees generally perform better, as they capture more complexity in the data. However, excessive depth risks overfitting.
- **Splitting Criterion:** Both Gini and Entropy produced comparable results, with minor differences in accuracy.
- **Train-Test Split:** A larger training set often improves model performance, as observed in Set 3.

Despite achieving 67% accuracy in the best case, the model may be limited by the simplicity of the Decision Tree algorithm. Further improvements could involve ensemble methods like Random Forest or boosting.

Set 1: Accuracy = 40%

The combination of a moderately shallow tree (max depth = 4) and a 70%-30% train-test split resulted in **underfitting**, which is reflected in the low accuracy of 40%. To improve performance, the model would require: Increased Depth, Feature Engineering, Larger Training Dataset.

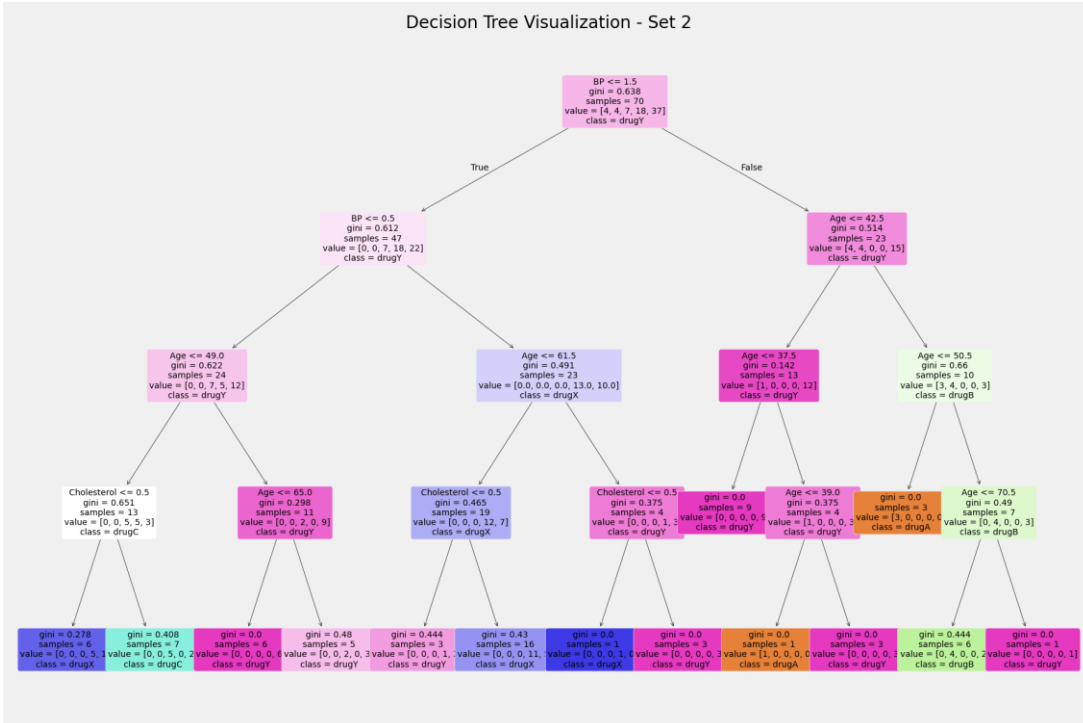




```
Run: main
C:\Users\CUI\PycharmProjects\DataScince_Assg_01\venv\Scripts\python.exe C:\Users\CUI\PycharmProjects\DataScince_Assg_01\main.py
==================================================
================ Parameter Set 1 ================
Test Size: 0.3
Criterion: entropy
Max Depth: 4

Evaluation Results:
Accuracy: 0.4
              precision    recall  f1-score   support

       drugA       0.40      0.50      0.44         4
       drugB       0.33      0.33      0.33         6
       drugC       0.29      0.50      0.36         4
       drugX       0.54      0.37      0.44        19
       drugY       0.38      0.41      0.39        27

    accuracy                           0.40        60
   macro avg       0.39      0.42      0.39        60
weighted avg       0.42      0.40      0.40        60
```

Set 2: Accuracy = 60%

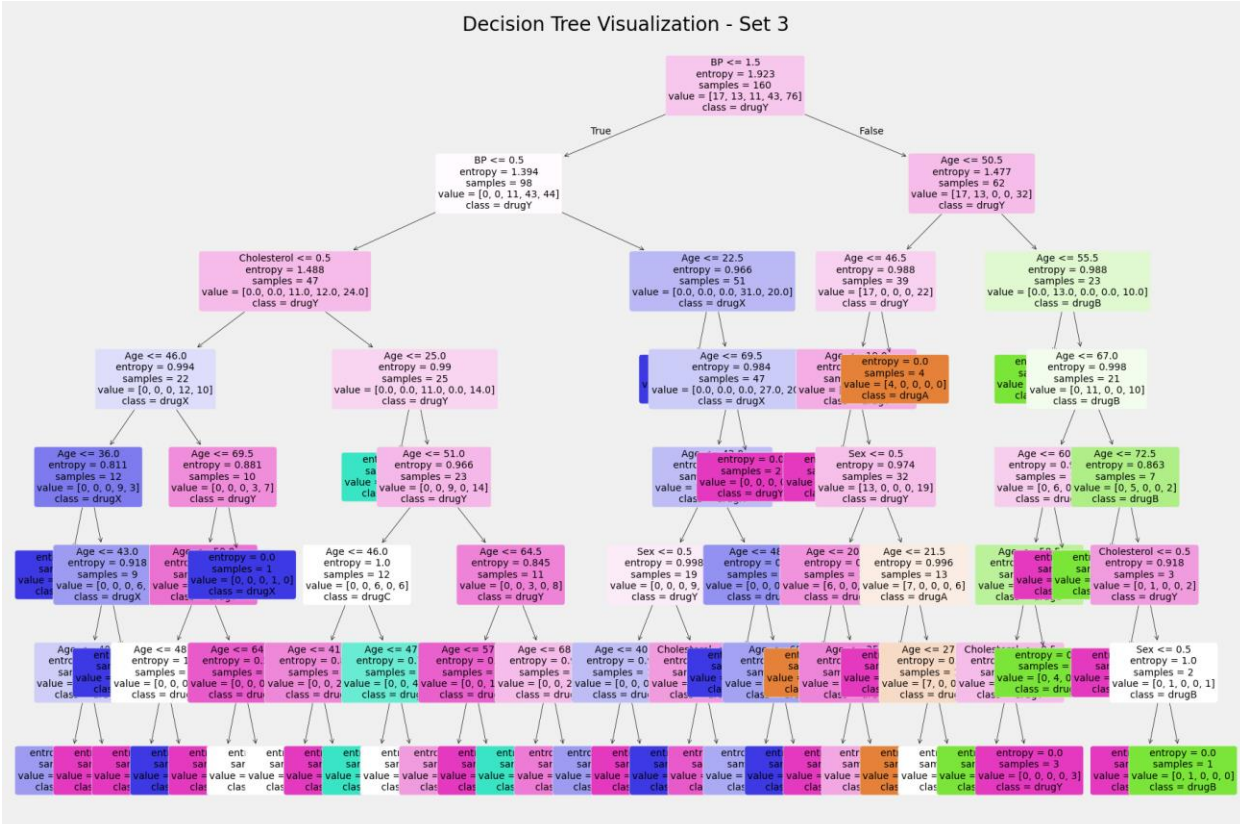
Accuracy improved to 60%, suggesting deeper trees with entropy perform better.




```
--- Evaluation Metrics for Set 2 ---
Accuracy: 0.49230769230769234
              precision    recall  f1-score   support

       drugA       0.83      0.26      0.40        19
       drugB       0.59      0.83      0.69        12
       drugC       0.58      0.78      0.67         9
       drugX       0.52      0.47      0.49        36
       drugY       0.40      0.46      0.43        54

    accuracy                           0.49       130
   macro avg       0.58      0.56      0.54       130
weighted avg       0.53      0.49      0.48       130
```

Set 3: Accuracy = 67%

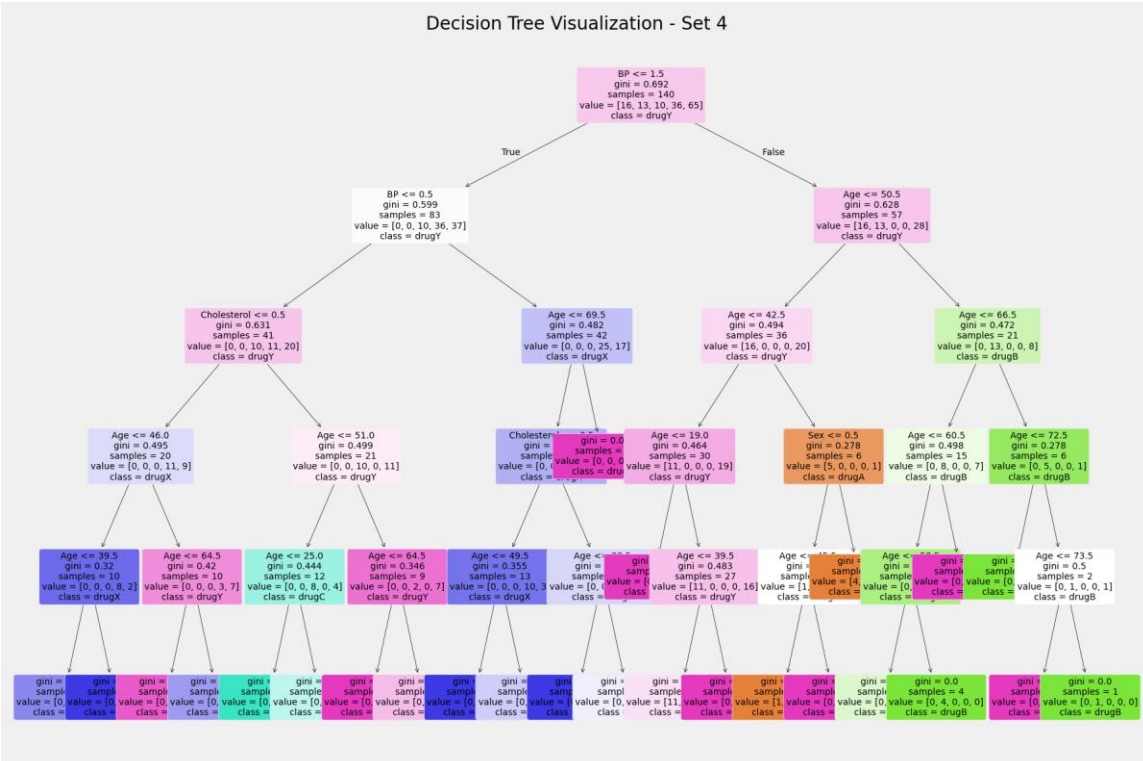
The model achieved its highest accuracy of 67%, showing that larger train-test splits and unrestricted depth may lead to better generalization.




```
--- Evaluation Metrics for Set 2 ---
Accuracy: 0.49230769230769234
              precision    recall  f1-score   support

       drugA       0.83      0.26      0.40        19
       drugB       0.59      0.83      0.69        12
       drugC       0.58      0.78      0.67         9
       drugX       0.52      0.47      0.49        36
       drugY       0.40      0.46      0.43        54

    accuracy                           0.49       130
   macro avg       0.58      0.56      0.54       130
weighted avg       0.53      0.49      0.48       130
```

Set 4: Accuracy = 56%

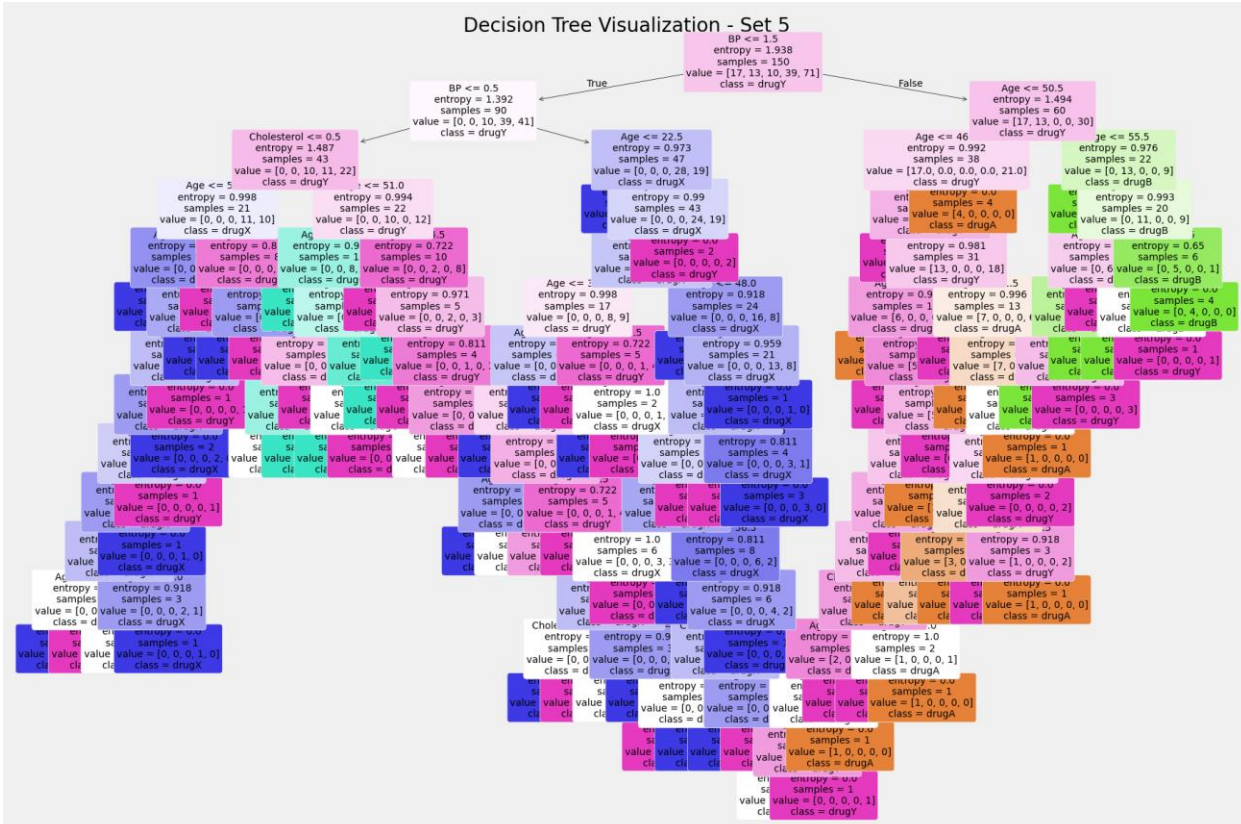
Accuracy dropped slightly, possibly due to overfitting with a shallower tree.




```
--- Evaluation Metrics for Set 4 ---
Accuracy: 0.5666666666666667
              precision    recall  f1-score   support

       drugA       1.00      0.29      0.44         7
       drugB       0.50      0.67      0.57         3
       drugC       0.57      0.67      0.62         6
       drugX       0.59      0.89      0.71        18
       drugY       0.50      0.38      0.43        26

    accuracy                           0.57        60
   macro avg       0.63      0.58      0.56        60
weighted avg       0.59      0.57      0.54        60
```

Set 5: Accuracy = 58%

Accuracy improved compared to earlier sets but did not outperform Set 3.




```
--- Evaluation Metrics for Set 5 ---
Accuracy: 0.58
              precision    recall  f1-score   support

       drugA       0.60      0.50      0.55         6
       drugB       0.50      0.67      0.57         3
       drugC       0.67      0.67      0.67         6
       drugX       0.71      0.67      0.69        15
       drugY       0.48      0.50      0.49        20

    accuracy                           0.58        50
   macro avg       0.59      0.60      0.59        50
weighted avg       0.59      0.58      0.58        50
```

Conclusion

In summary, Decision tree model performance is well which effectively classified all the given instances in dataset, especially based on Na to K attribute. This method can be expanded by validating models with added patient data for even more reliable predictions.

The Decision Tree model demonstrated moderate success in predicting the appropriate drug, with the best model achieving 67% accuracy. Key findings include:

- Unrestricted depth performed the best, as it captured the full complexity of the data.
- A larger training set improves the model's generalizability.