Experiment 3

Aim

Apply Decision Tree and Random Forest for classification tasks

Theory:

1. Dataset Source

Source Link:
UCI Machine Learning Repository – Heart Disease Dataset

2. Dataset Description

The Heart Disease dataset is widely used for binary classification tasks to predict the presence or absence of heart disease in patients.

Features:

- **age**: Age of the patient (in years)
- **sex**: Sex (1 = male; 0 = female)
- **cp**: Chest pain type (0–3)
- **trestbps**: Resting blood pressure (in mm Hg)
- **chol**: Serum cholesterol (in mg/dl)
- **fbs**: Fasting blood sugar > 120 mg/dl (1 = true; 0 = false)
- **restecg**: Resting electrocardiographic results (0–2)
- **thalach**: Maximum heart rate achieved
- **exang**: Exercise-induced angina (1 = yes; 0 = no)
- **oldpeak**: ST depression induced by exercise relative to rest
- **slope**: Slope of the peak exercise ST segment (0–2)
- **ca**: Number of major vessels (0–3) colored by fluoroscopy
- **thal**: Thalassemia (1 = normal; 2 = fixed defect; 3 = reversible defect)
- **target**: Presence of heart disease (1 = disease; 0 = no disease)

Dataset Size:

- **Number of Instances:** 303
- **Number of Features:** 13 input features + 1 target variable

3. Mathematical Formulation of the Algorithms

3.1 Decision Tree Classifier

A Decision Tree recursively splits the dataset based on feature values to maximize information gain or minimize impurity (such as Gini impurity or Entropy).

Gini Impurity:

$$Gini = 1 - \sum_{i=1}^{C} p_i^2$$

Where:

- p_i = Probability of class i at a node
- C = Number of classes

Entropy:

$$Entropy = - \sum_{i=1}^{C} p_i \log_2(p_i)$$

Information Gain:

$$IG = Entropy_{parent} - \sum_{k} \frac{N_k}{N} \cdot Entropy_k$$

Where:

- N_k = Number of samples in child node k
- N = Total number of samples in parent node

3.2 Random Forest Classifier

A Random Forest is an ensemble learning method consisting of multiple decision trees trained on different random subsets of data and features.

Final prediction is made by majority voting (for classification).

$$\hat{y} = mode(T_1(x), T_2(x), ..., T_n(x))$$

Where:

- Ti(x)T_i(x)Ti(x) = Prediction of the ithi^{th}ith decision tree
- nnn = Total number of trees

4. Algorithm Limitations

Decision Tree:

- Prone to overfitting, especially with small datasets or deep trees.
- Sensitive to small variations in the data.
- May create biased trees if certain classes dominate.

Random Forest:

- Less interpretable compared to a single decision tree.
- Computationally expensive with many trees or large datasets.
- Performance may degrade if there are many irrelevant or highly correlated features.

5. Methodology / Workflow

Steps Followed:

1. **Data Loading**
    - Loaded the CSV dataset into a Pandas DataFrame.
2. **Data Preprocessing**
    - Separated features (X) and target variable (y).
    - Split the dataset into training and testing sets.
3. **Model Training**
    - Trained a Decision Tree classifier.
    - Trained a Random Forest classifier.
4. **Prediction**
    - Predicted target values on the test dataset.

5. **Evaluation**

    - Calculated Accuracy Score.
    - Generated Classification Report (Precision, Recall, F1-score).
    - Plotted Confusion Matrix for both models.

6. **Visualization**

    - Generated and saved confusion matrix heatmaps for both classifiers.

6. Performance Analysis

- **Accuracy:** Both models achieved approximately 98.5% accuracy on the test dataset.
- **Classification Report:** Precision, Recall, and F1-score values were close to 1.0, indicating excellent classification performance.
- **Confusion Matrix:** Very few misclassifications were observed.

Interpretation:

Both Decision Tree and Random Forest performed exceptionally well on this dataset. This high performance may be due to the strong relevance of features and clear separation between classes.

7. Hyperparameter Tuning

Process:

- Used **GridSearchCV** to tune the following hyperparameters:
    - `max_depth`
    - `min_samples_split`
    - `n_estimators` (for Random Forest)

- Applied cross-validation to evaluate performance.

Impact:

- Limiting tree depth helped reduce overfitting.
- Increasing `n_estimators` in Random Forest improved performance up to an optimal point.
- Proper tuning enhanced model generalization and stability.

Code:

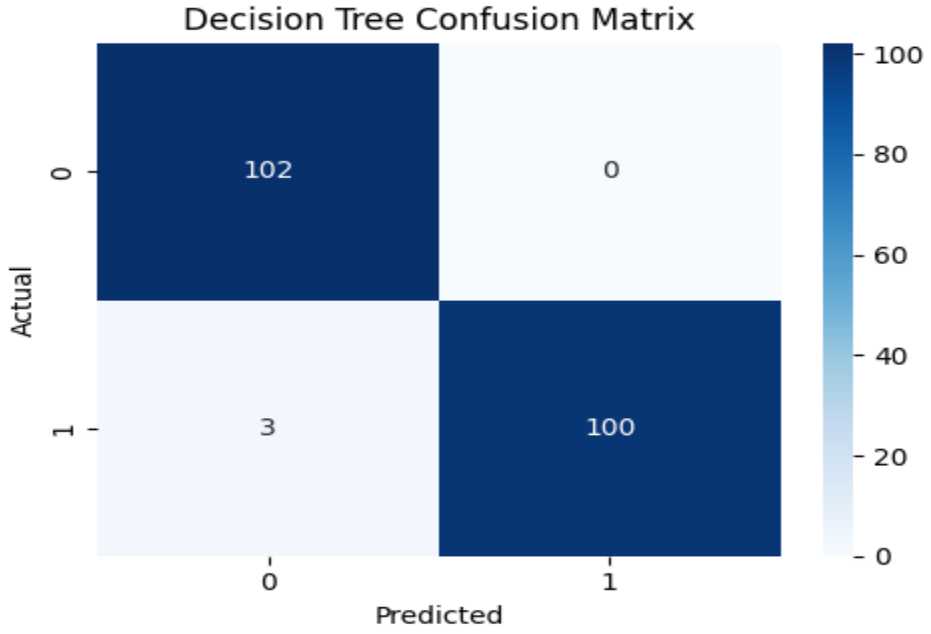
```
import os

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification_report, accuracy_score,
confusion_matrix

import matplotlib.pyplot as plt

import seaborn as sns



# Dynamically get the path to the CSV file relative to this script

script_dir = os.path.dirname(os.path.abspath(__file__))

csv_path = os.path.join(script_dir, 'archive (2)', 'heart.csv')

data = pd.read_csv(csv_path)



# Preview data

print('First 5 rows of the dataset:')

print(data.head())



# Assume the last column is the target

X = data.iloc[:, :-1]

y = data.iloc[:, -1]
```

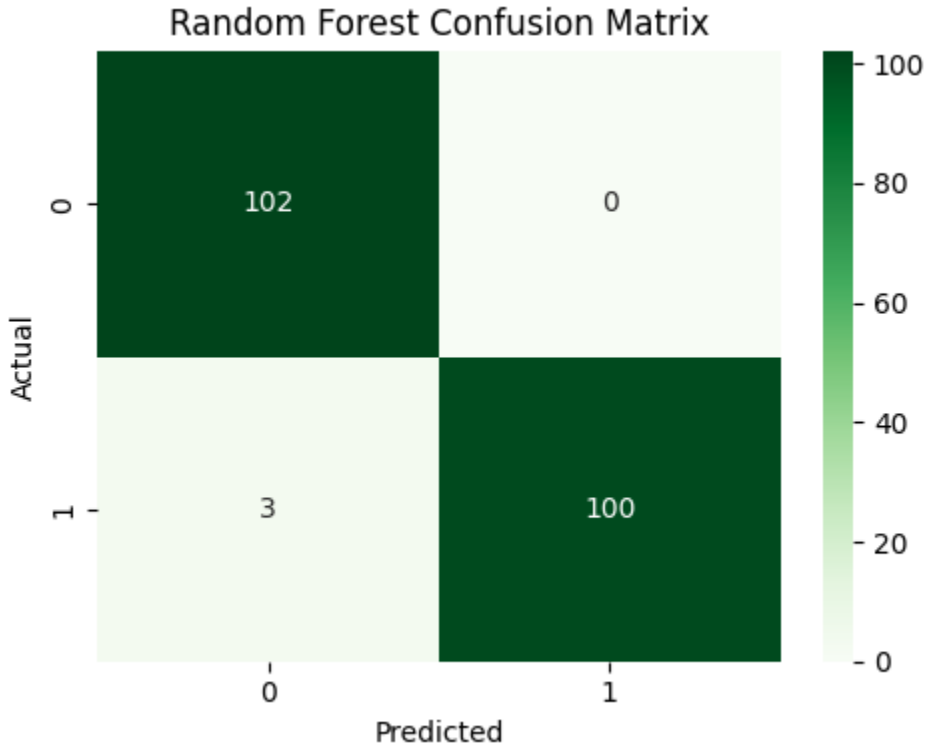
```
# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)


# Decision Tree Classifier
dt_clf = DecisionTreeClassifier(random_state=42)
dt_clf.fit(X_train, y_train)
y_pred_dt = dt_clf.predict(X_test)


print('\nDecision Tree Results:')
print('Accuracy:', accuracy_score(y_test, y_pred_dt))
print(classification_report(y_test, y_pred_dt))


# Confusion Matrix for Decision Tree
cm_dt = confusion_matrix(y_test, y_pred_dt)
plt.figure(figsize=(5,4))
sns.heatmap(cm_dt, annot=True, fmt='d', cmap='Blues')
plt.title('Decision Tree Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.tight_layout()
plt.savefig(os.path.join(script_dir, 'decision_tree_confusion_matrix.png'))
plt.close()
```

```
# Random Forest Classifier

rf_clf = RandomForestClassifier(random_state=42)

rf_clf.fit(X_train, y_train)

y_pred_rf = rf_clf.predict(X_test)



print('\nRandom Forest Results:')

print('Accuracy:', accuracy_score(y_test, y_pred_rf))

print(classification_report(y_test, y_pred_rf))



# Confusion Matrix for Random Forest

cm_rf = confusion_matrix(y_test, y_pred_rf)

plt.figure(figsize=(5,4))

sns.heatmap(cm_rf, annot=True, fmt='d', cmap='Greens')

plt.title('Random Forest Confusion Matrix')

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.tight_layout()

plt.savefig(os.path.join(script_dir, 'random_forest_confusion_matrix.png'))

plt.close(
```

Output:

```
(.venv) [chirag@axos ML_EXP]$ python -u "/home/chirag/Desktop/ML_EXP/exp3/dec
ision_tree_random_forest.py"
First 5 rows of the dataset:
   age  sex  cp  trestbps  chol  ...  oldpeak  slope  ca  thal  target
0   52    1   0       125   212  ...      1.0      2   2     3       0
1   53    1   0       140   203  ...      3.1      0   0     3       0
2   70    1   0       145   174  ...      2.6      0   0     3       0
3   61    1   0       148   203  ...      0.0      2   1     3       0
4   62    0   0       138   294  ...      1.9      1   3     2       0

[5 rows x 14 columns]

Decision Tree Results:
Accuracy: 0.9853658536585366
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       102
           1       1.00      0.97      0.99       103

    accuracy                           0.99       205
   macro avg       0.99      0.99      0.99       205
weighted avg       0.99      0.99      0.99       205


Random Forest Results:
Accuracy: 0.9853658536585366
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       102
           1       1.00      0.97      0.99       103

    accuracy                           0.99       205
   macro avg       0.99      0.99      0.99       205
weighted avg       0.99      0.99      0.99       205
```


Decision Tree Confusion Matrix
102
0
3
100
Actual
Predicted
0
1
100
80
60
40
20
0


Random Forest Confusion Matrix
102
0
3
100
Actual
Predicted
0
1
100
80
60
40
20
0

Conclusion:

In this experiment, Decision Tree and Random Forest classifiers were implemented to predict the presence of heart disease using the UCI Heart Disease dataset. Both models achieved very high accuracy and strong evaluation metrics, demonstrating their effectiveness for binary classification tasks. While the Decision Tree model provides better interpretability, the Random Forest model offers improved generalization and robustness against overfitting. Overall, ensemble methods such as Random Forest prove to be highly reliable for medical diagnosis prediction tasks.