Batch: IAI-2
Roll Number: 16010422234
Experiment Number: 8
Name: Chandana Ramesh Galgali

Aim of the Experiment: To implement Decision Tree Algorithm (ID3 using library functions)

Program/Steps:

Set up and train a decision tree classifier on the Titanic dataset and see how well the classifier performs on a validation set (80-20 train-test dataset). Find out accuracy and confusion matrix and plot created decision tree with following variations

1. Target Variable: Survived , remaining all input features

2. Target Variable: Survived , selecting subset of features as input

3. Target Variable: Survived , using transformed input feature (e.g. create new feature family = sibsp + parch, weighted_class = pclass*2 if pclass =1 ; pclass*3 if pclass =2 ; pclass*4 if pclass =3 etc)

Program:

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score, confusion_matrix

from sklearn.tree import plot_tree

import matplotlib.pyplot as plt

# Load the dataset

url =
"https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/stuff/titanic.
csv"

data = pd.read_csv(url)

# Preprocessing
```

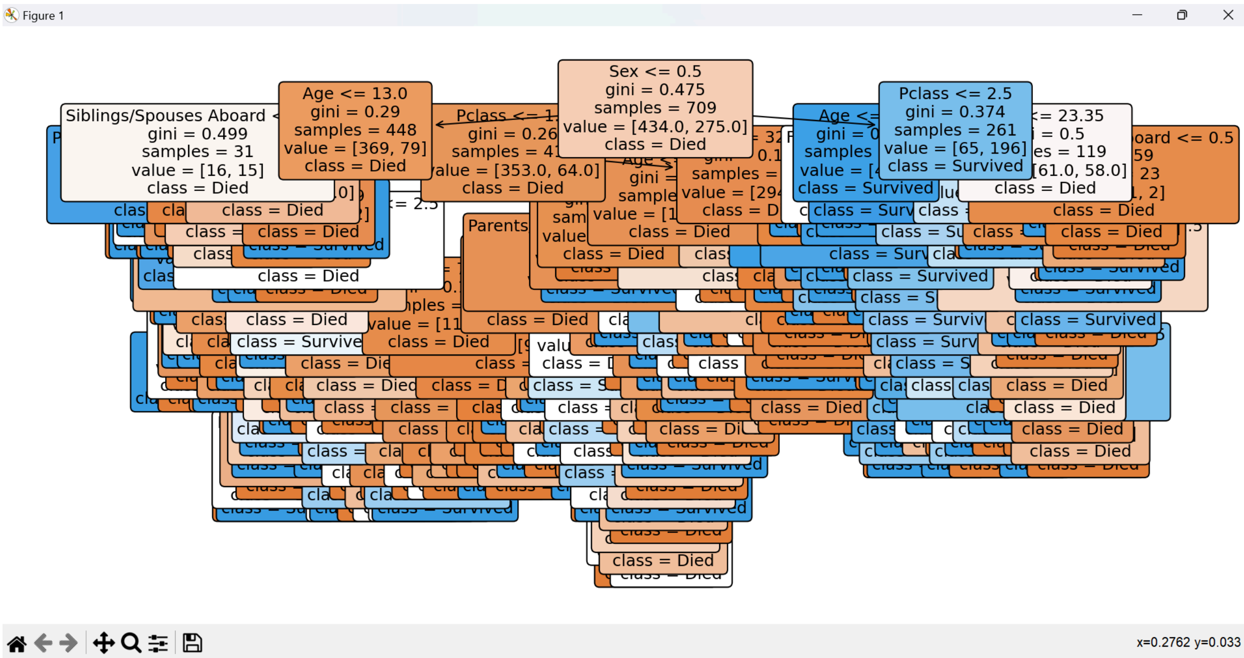
```
data['Age'].fillna(data['Age'].median(), inplace=True)

# Convert 'Sex' to a numeric variable

data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})

# Variation-1: Using All Input Features

# Define features and target

features = data.drop(['Survived', 'Name'], axis=1)  # Exclude 'Name' for
being non-predictive

target = data['Survived']

# Train-test split

X_train, X_test, y_train, y_test = train_test_split(features, target,
test_size=0.2, random_state=42)

# Decision Tree Classifier

classifier = DecisionTreeClassifier()

classifier.fit(X_train, y_train)

predictions = classifier.predict(X_test)

print("Using All Input Features:")

# Metrics

print("Accuracy:", accuracy_score(y_test, predictions))

print("Confusion Matrix:\n", confusion_matrix(y_test, predictions))

# Plot

plt.figure(figsize=(20,10))

plot_tree(classifier, filled=True, feature_names=features.columns,
class_names=['Died', 'Survived'], rounded=True, fontsize=12)

plt.show()

# Variation-2: Using a Subset of Features
```

```
features_subset = data[['Age', 'Fare', 'Sex']]  # Selected subset of features

# Train-test split

X_train, X_test, y_train, y_test = train_test_split(features_subset, target, test_size=0.2, random_state=42)

# Decision Tree Classifier

classifier_subset = DecisionTreeClassifier()

classifier_subset.fit(X_train, y_train)

predictions_subset = classifier_subset.predict(X_test)

print("\nUsing a Subset of Features:")

# Metrics

print("Accuracy:", accuracy_score(y_test, predictions_subset))

print("Confusion Matrix:\n", confusion_matrix(y_test, predictions_subset))

# Plot

plt.figure(figsize=(20,10))

plot_tree(classifier_subset, filled=True, feature_names=features_subset.columns, class_names=['Died', 'Survived'], rounded=True, fontsize=12)

plt.show()

# Variation-3: Using Transformed Input Features

# Feature engineering

data['Family'] = data['Siblings/Spouses Aboard'] + data['Parents/Children Aboard']

data['Weighted_Class'] = data['Pclass'].apply(lambda x: x*2 if x == 1 else (x*3 if x == 2 else x*4))

# Define features with transformations

features_transformed = data[['Age', 'Fare', 'Sex', 'Family', 'Weighted_Class']]
```

```python
# Train-test split

X_train, X_test, y_train, y_test = train_test_split(features_transformed,
target, test_size=0.2, random_state=42)

# Decision Tree Classifier

classifier_transformed = DecisionTreeClassifier()

classifier_transformed.fit(X_train, y_train)

predictions_transformed = classifier_transformed.predict(X_test)

print("\nUsing Transformed Input Features:")

# Metrics

print("Accuracy:", accuracy_score(y_test, predictions_transformed))

print("Confusion Matrix:\n", confusion_matrix(y_test,
predictions_transformed))

# Plot

plt.figure(figsize=(20,10))

plot_tree(classifier_transformed, filled=True,
feature_names=features_transformed.columns, class_names=['Died',
'Survived'], rounded=True, fontsize=12)

plt.show()
```

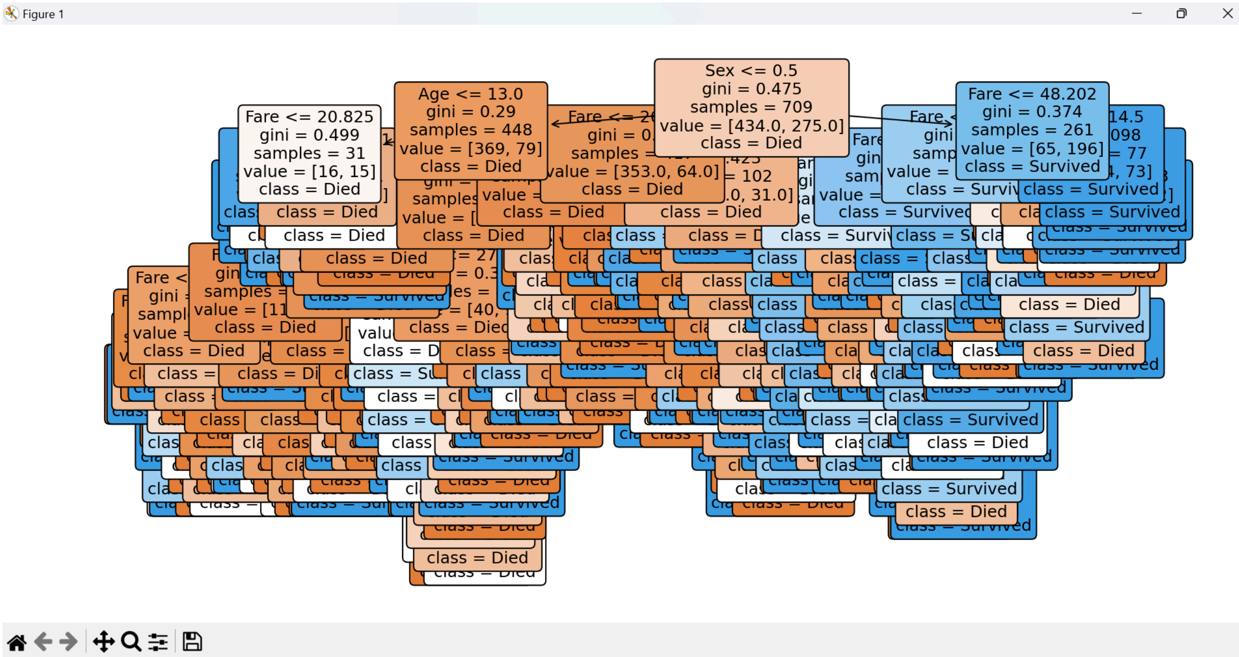
Output/Result: Variation-1

```
Using All Input Features:
Accuracy: 0.7078651685393258
Confusion Matrix:
 [[82 29]
 [23 44]]
```



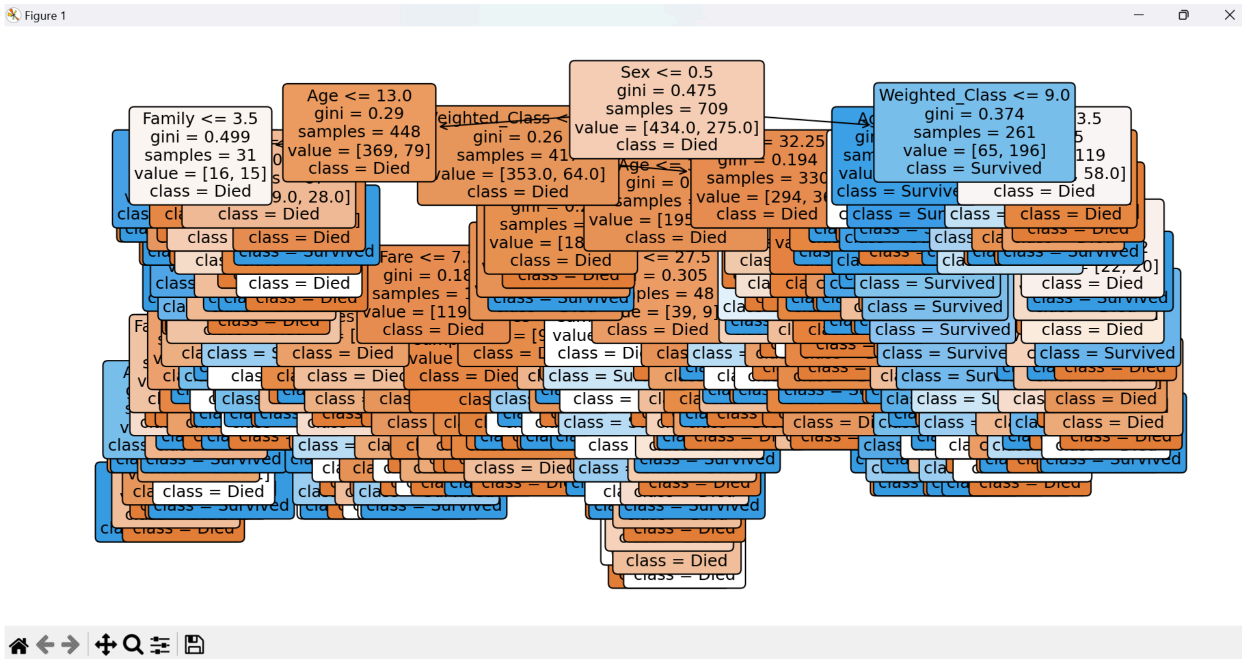
Variation-2

```
Using a Subset of Features:
Accuracy: 0.7303370786516854
Confusion Matrix:
 [[83 28]
 [20 47]]
```



Variation-3

```
Using Transformed Input Features:
Accuracy: 0.7528089887640449
Confusion Matrix:
 [[87 24]
 [20 47]]
```



Outcomes: Understand fundamentals of learning in AI.

Conclusion (Based on the Results and outcomes achieved):

Using the decision tree model on the Titanic dataset allows for straightforward interpretation of survival factors through a visual representation. The accuracy metric and confusion matrix provide quantitative measures of the model's performance, highlighting its strengths and areas for improvement. Feature engineering played a crucial role in enhancing the model's predictive power by introducing new, meaningful features based on existing data. This method offers a robust approach to understanding complex patterns in the data, making it a valuable tool for both predictive modeling and exploratory data analysis.

References:

1. Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, Second Edition, Pearson Publication
2. Jason Brownlee , Master Machine Learning Algorithms, eBook, 2017 ,Edition v1.12.
