Experiment 6

Construct Decision Tree and Random Forest models for Iris Flower Classification.

Aim: Construct Decision Tree and Random Forest models for Iris Flower Classification.

Theory:

Decision Tree :-

A Decision Tree is a supervised learning algorithm used for both classification and regression tasks. It works by splitting the data into subsets based on the value of input features. This process is represented visually as a tree, where internal nodes represent decision points based on features, and leaf nodes represent the outcome or class label. The tree continues splitting until it reaches pure leaf nodes or a stopping criterion. Splits are typically made using criteria like Gini Index or Entropy.

- **Root Node:** The top node that contains the best split.
- **Internal Node:** Represents a feature and a condition.
- **Leaf Node:** Represents a class label (prediction).

Decision Trees are simple to understand and interpret but are prone to overfitting, especially when deep.

Advantages :-

- Easy to understand and interpret
- No need for feature scaling or normalization
- Works for both classification and regression
- Handles both numerical and categorical data
- Can model non-linear relationships

Disadvantages :-

- Prone to overfitting
- Unstable to small data changes
- Biased with imbalanced data

Random Forest :-

A Random Forest is a supervised learning algorithm that builds a collection of decision trees and combines their outputs to improve overall performance. It operates by creating multiple trees during training, where each tree is trained on a different random subset of the data using a method called bootstrap sampling. Additionally, at each node, only a random subset of features is considered for splitting, which introduces diversity among the trees. The final prediction is made by majority voting in classification tasks or averaging in regression tasks. This ensemble approach reduces the risk of overfitting and increases the model's robustness and accuracy. Random Forest is particularly effective for high-dimensional data and handles both classification and regression problems efficiently.

Advantages :-

- Reduces overfitting compared to single decision trees
- Provides high accuracy and robust performance
- Works well with both classification and regression tasks

Disadvantages :-

- Slower training and prediction due to multiple trees
- Harder to interpret than a single decision tree
- Requires more memory and computational resources

Iris Dataset:

The Iris flower dataset is a classic dataset in machine learning introduced by Sir Ronald Fisher. It contains 150 samples from three species of Iris (setosa, versicolor, virginica), each described by four features:

- Sepal length
- Sepal width
- Petal length
- Petal width

CONCLUSION:

Decision Tree and Random Forest algorithms are powerful and reliable tools for classification problems. A Decision Tree is simple, easy to interpret, and works well for understanding how decisions are made based on input features. However, it can overfit the training data if not properly pruned. Random Forest, on the other hand, builds multiple decision trees and combines their predictions to improve accuracy and reduce overfitting, making it more robust and stable. When applied to the Iris dataset, which has clearly defined class boundaries and balanced features, both models achieve high performance. While the Decision Tree offers transparency, the Random Forest provides better generalization, making both algorithms valuable depending on the use case.

VIVA QUESTIONS:

1. What is the main difference between Decision Tree and Random Forest algorithms?
2. What criteria are commonly used for splitting nodes in a Decision Tree?
3. How does Random Forest reduce overfitting compared to a single Decision Tree?
4. What is bootstrap sampling in the context of Random Forest?
5. Why might Random Forest be preferred over a single Decision Tree in real-world applications?

PROGRAM:

 # Import required libraries

import numpy as np

import pandas as pd

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
import matplotlib.pyplot as plt

# Step 1: Load the Iris dataset
iris = load_iris()
X = iris.data          # Features
y = iris.target        # Target labels

# Step 2: Split into training and testing sets (70% train, 30% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

# Step 3: Train Decision Tree Classifier
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)
dt_pred = dt_model.predict(X_test)
```

 # Step 4: Evaluate Decision Tree Model

```
print("\n--- Decision Tree ---")

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

print("Classification Report:\n", classification_report(y_test, dt_pred))
```

 # Step 5: Visualize the Decision Tree

```
plt.figure(figsize=(12, 8))

plot_tree(dt_model, filled=True, feature_names=iris.feature_names,
class_names=iris.target_names)

plt.title("Decision Tree Visualization")

plt.show()
```

 # Step 6: Train Random Forest Classifier

```
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)

rf_model.fit(X_train, y_train)

rf_pred = rf_model.predict(X_test)
```

 # Step 7: Evaluate Random Forest Model

```
print("\n--- Random Forest ---")

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

print("Classification Report:\n", classification_report(y_test, rf_pred))
```