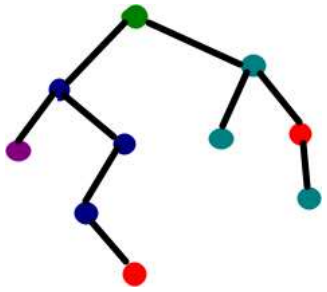
EX.NO : **DATE :**

IMPLEMENTATION OF DECISION TREE CLASSIFICATION TECHNIQUES

Decision Tree is one of the most powerful and popular algorithm. Decision-tree algorithm falls under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables.



AIM:

To implement a decision tree classification technique for gender classification using python.

EXPLANATION:

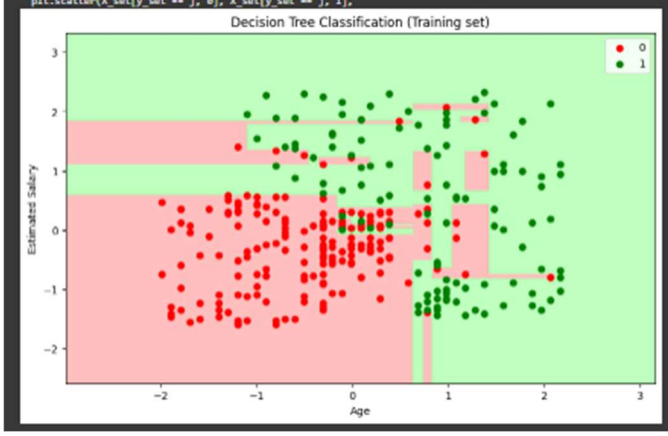
- Import tree from sklearn.
- Call the function DecisionTreeClassifier() from tree
- Assign values for X and Y.
- Call the function predict for Predicting on the basis of given random values for each given feature.
- Display the output.

CODE:

```
# Step 1: Mount Google Drive
from google.colab import drive
drive.mount('/content/gdrive')

# Step 2: Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix
from matplotlib.colors import ListedColormap

# Step 3: Load the dataset (update path if necessary)
dataset = pd.read_csv('/content/gdrive/My Drive/Colab Datasets/Social_Network_Ads.csv')

# Step 4: Prepare features (X) and target variable (y)
X = dataset.iloc[:, [2, 3]].values  # Assumes Age and EstimatedSalary are in columns 2 and 3
y = dataset.iloc[:, -1].values      # Assumes Purchased is the last column

# Step 5: Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)

# Step 6: Feature scaling
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Step 7: Train the Decision Tree Classifier
classifier = DecisionTreeClassifier(criterion='entropy', random_state=0)
classifier.fit(X_train, y_train)

# Step 8: Make predictions and print the confusion matrix
y_pred = classifier.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(cm)

# Step 9: Visualize Decision Tree Classification results on the training set
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(
    np.arange(start=X_set[:, 0].min() - 1, stop=X_set[:, 0].max() + 1, step=0.01),
    np.arange(start=X_set[:, 1].min() - 1, stop=X_set[:, 1].max() + 1, step=0.01)
)

plt.figure(figsize=(10, 6))
# Update ListedColormap definition here to match format
cmap_background = ListedColormap(['#FFAAAA', '#AAFFAA'])
cmap_points = ListedColormap(['red', 'green'])

# Decision boundary
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha=0.75, cmap=cmap_background)
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())

# Plotting data points
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c=cmap_points(i), label=j)

plt.title('Decision Tree Classification (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

OUTPUT:

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force_remount=True).
Confusion Matrix:
[[62  6]
 [ 3 29]]
<ipython-input-11-97844aa55b44b>:68: UserWarning: *c* argument looks like a single numeric RGB or RGBA sequence, which should be avoi
  plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
```



RESULT:

Thus Program is Executed Successfully And Output is Verified.