EXP. NO 9) DECISION TREE CLASSIFICATION

AIM :-

To classify the social network dataset using Decision tree analysis.

PSEUDOCODE:

1) Import libraries & locate the dataset
2) Define X (features) & y (label)
3) Scale features with std scaler.
4) Initialize & train DecisionTree classifier using entropy.
5) Predict on test Data and generate a confusion matrix
6) Prepare x_set & y_set for visualization

CODE:-

```
from google.colab import drive
drive.mount("/content/gdrive")

import pandas as pd
import numpy as np
from 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 sklearn.colors import ListedColors.

dataset = pd.read_csv("/content/gdrive/
                       MyDrive/Social Nw.csv")

# target values
X = dataset.iloc[:,[2,3]].values
y = dataset.iloc[:,-1]

# Split to tra. & test
X_train, X_test, y_train, y_test = train_test_split
(X, y, test_size=0.25, random_state=0)

sc = StandardScaler.
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
classifier = DecisionTreeClassifier(criterion = 'entropy',
                                    random_state = 0)

classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)

cm = confusion_matrix(y_train, y_test)
print("Confusion Matrix")
print(cm)

# visualize decision boundaries
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.contourf(X1, X2, classifier.predict(
  np.array([X1.ravel(), X2.ravel()]).T).reshape
  (X1.shape), alpha = 0.75, cmap = ListedColormap
                                   ('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())

# plot the 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 = ListedColormap(('red', 'green'))(i),
                                  label = j
```

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

Result :- The program is successfully executed and output is verified