Experiment NO. 6

Decision Tree Classification

Importing the libraries

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Importing the dataset

In [2]:
```
dataset = pd.read_csv('Social_Network_Ads.csv')
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, -1].values
```

Splitting the dataset into the Training set and Test set

In [3]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_
```

Feature Scaling

In [4]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

Training the Decision Tree Classification model on the Training set

In [5]:
```
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)
```

Out[5]:
```
DecisionTreeClassifier(criterion='entropy', random_state=0)
```

Predicting the Test set results

In [6]:
```
y_pred = classifier.predict(X_test)
```

Making the Confusion Matrix

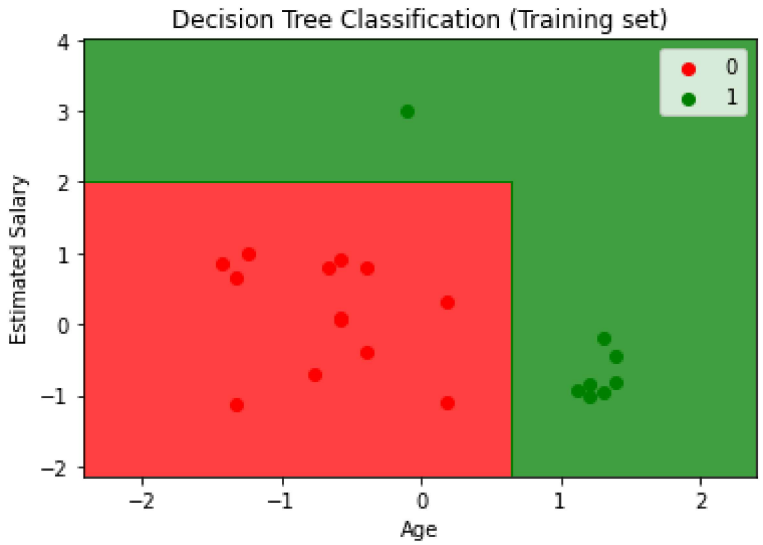
In [7]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[3 0]
 [0 5]]
```

Visualising the Training set results

In [8]:
```
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].ma
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].ma
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).resha
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
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()
```

```
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avo
ided as value-mapping will have precedence in case its length matches with *x* & *
y*.  Please use the *color* keyword-argument or provide a 2-D array with a single
row if you intend to specify the same RGB or RGBA value for all points.
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avo
ided as value-mapping will have precedence in case its length matches with *x* & *
y*.  Please use the *color* keyword-argument or provide a 2-D array with a single
row if you intend to specify the same RGB or RGBA value for all points.
```



Visualising the Test set results

In [9]:
```
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].ma
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].ma
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).resha
```

```
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
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 (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

```
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avo
ided as value-mapping will have precedence in case its length matches with *x* & *
y*.  Please use the *color* keyword-argument or provide a 2-D array with a single
row if you intend to specify the same RGB or RGBA value for all points.
*c* argument looks like a single numeric RGB or RGBA sequence, which should be avo
ided as value-mapping will have precedence in case its length matches with *x* & *
y*.  Please use the *color* keyword-argument or provide a 2-D array with a single
row if you intend to specify the same RGB or RGBA value for all points.
```