Name: Madhusudan D. Pangarkar

Class: TEIT Roll No.: 41

Sub: Laboratory Practice-I (Machine Learning) (314448)

------------------------------------------------------------------------------------------------

**Assignment No-03) Assignment on Classification Technique**

**Code:**

import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
# %% [markdown]  
# ## Prediction Models : Classification Algorithm (Supervised Machine Learning)  
#  
# 4. Decision Tree Classification  
  
# %% [markdown]  
# ### Preparing Data for Classification  
  
# %% [markdown]  
# If a candidate's Chance of Admit is greater than 80%, the candidate will receive the 1 label.  
#  
# If a candidate's Chance of Admit is less than or equal to 80%, the candidate will receive the 0 label.  
  
# %% [code]  
# reading the dataset  
df = pd.read\_csv("E:\ML Practicals\Admission\_Predict.csv")  
print("shape of dataset",df.shape)  
  
# %% [code]  
# it may be needed in the future.  
serialNo = df["Serial No."].values  
df.drop(["Serial No."], axis=1, inplace=True)  
  
df = df.rename(columns={'Chance of Admit ': 'Chance of Admit'})  
  
# %% [code]  
X = df.drop(["Chance of Admit"], axis=1)  
y = df["Chance of Admit"].values  
  
# %% [code]  
# separating train (80%) and test (%20) sets  
from sklearn.model\_selection import train\_test\_split  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.20, random\_state=101)  
  
# %% [code]  
# normalization  
from sklearn.preprocessing import MinMaxScaler  
  
scalerX = MinMaxScaler(feature\_range=(0, 1))  
X\_train[X\_train.columns] = scalerX.fit\_transform(X\_train[X\_train.columns])  
X\_test[X\_test.columns] = scalerX.transform(X\_test[X\_test.columns])  
  
# %% [code]  
y\_train\_01 = [1 if each > 0.8 else 0 for each in y\_train]  
y\_test\_01 = [1 if each > 0.8 else 0 for each in y\_test]  
  
# list to array  
y\_train\_01 = np.array(y\_train\_01)  
y\_test\_01 = np.array(y\_test\_01)  
  
# %% [markdown]  
# ### 4. Decision Tree Classification  
  
# %% [code]  
from sklearn.tree import DecisionTreeClassifier  
  
dtc = DecisionTreeClassifier()  
dtc.fit(X\_train, y\_train\_01)  
y\_pred\_dtc = dtc.predict(X\_test)  
print("score: ", dtc.score(X\_test, y\_test\_01))  
  
# %% [code]  
# confusion matrix  
from sklearn.metrics import confusion\_matrix  
  
cm\_dtc = confusion\_matrix(y\_test\_01, y\_pred\_dtc)  
print("confusion matrix for test data:decision tree\n",cm\_dtc)  
# print("y\_test\_01 == 1 :" + str(len(y\_test\_01[y\_test\_01==1]))) # 29  
cm\_dtc  
  
# %% [code]  
# cm visualization  
import seaborn as sns  
import matplotlib.pyplot as plt  
  
f, ax = plt.subplots(figsize=(5, 5))  
sns.heatmap(cm\_dtc, annot=True, linewidths=0.5, linecolor="red", fmt=".0f", ax=ax)  
plt.title("Test for Test Dataset:decision tree")  
plt.xlabel("predicted y values")  
plt.ylabel("real y values")  
plt.show()  
  
# %% [code]  
from sklearn.metrics import precision\_score, recall\_score  
  
print("precision\_score: ", precision\_score(y\_test\_01, y\_pred\_dtc))  
print("recall\_score: ", recall\_score(y\_test\_01, y\_pred\_dtc))  
  
from sklearn.metrics import f1\_score  
  
print("f1\_score: ", f1\_score(y\_test\_01, y\_pred\_dtc))  
  
# %% [markdown]  
# Test for Train Dataset:  
  
# %% [code]  
cm\_dtc\_train = confusion\_matrix(y\_train\_01, dtc.predict(X\_train))  
print("confusion matrix for test data:decision tree\n",cm\_dtc\_train)  
f, ax = plt.subplots(figsize=(5, 5))  
sns.heatmap(cm\_dtc\_train, annot=True, linewidths=0.5, linecolor="red", fmt=".0f", ax=ax)  
plt.xlabel("predicted y values")  
plt.ylabel("real y values")  
plt.title("Test for Train Dataset:decision tree")  
plt.show()

**Output:**

"C:\Python 310\python.exe" "E:/ML Practicals/ml\_pr3/main.py"

shape of dataset (400, 9)

score: 0.925

confusion matrix for test data:decision tree

[[65 3]

[ 3 9]]

precision\_score: 0.75

recall\_score: 0.75

f1\_score: 0.75

confusion matrix for test data:decision tree

[[215 0]

[ 0 105]]



