Q7.KNN

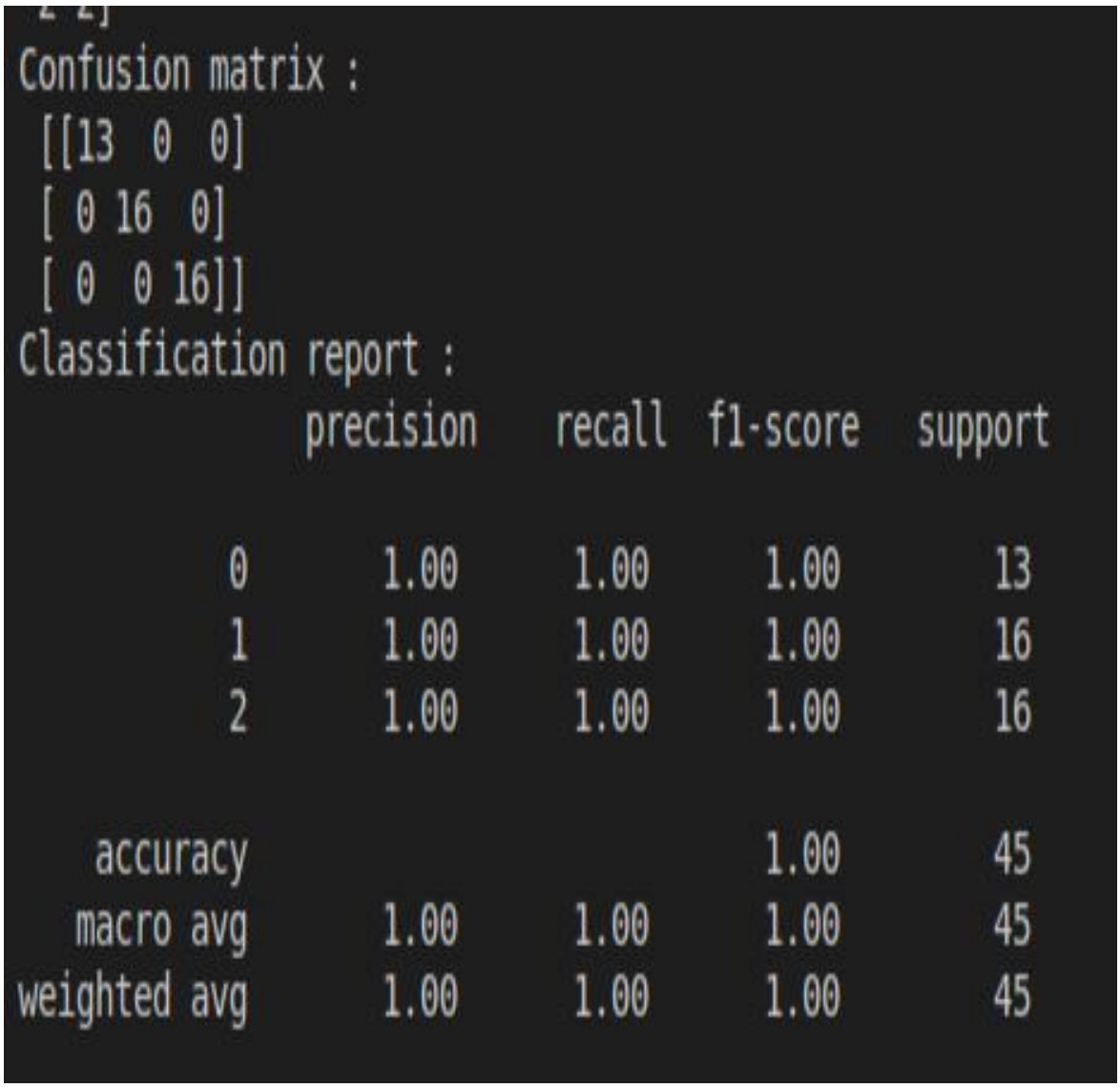
Write a program to implement K-Nearest neighbour algorithm to classify IRIS dataset. Printboth correct and wrong predictions using python ML libraries .classes can be used for this program

CODE:

```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report,
confusion_matrix
from sklearn import datasets

iris = datasets.load_iris()
iris_data = iris.data
iris_labels = iris.target

print(iris_data)
print(iris_labels)

x_train, x_test, y_train, y_test = train_test_split(iris_data,
iris_labels, test_size=0.30)

clsify = KNeighborsClassifier(n_neighbors=5)
clsify.fit(x_train, y_train)

y_pred = clsify.predict(x_test)

print('Confusion matrix : \n', confusion_matrix(y_test, y_pred))
```

```
print('Classification report : \n', classification_report(y_test, y_pred))
```

OUTPUT:



```
 2 2]
Confusion matrix :
 [[13  0  0]
 [ 0 16  0]
 [ 0  0 16]]
Classification report :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00        16
           2       1.00      1.00      1.00        16

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45
```

Q8.K Means

Apply Em algorithm to cluster a set of data stored in a .csv file. Use the same data for clustering using the K-means algorithms. Compare the results of these two algorithms and comment on the quality of clustering. you can add python ML library classes/API in the program.

CODE:

```
from sklearn.cluster import KMeans
from sklearn import preprocessing
from sklearn.mixture import GaussianMixture
from sklearn.datasets import load_iris
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

datasets = load_iris()
x = pd.DataFrame(datasets.data, columns=['Sepal_length',
'Sepal_width', 'Petal_length', 'Petal_width'])
y = pd.DataFrame(datasets.target, columns=['Targets'])
colourmap = np.array(['red', 'green', 'blue'])

plt.figure(figsize=(14, 7))

plt.subplot(1, 3, 1)
plt.scatter(x.Petal_length, x.Petal_width, c=colourmap[y.Targets],
s=40)
plt.title('Real')
```

```
kmeans = KMeans(n_clusters=3)
kmeans.fit(x)
predy = np.choose(kmeans.labels_, [0, 1, 2]).astype(np.int64)
plt.subplot(1, 3, 2)
plt.scatter(x.Petal_length, x.Petal_width, c=colourmap[predy],
s=40)
plt.title('KMeans')

scaler = preprocessing.StandardScaler()
scaler.fit(x)
xsa = scaler.transform(x)
xs = pd.DataFrame(xsa, columns=x.columns)

gmm = GaussianMixture(n_components=3)
gmm.fit(xs)
y_cluster_gmm = gmm.predict(xs)
plt.subplot(1, 3, 3)
plt.scatter(x.Petal_length, x.Petal_width,
c=colourmap[y_cluster_gmm], s=40)
plt.title('GMM Classification')

plt.tight_layout()
plt.show()
```

OUTPUT: