GAUTHAMI.U.G. 4MT17CS041

Date 18/10/20

Expt. No. 07

7. Write a program to implement k-nearest neighbour algorithm to classify the iris data. Print both correct and wrong predictions. Java/Python ML library classes can be used for this problem.

```
from sklearn.datasets import load-iris
from sklearn.neighbors import kneighbors Classifier
from sklearn.metrices import classification_report
import numpy as np
from sklearn.model_selection import train-test-split
from Sklearn.metrices import confusion_matrix
from sklearn.metrices import accuracy-score
 iris-dataset = load iris()

print("\n IRIS FEATURES\TARGET NAMES:\n" iris-dataset,
                          target_names)
for i in range(len(iris-dataset.target_names):
    print("\n [{0}]: [{1}]".format[i, iris-dataset.
                          target_names[i]))
#print("\n IRIS DATA:\n", iris-dataset["data"])

X_train, x-test, y-train, y-test = train_test_split
      (iris_dataset["data"], iris.dataset["target"], random state=0]

Classifier = kNeighbors Classifier(n_neighbors = 8, p=3,
                  metric = "euclidean")
classifier.fit(x-train, y-train)
 y-pred = classifier.predict(x-test)
```

Teacher's Signature :

```
cm = confusion_matrix (y_test, y-pred)
print (" Confusion matrix is as follows \n", cm)
print ("Accuracy matrices")
print("Classification_report (y-test, y_pred))
print("Correct prediction", accuracy-score(y-test, y-pred))
print ("wrong prediction", (1 - accuracy_score(y-test, y-pred)))
```

output :-

```
IRIS FEATURES\ TARGET NAMES:

['Setosa' 'versicolor' 'virginica']

[0]: [Setosa]
[1]: [versicolor]
[2]: [virginica]

KNeighborsClassifier(algorithm='auto', leaf_size=30,
            metric='euclidean' metric_params=None,
            n_jobs= None, n_neighbors=8, p=3
            weights= 'uniform')

Confusion matrix is as follows

   [[13  0  0]
    [0  15  1]
    [0  0  9]]

Accuracy metrics

            prediction    recall   f1_score   support
0           1.00          1.00     1.00       13
1           1.00          0.94     0.97       6
2           0.90          1.00     0.95       9
```

```
                                           0.97        38
accuracy
Macro avg           0.97        0.98       0.97        38
Weighted avg        0.98        0.97       0.97        38


correct prediction : 0.9736842105263158
wrong prediction : 0.02631578947368418
```