PAGE NO : 18
DATE : 18/12/20
EXP.NO. : 6

6 Write a program to implement K-nearest neighbor algorithm to classify the iris dataset Print both correct & wrong predictions. Java / Python ML library classes can be used for this problem.

```
from sklearn.datasets import load_iris
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import train_test_split
from sklearn.metrics 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]))
```

ARUN'S

PAGE NO : 19
DATE :
EXP.NO. : 6

```
# 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 = KNeighborsClassifier (n_neighbors=8,
        p=3, metric = "euclidean")
classifier.fit (X_train, y_train)
y_pred = classifier.predict (X_test)
cm = confusion_matrix (y_test, y_pred)
print ("Confusion matrix is as follows \n", cm)
print ("Accuracy metrics")
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)))
```

ARUN'S

output :

IRIS FEATURES \ TARGET NAMES :

['setosa' 'versicolor' 'virginica']

[0] : [setosa]

[1] : [versicolor]

[2] : [virginica]

KNeighborsClassifier(algorithm = 'auto', leaf_siz
metric = 'euclidian metric - params = Nor
n_jobs = None, n_neighbours = 8,
p = 3, weights = 'uniform')

Confusion matrix is as follows

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

Accuracy metrics

	precision	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

accuracy

Macro avg	0.97	0.98	0.97	38
weighted avg	0.98	0.97	0.97	38

Correct prediction : 0.973684210526315

wrong prediction : 0.026315789473684