

* write a program to implement k-nearest neighbour algorithm to classify the iris dataset. Print both correct and wrong predictions. Java/Python in-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 confusion_matrix
from sklearn.metrics import accuracy_score

iris_dataset = load_iris()

print("\n IRIS FEATURES/TARGET NAMES : \n", iris_dataset, target_name)
for i in range(len(iris_dataset.target_names)):
    print("\n [{}]: [{}]" . format(i, iris_dataset.target_names[i]))

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)

```

Teacher's Signature _____

```
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))
```

Teacher's Signature _____

output

IRIS FEATURE/TARGET NAMES

['Setosa', 'Versicolor', 'Virginica']

[0]: ['Setosa']

[1]: ['Versicolor']

[2]: ['Virginica']

K Neighbors Classifier (algorithm = "auto", leafsize = 30,
metric = 'euclidean', metric_params = None,
n_jobs = None, n_neighbors = 8, p = 3,
weights = 'uniform')

Confusion matrix is as follow

[[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

accuracy

0.97

38

Macro avg

0.97

0.98

0.97

38

weighted avg

0.98

0.97

0.97

38

Correct predictions : 0.973684210523158

wrong predictions : 0.02631578947368418