Experiment 2
31/7/2025

IMPLEMENT A CLASSIFIER ON OPEN SOURCE DATASET

Aim:
To implement a KNN classifier on the iris dataset

Objective:
- To understand the working of the KNN classification algorithm.
- To apply the KNN classifier on the standard iris dataset for predicting species classes.
- To evaluate the accuracy and performance of the classifier.
- To interpret the results and learn about the influence of the value 'K' on model performance.

Pseudocode:
- Import the necessary libraries (numpy, Pandas, sklearn).
- Load the iris-dataset from scikit-learn.
- Preprocess the dataset, normalize it using StandardScalar.
- Split the dataset into training and testing sets using train-test_split.
- Instantiate the KNN classifier with a chosen value of K.
- Fit the classifier on the training data.
- Predict the target values for test set.
- evaluate the classifier using metrics such as accuracy score.

Observation:

Accuracy : 1.0

Confusion Matrix :

```
[[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
```

Classification Report -

	Precision	recall	f1-score	support
0	1.00	1.00	1.00	10
1	1.00	1.00	1.00	9
2	1.00	1.00	1.00	11

Result: Successfully implemented KNN classifier on iris-dataset.

[3]:
```
!pip install scikit-learn
import pandas as pd
import numpy as np
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

```
Defaulting to user installation because normal site-packages is not writeable
Collecting scikit-learn
  Downloading scikit_learn-1.7.1-cp310-cp310-manylinux2014_x86_64.manylinux_2_17_x86_64.whl.metadata (11 kB)
Requirement already satisfied: numpy>=1.22.0 in /home/jupyter-ra2311047010041/.local/lib/python3.10/site-pac
learn) (2.2.3)
Collecting scipy>=1.8.0 (from scikit-learn)
  Downloading scipy-1.15.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (61 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.0/62.0 kB 114.1 kB/s eta 0:00:00 0:00:01
Collecting joblib>=1.2.0 (from scikit-learn)
  Downloading joblib-1.5.1-py3-none-any.whl.metadata (5.6 kB)
Collecting threadpoolctl>=3.1.0 (from scikit-learn)
  Downloading threadpoolctl-3.6.0-py3-none-any.whl.metadata (13 kB)
Downloading scikit_learn-1.7.1-cp310-cp310-manylinux2014_x86_64.manylinux_2_17_x86_64.whl (9.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 9.7/9.7 MB 200.3 kB/s eta 0:00:0000:0100:02
Downloading joblib-1.5.1-py3-none-any.whl (307 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 307.7/307.7 kB 330.0 kB/s eta 0:00:00a 0:00:01
Downloading scipy-1.15.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (37.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 37.7/37.7 MB 205.5 kB/s eta 0:00:0000:0100:05
Downloading threadpoolctl-3.6.0-py3-none-any.whl (18 kB)
[notice] To update, run: pip install --upgrade pip
```

4]:
```
iris = load_iris()
X = iris.data
y = iris.target
```

1]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

2]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

3]:
```
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
```

3]:
```
▾ KNeighborsClassifier
▸ Parameters
```

4]:
```
y_pred = knn.predict(X_test)
```

5]:
```
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 1.0
```

```
▸ Parameters
```

1]:
```
y_pred = knn.predict(X_test)
```

5]:
```
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 1.0
```

```
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 1.0

Confusion Matrix:
 [[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```