Training samples: 120, Testing samples: 30

Evaluating model performance
Accuracy: 1.0
Precision: 1.0
Recall: 1.0
F1 score: 1.0

Experiment - 2.

7-8-25 Implement a classifier using open-source dataset

AIM: To implement a supervised machine learning using an open source dataset

PSEUDO CODE:

* Importing libraries like pandas, scikit-learn, dataset metrics, K-Neighbour classifier
* Load the dataset
  Use dataset.load_iris() (iris dataset)
* Prepare the data
  Assign features to x target to y
* split into training and testing set
  use train_test_split(x, y, test_size=0.3, random_state=42)
* Instantiate the KNN classifier:
  knn = KNeighbours Classifier (n_neighbours = 3)
* Train the model
* Make the predictions
  y_pred = knn.predict (x_test)
* Evaluate the classifier:
  Calculate accuracy: metrics.accuracy_score (y_test, y_pred)

OBSERVATION:

The KNN classifier is trained on the iris dataset and tested with unseen data

Output is displayed

Lowering k can make the model more sensitive to noise, while larger k can smoother decision boundaries

RESULT:

KNN classifier was successfully implemented and tested using an open-source dataset.

File Edit View Run Kernel Tabs Settings Help

Filter files by name

/

Name	Last Modified
archive.zip	14 days ago
exam.py	4 months ago
iris.ipynb	14 days ago
irisdata.i...	6 days ago
new.ipynb	14 days ago
nnml.py	3 months ago
osexam.c	3 months ago
tictac.ipy...	4 months ago
Untitled.i...	14 days ago
untitled.txt	14 days ago
Untitled1...	14 days ago
untitled1....	6 days ago
Untitled2...	6 days ago
untitled2....	next year
week1.py	6 days ago
week2.py	6 days ago
week3.py	next year

week3.py | week1.py | week2.py | jupyter-ra2311047010013@c

```python
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

data = load_iris()
X, y = data.data, data.target
print(f"Dataset loaded with {X.shape[0]} samples and {X.shape[1]} features.\n")

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
print(f"Training samples: {X_train.shape[0]}, Testing samples: {X_test.shape[0]}\n")

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
print("Feature scaling completed.\n")

knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
print("Training completed.\n")

y_pred = knn.predict(X_test)
print("Predictions done.\n")

print("Evaluating model performance:")
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Precision:", precision_score(y_test, y_pred, average='macro'))
```