D. Khaja Nawaz

RA2311047010037

Deep Learning Techniques (LAB)

Date	Title	Sign
24/07/2025	Exploring the deep learning platforms	
31/07/25	Implement a classifier using open-source dataset	1, 7 Marks
31/07/25	Study of the classifiers with respect to statistical parameters	

LAB-2

31/07/25

Implement a classifier Using open-source dataset

*AIM :-

To implement a machine learning classifier using the k-Nearest algorithm on the iris dataset to classify iris flowers in their respective species based on petal and sepal features

* Objective :-

→ load and explore the Iris dataset
→ preprocess the dataset using normalization and train_test
→ TO apply the KNN algorithm for classification
→ TO evaluate the model's performance using accuracy, confusion matrix, and classification report.
→

* pseudocode :-

Start
1. Import required libraries (pandas, sklearn, matplotlib)
2. load the Iris dataset
3. split the dataset into training and testing sets
4. NORmalize the features using standardscaler
5. Initialize the KNN classifier
6. Train the model using training data
7. make predictions on test data
8. Evaluate the model using :
   → Accuracy score
   → classification report
   → confusion matrix
9. plot the confusion matrix using seaborn

End

Code

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.neighbours import kNeighorsclassifier
from sklearn.preprocessing import standardscaler
from sklearn. import accuracy_score, classification_report

# load the iris dataset
iris = load_iris()
X = iris.data
y = iris.target

# split the dataset

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                   test_size=0.3,
                                   random_state=0)

# Normalize the features using standard sclaer
scaler = standardscaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test

# create and train the KNN model

model = kNeighborclassifier(n_neighbors=3)
model.fit(X_train, y_train)

# predict test data
y_pred = model.predict(X_test)

# Accuracy
accuracy = accuracy_score(y_test, y_pred)

print("\nclassification report:")
print(a classification_report(y_test, y_pred, target_names
                              =iris.target_
                              names))
```

Output

Accuracy = 0.977

classification report

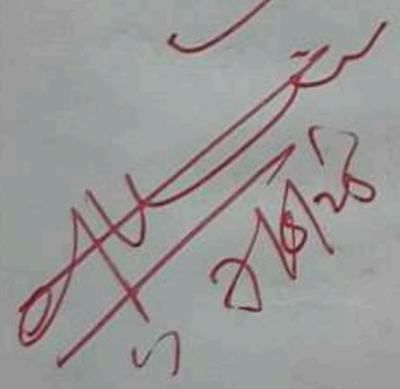
	precesion	recall	f1-score	support
setosa	1.00	1.00	1.00	16
versicolor	1.00	0.94	0.97	18
virginica	0.92	1.00	0.96	11
accuracy				45
Macro avg	0.97	0.98	0.98	45
weighted avg	0.98	0.98	0.98	45

```
plt.figure(figsize=(8,4))
  color = ['red', 'green', 'blue']
for i, label in enumerate(np.unique(y_test)):
                plt.scatter(X_test[y_test==label][:,0],
                            X_test[y_test==label][:,1],
        color=colors[i], label=target_names[label],
                                edgecolor='black')

plt.xlabel('petal length')
plt.ylabel('petal width')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

* Result :-

The K-Nearest Neighbors (KNN) classifier was successfully implemented on the Iris dataset using only two features (petal length and petal width)



Not Secure http://10.1.38.19/user/ra2311047010037/lab/tree/DLT/Lab2.ipynb

File Edit View Run Kernel Tabs Settings Help

Launcher | Lab2.ipynb

Code | Notebook | Python 3 (ipykernel)

Filter files by name

/ DLT /

Name	Last Modified
Lab2.ipynb	next year
Lab3.ipynb	next year

[1]:
```
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 classification_report, accuracy_score
```

[2]:
```
iris = load_iris()
X = iris.data
y = iris.target
```

[11]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```

[12]:
```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

[13]:
```
knn = KNeighborsClassifier(n_neighbors=3)
```

[14]:
```
knn.fit(X_train_scaled, y_train)
```

[14]: KNeighborsClassifier

▸ Parameters

Simple 0 3 Python 3 (ipykernel) | Idle Mem: 381.28 MB Mode: Command Ln 1, Col 1 Lab2.ipynb 1

File Edit View Run Kernel Tabs Settings Help

Filter files by name

/ DLT /

Name	Last Modified
Lab2.ipynb	next year
Lab3.ipynb	next year

Launcher | Lab2.ipynb

Code | Notebook | Python 3 (ipykerne

[14]:
```
knn.fit(X_train_scaled, y_train)
```

[14]: KNeighborsClassifier

▼ Parameters

n_neighbors	3
weights	'uniform'
algorithm	'auto'
leaf_size	30
p	2
metric	'minkowski'
metric_params	None
n_jobs	None

[15]:
```
y_pred = knn.predict(X_test_scaled)
```

[16]:
```
print(f"Accuracy: {accuracy_score(y_test, y_pred):.2f}\n")
print("Classification Report:")
print(classification_report(y_test, y_pred, target_names=iris.target_names))
```

File Edit View Run Kernel Tabs Settings Help

Launcher | Lab2.ipynb

Code | Notebook | Python 3 (ipyker

Filter files by name

/ DLT /

Name	Last Modified
Lab2.ipynb	next year
Lab3.ipynb	next year

metric_params	None
n_jobs	None

[15]:
```
y_pred = knn.predict(X_test_scaled)
```

[16]:
```
print(f"Accuracy: {accuracy_score(y_test, y_pred):.2f}\n")
print("Classification Report:")
print(classification_report(y_test, y_pred, target_names=iris.target_names))
```

```
Accuracy: 0.98

Classification Report:
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        16
  versicolor       1.00      0.94      0.97        18
   virginica       0.92      1.00      0.96        11

    accuracy                           0.98        45
   macro avg       0.97      0.98      0.98        45
weighted avg       0.98      0.98      0.98        45
```

[ ]: