

31-7-25

LAB-2

Deep Learning Classifier using Open Source Dataset

Aim: To implement logistic regression on the Iris dataset for multi-class classification.

Objective:

- Load and Split the Iris dataset.
- Train a logistic regression model.
- Evaluate its performance using Accuracy.

Pseudo Code:

1. Load Iris dataset.
2. Split into training and testing sets.
3. Initialize Logistic Regression Model.
4. Train model on training data.
5. Predict on test data.
6. Calculate and display accuracy.
7. Print Actual vs Predicted labels (from 5)

Observation:

→ The logistic regression model was trained on 80% of the Iris dataset and tested on the remaining 20%.

→ The Model Achieved high accuracy, indicating good performance on multi-class classification.

→ The predicted species labels closely matched

the actual labels for most test samples.

→ Logistic Regression worked well on Test Dataset due to its simplicity and linearly separable classes.

→ Minor misclassifications if any, could be due to overlap between class features.

→ Prescaling man. iters ensured proper convergence of the model.

Result:

Model Accuracy = ~ 1.00

Successfully Implemented a classifier using Appropriate Dataset.

~~1/1/2025~~

← → ↻

Not secure 10.138.19/user/ra2311047010017/lab/workspaces/auto-x/tree/DLT/classifier(opensourcedataset).ipynb

🔍 ☆

Verify it's you

File Edit View Run Kernel Tabs Settings Help

+

📁

📄

🔄

Filter files by name 🔍

/ DLT /

Name	Last Modified
classifier(o...	6 days ago
lab3.ipynb	next year

Launcher classifier(opensourcedataset)X +

📄 + ✂ 📄 📄 ▶ ⏮ ⏪ ⏩ ⏭ Code ▾

Notebook Python 3 (ipykernel)

[4]:

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

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

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = LogisticRegression(max_iter=200, random_state=42)
model.fit(X_train, y_train) # Teach the model

y_pred = model.predict(X_test) #Making predictions on the test data

accuracy = accuracy_score(y_test, y_pred) # Evaluating the accuracy

print(f"Model Accuracy: {accuracy:.2f}")

print("\nFirst 5 Actual vs. Predicted labels:")
for i in range(5):
    print(f"Actual: {iris.target_names[y_test[i]]}, Predicted: {iris.target_names[y_pred[i]]}")

Model Accuracy: 1.00

First 5 Actual vs. Predicted labels:
Actual: versicolor, Predicted: versicolor
Actual: setosa, Predicted: setosa
Actual: virginica, Predicted: virginica
Actual: versicolor, Predicted: versicolor
Actual: versicolor, Predicted: versicolor
```