# EXPT NO : 3 A python program to implement Logistic Model DATE: 4.10.24

**AIM:**

To write a python program to implement a Logistic Model.

**PROCEDURE:**

Implementing Logistic method using the iris dataset involve the following steps:

# Step 1: Import Necessary Libraries

First, import the libraries that are essential for data manipulation, visualization, and model building.

# Step 1: Import Necessary Libraries import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report

# Step 2: Load the Iris Dataset

The iris dataset can be loaded.

# Step 2: Load the Dataset

# For this example, we'll use a built-in dataset from sklearn. You can replace it with your dataset.

from sklearn.datasets import load\_iris

# Load the iris dataset data = load\_iris()

X = data.data

y = (data.target == 0).astype(int) # For binary classification (classifying Iris-setosa)

# Step 3: Data Preprocessing

Ensure the data is clean and ready for modeling. Since the Iris dataset is clean, minimal preprocessing is needed.

# Step 3: Prepare the Data

# Split the dataset into training and testing sets

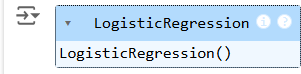
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Step 4 : Train a Model

**# Step 4: Create and Train the Model model = LogisticRegression()**

**model.fit(X\_train, y\_train)**

**OUTPUT :**



# Step 5 : Make Predictions

Use the model to make predictions based on the independent variable.

# Step 5: Make Predictions

y\_pred = model.predict(X\_test)

# Step 6 : Evaluate the Model

Evaluate the model performance.

# Step 6: Evaluate the Model

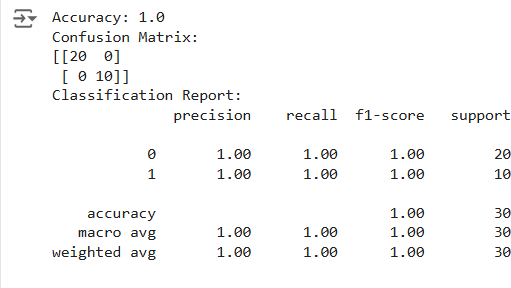
accuracy = accuracy\_score(y\_test, y\_pred) conf\_matrix = confusion\_matrix(y\_test, y\_pred) class\_report = classification\_report(y\_test, y\_pred) # Print evaluation metrics

print(f"Accuracy: {accuracy}") print("Confusion Matrix:")

print(conf\_matrix)

print("Classification Report:") print(class\_report)

# OUTPUT :



**Step 7 :Visualize the Results**

Plot the original data points and the fitted regression line.

# Step 7: Visualize Results (Optional)

x\_values = np.linspace(-10, 10, 100)

sigmoid\_values = 1 / (1 + np.exp(-x\_values))

# Plot the sigmoid function plt.figure(figsize=(10, 5))

plt.plot(x\_values, sigmoid\_values, label='Sigmoid Function', color='blue') plt.title('Sigmoid Function')

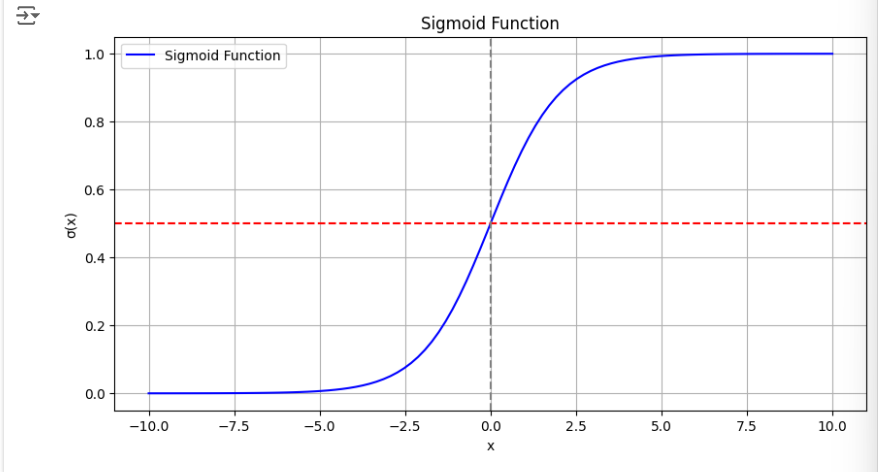
plt.xlabel('x')

plt.ylabel('σ(x)') plt.grid()

plt.axhline(0.5, color='red', linestyle='--') # Line at y=0.5 plt.axvline(0, color='gray', linestyle='--') # Line at x=0 plt.legend()

plt.show()

# OUTPUT :



**RESULT:**

This step-by-step process will help us to implement Logistic models using the Iris dataset and analyze their performance.