

EXPT NO : 3

A python program to implement Logistic Model

DATE:

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)
```

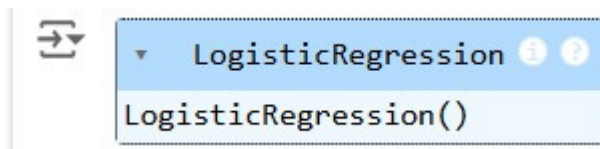
Step 4 : Train a Model

```
# 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 :

```
➡ Accuracy: 1.0
Confusion Matrix:
[[20  0]
 [ 0 10]]
Classification Report:
              precision    recall  f1-score   support

     0       1.00      1.00      1.00        20
     1       1.00      1.00      1.00        10

   accuracy          1.00
  macro avg          1.00
 weighted avg          1.00
```

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()

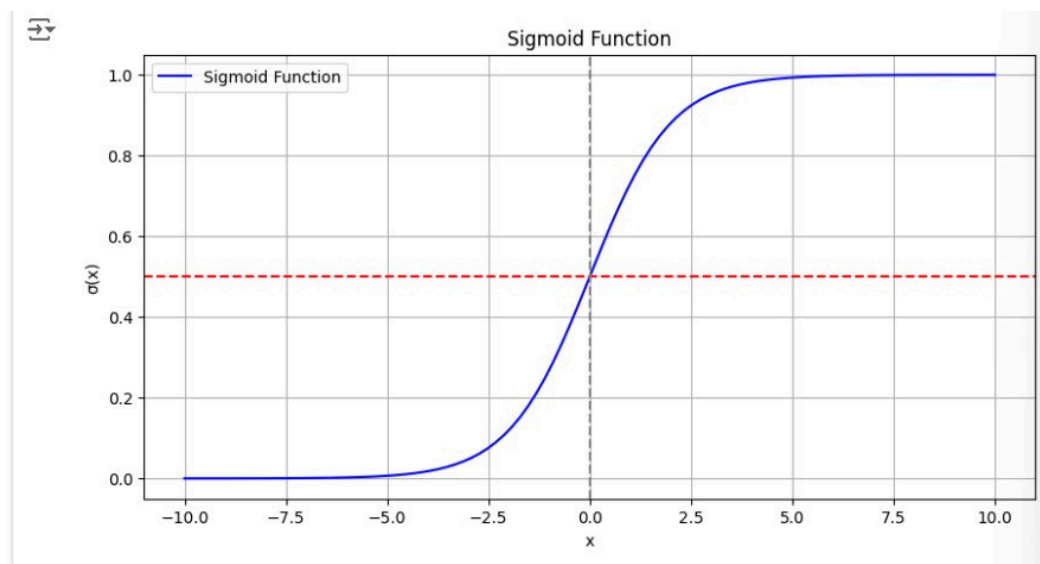
plt.axhline(0.5, color='red', linestyle='--')           # Line at y=0.5

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.