# EXPERIMENT 9

**Student Name: Samuel** **UID: 24MAI10018**

**Branch: CSE-AIML** **Section/Group: 24MAI-1**

**Semester: 2** **Date of Performance:**

**Subject Name: Machine Learning Lab** **Subject Code: 24CSH–651**

## AIM:

Implement Logistic Regression using Python.

## SOFTWARE REQUIREMENTS:

- Python IDE (e.g., Jupyter Notebook, PyCharm, etc.)
- NumPy Library.
- Pandas Library.
- Scikit-Learn Library.
- Matplotlib & Seaborn Libraries.

## THEORY:

**Logistic Regression Algorithm:** Logistic Regression is a supervised learning algorithm used for binary and multi-class classification tasks. It models the probability of a data point belonging to a particular class using the sigmoid function.

**Key Features of Logistic Regression:**

- **Probability-Based Classification**: Predicts the likelihood of a class using probabilities.
- **Logit (Sigmoid) Function**: Maps any real-valued number into a range between 0 and 1.
- **Decision Boundary**: Uses a threshold (e.g., 0.5) to classify data points.

**Mathematical Formulation:** The logistic function (sigmoid function) is defined as:

$$() = \frac{1}{1 + -}$$

where,

- = 0+11+22+⋯+
  - $w_i$ are the regression coefficients (weights).
  - $x_i$ are the input features.

The cost function used for optimization is the log-loss function:

( ) = − 2 1 [ log( ) + (1 − ) log(1 − ^)]
=1

where ^ is the predicted probability of class 1.

**Advantages of Logistic Regression:**

- Easy to implement and interpret.
- Works well when the relationship between features and target is linear.
- Computationally efficient for large datasets.

**Disadvantages:**

- Struggles with non-linear relationships unless feature transformation is applied.
- Sensitive to outliers.

**ALGORITHM:**

1. Load and preprocess the dataset.
2. Convert the target variable into binary/multi-class format if necessary.
3. Apply feature scaling (optional for better performance).
4. Train the Logistic Regression Model.
5. Predict the class labels for the test set.
6. Evaluate model performance using accuracy, precision, and recall.
7. Visualize the decision boundary (for 2D datasets).

## SOURCE CODE:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler from
sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score,
      classification_report, confusion_matrix

# Load dataset (Iris dataset)
iris = datasets.load_iris()
X = iris.data[:, :2] # Taking first two features for visualization
y = (iris.target != 0) * 1 # Converting to a binary classification
      problem (Setosa vs. Others)

# Split data into training and testing sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
      random_state=42)

# Standardizing the features (important for Logistic Regression)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Initialize Logistic Regression
Model log_reg = LogisticRegression()

# Train the model
log_reg.fit(X_train, y_train)

# Make predictions
y_pred = log_reg.predict(X_test)

# Compute accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"Model Accuracy: {accuracy * 100:.2f}%")

# Display Classification Report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)

# Visualizing Confusion Matrix
plt.figure(figsize=(6, 5))
sns.heatmap(conf_matrix, annot=True, cmap="Blues", fmt="d",
xticklabels=["Setosa", "Others"], yticklabels=["Setosa",
"Others"]) plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix - Logistic
Regression") plt.show()
```

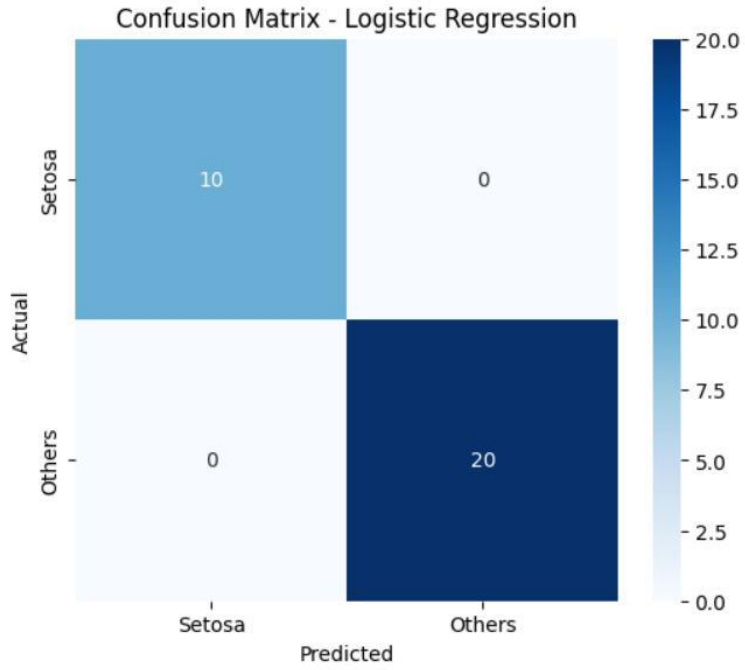
## OUTPUT:

```
Model Accuracy: 100.00%

 Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00        20

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```




## LEARNING OUTCOMES:

1. Understood the Logistic Regression Algorithm and its mathematical formulation.
2. Learned how sigmoid activation helps in classification.
3. Implemented Logistic Regression using Scikit-Learn.
4. Explored the importance of feature scaling for optimization.
5. Evaluated model performance using accuracy, classification report, and confusion matrix.
6. Understood the difference between Logistic Regression and Linear Regression.