Experiment 2

Title: To implement Logistic Regression

Lab Objective: To acquire fundamental knowledge of developing machine learning models.

Theory:

Logistic Regression is a widely used statistical and machine learning technique for classification problems. Unlike linear regression that predicts continuous outcomes, logistic regression estimates the probability that a given input belongs to a particular category, usually for binary classification such as Yes/No or 0/1. It's commonly applied in situations like predicting whether a student will pass based on study hours or whether a customer will make a purchase based on their age.

Types of Logistic Regression:

4. **Binomial Logistic Regression: Used for binary outcomes (e.g., Yes/No, Pass/Fail).**
5. **Multinomial Logistic Regression:** Applied when there are multiple categories without any order (e.g., "bike," "car," "truck")
6. **Ordinal Logistic Regression:** For ordered categories (e.g., "low," "medium," "high").

Logistic regression is extensively used in classification tasks across many fields.

Sigmoid (Logistic) Function:
The sigmoid function underpins logistic regression by converting any real number into a value between 0 and 1, which makes it suitable for modeling probabilities.

Mathematical Definition: $$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Where:

- $Z = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_n X_n$
- θ_i are model parameters (weights)
- X_i are the input features

Shape of the Sigmoid Curve

- The function has an S-shaped curve (hence the name "sigmoid").
- As $z \to +\infty$, $\sigma(z) \to 1$
- As $z \to -\infty$, $\sigma(z) \to 0$
- At $z=0$, $\sigma(0)=0.5$

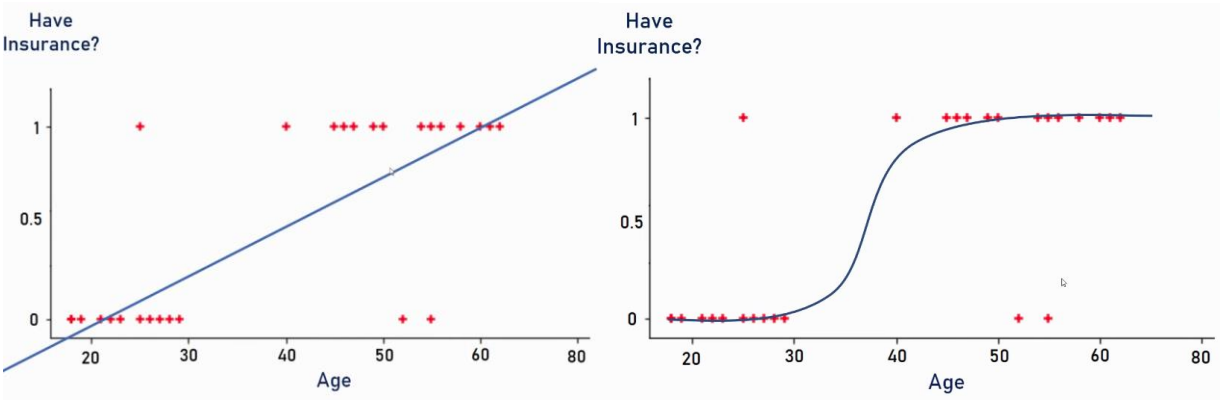
This makes it symmetric and smooth, which helps in optimization.

Usage of Sigmoid Function in Logistic Regression

3. Probability Estimation:
   Logistic regression uses the sigmoid function to map the linear combination of inputs into a probability.
   - Output close to 1 → Strong probability of belonging to class 1
   - Output close to 0 → Strong probability of belonging to class 0

4. Decision Boundary:
   Logistic regression makes predictions based on a threshold, typically 0.5.
   If $\sigma(z) \geq 0.5$, predict class = 1 else predict class = 0
   This threshold can be adjusted depending on the problem (e.g., medical diagnosis may use 0.3 to avoid missing positive cases).

Why not Linear Regression for Classification?



Linear regression can produce predictions outside the valid probability range [0,1] and struggles with sharp class boundaries, outliers, and imbalanced data. Logistic regression overcomes this by using the sigmoid function to map outputs between 0 and 1, providing meaningful probabilities and a clear decision boundary for classification. This allows it to model sudden changes in outcomes, like the increased likelihood of buying insurance with age.

Difference between Linear and Logistic Regression:

Aspect	**Linear Regression**	**Logistic Regression**
1. Purpose	Predicts continuous values (e.g., salary, house price).	Predicts categorical outcomes (e.g., yes/no, 0/1).
2. Output Range	`Output can be any real number` $(-\infty,+\infty)$	Output is restricted to [0,1] using sigmoid.

3. Type of Problem	Used for regression problems.	Used for classification problems.
4. Error Measurement	Uses Mean Squared Error (MSE) as cost function.	Uses Log Loss (Cross-Entropy Loss) as cost function.
5. Decision Boundary	No natural decision boundary; not ideal for classification.	Has a clear decision boundary (e.g., probability ≥ 0.5 ⇒ Class 1).
6. Interpretability	Coefficients represent the change in output per unit change in input.	Coefficients represent change in log-odds of the outcome.
7. Applications	Forecasting, trend analysis, demand prediction.	Spam detection, medical diagnosis, customer churn prediction.

Problem:

To implement a Logistic Regression model that predicts whether a person will buy insurance based on their age.

Steps:

Step 1: Import Required Libraries and Load Dataset
We first import the necessary Python libraries for handling data, visualization, and machine learning. Then, we load the dataset which contains the feature Age and the target variable HaveInsurance.

Step 2: Visualize Data using Scatter Plot
We plot a scatter plot between Age and HaveInsurance to observe the data distribution. This helps in understanding how the independent variable (Age) is related to the dependent variable (Insurance status).

Step 3: Define Feature and Target Variable
The independent variable (X) is taken as Age, and the dependent variable (y) is taken as HaveInsurance. This separation is important before training the model.

Step 4: Train-Test Split
The dataset is split into training and testing sets. The training set is used to fit the model, while the testing set is used to evaluate its performance.

Step 5: Create and Train the Model
We create a Logistic Regression model object and train (fit) it on the training data so that it learns the relationship between Age and Insurance status.

Step 6: Make Predictions
Using the trained model, we predict the insurance status for the test data. We also calculate the probabilities for each class (0 or 1).

Step 7: Evaluate Model Performance
We measure the performance of the model using accuracy. This tells us how many predictions were correct compared to the actual results in the test data.

Step 8: Visualize Logistic Curve
Finally, we plot the Logistic Regression curve (sigmoid function) along with the original data points. The sigmoid curve shows the probability of a person having insurance at different ages, and a decision threshold (0.5) is drawn to separate the two classes.

Program:

```
# Step 1: Import required libraries and load insurance dataset

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
# Load dataset
data = pd.read_csv("insurance_data.csv")
print("Dataset Head:")
print(data.head())

# Step 2: Visualize Data using scatter plot.

plt.scatter(data['age'], data['bought_insurance'], color='red')
plt.xlabel("Age")
plt.ylabel("Have Insurance?")
plt.title("Scatter Plot of Age vs Insurance By Rishikesh Singh")
plt.show()

# Step 3: Define Feature and Target variable.

X = data[['age']]
y = data['bought_insurance']

# Step 4: Train-Test Split.

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

 # Step 5: Create and Train the Model.

```
model = LogisticRegression()
model.fit(X_train, y_train)
```

 # Step 6: Make predictions.

```
y_pred = model.predict(X_test)
y_prob = model.predict_proba(X_test)
print("Probabilities :", np.array(y_prob))
print("Predicted :", y_pred)
print("Actual :", np.array(y_test))
```

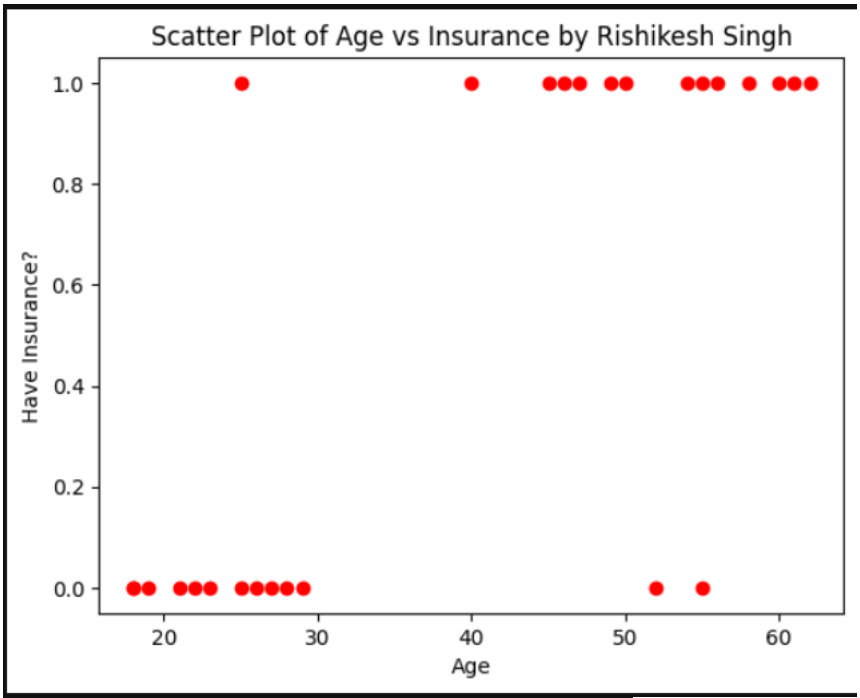
 # Step 7: Evaluate the model performance using accuracy metric.

```
accuracy = model.score(X_test, y_test)
print("Accuracy:", accuracy)

# Step 8: Visualize Logistic Curve. Plot the fitted sigmoid curve
along with the actual data points.

# Create age range for plotting the curve
age_range = np.linspace(10,90, 100).reshape(-1, 1)

# Get probabilities from your trained model
probabilities = model.predict_proba(age_range)[:, 1]

# Threshold
threshold = 0.5

# Plot
plt.figure(figsize=(10, 6))
plt.scatter(X, y, alpha=0.6, label='Data points')
plt.plot(age_range, probabilities, 'r-', linewidth=2,
label='Logistic Regression')
plt.axhline(y=threshold, color='green', linestyle='dotted',
linewidth=2, label=f'Decision Threshold ({threshold})')
plt.xlabel('Age')
plt.ylabel('Probability of Having Insurance')
plt.title('Logistic Regression: Age vs Insurance Status By
Rishikesh Singh')
plt.legend()
plt.grid(True, alpha=0.3)
plt.show()
```

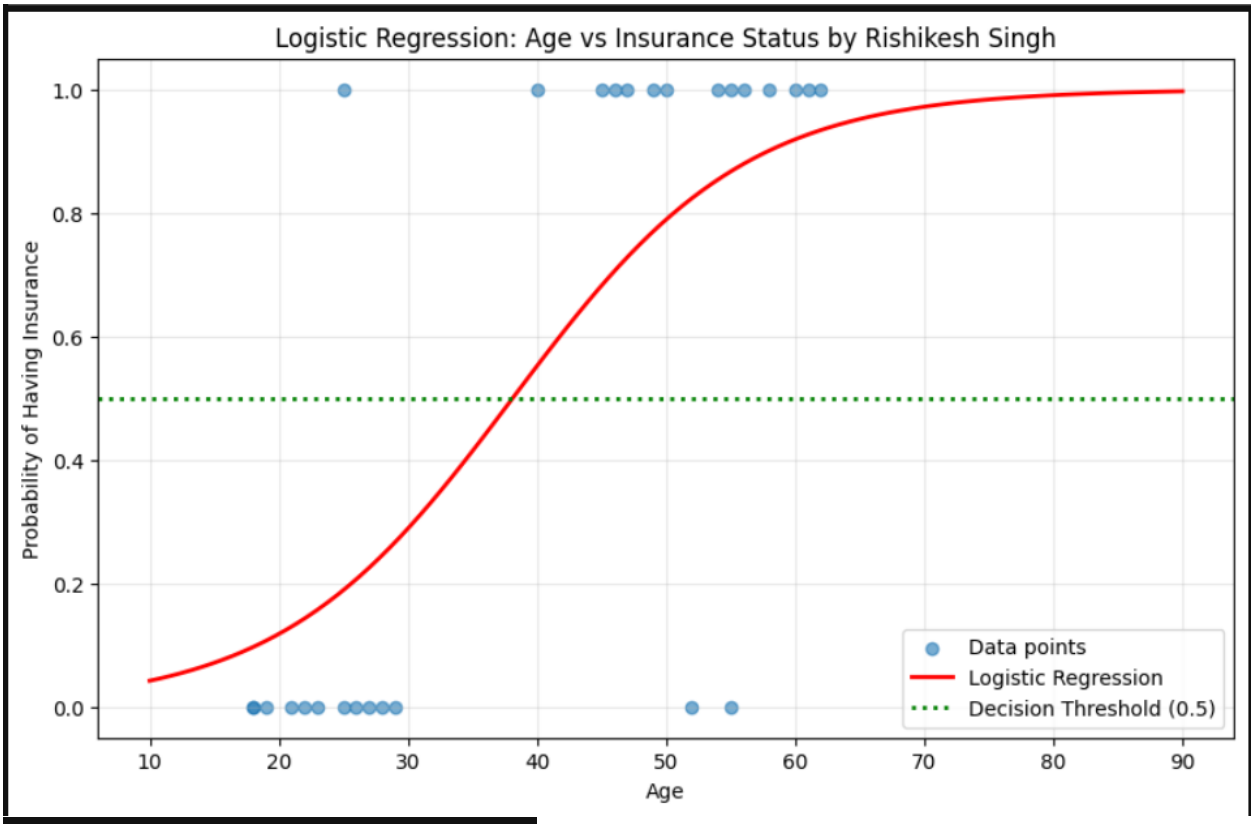
 # Step 10: Make predictions for new data. Example: Predict if a person aged 64 years will buy insurance.

```
print("Prediction for age 64:", model.predict([[56]]))
print("Probability for age 64:", model.predict_proba([[56]]))
```

Program Output:




```
Probabilities : [[0.87005015 0.12994985]
 [0.85697094 0.14302906]
 [0.90330864 0.09669136]
 [0.08095865 0.91904135]
 [0.75474474 0.24525526]
 [0.21099596 0.78900404]
 [0.07307146 0.92692854]]
Predicted : [0 0 0 1 0 1 1]
Actual : [0 0 0 1 0 1 1]
Accuracy: 1.0
```




```
Prediction for age 64: [1]
Probability for age 64: [[0.1207647 0.8792353]]
```

Conclusion: In this experiment, we implemented a Logistic Regression model to predict insurance purchase based on age. By using the sigmoid function, it maps predictions to valid probabilities and offers a clear decision boundary, making it more suitable than linear regression for classification tasks.