AIM - Implement linear and logistic regression models on real-world datasets.

THEORY-

What is Regression?

Regression is a supervised machine learning technique used to predict a dependent variable (target or outcome) based on one or more independent variables (features or predictors). It models the relationship between variables to make predictions.

Linear Regression

Linear regression is a statistical method that models the relationship between a dependent variable (y) and one or more independent variables (x) by fitting a linear equation to the observed data. It assumes a straight-line relationship between the variables and is used for predicting continuous outcomes. The goal is to find the best-fitting line that minimizes the sum of squared differences between actual and predicted values (using ordinary least squares).

Logistic Regression

Logistic regression is used when the dependent variable is binary (0 or 1, yes/no, disease/no disease).It predicts the probability that an observation belongs to a particular class using the logistic (sigmoid) function.

1. Dataset Source

The dataset used for this experiment is the **Heart Disease Dataset**, obtained from Kaggle.

Source:
https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset

2. Dataset Description

The Heart Disease Dataset is a real-world medical dataset used to analyze and predict heart disease based on clinical attributes.

It contains **1025 instances** and **14 attributes**, including features such as age, gender, chest pain type, resting blood pressure, cholesterol level, maximum heart rate, ECG results, and exercise-induced angina.

For **Linear Regression**, a **continuous variable** such as **cholesterol level (`chol`)** is used as the target.

For **Logistic Regression**, the **binary target variable (`target`)** is used to classify the presence or absence of heart disease.

3. Mathematical Formulation of the Algorithms

Linear Regression

Linear Regression

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$$

Cost function:

$$J(\beta) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Logistic Regression

Logistic regression uses the sigmoid function:

$$P(y = 1) = \frac{1}{1 + e^{-z}}$$

Where:

$$z = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$

Cost function:

$$J(\beta) = -\sum_{i=1}^{n} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

4. Algorithm Limitations

Linear Regression Limitations

- Assumes linear relationship between variables
- Sensitive to outliers

- Performs poorly with multicollinearity
- Not suitable for classification tasks

Logistic Regression Limitations

- Assumes linear decision boundary
- Not effective for complex non-linear patterns
- Sensitive to outliers
- Requires feature scaling

5. Methodology / Workflow

The experiment was carried out using the following steps:

1. Load the Heart Disease dataset
2. Perform exploratory data analysis and correlation analysis
3. Visualize feature relationships using a **heatmap**
4. Preprocess data and perform feature scaling
5. Split dataset into training and testing sets
6. Train Linear Regression model for continuous prediction
7. Train Logistic Regression model for classification
8. Evaluate models using appropriate metrics

6. Performance Analysis

- **Linear Regression** performance was evaluated using **Mean Squared Error (MSE)** to measure prediction error.
- **Logistic Regression** performance was evaluated using **Accuracy, Precision, Recall**, and **Confusion Matrix**.
- The **confusion matrix** provides a clear view of true positives, true negatives, false positives, and false negatives.
- The **heatmap** helped identify strongly correlated features influencing the target variable.

Overall, Logistic Regression showed good classification performance for heart disease prediction.

7. Hyperparameter Tuning

- Linear regression does not require significant hyperparameter tuning.
- Logistic regression performance was improved by tuning:
    - **Regularization parameter (C)**
    - **Penalty type (L1 / L2)**
- Proper tuning helped reduce overfitting and improve classification accuracy.

Load dataset

```
import pandas as pd

# Working URL (from a stable repo - identical to Kaggle heart disease UCI)
url =
"https://raw.githubusercontent.com/sharmaroshan/Heart-UCI-Dataset/master/heart.csv"

# Load the data
df = pd.read_csv(url)

# Check if it loaded (should print shape (303, 14) and first rows)
print("Data loaded successfully!")
print("Shape:", df.shape)
print("\nColumns:", df.columns.tolist())
print("\nFirst 5 rows:")
print(df.head())
```

Output:

```
Data loaded successfully!
Shape: (303, 14)

Columns: ['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'targ

First 5 rows:
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
0   63    1   3       145   233    1        0      150      0      2.3      0
1   37    1   2       130   250    0        1      187      0      3.5      0
2   41    0   1       130   204    0        0      172      0      1.4      2
3   56    1   1       120   236    0        1      178      0      0.8      2
4   57    0   0       120   354    0        1      163      1      0.6      2

   ca  thal  target
0   0     1       1
1   0     2       1
2   0     2       1
3   0     2       1
4   0     2       1
```

linear regression (predicting 'oldpeak' – a continuous value) and logistic regression (predicting 'target' – heart disease yes/no)

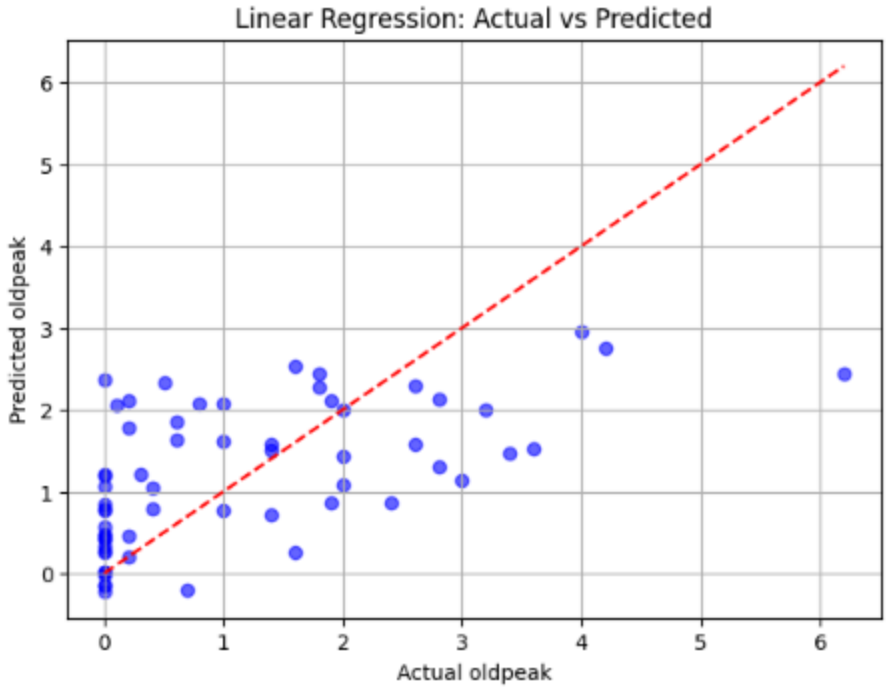
```
# Add these imports (if not already there)
```

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

# ─── LINEAR REGRESSION: Predict 'oldpeak' (continuous value) ───
print("\n" + "="*50)
print("LINEAR REGRESSION - Predicting oldpeak (ST depression)")
print("="*50)

X_lin = df.drop(['oldpeak', 'target'], axis=1)  # Features (drop what
we're predicting + binary target)
y_lin = df['oldpeak']  # Target for linear

X_train_lin, X_test_lin, y_train_lin, y_test_lin = train_test_split(
    X_lin, y_lin, test_size=0.2, random_state=42
)

lr = LinearRegression()
lr.fit(X_train_lin, y_train_lin)
y_pred_lin = lr.predict(X_test_lin)

print(f"MAE:  {mean_absolute_error(y_test_lin, y_pred_lin):.3f}")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_lin, y_pred_lin)):.3f}")
print(f"R²:   {r2_score(y_test_lin, y_pred_lin):.3f}")

# Plot
plt.figure(figsize=(7,5))
plt.scatter(y_test_lin, y_pred_lin, alpha=0.6, color='blue')
plt.plot([y_test_lin.min(), y_test_lin.max()], [y_test_lin.min(),
y_test_lin.max()], 'r--')
plt.xlabel("Actual oldpeak")
plt.ylabel("Predicted oldpeak")
plt.title("Linear Regression: Actual vs Predicted")
plt.grid(True)
```

```
plt.show()

# ——— LOGISTIC REGRESSION: Predict 'target' (0=no disease, 1=disease) ———
print("\n" + "="*50)
print("LOGISTIC REGRESSION - Predicting Heart Disease")
print("="*50)

X_log = df.drop('target', axis=1)  # Features
y_log = df['target']  # Binary target

X_train_log, X_test_log, y_train_log, y_test_log = train_test_split(
    X_log, y_log, test_size=0.2, random_state=42, stratify=y_log
)

log_model = LogisticRegression(max_iter=2000, random_state=42)
log_model.fit(X_train_log, y_train_log)
y_pred_log = log_model.predict(X_test_log)

acc = accuracy_score(y_test_log, y_pred_log)
print(f"Accuracy: {acc:.3f} ({acc*100:.1f}%)")

print("\nConfusion Matrix:")
print(confusion_matrix(y_test_log, y_pred_log))

print("\nClassification Report:")
print(classification_report(y_test_log, y_pred_log))

# Heatmap plot
plt.figure(figsize=(6,5))
sns.heatmap(confusion_matrix(y_test_log, y_pred_log), annot=True, fmt='d',
cmap='Greens', cbar=False)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix - Heart Disease')
plt.show()
```

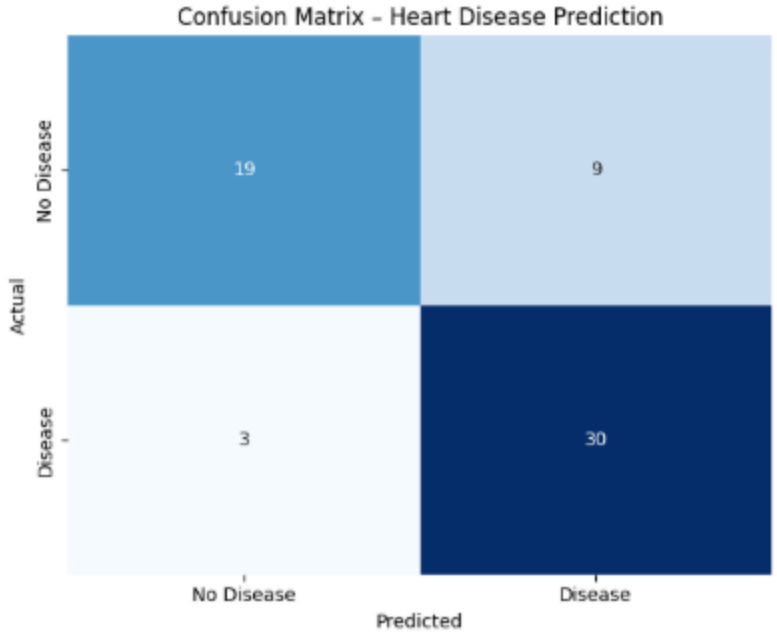
Output:

```
==================================================
LINEAR REGRESSION - Predicting oldpeak (ST depression)
==================================================
MAE:  0.874
RMSE: 1.123
R²:   0.321
```




```
==================================================
LOGISTIC REGRESSION - Predicting Heart Disease
==================================================
Accuracy: 0.803 (80.3%)
```

```
Confusion Matrix (raw numbers):
[[19  9]
 [ 3 30]]

Classification Report:
                precision    recall  f1-score   support

No Disease (0)       0.86      0.68      0.76        28
   Disease (1)       0.77      0.91      0.83        33

      accuracy                           0.80        61
     macro avg       0.82      0.79      0.80        61
  weighted avg       0.81      0.80      0.80        61
```




```
Done! ✓
```

You now have:

- Linear regression with plot
- Logistic regression with accuracy, classification report & confusion matrix heatmap