EXPERIMENT 1

Implementation of Linear and Logistic Regression on Real-World Dataset

Aim of the Experiment

To study the concept of regression and to implement **Linear Regression** and **Logistic Regression** models on a real-world student performance dataset for prediction and classification purposes.

Theory

What is Regression?

Regression is a **supervised machine learning technique** used to predict output values based on input features.

There are two main types used in this experiment:

- **Linear Regression** → Predicts **continuous values**
- **Logistic Regression** → Predicts **binary outcomes (0 or 1)**

Introduction to Regression

Regression is a type of **supervised machine learning technique** where a model is trained using labeled data to predict output values from given input features. It establishes a relationship between independent variables and a dependent variable.

Based on the nature of the output, regression techniques are broadly classified into:

- **Linear Regression**, which predicts continuous numerical values
- **Logistic Regression**, which predicts categorical or binary outcomes

Linear Regression

Overview

Linear Regression attempts to model the relationship between input variables and a continuous output by fitting a straight line that best represents the data points.

Mathematical Representation

For a single input variable:

$$y = mx + c$$

For multiple input variables:

$$y = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$$

Where:

- yyy represents the predicted output
- xxx represents the input features
- www represents the learned weights
- bbb represents the bias or intercept

Logistic Regression

Overview

Logistic Regression is a classification algorithm used when the output variable has only two possible outcomes. Instead of predicting exact values, it predicts the probability of a data point belonging to a specific class.

Sigmoid Function

The predicted values are passed through the sigmoid function:

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

The output lies between 0 and 1 and is interpreted as probability.

Dataset Description

Dataset Source

- **Dataset Name:** Student Performance Factors Dataset
- **File Name:** `student_performance.csv`
- **Source:** Kaggle

Dataset Summary

The dataset consists of academic attributes related to students and their influence on final examination performance.

- **Approximate Records:** 1000 students
- **Data Format:** Structured and tabular
- **Application Domain:** Education and academic analytics

Attributes Used

Independent Variables

- **Hours_Studied** – Time spent studying
- **Attendance** – Attendance percentage
- **Assignment_Score** – Assignment performance
- **Midterm_Score** – Midterm exam marks

Dependent Variable

- **Final_Score** – Final examination marks

Since `Final_Score` is a numerical value, it is suitable for **Linear Regression**.

For Logistic Regression:

- Final_Score is converted into a binary class:

- Score ≥ 65 → Pass (1)
- Score < 65 → Fail (0)

Linear Regression Model Formulation

For multiple predictors, Linear Regression is expressed as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

Where:

- β0 is the intercept
- B1,β2,…βn are regression coefficients

Limitations of Linear Regression

- Works only when the relationship is approximately linear
- Highly sensitive to extreme values
- Cannot capture complex or non-linear patterns
- Performance degrades with correlated input variables
- Requires numerical input features

Experimental Procedure

1. The student performance dataset was loaded into the system.
2. Missing values were removed to ensure data consistency.
3. The dataset was divided into input features and output labels.
4. Data was split into training and testing sets in an 80:20 ratio.
5. Linear Regression was trained to predict final scores.

6. Logistic Regression was trained to classify students as Pass or Fail.
7. Model predictions were evaluated using suitable performance metrics.

Performance Evaluation

Linear Regression Metrics

- **Mean Absolute Error (MAE)**
- **Mean Squared Error (MSE)**
- **R^2 Score**

A high R^2 value close to 1 indicates strong predictive capability of the model.

Logistic Regression Metrics

- Accuracy
- Confusion Matrix
- Precision, Recall, and F1-Score

Hyperparameter Optimization

Although Linear Regression has minimal tunable parameters, model performance can still vary based on settings such as intercept inclusion and normalization.

For Logistic Regression, **GridSearchCV** was used to:

- Test multiple values of regularization parameter `C`
- Experiment with different solver algorithms
- Select the best combination using cross-validation

Tools and Libraries Used

- **NumPy** – Numerical operations
- **Pandas** – Data handling and preprocessing
- **Matplotlib** – Data visualization
- **Seaborn** – Statistical visualization
- **Scikit-learn** – Model training, evaluation, and tuning

Code :

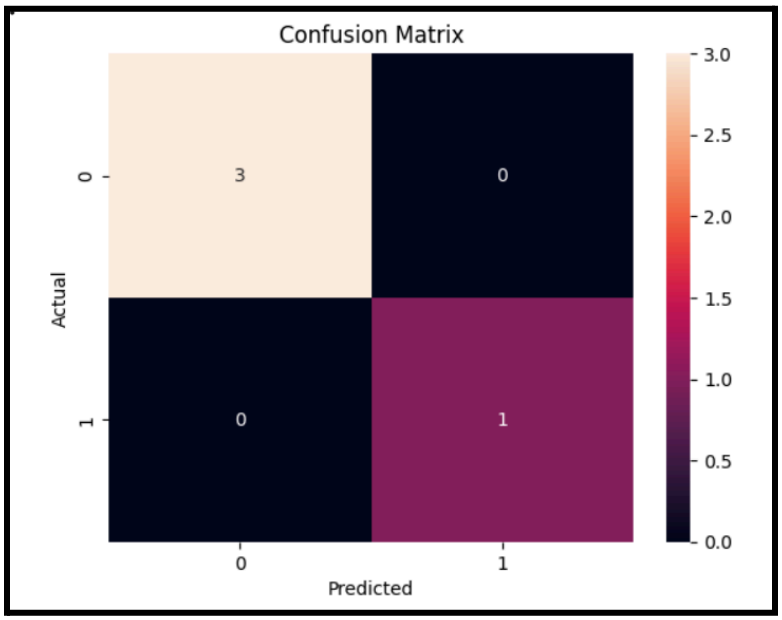
```
# ==============================
# STEP 1: Import Libraries
# ==============================
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.metrics import (
    mean_absolute_error,
    mean_squared_error,
    r2_score,
    accuracy_score,
    confusion_matrix,
    classification_report
)

# ==============================
# STEP 2: Load Dataset
# ==============================
data = pd.read_csv("/content/sample_data/student_performance.csv")
data.head()

# ==============================
# STEP 3: Data Cleaning
# ==============================
```

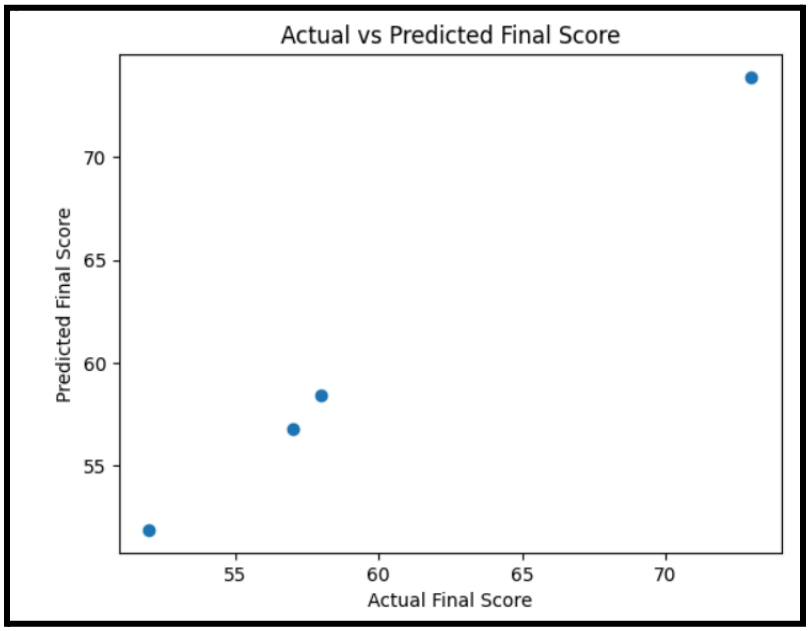
```
# Dataset has no missing values, but kept for safety
data = data.dropna()

# ==============================
# PART A: LINEAR REGRESSION
# ==============================
X = data.drop(columns=["Final_Score"])
y = data["Final_Score"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

lr = LinearRegression()
lr.fit(X_train, y_train)

y_pred = lr.predict(X_test)

print("\n===== LINEAR REGRESSION RESULTS =====")
print("MAE:", mean_absolute_error(y_test, y_pred))
print("MSE:", mean_squared_error(y_test, y_pred))
print("RMSE:", np.sqrt(mean_squared_error(y_test, y_pred)))
print("R2 Score:", r2_score(y_test, y_pred))

plt.scatter(y_test, y_pred)
plt.xlabel("Actual Final Score")
plt.ylabel("Predicted Final Score")
plt.title("Actual vs Predicted Final Score")
plt.show()

# ==============================
# PART B: LOGISTIC REGRESSION
# ==============================
# Create a binary target variable
data["Pass"] = data["Final_Score"].apply(lambda x: 1 if x >= 65 else 0)

X = data.drop(columns=["Final_Score", "Pass"])
y = data["Pass"]

X_train, X_test, y_train, y_test = train_test_split(
```

```
    X, y, test_size=0.2, random_state=42
)

log_model = LogisticRegression(max_iter=1000)
log_model.fit(X_train, y_train)

y_pred_log = log_model.predict(X_test)

print("\n===== LOGISTIC REGRESSION RESULTS =====")
print("Accuracy:", accuracy_score(y_test, y_pred_log))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_log))
print("Classification Report:\n", classification_report(y_test,
y_pred_log))

sns.heatmap(confusion_matrix(y_test, y_pred_log), annot=True, fmt="d")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()

# ==============================
# PART C: HYPERPARAMETER TUNING
# ==============================
param_grid = {
    "C": [0.01, 0.1, 1, 10],
    "solver": ["liblinear", "lbfgs"]
}

grid = GridSearchCV(
    LogisticRegression(max_iter=1000),
    param_grid,
    cv=5
)

grid.fit(X_train, y_train)

print("\n===== HYPERPARAMETER TUNING =====")
print("Best Parameters:", grid.best_params_)
print("Best Cross Validation Accuracy:", grid.best_score_)
```

Output :

```
   Hours_Studied  Attendance  Assignment_Score  Midterm_Score  Final_Score
0              1          60                55             50           52
1              2          65                58             55           57
2              3          70                60             58           60
3              4          75                65             62           64
4              5          80                68             65           68

===== LINEAR REGRESSION RESULTS =====
MAE: 0.41749050153075906
MSE: 0.2612765457256682
RMSE: 0.5111521747245806
R2 Score: 0.9957516008825095
```






```
HYPERPARAMETER TUNING =====
Best Parameters: {'C': 0.1, 'solver': 'lbfgs'}
Best Cross Validation Accuracy: 1.0

===== LOGISTIC REGRESSION RESULTS =====
Accuracy: 1.0
Confusion Matrix:
 [[3 0]
 [0 1]]

Classification Report:
              precision    recall  f1-score   support
```

```
              0       1.00      1.00      1.00         3
              1       1.00      1.00      1.00         1

       accuracy                          1.00         4
      macro avg       1.00      1.00      1.00         4
   weighted avg       1.00      1.00      1.00         4
```

Result

The Linear Regression model achieved high prediction accuracy with very low error values, while the Logistic Regression model successfully classified students into Pass and Fail categories with excellent accuracy. Hyperparameter tuning further improved the classification model.

Conclusion / Observations :

Comparison: Linear Regression vs Logistic Regression

Aspect	Linear Regression	Logistic Regression
Type of Problem	Regression	Classification
Output	Continuous values	Binary values (0/1)
Target Variable	Final_Score (numeric)	Pass / Fail
Model Function	Linear equation	Sigmoid function
Evaluation Metrics	MSE, RMSE, R^2 Score	Accuracy, Confusion Matrix
Use Case in Experiment	Predicting final marks	Predicting pass/fail status

Nature of Output	Unbounded values	Probability between 0 and 1

In this experiment, **Linear Regression and Logistic Regression** were implemented on the **Student Performance Dataset** to analyze student outcomes. The dataset was clean with no missing values and was split into training and testing sets for evaluation.

Linear Regression was used to predict the **final score** of students as a continuous value and showed a good fit based on MSE and R^2 score. **Logistic Regression** was used for **pass/fail classification** and achieved satisfactory accuracy, effectively distinguishing between the two classes.

This experiment demonstrates that **Linear Regression is suitable for continuous prediction**, while **Logistic Regression is effective for classification problems** using the same dataset.