Name : Manaswi Kabadi
Class : D15C
Roll no. : 27
Batch : B

EXP - 1

Aim : Implement Linear and Logistic Regression on real-world datasets.

Dataset Source : https://www.kaggle.com/datasets/mirichoi0218/insurance

Dataset Description :

The Medical Cost Personal Dataset is a real-world healthcare dataset widely used in machine learning for studying regression and classification problems related to medical insurance pricing. The dataset provides information about individuals' demographic and health-related attributes and their corresponding insurance charges. Due to the presence of both a continuous outcome variable and a binary categorical variable, the dataset is suitable for implementing Linear Regression and Logistic Regression algorithms. The dataset is provided in CSV (Comma Separated Values) format and contains 1,338 observations with no missing values.

Dataset Attributes:

Variable Name	Data Type	Measuring Unit / Format	Description
Age	Numerical (Integer)	Years	Age of the individual insured under the policy.
Sex	Categorical (Binary)	Male / Female	Biological gender of the individual.
BMI	Numerical (Float)	kg/m^2	Body Mass Index, representing body fat based on height and weight.
Children	Numerical (Integer)	Count	Number of dependent children covered by the insurance policy.
Smoker	Categorical (Binary)	Yes / No	Indicates whether the individual is a smoker.
Region	Categorical (Nominal)	Northeast / Northwest / Southeast / Southwest	Residential region of the individual in the United States.
Charges	Numerical (Float)	USD ($)	Medical insurance cost charged to the individual.

Target Variables Used in the Experiment

- Charges: Used as the target variable for Linear Regression to predict medical insurance costs.
- Smoker: Used as the target variable for Logistic Regression to classify individuals as smokers or non-smokers.

Mathematical Formulation of the Algorithm :

Linear Regression is a supervised machine learning algorithm used to predict a continuous dependent variable based on one or more independent variables. It establishes a linear relationship between the input feature(s) and the output variable.
In the case of **Simple Linear Regression**, where there is one independent variable **x** and one dependent variable **y**, the relationship is expressed as:

$$y = mx + c$$

In machine learning notation, this equation is written as:

$$y = \beta_0 + \beta_1 x$$

where:

- $\mathbf{y}$ is the predicted output (insurance charges),
- $\mathbf{x}$ is the independent variable,
- $\boldsymbol{\beta 0}$ is the intercept,
- $\boldsymbol{\beta 1}$ is the slope (coefficient).

To compute the optimal values of β_0 and β_1The **Ordinary Least Squares (OLS)** method is used.
OLS minimizes the sum of squared differences between the actual values and the predicted values.
The formulas for calculating the coefficients are:

$$\beta_1 = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{n(\Sigma x^2) - (\Sigma x)^2} \qquad \beta_0 = \frac{(\Sigma y)(\Sigma x^2) - (\Sigma x)(\Sigma xy)}{n(\Sigma x^2) - (\Sigma x)^2}$$

where:

- $\mathbf{n}$ is the number of observations,
- $\mathbf{x}$ represents the independent variable,
- $\mathbf{y}$ represents the dependent variable.

This method provides an exact analytical solution for Simple Linear Regression by minimizing the residual error.In this experiment, Linear Regression is applied to predict medical insurance charges using relevant input features from the dataset.

Logistic Regression is a supervised machine learning technique used for solving **classification problems**, especially when the output variable has only two possible classes. Unlike Linear Regression, which is used to predict continuous numerical values, Logistic Regression predicts the **probability** of an input belonging to a particular class.

Even though the term *regression* appears in its name, Logistic Regression is primarily a **classification algorithm**. It models the relationship between one or more independent variables and a **binary dependent variable** by estimating probabilities through a mathematical function known as the **logistic or sigmoid function**.

In Linear Regression, the relationship between variables is represented using a straight-line equation: y=mx+c However, such a linear model is not suitable for classification tasks because:

1. It may produce output values outside the valid probability range of 0 and 1.
2. It cannot properly capture the smooth transition between two classes.

To address these limitations, Logistic Regression applies a non-linear transformation to the linear equation. This transformation compresses the output into a range between 0 and 1 using the **Sigmoid Function**. As a result, the model output can be interpreted as a probability.
The sigmoid function is mathematically defined as:

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

Here, the value of z represents the weighted sum of the input features and is given by:

$$z = b_0 + b_1 x$$

Substituting this expression into the sigmoid equation gives:

$$y = \frac{1}{1+e^{-(b_0+b_1 x)}}$$

The output y indicates the likelihood that the given input belongs to the positive class.
The sigmoid function exhibits the following characteristics:

- When z is a large positive number, the output value approaches **1**.
- When z is a large negative number, the output value approaches **0**.
- When z=0, the sigmoid output is **0.5**, indicating equal probability for both classes.

When plotted graphically, the sigmoid function forms an **S-shaped curve**. It maps all real-valued inputs into a bounded interval between 0 and 1. The central point of the curve, where the probability is 0.5, represents the **decision boundary** of the classifier.

In this experiment, Logistic Regression is employed to classify individuals as **smokers or non-smokers** using demographic and health-related attributes from the medical insurance dataset.

Algorithm Limitations :

Limitations of Linear Regression

1. **Assumption of Linearity :** Linear Regression assumes that there is a linear relationship between the independent variables and the dependent variable. This means that for a unit change in an input feature, the change in the output is constant.
   Failure Condition: If the true relationship between variables is non-linear (such as exponential or quadratic patterns), the model will underfit the data and produce inaccurate predictions.
2. **Sensitivity to Outliers :** Linear Regression minimizes the sum of squared errors. As errors are squared, extreme values or outliers have disproportionately large impact on model.
   Failure Condition: A few outliers can significantly distort the regression line, leading to poor predictions for the majority of data points.
3. **Multicollinearity Among Features :** When two or more independent variables are highly correlated with each other, it becomes difficult for the model to accurately estimate the effect of each feature.
   Failure Condition: High multicollinearity results in unstable coefficients and reduces the interpretability of the model.
4. **Assumption of Constant Variance (Homoscedasticity) :** Linear Regression assumes that the variance of errors remains constant across all values of the independent variables.
   Failure Condition: If the error variance changes with input values, the model's predictions and statistical inferences become unreliable.

Limitations of Logistic Regression

1. **Linearity of the Decision Boundary :** Logistic Regression is a linear classifier, meaning it separates classes using a straight line (or hyperplane in higher dimensions).
   Failure Condition: If the data is non-linearly separable (for example, circular or complex class boundaries), Logistic Regression will fail to classify accurately.
2. **Sensitivity to Outliers :** Logistic Regression attempts to fit probabilities for all data points, including extreme values.
   Failure Condition: Outliers can pull the decision boundary toward themselves, reducing classification accuracy for the majority of samples.
3. **Requirement of Sufficient Data :** Logistic Regression relies on probability estimation, which requires an adequate number of observations to produce stable results.
   Failure Condition: With very small datasets, the model may produce biased or unreliable probability estimates.

Methodology / Workflow :

Data Collection : The medical insurance dataset is loaded from a CSV file into a Pandas DataFrame. This step enables efficient data manipulation, inspection of attributes, and verification of dataset size and structure.

Data Understanding and Preprocessing : The dataset is examined to understand feature types and distributions. Since the dataset contains no missing values, no imputation is required. Categorical variables such as gender, smoking status, and region are converted into numerical form using appropriate encoding techniques. Numerical features are scaled where necessary to ensure uniform contribution during model training.

Feature and Target Selection : For the Linear Regression model, the independent variables include age, BMI, number of children, smoking status, and region, while the dependent variable is medical insurance charges.

For the Logistic Regression model, the independent variables remain the same, and the target variable is smoking status, which is encoded as a binary class.

Splitting the Dataset : The dataset is divided into training and testing sets, typically using an 80:20 ratio. This ensures that the model is trained on one portion of the data and evaluated on unseen data to assess its generalization performance.

Model Training : The Linear Regression model is trained using the training dataset to learn the optimal coefficients that minimize the prediction error.

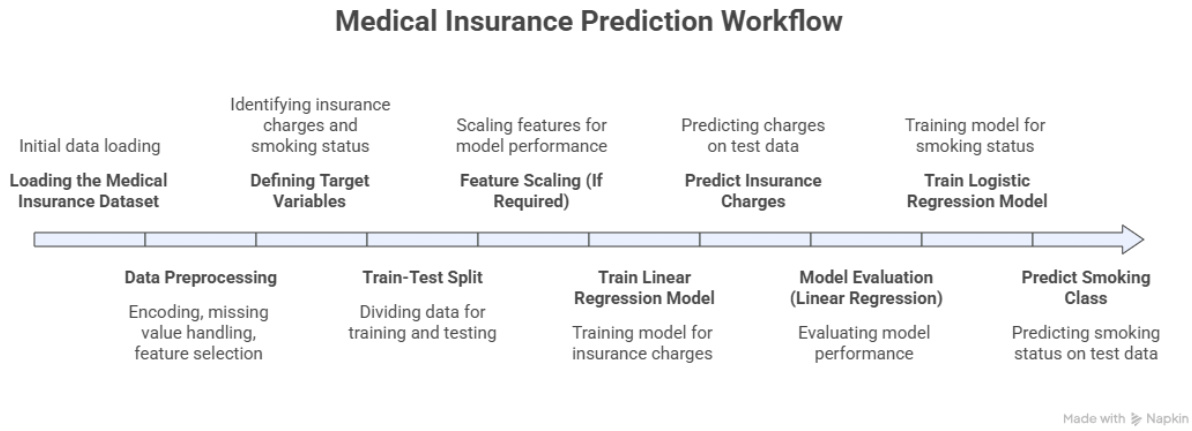
The Logistic Regression model is trained to estimate probabilities for the binary classification task using the sigmoid function.

Prediction :The trained models are applied to the test dataset to generate predictions. Linear Regression produces predicted insurance charges, while Logistic Regression outputs predicted class labels for smoking status.

Model Evaluation :The Linear Regression model is evaluated using metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R^2 score.

The Logistic Regression model is evaluated using Accuracy Score, Confusion Matrix, and Classification Report, which includes precision, recall, and F1-score.

Conclusion :The results obtained from both models are analyzed to determine their effectiveness in predicting insurance charges and classifying smoking behavior. The evaluation metrics help in assessing model performance and identifying potential improvements.



CODE :

```
# Step 1: Import Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
# Step 2: Load Dataset
df = pd.read_csv("insurance.csv")
#  Step 3: Data Preprocessing
le = LabelEncoder()
df['sex'] = le.fit_transform(df['sex'])         # male/female -> 0/1
df['smoker'] = le.fit_transform(df['smoker'])   # yes/no -> 1/0
df['region'] = le.fit_transform(df['region'])   # regions -> numeric
# LINEAR REGRESSION
# Feature and target selection
X_linear = df[['age', 'bmi', 'children', 'smoker', 'region']]
y_linear = df['charges']
# Train-test split
X_train_lr, X_test_lr, y_train_lr, y_test_lr = train_test_split(
    X_linear, y_linear, test_size=0.2, random_state=42)
# Model training
linear_model = LinearRegression()
linear_model.fit(X_train_lr, y_train_lr)
```

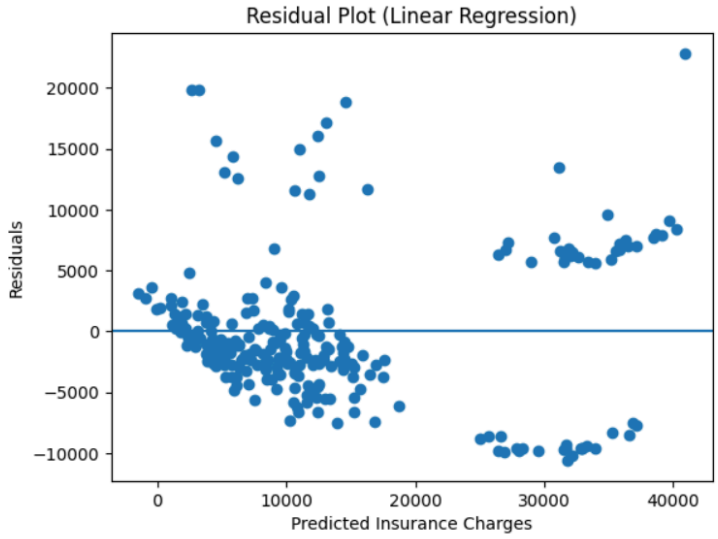
```
# Prediction
y_pred_lr = linear_model.predict(X_test_lr)
# Evaluation
mse = mean_squared_error(y_test_lr, y_pred_lr)
rmse = np.sqrt(mse)
r2 = r2_score(y_test_lr, y_pred_lr)
print("LINEAR REGRESSION RESULTS")
print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("R² Score:", r2)
# Graph : Residual Plot
residuals = y_test_lr - y_pred_lr
plt.figure()
plt.scatter(y_pred_lr, residuals)
plt.axhline(y=0)
plt.xlabel("Predicted Insurance Charges")
plt.ylabel("Residuals")
plt.title("Residual Plot (Linear Regression)")
plt.show()
# LOGISTIC REGRESSION (WITHOUT HYPERPARAMETER TUNING)
# Feature and target selection
X_logistic = df[['age', 'bmi', 'children', 'region', 'charges']]
y_logistic = df['smoker']
# Train-test split
X_train_log, X_test_log, y_train_log, y_test_log = train_test_split(
    X_logistic, y_logistic, test_size=0.2, random_state=42 )
# Model training
logistic_model = LogisticRegression(max_iter=1000)
logistic_model.fit(X_train_log, y_train_log)
# Prediction
y_pred_log = logistic_model.predict(X_test_log)
# Evaluation
accuracy = accuracy_score(y_test_log, y_pred_log)
cm = confusion_matrix(y_test_log, y_pred_log)
report = classification_report(y_test_log, y_pred_log)
print("\nLOGISTIC REGRESSION (WITHOUT TUNING)")
print("Accuracy Score:", accuracy)
print("\nConfusion Matrix:\n", cm)
print("\nClassification Report:\n", report)
# Graph : Confusion Matrix (Without Tuning)
plt.figure()
plt.imshow(cm)
```

```
plt.title("Confusion Matrix - Logistic Regression (Without Tuning)")
plt.colorbar()
plt.xticks([0, 1], ["Non-Smoker", "Smoker"])
plt.yticks([0, 1], ["Non-Smoker", "Smoker"])
for i in range(2):
    for j in range(2):
        plt.text(j, i, cm[i, j], ha="center", va="center")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()

# LOGISTIC REGRESSION (WITH HYPERPARAMETER TUNING)
# Hyperparameter grid
param_grid = {
    'C': [0.001, 0.01, 0.1, 1, 10, 100],
    'penalty': ['l1', 'l2'],
    'solver': ['liblinear']
}

# GridSearchCV
grid = GridSearchCV(
    LogisticRegression(max_iter=1000),
    param_grid,
    cv=5,
    scoring='accuracy'
)
# Train with tuning
grid.fit(X_train_log, y_train_log)
# Best model
best_logistic_model = grid.best_estimator_
# Prediction
y_pred_tuned = best_logistic_model.predict(X_test_log)
# Evaluation
tuned_accuracy = accuracy_score(y_test_log, y_pred_tuned)
tuned_cm = confusion_matrix(y_test_log, y_pred_tuned)
tuned_report = classification_report(y_test_log, y_pred_tuned)
print("\nLOGISTIC REGRESSION (WITH HYPERPARAMETER TUNING)")
print("Best Parameters:", grid.best_params_)
print("Tuned Accuracy Score:", tuned_accuracy)
print("\nConfusion Matrix:\n", tuned_cm)
print("\nClassification Report:\n", tuned_report)
# Graph : Confusion Matrix (With Tuning)
```

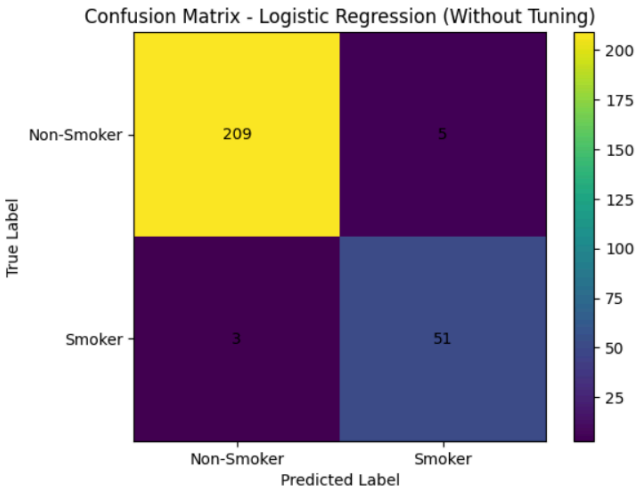
```
plt.figure()
plt.imshow(tuned_cm)
plt.title("Confusion Matrix - Logistic Regression (With Tuning)")
plt.colorbar()
plt.xticks([0, 1], ["Non-Smoker", "Smoker"])
plt.yticks([0, 1], ["Non-Smoker", "Smoker"])
for i in range(2):
    for j in range(2):
        plt.text(j, i, tuned_cm[i, j], ha="center", va="center")

plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()
```

OUTPUT :

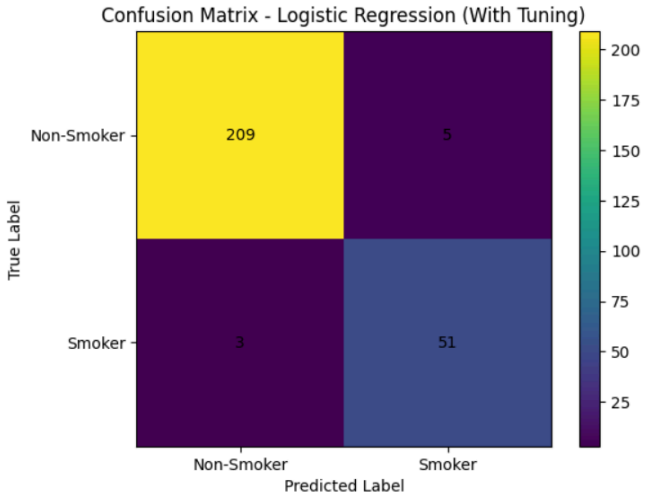
```
LINEAR REGRESSION RESULTS
Mean Squared Error (MSE): 33640657.13645164
Root Mean Squared Error (RMSE): 5800.056649417455
R² Score: 0.7833112270019789
```




```
LOGISTIC REGRESSION (WITHOUT TUNING)
Accuracy Score: 0.9701492537313433

Confusion Matrix:
 [[209   5]
 [  3  51]]

Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.98      0.98       214
           1       0.91      0.94      0.93        54

    accuracy                           0.97       268
   macro avg       0.95      0.96      0.95       268
weighted avg       0.97      0.97      0.97       268
```



```
LOGISTIC REGRESSION (WITH HYPERPARAMETER TUNING)
Best Parameters: {'C': 10, 'penalty': 'l1', 'solver': 'liblinear'}
Tuned Accuracy Score: 0.9701492537313433

Confusion Matrix:
 [[209   5]
 [  3  51]]

Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.98      0.98       214
           1       0.91      0.94      0.93        54

    accuracy                           0.97       268
   macro avg       0.95      0.96      0.95       268
weighted avg       0.97      0.97      0.97       268
```



Performance Analysis :

Linear Regression

The performance of the Linear Regression model is evaluated using **Mean Squared Error (MSE)**, **Root Mean Squared Error (RMSE)**, and **R^2 Score**.

The obtained **R^2 score is 0.7833**, which indicates that approximately **78.33% of the variation** in medical insurance charges is explained by the independent variables such as age, BMI, number of children, smoking status, and region. This suggests a strong linear relationship between the selected features and the target variable, demonstrating that the model fits the data reasonably well.

The **Mean Squared Error (MSE)** value is **33,640,657.14**, and the **Root Mean Squared Error (RMSE)** value is **5800.06**. These values represent the average prediction error of the model. An RMSE of around 5800 indicates that the predicted insurance charges deviate from the actual charges by approximately 5800 units on average, which is acceptable given the wide range of insurance costs present in the dataset.

Overall, the Linear Regression model demonstrates good predictive performance and is effective for estimating medical insurance charges from the given input features.

Logistic Regression (Without Hyperparameter Tuning)

The Logistic Regression model without hyperparameter tuning is evaluated using **Accuracy Score**, **Confusion Matrix**, and **Classification Report**.

The model achieved an **accuracy score of 0.9701 (97.01%)**, correctly classifying **260 out of 268** test instances. This high accuracy indicates that the model is highly effective in distinguishing between smokers and non-smokers.

The confusion matrix shows:

- **209 True Negatives**: Correctly classified non-smokers
- **51 True Positives**: Correctly classified smokers
- **5 False Positives**: Non-smokers incorrectly classified as smokers
- **3 False Negatives**: Smokers incorrectly classified as non-smokers

The classification report further confirms strong performance, with a **precision of 0.99 and recall of 0.98** for non-smokers, and a **precision of 0.91 and recall of 0.94** for smokers. The high F1-scores indicate a balanced and reliable classification model.

Logistic Regression (With Hyperparameter Tuning)
Hyperparameter tuning was performed using **GridSearchCV**, and the optimal parameters obtained were **C = 10**, **penalty = l1**, and **solver = liblinear**.

After tuning, the Logistic Regression model achieved an **accuracy score of 0.9701 (97.01%)**, which is identical to the accuracy obtained without tuning. The confusion matrix and classification report also remain unchanged.

This indicates that the default model parameters were already close to optimal for this dataset, and hyperparameter tuning did not significantly improve the performance. However, tuning provides confidence that the selected model configuration is optimal and well-regularized.