Assignment 1

Objective:

You are required to demonstrate your understanding of regression and classification models by applying **Linear Regression (Simple and Multiple)**, **Polynomial Regression**, and **Logistic Regression** on real-world datasets. You will analyze, build, evaluate, and compare models using **scikit-learn** and other relevant Python libraries (e.g., pandas, matplotlib, seaborn).

Tasks:

1. Regression Models

Choose a **regression dataset** (e.g., car prices, student performance, etc.) from Kaggle, UCI Machine Learning Repository, or any publicly available dataset.

1. **Exploratory Data Analysis (EDA):**
   - Load the data and clean it (handle missing values, data types, encoding, extra if any).
   - Visualize the data using appropriate plots (scatter plots, histograms, correlation heatmap).
2. **Preprocessing:**
   - Select appropriate numeric features for regression.
   - Scale the features if needed.
   - Split the data into **training and testing sets** (e.g., 80/20).
3. **Simple Linear Regression:**
   - Pick **one feature** that most correlates with the target variable.
   - Fit a simple linear regression model.
   - Visualize the regression line and report metrics (**R^2**, **MSE**).
4. **Multiple Linear Regression:**
   - Use **multiple features** to predict the target.
   - Compare results with the simple linear regression.
5. **Polynomial Regression:**
   - Apply **Polynomial Regression** on the best-performing simple linear feature.
   - Try polynomial degrees **2, 3, and 4**.
   - Scale the polynomial features if needed.
   - Compare models using visualizations and metrics.

2. Logistic Regression with Classification

Choose a **classification dataset** (e.g., Titanic survival, diabetes prediction, breast cancer, etc.)

1. **Preprocessing:**
   - Encode categorical variables and clean the data.
   - Scale features if needed.
   - Split the dataset into **training and testing sets** (e.g., 80/20 split).

2. **Modeling with Logistic Regression:**
    - Train a logistic regression model and Predict on your data.
3. **Model Evaluation:**
    - Generate a **confusion matrix**.
    - Visualize the confusion matrix using seaborn or matplotlib.

Deliverables:

Submit a **well-documented Jupyter Notebook** that includes:

- All code with comments and explanations.
- Data cleaning and preprocessing steps.
- Model training, evaluation, and comparison.
- Clear markdown sections for each part.
- A final markdown cell with a **summary of your findings**.