# Programming Assignment: Multiple feature regression and Multiclass classification

## Problem-1: House Price Prediction (Multiple Feature Regression)

**Objective:** Implement a multiple feature regression model to predict house prices based on various features. This problem will guide you through data loading, preprocessing, model training, and evaluation for a real-world regression task.

1. **Data Loading and Initial Exploration:**
   - Load a house price dataset (e.g., California Housing dataset from `sklearn.datasets.fetch_california_housing` or a similar publicly available dataset like Boston Housing).
   - Display the first 5 rows of the dataset, its shape, and a summary of its statistical properties using Pandas.
   - Identify the target variable (house price) and the feature variables.
2. **Data Preprocessing:**
   - Check for any missing values in the dataset and decide on an appropriate strategy to handle them (e.g., imputation or removal). Justify your choice.
   - Perform feature scaling on the numerical features using `StandardScaler` or `MinMaxScaler` from `sklearn.preprocessing`. Explain why feature scaling is important for some regression algorithms.
   - If applicable, handle categorical features using one-hot encoding or other appropriate techniques.
3. **Model Implementation - Multiple Linear Regression:**
   - Split the preprocessed dataset into training and testing sets (e.g., 80% training, 20% testing).
   - Implement Multiple Linear Regression from scratch using NumPy (i.e., using the normal equation or gradient descent).
   - Alternatively, use `sklearn.linear_model.LinearRegression` to train a model on the training data.
   - Make predictions on the test set.
4. **Model Evaluation:**
   - Calculate and report the Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared score on the test set.
   - Discuss what these metrics indicate about the model's performance.
5. **Feature Importance and Visualization (Optional/Bonus):**

- If using `sklearn.linear_model.LinearRegression`, extract the coefficients of the model.
- Visualize the relationship between the predicted prices and actual prices using a scatter plot.
- (Bonus) Create a bar chart showing the importance of each feature (based on absolute coefficient values) in predicting house prices.

This problem aims to solidify your understanding of the end-to-end process of building and evaluating a regression model for a practical machine learning task.

## Problem-2: Multiclass Classification with IRIS Dataset

**Objective:** Implement and evaluate multiclass classification algorithms on the Iris dataset. This problem will help you understand how models learn to classify multiple distinct categories based on input features.

1. **Data Loading and Initial Exploration:**
    - Load the Iris dataset using `sklearn.datasets.load_iris()`.
    - Display the first 5 rows of the feature data (`data`) and the target variable (`target`).
    - Print the names of the features (`feature_names`) and the target classes (`target_names`).
    - Describe the shape of the dataset and check for any missing values.
2. **Data Preprocessing and Splitting:**
    - Split the dataset into training and testing sets (e.g., 70% training, 30% testing) using `sklearn.model_selection.train_test_split`. Ensure `stratify` is used for the target variable to maintain class proportions.
    - Explain why stratification is important for multiclass classification problems.
    - Perform feature scaling on the numerical features using `StandardScaler` from `sklearn.preprocessing`. Apply the scaler to both training and testing sets separately (fit on training, transform both).
3. **Model Implementation - Logistic Regression (Multinomial):**
    - Implement a Logistic Regression model for multiclass classification using `sklearn.linear_model.LogisticRegression`.
    - Train the model on the scaled training data.
    - Make predictions on the scaled test set.
4. **Model Implementation - Support Vector Machine (SVM):**
    - Implement a Support Vector Machine classifier with a linear kernel using `sklearn.svm.SVC(kernel='linear')`.
    - Train the model on the scaled training data.
    - Make predictions on the scaled test set.

5. **Model Evaluation and Comparison:**
    - For both Logistic Regression and SVM models:
        - Calculate and report the accuracy score on the test set.
        - Display the classification report (precision, recall, f1-score) for each class.
        - Generate and display the confusion matrix.
    - Compare the performance of Logistic Regression and SVM on this dataset. Discuss which model performed better and why, considering the characteristics of the Iris dataset.
6. **Visualization of Decision Boundaries (Bonus):**
    - (Bonus) For one of the models, select two features and visualize the decision boundaries learned by the model. This will require training a new model using only two features for easier visualization. Clearly label the regions corresponding to each class.

## Evaluation Criteria

Your assignment will be evaluated based on the following:

- **Correctness:** Proper implementation of Regression and Classification algorithms.
- **Functionality:** The code runs without errors and produces expected outputs.
- **Visualization Quality:** Clarity and effectiveness of generated plots (regression line, decision boundary).
- **Code Quality:** Readability, comments, and adherence to Python best practices.
- **Understanding:** Demonstrated understanding of the underlying concepts of both algorithms.

## Due Date

Please submit your completed Google Colab notebook by Oct 9, 2025 5:00 PM GMT+5:30.

[Link to Google Colab Notebook to begin with Assignment-2]
PostMidSem_Assignment-2.ipynb