

# Programming Assignment-5 (PH227) : Data Preprocessing for ML model training

## Objective

This assignment focuses on the critical role of data preprocessing, feature selection, and robust model validation in developing high-performing machine learning models. You will implement these techniques and observe their impact on a regression task, highlighting the importance of thorough data preparation.

## Problem Description: Predicting Housing Prices

In this assignment, you will build a regression model to predict housing prices using a real-world dataset. The goal is to demonstrate how careful data handling, feature engineering, and proper validation can significantly improve model accuracy and generalization. Use sci-kit learn library for the tasks assigned like importing dataset, and ML models.

## Instructions

Complete the following programming exercises. For each section, you are expected to implement the logic from scratch where specified and provide discussions on your findings.

## Part 1: Data Loading and Initial Exploration

### 1. Load Dataset:

- Load the California Housing dataset using the sklearn library `sklearn.datasets.fetch_california_housing`) [\[Link\]](#).
- Display the first 5 rows, the shape, and a summary of statistical properties using Pandas.
- Identify the target variable (housing price) and the feature variables.

### 2. Initial Model Training (Baseline):

- Split the raw dataset (without any preprocessing) into training and testing sets (e.g., 80% training, 20% testing).
- Train following models on this raw data (import models from sklearn library)

- simple Linear Regression model  
(`sklearn.linear_model.LinearRegression`)
- Polynomial regression (degree=2)
- Decision Tree Regressor (use default parameters)
- Random Forest Regressor (use default parameters)
- Adaboost Regressor (use default parameters)
- Make predictions on the test set.
- Calculate and report the Mean Squared Error (MSE). This will serve as your baseline performance.

## Part 2: Data Preprocessing and Handling

In this section, you will systematically apply various data preprocessing techniques. You are encouraged to explore different strategies and justify your choices.

### 1. Outlier Detection and Handling:

- Construct a histogram and identify potential outliers.
- Implement a method to detect outliers (using the Inter Quartile Range (IQR) method ([resource link](#))).
- Apply removal strategy to handle outliers.

### 2. Feature Scaling:

- Perform feature scaling on numerical features using `StandardScaler` from `sklearn.preprocessing`.
- Explain why feature scaling is crucial for many machine learning algorithms.

### 3. Feature Engineering (Optional/Bonus):

- Based on your understanding of the dataset, create at least one new feature by combining or transforming existing features. Explain the rationale behind your new feature.

## Part 3: Comparing Model Performance

### 1. Final Model Evaluation:

- Train all the five models on the *entire* preprocessed data (80% training, 20% testing).
- Make predictions on the test data set.
- Calculate and print the MSE.

### 2. Discussion and Comparison:

- Compare the performance metrics (MSE) of your:
  - Five baseline ML models (Part 1).
  - Five models trained on the above preprocessed dataset (Part 2).

- Clearly articulate how each preprocessing step impacted the model's performance and which model performed well compared to other models.

## Submission Guidelines

- It is important that code is well structured and executed step by step as mentioned in this document.
- Submit a single Jupyter Notebook (.ipynb) file containing all your code and detailed explanations.
- Clearly separate each part and sub-part of the assignment using markdown headings.
- Ensure all code blocks are executed and produce the expected outputs.
- Ensure your code is well-structured, readable, and follows Python best practices.
- For any plots, include appropriate titles, axis labels, and legends.

## Evaluation Criteria

Your assignment will be evaluated based on the following:

- **Correctness:** Proper implementation of data preprocessing techniques.
- **Functionality:** The code runs without errors and produces expected outputs.
- **Evaluation Metrics:** Accurate calculation and reporting of performance metrics (MSE, R-squared).
- **Code Quality:** Readability, comments, and adherence to Python best practices.
- **Understanding and Discussion:** Demonstrated understanding of the underlying concepts, justification of choices, and insightful discussions on the impact of various techniques on model performance.

## Due Date

Please submit your completed Jupyter Notebook by **Oct 28, 2025 11:59 PM GMT+5:30**