Multiple Linear Regression from Scratch

CSE 3812
Artificial Intelligence Assignment
Department of CSE
United International University

October 7, 2025

Objective

The goal of this assignment is to implement **Multiple Linear Regression** from scratch, including:

- Data preprocessing and splitting
- Model parameter initialization
- Forward pass (matrix multiplication)
- Loss calculation
- Gradient descent parameter updates
- Model evaluation

1 Steps

1.1 1. Data Preprocess & Split

1. Drop unnecessary columns:

   Remove columns that are not useful for prediction or contain identifiers:
   Examples include:

   -ID or Serial No. — these are just row identifiers and carry no predictive power.

   -Unnamed: 0 — sometimes created accidentally when saving CSV.

   Any duplicate columns or irrelevant textual descriptions.

2. Preprocess the dataset according to the given guidelines (e.g., handle missing values, normalization, encoding if required).

3. Split the dataset into training and testing sets:

$$(X_{\text{train}}, y_{\text{train}}), \quad (X_{\text{test}}, y_{\text{test}})$$

1.2 2. Model Function

Define the hypothesis function:

$$\hat{y} = X\theta + b$$

where $\theta = [\theta_1, \theta_2, \ldots, \theta_n]^T$ are the model parameters and b is the bias term.

1. Initialize parameters:

$$\theta = \text{random small values}, \quad b = 0$$

2. Perform matrix multiplication between training features and parameters:

$$\hat{y}_{\text{train}} = X_{\text{train}}\theta + b$$

3. Compute the loss (Mean Squared Error):

$$L = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2$$

   where m is the number of training samples.

4. Compute gradients (partial derivatives):

$$\frac{\partial L}{\partial \theta} \quad \text{and} \quad \frac{\partial L}{\partial b}$$

1.3 3. Gradient Descent

1. Update parameters using the learning rate α:

$$\theta := \theta - \alpha\frac{\partial L}{\partial \theta}$$

$$b := b - \alpha\frac{\partial L}{\partial b}$$

2. Repeat the forward pass and parameter updates until the loss becomes very low or convergence is reached.

1.4 4. Evaluation

After training is complete:

1. Predict values for the test set:

$$\hat{y}_{\text{test}} = X_{\text{test}}\theta_{\text{final}} + b$$

2. Evaluate model performance using accuracy or error metrics such as Mean Squared Error (MSE) or R^2 score.

Submission Instructions

- Submit your Jupyter Notebook or Python file.
- Include all plots (loss curve, predictions vs. actual).
- Clearly label each step in your code.

Bonus (Optional)

Find the best learning rate for this problem (hyper-parameter tuning)

Dataset for Testing

You may use the “Salary Dataset” from Kaggle, which has **no missing values** according to its description. You can access it here:
https://www.kaggle.com/datasets/elikplim/concrete-compressive-strength-data-set

You can instruct students to download this for the assignment and use it for model training/testing.

Helper Code Snippet

Here is a simple Python helper code template (e.g. in your assignment prompt) to assist students in loading, splitting, and basic operations:

‘‘‘

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt

# load data
df = pd.read_csv(path)


X = df[columns that are features]
y = df[columns that is label] also apply necessary rehape

# Train, Test split
X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=test_size, random_state=random_state
    )
    return X_train, X_test, y_train, y_test


n = ... #no. of total columns
theta =  # set of parameters..initialized to a value
bias = 0.0
losses = []
```

```
lr = 0.01

# loop (until loss is minimal)
        # calculate y_pred
        # calculate loss

        losses.append(loss)
        # calculate gradients
        # update theta and bias



# calculate y_pred using final theta and bias and X_test,

# calculate MSE (compare y_pred with y_test)


# plotting loss curve
```

‘‘‘