
📘 Simple Linear Regression – Complete Notes

🔹 Introduction to Regression

Regression is a supervised learning technique used to model the relationship between **input features** (independent variables) and a **continuous output variable** (dependent variable).

✅ What is Regression?

- It's a **predictive modeling** technique.
- Used when the **target variable is continuous** (e.g., house price, temperature, salary).
- Unlike **classification**, which predicts categories, regression predicts **quantitative values**.

🔹 Types of Regression

Simple Linear Regression:

- Only **one independent variable** X and **one dependent variable** Y.
- A straight line is used to model the relationship.

Multiple Linear Regression:

- Involves two or more independent variables X_1, X_2, ..., X_n.

Equation:

$y = b_0 + b_1x_1 + b_2x_2 + \cdots + b_nx_n$

🔹 Simple Linear Regression (SLR)

Simple Linear Regression finds the **best-fitting straight line** to describe the relationship between a single input feature X and a target variable Y.

SLR Model:

$\hat{y} = mx + b$

Where:

- $\hat{y}$: predicted value of the target variable
- m: slope of the line (rate of change)
- x: input variable
- b: intercept (value of y when x = 0)

🔹 Objective of Linear Regression

Find best values for m and b that **minimize prediction error**.

Mean Squared Error (MSE) is the most used error metric:

$MSE = (1/n) \times \sum(y_i - \hat{y}_i)^2$

Goal: find m and b to minimize this error.

🔹 Python Code Example (Step-by-step)

```
# 1. Import Required Libraries

import numpy as np

import matplotlib.pyplot as plt


# 2. Dataset

X = np.array([1, 2, 3, 4, 5])

Y = np.array([2, 4, 5, 4, 5])
```

```
# 3. Prediction Function
def predict(x, m, b):
    return m * x + b

# 4. Find Slope and Intercept
n = len(X)
mean_x = np.mean(X)
mean_y = np.mean(Y)

numerator = np.sum((X - mean_x) * (Y - mean_y))
denominator = np.sum((X - mean_x) ** 2)

m = numerator / denominator
b = mean_y - m * mean_x

# 5. Predict and Plot
Y_pred = predict(X, m, b)

plt.scatter(X, Y, color='blue', label='Actual Data')
plt.plot(X, Y_pred, color='red', label='Regression Line')
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Simple Linear Regression')
plt.legend()
plt.grid(True)
plt.show()
```

🔹 Intuition Behind Simple Linear Regression

Imagine a scatterplot of data points.

What are we doing?

➡️ Drawing a **straight line** through the data points to **minimize the overall distance (residuals)** between points and the line.

- A **linear relationship** means: when X increases, Y increases or decreases proportionally.
- We minimize the **sum of squared residuals** (vertical distances).

🔹 How to Calculate Slope (m) and Intercept (b)

We use the **Least Squares Method**.

✅ **Slope formula:**

$$m = \sum(x_i - \bar{x})(y_i - \bar{y}) / \sum(x_i - \bar{x})^2$$

✅ **Intercept formula:**

$$b = \bar{y} - m \times \bar{x}$$

Where:

- $\bar{x}$: mean of X
- $\bar{y}$: mean of Y

These formulas ensure the **fitted line has minimum error (MSE)**.

🔍 Alternate Explanation (OLS Intuition)

- **Fit a line** that minimizes vertical distances from points.
- Center the data using **mean**.
- Slope shows how much Y changes for each unit change in X.
- Intercept aligns the line with the data center.

This is the idea of **Ordinary Least Squares (OLS)**.

✅ **Summary**

- Simple Linear Regression predicts a **continuous output** from one input.
- The equation:

$\hat{y} = mx + b$

- m: change in y per unit change in x
- b: predicted y when x = 0
- Goal: **minimize MSE**
- Parameters are estimated using the **Least Squares Method**
