Linear Regression is a foundational supervised learning algorithm used for **regression**—predicting a **continuous numerical value**—by modeling a straight-line relationship between variables.

Its core goal is to find the "line of best fit" through the data that minimizes the total prediction error.

## Theory of Linear Regression

### 1. The Core Equation

Linear regression mathematically models the relationship between an **independent variable** (the input, X) and a **dependent variable** (the output, Y) using a simple linear equation:

y^​=β0​+β1​x+ϵ

Where:

* **y^​** (pronounced 'y-hat'): The **Predicted Value** of the dependent variable.
* **x**: The **Independent Variable** (or feature, e.g., advertising spend).
* **β0​** (Beta-zero): The **Intercept**. This is the value of y^​ when x is 0.
* **β1​** (Beta-one): The **Slope** or **Coefficient**. This represents how much y^​ changes for every one-unit increase in x.
* **ϵ** (Epsilon): The **Error Term** (or residual), representing the difference between the actual observed value (y) and the predicted value (y^​).

### 2. Simple vs. Multiple Linear Regression

* **Simple Linear Regression:** Uses **one** independent variable (x) to predict the dependent variable (y^​), as shown in the equation above.
* Multiple Linear Regression: Uses two or more independent variables (x1​,x2​,…) to predict the dependent variable.  
    
  y^​=β0​+β1​x1​+β2​x2​+⋯+βn​xn​

### 3. The Estimation Method (Ordinary Least Squares)

The "best" line is determined using the **Ordinary Least Squares (OLS)** method.

* **Goal:** To find the values for the coefficients (β0​,β1​,…) that **minimize the sum of the squared errors** (or residuals).
* **Residual:** The vertical distance between the actual data point and the regression line. By squaring the errors, OLS ensures that positive and negative errors don't cancel each other out, and it heavily penalizes large errors.

## Practical Examples and Interpretations

### Example 1: Advertising Spend vs. Sales (Simple Linear Regression)

**Scenario:** A company wants to know how much its TV advertising budget impacts monthly sales.

| Variable | Role | Units |
| --- | --- | --- |
| **Sales** (Y) | **Dependent Variable** (Output) | Thousands of units |
| **TV Ad Spend** (X) | **Independent Variable** (Input) | Thousands of dollars |

Learned Equation Example:

Sales (in thousands)=5.0+0.045×TV Ad Spend (in thousands)

**Interpretation:**

* **Intercept (β0​=5.0):** If the company spends **$0** on TV advertising, the predicted baseline sales are **5,000 units**.
* **Coefficient (β1​=0.045):** For every **$1,000** increase in TV advertising spend, the predicted sales increase by **0.045 thousand** (i.e., **45 units**).

### Example 2: Sales Trend Forecasting (Multiple Linear Regression)

**Scenario:** Predicting sales based on both price and competitor activity.

| Variable | Role |
| --- | --- |
| **Sales** (y^​) | Predicted Sales Volume |
| **Price** (x1​) | Price of the Product |
| **Competitor Price** (x2​) | Average Price of Competitor Products |

Learned Equation Example:

Sales=1000−5.5×Price+2.0×Competitor Price

**Interpretation:**

* **Price Coefficient (β1​=−5.5):** Holding the competitor's price constant, every $1 increase in the product's price is associated with a **decrease of 5.5 sales units**. (A negative relationship, as expected).
* **Competitor Price Coefficient (β2​=2.0):** Holding the product's price constant, every $1 increase in the competitor's price is associated with an **increase of 2.0 sales units**. (A positive relationship, as customers switch to your product).

## Python Practical Example (Using Scikit-learn)

This example predicts sales based on a simplified "Advertising Spend" feature.

Python

import numpy as np  
from sklearn.linear\_model import LinearRegression  
  
# 1. Data Setup (Independent variable X, Dependent variable y)  
# Advertising Spend (in $1000s)  
X = np.array([10, 20, 30, 40, 50, 60]).reshape(-1, 1)   
# Sales (in $1000s)  
y = np.array([25, 45, 65, 80, 105, 120])   
  
# 2. Create and Train the Model  
model = LinearRegression()  
model.fit(X, y)  
  
# 3. Get Coefficients  
intercept = model.intercept\_  
slope = model.coef\_[0]  
  
print(f"Intercept (β0): {intercept:.2f}")  
print(f"Slope (β1): {slope:.2f}")  
# The resulting equation is approximately: Sales = 5.00 + 1.95 \* Ad\_Spend  
  
# 4. Make a Prediction  
# Predict sales for a new advertising spend of $70,000  
new\_ad\_spend = np.array([[70]])  
predicted\_sales = model.predict(new\_ad\_spend)  
  
print(f"\nPredicted Sales for $70k Ad Spend: ${predicted\_sales[0]:.2f}k")

**Output Interpretation:**

* The **Slope (β1​=1.95)** means that for every additional **$1,000** spent on advertising, the predicted sales increase by **$1,950**.
* The **Intercept (β0​=5.00)** means if $0 is spent on advertising, the predicted sales are **$5,000**.