Linear Regression on Boston Housing Dataset: From Scratch

ATharv Bhatt

October 3, 2025

1 Introduction

Linear regression is a fundamental algorithm for predicting continuous numerical outcomes. Here, we implement it from scratch to predict house prices (`medv`) using the number of rooms (`rm`) from the Boston Housing dataset.

2 Data Preprocessing

The dataset contains information on Boston houses:

- `rm` – Average number of rooms (feature)
- `medv` – Median house price (target)

We extract these columns:

```
X = data[’rm’].values
y = data[’medv’].values
```

—

3 Model Implementation

The linear regression model predicts the target as:

$$\hat{y} = mx + b$$

Where m is the slope and b is the intercept.
The loss function (Mean Squared Error) is:

$$L(m, b) = \frac{1}{n} \sum_{i=1}^{n} \big(y_i - (mx_i + b)\big)^2$$

Gradient descent updates the parameters iteratively:

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

$$\frac{\partial L}{\partial m} = -\frac{2}{n} \sum_{i=1}^{n} x_i (y_i - (m x_i + b)), \quad \frac{\partial L}{\partial b} = -\frac{2}{n} \sum_{i=1}^{n} (y_i - (m x_i + b))$$

—

4 Training

Weights are initialized as $m = 0, b = 0$, and updated over 1000 epochs with a learning rate of 0.0001:

```
m = 0
b = 0
learning_rate = 0.0001
epochs = 1000

for i in range(epochs):
    m, b = gradient_descend(m, b, X, y, data, learning_rate)
```

After training, the model parameters are printed:

```
print(m, b)
```

—

5 Conclusion

This project demonstrates:

- Simple linear regression for continuous target prediction
- Mean Squared Error loss function
- Gradient descent optimization

The model effectively captures the relationship between the number of rooms and house prices in the Boston Housing dataset.