Gradient Descent Optimization for Linear Regression

Introduction

This project demonstrates gradient descent for optimizing the weights of a simple linear regression equation:

ypred=x1w1+x2w2y_{\text{pred}} = x_1 w_1 + x_2 w_2ypred=x1w1+x2w2

where x1x_1x1 and x2x_2x2 are input features, and w1w_1w1 and w2w_2w2 are their respective weights. The goal is to minimize the squared error between the predicted output ypredy_{\text{pred}}ypred and the actual value yactualy_{\text{actual}}yactual.

Key Concepts

1. **Activation Functions: Defined but not directly used in this case.**
2. **Weight Initialization: Weights w1w_1w1 and w2w_2w2 are randomly initialized.**
3. **Loss Function: Mean Squared Error (MSE).**
4. **Gradient Descent: Updates weights to reduce error over multiple epochs.**

Training Process

- **Forward Pass: Compute predicted output using the linear equation.**
- **Error Calculation: Compute squared error.**
- **Gradient Calculation: Use partial derivatives to compute weight adjustments.**
- **Weight Update: Update weights using the gradient descent formula:**

wi=wi−η·∂E∂wiw_i = w_i - \eta \cdot \frac{\partial E}{\partial w_i}wi=wi−η·∂wi∂E

where η\etaη is the learning rate.

- **Repeat for 20 epochs to observe error reduction.**

Results & Visualization

Two key plots help visualize the optimization process:

1. **Error Reduction: Shows how the error decreases over epochs.**
2. **Prediction Trend: Shows how predicted values approach the actual target value over epochs.**

Conclusion

This experiment demonstrates how gradient descent effectively minimizes error in a simple linear regression model. By tuning parameters such as learning rate and number of epochs, optimization can be improved further.