Name: Ali Haider

Roll No: 21I-1522

Section: CS-A

Stochastic Gradient Descent (SGD) for Linear Regression

GitHub Repository: [https://github.com/alihaider-debug/SGD-in-ML]

Introduction

This report presents a comparative analysis of three Gradient Descent (GD) techniques -
Stochastic Gradient Descent (SGD), Batch Gradient Descent (BGD), and Mini-Batch Gradient Descent (MBGD) - for training a simple linear regression model. The impact of learning rate and momentum on convergence speed and model parameters is also examined.

Methodology

A synthetic dataset was created where y = 2x, and the three GD variants were implemented using Python.
The model updates weights based on the Mean Squared Error (MSE) loss function.
The comparison was conducted by analyzing convergence behavior and final model parameters.

Gradient Descent Variants

- Batch Gradient Descent (BGD): Updates weights after processing the entire dataset.
- Stochastic Gradient Descent (SGD): Updates weights after each data point, making it noisy but fast.
- Mini-Batch Gradient Descent (MBGD): Updates weights in small batches, offering a balance between stability and speed.

Observations & Results

1. Convergence Speed & Stability

- BGD took longer to converge but provided a smooth and stable descent.
- SGD was the fastest but exhibited fluctuations due to variance in weight updates.
- MBGD achieved a good trade-off between speed and stability.

2. Effect of Learning Rate

- A high learning rate (0.1) led to divergence (oscillations, failure to converge).
- A low learning rate (0.001) resulted in slow convergence.
- Optimal learning rate (0.01) balanced speed and stability.

3. Effect of Momentum

- Higher momentum (0.9) accelerated convergence and reduced fluctuations.
- No momentum (0.0) resulted in slower convergence and higher variance.
- Optimal momentum (0.8) provided a smooth convergence path.

Challenges Faced

- Finding an optimal learning rate required trial and error.
- SGD's instability led to difficulty in determining the stopping criteria.
- Computational overhead in large datasets for MBGD tuning.

Conclusion

From the analysis:

- Mini-batch Gradient Descent (MBGD) is the best choice for practical applications as it offers a balance between speedand stability.
- Hyperparameter tuning (learning rate & momentum) significantly impacts convergence.
- Momentum helps in overcoming oscillations and speeds up training.