NAME: MUSTAPHA YUSUF

MAT NO: 20CJ027456

CEN 520

**Introduction**

In this practical session, I applied key concepts from supervised machine learning to both synthetic and real-world datasets. The synthetic dataset modeled a relationship between “Projects Tracked and Closed” and “Efficiency Score.” Using NumPy, pandas, and Matplotlib, I built a simple linear regression model optimized with gradient descent.

After normalizing the input feature to zero mean and unit variance, I implemented gradient descent and observed the error decreasing consistently over 100 iterations. The model achieved a final Mean Squared Error (MSE) of approximately 4.3 and a Mean Absolute Error (MAE) of 1.73, indicating a decent fit for the synthetic internship dataset.

Visualization showed how the model learned to approximate the data, with a red regression line fitting well over the actual data points. The MSE plot showed a smooth descent, confirming the model converged properly with the learning rate of 0.01. When trying other values like 0.001 or 0.1, I observed that too small a learning rate slowed convergence, while too high a rate sometimes caused the loss to diverge.

A major challenge was choosing the correct cost function and tuning the learning rate. MSE gave smooth gradients but was sensitive to outliers, while MAE provided robustness but more complex optimization. Handling real-world datasets like changelog.csv required cleaning and identifying appropriate numeric features and targets. Normalization was critical to ensure consistent gradient steps across features.

**Challenges**

One key challenge was handling inconsistent or noisy real-world data. The changelog.csv file required cleaning and careful selection of feature and target columns. Additionally, determining an appropriate learning rate was critical. Too high, and the model diverged; too low, and convergence was very slow. I addressed this by experimenting with several learning rates (0.001, 0.01, 0.1) and plotting the MSE over iterations to visually assess convergence behavior.

Answers to Questions

* How does normalization affect the feature values?

Normalization centers the data around zero and scales it to have unit variance, making the gradient descent more stable and efficient.

* Why does MSE penalize larger errors more than MAE?

MSE squares the errors, which amplifies larger deviations more than MAE, which uses absolute values and treats all errors equally.

* How does the learning rate affect convergence?

A high learning rate can cause the model to overshoot and fail to converge, while a low rate results in slow learning. Optimal rates yield faster and smoother convergence.

* Why might the model perform differently on real vs. synthetic data?

Real-world data is noisier, more complex, and often includes outliers or nonlinear relationships. Synthetic data is usually cleaner and designed to follow a specific pattern.

* How does the choice of cost function (MSE vs. MAE) affect optimization?

MSE (Mean Squared Error) penalizes large errors more because it squares them, making it sensitive to outliers. This helps gradient descent converge to a minimum that prioritizes minimizing large deviations. MAE (Mean Absolute Error), on the other hand, treats all errors equally and is more robust to outliers. However, MAE leads to non-differentiable points (at zero), which can complicate optimization using gradient-based methods.

* How does gradient descent compare to scikit-learn's built-in linear regression?

scikit-learn's LinearRegression uses the closed-form Ordinary Least Squares (OLS) solution, which is mathematically exact and fast for small-to-medium datasets. Gradient descent, on the other hand, is an iterative method that's slower but can scale to massive datasets or be used in situations where OLS is infeasible (e.g., with online learning or very large feature sets).