

Multi-Variable Calculus

|  |  |
| --- | --- |
| Assignment 1 | Applications of Multi-Variable Calculus in Computer Science |
| Name | Muhammad Abdullah |
| Sap ID | 53457 |
| Section | BSCS-3B |
| Instructor | Dr. Maria Akram |

# Optimization in Machine Learning:

Optimization in machine learning is a process of finding the best model parameters by minimizing or maximizing objective functions, such as loss functions. A common objective in supervised learning is to minimize a loss function, which measures the discrepancy between predictions and true labels. For example, in linear regression, the loss function is the Mean Squared Error (MSE):

[1]

where:

* Yi​ is the true label for the iii-th data point,
* ​ is the predicted value for the iii-th data point,
* N is the total number of data points,
* w and b are the parameters of the model (weights and bias).

To minimize the loss function, Gradient Descent, a widely used optimization technique, is employed. It updates the parameters iteratively using the gradients of the loss function. The equation of loss function is given as,

[1]

[1]

where:

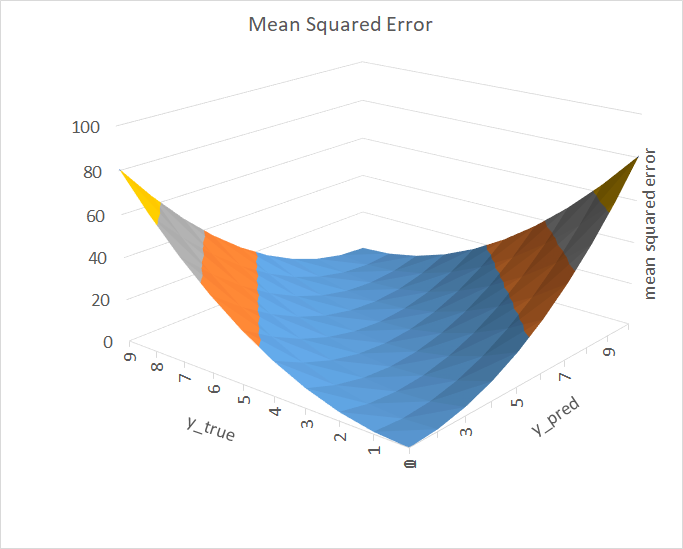
* ,​ are the gradients of the loss function for w and b,
* η is the learning rate, which controls the step size.

The gradients for the MSE loss function are calculated as follows:

[1]

[1]

The following figure shows the graph for mean squared error with respect to y\_true and y\_predict.



Here is the code for above graph,

|  |
| --- |
| **def mean\_squared\_error(actual, predicted):**  **sum\_square\_error = 0.0**  **for i in range(len(actual)):**  **sum\_square\_error += (actual[i] - predicted[i]) \*\* 2.0**  **mean\_square\_error = 1.0 / len(actual) \* sum\_square\_error**  **return mean\_square\_error**    **y\_true = [3, -0.5, 2, 7]**  **y\_pred = [2.5, 0.0, 2, 8]**    **mse1 = mean\_squared\_error(y\_true, y\_pred)**  **print(f"MSE = {mse1}")**  **# MSE = 0.375**      **from sklearn.metrics import mean\_squared\_error**  **mse2 = mean\_squared\_error(y\_true, y\_pred)**  **print(f"MSE = {mse2}")**  **# MSE = 0.375** |

Here, xi​ is the input feature for the i-th data point. By iteratively updating w and b, the loss function L(w,b) is minimized, leading to improved model predictions. Advanced optimization methods such as Stochastic Gradient Descent (SGD), Adam, and RMSprop refine these principles to handle large datasets, adaptive learning rates, and momentum for faster convergence. Visualizations, such as contour plots of the loss surface, illustrate the trajectory of optimization algorithms as they converge towards minima.

# References

[1] Deep Learning by Ian Goodfellow, Yoshua Bengio, and Aaron Courville.