**Difference between Cost function vs Loss function**

When calculating loss we consider only a single data point, then we use the term loss function.

Whereas, when calculating the sum of error for multiple data then we use the cost function. There is no major difference.

In other words, the loss function is to capture the difference between the actual and predicted values for a single record whereas cost functions aggregate the difference for the entire training dataset.

The Most commonly used loss functions are Mean-squared error and Hinge loss.

Mean-Squared Error(MSE): In simple words, we can say how our model predicted values against the actual values.

MSE = √(predicted value - actual value)2  
Hinge loss: It is used to train the machine learning classifier, which is

L(y) = max(0,1- yy)

Where y = -1 or 1 indicating two classes and y represents the output form of the classifier. The most common cost function represents the total cost as the sum of the fixed costs and the variable costs in the equation y = mx + b

**Cost function:**

In ML, cost functions are used to estimate how badly models are performing. Put simply, **a cost function is a measure of how wrong the model is in terms of its ability to estimate the relationship between X and y.** This is typically expressed as a difference or distance between the predicted value and the actual value. The cost function (you may also see this referred to as loss or error.) can be estimated by iteratively running the model to compare estimated predictions against “ground truth” — the known values of y.

The objective of a ML model, therefore, is to find parameters, weights or a structure that **minimises**the cost function.

**Minimizing the cost function: Gradient descent**

we know that models learn by minimizing a cost function, you may naturally wonder how the cost function is minimized — enter **gradient descent**. Gradient descent is an efficient optimization algorithm that attempts to find a local or global minima of a function.

Gradient descent enables a model to learn the gradient or *direction* that the model should take in order to reduce errors (differences between actual y and predicted y). Direction in the simple linear regression example refers to how the model parameters b0 and b1 should be tweaked or corrected to further reduce the cost function. As the model iterates, it gradually converges towards a minimum where further tweaks to the parameters produce little or zero changes in the loss — also referred to as convergence.

 

