OVERFITTING AND UNDERFITTING IN MACHINE LEARNING

Overfitting is the case where the overall cost is really small, but the generalization of the model is unreliable. This is due to the model learning “too much” from the training data set.The more we leave the model training the higher the chance of overfitting occurring. **We always want to find the trend, not fit the line to all the data points.** Overfitting (or high variance) leads to more bad than good.**Overfitting** occurs when a statistical model or machine learning algorithm **captures the noise** of the data. Intuitively, overfitting occurs when the model or the algorithm fits the data too well. Specifically, overfitting occurs if the model or algorithm shows **low bias** but **high variance**. Overfitting is often a result of an excessively complicated model, and it can be prevented by fitting multiple models and using validation or cross-validation to compare their predictive accuracies on test data.
To solve the problem of overfitting in our model we use regularization technique.They are three types of regularisation techniques to overcome overfitting.
1.L1 regularisation
2.L2 regularization
3.Elastic net

Underfitting is the case where the model has “ not learned enough” from the training data, resulting in low generalization and unreliable predictions.As you probably expected, underfitting (i.e. high bias) is just as bad for generalization of the model as overfitting. In high bias, the model might not have enough flexibility in terms of line fitting, resulting in a simplistic line that does not generalize well.**Underfitting** occurs when a statistical model or machine learning algorithm **cannot capture the underlying trend** of the data. Intuitively, underfitting occurs when the model or the algorithm does not fit the data well enough. Specifically, underfitting occurs if the model or algorithm shows **low variance** but **high bias**. Underfitting is often a result of an excessively simple model.
In order to overcome underfitting we have to model the expected value of target variable as nth degree polynomial yielding the general Polynomial.The training error will tend to **decrease** as we increase the degree d of the polynomial.

The mathematical presentation can be give as,

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$$

and

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \beta_{12} x_1 x_2 + \varepsilon$$

The training error will tend to **decrease** as we increase the degree d of the polynomial.At the same time, the cross validation error will tend to **decrease** as we increase d up to a point, and then it will **increase** as d is increased, forming a convex curve.