REGULARIZATION IN ML: BALANCING COMPLEXITY & GENERALIZATION

Swastik Dewan
School of Physical Sciences, NISER Bhubaneswar
swastik.dewan@niser.ac.in

Introduction

In the vast landscape of machine learning, overfitting lurks like a treacherous mirage. Imagine a model that memorizes every twist and turn of its training data, only to falter miserably when faced with new, unseen examples. Regularization steps in as our trusty guide, leading us away from the perilous cliffs of overfitting and toward a more reliable model.

Why Regularization matters

- **Complexity Control**
- **Overfitting Prevention**
- **Bias-Variance Tradeoff**
- **Feature Selection**
- **Multicollinearity Management**

Understanding Overfitting and Underfitting

Overfitting is a phenomenon that occurs when a Machine Learning model is constrained to the training set and not able to perform well on unseen data. That is when our model learns the noise in the training data as well. This is the case when our model memorizes the training data instead of learning the patterns in it.

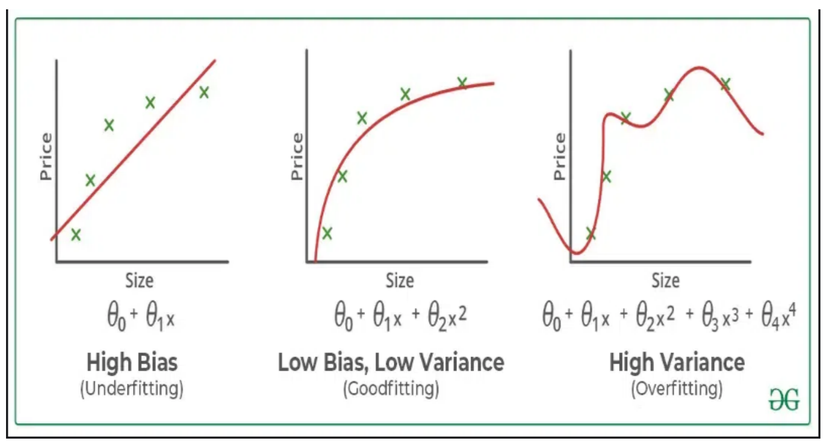

Figure 1: Graphical representation of different fittings

Underfitting on the other hand is the case when our model is not able to learn even the basic patterns available in the dataset. In the case of the underfitting model is unable to perform well even on the training data hence we cannot expect it to perform well on the validation data. This is the case when we are supposed to increase the complexity of the model or add more features to the feature set.

Bias-Variance Tradeoff

Bias measures the average difference between predicted values and true values. As bias increases, a model predicts less accurately on a training dataset. High bias refers to high error in training.

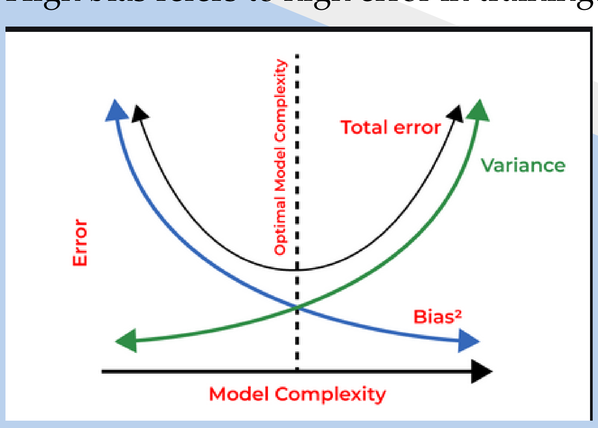

Figure 2: The bias-variance tradeoff demonstrates the inverse relationship between bias and variance.

Variance measures the difference between predictions across various realizations of a given model. As variance increases, a model predicts less accurately on unseen data. High variance refers to high error during testing and validation.

- **High Bias, Low Variance:** Underfitting
- **High Variance, Low Bias:** Overfitting
- **High-Bias, High-Variance:** The model cannot capture underlying patterns and is too sensitive to training data changes. On average, the model will generate unreliable and inconsistent predictions.
- **Low Bias, Low Variance:** Perfect scenario for a machine learning model where it can generalize well to unseen data and make consistent, accurate predictions.

Types of Regularization

1. **Lasso regression (or L1 regularization)** is a regularization technique that penalizes high-value, correlated coefficients. It introduces a regularization term (also called, penalty term) into the model's sum of squared errors (SSE) loss function. This penalty term is the absolute value of the sum of coefficients. Controlled in turn by the hyperparameter lambda (λ), it reduces select feature weights to zero. Lasso regression thereby removes multicollinear features from the model altogether.

$$Cost = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y}_i)^2 + \lambda \sum_{i=1}^{M} |w_i|$$

2. **Ridge regression (or L2 regularization)** is regularization technique that similarly penalizes high-value coefficients by introducing a penalty term in the SSE loss function. It differs from lasso regression however. First, the penalty term in ridge regression is the squared sum of coefficients rather than the absolute value of coefficients. Second, ridge regression does not enact feature selection. While lasso regression's penalty term can remove features from the model by shrinking coefficient values to zero, ridge regression only shrinks feature weights towards zero but never to zero.

$$Cost = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y}_i)^2 + \lambda \sum_{i=1}^{M} w_i^2$$

3. **Elastic net regularization** essentially combines both ridge and lasso regression but inserting both the L1 and L2 penalty terms into the SSE loss function. Elastic net inserts both of the penalty values into the cost function (SSE) equation. In this way, elastic net addresses multicollinearity while also enabling feature selection. α is a hyperparameter that controls the ratio of the L1 and L2 regularization.

$$Cost = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y}_i)^2 + \lambda((1-\alpha)\sum_{i=1}^{M} |w_i| + \alpha \sum_{i=1}^{M} w_i^2)$$

Case Study: Applying Dropout to CNNs

Figure 3: Applying dropout to prevent neural networks from overfitting (a) standard neutral net and (b) after applying dropout

The dropout is used to randomly "turn off" or "ignore" neurons (with a predefined probability, often every other neuron) as well as peripheral neurons and the other neurons will have to step in and handle the representation required to make predictions for the missing neurons. Thus, with fewer neurons, the network is more responsive and can learn faster. At the end of the learning session, the "turned off" neurons are "turned back on" (with their original weights).

Figure 4: Training and validation accuracy

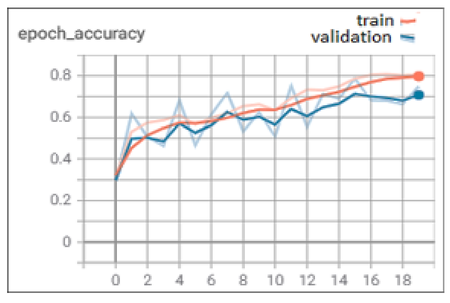

Figure 5: Training and validation accuracy after dropout

References

1. https://www.researchgate.net/publication/355949412_Dropout_a_basic_and_effective_regularization_method_for_a_deep_learning_model_A_case_study
2. https://www.ibm.com/topics/regularization
3. https://www.geeksforgeeks.org/regularization-in-machine-learn