(https://google.com/racialequity?authuser=0)

Regularization for Simplicity: L_2 Regularization

ated Time: 7 minutes

Consider the following **generalization curve**, which shows the loss for both the training set and validation set against the number of training iterations.

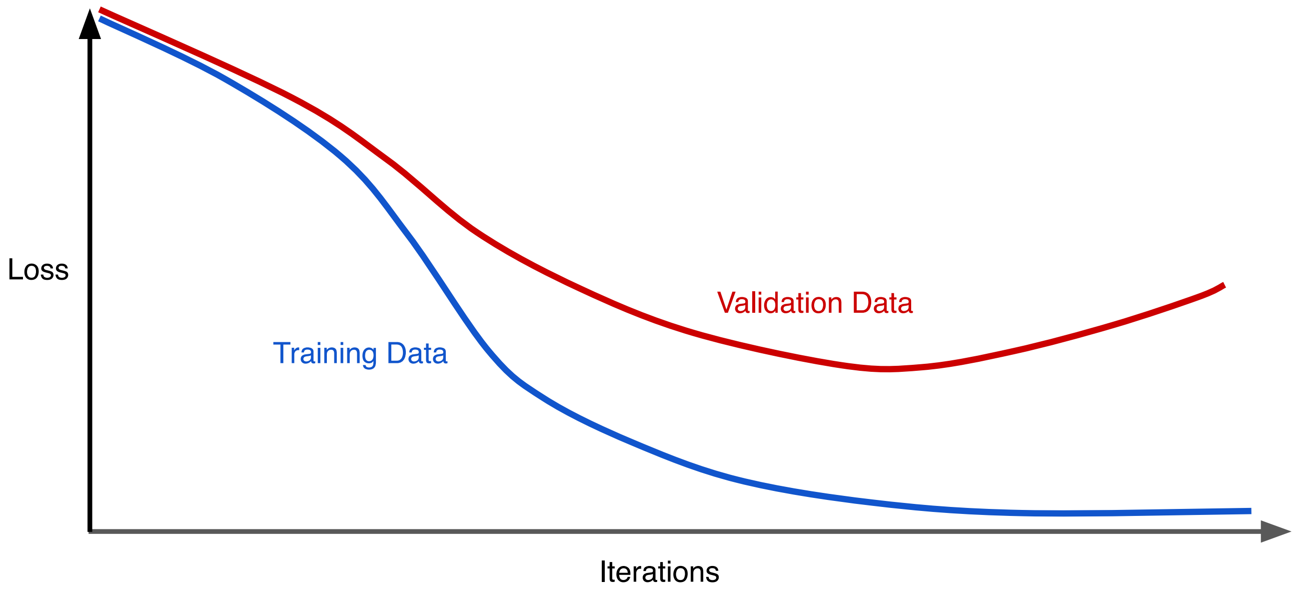

Figure 1. Loss on training set and validation set.

Figure 1 shows a model in which training loss gradually decreases, but validation loss eventually goes up. In other words, this generalization curve shows that the model is overfitting (https://developers.google.com/machine-learning/crash-course/generalization/peril-of-overfitting?authuser=0) to the data in the training set. Channeling our inner Ockham (https://developers.google.com/machine-learning/crash-course/generalization/peril-of-overfitting?authuser=0#ockham) , perhaps we could prevent overfitting by penalizing complex models, a principle called **regularization**.

In other words, instead of simply aiming to minimize loss (empirical risk minimization):

$$\text{minimize(Loss(Data|Model))}$$

we'll now minimize loss+complexity, which is called **structural risk minimization**:

$$\text{minimize(Loss(Data|Model)} + \text{complexity(Model))}$$

Our training optimization algorithm is now a function of two terms: the **loss term**, which measures how well the model fits the data, and the **regularization term**, which measures model complexity.

Machine Learning Crash Course focuses on two common (and somewhat related) ways to think of model complexity:

- Model complexity as a function of the *weights* of all the features in the model.
- Model complexity as a function of the *total number of features* with nonzero weights. (A later module (https://developers.google.com/machine-learning/crash-course/regularization-for-sparsity/l1-regularization?authuser=0) covers this approach.)

If model complexity is a function of weights, a feature weight with a high absolute value is more complex than a feature weight with a low absolute value.

We can quantify complexity using the ***L_2* regularization** formula, which defines the regularization term as the sum of the squares of all the feature weights:

$$L_2\text{ regularization term} = ||\boldsymbol{w}||_2^2 = w_1^2 + w_2^2 + \ldots + w_n^2$$

In this formula, weights close to zero have little effect on model complexity, while outlier weights can have a huge impact.

For example, a linear model with the following weights:

$$\{w_1 = 0.2, w_2 = 0.5, w_3 = 5, w_4 = 1, w_5 = 0.25, w_6 = 0.75\}$$

Has an L_2 regularization term of 26.915:

$$w_1^2 + w_2^2 + \boldsymbol{w_3^2} + w_4^2 + w_5^2 + w_6^2$$

$$= 0.2^2 + 0.5^2 + \mathbf{5^2} + 1^2 + 0.25^2 + 0.75^2$$

$$= 0.04 + 0.25 + \mathbf{25} + 1 + 0.0625 + 0.5625$$

$$= 26.915$$

But w_3 (bolded above), with a squared value of 25, contributes nearly all the complexity. The sum of the squares of all five other weights adds just 1.915 to the L_2 regularization term.

:rms

- neralization curve ://developers.google.com/machine-ıg/glossary?authuser=0#generalization_curve)
- erfitting ://developers.google.com/machine-ıg/glossary?authuser=0#overfitting)
- ɪuctural risk minimization ://developers.google.com/machine-ıg/glossary?authuser=0#SRM)
- L_2 regularization (https://developers.google.com/machine-learning/glossary?authuser=0#L2_regularization)
- regularization (https://developers.google.com/machine-learning/glossary?authuser=0#regularization)

Help Center (https://support.google.com/machinelearningeducation?authuser=0)

Previous
← Video Lecture (https://developers.google.com/machine-learning/crash-course/regularization-for-simplicity/video-lecture?authuser=0)

Next
Lambda → (https://developers.google.com/machine-learning/crash-course/regularization-for-simplicity/lambda?authuser=0)