**Regularization**

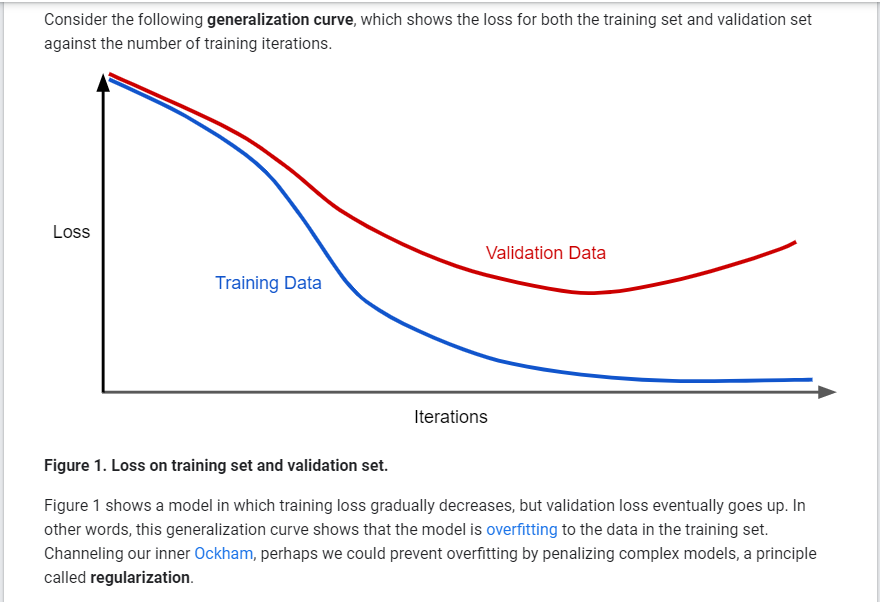
* How to define complexity(Model)?
* Prefer smaller weights
* Diverging from this should incur a cost
* Can encode this idea via L2 **regularization** (a.k.a. ridge)
  + *complexity(model) = sum of the squares of the weights*
  + Penalizes really big weights
  + For linear models: prefers flatter slopes
  + Bayesian prior:
    - weights should be centered around zero
    - weights should be normally distributed

## A Loss Function with L2 Regularization

Loss(Data|Model)+λ(w12+…+wn2)

Where:  
  
Loss: Aims for low training error λ: Scalar value that controls how weights are balanced w12+…+wn2: Square ofL2norm

# Regularization for Simplicity: L₂ Regularization



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

minimize(Loss(Data|Model))

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

minimize(Loss(Data|Model) + 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.

# Regularization for Simplicity: Lambda

Model developers tune the overall impact of the regularization term by multiplying its value by a scalar known as **lambda** (also called the **regularization rate**). That is, model developers aim to do the following:

minimize(Loss(Data|Model)+λ complexity(Model))

Performing *L2* regularization has the following effect on a model

* Encourages weight values toward 0 (but not exactly 0)
* Encourages the mean of the weights toward 0, with a normal (bell-shaped or Gaussian) distribution.

