Optimization and Regularization

https://github.com/fchollet/deep-learning-with-python-notebooks/blob/master/4.4-overfitting-and-underfitting.ipynb

https://cs231n.github.io/classification/

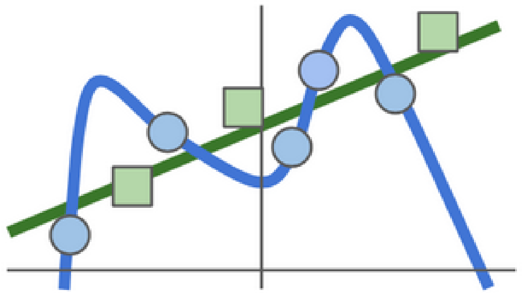
Generic formula for Optimization:

- Data loss + Regularization

$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}_{\text{Data loss: Model predictions should match training data}} + \underbrace{\lambda R(W)}_{\text{Regularization: Model should be “simple”, so it works on test data}}$$

Data loss: Model predictions should match training data

Regularization: Model should be “simple”, so it works on test data



Basic intuition:

A "simple model" in the context of regularization is a model where the distribution of parameter values has less entropy or a model with fewer parameters altogether.

Thus a common way to mitigate overfitting is to put constraints on the complexity of a network by forcing its weights to only take small values, which makes the distribution of weight values more "regular".

This is called "weight regularization", and it is done by adding to the loss function of the network a cost associated with having large weights.

L1 and L2 norm regularization:

- L1 regularization: the cost added is proportional to the absolute value of the weights coefficients (i.e. to what is called the "L1 norm" of the weights).

- L2 regularization: the cost added is proportional to the square of the value of the weights coefficients (i.e. to what is called the "L2 norm" of the weights).

$$||\mathbf{w}||_1 = |w_1| + |w_2| + ... + |w_N|$$

1-norm (also known as L1 norm)

$$\|\mathbf{w}\|_2 = \left(|w_1|^2 + |w_2|^2 + \ldots + |w_N|^2\right)^{\frac{1}{2}}$$

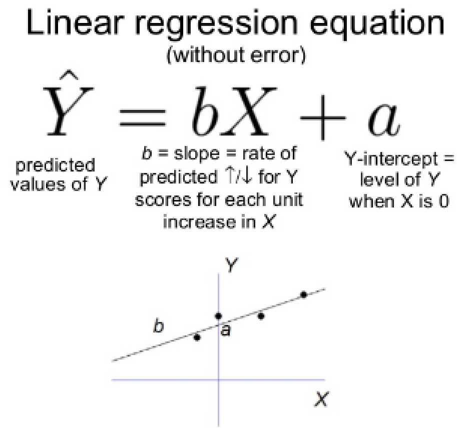
2-norm (also known as L2 norm or Euclidean norm)

$$\|\mathbf{w}\|_p = (|w_1|^p + |w_2|^p + \ldots + |w_N|^p)^{\frac{1}{p}}$$

p-norm

Example: Linear regression

Single variable (X):
#features = 1


Linear regression equation
(without error)

$$\hat{Y} = bX + a$$

predicted values of Y
b = slope = rate of predicted ↑/↓ for Y scores for each unit increase in X
Y-intercept = level of Y when X is 0
Y
b
a
X

Multiple variables (x1, … , x_n):
#features = n

$$\hat{y} = w_1x_1 + w_2x_2 + \ldots + w_Nx_N + b$$

Linear regression with L1 and L2 regularization (lasso regression):

L1 Regularization

$$\text{Cost} = \sum_{i=0}^{N}\left(y_i - \sum_{j=0}^{M} x_{ij}W_j\right)^2 + \lambda \sum_{j=0}^{M} |W_j|$$

L2 Regularization

$$\text{Cost} = \underbrace{\sum_{i=0}^{N}\left(y_i - \sum_{j=0}^{M} x_{ij}W_j\right)^2}_{\text{Loss function}} + \underbrace{\lambda \sum_{j=0}^{M} W_j^2}_{\text{Regularization Term}}$$

Overview:

- There's two ways to do regularization
  - First thing is you can constrain your model class to just not contain the more powerful, more complex models
  - Second thing is you can add this soft penalty where the model still has access to more complex models.
- In plain words, you add this soft constraint saying that "if you want to use these more complex models, you need to overcome this penalty for using their complexity!".

$$x = [1, 1, 1, 1]$$

$$w_1 = [1, 0, 0, 0]$$
$$w_2 = [0.25, 0.25, 0.25, 0.25]$$

- L2 regularization = Weight decay = Euclidean norm of weight vector W = Squared norm = Penalizing euclidean norm of weight vector (ex. w_2)
  - 1) Prefer to spread that influence across all the different values in x
  - and 2) Depend on the entire x vector rather than depending only on certain elements of the x vector
  - The better the element of W spreads, the loss complex
- L1 regularization = Penalizing L1 norm of weight vector = Encouraging sparsity in matrix W (ex. w_1)
  - measure model complexity by the number of zeros in the weight vector
  - The more zeros, the less complex