Deep Learning Seminar

Chapter 7. Regularization for Deep Learning

Keonwoo Noh

Department of Information and Communication Engineering

DGIST

2017.08.04

Chapter 7. Regularization for Deep Learning

- **Part 1**
  - 7.1 Parameter Norm Penalties
  - 7.2 Norm Penalties as Constrained Optimization
  - 7.3 Regularization and Under-Constrained Problems
  - 7.4 Dataset Augmentation
  - 7.5 Noise Robustness
  - 7.6 Semi-Supervised Learning
  - 7.7 Multitask Learning
  - 7.8 Early stopping

Chapter 7. Regularization for Deep Learning

- Part 2
  - 7.9 Parameter Tying and Parameter Sharing
  - 7.10 Sparse Representation
  - 7.11 Bagging and Other Ensemble Method
  - 7.12 Dropout
  - 7.13 Adversarial Training
  - 7.14 Tangent Distance, Tangent Prop and Manifold Tangent Classifier

Introduction to Regularization

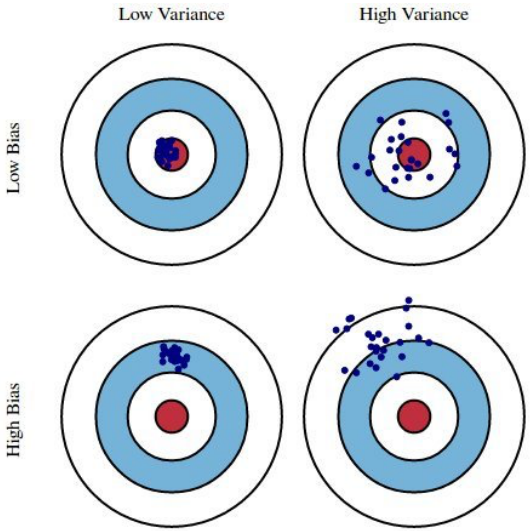
Definition and objective

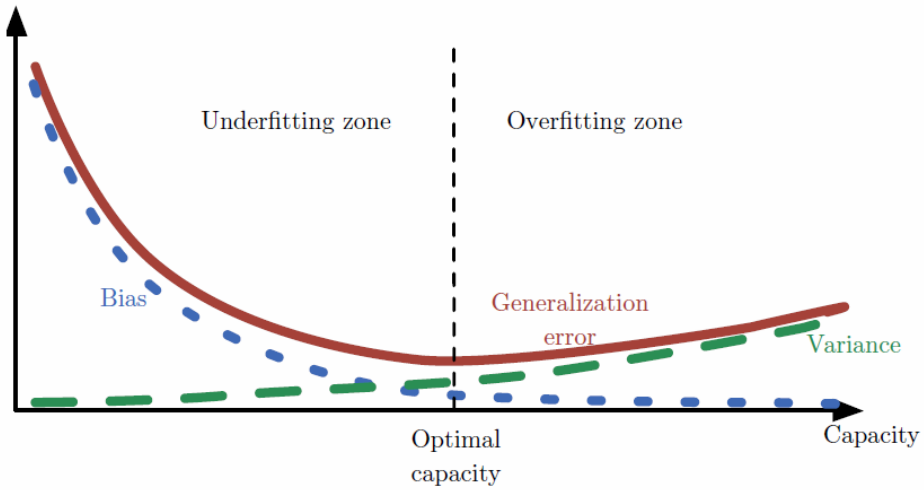
- **Any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error**

- **The goal of Regularization is to prevent overfitting by imposing some strategies such as :**
  - Put extra constraints on a machine learning model
  - Add extra terms in the objective function
  - Impose ensemble method

Bias variance trade off

- **Regularization of an estimator works by trading increased bias for reduced variance**



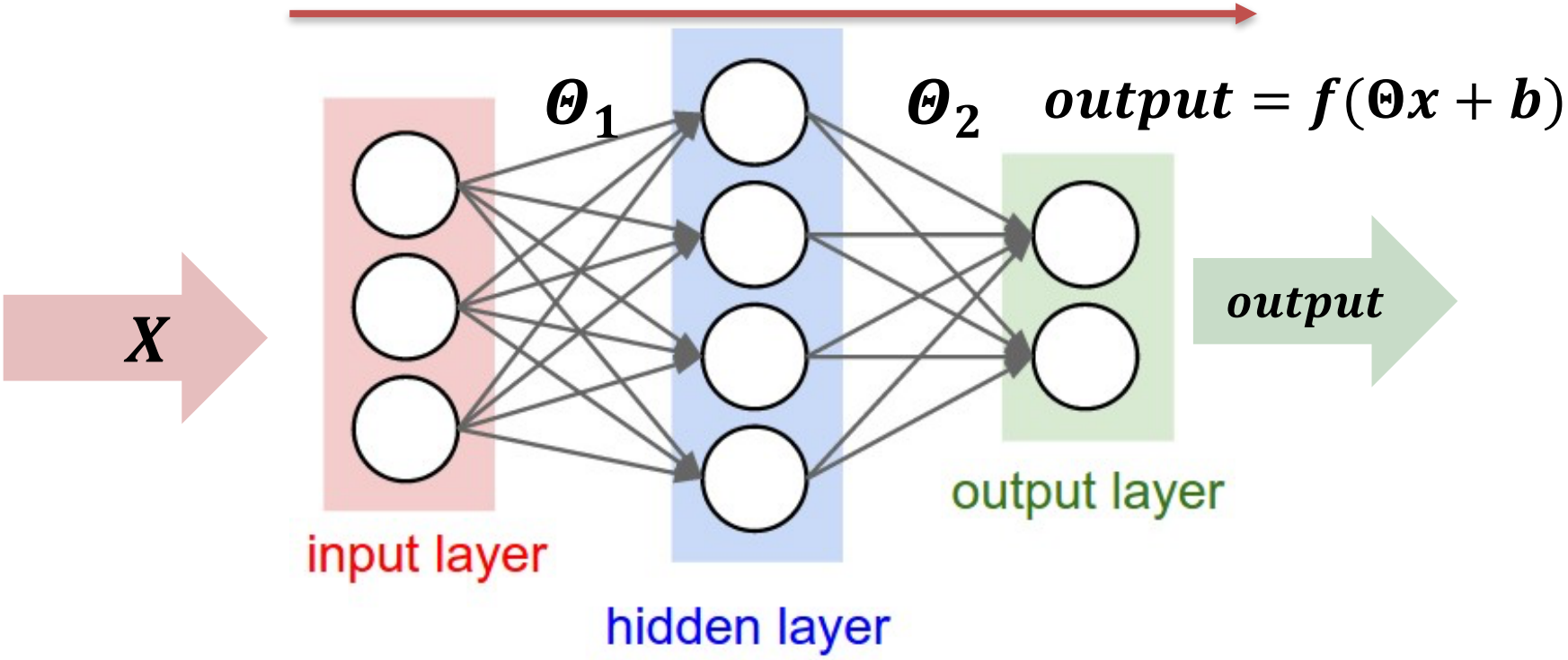


Source : http://www.kdnuggets.com/2016/08/bias-variance-tradeoff-overview.html

Parameter Norm Penalties

Supervised learning (Inference)

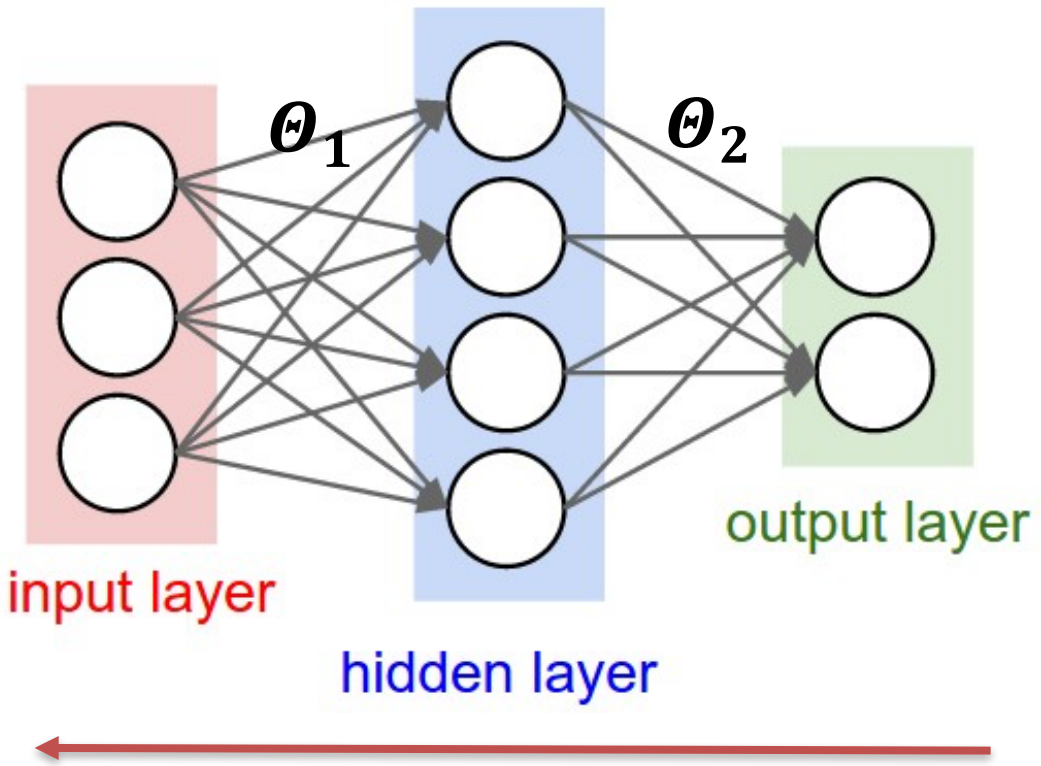
- Predict (Label/Dependent variable) given data X



source : http://cs231n.github.io/neural-networks-1/

Supervised learning (Learning)

- Update θ through gradient of objective function ∇J



$$\nabla J(\Theta; X, y)$$

+ Norm penalty

source : http://cs231n.github.io/neural-networks-1/

Norms (Definition)

- A way to measure the size of vector

$$||x||_p = (\sum_i |x_i|^p)^{\frac{1}{p}}$$

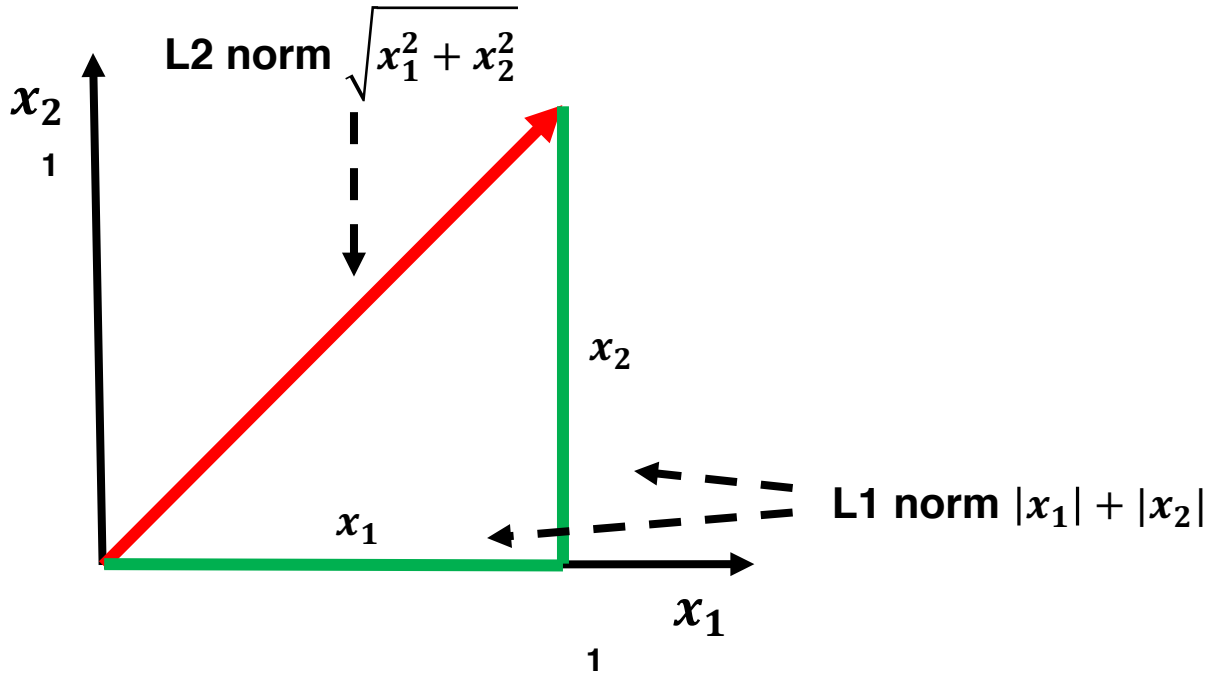
- Thus, L1 norm and L2 norm are respectively :

$$||x||_1 = \sum_i |x_i|$$

$$||x||_2 = \sqrt{\sum_i |x_i|^2}$$

Norms (Example)

- $x^T = [1, 1]$



$$||x||_1 = \sum_i |x_i| = 2$$

$$||x||_2 = \sqrt{\sum_i |x_i|^2} = \sqrt{2}$$

Squared L2 Norm

- Squared L2 norm is used instead of original L2 norm for regularization in machine learning task

- All of the derivatives of the L2 norm depend on the entire vector

- The derivatives of the squared L2 norm with respect to each element of x each depend only on the corresponding element of x

Squared L2 Norm

- $x^T = [x_1, x_2]$
- **L2 norm**

$$f(x) = \sqrt{x_1^2 + x_2^2}$$

$$f'(x) = x_1(x_1^2 + x_2^2)^{-\frac{1}{2}}$$

- **Squared L2 norm**

$$f(x) = x_1^2 + x_2^2$$

$$f'(x) = 2x_1$$

Norms (Matrix norm)

- L1 norm and L2 norm are defined for the way to measure size of vector

- Sometimes machine learning task would require a size of matrix as well

- Frobenius norm is :
  - A way to measure the size of matrix
  - Known as ‘Matrix norm’

$$||A||_F = \sqrt{\sum_{i,j} A_{i,j}^2},$$

Norm penalties

- Limiting the capacity of models by adding norm penalty $\Omega(\Theta)$ to the objective function J

Regularized objective function

Penalty term

$$\tilde{J}(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha\Omega(\boldsymbol{\theta})$$

Original objective function

- Not modifying the model in inference phase, but adding penalties to the objective function in learning phase

- Also known as weight decay

Modified objective function

- $\tilde{J}$ will decrease both J and Θ

- Setting α to 0 results in no regularization and larger values α correspond to more regularization

Regularized objective function

Penalty term

$$\tilde{J}(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha\Omega(\boldsymbol{\theta})$$

Original objective function

L2 norm Regularization

- **Substituting squared L2 norm to the $\Omega(\Theta)$**

$||\theta||_2^2$

$$\tilde{J}(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha\Omega(\boldsymbol{\theta})$$

$$\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \frac{\alpha}{2}\boldsymbol{w}^\top \boldsymbol{w} + J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}),$$

- **Calculating gradient**

$$\nabla_{\boldsymbol{w}}\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha\boldsymbol{w} + \nabla_{\boldsymbol{w}}J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}).$$

L2 norm Regularization

- **Calculating gradient**

$$\nabla_{\boldsymbol{w}} \tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha \boldsymbol{w} + \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}).$$

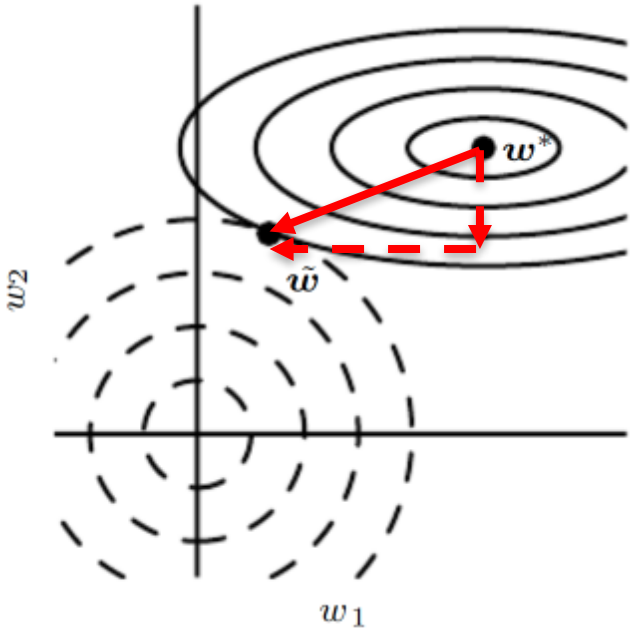
- **Applying weight update**

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \epsilon \left(\alpha \boldsymbol{w} + \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) \right).$$

$$\boldsymbol{w} \leftarrow (1 - \epsilon\alpha)\boldsymbol{w} - \epsilon \nabla_{\boldsymbol{w}} J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}).$$

Effect of L2 norm Regularization

- Only directions along which the parameters contribute significantly to reducing the objective function are preserved relatively intact



L1 norm Regularization

- **Substituting L1 norm to the $\Omega(\Theta)$**

$$\tilde{J}(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha\Omega(\boldsymbol{\theta})$$

$||\theta||_1$

$$\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha||\boldsymbol{w}||_1 + J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}),$$

- **Calculating gradient**

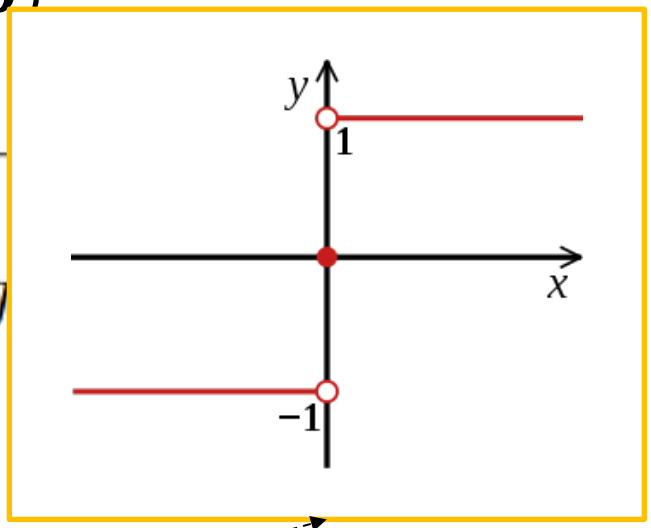
$$\nabla_{\boldsymbol{w}}\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha\text{sign}(\boldsymbol{w}) + \nabla_{\boldsymbol{w}}J(\boldsymbol{X}, \boldsymbol{y}; \boldsymbol{w})$$

L1 norm Regularization

- **Substituting L1 norm to the $\Omega(\Theta)$**

$$\tilde{J}(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) +$$

$$\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha||\boldsymbol{w}||_1 + J$$



- **Calculating gradient**

$$\nabla_{\boldsymbol{w}} \tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha \text{sign}(\boldsymbol{w}) + \nabla_{\boldsymbol{w}} J(\boldsymbol{X}, \boldsymbol{y}; \boldsymbol{w})$$

Comparison L1 and L2 norm Regularization

- The L1 norm is commonly used in machine learning when the difference between zero and nonzero elements is very important

- L2 regularization does not cause the parameters to become zero

- L1 regularization may cause the parameters to become zero for large enough α

Norm Regularization without bias

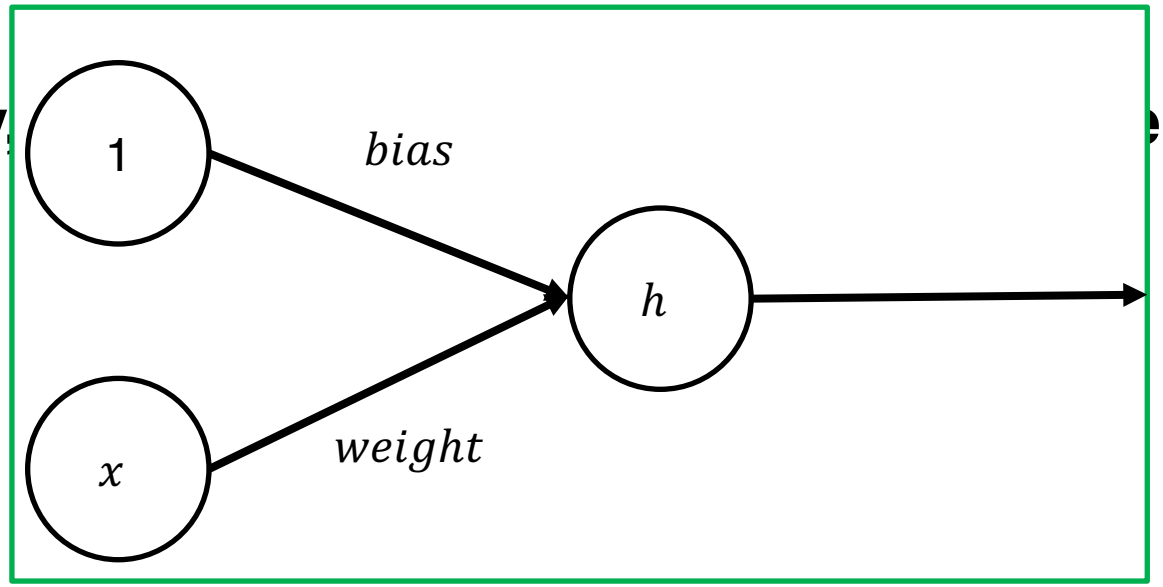
- **Usually, bias of each weight is excluded in penalty terms**

$$\tilde{J}(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha\Omega(w)$$

- **The reason is :**
  - The biases typically require **less data** to fit than the weights
  - Each weight specifies how two variables interact while biases specify **interact of one variables**
  - Regularizing the bias parameters can **cause underfitting**

Norm Regularization without bias

- Usually, we only penalty terms



- The reason is :
  - The biases typically require **less data** to fit than the weights
  - Each weight specifies how two variables interact while biases specify **interact of one variables**
  - Regularizing the bias parameters can **cause underfitting**

Pros and Cons

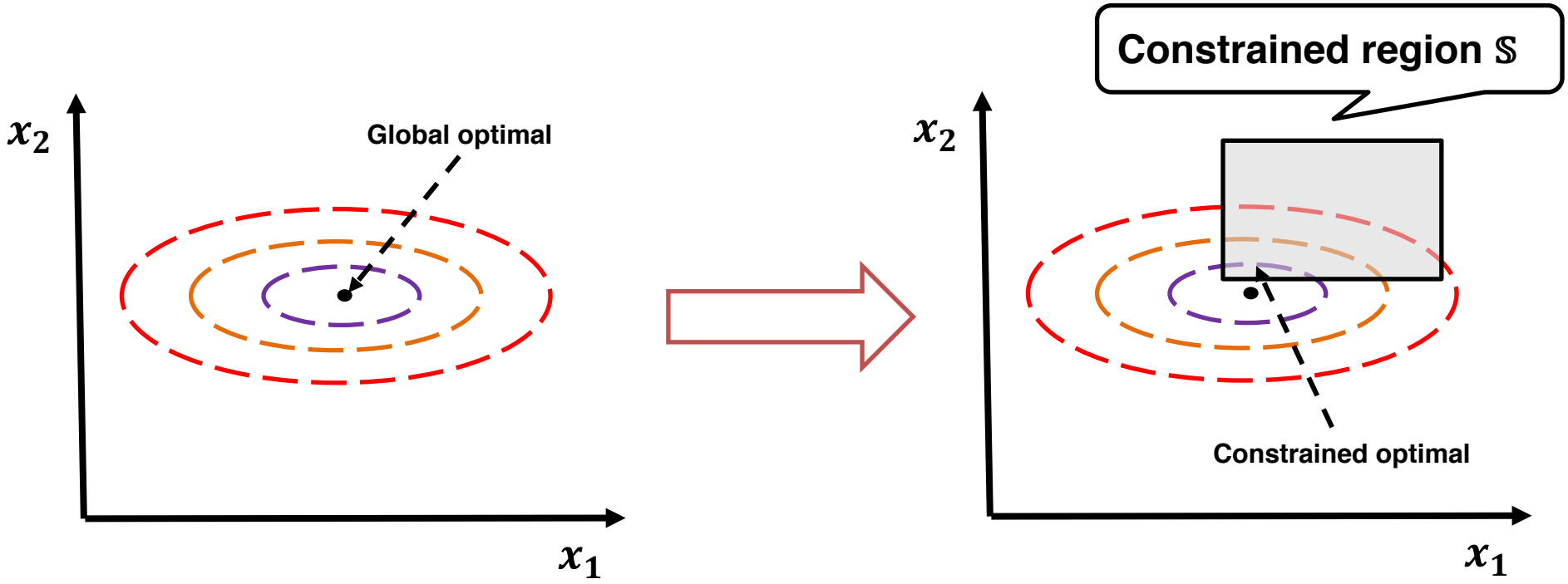
- Norm regularization doesn't modify formula of inference phase

- Norm regularization sometimes causes local minima

Norm Penalties as Constrained Optimization

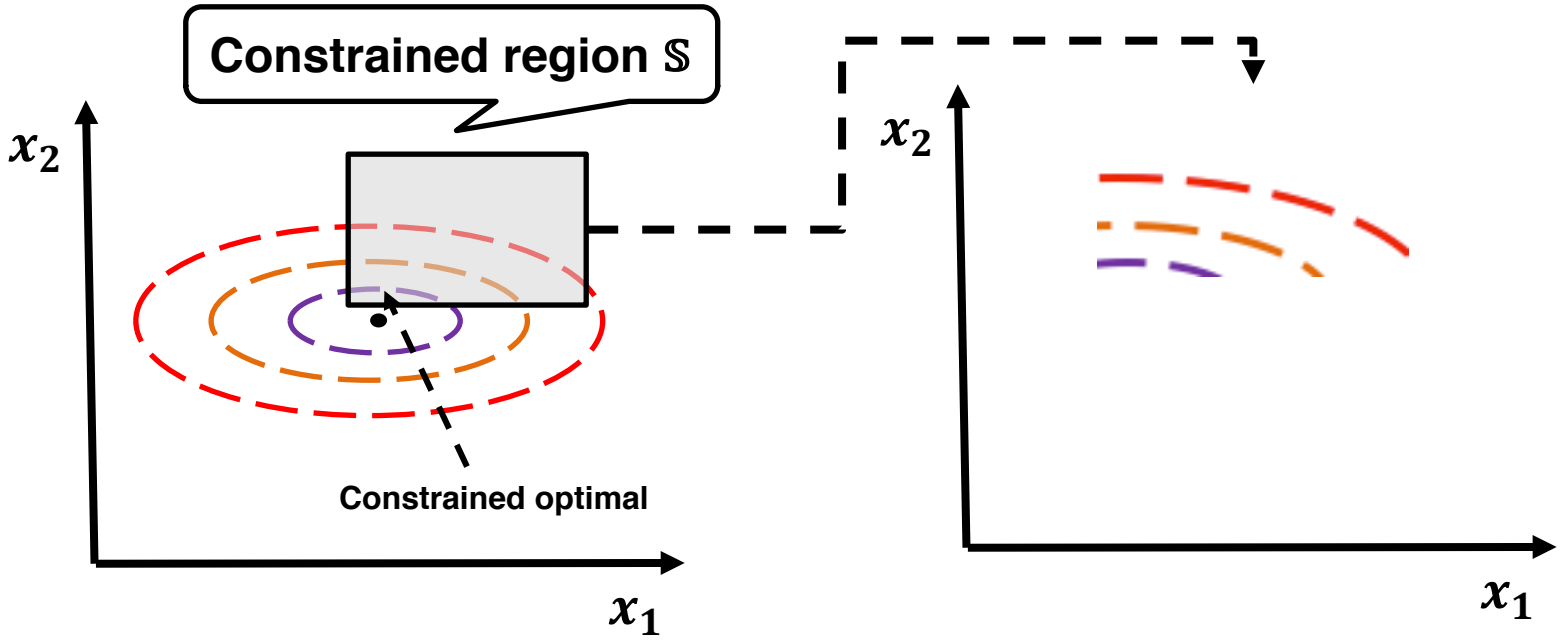
Constrained optimization

- Sometimes one may wish to find the maximal or minimal value of $f(x)$ for value of x in some set $\mathbb{S}$



Expression of constrained function

- To express function with constrained condition is difficult



Generalized Lagrange function

- A possible approach is to design a different, unconstrained optimization problem whose solution can be converted into a solution to the original constrained problem

- The unconstrained optimization function is called “Generalized Lagrange function”

Generalized Lagrange function

- **Generalized Lagrange function is defined as:**

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\alpha}) = f(\boldsymbol{x}) + \sum_i \lambda_i g^{(i)}(\boldsymbol{x}) + \sum_j \alpha_j h^{(j)}(\boldsymbol{x}).$$

- **Where the constrained region is:**

$$\mathbb{S} = \{\boldsymbol{x} | \forall \boldsymbol{i}, \boldsymbol{g}^{(\boldsymbol{i})}(\boldsymbol{x}) = \boldsymbol{0} \; \boldsymbol{and} \; \forall \boldsymbol{j}, \boldsymbol{h}^{(\boldsymbol{j})} \leq \boldsymbol{0}\}$$

- **We can find optimal x in region $\mathbb{S}$ by solving:**

$$\min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda}} \max_{\boldsymbol{\alpha}, \boldsymbol{\alpha} \geq 0} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\alpha}).$$

Norm Penalties with respect to Constrained Optimization

- **Cost function regularized by a parameter norm penalty**

$$\tilde{J}(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha\Omega(\boldsymbol{\theta}).$$

Original cost function — norm penalty, constrained term

- **If we wanted to constrain $\Omega(\theta)$ to be less than some constant k, we could construct a generalized Lagrange function**

$$\mathcal{L}(\boldsymbol{\theta}, \alpha; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha(\Omega(\boldsymbol{\theta}) - k).$$

Norm Penalties with respect to Constrained Optimization

- **Generalized Lagrange function**

$$\mathcal{L}(\boldsymbol{\theta}, \alpha; \boldsymbol{X}, \boldsymbol{y}) = J(\boldsymbol{\theta}; \boldsymbol{X}, \boldsymbol{y}) + \alpha(\Omega(\boldsymbol{\theta}) - k).$$

- **The solution to the constrained problem is given by**

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \max_{\alpha, \alpha \geq 0} \mathcal{L}(\boldsymbol{\theta}, \alpha).$$

- **α must increase whenever $\Omega(\theta) > k$ and decrease whenever $\Omega(\theta) < k$**

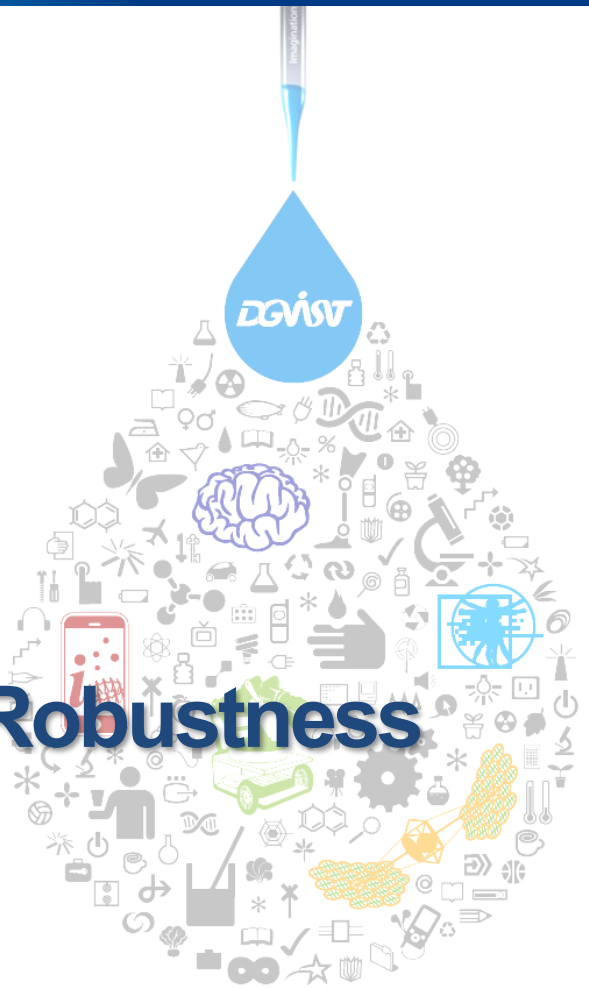
Norm Penalties with respect to Constrained Optimization

- To gain some insight into the effect of the constraint, we can fix α^* and view the problem as just a function of θ :

$$\theta^* = argmin_\theta \mathcal{L}(\theta, \alpha^*) = argmin_\theta [J(\theta; X, y) + \alpha^* \Omega(\theta)]$$

- This is exactly the same as the regularized training problem of minimizing $\tilde{J}$

Data Augmentation and Noise Robustness



Introduction to Dataset Augmentation

- The best way to make a machine learning model generalize better is to train it on more data

- Dataset augmentation is a technique that creating fake data and adding it to the training set

Original data

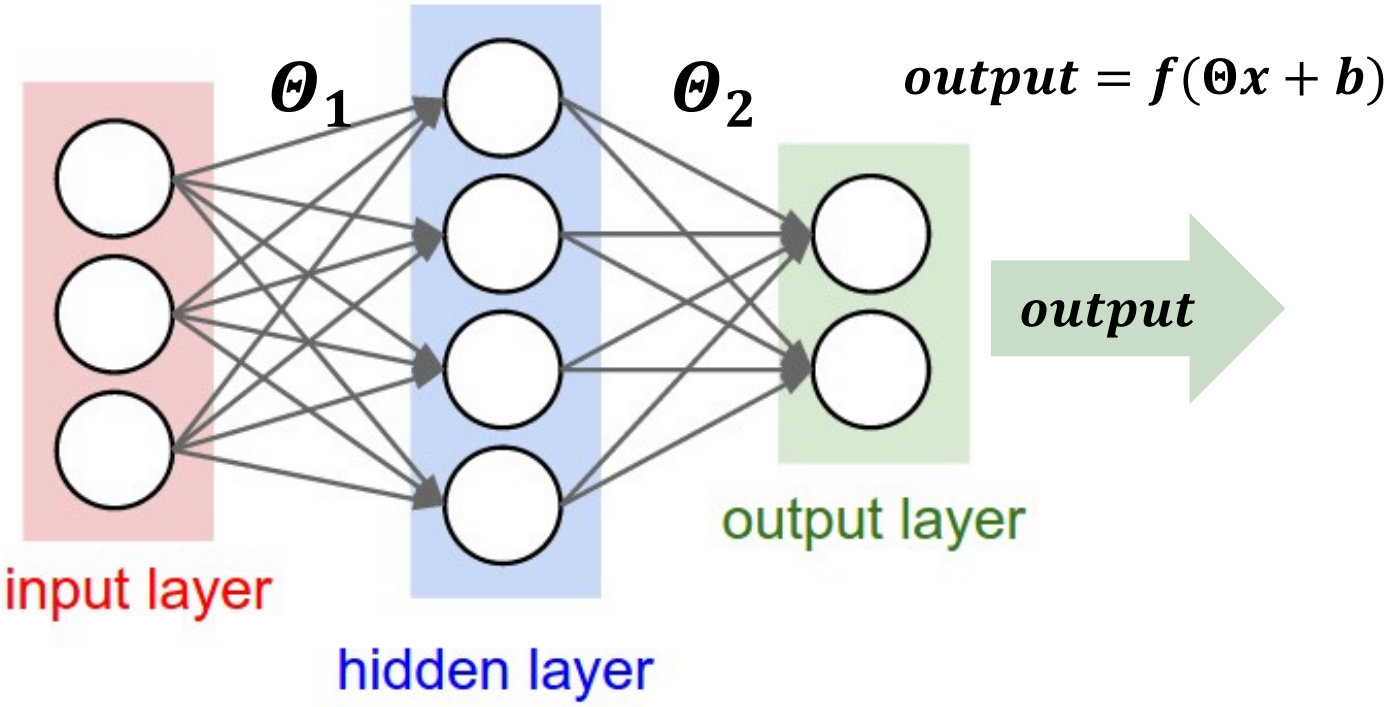
Artificial data

Image source : https://www.pexels.com/photo

Injecting noise (Training data)

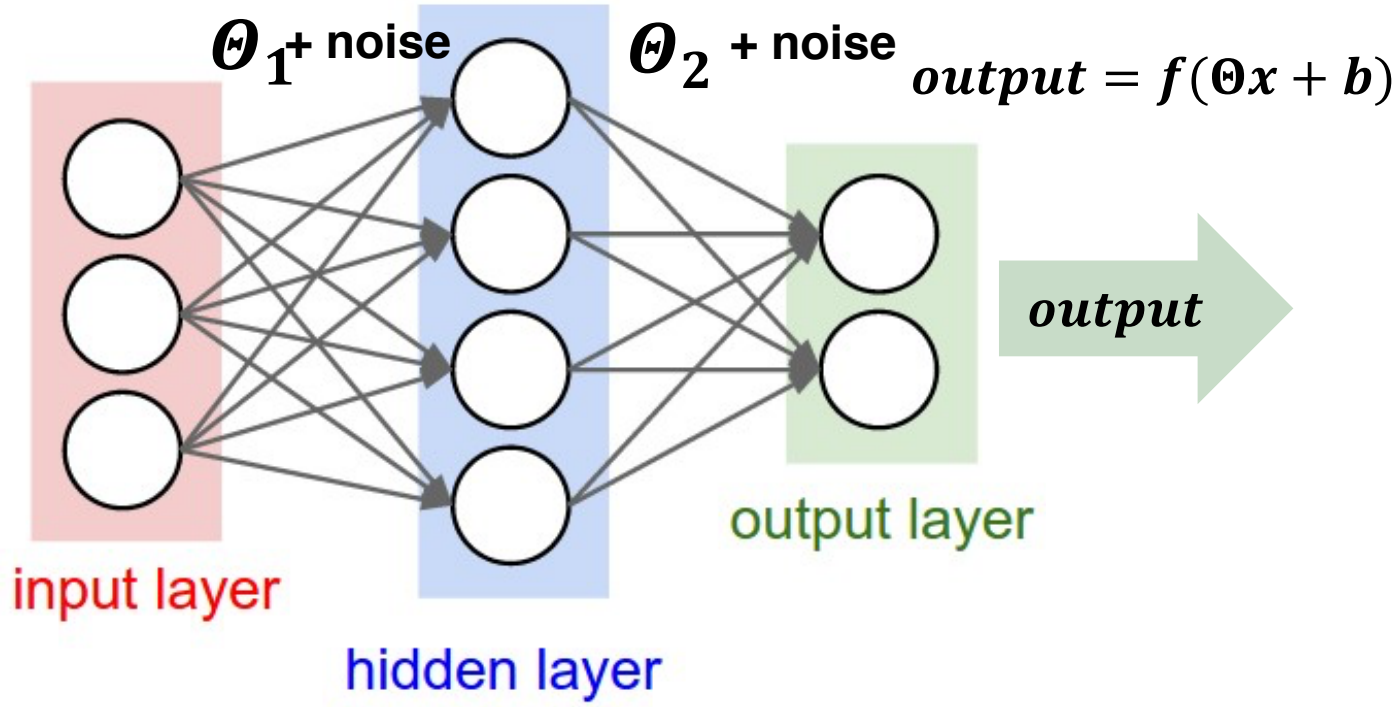
- Injecting random noise into input data to improve robustness

Random
Noise Filter



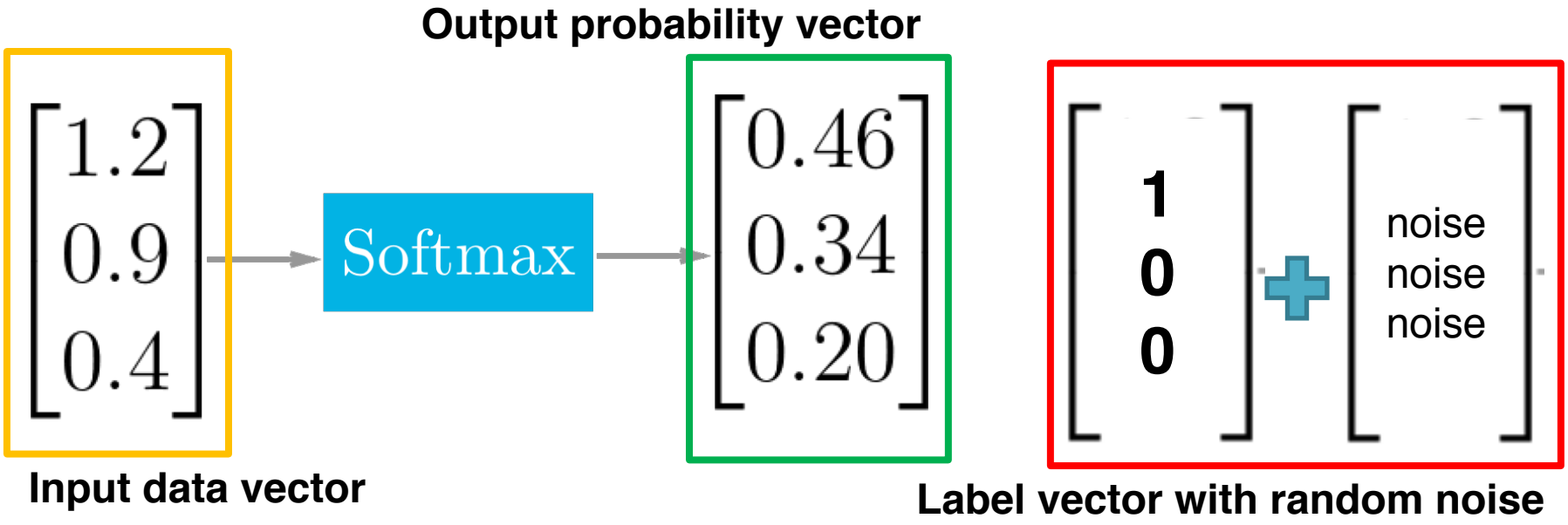
Injecting noise (Weight)

- Injecting random noise into weight to improve robustness
- This makes the model relatively insensitive to small variations in the weights



Injecting noise (Label)

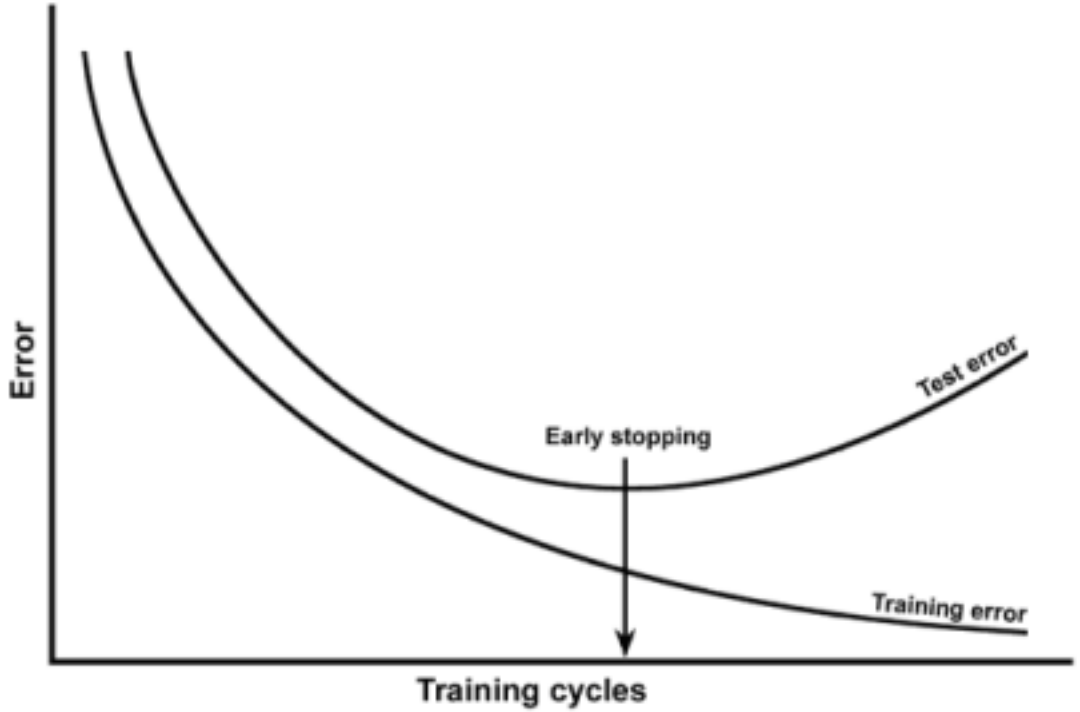
- Most datasets have some amount of mistakes in the y labels



Early Stopping

Necessity of early stopping

- Learning too much iterations causes overfitting



Feature of early stopping

- **Early stopping regards iteration number as hyper parameter and find optimal value of it**

- **Computation cost to find optimal iteration number is negligible**
  - This can be parallelized while training process on separate machine

- **Early stopping doesn't affect the formula of cost function**