Deep Learning Seminar
Chapter 5. Machine Learning Basics
- Part 2 -

Hyun-Lim YANG

Department of Information and Communication Engineering

DGIST

2017.07.20

Contents

Chapter 5. Machine Learning Basics

- **Part 1**
  - 5.1 Learning algorithms
  - 5.2 Capacity, overfitting and underfitting
  - 5.3 Hyperparameters and validation sets
  - 5.4 Estimators, bias and variance
  - 5.5 Maximun likelihood estimation
- **Part 2**
  - 5.6 Bayesian statistics
  - 5.7 Supervised learning algorithms
  - 5.8 Unsupervised learning algorithms
  - 5.9 Stochastic gradient descent
  - 5.10 Building a machine learning algorithm
  - 5.11 Challenges motivating deep learning

Chapter 5. Machine Learning Basics

- Part 1
  - 5.1 Learning algorithms
  - 5.2 Capacity, overfitting and underfitting
  - 5.3 Hyperparameters and validation sets
  - 5.4 Estimators, bias and variance
  - 5.5 Maximun likelihood estimation
- Part 2
  - 5.6 Bayesian statistics
  - 5.7 Supervised learning algorithms
  - 5.8 Unsupervised learning algorithms
  - 5.9 Stochastic gradient descent
  - 5.10 Building a machine learning algorithm
  - 5.11 Challenges motivating deep learning

Bayesian statistics

Chapter 5. Machine Learning Basics

- **Part 1**
  - 5.1 Learning algorithms
  - 5.2 Capacity, overfitting and underfitting
  - 5.3 Hyperparameters and validation sets
  - 5.4 Estimators, bias and variance
  - 5.5 Maximun likelihood estimation

Frequentist statistics
Point estimation

- **Part 2**
  - 5.6 Bayesian statistics
  - 5.7 Supervised learning algorithms
  - 5.8 Unsupervised learning algorithms
  - 5.9 Stochastic gradient descent
  - 5.10 Building a machine learning algorithm
  - 5.11 Challenges motivating deep learning

Bayesian statistics

Bayesian statistics

- **Bayesian perspective**
  - Uses probability to reflect degrees of certainty of states of knowledge
  - The dataset is directly observed and so is not random
  - Parameter θ is represented as random variable
- **The prior**
  - We represent our knowledge of θ using the prior probability distribution, notation with $p(\theta)$, before observing data
  - Select broad priori distribution (with high degree of uncertainty), such as finite range of volume, with a uniform distribution, or Gaussian.

Mathematical description

- **Set of data samples** $\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(m)}\}$
- **The dataset is directly observed and so is not random**
- **Parameter** $\boldsymbol{\theta}$ **is represented as random variable**
- **Combine the data likelihood with the prior via Bayes' rule:**

$$\boldsymbol{p}\left(\boldsymbol{\theta} \middle| \boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(m)}\right) = \frac{\boldsymbol{p}\left(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(m)} \middle| \boldsymbol{\theta}\right) \boldsymbol{p}(\boldsymbol{\theta})}{\boldsymbol{p}(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(m)})}$$

Mathematical description

- Set of data samples $\{x^{(1)}, x^{(2)}, \cdots, x^{(m)}\}$
- The dataset is directly observed and so is not random
- Parameter θ is represented as random variable
- Combine the data likelihood with the prior via Bayes' rule:

likelihood prior

$$\boxed{p\left(\theta \middle| x^{(1)}, \cdots, x^{(m)}\right)} = \frac{p\left(x^{(1)}, \cdots, x^{(m)} \middle| \theta\right) p(\theta)}{p(x^{(1)}, \cdots, x^{(m)})}$$

Bayesian inference

Relative to MLE

- **Make prediction using a full distribution over θ**
- **After observing m samples, predict distribution over the next data sample, $x^{(m+1)}$, is given by:**

$$p\left(x^{(m+1)} \middle| x^{(1)}, \cdots, x^{(m)}\right) = \int p\left(x^{(m+1)} \middle| \theta\right) p\left(\theta \middle| x^{(1)}, \cdots, x^{(m)}\right) d\theta$$

- **Prior distribution has influence by shifting probability toward the parameter space**
- **Bayesian method typically generalize much better**
- **But high computational cost**

Maximum A Posterior (MAP) Estimation

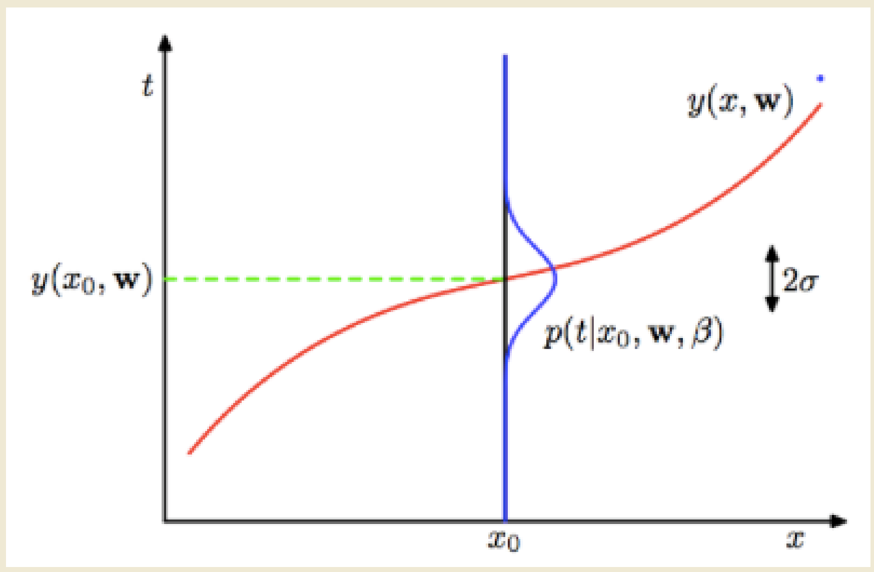
- **Chose the point of maximal posterior probability**

$$\theta_{MAP} = \underbrace{\arg\max_{\theta} p(\theta|x)}_{\text{posterior}} = \underbrace{\arg\max_{\theta} \log p(x|\theta)}_{\text{likelihood}} + \boxed{\underbrace{\log p(\theta)}_{\text{prior}}}$$

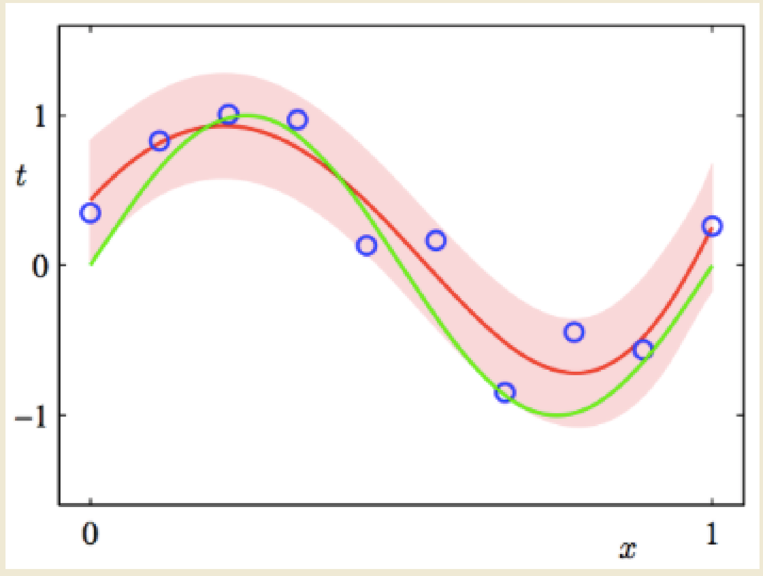
Similar with weight decay term

- **Has the advantage of leveraging information that is brought by the prior**
- **Additional information helps the variance of MAP estimation**
- **But it increase bias**
- **Regularized estimation strategies can be interpreted as making the MAP approximation**

MLE vs MAP

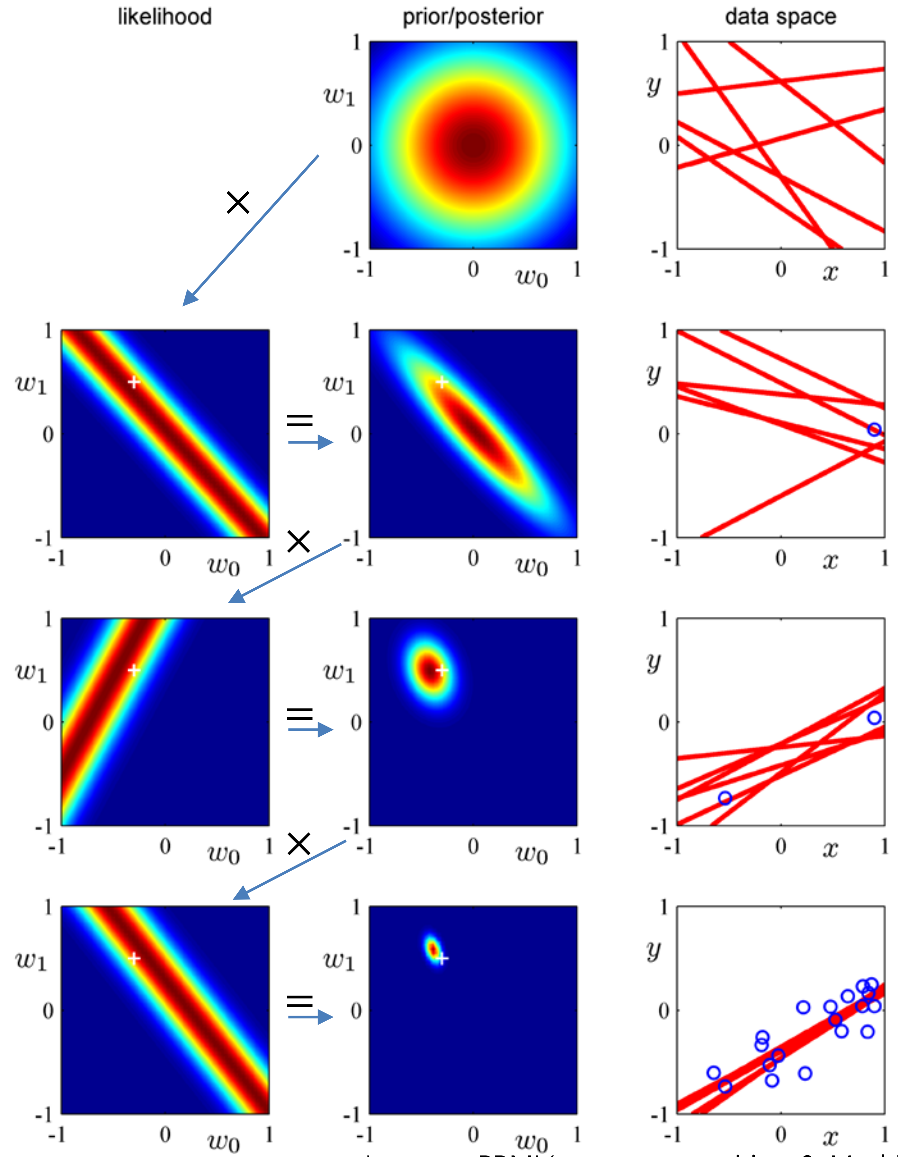


<MLE>



<MAP>

Bayesian statistics



1st sampled data point

2nd sampled data point

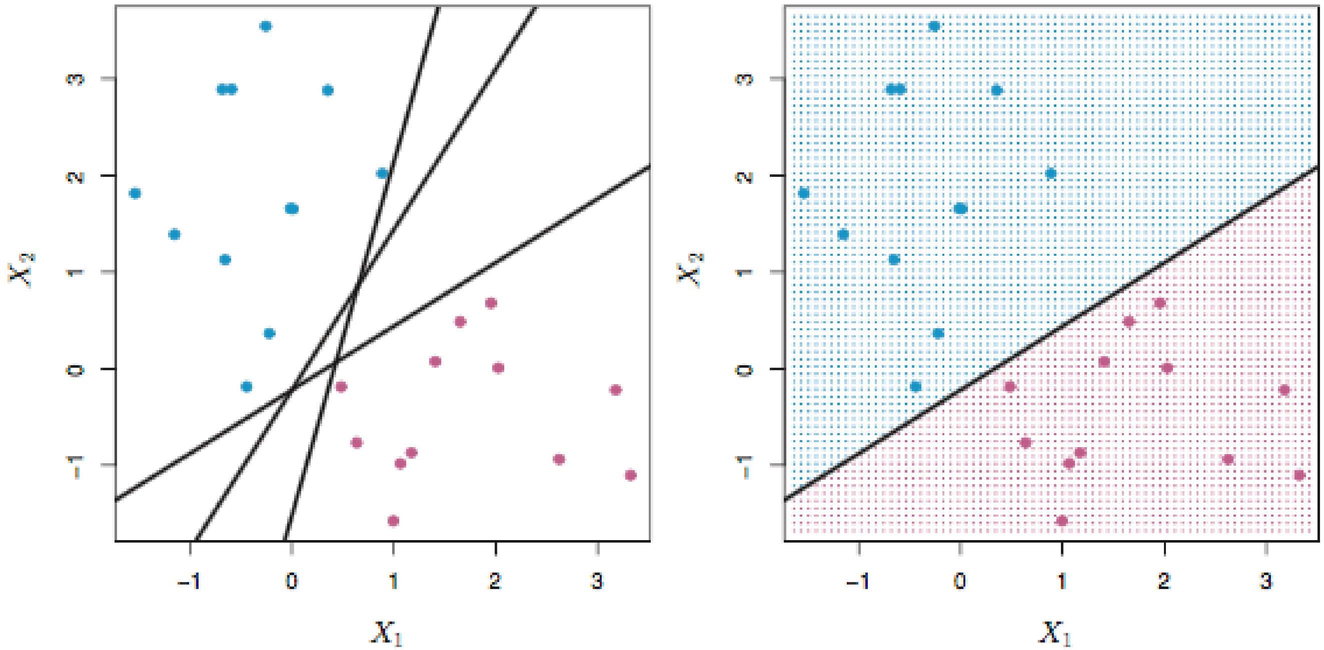
20th sampled data point

*source : PRML(pattern recognition & Machine Learning) Textbook

Supervised Learning Algorithms

Support Vector Machine (SVM)

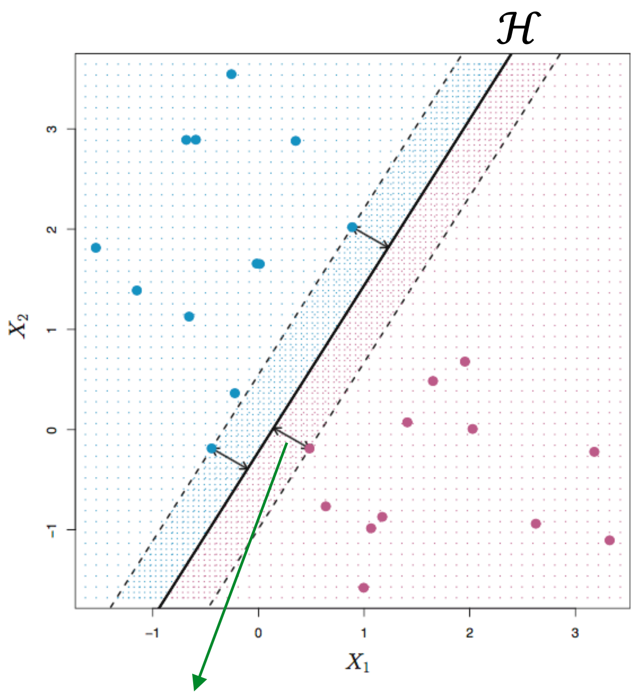
- **Definition**
  - Two-class classification problem in direct way
  - Find a plane that separates the classes in feature space as far as possible
- **Separating Hyperplane**



* *Image from "Introduction to Statistical Learning with R", springer*

Support Vector Machine (SVM)

Maximal Margin Classifier



Margin M

$$\left|\frac{|1-b|}{\|w\|} - \frac{|-1-b|}{\|w\|}\right| = \frac{1}{2}\frac{2}{\|w\|^2} = \frac{1}{\|w\|^2}$$

Let separating Hyperplane $\mathcal{H}$ *as*

$$w^T x + b = 0$$

if we rescale the margin with 1, *then*

$$w^T x + b \le -1 \; (for\ y_i = -1)$$
$$w^T x + b \ge +1 \; (for\ y_i = 1)$$

combining two equation,

$$y_i\left(w^T x^i + b\right) - 1 \ge 0$$

so, the maximal margin is as follows:

$$\textbf{Minimize } \|w\|$$
$$\textbf{subject to } \; y_i(w^T x + b) - 1 \ge 0, \qquad i = 1, \dots, k$$

with generalization,

$$\max_{w} \; M$$
$$\textbf{subject to } \; y_i(w^T x) \ge M, \qquad i = 1, \dots, k$$
$$\|w\|^2 = 1$$

* Image from "Introduction to Statistical Learning with R", springer

Support Vector Machine (SVM)

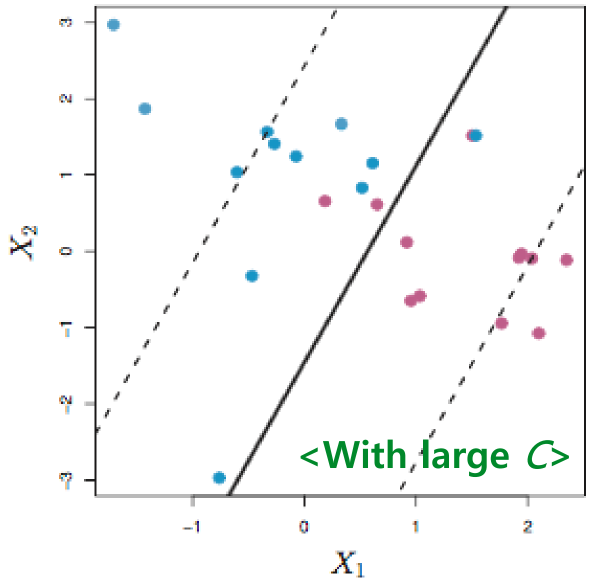
Support Vector Classifier

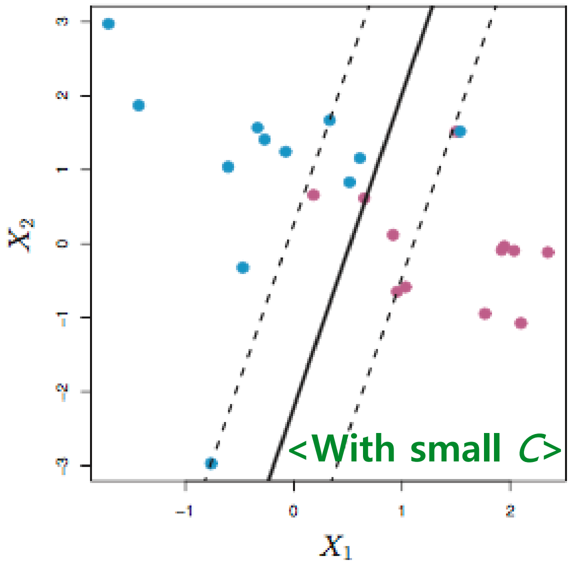
$$\max_{w,\varepsilon} \; \boldsymbol{M}$$

$$\textit{subject to} \;\; y_i(w^T x) \geq M(1-\varepsilon_i)$$

$$\|w\|^2 = 1$$

$$\varepsilon_i \geq 0 \,, \sum_{i=1}^{n} \varepsilon_i \leq C$$



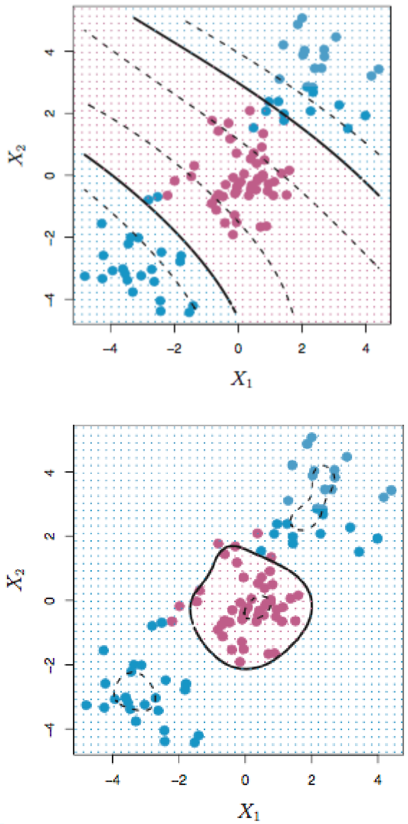


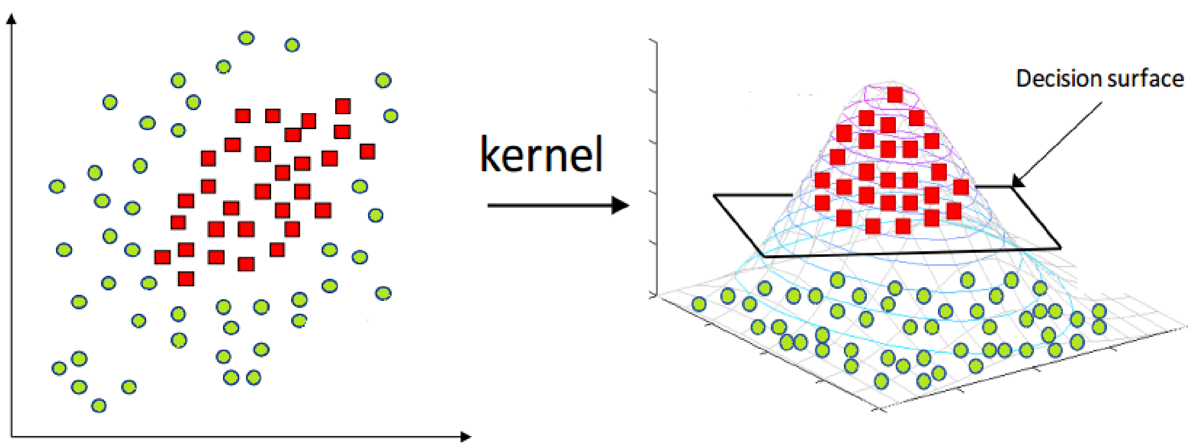
* Image from "Introduction to Statistical Learning with R", springer

Support Vector Machine (SVM)

Kernel Method (SVM)

- In SVM, we **just need to calculate inner product of vectors**
- If the data is not linear separable, we send the data to more high dimensional space and make Support Vector Classifier



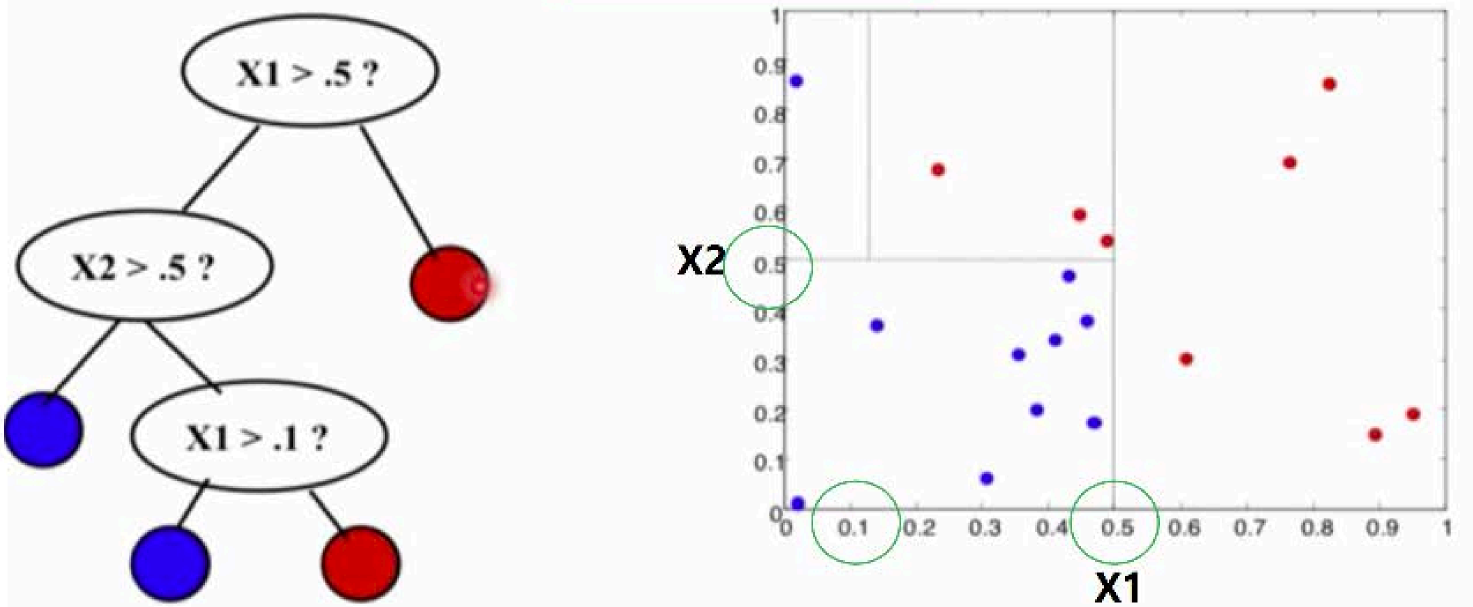


** Image from “Introduction to Statistical Learning with R”, springer*

Tree Based Methods

Decision Tree Classification

- "Split" input into cases
  - Usually based on a single variable
  - Recurse down until we reach a decision
  - Continuous vars: choose split point



* *Slides from Seo Hui(LG Electronics), "Gradient Boosting Model"*

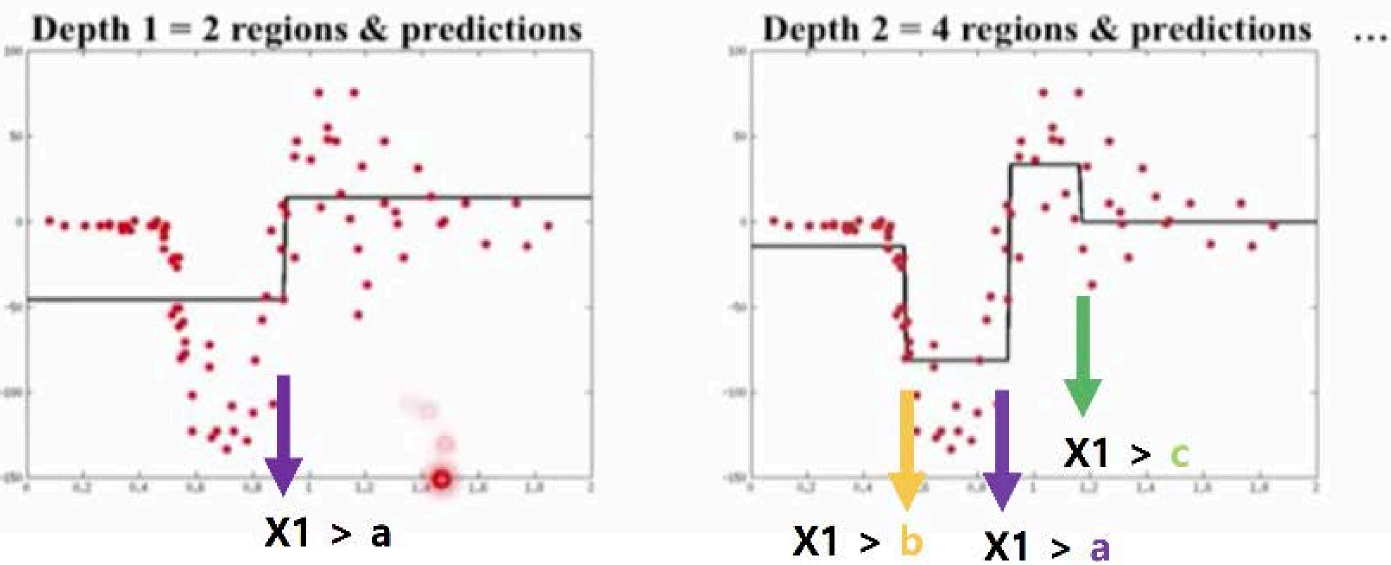
Tree Based Methods

- Decision Tree Regression

  - Exactly the same
  - Predict real valued numbers at leaf nodes

  - Examples on a single scalar feature:

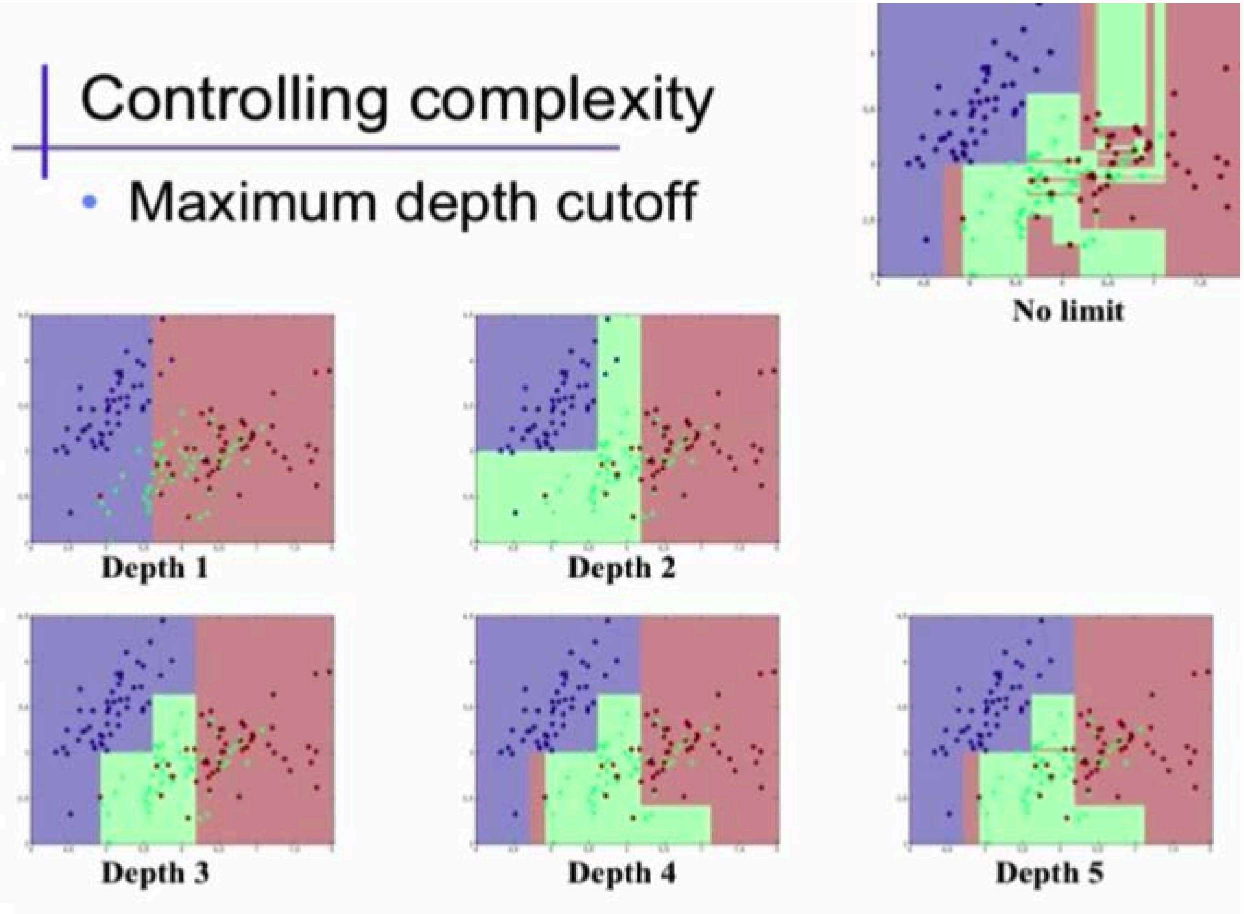
Real value Y



* *Slides from Seo Hui(LG Electronics), "Gradient Boosting Model"*

Tree Based Methods

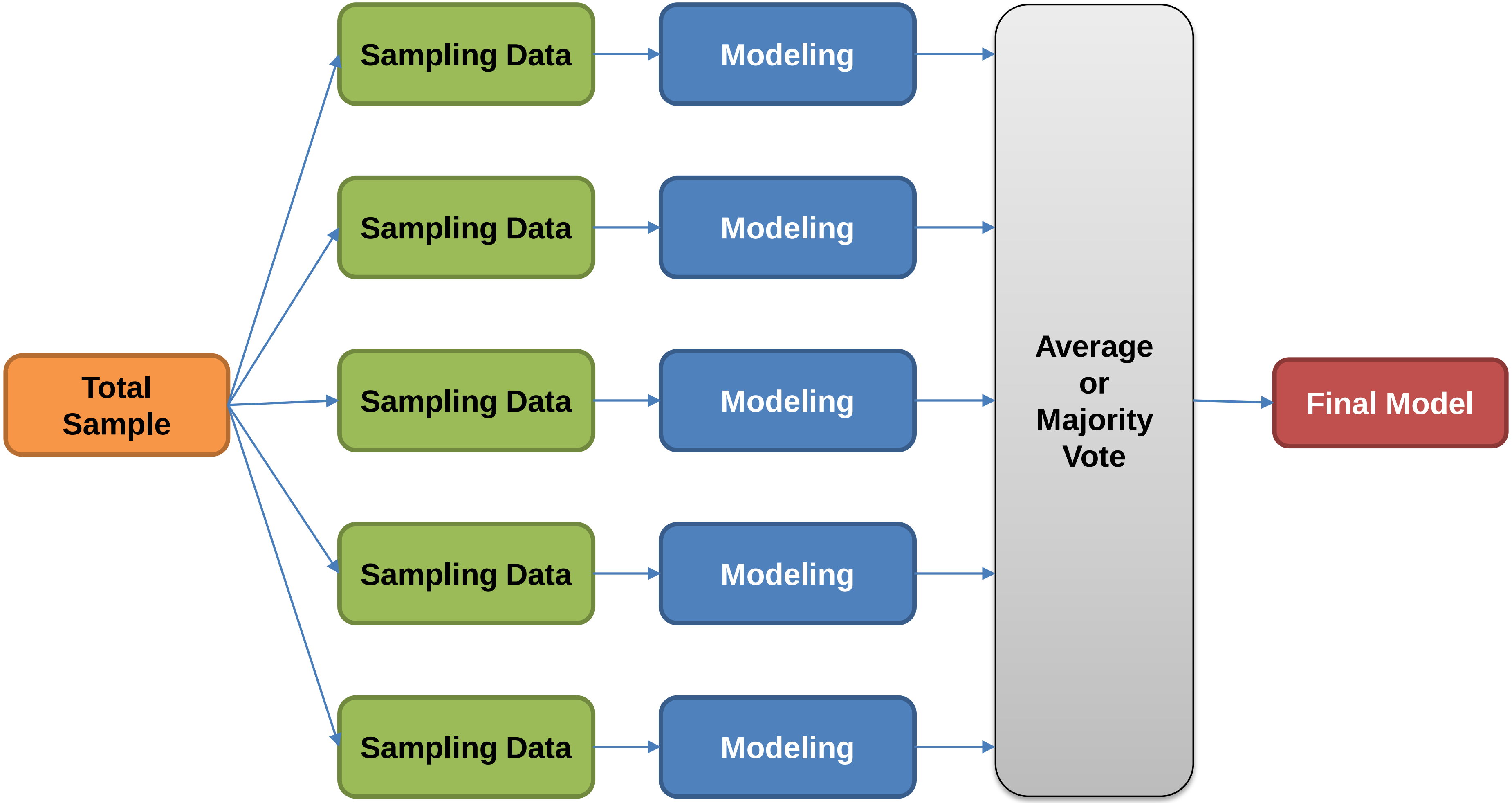
- **Decision Tree Complexity**



* *Slides from Seo Hui(LG Electronics), "Gradient Boosting Model"*

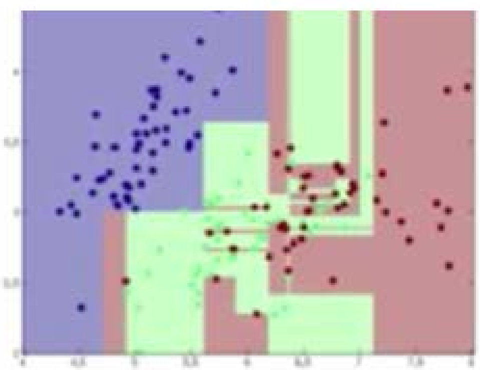
Tree Based Methods

- **Bagging**

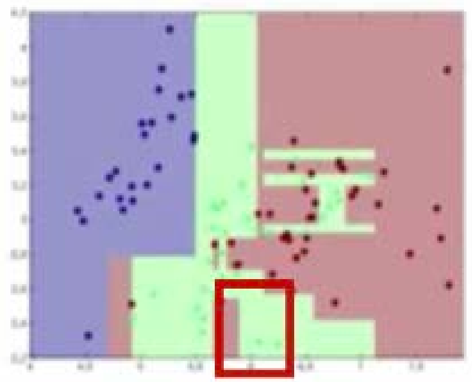


Tree Based Methods

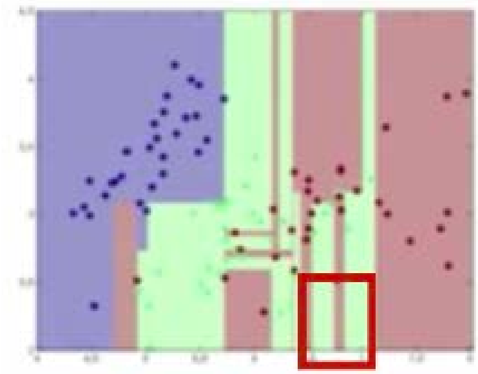
- **Bagging**



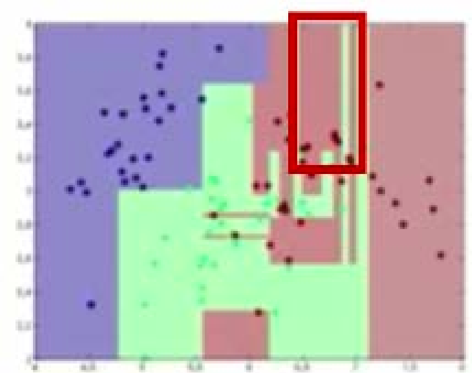
<Model with Full dataset>



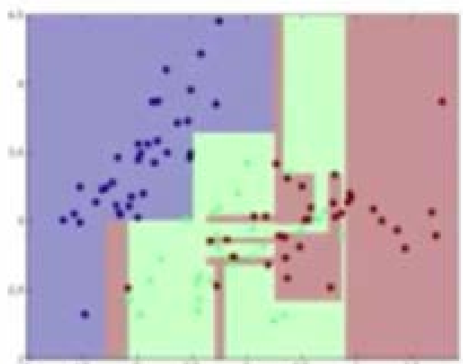
<Model No. 1>



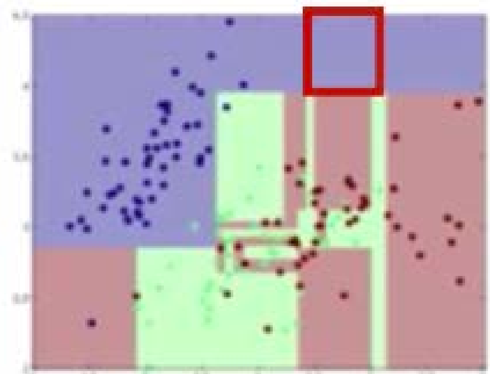
<Model No. 2>



<Model No. 3>



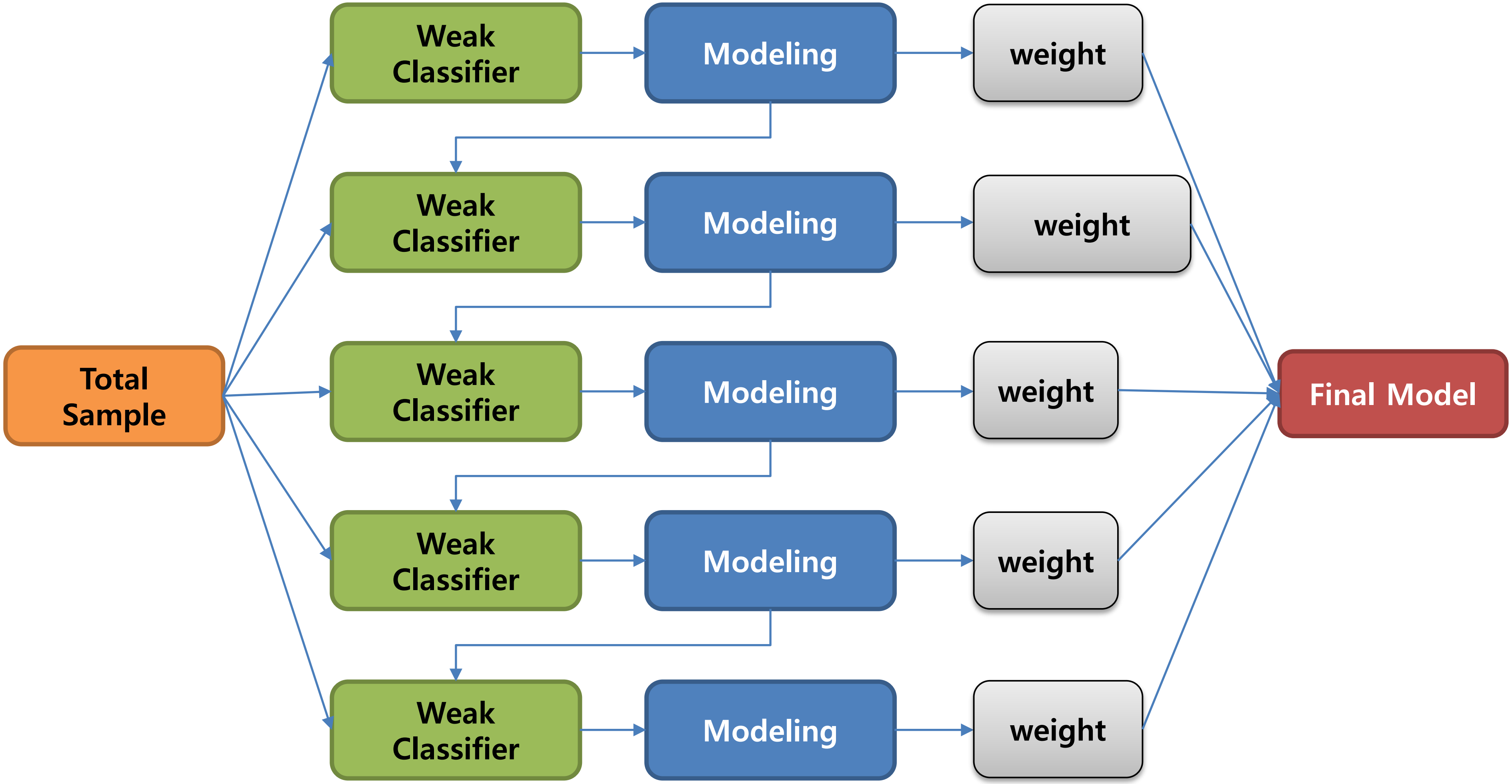
<Model No. 4>



<Model No. 5>

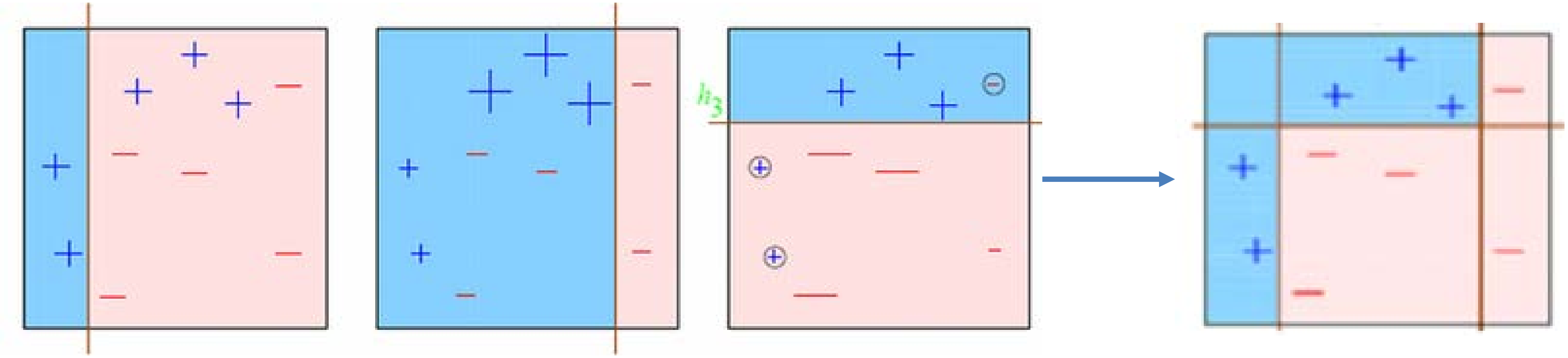
Tree Based Methods

- **Boosting**



Tree Based Methods

- **Boosting**



  - Let the problem which we should classify '+', and '-' with tree-based classifier
  - First, a **weak classifier** classify the label with left-sided vertical single line
  - Then, **weight to the incorrect points**(large annotated '+' in second figure), and **do weak classify again**(right-sided line)
  - Repeat those procedure, and finally merge the weak classifiers

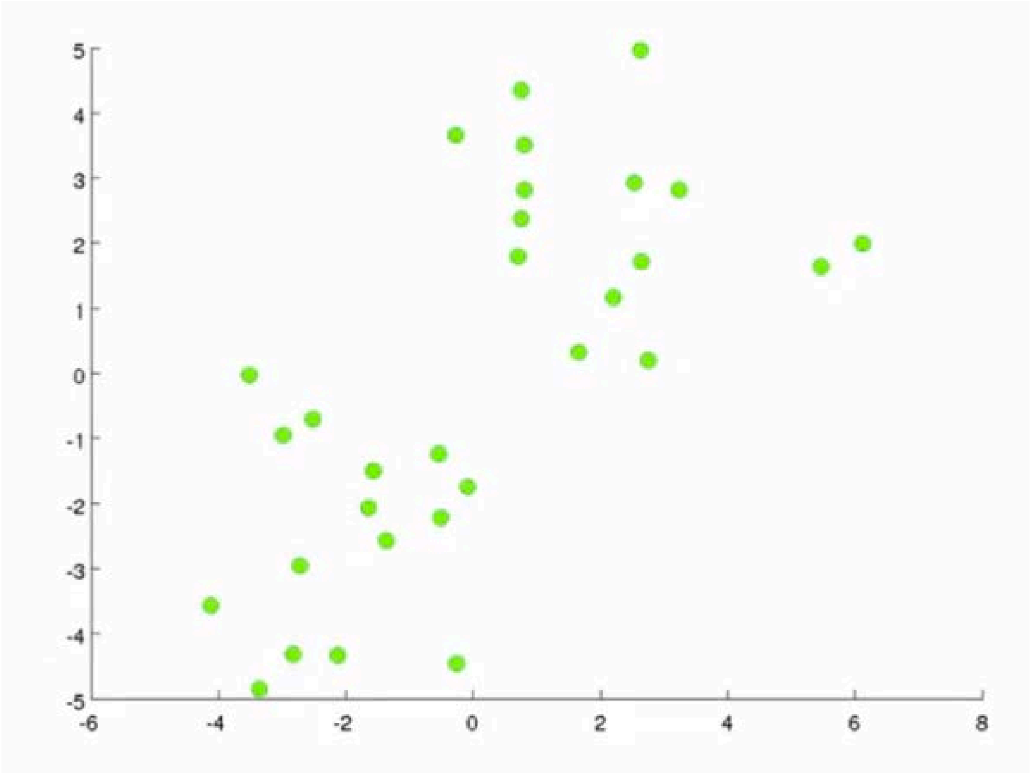
Another Supervised Learning Algorithms

- Linear Regression
  - Ridge
  - Lasso
- Logistic Regression
- LDA (Linear Discriminant Analysis)
- Random Forest
- KNN (K-Nearest Neighbor)
- Naïve Bayes
- Neural Network (MLP)
- …

Unsupervised Learning Algorithms

K-means clustering

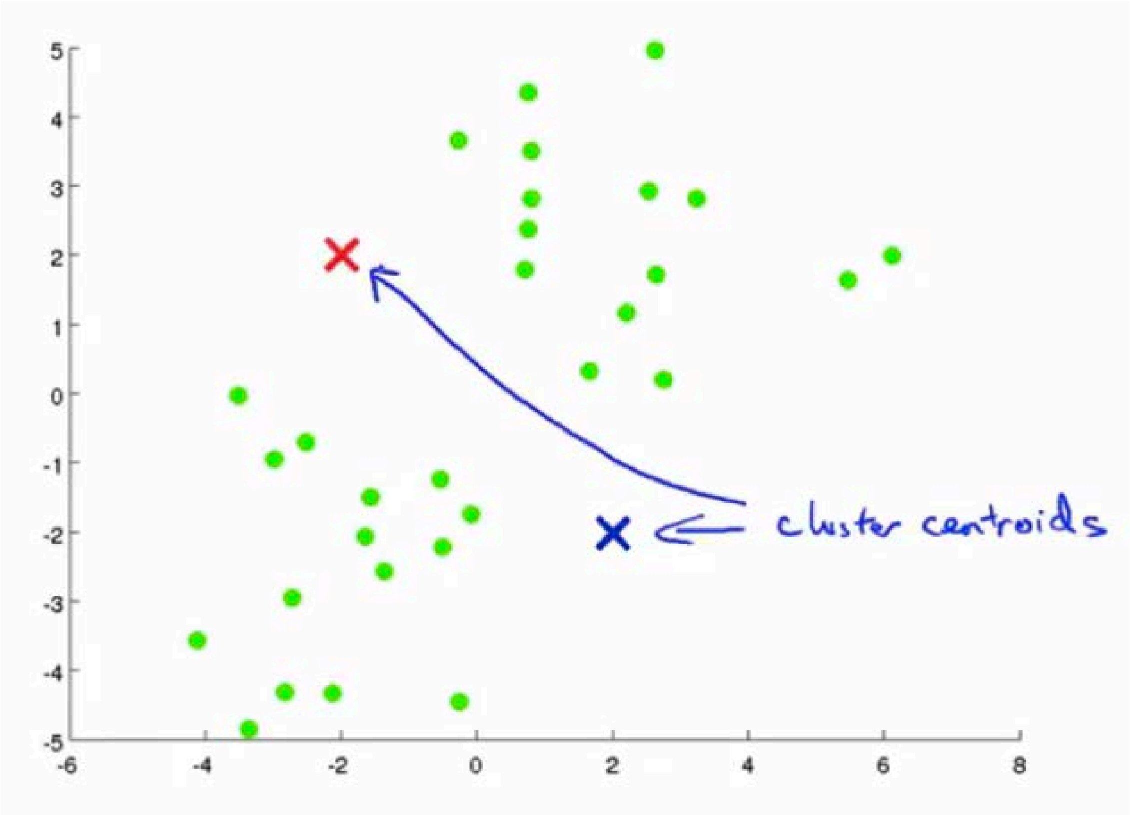
- **Find the K clusters that best describes the data**



** Slides from Andrew Ng(Stanford Univ.), "Machine Learning"*

K-means clustering

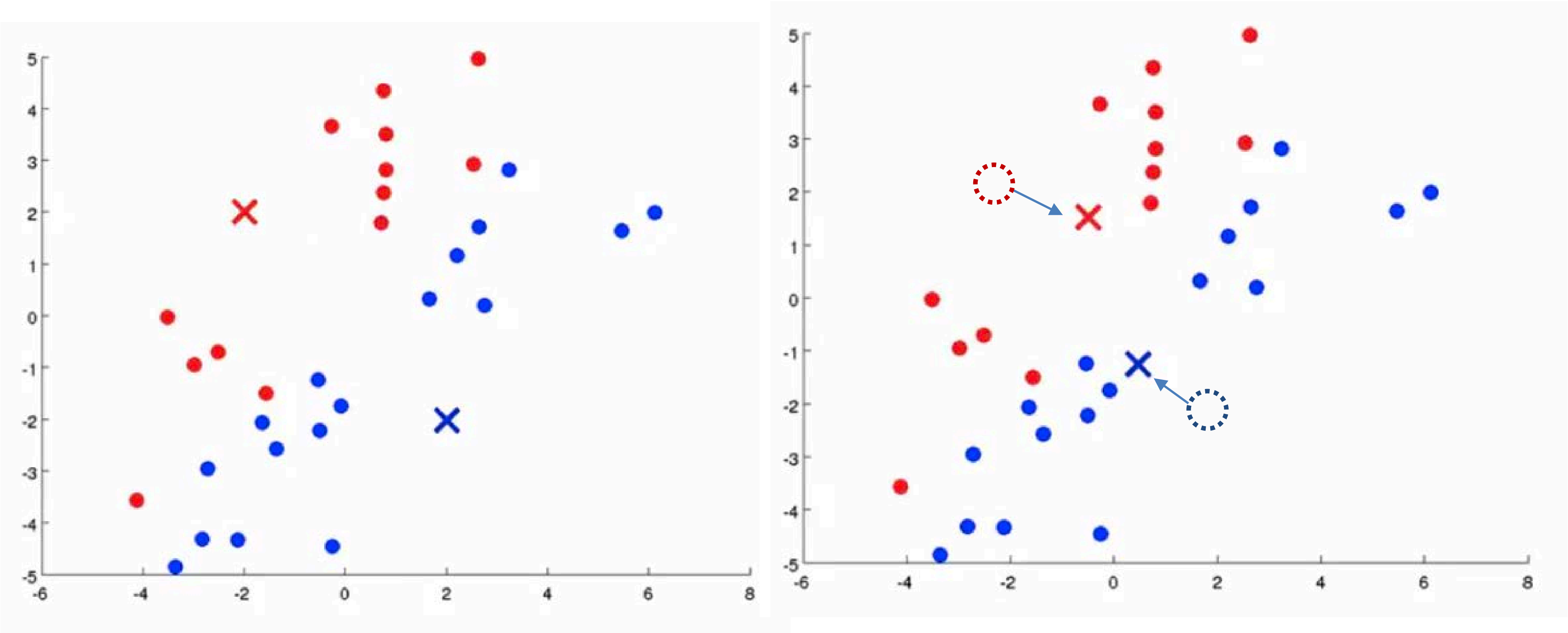
- **Number of cluster k = 2,**
  - Randomly initialize "centroids"



* *Slides from Andrew Ng(Stanford Univ.), "Machine Learning"*

K-means clustering

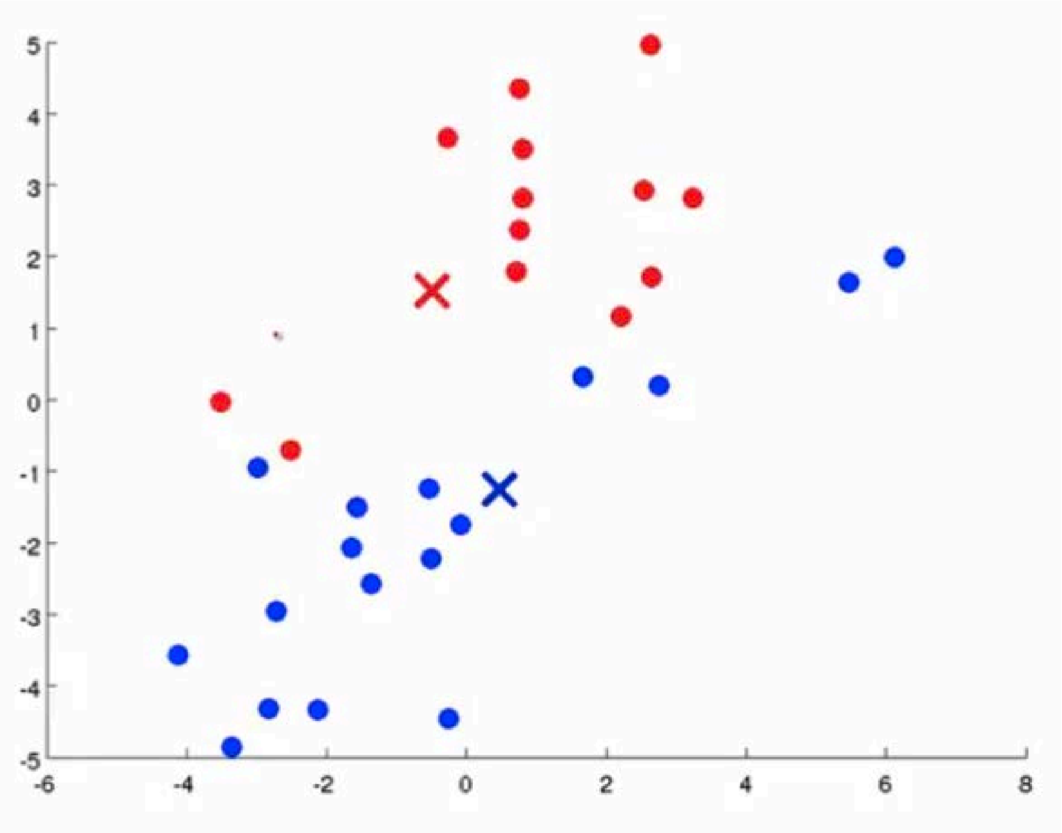
- **Number of cluster k = 2,**
  - Assign cluster membership
  - Update the cluster centroid (average of the data points in each cluster)



* *Slides from Andrew Ng(Stanford Univ.), "Machine Learning"*

K-means clustering

- **Number of cluster k = 2,**
  - Update cluster membership
  - Repeat those procedure until no membership update



** Slides from Andrew Ng(Stanford Univ.), "Machine Learning"*

Another Unsupervised Learning Algorithms

- PCA (Principal Component Analysis)
- ICA (Independent Component Analysis)
- ARM (Association Rule Mining)
  - Apriori rule
  - FP-growth
  - Eclat algorithm
- Expectation Maximization
- Density Estimation
- …

Stochastic Gradient Descent (SGD)

Gradient Descent

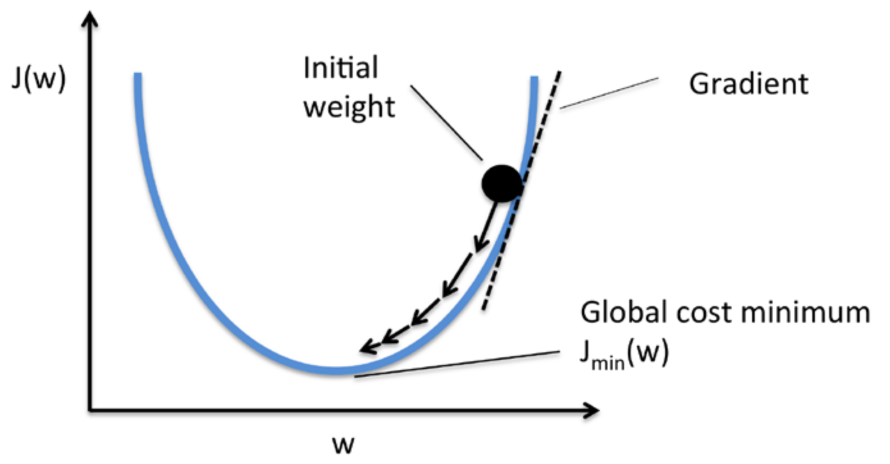
- **The method for parameter update**
- **Consider the model cost function $\boldsymbol{J(\theta)}$**

$$J(\theta) = \mathbb{E}_{x,y\sim\hat{p}_{data}} L(x,y,\theta) = \frac{1}{m}\sum_{i=1}^{m} L(x^{(i)},y^{(i)},\theta)$$

$$where, L(x,y,\theta) = -\log p(y|x;\theta)$$

- **Gradient of $\boldsymbol{J(\theta)}$ respect to $\boldsymbol{\theta}$ is:**

$$\nabla_\theta J(\theta) = g = \frac{1}{m}\nabla_\theta \sum_{i=1}^{m} L(x^{(i)},y^{(i)},\theta)$$



- **Update the new parameter $\boldsymbol{\theta_{new}}$**

$$\theta_{new} \leftarrow \theta - \epsilon g$$

$$where, epsilon\ \epsilon\ is\ the\ learning\ rate$$

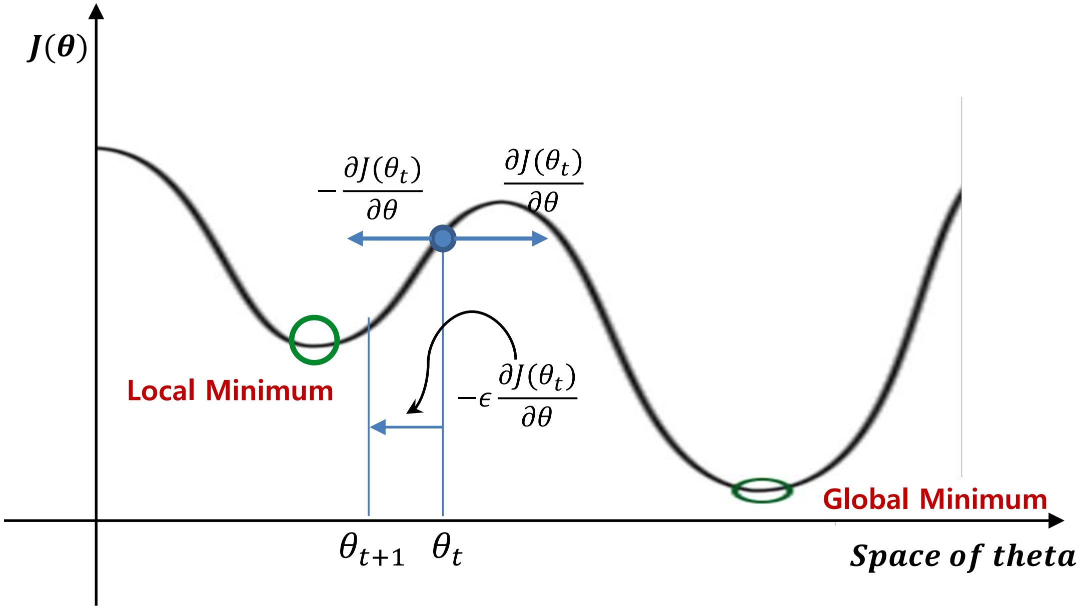
Limitation of Gradient Descent

- **Issue of local minimum**

$$Objective: \quad min \;\; J(\theta)$$

$$\theta_{t+1} = \theta_t - \epsilon \frac{\partial J(\theta)}{\partial \theta}$$

$(\epsilon : Learning\, rate)$



- **If the starting point for gradient descent was chosen inappropriately, cannot reach global minimum**

Stochastic Gradient Descent (SGD)

- **The SGD method**
  - Extension of gradient descent
  - Nearly all of deep learning is powered by this method
    (deep learning's cost space is not convex)
- **Using batch learning (= epoch learning)**
  - Calculate the loss function with batch(sample)

$$J_i(\theta) = L(x^{(i)}, y^{(i)}, \theta)$$

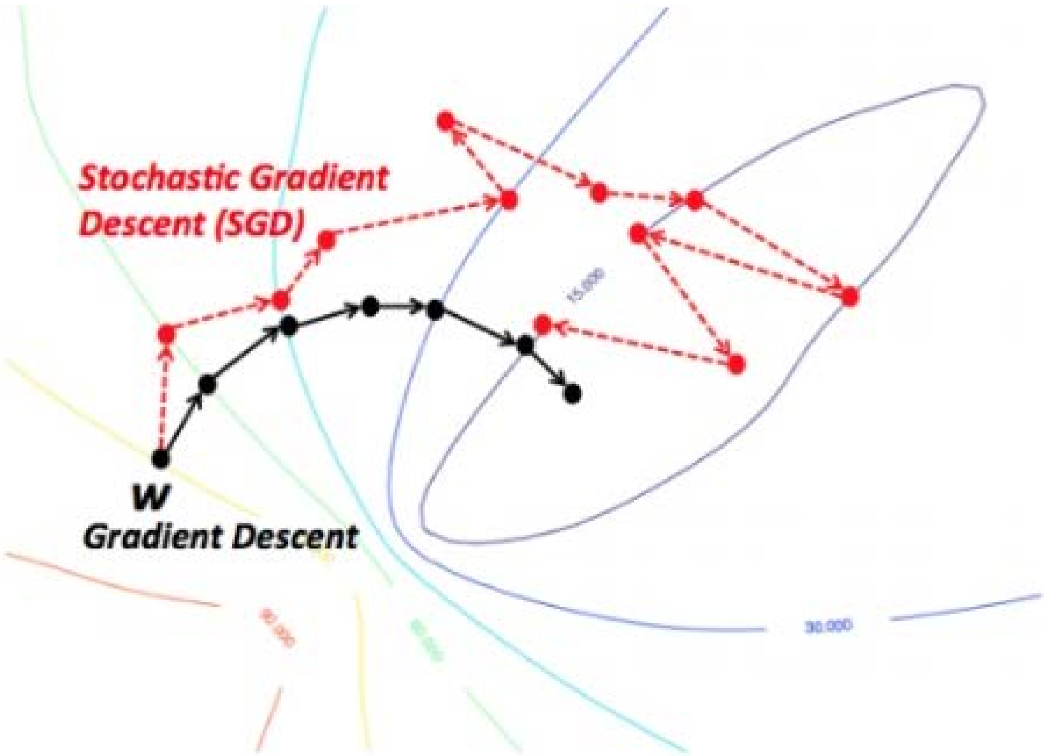
  - Update the new parameter θ_{new} with gradient of batch loss function

$$\theta_{new} \leftarrow \theta - \epsilon \nabla_{\theta} J_i(\theta)$$

  - At each update, loss function will be changed

SGD vs GD

- **GD goes in steepest descent direction, but slower to compute per iteration for large datasets**
- **SGD can be viewed as noisy descent, but faster per iteration**



** Slides from Veit-Trung TRAN(Hanoi Univ. of S&T), "From neural network to deep learning"*

The next Deep Learning Seminar

[Part 2] Deep Networks: Modern Practice

Chapter 6. Deep Feedforward Networks

Thank you

Any Questions?