Machine & Deep learning basics [AICS305]
Machine learning for cyber security II

고려대학교

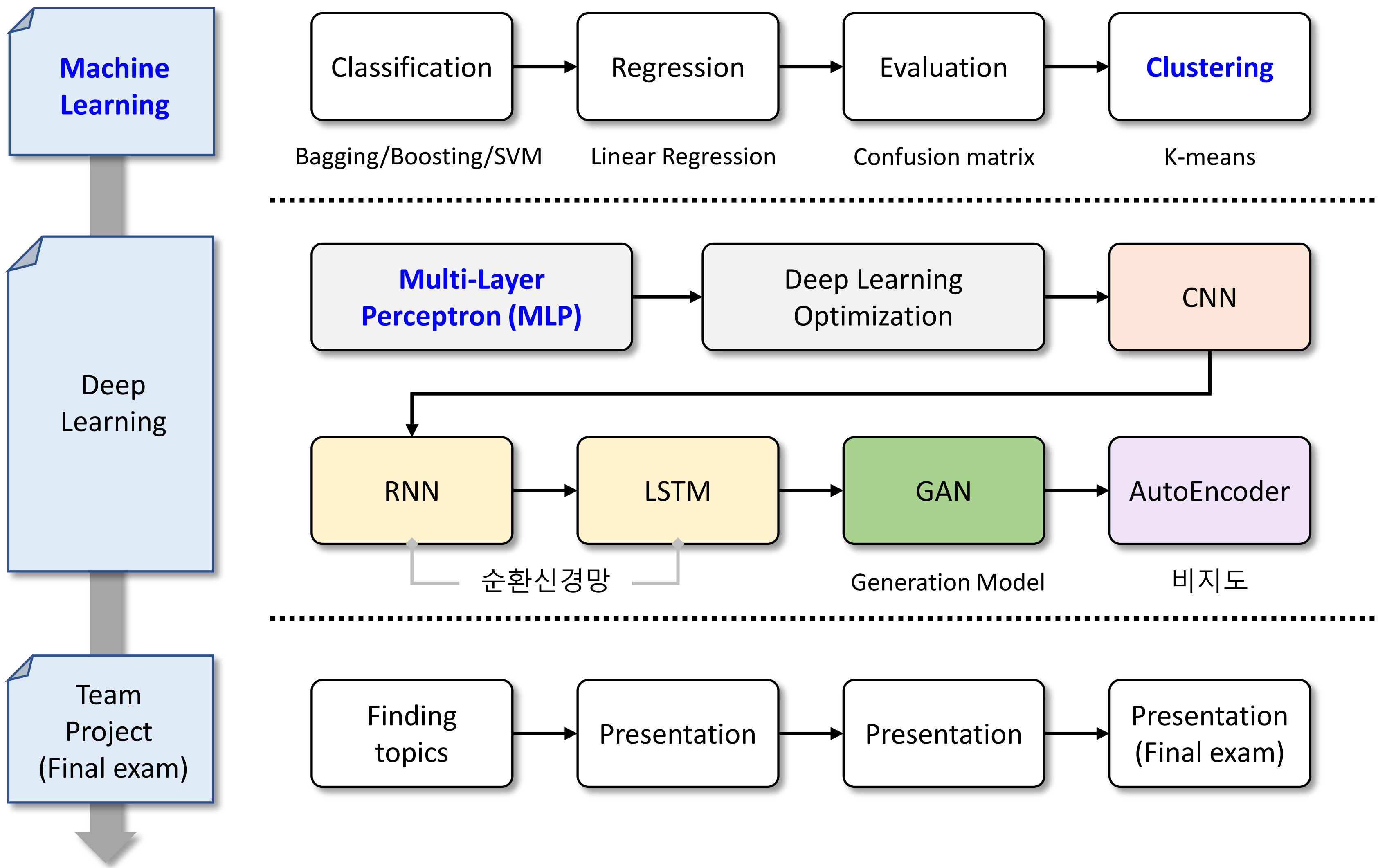
인공지능사이버보안학과

Prof. Mee Lan Han (aeternus1203@gmail.com)

KOREA UNIVERSITY

Machine Learning vs. Deep Learning

■Study Plan



CONTENTS

- **Clustering (Model)**
  - K-means
  - EM
  - DB Scan

Clustering

■ 군집화 (Clustering)

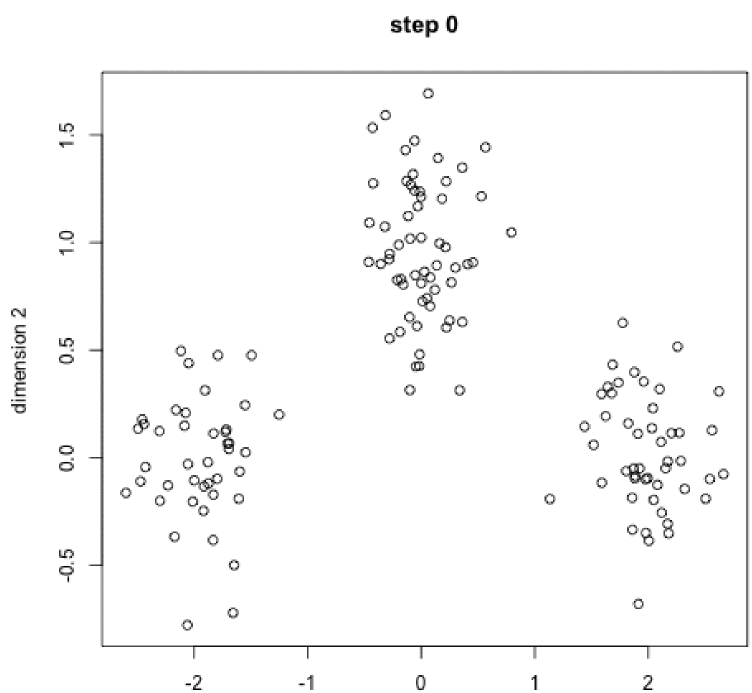
- □ 레이블이 지정 되어있지 않은 데이터를 학습하는 대표적인 알고리즘 -> 비지도학습
- □ 데이터들의 특성을 고려해 데이터 집단 (즉, 클러스터)을 정의
- □ 데이터 집단을 대표할 수 있는 **중심점**을 찾는 것으로 데이터 마이닝을 하는 방법

- □ Clustering algorithms
  - **K-means**
    - 각 클러스터의 중심과 데이터들의 평균 거리를 이용하여 k개의 클러스터로 데이터 학습
  - **Expectation-Maximization (EM)**
    - Expectation (기대값)과 Maximization (최대화) 으로 나뉘어 수렴할 때까지 반복하는 방식
  - **DBSCAN**
    - 밀도에 따라 클러스터링 하는 알고리즘

K-means algorithm

■ K-means

- □ 미리 정해 놓은 각 군집의 프로토타입에 각 객체가 얼마나 유사한가 (혹은 가까운가)를 측정하여 군집을 형성하는 기법
- □ K는 군집의 수 (number of clusters)
  - 클래스 (레이블, 정답지) 가 없는 데이터에서는 몇 개의 클로스터가 존재하는지 모름
  - K-평균 군집화 알고리즘에서는 분류할 클러스터의 수를 미리 정함



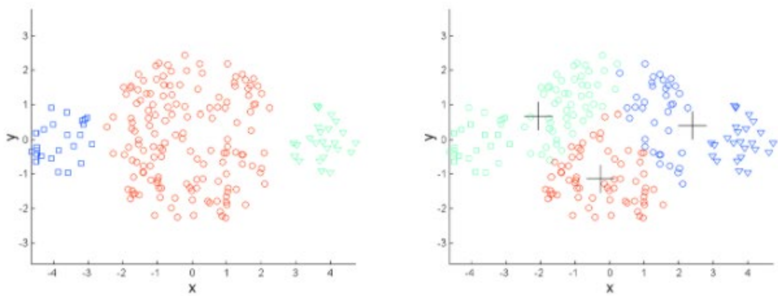
- **클러스터 (K) 설정 후 K에 따라 랜덤하게 center point를 잡게됨**
- **Data들은 각 center point끼리의 거리를 측정**
- **측정된 거리 벡터의 평균을 구하는 것으로 각 그룹의 center를 recompute함**
- **이 과정을 그룹 center point가 바뀌지 않을 때까지 반복**

[참고자료] https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68

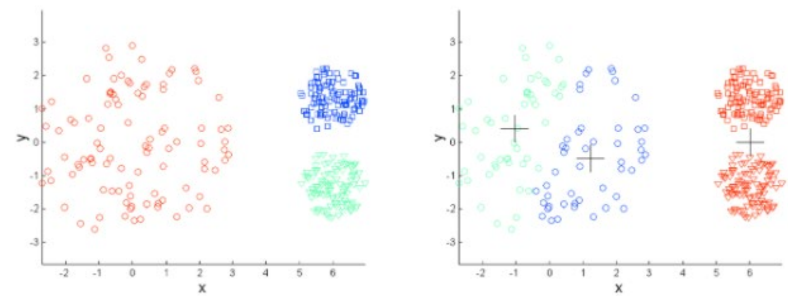
K-means algorithm

■ K-means

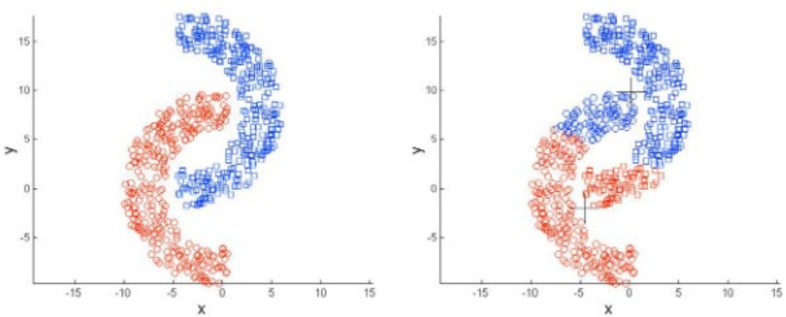
- 각 군집 중심의 초기값을 랜덤하게 정하기 때문에, 초기값 위치에 따라 원하는 결과가 나오지 않을 수도 있음

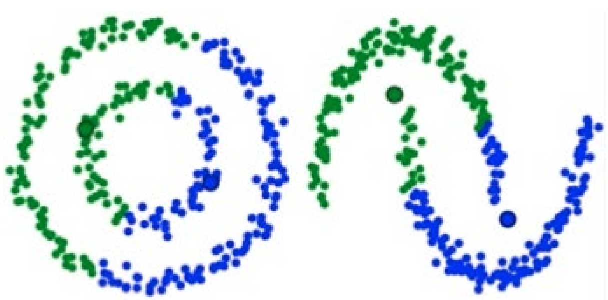


- 군집의 크기가 다를 경우 제대로 작동하지 않을 수 있음



- 군집의 밀도가 다를 경우 제대로 작동하지 않을 수 있음





- 데이터 분포가 특이한 케이스일 경우 제대로 작동하지 않을 수 있음

Expectation-Maximization (EM)

■ Expectation-Maximization (EM) using Gaussian Mixture Models

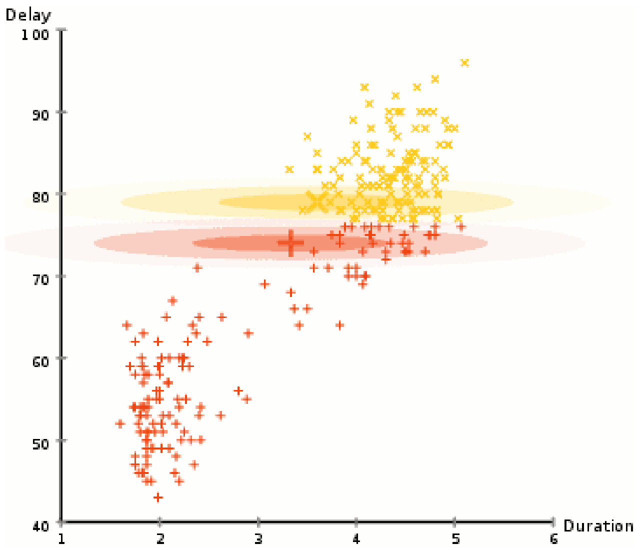
□ K-means는 cluster center를 알기 위해 평균 값을 naive하게 사용한다는 단점이 존재함

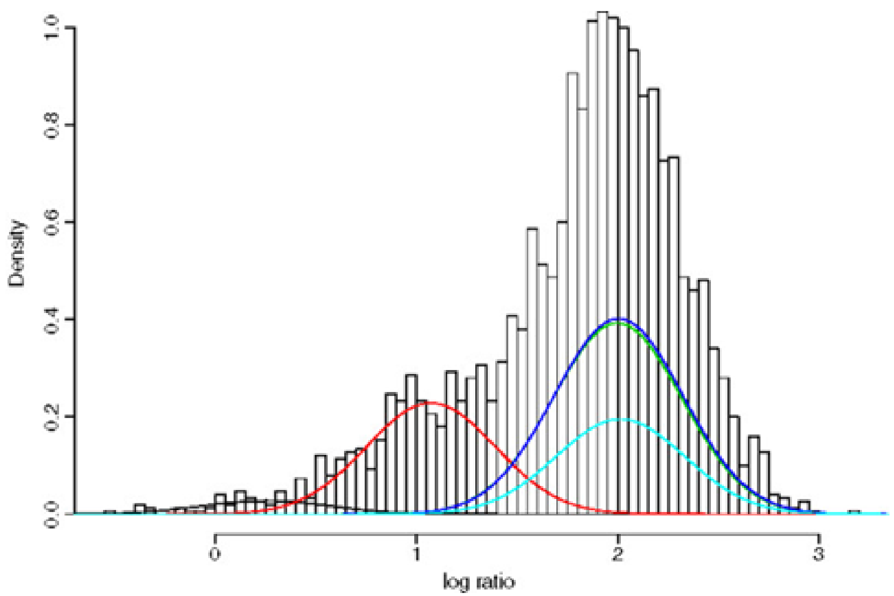
□ 이 전 페이지의 특이한 데이터 셋 분포의 경우 평균을 사용하는 원리로 인해 군집 실패!!

□ <u>**Gaussian Mixture Models (GMM)**</u>

- 혼합 모델은 통계학에서 전체 집단안의 하위 집단의 존재를 나타내기 위한 확률 모델
- 전체 데이터를 몇 개의 가우시안 분포로 표현할 수 있다고 가정하여 각 분포에 속할 확률이 높은 데이터로 군집을 형성하는 기법
- 각 클러스터를 위한 가우시안 파라메터 (e.g. 평균(μ), 분산(σ), <u>확률(π))</u> 사용

* 해당 데이터가 해당 하위집단에 속할 확률





[참고자료] https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68

Expectation-Maximization (EM)

■ Expectation-Maximization (EM) using Gaussian Mixture Models

□ 기댓값 최대화 알고리즘 -> **관측되지 않는 잠재변수에 의존하는 확률 모델**

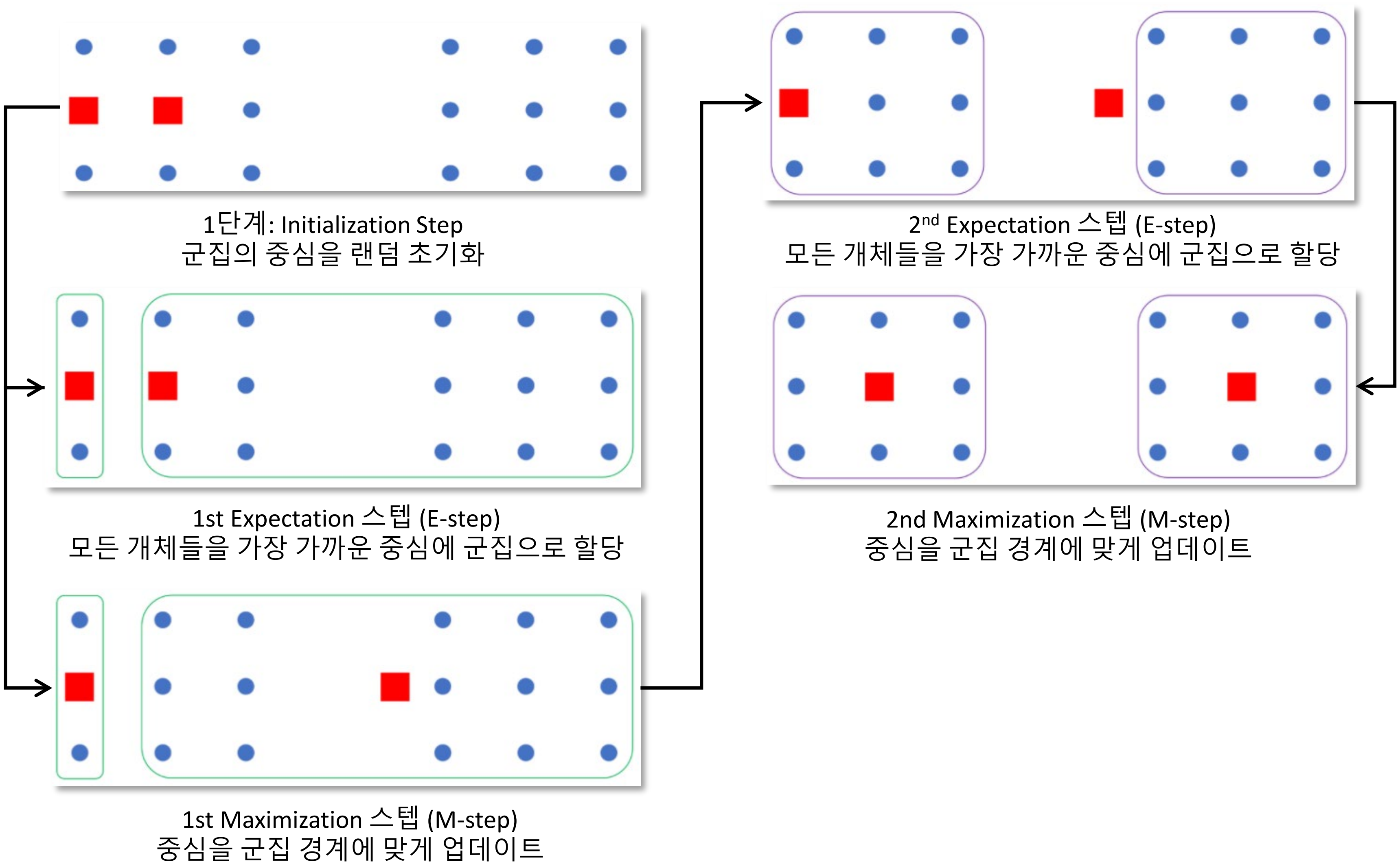
□ 최대우도값 (Likelihood function) 을 갖는 매개변수를 찾는 반복적인 알고리즘

- **1단계: Initialization Step**
  - 매개변수 θ를 임의의 값으로 초기화
- **2단계: Expectation Step (E-Step)**
  - 주어진 매개변수 값에 관한 잠재변수 z값을 추정
- **3단계: Maximization Step (M-Step)**
  - 2단계에서 얻은 잠재변수 값을 이용하여 매개변수 θ값을 다시 추정
- **4단계: Convergence Step**
  - 매개변수 θ값과 잠재변수 z값이 수렴할 때까지 2,3단계를 반복

□ E-step 과 M-step 을 반복 적용해도 결과가 바뀌지 않거나 (=해가 수렴), 사용자가 정한 반복 수를 채우게 되면 학습이 완성됨

Expectation-Maximization (EM)

- **Expectation-Maximization (EM) using Gaussian Mixture Models**
  - 군집 수 k=2설정

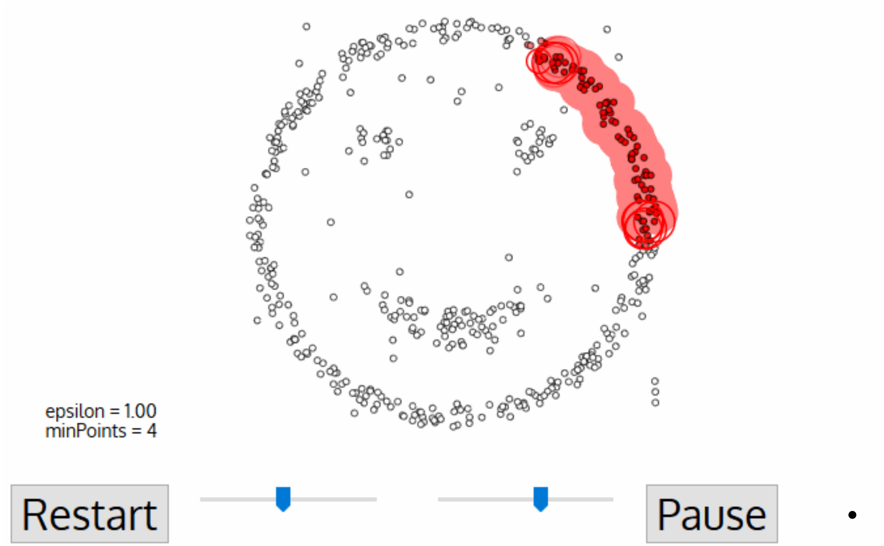


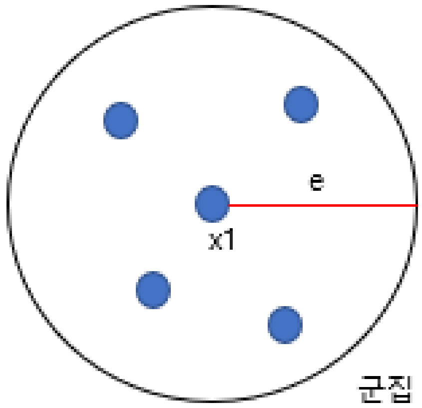
DBSCAN

■ Density-Based Spatial Clustering of Applications with Noise

- □ 밀도 기반 데이터 클러스터링 알고리즘
- □ 밀집되어 있는 지역을 하나의 군집으로 정의

- □ **DBSCAN의 핵심: <u>밀도가 높은 지역에 대한 정의</u>**
  - 밀도가 높은 지역을 정의하기 위해서 두 가지의 개념이 사용
    - **지정거리 (e, eps)** : 지정된 데이터 포인터를 기준으로 하여 군집을 탐색할 거리를 의미함
    - **데이터 개수 (n)** : 지정거리 이내 필요한 최소 데이터 개수를 의미





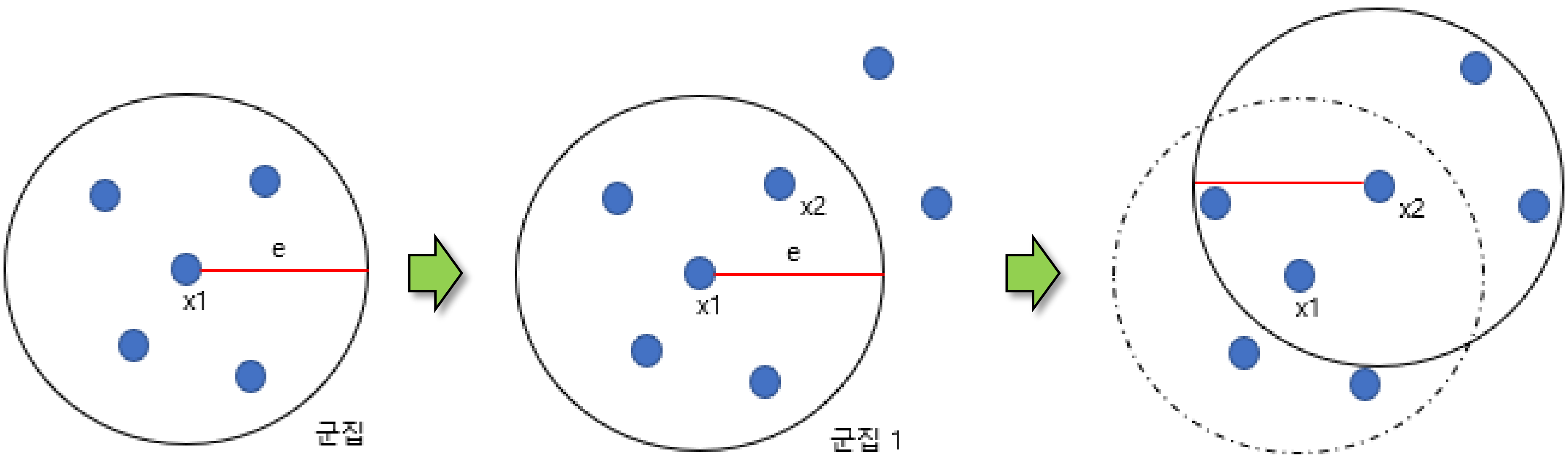
- **X_1 데이터 기준으로 e = 2, n = 5**
- **밀도가 높은 지역으로 선정되며 하나의 군집으로 할당**
- **X_1 은 core point로 정의됨**

[참고자료] https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68

DBSCAN

■ Density-Based Spatial Clustering of Applications with Noise

□ X_1 데이터 기준으로 e = 2, n = 5
□ X_1 은 이미 core point로 정의됨
□ X_1 을 core point를 갖는 밀도 높은 지역(군집 1) 안에 x2 데이터는 현재 core point의 요건을 만족함
→ X_2**를 기점으로 e 거리 안에 5개의 샘플 이상이 존재**

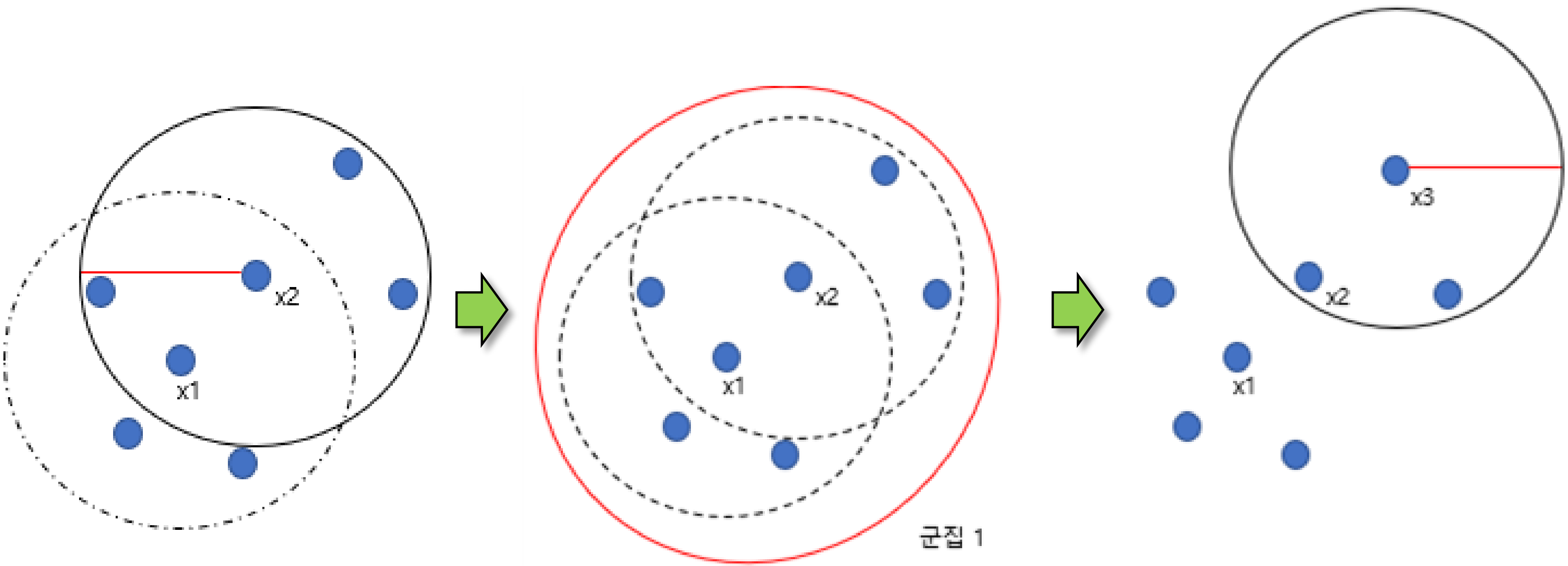


- X_1 core point로 하는 군집 안에 X_2 가 다른 데이터를 포함하여 core point가 됨
- 밀도가 높은 지역이 두 개가 형성되며, 이 경우 밀도가 높은 두 개의 지역이 서로 연결됨
- 처음에 할당했던 군집 1을 확장하여 밀도 높은 지역 두개를 포함하는 방식

DBSCAN

■ Density-Based Spatial Clustering of Applications with Noise

- □ 밀도가 높은 지역이 연결된 경우 군집을 확장하면서 군집화를 하는 기법
- □ 군집을 확장하면 어느 순간 확장을 멈추게 되는 지점이 존재함!!!
- □ 더 이상 연결된 밀도 높은 지역이 없는 상황 발생 → **core point가 없는 상황**

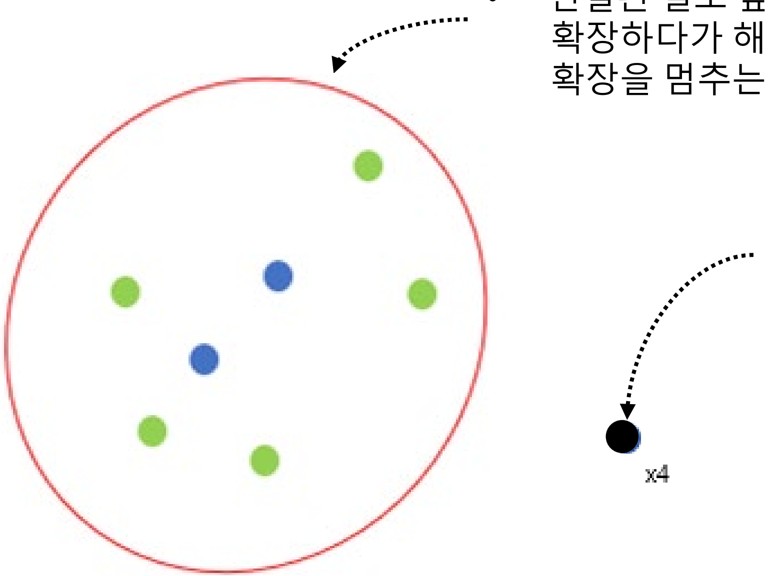


- X_3를 기점으로 e 거리 안에 5개의 샘플 이상이 존재하지 않음 (**밀도가 높은 지역을 형성할 수 없음**)
- e = 2, n = 5 만족하지 않음

DBSCAN

■ Density-Based Spatial Clustering of Applications with Noise

- X_3: Border point (경계 포인트)
- 연결된 밀도 높은 지역들을 통해 군집을 확장하다가 해당 경계 포인트에서 확장을 멈추는 지점



- X_4: Noise point (노이즈 포인트)
- core point, border point가 아닌 데이터로 어떤 군집에도 속하지 못한 데이터

① 거리 척도, e (지정 거리), n (필요 최소 샘플 수) 설정을 통해 밀도 높은 지역 정의

② 밀도 높은 지역을 만족하는 core point를 찾고 그 지역을 군집으로 할당

③ 해당 밀도 높은 지역 안에 core point를 만족하는 데이터가 있다면 그 지역을 포함하여 군집 확장

④ 해당 밀도 높은 지역 안에 더 이상 core point를 정의할 수 없을 때까지 2~3 단계 반복

⑤ 어떤 군집에도 해당되지 않은 데이터 noise point로 정의

Practicum

■ Clustering Practicum

□ K-means

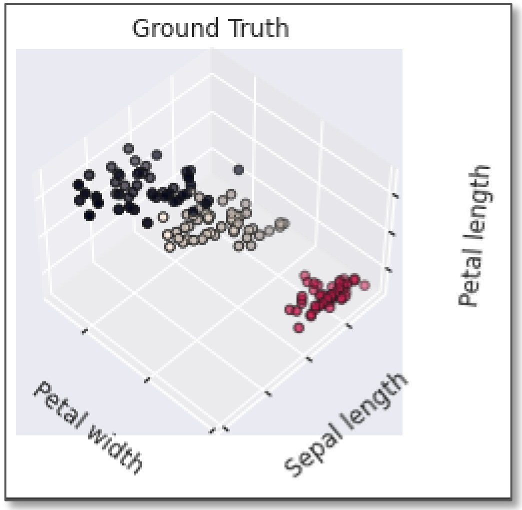
```
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

## sklearn 모델의 동일한 결과 출력 위해 선언
np.random.seed(5)

## 데이터 생성
df = pd.DataFrame(columns=['height', 'weight'])
df.loc[0] = [185,60]
df.loc[1] = [180,60]
df.loc[2] = [185,70]
df.loc[3] = [165,63]
df.loc[4] = [155,68]
df.loc[5] = [170,75]
df.loc[6] = [175,80]
```

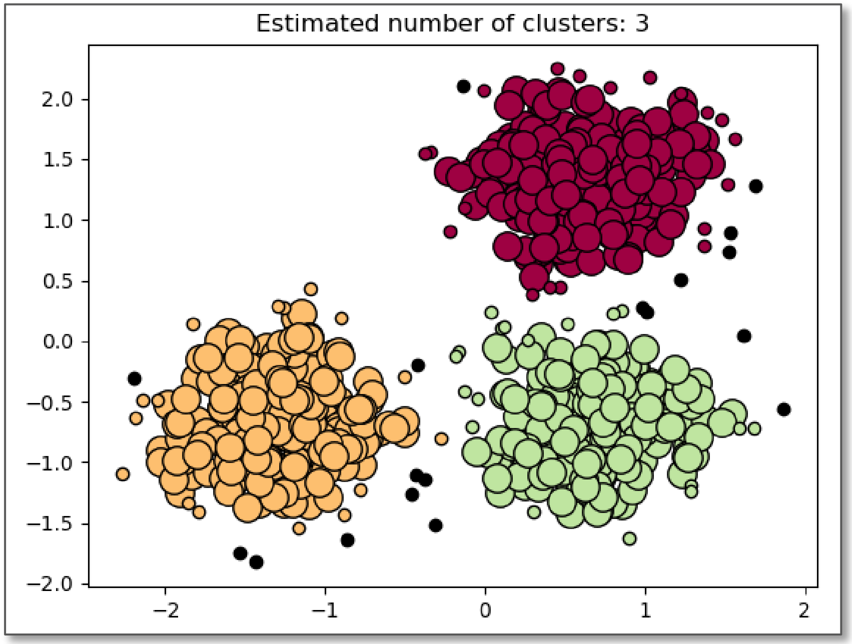
```
df.head(7)

## 데이터 시각화
sns.lmplot('height', 'weight',
           data=df, fit_reg=False,
           scatter_kws={"s": 200})

## K-means 군집화
data_points = df.values
kmeans = KMeans(n_clusters=3).fit(data_points)

## 각 군집의 중심 위치 확인
kmeans.cluster_centers_

## 데이터가 어느 군집에 소속되어 있는지 저장
df['cluster_id'] = kmeans.labels_

df.head(12)
```

Practicum

■ Clustering Practicum

□ K-means and DBSCAN





CONTENTS

- **Neural Network**
  - Neural Network
  - Perceptron
  - Multi-Layer Perceptron (MLP)
  - Activation function

Neural Network

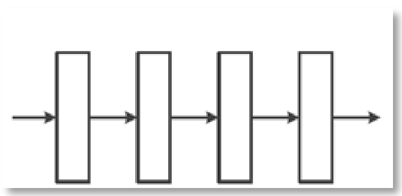
■ Neural Network

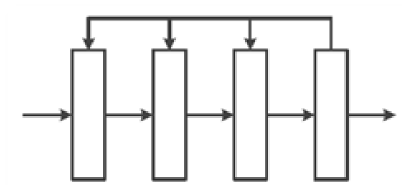
- □ 기계 학습 역사에서 가장 오래된 기계 학습 모델
- □ 현재 가장 다양한 형태를 가짐

퍼셉트론 -> 다층 퍼셉트론 -> 딥러닝 (기계학습의 주류 기술!!)

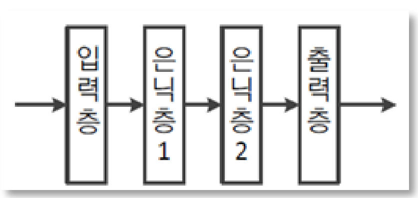
□ **신경망 모델**

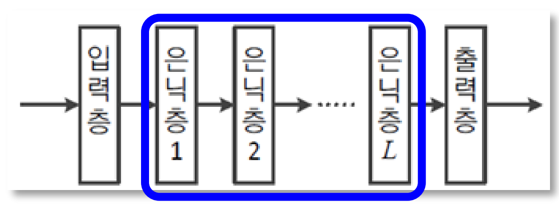
- 전방 신경망 (Feed-forward) 과 순환 (Recurrent)신경망





- 얕은 신경망 과 깊은 신경망





더 많은 은닉층을 가진 신경망

- 결정론 (Deterministic) 신경망과 확률적 (Stochastic) 신경망

Neural Network

■ Perceptron

□ 1세대 딥러닝

□ **<u>다수의 신호 (n개)를 입력으로 받아 특정한 연산을 거쳐 하나의 값을 출력하는 방식</u>**

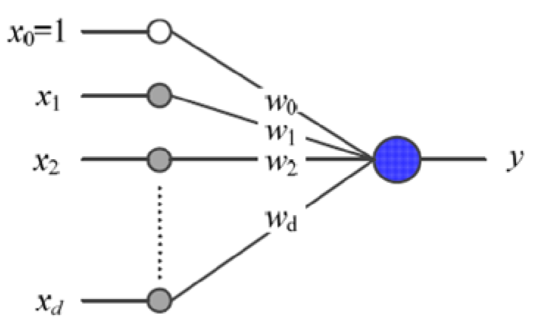
□ 입력층은 연산을 하지 않으므로 퍼셉트론은 단일 층 구조라고 간주

□ 출력층은 한 개의 노드

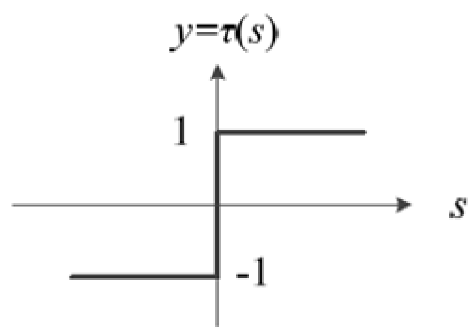
□ i번째 입력층 노드와 출력층을 연결하는 에지는 가중치 w_i를 가짐

□ 단점

- 간단한 xor 연산도 학습하지 못하는 문제 발생 (XOR 문제 ***75%가 정확률*** 한계)



$x_1, \ldots x_d$ (입력신호) $w_1, \ldots w_d$ (가중치) y (출력신호)



<u>계**단함수**</u>를 활성함수로 이용

$$y = \begin{cases} 0 & (w_1x_1 + w_2x_2 \le \boldsymbol{\theta}) \\ 1 & (w_1x_1 + w_2x_2 > \boldsymbol{\theta}) \end{cases}$$

참 or 거짓인지만 판단
(0 vs 1)

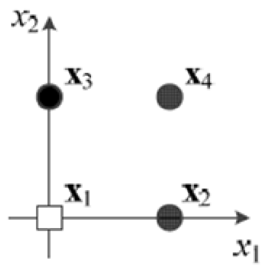
Neural Network

■ Perceptron 동작

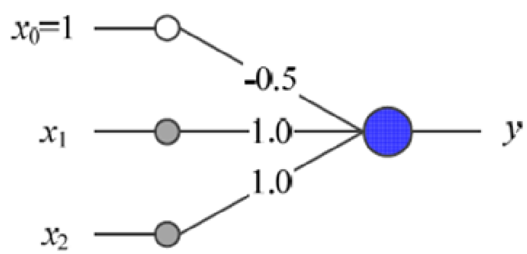
□ 2차원 특징 벡터로 표현되는 샘플 4개를 가진 훈련집합 X, Y

- $X = \{x_1, x_2, x_3, x_4\}$
- $Y = \{y_1, y_2, y_3, y_4\}$

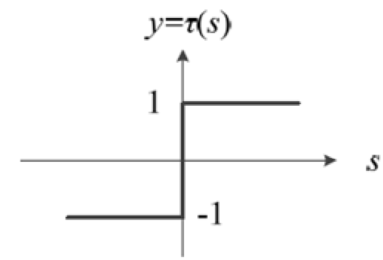
$$\mathbf{x}_1 = \begin{pmatrix}0\\0\end{pmatrix},\ y_1 = -1,\ \ \mathbf{x}_2 = \begin{pmatrix}1\\0\end{pmatrix},\ y_2 = 1,\ \ \mathbf{x}_3 = \begin{pmatrix}0\\1\end{pmatrix},\ y_3 = 1,\ \ \mathbf{x}_4 = \begin{pmatrix}1\\1\end{pmatrix},\ y_4 = 1$$



Training set



Perceptron



Activation function

✓ **OR논리 게이트를 이용한 퍼셉트론 동작**

$\boldsymbol{X}_1 : \boldsymbol{S} = -0.5 + 0 * 1.0 + 0 * 1.0 = -0.5$ $\qquad y = -1$

$\boldsymbol{X}_2 : \boldsymbol{S} = -0.5 + 1 * 1.0 + 0 * 1.0 = 0.5$ $\qquad y = 1$

$\boldsymbol{X}_3 : \boldsymbol{S} = -0.5 + 0 * 1.0 + 1 * 1.0 = 0.5$ $\qquad y = 1$

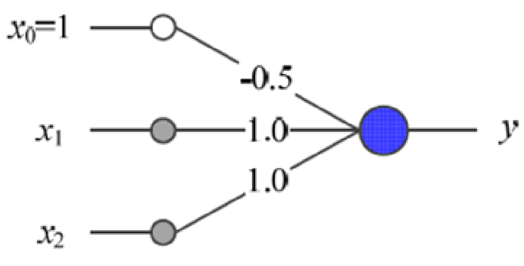
$\boldsymbol{X}_4 : \boldsymbol{S} = -0.5 + 1 * 1.0 + 1 * 1.0 = 1.5$ $\qquad y = 1$

Neural Network

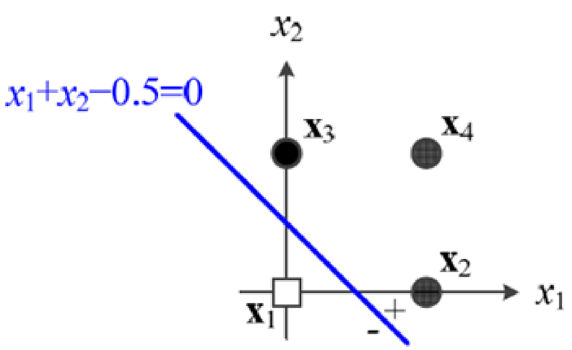
■ Perceptron 동작

- □ 퍼셉트론에 해당하는 결정 경계
- □ 결정 경계 $d(X) = d(x_1, x_2) = w_1x_1 + w_2x_2 + w_0 = 0\}$ → $x_1 + x_2 - 0.5 = 0$
  - w_1과 w_2는 직선의 방향, w_0은 절편을 결정
  - 결정 경계는 전체 공간을 +1과 -1의 두 부분공간으로 분할하는 분류기 역할

  - 2차원은 결정 경계
  - 3차원은 결정 평면
  - 4차원 이상은 결정 초평면



Perceptron

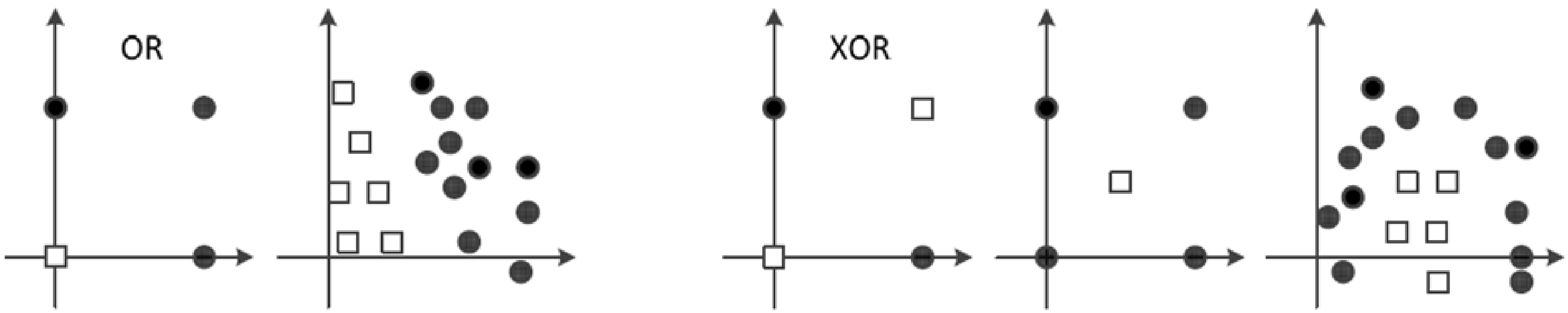


Neural Network

■ Multi-layered Perceptron

□ Perceptron 의 한계

- 선형분리가 가능한 상황과 불가능한 상황



(a) 선형분리 가능

(b) 선형분리 불가능

□ 민스키의『Perceptrons』

- 퍼셉트론의 한계를 지적하고 다층 구조를 이용한 극복 방안 제시 <u>-> 당시 기술로는 실현 불가능</u>
- 70년대 웨어보스는 박사 -> **<u>오류 역전파 알고리즘</u>** 제안
- 80년대 루멜하트 **<u>다층 퍼셉트론</u>** 이론 정립하여 신경망 부활

Neural Network

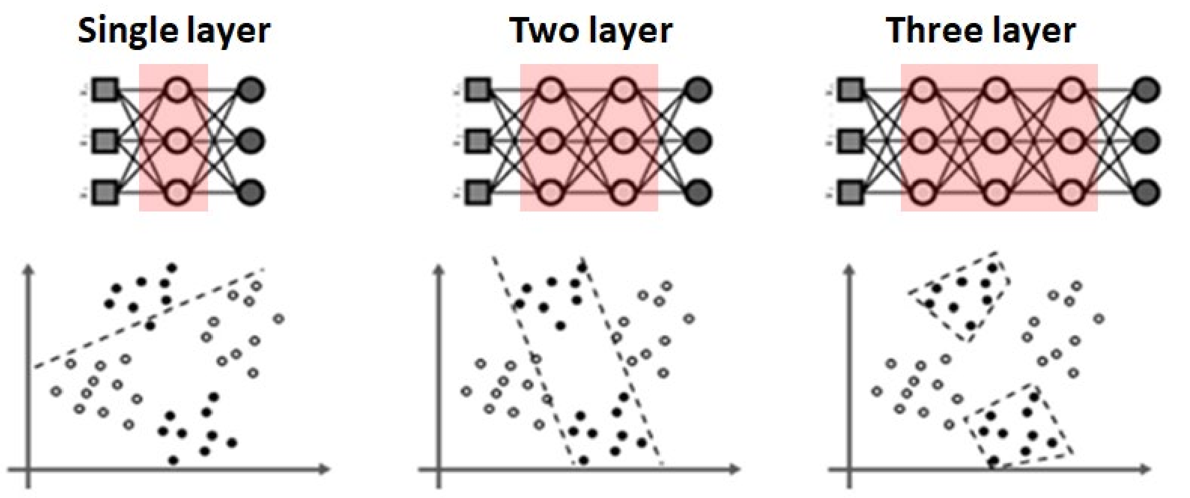
■ Multi-layered Perceptron

□ 2세대 딥러닝

□ 입력과 출력 사이에 하나 이상의 은닉층 (hidden layer)을 추가해 학습

□ **Key points!!**

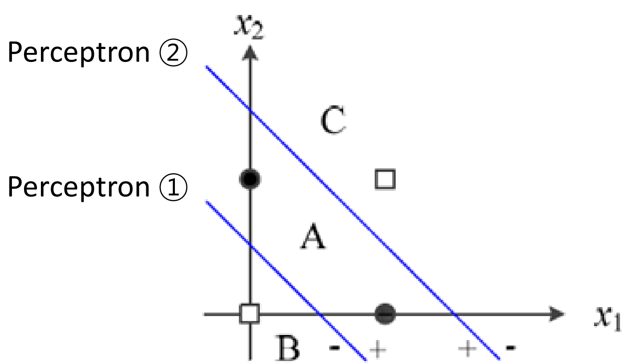
- **은닉층의 존재** - 특징 공간을 분류하는데 훨씬 유리한 새로운 특징 공간으로 변환시키는 역할
- **시그모이드 활성함수** – Soft decision-making 이 가능 (융통성 있는 의사결정 가능)
- **오류 역전파 알고리즘** 사용
  - 다층 퍼셉트론이 순차적으로 이어진 구조
  - 은닉층의 개수가 증가할수록 가중치의 개수도 증가 -> 학습이 어려움
  - 역방향으로 진행하면서 한 번에 한 층씩 그레이디언트를 계산하고 가중치를 갱신하는 방식 사용



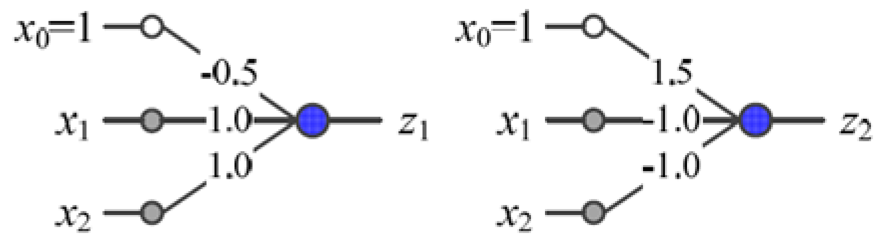
Neural Network

■ 특징 공간 변환

- □ Perceptron 2개를 사용한 XOR 문제의 해결
  - Perceptron ①과 Perceptron ②가 모두 +1이면 ● 부류이고 그렇지 않으면 □ 부류임



Perceptron 2개를 이용한 공간 분할



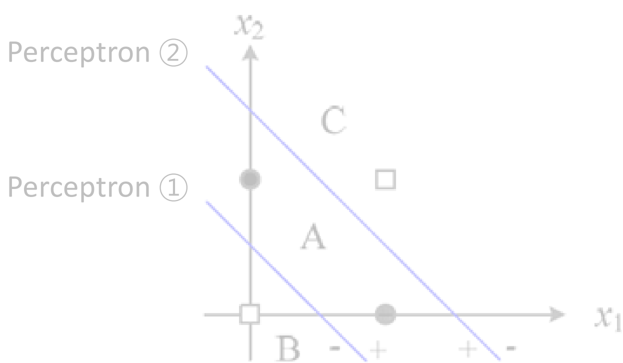
Perceptron ①　　Perceptron ②

Neural Network

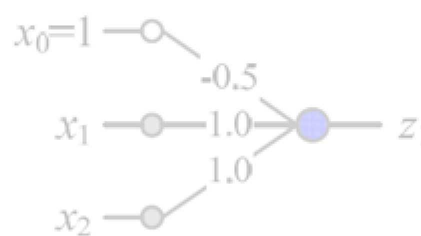
■ 특징 공간 변환

□ **Perceptron 2**개를 **병렬**로 결합

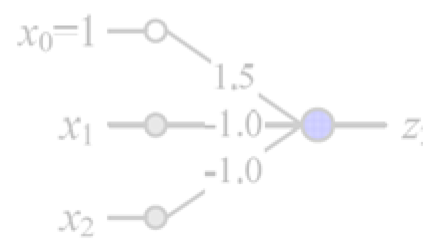
- 원래 공간 $X = (x_1, x_2)^T$를 새로운 특징 공간 $Z = (z_1, z_2)^T$로 변환
- 새로운 특징 공간 $\mathbf{z}$에서는 선형 분리 가능함



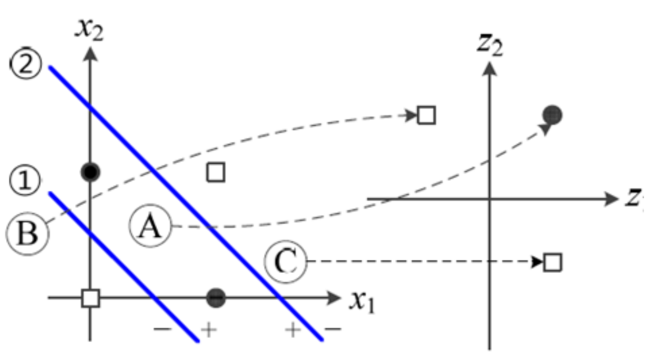
Perceptron 2개를 이용한 공간 분할



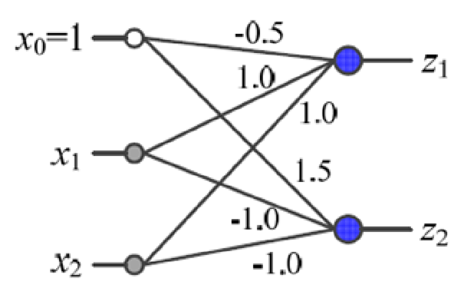
Perceptron ①



Perceptron ②



원래 특징 공간 x를 새로운 특징 공간 z로 변환



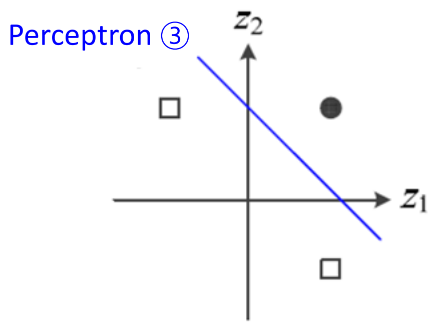
2개의 Perceptron을 병렬로 결합

Neural Network

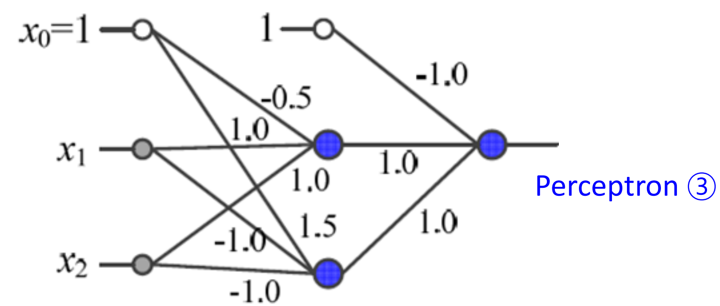
■ 특징 공간 변환

□ **Perceptron 1**개를 **순차**로 결합

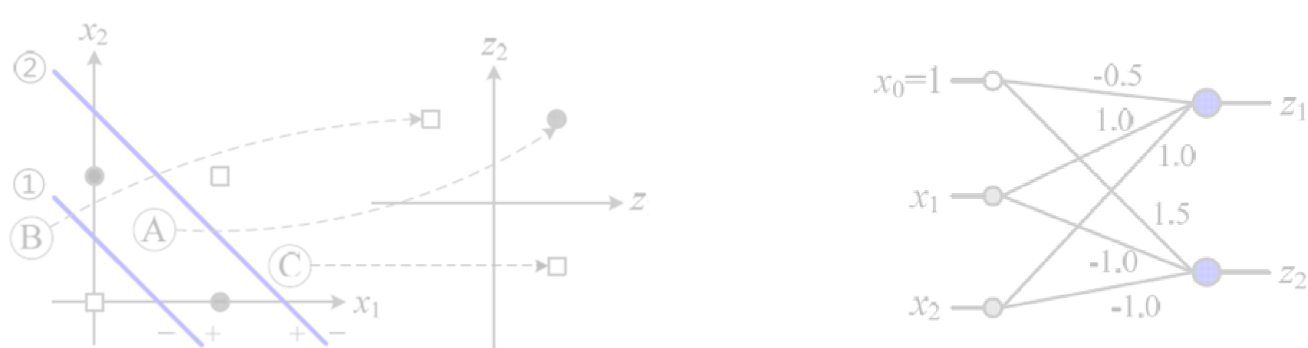
- 새로운 특징 공간 $Z = (z_1, z_2)$ 에서 선형 분리를 수행하는 퍼셉트론③ 을 순차 결합
- Multi-layered neural network 이 생성됨



새로운 특징 공간에서 분할



3개의 Perceptron을 결합한 다층 Perceptron



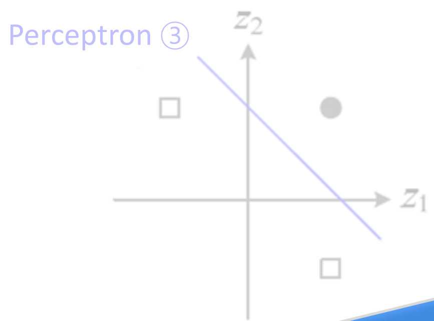
원래 특징 공간 x를 새로운 특징 공간 z로 변환

2개의 Perceptron을 병렬로 결합

Neural Network

■ 특징 공간 변환

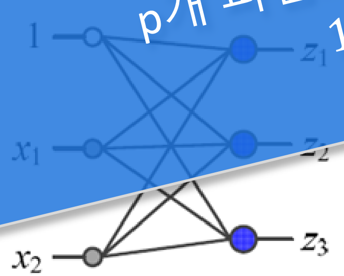
- □ **Perceptron 1**개를 **순차**로 결합
  - 3개 퍼셉트론을 결합하면, 2차원 공간을 7개 영역으로 나누고 각 영역을 3차원 점으로 변환
  - 활성함수 – 계단함수를 사용하여 영역을 점으로 변환

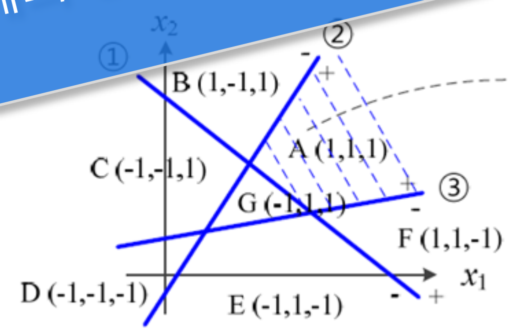


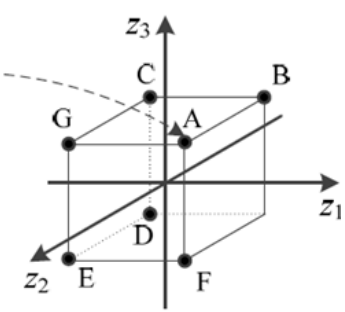
새로운 특징 공간에서 분할

Perceptron을 결합한 다층 Perceptron

p개 퍼셉트론을 결합하면 p차원 공간으로 변환
$1 + \sum_{i=1}^{p} i$개의 영역으로 분할







Perceptron 3개 결합

7개 부분 공간으로 나눔

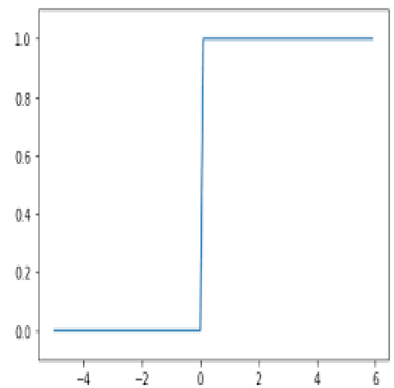
3차원 공간의 점으로 매핑

Neural Network

■ Activation function

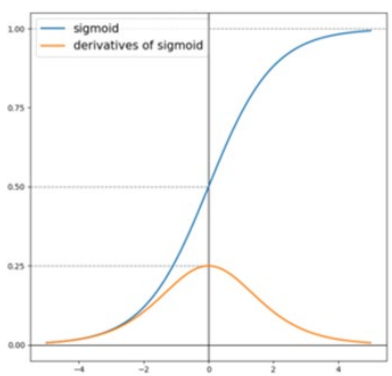
□ Hard & Soft 공간 분할

- 계단함수는 딱딱한 의사결정 (영역을 점으로 변환)

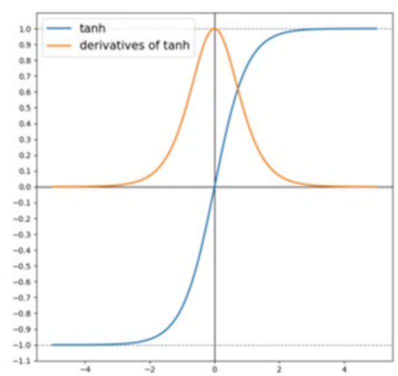


- 그 외 활성함수는 부드러운 의사결정 (영역을 영역으로 변환)

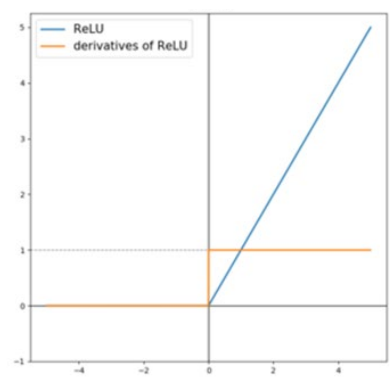
Sigmoid



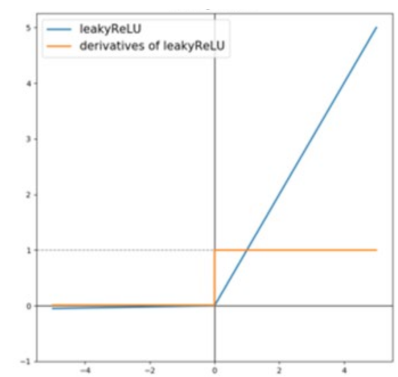
Tanh



ReLU



LeakyReLU

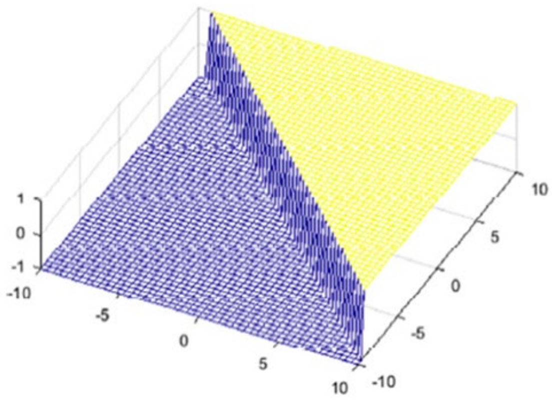


Neural Network

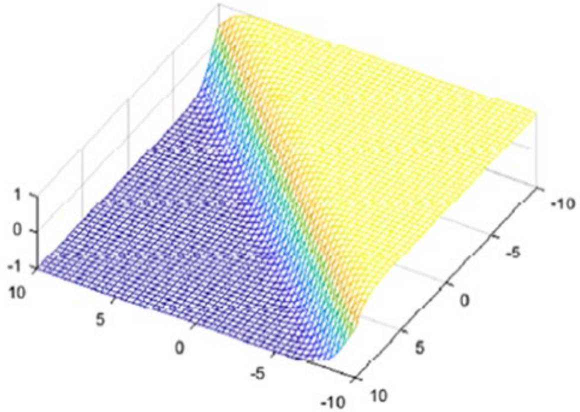
■ Activation function

□ Hard & Soft 공간 분할

- 계단함수는 딱딱한 의사결정 (영역을 점으로 변환)



- 그 외 활성함수는 부드러운 의사결정 (영역을 면으로 변환)



Neural Network

■ Activation function

□ **신경망이 사용하는 다양한 활성함수**

- 로지스틱 시그모이드와 하이퍼볼릭 탄젠트는 a가 커질수록 계단함수에 가까워짐
- 모두 1차 도함수 계산이 빠름 (특히 ReLU는 비교 연산 한 번)

Name	Function	1th a derived function	Range
$Step$	$r(s) = \begin{cases} 1 & s \geq 0 \\ -1 & s < 0 \end{cases}$	$r'(s) = \begin{cases} 0 & s \neq 0 \\ \text{불가} & s = 0 \end{cases}$	-1과 1
$Sigmoid$	$r(s) = \frac{1}{1 + e^{-as}}$	$r'(s) = ar(s)(1 - r(s))$	(0~1)
$Tanh$	$r(s) = \frac{2}{1 + e^{-as}} - 1$	$r'(s) = \frac{a}{2}(1 - r(s)^2)$	(-1~1)
$ReLU$	$r(s) = max(0, s)$	$r'(s) = \begin{cases} 0 & s < 0 \\ 1 & s > 0 \\ \text{불가} & s = 0 \end{cases}$	(0~∞)

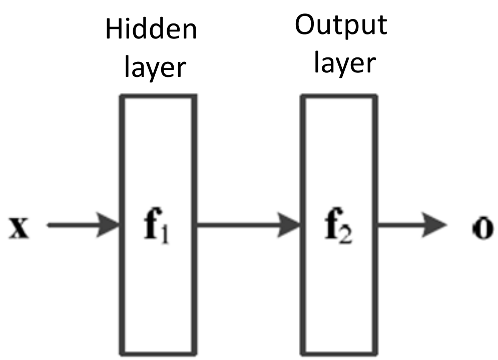
- 퍼셉트론: 계단함수
- 다층 퍼셉트론: 시그모이드와 하이퍼볼릭 탄젠트
- 딥러닝: ReLU

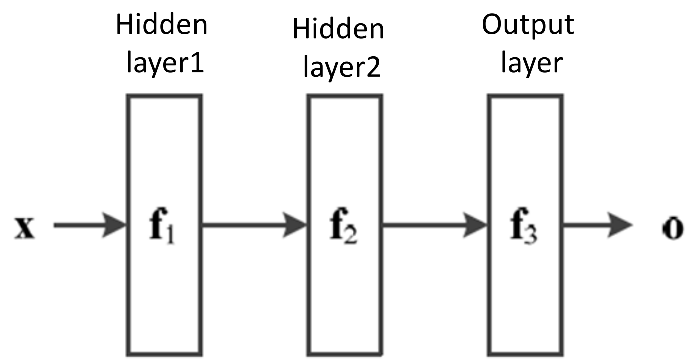
Neural Network

■ Architecture

□ 특징 벡터 x를 출력 벡터 **o**로 매핑하는 함수로 간주할 수 있음

- 2개 Perceptron: $\mathbf{o} = \boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{f_2}(\boldsymbol{f_1}(\boldsymbol{x}))$
- 3개 Perceptron: $\mathbf{o} = \boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{f_3}(\boldsymbol{f_2}(\boldsymbol{f_1}(\boldsymbol{x})))$

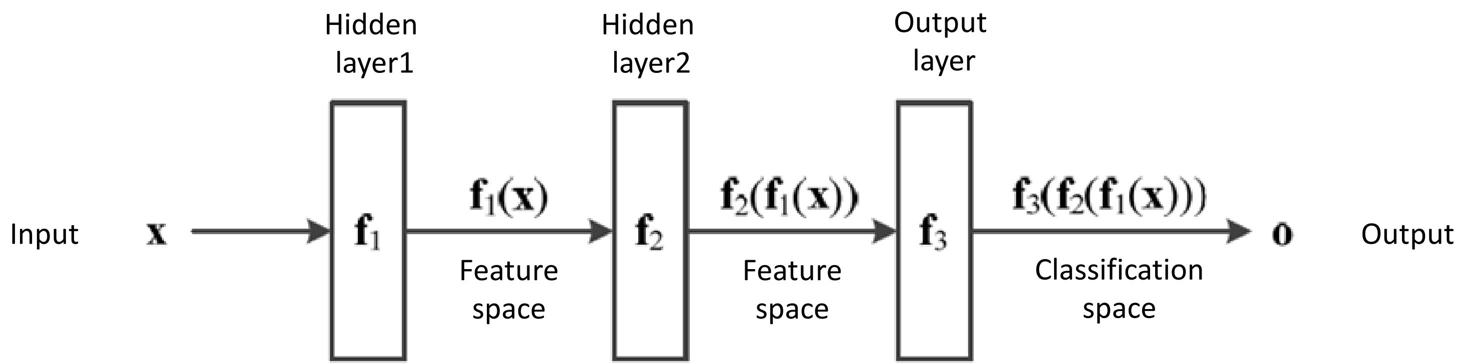




Neural Network

■ Architecture

- □ 은닉층은 특징 벡터를 분류에 더 유리한 새로운 특징 공간으로 변환시켜 줌
- □ 딥러닝은 더 많은 Hidden layer를 거쳐 특징학습 (feature learning)을 함



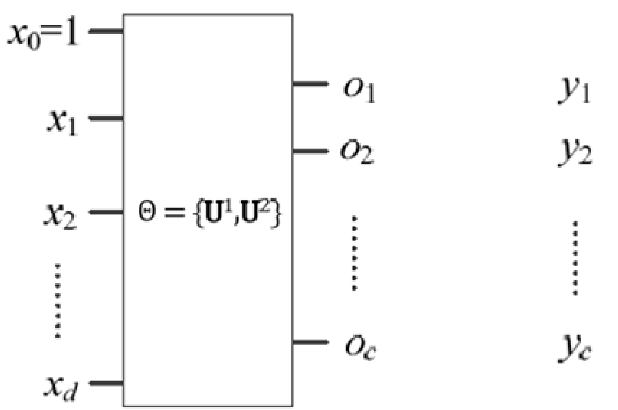
특징 추출기로써의 은닉층

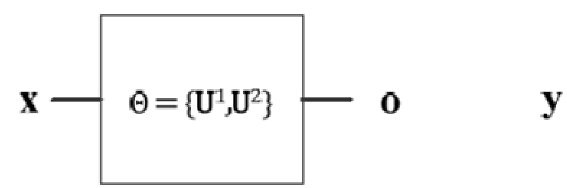
Neural Network

■ Goals of Machine Learning

- □ 모든 샘플을 옳게 분류하는 함수 f(x) 를 찾는 것을 목표로 함
  - $Y = f(X)$
  - $Y = f(X_i), i = 1,2,\ldots,n$
- □ Loss function MSE (mean squared error) 평균 제곱 오차로 정의
- □ 2개 Perceptron에 대한 가중치 행렬: $\Theta = \{\mathbf{U}^1, \mathbf{U}^2\}$

$$J(\Theta) = \frac{1}{n}\sum_{i=1}^{n}\|y - o(\Theta)\|^2$$





Input Vector | Output Vector | Classification Vector

Input Vector | Output Vector | Classification Vector

Neural Network

■ Goals of Machine Learning

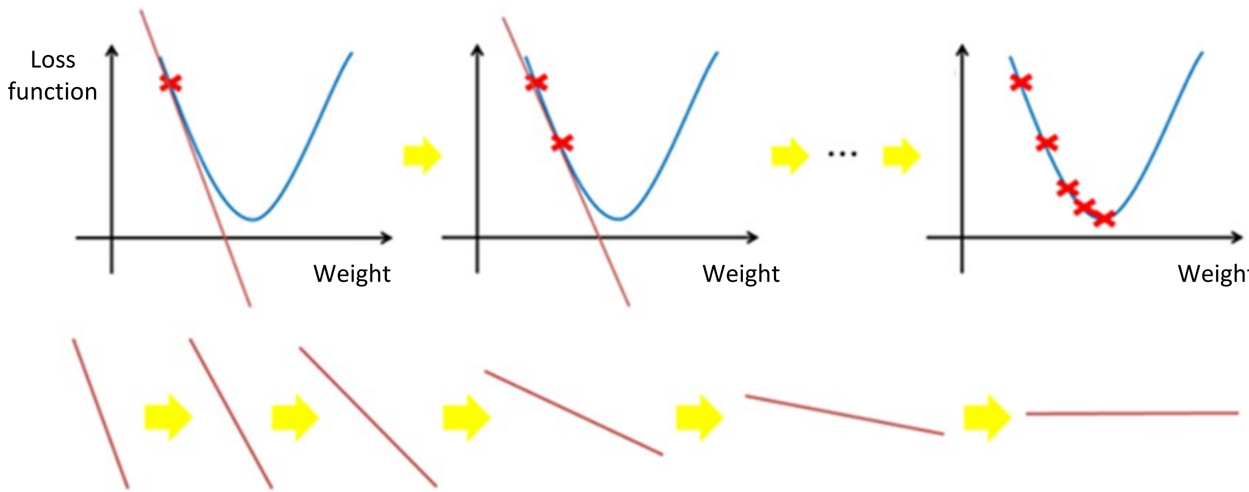
□ 가중치 행렬: $\Theta = \{\mathbf{U}^1, \mathbf{U}^2\}$

$$J(\Theta) = \frac{1}{n}\sum_{i=1}^{n} \|y - o(\Theta)\|^2$$

□ $J(\Theta) = J(\{\mathbf{U}^1, \mathbf{U}^2\})$의 최저점을 찾아주는 **경사하강법**

□ **경사하강법 (Gradient Descent)**

- 함수의 기울기(경사)를 구하여 기울기가 낮은 쪽으로 계속 이동시켜 극값 (최적값)에 이를 때까지 반복하는 과정
- 해당 함수의 최소값 위치를 찾기 위해 비용 함수 (Cost Function)의 경사 반대 방향으로 정의한 Step Size를 가지고 조금씩 움직여 가면서 최적의 파라미터를 찾으려는 방법
  학습률
- 경사는 파라미터에 대해 편미분한 벡터를 의미



Thank you

KOREA UNIVERSITY