

# 인공지능

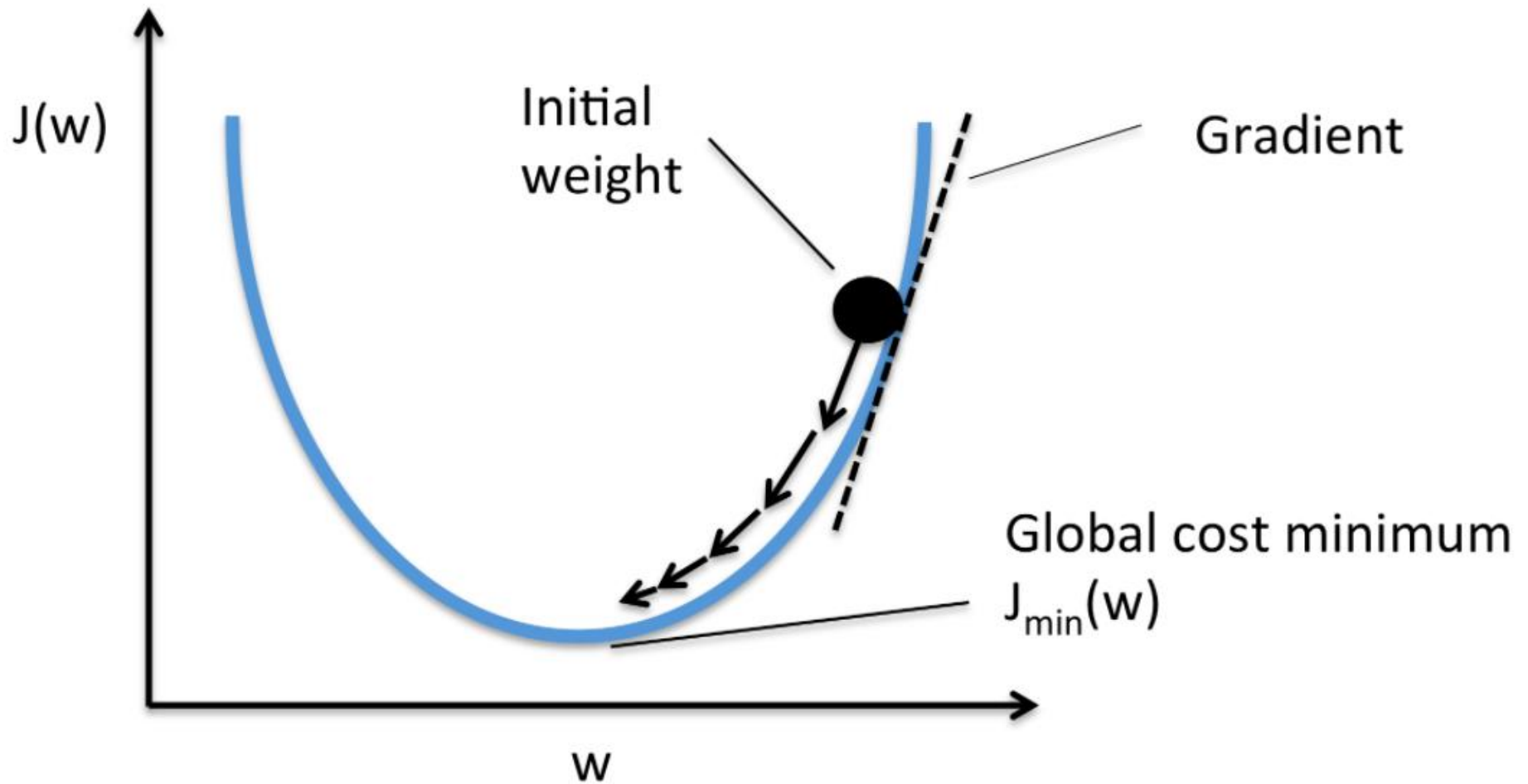
## [ 6. 비지도 학습 ]

소프트웨어융합대학  
소프트웨어학부

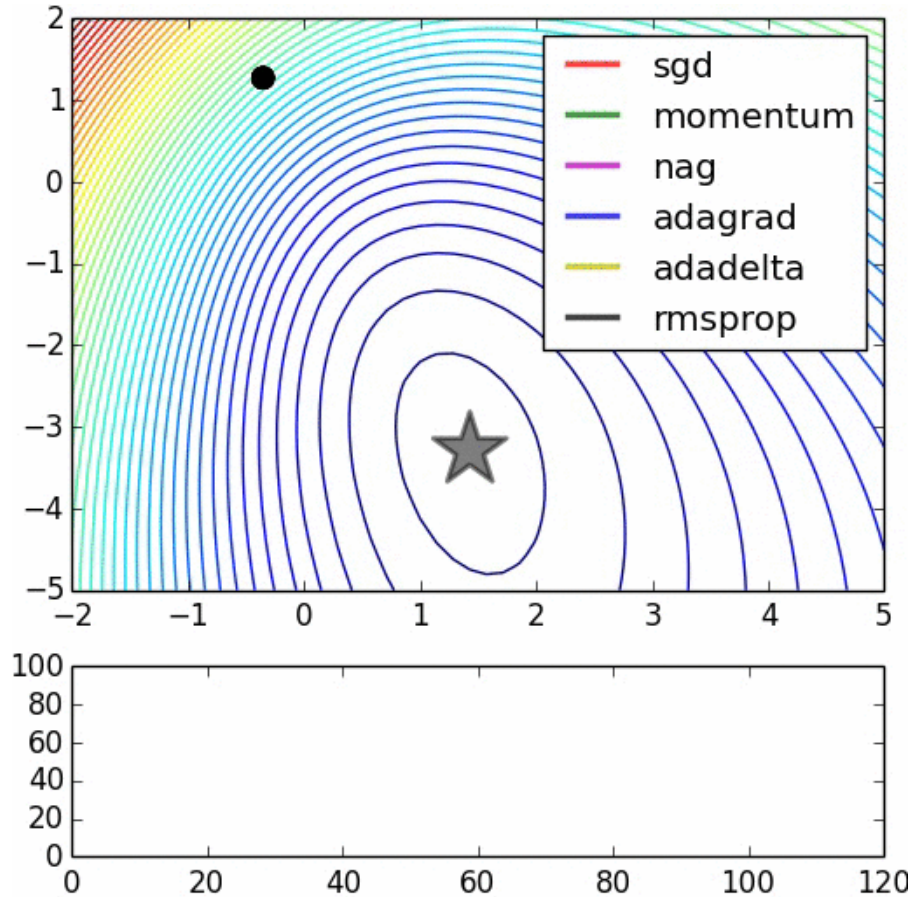
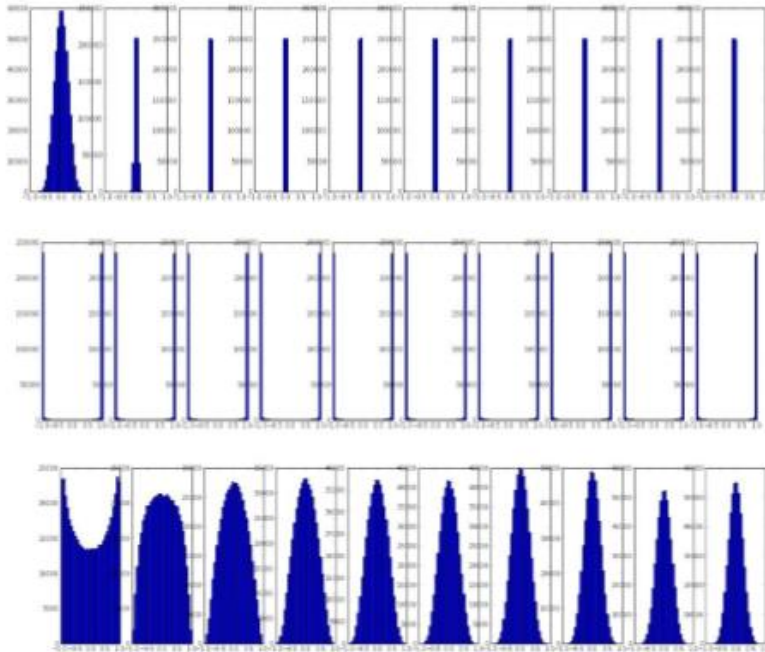
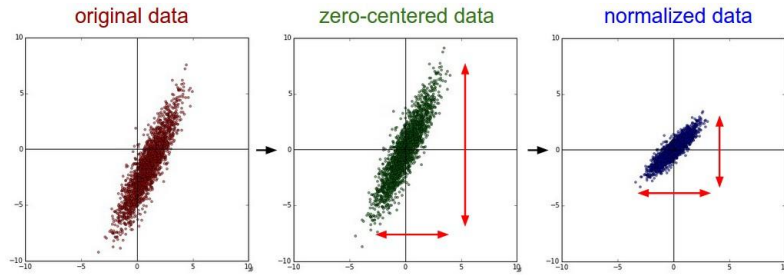
본 자료는 해당 수업의 교육 목적으로만 활용될 수 있음.  
일부 내용은 다른 교재와 논문으로부터 인용되었으며, 모든 저작권은 원 교재와 논문에 있음.

# 지난 장에는 뭐했지?

## 5.1. 목적함수

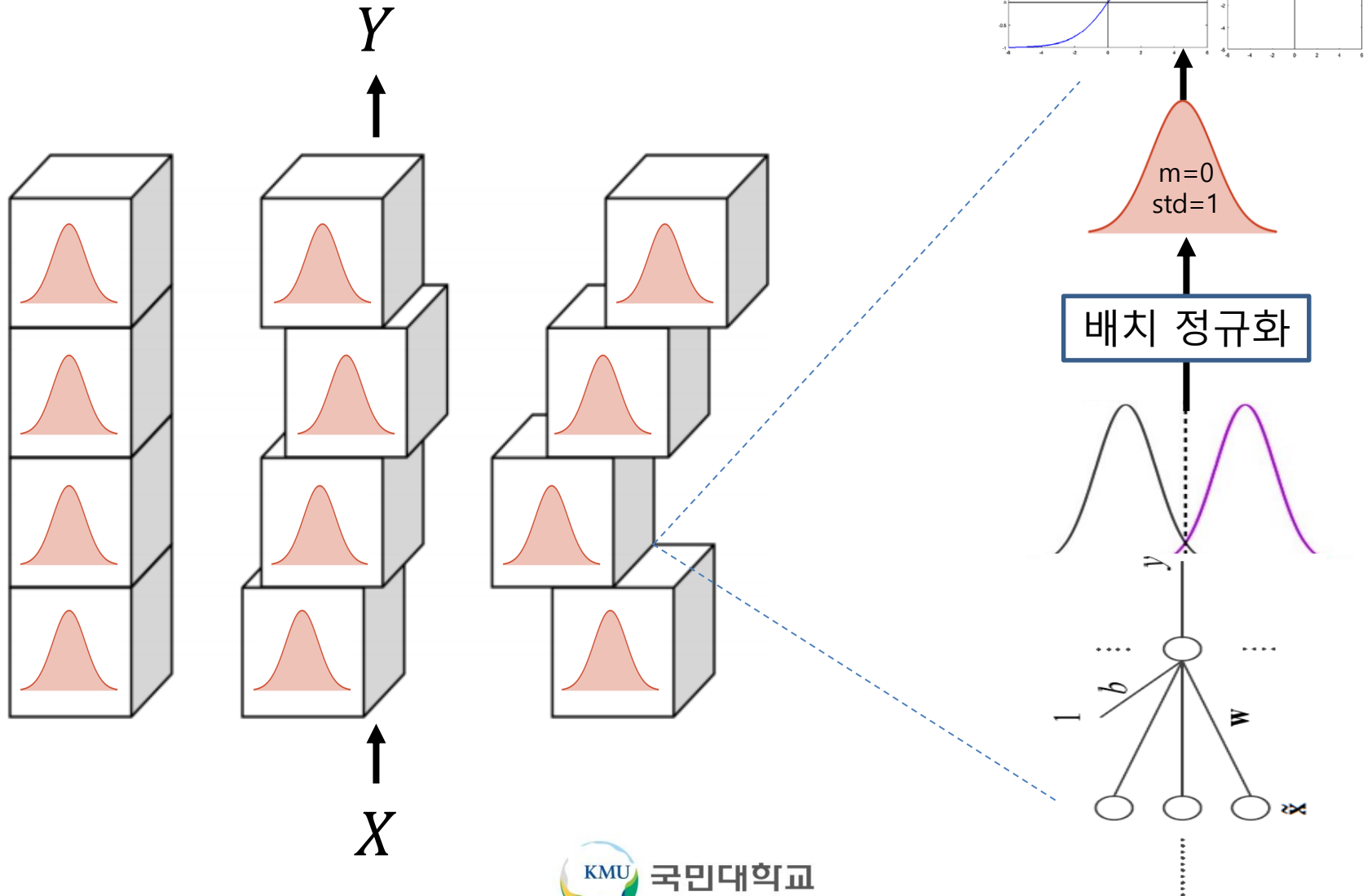


## 5.2. 성능 향상을 위한 조언

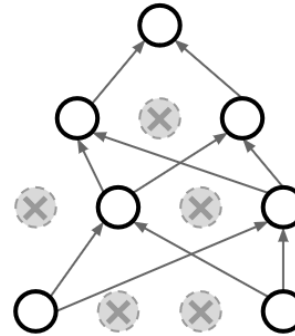
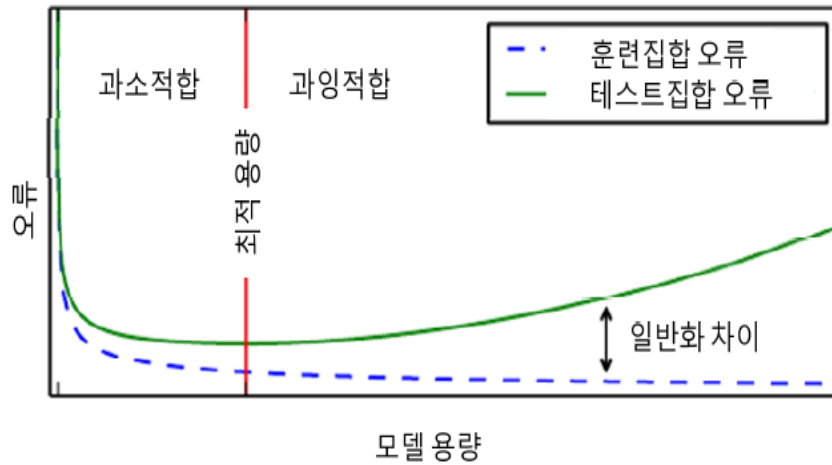


## 5.2. 성능 향상을 위한 조언

### ■ 공변량 시프트와 배치 정규화



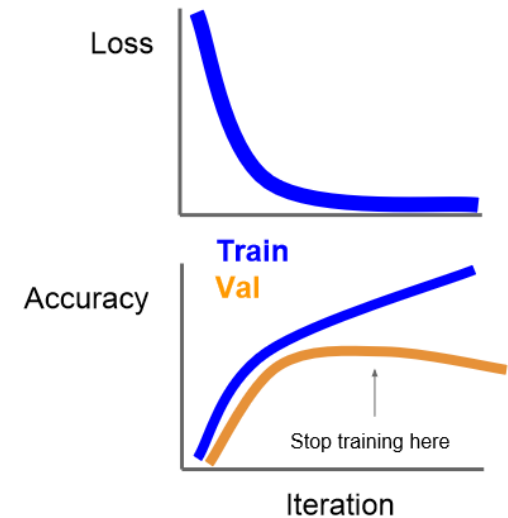
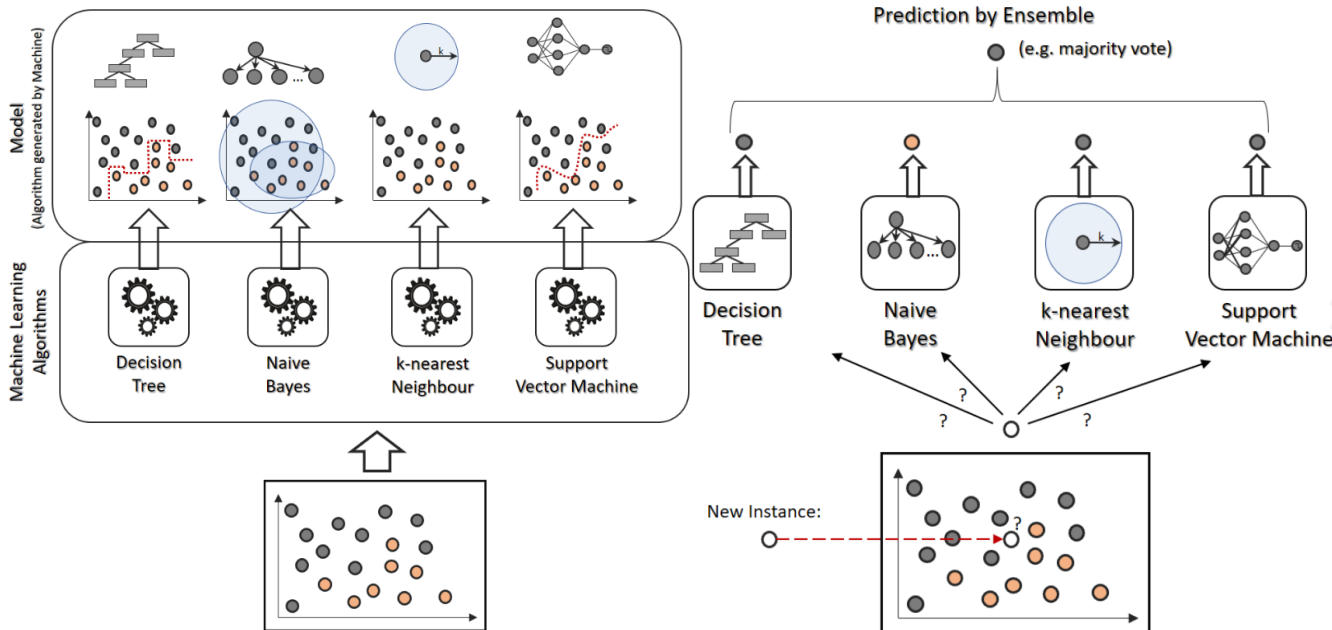
## 5.3. 5.4. 규제



Forces the network to have a redundant representation;  
Prevents co-adaptation of features



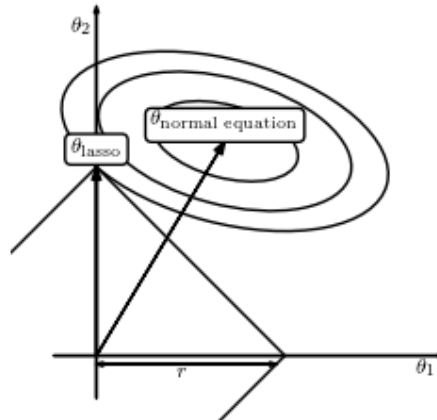
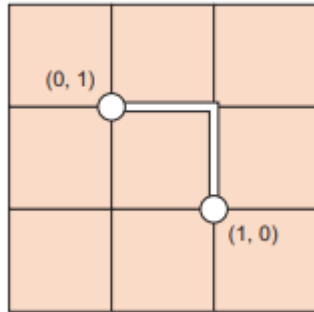
그림 5-18 학습 모델의 용량과 일반화 능력의 관계



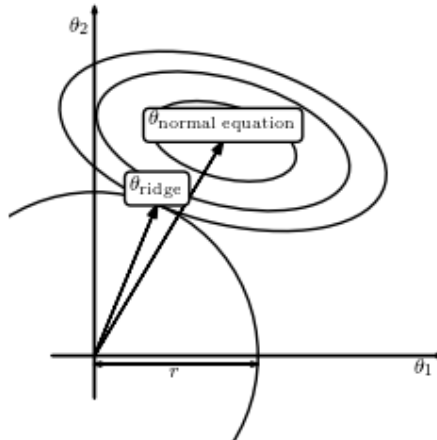
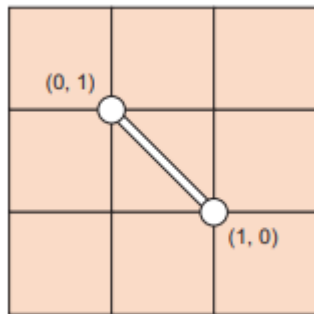
## 5.3. 5.4. 규제

$$\underbrace{J_{\text{regularized}}(\Theta; \mathbb{X}, \mathbb{Y})}_{\text{규제를 적용한 목적함수}} = \underbrace{J(\Theta; \mathbb{X}, \mathbb{Y})}_{\text{목적함수}} + \underbrace{\lambda R(\Theta)}_{\text{규제 항}}$$

### ■ L1 (lasso) 놈

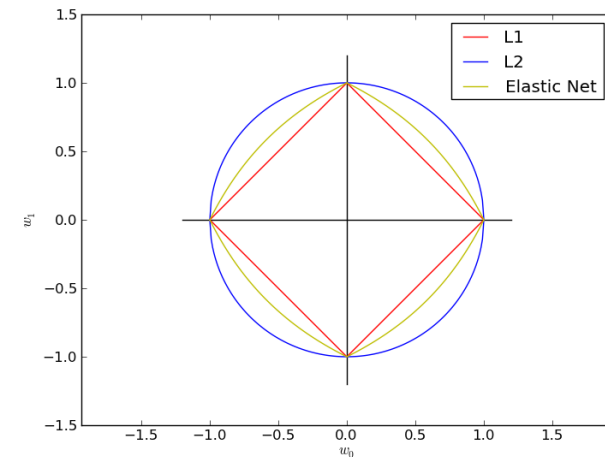


### ■ L2 (ridge) 놈

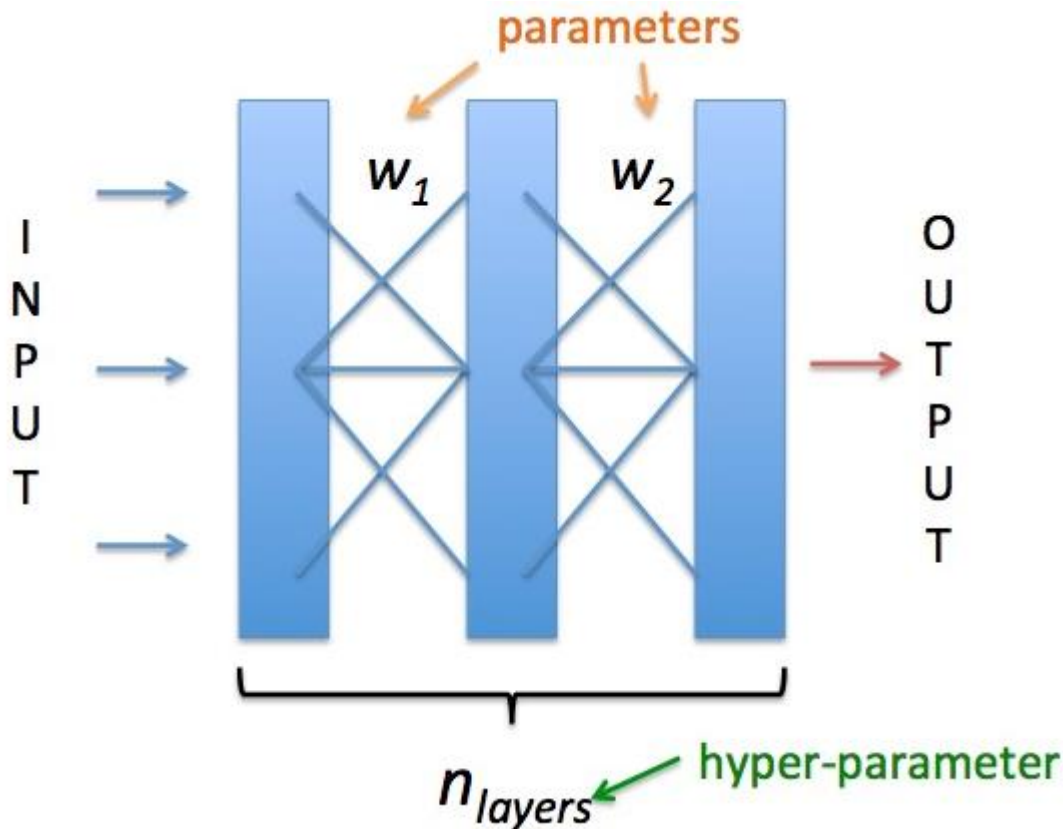


### ■ Elastic Net (L1+L2)

Regression with L2	Regression with L1
Not very robust	Robust
Stable solution	Unstable solution
Always one solution	Possibly multiple solutions
No feature selection	Built-in feature selection
Non-sparse outputs	Sparse outputs
Computational efficient due to having analytical solutions	Computational inefficient on non-sparse cases



## 5.5. 하이퍼 매개변수 최적화



### Coarse to fine search

```
val_acc: 0.412000, lr: 1.405200e-04, reg: 4.793504e-01, (1 / 100)
val_acc: 0.214000, lr: 7.231880e-06, reg: 2.321281e-04, (2 / 100)
val_acc: 0.208000, lr: 2.119571e-06, reg: 8.011057e+01, (3 / 100)
val_acc: 0.196000, lr: 1.551131e-05, reg: 4.374936e-05, (4 / 100)
val_acc: 0.079000, lr: 1.753300e-05, reg: 1.200424e+03, (5 / 100)
val_acc: 0.223000, lr: 4.215120e-05, reg: 4.196174e+01, (6 / 100)
val_acc: 0.441000, lr: 1.750259e-04, reg: 2.110007e-04, (7 / 100)
val_acc: 0.241000, lr: 6.749231e-05, reg: 4.226411e+01, (8 / 100)
val_acc: 0.482000, lr: 4.296863e-04, reg: 6.642555e-01, (9 / 100)
val_acc: 0.079000, lr: 5.401602e-06, reg: 1.599820e+04, (10 / 100)
val_acc: 0.154000, lr: 1.618500e-06, reg: 4.925252e-01, (11 / 100)
```

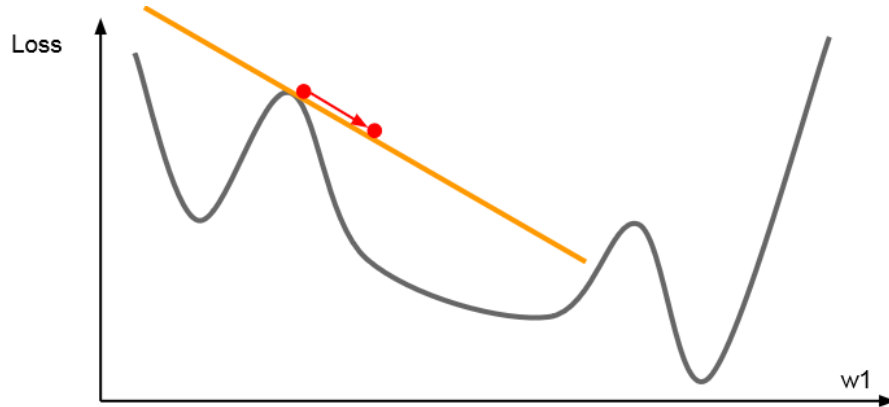
```
val_acc: 0.527000, lr: 5.340517e-04, reg: 4.097824e-01, (0 / 100)
val_acc: 0.492000, lr: 2.279404e-04, reg: 9.991345e-04, (1 / 100)
val_acc: 0.512000, lr: 8.600827e-04, reg: 1.349727e-02, (2 / 100)
val_acc: 0.461000, lr: 1.028377e-04, reg: 1.220193e-02, (3 / 100)
val_acc: 0.460000, lr: 1.113730e-04, reg: 5.244309e-02, (4 / 100)
val_acc: 0.490000, lr: 9.477776e-04, reg: 2.001293e-03, (5 / 100)
val_acc: 0.469000, lr: 1.484369e-04, reg: 4.328313e-01, (6 / 100)
val_acc: 0.522000, lr: 5.586261e-04, reg: 2.312685e-04, (7 / 100)
val_acc: 0.530000, lr: 5.800183e-04, reg: 8.259964e-02, (8 / 100)
val_acc: 0.489000, lr: 1.979168e-04, reg: 1.016889e-04, (9 / 100)
val_acc: 0.496000, lr: 2.836031e-04, reg: 2.406271e-03, (10 / 100)
val_acc: 0.475000, lr: 2.021162e-04, reg: 2.287807e-01, (11 / 100)
val_acc: 0.460000, lr: 1.135527e-04, reg: 3.905040e-02, (12 / 100)
val_acc: 0.515000, lr: 6.947668e-04, reg: 1.562888e-02, (13 / 100)
val_acc: 0.531000, lr: 9.471549e-04, reg: 1.433895e-03, (14 / 100)
val_acc: 0.509000, lr: 3.140080e-04, reg: 2.057510e-01, (15 / 100)
val_acc: 0.514000, lr: 6.438349e-04, reg: 3.033781e-01, (16 / 100)
val_acc: 0.502000, lr: 3.921784e-04, reg: 2.787126e-04, (17 / 100)
val_acc: 0.509000, lr: 9.752279e-04, reg: 2.050065e-03, (18 / 100)
val_acc: 0.500000, lr: 2.412948e-04, reg: 4.997821e-04, (19 / 100)
val_acc: 0.466000, lr: 1.319314e-04, reg: 1.189915e-02, (20 / 100)
val_acc: 0.510000, lr: 0.039527e-04, reg: 1.520291e-02, (21 / 100)
```



## 5.6. 경사 하강법 보완 기법

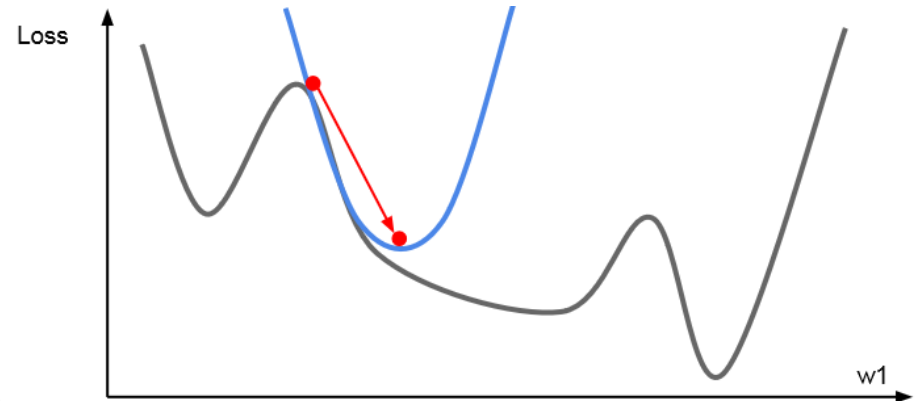
### 1차 미분의 최적화

- 그래디언트 사용하여 선형 근사 사용
- 이차 최소화 근사



### 2차 미분의 최적화

- 그래디언트와 헤시안을 사용하여 2차 근사 사용
- 이차 최소값 근사



# 오늘 수업에는 뭐하지?

# PREVIEW

## ■ 지도 학습과 비지도 학습

- 지금까지는 훈련집합으로  $\mathbb{X}$ 와  $\mathbb{Y}$ 가 주어지는 지도 학습 supervised learning
- 6장은  $\mathbb{X}$ 만 주어지는 비지도 학습 unsupervised learning

## ■ 다양한 응용

- 맞춤 광고, 차원 축소, 데이터 압축, 데이터 가시화, 특징 추출, 생성 모델 구축 등

## ■ 현대 기계 학습에서 더욱 중요해짐

# 각 절에서 다루는 내용

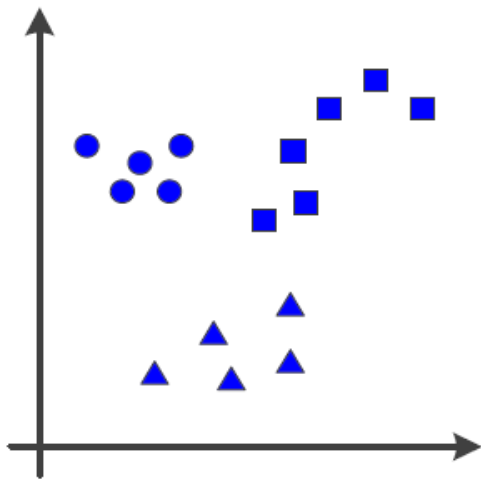
- 6.1절 비지도 학습을 지도 학습, 준지도 학습과 비교한다.
- 6.2절 비지도 학습의 일반 연산으로 군집화 밀도 추정 공간 변환을 소개한다.
- 6.3절 군집화 알고리즘으로  $k$ -평균과 친밀도 전파 알고리즘을 설명한다.
- 6.4절 밀도 추정 방법으로 커널 밀도 추정과 가우시안 혼합을 설명한다.
- 6.5절 기계 학습에서 공간 변환의 중요성을 강조한다.
- 6.6절 선형 인자 모델로서 PCA, ICA, 희소 코딩을 소개한다.
- 6.7절 오토인코더를 소개하고 규제 오토인코더로서 SAE, DAE, CAE를 설명한다.
- 6.8절 매니폴드 개념을 소개하고 IsoMap, LLE, t-SNE라는 매니폴드 학습 기법을 설명한다.



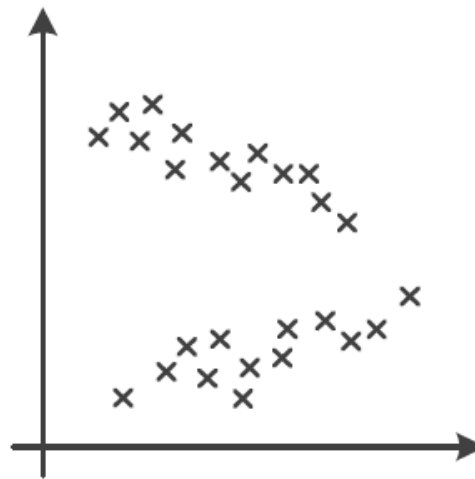
## 6.1 지도 학습과 비지도 학습, 준지도 학습

### ■ 세 가지 유형의 학습

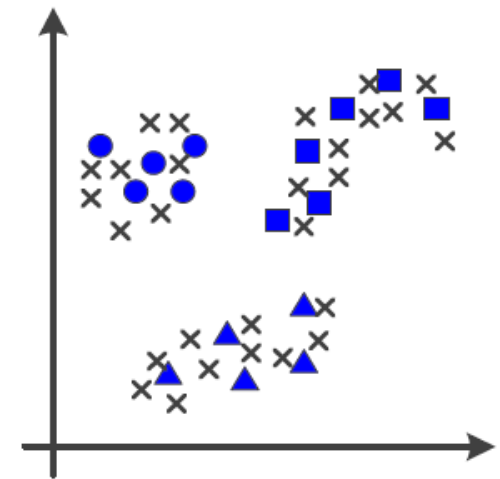
- **지도 학습** supervised learning: 모든 훈련 샘플이 레이블 정보를 가짐
- **비지도 학습** unsupervised learning: 모든 훈련 샘플이 레이블 정보를 가지지 않음 ← 6장의 주제
- **준지도 학습** semi-supervised learning: 레이블을 가진 샘플과 가지지 않은 샘플이 섞여 있음 ← 7장의 주제



(a) 지도 학습



(b) 비지도 학습



(c) 준지도 학습

**그림 6-1** 기계 학습의 유형(속이 찬 샘플은 레이블이 있고, x 표시된 샘플은 레이블이 없음)

## 6.1 지도 학습과 비지도 학습, 준지도 학습

### ■ 기계 학습이 사용하는 두 종류의 지식

- 훈련집합을 통한 전체 데이터에 대한 지식
- 사전 지식 prior knowledge (세상의 일반적인 규칙)

### ■ 중요한 두 가지 **사전 지식** prior beliefs

- **매니폴드 가정** manifold hypothesis: 데이터집합은 하나의 매니폴드 또는 여러 개의 매니폴드를 구성하며, 모든 샘플은 매니폴드와 가까운 곳에 있다. 매니폴드와 매니폴드 가정은 6.8.1절에서 자세히 설명한다.
- **매끄러움 가정** smoothness hypothesis: 샘플은 어떤 요인에 의해 변화한다. 예를 들어, 장면과 카메라 위치를 고정한 상태에서 조명을 조금씩 변화하면서 영상을 획득한 경우, 획득된 영상 샘플은 특징 공간에서 위치가 조금씩 바뀔 것이다. 이때 [그림 6-1(b)]와 같이 매끄러운 곡면을 따라 위치가 변한다.  
(=local constancy prior)

### ■ 비지도 학습과 준지도 학습은 **사전 지식을 더 명시적으로 사용**

- 매니폴드와 매끄러움 사전 지식에 의해

우리가 학습하는 함수는 작은 영역 내에서는 많이 변하지 않음을 가정함

$$f(x) \approx f(x + \epsilon), \epsilon: \text{작은 변화}$$

- 예로 좋은 입력들의 경우, 정답이 같으면 유사한 입력 특징 공간에 존재할 확률이 높음

## 6.2 비지도 학습

- 6.2.1 비지도 학습의 일반 과업
- 6.2.2 비지도 학습의 응용 과업

## 6.2.1 비지도 학습의 일반 과업

### ■ 세 가지 일반 과업

- **군집화**clustering: 유사한 샘플을 모아 **같은 그룹으로 묶는 일**
- **밀도 추정**density (probability mass function) estimation: 데이터로부터 **확률분포를 추정하는 일**
- **공간 변환**space transformation: 원래 특징 공간을 **저차원 또는 고차원 공간으로 변환하는 일**

### ■ 데이터에 내재한 구조를 잘 파악하여 새로운 정보를 발견해야 함

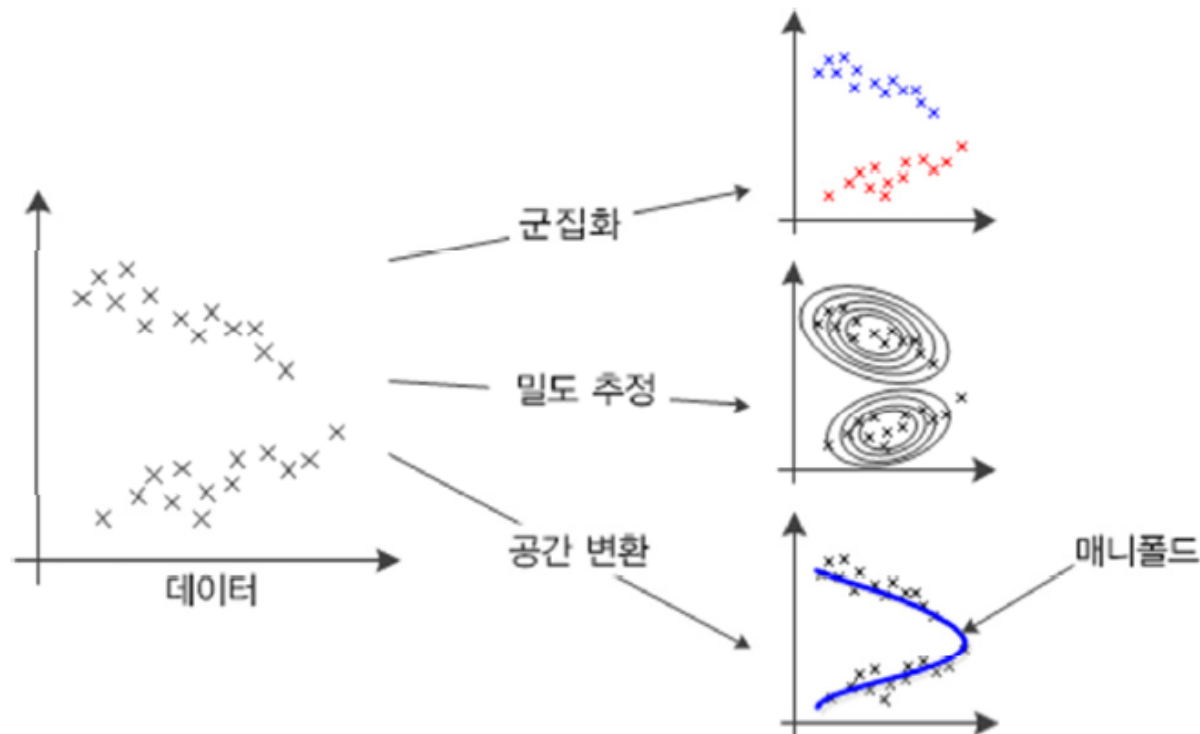


그림 6-2 비지도 학습의 군집화, 밀도 추정, 공간 변환 과업이 발견하는 정보



## 6.2.2 비지도 학습의 응용 과업

### ■ 아주 많은 응용 (서로 밀접하게 관련됨)

#### ■ 군집화의 응용

- 영상 분할
- 유전자 데이터 분석
- 맞춤 광고, SNS 실시간 검색어 분석을 통해 사람들의 관심 파악하여 제안

#### ■ 밀도 추정의 응용

- 분류, 생성 모델 구축 등

#### ■ 공간 변환의 응용

- 데이터 가시화, 데이터 압축, 특징 추출 (표현 학습) 등

## 6.3 군집화

- 6.3.1  $k$ -평균 알고리즘
- 6.3.2 친밀도 전파 알고리즘

## 6.3 군집화

### ■ 군집화 문제

- $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ 에서 식 (6.1)을 만족하는

군집 cluster or group 집합  $C = \{c_1, c_2, \dots, c_k\}$ 를 찾아내는 작업

$$\left. \begin{array}{l} c_i \neq \emptyset, i = 1, 2, \dots, k \\ \bigcup_{i=1}^k c_i = \mathbb{X} \\ c_i \cap c_j = \emptyset, i \neq j \end{array} \right\} \quad (6.1)$$

- 군집의 개수  $k$ 는 주어지는 경우와 자동으로 찾아야 하는 경우가 있음
- 군집화를 부류 발견 작업이라 부르기도 함

■ 군집화의 **주관성**: 군집의 개수, 최적의 거리 측정 방식을 설정해야 함

■ 군집의 개수  $K$

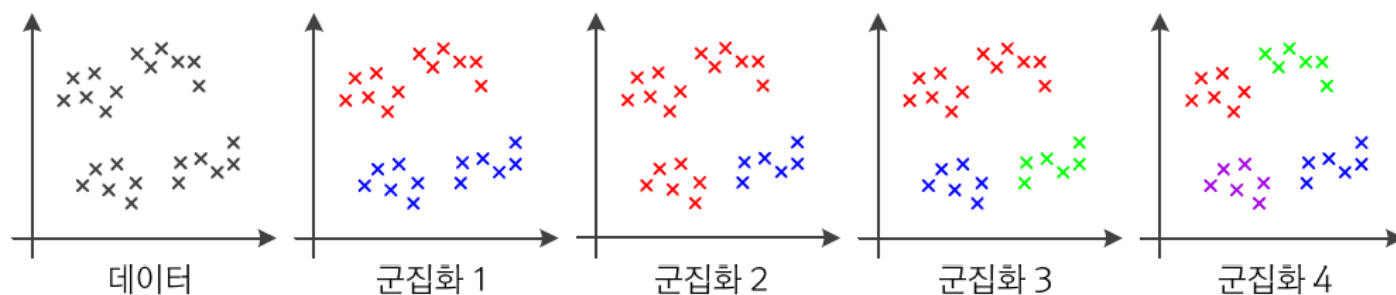


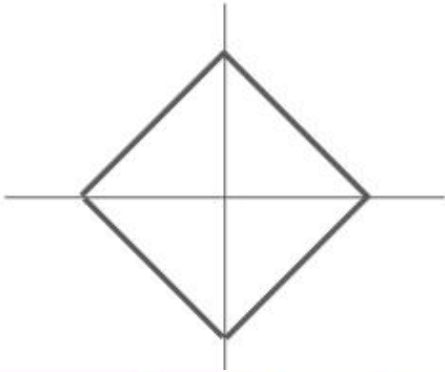
그림 6-3 군집화의 주관성

## 6.3 군집화

### ■ 거리 측정법 distance metric

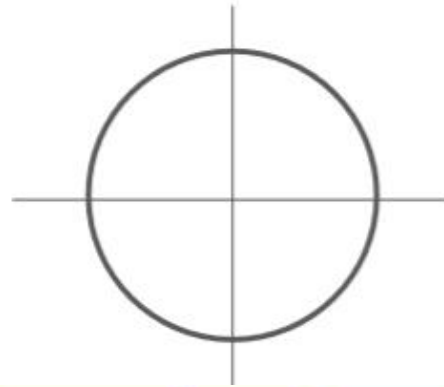
- L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$



- L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$

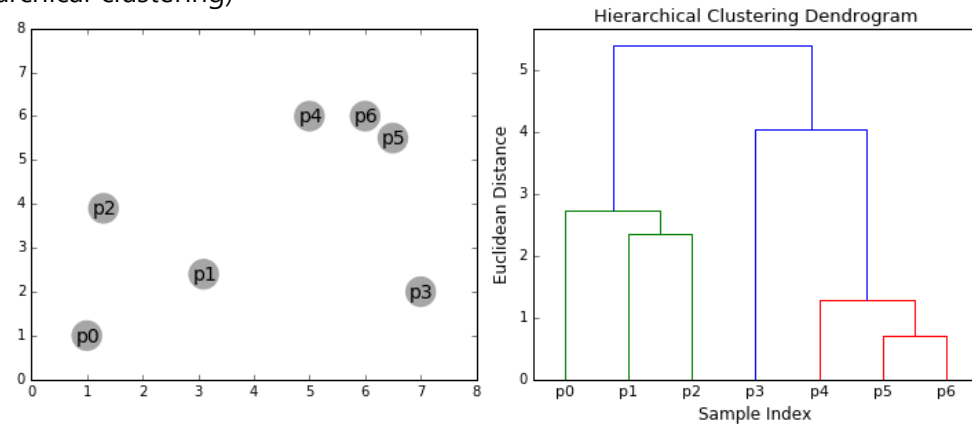


## 6.3 군집화

### ■ 분류

#### ■ 연결 기준 군집화(connectivity-based clustering (hierarchical clustering))

- 예) single-linkage clustering



#### ■ 중심 기준 군집화(centroid-based clustering)

- 예) k-means clustering

#### ■ 분포 기준 군집화(distribution-based clustering)

- 예) gaussian mixture model clustering

#### ■ 밀도 기준 군집화(density-based clustering)

- 예) DBSCAN

## 6.3.1 $k$ -평균 알고리즘

### ■ $k$ -평균 군집화 $k$ -mean clustering 알고리즘의 특성

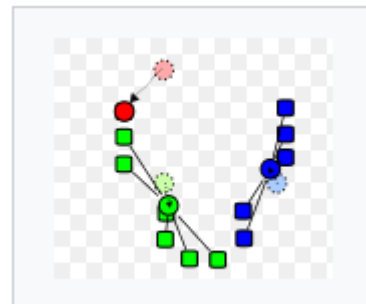
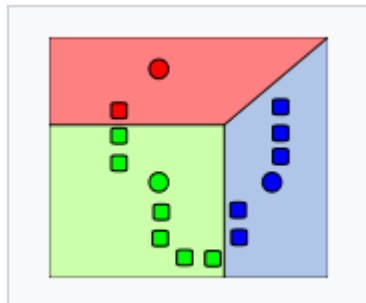
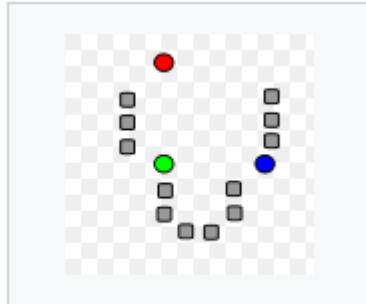
- 원리 단순하지만 성능이 좋아 인기 좋음
- 직관적으로 이해하기 쉽고 구현 쉬움
- 군집 개수  $k$ , 거리측정 방법을 설정해야 함

#### 알고리즘 6-1 $k$ -평균

입력: 훈련집합  $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ , 군집의 개수  $k$

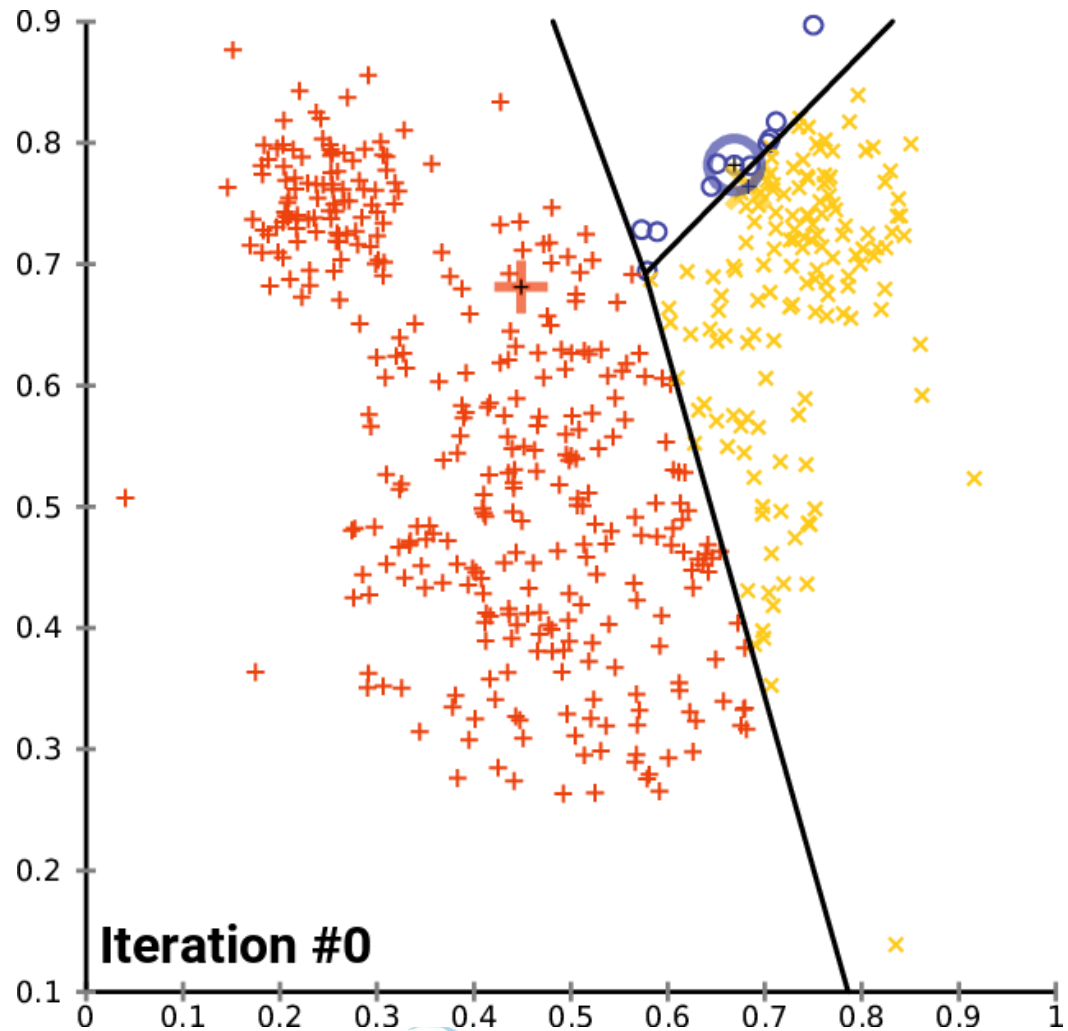
출력: 군집집합  $C = \{c_1, c_2, \dots, c_k\}$

```
1   $k$ 개의 군집 중심  $Z = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_k\}$ 를 초기화한다.
2  while (true)
3      for ( $i=1$  to  $n$ )
4           $\mathbf{x}_i$ 를 가장 가까운 군집 중심에 배정한다.
5          if (라인 3~4에서 이루어진 배정이 이전 루프에서의 배정과 같으면) break
6      for ( $j=1$  to  $k$ )
7           $\mathbf{z}_j$ 에 배정된 샘플의 평균으로  $\mathbf{z}_j$ 를 대체한다.
8  for ( $j=1$  to  $k$ )
9       $\mathbf{z}_j$ 에 배정된 샘플을  $c_j$ 에 대입한다.
```



## 6.3.1 $k$ -평균 알고리즘

### ■ 동작 예제



## 6.3.1 $k$ -평균 알고리즘

### ■ $k$ -평균과 $k$ -중앙객체 medoids 군집

- $k$ -평균은 [알고리즘 6-1]의 라인 7에서 샘플의 평균으로 군집 중심을 갱신 (잡음에 민감)
- $k$ -중앙객체는 실제 존재하는 객체들 중 하나를 뽑아 대표 객체로 선정하고 이 객체를 중심으로 군집 중심을 갱신 ( $k$ -평균에 비해 잡음에 둔감)
  - 중앙객체 medoids: 객체 집합에서 수학적으로 대표적인 객체

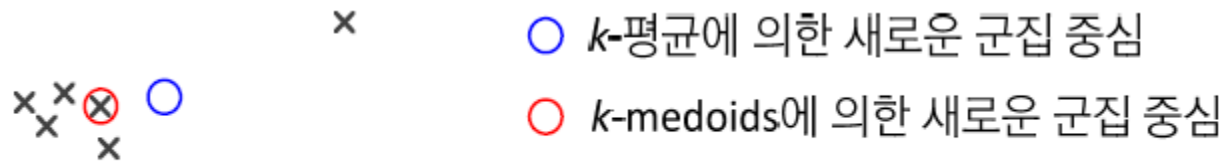


그림 6-4  $k$ -평균과  $k$ -medoids가 군집 중심을 갱신하는 과정

### ■ 최적화 문제로 해석

- $k$ -평균은 식 (6.2)의 목적함수를 최소화하는 알고리즘
- 행렬  $\mathbf{A}$ 는 군집 배정 정보를 나타내는  $k \times n$  행렬  
( $i$ 번째 샘플이  $j$ 번째 군집에 배정되었다면  $a_{ji}$ 는 1, 그렇지 않으면 0)

$$J(\mathbf{Z}, \mathbf{A}) = \sum_{i=1}^n \sum_{j=1}^k a_{ji} \text{dist}(\mathbf{x}_i, \mathbf{z}_j) \quad (6.2)$$



## 6.3.1 $k$ -평균 알고리즘

### 예제 6-1 $k$ -평균의 동작

[그림 6-5]는 훈련집합이 7개의 샘플을 가진  $n=7$ 인 예를 보여 준다. 좌표는 다음과 같다.

$$\mathbf{x}_1 = \begin{pmatrix} 18 \\ 5 \end{pmatrix}, \mathbf{x}_2 = \begin{pmatrix} 20 \\ 9 \end{pmatrix}, \mathbf{x}_3 = \begin{pmatrix} 20 \\ 14 \end{pmatrix}, \mathbf{x}_4 = \begin{pmatrix} 20 \\ 17 \end{pmatrix}, \mathbf{x}_5 = \begin{pmatrix} 5 \\ 15 \end{pmatrix}, \mathbf{x}_6 = \begin{pmatrix} 9 \\ 15 \end{pmatrix}, \mathbf{x}_7 = \begin{pmatrix} 6 \\ 20 \end{pmatrix}$$

군집의 개수  $k=3$ 이라 하자. 맨 왼쪽 그림은 초기 군집 중심을 보여 준다. [알고리즘 6-1]의 라인 3~4는 7개 샘플을 아래와 같이 배정할 것이다.

$$\{\mathbf{x}_1\} \text{은 } \mathbf{z}_1, \{\mathbf{x}_2\} \text{은 } \mathbf{z}_2, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7\} \text{은 } \mathbf{z}_3$$

이 배정을 행렬  $\mathbf{A}$ 로 표현하면 다음과 같다.

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

## 6.3.1 $k$ -평균 알고리즘

[그림 6-5]의 가운데 그림은 새로 계산한 군집 중심이다.  $\mathbf{z}_1 = (18, 5)^T$ ,  $\mathbf{z}_2 = (20, 9)^T$ ,  $\mathbf{z}_3 = (12, 16.2)^T$  이고, 식 (6.2)에 대입하면  $J = 244.8$ 이 된다. 이때 거리함수  $\text{dist}$ 로 식 (1.7)의 유클리디언 거리를 사용한다.

두 번째 루프를 실행하면 행렬  $\mathbf{A}$ 는 아래와 같이 바뀐다. 군집 중심은  $\mathbf{z}_1 = (18, 5)^T$ ,  $\mathbf{z}_2 = (20, 13.333)^T$ ,  $\mathbf{z}_3 = (6.667, 16.667)^T$ 이다. 이것을 식 (6.2)에 대입하면  $J = 58.00$ 이 된다. [그림 6-5]의 맨 오른쪽 그림은 두 번째 루프 수행 후의 상황이다.

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

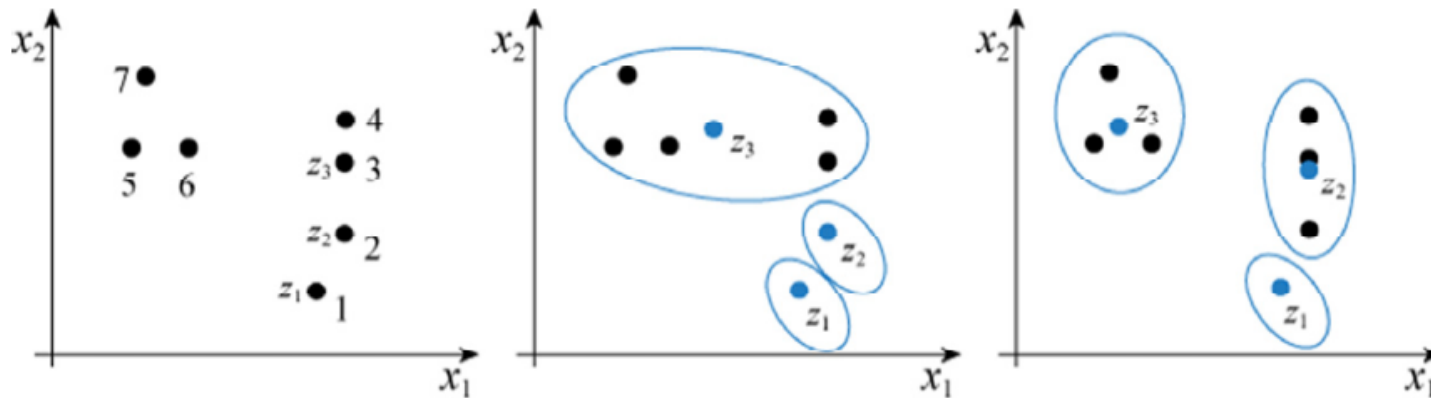


그림 6-5  $k$ -평균의 동작 예제

## 6.3.1 $k$ -평균 알고리즘

### ■ 다중 시작 $k$ -평균

- $k$ -평균은 [알고리즘 6-1]의 라인 1에서 초기 군집 중심이 달라지면 최종 결과가 달라짐
- 다중 시작은 서로 다른 초기 군집 중심을 가지고 여러 번 수행한 다음,  
가장 좋은 품질의 해를 취함

#### 알고리즘 6-2 다중 시작 $k$ -평균

입력: 훈련집합  $\mathbb{X} = \{x_1, x_2, \dots, x_n\}$ , 군집의 개수  $k$ , 다중 시작 횟수  $t$

출력: 군집집합  $C = \{c_1, c_2, \dots, c_k\}$

```
1  for ( $i=1$  to  $t$ )
2       $\mathbb{X}$ 에서 임의로  $k$ 개 샘플을 뽑는다.
3      라인 2에서 뽑은 샘플을 초기 군집 중심으로 삼고, [알고리즘 6-1]의  $k$ -평균을 수행한다.
4       $k$ -평균이 출력한 해를 가지고 식 (6.2)의 목적함숫값을 계산한다.
5   $t$ 개의 해 중 목적함숫값이 가장 작은 해를 최종해로 취한다.
```

## 6.3.1 $k$ -평균 알고리즘

### ■ EM (expectation maximization) 기초

- $k$ -평균에서 훈련집합  $\mathbb{X}$ 와 군집집합  $C$  (행렬  $A$ )는 각각 입력단과 출력단에서 관찰 가능
- 중간 단계의 임시 변수  $Z$  (입출력단에서 보이지 않기 때문에 은닉변수<sup>latent variable</sup>라 부름)
- $k$ -평균의 할당<sup>assignment</sup>과 갱신<sup>update</sup> 과정은

$Z$ 의 추정 (E 단계)과  $A$ 의 추정 (M 단계)을 번갈아 가면 수행하는 EM 알고리즘과 유사

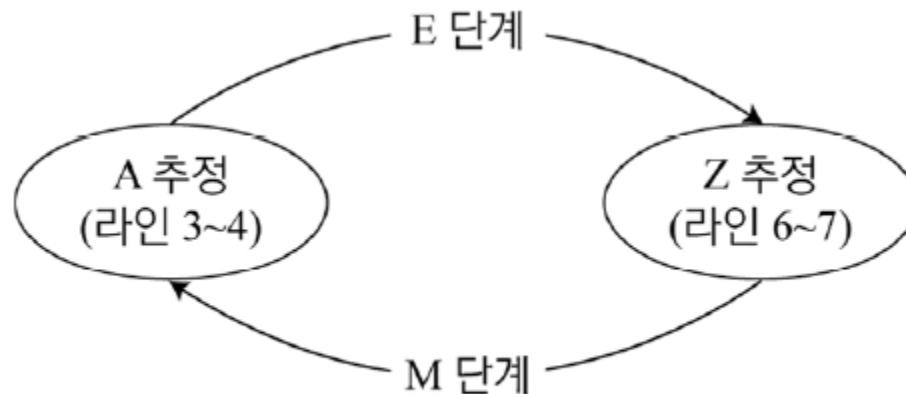


그림 6-6  $k$ -평균을 EM 알고리즘으로 해석

## 6.3.2 친밀도 전파 알고리즘

### ■ 친밀도 전파 affinity propagation 알고리즘

- 데이터들간의 신호 전달 message passing 개념을 사용하여

모든 데이터가 친밀도 기준에 따라 자신을 대표할 중심 데이터를 선택하고 군집화 함

- 책임 responsibility 행렬 **R**과 가용 availability 행렬 **A**라는 두 종류의 친밀도 행렬을 이용하여 군집화
  - $r_{ik}$ : 데이터  $x_k$ 가 데이터  $x_i$ 의 대표가 되어야 한다는 책임의 근거
  - $a_{ik}$ : 데이터  $x_i$ 가 데이터  $x_k$ 를 대표로 선택될 가능성의 근거
- 군집 개수  $k$ 를 자동으로 추정하여 알아냄 ( $k$ -평균 군집의 단점 해소)

### ■ 샘플 $i$ 와 $k$ 의 유사도 $s_{ik}$

$$s_{ik} = -\|\mathbf{x}_i - \mathbf{x}_k\|_2^2, \quad i \neq k \text{이고 } i, k = 1, 2, \dots, n \quad (6.3)$$

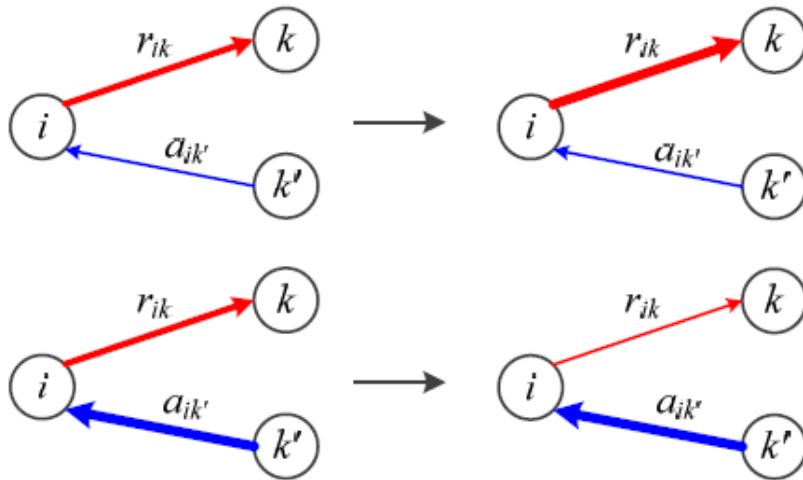
- 유사도를 이용하여 책임 행렬과 가용 행렬을 계산
- 유클리디언 거리의 제곱에 음수
  - 가까울수록, 즉 유사할수록 큰 값을 가짐

## 6.3.2 친밀도 전파 알고리즘

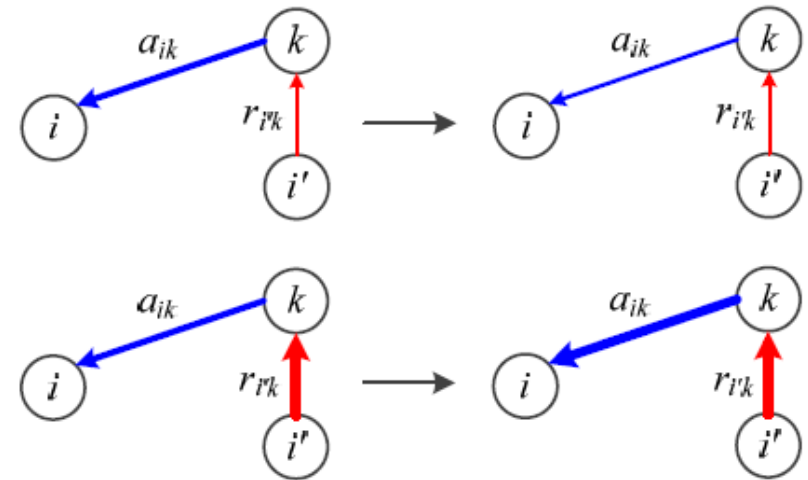
### ■ 책임 행렬 **R**과 가용 행렬 **A**의 계산

$$r_{ik} = s_{ik} - \max_{k' \neq k} (a_{ik'} + s_{ik'}) \quad (6.4)$$

$$a_{ik} = \min \left( 0, r_{kk} + \sum_{i' \neq i, k} \max(0, r_{i'k}) \right), i \neq k \quad (6.5)$$



(a)  $r_{ik}$ 의 계산



(b)  $a_{ik}$ 의 계산

그림 6-7 친밀도 계산 과정

## 6.3.2 친밀도 전파 알고리즘

### ■ 자가 유사도 $s_{kk}$

- 유사도의 최솟값, 중앙값<sup>median</sup>, 최댓값 중에서 선택 (하이퍼 매개변수임)
- 최솟값은 적은 수의 군집, 최댓값은 많은 수의 군집을 생성. 중앙값은 중간 정도

### ■ 자가 친밀도 $r_{kk}$ 와 $a_{kk}$

- $r_{kk}$ 는 식 (6.4)를 그대로 사용. 즉  $r_{kk} = s_{kk} - \max_{k' \neq k} (a_{kk'} + s_{kk'})$
- $a_{kk}$ 는 식 (6.6)으로 계산

$$a_{kk} = \sum_{i' \neq k} \max(0, r_{i'k}) \quad (6.6)$$

## 6.3.2 친밀도 전파 알고리즘

### 알고리즘 6-3 친밀도 전파 군집화

입력: 훈련집합  $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ , 군집 개수 선택사항  $\in \{\text{최솟값}, \text{메디안}, \text{최댓값}\}$ , 댐핑 인자  $\lambda$

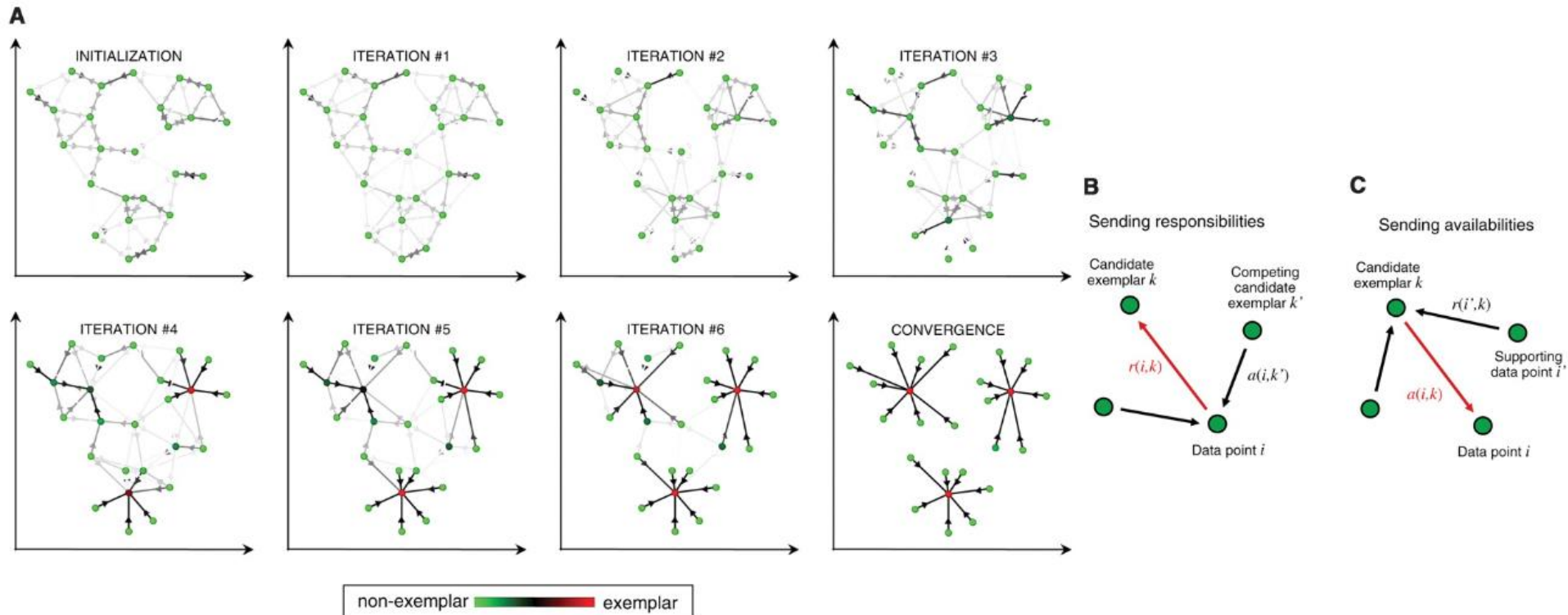
출력: 군집 정보  $\mathbf{z} = (z_1, z_2, \dots, z_n)^T$

```
1 for (모든 샘플 쌍  $i$ 와  $k$ 에 대해) if ( $i \neq k$ )  $s_{ik} = -\|\mathbf{x}_i - \mathbf{x}_k\|_2^2$  // 식 (6.3)
2 for ( $k = 1$  to  $n$ )  $s_{kk}$  = 선택 사항에 따라 라인 1의 유사도의 최솟값, 중앙값, 또는 최댓값
3 for (모든 샘플 쌍  $i$ 와  $k$ 에 대해)  $a_{ik}^0 = 0, r_{ik}^0 = 0$ 
4  $t = 0$ 
5 repeat
6    $t++$ 
7   for (모든 샘플 쌍  $i$ 와  $k$ 에 대해)
8      $r_{ik}^t = s_{ik} - \max_{k' \neq k} (a_{ik'}^t + s_{ik'})$  // 식 (6.4)
9      $r_{ik}^t = \lambda r_{ik}^t + (1 - \lambda) r_{ik}^{t-1}$ 
10    for (모든 샘플 쌍  $i$ 와  $k$ 에 대해)
11      if ( $i \neq k$ )  $a_{ik}^t = \min(0, r_{kk}^t + \sum_{i' \neq i, k} \max(0, r_{i'k}^t))$  // 식 (6.5)
12      else  $a_{kk}^t = \sum_{i' \neq k} \max(0, r_{i'k}^t)$  // 식 (6.6)
13       $a_{ik}^t = \lambda a_{ik}^t + (1 - \lambda) a_{ik}^{t-1}$  // 라인 9과 13은
14 until (멈춤 조건) // 진자 현상을 방지하기 위해 감폭 인자 damping factor 적용
15 for ( $i = 1$  to  $n$ )
16    $\hat{k} = \operatorname{argmax}_{k=1, n} (a_{ik}^t + r_{ik}^t)$ 
17   if ( $\hat{k} = i$ )  $z_i = i$  // 이 샘플은 군집 중심임 // 자신의 자가 친밀도가 최고이면 군집 대표
18   else  $z_i = \hat{k}$  // 이 샘플은  $\hat{k}$  군집 중심에 속함
```



## 6.3.2 친밀도 전파 알고리즘

### ■ 알고리즘 세부 과정

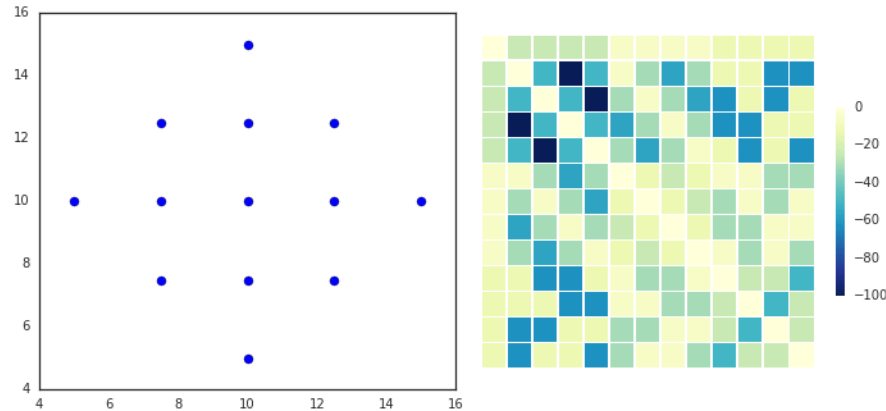


(B) "책임"  $r(i, k)$ : 데이터 포인트  $\rightarrow$  후보 표본  
각 데이터 포인트가 다른 후보 표본보다 후보 표본을 선호하는 정도

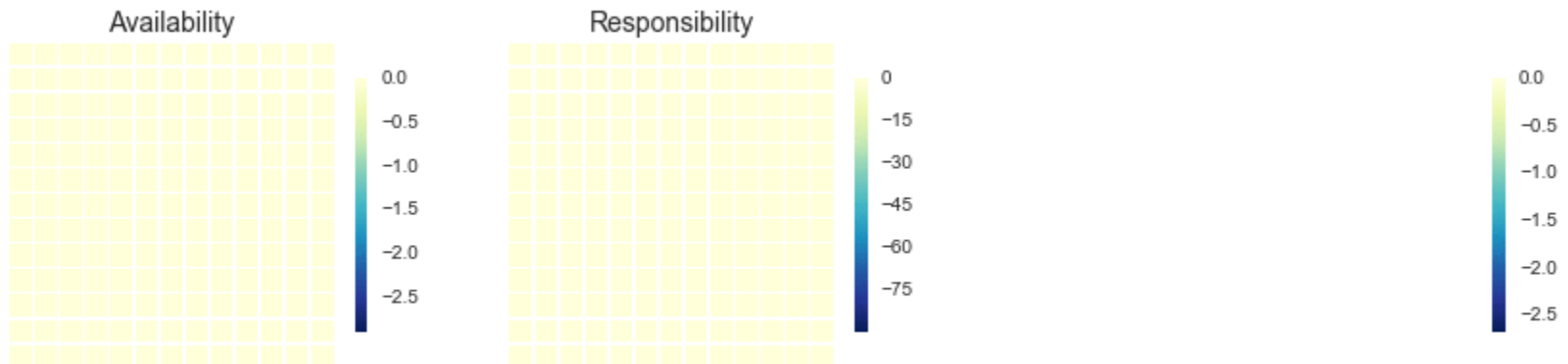
(C) "가용도"  $a(i, k)$ : 후보 표본  $\rightarrow$  데이터 포인트  
각 후보 표본이 데이터 포인트의 군집 중심으로 사용될 수 있는 정도

## 6.3.2 친밀도 전파 알고리즘

### ■ 알고리즘 동작 유사도 계산

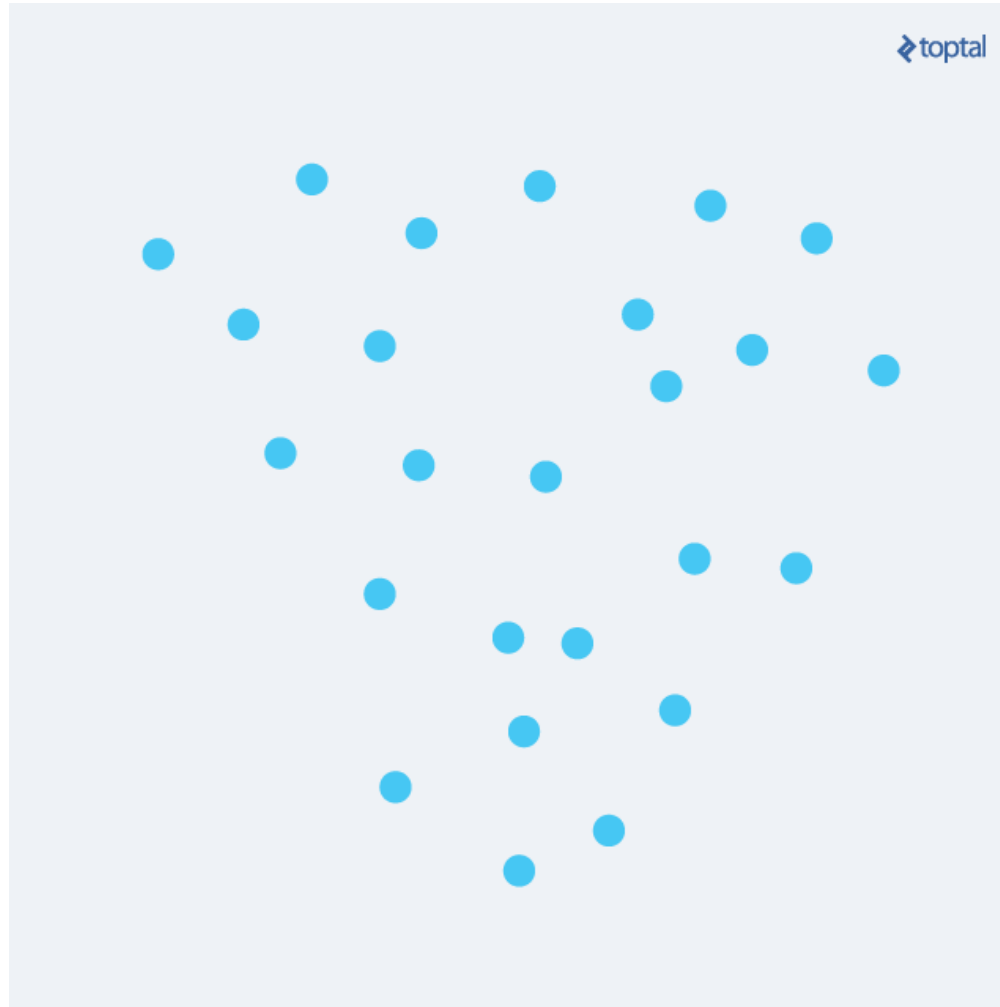


대표 추출 = 책임+가용 > 0



## 6.3.2 친밀도 전파 알고리즘

### ■ 동작 예제



## 6.4 밀도 추정

- 6.4.1 커널 밀도 추정
- 6.4.2 가우시안 혼합
- 6.4.3 EM 알고리즘

### ■ 밀도 추정 문제

- **모수적**parametric **추정**: 정해진 확률 밀도 함수로 추정 (=몇 개의 매개변수로 확률 분포 정의)
- **비모수적**nonparametric **추정**: 관측치만으로 데이터의 분포를 추정
- 어떤 점  $\mathbf{x}$ 에서 데이터가 발생할 확률,  
즉 **확률분포  $P(\mathbf{x})$ 를 구하는 문제**
- 예를 들어, 그림 6-8에서  $P(\mathbf{x}_1) > P(\mathbf{x}_2) > P(\mathbf{x}_3)$

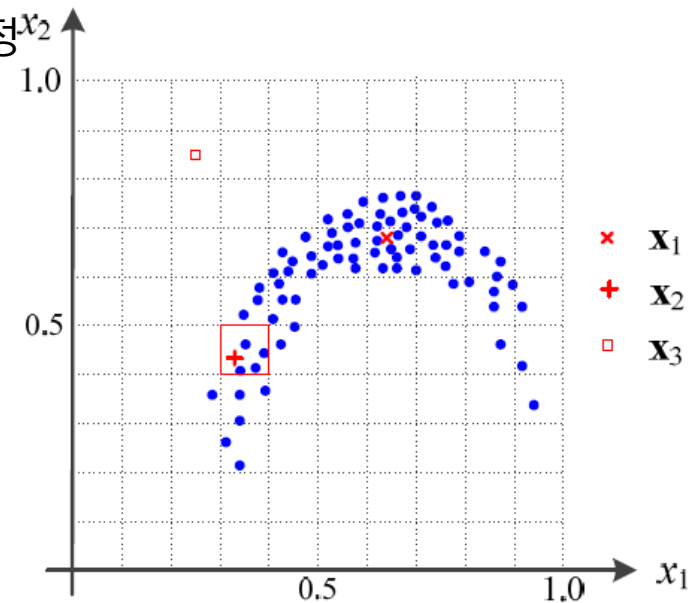


그림 6-8 밀도 추정 문제

## 6.4.1 커널 밀도 추정

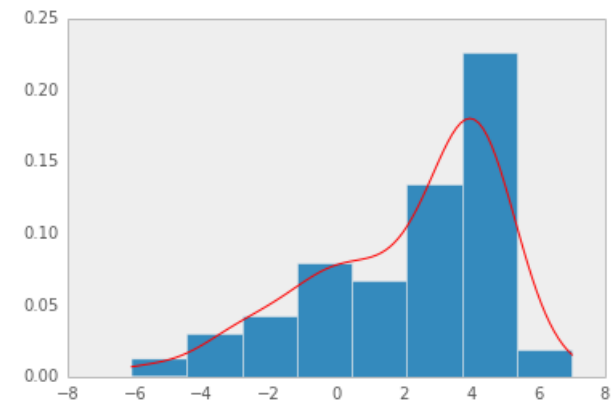
### ■ 히스토그램 histogram 추정법

- 특징 공간을 칸<sup>bin</sup>의 집합으로 분할한 다음, 칸에 있는 샘플의 빈도를 세어 식 (6.7)로 추정

- 예,  $P(\mathbf{x} = +) = \frac{4}{80} = 0.05$

$$P(\mathbf{x}) = \frac{bin(\mathbf{x})}{n} \quad (6.7)$$

- 여러 문제점
  - 매끄럽지 못하고 (경계에서의 불연속성) 계단 모양을 띠는 확률밀도함수가 됨
  - 칸의 크기와 위치에 민감함



## 6.4.1 커널 밀도 추정

### ■ 커널 밀도 추정법 kernel density estimation method (비모수 방법)

- 커널함수를 이용한 확률 밀도 함수를 만들어 히스토그램 추정의 단점을 개선
  - 커널kernel (=알맹이): 원점 대칭이면서 1인 적분값을 가지는 함수로 정의
- 점  $\mathbf{x}$ 에 [그림 6-9]가 예시하는 커널을 씌우고 커널 안에 있는 샘플의 가중합을 이용함
- 대역폭bandwidth  $h$ 의 크기가 중요

$$P_h(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n K_h(\mathbf{x} - \mathbf{x}_i) = \frac{1}{nh^d} \sum_{i=1}^n K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) \quad (6.8)$$

여기서  $K_h(\mathbf{x}) = \frac{1}{h^d} K\left(\frac{\mathbf{x}}{h}\right)$

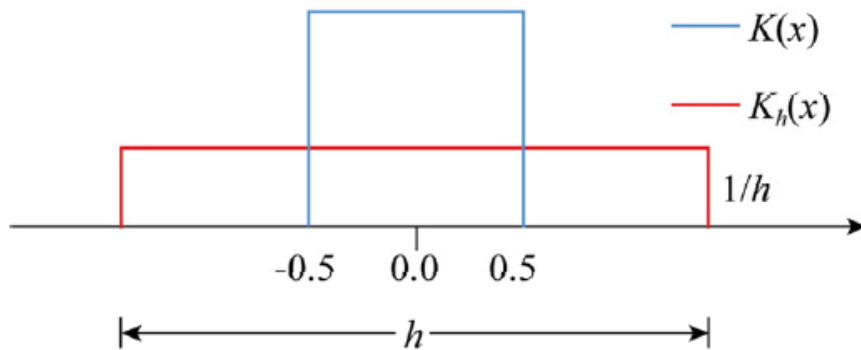
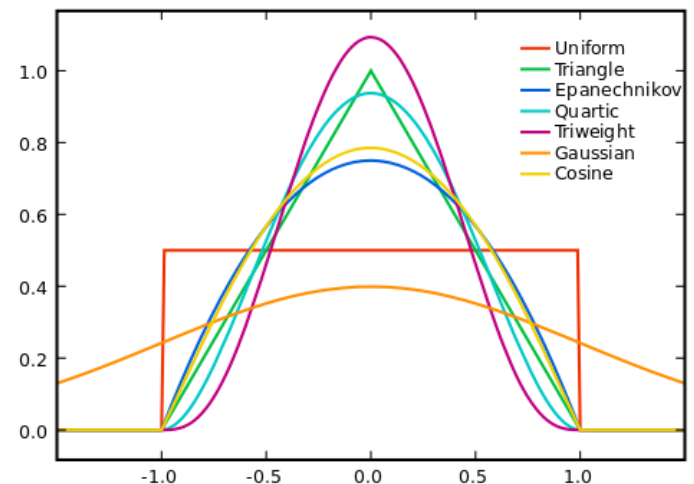


그림 6-9 표준커널함수  $K$ 와 크기 변환된 커널함수  $K_h$

커널 함수의 종류



## 6.4.1 커널 밀도 추정

### ■ 히스토그램 방법과 커널 밀도 추정법의 비교

- 관측된 샘플마다 해당 값을 중심으로 하는 커널 함수 생성  $K_h(x - x_i)$
- 얻어진 커널 함수의 결과를 더하여 전체 개수로 나눔

→ 커널 밀도 추정법은 매끄러운 확률밀도함수를 추정함

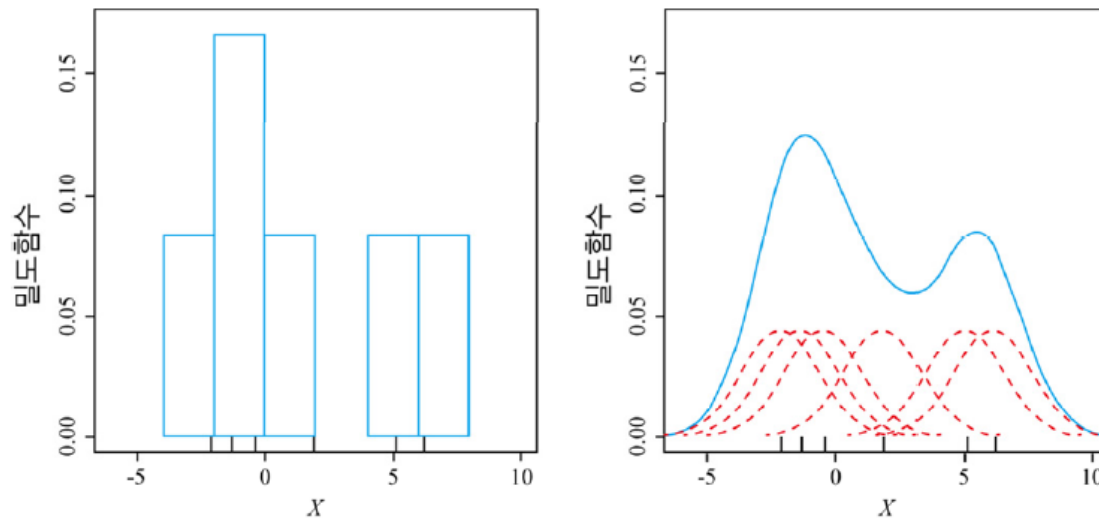
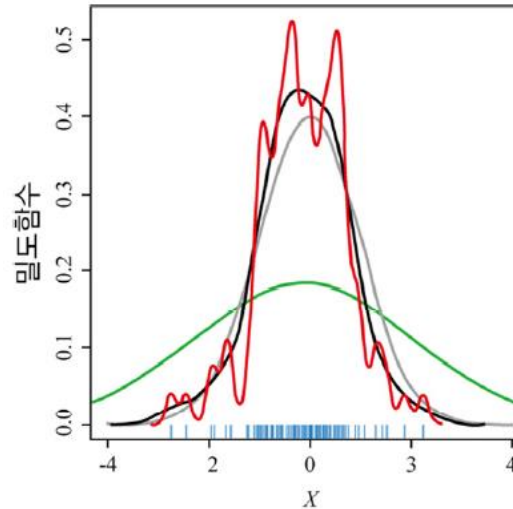


그림 6-10 히스토그램 방법(왼쪽)과 커널 밀도 추정법(오른쪽)의 비교

## 6.4.1 커널 밀도 추정

### ■ 커널 밀도 추정법에서 대역폭 $h$ 의 중요성

- $h$ 가 너무 작으면 (빨강) 뾰족뾰족한 모양,  $h$ 가 너무 크면 (녹색) 뭉개짐,



적절하게 설정해야 함 (검정)

그림 6-11 대역폭이 확률밀도함수 추정에 미치는 영향

### ■ 커널 밀도 추정 기법의 근본적 문제점

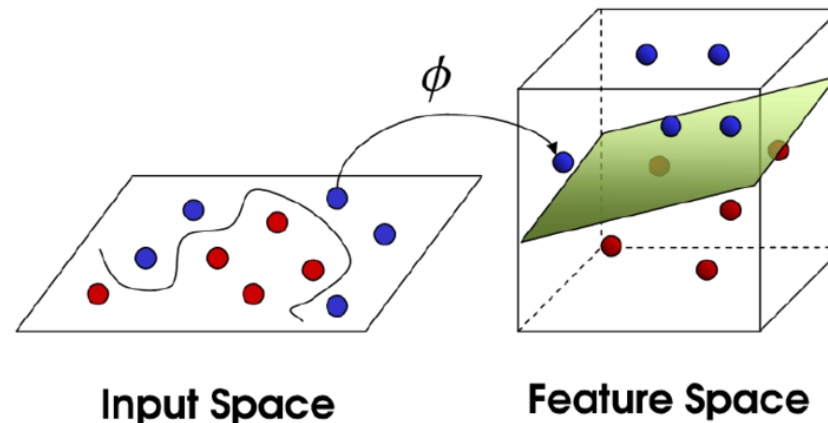
- 샘플을 모두 저장하고 있어야 하는 메모리 기반 방법  
(새로운 샘플이 주어질 때마다 식 (6.8)을 처음부터 다시 계산)
- 데이터 희소성 (차원의 저주)  
→ 데이터가 낮은 차원인 경우로 국한하여 활용



## 6.4.1 커널 밀도 추정

### ■ (참고) 커널 함수 kernel function

- 커널 밀도 추정법의 예와 같이 확률 밀도 추정에서 활용
- 차원 변환에서도 활용
  - 원공간 input space에서 선형 분리가 안되는 데이터들을 공간 변형을 통해 고차원 공간 feature space으로 옮기고 선형 분리되도록 함



### → 커널 대치 kernel trick

- 서포트벡터머신 support vector machine (SVM)에서 활용됨
- 실제 데이터를 새로운 공간으로 변형하지 않고 변형된 공간의 데이터간의 거리를 구함
- 예) 다항식 커널, 가우시안 커널 gaussian kernel 혹은 radial basis function

## 6.4.2 가우시안 혼합

### ■ 가우시안(gaussian)을 이용한 방법 (모수적 방법)

- 데이터가 가우시안 분포를 따른다고 가정하고 평균 벡터  $\mu$ 와 공분산 행렬  $\Sigma$ 를 추정함

$$P(\mathbf{x}) = N(\mathbf{x}; \mu, \Sigma) = \frac{1}{\sqrt{|\Sigma|}\sqrt{(2\pi)^d}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1}(\mathbf{x} - \mu)\right)$$

이때  $\mu = \frac{1}{n} \sum_{i=1,n} \mathbf{x}_i, \Sigma = \frac{1}{n} \sum_{i=1,n} (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T$

(6.9)

### ■ 대부분 데이터가 하나의 가우시안으로 불충분 ([그림 6-12]의 오른쪽)

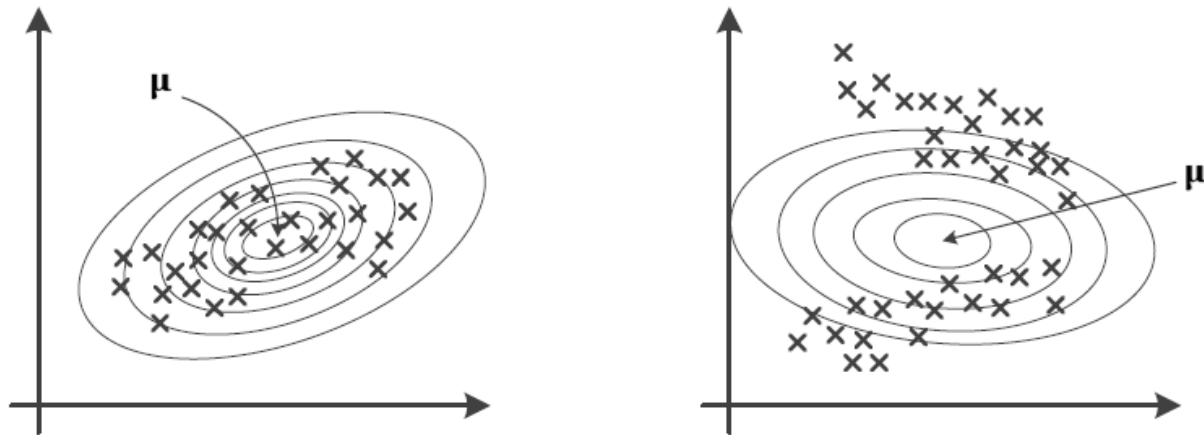


그림 6-12 하나의 가우시안으로 밀도 추정

## 6.4.2 가우시안 혼합

### ■ 가우시안 혼합 gaussian mixture

- [그림 6-13]은 2개의 가우시안을 사용한 예

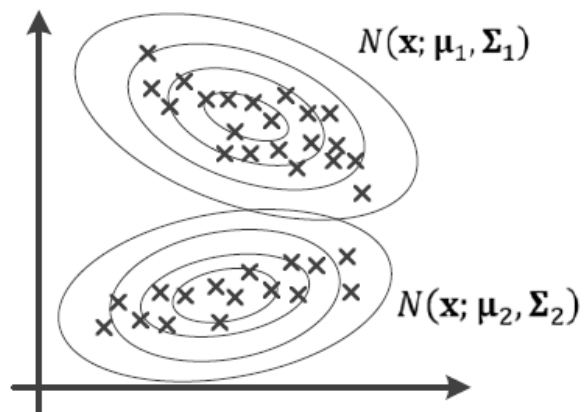


그림 6-13 가우시안 혼합으로 밀도 추정

$$P(\mathbf{x}) = \frac{21}{37} N(\mathbf{x}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) + \frac{16}{37} N(\mathbf{x}; \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$$

### ■ $k$ 개의 가우시안으로 일반화하면,

- 확률분포  $P(\mathbf{x})$ 는  $k$ 개 가우시안의 선형 결합으로 표현(식 (6.10))

$$P(\mathbf{x}) = \sum_{j=1}^k \pi_j N(\mathbf{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \quad (6.10)$$

## 6.4.2 가우시안 혼합

- 주어진 데이터와 추정해야 할 매개변수를 정리하면,

주어진 데이터: 훈련집합  $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ , 가우시안의 개수  $k$

추정해야 할 매개변수집합:  $\Theta = \{\boldsymbol{\pi} = (\pi_1, \pi_2, \dots, \pi_k), (\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), (\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2), \dots, (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}$

- 최대 우도 maximum likelihood를 이용한 최적화 문제로 공식화

$$P(\mathbb{X}|\Theta) = \prod_{i=1}^n P(\mathbf{x}_i|\Theta) = \prod_{i=1}^n \left( \sum_{j=1}^k \pi_j N(\mathbf{x}_i; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \right) \quad (6.11)$$

$$\log P(\mathbb{X}|\Theta) = \sum_{i=1}^n \log \left( \sum_{j=1}^k \pi_j N(\mathbf{x}_i; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \right) \quad (6.12)$$

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmax}} \log P(\mathbb{X}|\Theta) \quad (6.13)$$

→ 주어진  $x$ 가 발생할 가능성이 가장 큰  $\theta$ 를 찾는 문제

## 6.4.3 EM 알고리즘

### ■ EM 알고리즘을 이용한 식 (6.13)의 풀이

- $\theta$ 를 모르므로 난수로 설정하고 출발 ([그림 6-14]의 예시 2개의 가우시안을 사용)

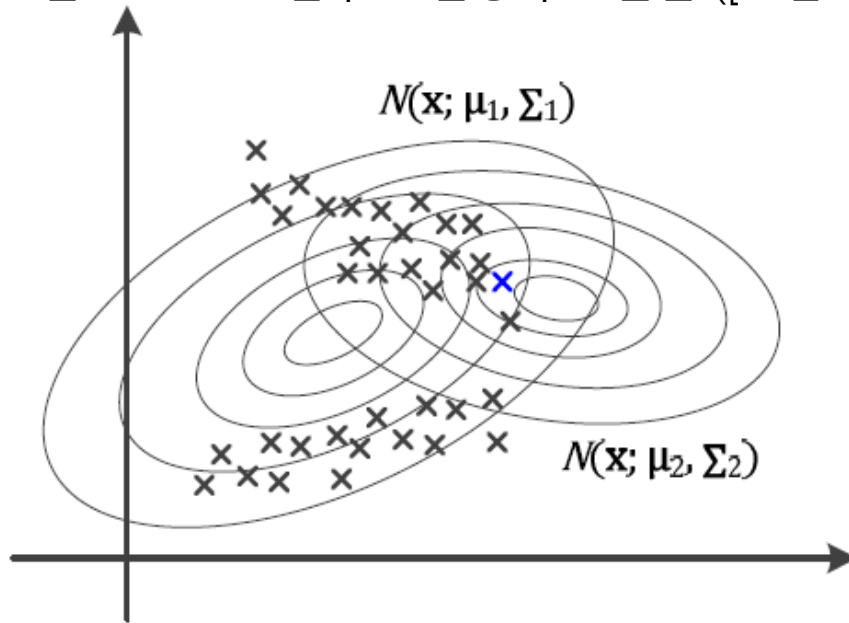


그림 6-14 샘플의 소속 확률을 어떻게 추정할 것인가

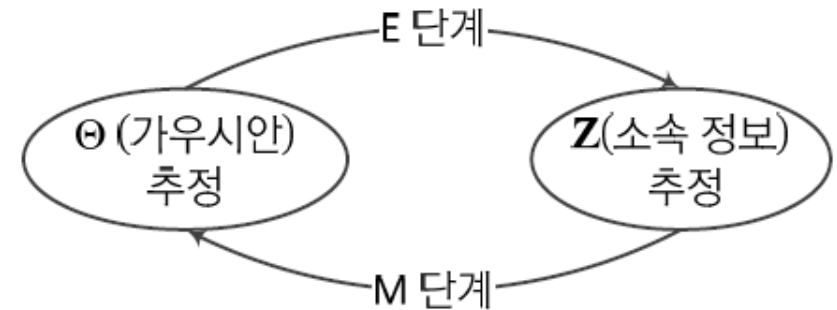
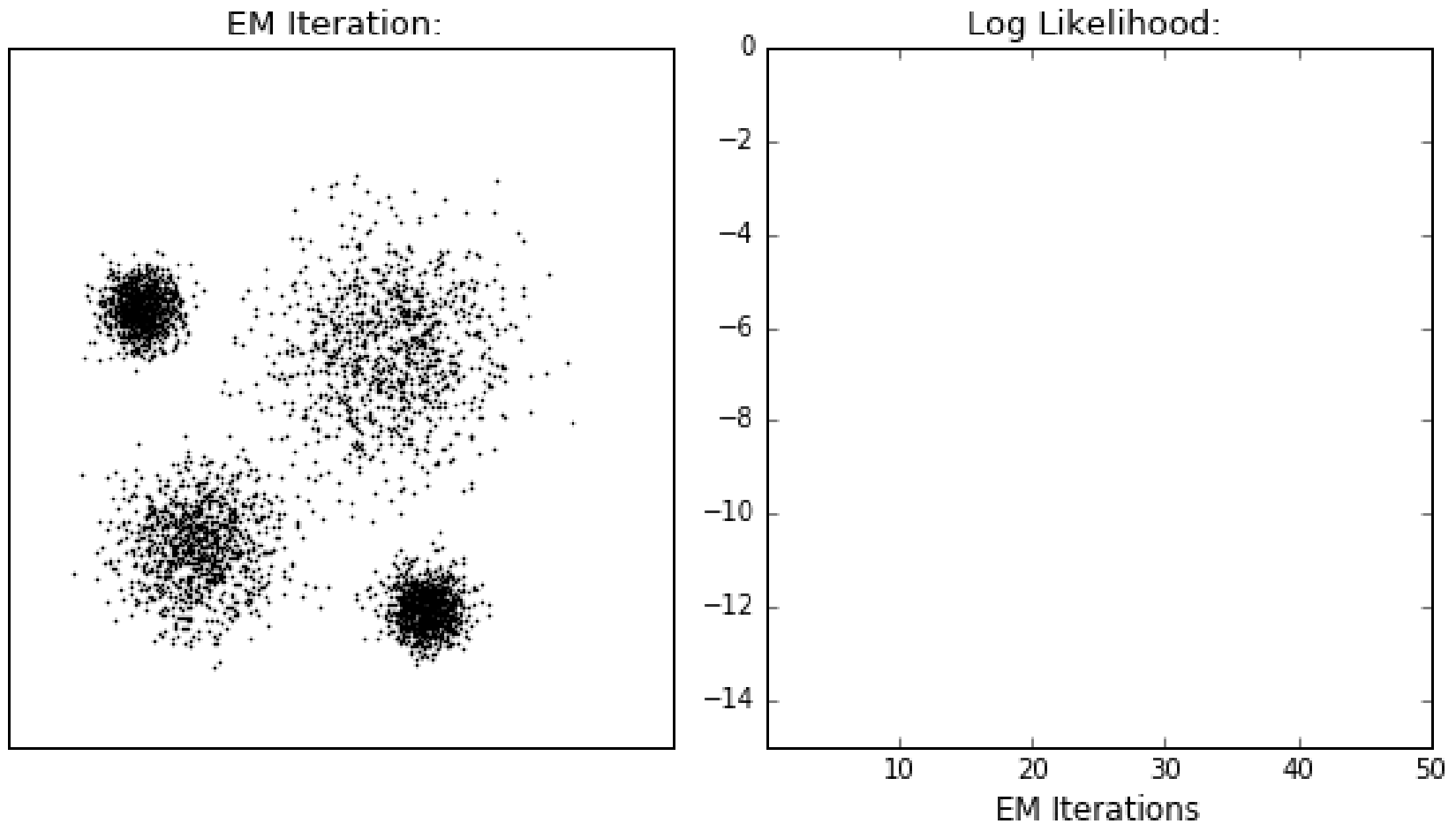


그림 6-15 가우시안 혼합을 위한 EM 알고리즘

- 과정: 가우시안으로 샘플의 소속 정보 개선 (E단계)
  - 샘플의 소속 정보로 가우시안 개선 (M단계)
  - 가우시안으로 샘플의 소속 정보 개선 (E단계)
  - 샘플의 소속 정보로 가우시안 개선 (M단계)
  - ..... ([그림 6-15])

## 6.4.3 EM 알고리즘

### ■ EM 알고리즘 동작 예제



## 6.4.3 EM 알고리즘

### ■ 가우시안 혼합을 위한 EM 알고리즘

#### 알고리즘 6-4 가우시안 혼합을 위한 EM 알고리즘

입력: 훈련집합  $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ , 가우시안의 개수  $k$

출력: 최적의 가우시안과 혼합 계수  $\Theta = \{\boldsymbol{\pi} = (\pi_1, \pi_2, \dots, \pi_k), (\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), (\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2), \dots, (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}$

```
1   $\Theta$ 를 초기화한다.  
2  while (!멈춤조건)  
3     $\Theta$ 를 이용하여 소속확률 행렬  $\mathbf{Z}$ 를 추정한다. // E단계  
4     $\mathbf{Z}$ 를 이용하여  $\Theta$ 를 추정한다. // M단계
```

- 라인 3과 라인 4를 위한 수식
- $z_{ji}$ 는  $\mathbf{x}_i$ 가  $j$ 번째 가우시안에 속할 확률

$$z_{ji} = \frac{\pi_j N(\mathbf{x}_i; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{q=1}^k \pi_q N(\mathbf{x}_i; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)} \quad (6.14)$$

$$\left. \begin{aligned} \boldsymbol{\mu}_j &= \frac{1}{n_j} \sum_{i=1}^n z_{ji} \mathbf{x}_i \\ \boldsymbol{\Sigma}_j &= \frac{1}{n_j} \sum_{i=1}^n z_{ji} (\mathbf{x}_i - \boldsymbol{\mu}_j)(\mathbf{x}_i - \boldsymbol{\mu}_j)^T \\ \pi_j &= \frac{n_j}{n} \\ \text{이때 } n_j &= \sum_{i=1}^n z_{ji} \end{aligned} \right\} \quad (6.15)$$

## 6.5 공간 변환의 이해

### ■ 간단한 상황 예시

- 2개 군집을 가진 [그림 6-16]의 2차원 특징 공간을 극좌표 polar coordinates 공간으로 변환하면 1차원만으로 2개 군집 표현 가능

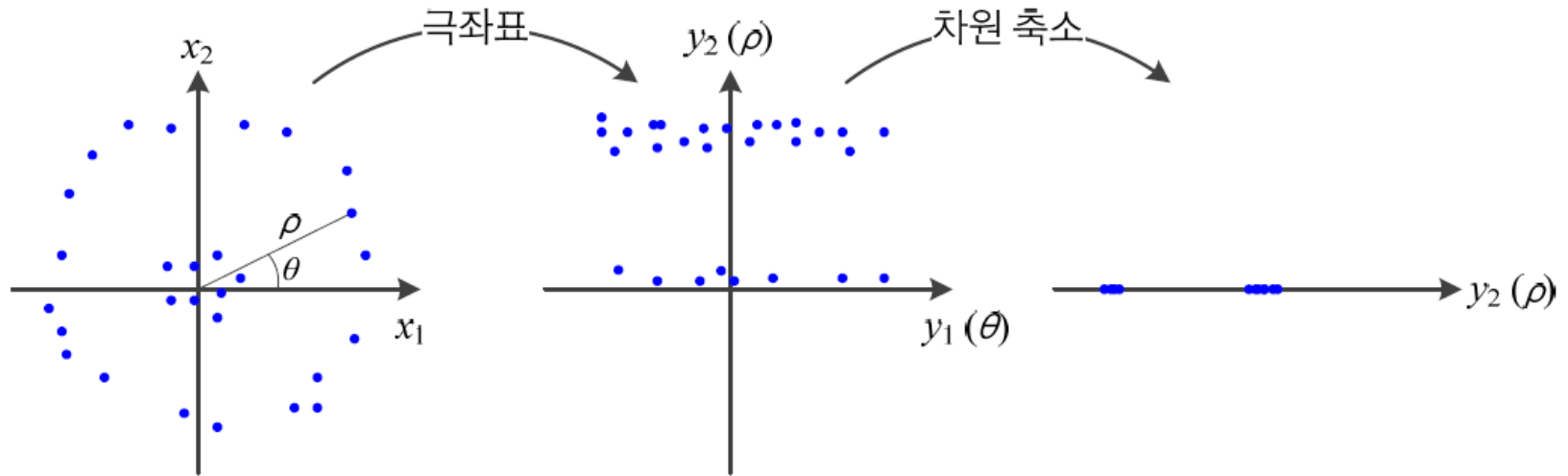


그림 6-16 공간 변환의 예

### ■ 실제 문제에서는 비지도 학습을 이용하여

최적의 공간 변환을 자동으로 알아내야 함



## 6.5 공간 변환의 이해

### ■ 인코딩 encoding과 디코딩 decoding

- 원래 공간을 다른 공간으로 변환하는 인코딩 과정 ( $f$ ),  
변환 공간을 원래 공간으로 역변환하는 디코딩 과정 ( $g$ )

$$\hat{\mathbf{x}} = g(f(\mathbf{x})) \quad (6.16)$$

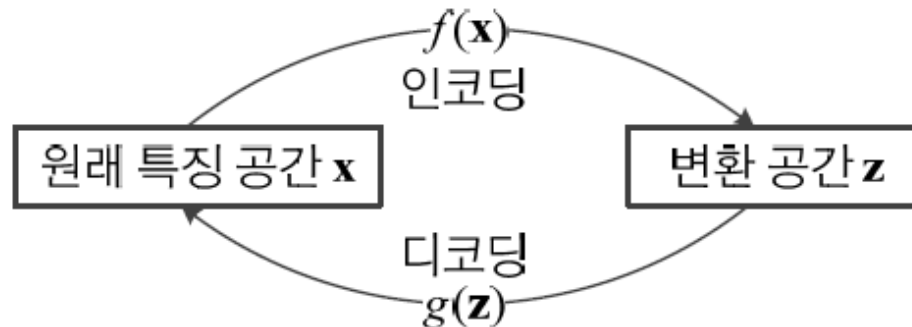


그림 6-17 공간 변환과 역변환

$\mathbf{z}$ : 인자<sup>factor</sup> 혹은  
잠복변수<sup>latent variable</sup> (은닉변수<sup>hidden variable</sup>)라고 함

- 예) 데이터 압축의 경우, 역변환으로 얻은  $\hat{\mathbf{x}}$ 은 원래 신호  $\mathbf{x}$ 와 가급적 같아야 함
- 예) 데이터 가시화에서는 2차원 또는 3차원의  $\mathbf{z}$  공간으로 변환

## 6.6 선형 인자 모델

- 6.6.1 주성분 분석
- 6.6.2 독립 성분 분석
- 6.6.3 희소 코딩

## 6.6 선형 인자 모델

### ■ 선형 인자 모델 linear factor models

- 선형 연산을 이용한 공간 변환 기법
- 선형 연산을 사용하므로 행렬 곱으로 인코딩 (식 (6.17))과 디코딩 (식 (6.18)) 과정을 표현

$$f: \mathbf{z} = \mathbf{W}_{enc}\mathbf{x} + \boldsymbol{\alpha}_{enc} \quad (6.17)$$

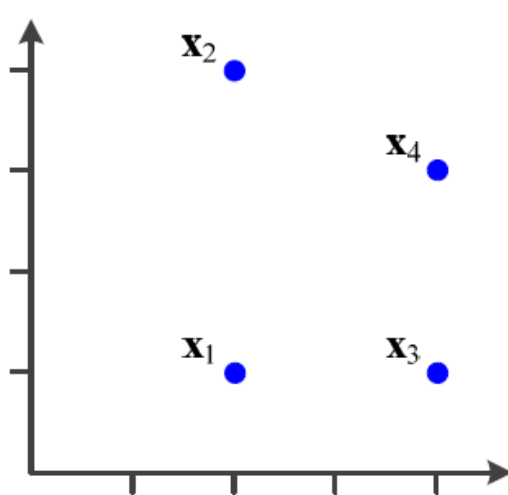
$$g: \mathbf{x} = \mathbf{W}_{dec}\mathbf{z} + \boldsymbol{\alpha}_{dec} \quad (6.18)$$

- $\boldsymbol{\alpha}$ 는 데이터를 원점으로 이동하거나 잡음을 추가하는 등의 역할
- 인자  $\mathbf{z}$ 와 추가 항  $\boldsymbol{\alpha}$ 에 따라 여러 가지 모델이 존재
  - $\mathbf{z}$ 에 확률 개념이 없고  $\boldsymbol{\alpha}$ 를 생략하면 주성분 분석 principal component analysis (PCA) (6.6.1절)  
:관찰 벡터  $\mathbf{x}$ 와 인자  $\mathbf{z}$ 는 결정론적인 1:1 매핑 관계
  - $\mathbf{z}$ 와  $\boldsymbol{\alpha}$ 가 가우시안 분포를 따른다고 가정하면 확률 주성분 분석 probabilistic PCA
  - $\mathbf{z}$ 가 비가우시안 분포를 따른다고 가정하는 독립 성분 분석 independent component analysis (ICA) (6.6.2절)

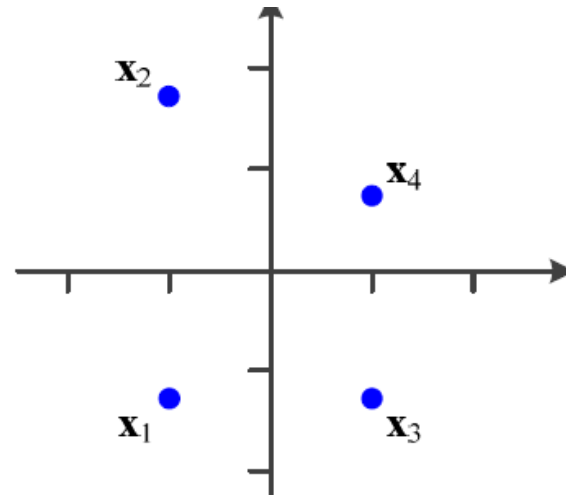
## 6.6.1 주성분 분석

- 데이터를 원점 중심으로 옮기는 **전처리**를 먼저 수행

$$\left. \begin{array}{l} \mathbf{x}_i = \mathbf{x}_i - \boldsymbol{\mu}, \quad i = 1, 2, \dots, n \\ \text{이때 } \boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i \end{array} \right\} \quad (6.19)$$



(a) 원래 훈련집합  $\mathbb{X}$



(b)  $\mathbb{X}$ 에 식 (6.19)를 적용한 이후

그림 6-18  $\mathbb{X}$ 의 평균을 0으로 변환

## 6.6.1 주성분 분석

### ■ 주성분 분석<sup>PCA</sup>이 사용하는 변환식

- 일반적인 선형 변환식인 식 (6.17)에서  $\mathbf{z}$ 에 확률 개념이 없고  $\alpha$ 를 생략하면 주성분 분석
- 변환 행렬  $\mathbf{W}$ 는  $d * q$ 로서 주성분 분석은  $d$ 차원의  $\mathbf{x}$ 를  $q$ 차원의  $\mathbf{z}$ 로 변환 ( $q < d$ )
  - $\mathbf{W}$ 의  $j$ 번째 열 벡터와의 내적  $\mathbf{u}_j^T \mathbf{x}$ 는  $\mathbf{x}$ 를  $\mathbf{u}_j$ 가 가리키는 축으로 투영

$$\left. \begin{aligned} \mathbf{z} &= \mathbf{W}^T \mathbf{x} \\ \text{이때 } \mathbf{W} &= (\mathbf{u}_1 \ \mathbf{u}_2 \ \cdots \ \mathbf{u}_q) \text{이고, } \mathbf{u}_j = (u_{1j}, u_{2j}, \cdots, u_{dj})^T \end{aligned} \right\} \quad (6.20)$$

- 예, 2차원을 1차원으로 변환하는 상황 ( $d = 2, q = 1$ )

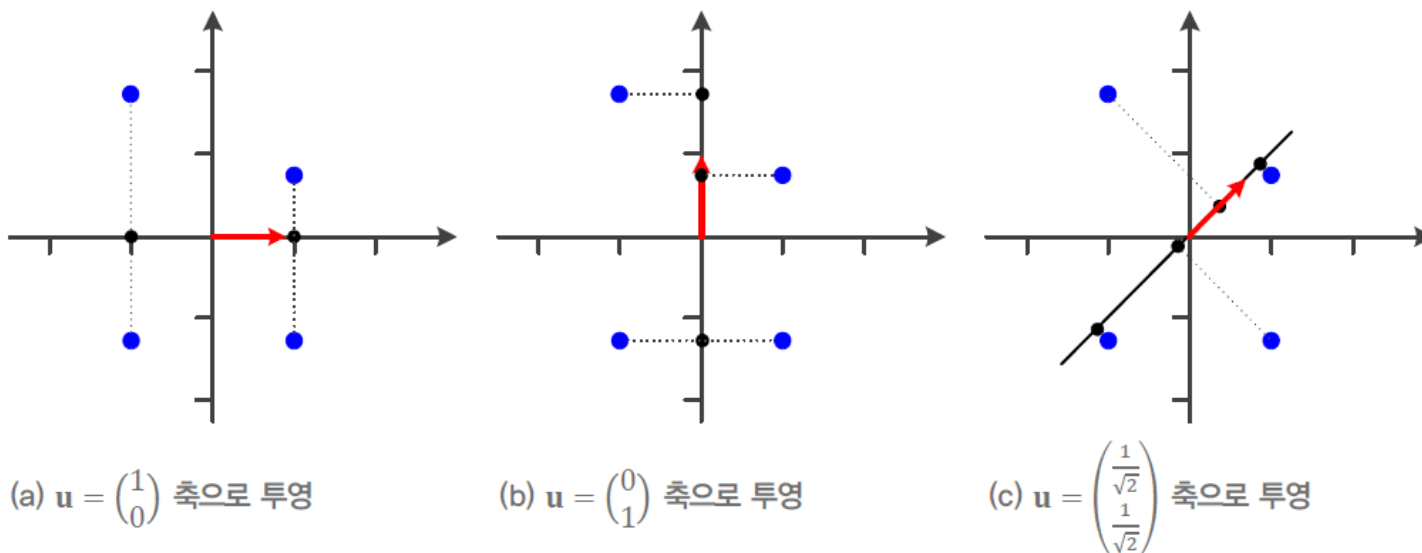


그림 6-19 투영에 의해 2차원을 1차원으로 변환

## 6.6.1 주성분 분석

### ■ 주성분 분석의 목적

- 손실을 최소화하면서 저차원으로 변환하는 것
  - [그림 6-19]에서 정보 손실 예
    - [그림 6-19(a)]는  $x_1$ 과  $x_2$  쌍,  $x_3$ 과  $x_4$  쌍이 같은 점으로 변환되는 정보 손실
    - [그림 6-19(b)]는  $x_1$ 과  $x_3$  쌍이 같은 점으로 변환되는 정보 손실
    - [그림 6-19(c)]는 4개 점이 모두 다른 점으로 변환되어 정보 손실이 가장 적음
- 주성분 분석은 변환된 훈련집합  $\mathbb{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n\}$ 의 분산이 클수록 정보 손실이 적다고 판단

## 6.6.1 주성분 분석

### 예제 6-2 [그림 6-19]의 세 가지 경우의 분산

[그림 6-18(a)]의 훈련집합에 식 (6.19)를 적용하기 전과 후는 다음과 같다.

$$\mathbf{x}_1 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \mathbf{x}_2 = \begin{pmatrix} 2 \\ 4 \end{pmatrix}, \mathbf{x}_3 = \begin{pmatrix} 4 \\ 1 \end{pmatrix}, \mathbf{x}_4 = \begin{pmatrix} 4 \\ 3 \end{pmatrix} \Rightarrow \mathbf{x}_1 = \begin{pmatrix} -1 \\ -1.25 \end{pmatrix}, \mathbf{x}_2 = \begin{pmatrix} -1 \\ 1.75 \end{pmatrix}, \mathbf{x}_3 = \begin{pmatrix} 1 \\ -1.25 \end{pmatrix}, \mathbf{x}_4 = \begin{pmatrix} 1 \\ 0.75 \end{pmatrix}$$

[그림 6-19(a)]의  $\mathbf{u} = (1 \ 0)^T$  축으로 투영된 점은 다음과 같다.  $z_1 \sim z_4$ 의 분산은 1.00이다.

$$z_1 = (1 \ 0) \begin{pmatrix} -1 \\ -1.25 \end{pmatrix} = -1, \quad z_2 = (1 \ 0) \begin{pmatrix} -1 \\ 1.75 \end{pmatrix} = -1, \quad z_3 = (1 \ 0) \begin{pmatrix} 1 \\ -1.25 \end{pmatrix} = 1, \quad z_4 = (1 \ 0) \begin{pmatrix} 1 \\ 0.75 \end{pmatrix} = 1$$

이제 [그림 6-19(c)]의  $\mathbf{u} = \left(\frac{1}{\sqrt{2}} \ \frac{1}{\sqrt{2}}\right)^T$  축으로 투영된 점을 구해 보자.  $z_1 \sim z_4$ 의 분산은 1.0930이다.

$$z_1 = \left(\frac{1}{\sqrt{2}} \ \frac{1}{\sqrt{2}}\right) \begin{pmatrix} -1 \\ -1.25 \end{pmatrix} = -1.591, \quad z_2 = \left(\frac{1}{\sqrt{2}} \ \frac{1}{\sqrt{2}}\right) \begin{pmatrix} -1 \\ 1.75 \end{pmatrix} = 0.530,$$

$$z_3 = \left(\frac{1}{\sqrt{2}} \ \frac{1}{\sqrt{2}}\right) \begin{pmatrix} 1 \\ -1.25 \end{pmatrix} = -0.177, \quad z_4 = \left(\frac{1}{\sqrt{2}} \ \frac{1}{\sqrt{2}}\right) \begin{pmatrix} 1 \\ 0.75 \end{pmatrix} = 1.237$$

따라서  $\mathbf{u} = \left(\frac{1}{\sqrt{2}} \ \frac{1}{\sqrt{2}}\right)^T$  축이  $\mathbf{u} = (1 \ 0)^T$ 보다 우수하다고 할 수 있다. 그렇다면  $\mathbf{u} = \left(\frac{1}{\sqrt{2}} \ \frac{1}{\sqrt{2}}\right)^T$  보다 더 좋은 축이 있을까? 이제부터 최적해를 찾는 방법을 살펴보자.

## 6.6.1 주성분 분석

### ■ 주성분 분석의 최적화 문제

**문제 6.1**  $\mathbb{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n\}$ 의 분산을 최대화하는  $q$ 개의 축, 즉  $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_q$ 를 찾아라. 이 단위 벡터는 식 (6.20)에 따라 변환 행렬  $\mathbf{W}$ 를 구성한다.

- $q = 1$ 로 국한하고 분산을 쓰면,

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^n (z_i - \bar{z})^2 = \frac{1}{n} \sum_{i=1}^n z_i^2 = \frac{1}{n} \sum_{i=1}^n (\mathbf{u}^T \mathbf{x}_i)^2 = \mathbf{u}^T \Sigma \mathbf{u} \quad (6.21)$$

$\uparrow$  전처리를 통해 평균 0       $\uparrow$

$$var(\mathbf{x}^T \mathbf{u}) = \frac{1}{n} (\mathbf{x}^T \mathbf{u})^T (\mathbf{x}^T \mathbf{u}) = \frac{1}{n} \mathbf{u}^T \mathbf{x} \mathbf{x}^T \mathbf{u} = \mathbf{u}^T \left( \frac{\mathbf{x} \mathbf{x}^T}{n} \right) \mathbf{u}$$

- [문제 6.1]을 바꾸어 쓰면,

**문제 6.2** 식 (6.21)의 분산  $\sigma^2$ 을 최대로 하는  $\mathbf{u}$ 를 찾아라.

자기 공분산:  $\frac{1}{n} \begin{pmatrix} \text{dot}(X_1, X_1) & \text{dot}(X_1, X_2) & \dots & \text{dot}(X_1, X_d) \\ \text{dot}(X_2, X_1) & \text{dot}(X_2, X_2) & \dots & \text{dot}(X_2, X_d) \\ \vdots & \vdots & \ddots & \vdots \\ \text{dot}(X_d, X_1) & \text{dot}(X_d, X_2) & \dots & \text{dot}(X_d, X_d) \end{pmatrix}$





## 6.6.1 주성분 분석

### ■ 주성분 분석의 최적화 문제

- $\mathbf{u}$ 가 단위 벡터라는 사실을 적용하여 문제를 다시 쓰면,

문제 6.3  $L(\mathbf{u}) = \mathbf{u}^T \Sigma \mathbf{u} + \lambda(1 - \mathbf{u}^T \mathbf{u})$ 를 최대로 하는  $\mathbf{u}$ 를 찾아라.

- $L(\mathbf{u})$ 를  $\mathbf{u}$ 로 미분하면,  $\frac{\partial L(\mathbf{u})}{\partial \mathbf{u}} = 2\Sigma \mathbf{u} - 2\lambda \mathbf{u}$
- $\frac{\partial L}{\partial \mathbf{u}} = 0$ 을 풀면,

$$\Sigma \mathbf{u} = \lambda \mathbf{u} \quad (6.22)$$

### ■ 주성분 분석의 학습 알고리즘

1. 훈련집합으로 공분산 행렬  $\Sigma$ 를 계산한다.
2. 식 (6.22)를 풀어  $d$ 개의 고윳값과 고유벡터를 구한다.
3. 고윳값이 큰 순서대로  $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_d$ 를 나열한다. (이들을 주성분 principal component이라 부름)
4.  $q$ 개의 주성분  $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_q$ 를 선택하여 식 (6.20)에 있는 행렬  $\mathbf{W}$ 에 채운다.

## 6.6.1 주성분 분석

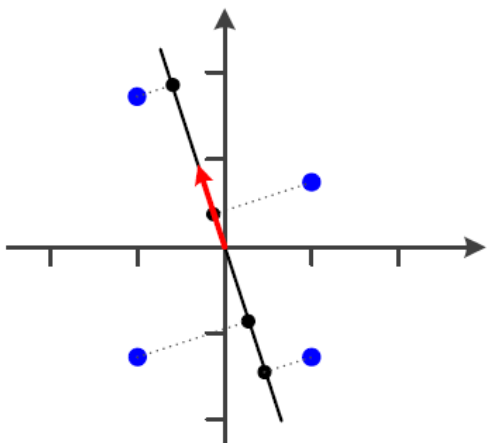
### 예제 6-3 PCA 수행

식 (6.22)를 풀어 [그림 6-18]에 있는 데이터의 최적해를 구해 보자. 먼저 공분산 행렬  $\Sigma$ 와  $\Sigma$ 의 고윳값과 고유 벡터를 구하면 다음과 같다. 공분산을 구하는 방법은 2장의 식 (2.39)를 참조하라.

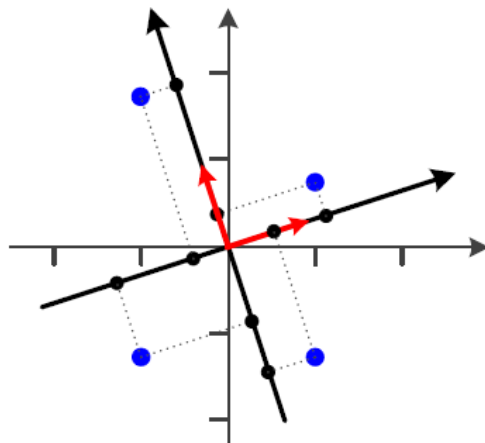
$$\Sigma = \begin{pmatrix} 1.000 & -0.250 \\ -0.250 & 1.688 \end{pmatrix}$$

$$\lambda_1 = 1.7688, \mathbf{u}_1 = \begin{pmatrix} -0.3092 \\ 0.9510 \end{pmatrix}, \lambda_2 = 0.9187, \mathbf{u}_2 = \begin{pmatrix} -0.9510 \\ -0.3092 \end{pmatrix}$$

고유 벡터 2개 중 고윳값이 큰  $\mathbf{u}_1$ 을 선택하고,  $\mathbf{u}_1$ 에 샘플 4개를 투영하면 [그림 6-20(a)]가 된다. 변환된 점의 분산은 1.7688로 [그림 6-19]에 있는 축보다 훨씬 크다는 사실을 확인할 수 있다.  $\mathbf{u}_1$ 은 PCA 알고리즘으로 찾은 최적으로서 더 좋은 축은 없다.



(a) 축 1개 사용



(b) 축 2개 사용

그림 6-20 PCA가 찾은 최적 변환

## 6.6.1 주성분 분석

### ■ 디코딩 과정

- 역변환은  $\mathbf{x} = (\mathbf{W}^T)^{-1} \mathbf{z}$ 인데,  $\mathbf{W}$ 가 정규직교<sup>orthonormal</sup> 행렬이므로 식 (6.23)이 됨

$$\tilde{\mathbf{x}} = \mathbf{W} \mathbf{z} \quad (6.23)$$

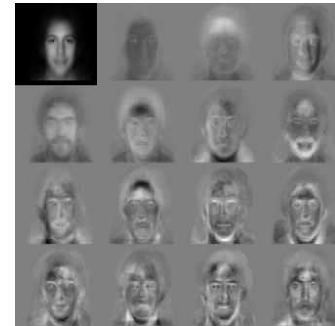
- $q = d$ 로 설정하면  $\mathbf{W}$ 가  $d * d$ 이고  $\tilde{\mathbf{x}}$ 는 원래 샘플  $\mathbf{x}$ 와 같게 됨 ([그림 6-20(b)]의 예시)
  - 원래 공간을 단지 일정한 양만큼 회전하는 것에 불과

### ■ 실제로는 $q < d$ 로 설정하여 차원 축소를 꾀함

- 많은 응용이 있음
  - 데이터 압축
  - $q = 2$  또는  $q = 3$ 으로 설정하여 2차원 또는 3차원으로 축소하여 데이터 가시화
  - 고유얼굴<sup>eigen faces</sup> 기법: 256\*256 얼굴 영상 ( $d = 65536$ )을

$q = 7$ 차원으로 변환하여 얼굴 인식 (정면 얼굴에 대해 96% 정확률)

→ 상위 몇 개의 고유벡터들이 대부분 정보를 가짐



## 6.6.2 독립 성분 분석

### ■ 블라인드 원음 분리 문제

- 실제 세계에서는 여러 신호가 섞여 나타남 ([그림 6-21]은 음악과 대화가 섞이는 예)

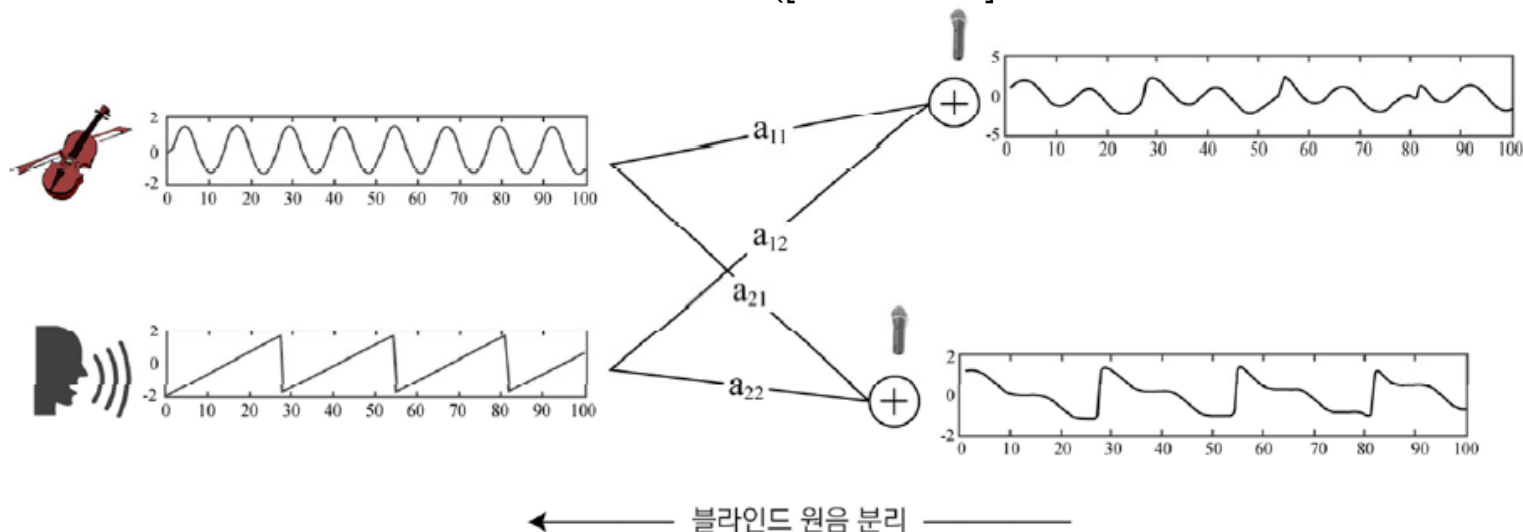


그림 6-21 블라인드 원음 분리 문제

- 마이크로 측정한 혼합 신호로부터 원음(음악과 목소리)을 복원할 수 있나?  
→ 블라인드 원음 분리 문제라 부르며 독립 성분 분석 기법으로 해결 가능
- 아주 많은 예, 뇌파와 다른 장기 신호가 섞인 EEG, 장면과 잡음이 섞인 영상, ...

## 6.6.2 독립 성분 분석

### ■ 문제 정의

#### ■ 표기

- 원래 신호를  $z_1(t)$ 와  $z_2(t)$ , 측정된 혼합 신호를  $x_1(t)$ 와  $x_2(t)$ 로 표기
- $t$  순간에 획득된  $\mathbf{x}_t = (x_1(t), x_2(t))^T$ 를 훈련 샘플로 취함
- 따라서 훈련집합은  $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$
- 블라인드 원음 분리 문제는  $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ 로부터  $\mathbb{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n\}$ 를 찾는 문제

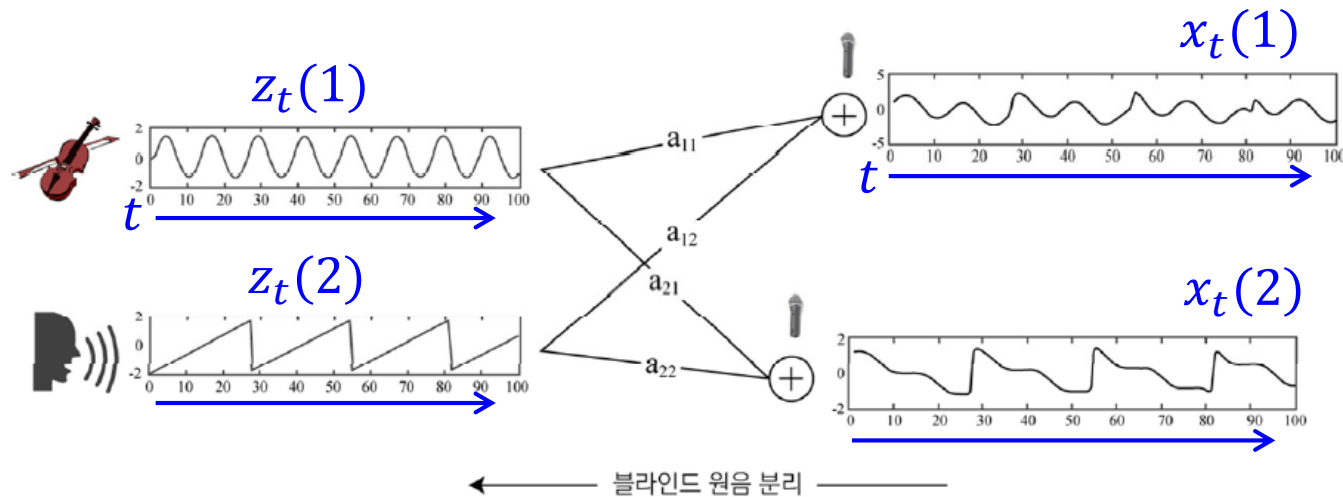


그림 6-21 블라인드 원음 분리 문제

## 6.6.2 독립 성분 분석

### ■ 문제 공식화

- 혼합 신호  $\mathbf{x}$ 를 원래 신호  $\mathbf{z}$ 의 선형 결합으로 표현 가능 ( $z_1(t)$ 와  $z_2(t)$ 가 독립이라는 가정)

$$\begin{cases} x_1 = a_{11}z_1 + a_{12}z_2 \\ x_2 = a_{21}z_1 + a_{22}z_2 \end{cases} \quad (6.24)$$

- 행렬 표기로 쓰면,

$$\mathbf{x} = \mathbf{A}\mathbf{z} \quad (6.25)$$

- 블라인드 원음 분리 문제란  $\mathbf{A}$ 를 구하는 것.  $\mathbf{A}$ 를 알면, 식 (6.26)으로 원음 복원

$$\tilde{\mathbf{z}} = \mathbf{W}\mathbf{x}, \text{ 이때 } \mathbf{W} = \mathbf{A}^{-1} \quad (6.26)$$

### ■ 식 (6.25)는 과소 조건 문제

- 정수 하나를 주고 어떤 두 수의 곱인지 알아내라는 문제와 비슷함  
(예를 들어, 32는  $1 \times 32$ ,  $2 \times 16$ ,  $4 \times 8$  등 여러 답이 가능)  $\leftarrow$  추가 조건을 주면 유일 해가 가능
- 문제도 과소 적합
- 추가 조건을 이용하여 식 (6.25)의 해를 찾음  $\rightarrow$  독립성 가정과 비가우시안 가정

## 6.6.2 독립 성분 분석

### ■ 독립성 가정

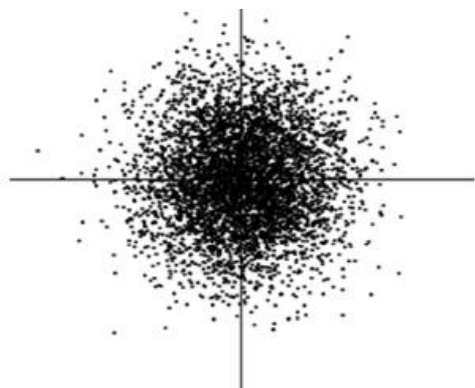
- 원래 신호가 서로 독립이라는 가정 (예, 음악과 대화는 서로 무관하게 발생함)

$$P(\mathbf{z}) = P(z_1, z_2, \dots, z_d) = \prod_{j=1}^d P(z_j) \quad (6.27)$$

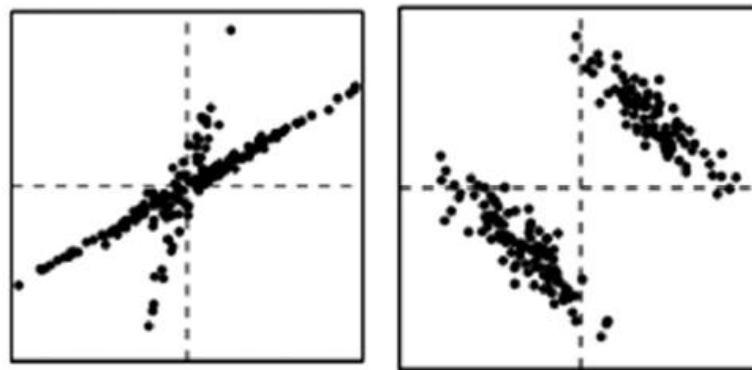
### ■ 비가우시안 nongaussianity 가정

- 원래 신호가 가우시안이라면

혼합 신호도 ([그림 6-22(a)]처럼) 가우시안이 되므로 분리할 실마리 없음  
하지만, 비가우시안이면 ([그림 6-22(b)]처럼) 실마리가 있음



(a) 확률변수가 가우시안일 때



(b) 확률변수가 비가우시안일 때

그림 6-22 서로 독립인 두 확률변수의 결합 분포

## 6.6.2 독립 성분 분석

### ■ 독립 성분 분석ICA의 문제 풀이

- 원래 신호  $z$ 의 비가우시안인 정도를 최대화하는 가중치를 구하는 전략 사용
  - 원래 신호를 식으로 쓰면,

$$\left. \begin{aligned} z_j &= w_{j1}x_1 + w_{j2}x_2 \\ \text{행렬 형태로 쓰면 } z_j &= \mathbf{w}_j \mathbf{x} \end{aligned} \right\} \quad (6.28)$$

- 비가우시안을 최대화하는 가중치를 구하는 식을 쓰면,

$$\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\operatorname{argmax}} \check{G}(z_j) \quad (6.29)$$

- $\check{G}$ 는 비가우시안 정도를 측정하는 함수
- 주로 식 (6.31)의 **첨도 kurtosis**를 사용
  - » 첨도: (분포상의) 뾰족함의 정도를 나타내는 수치

$$\text{kurtosis}(z_j) = \frac{1}{n} \sum_{i=1}^n z_{ji}^4 - 3 \left( \frac{1}{n} \sum_{i=1}^n z_{ji}^2 \right)^2 \quad (6.31)$$

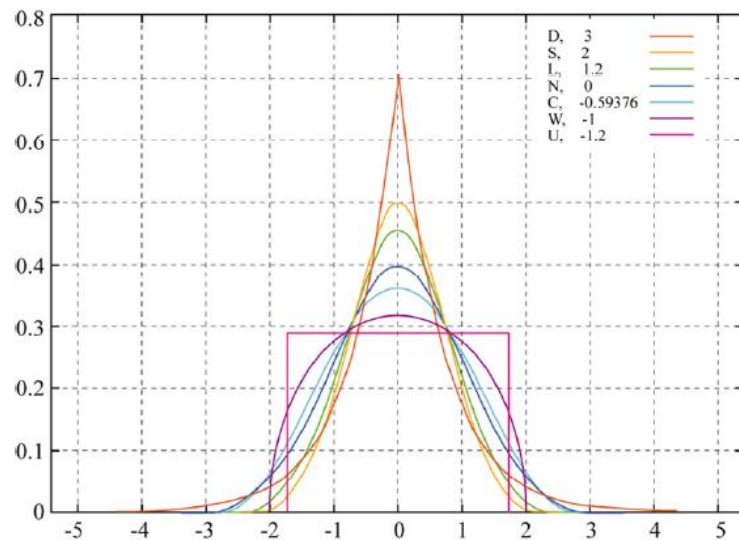


그림 6-23 여러 가지 분포의 첨도 측정

가우시안 (N)의 첨도 0

가우시안을 기준, 분포가 뾰족할수록 첨도 커지고  
평퍼짐해질수록 음수



## 6.6.2 독립 성분 분석

### ■ 독립 성분 분석 학습

#### 1. 전처리 수행

- 훈련집합  $\mathbf{x}$ 의 평균이  $\mathbf{0}$ 이 되도록 이동centering (식 (6.19) 적용)
- 식 (6.30)의 화이트닝whitening 변환 적용

$$\mathbf{x}'_i = \left( \mathbf{D}^{-\frac{1}{2}} \mathbf{V}^T \right) \mathbf{x}_i, i = 1, 2, \dots, n \quad (6.30)$$

#### 2. 식 (6.29)를 풀어 최적 가중치 구함

예. 신경망 최적화와 유사하게  $\mathbf{w}$ 를 초기화하고 첨도를 구하고, 첨도가 작아지는 방향으로 개선

### ■ PCA와 ICA 비교

- ICA는 비가우시안과 독립성 가정, PCA는 가우시안과 비상관을 가정
- ICA는 4차 모멘트까지 사용, PCA는 2차 모멘트까지 사용
- ICA로 찾은 축은 수직 아님, PCA로 찾은 축은 서로 수직
- ICA는 주로 블라인드 원음 분리 문제를 푸는데, PCA는 차원 축소 문제를 푼다

## 6.6.3 희소 코딩

### ■ 기저함수 basis function 또는 기저벡터 basis vector의 선형 결합으로 신호를 표현

- 푸리에 변환 Fourier transform ([그림 6-24(a)) 또는 웨이블릿 변환 wavelet transform 등

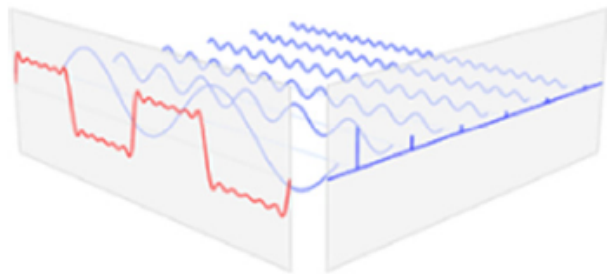
### ■ 희소 코딩 sparse coding

- 사전 dictionary  $\mathbf{D}$ 를 구성하는 기저 벡터 (단어 atoms)  $\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_m$ 의 선형 결합으로 신호(영상)  $\mathbf{x}$ 를 표현

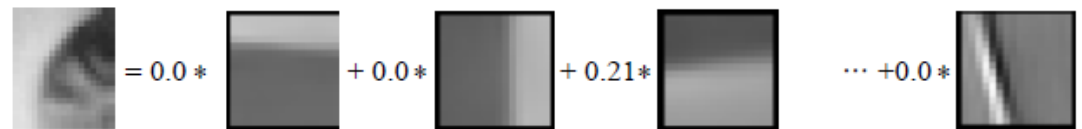
$$\mathbf{x} = \mathbf{D}\mathbf{a}$$

이때  $\mathbf{D} = (\mathbf{d}_1 \ \mathbf{d}_2 \ \dots \ \mathbf{d}_m)$

(6.32)



(a) 푸리에 변환

The equation shows a grayscale image  $\mathbf{x}$  on the left, followed by an equals sign and a series of terms:  $0.0 * \mathbf{d}_1 + 0.0 * \mathbf{d}_2 + 0.21 * \mathbf{d}_3 + \dots + 0.0 * \mathbf{d}_m$ . Each  $\mathbf{d}_i$  is represented by a small grayscale image patch.

사전  $\mathbf{D} = \left\{ \begin{array}{c} \mathbf{d}_1 \ \mathbf{d}_2 \ \mathbf{d}_3 \ \dots \ \mathbf{d}_m \end{array} \right\}$

희소 코드  $\mathbf{a} = (0.0, 0.0, 0.21, \dots, 0.0)$

(b) 희소 코딩

그림 6-24 신호를 기저함수 또는 기저 벡터의 선형 결합으로 근사 표현

## 6.6.3 희소 코딩

### ■ 희소 코딩이 다른 변환 기법과 다른 점

- 비지도 학습이 사전 (즉 기저 벡터)를 자동으로 알아냄 (푸리에 변환은 삼각함수를 사용함)  
→ 희소 코딩은 데이터에 맞는 기저 벡터를 사용하는 셈
- 사전의 크기 ( $m$ )를 기저벡터의 크기 ( $d$ )보다 과잉 완벽하게 채정 over-complete spanning set ( $m > d$ )
- 희소 코드  $\mathbf{a}$ 를 구성하는 요소 대부분이 0 값을 가짐

### ■ 희소 코딩 구현

- 최적의 사전과 최적의 희소 코드를 알아내야 함
- $\phi$ 는 희소 코드의 희소성을 강제 (많은 계수들을 0으로 만듦)하는 규제항

$$\hat{\mathbf{D}}, \hat{\mathbf{A}} = \underset{\mathbf{D}, \mathbf{A}}{\operatorname{argmin}} \sum_{i=1}^n \|\mathbf{x}_i - \mathbf{D}\mathbf{a}_i\|_2^2 + \lambda \phi(\mathbf{a}_i) \quad (6.33)$$

- 결과적으로  $x = 0 \cdot d_1 + 0 \cdot d_2 + 0.4 \cdot d_3 + \dots + 0 \cdot d_{m-1} + 0.1 \cdot d_m$ 처럼

최대한 많이 0이 되게 하는 선형 계수를 찾는 학습 알고리즘

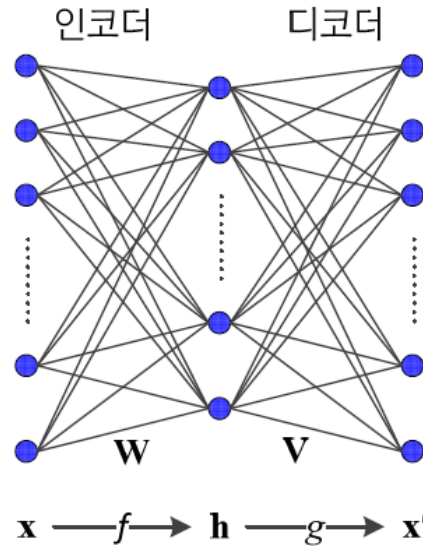
## 6.7 오토인코더

- 6.7.1 규제 오토인코더
- 6.7.2 적층 오토인코더

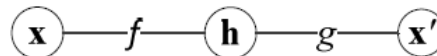
## 6.7 오토인코더

### ■ 오토인코더 autoencoder

- 특징 벡터  $\mathbf{x}$ 를 입력 받아 동일한 또는 유사한 벡터  $\mathbf{x}'$ 를 출력하는 신경망



(a) 오토인코더의 구조와 동작



(b) 축약형

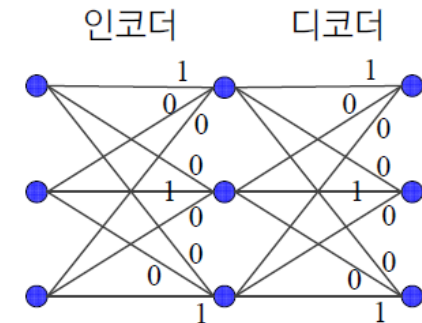


그림 6-26 단위 행렬로 오토인코더 구현

- 단순 복사하는 단위 행렬([그림 6-26])은 무의미

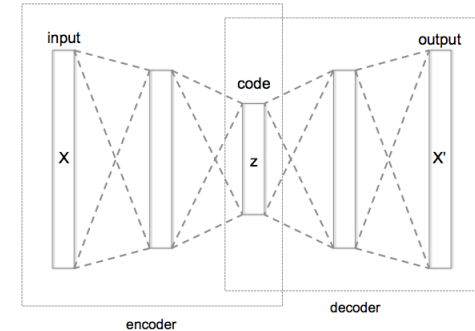
$$\mathbf{x} = g(f(\mathbf{x})) = \mathbf{VW}\mathbf{x} = \mathbf{I}\mathbf{x}$$

- 일반적으로 차원 감소를 위해 데이터 집합에 대한 표현을 학습함
- 여러 가지 규제 기법 적용하여 유용한 신경망으로 활용

## 6.7 오토인코더

### ■ 병목 구조 오토인코더의 동작 원리

- $m < d$ 인 구조



(예, 256\*256 영상을 입력 받아 256\*256 영상을 출력하는 경우  $d = 65536$ 인데  $m = 1024$ 로 설정)

- 은닉층의  $\mathbf{h}$ 는 훨씬 적은 메모리로 데이터 특징 표현. 필요하다면 디코더로 원래 데이터 복원
- $\mathbf{h}$ 는  $\mathbf{x}$ 의 핵심 정보를 표현  $\rightarrow$  특징 추출, 영상 압축 등의 응용

### ■ 여러 형태의 오토인코더

- 은닉 노드의 개수에 따라  $m < d$ ,  $m = d$ ,  $m > d$  구조
- 활성화함수에 따라 선형 (식 (6.34))과 비선형 (식 (6.35))

$$\left. \begin{aligned} \mathbf{h} &= f(\mathbf{x}) = \mathbf{W}\mathbf{x} \\ \mathbf{x} &= g(\mathbf{h}) = \mathbf{V}\mathbf{h} \end{aligned} \right\} \quad (6.34)$$

$$\left. \begin{aligned} \mathbf{h} &= f(\mathbf{x}) = \tau_{\text{encode}}(\mathbf{W}\mathbf{x}) \\ \mathbf{x} &= g(\mathbf{h}) = \tau_{\text{decode}}(\mathbf{V}\mathbf{h}) \end{aligned} \right\} \quad (6.35)$$

## 6.7 오토인코더

### ■ 오토인코더의 학습

- 주어진 데이터는 훈련집합  $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ ,

알아내야 하는 매개변수는  $f$ 와  $g$ 라는 매핑함수, 즉 가중치집합  $\theta = \{\mathbf{W}, \mathbf{V}\}$

- 오토인코더 학습을 최적화 문제로 쓰면,

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^n L(\mathbf{x}_i, g(f(\mathbf{x}_i))) \quad (6.36)$$

$$L(\mathbf{x}_i, g(f(\mathbf{x}_i))) = \|\mathbf{x}_i - g(f(\mathbf{x}_i))\|_2^2 \quad (6.37)$$

## 6.7.1 규제 오토인코더

### ■ 여러 규제 기법을 적용

- $m > d$ 인 상황에서도 단순 복사를 피할 수 있음

← 충분히 큰 모델을 사용하되 적절한 규제 기법을 적용하는 현대 기계 학습 추세를 따름

### ■ SAE (sparse autoencoder): 희소한 독립적 인자들로 표현 가능

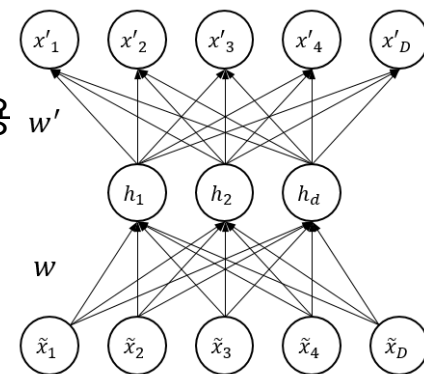
- 은닉 벡터  $\mathbf{h}_i$ 가 희소하도록 강제화 (0이 아닌 요소의 개수를 적게 유지)
- $\phi(\mathbf{h}_i)$ : 벡터  $\mathbf{h}_i$ 가 희소하도록 강제하는 규제항 (가중치 감쇄) + 활성화함수 제한 규제

$$\text{SAE: } \hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \sum_{i=1}^n L(\mathbf{x}_i, g(f(\mathbf{x}_i))) + \lambda \phi(\mathbf{h}_i) \quad (6.38)$$

### ■ DAE (denoising autoencoder): 입력의 부분 손실에도 견고한 표현 학습

- 잡음을 추가한 다음, 원본을 복원하도록 학습하는 원리
- 특징 벡터  $\mathbf{x}_i$ 에 적절한 양의 잡음을 추가한  $\tilde{\mathbf{x}}_i$ 를 입력으로 사용

$$\text{DAE: } \hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \sum_{i=1}^n L(\mathbf{x}_i, g(f(\tilde{\mathbf{x}}_i))) \quad (6.39)$$



$$\tilde{\mathbf{x}} = \mathbf{x} + \text{noise}$$



## 6.7.1 규제 오토인코더

- CAE (contractive autoencoder): 데이터의 작은 변화에 견고한 표현 학습
  - 인코더함수  $f$ 의 야코비안 행렬의 프로베니우스 놈을 작게 유지

$$\left. \begin{aligned} \text{CAE: } \hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} \sum_{i=1}^n L(\mathbf{x}_i, g(f(\mathbf{x}_i))) + \lambda \phi(\mathbf{x}_i, \mathbf{h}_i) \\ \text{이때 } \phi(\mathbf{x}_i, \mathbf{h}_i) = \left\| \frac{\partial f}{\partial \mathbf{x}} \right\|_F^2 \end{aligned} \right\} \quad (6.40)$$

- CAE는 공간을 축소<sup>contraction</sup>하는 효과  $\rightarrow$  SAE + DAE 효과

## 6.7.2 적층 오토인코더

### ■ 오토인코더는 얇은 신경망

- 은닉층이 하나뿐임 → 표현력에 한계
- 여러 층으로 쌓으면 용량이 커짐

### ■ 적층 오토인코더 stacked autoencoder

- 층별 예비학습 layerwise pretraining을 이용하여 [그림 6-29]처럼 깊은 신경망을 만들

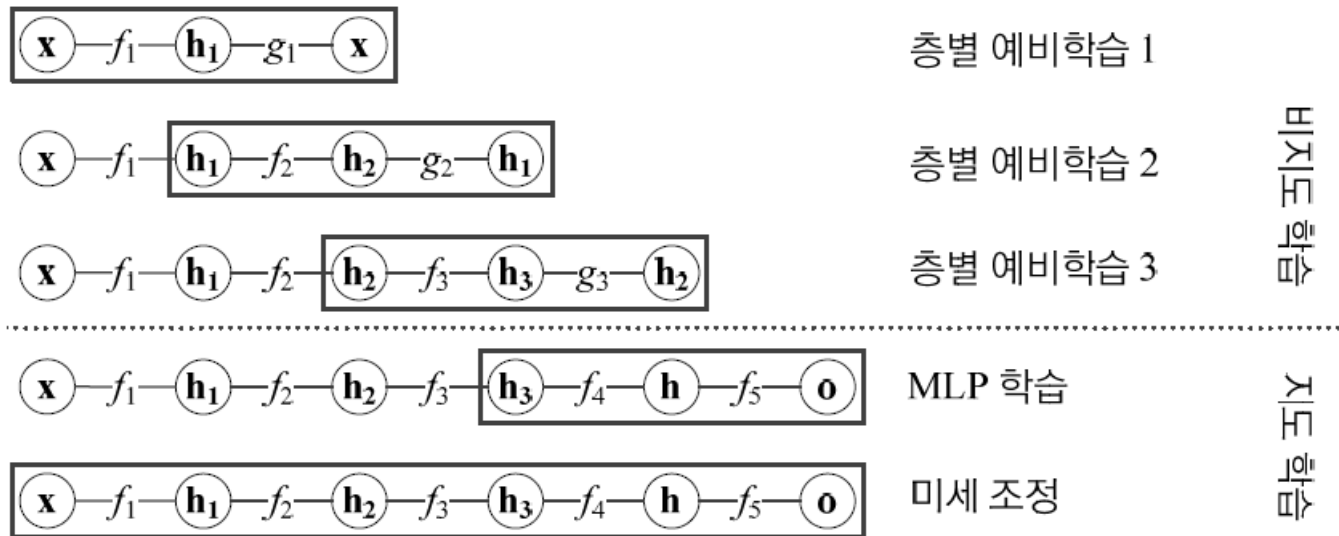


그림 6-29 적층 오토인코더의 학습 과정

## 6.7.2 적층 오토인코더

### ■ 적층 오토인코더를 지도 학습 (분류)에 활용하는 경우의 학습 과정

1. 층별 예비학습을 필요한 만큼 수행한다. ( $\mathbb{X}$ 만 가지고 비지도 학습) // [그림6-29]는 1-3번 수행
2. 마지막 층의 출력을 입력으로 하여, MLP를 학습한다. ( $\mathbb{X}$ 와  $\mathbb{Y}$ 를 가지고 지도 학습)  
// [그림6-29]는  $\mathbf{h}_3$ 가 마지막 층의 출력임
3. 신경망 전체를 한꺼번에 추가로 학습한다. // 미세 조정 단계

### ■ 층별 예비학습의 변천사

- 깊은 구조의 MLP 학습이 번번이 실패하던 상황에서  
2006년에 힌튼 교수가 층별 예비학습 아이디어를 제안하고, 성공적인 성능을 입증
- 딥러닝의 가능성을 확인, 현재 딥러닝이 기계 학습을 주도
- 여러 가지 기술 향상으로 현재는 층별 예비학습을 별로 사용하지 않음

## 6.8 매니폴드 학습

- 6.8.1 매니폴드란?
  - 6.8.2 IsoMap
  - 6.8.3 LLE
  - 6.8.4 t-SNE
  - 6.8.5 귀납적 학습 모델과 전이유추 학습 모델
- 
- 오토인코더는 데이터 구조를 간접적으로 표현
  - 반면 매니폴드 학습은 데이터의 비선형 구조를 직접적으로 반영

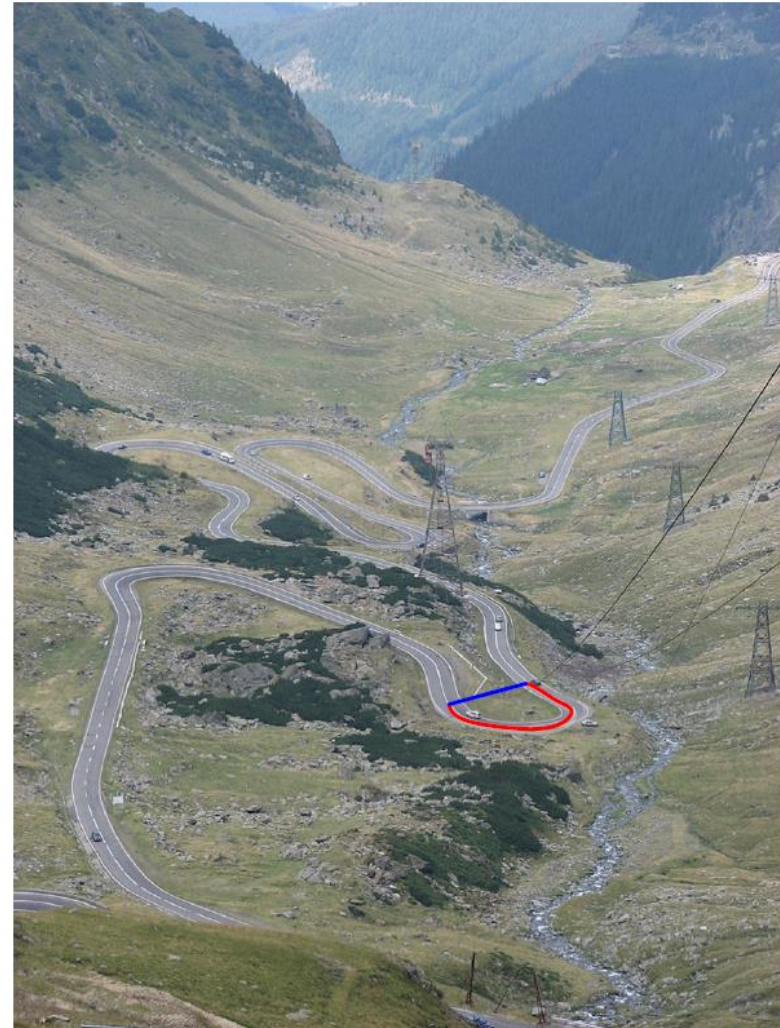
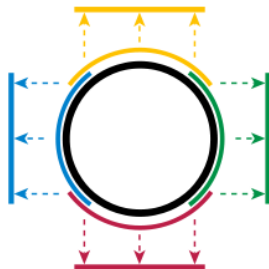
## 6.8.1 매니폴드란?

### ■ 매니폴드 manifold (많은; 여러가지의)

- 매니폴드는 고차원 공간에 내재한 저차원 공간
- [그림 6-30]에서 도로가 매니폴드에 해당
- 자동차 위치를 데이터로 간주하면,

$$\mathbf{x} = (\text{위도}, \text{경도}, \text{고도})^T$$

- 이 3차원(고차원) 공간 데이터를  $\mathbf{x} = (\text{기준점에서의 거리})^T$  라는 1차원(저차원) 공간, 즉 매니폴드로 표현할 수 있음
- 보통 매니폴드는 비선형 공간이지만 지역적으로 살펴보면 선형 구조

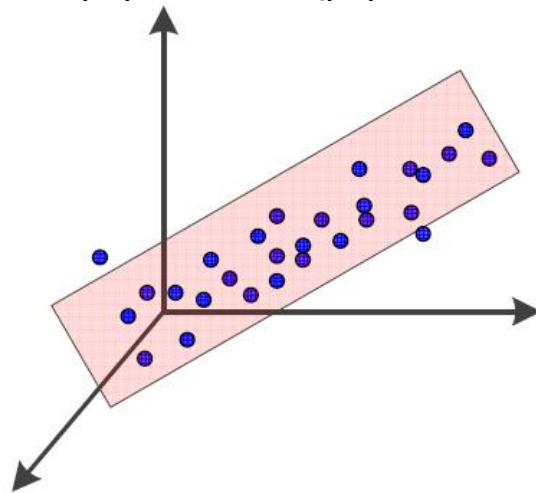


## 6.8.1 매니폴드란?

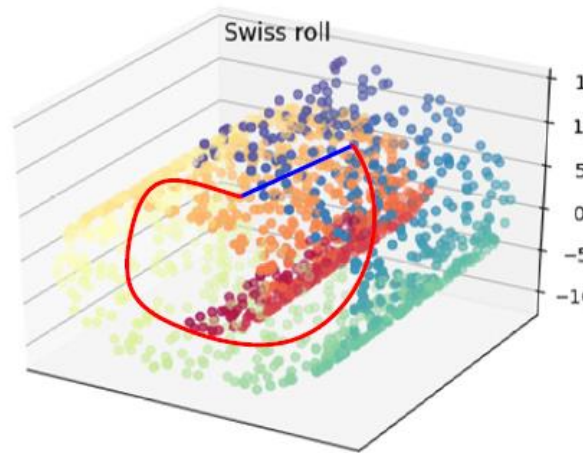
### ■ 매니폴드 가정

“real-world data presented in high-dimensional spaces are expected to concentrate in the vicinity of a manifold  $M$  of much lower dimensionality  $d_M$ , embedded in high-dimensional input space  $R^d$ . 고차원 공간에 주어진 실제 세계의 데이터는 고차원 입력 공간  $R^d$ 에 내재한 훨씬 저차원인  $d_M$ 차원 매니폴드의 인근에 집중되어 있다.”

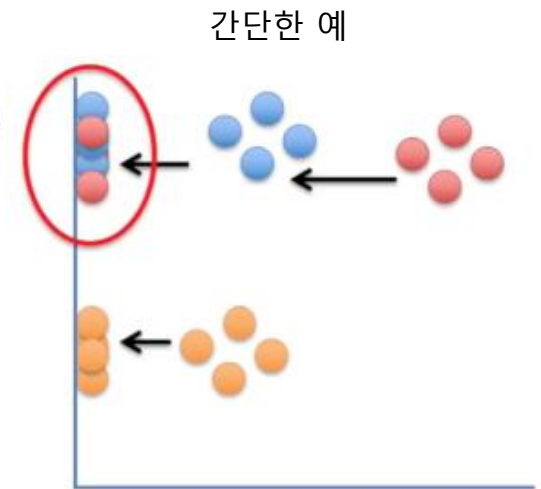
#### ■ 인위적인 상황 예시



(a) PCA로 찾은 매니폴드가 적절한 상황



(b) 비선형 매니폴드가 필요한 상황



간단한 예

→ 차원감소를 통해서 두 군집의 차이가 없어짐

그림 6-31 매니폴드 가정

### ■ 매니폴드를 어떻게 찾고, 어떻게 표현할 것인가?

## 6.8.2 IsoMap

### ■ IsoMap 알고리즘

- 최근접 이웃 그래프 구축
  - 첫째, 각 점은  $k$ -최근접 이웃을 구하여 거리를  $n * n$  행렬  $\mathbf{M}$ 에 채움
  - 둘째, 빈 곳은 최단 경로의 shortest path 길이로 채움
- $\mathbf{M}$ 의 고유 벡터를 계산하고, 큰 순서대로  $d_{low}$ 개의 고유 벡터를 선택
  - 이들 고유 벡터가 새로운 저차원 공간 형성
  - $i$ 번째 샘플의  $k$ 번째 좌표는  $\sqrt{\lambda_k} v_k^i$ 로 변환( $v_k^i$ 는  $\lambda_k$ 에 해당하는 고유 벡터의  $i$ 번째 요소)
- 예) 2차원으로 변환하는 경우,  $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ 의  $\mathbf{x}_i$ 는  $(\sqrt{\lambda_1} v_1^i, \sqrt{\lambda_2} v_2^i)^T$ 로 변환됨

### ■ $\mathbf{M}$ 의 크기가 방대한 문제점

## 6.8.3 LLE(locally linear embedding)

### ■ LLE 알고리즘

- 거리 행렬  $\mathbf{M}$  대신에 식 (6.42)의 함수  $\epsilon$ 을 최소로 하는 가중치 행렬  $\mathbf{W}$ 를 사용함

$$\epsilon(\mathbf{W}) = \sum_{i=1}^n \left\| \mathbf{x}_i - \sum_{\mathbf{x}_j \in \{\mathbf{x}_i \text{의 이웃}\}} w_{ij} \mathbf{x}_j \right\|_2^2 \quad (6.42)$$

- $\mathbf{x}_i$ 를  $k$ -최근접 이웃의 선형 결합  $\sum_{\mathbf{x}_j \in \{\mathbf{x}_i \text{의 이웃}\}} w_{ij} \mathbf{x}_j$ 로 근사화하는 셈 ([그림 6-32 왼쪽])

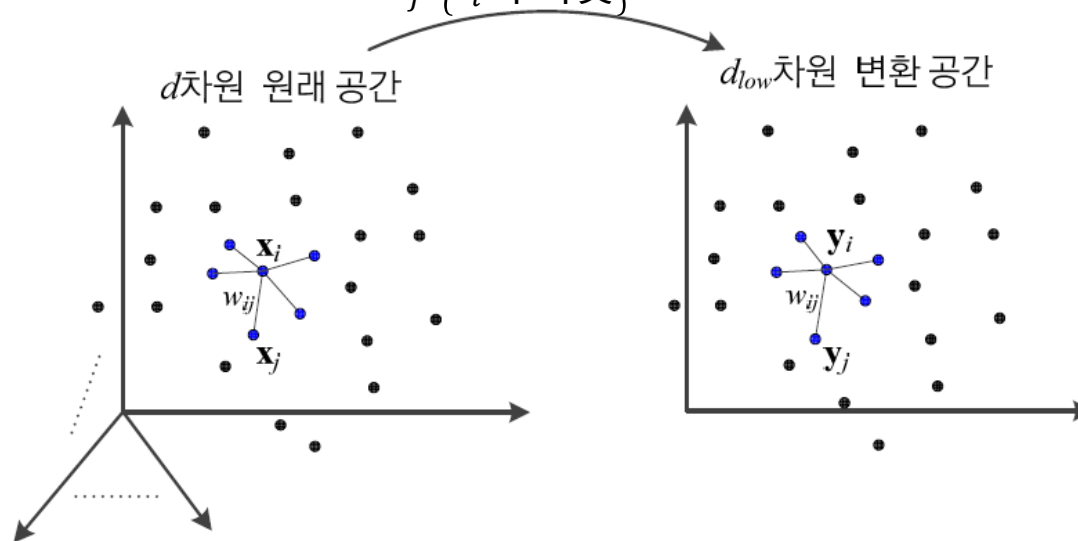


그림 6-32 LLE에서 가중치 행렬



## 6.8.3 LLE

- 저차원 공간에서는,

- 변환된 저차원 공간의 점을  $\mathbb{X}' = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n\}$ 이라면

식 (6.43)을 최소화하는  $\mathbb{X}'$ 를 찾아야 함

$$\phi(\mathbb{X}') = \sum_{i=1}^n \left\| \mathbf{y}_i - \sum_{\mathbf{y}_j \in \{\mathbf{y}_i \text{의 이웃}\}} w_{ij} \mathbf{y}_j \right\|_2^2 \quad (6.43)$$

- 고차원 원래 공간에서의 식 (6.42)와 저차원 변환 공간에서 식 (6.43)을 비슷하게 유지함으로써 원래 데이터  $\mathbb{X}$ 와 변환된 데이터  $\mathbb{X}'$ 가 비슷한 구조를 가짐

## 6.8.4 t-SNE (t-distributed stochastic neighbor embedding)

- 현재 t-SNE는 매니폴드 공간 변환 기법 중에서 가장 뛰어남

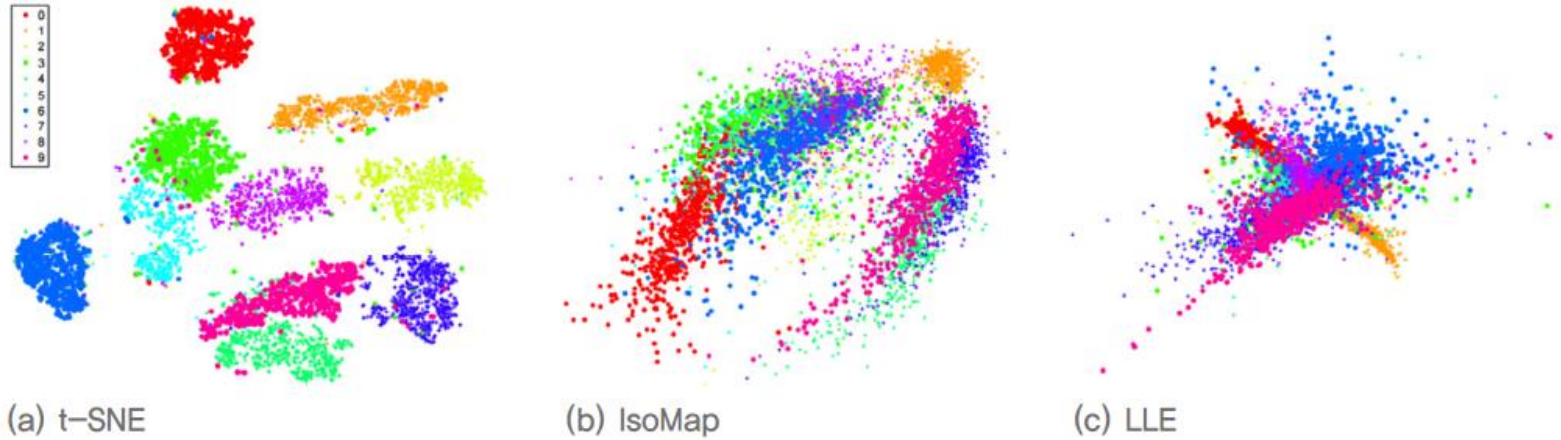


그림 6-33 MNIST를 이용한 매니폴드 학습 기법의 성능 비교

- 원래 공간에서 유사도 측정

- $\mathbf{x}_i$ 와  $\mathbf{x}_j$ 의 유사도를 식 (6.45)의 확률로 측정

$$p_{j|i} = \frac{\exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_k\|_2^2}{2\sigma_i^2}\right)} \quad (6.44)$$

$$p_{ij} = p_{ji} = \frac{p_{j|i} + p_{i|j}}{2n} \quad (6.45)$$

## 6.8.4 t-SNE (t-distributed stochastic neighbor embedding)

- 변환된 공간에서의 유사도는 학생-t 분포 student-t distribution로 측정

- $y_i$ 와  $y_j$ 는 변환된 공간에서의 점

$$q_{ij} = \frac{(1 + \|y_i - y_j\|_2^2)^{-1}}{\sum_{k \neq i} (1 + \|y_i - y_k\|_2^2)^{-1}} \quad (6.46)$$

- 원래 데이터와 변환된 데이터의 구조가 비슷해야 하므로,

- 확률 분포  $P$ 와  $Q$ 가 비슷할수록 좋음
- 비슷한 정도를 측정하기 위해 식 (6.47)의 KL 다이버전스를 사용

$$J(\mathbb{X}') = KL(P \parallel Q) = \sum_{i=1}^n \sum_{j=1}^n p_{ij} \log \left( \frac{p_{ij}}{q_{ij}} \right) \quad (6.47)$$

## 6.8.4 t-SNE (t-distributed stochastic neighbor embedding)

### ■ 학습 알고리즘

- 목적함수  $J$ 를 최소로 하는, 즉  $P$ 와  $Q$ 의 KL 다이버전스를 최소로 하는  $\mathbb{X}'$ 를 찾는 문제
- 경사 하강법을 이용(식 (6.48))

$$\frac{\partial J}{\partial \mathbf{y}_i} = 4 \sum_{j=1}^n (p_{ij} - q_{ij}) (\mathbf{y}_i - \mathbf{y}_j) \left(1 + \|\mathbf{y}_i - \mathbf{y}_j\|_2^2\right)^{-1} \quad (6.48)$$

#### 알고리즘 6-5 t-SNE

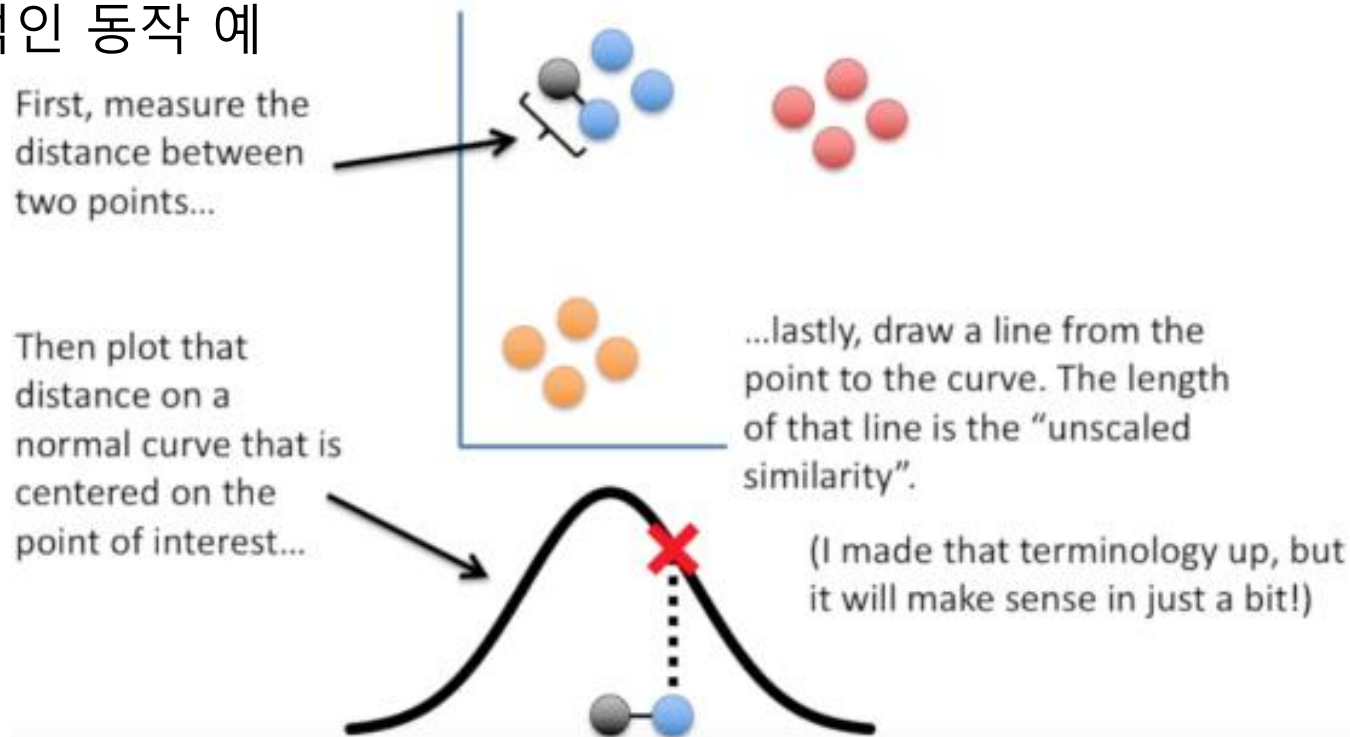
**입력:**  $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ , 반복 횟수  $T$ , 학습률  $\rho$ , 모멘텀 계수  $\alpha$

**출력:**  $\mathbb{X}' = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n\}$

- 1  $\mathbb{X}$ 의 모든 샘플 쌍에 대해 식 (6.45)로  $p_{ij}$ 를 계산한다.
- 2  $N(0, 10^{-4}\mathbf{I})$  가우시안 분포로부터 초기해  $\mathbb{X}'^{(0)} = \{\mathbf{y}_1^{(0)}, \mathbf{y}_2^{(0)}, \dots, \mathbf{y}_n^{(0)}\}$ 을 샘플링한다.
- 3 for ( $t=1$  to  $T$ )
- 4     식 (6.46)으로  $\mathbb{X}'^{(t-1)}$ 의 모든 쌍에 대해  $q_{ij}$ 를 계산한다.
- 5     for ( $i=1$  to  $n$ )
- 6         식 (6.48)로 그래디언트  $\frac{\partial J}{\partial \mathbf{y}_i}$ 를 계산한다.
- 7         if ( $t > 1$ )  $\mathbf{y}_i^{(t)} = \mathbf{y}_i^{(t-1)} + \eta \frac{\partial J}{\partial \mathbf{y}_i} + \alpha(\mathbf{y}_i^{(t-1)} - \mathbf{y}_i^{(t-2)})$  // 학습률과 모멘텀 적용
- 8         else  $\mathbf{y}_i^{(t)} = \mathbf{y}_i^{(t-1)} + \eta \frac{\partial J}{\partial \mathbf{y}_i}$

## 6.8.4 t-SNE (t-distributed stochastic neighbor embedding)

### ■ 직관적인 동작 예

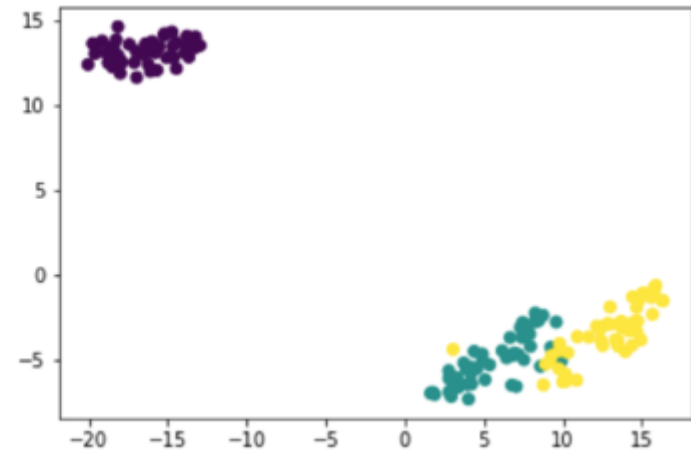
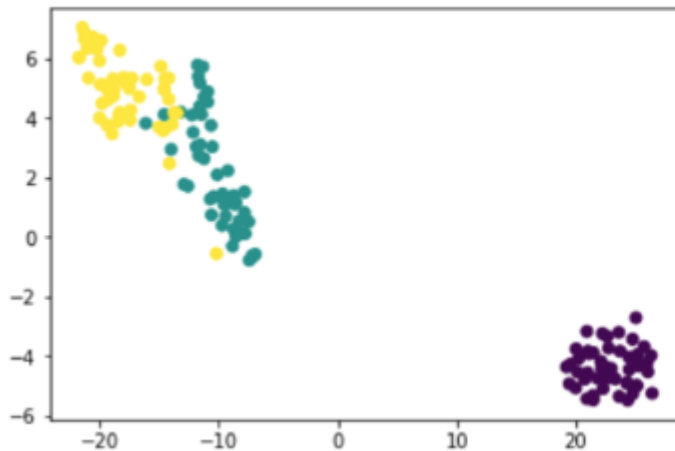


1. 임의의 점 선택 (검정 점)
2. 선택된 검정 점으로부터 다른 점들까지의 거리 측정
3. t 분포의 검정 점을 중심 (기준)에 있다고 할 때, 다른 점들은 상대적인 거리에 의해 값 선택  
이 값을 유사도 (친밀도)로 하고, 가까운 값끼리 군집

## 6.8.4 t-SNE (t-distributed stochastic neighbor embedding)

### ■ 단점

- 낮은 차원으로 잘 변환하는 장점이 있지만
- 수행할 때마다 축의 위치가 변화하는 특성 때문에 모양이 매번 변화함
- 따라서 데이터의 고유 특성을 시각화하는 분석에는 적합하지만 학습 특징으로 부적합함



## 6.8.5 귀납적 학습 모델과 전이유추 학습 모델

### ■ 전이유추<sup>transductive</sup> 학습 모델

- 관찰된 훈련 사례에서 시험 사례로 추론됨
- 훈련집합 이외의 새로운 샘플을 처리할 능력이 없는 모델
- IsoMap, LLE, t-SNE는 모두 트랜스덕티브 모델
- 데이터 가시화라는 목적에 관한 한 PCA나 오토인코더와 같은 귀납적 모델보다 성능이 뛰어남

“If you possess a restricted amount of information for solving some problem, try to solve the problem directly and never solve a more general problem as an intermediate problem. 어떤 문제를 풀 때 데이터가 제한되어 있으면, 그 문제를 직접 풀어야 한다. 중간 문제로서 좀 더 일반적인 문제를 풀 필요가 전혀 없다.”

### ■ 귀납적<sup>induction</sup> 모델

- 관찰된 훈련 사례에서 일반적 규칙에 이르기 까지 추론한 다음 시험 사례 적용
- 훈련집합 이외의 새로운 샘플을 처리할 능력이 있는 모델
- IsoMap, LLE, t-SNE를 제외한 지금까지 공부한 모든 모델

### ■ 주어진 문제에 따라 둘 중 적절한 것을 선택하는 지혜 필요