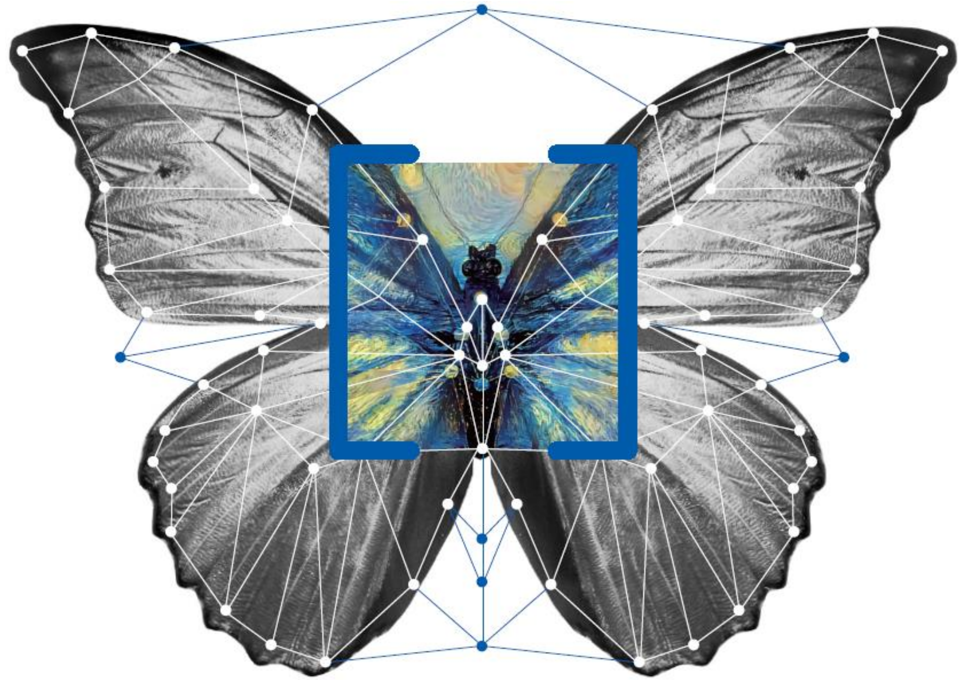
한빛아카데미
Hanbit Academy, Inc.



MACHINE LEARNING 기계 학습

오일석 지음

6장. 비지도 학습

6.5 공간 변환의 이해

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

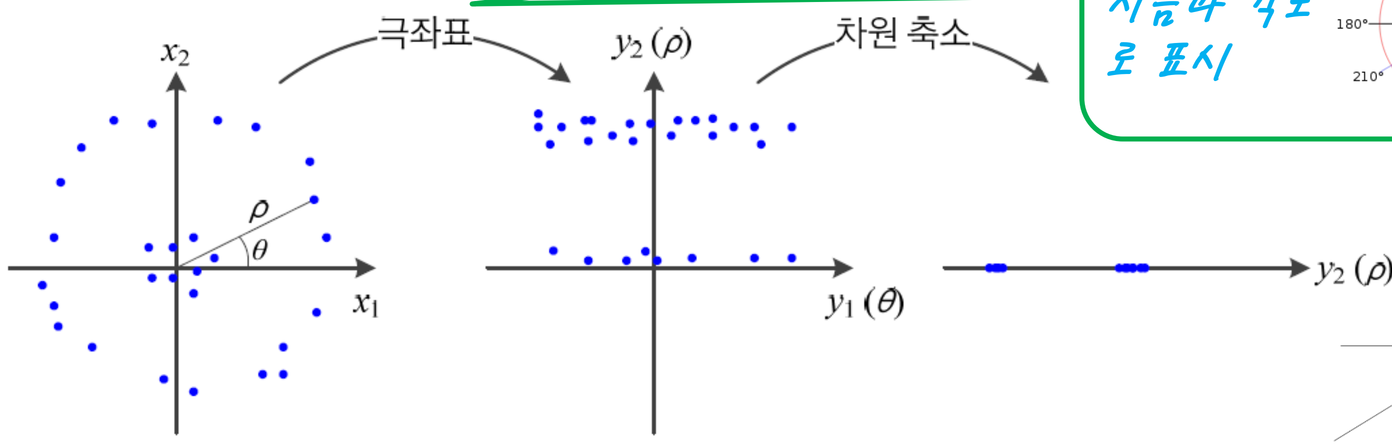
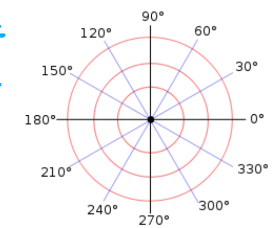
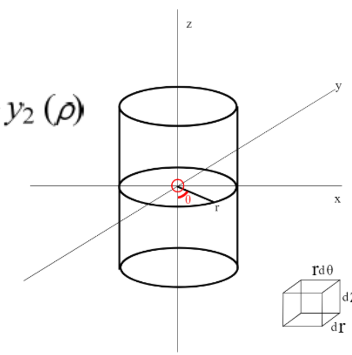


그림 6-16 공간 변환의 예

좌표계를 반지름과 각도로 표시





cylindrical coordinate system

- 실제 문제에서는 비지도 학습을 이용하여 최적의 공간 변환을 자동으로 알아내야 함

6.5 공간 변환의 이해

인코딩과 디코딩

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

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

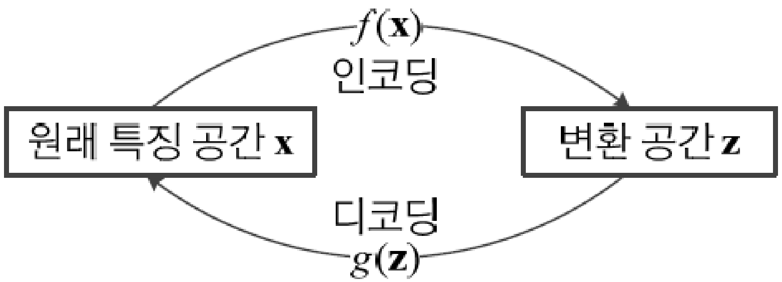


그림 6-17 공간 변환과 역변환

- 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. 디코딩은 불필요

또는 불가능

6.6 선형 인자 모델

- 6.6.1 주성분 분석
- 6.6.2 독립 성분 분석

6.6 선형 인자 모델

선형 인자 모델

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

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

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

- $\boldsymbol{\alpha}$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 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 PCA[probabilistic PCA]
  - $\mathbf{z}$가 비가우시안 분포를 따른다고 가정하는 ICA(6.6.2절)

6.6.1 주성분 분석

- 데이터를 원점 중심으로 옮기는 전처리를 먼저 수행

$$\left.\begin{aligned} &\mathbf{x}_i = \mathbf{x}_i - \boldsymbol{\mu}, \qquad i = 1,2,\cdots,n \\ &\text{이때 } \boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i \end{aligned}\right\} \qquad (6.19)$$

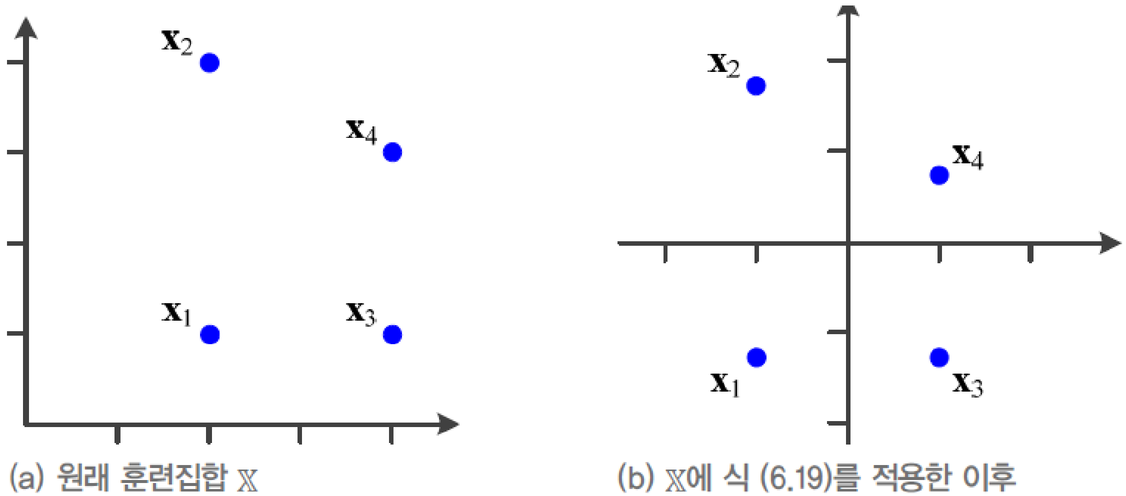


그림 6-18 𝕏의 평균을 0으로 변환

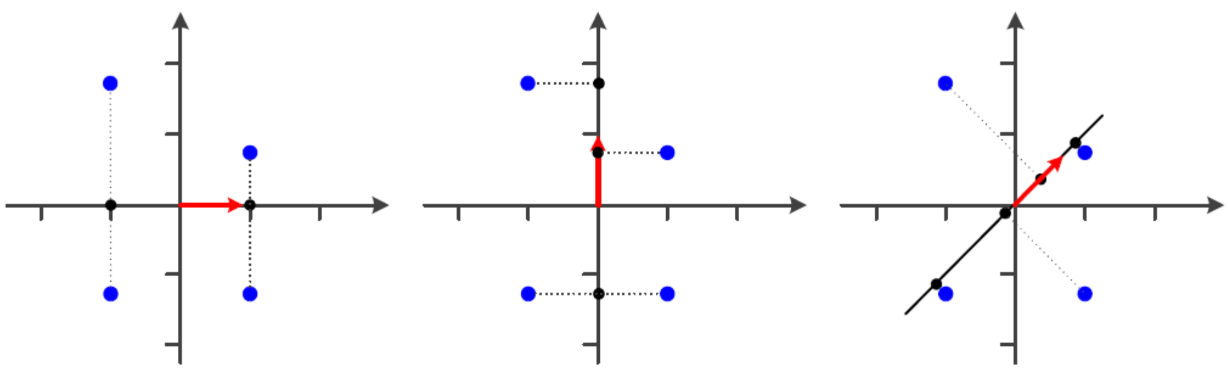
6.6.1 주성분 분석(Principal Component Analysis; PCA)

주성분 분석이 사용하는 변환식

- 다양한 세부분석을 위해 필요에 따라 고차원의 특징벡터를 저차원의 특징벡터로 변환
- 변환 행렬 $\mathbf{W}$는 $d * q$로서 주성분 분석은 d차원의 $\mathbf{x}$를 q차원의 $\mathbf{z}$로 변환 $(q < d)$
  - $\mathbf{W}$의 j번째 열 벡터와의 내적 $\mathbf{u}_j^{\mathrm{T}}\mathbf{x}$는 $\mathbf{x}$를 $\mathbf{u}_j$가 가리키는 축으로 투영

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

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



(a) $\mathbf{u} = \binom{1}{0}$ 축으로 투영 (b) $\mathbf{u} = \binom{0}{1}$ 축으로 투영 (c) $\mathbf{u} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}$ 축으로 투영

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 $\mathbf{x}_1$과 $\mathbf{x}_2$ 쌍, $\mathbf{x}_3$과 $\mathbf{x}_4$ 쌍이 같은 점으로 변환되는 정보 손실
      - [그림 6-19(b)]는 $\mathbf{x}_1$과 $\mathbf{x}_3$ 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 $\mathbb{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n\}$의 분산이 클수록 정보 손실이 적다고 판단

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 $\mathbf{u}=(1\ 0)^{\mathrm{T}}$ 축으로 투영된 점은 다음과 같다. $z_1 \sim z_4$의 분산은 1.0이다.

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

이제 [그림 6-19(c)]의 $\mathbf{u}=\left(\frac{1}{\sqrt{2}}\ \frac{1}{\sqrt{2}}\right)^{\mathrm{T}}$ 축으로 투영된 점을 구해 보자. $z_1 \sim z_4$의 분산은 1.093이다.

$$z_1=\left(\frac{1}{\sqrt{2}}\frac{1}{\sqrt{2}}\right)\begin{pmatrix}-1\\-1.25\end{pmatrix}=-1.591,\ \ z_2=\left(\frac{1}{\sqrt{2}}\frac{1}{\sqrt{2}}\right)\begin{pmatrix}-1\\1.75\end{pmatrix}=0.530,$$
$$z_3=\left(\frac{1}{\sqrt{2}}\frac{1}{\sqrt{2}}\right)\begin{pmatrix}1\\-1.25\end{pmatrix}=-0.177,\ \ z_4=\left(\frac{1}{\sqrt{2}}\frac{1}{\sqrt{2}}\right)\begin{pmatrix}1\\0.75\end{pmatrix}=1.237$$

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

6.6.1 주성분 분석

■ PCA의 최적화 문제

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

$\mathbf{z} = \mathbf{W}^{\mathrm{T}}\mathbf{x}$
이때 $\mathbf{W} = (\mathbf{u}_1\ \mathbf{u}_2\ \cdots\ \mathbf{u}_q)$이고, $\mathbf{u}_j = (u_{1j}, u_{2j}, \cdots, u_{dj})^{\mathrm{T}}$

- $q = 1$로 국한하고 분산을 쓰면,

최대한 단순화시키기 위해

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

샘플평균=0

- [문제 6.1]을 바꾸어 쓰면,

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

6.6.1 주성분 분석

PCA의 최적화 문제

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

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

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

$$\mathbf{\Sigma u} = \lambda\mathbf{u} \tag{6.22}$$

주성분 분석의 학습 알고리즘

1. 훈련집합으로 공분산 행렬 $\mathbf{\Sigma}$를 계산한다.
2. 식 (6.22)를 풀어 d개의 고윳값과 고유 벡터를 구한다.
3. 고윳값이 큰 순서대로 $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \cdots, \mathbf{u}_d$를 나열한다. (이들을 주성분이라 부름)
4. q개의 주성분 $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_q$를 선택하여 식 (6.20)에 있는 행렬 $\mathbf{W}$에 채운다.

6.6.1 주성분 분석

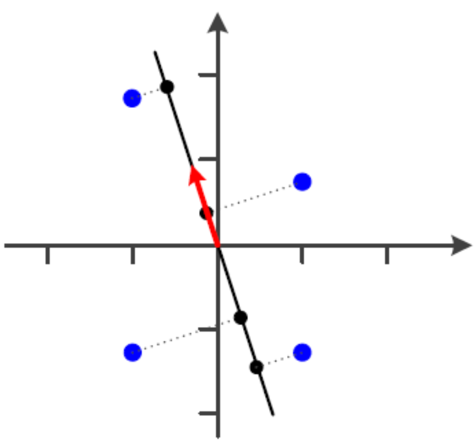
예제 6-3 **PCA 수행**

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

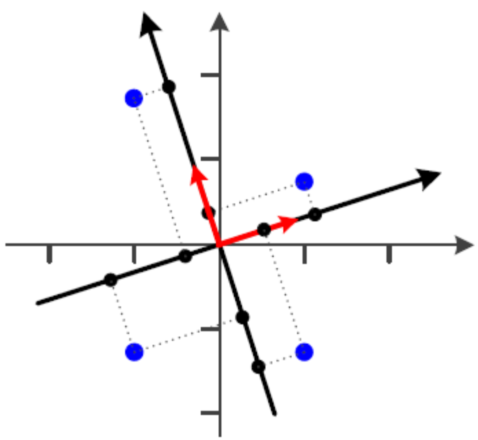
$$\boldsymbol{\Sigma} = \begin{pmatrix} 1.000 & -0.250 \\ -0.250 & 1.688 \end{pmatrix}$$

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

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 $\mathbf{x} = \left(\mathbf{W}^{\mathrm{T}}\right)^{-1}\mathbf{z}$인데, $\mathbf{W}$가 정규직교 행렬이므로 식 (6.23)이 됨

$$\tilde{\mathbf{x}} = \mathbf{W}\mathbf{z} \tag{6.23}$$

- $q = d$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 기법: 256*256 얼굴 영상($d = 65536$)을 $q = 7$차원으로 변환하여 얼굴 인식(정면 얼굴에 대해 96% 정확률) → 상위 몇 개의 고유 벡터가 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)

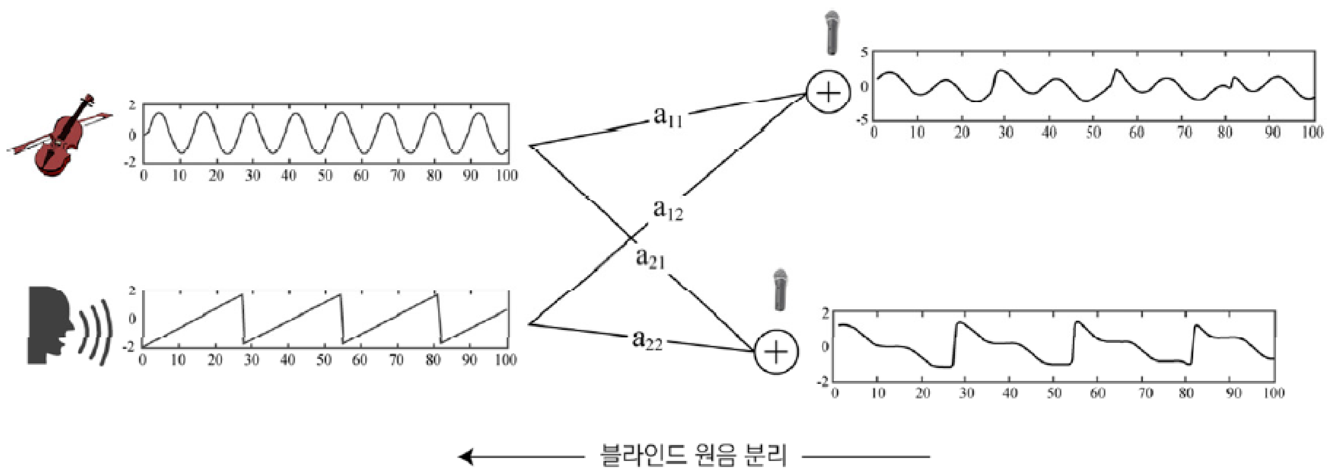


그림 6-21 블라인드 원음 분리 문제

- 마이크로 측정한 혼합 신호로부터 원음(음악과 목소리)을 복원할 수 있나? → 블라인드 원음 분리 문제라 부르며 독립 성분 분석 기법으로 해결 가능

사전정보 없거나 미흡

- 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 $\mathbf{x}_t = (x_1(t), x_2(t))^{\mathrm{T}}$를 훈련 샘플로 취함. 따라서 훈련집합은 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$
- 블라인드 원음 분리 문제는 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$로부터 $\mathbb{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n\}$를 찾는 문제

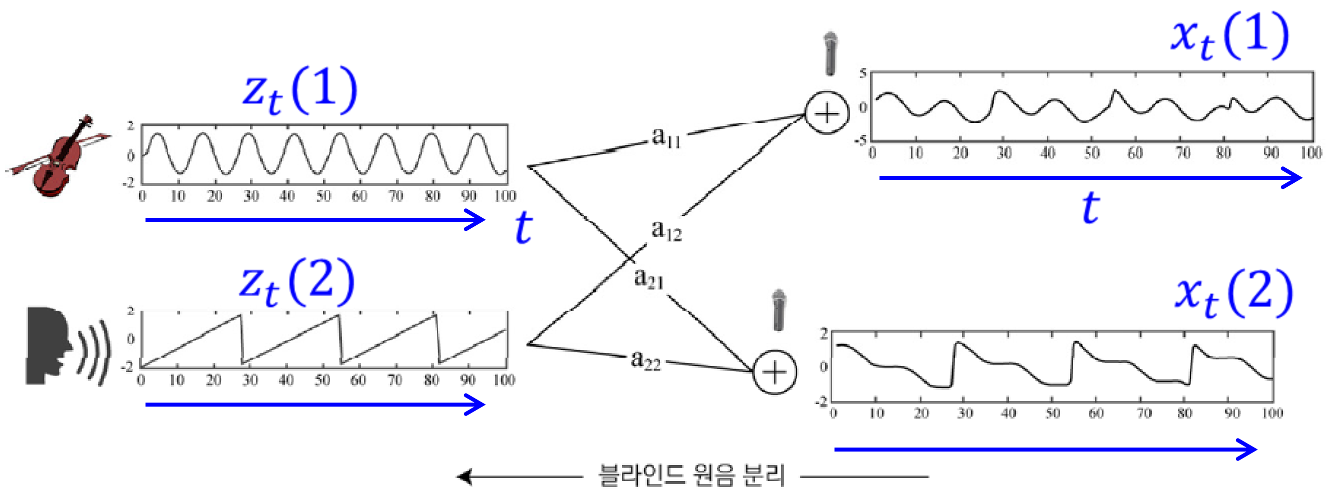


그림 6-21 블라인드 원음 분리 문제

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)

$$\left.\begin{aligned} x_1 &= a_{11}z_1 + a_{12}z_2 \\ x_2 &= a_{21}z_1 + a_{22}z_2 \end{aligned}\right\} \quad (6.24)$$

- 행렬 표기로 쓰면,

$$\mathbf{x} = \mathbf{Az} \quad (6.25)$$

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

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

식 (6.25)는 과소 조건 문제

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

이 가정을 벗어나면 사용불가능

6.6.2 독립 성분 분석

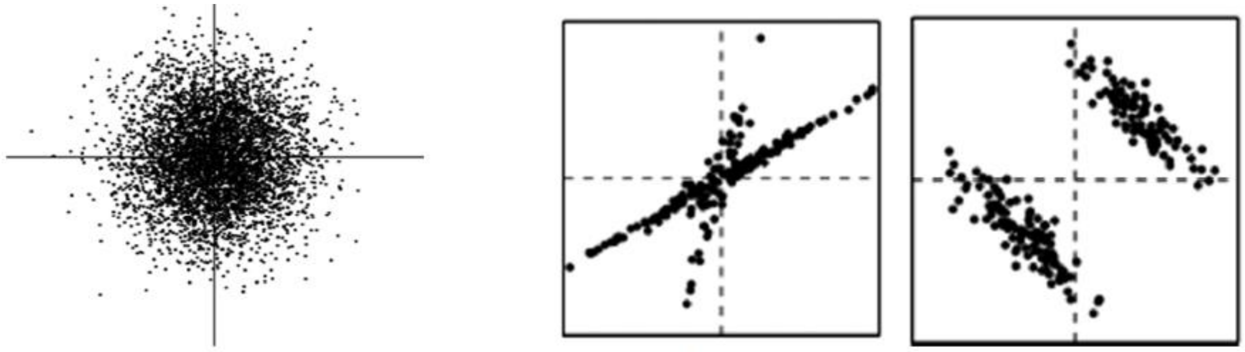
독립성 가정

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

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

비가우시안 가정

- 원래 신호가 가우시안이라면 혼합 신호도 ([그림 6-22(a)]처럼) 가우시안이 되므로 분리할 실마리 없음. 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

ICA의 문제 풀이

- 원래 신호의 비가우시안인 정도를 최대화하는 가중치를 구하는 전략 사용
  - 원래 신호를 식으로 쓰면,

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

  - 비가우시안을 최대화하는 가중치를 구하는 식을 쓰면,

$$\hat{\mathbf{w}}_j = \underset{\mathbf{w}_j}{\operatorname{argmax}}\, \breve{G}(\mathbb{z}_j) \qquad (6.29)$$

    - $\breve{G}$는 비가우시안 정도를 측정하는 함수
    - 주로 식 (6.31)의 첨도를 사용

분포가 뾰족한 모양일수록 첨도 값이 커짐

$$kurtosis(\mathbb{z}_j) = \frac{1}{n}\sum_{i=1}^{n} z_{ji}^4 - 3\left(\frac{1}{n}\sum_{i=1}^{n} z_{ji}^2\right)^2 \qquad (6.31)$$

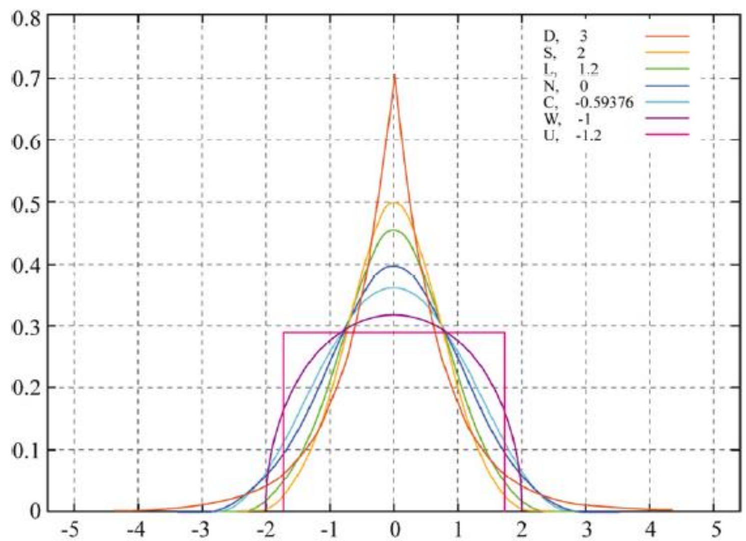


그림 6-23 여러 가지 분포의 첨도 측정

6.6.2 독립 성분 분석

ICA 학습

1. 전처리 수행
   - 훈련집합 $\mathbb{X}$의 평균이 $\mathbf{0}$이 되도록 이동(식 (6.19) 적용)
   - 식 (6.30)의 화이트닝 변환 적용

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

2. 식 (6.29)를 풀어 최적 가중치 구함

PCA와 ICA 비교

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

sklearn.decomposition.PCA

■ Principal component analysis (PCA)

(https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html#sklearn.decomposition.PCA)

- 주요 파라미터

- n_components: The number of components to keep.

  if n_components is not set all components are kept:

  n_components == min (n_samples, n_features)

fit(*X*) :Fit the model with X

transform(*X*) :apply the dimensionality reduction on X

Examples

```
>>> import numpy as np
>>> from sklearn.decomposition import PCA
>>> X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])
>>> pca = PCA(n_components=2) >>>
pca.fit(X) PCA(n_components=2)
```

sklearn.decomposition. FastICA

■ Fast algorithm for Independent Component Analysis (ICA)
(https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.FastICA.html#sklearn.decomposition.FastICA)

- 주요 파라미터

\- n_components: The number of components to use. If None is passed, all are used.

fit(*X*) :Fit the model to X

transform(*X*) :Recover the sources from X

Examples

```
>>> from sklearn.datasets import load_digits
>>> from sklearn.decomposition import FastICA
>>> X, _ = load_digits(return_X_y=True)
>>> transformer = FastICA(n_components=7, ... random_state=0, ... whiten='unit-variance')
>>> X_transformed = transformer.fit_transform(X)
```

파이썬으로 GMM 연습

- 아래의 코드를 한번씩 사용해보기

plot_iris_dataset.py

plot_pca_iris.py

plot_ica_blind_source_separation.py

plot_ica_vs_pca.py