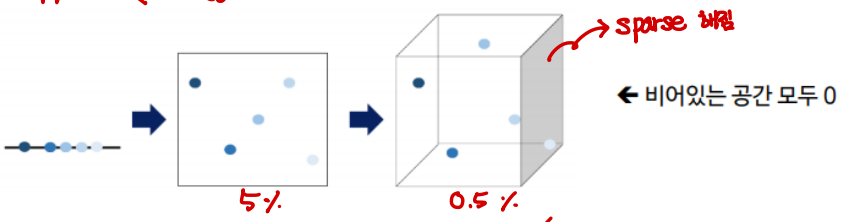
차원 축소

무인이동체공학과 17011882 김 우 혁

- 데이터 전처리 中 정규화, 차원축소, 영상처리
- 차원 축소는 데이터 분석 과정 중 전처리에서 주로 사용됨/ 목적: 고차원 → 저차원

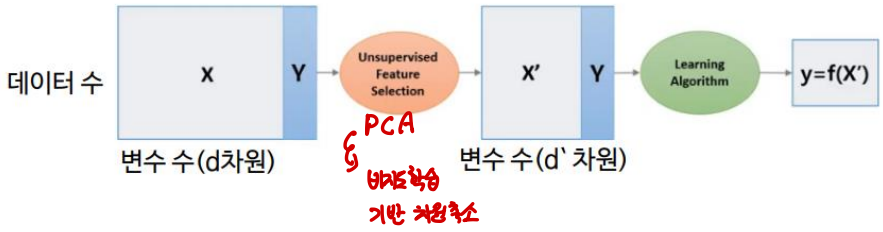
- 차원의 저주
- 차원이 증가할수록 동일 정보량을 표현하기 위해 필요한 데이터의 수는 지수적으로 증가
  - 차원이 증가 ~ 모델의 성능 저하 ➔ 개별 차원 내 학습할 데이터 수 Sparse 해짐
  - 데이터의 수 < 변수의 수 → 차원의 저주 발생



  - 일반적으로 intrinsic dimension(고유 차원)은 original dimension보다 상대적으로 작음 → 여기서, (고유 차원: 표현력을 상실하지 X 정도의 차원)
- 차원이 높을수록 발생하는 문제: Noise 증가, 성능 감소, 계산 복잡도 증가
  - 모든 변수는 서로 상관관계 0
- 해결책 → 중요한 특성만 사용!

- 차원 축소 목적: 모델의 성능을 최대로 해주는 변수의 일부 셋을 찾는 것

- 지도학습 기반 차원 축소: 학습결과가 피드백 되어 Feature Selection → 정답(라벨)과 예측값 비교 에러 보정 → 반복 학습
- 비지도학습 기반 차원 축소: 지도학습처럼 피드백을 통한 Feature Selection 반복 없음 → 정답(라벨)이 없으므로!



일반적으로 '선택' 보다 자주 사용 — 변환 함수 필요

⇨ 변수/피쳐 추출: 에측 변수의 변환을 통해 새로운 변수 추출 (변환 함수 필요)

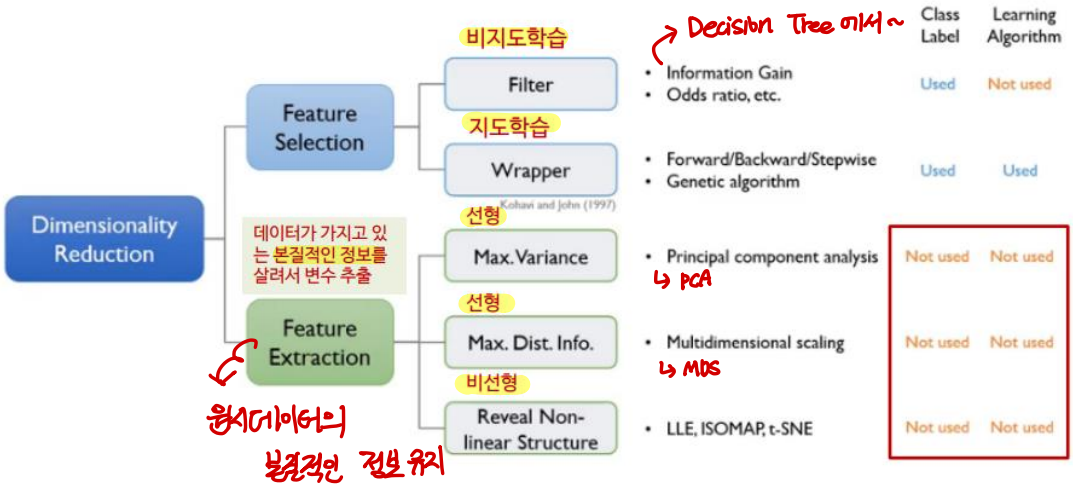
- 장점: 변수간 상관관계 고려, 변수의 개수를 "많이" 줄일 수 있음
- 단점: 추출된 변수의 해석이 어려움

기존 변수들 이용 변환

$$z = f(x_1, x_2, \ldots, x_{100})$$

새로운 변수 — 선형 결합

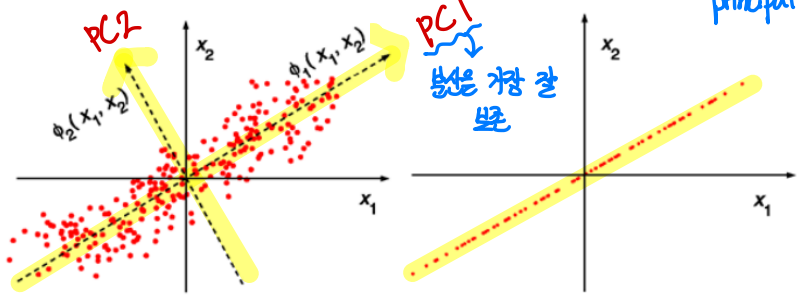
- 차원 축소 방법



- 차원 축소: PCA

  - **주성분 분석의 목적**
    - 차원을 줄이는 비지도 학습 방법 중 한가지
    - 사영 후 원 데이터의 분산(variance)을 최대한 보존할 수 있는 기저를 찾아 차원을 줄이는 방법

축, principal Axis
principal Components (PC)



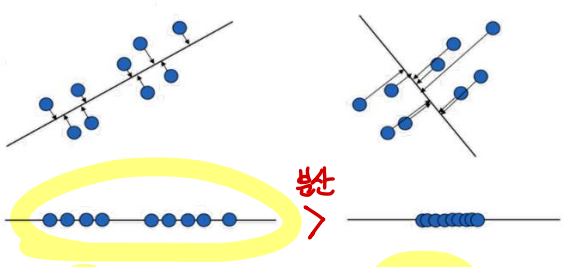
(a) PCA basis (b) PCA reduction to 1D

★ 우리의 기본적인 질문

Q. 데이터들을 정사영 시켜 차원을 낮춘다면, 어떤 basis에 데이터들을 정사영 시켜야 원래의 데이터 구조를 제일 잘 유지할까?

방법 ⇒ 선형 변환 → 하나의 벡터 공간을 선형적으로 다른 공간으로 mapping 하는 개념 가짐

- 데이터를 사영(projection)시킬 경우 손실되는 정보의 양이 적은 쪽의 기저(축)를 선택
- 아래 예시 → 왼쪽 기저(축)가 원 데이터의 분산을 최대로 유지하므로 왼쪽의 기저 축을 주성분으로 선택하는 것이 좋음



⇨ 대이터(X) 사영 변환 후 (Z)에도 분산이 보존하는 기저(a)를 찾는 것

$$Z_1 = \alpha_1^T X = \alpha_{11}X_1 + \alpha_{12}X_2 + \cdots + \alpha_{1p}X_p$$
$$Z_2 = \alpha_2^T X = \alpha_{21}X_1 + \alpha_{22}X_2 + \cdots + \alpha_{2p}X_p$$
$$\vdots$$
$$Z_p = \alpha_p^T X = \alpha_{p1}X_1 + \alpha_{p2}X_2 + \cdots + \alpha_{pp}X_p$$

$X_1, X_2, \ldots, X_p$: 원 데이터 P 개 변수

$\alpha_i = [\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{ip}]$: i 번째 기저(basis) 또는 계수(loading)

$Z_1, Z_2, \ldots, Z_p$: 각 기저로 사영 변환 후 변수 (주성분)

- 주성분 분석(PCA): 수리적 배경

- 공분산(Covariance): 변수의 상관 정도 / X: 입력 데이터 (n개의 데이터, d개의 변수)

$$Cov(X) = \frac{1}{n}(X - \bar{X})(X - \bar{X})^T$$

X의 평균

[dxd] [dxn] [nxd]

- 데이터 셋의 전체 분산(Total variance)

  → $\mathrm{tr}[\mathrm{Conv}(x)] = \mathrm{Conv}(X)_{11} + \mathrm{Conv}(X)_{22} + \cdots + \mathrm{Conv}(X)_{dd}$

예시)

$$A = \begin{bmatrix} 1 & 3 & 5 \\ 5 & 4 & 1 \\ 3 & 8 & 6 \end{bmatrix}, \quad \mathrm{Conv}(A) = \begin{bmatrix} 2.67 & 0.67 & -2.67 \\ 0.67 & 4.67 & 2.33 \\ -2.67 & 2.33 & 4.67 \end{bmatrix}$$

1과 2의 상관관계

대각은 다 더하면 데이터셋의 전체 분산

- **사영 (Projection)**

$$(\vec{b} - p\vec{a})^{\mathrm{T}}\vec{a} = 0 \;\Rightarrow\; \vec{b}^{\mathrm{T}}\vec{a} - p\vec{a}^{\mathrm{T}}\vec{a} = 0 \;\Rightarrow\; p = \frac{\vec{b}^{\mathrm{T}}\vec{a}}{\vec{a}^{\mathrm{T}}\vec{a}}$$

$$\vec{x} = p\vec{a} = \frac{\vec{b}^{\mathrm{T}}\vec{a}}{(\vec{a}^{\mathrm{T}}\vec{a})}\vec{a}$$

$(\vec{a}^{\mathrm{T}}\vec{a}) \to 1$

If $\vec{a}$ is unit vector

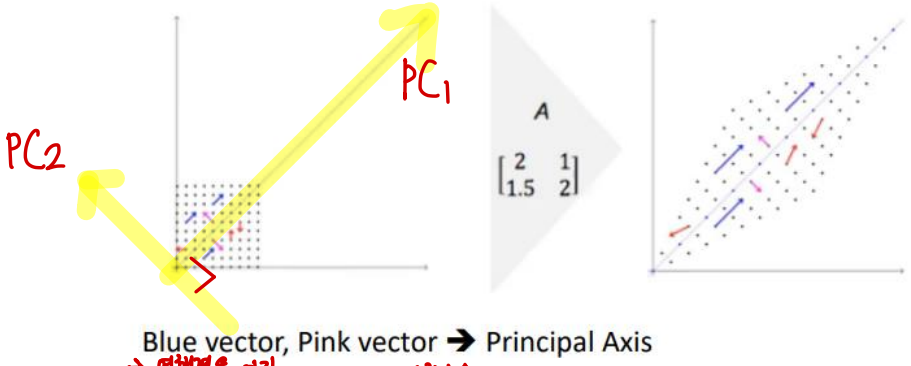
$$p = \vec{b}^{\mathrm{T}}\vec{a} \;\Rightarrow\; \vec{x} = p\vec{a} = (\vec{b}^{\mathrm{T}}\vec{a})\vec{a}$$

x: 사영후 벡터, p: 직교 사영을 위한 스케일러
b = 데이터, a = 기저축 (PC)

- **고유값(eigenvalue)과 고유벡터(eigenvector)**

$$\mathbf{Ax} = \lambda\mathbf{x} \quad \rightarrow \quad (\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = 0$$

- **벡터에 행렬을 곱하는 것은 선형 변환의 의미를 가짐**
  - 고유벡터는 변환에 의해 방향 변화가 발생하지 않음
  - 고유벡터의 크기 변화는 λ 만큼



Blue vector, Pink vector ➔ Principal Axis

⇨ 행렬 A가 Non-singular (↳ 역행렬은 가짐) 하다면, d개의 (차원의 수) 고유값과 고유벡터가 존재함 (↳ 비율이 없다)

⇨ 고유벡터는 서로 직교함(orthogonal)

⇨ $\mathrm{tr}(A) = \lambda_1 + \lambda_2 + \cdots + \lambda_d$

★ PCA의 방법성 → 사영된 데이터 분산의 최대화

Step2: 최적화 문제 정의

- 데이터 X를 기저 벡터 W에 사영(projection)하면, 사영 후 분산은 다음과 같음

$$V = \frac{1}{n}(\mathbf{w}^T\mathbf{X})(\mathbf{w}^T\mathbf{X})^T = \frac{1}{n}\mathbf{w}^T\mathbf{X}\mathbf{X}^T\mathbf{w} = \mathbf{w}^T\mathbf{S}\mathbf{w}$$

  - S는 X의 covariace matrix → 데이터의 구조를 선형변환해, 즉 특정 성분의 변동이 얼마나 largest한가를 나타냄.

★

- PCA의 목적은 사영 이후 분산 V를 최대화 하는 것

$$\max \mathbf{w}^T\mathbf{S}\mathbf{w}$$
$$\text{s.t. } \mathbf{w}^T\mathbf{w} = 1$$

➡ $S = \begin{pmatrix} 0.6166 & 0.6154 \\ 0.6154 & 0.7166 \end{pmatrix}$

Step3: 최적화 문제 솔루션

- 라그랑지 멀티플라이어(Lagrangian multiplier) 적용

$$\begin{bmatrix} \max & \mathbf{w}^T\mathbf{S}\mathbf{w} \\ s.t. & \mathbf{w}^T\mathbf{w} = 1 \end{bmatrix}$$ → 우리가 찾고자 하는 것

S는 X의 covariance matrix
W는 S의 eigen vector
람다는 S의 eigen value

$$L = \mathbf{w}^T\mathbf{S}\mathbf{w} - \lambda(\mathbf{w}^T\mathbf{w} - 1)$$

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{S}\mathbf{w} - \lambda\mathbf{w} = 0 \Rightarrow (\mathbf{S} - \lambda\mathbf{I})\mathbf{w} = 0$$

$$Eigenvectors = \begin{bmatrix} 0.6779 & -0.7352 \\ 0.7352 & 0.6779 \end{bmatrix} \qquad Eigenvalues = (1.2840 \quad 0.0491)$$

Step4: 주축 정렬

- Eigenvalue 에 해당되는 eigenvectors를 순서대로 정렬

$$Eigenvectors = \begin{bmatrix} 0.6779 & -0.7352 \\ 0.7352 & 0.6779 \end{bmatrix} \qquad Eigenvalues = (1.2840 \quad 0.0491)$$

- W_1은 Eigen vector이고 λ_1은 대응되는 eigen value 이다.
- (아래 식의 유도에 의하면) W_1에 사영된 데이터의 분산은 λ_1 이다.

$$\mathbf{v} = (\mathbf{w}_1^T\mathbf{X})(\mathbf{w}_1^T\mathbf{X})^T = \mathbf{w}_1^T\mathbf{X}\mathbf{X}^T\mathbf{w}_1 = \mathbf{w}_1^T\mathbf{S}\mathbf{w}_1$$

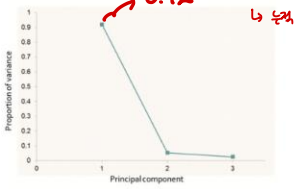
$$\text{Since } \mathbf{S}\mathbf{w}_1 = \lambda_1\mathbf{w}_1, \quad \mathbf{w}_1^T\mathbf{S}\mathbf{w}_1 = \mathbf{w}_1^T\lambda_1\mathbf{w}_1 = \lambda_1\mathbf{w}_1^T\mathbf{w}_1 = \lambda_1$$

- 첫번째 주성분으로 설명가능한 데이터 비율 $= \frac{\lambda_2}{\lambda_1 + \lambda_2} = \frac{0.0491}{1.2840 + 0.0491} = 0.96$ (1.2840)

★ 이렇게 우리는 d개의 주축들 중 일정 개수 이상의 성능을 내도록 선택해서 사용!

주성분 개수 선정 법 ➔ 몇개의 주성분을 사용해야 할까?

- 선택 방법 #1
  - 고유 값 감소율이 유의미하게 낮아지는 Elbow Point에 해당하는 주성분을 선택
- 선택 방법 #2
  - 일정 수준 이상의 분산 비를 보존하는 최소의 주성분을 선택 (보통 70% 이상)



0.92 ~> PC1만 선택해도 OK
↳ 문제

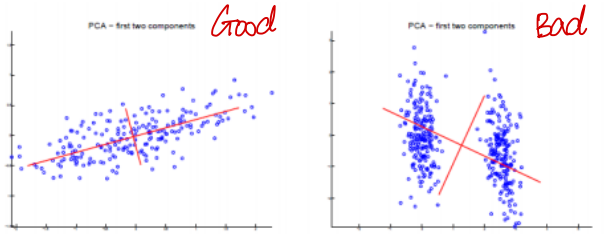
첫번째 주성분으로 설명가능한 데이터 비율 $= \frac{\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3}$

$= \frac{2.7596}{0.0786 + 0.1618 + 2.7596}$

$= 0.920$

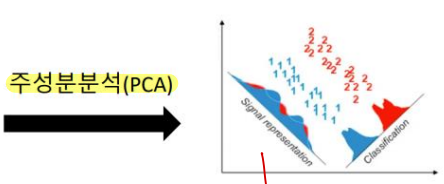
- **주성분 분석 한계**

- [1] 데이터 분포가 가우시안이 아니거나 다중 가우시안 자료들에 대해서는 적용하기 어려움



- [2] 분류 문제를 위해 디자인되지 않음, 즉 분류 성능 향상을 보장하지 못함

즉, 재가공된 데이터가 특유의 성질을 놓여줄 것이라고 보장 X



☆ 지도학습 ↑

~> 이 또한, 차원 축소

↳ 분산을 크게 하는 방향 ⇒ 구현 ↓

- **랜덤 PCA의 개념**
  - 자료의 크기 또는 특성변수의 크기가 매우 크면 주성분 W 를 구하기 위한 SVD 계산이 불가능하거나 시간이 많이 소요됨
  - 이런 경우 Randomized PCA 가 유용
  - Randomized PCA는 QR 분해를 이용하여 행렬의 SVD를 수행함

문제 ←

↳ 연산량 ↑

↳ 고유값과 고유벡터를 구하기 위한 알고리즘 ~

- **커널 PCA 개념**
  - PCA는 선형 변환이고 Kernelized PCA는 비선형 변환임
  - SVM의 커널트릭을 PCA에서도 사용
  - 특성 변수 x를 비선형 h(x)로 번환한 후 이에 대해 PCA를 하여 차원 축소를 하는 방법임