

선형 대수 (Linear Algebra)

자연어처리 텍스트마이닝

벡터

벡터와 행렬 (Vector & Matrix)

벡터

벡터는 크기와 방향을 가지고 있는 개념을 표현한 것이다.
자연어 처리에서 벡터를 많이 사용한다. 단어나 문장을 텍스트
그대로 사용할 수 없기 때문에 단어를 표현하거나 문장을
표현할 때 벡터를 사용한다.

$$x = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, y = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}$$

벡터의 덧셈

두 벡터간 덧셈은 각 벡터의 원소들간 값을 더하여 계산한다.

$$x = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, y = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

$$x + y = \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 1 + 3 \\ 2 + 4 \end{bmatrix} = \begin{bmatrix} 4 \\ 6 \end{bmatrix}$$

벡터의 뺄셈

마찬가지로 두 벡터간 뺄셈은 각 벡터의 원소들간 값을 빼서 계산한다.

$$x = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, y = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

$$x - y = \begin{bmatrix} 1 \\ 2 \end{bmatrix} - \begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 1 - 3 \\ 2 - 4 \end{bmatrix} = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$$

벡터의 스칼라 곱셈/나눗셈

벡터에 스칼라 곱셈과 나눗셈을 적용하면 스칼라 값을 각 원소에 곱하거나 나누어 계산한다.

$$x = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad 2x = 2 \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \cdot 1 \\ 2 \cdot 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$$

$$\frac{1}{2}x = \frac{1}{2} \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \cdot 1 \\ \frac{1}{2} \cdot 2 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$$

벡터의 norm

벡터의 노름(norm)은 벡터의 크기/길이를 의미한다.

$$\|x\| = \sqrt{\sum_{i=1}^n x_i^2} = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$$

$$x = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$\|x\| = \sqrt{1^2 + 2^2} = \sqrt{5} \approx (2.236)$$

벡터의 내적(dot product)

두 벡터의 내적은 두 벡터의 개별 성분의 곱의 합이다. 두 개의 벡터 x 와 y 가 있으면 내적은 다음과 같이 정의된다.

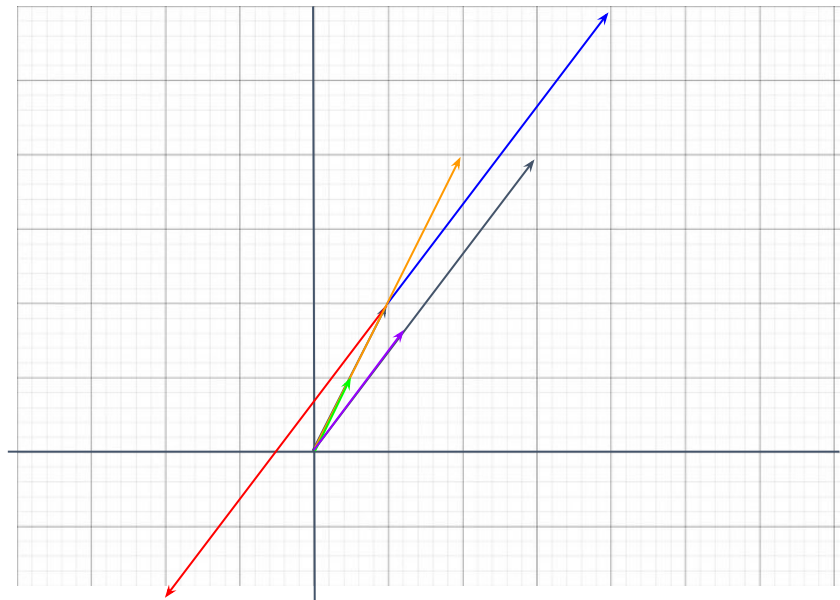
$$x \cdot y = x_1 y_1 + x_2 y_2 + \dots x_n y_n$$

$$x = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, y = \begin{bmatrix} -3 \\ 4 \end{bmatrix}$$

$$x \cdot y = x_1 y_1 + x_2 y_2 = (1 \cdot -3) + (2 \cdot 4) = 5$$

벡터의 기하학적 표현

$$x = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, y = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$



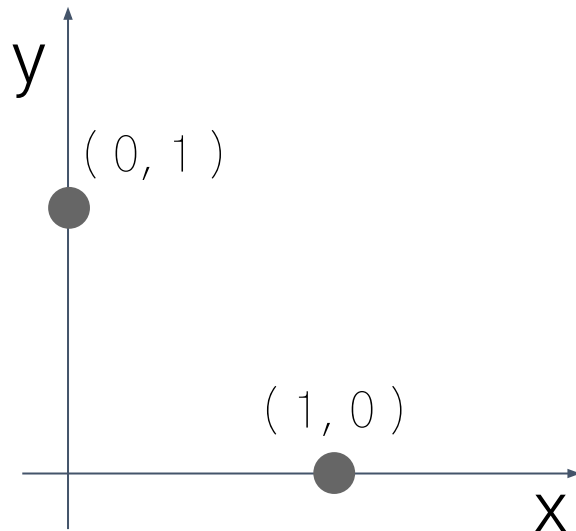
벡터의 직교

두 벡터 x 와 y 가 내적이 0이면 서로 직교한다.

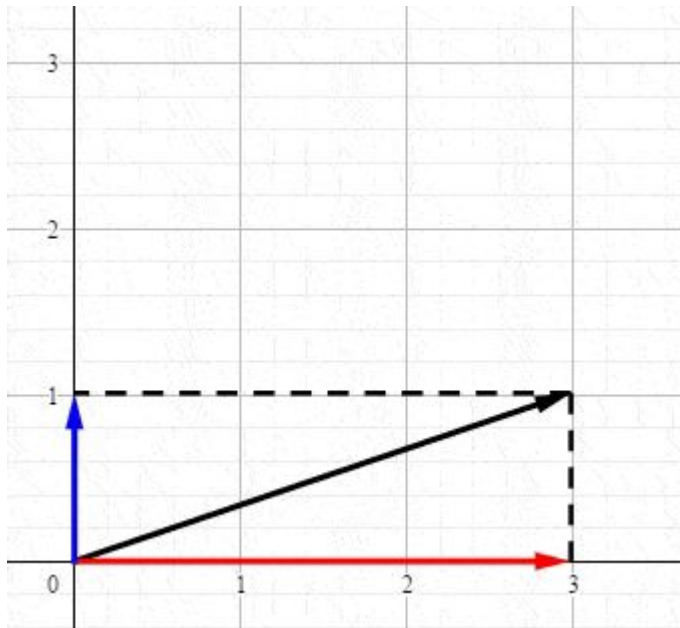
$$x \cdot y = 0$$

$$x = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, y = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$x \cdot y = x_1 y_1 + x_2 y_2 = (1 \cdot 0) + (0 \cdot 1) = 0$$



벡터의 분해



행렬

벡터와 행렬 (Vector & Matrix)

행렬

행렬은 2차원 숫자 배열이다. 행렬은 벡터의 묶음으로 생각할 수 있다.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

행렬의 곱셈

$$c_{ij} = \sum_k a_{ik} \cdot b_{kj} = a_{i1} \cdot b_{1j} + a_{i2} \cdot b_{2j} + \dots + a_{ik} \cdot b_{kj}$$

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}, B = \begin{bmatrix} 4 & 3 \\ 2 & 1 \end{bmatrix} \quad C = A \cdot B = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \cdot \begin{bmatrix} 4 & 3 \\ 2 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \cdot 4 + 2 \cdot 2 & 1 \cdot 3 + 2 \cdot 1 \\ 3 \cdot 4 + 4 \cdot 2 & 3 \cdot 3 + 4 \cdot 1 \\ 5 \cdot 4 + 6 \cdot 2 & 5 \cdot 3 + 6 \cdot 1 \end{bmatrix} = \begin{bmatrix} 8 & 5 \\ 20 & 13 \\ 32 & 21 \end{bmatrix}$$

행렬의 곱셈

$$c_{ij} = \sum_k a_{ik} \cdot b_{kj} = a_{i1} \cdot b_{1j} + a_{i2} \cdot b_{2j} + \dots + a_{ik} \cdot b_{kj}$$

$$A = [a_1, a_2, \dots, a_n], B = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{bmatrix}$$

$$A \cdot B = [a_1, a_2, \dots, a_n] \cdot \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{bmatrix}$$

$$= a_1 b_1 + a_2 b_2 + \dots + a_n b_n = \sum_{i=1}^n a_i b_i$$

대각 행렬 (diagonal matrix)

행과 열의 크기가 같은 정방행렬 비대각 요소의 값이 모두 0이고 대각 요소만 0이 아닌 값을 가진 행렬

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \end{bmatrix}$$

직교 행렬 (Orthogonal matrix)

행벡터와 열벡터가 직교를 이루는 행렬

$$X^T X = X X^T = I$$

$$X X^T = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = I$$

단위 행렬 (Identify matrix)

행과 열의 크기가 같은 정방행렬의 비대각 요소의 값이 모두 0이고 대각 요소만 1의 값을 가진 행렬

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

전치 행렬 (Transposed matrix)

전치 행렬은 행과 열이 바뀐 행렬

$$X = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, X^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

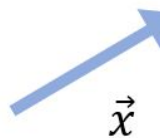
역행렬 (Inverse matrix)

역행렬은 임의행렬 X 에 X^{-1} 을 곱했을 때 단위 행렬 I 가 되는
행렬 X^{-1}

선형 변환(Linear Transformation)

벡터에 행렬을 곱

$$Av = b$$



\vec{x}

vector

1: Translation

$$\begin{bmatrix} Q_x \\ Q_y \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

2: Scaling

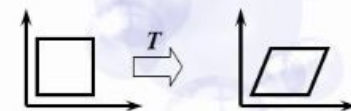
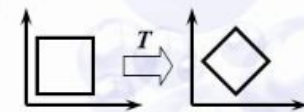
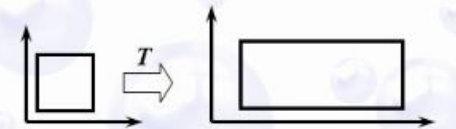
$$\begin{bmatrix} Q_x \\ Q_y \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} P_x \\ P_y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

3: Rotation

$$\begin{bmatrix} Q_x \\ Q_y \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} P_x \\ P_y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

4: Shearing

$$\begin{bmatrix} Q_x \\ Q_y \end{bmatrix} = \begin{bmatrix} 1 & h \\ g & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

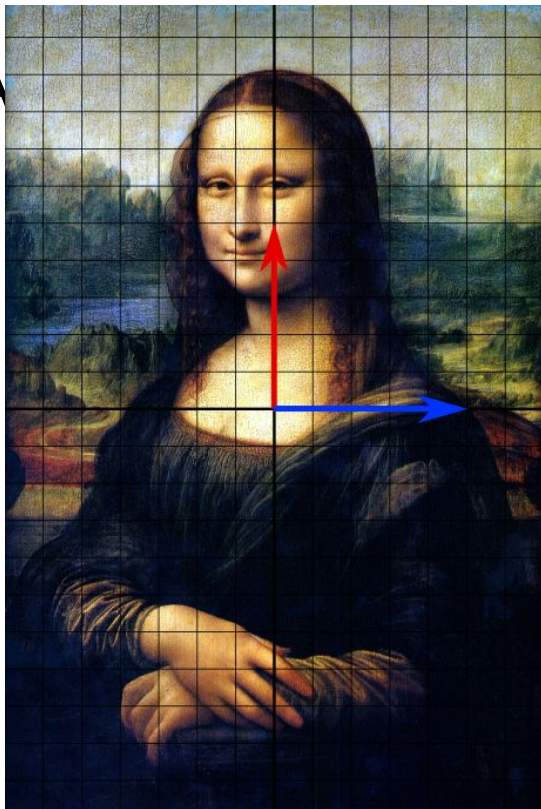


$A\vec{x}$

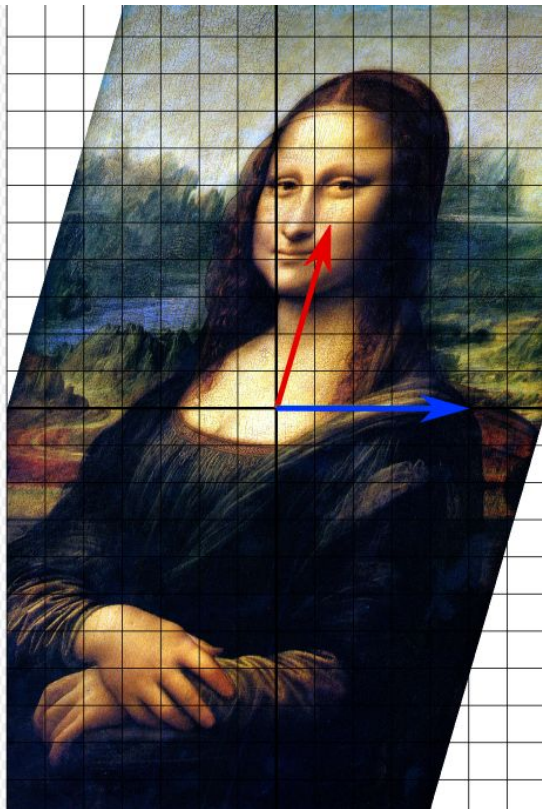
transformed vector

고유 벡터(Eigen Vector)

행렬 A



변합니다



확률

확률 (Dimension Reduction)

기대값 (Expected Value)

어떤 확률을 가진 사건을 무한히 반복했을 경우 얻을 수 있는 값의 평균으로서 기대할 수 있는 값

ex) 주사위를 무한히 굴렸을때 주사위 숫자는 3.5

$$E(x) = 1*1/6 + 2*1/6 + 3*1/6 + 4*1/6 + 5*1/6 + 6*1/6 = 3.5$$

분산 (Variance)

- 분포가 평균으로부터 얼마나 산포되어 있는 정도
- 편차의 제곱의 기댓값

확률변수 X 에 대해 분산 $\text{Var}(X)$ 는

$$\text{Var}(X) = E \left[(X - E(X))^2 \right]$$

이 때, $E(X) = \mu$ 라고 하면,

$$\text{Var}(X) = E \left[(X - \mu)^2 \right]$$

표준편차(Standard Deviation)

- 분산에 제곱근을 붙인 값

확률변수 X 의 표준편차 σ 는

$$\sigma = \sqrt{\text{Var}(X)}$$

$$\sigma^2 = \text{Var}(X)$$

공분산(Covariance)

- 두 개의 확률변수의 관계를 보여주는 값 - 확률변수 X 와 Y 에 대해 X 가 변할 때 Y 가 변하는 정도를 나타내는 값

$$\text{Cov}(X, Y) = E[(X - \mu_x)(Y - \mu_y)]$$

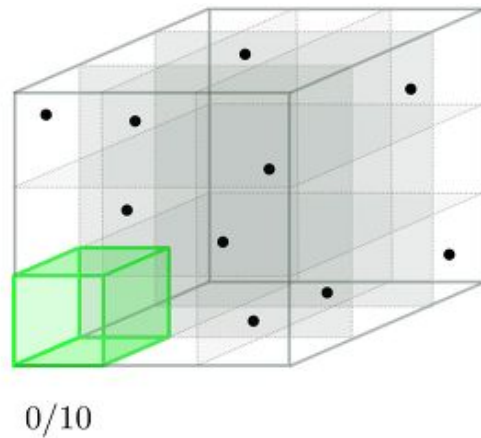
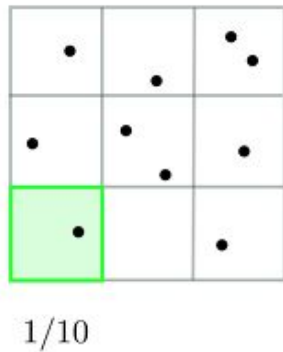
차원 축소 개요

차원 축소 (Dimension Reduction)

차원의 저주

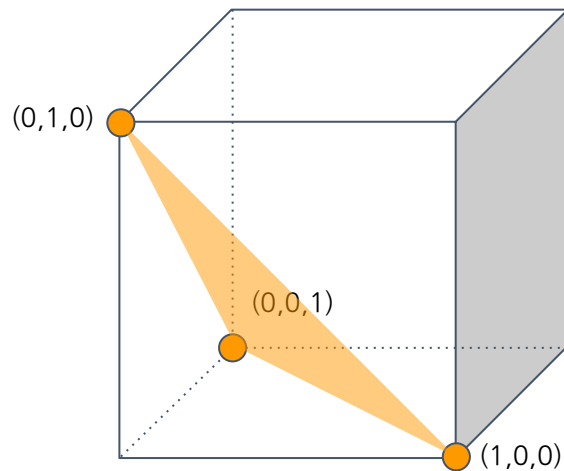
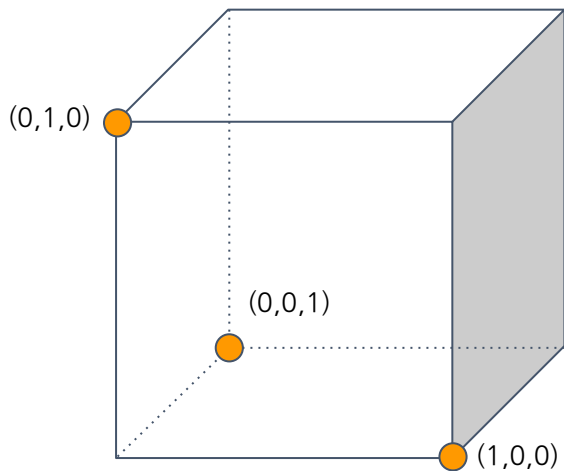
- 차원 증가할 수록 데이터의 표현이 어려움
- 차원 증가할 수록 연산(분석) 어려움

=> 차원 축소가 필요함



차원 축소 개요

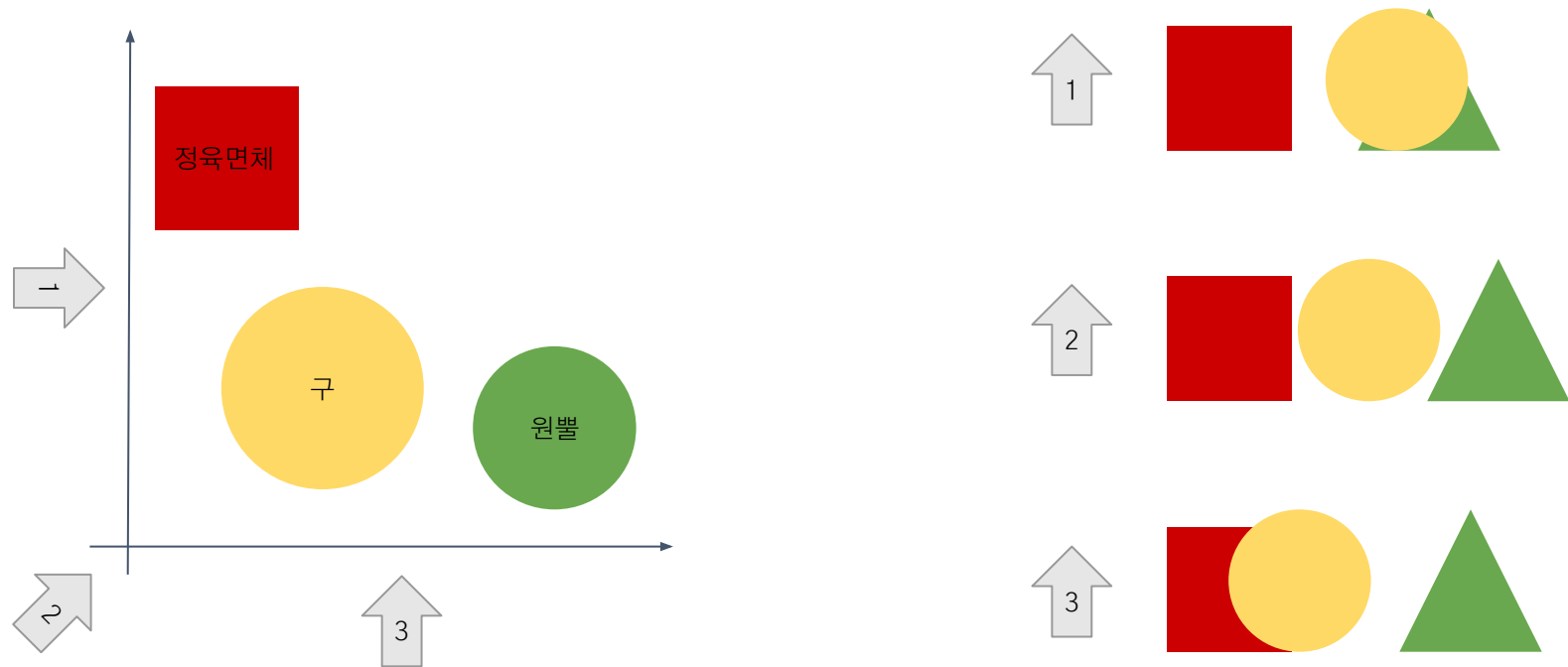
- 입력 데이터의 차원이 클 경우, 차원의 저주와 학습 속도가 저하됨
- 같은 정보를 표현하는데 낮은 차원을 사용하여 정보를 표현하는 것을 차원 축소라고 함



차원 축소의 예

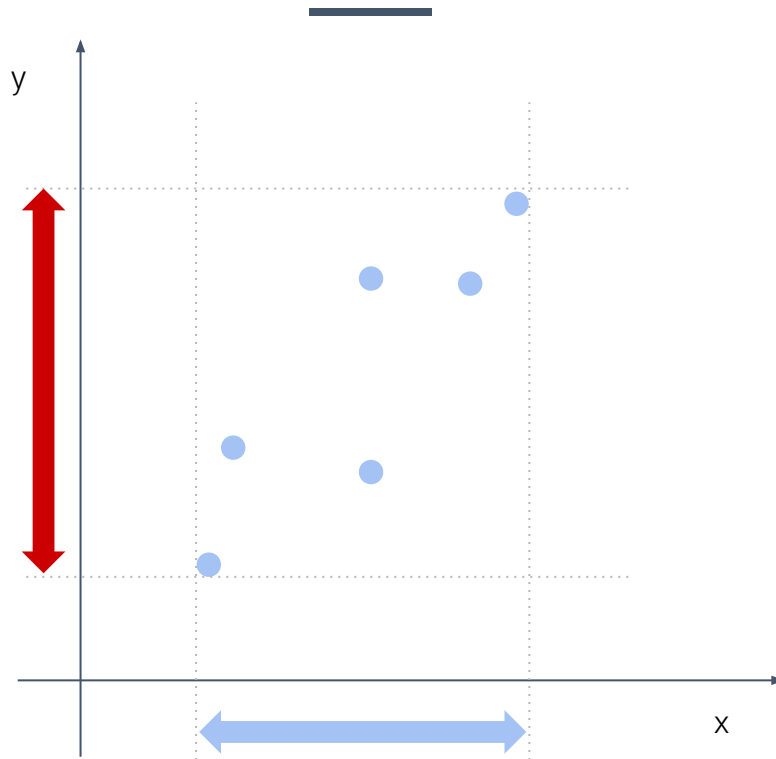
- 예시 1)
 - 국어 : 80점, 영어 : 60점, 수학 : 90점, 과학 : 80점
 - 평균 = $(80 + 60 + 90 + 80)/4 = 77.5$ 점
 - 4개의 차원을 “평균”이라는 1개의 차원으로 축소
- 예시 2)
 - 체질량지수(BMI) = $\text{몸무게} / \text{키}^2$
 - 몸무게, 키를 1개 차원으로 축소
- 차원축소시 정보의 손실이 발생한다. 차원 축소에서 가장 중요한 것은 정보 손실을 최소화하는 것

차원 축소의 기하학적 의미 (1)

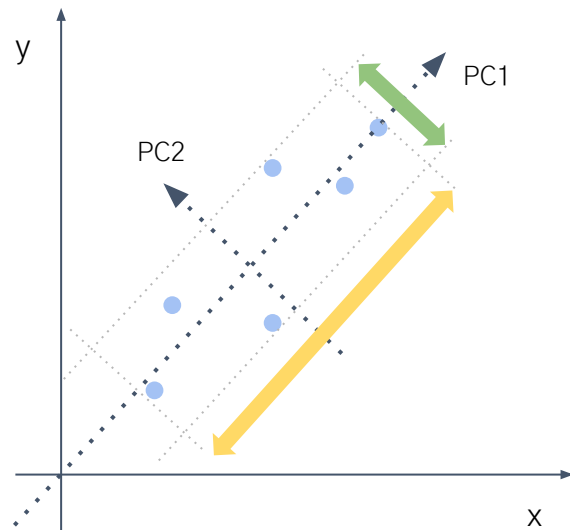
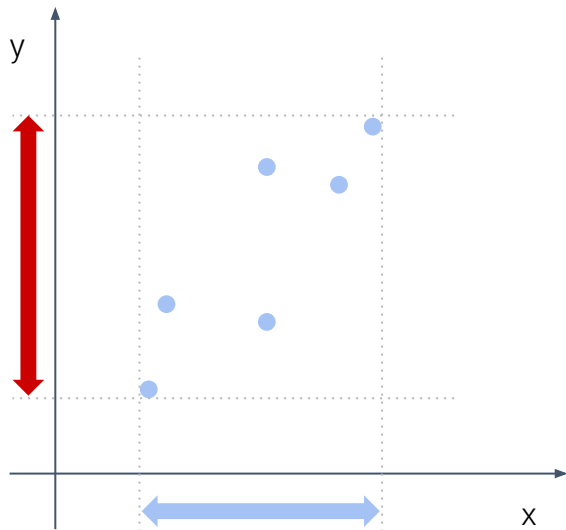


바라보는 시선에 따라 분류의 정도가 달라질 수 있다.

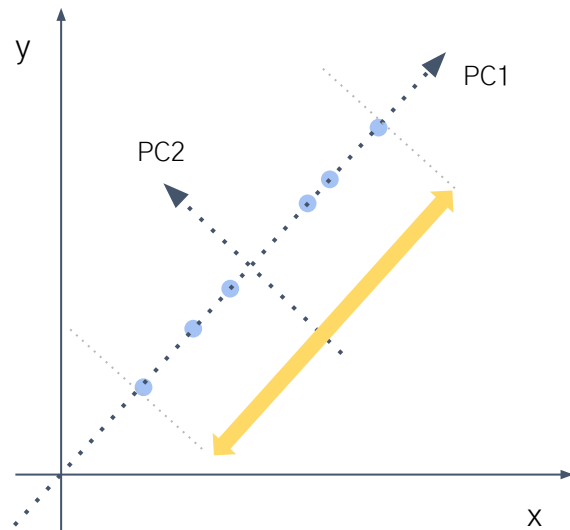
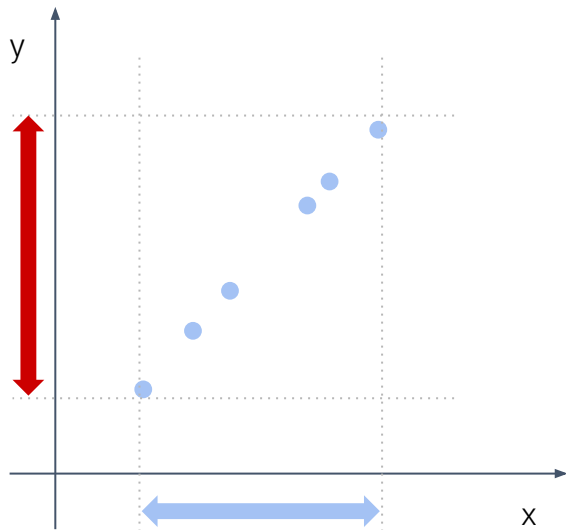
차원 축소의 기하학적 의미 (2)



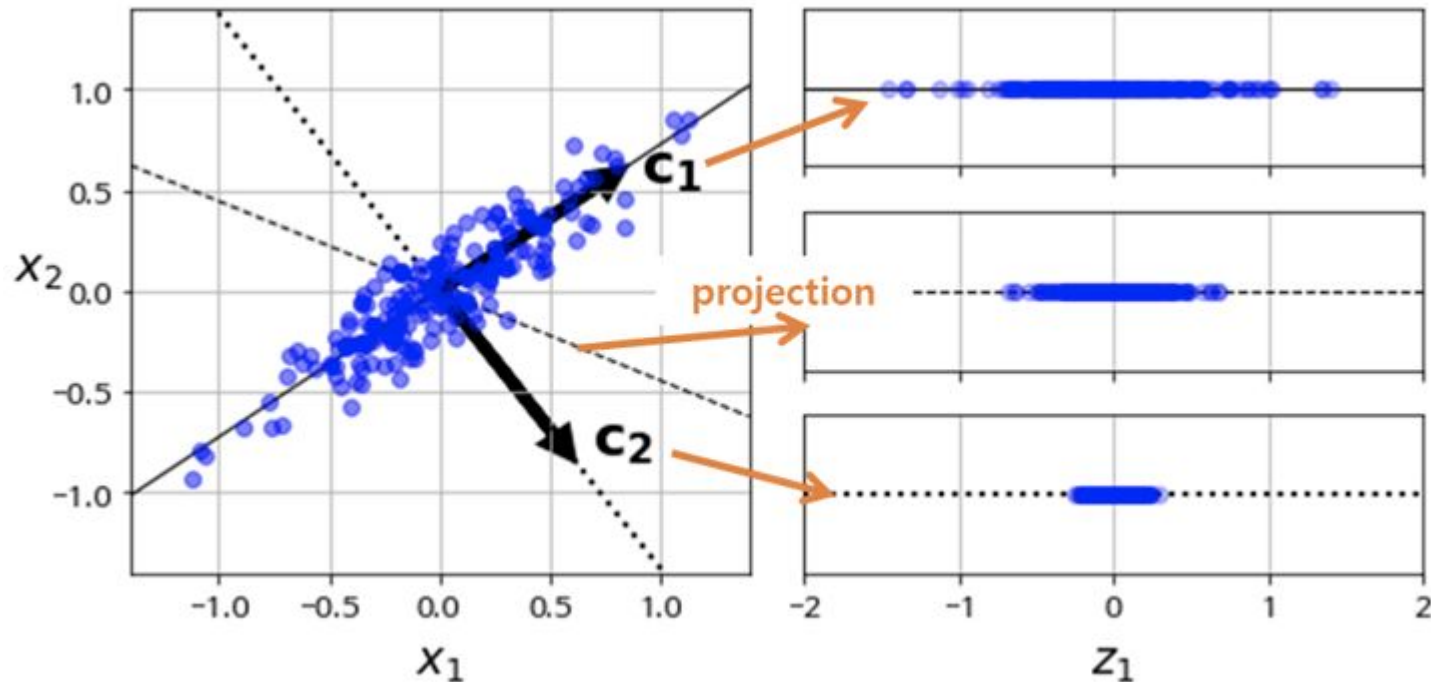
차원 축소의 기하학적 의미 (3)



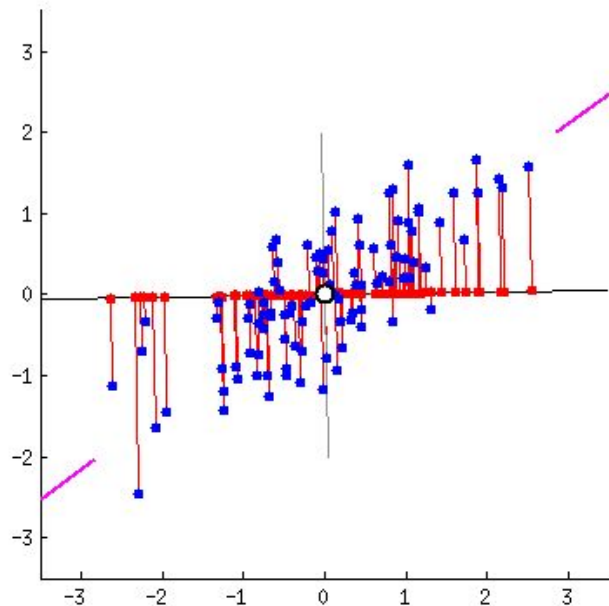
차원 축소의 기하학적 의미 (4)



차원 축소의 기하학적 의미 (5)



차원 축소의 기하학적 의미 (6)



<https://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues>

주성분 분석

(PCA, Principal Component Analysis)

차원 축소 (Dimension Reduction)

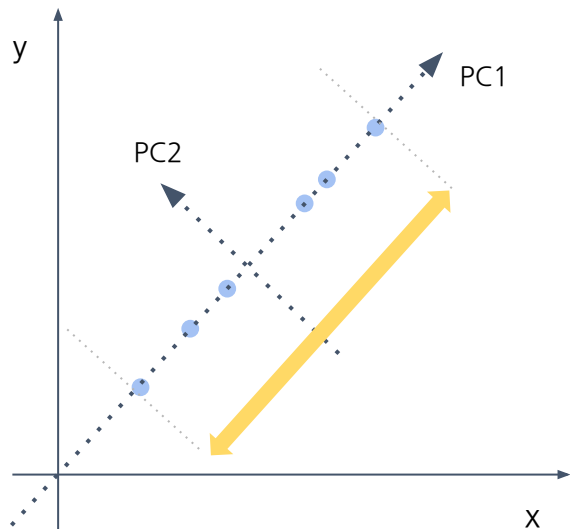
주성분 분석(PCA)

주성분 분석(主成分分析, Principal component analysis; PCA)은 고차원의 데이터를 저차원의 데이터로 환원시키는 기법이다. 서로 연관 가능성이 있는 고차원 공간의 표본들을 선형 연관성이 없는 저차원 공간(주성분)의 표본으로 변환하기 위해 직교 변환을 사용한다. 주성분의 차원수는 원래 표본의 차원수보다 작거나 같다. 주성분 분석은 데이터를 한개의 축으로 사상시켰을 때 그 분산이 가장 커지는 축을 첫 번째 주성분, 두 번째로 커지는 축을 두 번째 주성분으로 놓이도록 새로운 좌표계로 데이터를 선형 변환한다. 이와 같이 표본의 차이를 가장 잘 나타내는 성분들로 분해함으로써 여러가지 응용이 가능하다. 이 변환은 첫째 주성분이 가장 큰 분산을 가지고, 이후의 주성분들은 이전의 주성분들과 직교한다는 제약 아래에 가장 큰 분산을 갖고 있다는 식으로 정의되어있다. 중요한 성분들은 공분산 행렬의 고유 벡터이기 때문에 직교하게 된다.

https://ko.wikipedia.org/wiki/%EC%A3%BC%EC%84%B1%EB%B6%84_%EB%B6%84%EC%84%9D

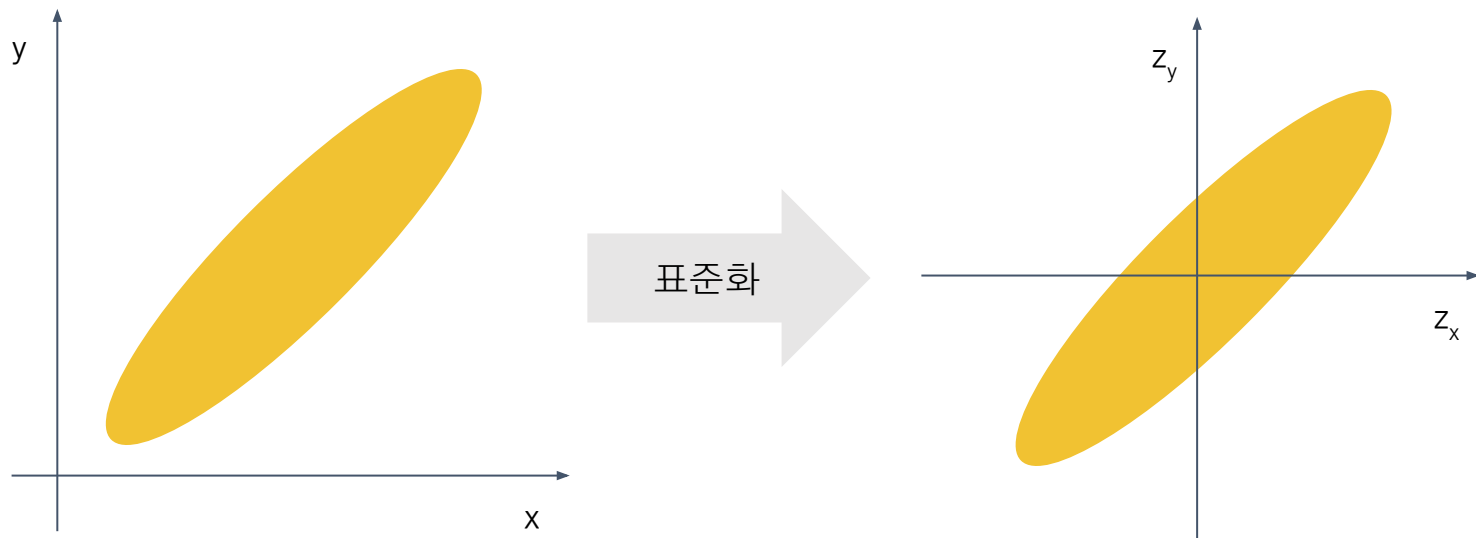
다차원 데이터의 정보를 가능한 한 손실 없이 저차원 공간에 압축하는 것

주성분 분석 개요



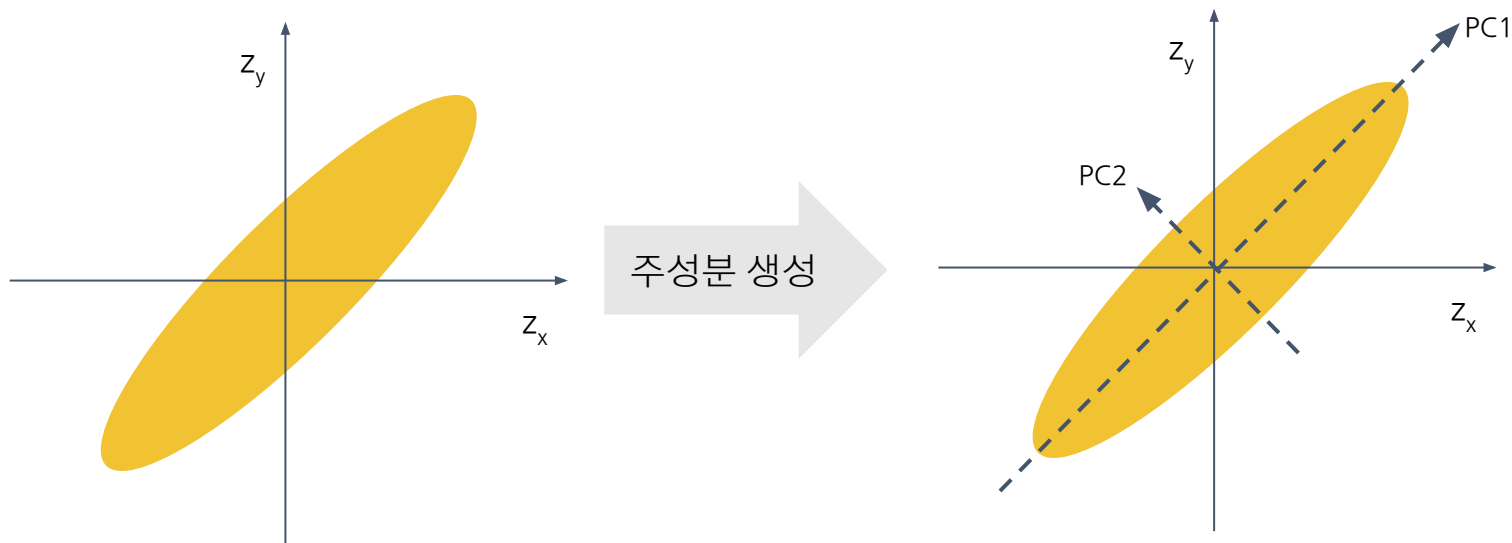
- 첫번째 주성분 축생성
 - 분산이 가장 큰(데이터가 가장 넓게 퍼져있는) 방향을 구한다
 - 그 방향으로 첫번째 축을 만든다
- 두번째 주성분 축생성
 - 첫번째 축과 90도 직교하며, 분산이 두번째로 큰 방향을 구한다.
 - 그 방향으로 두번째 축을 만든다
- 세번째 주성분 축생성
 - ...

주성분 분석 절차(1)



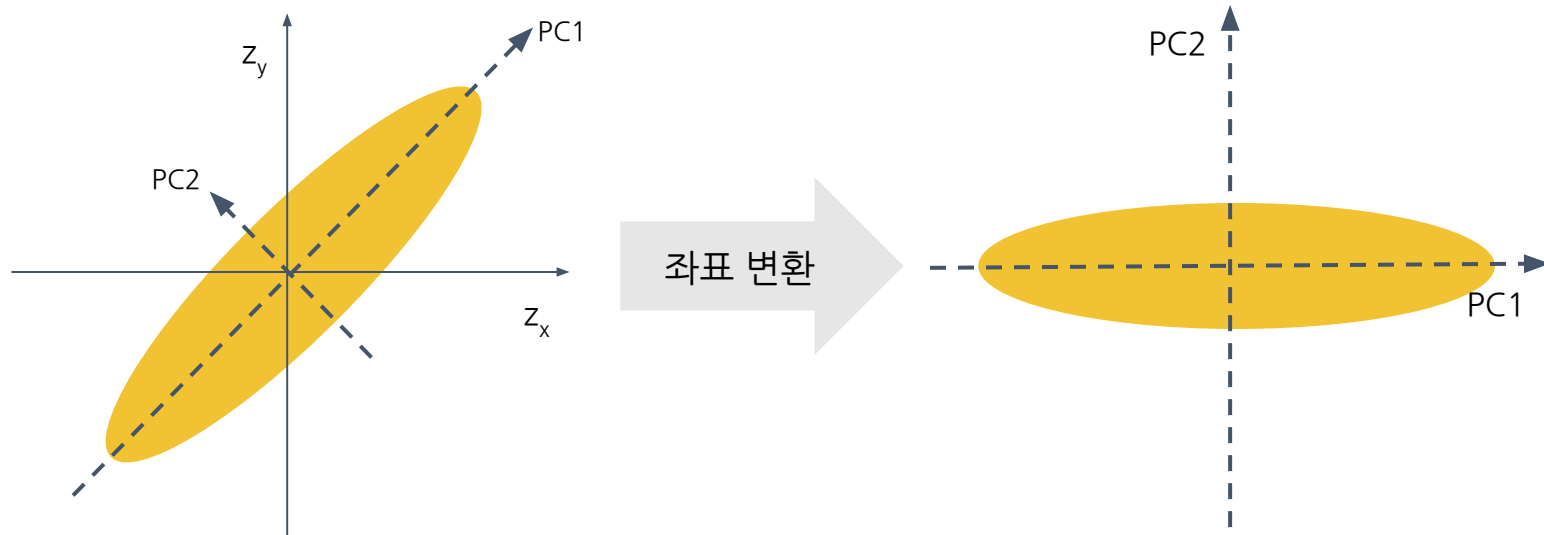
데이터를 표준화(평균 0, 표준편차 1)

주성분 분석 절차(2)



분산이 가장 큰(데이터가 가장 많이 퍼져있는) 방향을 찾아서 PC1을 생성 (반복)

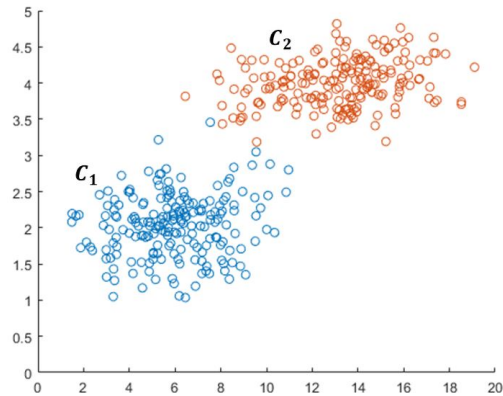
주성분 분석 절차(3)



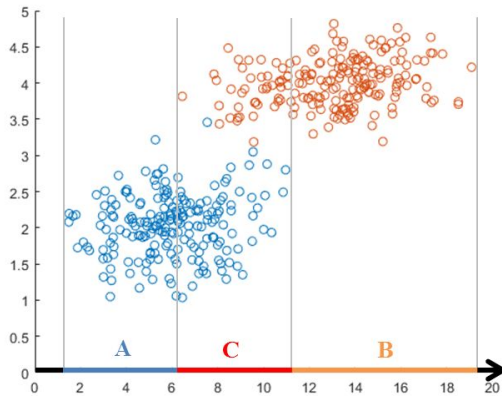
구해진 주성분 축으로 좌표를 변환

주성분 분석 예시 (1)

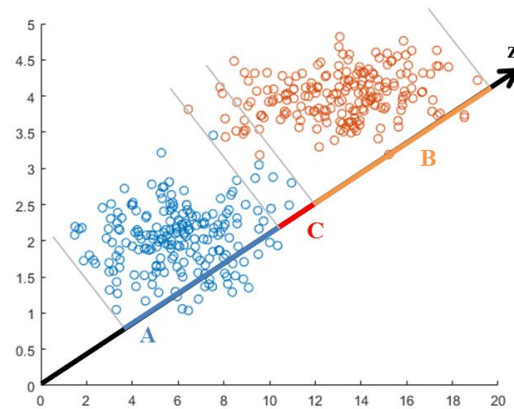
- 2차원의 분류(Classification) 결과를 1차원으로 축소



분류(Classification) 결과



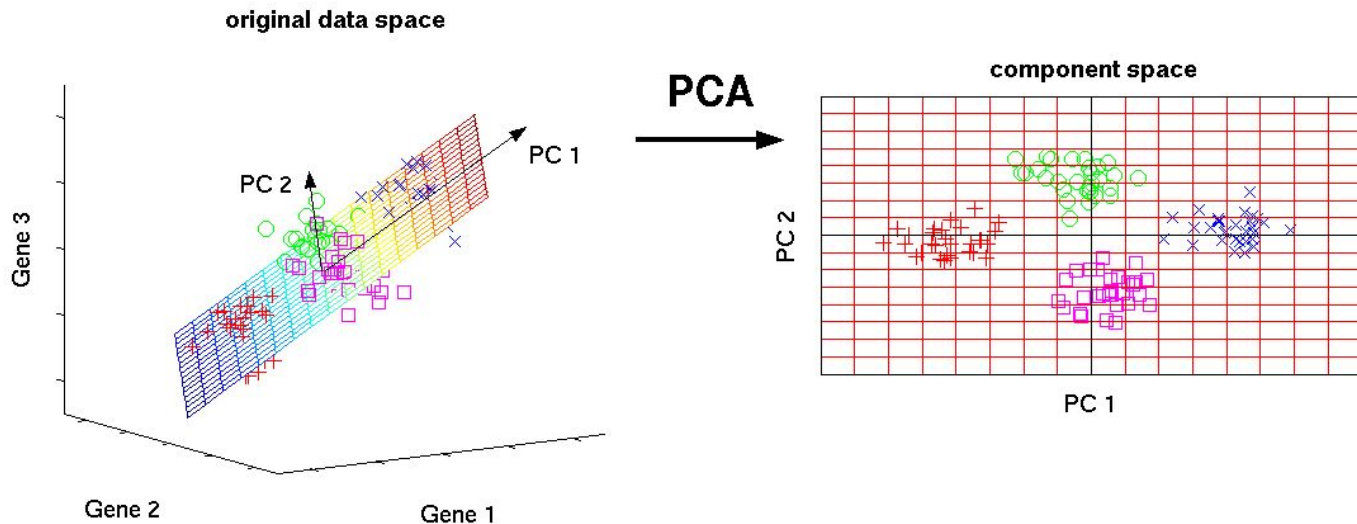
x축에 사상



z축에 사상

주성분 분석 예시 (2)

- 3차원의 분류(Classification) 결과를 2차원으로 축소



공분산 행렬

(Covariance Matrix)

차원 축소 (Dimension Reduction)

공분산 행렬의 의미

- 공분산 행렬의 의미
 - 데이터의 의미 : 각 feature의 움직임이 얼마나 유사한가
 - 수학적 의미 : 선형변환
- PCA
 - 공분산 행렬의 eigenvector, eigenvalue
 - PCA = Projecting data onto eigenvector of 공분산 행렬

공분산 행렬의 의미 - 데이터 구조 (1)

- 어떤 행렬 X
 - 행은 sample, 열은 feature를 의미
 - 열(feature)의 평균 값을 0으로 조정 (=각 feature의 값에 평균을 뺀 상태)

$$X = \begin{pmatrix} | & | & | & \dots & | \\ X_1 & X_2 & X_3 & \dots & X_d \\ | & | & | & & | \end{pmatrix} \in \mathbb{R}^{n \times d}$$

공분산 행렬의 의미 - 데이터 구조 (1)

$$\mathbf{C}(\mathbf{X}) = \begin{bmatrix} \text{Var}(X_1) & \text{Cov}(X_1, X_2) & \cdots & \text{Cov}(X_1, X_K) \\ \text{Cov}(X_2, X_1) & \text{Var}(X_2) & \cdots & \text{Cov}(X_2, X_K) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(X_K, X_1) & \text{Cov}(X_K, X_2) & \cdots & \text{Var}(X_K) \end{bmatrix}$$

$$\text{Cov}(x, y) = \frac{\sum (x - \bar{x})(y - \bar{y})}{n} \quad \text{or}$$

$$\text{Cov}_{xy} = \frac{\sum (x - \bar{x})(y - \bar{y})}{n-1}$$

공분산 행렬의 의미 - 데이터 구조 (2)

- 데이터의 의미 : 각 feature의 움직임이 얼마나 유사한가
 - 행렬 X의 공분산 행렬 계산

$$Cov(x,y) = \frac{\sum (x - \bar{x})(y - \bar{y})}{n} \quad or$$

$$Cov_{xy} = \frac{\sum (x - \bar{x})(y - \bar{y})}{n-1}$$

$$X^T X = \begin{pmatrix} \text{---} & X_1 & \text{---} \\ \text{---} & X_2 & \text{---} \\ & \dots & \\ \text{---} & X_d & \text{---} \end{pmatrix} \begin{pmatrix} | & | & & | \\ X_1 & X_2 & \dots & X_d \\ | & | & & | \end{pmatrix}$$

공분산 행렬의 의미 - 데이터 구조 (3)

- 데이터의 의미 : 각 feature의 움직임이 얼마나 유사한가
 - 행렬 X의 공분산 행렬 계산

$$\begin{aligned}
 X^T X &= \begin{pmatrix} \text{---} & X_1 & \text{---} \\ \text{---} & X_2 & \text{---} \\ & \dots & \\ \text{---} & X_d & \text{---} \end{pmatrix} \begin{pmatrix} | & | & & | \\ X_1 & X_2 & \dots & X_d \\ | & | & & | \end{pmatrix} \\
 &= \begin{pmatrix} \text{dot}(X_1, X_1) & \text{dot}(X_1, X_2) & \dots & \text{dot}(X_1, X_d) \\ \text{dot}(X_2, X_1) & \text{dot}(X_2, X_2) & \dots & \text{dot}(X_2, X_d) \\ \vdots & \vdots & \ddots & \vdots \\ \text{dot}(X_d, X_1) & \text{dot}(X_d, X_2) & \dots & \text{dot}(X_d, X_d) \end{pmatrix}
 \end{aligned}$$

공분산 행렬의 의미 - 데이터 구조 (4)

- 데이터의 구조적 의미 : 각 feature의 변동이 얼마나 닮았나 행렬 X의 공분산 행렬 계산
 - 공분산 행렬의 $\text{dot}(X_1, X_1)$ 은 X_1 의 분산을 의미
 - 공분산 행렬의 $\text{dot}(X_1, X_2)$ 은 X_1 과 X_2 의 공분산을 의미

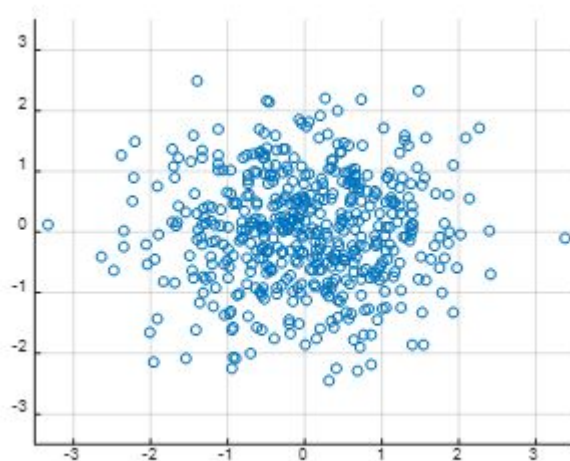
$$\frac{X^T X}{n} = \frac{1}{n} \begin{bmatrix} \text{dot}(X_1, X_1) & \text{dot}(X_1, X_2) & \cdots & \text{dot}(X_1, X_d) \\ \text{dot}(X_2, X_1) & \text{dot}(X_2, X_2) & \cdots & \text{dot}(X_2, X_d) \\ \vdots & \vdots & \ddots & \vdots \\ \text{dot}(X_d, X_1) & \text{dot}(X_d, X_2) & \cdots & \text{dot}(X_d, X_d) \end{bmatrix}$$

분산

공분산

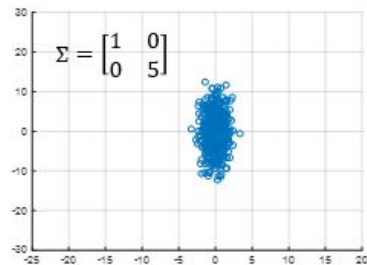
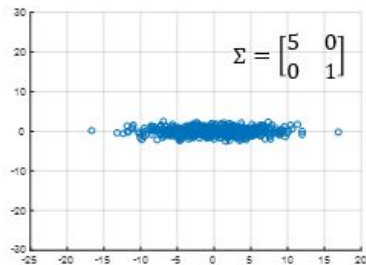
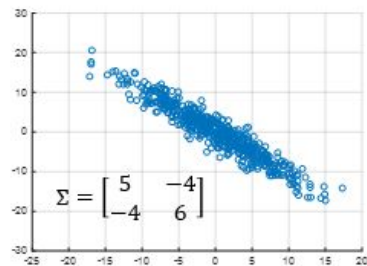
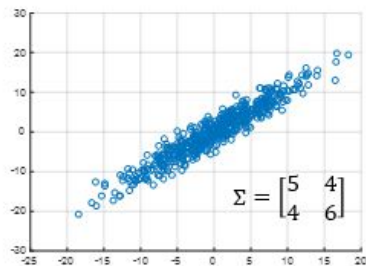
공분산 행렬의 의미 - 수학적 의미 (1)

- 수학적 의미 : 선형 변환 (shearing)




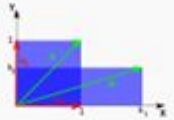
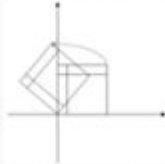


공분산 행렬의 의미 - 수학적 의미 (2)

- 수학적 의미 : 선형 변환 (shearing)



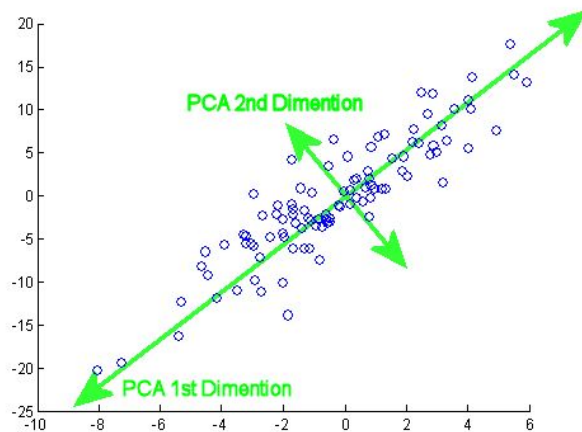
공분산 행렬의 의미 - 수학적 의미 (2)

- 수학적 의미 : 선형 변환 (shearing)

	scaling	unequal scaling	rotation	horizontal shear	hyperbolic rotation
illustration					
matrix	$\begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix}$	$\begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}$	$\begin{bmatrix} c & -s \\ s & c \end{bmatrix}$ $c = \cos \theta$ $s = \sin \theta$	$\begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} c & s \\ s & c \end{bmatrix}$ $c = \cosh \varphi$ $s = \sinh \varphi$

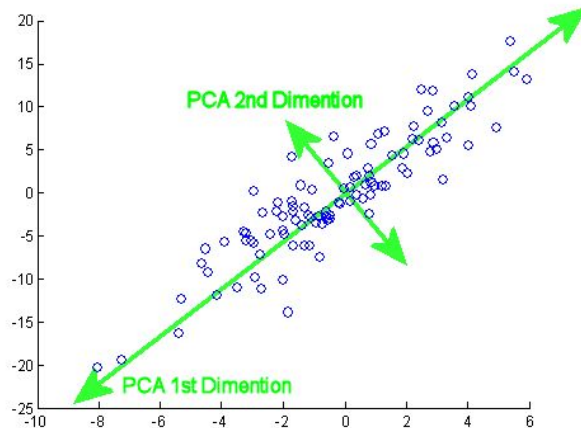
PCA (Principal Component Analysis)

- PCA 알고리즘은 데이터 구조를 잘 살리면서 차원을 감소할 수있게 하는 방법
 - 정사영 이후 데이터의 분포(=분산)이 제일 큰 것이 좋음
 - 공분산 행렬로 선형 변환할 때, 주축에 대해 정사영하는 것이 좋음



PCA (Principal Component Analysis)

- PCA 알고리즘은 데이터 구조를 잘 살리면서 차원을 감소할 수있게 하는 방법
 - 선형변환의 주축을 eigenvector라 부름
 - eigenvector를 찾는다는 것은 “선형변환 이후 크기만 바뀌고 방향은 바뀌지 않는 벡터를 찾는것”
 - 이는 정사영 후 데이터 분포(분산)가 가장 큰 결과를 얻기위해 eigenvector에 정사영



PCA (Principal Component Analysis)

- PCA 알고리즘은 데이터 구조를 잘 살리면서 차원을 감소할 수있게 하는 방법
 - 3차원 데이터는?
 - 공분산 행렬을 고유값 분해하면 3개의 eigenvector가 나옴
 - 이 중 고유값이 큰순으로 2개의 eigenvector에 정사영 하면 2차원으로 축소됨
 - N차원 데이터는?
 - 공분산 행렬을 고유값 분해하면 N개의 eigenvector가 나옴
 - 이 중 고유값이 큰순으로 k개의 eigenvector에 정사영 하면 k차원으로 축소됨

고유값 분해

(Eigenvalue Decomposition)

차원 축소

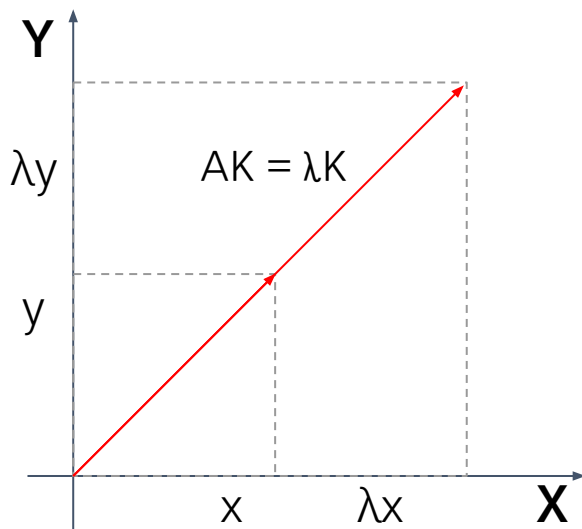
고유값, 고유벡터 의미

- 정방행렬 $A : N \times N$
- $AK = \lambda K$
 - K : 고유벡터(eigenvector)
 - λ : 고유값(eigenvalue)
- 임의의 정방행렬 K 에 대해 고유값, 고유벡터를 찾아내는 것을 **고유값 분해**라고 함

$$AE = E\Lambda$$
$$A = E\Lambda E^{-1}$$

고유값, 고유벡터 의미

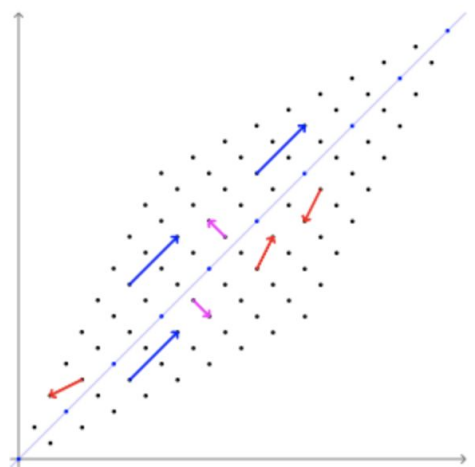
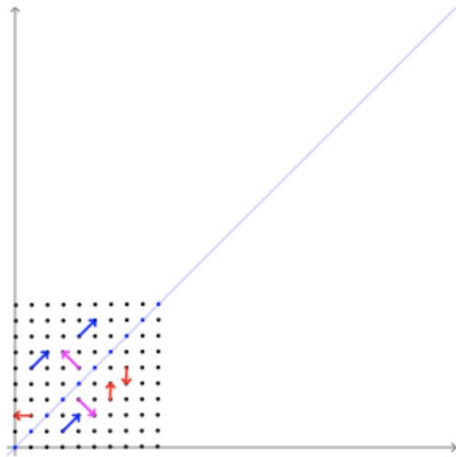
- 고유값, 고유벡터의 의미
 - A를 선형변환 행렬로 보았을 때 그 안에서 크기(scale)은 변하지만 방향을 유지되는 벡터(eigenvector)가 존재하는가



https://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors

고유값, 고유벡터 의미

- 고유값, 고유벡터의 의미
 - A를 선형변환 행렬로 보았을 때 그 안에서 크기(scale)은 변하지만 방향을 유지되는 벡터(eigenvector)가 존재하는가
=> 주성분 축을 찾는다

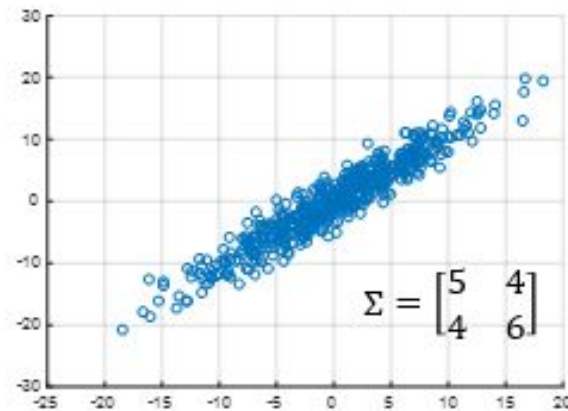
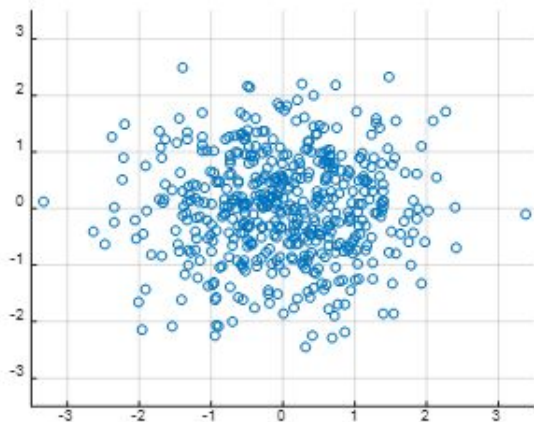


https://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors

$$AK = \lambda K \cdots (1)$$

고유값, 고유벡터 의미

- 고유값, 고유벡터의 의미
 - A를 선형변환 행렬로 보았을 때 그 안에서 크기(scale)은 변하지만 방향을 유지되는 벡터(eigenvector)가 존재하는가
=> 주성분 축을 찾는다



https://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors

고유벡터 활용

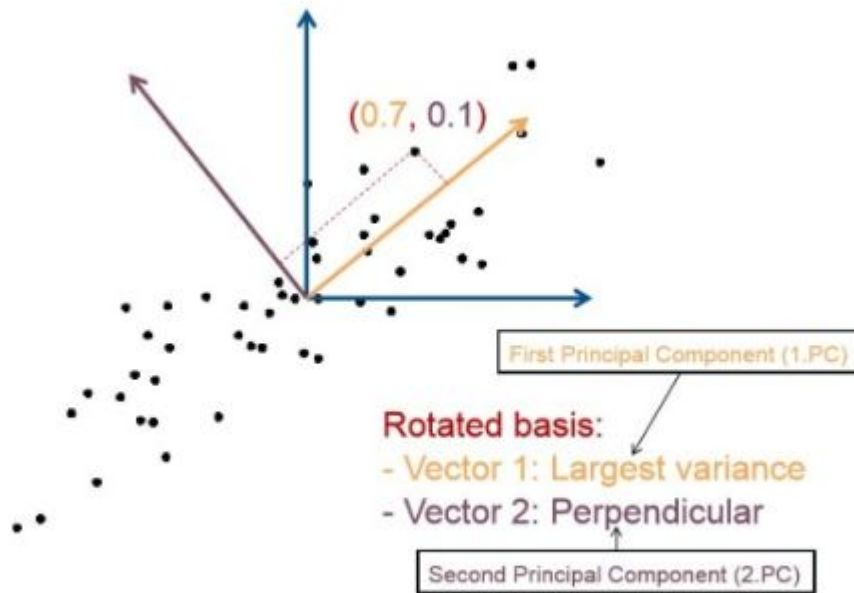
- 행렬 분해
 - 고유값 분해 (Eigendecomposition)
 - 특이값 분해 (SVD, Singular Value Decomposition)
- PCA (Principal Component Analysis)
 - 차원축소
- 그래프 분석
 - 그래프 영향도 측정 : Google PageRank
- 신호처리
 - 영상처리 : Eigenface



고유벡터 - 복소수

- 행렬의 고유값 분해시 복소수 (complex number)가 고유값 (eigenvalue), 고유벡터 (eigenvector)에 등장 할 수 있다.
- 대칭 행렬 vs 비대칭 행렬
 - 분해하고자 하는 행렬이 대칭행렬 (symmetric matrix)의 경우 복소수가 계산되지 않음
 - 분해하고자 하는 행렬이 비대칭행렬 (non-symmetric matrix)의 경우 복소수가 계산될 수 있음
- PCA
 - PCA의 경우 공분산 행렬 (대칭 행렬)을 사용하여 고유값 분해를 수행하기 때문에 복소수가 계산되지 않음

정리



특이값 분해

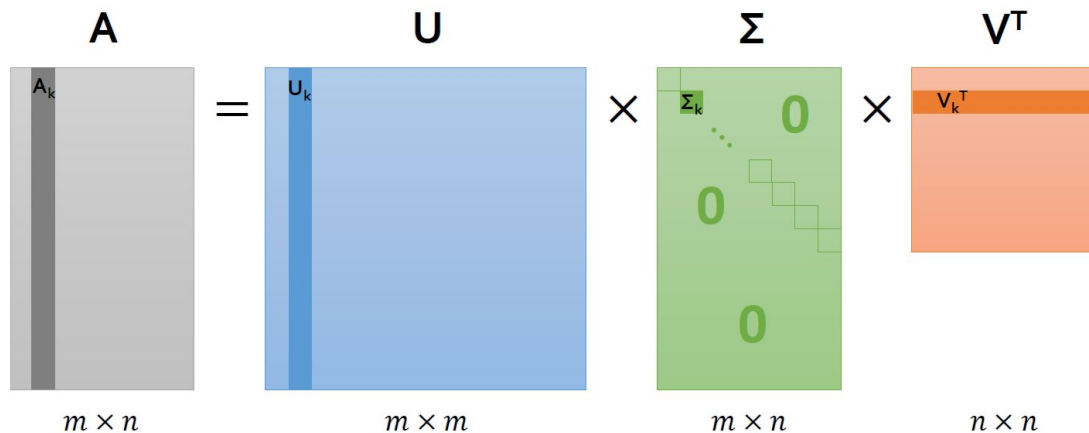
(SVD, Singular Value Decomposition)

차원 축소

특이값 분해 (SVD) (1)

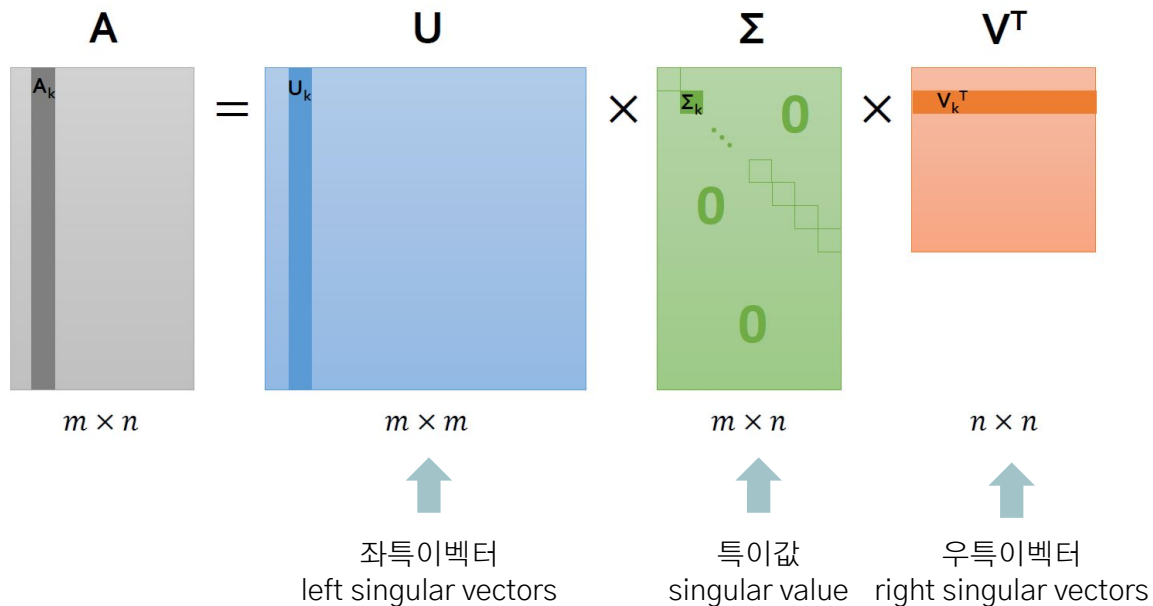
- 특이값 분해(Singular Value Decomposition, SVD)는 임의의 $m \times n$ 차원의 행렬 A 에 대하여 다음과 같이 행렬을 분해 (decomposition)
- 고유값 분해가 정방 행렬에만 적용가능한데 비교하러 비정방형 행렬에도 적용이 가능

$$A_k = U_k \Sigma_k V_k^T$$



특이값 분해 (SVD) (2)

$$A_k = U_k \Sigma_k V_k^T$$



특이값 분해 (SVD) (3)

- 특이값 분해(Singular Value Decomposition, SVD)는 임의의 $m \times n$ 차원의 행렬 A 에 대하여 다음과 같이 행렬을 분해 (decomposition)
- 고유값 분해가 정방 행렬에만 적용가능한데 비해 비정방행 행렬에도 적용이 가능

$$A = U\Sigma V^T$$

여기서 각 3개의 행렬은 다음과 같은 조건을 만족합니다.

$$U : m \times m \text{ 직교행렬 } (AA^T = U(\Sigma\Sigma^T)U^T)$$

$$V : n \times n \text{ 직교행렬 } (A^TA = V(\Sigma^T\Sigma)V^T)$$

$$\Sigma : m \times n \text{ 직사각 대각행렬}$$

직교행렬 (Orthogonal matrix)

선형대수학에서, 직교 행렬(直交行列, orthogonal matrix)은 행벡터와 열벡터가 유클리드 공간의 정규 직교 기저를 이루는 실수 행렬이다.

$$Q^T Q = Q Q^T = I$$

$$Q^{-1} = Q^T$$

$$Q^T Q = I \rightarrow \underbrace{\begin{bmatrix} - & \mathbf{q}_1^T & - \\ & \vdots & \\ - & \mathbf{q}_n^T & - \end{bmatrix}}_{Q^T} \underbrace{\begin{bmatrix} | & & | \\ \mathbf{q}_1 & \cdots & \mathbf{q}_n \\ | & & | \end{bmatrix}}_Q = \underbrace{\begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{bmatrix}}_I$$

대각행렬 (Diagonal matrix)

선형대수학에서, 대각행렬(對角行列, diagonal matrix)은 주대각선을 제외한 곳의 원소가 모두 0인 정사각행렬이다.

대각행렬은 0일 수도, 아닐 수도 있는 대각원소들에 의해 결정된다. $n \times n$ 행렬 $D=[d_{i,j}]$ 가 대각행렬일 필요충분조건은

임의의 $i, j \in \{1, 2, \dots, n\}$ 에 대해, $i \neq j$, $d_{i,j} = 0$

$$\begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{bmatrix}$$

$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \end{bmatrix}$$

$$\begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \\ 0 & 0 & 0 \end{bmatrix}$$

특이값 분해 (SVD) (4)

$$U: m \times m$$

$$V: n \times n$$

$$\Sigma: m \times n$$

$$A = U\Sigma V^T$$

$$(AA^T = U(\Sigma\Sigma^T)U^T)$$

$$(A^T A = V(\Sigma^T \Sigma)V^T)$$

특이값 분해 (SVD) (5)

$$\begin{aligned}
 AA^T &= (U\Sigma V^T)(U\Sigma V^T)^T \\
 &= (U\Sigma V^T)(V\Sigma^T U^T) \quad V^T V = I \\
 &= U(\Sigma\Sigma^T)U^T \\
 &= U\Lambda U^T
 \end{aligned}$$

$$\begin{aligned}
 A^T A &= (U\Sigma V^T)^T (U\Sigma V^T) \\
 &= (V\Sigma^T U^T)(U\Sigma V^T) \quad U^T U = I \\
 &= V(\Sigma\Sigma^T)V^T \\
 &= V\Lambda V^T
 \end{aligned}$$

$$\sigma_i = \sqrt{\lambda_i}$$

$$\Sigma\Sigma^T = \Sigma^T\Sigma = \Lambda$$

특이값 분해 (SVD) (6)

$$A_k = U_k \Sigma_k V_k^T$$

$$A = U \Sigma V^T = \begin{pmatrix} | & | & \cdots & | \\ \vec{u}_1 & \vec{u}_2 & & \vec{u}_n \\ | & | & & | \end{pmatrix} \begin{pmatrix} \sigma_1 & & & 0 \\ & \sigma_2 & & 0 \\ & & \ddots & 0 \\ & & & \sigma_m & 0 \end{pmatrix} \begin{pmatrix} - & \vec{v}_1^T & - \\ - & \vec{v}_2^T & - \\ & \vdots & \\ - & \vec{v}_n^T & - \end{pmatrix}$$

$$= \sigma_1 \vec{u}_1 \vec{v}_1^T + \sigma_2 \vec{u}_2 \vec{v}_2^T + \cdots + \sigma_m \vec{u}_m \vec{v}_m^T$$



좌특이벡터
left singular vectors



특이값
singular value



우특이벡터
right singular vectors

특이값 분해 (SVD) (7)

$$A_k = U_k \Sigma_k V_k^T$$

$$A = \begin{pmatrix} 3 & 6 \\ 2 & 3 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \text{ 일 때,}$$

*A*의 특이값 분해 (Singular Value Decomposition)은

$$A = U \Sigma V^T = \begin{pmatrix} 0.881 & -0.471 & 0 & 0 \\ 0.471 & 0.881 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 7.605 & 0 \\ 0 & 0.394 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0.471 & -0.881 \\ 0.881 & 0.471 \end{pmatrix}^T$$

U



AA^T 의 고유벡터
(eigenvectors of AA^T)

Σ



$\sigma_i = \sqrt{\lambda_i}$
($\lambda_1 \geq \lambda_2 \geq \dots \geq 0$ 이며,
대각원 소외 외 모두 0)

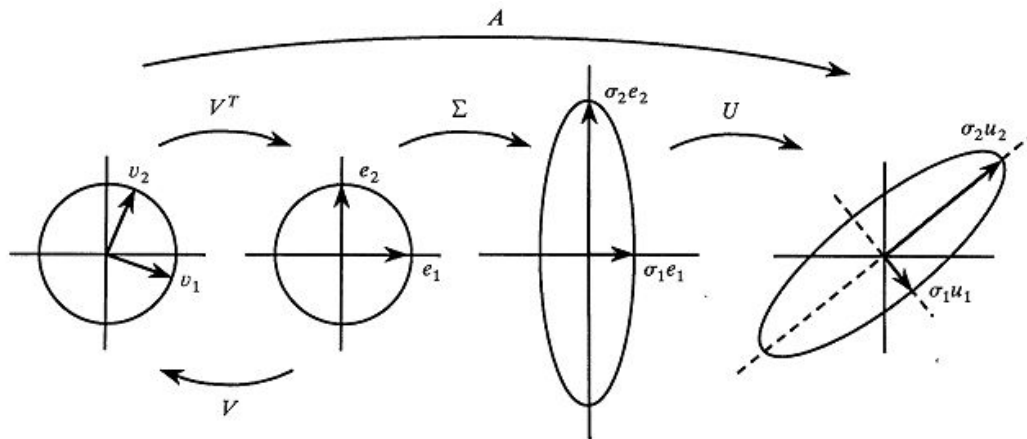
V^T



$A^T A$ 의 고유벡터
(eigenvectors of $A^T A$)

특이값 분해 기하학적 의미

- $A = U\Sigma V^T$ 에서 U, V 는 직교행렬, Σ 는 대각행렬이므로 A 는 x 를 먼저 V^T 에 의해 회전시킨 후 Σ 로 스케일을 변화시키고 다시 U 로 회전시키는 것



https://en.wikipedia.org/wiki/Singular_value_decomposition

Thin SVD

- Σ 행렬 아랫부분 0으로 채워진 영역(비대각 파트)을 제거.
- 원복이 가능

$$A = U_s \Sigma V^T$$

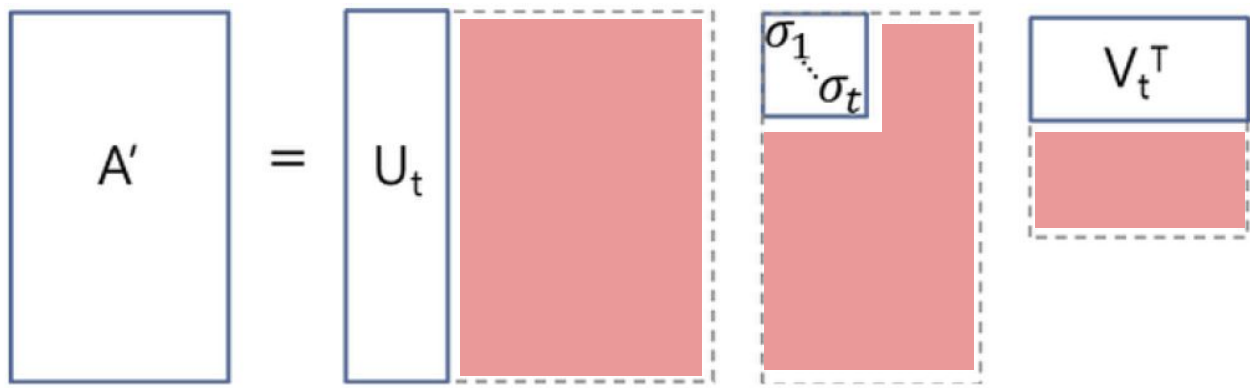
Compact SVD

- Σ 행렬에서 비대각파트 뿐만 아니라 특이값 중에서도 0인 부분을 제거한 형태
- 특이값(대각파트)이 0 초과인 값만 남겨 두고 제거
- 원복이 가능

$$A = U_r \Sigma V_r^T$$

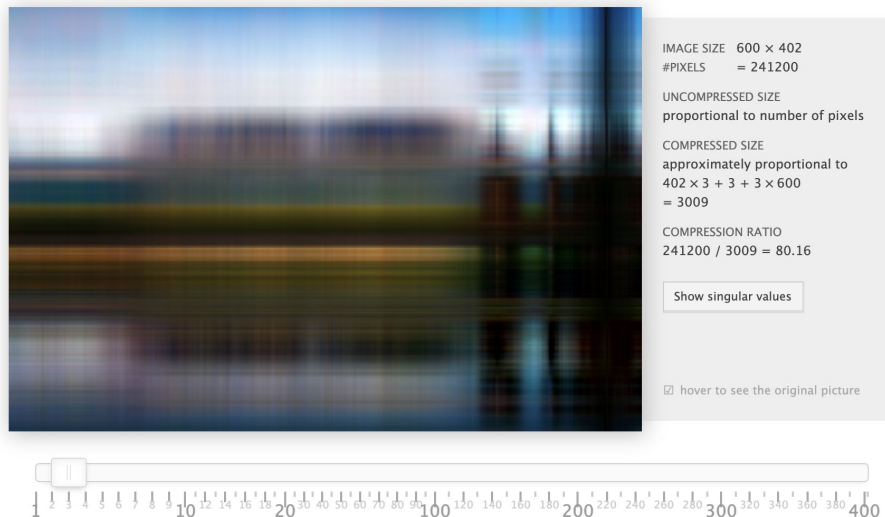
Truncated SVD

- Σ 행렬의 특이값 중 상위 n 개를 선택하여 나머지를 제거한 형태
- **원복이 불가능** (= 정보 손실 발행 = 정보 압축)
- 정보 압축(=정보 손실)을 했음에도 원 행렬에 대한 근사가 가능. (LSA에서 사용)



특이값 분해 활용 예시

Image Compression with Singular Value Decomposition



<http://timbaumann.info/svd-image-compression-demo/>

t-SNE

(t-Stochastic Neighbor Embedding)

차원 축소 (Dimension Reduction)