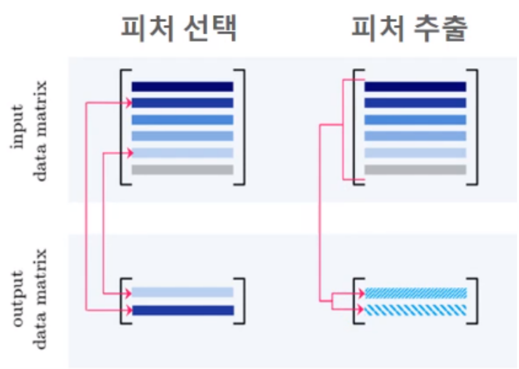
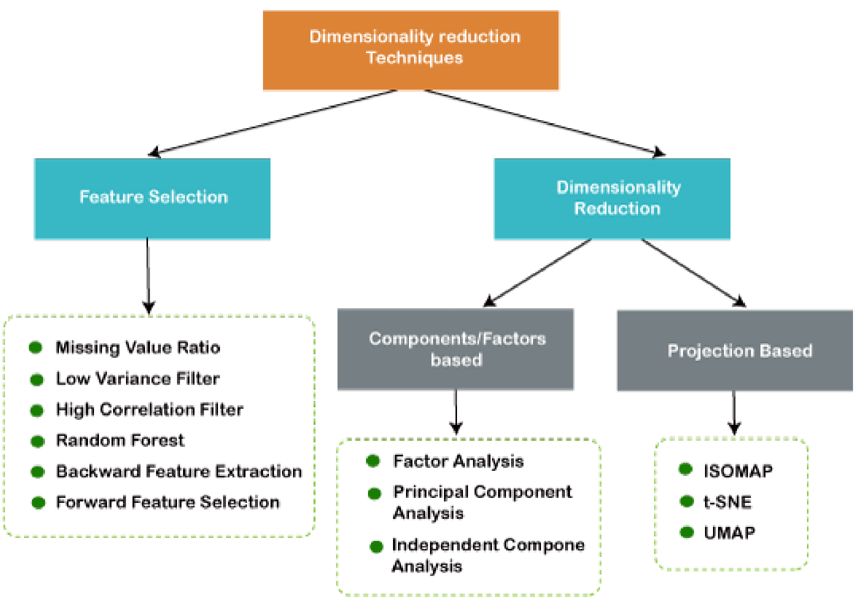
Dimensionality Reduction (1)

Dimensionality Reduction

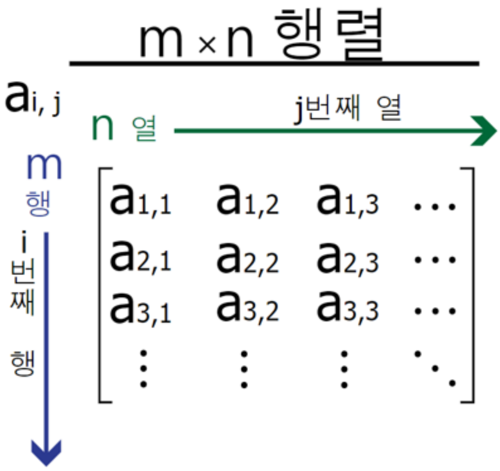
: 어떠한 목적에 따라서 데이터의 양을 줄여 잠재적인 요소를 추출하는 방법. 시간 & 공간 복잡도를 감소시키고, 적은 데이터셋에 대해서도 안정적인 결과를 얻으며, 결과해석을 용이하게 만들기 위함

- feature selection: 특정 피처에 종속성이 강한 불필요한 피처 제거
- feature extraction: 기존 피처를 저차원으로 압축시켜 추출함.





Matrix Property



Idempotent (멱등행렬)

- 자기 자신과 곱해서 자기자신이 나오는 행렬. ex) 단위행렬

> 💡 단위행렬 (identity matrix): 주대각선 성분이 모두 1이고 그 외는 0인 행렬. I

Square Matrix (정방행렬)

- 행과 열의 크기가 같은 행렬. $m = n$
- n차 정방행렬

Symmetric (대칭행렬)

- 기존 행렬과 전치 행렬이 같은 행렬. 즉 $A = A^T$

> 💡 전치행렬 (transposed matrix): 원래의 행렬에서 행과 열을 맞바꾼 행렬. A^T
>
> $$\mathrm{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \Rightarrow \mathrm{A}^T = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix}$$
>
> $$\mathrm{B} = \begin{bmatrix} x & y \\ z & w \end{bmatrix} \Rightarrow \mathrm{B}^T = \begin{bmatrix} x & z \\ y & w \end{bmatrix}$$
>
> $$\mathrm{C} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -3 & 5 & -2 & 7 \end{bmatrix} \Rightarrow \mathrm{C}^T = \begin{bmatrix} 1 & -3 \\ 1 & 5 \\ 1 & -2 \\ 1 & 7 \end{bmatrix}$$
>
> **전치행렬 성질**
>
> 1. $(\mathrm{A}^T)^T = \mathrm{A}$
> 2. $(\mathrm{A}+\mathrm{B})^T = \mathrm{A}^T + \mathrm{B}^T$
> 3. $(\mathrm{AB})^T = \mathrm{B}^T \mathrm{A}^T$
> 4. $(k\mathrm{A})^T = k\,\mathrm{A}^T$ (k는 임의의 상수)

Orthogonal (직교행렬)

- 모든 행렬이 orthogonal set, 즉 직교를 이루는 행렬.
- $A^T A = I,\ A^{-1} = A^T$

💡 역행렬 (inverse matrix) A^{-1}

$$A=\begin{bmatrix}1 & 3\\1 & 4\end{bmatrix},\ B=\begin{bmatrix}4 & -3\\-1 & 1\end{bmatrix} \Rightarrow AB=\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix},\ BA=\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$$

B는 A의 역행렬이다.(= A는 B의 역행렬이다.)

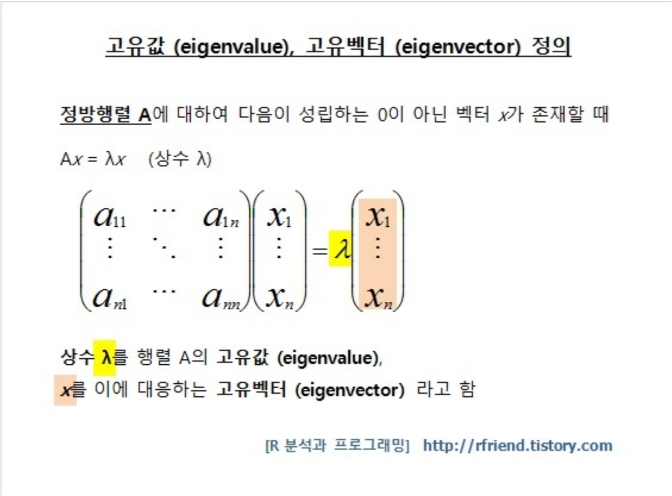
Singular (특이행렬)

- 정방행렬에 A에 대해서 $AB = I$를 만족하여 A의 역행렬이 되는 같은 크기의 행렬 B가 존재하지 않을 때 A는 특이행렬이 된다
- 위 조건을 만족시킬시 A와 B는 정칙행렬 (nonsingular matrix)이 되고, 서로가 서로의 역행렬이 됨

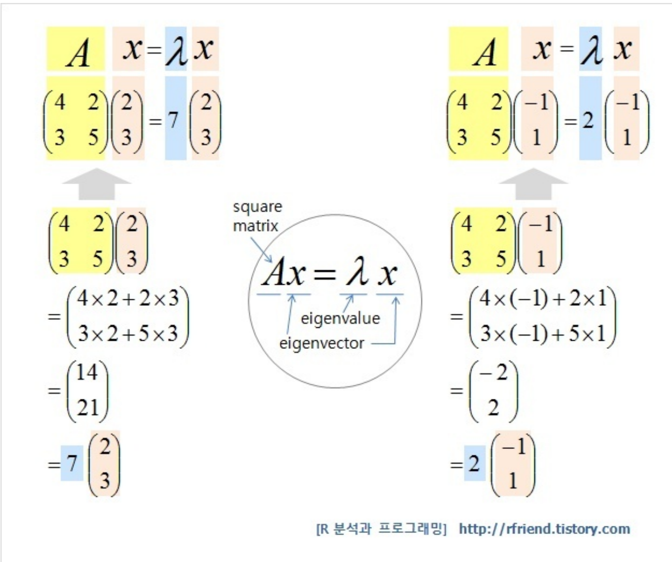
Eigen Decomposition

Eigenvalue, Eigenvector

: Eigenvector 선형변환에 의한 결과가 자기 자신의 상수배 λ가 되는 0이 아닌 벡터고, 그 상수배를 eigenvalue이라 함

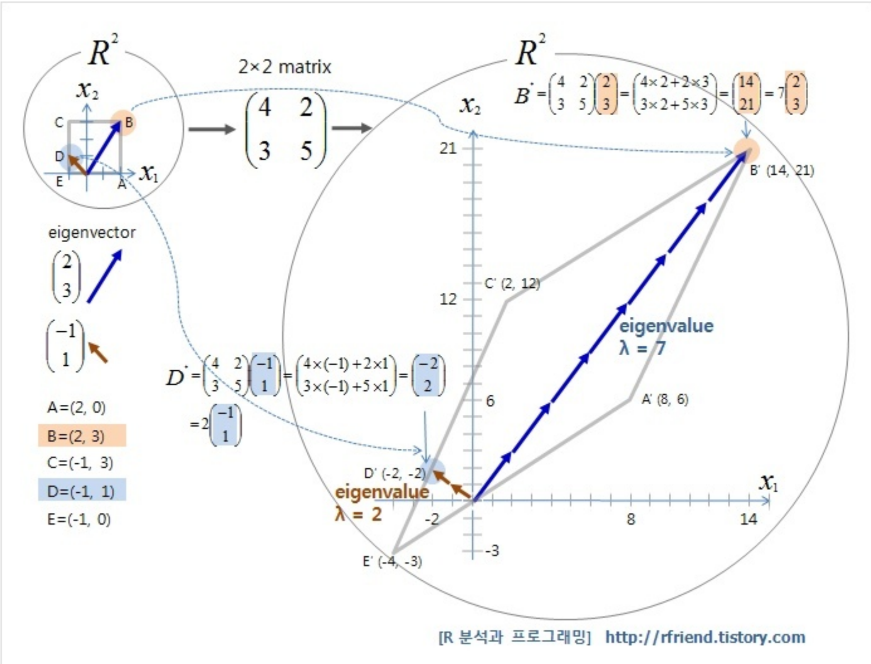


이러한 성질에 의해 벡터 x에 대해 정방행렬 A를 곱하는 결과와 λ를 곱하는 결과가 같음.



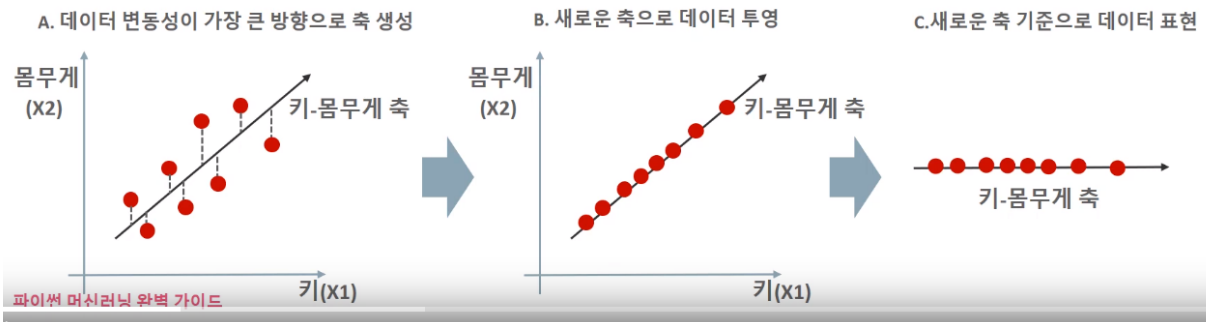
⇒ 행렬의 곱의 결과가 원래 벡터와 **방향**은 같고, **배율**만 상수 λ만큼 비례해서 변함

Ex) 왼쪽 공간이 정방행렬 (4, 3 2, 5)에 의해 오른쪽으로 바뀌는 과정. 방향은 똑같고, 배율만 eigenvalue λ 배수 (7배 2배)만큼 변함



PCA

: 고차원의 원본 데이터를 저 차원의 부분 공간으로 투영하여 데이터를 축소시키는 방법. 데이터 분단위



이때 데이터를 어떤 벡터에 내적하는 것이 최적의 결과를 내주는가? ⇒ 고유벡터.

과정

1. 데이터의 공분산 행렬 추출

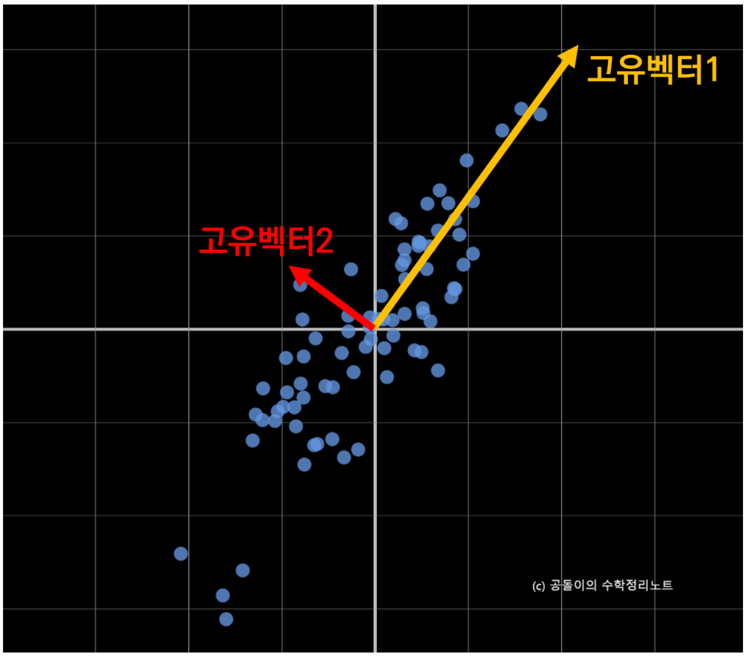
💡 공분산 행렬은 feature 쌍들의 변동이 얼마만큼 함께 변하는지 나타낸다. 대칭행력이기 때문에 eigenvalue로 분해될 수 있음

2. 행렬의 eigenvector 계산

💡 공분산행렬의 eigenvector는 데이터가 어떤 방향으로 분산되어 있는지 나타낸다

3. eigenvalue가 가장 큰 순으로 k개의 eigenvector 추출
4. eigenvalue가 가장 큰 순으로 추출된 eigenvector를 이용해 원본 데이터를 새롭게 변환

💡 eigenvalue는 그 벡터의 방향으로 얼마만큼의 크기로 벡터공간이 늘려지는 지 알려준다. 따라서 고유값이 큰 순서대로 정렬하면 결과적으로 중요한 순서대로 주성분을 구해게 됨



참고

https://www.javatpoint.com/dimensionality-reduction-technique

https://docs.sangyunlee.com/ml/analysis/undefined-1 https://rfriend.tistory.com/181

https://angeloyeo.github.io/2019/07/27/PCA.html

https://hongsusoo.github.io/ai math/math_matrix2/