Statistics II

for Machine Learning

Chap. 6: SVD and PCA

Oh, Hyung Sool

SVD(Singular Vector Decomposition)

- 차원(dimension)이 다른 벡터로 매핑하기 위해서는 직사각형 행렬(rectangular matrix)이 필요 ➔ **선형변환**

$$R^2 : \begin{bmatrix}2\\1\end{bmatrix} \quad \overset{?}{\rightleftarrows} \quad R^3 : \begin{bmatrix}1\\2\\3\end{bmatrix}$$

ex) $\begin{bmatrix}\mathbf{3} & \mathbf{-2} & \mathbf{1}\\ \mathbf{2} & \mathbf{1} & \mathbf{-1}\end{bmatrix}\begin{bmatrix}1\\2\\3\end{bmatrix} = \begin{bmatrix}2\\1\end{bmatrix}$

: 3차원의 행렬(3×1)에 대해 2차원 행렬(2×3)로 선형 변환을 하면 2차원 행렬이 된다.

- R^m : [$A: m \times n$] : 직사각형 행렬 A는 $n(or\ m)$ 차원의 벡터를 취하여 $m(or\ n)$ 차원의 벡터로 변환할 수 있다.

SVD(Singular Vector Decomposition)

- $\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\end{bmatrix}$: "Dimension Eraser"

  ex) $\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\end{bmatrix}\begin{bmatrix}1\\2\\7\end{bmatrix} = \begin{bmatrix}1\\2\end{bmatrix}$ $\quad\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\end{bmatrix}\begin{bmatrix}1\\2\\1\end{bmatrix} = \begin{bmatrix}1\\2\end{bmatrix}$ $\quad\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\end{bmatrix}\begin{bmatrix}1\\2\\11\end{bmatrix} = \begin{bmatrix}1\\2\end{bmatrix}$

- $\begin{bmatrix}1 & 0\\0 & 1\\0 & 0\end{bmatrix}$: "Dimension Adder"

  ex) $\begin{bmatrix}1 & 0\\0 & 1\\0 & 0\end{bmatrix}\begin{bmatrix}1\\2\end{bmatrix} = \begin{bmatrix}1\\2\\0\end{bmatrix}$ $\quad\begin{bmatrix}1 & 0\\0 & 1\\0 & 0\end{bmatrix}\begin{bmatrix}3\\6\end{bmatrix} = \begin{bmatrix}3\\6\\0\end{bmatrix}$ $\quad\begin{bmatrix}1 & 0\\0 & 1\\0 & 0\end{bmatrix}\begin{bmatrix}4\\8\end{bmatrix} = \begin{bmatrix}4\\8\\0\end{bmatrix}$

- $\underset{2\times2}{\begin{bmatrix}4 & 0\\0 & 2\end{bmatrix}}\underset{2\times3}{\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\end{bmatrix}} = \underset{2\times3}{\begin{bmatrix}4 & 0 & 0\\0 & 2 & 0\end{bmatrix}}$ $\qquad\underset{2\times3}{\begin{bmatrix}4 & 0 & 1\\0 & 2 & 1\end{bmatrix}}\underset{3\times2}{\begin{bmatrix}1 & 0\\0 & 1\\0 & 0\end{bmatrix}} = \underset{2\times2}{\begin{bmatrix}4 & 0\\0 & 2\end{bmatrix}}$

SVD(Singular Vector Decomposition)

- 대칭 행렬(Square Matrix)을 인위적으로 만드는 방법

$$\underset{A^{(2\times3)}}{\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix}}\ \underset{A^{T(3\times2)}}{\begin{bmatrix}1 & 4\\2 & 5\\3 & 6\end{bmatrix}} = \underset{AA^{T(2\times2)}}{\begin{bmatrix}14 & 32\\32 & 77\end{bmatrix}}$$

: (2×2) 정사각형 행렬이 되며, 대칭 행렬이 된다.

전치행렬을 왼쪽에 두면

$$\underset{A^{T(3\times2)}}{\begin{bmatrix}1 & 4\\2 & 5\\3 & 6\end{bmatrix}}\ \underset{A^{(2\times3)}}{\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix}} = \underset{AA^{T(3\times3)}}{\begin{bmatrix}17 & 22 & 27\\22 & 29 & 36\\27 & 36 & 45\end{bmatrix}}$$

: (3×3) 정사각형 행렬이 되며, 대칭 행렬이 된다.

SVD(Singular Vector Decomposition)

- 이런 특성을 일반화 하면,

$$A^{(m\times n)}$$

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

$$AA^{T(m\times m)} : S_L$$

$$\begin{bmatrix} 14 & 32 \\ 32 & 77 \end{bmatrix}$$

$$A^T A^{(n\times n)} : S_R$$

$$\begin{bmatrix} 17 & 22 & 27 \\ 22 & 29 & 36 \\ 27 & 36 & 45 \end{bmatrix}$$

$$[\ S_L\ ] \quad \begin{bmatrix} | & | \\ \vec{u_1} & \vec{u_2} \\ | & | \end{bmatrix} \xrightarrow{\textit{Left Singular Vectors}} A$$

$$[\ S_R\ ] \begin{bmatrix} | & | & | \\ \vec{v_1} & \vec{v_2} & \vec{v_3} \\ | & | & | \end{bmatrix} \xrightarrow{\textit{Right Singular Vectors}} A$$

- 이때의 $\vec{u_1}$, $\vec{u_2}$, $\vec{v_1}$, $\vec{v_2}$, $\vec{v_3}$: 모두 **고유벡터** 이다.

SVD(Singular Vector Decomposition)

- $\begin{matrix} | & | \\ \vec{u}_1 & \vec{u}_2 \\ | & | \end{matrix}$ 고유값 : λ_1 , λ_2 : $\lambda_1 > \lambda_2$

- $\begin{matrix} | & | & | \\ \vec{v}_1 & \vec{v}_2 & \vec{v}_3 \\ | & | & | \end{matrix}$ 고유값 : λ_1 , λ_2 , λ_3 : $\lambda_1 > \lambda_2 > \lambda_3$

- $\boldsymbol{S_L}$ 의 고유값 : λ_{L1} , λ_{L2} 과 $\boldsymbol{S_R}$ 의 고유값 : λ_{R1} , λ_{R2} , λ_{R3} 에서 $\boldsymbol{\lambda_{L1} = \lambda_{R1}}$, $\boldsymbol{\lambda_{L2} = \lambda_{R2}}$

- $\sqrt{\lambda_1} = \sigma_1$, $\sqrt{\lambda_2} = \sigma_2$: 벡터 $\boldsymbol{A}$의 특이값(***Singular Values***) 이라고 한다.

SVD(Singular Vector Decomposition)

- 이런 특성을 일반화 하면, 모든 행렬은 3가지의 특별한 행렬(singular matrix)로 분해할 수 있다.

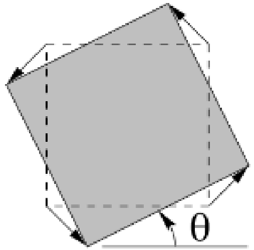
Rectangular matrix $\boldsymbol{A} = \underline{\mathrm{U}}\ \underline{\Sigma}\ \underline{V^T}$

U: *Orthogonal matrix*

Σ: *Diagonal matrix*

V^T: *Orthogonal matrix*

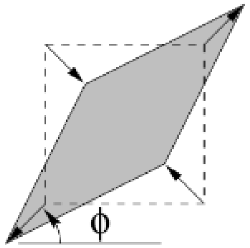
$$\underset{2\times 3}{\begin{bmatrix} ? & ? & ? \\ ? & ? & ? \end{bmatrix}} = \begin{bmatrix} | & | \\ \vec{u}_1 & \vec{u}_2 \\ | & | \end{bmatrix} \begin{bmatrix} \boldsymbol{\sigma_1} & 0 & 0 \\ 0 & \boldsymbol{\sigma_2} & 0 \end{bmatrix} \begin{bmatrix} - & \vec{v}_1 & - \\ - & \vec{v}_2 & - \\ - & \vec{v}_3 & - \end{bmatrix}$$



$\underline{\boldsymbol{U}}$: *Rotation*

$\underline{\boldsymbol{\Sigma}}$: *Stretching*

$\underline{\boldsymbol{V^T}}$: *Inverse Rotation*



SVD(Singular Vector Decomposition)

$$\boldsymbol{A} = \underbrace{\mathrm{U}}_{Rotation}\ \underbrace{\Sigma}_{Stretching}\ \underbrace{V^T}_{Inverse\ Rotation}$$

- 특이값 분해(SVD)는 직사각 행렬($\boldsymbol{A}$) 을 특이값(Σ)과 특정한 구조(U, V^T)로 분해하는 것
- 형태적인 변환은 *Diagonal matrix* Σ 의 *Stretching* 값에 따라 달라지게 되고, 차원의 변화는 U,V 에 따른다.
- $\mathrm{U} = \begin{bmatrix} | & | \\ \overrightarrow{u_1} & \overrightarrow{u_2} \\ | & | \end{bmatrix}$
  - ✓ 열 벡터들은 **서로 수직이고 크기가 1인 단위 벡터**이다.
  - ✓ U 의 각 열 벡터 $\overrightarrow{u_i}$ 는 *A*의 열들을 선형 독립적으로 나타내는 **새로운 기저(basis)** 역할을 한다.
  - ✓ 첫 번째 열 벡터는 가장 큰 특이값에 대응하는 특이 벡터로, 두 번째 열 벡터는 두 번째로 큰 특이값에 대응하는 특이 벡터로 해석
- $V^T = \begin{bmatrix} - & \overrightarrow{v_1} & - \\ - & \overrightarrow{v_2} & - \\ - & \overrightarrow{v_3} & - \end{bmatrix}$
  - ✓ 행 벡터들은 **서로 수직이고 크기가 1인 단위 벡터**이다.
  - ✓ V^T의 각 행 벡터 $\overrightarrow{v_i}$는 *A*의 행들을 선형 독립적으로 나타내는 **새로운 기저(basis)** 역할을 한다.
  - ✓ 첫 번째 행 벡터는 가장 큰 특이값에 대응하는 특이 벡터로, 두 번째 행 벡터는 두 번째로 큰 특이값에 대응하는 특이 벡터로 해석

고유값 분해 & 특이값 분해

- 모든 행렬은 특정한 구조로 **분해가 가능**한다: 대수학에서의 인수분해처럼!!
  - *Rotation Factor* : 방향 변환
  - *Stretching Factor* : 크기 변환 ➔ ***Scaling***

- 정방 행렬(Square Matrix)은 고유값 분해로, 직사각 행렬(Reactangular Matrix)은 특이값 분해로

- 정방 행렬(Square Matrix)의 고유값 분해
  - $AX = \lambda X = \ Stretching\ Factor(Scaling) \times Eigenvectors =$ 변화의 크기(분산) × 변화의 주축
  - A를 고유값으로 분해하는 것의 의미는
    - A로 인한 **변화의 크기**를 찾아내는 것
    - A로 인한 **변화가 생기는 주축(주요 요인)**을 찾아내는 것

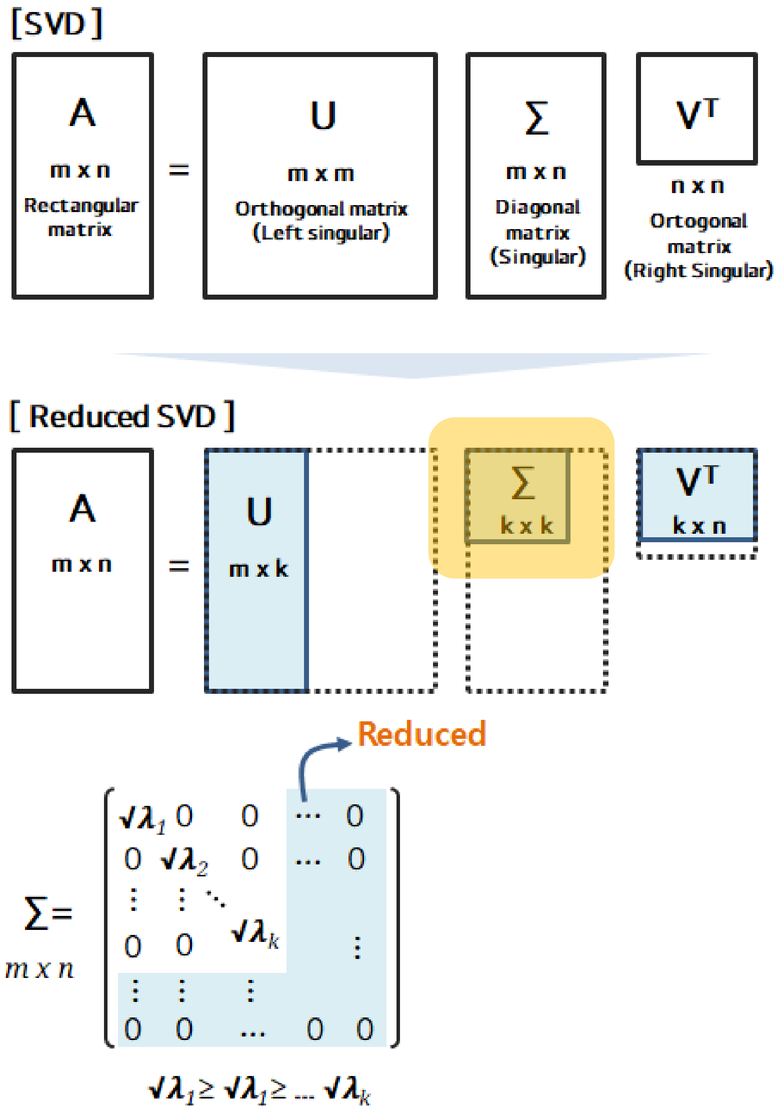
고유값 분해 & 특이값 분해

- 직사각 행렬(Square Matrix)을 고유값으로 분해하기 위해서는
  - 먼저, 직사각 행렬 $\boldsymbol{A}^{(\boldsymbol{m}\times\boldsymbol{n})}$를 정방 행렬 형태로 만들어야만 한다.
  - 정방 행렬인 $\boldsymbol{AA}^{\boldsymbol{T}(\boldsymbol{m}\times\boldsymbol{m})} = \boldsymbol{U}$ 과 $\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}^{(\boldsymbol{n}\times\boldsymbol{n})} = \boldsymbol{V}$ 를 만든다. $\Rightarrow$ $\boldsymbol{A} = \boldsymbol{U\Sigma V}^{\boldsymbol{T}}$

- 직사각 행렬(Square Matrix)의 특이값 분해
  - $A = U\Sigma V^T \quad \Rightarrow \quad AV = U\Sigma$
  - $AV = U\Sigma = Rotation\ Factor \times Stretching\ Factor(Scaling) =$ **직교하는 방향(주축)** $\times$**변화의 크기(분산)**
  - A를 고유값으로 분해하는 것의 의미는
    - A로 인한 **변화의 크기**를 찾아내는 것
    - A로 인한 **변화가 생기는 주축**, 즉 데이터의 **주요 요인** 찾아내는 것

- 고유값 분해: 주축을 이루는 벡터(**주요 요인**)와 변화의 크기(**분산**)를 찾는 것
- 특이값 분해: 직교하는 벡터들(**주요 요인**)과 그 방향에서의 크기(**분산**)를 찾는 것

SVD(Singular Vector Decomposition)



- reduced SVD는 특이값들 중에서 0인 것들을 제외한 k 차원 만을 가지고 한다.
- reduced SVD는 PCA 에서 중요한 개념

SVD(Singular Vector Decomposition)

```
cross_matrix <- matrix(c(0, 0, 0, 1, 1, 0, 0, 0,
                         0, 0, 0, 1, 1, 0, 0, 0,
                         0, 0, 0, 1, 1, 0, 0, 0,
                         1, 1, 1, 1, 1, 1, 1, 1,
                         1, 1, 1, 1, 1, 1, 1, 1,
                         1, 1, 1, 1, 1, 1, 1, 1,
                         1, 1, 1, 1, 1, 1, 1, 1,
                         0, 0, 0, 1, 1, 0, 0, 0,
                         0, 0, 0, 1, 1, 0, 0, 0,
                         0, 0, 0, 1, 1, 0, 0, 0), byrow = TRUE, nrow=10)

cross_matrix
cross_svd <- svd(cross_matrix)
str(cross_svd)

round(cross_svd$u[,c(1,2)] %*% diag(cross_svd$d[c(1,2)]) %*% t(cross_svd$v[,c(1,2)]))
```

```
> cross_matrix
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
 [1,]    0    0    0    1    1    0    0    0
 [2,]    0    0    0    1    1    0    0    0
 [3,]    0    0    0    1    1    0    0    0
 [4,]    1    1    1    1    1    1    1    1
 [5,]    1    1    1    1    1    1    1    1
 [6,]    1    1    1    1    1    1    1    1
 [7,]    1    1    1    1    1    1    1    1
 [8,]    0    0    0    1    1    0    0    0
 [9,]    0    0    0    1    1    0    0    0
[10,]    0    0    0    1    1    0    0    0
```

```
> cross_svd <- svd(cross_matrix)
> str(cross_svd)
List of 3
 $ d: num [1:8] 6.00 2.83 1.99e-16 2.13e-32 5.90e-49 ...
 $ u: num [1:10, 1:8] -0.154 -0.154 -0.154 -0.463 -0.463 ..
 $ v: num [1:8, 1:8] -0.309 -0.309 -0.309 -0.463 -0.463 ...
```

```
> round(cross_svd$u[,c(1,2)] %*% diag(cross_svd$d[c(1,2)]) %*% t(cross_svd$v[,c(1,2)]))
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
 [1,]    0    0    0    1    1    0    0    0
 [2,]    0    0    0    1    1    0    0    0
 [3,]    0    0    0    1    1    0    0    0
 [4,]    1    1    1    1    1    1    1    1
 [5,]    1    1    1    1    1    1    1    1
 [6,]    1    1    1    1    1    1    1    1
 [7,]    1    1    1    1    1    1    1    1
 [8,]    0    0    0    1    1    0    0    0
 [9,]    0    0    0    1    1    0    0    0
[10,]    0    0    0    1    1    0    0    0
```

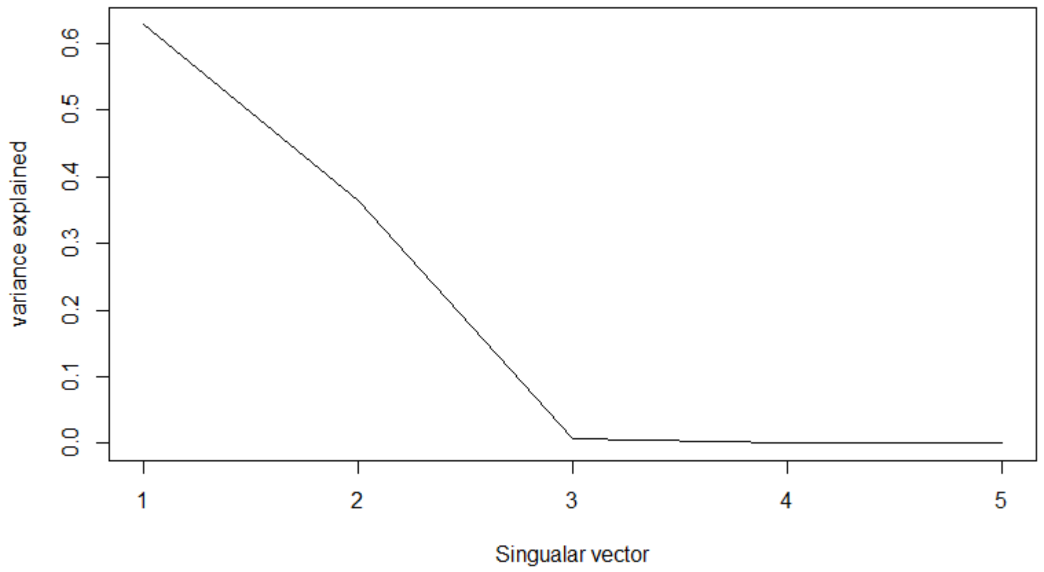
SVD(Singular Vector Decomposition)

```
> movieRating
    스타워즈 아바타 혹성탈출 사랑과영혼 타이타닉
u1        1      1        1          0        0
u2        3      3        3          0        0
u3        4      4        4          0        0
u4        5      5        5          0        0
u5        0      0        0          4        4
u6        0      0        0          5        5
u7        0      0        0          2        2

> movieRating_svd <- svd(movieRating); movieRating_svd
$d
[1] 12.369317  9.486833  0.000000  0.000000  0.000000

$u
          [,1]       [,2]       [,3]       [,4]       [,5]
[1,] -0.140028  0.0000000  0.4174829 -0.5601120 -0.4200840
[2,] -0.420084  0.0000000  0.1538365 -0.2063933  0.8452050
[3,] -0.560112  0.0000000  0.2051153  0.7248090 -0.2063933
[4,] -0.700140  0.0000000 -0.3398907 -0.3439888 -0.2579916
[5,]  0.000000 -0.5962848  0.6444444  0.0000000  0.0000000
[6,]  0.000000 -0.7453560 -0.4444444  0.0000000  0.0000000
[7,]  0.000000 -0.2981424 -0.1777778  0.0000000  0.0000000

$v
           [,1]       [,2]       [,3]       [,4]       [,5]
[1,] -0.5773503  0.0000000  0.0000000  0.0000000  0.8164966
[2,] -0.5773503  0.0000000  0.0000000 -0.7071068 -0.4082483
[3,] -0.5773503  0.0000000  0.0000000  0.7071068 -0.4082483
[4,]  0.0000000 -0.7071068 -0.7071068  0.0000000  0.0000000
[5,]  0.0000000 -0.7071068  0.7071068  0.0000000  0.0000000
```




```
> cor(t(U%*%D))
     [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,]    1    1    1    1   -1   -1   -1
[2,]    1    1    1    1   -1   -1   -1
[3,]    1    1    1    1   -1   -1   -1
[4,]    1    1    1    1   -1   -1   -1
[5,]   -1   -1   -1   -1    1    1    1
[6,]   -1   -1   -1   -1    1    1    1
[7,]   -1   -1   -1   -1    1    1    1
> cor(D%*%t(V))
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    1    1   -1   -1
[2,]    1    1    1   -1   -1
[3,]    1    1    1   -1   -1
[4,]   -1   -1   -1    1    1
[5,]   -1   -1   -1    1    1
```

```
> D <- diag(movieRating_svd$d[1:2]);D
         [,1]     [,2]
[1,] 12.36932 0.000000
[2,]  0.00000 9.486833
```

SVD(Singular Vector Decomposition)

특이값: 10개

특이값: 50개

특이값: 100개

원 이미지

PCA(Principle Component Ana.)