17.1. Phương pháp phân tích suy biến

17.1.1. Hệ trực giao và ma trận trực giao

Hệ trực giao và ma trận trực giao là những khái niệm cơ bản trong đại số tuyến tính. Thông qua hệ trực giao ta có thể tìm được hệ cơ sở của các không gian cao chiều.

Hệ trực giao: Một hệ véc tơ cơ sở $\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_D\} \in \mathbb{R}^K$ được gọi là một *hệ trực giao* (*orthogonal*) nếu thỏa mãn hệ điều kiện: $$\begin{cases} ||\mathbf{u}_i||_2^2 > 0 \\ \mathbf{u}_i^\intercal \mathbf{u}_j = 0 \; \forall i \neq j \end{cases}$$

Ký hiệu $||\mathbf{u}_i||_2^2$ chính là bình phương của norm chuẩn bậc hai (L_2 norm) của véc tơ $\mathbf{u}_i$. Như vậy các chiều của hệ trực giao là vuông góc với nhau đôi một.

Hệ trực chuẩn: *Hệ trực chuẩn* (*orthonormal*) là một trường hợp đặc biệt của hệ trực giao khi giá trị của chuẩn bậc hai $||\mathbf{u}_i||_2^2 = 1$, $\forall i$. Một tập hợp các véc tơ đơn vị bất kỳ $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_Q\}$ trong không gian K chiều ($Q \leq K$) sẽ tạo thành một hệ trực chuẩn.

Ma trận trực giao: *Ma trận trực giao* (*orthogonal matrix*) là ma trận vuông thỏa mãn các dòng và cột của nó là một hệ trực chuẩn. Điều đó có nghĩa là một ma trận trực giao $\mathbf{U} \in \mathbb{R}^{D \times D}$ thoả mãn:

$$\mathbf{U}^\intercal \mathbf{U} = \mathbf{I}_D$$

Với $\mathbf{I}_D$ là ma trận đơn vị với kích thước D chiều.

Một phép xoay dựa trên hệ trực chuẩn sẽ bảo toàn giá trị tích vô hướng giữa hai véc tơ. Tính chất thú vị này có thể chứng minh như sau: Hai véc tơ bất kì $\mathbf{x}$ và $\mathbf{y}$ có kết quả khi thực hiện phép xoay theo ma trận trực giao $\mathbf{U}$ lần lượt là $\mathbf{Ux}$ và $\mathbf{Uy}$. Khi đó:

$$(\mathbf{Ux})^\intercal (\mathbf{Uy}) = \mathbf{x}^\intercal \mathbf{U}^\intercal \mathbf{Uy} = \mathbf{x}^\intercal \mathbf{I}_n \mathbf{y} = \mathbf{x}^\intercal \mathbf{y}$$

Ma trận hiệp phương sai: Ma trận hiệp phương sai (*covariance matrix*) của một ma trận dữ liệu đầu vào $\mathbf{X} \in \mathbb{R}^{n \times d}$ có các véc tơ cột lần lượt là $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_d]$ là một ma trận vuông và bán xác định dương. Phần tử σ_{ij} của ma trận hiệp phương sai của $\mathbf{X}$ chính là hiệp phương sai giữa hai véc tơ cột $\mathbf{x}_i$ và $\mathbf{x}_j$ theo công thức:

$$\sigma_{ij} = \text{cov}(\mathbf{x}_i, \mathbf{x}_j) = [\mathbf{x}_i - \mathbf{E}(\mathbf{x}_i)][\mathbf{x}_j - \mathbf{E}(\mathbf{x}_j)] = [\mathbf{x}_i - \bar{\mathbf{x}_i}][\mathbf{x}_j - \bar{\mathbf{x}_j}] \quad (1)$$

Với $\bar{\mathbf{x}_i}$ là véc tơ kích thước d có các giá trị bằng nhau và bằng trung bình của véc tơ cột của $\mathbf{x}_i$.

Hiệp phương sai sẽ đo lường mối quan hệ cùng chiều hoặc nghịch chiều giữa hai biến. Hiệp phương sai của một biến với chính nó chính là phương sai đo lường mức độ biến động của biến. Ma trận hiệp phương sai sẽ cho ta biết quan hệ giữa các cột của ma trận và phương sai của từng cột.

Giả sử $\bar{\mathbf{X}} = [\bar{\mathbf{x}_1}, \bar{\mathbf{x}_2}, \ldots, \bar{\mathbf{x}_d}]$ là ma trận trung bình của toàn bộ các quan sát của $\mathbf{X}$. Ma trận $\bar{\mathbf{X}}$ có các dòng là bằng nhau và bằng giá trị của điểm trung bình. Khi đó ma trận hiệp phương sai còn được tính theo công thức:

$$\text{cov}(\mathbf{X}, \mathbf{X}) = (\mathbf{X} - \bar{\mathbf{X}})^\intercal (\mathbf{X} - \bar{\mathbf{X}})$$

Ma trận hiệp phương sai ẩn chứa nhiều tính chất thú vị mà trong phương pháp PCA chúng ta sẽ có dịp gặp lại ma trận này.

Ma trận đường chéo: Một ma trận $\mathbf{D}$ là ma trận đường chéo khi các phần tử của nó thỏa mãn:

$$d_{ii} \neq 0, d_{ij} = 0 \; \forall i \neq j$$

Hay nói cách khác ma trận có các phần tử trên đường chéo chính khác 0 và các phần tử còn lại bằng 0. Ma trận đường chéo có thể không vuông. Ma trận đơn vị $\mathbf{I}_n$ là một dạng ma trận đường chéo khi nó vừa là một ma trận vuông và đồng thời các phần tử trên đường chéo chính bằng 1. Ngoài ra ta có thể nhận

Contents

thấy mối liên hệ giữa **ma trận trực giao** và **ma trận đường chéo** đó là một ma trận $\mathbf{U} \in \mathbb{R}^{D \times D}$ có các cột tạo thành một hệ trực giao thì tích của nó với ma trận chuyển vị của nó sẽ tạo thành một ma trận đường chéo.

$$\mathbf{U}^\intercal\mathbf{U} = \begin{bmatrix} \mathbf{u}_1^2 & \mathbf{u}_1\mathbf{u}_2 & \mathbf{u}_1\mathbf{u}_3 & \dots & \mathbf{u}_1\mathbf{u}_D \\ \mathbf{u}_2\mathbf{u}_1 & \mathbf{u}_2^2 & \mathbf{u}_2\mathbf{u}_3 & \dots & \mathbf{u}_2\mathbf{u}_D \\ \vdots & \vdots & \ddots & \vdots & \\ \mathbf{u}_D\mathbf{u}_1 & \mathbf{u}_D\mathbf{u}_2 & \mathbf{u}_D\mathbf{u}_3 & \dots & \mathbf{u}_D^2 \end{bmatrix} = \begin{bmatrix} \mathbf{u}_1^2 & 0 & 0 & \dots & 0 \\ 0 & \mathbf{u}_2^2 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & 0 & \dots & \mathbf{u}_D^2 \end{bmatrix}$$

17.1.2. Các tính chất liên quan tới trace của ma trận

Trace của ma trận: hàm *trace* của ma trận $\mathbf{A}$ kí hiệu là $\mathbf{trace(A)}$ là tổng của tất cả các phần tử trên đường chéo của ma trận đó. Một số đẳng thức của *trace*:

1. $\mathbf{trace(A)} = \mathbf{trace(A^\intercal)}$. Điều này là hiển nhiên do phép chuyển vị không làm thay đổi các vị trí trên đường chéo chính của $\mathbf{A}$.
2. $\mathbf{trace(AB)} = \mathbf{trace(BA)}$. Chứng minh tính chất này khá đơn giản. Giả sử $\mathbf{A}^{(i)}$ là vector dòng thứ i của $\mathbf{A}$ và $\mathbf{B}_j$ là vector cột thứ j của $\mathbf{B}$. Khi đó phần tử AB_{ij} ở dòng thứ i cột thứ j của ma trận tích $\mathbf{AB}$ là:

$$AB_{ij} = \mathbf{A}^{(i)}\mathbf{B}_j = \sum_k (a_{ik}b_{kj})$$

Do đó:

$$\mathbf{trace(AB)} = \sum_i AB_{ii} = \sum_i \sum_k (a_{ik}b_{ki}) \quad (1)$$

Hoàn toàn biến đổi tương tự:

$$\mathbf{trace(BA)} = \sum_i BA_{ii} = \sum_i \sum_k (b_{ik}a_{ki}) \quad (2)$$

Ta nhận thấy chỉ số i, k bình đẳng trong cả 2 biểu thức (1) và (2) nên nếu hoán vị i và k cho nhau không làm thay đổi kết quả tổng. Mặt khác phép hoán vị này sẽ biến biểu thức (1) thành (2) nên suy ra giá trị của 2 biểu thức là bằng nhau.

1. $\mathbf{trace(ABC)} = \mathbf{trace(CAB)} = \mathbf{trace(BCA)}$. Tính chất này suy ra từ tính chất 2.
2. $\mathbf{trace(A + B)} = \mathbf{trace(A)} + \mathbf{trace(B)}$. Dễ dàng suy ra tính chất này do $\mathbf{A, B}$ phải cùng kích thước và vị trí các phần tử trên đường chéo chính của 2 ma trận này là trùng nhau.

17.1.3. Véc tơ riêng và trị riêng

Véc tơ riêng và trị riêng: Véc tơ riêng của một ma trận vuông $\mathbf{A} \in \mathbb{R}^{n \times n}$ là một véc tơ đặc biệt vì nó sở hữu tính chất khi nhân với ma trận $\mathbf{A}$ thì kết quả thu được chính là véc tơ đó nhân với một đại lượng vô hướng λ:

$$\mathbf{Ax} = \lambda\mathbf{x}$$

Chúng ta gọi $\mathbf{x}$ là vector riêng (*Eigenvector*) ứng với trị riêng (*Eigenvalue*) λ. Khai triển biểu thức trên để đưa về dạng tích của $\mathbf{x}$:

$$\begin{aligned} \mathbf{Ax} - \lambda\mathbf{x} &= 0 \\ \mathbf{Ax} - \lambda\mathbf{I}_n\mathbf{x} &= 0 \\ (\mathbf{A} - \lambda\mathbf{I}_n)\mathbf{x} &= 0 \quad (3) \end{aligned}$$

Dòng thứ 2 đạt được là nhờ tích của một ma trận với ma trận đơn vị thì bằng chính nó ($\mathbf{I}_n$ là kí hiệu cho ma trận đơn vị bậc n).

Trị riêng và vector riêng của ma trận có những tính chất đặc biệt sau:

1. Nếu $\mathbf{x}$ là vector riêng tương ứng của trị riêng λ thì $k\mathbf{x}$ cũng là một vector riêng của λ. Tính chất này cho thấy một trị riêng có thể có nhiều vector riêng. Tuy nhiên với một vector riêng chỉ có duy nhất một trị riêng.
2. Một ma trận có n trị riêng bao gồm cả lặp lại và trị riêng phức. Điều này có được là vì trị riêng là nghiệm của một đa thức bậc n. Thật vậy, biểu thức (3) về bản chất là tổ hợp tuyến tính của các cột ma trận $\mathbf{A} - \lambda\mathbf{I}_n$ với các phần tử của vector $\mathbf{x}$. Do kết quả bằng 0 nên các cột của ma trận $\mathbf{A} - \lambda\mathbf{I}_n$ là phụ thuộc tuyến tính. Từ đó suy ra $\mathbf{det}(\mathbf{A} - \lambda\mathbf{I}_n) = 0$. Triển khai định thức này ta thu được một đa thức $\mathbf{P}_n(\lambda)$ bậc n. Do đó ma trận có n trị riêng bao gồm cả lặp và phức.
3. Khi ma trận $\mathbf{A}$ là ma trận đối xứng thì các trị riêng của $\mathbf{A}$ là các số thực.
4. Khi ma trận $\mathbf{A}$ là ma trận xác định dương thì các trị riêng của nó là các số thực dương và nếu ma trận $\mathbf{A}$ là ma trận bán xác định dương thì các trị riêng của nó không âm. Chứng minh như sau: $\lambda\mathbf{x}^\intercal\mathbf{x} = \mathbf{x}^\intercal\mathbf{A}\mathbf{x} > 0$ khi $\mathbf{A}$ xác định dương. Mặt khác $\lambda\mathbf{x}^\intercal\mathbf{x} = \lambda||\mathbf{x}||_\mathbf{2}^\mathbf{2}, ||\mathbf{x}||_\mathbf{2}^\mathbf{2} > 0\ \forall\mathbf{x} \neq 0$. Suy ra $\lambda > 0$, như vậy mọi trị riêng của $\mathbf{A}$ đều dương. Chứng minh tương tự cho trường hợp $\mathbf{A}$ bán xác định đương.
5. Tổng các phần tử trên đường chéo chính của ma trận $\mathbf{A} \in \mathbb{R}^{n\times n}$ thì bằng tổng các trị riêng. Để chứng minh công thức này cần sử dụng đến phép phân tích riêng sẽ được trình bày bên dưới. Khi ma trận $\mathbf{A}$ độc lập tuyến tính nó có thể biểu diễn dưới dạng phân tích riêng như sau:

$$\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}$$

Áp dụng hằng đẳng thức $\mathbf{trace}(\mathbf{AB}) = \mathbf{trace}(\mathbf{BA})$ ta có:

$$\mathbf{trace}(\mathbf{A}) = \mathbf{trace}(\mathbf{P}\mathbf{D}\mathbf{P}^{-1}) = \mathbf{trace}((\mathbf{P}\mathbf{D})\mathbf{P}^{-1}) = \mathbf{trace}(\mathbf{P}^{-1}\mathbf{P}\mathbf{D}) = \mathbf{trace}(\mathbf{D})$$

Từ đó suy ra tổng các phần tử trên đường chéo chính của ma trận $\mathbf{A}$ bằng tổng các trị riêng. 6. Đinh thức của ma trận $\mathbf{A} \in \mathbb{R}^{n\times n}$ thì bằng tích các trị riêng của nó. Sử dụng phép phân tích riêng đối với ma trận $\mathbf{A}$ độc lập tuyến tính ta có:

$$\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1} \Rightarrow \mathbf{det}(\mathbf{A}) = \mathbf{det}(\mathbf{P}\mathbf{D}\mathbf{P}^{-1}) = \mathbf{det}(\mathbf{P}).\mathbf{det}(\mathbf{D}).\mathbf{det}(\mathbf{P}^{-1}) = \mathbf{det}(\mathbf{D}) = \prod_{i=1}^{n}\lambda_i$$

17.1.4. Phép phân tích riêng (*EigenDecomposition*)

Phép phân tích riêng: Phép phân tích riêng (EigenDecomposition) là một dạng phân rã ma trận (*matrix factorization*). Nó có mối liên hệ bền chặt với SVD mà chúng ta sẽ tìm hiểu bên dưới. Phép phân tích riêng sẽ phân tích một ma trận vuông độc lập tuyến tính $\mathbf{A} \in \mathbb{R}^{n\times n}$ thành tích của ma trận vuông $\mathbf{P} \in \mathbb{R}^{n\times n}$ khả nghịch với ma trận đường chéo $\mathbf{D} \in \mathbb{R}^{n\times n}$ theo công thức:

$$\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}$$

Đẳng thức trên tương đương với:

$$\mathbf{A}\mathbf{P} = \mathbf{P}\mathbf{D}$$

Bây giờ ta chỉ xét đến cột thứ i của cả 2 ma trận bên vế trái và phải:

$$\mathbf{A}\mathbf{p}_i = \mathbf{P}\mathbf{d}_i$$

Trong đó $\mathbf{p}_i$, $\mathbf{d}_i$ lần lượt là cột thứ i của ma trận $\mathbf{P}$ và $\mathbf{D}$. Mặt khác do $\mathbf{D}$ là ma trận đường chéo nên $\mathbf{d}_i$ chỉ có duy nhất một phần tử khác 0 là d_{ii} nên $\mathbf{P}\mathbf{d}_i = d_{ii}\mathbf{p}_i$. Như vậy:

$$\mathbf{A}\mathbf{p}_i = d_{ii}\mathbf{p}_i$$

Ta có thể thấy d_{ii} chính là trị riêng của ma trận $\mathbf{A}$ và $\mathbf{p}_i$ là các vector riêng tương ứng của d_{ii}.

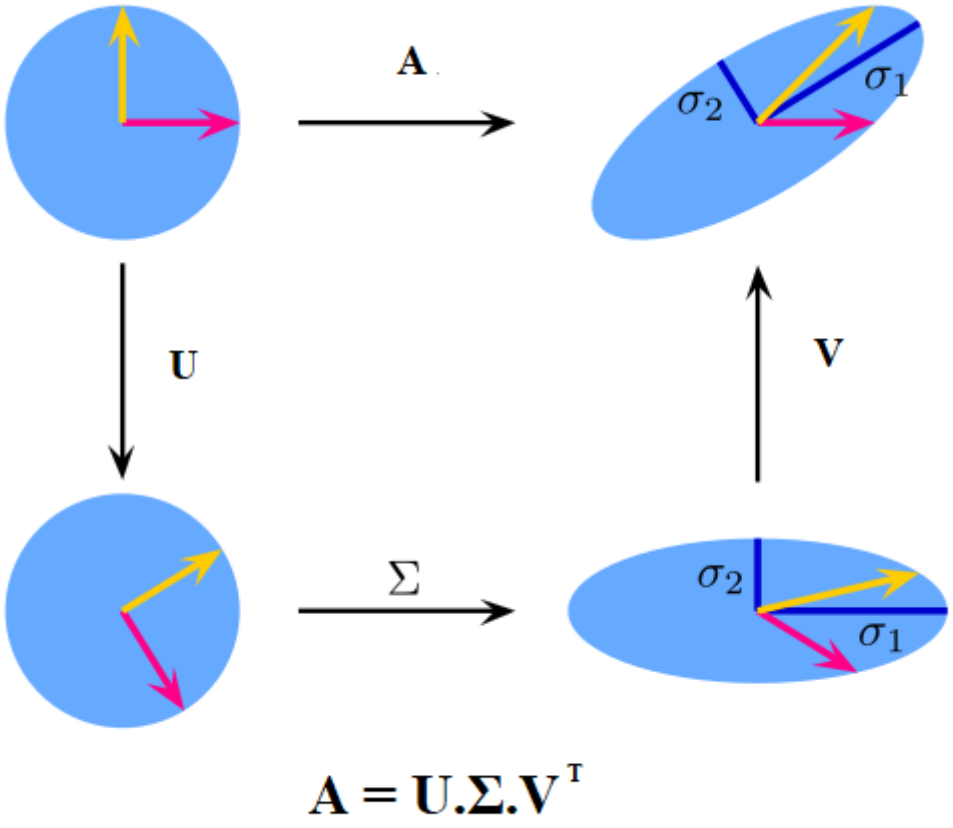
Như vậy điểm đặc biệt của phân tích riêng đó là đường chéo chính của $\mathbf{D}$ là các trị riêng của ma trận $\mathbf{A}$ và các cột của $\mathbf{P}$ là các vector riêng tương ứng với trị riêng nằm trên đường chéo chính. Ngoài ra phép phân tích riêng không là duy nhất. Nếu ma trận trực giao $\mathbf{P}$ thỏa mãn phương trình phân tích riêng thì ma trận $k\mathbf{P}$ cũng thỏa mãn phương trình phân tích riêng đó.

17.1.5. Phân tích suy biến (SVD)

Phép phân tích suy biến (*Singular Value Decomposition*) được viết tắt là `SVD` là một dạng `matrix factorization` khái quát hơn so với phép phân tích riêng đã trình bày ở trên. Theo đó phương pháp này cũng nhằm phân rã một ma trận thành tích của ba ma trận số thực hoặc số phức. Trong đó hai ma trận ở vị trí đầu tiên và cuối cùng là ma trận trực giao có thể không là khả nghịch của nhau và ở giữa là ma trận đường chéo có thể không vuông.

$$\mathbf{A}_{mn} = \mathbf{U}_{mm}\mathbf{\Sigma}_{mn}\mathbf{V}^{\intercal}_{nn}$$

Kích thước ma trận được để ở bên dưới chân ma trận, tức là ma trận $\mathbf{A}_{mn} \in \mathbb{R}^{m \times n}$. Trong công thức trên thì $\mathbf{U}_{mm}, \mathbf{V}_{nn}$ là các ma trận trực giao và $\mathbf{\Sigma}_{mn}$ là ma trận đường chéo.



Hình 3: Minh họa biến đổi hình học của phép phân tích suy biến. (Nguồn wikipedia)

Phép phân tích suy biến sẽ lần lượt trải qua:

- phép xoay (rotation): Từ một điểm dữ liệu $\mathbf{x} \in \mathbb{R}^m$ trong không gian gốc, chúng ta nhân với ma trận $\mathbf{U}_{mm}$ để thực hiện xoay trục. Tại phép biến đổi này không làm thay đổi tích vô hướng của hai điểm dữ liệu. Thật vậy, giả sử hai véc tơ $\mathbf{x}_i$ và $\mathbf{x}_j \in \mathbb{R}^m$. Phép xoay sẽ giúp biến đổi các véc tơ thành hai véc tơ mới là $\mathbf{y}_i = \mathbf{U}\mathbf{x}_i$ và $\mathbf{y}_j = \mathbf{U}\mathbf{x}_j$. Tích vô hướng của chúng:

$$\mathbf{y}_i^{\intercal}\mathbf{y}_j = \mathbf{x}_i^{\intercal}\mathbf{U}^{\intercal}\mathbf{U}\mathbf{x}_j = \mathbf{x}_i^{\intercal}\mathbf{I}_n\mathbf{x}_j = \mathbf{x}_i^{\intercal}\mathbf{x}_j$$

- phép nới rộng (scaling): Kết quả thu được sau phép xoay tiếp tục được nhân với ma trận đường chéo. Khi đó mỗi một chiều của dữ liệu sẽ được scale lên số lần chính bằng giá trị của trị riêng nằm trên đường chéo chính.
- Tiếp tục lại là một phép xoay: Sau phép nới rộng thì dữ liệu đã được chuyển về không gian thấp chiều. Ta lại tiếp tục thực hiện một phép xoay bằng cách nhân với ma trận $\mathbf{V}_{nn}$

17.1.6. Mối liên hệ giữa phân tích suy biến và phân tích riêng.

Ta có thể thấy phân tích suy biến khác với phân tích riêng ở chỗ nó áp dụng cho ma trận bất kì mà không yêu cầu ma trận đó phải vuông. Để thấy được mối liên hệ mật thiết giữa phép phân tích suy biến và phân tích riêng sau khi khai triển tích $\mathbf{A}^{\intercal}\mathbf{A}$:

$$\begin{aligned}\mathbf{A}^\intercal\mathbf{A} &= (\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\intercal)^\intercal\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\intercal \\ &= \mathbf{V}\mathbf{\Sigma}^\intercal\underbrace{\mathbf{U}^\intercal\mathbf{U}}_{\mathbf{I}_n}\mathbf{\Sigma}\mathbf{V}^\intercal \\ &= \mathbf{V}\mathbf{\Sigma}^\intercal\mathbf{\Sigma}\mathbf{V}^\intercal\end{aligned}$$

Như vậy $\mathbf{A}^\intercal\mathbf{A}$ là một phân tích riêng của ma trận trực giao $\mathbf{V}$ và ma trận đường chéo $\mathbf{\Sigma}^\intercal\mathbf{\Sigma}$. Hoàn toàn tương tự ta cũng có $\mathbf{A}\mathbf{A}^\intercal$ là một phân tích riêng của ma trận trực giao $\mathbf{U}$ và ma trận đường chéo $\mathbf{\Sigma}\mathbf{\Sigma}^\intercal$. Ngoài ra $\mathbf{\Sigma}^\intercal\mathbf{\Sigma}$ là một ma trận đường chéo có các thành phần trên đường chéo chính lần lượt là $\sigma_1^2, \sigma_2^2, \ldots, \sigma_n^2 > 0$. Các phần tử σ_i trên đường chéo chính của $\mathbf{\Sigma}$ được gọi là giá trị suy biến (*singular values*) của ma trận $\mathbf{A}$. Các cột của ma trận $\mathbf{V}$ là hệ vector riêng ứng với trị riêng của $\mathbf{\Sigma}^\intercal\mathbf{\Sigma}$ và còn được gọi là hệ vector suy biến trái (left-singular vectors). Các cột của $\mathbf{U}$ được gọi là hệ vector suy biến phải (*right-singular vectors*).

Một điều khá may mắn đó là trong python chúng ta có thể tính toán các ma trận $\mathbf{U}, \mathbf{V}$ và $\mathbf{\Sigma}$ trong phép phân tích suy biến của một ma trận bất kỳ thông qua module `scipy.linalg` chỉ với một vài dòng code.

```
import scipy.linalg as ln
import numpy as np
m, n = 2, 3
n_diag = min(m, n)
#Init normal standard random variable A with size (m, n)
A = np.random.rand(m, n)
U, S_diag, V = ln.svd(A)
#Create diagonal matrix S based on diagonal
S = np.zeros((n_diag, n_diag))
np.fill_diagonal(S, S_diag)
if m > n:
    S = np.concatenate((S, np.zeros((1, n))), axis = 0)
elif m < n:
    S = np.concatenate((S, np.zeros((m, 1))), axis = 1)

print('Matrix A: \n %s \n'%A)
print('orthogonal matrix U: \n %s \n'%U)
print('Check Frobenius U^TU-I: \n %s \n'%ln.norm(np.dot(U.T,U)-np.eye(m, m), 'fro'))
print('orthogonal matrix V: \n %s \n'%V)
print('Check Frobenius V^TV-I: \n %s \n'%ln.norm(np.dot(V.T,V)-np.eye(n, n), 'fro'))
print('Diagonal matrix S: \n %s \n'%S_diag)
print('Matrix S: \n %s \n'%S)
print('Check Frobenius U.S.V - A: \n %s \n'%ln.norm(np.dot(U, S.dot(V))-A,'fro'))
```

```
Matrix A:
 [[0.15141772 0.33360862 0.92847382]
 [0.20496416 0.50481926 0.93969943]]

orthogonal matrix U:
 [[-0.67626611  0.73665741]
 [-0.73665741 -0.67626611]]

Check Frobenius U^TU-I:
 3.141612046725681e-16

orthogonal matrix V:
 [[-0.17225085 -0.4061678  -0.89741705]
 [-0.24476148 -0.8648147   0.43839178]
 [-0.95416008  0.29516648  0.04955084]]

Check Frobenius V^TV-I:
 1.2243466275233446e-15

Diagonal matrix S:
 [1.47103504 0.11058657]

Matrix S:
 [[1.47103504 0.         0.        ]
 [0.         0.11058657 0.        ]]

Check Frobenius U.S.V - A:
 9.686656125337893e-16
```

17.2. Các dạng phân tích suy biến SVD

Thông thường việc phân tích suy biến một ma trận có kích thước lớn sẽ rất lâu vì trước tiên ta phải giải phương trình đặc trưng để tìm ra các giá trị đặc trưng, từ đó suy ra ma trận đường chéo $\mathbf{\Sigma}$. Tiếp theo để tìm ra ma trận trực giao $\mathbf{V}$ ta xuất phát từ phương trình $\mathbf{A}^\intercal\mathbf{A} = \mathbf{V}\mathbf{\Sigma}^\intercal\mathbf{\Sigma}\mathbf{V}^\intercal$ để suy ra

$$\mathbf{A}^\intercal\mathbf{A}\mathbf{V} = \mathbf{V}\mathbf{\Sigma}^\intercal\mathbf{\Sigma}\underbrace{\mathbf{V}^\intercal\mathbf{V}}_{\mathbf{I}_n} = \mathbf{V}\mathbf{\Sigma}^\intercal\mathbf{\Sigma}$$

Như vậy đối với một cột $\mathbf{v}_i$ bất kỳ của ma trận $\mathbf{V}$ ta có:

$$\mathbf{A}^\intercal\mathbf{A}\mathbf{v}_i = \sigma_i^2\mathbf{v}_i$$

Hay nói cách khác các cột $\mathbf{v}_i$ chính là một véc tơ riêng ứng với trị riêng σ_i^2 của ma trận $\mathbf{A}^\intercal\mathbf{A}$.

Cách tìm ma trận $\mathbf{U}$ cũng được suy ra tương tự từ phương trình phân tích riêng $\mathbf{A}\mathbf{A}^\intercal = \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}^\intercal\mathbf{U}^\intercal$. Quá trình này phải trải qua nhiều bước và khi kích thước ma trận lớn, chi phí thời gian và lưu trữ sẽ rất lớn. Vì vậy các dạng giảm chiều SVD sẽ có tác dụng rút gọn quá trình tính toán.

17.2.1. Phương pháp làm mỏng SVD

Hình 4: Phương pháp làm mỏng SVD. Ma trận suy biến trái đã được giảm từ $m \times m$ về còn $m \times n$. Ma trận đường chéo giảm từ $m \times n$ về $n \times n$.

Xuất phát từ ý tưởng số quan sát thường lớn gấp rất nhiều lần so với số chiều hay $m >> n$ trong hầu hết các trường hợp của ma trận $\mathbf{A}$ nên thay vì phải tính toán bộ ma trận $\mathbf{U}_{mm}$ ta sẽ chỉ tính n cột đầu tiên là $\mathbf{U}_{mn}$. Số chiều của ma trận đường chéo $\mathbf{\Sigma}_{mn}$ cũng giảm xuống thành $\mathbf{\Sigma}_{nn}$. Khi đó ma trận $\mathbf{A}$ được biểu diễn dưới dạng:

$$\mathbf{A} = \mathbf{U}_{mn}\mathbf{\Sigma}_{nn}\mathbf{V}_{nn}^\intercal$$

Như vậy số lượng các trị riêng cần tìm chỉ còn n và số lượng vector riêng chỉ còn $2n$ (n cột của ma trận $\mathbf{U}_{mn}$ và n cột của ma trận $\mathbf{V}_{nn}$).

17.2.2 . Phương pháp Compact SVD.

Ta có thể biểu diễn ma trận $\mathbf{A}$ dưới dạng tổng của tích vô hướng giữa vector cột $\mathbf{u}_i \in \mathbb{R}^m$ của $\mathbf{U}_{mm}$ và vector dòng $\mathbf{v}_i \in \mathbb{R}^n$ của $\mathbf{V}_{nn}^\intercal$ như sau:

$$\mathbf{A} = \sum_{i=1}^{n}\mathbf{u}_i\sigma_i\mathbf{v}_i \tag{4}$$

Các vector $\mathbf{u}_i$ và $\mathbf{v}_i$ là các hệ cơ sở độc lập tuyến tính. Thông thường trong ma trận đường chéo $\mathbf{\Sigma}_{nn}$ chỉ một lượng lớn các trị riêng có lớn hơn 0. Các trị riêng còn lại đều xấp xỉ 0. Do đó chỉ tại r vị trí dòng và cột tương ứng với các trị riêng đủ lớn ta mới thực hiện tính toán SVD. Biểu diễn ma trận $\mathbf{A}_{mn}$ dưới dạng compact SVD như sau:

$$\mathbf{A} = \mathbf{U}_r\mathbf{\Sigma}_r\mathbf{V}_r^\intercal$$

Trong đó các ma trận $\mathbf{U_r}, \mathbf{\Sigma_r}, \mathbf{V_r}^\mathbf{T}$ lần lượt là các ma trận sau khi đã rút gọn các dòng và cột để chỉ giữ lại các vị trí tương ứng với σ_i đủ lớn. Nếu $r << n$ thì cách tính này tiết kiệm được nhiều số lượng tính toán và lưu trữ hơn so với phương pháp làm mỏng SVD.

17.2.3. Phương pháp Truncate SVD.

Trong phương pháp này ta giả định ma trận $\mathbf{A}$ là ma trận bán xác định dương. Khi đó mọi trị riêng của nó đều không âm. Phương pháp Truncate SVD cũng tương tự như Compact SVD. Tuy nhiên thay vì các dòng và cột tương ứng với trị riêng đủ lớn trên ma trận đường chéo $\mathbf{\Sigma}_{mn}$ thì chúng ta sẽ chỉ lấy ra t dòng và cột ứng với top t các trị riêng $\sigma_1 > \sigma_2 > \ldots > \sigma_t > 0$ lớn nhất của $\mathbf{A}$ từ $\mathbf{U}, \mathbf{V}^\intercal$. Phần còn lại của ma trận sẽ bị loại bỏ. Như vậy trong phương pháp Truncate SVD ta sẽ thu được ma trận xấp xỉ của ma trận $\mathbf{A}$ là ma trận:

$$\hat{\mathbf{A}} = \mathbf{U}_t\mathbf{\Sigma}_t\mathbf{V}_t^\intercal$$

Hoặc ta có thể biểu diễn dưới dạng tổng của tích vô hướng các vector cột và dòng của $\mathbf{U}, \mathbf{V}^\intercal$ như sau:

$$\mathbf{A} = \sum_{i=1}^{t} \mathbf{u}_i \sigma_i \mathbf{v}_i \tag{5}$$

Khi đó ta còn tính được khoảng cách norm Frobenius giữa $\hat{\mathbf{A}}$ và $\mathbf{A}$ chính bằng tổng bình phương của các trị riêng còn lại từ σ_{t+1} đến σ_n (với giả định $\mathbf{A}$ có n trị riêng) như sau:

$$\begin{aligned}
||\mathbf{A} - \hat{\mathbf{A}}||_F^2 &= (\mathbf{A} - \hat{\mathbf{A}})^\intercal (\mathbf{A} - \hat{\mathbf{A}}) \\
&= (\sum_{i=t+1}^{n} \mathbf{u_i} \sigma_\mathbf{i} \mathbf{v_i})^\intercal (\sum_{i=t+1}^{n} \mathbf{u_i} \sigma_\mathbf{i} \mathbf{v_i}) \\
&= (\sum_{i=t+1}^{n} \sigma_\mathbf{i} \mathbf{v_i}^\intercal \mathbf{u_i}^\intercal)(\sum_{i=t+1}^{n} \sigma_\mathbf{i} \mathbf{u_i} \mathbf{v_i}) \\
&= \sum_{i=t+1}^{n} \sum_{j=t+1}^{n} \sigma_\mathbf{i} \sigma_\mathbf{j} \mathbf{v_i}^\intercal \mathbf{u_i}^\intercal \mathbf{u_j} \mathbf{v_j} \\
&= \sum_{i=t+1}^{n} \sigma_\mathbf{i}^\mathbf{2} \mathbf{v_i}^\intercal \mathbf{u_i}^\intercal \mathbf{u_i} \mathbf{v_i} \\
&= \sum_{i=t+1}^{n} \sigma_\mathbf{i}^\mathbf{2} \mathbf{v_i}^\intercal \mathbf{v_i} \\
&= \sum_{i=t+1}^{n} \sigma_\mathbf{i}^\mathbf{2}
\end{aligned}$$

Dấu bằng thứ 3 là vì lấy đẳng thức (4) trừ đi (5). Dấu bằng thứ 5 xảy ra là do ma các vector cột của $\mathbf{U, V}$ là những hệ trực giao nên $\mathbf{u_i}^\intercal \mathbf{u_j} = 0, \mathbf{v_i}^\intercal \mathbf{v_j} = 0,\ \forall 1 \leq i \neq j \leq n$. Các dấu bằng 6, 7 là do $\mathbf{u_i}^\intercal \mathbf{u_i} = 1, \mathbf{v_i}^\intercal \mathbf{v_i} = 1,\ \forall 1 \leq i \leq n$. Như vậy ta đã hiểu lý do tại sao chúng ta chỉ chọn ra top t trị riêng có giá trị lớn nhất. Khi đó sai số của 2 ma trận sẽ là nhỏ nhất vì bằng tổng bình phương của các trị riêng còn lại. Phương pháp truncate SVD còn cho ta biết được tỷ lệ phần trăm lượng thông tin lưu giữ trong ma trận xấp xỉ thông qua công thức:

$$\frac{\sum_{i=1}^{t} \sigma_i^2}{\sum_{j=1}^{n} \sigma_j^2}$$

17.2.4. Thuật toán PCA

Trong thuật toán PCA chúng ta mong muốn giảm chiều dữ liệu sao cho thông tin về độ biến động của dữ liệu trong ma trận được bảo toàn.

Bạn còn nhớ về ma trận hiệp phương sai của ma trận dữ liệu đầu vào $\mathbf{X} \in \mathbb{R}^{n \times d}$ được trình bày ở mục 2. Xin nhắc lại đây là một ma trận có tính chất vuông, đối xứng và bán xác định dương có công thức như sau:

$$\mathbf{A} \triangleq \text{cov}(\mathbf{X}, \mathbf{X}) = (\mathbf{X} - \bar{\mathbf{X}})^\intercal (\mathbf{X} - \bar{\mathbf{X}})$$

Trong đó ma trận $\bar{\mathbf{X}}$ có các dòng là bằng nhau và bằng giá trị của điểm trung bình của toàn bộ ma trận $\mathbf{X}$. Điểm này đại diện cho toàn bộ các quan sát trong ma trận $\mathbf{X}$.

Thuật toán PCA sẽ dựa trên phép phân tích suy biến `SVD` để tìm ra một ma trận sấp xỉ với ma trận hiệp phương sai $\mathbf{A}$ bằng phương pháp `truncate SVD`.Điều đó có nghĩa rằng chúng ta sẽ tìm ra các ma trận $\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}$ sao cho:

$$\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^\intercal$$

và sau đó lọc ra t trị riêng lớn nhất của $\mathbf{\Sigma}$ để giảm chiều dữ liệu. Khi đó ma trận vuông $\mathbf{U}$ kích thước $n \times n$ sẽ tiêu giảm xuống thành ma trận thấp chiều kích thước $n \times t$ với $n >> t$.

Các điểm dữ liệu từ ma trận $\mathbf{X}$ sẽ được chiếu sang hệ cơ sở mới dựa trên phép nhân với ma trận $\mathbf{U}_{nt}$ theo công thức:

$$\mathbf{y} = \mathbf{U}_{nt} \mathbf{x}$$

17.4. Ví dụ về PCA trên sklearn

Có khá nhiều các phương pháp giảm chiều dữ liệu khác nhau được hỗ trợ trên sklearn như `PCA, LDA, Dictionary Learning, ...` . Bạn có thể theo dõi những thuật toán này tại sklearn-decomposition. Để sử dụng PCA trên sklearn chúng ta có thể sử dụng module sklearn.decomposition.PCA.

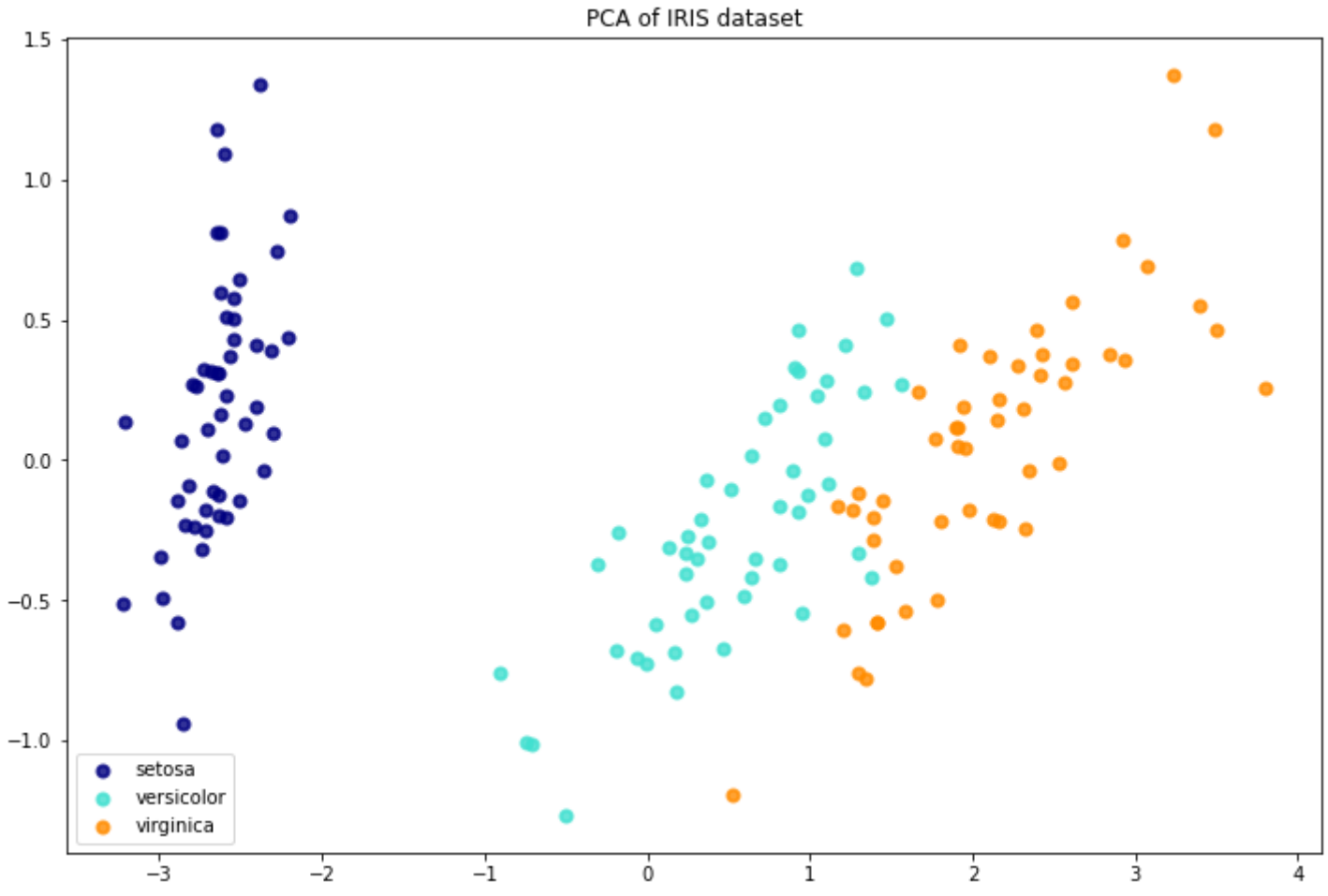
```
PCA(n_components=None,
  *, copy=True,
  whiten=False,
  svd_solver='auto',
  tol=0.0,
  iterated_power='auto',
  random_state=None)
```

Trong đó `n_components` là số lượng chiều mà chúng ta sẽ chiếu trong không gian giảm chiều. `svd_solver` là phương pháp SVD sử dụng để phân tích suy biến ma trận hiệp phương sai. Thông thường ta chỉ cần quan tâm và điền số chiều `n_components` là được. Số chiều này phải nhỏ hơn số lượng quan sát và số lượng `max_features`.

Tiếp theo trong ví dụ mẫu chúng ta sẽ áp dụng sklearn vào để giảm chiều bộ dữ liệu `iris_dataset`. Bộ dữ liệu này bao gồm 4 quan sát là độ dài và rộng cánh hoa (`Pental.Length, Pental.Width`), độ dài và rộng đài hoa (`Sepal.Length, Sepal.Width`) thuộc về ba giống hoa là `sentosa, virginica,` .

Chúng ta sẽ giảm chiều bộ dữ liệu này từ 4 chiều về 2 chiều và tiến hành visualize để kiểm tra yếu tố phân cụm của các giống hoa.

```
from sklearn import datasets
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

iris = datasets.load_iris()

X = iris.data
y = iris.target
target_names = iris.target_names

pca = PCA(n_components=2)
X_r = pca.fit(X).transform(X)

plt.figure(figsize=(12, 8))
colors = ['navy', 'turquoise', 'darkorange']
lw = 2

for color, i, target_name in zip(colors, [0, 1, 2], target_names):
    plt.scatter(X_r[y == i, 0], X_r[y == i, 1], color=color, alpha=.8, lw=lw,
                label=target_name)
plt.legend(loc='best', shadow=False, scatterpoints=1)
plt.title('PCA of IRIS dataset')

plt.show()
```



Chúng ta có thể thấy PCA đã giảm chiều dữ liệu từ 4 chiều xuống 2 chiều. Khi visualize những lớp trên không gian hai chiều thì chúng ta thấy có sự phân biệt giữa các cụm theo loài hoa. Điều đó cho thấy PCA đã bảo toàn được thông tin của dữ liệu đầu vào rất tốt trên không gian thấp chiều.

17.5. Tổng kết

Như vậy qua bài này chúng ta đã cùng nhau tìm hiểu về phương pháp giảm chiều dữ liệu PCA. Đây là một phương pháp giảm chiều dữ liệu tuyến tính khá hiệu quả dựa trên phương pháp phân tích suy biến SVD ma trận hiệp phương sai. Các biến trong không gian giảm chiều sẽ là tổ hợp tuyến tính của các biến đầu vào gốc nhân với hệ véc tơ cơ sở của ma trận suy biến trái. Như vậy sử dụng PCA sẽ không cần phải bỏ sót biến và đồng thời giữ được thông tin về độ biến động của dữ liệu.

Ngoài ra chúng ta cũng được làm quen với các khái niệm về phép phân tích riêng, phép phân tích suy biến, hệ trực giao, hệ trực chuẩn và các đại lượng như véc tơ riêng tương ứng với trị riêng trong đại số tuyến tính. Đây là những kiến thức tương đối quan trọng và được ứng dụng nhiều trong đại số tuyến tính mà bạn đọc cần ghi nhớ.

17.6. Bài tập

Lựa chọn một trong các bộ dữ liệu sau:

- BreastCancer về chuẩn đoán ung thư vú.
- diabetes chuẩn đoán bệnh tiểu đường.
- hmeq phân loại hồ sơ cho vay mua nhà.
- BonstonHousing dự báo giá nhà ở Bonston.
- churn customer dự đoán khách hàng rời bỏ.

Hãy thực hiện các bài tập sau:

1. Giảm chiều dữ liệu của bộ dữ liệu về còn 2 chiều.
2. Thực hiện visualization các nhóm dữ liệu theo 2 chiều và nhận xét.
3. Phát hiện các điểm outliers từ đồ thị visualize.

Tiếp theo, sử dụng bộ dữ liệu movielen hãy thực hiện:

1. phân tích suy biến SVD đối với ma trận rating.
2. Tìm biểu diễn của các bộ phim trong không gian thấp chiều.
3. Sử dụng cosine-similarity trên không gian thấp chiều, hãy xây dựng thuật toán khuyến nghị những bộ phim tương đồng dựa trên bộ phim mà tác giả đã xem.

17.7. Tài liệu

1. https://builtin.com/data-science/step-step-explanation-principal-component-analysis
2. https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c
3. https://cs229.stanford.edu/notes2021fall/cs229-notes10.pdf
4. https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202
5. https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/principal-component-analysis-pca/
6. https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html
7. https://scikit-learn.org/stable/auto_examples/decomposition/plot_faces_decomposition.html#sphx-glr-auto-examples-decomposition-plot-faces-decomposition-py

Previous
17. Giảm chiều dữ liệu

Next
Tích phân Riemann và định lý Fubini

By Pham Dinh Khanh