CSE: **Faculty of Computer Science and Engineering**
Thuyloi University

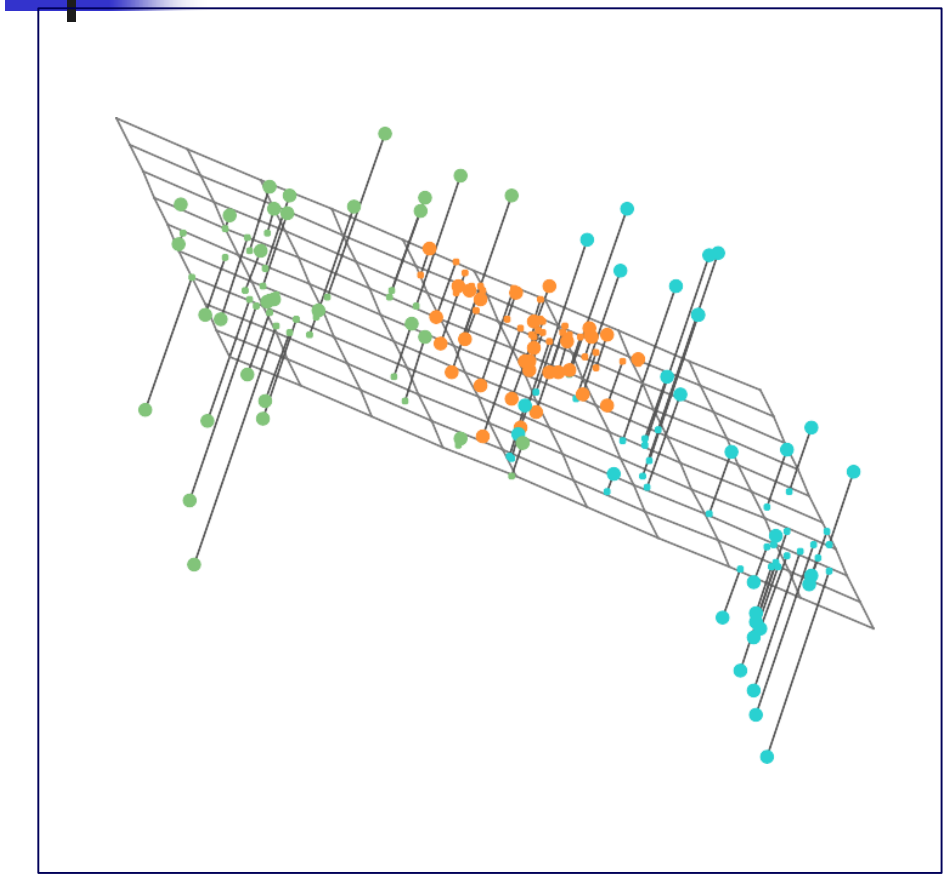
Phân tích thành phần chính

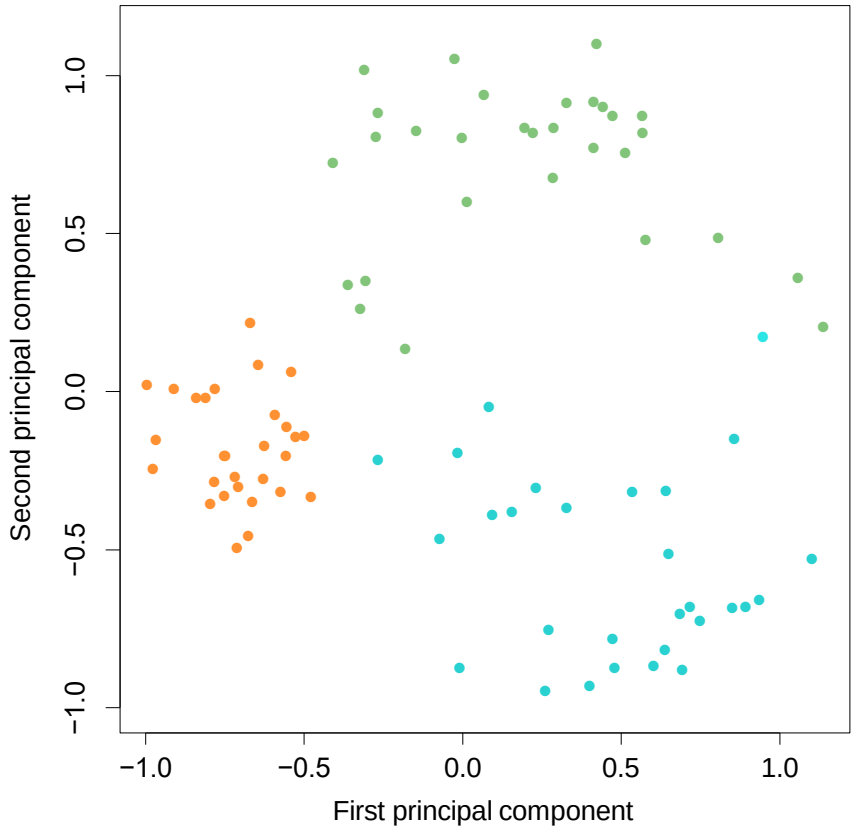
(Principal component analysis, PCA)

Giới thiệu

Dimensionality Reduction (giảm chiều dữ liệu) là một trong những kỹ thuật quan trọng trong Machine Learning. Các feature vectors trong các bài toán thực tế có thể có số chiều rất lớn, tới vài nghìn. Ngoài ra, số lượng các điểm dữ liệu cũng thường rất lớn. Nếu thực hiện lưu trữ và tính toán trực tiếp trên dữ liệu có số chiều cao này thì sẽ gặp khó khăn cả về việc lưu trữ và tốc độ tính toán. Vì vậy, giảm số chiều dữ liệu là một bước quan trọng trong nhiều bài toán. Đây cũng được coi là một phương pháp nén dữ liệu.

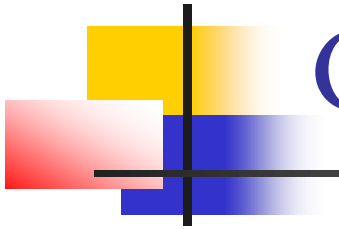
Giới thiệu




Second principal component
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0
First principal component

Giới thiệu

- Bài toán giảm chiều: tìm hàm số sao cho
  - ✓ Input: $\mathbf{x} \in \mathbb{R}^D$ với D rất lớn
  - ✓ Output: $\mathbf{z} \in \mathbb{R}^K$ $với i$ $K < D$



Giới thiệu

- Phương pháp *Principal Component Analysis* (PCA):
  - ✓ Dựa trên quan sát rằng dữ liệu thường không phân bố ngẫu nhiên trong không gian mà thường phân bố gần các đường/mặt đặc biệt nào đó.
  - ✓ PCA xem xét một trường hợp đặc biệt khi các mặt đặc biệt đó có dạng tuyến tính là các không gian con (subspace).

Nhắc lại một số kiến thức toán học liên quan

Norm 2 của ma trận

Chúng ta sẽ làm quen với 1 lớp các norm cho ma trận được định nghĩa dựa trên norm của vector. Lớp các norms này còn được gọi là *Induced Norms*

- Giả sử hàm số $||x||_\alpha$ là một norm bất kỳ của vector x. Ứng với norm này, định nghĩa norm tương ứng cho ma trân A:

$$||\mathbf{A}||_\alpha = \max_{\mathbf{x}} \frac{||\mathbf{Ax}||_\alpha}{||\mathbf{x}||_\alpha}$$

Chú ý:

- Ma trận A có thể không vuông và số cột của nó bằng với số chiều của x
- Bản thân việc tính toán norm của ma trận là việc giải một bài toán tối ưu
- Hàm tối ưu có cả tử số và mẫu số là các norm trên vectors

Nhắc lại một số kiến thức toán học liên quan

- Norm 2 của ma trận A:

$$||\mathbf{A}||_2 = \max_{\mathbf{x}} \frac{||\mathbf{Ax}||_2}{||\mathbf{x}||_2} \quad (1)$$

  - Nếu x là nghiệm của bài toán tối ưu (1) thì kx cũng là nghiệm với k là một số thực khác không bất kỳ.
  - Không mất tính tổng quát, ta có thể giả sử mẫu số bằng 1. Khi đó, bài toán tối ưu (1) có thể được viết dưới dạng:

$$||\mathbf{A}||_2 = \max_{||\mathbf{x}||_2=1} ||\mathbf{Ax}||_2 \quad (2)$$

Nhắc lại một số kiến thức toán học liên quan

- Nói cách khác, ta cần đi tìm x sao cho:

$$\mathbf{x} = \text{argmax}_{\mathbf{x}} ||\mathbf{Ax}||_2^2$$
$$\text{s.t.:} \quad ||\mathbf{x}||_2^2 = 1 \qquad (3)$$

  - Bài toán (3) có thể được giải bằng Phương pháp nhân tử Lagrange vì ràng buộc là một phương trình.

- Lagrangian của Bài toán (3) là:

$$\mathcal{L}(\mathbf{x}, \lambda) = ||\mathbf{Ax}||_2^2 + \lambda(1 - ||\mathbf{x}||_2^2)$$

Nhắc lại một số kiến thức toán học liên quan

- Nghiệm của bài toán (3) sẽ thoả mãn hệ phương trình:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}} = 2\mathbf{A}^T\mathbf{A}\mathbf{x} - 2\lambda\mathbf{x} = \mathbf{0} \quad (4)$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = 1 - ||\mathbf{x}||_2^2 = 0 \quad (5)$$

- Từ (4) ta có:

$$\mathbf{A}^T\mathbf{A}\mathbf{x} = \lambda\mathbf{x} \quad (6)$$

- Điều này suy ra rằng λ là một trị riêng của A^TA và x là 1 vector riêng ứng với trị riêng đó. Tiếp tục nhân hai vế của (6) với x^T vào bên trái, ta có:

$$\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} = \lambda\mathbf{x}^T\mathbf{x} = \lambda$$

Nhắc lại một số kiến thức toán học liên quan

$$\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} = \lambda\mathbf{x}^T\mathbf{x} = \lambda$$

- Vế trái chính là $||Ax||_2^2$ chính là hàm mục tiêu trong (3). Vậy hàm mục tiêu đạt giá trị lớn nhất khi λ đạt giá trị lớn nhất. Nói cách khác, λ chính là trị riêng lớn nhất của $\mathrm{A}^T\mathrm{A}$

Biểu diễn vector trong các hệ cơ sở khác nhau

- Trong không gian D chiều , toạ độ của mỗi điểm được xác định dựa trên một hệ toạ độ nào đó. Ở các hệ toạ độ khác nhau, toạ độ của mỗi điểm cũng khác nhau.
- Tập hợp các vector $e_1, \dots, e_D$ mà mỗi vector e_d có đúng 1 phần tử bằng 1 ở thành phần thứ d và các phần tử còn lại bằng 0, được gọi là hệ cơ sở đơn vị (hoặc hệ đơn vị) trong không gian D chiều.
- Nếu xếp các vector $e_d, d = 1,2, \dots, D$ theo đúng thứ tự đó, ta sẽ được ma trận đơn vị D chiều.
- Mỗi vector cột $\mathbf{x} = [x_1, x_2, \dots, x_D] \in \mathbb{R}^D$, biểu diễn của nó trong hệ đơn vị là:

$$\mathbf{x} = x_1\mathbf{e}_1 + x_2\mathbf{e}_2 + \cdots + x_D\mathbf{e}_D$$

Biểu diễn vector trong các hệ cơ sở khác nhau

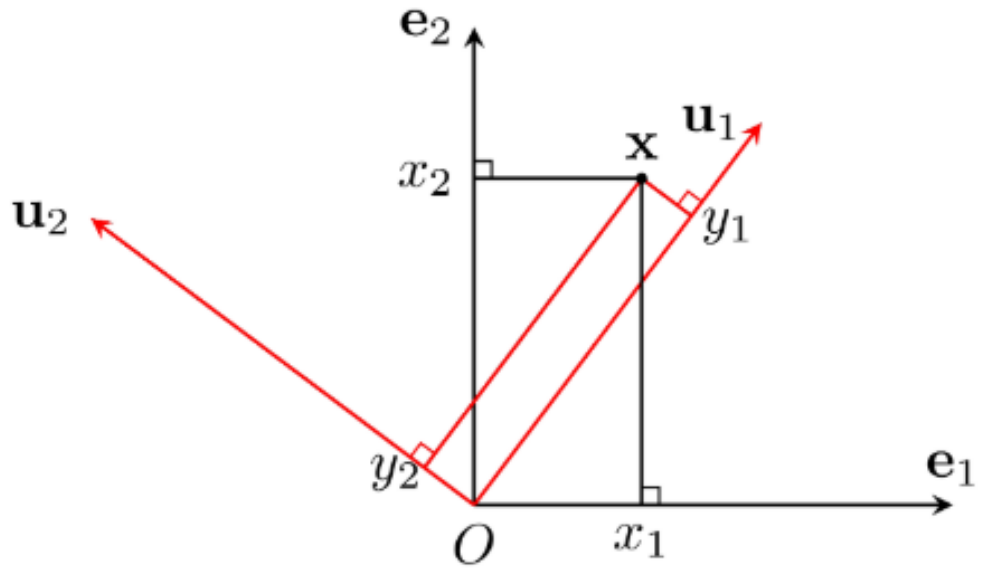
- Giả sử có một hệ cơ sở khác $u_1, u_2, \ldots, u_D$ (các vector này độc lập tuyến tính), vậy thì biểu diễn của vector x trong hệ cơ sở mới này có dạng:

$$\mathbf{x} = y_1\mathbf{u}_1 + y_2\mathbf{u}_2 + \cdots + y_D\mathbf{u}_D = \mathbf{U}\mathbf{y}$$

- U là ma trận mà cột thứ d của nó chính là vector u_d. Lúc này, vector y chính là biểu diễn của x trong hệ cơ sở mới.
- Ví dụ về việc chuyển đổi toạ độ trong các hệ cơ sở khác nhau:

✓ Trong hệ trục Oe_1e_2: $X=(x_1, x_2)$

✓ Trong hệ trục Ou_1u_2: $X=(y_1, y_2)$



Hàm số Trace

- Hàm số trace xác định trên tập các ma trận vuông được sử dụng rất nhiều trong tối ưu vì những tính chất đẹp của nó.
- Hàm trace trả về tổng các phần tử trên đường chéo của một ma trận vuông.
- Các tính chất quan trọng của hàm trace
  - $\text{trace}(\mathbf{A}) = \text{trace}(\mathbf{A}^T)$
  - $\text{trace}(k\mathbf{A}) = k\text{trace}(\mathbf{A})$ với k là một số bất kỳ.
  - $\text{trace}(\mathbf{AB}) = \text{trace}(\mathbf{BA})$

Hàm số Trace

- $||\mathbf{A}||_F^2 = \text{trace}(\mathbf{A}^T\mathbf{A}) = \text{trace}(\mathbf{A}\mathbf{A}^T)$ với $\mathbf{A}$ là ma trận bất kỳ, có thể không vuông.
- $\text{trace}(\mathbf{A}) = \sum_{i=1}^{D} \lambda_i$ với $\mathbf{A}$ là một ma trận vuông và $\lambda_i, i = 1, 2, \ldots, N$ là toàn bộ các trị riêng của nó, có thể phức hoặc lặp. Việc chứng minh tính chất này có thể được dựa trên ma trận đặc trưng của $\mathbf{A}$ và định lý Viète. Tôi xin được bỏ qua.

Kỳ vọng và ma trận hiệp phương sai

Dữ liệu một chiều

- Cho N giá trị $x_1, x_2, \ldots, x_N$. *Kỳ vọng* và *phương sai* của bộ dữ liệu này được định nghĩa là:

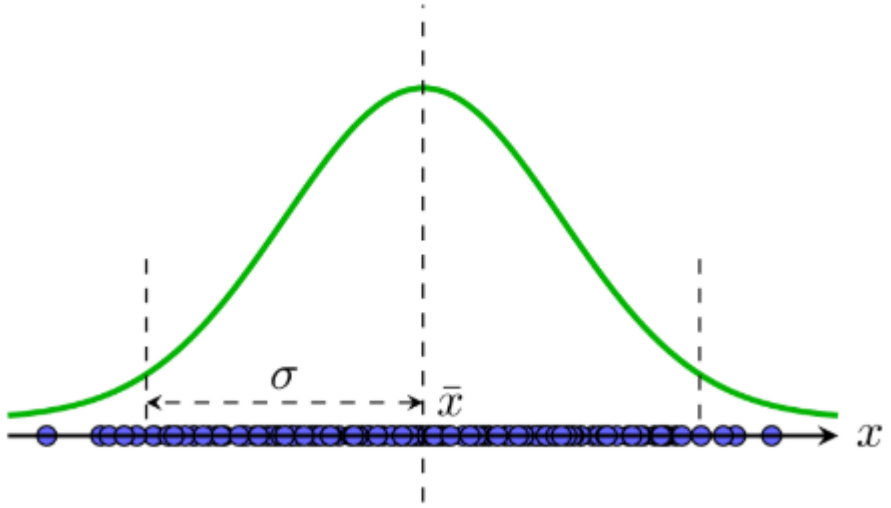
$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x_n = \frac{1}{N}\mathbf{X1}$$

$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})^2$$

Căn bậc hai của phương sai, σ được gọi là độ lệch chuẩn (standard deviation) của dữ liệu.

Kỳ vọng và ma trận hiệp phương sai

- Phương sai càng nhỏ thì các điểm dữ liệu càng gần với kỳ vọng, tức các điểm dữ liệu càng giống nhau. Phương sai càng lớn thì ta nói dữ liệu càng có tính phân tán



Kỳ vọng và ma trận hiệp phương sai

Dữ liệu nhiều chiều

- Cho *N* điểm dữ liệu được biểu diễn bởi các vector cột $x_1, \ldots, x_N$, khi đó, *vector kỳ vọng* và *ma trận hiệp phương sai* của toàn bộ dữ liệu được định nghĩa là:

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T = \frac{1}{N}\hat{\mathbf{X}}\hat{\mathbf{X}}^T$$

- Trong đó $\hat{X}$ được tạo bằng cách trừ mỗi cột của *X* đi $\bar{x}$:

$$\hat{\mathbf{x}}_n = \mathbf{x}_n - \bar{\mathbf{x}}$$

Kỳ vọng và ma trận hiệp phương sai

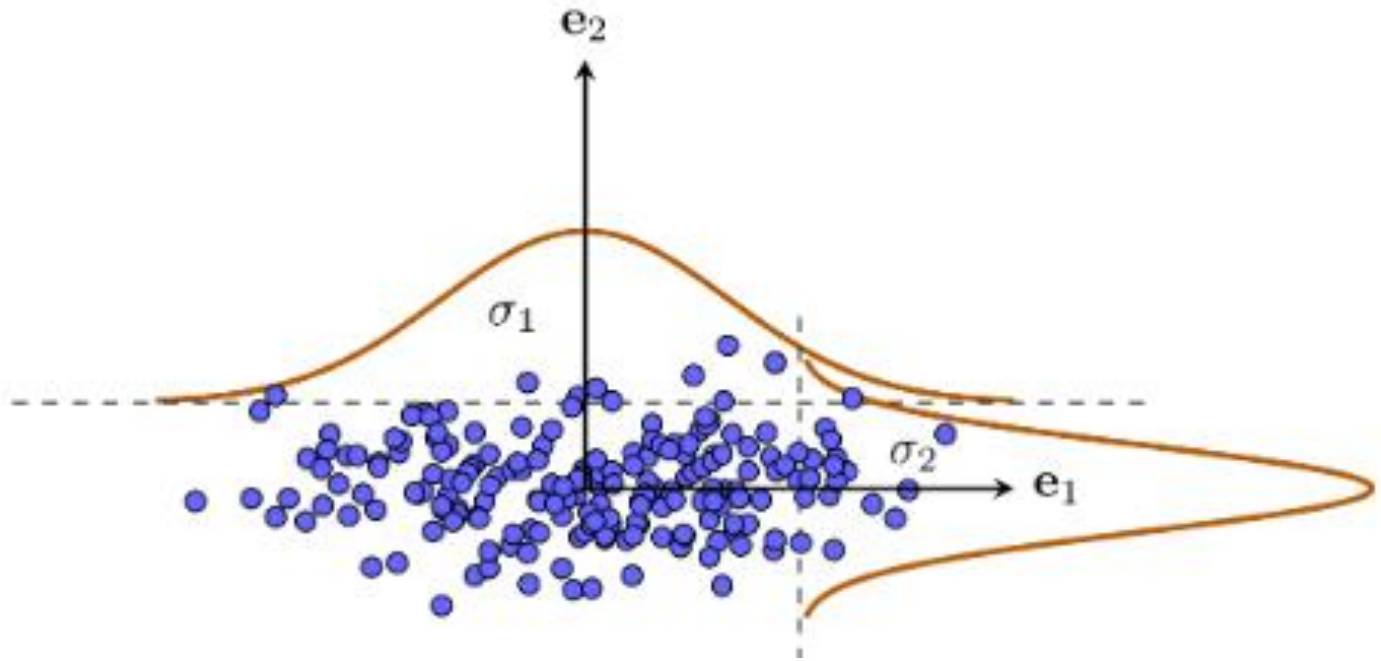
Có một vài điểm lưu ý về ma trận hiệp phương sai:

- Ma trận hiệp phương sai là một ma trận đối xứng, hơn nữa, nó là một ma trận nửa xác định dương.
- Mọi phần tử trên đường chéo của ma trận hiệp phương sai là các số không âm. Chúng cũng chính là phương sai của từng chiều của dữ liệu.
- Các phần tử ngoài đường chéo $s_{ij}, i \neq j$ thể hiện sự tương quan giữa thành phần thứ i và thứ j của dữ liệu (là hiệp phương sai). Giá trị này có thể dương, âm hoặc bằng 0. Khi nó bằng 0, ta nói rằng hai thành phần i và j trong dữ liệu là không tương quan (uncorrelated).
- Nếu ma trận hiệp phương sai là ma trận đường chéo, ta có dữ liệu hoàn toàn không tương quan giữa các chiều.

Kỳ vọng và ma trận hiệp phương sai

- Không gian 2 chiều mà hai chiều không tương quan.

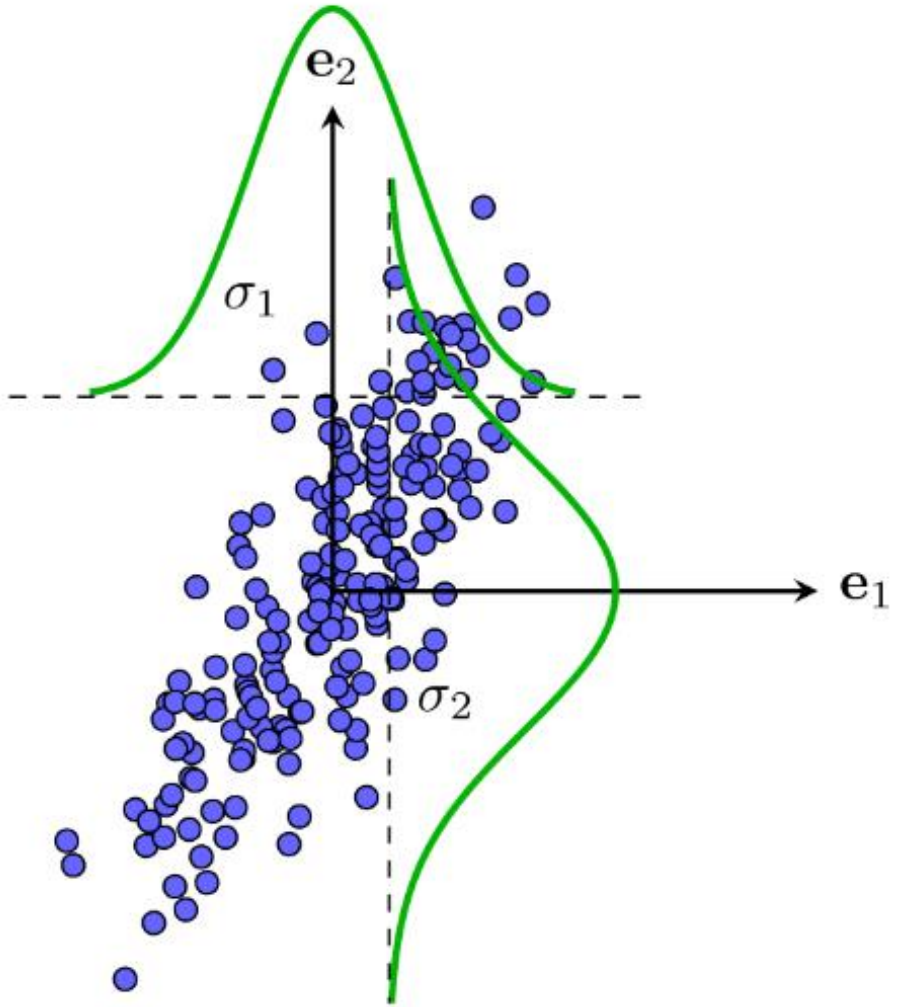
Trong trường hợp này, ma trận hiệp phương sai là ma trận đường chéo với hai phần tử trên đường chéo là σ1, σ2, đây cũng chính là hai trị riêng của ma trận hiệp phương sai và là phương sai của mỗi chiều dữ liệu.



Kỳ vọng và ma trận hiệp phương sai

- Dữ liệu trong không gian hai chiều có tương quan.

Theo mỗi chiều, ta có thể tính được kỳ vọng và phương sai. Phương sai càng lớn thì dữ liệu trong chiều đó càng phân tán. Trong ví dụ này, dữ liệu theo chiều thứ hai phân tán nhiều hơn so so với chiều thứ nhất.

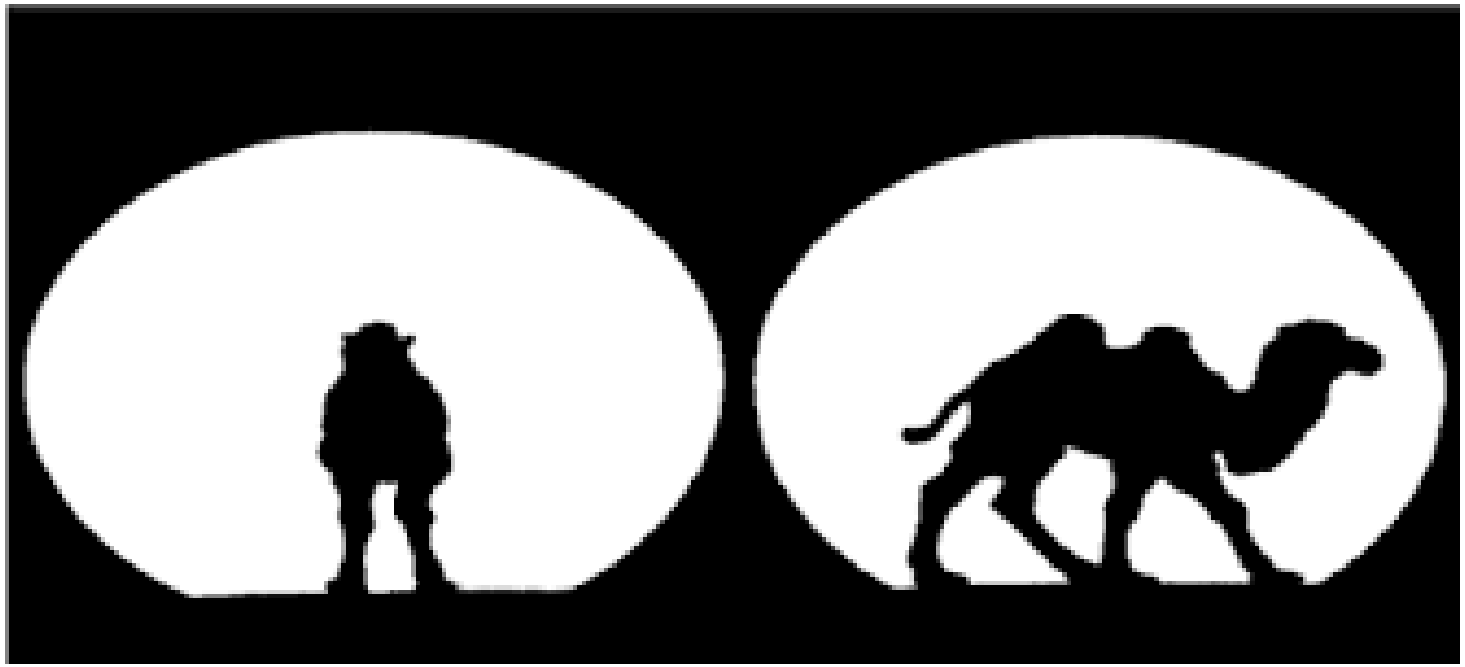


Principal Component Analysis

- Cách đơn giản nhất để giảm chiều dữ liệu từ D về $K < D$ là chỉ giữ lại K phần tử *quan trọng nhất*. Tuy nhiên, chúng ta gặp phải:
  - chưa biết xác định thành phần nào là quan trọng hơn thành phần nào.
  - trong trường hợp xấu nhất, lượng thông tin mà mỗi thành phần mang là như nhau, bỏ đi thành phần nào cũng dẫn đến việc mất một lượng thông tin lớn.
- Cách giải quyết:
  - biểu diễn các vector dữ liệu ban đầu trong một hệ cơ sở mới mà trong hệ cơ sở mới đó, *tầm quan trọng* giữa các thành phần là khác nhau rõ rệt
  - Sau đó bỏ qua những thành phần ít quan trọng nhất.

Principal Component Analysis

- Ví dụ: có hai camera để chụp một con lạc đà, một camera đặt phía trước mặt con lạc đà và một camera đặt phía ngang con lạc đà:

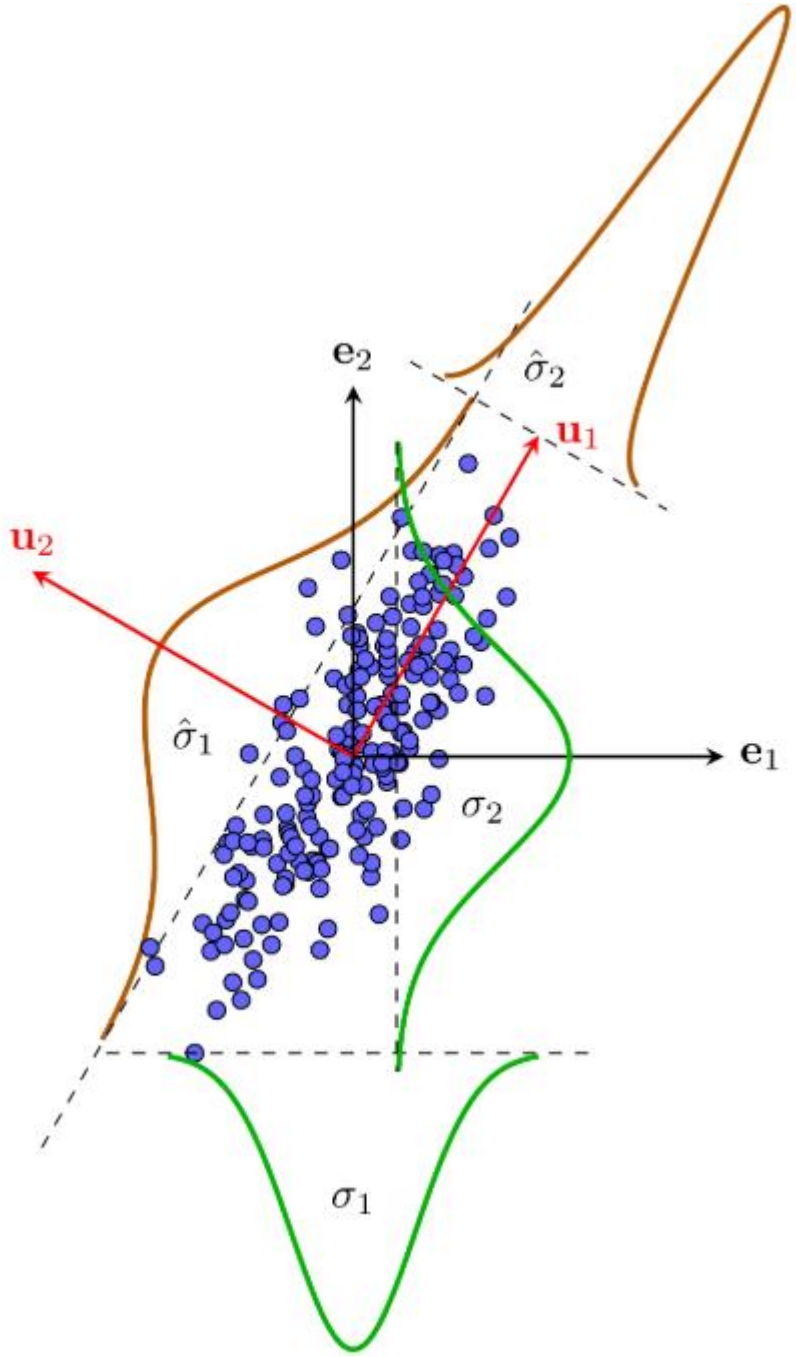


Principal Component Analysis

- PCA là phương pháp đi tìm một hệ trực chuẩn mới sao cho trong hệ này, các thành phần quan trọng nhất nằm trong K thành phần đầu tiên.
- Để cho đơn giản trong tính toán, PCA sẽ tìm một hệ trực chuẩn để làm cơ sở mới.

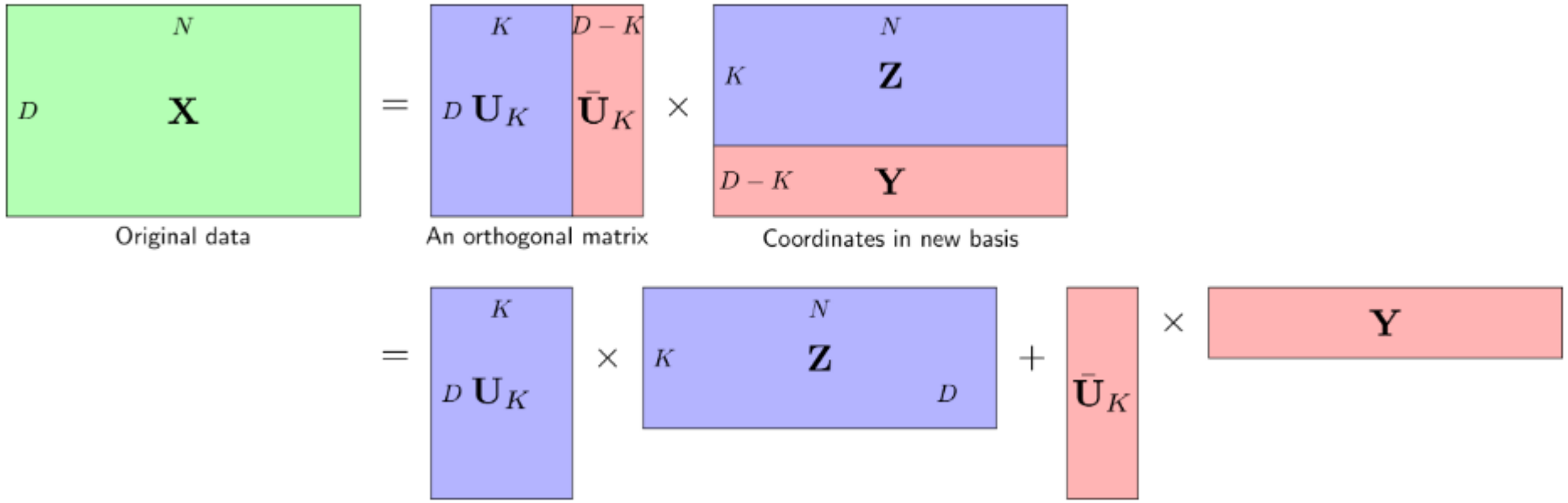
Principal Component Ana

- Dưới góc nhìn Thống kê, PCA có thể được coi là phương pháp đi tìm một hệ cơ sở trực chuẩn đóng vai trò một phép xoay, sao cho trong hệ cơ sở mới này, phương sai theo một số chiều nào đó là rất nhỏ, và ta có thể bỏ qua.



Principal Component Analysis

- GS hệ cơ sở trực chuẩn mới là U và chúng ta muốn giữ lại K toạ độ trong hệ cơ sở mới:



Principal Component Analysis

- Quan sát hình vẽ trên với cơ sở mới $U = [UK, \bar{U}_K]$ là một hệ trực chuẩn với U_K là ma trận con tạo bởi K cột đầu tiên của U.
- Với cơ sở mới này, ma trận dữ liệu có thể được viết thành:

$$\mathbf{X} = \mathbf{U}_K\mathbf{Z} + \bar{\mathbf{U}}_K\mathbf{Y} \quad (8)$$

- Từ đây ta suy ra:

$$\begin{bmatrix} \mathbf{Z} \\ \mathbf{Y} \end{bmatrix} = \begin{bmatrix} \mathbf{U}_K^T \\ \bar{\mathbf{U}}_K^T \end{bmatrix} \mathbf{X} \Rightarrow \begin{matrix} \mathbf{Z} = \mathbf{U}_K^T\mathbf{X} \\ \mathbf{Y} = \bar{\mathbf{U}}_K^T\mathbf{X} \end{matrix} \quad (9)$$

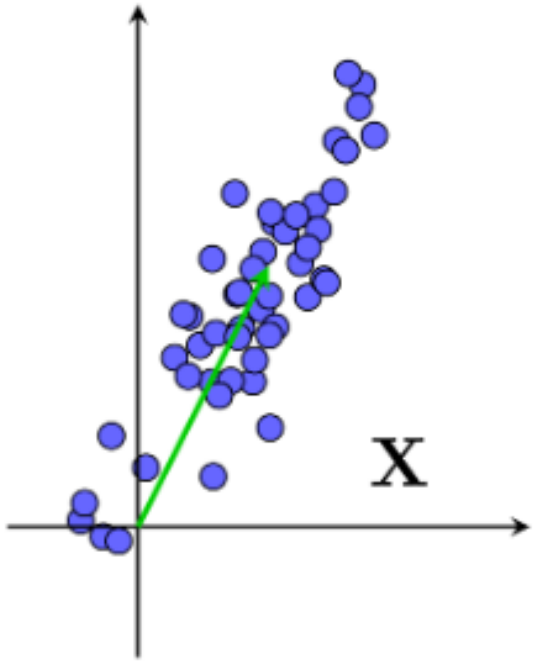
- Mục đích của PCA là đi tìm ma trận trực giao **U** sao cho phần lớn thông tin được giữ lại ở phần màu xanh $\mathbf{U}_K\mathbf{Z}$ và phần màu đỏ $\bar{\mathbf{U}}_K\mathbf{Y}$ sẽ được lược bỏ và thay bằng một ma trận không phụ thuộc vào từng điểm dữ liệu.

Principal Component Analysis

Các bước thực hiện PCA

Bước 1: Tính vector kỳ vọng của toàn bộ dữ liệu:

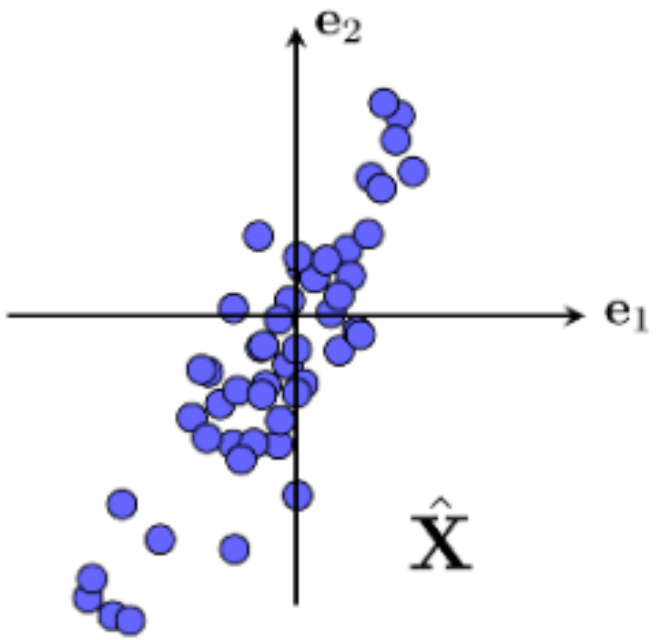
$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$



Principal Component Analysis

Bước 2: Trừ mỗi điểm dữ liêu đi vector kỳ vong của toàn bộ dữ liệu:

$$\hat{\mathbf{x}}_n = \mathbf{x}_n - \bar{\mathbf{x}}$$



Principal Component Analysis

Bước 3: Tính ma trận hiệp phương sai S:

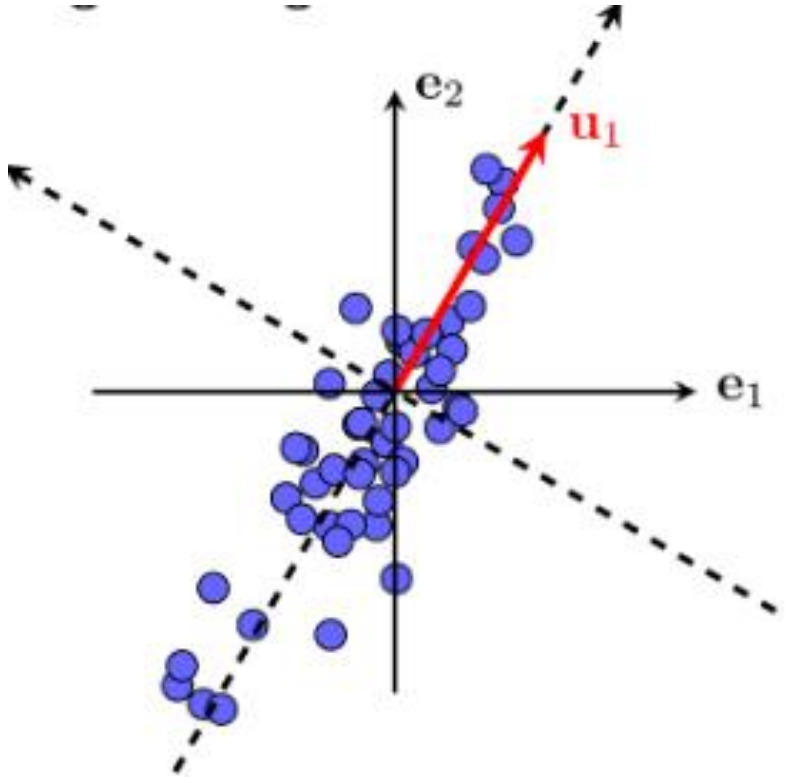
$$\mathbf{S} = \frac{1}{N}\hat{\mathbf{X}}\hat{\mathbf{X}}^T$$

Bước 4: Tính các trị riêng và vector riêng của ma trận S $(\lambda_1, \mathbf{u}_1), \ldots, (\lambda_D, \mathbf{u}_D)$ sắp xếp chúng theo thứ tự giảm dần của trị riêng.

(lưu ý về sự trực giao của các u_i)

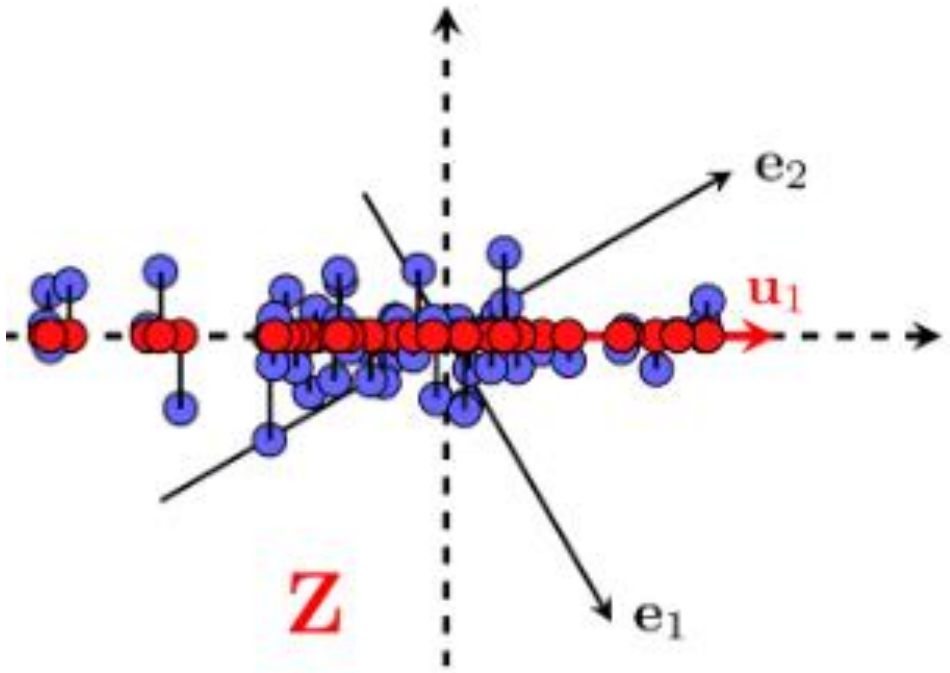
Principal Component Analysis

Bước 5: Chọn K vector riêng ứng với K trị riêng lớn nhất để xây dựng ma trận $\mathbf{U}_K$ có các cột tạo thành một hệ trực giao. K vectors này là các thành phần chính



Principal Component Analysis

Bước 6: Chiếu dữ liệu ban đầu đã chuẩn hoá $\hat{X}$ xuống không gian con tìm được. Dữ liệu mới chính là toạ độ của các điểm dữ liệu trên không gian mới $\mathbf{Z} = \mathbf{U}_K^T \hat{\mathbf{X}}$



Ví dụ trên Python

- Ví dụ về tập dữ liệu Iris bao gồm 4 thuộc tính và 3 nhãn tương ứng với 3 loài hoa.

✓ Rất khó để có thể nhận biết rằng 4 thuộc tính này có phân tách với nhau theo mỗi loài hay không vì cần biểu diễn không gian này trên dữ liệu 4 chiều.

✓ Vì vậy, thuật toán giảm chiều dữ liệu giúp đưa về không gian 2 chiều để dễ dàng trực quan hóa trên hệ toạn độ Oxy, đổi lại là chúng ta phải chấp nhận mất mát đi một lượng thông tin.

✓ Dữ liệu mới dễ dàng phân tích hơn, có thể thấy lớp nào dễ nhầm lẫn với nhau, mức độ tách biệt giữa các lớp,...

Ví dụ trên Python

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import pandas as pd
class PCA:
  def __init__(self, n_dimention: int):
    self.n_dimention = n_dimention

  def fit_transform(self, X):
    #Tính vector trung bình, sau đó trừ các điểm dữ liệu cho vector đó:
    mean = np.mean(X, axis=0)
    X = X - mean

    #Tìm ma trận hiệp phương sai:
    cov = X.T.dot(X) / X.shape[0]
```

Ví dụ trên Python

```
#Tính trị riêng, vector riêng:
    eigen_values, eigen_vectors, = np.linalg.eig(cov)

#Sắp các giá trị riêng từ lớn đến nhỏ, rồi chọn k vector riêng tương ứng
để tạo ma trận U
    select_index = np.argsort(eigen_values)[::-1][:self.n_dimention]
    U = eigen_vectors[:, select_index]

#Ánh xạ dữ liệu sang không gian mới
    X_new = X.dot(U)
    return X_new

if __name__ == "__main__":
    data = pd.read_csv('iris.csv',header=None)
    X = data.values[:,:4]
    Y = data.values[:,4]
```

Ví dụ trên Python

```
    print(X)
    print(Y)
    pca = PCA(n_dimention=2)
    new_X = pca.fit_transform(X)
    print(new_X)

for label in set(Y):
    X_class = new_X[Y == label]
    plt.scatter(X_class[:, 0], X_class[:, 1], label=label)

#plt.legend()
plt.show()
```

Ví dụ trên Python

- Ta thấy rằng số chiều dữ liệu là 116×98=11368 là một số khá lớn.
- Tuy nhiên, vì chỉ có tổng cộng 15×11=165 bức ảnh nên ta có thể nén các bức ảnh này về dữ liệu mới có chiều nhỏ hơn 165. Ở đây, ta chọn K=100.