

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA CÔNG NGHỆ THÔNG TIN  
KHOA CHUYÊN NGÀNH VỀ THỊ GIÁC MÁY TÍNH  
VÀ KỸ THUẬT ĐIỀU KHIỂN TỰ ĐỘNG  
MÔN HỌC: PHÂN TÍCH THỐNG KÊ DỮ LIỆU NHIỀU BIẾN



SEMINAR ĐỒ ÁN  
**BÁO CÁO**

*Tài liệu này mô tả nội dung của đề tài cho môn học  
Phân tích thống kê dữ liệu nhiều biến*

Giáo viên hướng dẫn  
*Lý Quốc Ngọc  
Nguyễn Mạnh Hùng  
Phạm Thanh Tùng*

Sinh viên thực hiện  
*Phan Trí Nguyên  
Nguyễn Mạnh Hùng  
Lưu Hoàng Minh*

TP.Hồ Chí Minh, ngày 25 tháng 4 năm 2023

# MỤC LỤC

<b>TRANG BÌA.....</b>	<b>1</b>
<b>MỤC LỤC .....</b>	<b>2</b>
<b>1 TÓM TẮT .....</b>	<b>4</b>
1.1 Thông tin sinh viên thực hiện .....	4
1.2 Thông tin đồ án môn học .....	4
<b>2 GIỚI THIỆU .....</b>	<b>5</b>
2.1 ĐỘNG LỰC NGHIÊN CỨU .....	5
2.2 PHÁT BIỂU BÀI TOÁN.....	6
2.2.1 Giải thích cơ bản về đầu vào và đầu ra của hệ thống .....	6
2.2.2 Framework của hệ thống .....	6
2.2.3 Công dụng của bài toán PCA .....	7
2.2.4 Giới hạn của bài toán.....	7
2.2.5 Đóng góp của nhóm .....	7
<b>3 CÁCH TIẾP CẬN BÀI TOÁN PHÂN TÍCH THÀNH PHẦN CHÍNH .....</b>	<b>8</b>
3.1 CÁCH ĐẠI SỐ .....	8
3.2 CÁCH HÌNH HỌC .....	8
3.3 XÁC ĐỊNH THÀNH PHẦN CHÍNH.....	9
Các giá trị riêng của ma trận covariance.....	9
3.4 NGUYÊN LÝ TÌM MA TRẬN A BIẾN ĐỔI CỦA THUẬT TOÁN PCA .....	9
<b>4 PHƯƠNG PHÁP TOÁN HỌC.....</b>	<b>12</b>
4.1 MIÊU TẢ CHI TIẾT PHƯƠNG PHÁP .....	12
4.2 BÀI TOÁN VÍ DỤ ÁP DỤNG THUẬT TOÁN PCA .....	13
<b>5 THỰC NGHIỆM.....</b>	<b>19</b>
5.1 TẬP DỮ LIỆU THỰC NGHIỆM.....	19
5.2 SOURCE CODE NHÓM PHÁT TRIỂN .....	20
5.3 KẾT QUẢ THỰC NGHIỆM .....	26
5.2.1 Biểu diễn đồ thị với ba thành phần chính.....	26
5.2.2 Biểu diễn đồ thị với hai thành phần chính khi áp dụng thuật toán PCA .....	27
5.2.3 Đánh giá kết quả.....	27

5.4 HƯỚNG DẪN SỬ DỤNG SOURCE CODE.....	28
<b>6 KẾT LUẬN .....</b>	<b>32</b>
<b>7 TÀI LIỆU THAM KHẢO .....</b>	<b>33</b>

## 1

## TÓM TẮT

## 1.1 Thông tin sinh viên thực hiện

Nhóm	MSSV	Họ và tên	Địa chỉ email	Lớp học
10	20127578	Phan Trí Nguyên	20127578@student.hcmus.edu.vn	20TGMT01
	20127030	Nguyễn Mạnh Hùng	20127030@student.hcmus.edu.vn	20TGMT01
	20127048	Lưu Hoàng Minh	20127048@student.hcmus.edu.vn	20TGMT01

## 1.2 Thông tin đề án môn học

STT	Tên đề án	Môi trường lập trình
2	Phân tích thành phần chính	Phần mềm: Visual Studio Code, Google Colabotary , Github, Zoom.  Ngôn ngữ lập trình: Python

# 2

## GIỚI THIỆU

### 2.1 ĐỘNG LỰC NGHIÊN CỨU

Một trong những thách thức phổ biến nhất trong việc phân tích dữ liệu (phân tích văn bản, ảnh, video, v.v.) là sự đa chiều và thừa của dữ liệu.

Thường xảy ra những trường hợp mà số lượng điểm dữ liệu rất nhỏ, nhưng mỗi điểm lại có một lượng lớn đặc trưng.



**Hình 7.** Hình ảnh mẫu từ bộ dữ liệu thử nghiệm MNIST

Lấy một ví dụ về tập dữ liệu MNIST [1] (Modified National Institute of Standards and Technology database) là một phần của cơ sở dữ liệu rộng lớn chứa các hình ảnh chữ viết tay (**Hình 7**) được sử dụng để huấn luyện các mô hình Học máy khác nhau.

- Số lượng hình ảnh huấn luyện: 60,000
- Số lượng hình ảnh kiểm tra: 10,000

Mỗi hình ảnh có kích thước 28x28 điểm ảnh. Giá trị của mỗi điểm ảnh nằm trong khoảng từ 0 đến 255, biểu thị độ sáng hoặc tối của điểm ảnh đó.

Tập dữ liệu huấn luyện (.csv) có 785 cột. Cột đầu tiên biểu thị nhãn, tức là chữ số nào đã được vẽ, và 784 giá trị còn lại biểu thị giá trị của các điểm ảnh trong hình ảnh.

label	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	...	pixel774	pixel775	pixel776	pixel777	pixel778	pixel779	pixel780	pixel781
0	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0
3	4	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0

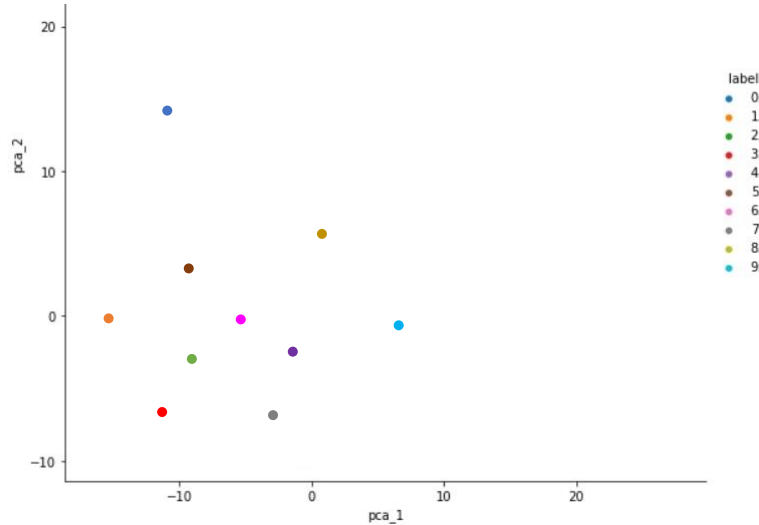
**Thẻ hiện 5 dòng đầu tiên trong tập dữ liệu thực nghiệm MNIST**

Có thể thấy rõ từ hình minh họa trên rằng số lượng điểm dữ liệu chỉ có mười (mười chữ số viết tay từ 0 đến 9), nhưng mỗi điểm có  $28 \times 28 = 785$  pixel.

Trong những tình huống như vậy, việc xây dựng một mô hình có thể mô tả tất cả các đặc trưng và giúp chúng ta hiểu bản chất của dữ liệu là không khả thi. Để vượt qua điều này, người ta cố gắng tăng số lượng điểm dữ liệu hoặc giảm số chiều của mỗi điểm.

Trong chủ đề này, báo cáo sẽ đi qua một nghiên cứu về phân tích thành phần chính (PCA) - một kỹ thuật giảm chiều dữ liệu giúp chuyển đổi một bộ dữ liệu có số chiều cao (bộ dữ liệu có nhiều đặc trưng / biến) thành một bộ dữ liệu có số chiều thấp.

Với bảng thống kê được cung cấp, khi áp dụng PCA để giảm số chiều xuống chỉ còn hai, chúng ta thu được biểu đồ như sau:



Với đồ thị trên, dữ liệu đã được chuyển từ 785 chiều xuống còn 2 chiều. Các điểm dữ liệu của nhãn 0.0 (nhãn màu xanh đậm) có sự khác biệt lớn so với các nhãn khác. Việc xác định bản chất của dữ liệu này giúp chúng ta đưa ra các quyết định hợp lý. Trong ví dụ này, các hình ảnh chữ số viết tay của nhãn 0.0 có một số đặc điểm độc đáo so với các điểm dữ liệu khác.

Đó là một ví dụ cơ bản về tính hữu dụng của PCA trong khai thác dữ liệu nằm ở khả năng giảm chiều dữ liệu mà vẫn giữ lại các đặc trưng quan trọng nhất. Điều này không chỉ giúp cho việc phân tích dữ liệu hiệu quả hơn mà còn giúp đơn giản hóa quá trình trực quan hóa và giải thích dữ liệu.

## 2.2 PHÁT BIỂU BÀI TOÁN

### 2.2.1 Giải thích cơ bản về đầu vào và đầu ra của hệ thống

Đầu vào của hệ thống:  $X = [X_1|X_2|\dots|X_p]$  là các vectors đặc trưng ( $p$  đặc trưng) cần phải giảm số chiều.

Đầu ra của hệ thống:  $Y$  là tập dữ liệu trong không gian mới của  $X$  và giữ được các đặc trưng chính, có số chiều là  $m$  với  $m \ll p$ .

Ấn số cần tìm: Ta cần tìm phép biến đổi tuyến tính  $A$  sao cho phép biến đổi  $Y = XA$ , có kết quả gồm các tính chất bao gồm độ phương sai dữ liệu lớn nhất có thể và cực tiểu hóa việc mất mát thông tin.

### 2.2.2 Framework của hệ thống

Mục tiêu của thuật toán là để trích xuất thông tin quan trọng từ dữ liệu thống kê để biểu diễn nó dưới dạng một tập hợp các giá trị trực giao mới, với các biến được gọi là các thành phần chính (Principal Component).

Các giá trị trực giao trong không gian mới được xây dựng sao cho trên mỗi trục, độ biến thiên của dữ liệu trên đó là lớn nhất có thể, đồng thời số chiều trong không gian trục mới phải nhỏ hơn số chiều trong không gian cũ.

Framework chung của hệ thống này bao gồm:

Giai đoạn	Framework
Giai đoạn 1	Nhập vào một số thư viện cần thiết để áp dụng các thuật toán PCA cho dữ liệu được cung cấp.
Giai đoạn 2	Đọc dữ liệu từ tệp CSV được cung cấp.
Giai đoạn 3	Nhập các tham số đầu vào cần thiết để chạy chương trình.
Giai đoạn 4	Viết từng hàm thực hiện một số phân tích đa biến cơ bản với trực quan hóa.
Giai đoạn 5	Tải các tham số đầu vào vào các hàm PCA và in kết quả ra (một biểu đồ).

### 2.2.3 Công dụng của bài toán PCA

Số thứ tự	Miêu tả công dụng
1	Rút trích thông tin quan trọng nhất từ tập dữ liệu.
2	Giảm số chiều dữ liệu với khả năng cực tiểu hóa việc mất mát thông tin và cực đại hóa độ biến thiên của dữ liệu.
3	Xử lý vấn đề và phân tích sự tương quan giữa các biến trong dữ liệu ban đầu bằng cách sử dụng các biến mới mà thuật toán đã tìm được.

### 2.2.4 Giới hạn của bài toán

Bài toán Phân tích thành phần chính hoạt động tốt khi trong bộ dữ liệu có nhiều biến có tương quan tuyến tính với nhau.

Ngược lại bài toán lại thể hiện rõ ràng những bất lợi khi:

- Bộ dữ liệu gốc có nhiều biến độc lập với nhau. Khi đó giá trị covariance matrix gần như xấp xỉ bằng 0, vì vậy các biến có sự trùng lặp nhiều dẫn đến việc rút gọn dữ liệu trở nên kém hiệu quả.
- Mỗi quan hệ giữa các biến trong dữ liệu gốc là phi tuyến tính.

### 2.2.5 Đóng góp của nhóm

Đầu tiên, các thành viên trong nhóm đã cung cấp một lời giải thích rõ ràng và công thức khoa học về bài toán Phân tích thành phần chính, bắt đầu từ định nghĩa và mục đích của nó cho các bước liên quan đến việc thực hiện nó. Giải thích này sẽ hữu ích cho bất cứ ai muốn tìm hiểu về PCA, bất kể kiến thức nền tảng của họ.

Bên cạnh đó, việc áp dụng PCA cho một vấn đề trong thế giới thực, cụ thể là việc phân loại bộ dữ liệu IRIS. Ứng dụng này rất hữu ích để chứng minh làm thế nào PCA có thể được sử dụng trong thực tế để giảm tính kích thước của bộ dữ liệu và cải thiện độ chính xác phân loại.

Cuối cùng nhưng không kém phần quan trọng, nhóm cũng đã cung cấp một số hình ảnh trực quan giúp minh họa các khái niệm về PCA, bao gồm các sơ đồ phân tán của các bộ dữ liệu gốc được chuyển đổi và biểu đồ phương sai được giải thích tích lũy. Những trực quan này rất hữu ích để hiểu tác động của PCA đối với bộ dữ liệu và để đánh giá chất lượng của kết quả PCA.

# 3

## CÁCH TIẾP CẬN BÀI TOÁN PHÂN TÍCH THÀNH PHẦN CHÍNH

### 3.1 CÁCH ĐẠI SỐ

Các thành phần chính là các tổ hợp tuyến tính của  $n$  biến gốc  $X_1, X_2, \dots, X_p$  sao cho:

- Thành phần chính thứ nhất có phương sai lớn nhất có thể.
- Thành phần chính thứ 2 có phương sai lớn nhất có thể và trực giao với thành phần chính thứ nhất.
- Cứ tiếp tục như threshold đến thành phần chính cuối cùng.

### 3.2 CÁCH HÌNH HỌC

Thực hiện phép xoay để chuyển về một hệ trục tọa độ mới có khả năng biểu diễn dữ liệu tốt hơn, cho phép khám phá được những liên kết tiềm ẩn trong nội bộ dữ liệu cũ mà ta không thể thấy rõ trên hệ trục cũ. Các trục tọa độ mới phải được đảm bảo luôn trực giao đôi một với nhau.

Các thành phần chính sẽ được biểu diễn một hệ trục tọa độ mới thu được bằng cách xoay các trục gốc tới tập hợp các trục mới:

- Thành phần chính thứ nhất có hướng mà mô tả được lớn nhất sự biến thiên của dữ liệu.
- Thành phần chính thứ hai có hướng mà mô tả được lớn nhất sự biến thiên của dữ liệu và trực giao với thành phần chính thứ nhất.

Cứ tiếp tục như thế cho đến khi thành phần chính cuối cùng có hướng mà mô tả được ít nhất sự biến thiên của dữ liệu và trực giao với tất cả các thành phần chính khác.

Lưu ý: Tập hợp các biến mới  $y_1, y_2, y_3, \dots, y_p$  được xây dựng sao cho:

- $y_1$  có thể mô tả được nhiều nhất có thể sự biến thiên của dữ liệu gốc trong số tất cả các tổ hợp tuyến tính tập từ  $x_1, x_2, x_3, \dots, x_p$ .
- $y_2$  được chọn sao cho có thể mô tả được nhiều nhất có thể sự biến thiên của dữ liệu còn lại và không tương quan với  $y_1$ .

Khi đó các biến  $y_1, y_2, y_3, \dots, y_m$  được xây dựng như trên được gọi là thành phần chính. Các thành phần này tạo thành một hệ trục tọa độ trực giao mới, với  $m \ll p$  nhưng vẫn thể hiện đầy đủ hầu hết sự biến thiên của các biến gốc  $x_1, x_2, x_3, \dots, x_p$ . Đây chính là việc giảm số chiều dữ liệu trong thuật toán này.



### 3.3 XÁC ĐỊNH THÀNH PHẦN CHÍNH

Các giá trị riêng của ma trận covariance

Các giá trị riêng của ma trận covariance trong thuật toán PCA, giảm dần về mặt giá trị theo hàm logarit. Bởi vì, các thành phần chính  $y_1, \dots, y_m$  lần lượt thỏa các điều kiện sau:

- Thành phần chính thứ 1 là tổ hợp tuyến tính  $a_1^T * x$  mà làm cực đại  $\text{var}(y_1) = a_1^T * S * a_1$  với ràng buộc  $a_1^T * a_1 = 1$ .
- Thành phần chính thứ 2 là tổ hợp tuyến tính  $a_2^T * x$  mà làm cực đại  $\text{var}(y_2) = a_2^T * S * a_2$  với ràng buộc  $a_1^T * a_1 = 1$  và  $\text{cov}[a_1^T * x, a_2^T * x] = a_1^T * S * a_2 = 0$ .
- Tổng quát cho thành phần chính thứ  $j$  là tổ hợp tuyến tính  $(a_j^T) * x$  mà làm cực đại  $\text{var}(y_j) = a_j^T * S * a_j$  với ràng buộc  $a_j^T * a_j = 1$  và  $\text{cov}[a_j^T * x, a_k^T * x] = a_j^T * S * a_k = 0$ , với mọi giá trị của  $k < j$ .

Như vậy, các thành phần chính về sau sẽ biểu diễn lượng phương sai còn lại trong tập dữ liệu vì tính trực giao đôi một giữa các thành phần chính trước đó. Vì thế, giá trị riêng  $\lambda$  có xu hướng suy giảm nhanh cho vài dữ liệu gốc ban đầu, và dần dần giảm chậm lại đối với các giá trị riêng còn lại để biểu diễn các thành phần chính còn lại trong dữ liệu.

### 3.4 NGUYÊN LÝ TÌM MA TRẬN A BIẾN ĐỔI CỦA THUẬT TOÁN PCA

Ta có:  $Y = XA \Leftrightarrow Y = A^T X$  (1)

Với  $A$  là ma trận chuyển giao, là ẩn số cần tìm. Nhưng do  $A$  cũng là ma trận trực giao nên  $A^T = A^{-1}$ . Như vậy, từ **biểu thức (1)**, ta có:

$$Y = A^T X \Leftrightarrow X = (A^T)^{-1} Y \Leftrightarrow X = (A^{-1})^{-1} Y \Leftrightarrow X = A Y \Leftrightarrow X = \sum_{i=1}^m Y_i * A_i$$

Trong đó,  $A_i$  là các vector cột của ma trận  $A$  và  $Y_i$  là các vector cột của ma trận  $Y$ .

Như vậy ta có thể nói, ma trận biến đổi  $A$  là ma trận cần tìm sao cho thỏa mãn 2 tính chất của phép biến đổi sau:

$$\begin{cases} Y = XA \\ X = AY \end{cases}$$

Ta gọi  $a$  là trục mới trong không gian mới cần tìm,  $a \in \mathbb{R}^m$  và  $a^T a = 1$ . Khi đó, tọa độ mới của  $x_i$  trên trục  $a$  được tính  $y_i = a^T x_i$

Mục tiêu của chúng ta là làm  $y_i$  max và cực tiểu hóa độ sai số nhằm cho  $a$  biểu diễn tốt nhất.

Đối với cực đại hóa  $y_i$ , ta lần lượt đi thực hiện Cực đại tổng  $\sum (y_i)^2$ , với bình phương để loại bỏ các số âm. Ta có:

$$\sum_{i=1}^n y_i^2 = \sum_{i=1}^n y_i * y_i = \sum_{i=1}^n (a^T x_i) * (a^T x_i) = a^T X X^T a$$

Để tìm được vector  $a$  thỏa mãn  $a^T X X^T a$  đạt giá trị cực đại thì ta chuyển về pp toán tử Lagrange như sau:

$$L(a, \lambda) = a^T X X^T a + \lambda(1 - a^T a)$$

Với  $XX^T$  chính là ma trận hiệp phương sai  $\Sigma$  của  $X$ .

Điều kiện để tính cực trị:

$$\frac{\partial F}{\partial a} = 0 \text{ và } \frac{\partial F}{\partial \lambda} = 0$$

Xét điều kiện trên, ta có  $D = a^T \Sigma - a^T \lambda = 0 \Leftrightarrow \Sigma a = \lambda a \Rightarrow$  Như vậy  $a$  là các vector riêng ứng với  $\lambda$  là các giá trị riêng của  $\Sigma$  (2)

Đối với cực tiểu hóa độ sai số để cực tiểu hóa độ mất mát thông tin ở mức tối đa thì ta có:

$$X \sim \hat{X} = \sum_{i=1}^m y_i a_i \quad (m < n) \Rightarrow \text{Sai số } E \sim \|X - \hat{X}\|^2 \Leftrightarrow E = \|\sum_{i=1}^m y_i a_i\|^2$$

Do các vector  $a_i$  là các vector trực giao nên:

$$\begin{aligned} E &= \sum_{i=1}^m y_i^2 \\ &= \sum_{i=1}^m y_i * y_i \\ &= \sum_{i=1}^m (a_i^T x_i) * (a_i^T x_i) \\ &= \sum_{i=1}^m a_i^T X X^T a_i \end{aligned}$$

Với  $a_i^T * a_j = 1$  nếu  $i = j$  và  $a_i^T * a_j = 0$  nếu  $i$  khác  $j$

Tối ưu hóa sai số cực tiểu bằng phương pháp nhân tử Lagrange:

$$L(a_i, \lambda_i) = \sum_{i=1}^n a_i^T X X^T a_i + \sum_{i=1}^n \lambda_i (1 - a_i^T a_i)$$

Với  $XX^T$  chính là ma trận tương quan  $\Sigma$  của  $X$ .

Điều kiện để tính cực trị:

$$\frac{\partial F}{\partial a_i} = 0 \text{ và } \frac{\partial F}{\partial \lambda_i} = 0$$

Xét điều kiện trên, ta có  $E = a_i^T \Sigma - a_i^T \lambda_i = 0 \Leftrightarrow \Sigma a_i = \lambda_i a_i$ . Khi đó ta chọn  $a_i$  là các vector riêng của ma trận  $\Sigma$  (3).

Như vậy, từ (2) và (3) ta có thể thấy để tìm được các trục làm cực đại phương sai và cực tiểu độ mất mát thông tin thì ta đều tính được từ các vector riêng ứng với các trị riêng của ma trận tương quan  $\Sigma$ . Thông qua phép tính tìm trị riêng, ta tính được các giá trị riêng như sau:

$$\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \dots \geq \lambda_{m-1} \geq \lambda_m \geq \lambda_{m+1} \geq \dots \geq \lambda_n \geq 0$$

Qua đó, các giá trị riêng này có giá trị giảm dần theo hàm logarit.

Thông qua điều kiện của công thức (3), để tính được các vector riêng sao cho thu được E có giá trị nhỏ nhất thì ta chọn các giá trị riêng từ  $m+1$  đến  $n$  ứng với top các giá trị riêng nhỏ nhất.

Để cực đại hóa phương sai của  $y_i$ , theo điều kiện của công thức số (2), ta có:

$$\sigma^2(y_i) = a_i^T X X^T a_i = a_i^T \lambda_i a_i = \lambda_i$$

Như vậy, ta chọn  $m$  giá trị đầu tiên của trị riêng ứng với top các trị riêng lớn nhất để cực đại hóa phương sai của tập dữ liệu trong không gian mới  $Y$ . Đồng thời, khi chọn các  $m$  giá trị lớn nhất thì tổng bình phương các giá trị còn lại sẽ cho kết quả nhỏ nhất, điều này sẽ giúp thỏa mãn yêu cầu cực tiểu độ sai lệch thông tin ở mức tối đa.

**Tổng kết:** Sau khi chọn được  $m$  các giá trị  $\lambda$  lớn nhất thì ta sẽ thu được các vector riêng tương ứng lần lượt là  $a_1, a_2, \dots, a_m$  với  $m$  trị riêng. Lúc này, ta chỉ cần xây dựng ma trận biến đổi trực giao  $A$  cần tìm bằng cách  $A = [a_1 | a_2 | \dots | a_m]$  với  $a_i$  là các vector cột trong ma trận  $A$ .

## 4

## PHƯƠNG PHÁP TOÁN HỌC

## 4.1 MIÊU TẢ CHI TIẾT PHƯƠNG PHÁP

Việc miêu tả chi tiết cho thuật toán Phân tích thành phần chính sẽ được thông qua 6 bước sau:

Bước 1: Tính các vector kỳ vọng của các dữ liệu và dời tọa độ về gốc dữ liệu

Ta tính vector kỳ vọng của toàn bộ dữ liệu bằng công thức sau:  $\hat{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i$

Sau đó trừ mỗi điểm dữ liệu đi vector kỳ vọng của toàn bộ dữ liệu, nhằm thiết lập gốc tọa độ tính toán theo tập dữ liệu:  $\hat{\mathbf{x}}_n = \mathbf{x}_n - \hat{\mathbf{x}}$

Bước 2: Sau khi chúng ta thực hiện bước dời tọa độ về gốc dữ liệu, ta sẽ tính ma trận hiệp phương sai dựa trên tập dữ liệu mới

$$\mathbf{C} = \frac{1}{n} \sum_{i=1}^n \hat{\mathbf{x}}_i \cdot \hat{\mathbf{x}}_i^T$$

Bước 3: Tìm trị riêng của ma trận hiệp phương sai bằng công thức sau

$$\det|\mathbf{A} - \lambda \mathbf{I}| = 0$$

Bằng cách thế A bằng ma trận hiệp phương sai và lambda nhân với ma trận đơn vị I. Sau đó ta thực hiện trừ 2 ma trận và cuối cùng tính giá trị det của ma trận thì ta sẽ được các trị riêng.

Bước 4: Tìm vector riêng tương ứng với các trị riêng

$$\mathbf{A} \cdot \mathbf{v} = \lambda \cdot \mathbf{v}$$

Ta thế A bằng ma trận hiệp phương sai và lambda bằng các trị riêng. Sau khi giải ra ta sẽ được một biểu thức  $\mathbf{Y} = \mathbf{A} \cdot \mathbf{X}$ , ta đã có được vector trị riêng của trị riêng đó.

Bước 5: Sắp xếp vector trị riêng dựa trên trị riêng tương ứng của nó, với trị riêng nào lớn hơn thì xếp đầu tiên, ta được

Sau đó ghép 2 vector trị riêng lại và ta được một vector V, với cột đầu tiên tương ứng với vector trị riêng mà có trị riêng lớn nhất, và tương tự cho các cột sau.

Bước 6: Tạo dựng ma trận chuyển đổi và tiến hành chuyển đổi

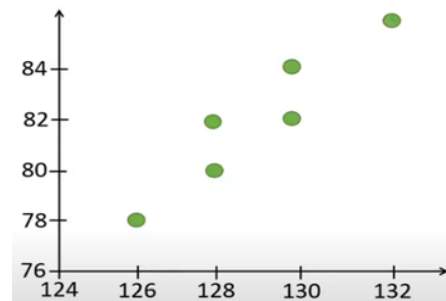
Ta sẽ tạo một ma trận D chứa dữ liệu được dời ban đầu. Sau đó ta tiến hành lấy ma trận D nhân ma trận V.

## 4.2 BÀI TOÁN VÍ DỤ ÁP DỤNG THUẬT TOÁN PCA

Dữ liệu mẫu ta sẽ lấy từ 2 giá trị huyết áp của tâm thu (systolic) và tâm trương (diastolic) của 6 người khác nhau:

Huyết áp tâm thu	Huyết áp tâm trương
126	78
128	80
128	82
130	82
130	84
132	86

**Bảng 5**



**Hình 2**

Dựa vào biểu đồ **Bảng 5** cho thấy, người đầu có giá trị huyết áp tâm trương (Systolic Blood Pressure – SBP) là 78 và giá trị huyết áp tâm thu là 126.

Người thứ 2 có giá trị huyết áp tâm trương (Diastolic Blood Pressure-DBP) là 80 và giá trị huyết áp tâm thu là 128.

Từ đó ta sẽ vẽ được một biểu đồ dataset gồm 2 chiều tương ứng với 2 thành phần của SBP và DBP (**Hình 2**), khi đó, ta thấy có sự tương quan tốt giữa 2 giá trị này.

**Bước 1:** Ta sẽ tính các vector kỳ vọng của các dữ liệu và dời tọa độ về gốc dữ liệu để thực hiện tính toán.

Đầu tiên ta tính được mean của cột giá trị huyết áp tâm thu (systolic) là 129 sau đó t lấy giá trị ban đầu trừ đi 129 ta sẽ được các giá trị trong cột sau (**Bảng 6**), khi đó ta sẽ thu được các kết quả như **Bảng 7**. Các giá trị này cho thấy khoảng cách giữa các giá trị ban đầu so với mean.

Huyết áp tâm thu	Huyết áp tâm trương
$126 - 129 = -3$	$78 - 82 = -4$
$128 - 129 = -1$	$80 - 82 = -2$
$128 - 129 = -1$	$82 - 82 = 0$
$130 - 129 = 1$	$82 - 82 = 0$
$130 - 129 = 1$	$84 - 82 = 2$
$132 - 129 = 3$	$86 - 82 = 4$

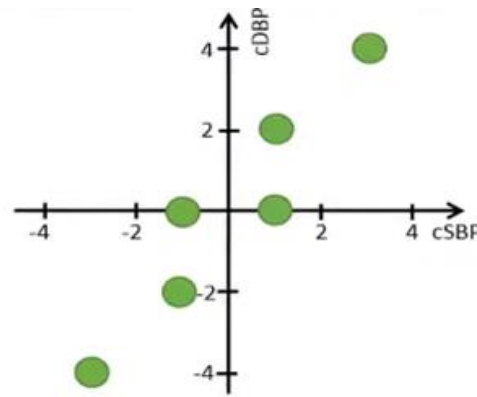
**Bảng 6**

Huyết áp tâm thu	Huyết áp tâm trương
-3	-4
-1	-2
-1	0
1	0
1	2
3	4

**Bảng 7**

Làm điều tương tự với cột huyết áp tâm trương (diastolic) thì ta sẽ được bảng gồm 2 cột Huyết áp tâm thu và Huyết áp tâm trương. Lúc này ta đã thành công dời tọa độ về gốc dữ liệu, bước này sẽ giúp chúng ta dễ dàng tính toán hơn khi chúng ta xoay chiều dữ liệu (**Hình 3**).

**Bước 2:** Sau khi chúng ta thực hiện bước dời tọa độ về gốc dữ liệu, ta sẽ tính ma trận hiệp phương sai dựa trên tập dữ liệu mới và kết quả thu được dựa vào **Bảng 8**.



Hình 3

	SBP	DBP
SBP	4.4	5.6
DBP	5.6	8.0

Bảng 8

Ở đây ta có đường chéo chính tại **Bảng 9** là giá trị phương sai của mỗi dữ liệu dựa vào **biểu thức (4)**.

	SBP	DBP
SBP	4.4	5.6
DBP	5.6	8.0

Bảng 9

$$\text{var}(\text{cSBP}) = \frac{1}{n-1} \sum_{i=1}^n (\text{cSBP}_i - \overline{\text{cSBP}})^2 \quad (4)$$

Khi chúng ta tính toán phương sai của dữ liệu được chuẩn hóa thì mean của giá trị centered SBP (hoặc DBP) sẽ luôn bằng 0.

	SBP	DBP
SBP	4.4	5.6
DBP	5.6	8.0

$$\begin{aligned}
 \text{var}(\text{cSBP}) &= \frac{1}{n-1} \sum_{i=1}^n (\text{cSBP}_i - \overline{\text{cSBP}})^2 \\
 &= \frac{((-3)^2 + (-1)^2 + (-1)^2 + 1^2 + 1^2 + 3^2)}{6-1} \\
 &= \frac{22}{5} \\
 &= 4.4
 \end{aligned}$$

Từ đó ta thấy rằng chỉ cần tính tổng của bình phương các giá trị trong cột và chia cho kích thước của tập dữ liệu trừ đi 1.

Sau đó ta tính tương tự cho giá trị còn lại (**Bảng 10**):

	SBP	DBP
SBP	4.4	5.6
DBP	5.6	8.0

Bảng 10

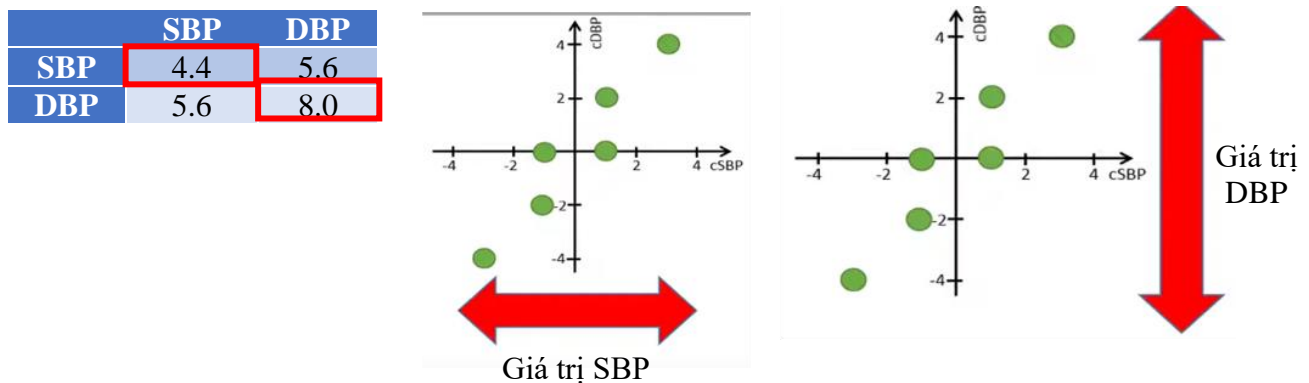
$$\begin{aligned}
 \text{var}(\text{cDBP}) &= \frac{1}{n-1} \sum_{i=1}^n (\text{cDBP}_i - \overline{\text{cDBP}})^2 \\
 &= \frac{((-4)^2 + (-2)^2 + 0^2 + 0^2 + 2^2 + 4)}{6-1} \\
 &= \frac{40}{5} \\
 &= 8.0
 \end{aligned}$$

Cuối cùng ta tính hiệp phương sai (giá trị này đo độ lan ra của 2 dữ liệu) bằng công thức sau:

$$\text{cov}(\text{sSBP}, \text{cDBP}) = \frac{1}{n-1} \sum (\text{cSBP}_i - \overline{\text{cSBP}}) \cdot (\text{cDBP}_i - \overline{\text{cDBP}}) \quad (5)$$

Tại đây, ta tính bằng cách nhân 2 giá trị của 1 dòng tương ứng của 2 cột dữ liệu về bên phải của **công thức 5**, sau đó ta sẽ lấy dữ liệu của 2 cột của người đầu tiên nhân với nhau và tương tự. Cuối cùng ta sẽ cho kích thước của dữ liệu trừ 1. Từ bảng giá trị trên ta thấy rằng:

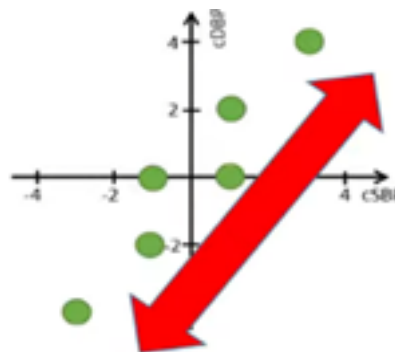
Độ lan ra (spread) của giá trị SBP (4.4) thấp hơn giá trị DBP (8.0) và được minh họa như **hình 4**.



Hình 4

Và giá trị hiệp phương sai sẽ nằm ở đâu đó giữa 2 giá trị được minh họa như **hình 5**.

	SBP	DBP
SBP	4.4	5.6
DBP	5.6	8.0



Hình 5

**Bước 3:** Ta sẽ tìm trị riêng của ma trận hiệp phương sai bằng **công thức 6**:

$$\det|A - \lambda I| = 0 \quad (6)$$

Ta thế A bằng ma trận hiệp phương sai và lambda nhân với ma trận đơn vị I (cùng kích thước với ma trận hiệp phương sai)

$$\det \left| \begin{bmatrix} 4.4 & 5.6 \\ 5.6 & 8.0 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right| = 0$$

Sau đó ta thực hiện trừ 2 ma trận ta được ma trận như sau:

$$\det \left| \begin{bmatrix} (4.4 - \lambda) & 5.6 \\ 5.6 & (8.0 - \lambda) \end{bmatrix} \right| = 0$$

Tiếp theo ta tính giá trị det của ma trận này bằng cách lấy tích huyền trừ tích sất. ta được biểu thức như sau:

$$(4.4 - \lambda)(8.0 - \lambda) - 5.6 \cdot 5.6 = 0$$

$$\Leftrightarrow 3.84 - 12.4\lambda + \lambda^2 = 0 \quad (7)$$

Sau khi giải ra ta sẽ được 2 trị riêng dựa vào **biểu thức 7**, ta sẽ có được kết quả như sau:

$$\lambda_1 = 0.32 \text{ và } \lambda_2 = 12.08$$

**Bước 4:** Ta sẽ tìm vector riêng tương ứng với các trị riêng:

$$A \cdot v = \lambda \cdot v \quad (8)$$

Với  $\lambda_2 = 12.08$ , ta thế vào **biểu thức 8**, ta được:

$$\begin{bmatrix} 4.4 & 5.6 \\ 5.6 & 8.9 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = 12.08 \begin{bmatrix} x \\ y \end{bmatrix}$$

Sau đó ta nhân ma trận hiệp phương sai với vector cột x và y và sau đó nhân trị riêng với vector tương tự thì ta sẽ được biểu thức như sau:

$$\begin{cases} 4.4x + 5.6y = 12.08x \\ 5.6x + 8.0y = 12.08y \end{cases}$$

$$\Leftrightarrow \begin{cases} 5.6y = 7.68x \\ 5.6x = 4.08y \end{cases}$$

$$\Leftrightarrow \begin{cases} y = 1.37x \\ 1.37x = y \end{cases}$$

Cuối cùng ta giải ra được biểu thức  $y = 1.37x$

Từ đó ta được vector trị riêng của ma trận hiệp phương sai:  $v_2 = \begin{bmatrix} 1 \\ 1.37 \end{bmatrix}$ . Tại đây ta tối giản  $v_2$  bằng cách chia 2 vế cho  $\frac{17}{10}$ , ta được:  $v_2 = \begin{bmatrix} 1 \\ 1.37 \end{bmatrix} = \begin{bmatrix} 0.59 \\ 0.81 \end{bmatrix}$  với trị riêng tương ứng  $\lambda_2 = 12.08$  **(9)**

Làm tương tự ta sẽ vector trị riêng còn lại:  $v_1 = \begin{bmatrix} -0.81 \\ 0.59 \end{bmatrix}$  với  $\lambda_1 = 0.32$  **(10)**



**Bước 5:** Tiếp đến ta sẽ chuẩn hóa và sắp xếp vector trị riêng dựa trên trị riêng tương ứng của nó (2 biểu thức (9) và (10)), với trị riêng nào lớn hơn thì xếp đầu tiên, ta được:

$$\begin{cases} v_2 = \begin{bmatrix} 0.59 \\ 0.81 \end{bmatrix}, \text{ với } \lambda_2 = 12.08 \\ v_1 = \begin{bmatrix} -0.81 \\ 0.59 \end{bmatrix}, \text{ với } \lambda_1 = 0.32 \end{cases}$$

Sau đó ghép 2 vector trị riêng lại  $v_1$  và  $v_2$ , ta được một vector V như sau:

$$V = \begin{bmatrix} 0.59 & -0.81 \\ 0.81 & 0.59 \end{bmatrix}$$

Cột đầu tiên tương ứng với vector trị riêng mà có trị riêng lớn nhất, và tương tự cho các cột sau.

**Bước 6:** Ta tạo dựng ma trận chuyển đổi và tiến hành chuyển đổi.

Tại đây ta sẽ tiến hành chuyển đổi dữ liệu gốc được dời ban đầu trở nên không tương quan với nhau.

Ta sẽ tạo một ma trận D chứa dữ liệu gốc được dời ban đầu:

$$D = \begin{bmatrix} -3 & -4 \\ -1 & -2 \\ -1 & 0 \\ 1 & 0 \\ 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Huyết áp tâm thu	Huyết áp tâm trương
-3	-4
-1	-2
-1	0
1	0
1	2
3	4

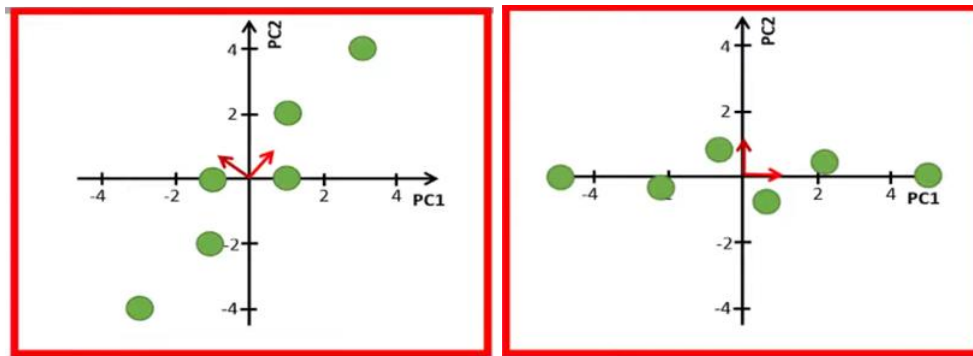
Sau đó ta tiến hành lấy ma trận D nhân ma trận V:

$$DV = \begin{bmatrix} -3 & -4 \\ -1 & -2 \\ -1 & 0 \\ 1 & 0 \\ 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 0.59 & -0.81 \\ 0.81 & 0.59 \end{bmatrix} = \begin{bmatrix} -5.0 & 0.1 \\ -2.2 & -0.4 \\ -0.6 & 0.8 \\ 0.6 & -0.8 \\ 2.2 & 0.4 \\ 5.0 & -0.1 \end{bmatrix}$$

Ta sẽ được ma trận chứa các dữ liệu đã được chuyển đổi như sau:

$$\begin{bmatrix} -5.0 & 0.1 \\ -2.2 & -0.4 \\ -0.6 & 0.8 \\ 0.6 & -0.8 \\ 2.2 & 0.4 \\ 5.0 & -0.1 \end{bmatrix}$$

Những giá trị này được gọi là giá trị thành phần chính và nó đại diện cho dữ liệu được dời ban đầu trong gốc tọa độ mới sau khi xoay (hình 6).



Hình 6

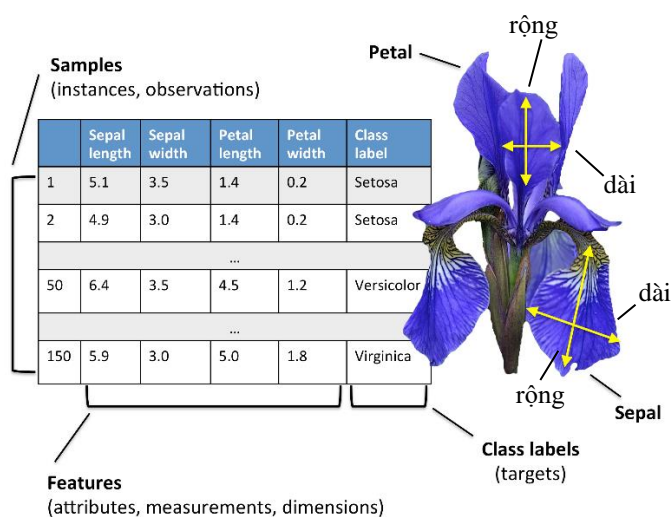
Ta thấy rằng sau khi xoay chiều tọa độ theo chiều kim đồng hồ cho tới khi hai trục x và y trùng hướng với hai vector trị riêng. Nếu như ta muốn chiếu các giá trị thành phần chính lên mặt phẳng tọa độ mới này thì ta sẽ chia hai cột ra thành 2 cột giá trị PC1, và PC2 :

PC1	PC2
-5.0	0.1
-2.2	-0.4
-0.6	0.8
0.6	-0.8
2.2	0.4
5.0	-0.1

# 5

## THỰC NGHIỆM

### 5.1 TẬP DỮ LIỆU THỰC NGHIỆM



Đối với tập dữ liệu Iris CSV (**Hình 1**), tập dữ liệu này được sử dụng trong báo cáo khoa học của R.A. Fisher năm 1936 và có thể được tìm thấy trực tuyến bằng cách tìm kiếm tập tin iris.csv để kiểm tra các thuật toán.

Tập dữ liệu này bao gồm ba loài hoa Iris (Iris-setosa, Iris-versicolor và Iris-virginica), mỗi loài có 50 mẫu riêng biệt, cùng với một số đặc điểm của mỗi loài hoa, bao gồm chiều dài đài hoa, chiều rộng đài hoa, chiều dài cánh hoa và chiều rộng cánh hoa.

Do đó, chúng ta sẽ có bốn tham số chính tương ứng với một mô hình dữ liệu 4D được minh họa theo **Bảng 1**.

**Hình 1:** Hình ảnh minh họa cho tập dữ liệu Iris.

Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
4	4.6	3.1	1.5	0.2	Iris-setosa
5	5	3.6	1.4	0.2	Iris-setosa
6	5.4	3.9	1.7	0.4	Iris-setosa
7	4.6	3.4	1.4	0.3	Iris-setosa
8	5	3.4	1.5	0.2	Iris-setosa
9	4.4	2.9	1.4	0.2	Iris-setosa
10	4.9	3.1	1.5	0.1	Iris-setosa
11	5.4	3.7	1.5	0.2	Iris-setosa
12	4.8	3.4	1.6	0.2	Iris-setosa
13	4.8	3	1.4	0.1	Iris-setosa
14	4.3	3	1.1	0.1	Iris-setosa
15	5.8	4	1.2	0.2	Iris-setosa
16	5.7	4.4	1.5	0.4	Iris-setosa
17	5.4	3.9	1.3	0.4	Iris-setosa
18	5.1	3.5	1.4	0.3	Iris-setosa
19	5.7	3.8	1.7	0.3	Iris-setosa

**Bảng 1:** Các mẫu đầu tiên trong tập dữ liệu hoa Iris bao gồm 19 mẫu đầu tiên với 4 đặc trưng của ba loài hoa khác nhau, bao gồm chiều dài đài hoa, chiều rộng đài hoa, chiều dài cánh hoa và chiều rộng cánh hoa.

## 5.2 SOURCE CODE NHÓM PHÁT TRIỂN

Nhóm đã phát triển và chạy thành công thuật toán Phân tích thành phần chính trên tập dữ liệu về loài hoa Iris nhằm giảm chiều dữ liệu từ 4 về còn lại 2 chiều dữ liệu. Sau đây là [đường link đi đến Google Colabotary](#) của nhóm đã thực nghiệm.

Toàn bộ hệ thống source code được biểu diễn như sau:

Tên hàm	Miêu tả hàm
<pre>import numpy as np import pandas as pd import seaborn as sns from matplotlib import pyplot as plt</pre>	<ul style="list-style-type: none"> <li>- Cập nhật một số thư viện cần thiết cho chương trình:             <ul style="list-style-type: none"> <li>- <i>numpy</i>: một thư viện Python dùng để làm việc với mảng. Thư viện này cũng có các chức năng để làm việc trong miền đại số tuyến tính, biến đổi phạm vi và ma trận.</li> <li>- <i>pandas</i>: một công cụ thao tác và phân tích dữ liệu nguồn mở nhanh, mạnh mẽ, linh hoạt và dễ sử dụng, được xây dựng dựa trên ngôn ngữ lập trình Python.</li> <li>- <i>matplotlib</i>: một thư viện toàn diện để tạo trực quan hóa tĩnh, hoạt hình và tương tác trong Python</li> <li>- <i>seaborn</i>: một thư viện trực quan hóa dữ liệu Python dựa trên matplotlib. Nó cung cấp một giao diện cấp cao để vẽ đồ họa thống kê hấp dẫn và nhiều thông tin.</li> </ul> </li> </ul>

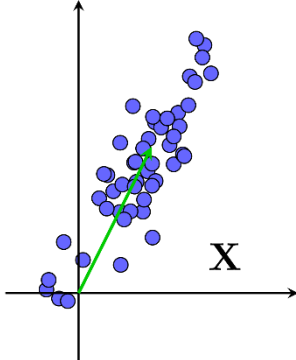
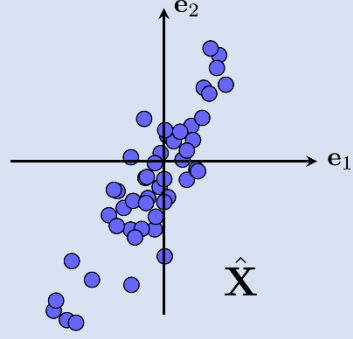
### Xử lý dữ liệu để thực nghiệm thuật toán PCA

<pre># step 1 df = pd.read_csv('Iris.csv')</pre>	<ul style="list-style-type: none"> <li>- Đầu tiên tại step 1: chúng ta sử dụng hàm <code>read_csv()</code> từ thư viện Pandas để tải dữ liệu từ tệp CSV vào đối tượng Pandas DataFrame có tên là <code>df</code>.</li> <li>- <b>Input</b>: tệp CSV có tên "Iris.csv" chứa tập dữ liệu Iris</li> <li>- <b>Output</b>: một khung dữ liệu gấu trúc có tên là "df" chứa dữ liệu từ tệp CSV.</li> </ul>
--	--

Tên hàm	Miêu tả hàm
<pre># step 2 x = df[["SepalLengthCm", "SepalWidthCm",         "PetalLengthCm", "PetalWidthCm"]].to_numpy()</pre>	<ul style="list-style-type: none"> <li>- Đến với bước step 2 tiếp theo, bước này trích xuất các giá trị của bốn cột - độ dài đài hoa, độ rộng đài hoa, độ dài cánh hoa và độ rộng cánh hoa; từ khung dữ liệu gấu trúc "df" và lưu trữ chúng trong một mảng gọn gàng có tên là "X". Bước này tạo mảng 2 chiều có 150 hàng và 4 cột.</li> <li>- <b>Input:</b> khung dữ liệu gấu trúc "df" chứa bộ dữ liệu Iris.</li> <li>- <b>Output:</b> một mảng gọn gàng có tên là "X" chứa các giá trị của bốn cột giá trị cho tất cả 150 quan sát trong tập dữ liệu.</li> </ul>
<pre># step 3 list_label = df["Species"].to_numpy()</pre>	<ul style="list-style-type: none"> <li>- Cuối cùng tại step 3, chương trình thực hiện việc này trích xuất các giá trị của cột "Loài" từ khung dữ liệu gấu trúc "df" và lưu trữ chúng trong một mảng gọn gàng có tên là "list_label". Mảng này chứa các nhãn loài cho mỗi trong số 150 quan sát trong tập dữ liệu. Có ba loài trong tập dữ liệu: Iris-setosa, Iris-versicolor và Iris-virginica.</li> <li>- <b>Input:</b> tệp CSV có tên "Iris.csv" chứa tập dữ liệu Iris.</li> <li>- <b>Output:</b> một mảng gọn gàng có tên là "list_label" chứa các nhãn loài cho mỗi trong số 150 quan sát trong bộ dữ liệu.</li> </ul>

## Xây dựng lớp PCA

<pre># step 0 self.n_dimention = n_dimention</pre>	<ul style="list-style-type: none"> <li>- Đầu tiên tại bước khởi tạo step 0, chuẩn bị dữ liệu cần giảm chiều là X với kích thước (n_sample, n_feature), tương ứng mỗi hàng là 1 mẫu dữ liệu có n_feature thuộc tính.</li> </ul>
<pre># step 1 mean = np.mean(X, axis=0)</pre>	<ul style="list-style-type: none"> <li>- Đến với step 1, chương trình tính toán mean vector được sử dụng hàm thư viện Numpy để tính mean vector, với thông số đầu vào gồm:             <ol style="list-style-type: none"> <li>a) ma trận X gồm ba đặc trưng của bộ dataset iris</li> <li>b) axis = 0 để tính mean vector theo ma trận cột</li> </ol> </li> </ul>

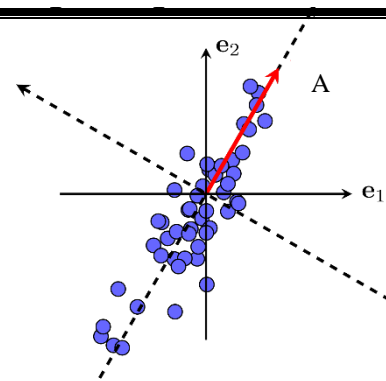
Tên hàm	Miêu tả hàm
	
<pre># step 2 X = X - mean</pre>	<p>- Tiếp theo tại step 2, chương trình sẽ trừ mỗi điểm dữ liệu cho vector kỳ vọng: <math>X_k = X_k - X_{mean}</math> với <math>k = 1, 2, \dots, n\_sample</math> và <math>X_{mean}</math> là vector trung bình của tất cả các điểm dữ liệu.</p> 
<pre># step 3 cov = X.T.dot(X) / X.shape[0]</pre>	<p>- Tại step 3, biểu thị cho việc tính ma trận hiệp phương sai với công thức như sau:</p> $S = \frac{1}{n - sample} * X^T * X$
<pre># step 4 eigen_values, eigen_vectors, = np.linalg.eig(cov)</pre>	<p>- Tính toán tìm trị riêng, vector riêng của ma trận S tương ứng với <math>(\lambda_1, u_1), (\lambda_2, u_2), \dots, (\lambda_D, u_D)</math>.</p>
<pre># step 5 select_index = np.argsort(eigen_values) [:::-1][:self.n_dimension]</pre>	<p>- Sắp xếp chúng theo thứ tự giảm dần của trị riêng.</p>

## Tên hàm

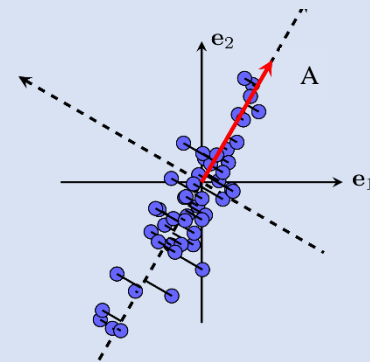
## Miêu tả hàm

```
# step 6
A = eigen_vectors[:, select_index]
```

```
# step 7
Y = X.dot(A)
```



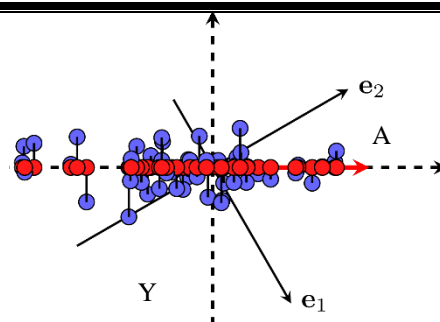
- Chọn K vector riêng ứng với K trị riêng lớn nhất để xây dựng ma trận A có các cột tạo thành một hệ trực giao.
- K vectors này, còn được gọi là các thành phần chính, tạo thành một không gian con gần với phân bố của dữ liệu ban đầu đã chuẩn hoá.



- Sắp ánh xạ không gian ban đầu sang không gian k chiều:  $Y = X * A$
- Y là dữ liệu mới chính bằng tọa độ của các điểm dữ liệu trên không gian mới mà chúng ta cần tìm.
- Tại đây nhân này, ta có thể hiểu rằng việc nhân này là lấy từng mẫu dữ liệu nhân với từng vector riêng, khi đó mỗi mẫu dữ liệu ban đầu sẽ được nhân với k vector nên sẽ có k chiều.

## Tên hàm

## Miêu tả hàm



## In kết quả biểu đồ được ứng dụng thuật toán PCA nhằm giảm số chiều

```
# step 1:
pca = PCA(n_dimension=2)
```

- Việc đầu tiên cần thực hiện thuật toán Phân tích thành phần chính là lớp PCA đã xác định trước đó, với số chiều là một số nguyên được đặt thành 2. Điều này làm sẽ giảm số chiều của dữ liệu đầu vào X từ 3 xuống còn 2.
- **Input:** một số nguyên “n\_dimension” đại diện cho số thứ nguyên mà thuật toán PCA sẽ chiếu dữ liệu lên.
- **Output:** một biến đại diện cho lớp PCA, có thể được sử dụng để điều chỉnh và biến đổi dữ liệu bằng PCA.

```
# step 2:
Y = pca.fit_transform(X)
```

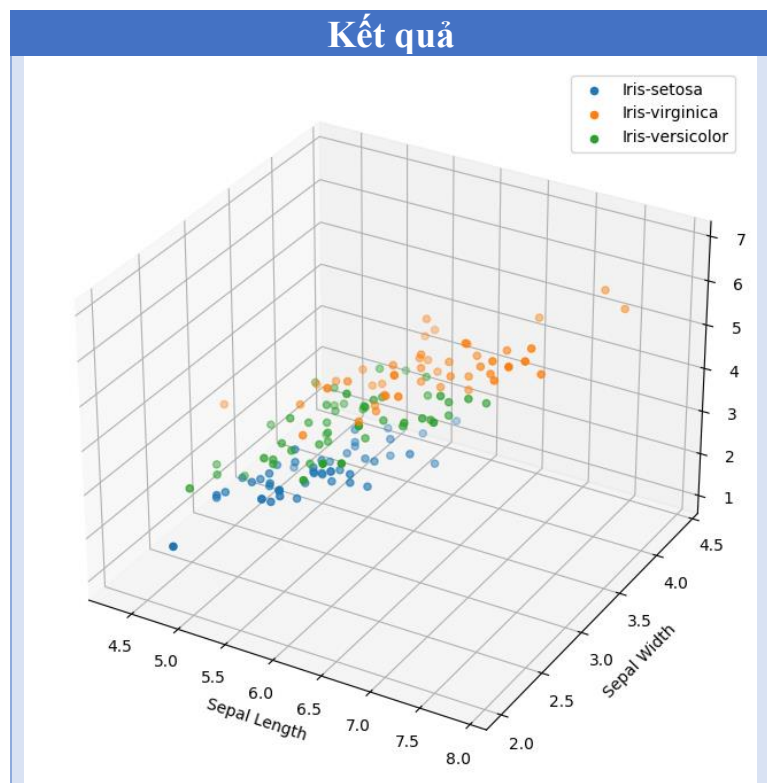
- Áp dụng giảm kích thước cho dữ liệu đầu vào X và trả về tập dữ liệu mới new\_X với chỉ còn 2 chiều.
- Hàm fit\_transform của lớp PCA tính toán các thành phần chính bằng cách thực hiện phân tách giá trị riêng của ma trận hiệp phương sai, các bước thực hiện thì như chúng ta đã đi qua chi tiết bên trên bước 5.
- **Input:** một mảng NumPy “X” chứa các giá trị đặc trưng của bộ dữ liệu Iris.
- **Output:** một mảng NumPy “Y” chứa dữ liệu đã chuyển đổi, trong đó các giá trị tính năng ban đầu đã được chiếu lên các thành phần chính do thuật toán PCA tìm thấy.



Tên hàm	Miêu tả hàm
<pre># step 3: df_new = pd.DataFrame(data=Y, columns=["PC1", "PC2"])</pre>	<ul style="list-style-type: none"> <li>- Tạo một khung dữ liệu Pandas DataFrame mới, đặt tên là <code>df_new</code> với đầu vào là dữ liệu vừa thực thi tại bước 2, các cột gồm 2 giá trị tương ứng với biểu đồ 2 chiều, PC1 và PC2.</li> <li>- <b>Input:</b> một mảng NumPy “Y” chứa dữ liệu đã chuyển đổi được trả về từ thuật toán PCA.</li> <li>- <b>Output:</b> một Pandas DataFrame “df_new” có hai cột có tên là “PC1” và “PC2”, chứa dữ liệu được chuyển đổi từ hai thành phần chính đầu tiên.</li> </ul>
<pre># step 4: df_new["Species"] = list_label  plt.figure(figsize=(10, 8))</pre>	<ul style="list-style-type: none"> <li>- Tại bước 3, chúng ta đã có được 2 thành phần gồm PC1 và PC2, tại bước 4 này, ta sẽ thêm một cột mới vào khung dữ liệu <code>df_new</code> với biến mục tiêu Y, chứa nhãn loài của từng mẫu trong X.</li> <li>- <b>Input:</b> một khung dữ liệu Pandas “df_new” chứa dữ liệu được chuyển đổi và hai cột có tên “PC1” và “PC2”.</li> <li>- <b>Output:</b> một biểu đồ phân tán cho thấy sự phân bố của các điểm dữ liệu được chuyển đổi trong không gian hai chiều được xác định bởi hai thành phần chính đầu tiên.</li> </ul>
<pre># step 5: sns.scatterplot(x="PC1", y="PC2", hue="Species", data=df_new)</pre>	<ul style="list-style-type: none"> <li>- Tạo một scatter plot bằng thư viện Seaborn, trong đó: <ul style="list-style-type: none"> <li>a) x và y: đại diện cho hai thành phần chính</li> <li>b) hue: đại diện cho biến mục tiêu (Species)</li> <li>c) data: là khung dữ liệu chứa dữ liệu chính là <code>df_new</code> tại step 4</li> </ul> </li> <li>- <b>Input:</b> một khung dữ liệu Pandas “df_new” chứa dữ liệu được chuyển đổi và hai cột có tên “PC1” và “PC2” và một cột có tên “Species” chỉ định nhãn loài cho từng điểm dữ liệu.</li> <li>- <b>Output:</b> một biểu đồ phân tán cho thấy sự phân bố của các điểm dữ liệu được biến đổi trong không gian hai chiều được xác định bởi hai thành phần chính đầu tiên. Trục x biểu thị các giá trị của thành phần chính đầu tiên (PC1), trục y biểu thị các giá trị của thành phần chính thứ hai (PC2) và màu của từng điểm dữ liệu được xác định bởi nhãn loài tương ứng của nó.</li> </ul>

## 5.3 KẾT QUẢ THỰC NGHIỆM

### 5.2.1 Biểu diễn đồ thị với ba thành phần chính

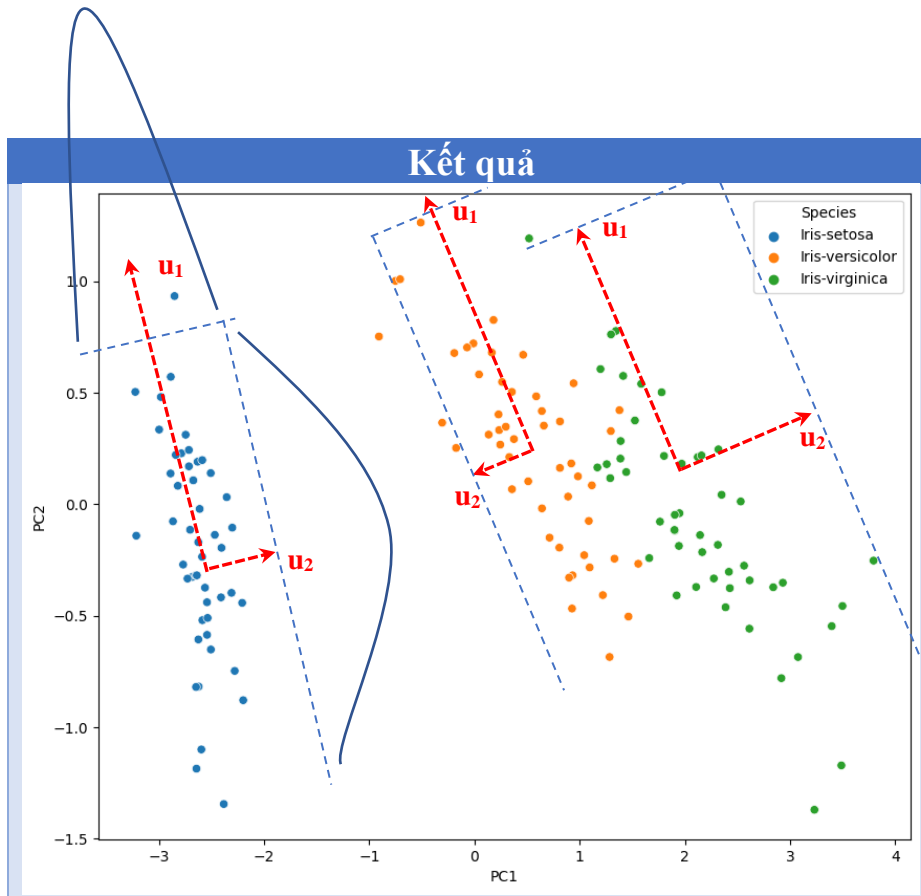


**Bảng 2:** Kết quả thực nghiệm của chương trình, một biểu đồ 3 chiều sử dụng 3 thành phần trong tập dữ liệu dùng để biểu thị mức độ phân tán của từng bộ dữ liệu trong tập dữ liệu Iris gốc.

Dựa vào **Bảng 2**, ban đầu nhóm đã thực hiện việc biểu diễn đồ thị dựa vào ba giá trị của tập dữ liệu gồm chiều dài đài hoa, chiều rộng đài hoa và chiều dài cánh hoa.

Nhìn vào biểu đồ, các tập điểm dữ liệu của 3 loài hoa Iris này phân bố hầu như khá trùng lẫn nhau và khó cho người dùng phân tích dựa trên biểu đồ 3 chiều này.

### 5.2.2 Biểu diễn đồ thị với hai thành phần chính khi áp dụng thuật toán PCA



Bằng cách áp dụng thuật toán Phân tích thành phần chính nhóm đã cài đặt cho chương trình, chương trình đã thể hiện được một biểu đồ kết quả qua **Bảng 3** hiển thị phân phối các điểm dữ liệu trong hai thành phần chính với các màu khác nhau đại diện cho từng loài Iris sau khi được giảm số chiều của dữ liệu từ 4 xuống còn lại 2 chiều.

**Bảng 3:** PCA 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.

### 5.2.3 Đánh giá kết quả

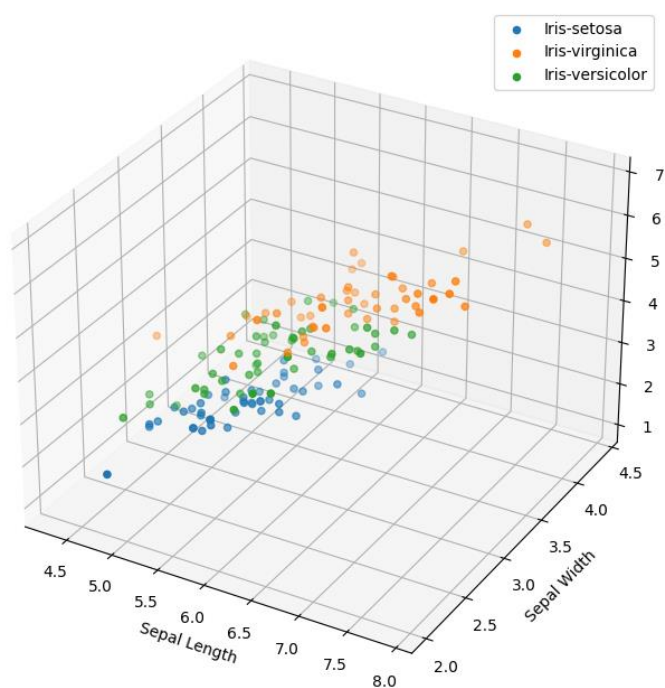
Dựa vào biểu đồ thể hiện trong **Bảng 4**, ta cho thấy sự tách biệt rõ ràng giữa ba loài Iris: Setosa, Versicolor và Virginica. Loài Setosa tách biệt rõ ràng với hai loài còn lại, tạo thành một cụm riêng biệt ở góc dưới bên trái của ô. Mặt khác, các điểm dữ liệu của hai loài hoa Versicolor và Virginica lại phân bố khá trùng lặp với nhau, nhưng vẫn tạo thành hai cụm riêng biệt ở góc trên bên phải của ô. Điều này có thể đưa ra một kết luận rằng 2 loài hoa Versicolor và Virginica có đặc điểm tương đồng với nhau về ngoại hình (đài hoa và cánh hoa), trong khi đó loài hoa Setosa lại có đặc điểm tương đối khác nhau về ngoại hình với hai loài hoa còn lại.

Hơn nữa, biểu đồ thể hiện rằng thành phần chính đầu tiên (PC1) quan trọng hơn thành phần chính thứ hai (PC2) trong việc phân biệt các loài Iris. PC1 có thể nắm bắt khoảng 92% tổng số biến thể trong dữ liệu, trong khi PC2 chỉ chiếm khoảng 6%. Điều này chỉ ra rằng phần lớn thông tin trong dữ liệu chỉ có thể được giải thích bởi PC1.

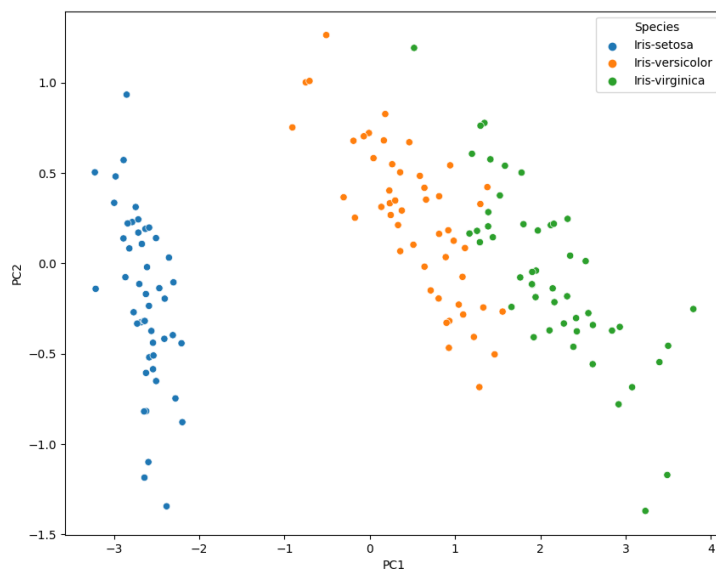
Nhìn chung, thuật toán PCA đã giảm kích thước của bộ dữ liệu Iris một cách hiệu quả trong khi vẫn giữ được các tính năng quan trọng nhất, cho phép phân tích dữ liệu hiệu quả hơn. Biểu đồ thu được cho thấy rõ ràng sự phân bố và phân tách của các loài Iris khác nhau trong không gian hai chiều.

Xét về thời gian chương trình thực nghiệm thuật toán PCA, với sự hỗ trợ từ những thư viện Numpy, Pandas, Seaborn và Matplotlib nên cho thời gian thuật toán chạy với tốc độ rất tốt.

Đồ thị với ba thành phần chính gốc



Đồ thị với hai thành phần chính sau khi áp dụng PCA



Bảng 4: PCA dưới góc nhìn Thống kê.

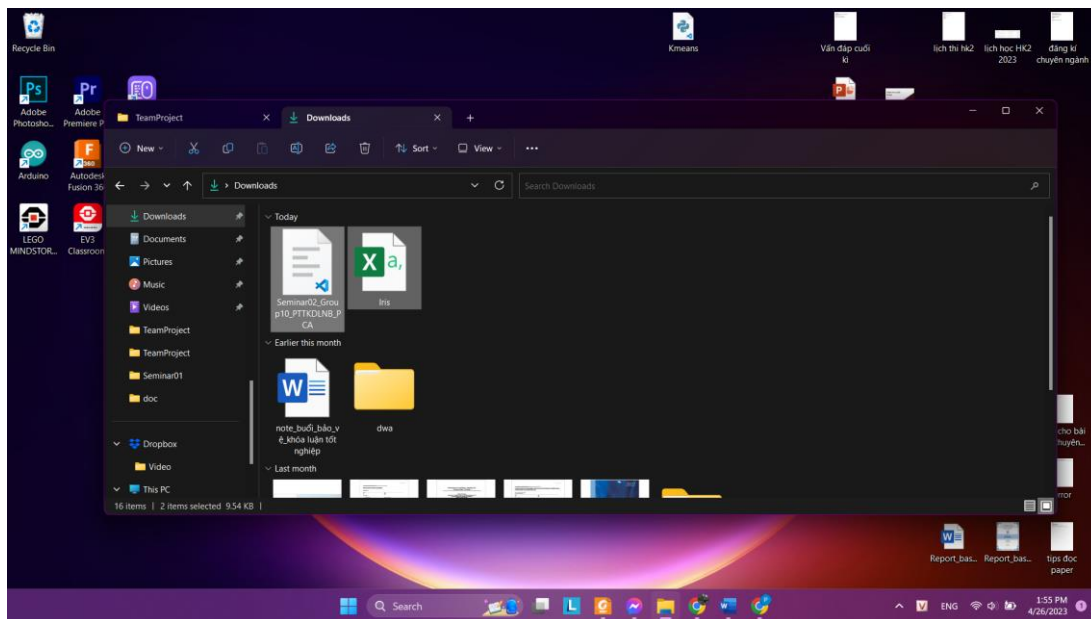
## 5.4 HƯỚNG DẪN SỬ DỤNG SOURCE CODE

Với chương trình mà nhóm đã phát triển, source code được phát triển và thực nghiệm dựa trên bộ tập dữ liệu về loài hoa Iris đã được miêu tả chi tiết tại mục [5.1]. Để có được tập dữ liệu này, người dùng có thể tìm kiếm trên Internet vì đây là tập dữ liệu mở open source hoặc nhóm cũng đã chuẩn bị sẵn tập dữ liệu này.

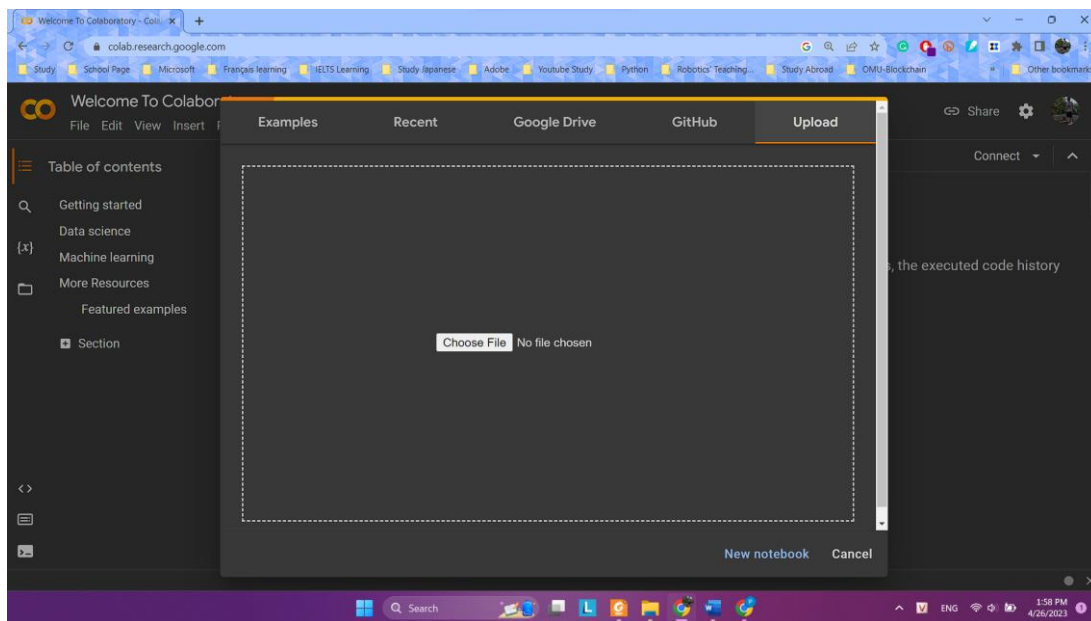
Nhóm cũng đã chuẩn bị sẵn quay một video nhằm ghi lại việc tải về tập dữ liệu thực nghiệm và source code chương trình này nhằm giúp cho người dùng dễ dàng tiếp cận với cách chạy source code của nhóm. Sau đây là đường link đến với [video demo của nhóm](#).

Sau đây là các bước miêu tả chi tiết hướng dẫn sử dụng source code đồ án Phân tích thành phần chính của nhóm.

Bước 1: Nhóm đã sử dụng trình duyệt Google Colabotary để chạy source code, và sau đây là đường link đến với [source code PCA](#) và đường link đến [tập dữ liệu Iris.csv](#). Người dùng tải source code và tập dữ liệu thực nghiệm về PC.



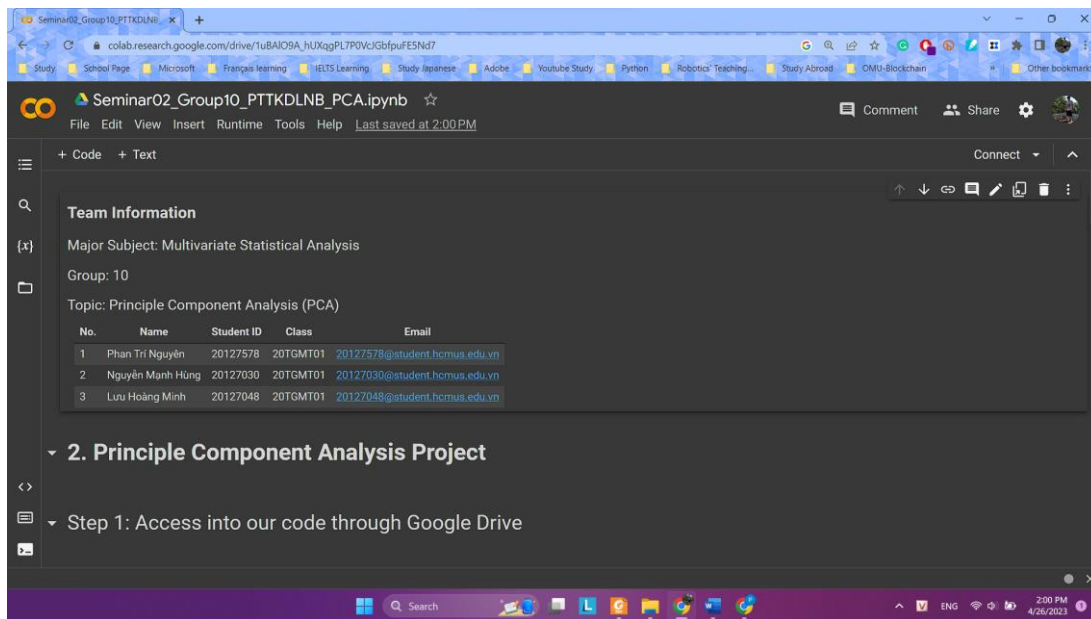
Bước 2: Sau khi đã có được source code và tập dữ liệu Iris.csv. Người dùng tải source code này lên trên Google Colabotary để chạy chương trình như sau. Mở trình duyệt [Google Colabotary](https://colab.research.google.com), sau chọn **Upload** để tải source code PCA lên



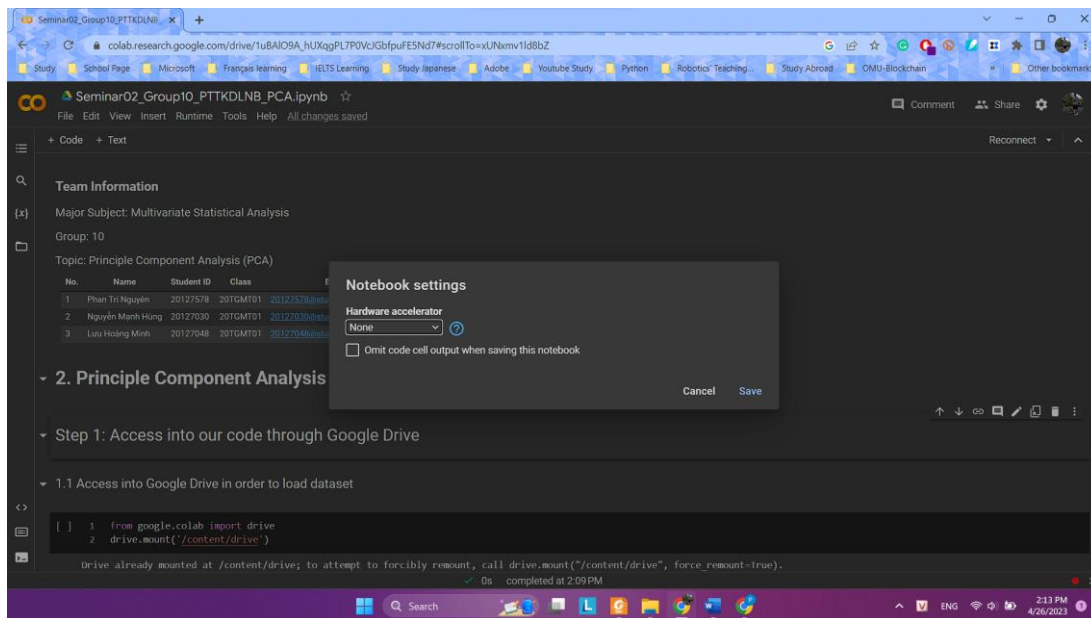
Tiếp theo nhấn vào khung **Choose File**, một Folder file sẽ hiện lên, người dùng chọn vào tệp Seminar02\_Group10\_PTTKDLNB\_PCA chính là source code của nhóm, chọn Open và đợi cho Google Colab mở source code.

Nếu như màn hình người dùng hiển thị ra source code như hình sau, tức là người dùng đã mở thành công chương trình của nhóm.



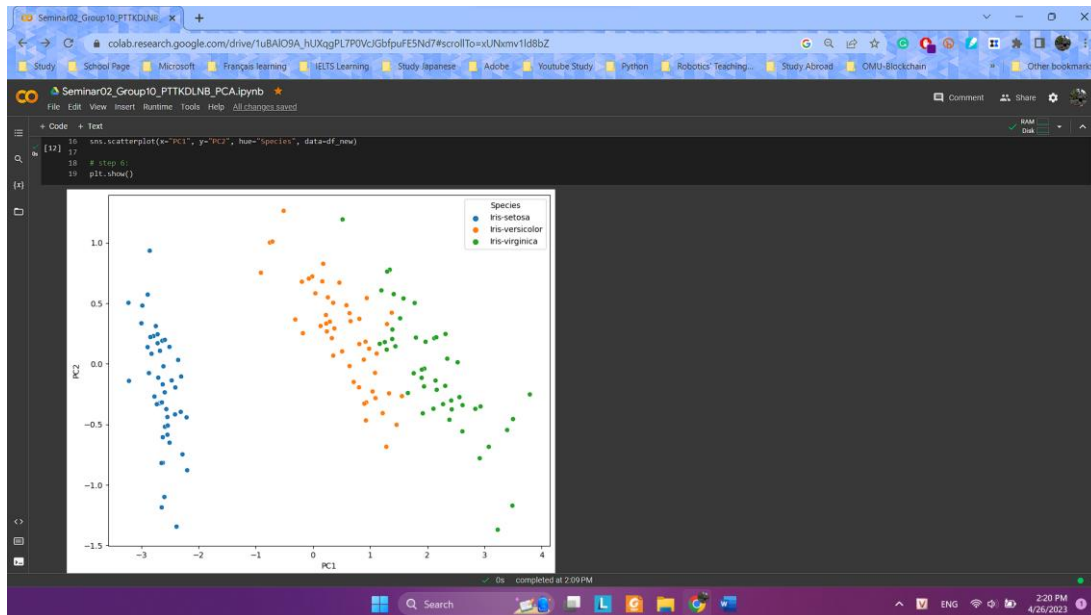


Bước 3: Tiếp theo người dùng có thể chọn vào phần cứng mặc định hoặc GPU đều được vì chương trình nhóm đã phát triển không tiêu tốn quá nhiều tài nguyên bộ nhớ xử lý, tại đây nhóm sẽ demo định dạng phần cứng mặc định của Google Colab để không tiêu tốn dung lượng đáng kể.



Bước 4: Đến bước 4, người dùng có thể chạy từng dòng lệnh code trong chương trình hoặc chọn vào mục **Runtime** => **Run all** để chạy toàn bộ. Một điều lưu ý rằng vì chương trình cần tập dữ liệu thực nghiệm, vì vậy dataset sẽ được tải về từ link github nhóm đã đăng từ trước, người dùng sẽ được yêu cầu cho phép Google Colab liên kết với Google Drive nhằm tải tập dữ liệu cho chương trình thực nghiệm thuật toán Phân tích thành phần chính.

Bước 5: Khi chương trình chạy ra đến kết quả cuối cùng như màn hình demo chứng tỏ người dùng đã thực thi thành công chương trình demo thuật toán PCA.



# 6

## KẾT LUẬN

Kết luận lại quá trình nghiên cứu, Phân tích thành phần chính (Principle Component Analysis-PCA) là một kỹ thuật rất hữu ích trong việc giảm số chiều dữ liệu và phân tích dữ liệu đa biến.

Vấn đề ban đầu của PCA là tìm cách biến đổi dữ liệu ban đầu thành các thành phần chính độc lập với nhau và giảm số chiều dữ liệu một cách tối đa. Với hai phương pháp tính toán PCA, bao gồm cách đại số và cách hình học, PCA cho phép ta giảm kích thước của dữ liệu trong không gian nhiều chiều, giữ lại những thông tin quan trọng nhất và loại bỏ những thông tin không quan trọng.

Nguyên tắc giải quyết vấn đề của PCA là tìm ra các hệ số trọng số sao cho tổng phương sai giữ lại của dữ liệu sau khi giảm số chiều là lớn nhất. Thực hiện PCA đòi hỏi các bước tiền xử lý dữ liệu, tính toán ma trận hiệp phương sai và giá trị riêng của nó, tìm các thành phần chính và xác định chiều của không gian mới để giảm số chiều của dữ liệu.

Cuối cùng nhóm đã học hỏi và thực hiện áp dụng thành công PCA vào một tập dataset về các bông hoa Iris giúp giảm chiều dữ liệu và giữ lại được những thông tin giúp chúng ta vẫn có thể nhận biết được, từ đó giúp người trồng đưa ra kết luận về cấu trúc phát triển của từng loại hoa sẽ phát triển tốt ở khu vực nào và trồng từng loại phù hợp với điều kiện sinh sống của chúng.



## 7

## TÀI LIỆU THAM KHẢO

- [1] Ritik Arora, Applying PCA on MNIST dataset [May 13, 2022]
- [2] Sebastian Raschka, Principal Component Analysis in 3 Simple Steps [Jan 27, 2015]
- [3] Principal Component Analysis (PCA) in Python Tutorial [Jan 2020]
- [4] Tuong Nguyen Huy, ML From Scratch: PCA . Data Dimension Reduction Algorithm by [Oct 12, 2022]
- [5] Richard A. Johnson, Dean W. Wichern, Applied Multivariate Statistical Analysis 6th Edition [April 02, 2007]
- [6] Wolfgang Karl Härdle, Léopold Simar, Applied Multivariate Statistical Analysis, 3rd Edition
- [7] Edouard Duchesnay, Tommy Löfstedt, Statistics and Machine Learning in Python Release 0.1 [Feb 03, 2017]
- [8] Sasan Karamizadeh, Shahidan M. Abdullah, Azizah A. Manaf, Mazdak Zamani, Alireza Hooman, An Overview of Principal Component Analysis by [May, 2013]
- [9] Sidharth Prasad Mishra, Uttam Sarkar, Subhash Taraphder, Sanjay Datta, Devi Prasanna Swain, Reshma Saikhom, Sasmita Panda and Menalsh Laishram, Multivariate Statistical Data Analysis- Principal Component Analysis (PCA) [May 01, 2017]
- [10] Principal Component Analysis Part 1: The Different Formulations. | by Aadithya Sankar | Towards Data Science
- [11] Frieder Heinrich Baumann, Recent Advances in VLSI Characterization using the TEM [August 2019]
- [12] Principal Component Analysis Eliminate Noise In The Data – Cross Validated
- [13] Kai Zhao, Feature Extraction using Principal Component Analysis — A Simplified Visual Demo | Towards Data Science
- [14] Feature extraction using PCA - Computer vision for dummies
- [15] Etika Kartikadarma, Sari Wijayanti, Sari Ayu Wulandari, Fauzi Adi Rafrastara, Principle Component Analysis for Classification of the Quality of Aromatic Rice [14 Sep 2020] published in International Journal of Computer Science and Information Security (IJCSIS)