

Học máy không giám sát

Nguyễn Thanh Tùng
Khoa Công nghệ thông tin – Đại học Thủ Lợi
tungnt@tlu.edu.vn

Bài giảng có sử dụng hình vẽ trong cuốn sách “An Introduction to Statistical Learning with Applications in R” với sự
cho phép của tác giả, có sử dụng slides các khóa học CME250 của ĐH Stanford và IOM530 của ĐH Southern California

Học máy không giám sát

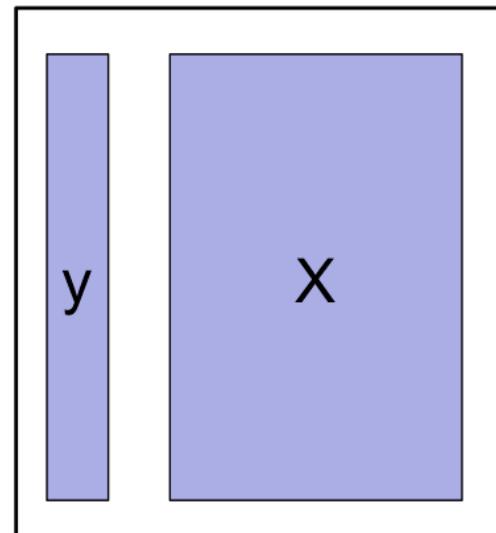
- *Học không giám sát*: tập các công cụ thống kê xử lý dữ liệu chỉ có biến đầu vào, không có biến đích
 - Ta chỉ có X's mà không có các nhãn Y
 - Mục tiêu: phát hiện các mẫu/các đặc tính của dữ liệu
 - vd. trực quan hóa hoặc diễn giải dữ liệu nhiều chiều

Học có giám sát vs. không giám sát

Học máy có giám sát: cả X và Y đều đã biết

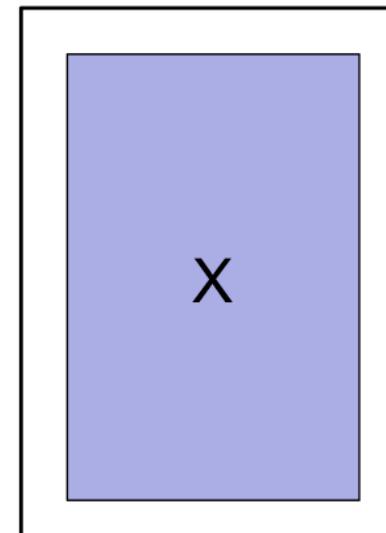
Học máy không giám sát: chỉ biết X

(a)



Học có giám sát

(b)



Học không giám sát

Học không giám sát

- Ví dụ ứng dụng:
 - Biết các mô ung thư của n bệnh nhân bị ung thư vú, cần xác định các nhóm nhỏ (subtypes) chưa biết gây nên ung thư vú
 - Các thí nghiệm về biểu diễn Gen chứa hàng ngàn biến

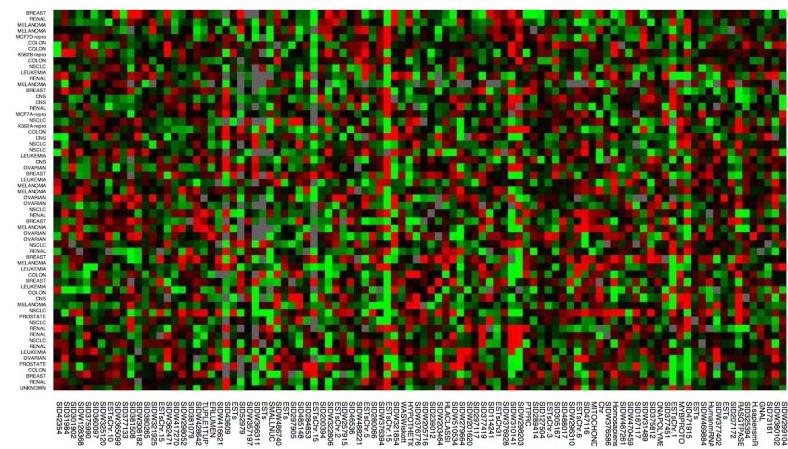
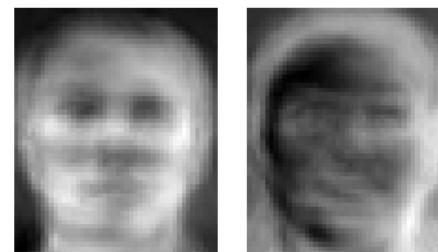


Figure 1.3, ESL

Học không giám sát

- Ví dụ ứng dụng:

- Cho một tập các tài liệu văn bản, cần xác định tập các tài liệu có chung chủ đề như thể thao, chính trị, ca nhạc...
- Cho các ảnh khuôn mặt có số chiều cao, tìm một biểu diễn đơn giản/thu gọn của các ảnh này để đưa vào bộ phân lớp nhận dạng khuôn mặt



(AT&T Laboratories
Cambridge)

Học không giám sát

- Tại sao học không giám sát luôn thách thức lớn?
 - Phân tích khám phá dữ liệu (Exploratory data analysis) – mục tiêu không được định nghĩa rõ ràng
 - Khó đánh giá hiệu năng – không biết được đáp án đúng (“right answer” unknown)
 - Xử lý dữ liệu với số chiều lớn

Sắp xếp các đối tượng vào các nhóm



Có bao nhiêu clusters



Có bao nhiêu clusters?



2 clusters



4 clusters



6 clusters

Tuỳ thuộc vào “resolution” !

Học không giám sát

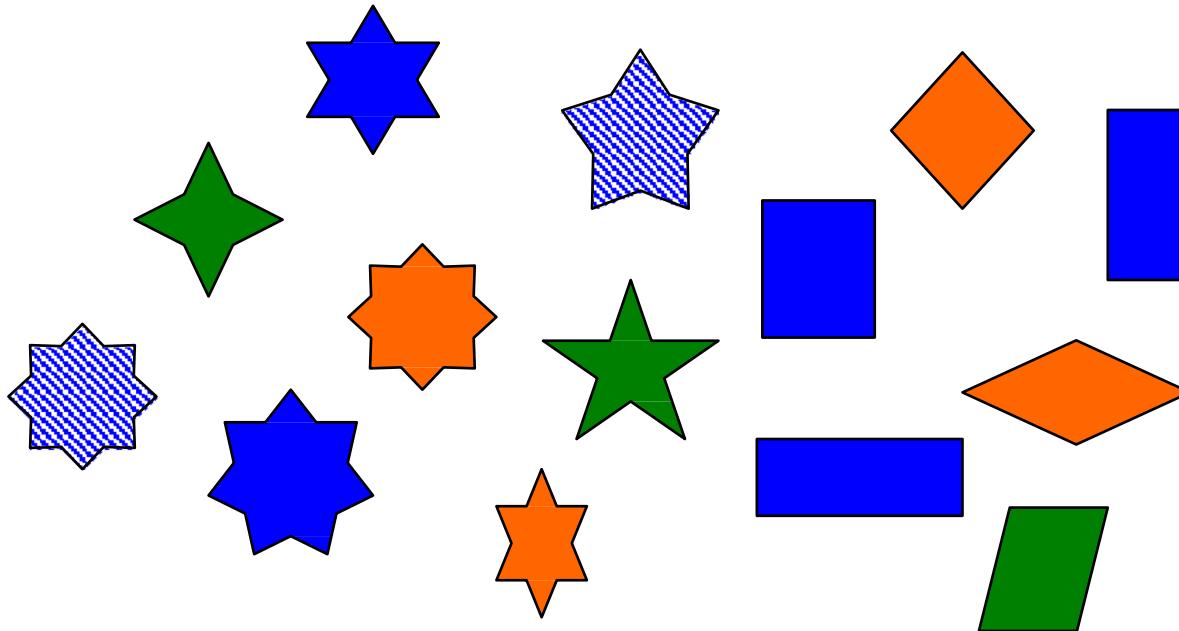
- Hai cách tiếp cận:
 - *Phân tích cụm (Cluster analysis)*
 - Xác định các nhóm mẫu đồng nhất (có các đặc tính chung)
 - *Giảm chiều dữ liệu (Dimensionality Reduction)*
 - Tìm cách biểu diễn với số chiều thấp hơn dựa trên tính chất và trực quan hóa dữ liệu

Phân tích cụm & K-means

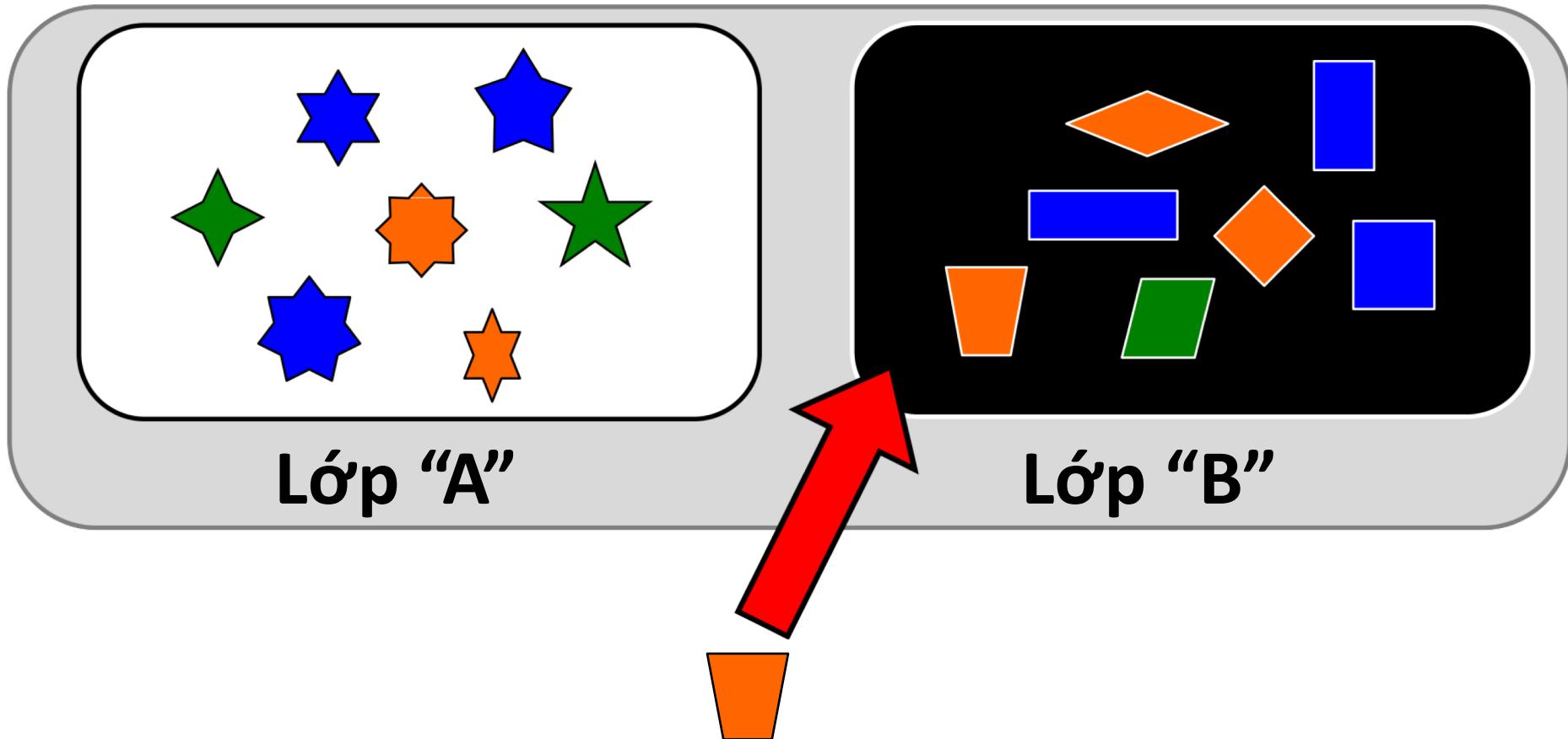
Phân cụm

- *Phân cụm*: là tập các phương pháp nhằm tìm ra các nhóm con trong dữ liệu
 - Các mẫu có đặc điểm chung trong cùng 1 nhóm nhưng khác với các mẫu ở ngoài nhóm
 - Việc gom nhóm là phân tích cấu trúc dữ liệu nội tại, điều này khác với phân lớp

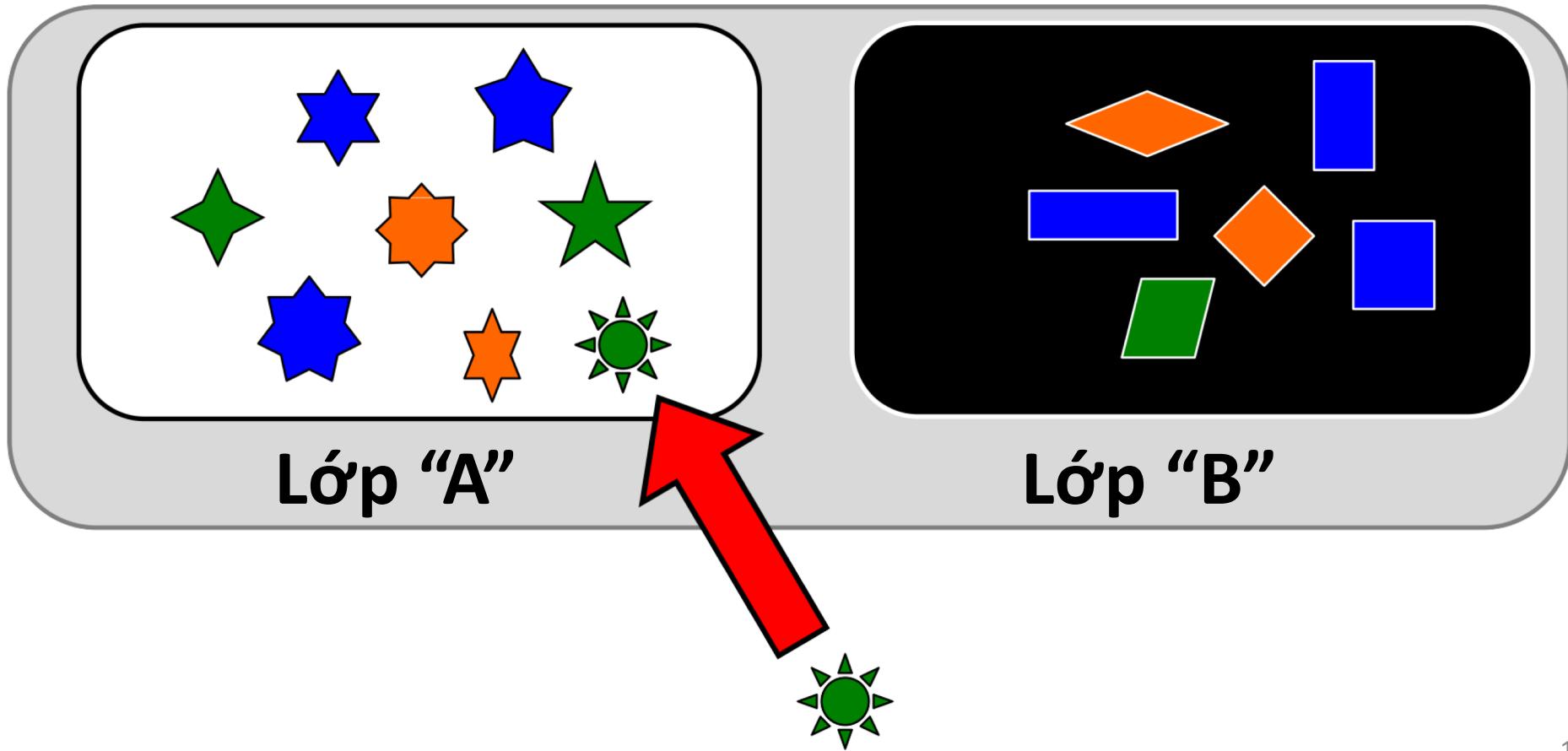
Phân cụm vs. Phân lớp



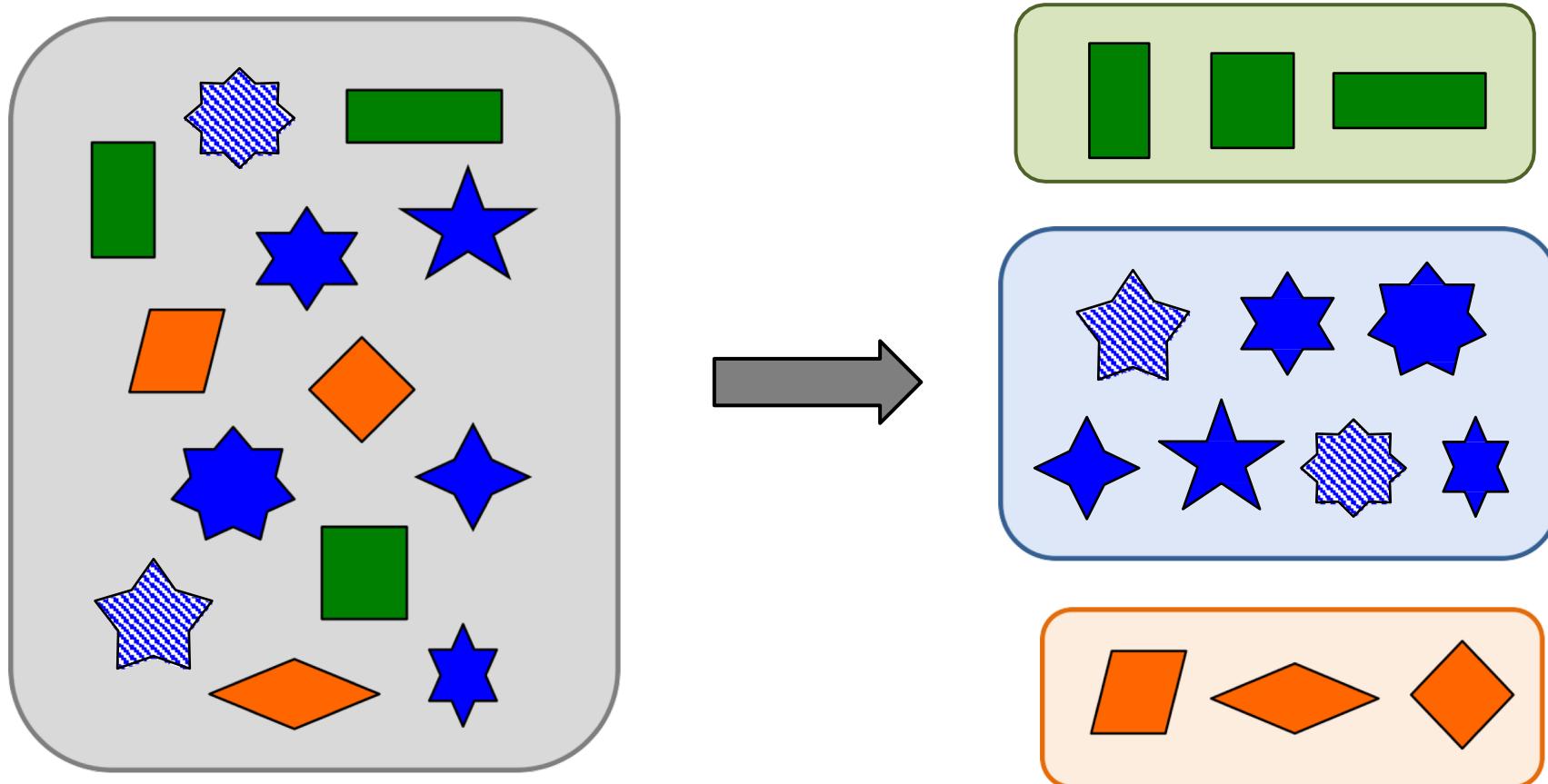
Phân lớp



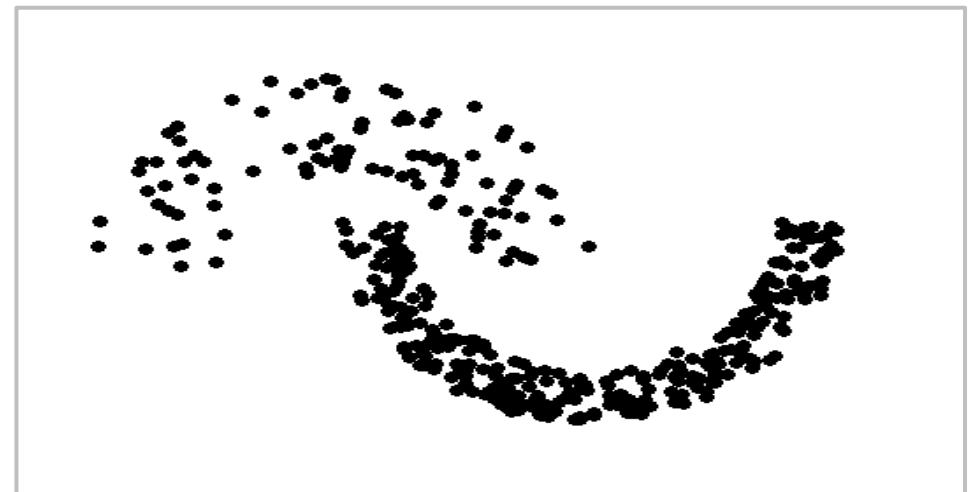
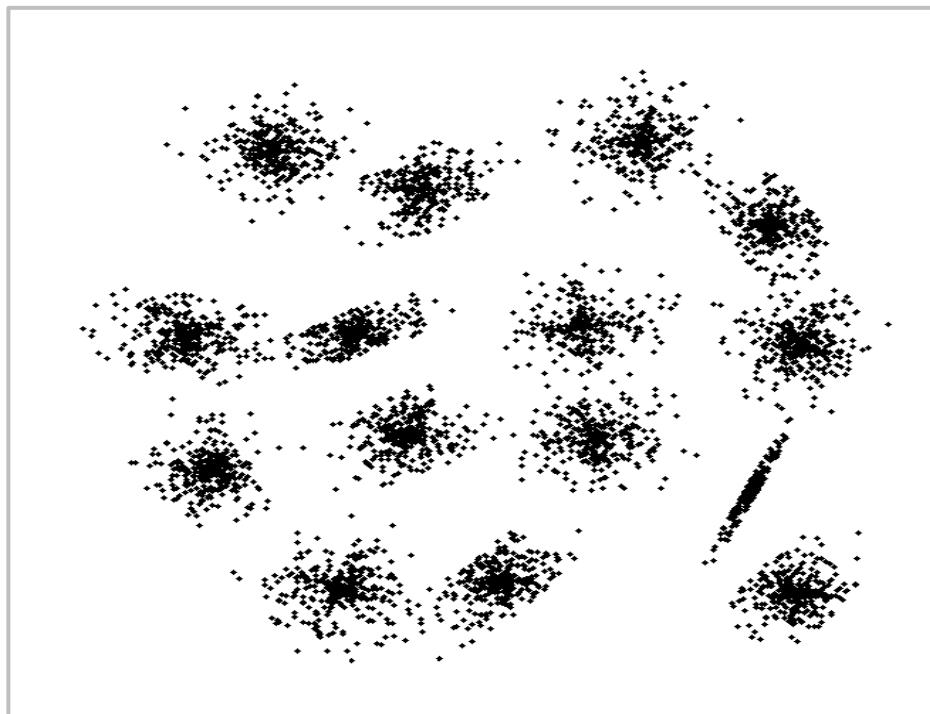
Phân lớp



Phân cụm

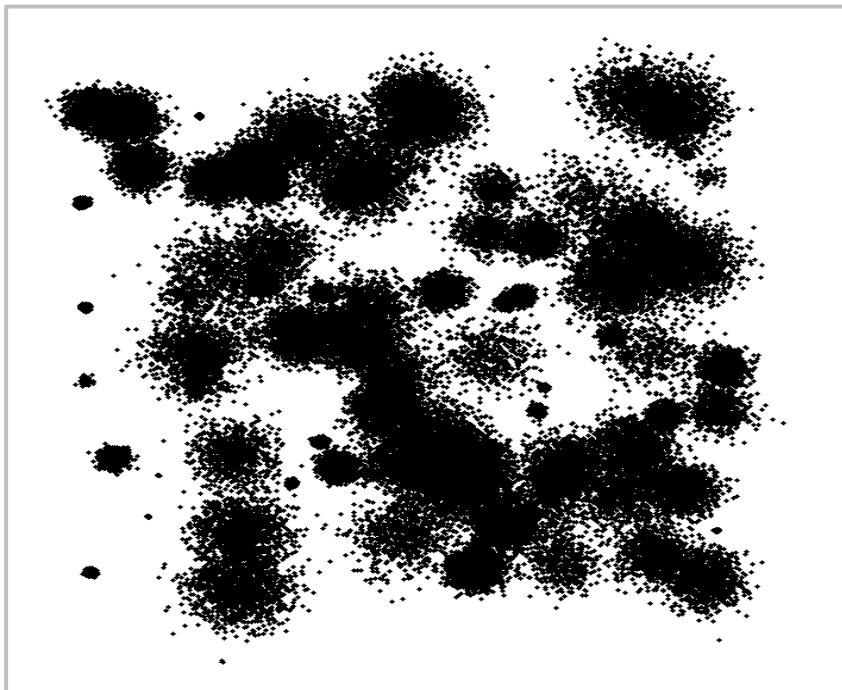


Phân cụm



Dữ liệu lấy từ: <http://cs.joensuu.fi/sipu/datasets/>

Phân cụm



Dữ liệu lấy từ: <http://cs.joensuu.fi/sipu/datasets/>

Phân cụm

- Các kiểu mô hình phân cụm
 - Hai mô hình phân cụm thông dụng:
 - Phương pháp dựa trên tâm cụm (Centroid-based)
 - Phương pháp phân cấp (Hierarchical)
 - Các mô hình khác:
 - Phân cụm dựa trên mô hình (Model-based)
 - Mỗi cụm được thể hiện bằng một phân bố thống kê tham số
 - Dữ liệu là một hỗn hợp các phân bố
 - Khái niệm phân cụm fuzzy cứng vs. mềm
 - Cứng (Hard): Các mẫu được chia thành các cụm riêng biệt
 - Mềm (Soft): Các mẫu có thể thuộc nhiều hơn 1 cụm

Phương pháp phân cấp

- Phương pháp phân cấp (phân cụm cây)

- Các cụm dựa trên khoảng cách giữa các mẫu
- Hiển thị theo phân cấp mà không theo cách phân hoạch dữ liệu

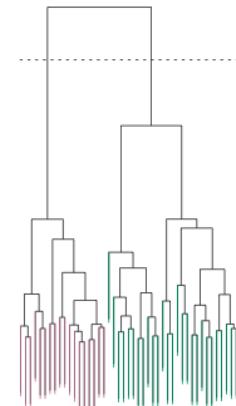
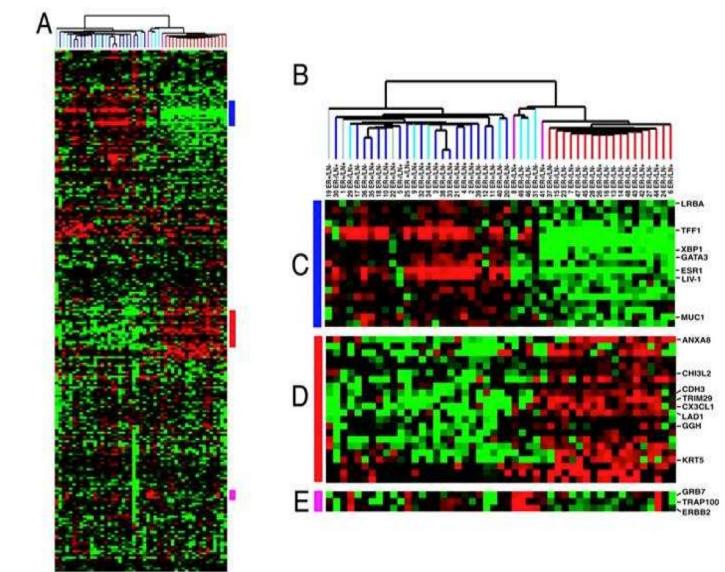


Figure 10.9 , ISL 2013



Sørlie, Therese, et al. (2003) "Repeated observation of breast tumor subtypes in independent gene expression data sets," PNAS.

Phân cụm K-means

- Gom nhóm dữ liệu thành K cụm riêng biệt
 - Mỗi cụm K được định nghĩa bởi 1 véc tơ tâm cụm (centroid)
 - Tâm cụm: giá trị trung bình của tất cả các đối tượng trong cụm
 - Mỗi đối tượng gán cho 1 cụm đơn (tâm cụm gần nhất)
 - Yêu cầu số lượng cụm đầu vào K
 - “Phân cụm tốt” cực tiểu sự biến đổi giữa các cụm
 - “Tính tương tự (Similarity)” đo theo khoảng cách Euclidean

Phân cụm K-means

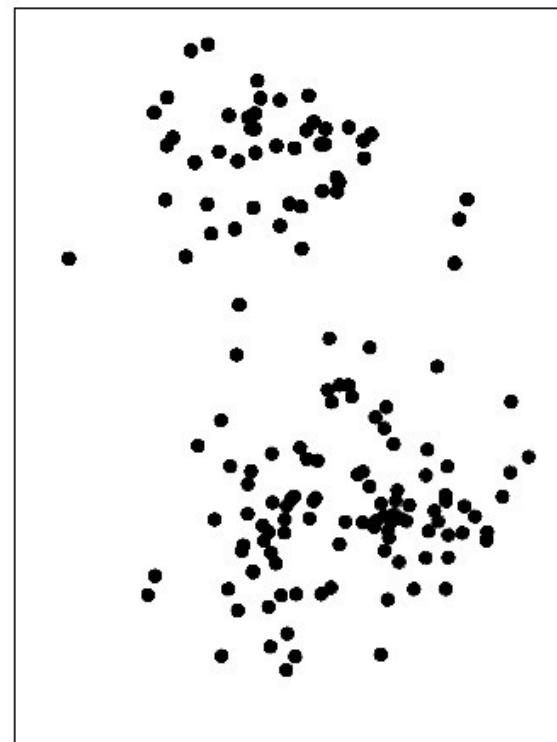


Figure 10.5 , ISL 2013*

*Một số hình vẽ trong bài trình bày này được lấy từ cuốn "*An Introduction to Statistical Learning, with applications in R*" (Springer, 2013) với sự đồng ý của các tác giả: G. James, D. Witten, T. Hastie and R. Tibshirani

Phân cụm K-means

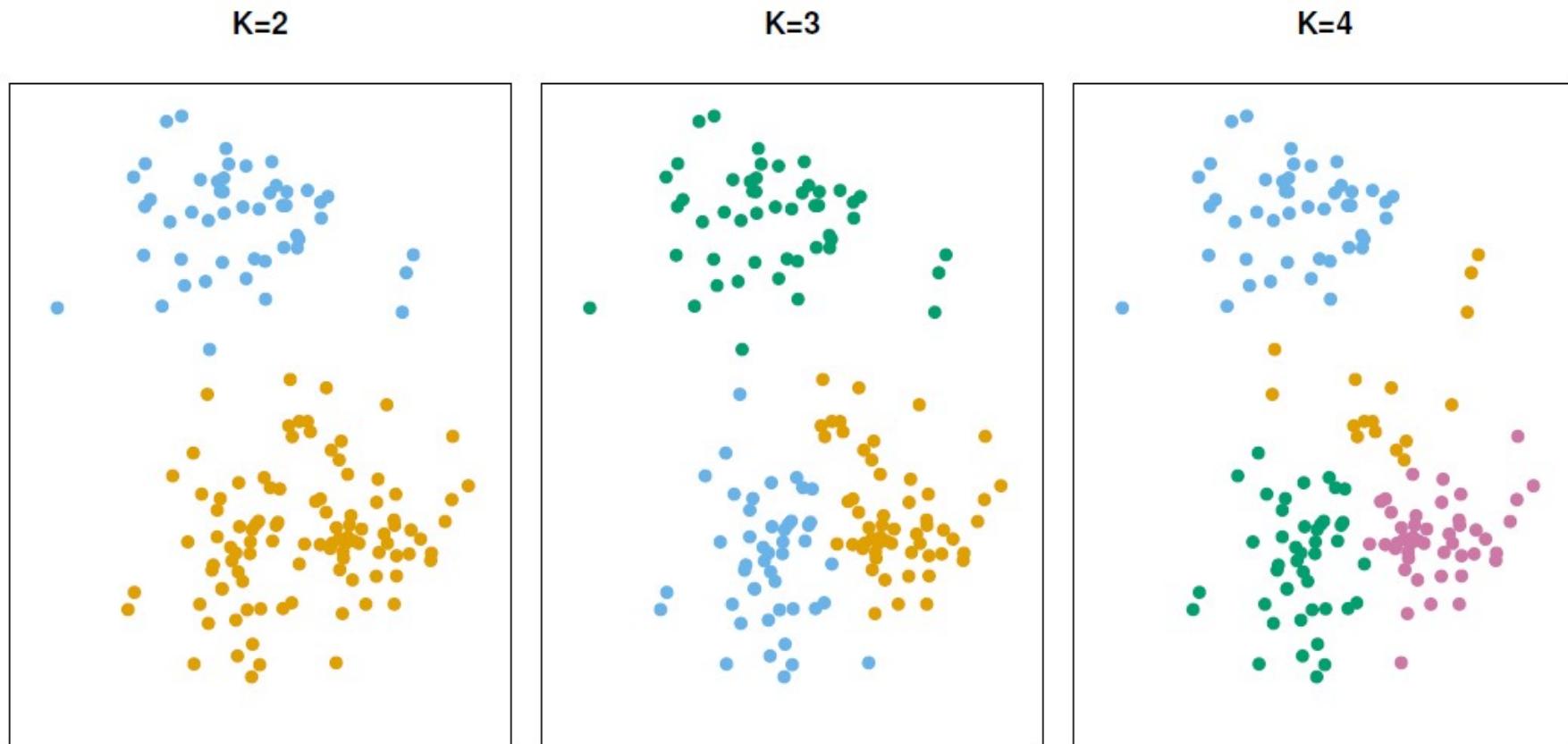


Figure 10.5 , ISL 2013

Dữ liệu

case	sex	glasses	moustach	smile	hat
------	-----	---------	----------	-------	-----

1	m	y	n	y	n
2	f	n	n	y	n
3	m	y	n	n	n
4	m	n	n	n	n
5	m	n	n	y?	n
6	m	n	y	n	y
7	m	y	n	y	n
8	m	n	n	y	n
9	m	y	y	y	n
10	f	n	n	n	n
11	m	n	y	n	n
12	f	n	n	n	n

Đo lường khoảng cách (distance)

- Euclidean $\sqrt{[\sum (y - x)^2]}$
- Squared Euclidean $\sum (y - x)^2$ (thông dụng)
- City-Block $\sum |y - x|$
- Chebychev $\max |y - x|$
- Cosine $\cos r_{xy}$

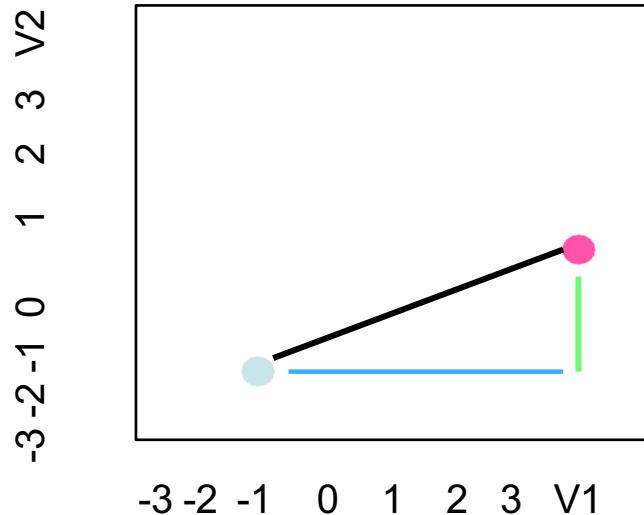
Euclidean

$$X = 2, 0$$

$$Y = -2, -2$$

$$\sqrt{(-2 - 2)^2 + (-2 - 0)^2}$$

$$(4^2 + 2^2) = \sqrt{20} = 4.47$$



$$\sqrt{\sum (y - x)^2}$$

Khoảng cách giữa X và Y = 4.47

Squared Euclidean

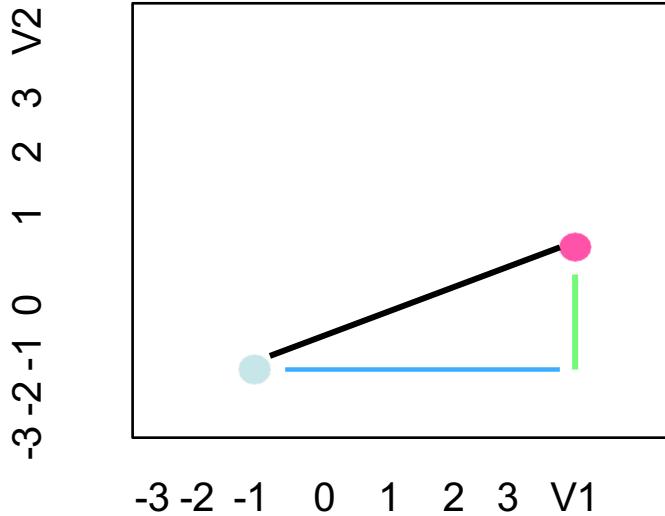
$$\sum (y - \textcolor{magenta}{x})^2$$

$$X = 2, 0$$

$$Y = -2, -2$$

$$([-2 - 2]^2 + [-2 - 0])^2$$

$$(4^2 + 2^2) = 20$$



Squared Euclidean

Tính toán distance

	X	Y		$\ v_3 - v_2 \ _1 = \delta_1(v_3, v_2)$
(1)	4	4	v_1	$= 15-8 + 8-4 = 11$
(2)	8	4	v_2	$\ v_3 - v_2 \ _2 = \delta_2(v_3, v_2)$
(3)	15	8	v_3	$= [(15-8)^2 + (8-4)^2]^{1/2}$
(4)	24	4	v_4	$= 65^{1/2} \sim 8.062$
(5)	24	12	v_5	$\ v_3 - v_2 \ _\infty =$
				$\delta_\infty(v_3, v_2) =$
				$\max(15-8 , 8-4) = 7$

Ví dụ với 5 điểm: Euclidean distance

	X	Y	(1)	(2)	(3)	(4)	(5)	
(1)	4	4	(1)	-	4.0	11.7	20.0	21.5
(2)	8	4	(2)		-	8.1	16.0	17.9
(3)	15	8	(3)			-	9.8	9.8
(4)	24	4	(4)				-	8.0
(5)	24	12	(5)					-

Ma trận tương tự (Dissimilarity matrix)

(1) (2) (3) (4) (5)

* 4.0 11.7 20.0 21.5 (1)

* 8.1 16.0 17.9 (2)

* 9.8 9.8 (3)

* 8.0 (4)

* (5)

Phân cụm K-means

- Các tâm cụm cực tiểu sự biến đổi giữa các cụm

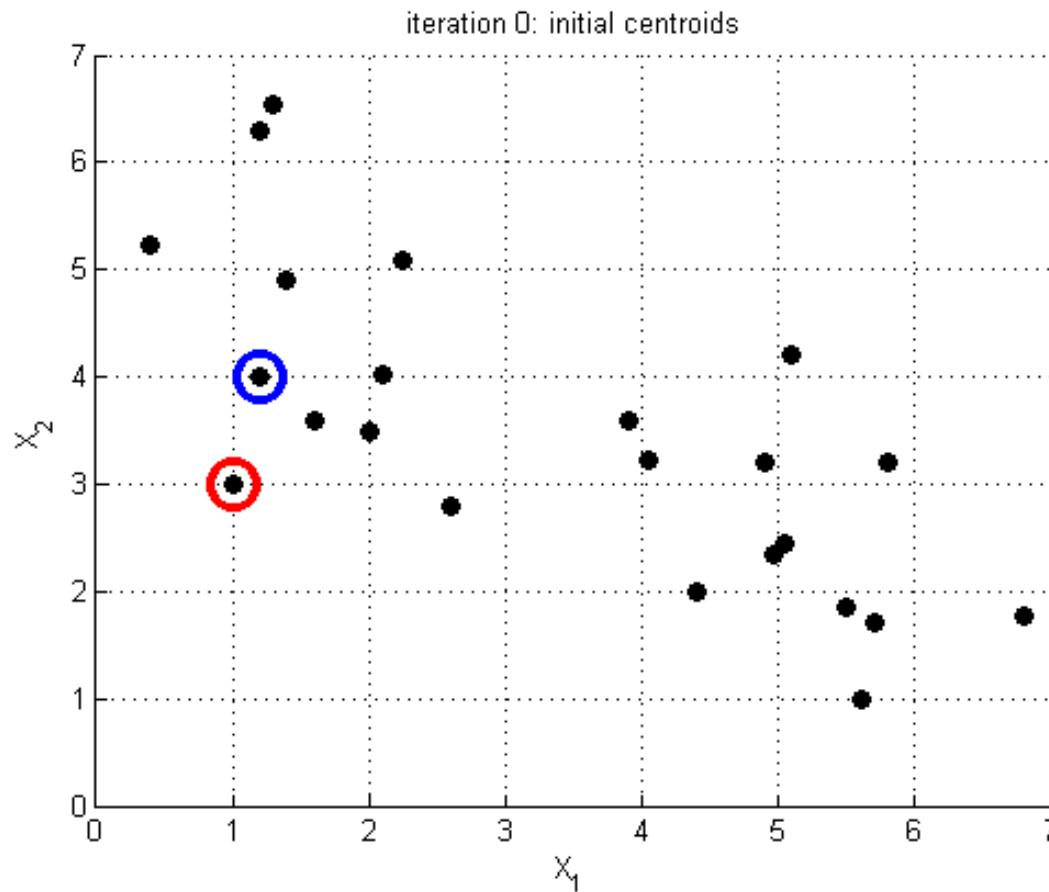
$$J = \frac{1}{n} \sum_{i=1}^K \sum_{x \in c_i} |x - \mu_i|^2$$

- Các tâm cụm (trung tâm của cụm): $\mu_i = \frac{1}{|c_i|} \sum_{x \in c_i} x$
- Bài toán cực tiểu hóa này là tối ưu tổ hợp
 - Giải pháp cho cực tiểu hóa địa phương ta sử dụng phương pháp lặp

Thuật toán K-means

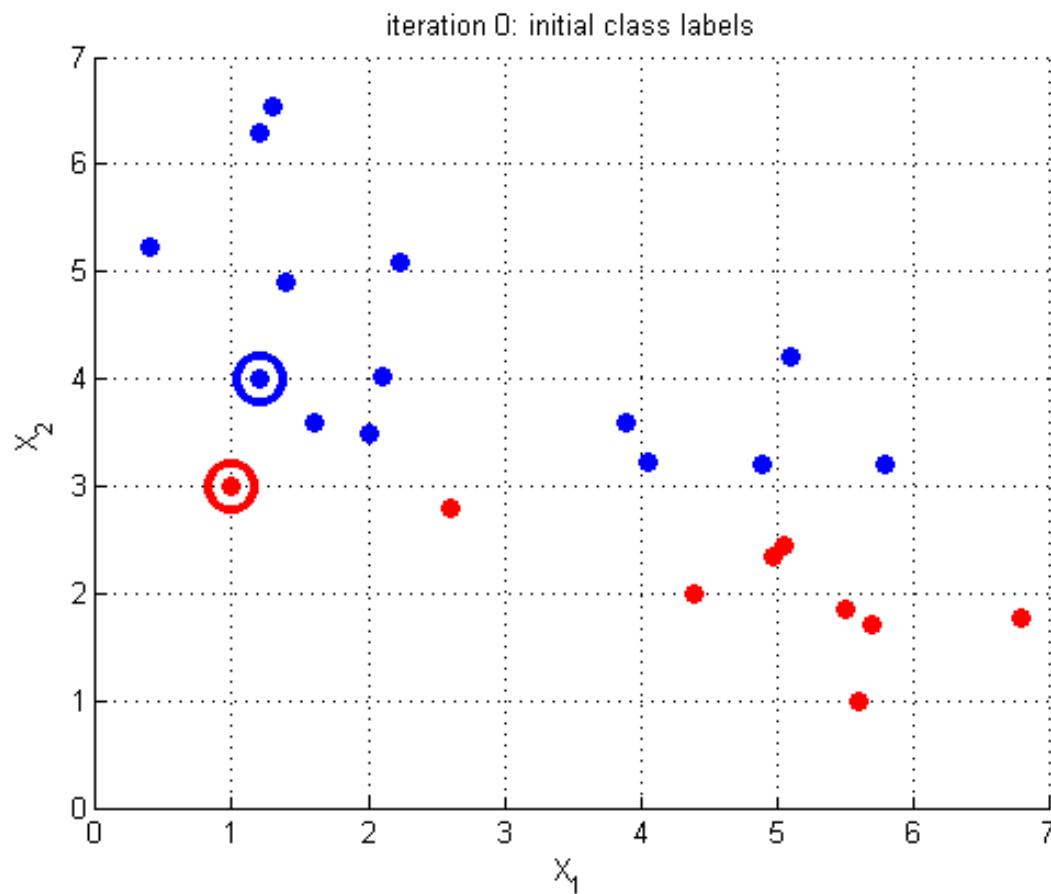
- 1) Khởi tạo chọn ngẫu nhiên K tâm cụm
- 2) Phân hoạch dữ liệu bằng cách gán mỗi đối tượng vào cụm mà nó gần tâm nhất
- 3) Tính các tâm cụm mới trong mỗi cụm
- 4) Lặp lại 2 và 3 cho đến khi thỏa mãn điều kiện
 - “thỏa mãn điều kiện” khi các tâm cụm ổn định và các đối tượng không dịch chuyển giữa các cụm

Thuật toán K-means



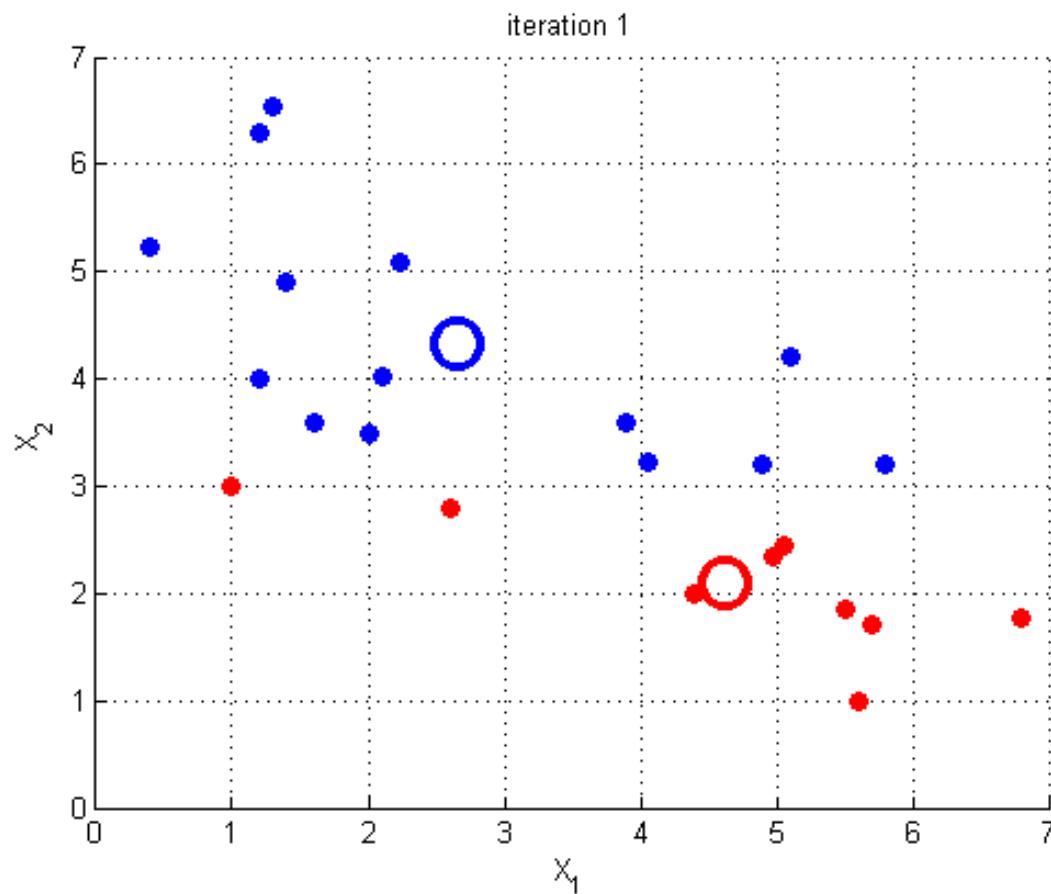
Khởi tạo tâm cụm

Thuật toán K-means



Khởi tạo tâm cụm
Gán các cụm ban đầu

Thuật toán K-means

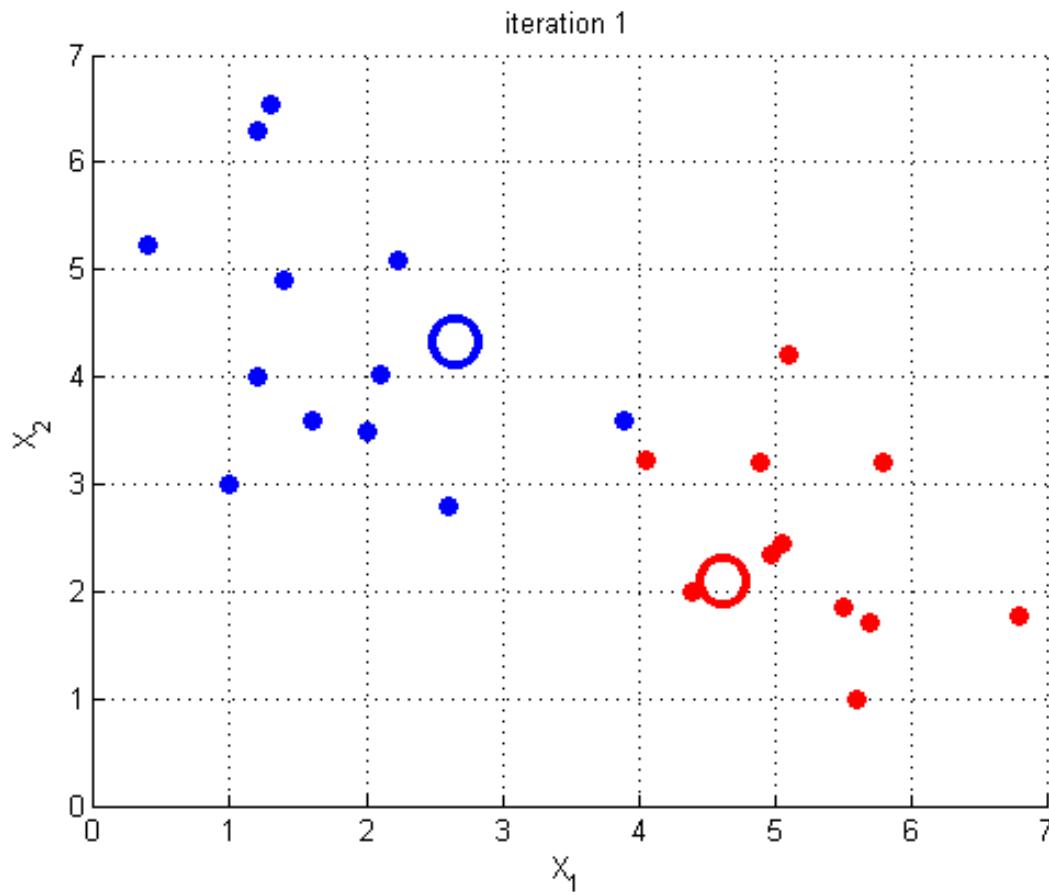


Khởi tạo tâm cụm

Gán các cụm ban đầu

Cập nhật các tâm cụm

Thuật toán K-means



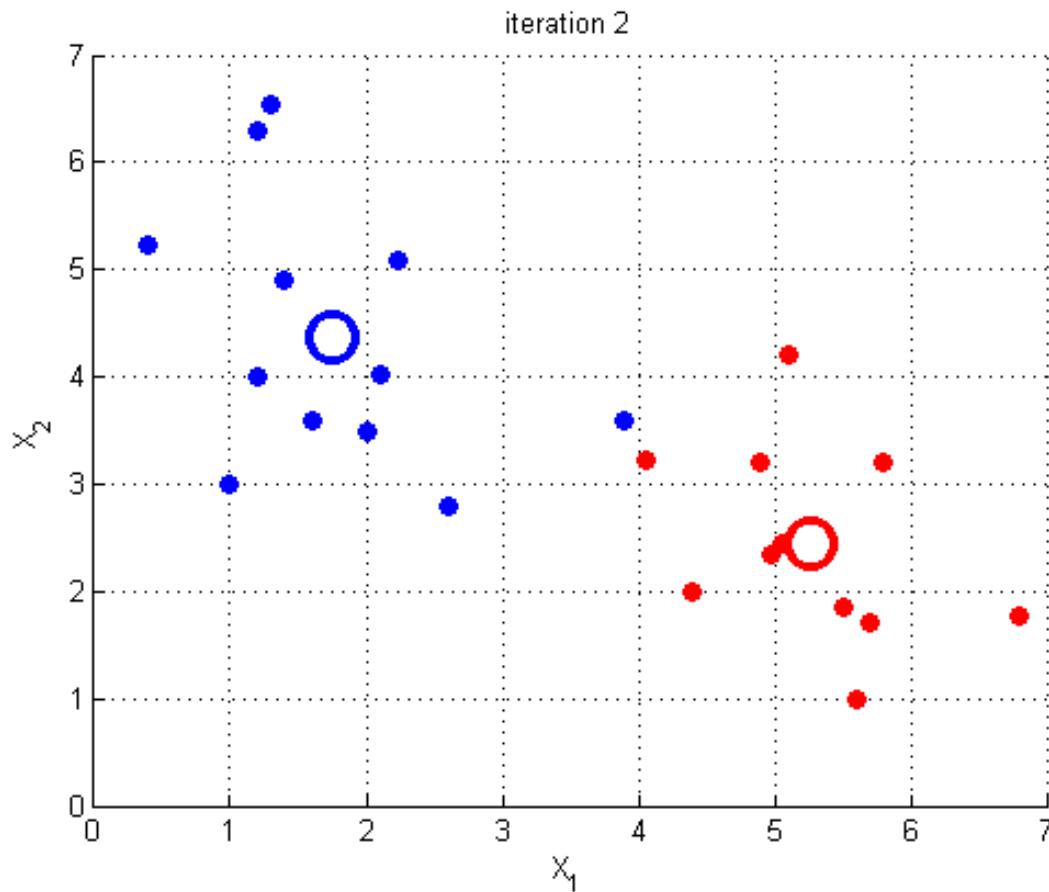
Khởi tạo tâm cụm

Gán các cụm ban đầu

Cập nhật các tâm cụm

Gán lại các cụm

Thuật toán K-means



Khởi tạo tâm cụm

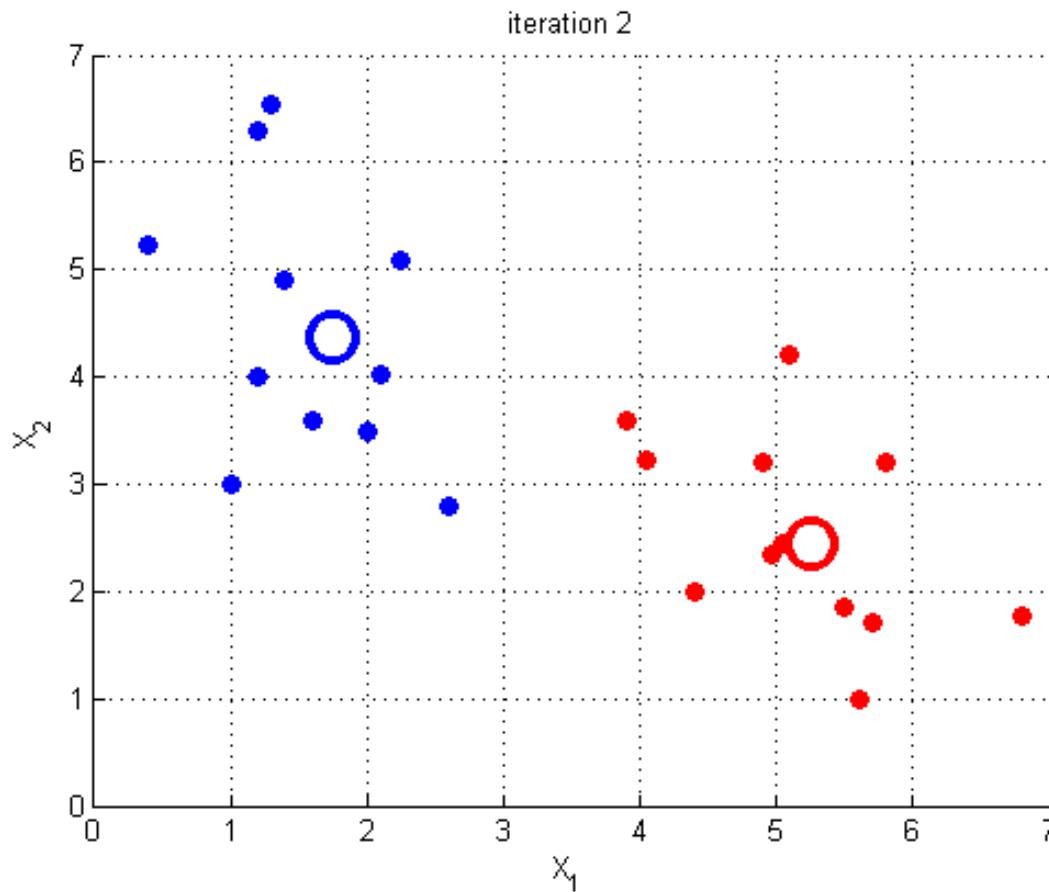
Gán các cụm ban đầu

Cập nhật các tâm cụm

Gán lại các cụm

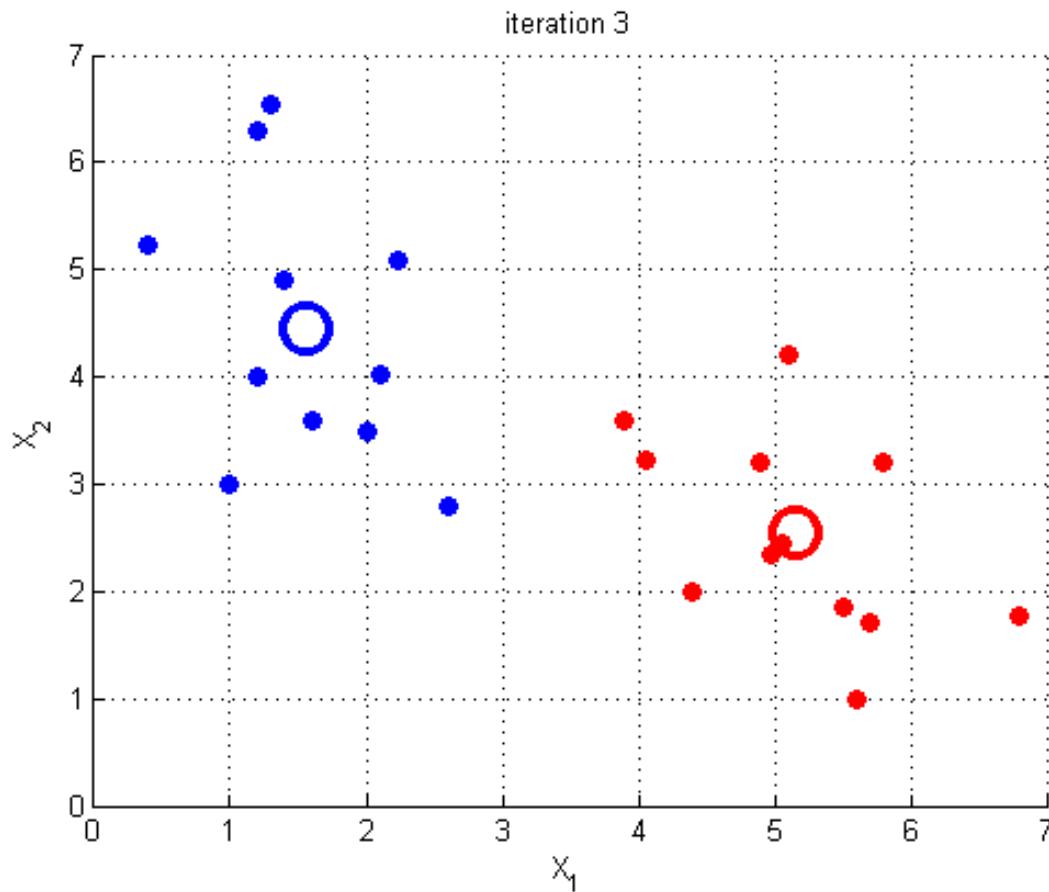
Cập nhật tâm cụm

Thuật toán K-means

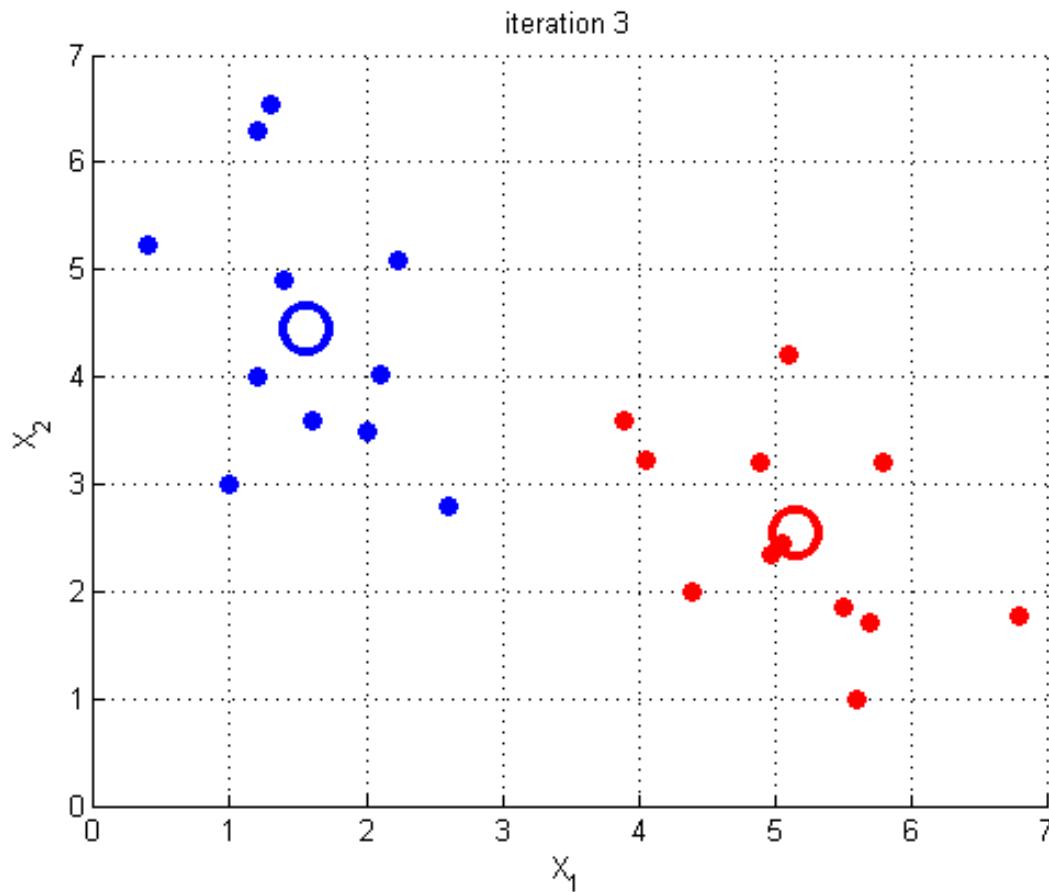


- Khởi tạo tâm cụm
- Gán các cụm ban đầu
- Cập nhật các tâm cụm
- Gán lại các cụm
- Cập nhật tâm cụm
- Gán lại các cụm

Thuật toán K-means

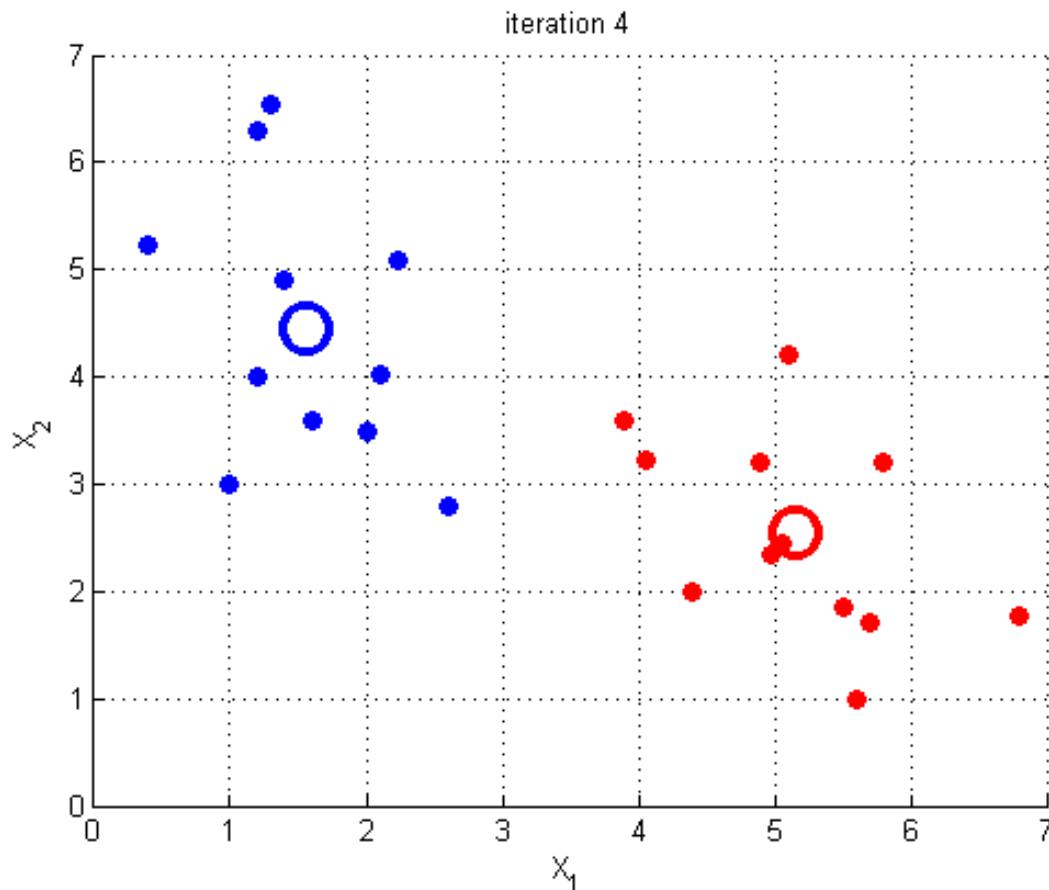


Thuật toán K-means



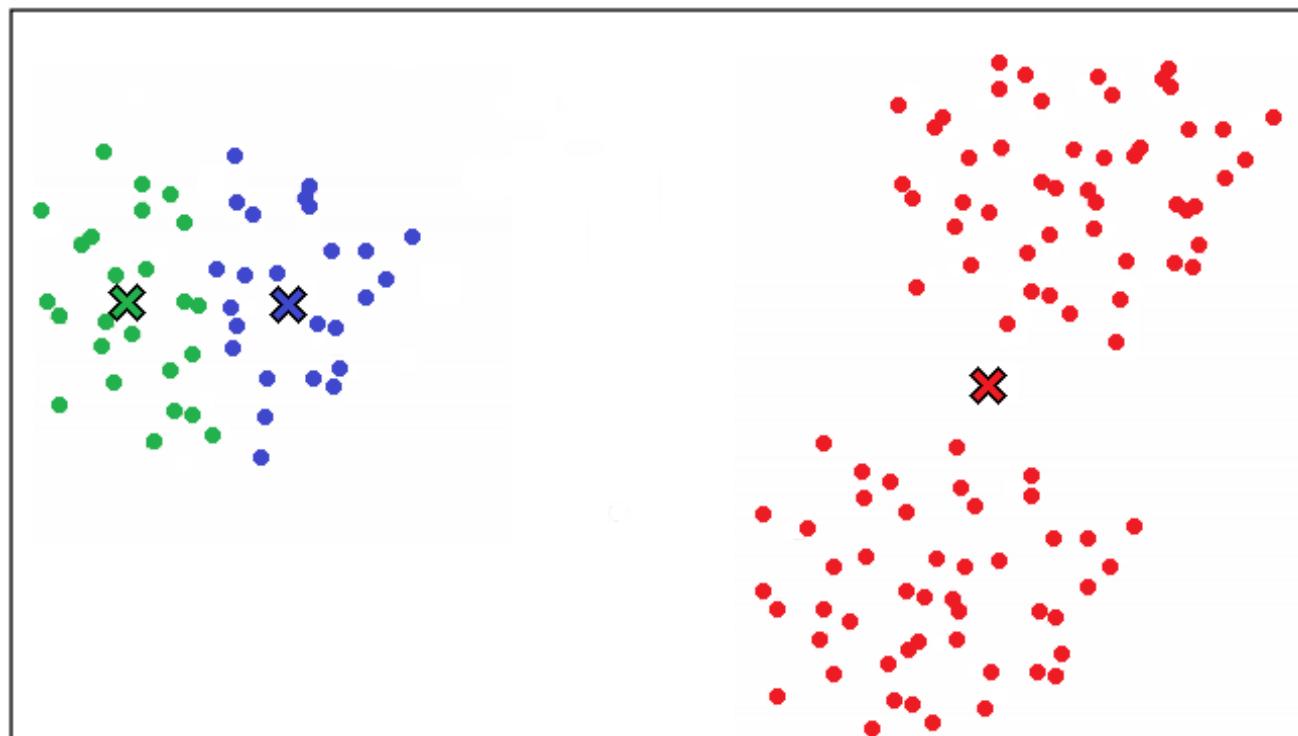
- Khởi tạo tâm cụm
- Gán các cụm ban đầu
- Cập nhật các tâm cụm
- Gán lại các cụm
- Cập nhật tâm cụm
- Gán lại các cụm
- Cập nhật tâm cụm
- Gán lại các cụm

Thuật toán K-means



Thuật toán K-means

- Khởi tạo không tốt dẫn đến kết quả phân cụm kém

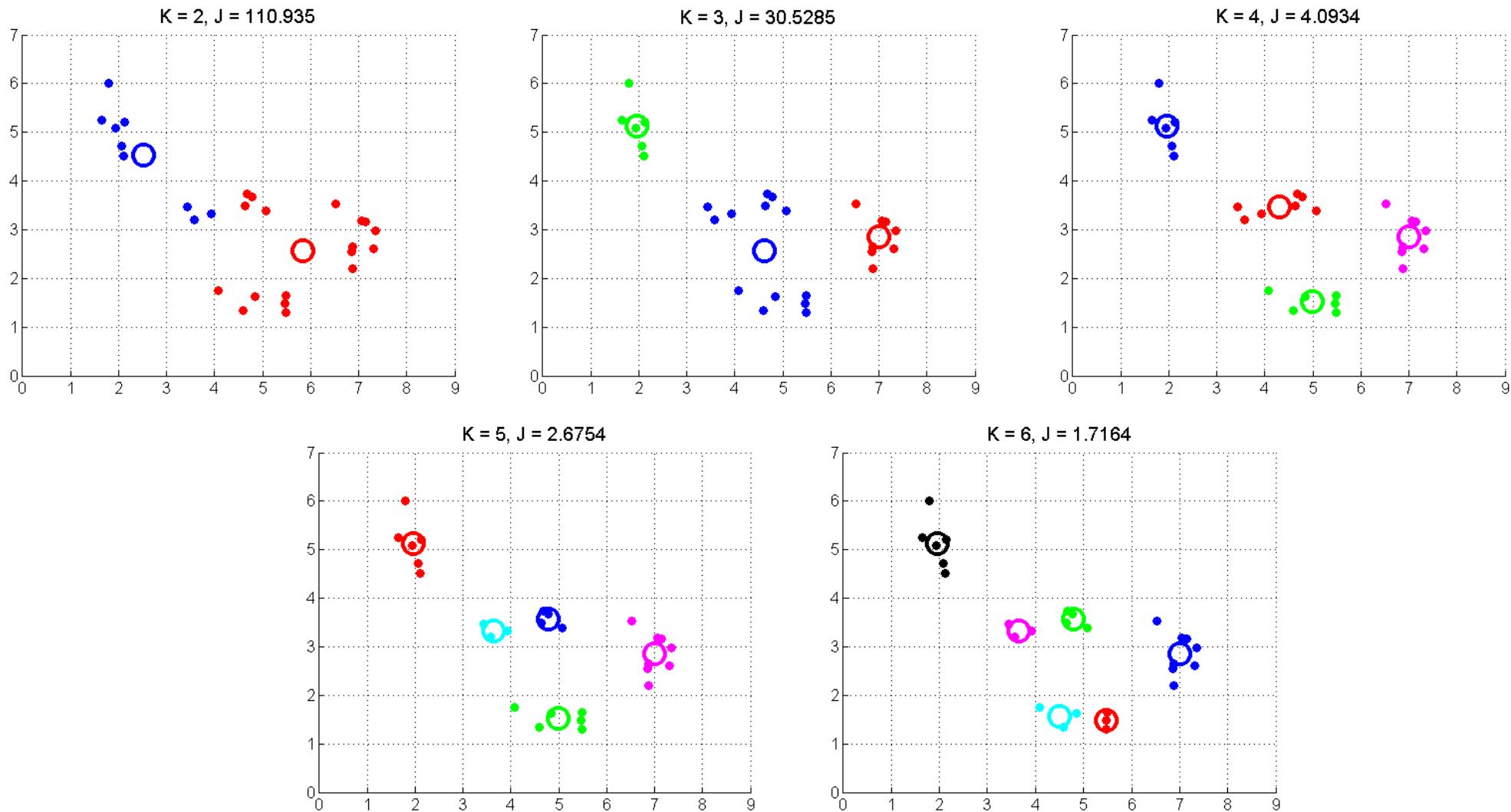


Khởi tạo tâm cụm

- Chọn ngẫu nhiên trong K đối tượng
- Phân hoạch ngẫu nhiên dữ liệu
- Chọn K điểm xa nhau “far apart”
- Khởi tạo bằng cách sử dụng kết quả của phương pháp phân cụm khác

Bao nhiêu cụm?

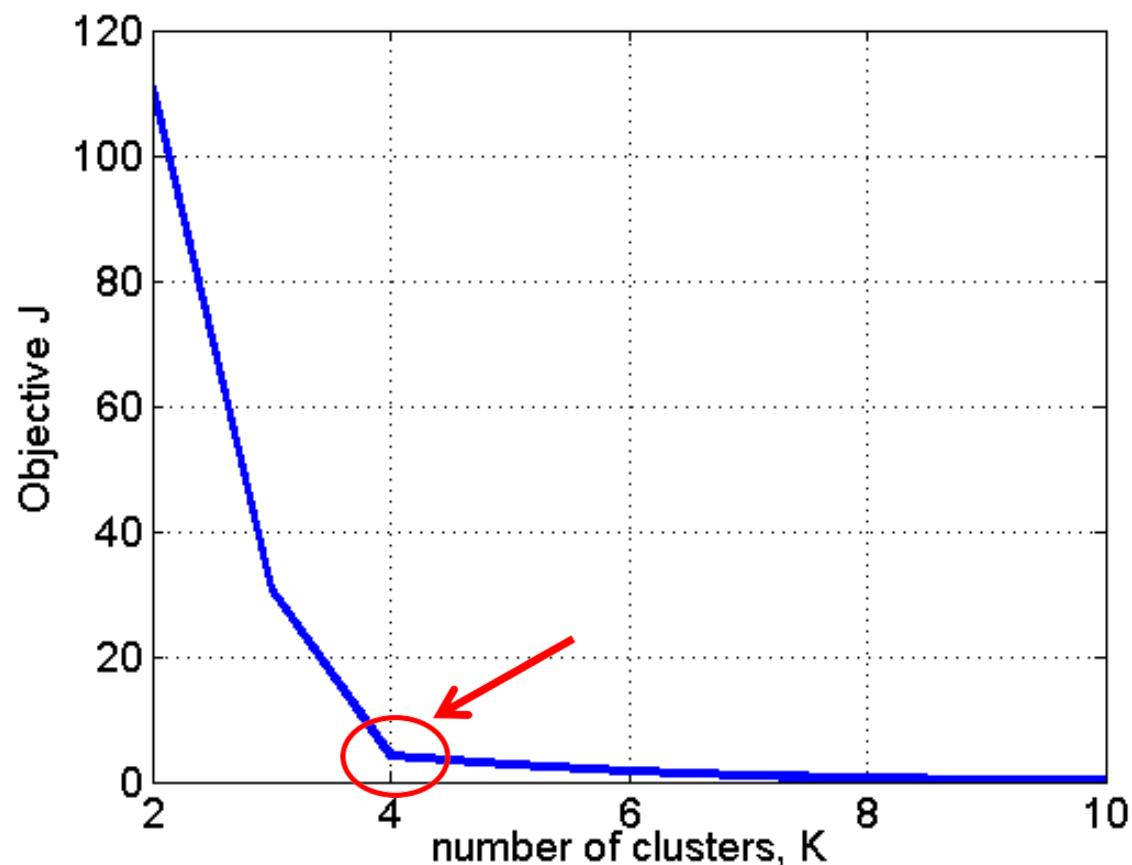
- K-means yêu cầu đầu vào K (# cụm)
 - Ta cần hiểu về bài toán ứng dụng để chọn K
 - Ngược lại, việc chọn K được xác định từ dữ liệu



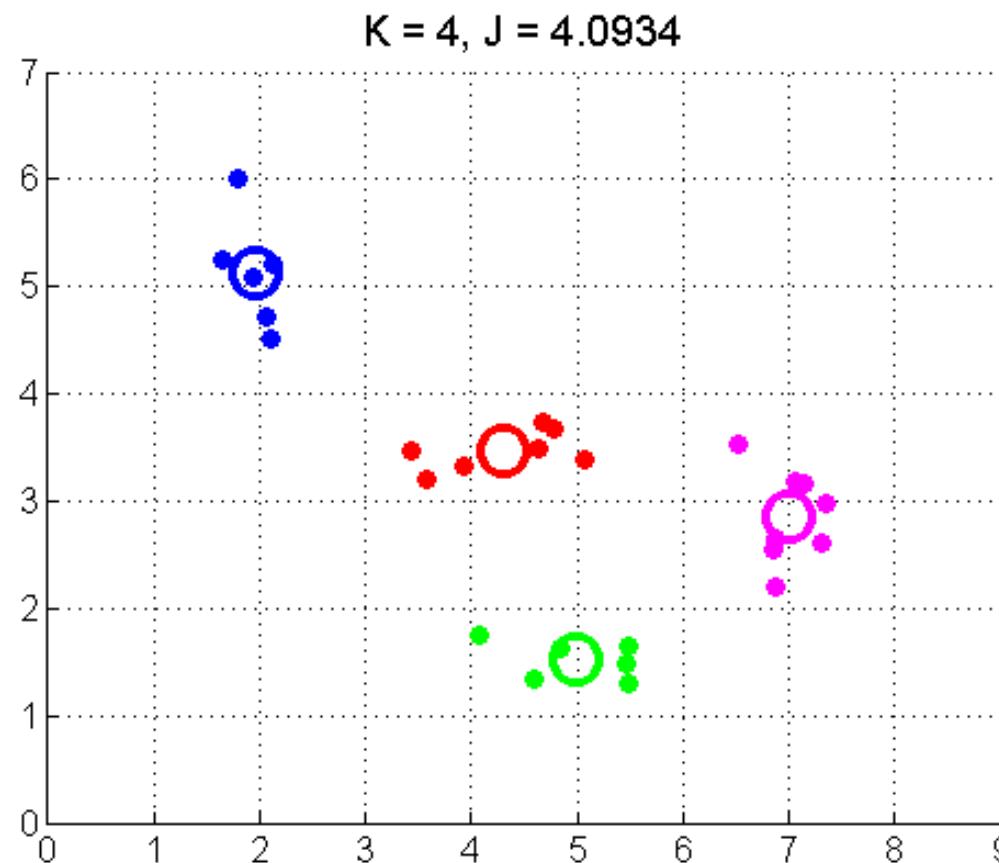
Bao nhiêu cụm?

- Không thể tính được giá trị K để cực tiểu mục tiêu J
 - J giảm đồng thời với tăng K
- Phương pháp dựa trên kinh nghiệm (Heuristic):
 - Với mỗi giá trị ứng viên của K ,
 - Tính toán phân cụm bằng K-means M lần, tìm mục tiêu nhỏ nhất J_K
 - Tìm điểm “khuỷu tay (elbow)” trong đường mục tiêu (K vs J_K)

Bao nhiêu cụm?



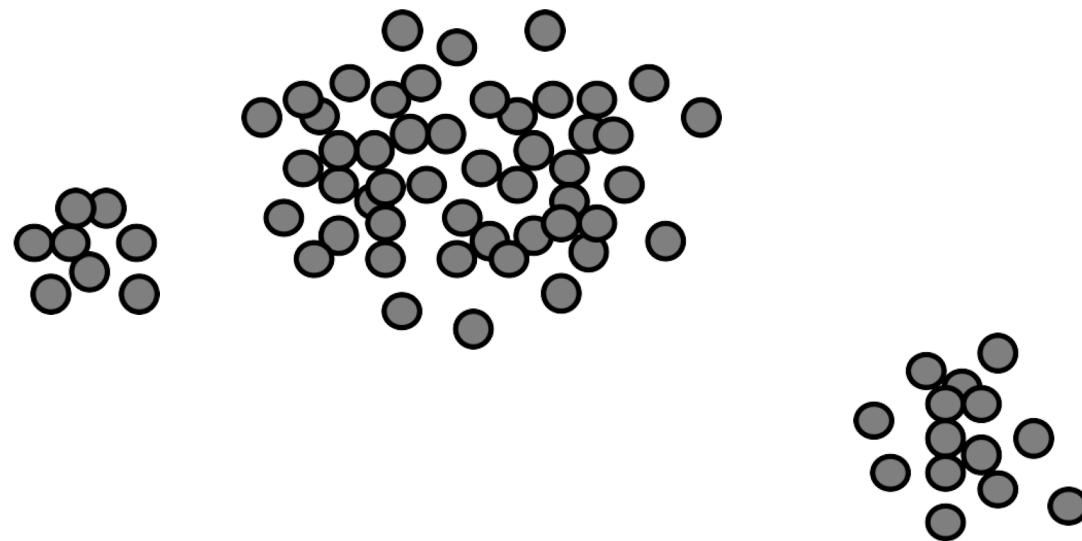
Bao nhiêu cụm?



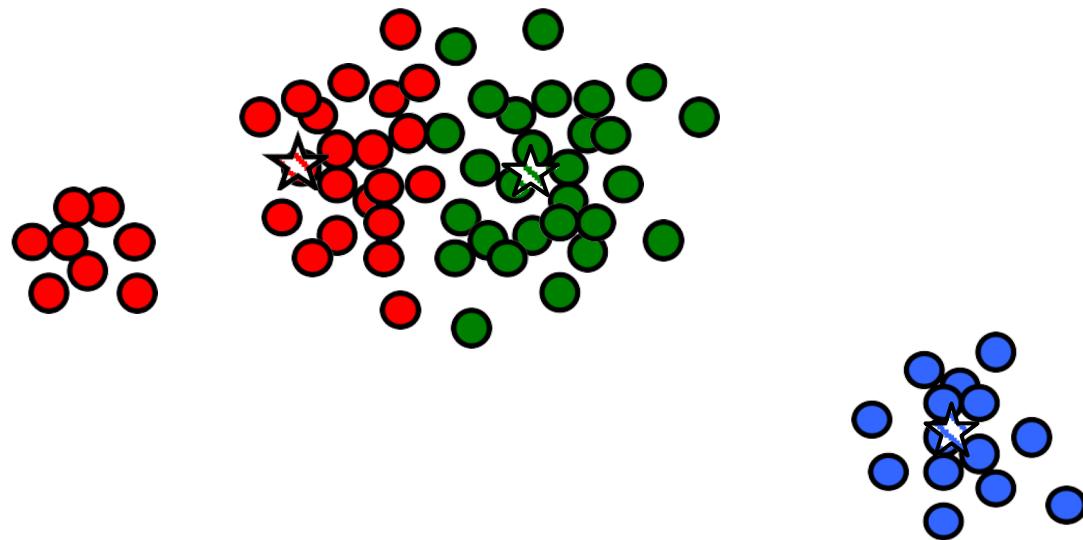
Thuật toán K-means

- **Ưu điểm**
 - Dễ cài đặt
 - Luôn hội tụ với số lần lặp ít
 - Có thể triển khai trên những tập dữ liệu với số chiều lớn
- **Nhược điểm**
 - Giá trị K là tham số đầu vào (khó xác định tối ưu)
 - Thuật toán lặp trả về cực tiểu địa phương*

Thuật toán K-means



Thuật toán K-means



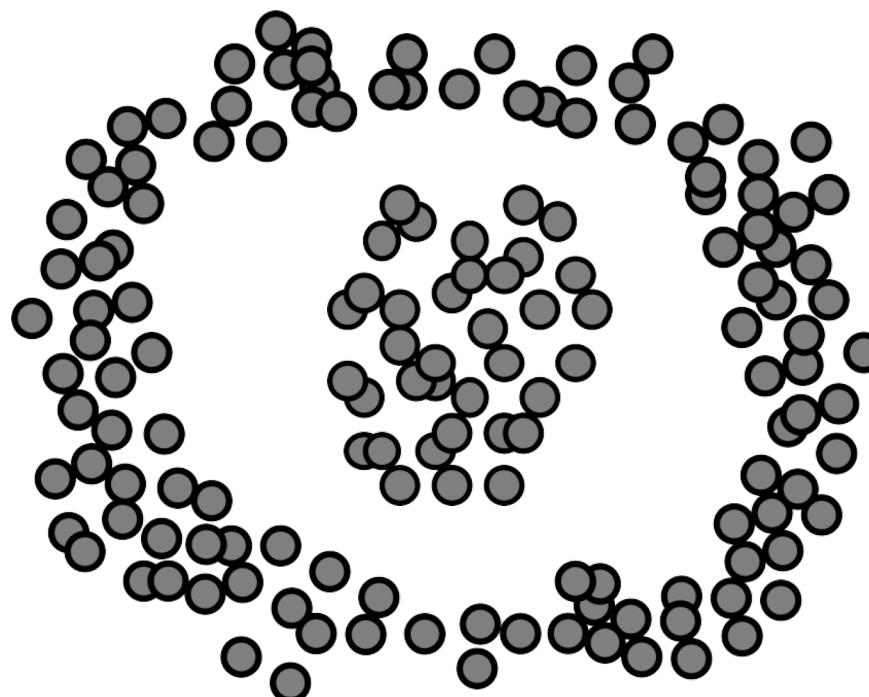
40

50

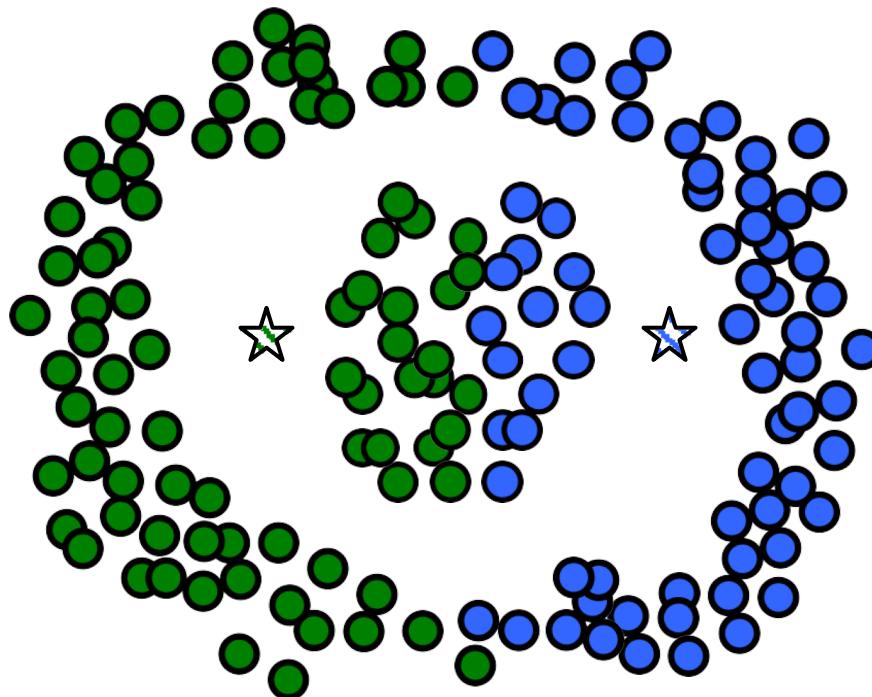
Thuật toán K-means

- **Ưu điểm**
 - Dễ cài đặt
 - Luôn hội tụ với số lần lặp ít
 - Có thể triển khai trên những tập dữ liệu với số chiều lớn
- **Nhược điểm**
 - Giá trị K là tham số đầu vào (khó xác định tối ưu)
 - Thuật toán lặp trả về cực tiểu địa phương*
 - Giả thiết tất cả các cụm hình cầu và có kích thước xấp xỉ nhau *

Thuật toán K-means



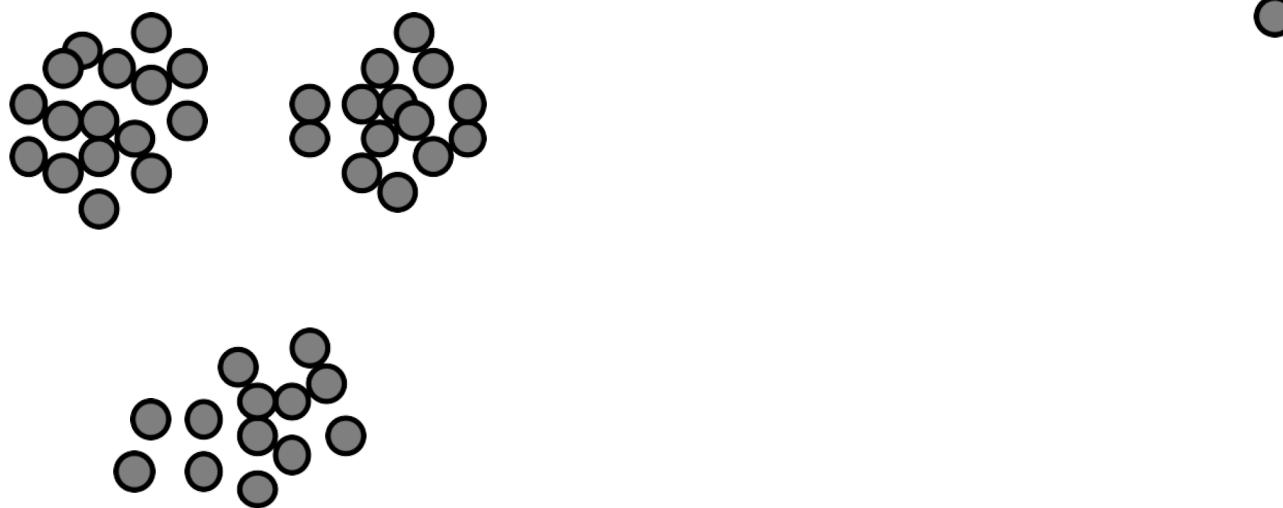
Thuật toán K-means



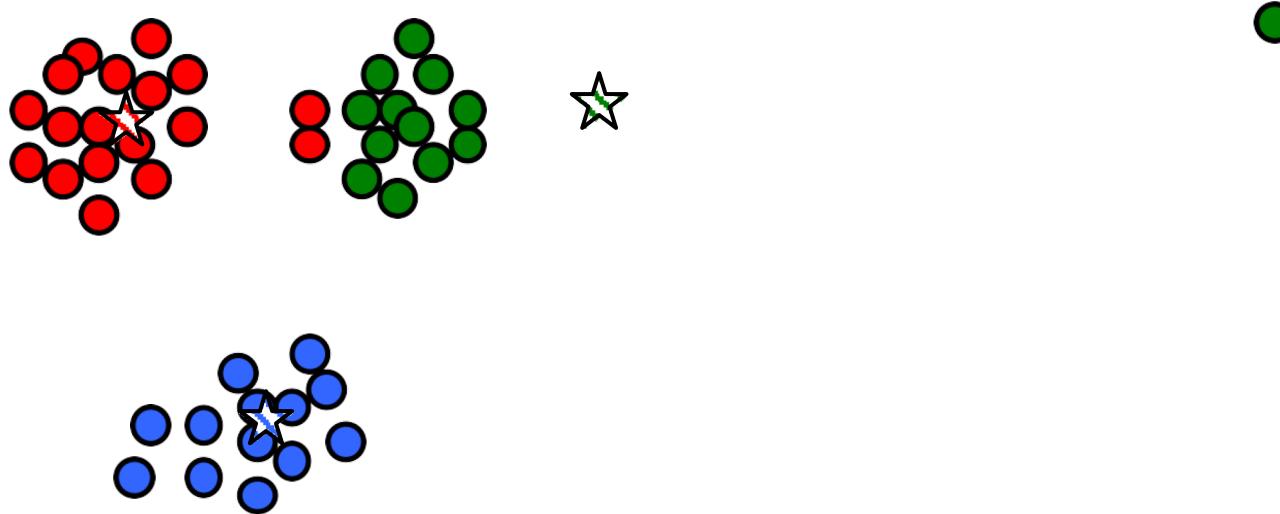
Thuật toán K-means

- **Ưu điểm**
 - Dễ cài đặt
 - Luôn hội tụ với số lần lặp ít
 - Có thể triển khai trên những tập dữ liệu với số chiều lớn
- **Nhược điểm**
 - Giá trị K là tham số đầu vào (khó xác định tối ưu)
 - Thuật toán lặp trả về cực tiểu địa phương*
 - Giả thiết tất cả các cụm hình cầu và có kích thước xấp xỉ nhau *
 - Nhạy với các phần tử ngoại lai*

Thuật toán K-means



Thuật toán K-means



Thuật toán K-means

- **Ưu điểm**
 - Dễ cài đặt
 - Luôn hội tụ với số lần lặp ít
 - Có thể triển khai trên những tập dữ liệu với số chiều lớn
- **Nhược điểm**
 - Giá trị K là tham số đầu vào (khó xác định tối ưu)
 - Thuật toán lặp trả về cực tiểu địa phương*
 - Giả thiết tất cả các cụm hình cầu và có kích thước xấp xỉ nhau *
 - Nhạy với các phần tử ngoại lai*
 - ***một số nhược điểm được khắc phục bằng vài biến thể của K-means**

Thuật toán K-means

- Khắc phục nhược điểm
 - Khởi tạo không tốt → ta chạy thuật toán nhiều lần
 - K-medians: Tâm cụm được tính bằng giá trị trung vị thay cho giá trị trung bình của K-means
 - K-medoids
 - Yêu cầu: “tâm cụm” phải là 1 trong các điểm dữ liệu
 - xử lý tốt hơn các phần tử ngoại lai
 - linh hoạt hơn – có thể dùng nhiều độ đo
 - *nhưng* thời gian tính toán lâu hơn vì phải tính các tâm cụm

Ví dụ: Phân đoạn/nén ảnh



Ảnh: Trọng Vũ

Phân đoạn/nén ảnh

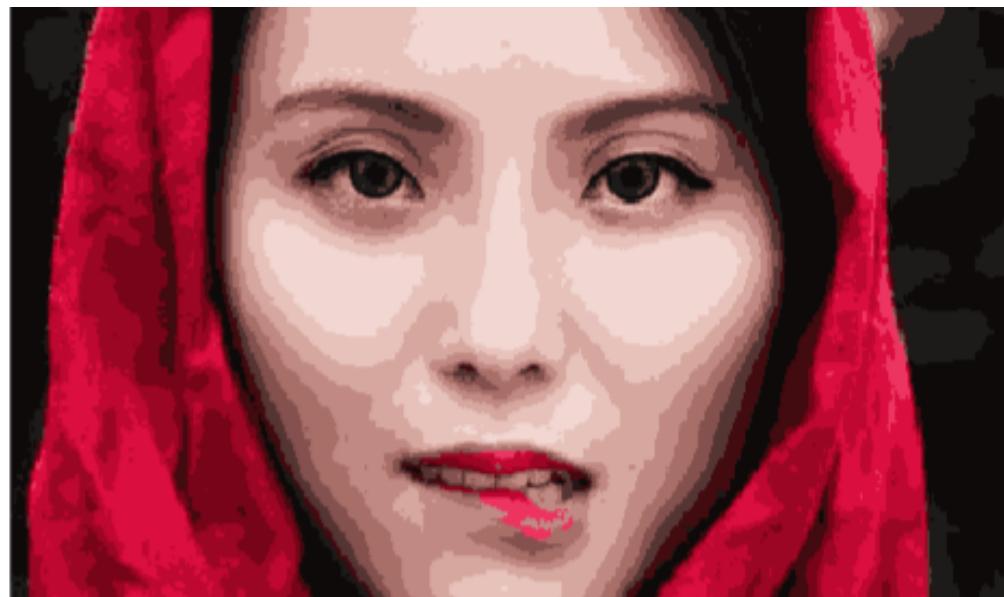
- Ảnh → điểm ảnh (pixels) → véc tơ RGB (colors)
- Áp dụng K-means tập hợp các véc tơ RGB
 - Một véc tơ RGB ứng với 1 điểm ảnh
 - Các cụm thể hiện các màu giống nhau
- Thay thế mỗi điểm ảnh bằng một tâm cụm liên quan
 - Kết quả trên ảnh với K màu khác nhau

Phân đoạn/nén ảnh



Chất lượng nén ảnh với số lượng
cluster khác nhau. Ảnh: K=5

Phân đoạn/nén ảnh



K=10

Phân đoạn/nén ảnh



K=15

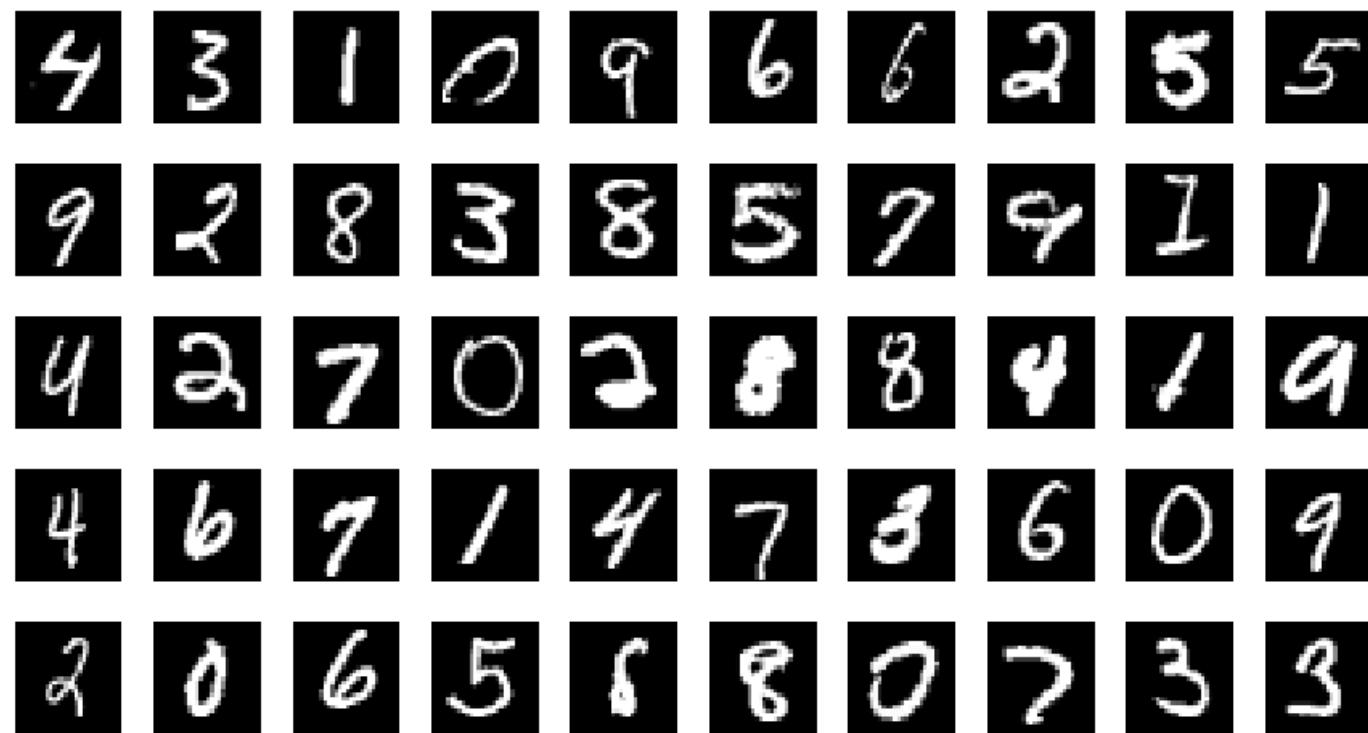


K=20

Thuật toán K-means

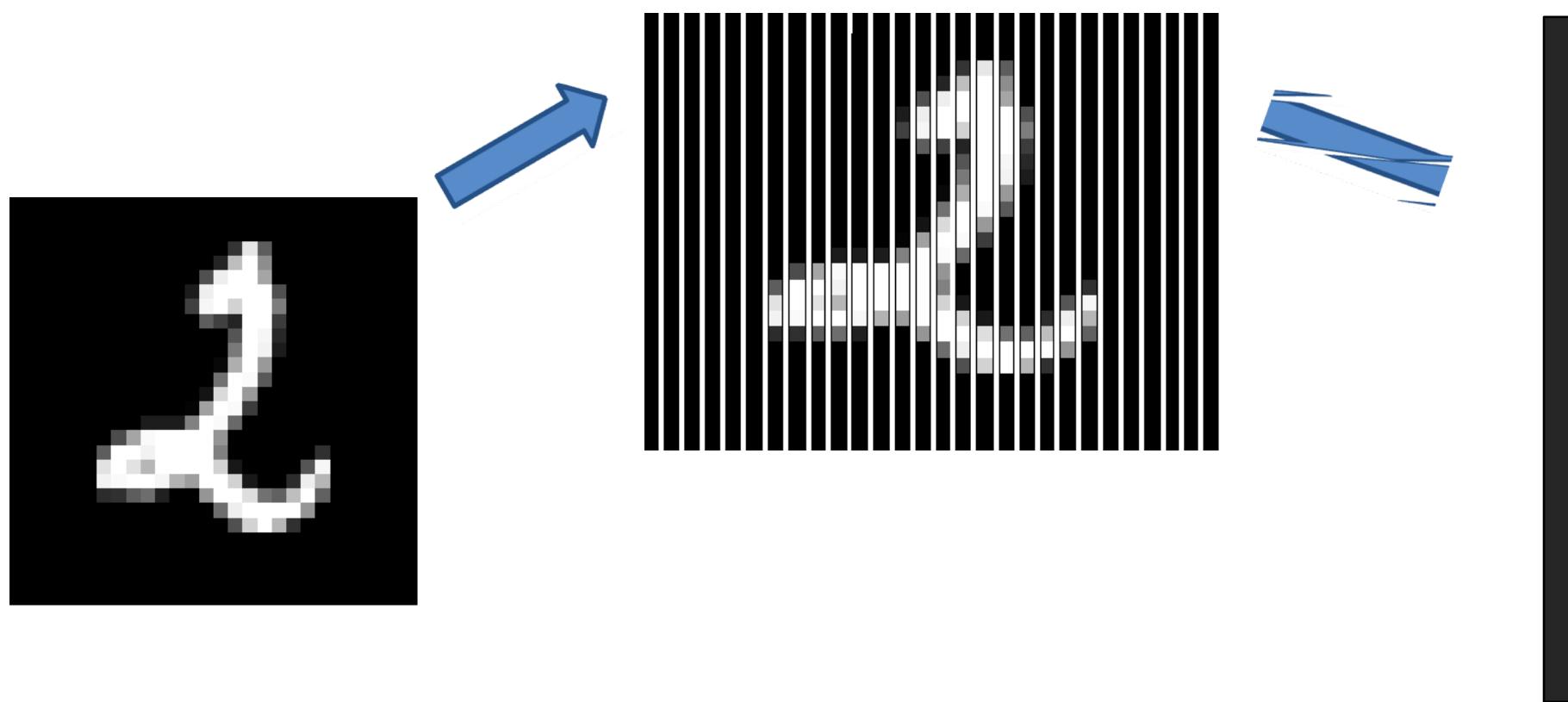
- Chúng ta thực hiện thuật toán với dữ liệu có 2–3 thuộc tính (rất dễ để minh họa)
 - Trong thực tế, ta thường gặp nhiều hơn 2 thuộc tính khi phân tích dữ liệu
- Phân cụm sẽ khó khăn hơn rất nhiều khi gấp số chiều lớn

Phân cụm chữ viết tay



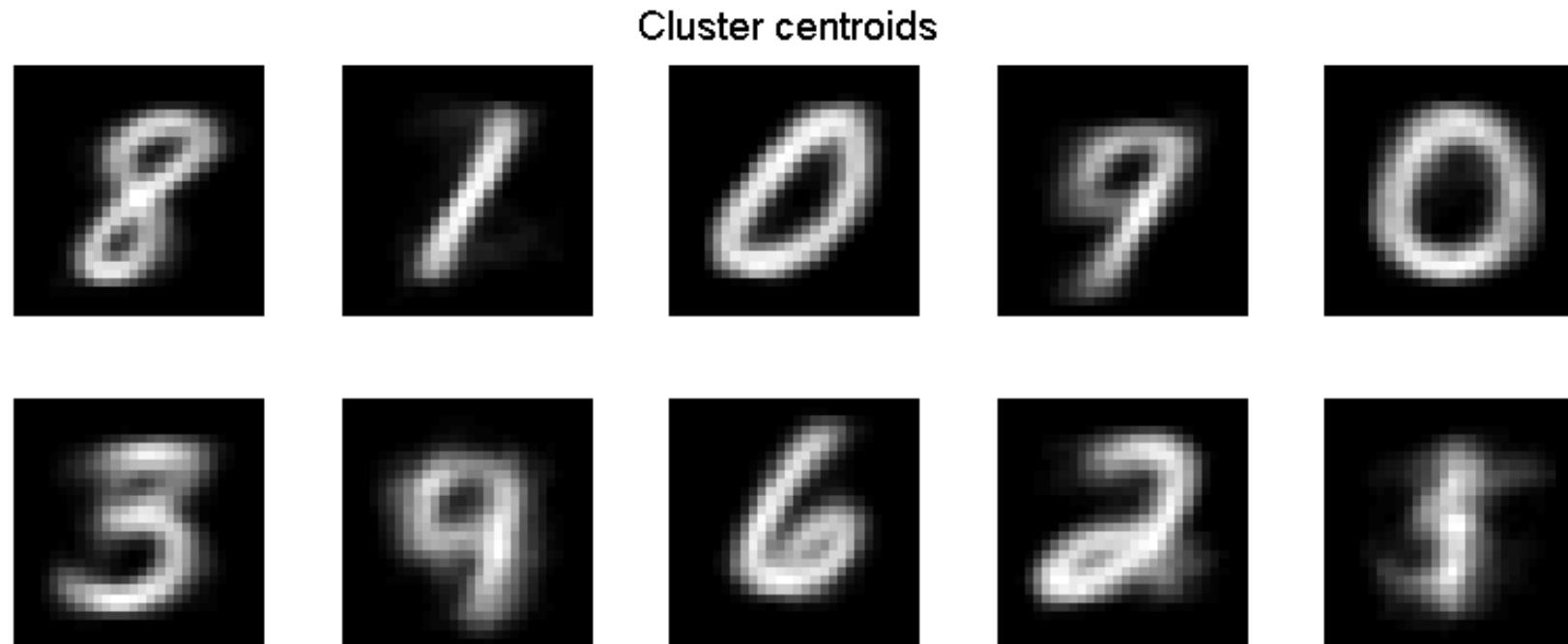
MNIST dataset: <http://cis.jhu.edu/~sachin/digit/digit.html>

Phân cụm chữ viết tay



Phân cụm chữ viết tay

- Áp dụng K-means, sử dụng $K = 10$



Phân cụm chữ viết tay

Class A Class B Class C Class D Class E Class F Class G Class H Class I Class J

8 1 5 7 0 2 8 6 2 5

8 2 0 7 0 3 4 6 2 1

8 1 0 4 0 5 4 6 2 4

8 1 0 9 0 3 2 6 2 1

8 1 0 7 0 3 4 6 2 1

8 9 0 9 5 5 5 6 2 1

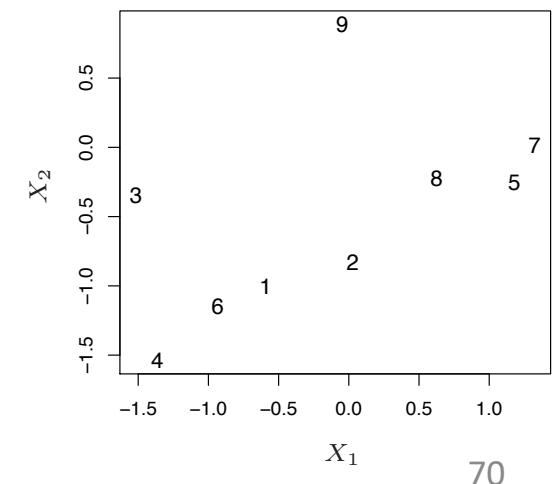
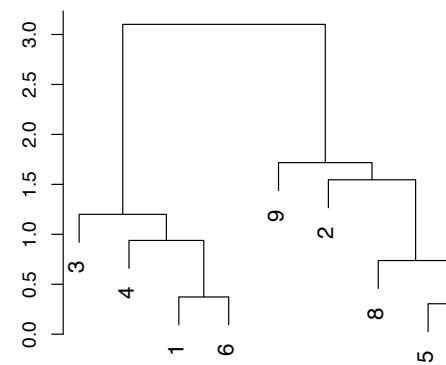
8 1 0 9 0 5 9 6 2 5

Phân cụm phân cấp

- Phân cụm theo phương pháp K-Means yêu cầu chọn tham số đầu vào là số lượng cụm K
- Nếu ta không muốn làm theo cách trên, ta có thể dùng phương pháp phân cụm phân cấp
- Phân cụm phân cấp có ưu điểm là hiển thị các quan sát (mẫu) dạng hình cây nên dễ hình dung, được gọi là phân cụm theo cấu trúc cây (Dendrogram)

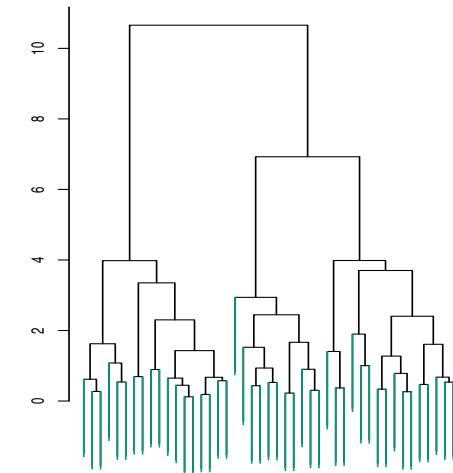
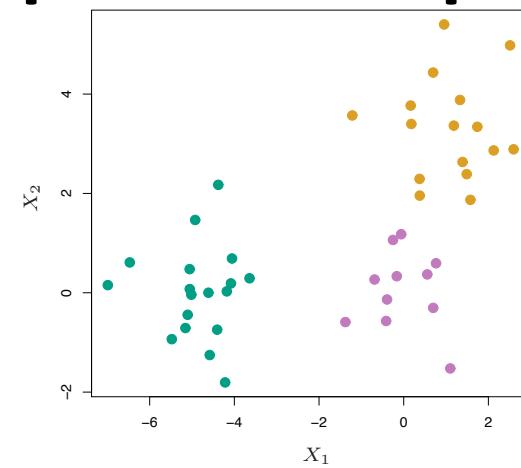
Phân cụm phân cấp

- Đầu tiên nhập các điểm gần nhau nhất (5 và 7)
- Độ cao của việc hợp nhất (theo trực dọc) phản ánh độ tương tự của các điểm
- Sau khi các điểm được hợp nhất, chúng được xem như 1 mẫu để tiếp tục tiến hành giải thuật



Diễn giải phương pháp phân cấp

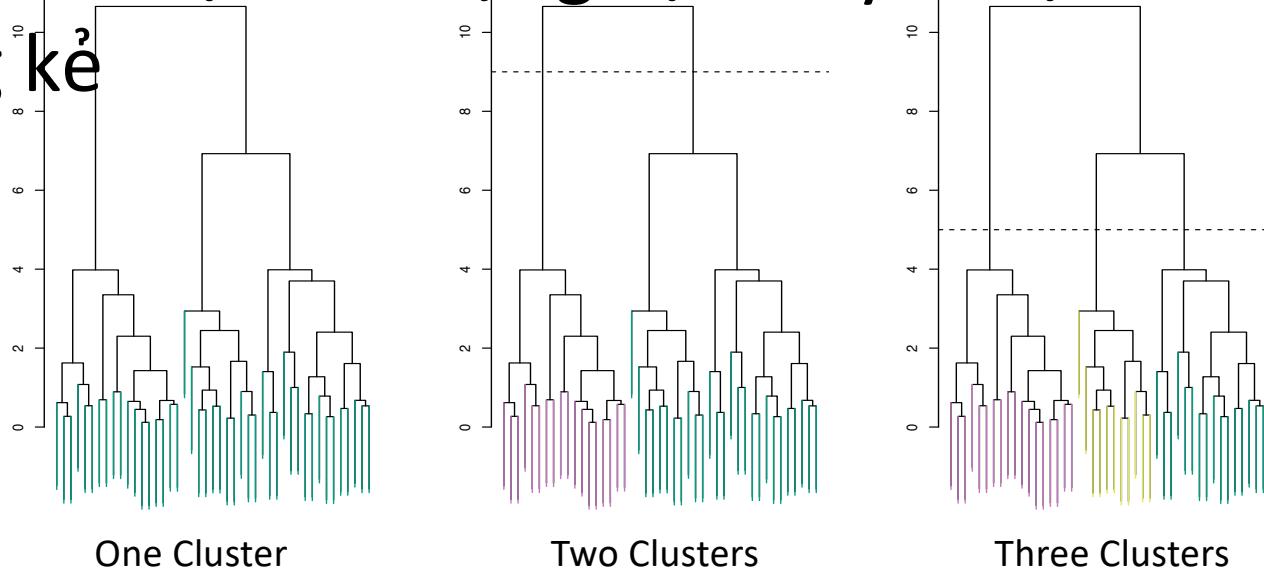
- Mỗi “lá” của cây phân cấp biểu diễn một trong 45 mẫu
- Phần đáy của cây, mỗi mẫu là 1 lá riêng biệt. Tuy nhiên, càng lên cao các lá sẽ hợp nhất với nhau. Việc này thể hiện các mẫu có độ tương tự với các mẫu khác.
- Khi di chuyển cao lên phần ngọn của cây, số lượng mẫu đã được hợp nhất. Trước đó (phần dưới của cây) với 2 mẫu hợp nhất, chúng có chung đặc tính (gần) với nhau.



Lựa chọn các cụm

Để chọn các cụm ta kẻ đường thẳng ngang cây phân cấp

Ta có thể chọn số lượng cụm tùy thuộc vào vị trí đường kẻ



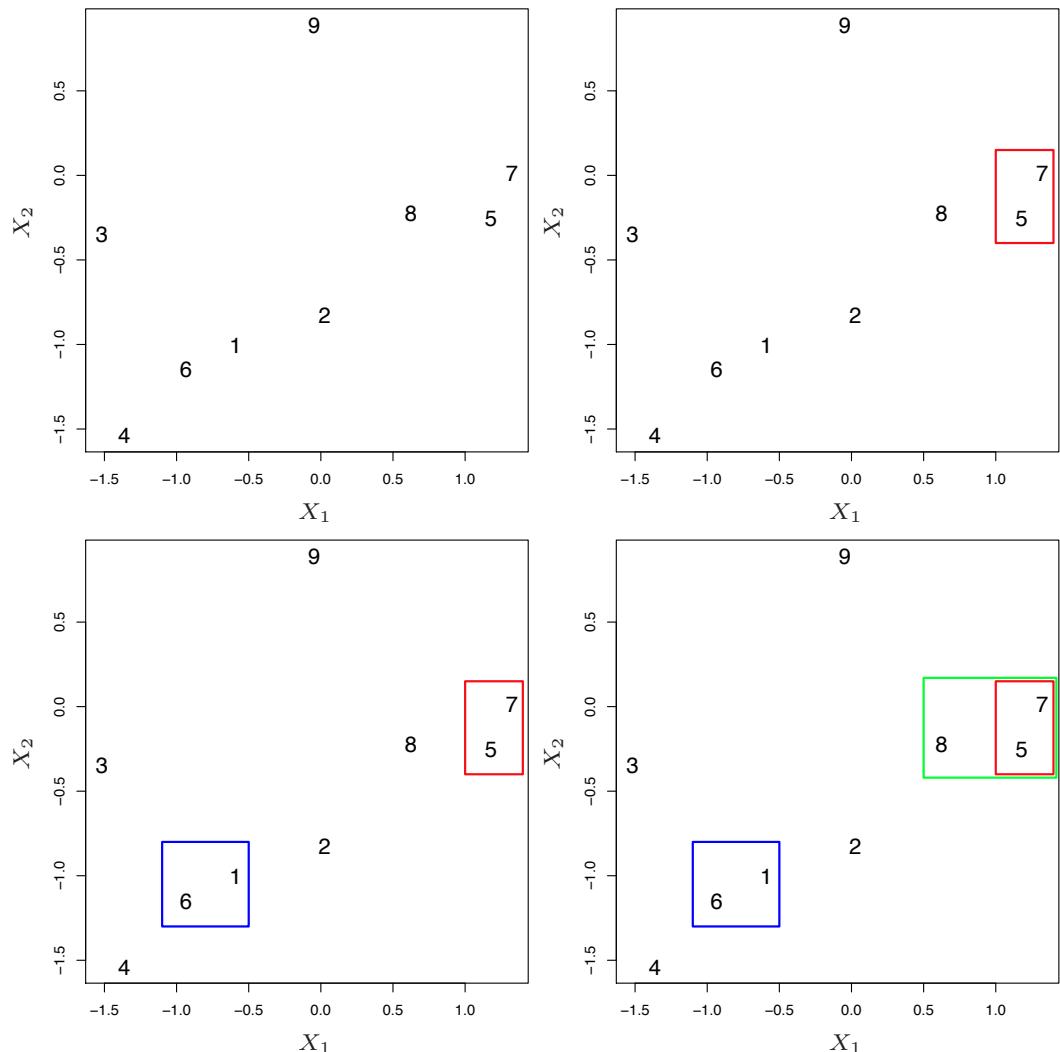
Giải thuật (trộn các cụm)

Phân cụm bằng cấu trúc cây:

- Khởi tạo với mỗi điểm là 1 cụm riêng biệt (n cụm), chính là 1 nút trong dendrogram
- Tính toán độ tương tự (gần) giữa các điểm/cụm
- Hợp nhất 2 cụm mà chúng có độ tương tự cao nhất, ta còn lại $n-1$ cụm
- Hợp nhất 2 cụm tiếp theo có độ tương tự cao nhất, ta còn lại $n-2$ cụm
- Quá trình trên tiếp tục cho đến khi chỉ còn 1 cụm (là nút gốc trong dendrogram)

Ví dụ

Bắt đầu với 9 cụm
Hợp nhất 5 và 7
Hợp nhất 6 và 1
Hợp nhất cụm (5,7) với 8.
Quá trình tiếp tục cho
đến khi tất cả các cụm
được hợp nhất.



Ta định nghĩa sự khác biệt ntn?

Việc triển khai phương pháp phân cấp cần giải quyết vấn đề khá hiển nhiên, đó là làm sao để định nghĩa sự khác biệt (dissimilarity) hoặc mối liên kết (linkage) giữa cụm hợp nhất (5, 7) và cụm 8?

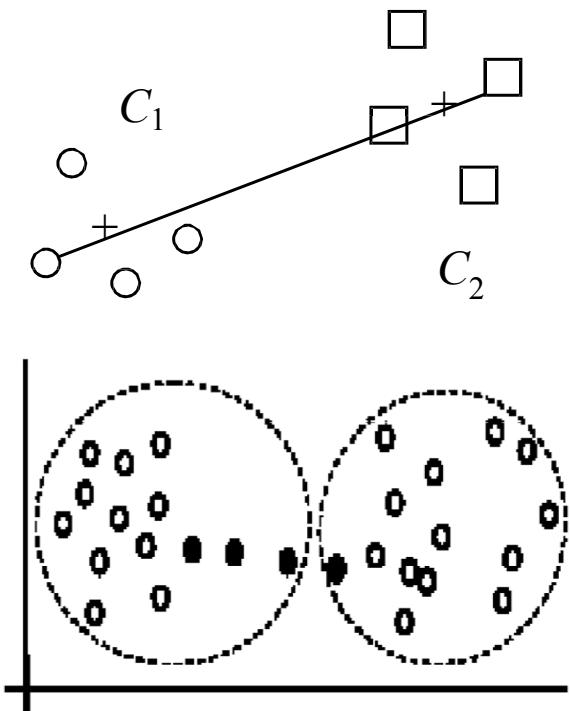
Có 4 lựa chọn:

- Liên kết đầy (Complete Linkage)
- Liên kết đơn (Single Linkage)
- Liên kết trung bình giữa các nhóm (Average Linkage)
- Liên kết tâm (Centroid Linkage)

Các phương pháp liên kết

Liên kết đầm: Khoảng cách giữa 2 cụm là khoảng cách lớn nhất giữa 2 mẫu tương ứng của 2 cụm đó

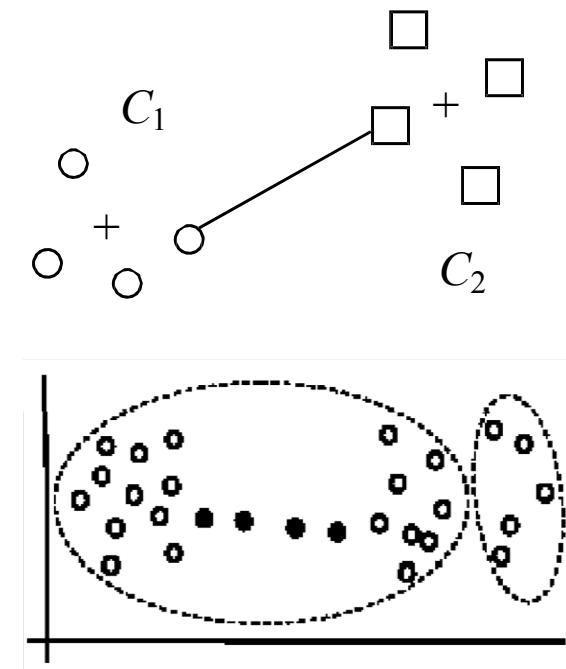
- Nhạy cảm (gặp lỗi phân cụm) đối với các ngoại lai (outliers)
- Có xu hướng sinh ra các cụm có dạng “bụi cây” (clumps)



[Liu, 2006]

Các phương pháp liên kết

Liên kết đơn: Khoảng cách giữa 2 cụm là khoảng cách nhỏ nhất giữa các mẫu (các thành viên) của 2 cụm đó. Có xu hướng sinh ra các cụm có dạng “chuỗi dài” (long chain)



[Liu, 2006]

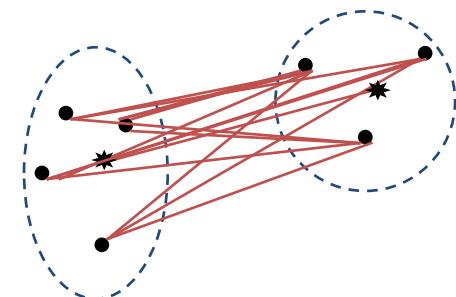
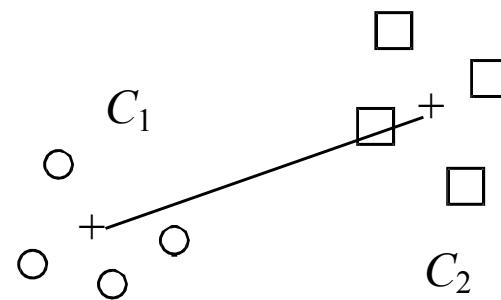
Các phương pháp liên kết

Liên kết trung bình: Khoảng cách trong liên kết trung bình (Average-link) là sự thỏa hiệp giữa các khoảng cách trong liên kết hoàn toàn (Complete-link) và liên kết đơn (Single-link)

- Để giảm mức độ nhạy cảm (khả năng lỗi) của phương pháp phân cụm dựa trên liên kết đầy đối với các ngoại lai (outliers)
- Để giảm xu hướng sinh ra các cụm có dạng “chuỗi dài” của phương pháp phân cụm dựa trên liên kết đơn (dạng “chuỗi dài” không phù hợp với khái niệm tự nhiên của một cụm)
- Khoảng cách giữa 2 cụm là khoảng cách trung bình của tất cả các cặp mẫu (mỗi mẫu thuộc về một cụm)

Các phương pháp liên kết

Liên kết tâm: Khoảng cách giữa các tâm của các mẫu tương ứng

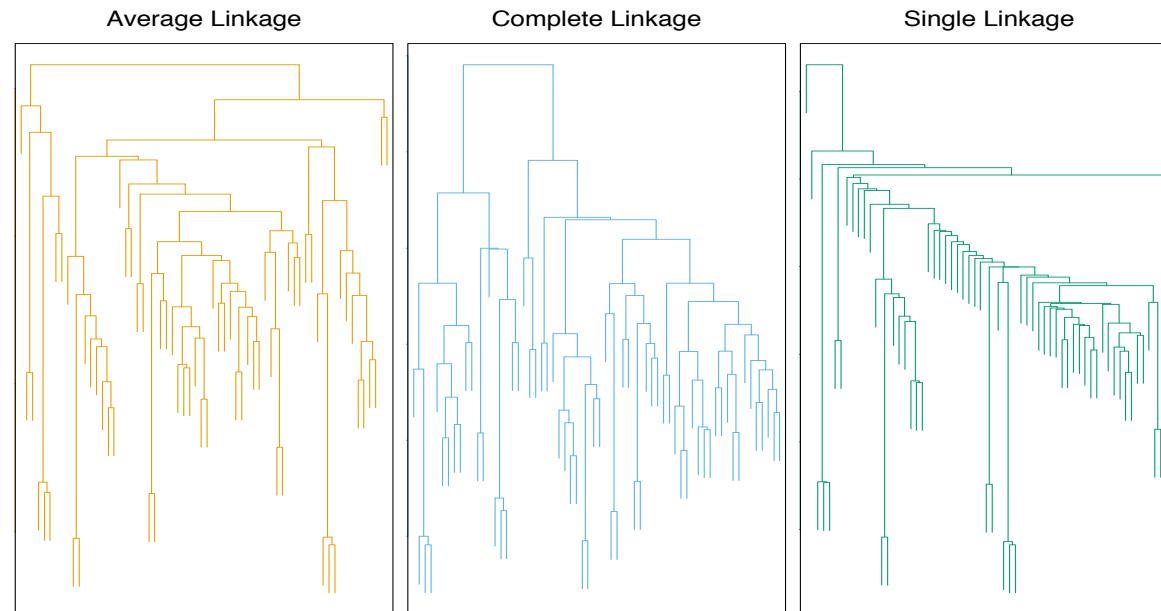


Mối liên kết rất quan trọng

Dưới đây ta có 3 kết quả phân cụm trên cùng 1 bộ dữ liệu

Phương pháp tính mối liên kết khác nhau nhưng kết quả đem lại rất khác xa nhau

Phương pháp liên kết đầy và liên kết trung bình đường như có cỡ cụm như nhau, tuy nhiên liên kết đơn lại cho số cụm nhiều hơn vì mỗi lá của cây được hợp nhất từng lần một



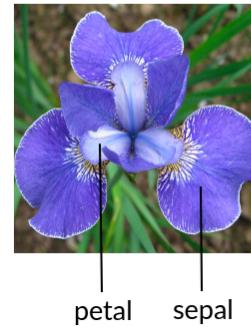
Clustering in Python

S.Length	S.Width	P.Length	P.Width	Species
5.1	3.5	1.4	0.2	I.setosa
4.9	3	1.4	0.2	I.setosa
4.7	3.2	1.3	0.2	I.setosa
4.6	3.1	1.5	0.2	I.setosa
5	3.6	1.4	0.2	I.setosa
5.4	3.9	1.7	0.4	I.setosa
4.6	3.4	1.4	0.3	I.setosa
5	3.4	1.5	0.2	I.setosa
4.4	2.9	1.4	0.2	I.setosa
5.1	3.8	1.9	0.4	I.setosa
4.8	3	1.4	0.3	I.setosa
5.1	3.8	1.6	0.2	I.setosa
4.6	3.2	1.4	0.2	I.setosa
5.3	3.7	1.5	0.2	I.setosa
5	3.3	1.4	0.2	I.setosa
7	3.2	4.7	1.4	I.versicolor
6.4	3.2	4.5	1.5	I.versicolor
6.9	3.1	4.9	1.5	I.versicolor
5.5	2.3	4	1.3	I.versicolor
6.5	2.8	4.6	1.5	I.versicolor
5.7	2.8	4.5	1.3	I.versicolor
6.3	3.3	4.7	1.6	I.versicolor
4.9	2.4	3.3	1	I.versicolor
6.6	2.9	4.6	1.3	I.versicolor
5.2	2.7	3.9	1.4	I.versicolor
5	2	3.5	1	I.versicolor
6.2	2.8	4.8	1.8	I.virginica
6.1	3	4.9	1.8	I.virginica
6.4	2.8	5.6	2.1	I.virginica
7.2	3	5.8	1.6	I.virginica
7.4	2.8	6.1	1.9	I.virginica
7.9	3.8	6.4	2	I.virginica

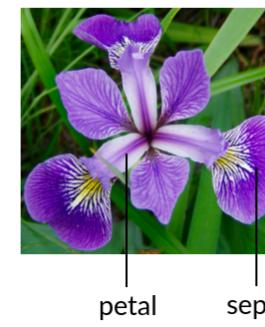
Fisher's iris data (famous)

- Có 3 loài hoa: setosa, versicolor, virginica
- 4 tiêu chí: S.Length S.Width P.Length, P.Width
- Mục tiêu: dựa vào tiêu chí để phân loại các loài hoa

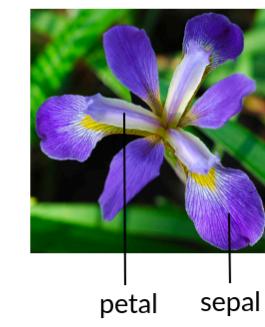
iris setosa



iris versicolor



iris virginica



```
iris = pd.read_csv('Iris.csv')
x = iris.iloc[:, [0, 1, 2, 3]].values
```

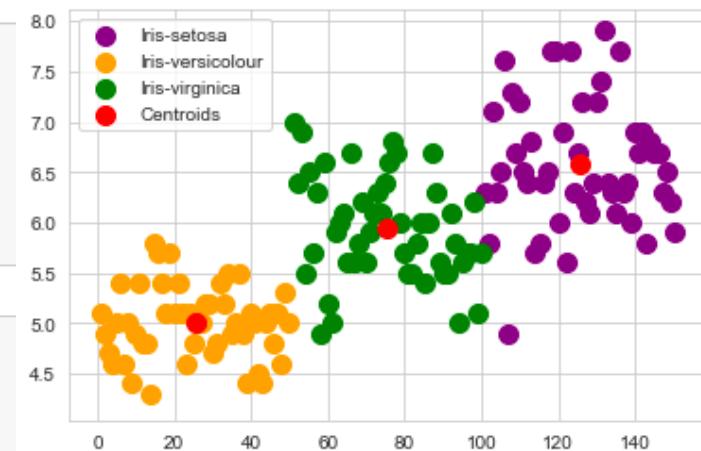
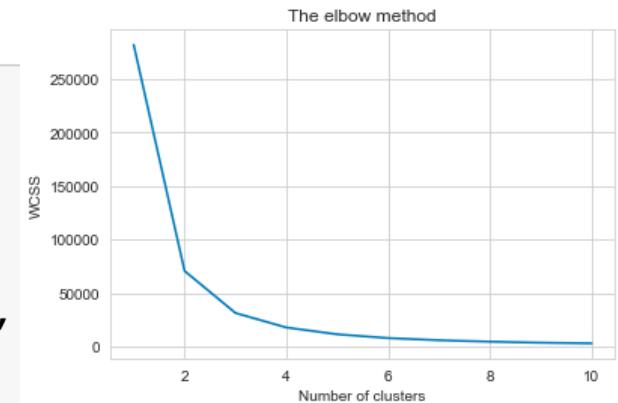
Clustering in Python

```
#Finding the optimum number of clusters for k-means classification
from sklearn.cluster import KMeans
wcss = []

for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter = 300,
                    n_init = 10, random_state = 0)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
```

```
plt.plot(range(1, 11), wcss)
plt.title('The elbow method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS') #within cluster sum of squares
plt.show()
```

```
kmeans = KMeans(n_clusters = 3, init = 'k-means++', max_iter = 300,
                 n_init = 10, random_state = 0)
y_kmeans = kmeans.fit_predict(x)
```



```
data = load_iris()
df = data.data
#Selecting certain features based on which clustering is done
df = df[:,1:3]

#Creating the model

agg_clustering = AgglomerativeClustering(n_clusters = 3, affinity = 'euclidean', linkage = 'ward')

#predicting the labels

labels = agg_clustering.fit_predict(df)

#Plotting the results

plt.figure(figsize = (8,5))
plt.scatter(df[labels == 0 , 0] , df[labels == 0 , 1] , c = 'red')
plt.scatter(df[labels == 1 , 0] , df[labels == 1 , 1] , c = 'blue')
plt.scatter(df[labels == 2 , 0] , df[labels == 2 , 1] , c = 'green')
plt.show()
```

Nguồn: <https://www.askpython.com/python/examples/hierarchical-clustering>

```

#Importing libraries
from sklearn.datasets import load_iris
from sklearn.cluster import AgglomerativeClustering
import numpy as np
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram , linkage

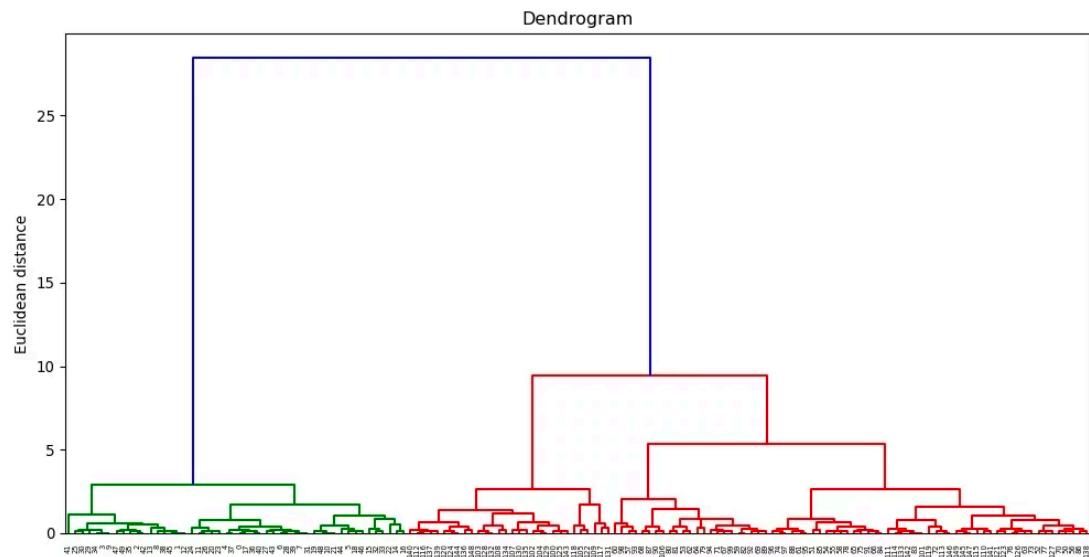
#Getting the data ready

data = load_iris()
df = data.data
#Selecting certain features based on
df = df[:,1:3]

#Linkage Matrix
Z = linkage(df, method = 'ward')

#plotting dendrogram
dendro = dendrogram(Z)
plt.title('Dendrogram')
plt.ylabel('Euclidean distance')
plt.show()

```



Nguồn: <https://www.askpython.com/python/examples/hierarchical-clustering>

Tóm lược

- Phân tích cụm: Phương pháp rất hữu hiệu để "nhận dạng" nhóm dựa vào các biến quan sát
- Chủ yếu dựa vào khoảng cách giữa các nhóm dựa trên dữ liệu thực tế
- Một phương pháp rất có ích trong thời đại "Big Data"

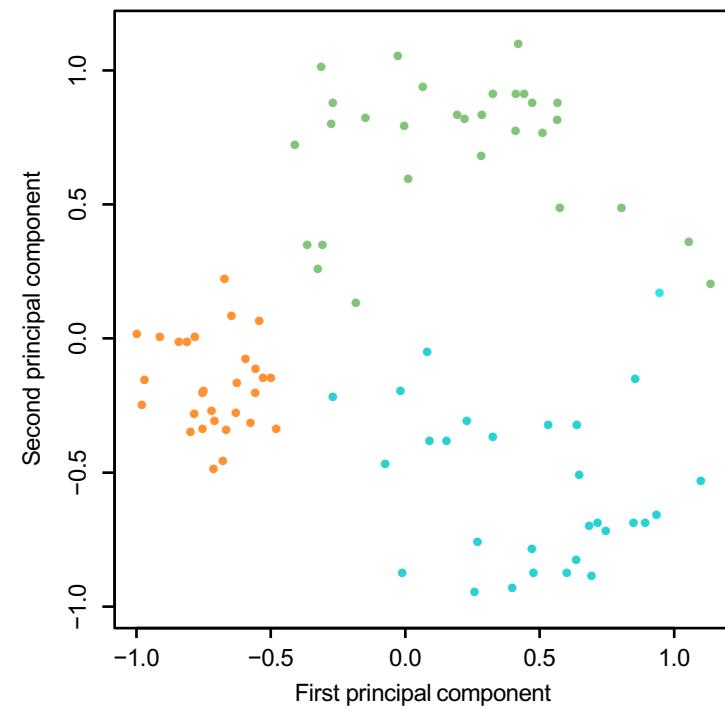
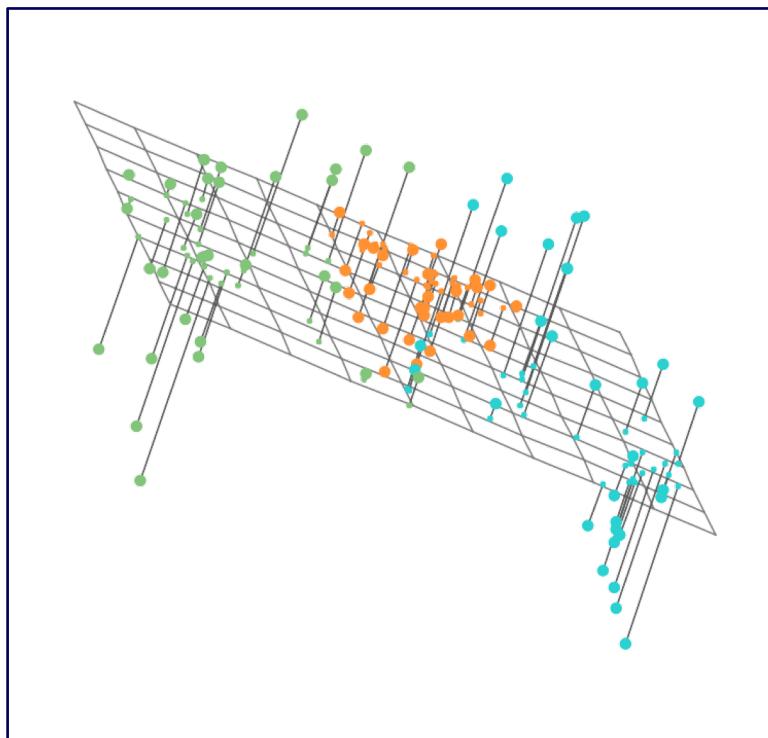
Câu hỏi?

Phân tích thành phần chính

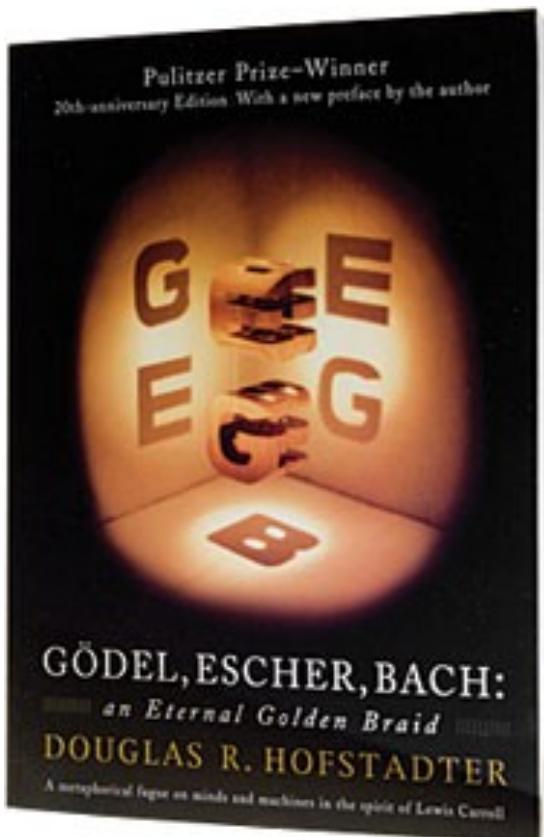
Principal Component Analysis (PCA)

Các slides ở mục này tham khảo từ bài giảng của GS. Nguyễn Văn Tuấn

Giảm chiều dữ liệu



Phép chiếu



Phân tích thành phần chính

- Đề xướng bởi Pearson (1901) và Hotelling (1933)
- Mô tả sự biến thiên của nhiều biến số có liên quan nhau bằng ***một nhóm biến số mới không liên quan đến nhau***

Phân tích thành phần chính

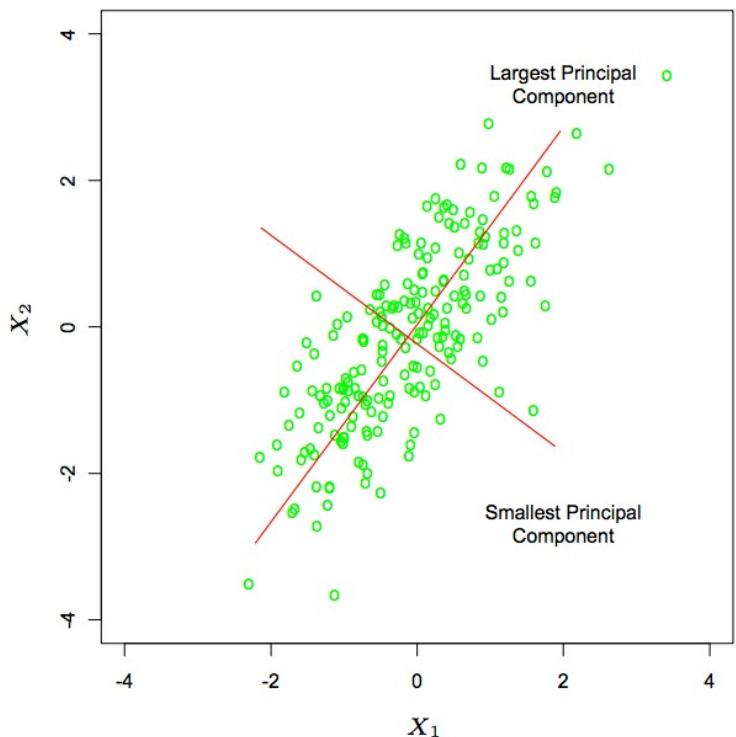
- Nếu hai biến (hay 2 items) có tương quan với nhau
 - Chúng có thể phản ảnh một hiện tượng tiềm ẩn (hay một yếu tố không quan sát được – latent factor)
 - Nếu chúng phản ảnh một latent variable, thì tổng hợp chúng thành 1 biến là hợp lí
- Các biến latent variables còn gọi là "factors" hay "**principal components**"

Phân tích thành phần chính

- Trong không gian mới, các liên kết tiềm ẩn của dữ liệu có thể được khám phá
- Ví dụ: Thị trường ta quan tâm có hàng ngàn mã cổ phiếu làm cách nào để khi quan sát dữ liệu từ hàng ngàn cổ phiếu này ta hình dung được xu hướng của toàn thị trường...

nguồn: <http://phvu.net/>

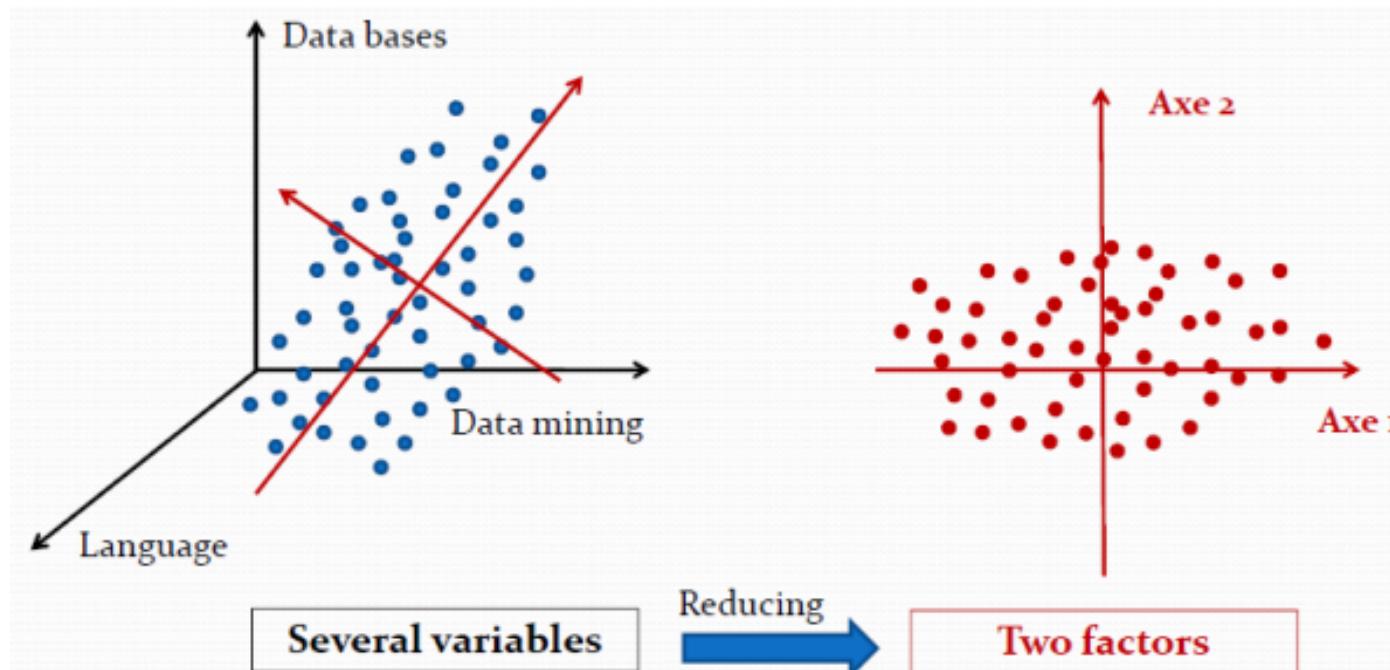
Phân tích thành phần chính



Minh họa PCA: phép chiếu lên các trục tọa độ khác nhau có thể cho cách nhìn rất khác nhau về cùng một dữ liệu.

nguồn: <http://phvu.net/>

Phân tích thành phần chính



Giả sử tập dữ liệu ban đầu (tập điểm màu xanh) được quan sát trong không gian 3 chiều (trục màu đen) như hình bên trái. Rõ ràng 3 trục này không biểu diễn được tốt nhất mức độ biến thiên của dữ liệu. PCA do đó sẽ tìm hệ trục tọa độ mới (là hệ trục màu đỏ trong hình bên trái). Sau khi tìm được không gian mới, dữ liệu sẽ được chuyển sang không gian này để được biểu diễn như trong hình bên phải. Rõ ràng hình bên phải chỉ cần 2 trục tọa độ nhưng biểu diễn tốt hơn mức độ biến thiên của dữ liệu so với hệ trục 3 chiều ban đầu.

nguồn: <http://phvu.net/>

Bối cảnh và dữ liệu

- Bối cảnh: chúng ta có một ma trận dữ liệu gồm n hàng và p biến số x_1, x_2, \dots, x_p
- PCA tìm cách hoán chuyển các x_i thành p biến mới (y_i) nhưng không có liên quan với nhau!

Khi các biến liên quan đến nhau

- Cách đơn giản nhất là chỉ lưu lại 1 biến duy nhất (bỏ các biến còn lại) – không hợp lý
- Cho trọng số mỗi biến. Trọng số nào?
- Tiêu chuẩn nào?

Khi các biến liên quan đến nhau

- Tìm phương pháp hoán chuyển ma trận \mathbf{X} ($n \times p$) sao cho

$$Y = \boldsymbol{\delta}^T \mathbf{X} = \delta_1 X_1 + \delta_2 X_2 + \dots + \delta_p X_p$$

Trong đó: $\boldsymbol{\delta} = (\delta_1, \delta_2, \dots, \delta_p)^T$ là cột vector gồm trọng số sao cho:

$$\delta_1^2 + \delta_2^2 + \dots + \delta_p^2 = 1$$

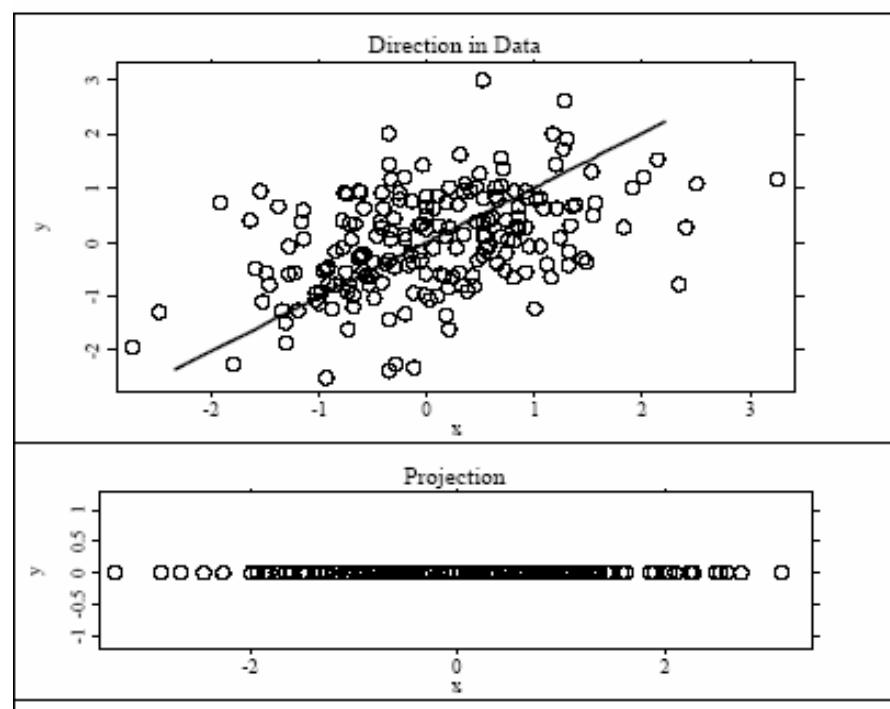
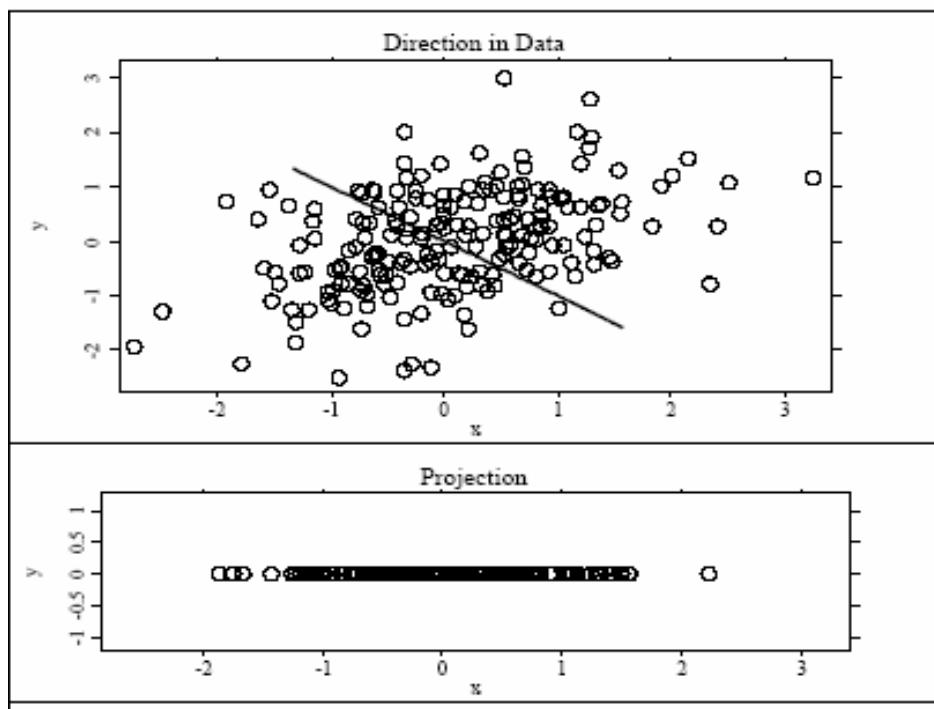
Tiêu chuẩn

- Tối đa hoá phương sai của dữ liệu dựa trên các biến Y
- Tìm δ sao cho

$$\text{Var}(\delta^T \mathbf{X}) = \delta^T \text{Var}(\mathbf{X}) \delta \text{ tối đa}$$

- Ma trận $\mathbf{C} = \text{Var}(\mathbf{X})$ là hiệp biến (covariance) của các biến X_i

Thử tưởng tượng ...



Variance-covariance matrix

$$\begin{pmatrix} v(x_1) & c(x_1, x_2) & \dots & c(x_1, x_p) \\ c(x_1, x_2) & v(x_2) & \dots & c(x_2, x_p) \\ \vdots & \vdots & \ddots & \vdots \\ c(x_1, x_p) & c(x_2, x_p) & \dots & v(x_p) \end{pmatrix}$$

Có nghĩa là chúng ta tìm ...

- Hướng của δ được xác định bởi véc tơ riêng (vector eigen) γ_1 tương đương với giá trị riêng (eigen) lớn nhất của ma trận C
- Vector thứ 2 cũng trực giao (orthogonal, tức không liên quan) với vector 1
- v.v.

Do đó, PCA cung cấp

- Một nhóm biến mới (Y_i) là hàm số tuyến tính của các biến X_j :

$$Y_i = a_{i1}X_1 + a_{i2}X_2 + \dots + a_{ip}X_p ; i = 1, 2, \dots, p$$

- Biến mới Y_i được tạo ra theo mức độ quan trọng nhưng suy giảm theo độ quan trọng
- Chúng được gọi là "**principal components**"

Tính toán eigenvalue và eigenvector

- Giá trị eigen λ_i được xác định bằng cách giải phương trình
$$\det(C - \lambda I) = 0$$
- Vector eigen là những cột của ma trận A với đặc điểm

$$C = A D A^T$$

- Trong đó

$$D = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \dots & \lambda_p \end{pmatrix}$$

Diễn giải PCA

- Những biến mới (Principal Components) có phương sai bằng giá trị eigen:

$$Var(Y_i) = \lambda_i \text{ cho tất cả } i = 1, 2, \dots, p$$

- Giá trị λ_i nhỏ \Leftrightarrow phương sai thấp \Leftrightarrow dữ liệu thay đổi nhỏ về hướng của Y_i
- Mức độ quan trọng của mỗi PC given by $\lambda_i / \sum \lambda_i$

Cần bao nhiêu PC?

- Số PCs sao cho tỉ lệ phương sai giải thích được từ 50-70%
- Kaiser criterion: giữ PCs với eigenvalues >1
- Scree plot: thể hiện khả năng PC giải thích phương sai của dữ liệu

Diễn giải ý nghĩa của PC

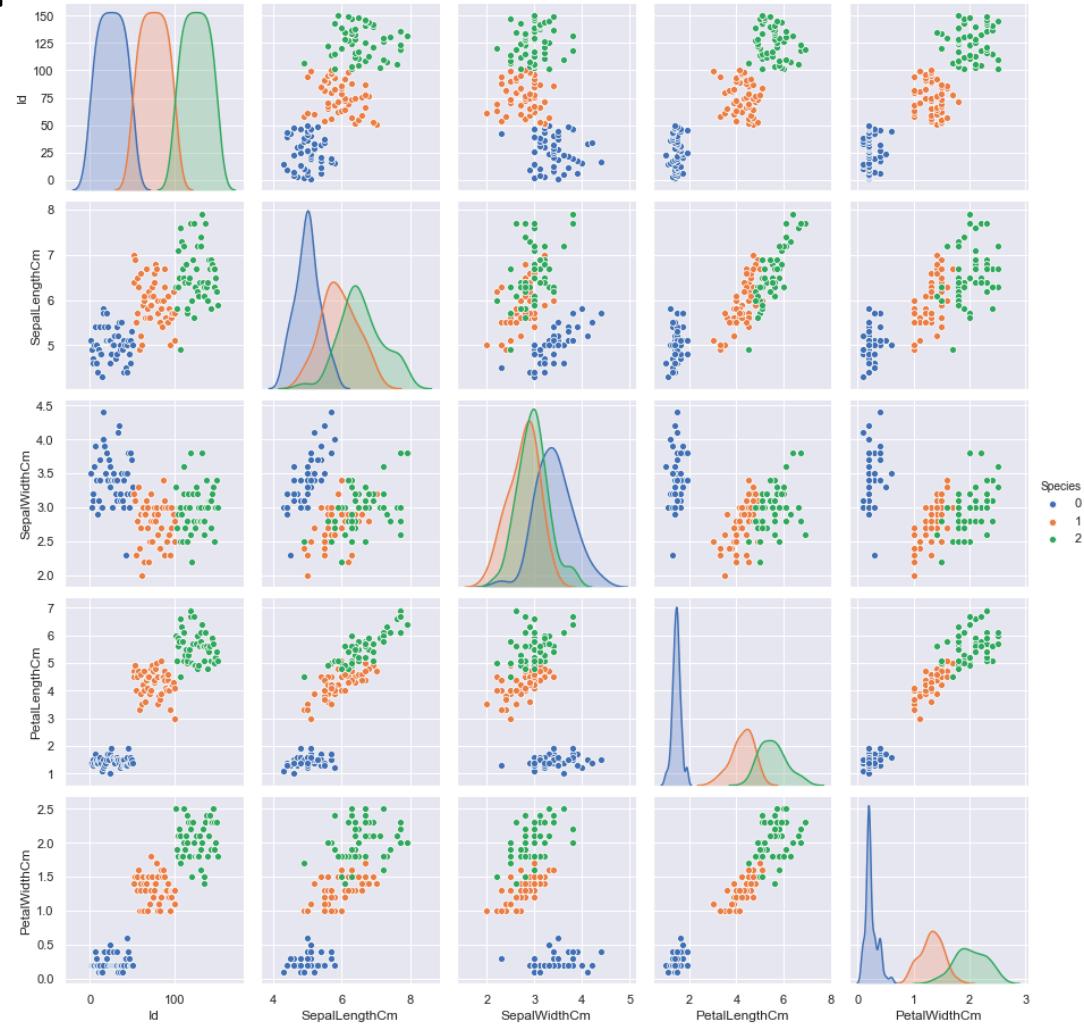
- Trọng số của biến số trong mỗi PC:
- Nếu $Y_1 = 0.89X_1 + 0.15X_2 - 0.77X_3 + 0.51X_4$
- Thì X_1 và X_3 có trọng số cao nhất, và là biến quan trọng nhất
- Xem mối tương quan giữa các biến X_i và PC

Các bước phân tích

1. Chuẩn bị dữ liệu (chuẩn hoá dữ liệu)
2. Tính ma trận covariance hoặc correlation
3. Tính eigenvalues của ma trận covariance
4. Chọn components

Quan sát dữ liệu

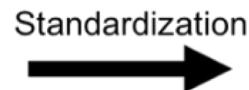
```
import seaborn as sns  
p=sns.pairplot(iris, hue = 'Species')
```



Bước 1: Chuẩn hoá data

```
from sklearn.preprocessing import MinMaxScaler  
scaler = MinMaxScaler()  
X=scaler.fit_transform(X)  
X_train, X_test, y_train, y_test = train_test_split(X, y,  
test_size = 0.4, random_state=20, stratify=y)
```

	sepal length	sepal width	petal length	petal width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

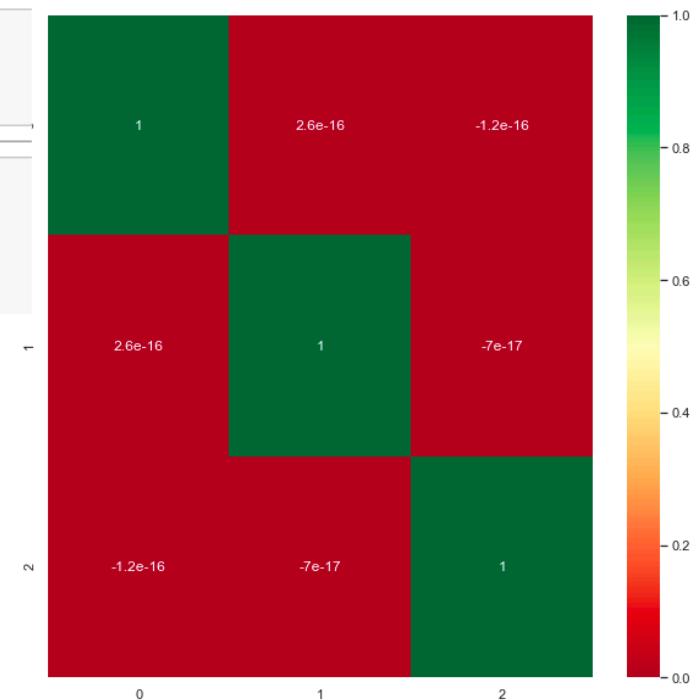
Standardization 

	sepal length	sepal width	petal length	petal width
0	-0.900681	1.032057	-1.341272	-1.312977
1	-1.143017	-0.124958	-1.341272	-1.312977
2	-1.385353	0.337848	-1.398138	-1.312977
3	-1.506521	0.106445	-1.284407	-1.312977
4	-1.021849	1.263460	-1.341272	-1.312977

Bước 2: Tính toán hệ số tương quan

```
pca=PCA(n_components=3)  
X_new=pca.fit_transform(X)
```

```
plt.figure(figsize=(10,10))  
corr_x = pd.DataFrame(X_new)  
p=sns.heatmap(corr_x.corr(), annot=True,cmap='RdYlGn' )
```



Bước 3: Tính toán Covariance

```
pca.get_covariance()

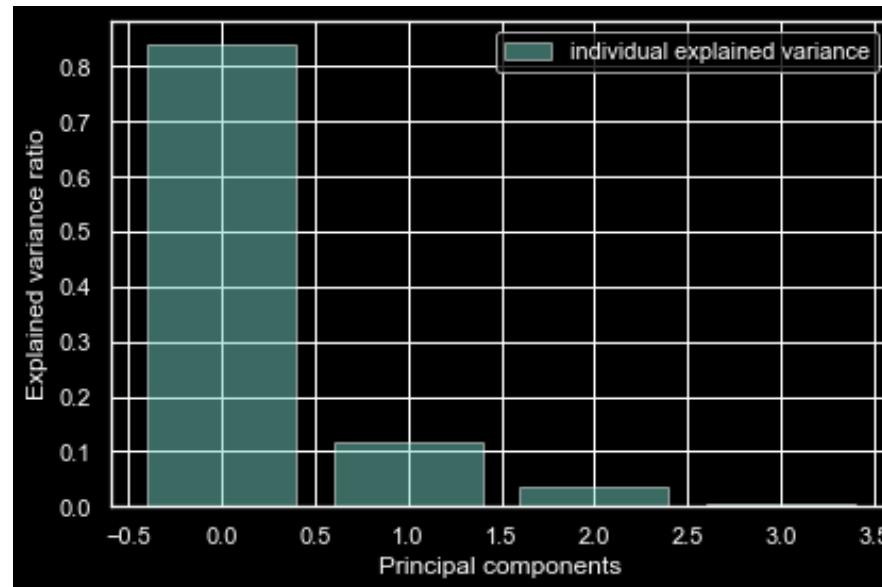
array([[ 0.05290845, -0.00454496,  0.05996621,  0.05982683],
       [-0.00454496,  0.03263959, -0.02271983, -0.02048285],
       [ 0.05996621, -0.02271983,  0.08943348,  0.09155279],
       [ 0.05982683, -0.02048285,  0.09155279,  0.1011136 ]])

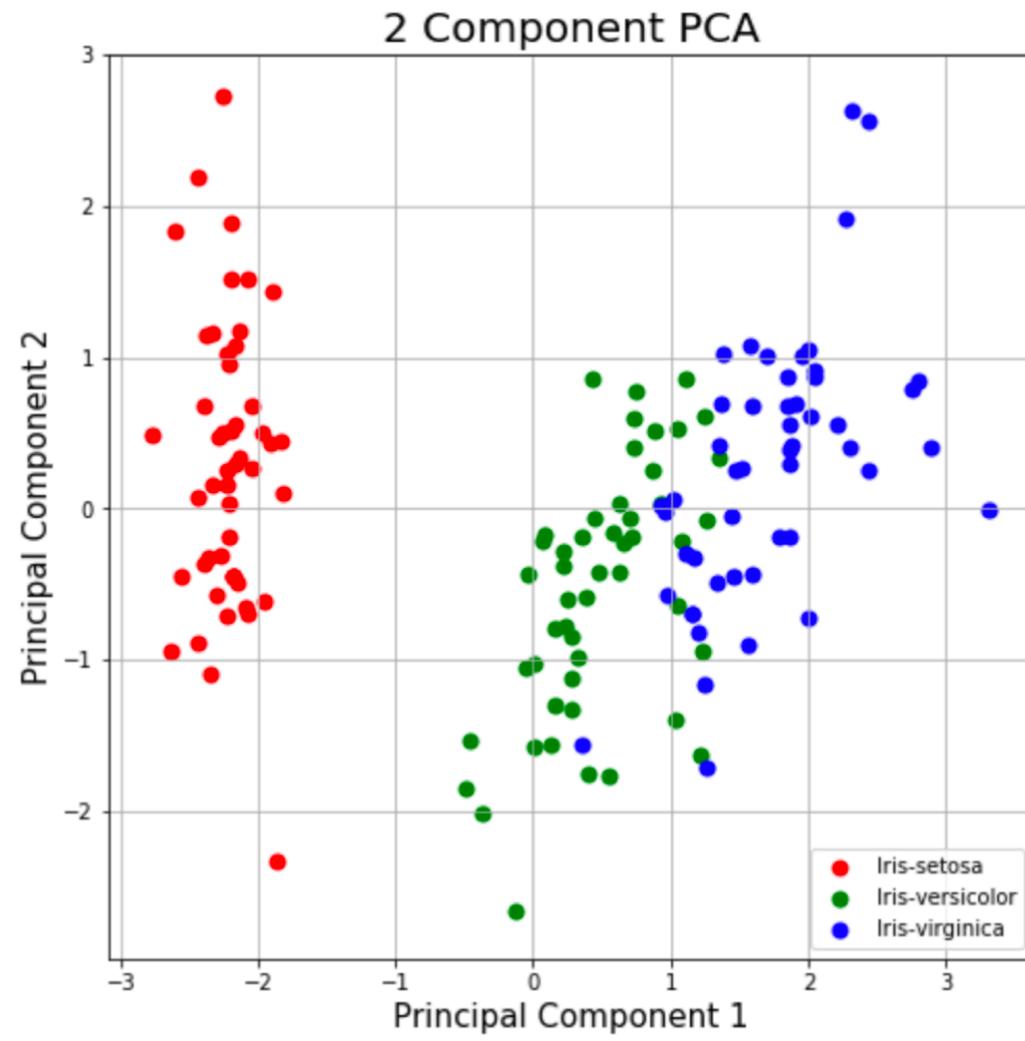
explained_variance=pca.explained_variance_ratio_
explained_variance

array([0.84141901, 0.11732474, 0.03490564, 0.00635061])
```

Bước 4: Phân tích PC

```
pca=PCA(n_components=3)  
X_new=pca.fit_transform(X)
```





Tóm tắt

- PCA là một phương pháp giảm độ liên quan đa chiều của dữ liệu
- Ý tưởng: tìm các biến số mới (PC) là hàm số của các biến số gốc sao cho các PC không tương quan với nhau (orthogonal)
- PC đầu tiên giải thích nhiều phương sai nhất; PC 2 giải thích tỉ lệ phương sai ít hơn PC 1, v.v.

Questions?