

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO
NGÀNH CÔNG NGHỆ THÔNG TIN



BÁO CÁO ĐỒ ÁN 1

TÌM HIỂU THUẬT TOÁN K-MEANS VÀ ỨNG DỤNG

GVHD : TS. Lê Văn Vinh
SVTH1 : Võ Gia Huy
SVTH2 : Võ Minh Huy
LỚP : 16110CL2

MSSV
16110092
16110093

Tp. Hồ Chí Minh, tháng 10 năm 2018

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO
NGÀNH CÔNG NGHỆ THÔNG TIN



BÁO CÁO ĐỒ ÁN 1

TÌM HIỂU THUẬT TOÁN K-MEANS VÀ ỨNG DỤNG

GVHD : TS. Lê Văn Vinh
SVTH1 : Võ Gia Huy
SVTH2 : Võ Minh Huy
LỚP : 16110CL2

MSSV
16110092
16110093

Tp. Hồ Chí Minh, tháng 10 năm 2018

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Giáo viên hướng dẫn

(ký và ghi họ tên)

NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Giáo viên phản biện

(ký và ghi họ tên)

LỜI CẢM ƠN

Trên thực tế không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù ít hay nhiều, dù trực tiếp hay gián tiếp của người khác. Nay chúng em xin được phép gửi lời cảm ơn chân thành này đến thầy Lê Văn Vinh, người đã trực tiếp hỗ trợ chúng em trong suốt quá trình định hướng chọn đề tài, hướng dẫn, nhận xét và góp ý cũng như cung cấp tài liệu tham khảo. Nếu không có những lời hướng dẫn, những kinh nghiệm thực tiễn của thầy thì chúng em nghĩ rằng bài thu hoạch này sẽ khó có thể hoàn thiện và hoàn thành đúng thời hạn được. Một lần nữa, chúng em xin cảm ơn thầy.

Chúng em cũng xin chân thành cảm ơn các quý thầy cô trong khoa Công Nghệ Thông Tin đã giúp đỡ hỗ trợ kiến thức cũng như giải đáp thắc mắc của chúng em. Cùng với đó, chúng em xin được gửi cảm ơn đến các bạn cùng khóa đã cung cấp nhiều thông tin và kiến thức hữu ích giúp cho chúng em hoàn thiện đề tài hơn.

Bài thu hoạch được thực hiện trong khoảng thời gian gần 10 tuần. Khoảng thời gian có hạn, cùng với kiến thức còn hạn chế và còn nhiều bờ ngõ khác do đó thiếu sót là điều không thể tránh khỏi nên chúng em rất mong nhận được những ý kiến đóng góp quý báu của các quý Thầy Cô để kiến thức của chúng em được hoàn thiện hơn sau này. Chúng em xin chân thành cảm ơn.

Thành phố Hồ Chí Minh, ngày 30 tháng 10 năm 2018

Sinh viên thực hiện 1

Sinh viên thực hiện 2

(ký và ghi họ tên)

(ký và ghi họ tên)

TÓM TẮT

Trong những năm trở lại đây, công nghệ thông tin đã và đang không ngừng phát triển một cách mạnh mẽ. Sự ra đời của công nghệ thông tin đã một phần nào giúp cho xã hội, đời sống của người dân vươn lên một tầm cao mới và một cách rõ rệt, ngoài ra công nghệ thông tin còn góp phần không nhỏ cho sự phát triển của nhân loại khắp nơi trên thế giới.

Trong thế kỷ 21 như hiện nay, trí tuệ nhân tạo là một lĩnh vực rất nóng và rất có tiềm năng phát triển do việc tính toán của máy lúc nào cũng nhanh và chính xác hơn con người. Tại sự kiện Google I/O 2018 vừa diễn ra, “gã khổng lồ tìm kiếm” đã có màn trình diễn AI đầy ấn tượng trước hàng triệu người có mặt tại sân khấu cũng như qua các phương tiện truyền thông. Những người có mặt trực tiếp đã phải “Ồ” lên đầy ngạc nhiên, thích thú khi trợ lý ảo Google Assistant thực hiện cuộc gọi đặt lịch cắt tóc mà phải người nghe không hề biết đang nói chuyện với một AI. Sự thông minh và linh hoạt của AI được kỳ vọng sẽ giúp ích cho con người rất nhiều trong tương lai, giúp tăng năng suất và hiệu quả lao động. Do đó chúng em đã chọn đề tài **“Tìm hiểu về thuật toán k-means và ứng dụng”** để một phần biết được một phần kiến thức về trí tuệ nhân tạo, cách thức hoạt động và những ứng dụng của chúng trong cuộc sống.

Mục lục

1. Mô tả project.....	10
2. Quá trình thực hiện Project.....	10
2.1. Tìm hiểu sơ lược về thuật toán k-means	10
2.1.1. Thuật toán gom cụm.....	10
2.1.2. Định nghĩa thuật toán k-means	11
2.1.3. Sơ đồ khối.....	11
2.1.4. Thuật toán k-means	11
2.2. Mẫu thiết kế code	12
2.2.1. Liệt kê các lớp sử dụng trong chương trình.....	12
2.2.2. Mô tả phương thức code trong project	12
2.3. Trình bày phần đồ hoạ (Trình bày màn hình phần đồ hoạ).....	15
2.4. Ứng dụng của thuật toán k-means	23
3. Mô tả phân công công việc.....	28
4. Kết luận.....	28
5. Tài liệu tham khảo	28

Danh mục các hình

Hình 2.1 Sơ đồ khối k-means	11
Hình 2.2 Class Toạ Độ	15
Hình 2.3 Class Trọng Tâm	16
Hình 2.4 Khởi tạo và in ra các trọng tâm ngẫu nhiên.....	16
Hình 2.5 Khởi tạo toạ độ ban đầu.....	17
Hình 2.6 Tính khoảng cách tính trọng tâm.....	17
Hình 2.7 Cập nhật cluster và kết quả.....	18
Hình 2.8 Xuất ra trọng tâm và toạ độ	18
Hình 2.9 Kết quả trọng tâm lần đầu	19
Hình 2.10 Kết quả trọng tâm chạy lại lần 2 sau khi cập nhật cluster	19
Hình 2.11 Kết quả trọng tâm chạy lại lần 3 sau khi cập nhật cluster	20
Hình 2.12 Khởi tạo điểm để hiển thị lên trục toạ độ	21
Hình 2.13 Code thuật toán k-means	22
Hình 2.14 Các điểm sau khi phân cụm.....	22
Hình 2.15 Các điểm phân cụm bằng thư viện có sẵn	23
Hình 2.16 Vị trí các viên thuốc trên toạ độ	24
Hình 2.17 Vị trí cập nhật lần một các viên thuốc trên toạ độ.....	25
Hình 2.18 Vị trí cập nhật lần hai các viên thuốc trên toạ độ	26

Danh mục các bảng

Bảng 2.1 Liệt kê các lớp trong project	12
Bảng 2.2 Mô tả các phương thức code trong project	14
Bảng 2.3 Code và k-means trên toạ độ.....	23
Bảng 2.4 Thuộc và thuộc tính của thuốc	23
Bảng 2.5 Kết quả phân nhóm	27
Bảng 3.1 Mô tả phân công công việc	28

1. Mô tả project

Sản phẩm muốn xây dựng là 1 phần mềm có tính demo về trí tuệ nhân tạo k-means. Được sử dụng khi muốn chia một số điểm trên toạ độ thành 1 số nhóm. Mục tiêu của thuật toán này là tìm nhóm trong dữ liệu, với số lượng các nhóm đại diện bởi các biến K. Thuật toán hoạt động lặp lại để gán mỗi điểm dữ liệu cho một trong các nhóm K dựa trên các tính năng được cung cấp. Các điểm dữ liệu được phân cụm dựa trên tính tương tự về tính năng. Kết quả của thuật toán phân cụm K-means là: Các trung tâm của các cụm K, có thể được sử dụng để gán nhãn dữ liệu mới. Nhãn cho dữ liệu thử (mỗi điểm dữ liệu được gán cho một cụm duy nhất). Mỗi trọng tâm của một cụm là một tập hợp các giá trị tính năng mà nó xác định các nhóm kết quả. Kiểm tra trọng số tính năng trọng tâm có thể được sử dụng để phân tích chất lượng loại của nhóm từng đại diện.

2. Quá trình thực hiện Project

2.1. Tìm hiểu sơ lược về thuật toán k-means

2.1.1. Thuật toán gom cụm

Thuật toán gom cụm (Clustering) là một kỹ thuật Machine Learning có liên quan đến việc nhóm các điểm dữ liệu. Với một tập hợp các điểm dữ liệu, chúng ta có thể sử dụng thuật toán phân cụm để phân loại từng điểm dữ liệu thành một nhóm cụ thể. Về lý thuyết, các điểm dữ liệu nằm trong cùng một nhóm phải có các thuộc tính và các tính năng tương tự, trong khi các điểm dữ liệu trong các nhóm khác nhau phải có các thuộc tính và tính năng không giống nhau. Clustering là một phương pháp học tập không giám sát và là một kỹ thuật phổ biến để phân tích dữ liệu thống kê được sử dụng trong nhiều lĩnh vực.

Thuật toán clustering được chia làm 2 nhóm chính:

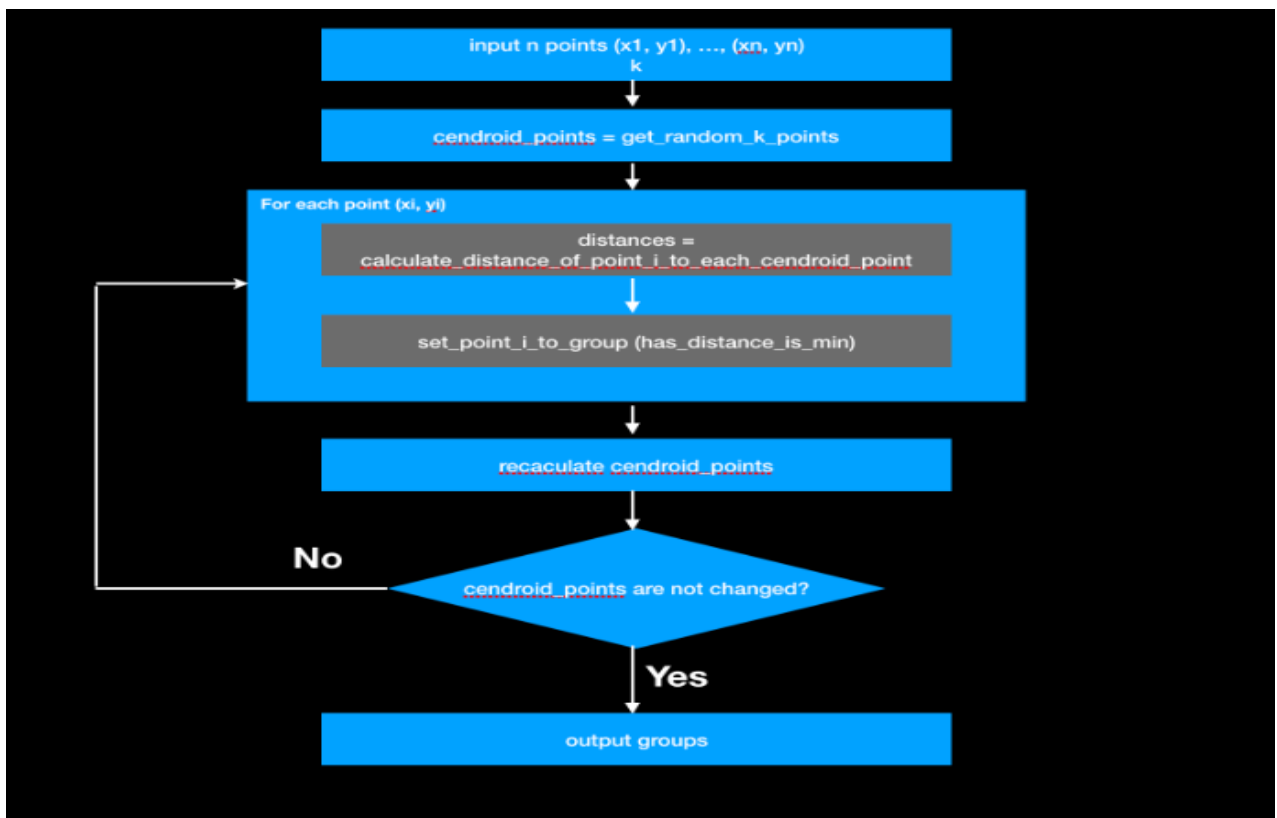
- distance-based clustering algorithms: thuật toán phân nhóm dựa trên khoảng cách.
- density-based clustering algorithms: thuật toán phân nhóm dựa trên mật độ

2.1.2. Định nghĩa thuật toán k-means

K-means nó là 1 thuật toán để phân nhóm 1 tập các đối tượng dựa trên các thuộc tính của chúng.

K-means thuộc nhóm distance-based clustering algorithms: thuật toán phân nhóm dựa trên khoảng cách

2.1.3. Sơ đồ khối



Hình 2.1 Sơ đồ khối k-means

2.1.4. Thuật toán k-means

Thuật toán k-means có 2 bước chính được lặp đi lặp lại cho đến khi trọng tâm không thay đổi:

Bước 1: Tính trọng tâm

Bước 2: Cập nhật Cluster

2.2. Mẫu thiết kế code

2.2.1. Liệt kê các lớp sử dụng trong chương trình

TT	Tên lớp	Tên các SV phụ trách viết.	Mục đích chính của lớp trong chương trình
1	Toạ độ	Võ Gia Huy	Khởi tạo các thuộc tính của toạ độ gồm x,y
2	Trọng tâm	Võ Minh Huy	Khởi tạo thuộc tính của trọng tâm cần phân cụm

Bảng 2.1 Liệt kê các lớp trong project

2.2.2. Mô tả phương thức code trong project

TT	Phương thức	Mục đích	Tên file, số thứ tự dòng chứa khai báo	Tên SV phụ trách viết
1	KhoiTaoTrongTam() Input: s1, s2 Output: trongtam[0].get_x, trongtam[1].get_x, trongtam[0].get_y, trongtam[1].get_y Pseudo code: không có vì đơn giản	Chọn ra hai trọng tâm ngẫu nhiên từ các điểm đã nhập trước	Kmeans.ipynb (In [3])	Võ Minh Huy

2	<p>KhoiTaoToaDo()</p> <p>Input: không có</p> <p>Output: data[j].get_x(), data[j].get_y(), data[j].get_cluster()</p> <p>Pseudo code: không có vì đơn giản</p>	Tạo Cluster ban đầu cho các điểm	Kmeans.ipynb (In [4])	Võ Gia Huy
3	<p>TinhKhoangCach(Xtrongtam, x, Ytrongtam, y)</p> <p>Input: Xtrongtam, x, Ytrongtam, y</p> <p>Output: không có</p> <p>Pseudo code: không có vì đơn giản</p>	Tính khoảng cách từ cách điểm đến trọng tâm	Kmeans.ipynb (In [5])	Võ Minh Huy
4	<p>TinhTrongTam()</p> <p>Input: TongToaDoX1 = 0 TongToaDoY1 = 0 TongToaDoX2 = 0 TongToaDoY2 = 0 SoDiemToaDo1 = 0 SoDiemToaDo2 = 0</p> <p>Output: trongtam[0].get_x trongtam[1].get_x trongtam[0].get_y trongtam[1].get_y</p> <p>Pseudo code: không có vì đơn giản</p>	Tính lại trọng tâm	Kmeans.ipynb (In [6])	Võ Minh Huy

5	<p>CapNhatCluster()</p> <p>Input: KhoảngCachNganNhat = 1000 Cluster = 0</p> <p>Output: data[i].get_x() data[i].get_y() data[i].get_cluster()</p> <p>Pseudo code: không có vì đơn giản</p>	Cap nhat lai cluster	Kmeans.ipynb (In [7])	Võ Minh Huy
6	<p>Kmeans()</p> <p>Input: không có</p> <p>Output: không có</p> <p>Pseudo code: không có vì đơn giản</p>	<p>chạy vòng lặp TinhTrongTam() và CapNhatCluster() đến khi trọng tâm không đổi thì dừng lại</p>	Kmeans.ipynb (In [8])	Võ Minh Huy
7	<p>inKetQua()</p> <p>Input: không có</p> <p>Output: data[j].get_x() data[j].get_y() data[j].get_cluster()</p> <p>Pseudo code: không có vì đơn giản</p>	xuất ra kết quả cuối cùng	Kmeans.ipynb (In [9])	Võ Minh Huy

Bảng 2.2 Mô tả các phương thức code trong project

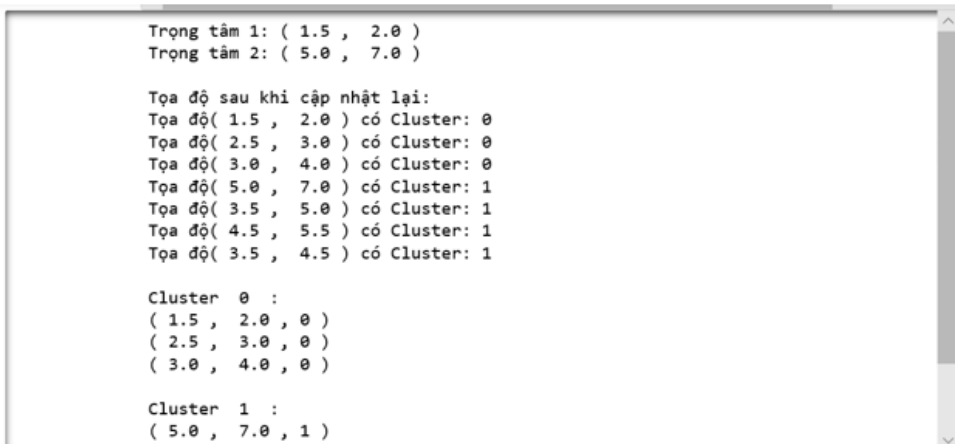
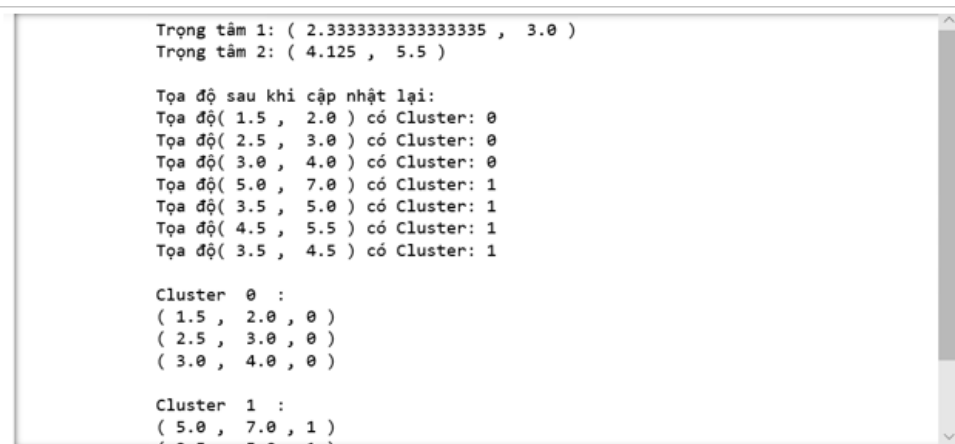
2.3. Trình bày phần đồ họa (Trình bày màn hình phần đồ họa)

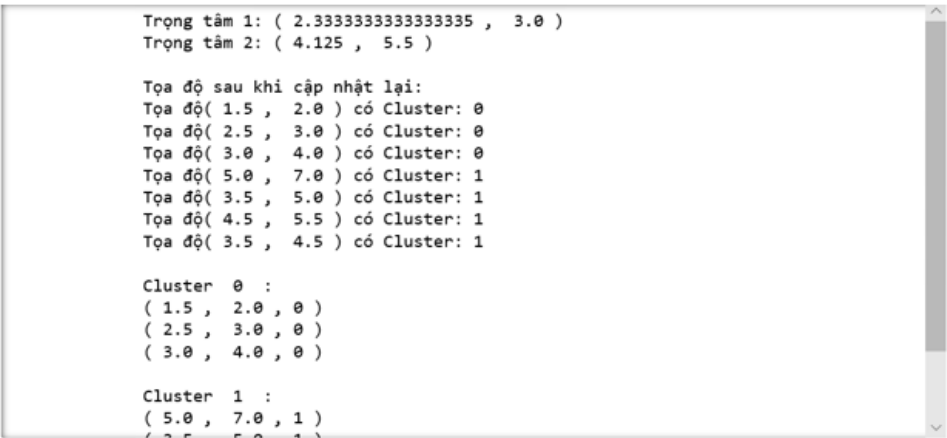
TT	Màn hình/Cửa sổ/Dialog	Người thiết kế & giải thích ngắn gọn	Mục đích chính của màn hình
1	<p>Màn hình code class tọa độ:</p> <pre> In [1]: import math ToaDoChoTruoc = [[1.5,2.0], [2.5,3.0], [3.0,4.0], [5.0,7.0], [3.5,5.0], [4.5,5.5], [3.5,4.5]] class ToaDo: def __init__(self, x, y): self.x = x self.y = y def set_x(self, x): self.x = x def get_x(self): return self.x def set_y(self, y): self.y = y def get_y(self): return self.y def set_cluster(self, cluster): self.cluster = cluster def get_cluster(self): return self.cluster </pre> <p><i>Hình 2.2 Class Tọa Độ</i></p>	Võ Minh Huy Khởi tạo class tọa độ	Cách import math tọa độ cho trước và tạo class tọa độ trên python
2	Màn hình code class Trọng Tâm:	Võ Gia Huy	Khai báo và tạo class

	<pre> In [2]: class TrongTam: def __init__(self, x, y): self.x = x self.y = y def set_x(self, x): self.x = x def get_x(self): return self.x def set_y(self, y): self.y = y def get_y(self): return self.y </pre> <p><i>Hình 2.3 Class Trọng Tâm</i></p>	Tạo class Trọng Tâm	trọng tâm
3	<p>Màn hình code Khởi tạo Trọng Tâm:</p> <pre> In [3]: trongtam = [] #ArrayList chứa các trọng tâm s = [] #ArrayList để lưu đỉnh của trọng tâm def KhoiTaoTrongTam(): s1 = int(input("Nhập đỉnh trọng tâm thứ 1: ")) s2 = int(input("Nhập đỉnh trọng tâm thứ 2: ")) s.append(s1) #Thêm đỉnh s1 vào s s.append(s2) #Thêm đỉnh s2 vào s trongtam.append(TrongTam(ToaDoChoTruoc[s[0]][0], ToaDoChoTruoc[s[0]][1])) #t trongtam.append(TrongTam(ToaDoChoTruoc[s[1]][0], ToaDoChoTruoc[s[1]][1])) #t print("Các trọng tâm được chọn ngẫu nhiên là: ") print("(", trongtam[0].get_x(), ", ", trongtam[0].get_y(), ")") print("(", trongtam[1].get_x(), ", ", trongtam[1].get_y(), ")") print() return </pre> <p><i>Hình 2.4 Khởi tạo và in ra các trọng tâm ngẫu nhiên</i></p>	Võ Minh Huy In ra trọng tâm ngẫu nhiên ngẫu nhiên	Cách tạo trọng tâm ngẫu nhiên và xuất ra màn hình
4	<p>Màn hình code khởi tạo tọa độ</p>	Võ Minh Huy Tạo các cluster	Lưu lại các điểm, khởi tạo tọa độ các điểm ban đầu

	<pre> In [4]: data = [] #ArrayList Lưu lại các điểm sau khi đã được khởi tạo cluster def KhoiTaoToaDo(): print("Tọa độ các điểm ban đầu : ") for i in range(len(ToaDoChoTruoc)): #ToaDoChoTruoc có length = 7 newPoint = ToaDo(ToaDoChoTruoc[i][0], ToaDoChoTruoc[i][1]) if(i == s[0]): newPoint.set_cluster(0) #Tạo cluster ban đầu cho trọng tâm 1 elif(i == s[1]): newPoint.set_cluster(1) #Tạo cluster ban đầu cho trọng tâm 2 else: newPoint.set_cluster(None) #Tạo cluster ban đầu cho các điểm tọa độ data.append(newPoint) for j in range(len(ToaDoChoTruoc)): print("Tọa độ(", data[j].get_x(), ", ", data[j].get_y(), ") có Cluster: ") print() return </pre> <p><i>Hình 2.5 Khởi tạo tọa độ ban đầu</i></p>	ban đầu	và in ra màn hình
5	<p>Màn hình code tính khoảng cách tính trọng tâm</p> <pre> In [5]: def TinhKhoangCanh(Xtrongtam, x, Ytrongtam, y): return math.sqrt(math.pow((Xtrongtam - x), 2) + math.pow((Ytrongtam - y), 2)) In [6]: def TinhTrongTam(): TongToaDoX1 = 0 TongToaDoY1 = 0 TongToaDoX2 = 0 TongToaDoY2 = 0 SoDiemToaDo1 = 0 SoDiemToaDo2 = 0 for j in range(2): for k in range(len(data)): if(data[k].get_cluster() == j): #Xét các điểm tọa độ thuộc cùng 1 cluster if(j == 0): TongToaDoX1 += data[k].get_x() #Tổng các tọa độ x trong cùng 1 cluster TongToaDoY1 += data[k].get_y() #Tổng các tọa độ y trong cùng 1 cluster SoDiemToaDo1 += 1 #Nếu có các điểm thuộc cùng 1 cluster sẽ tăng lên 1 if(SoDiemToaDo1 > 0): trongtam[j].set_x(TongToaDoX1 / SoDiemToaDo1) #Tính trung bình cộng tọa độ x của trọng tâm trongtam[j].set_y(TongToaDoY1 / SoDiemToaDo1) #Tính trung bình cộng tọa độ y của trọng tâm elif(j == 1): TongToaDoX2 += data[k].get_x() TongToaDoY2 += data[k].get_y() SoDiemToaDo2 += 1 if(SoDiemToaDo2 > 0): trongtam[j].set_x(TongToaDoX2 / SoDiemToaDo2) trongtam[j].set_y(TongToaDoY2 / SoDiemToaDo2) print("Trọng tâm 1: (", trongtam[0].get_x(), ", ", trongtam[0].get_y(), ")") print("Trọng tâm 2: (", trongtam[1].get_x(), ", ", trongtam[1].get_y(), ")") print() return </pre> <p><i>Hình 2.6 Tính khoảng cách tính trọng tâm</i></p>	Võ Minh Huy Tính khoảng cách và tính trọng tâm	Tính khoảng cách của các điểm so với trọng tâm và tính trọng tâm
6	<p>Màn hình code cập nhật lại cluster</p>	Võ Minh Huy Cập nhật lại tọa độ, trọng	Cập nhật lại tọa độ các điểm sẽ đi về nơi có

	<pre> In [7]: def CapNhatCluster(): print("Tọa độ sau khi cập nhật lại:") for i in range(len(ToaDoChoTruoc)): KhoangCachNganNhat = 1000 #Giả sử cho trước khoảng cách ngắn nhất từ các điểm tọa độ đến trọng tâm là 1000 Cluster = 0 #Khởi tạo Cluster ban đầu là 0 for j in range(2): KhoangCach = TinhKhoangCanh(trongtam[j].get_x(), data[i].get_x(), trongtam[j].get_y(), data[i].get_y()) if(KhoangCach < KhoangCachNganNhat): #So sánh khoảng cách từ các tọa độ đến trọng tâm KhoangCachNganNhat = KhoangCach #Nếu nhỏ hơn sẽ cập nhật lại khoảng cách Cluster = j #Cluster tại điểm sẽ bằng Cluster trọng tâm có khoảng cách gần nhất data[i].set_cluster(Cluster) if(data[i].get_cluster() is None or data[i].get_cluster() != Cluster): #Cập nhật lại Cluster data[i].set_cluster(Cluster) print("Tọa độ(", data[i].get_x(), ", ", data[i].get_y(), ") có Cluster:", data[i].get_cluster()) In [8]: def Kmeans(): TinhTrongTam() CapNhatCluster() print() return In [9]: def InKetQua(): for i in range(2): print("Cluster ", i, " :") for j in range(len(ToaDoChoTruoc)): if(data[j].get_cluster() == i): print("(" , data[j].get_x(), ", ", data[j].get_y(), ", ", data[j].get_cluster(),")") print() return </pre> <p style="text-align: center;"><i>Hình 2.7 Cập nhật cluster và kết quả</i></p>	tâm và in kết quả	trọng tâm gần nhất, xuất kết quả
7	<p>Màn hình xuất ra của thuật toán khi khởi tạo trọng tâm và tọa độ</p> <p>KhởiTaoTrongTam()</p> <pre> In [10]: KhởiTaoTrongTam() Nhập đỉnh trọng tâm thứ 1: 0 Nhập đỉnh trọng tâm thứ 2: 3 Các trọng tâm được chọn ngẫu nhiên là: (1.5 , 2.0) (5.0 , 7.0) </pre> <p>KhởiTaoToaDo()</p> <pre> In [11]: KhởiTaoToaDo() Tọa độ các điểm ban đầu : Tọa độ(1.5 , 2.0) có Cluster: 0 Tọa độ(2.5 , 3.0) có Cluster: None Tọa độ(3.0 , 4.0) có Cluster: None Tọa độ(5.0 , 7.0) có Cluster: 1 Tọa độ(3.5 , 5.0) có Cluster: None Tọa độ(4.5 , 5.5) có Cluster: None Tọa độ(3.5 , 4.5) có Cluster: None </pre> <p style="text-align: center;"><i>Hình 2.8 Xuất ra trọng tâm và tọa độ</i></p>	Võ Minh Huy Hiển thị kết quả hàm khởi tạo	Hiển thị kết quả tọa độ trọng tâm được chọn ngẫu nhiên khởi tạo tọa độ và cluster của các điểm ngẫu nhiên

8	<p>Màn hình kết quả chạy lần đầu tiên</p>  <pre> Trọng tâm 1: (1.5 , 2.0) Trọng tâm 2: (5.0 , 7.0) Tọa độ sau khi cập nhật lại: Tọa độ(1.5 , 2.0) có Cluster: 0 Tọa độ(2.5 , 3.0) có Cluster: 0 Tọa độ(3.0 , 4.0) có Cluster: 0 Tọa độ(5.0 , 7.0) có Cluster: 1 Tọa độ(3.5 , 5.0) có Cluster: 1 Tọa độ(4.5 , 5.5) có Cluster: 1 Tọa độ(3.5 , 4.5) có Cluster: 1 Cluster 0 : (1.5 , 2.0 , 0) (2.5 , 3.0 , 0) (3.0 , 4.0 , 0) Cluster 1 : (5.0 , 7.0 , 1) </pre> <p><i>Hình 2.9 Kết quả trọng tâm lần đầu</i></p>	Võ Minh Huy	Hiện thị kết quả tọa độ trọng tâm chạy lần 1 và chia các điểm về hai cluster
9	<p>Màn hình kết quả chạy lần 2</p>  <pre> Trọng tâm 1: (2.333333333333335 , 3.0) Trọng tâm 2: (4.125 , 5.5) Tọa độ sau khi cập nhật lại: Tọa độ(1.5 , 2.0) có Cluster: 0 Tọa độ(2.5 , 3.0) có Cluster: 0 Tọa độ(3.0 , 4.0) có Cluster: 0 Tọa độ(5.0 , 7.0) có Cluster: 1 Tọa độ(3.5 , 5.0) có Cluster: 1 Tọa độ(4.5 , 5.5) có Cluster: 1 Tọa độ(3.5 , 4.5) có Cluster: 1 Cluster 0 : (1.5 , 2.0 , 0) (2.5 , 3.0 , 0) (3.0 , 4.0 , 0) Cluster 1 : (5.0 , 7.0 , 1) </pre> <p><i>Hình 2.10 Kết quả trọng tâm chạy lại lần 2 sau khi cập nhật cluster</i></p>	Võ Minh Huy	Hiện thị kết quả tọa độ trọng tâm chạy lần 2 và phân lại các điểm về các cluster
10	<p>Màn hình kết quả chạy lần 3</p>	Võ Minh Huy	Hiện thị kết quả tọa độ trọng

	 <p>Trọng tâm 1: (2.3333333333333335 , 3.0) Trọng tâm 2: (4.125 , 5.5)</p> <p>Tọa độ sau khi cập nhật lại: Tọa độ(1.5 , 2.0) có Cluster: 0 Tọa độ(2.5 , 3.0) có Cluster: 0 Tọa độ(3.0 , 4.0) có Cluster: 0 Tọa độ(5.0 , 7.0) có Cluster: 1 Tọa độ(3.5 , 5.0) có Cluster: 1 Tọa độ(4.5 , 5.5) có Cluster: 1 Tọa độ(3.5 , 4.5) có Cluster: 1</p> <p>Cluster 0 : (1.5 , 2.0 , 0) (2.5 , 3.0 , 0) (3.0 , 4.0 , 0)</p> <p>Cluster 1 : (5.0 , 7.0 , 1) (3.5 , 5.0 , 1) (4.5 , 5.5 , 1) (3.5 , 4.5 , 1)</p>	Hiện thị kết quả chạy lần 3	tâm chạy lần 3 và phân lại các điểm về các cluster
11	<p>Màn hình code minh họa trên trục tọa độ</p> <pre>In [18]: import numpy as np import matplotlib.pyplot as plt from scipy.spatial.distance import cdist In [19]: #Khởi tạo 7 điểm tọa độ ban đầu cho trước và số Cluster cov = [[1, 0], [0, 1]] N = 300 #Với mỗi điểm tọa độ sẽ có 300 điểm lân cận X0 = np.random.multivariate_normal(ToaDoChoTruoc[0], cov, N) X1 = np.random.multivariate_normal(ToaDoChoTruoc[1], cov, N) X2 = np.random.multivariate_normal(ToaDoChoTruoc[2], cov, N) X3 = np.random.multivariate_normal(ToaDoChoTruoc[3], cov, N) X4 = np.random.multivariate_normal(ToaDoChoTruoc[4], cov, N) X5 = np.random.multivariate_normal(ToaDoChoTruoc[5], cov, N) X6 = np.random.multivariate_normal(ToaDoChoTruoc[6], cov, N) X = np.concatenate((X0, X1, X2, X3, X4, X5, X6), axis = 0) cluster = 2 original_label = np.asarray([0]*N + [1]*N + [2]*N + [3]*N + [4]*N + [5]*N + [6]*N).T</pre>	Võ Gia Huy Code đồ họa khởi tạo 7 điểm ngẫu nhiên với mỗi điểm sẽ có thêm 300 điểm lân cận	Dựa vào code đã nghiên cứu áp dụng tạo đồ họa trên trục tọa độ biểu diễn ra màn hình các điểm nằm lộn xộn đan xen nhau

```

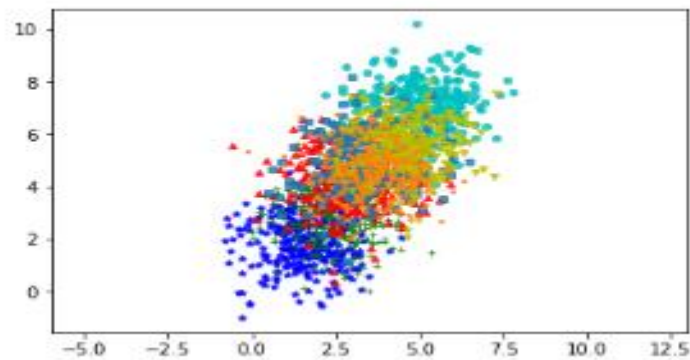
In [20]: #Hiển thị các điểm tọa độ lên đồ thị Kmeans
def DoThiKmeans(X, label):
    K = np.amax(label) + 1
    X0 = X[label == 0, :]
    X1 = X[label == 1, :]
    X2 = X[label == 2, :]
    X3 = X[label == 3, :]
    X4 = X[label == 4, :]
    X5 = X[label == 5, :]
    X6 = X[label == 6, :]

    #Xác định hình dạng cho các tọa độ
    plt.plot(X0[:, 0], X0[:, 1], 'b*', markersize = 4, alpha = .8)
    plt.plot(X1[:, 0], X1[:, 1], 'g+', markersize = 4, alpha = .8)
    plt.plot(X2[:, 0], X2[:, 1], 'r^', markersize = 4, alpha = .8)
    plt.plot(X3[:, 0], X3[:, 1], 'co', markersize = 4, alpha = .8)
    plt.plot(X4[:, 0], X4[:, 1], 's', markersize = 4, alpha = .8)
    plt.plot(X5[:, 0], X5[:, 1], 'yv', markersize = 4, alpha = .8)
    plt.plot(X6[:, 0], X6[:, 1], '.:', markersize = 4, alpha = .8)

    plt.axis('equal')
    plt.plot()
    plt.show()

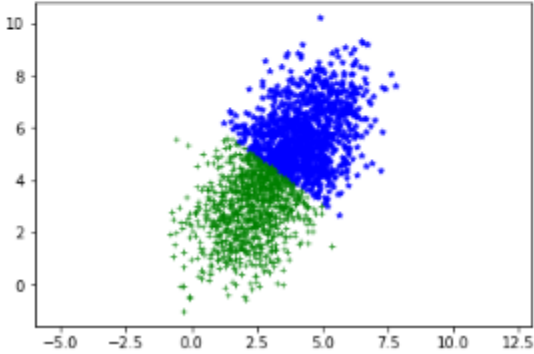
DoThiKmeans(X, original_label)

```



Hình 2.12 Khởi tạo điểm để hiển thị lên trục tọa độ

12	<p>Màn hình code thuật toán k-mean trên trục tọa độ</p> <pre> In [21]: def KhoiTaoTrongTam(X, cluster): return X[np.random.choice(X.shape[0], cluster, replace=False)] def TinhKhoangCach(X, TrongTam): D = cdist(X, TrongTam) #tính khoảng cách giữa các tọa độ với trọng tâm return np.argmin(D, axis = 1) #Trở về khoảng cách ngắn nhất def CapNhatTrongTam(X, labels, cluster): TrongTam = np.zeros((cluster, X.shape[1])) for k in range(cluster): Xk = X[labels == k, :] TrongTam[k, :] = np.mean(Xk, axis = 0) return TrongTam def XetDieuKienTrongTam(TrongTam, TrongTamMoi): return (set([tuple(a) for a in TrongTam]) == set([tuple(a) for a in TrongTamMoi])) #So sánh In [22]: def kmeans(X, cluster): TrongTam = [KhoiTaoTrongTam(X, cluster)] labels = [] it = 0 while True: labels.append(TinhKhoangCach(X, TrongTam[-1])) #Tính khoảng cách giữa các điểm và trọng tâm TrongTamMoi = CapNhatTrongTam(X, labels[-1], cluster) #Cập nhật lại trọng tâm mới if XetDieuKienTrongTam(TrongTam[-1], TrongTamMoi): #Thuật toán sẽ dừng khi trọng tâm không thay đổi break TrongTam.append(TrongTamMoi) it += 1 return (TrongTam, labels, it) </pre> <p><i>Hình 2.13 Code thuật toán k-means</i></p>	Võ Gia Huy	<p>Áp dụng code lý thuyết bên trên (cập nhật đến khi trọng tâm không đổi) để làm đồ họa trên trục đồ thị</p>
13	<p>Màn hình trục tọa độ sau khi chia thành 2 cụm</p> <pre> In [23]: #Đồ thị Kmeans sau khi cập nhật với số Cluster = 2 (TrongTam, labels, it) = kmeans(X, cluster) print(TrongTam[-1]) DoThiKMeans(X, labels[-1]) [[2.24437332 2.9461836] [4.29873487 5.73555845]] </pre> <p><i>Hình 2.14 Các điểm sau khi phân cụm</i></p>	Võ Gia Huy	<p>Các điểm được phân về hai cụm sau khi phân cụm trên đồ thị</p>

14	<p>Màn hình phân cụm dùng hàm có sẵn</p> <pre> In [24]: #Đồ thị Kmeans sử dụng hàm có sẵn from sklearn.cluster import KMeans kmeans = KMeans(n_clusters = 2, random_state=0).fit(X) print(kmeans.cluster_centers_) pred_label = kmeans.predict(X) DoThiKmeans(X, pred_label) </pre> <pre> [[4.29768635 5.73442905] [2.24347736 2.94462232]] </pre>  <p>Hình 2.15 Các điểm phân cụm bằng thư viện có sẵn</p>	<p>Võ Gia Huy</p> <p>Kết quả đồ thị</p>	<p>Các điểm được phân về hai cụm sau khi phân cụm trên đồ thị dựa trên thư viện có sẵn</p>
----	---	---	--

Bảng 2.3 Code và k-means trên toạ độ

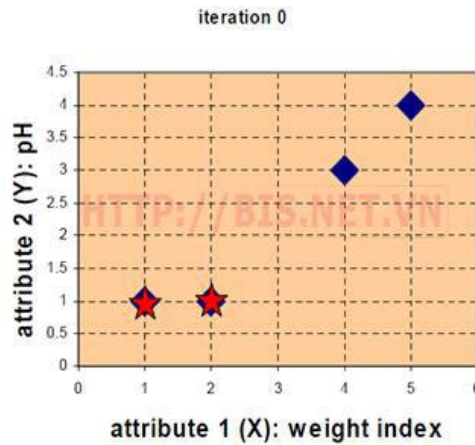
2.4. Ứng dụng của thuật toán k-means

Giả sử ta có 4 loại thuốc A, B, C, D, mỗi loại thuốc được biểu diễn bởi 2 đặc trưng X và Y như sau. Mục đích của ta là nhóm các thuốc đã cho vào 2 nhóm (K=2) dựa vào các đặc trưng của chúng.

Object	Feature (X): Weight index	Feature 2 (Y): pH
Medicine A	1	1
Medicine B	2	1
Medicine C	4	3
Medicine D	5	4

Bảng 2.4 Thuốc và thuộc tính của thuốc

Bước 1. Giả sử ta chọn A là tâm của nhóm thứ nhất (tọa độ tâm nhóm thứ nhất $c_1(1,1)$) và B là tâm của nhóm thứ 2 (tọa độ tâm nhóm thứ hai $c_2(2,1)$).



Hình 2.16 Vị trí các viên thuốc trên toạ độ

Bước 2. Tính khoảng cách từ các đối tượng đến tâm của các nhóm (Khoảng cách Euclidean)

$$D^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \quad \begin{array}{l} c_1 = (1,1) \text{ group } - 1 \\ c_2 = (2,1) \text{ group } - 2 \end{array}$$

A B C D

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{array}{l} X \\ Y \end{array}$$

Mỗi cột trong ma trận khoảng cách (D) là một đối tượng (cột thứ nhất tương ứng với đối tượng A, cột thứ 2 tương ứng với đối tượng B,). Hàng thứ nhất trong ma trận khoảng cách biểu diễn khoảng cách giữa các đối tượng đến tâm của nhóm thứ nhất (c_1) và hàng thứ 2 trong ma trận khoảng cách biểu diễn khoảng cách của các đối tượng đến tâm của nhóm thứ 2 (c_2).

Ví dụ, khoảng cách từ loại thuốc C = (4,3) đến tâm $c_1(1,1)$ là 3.61 và đến tâm $c_2(2,1)$ là 2.83 được tính như sau:

$$c_1 = (1,1) \quad \sqrt{(4-1)^2 + (3-1)^2} = 3.61$$

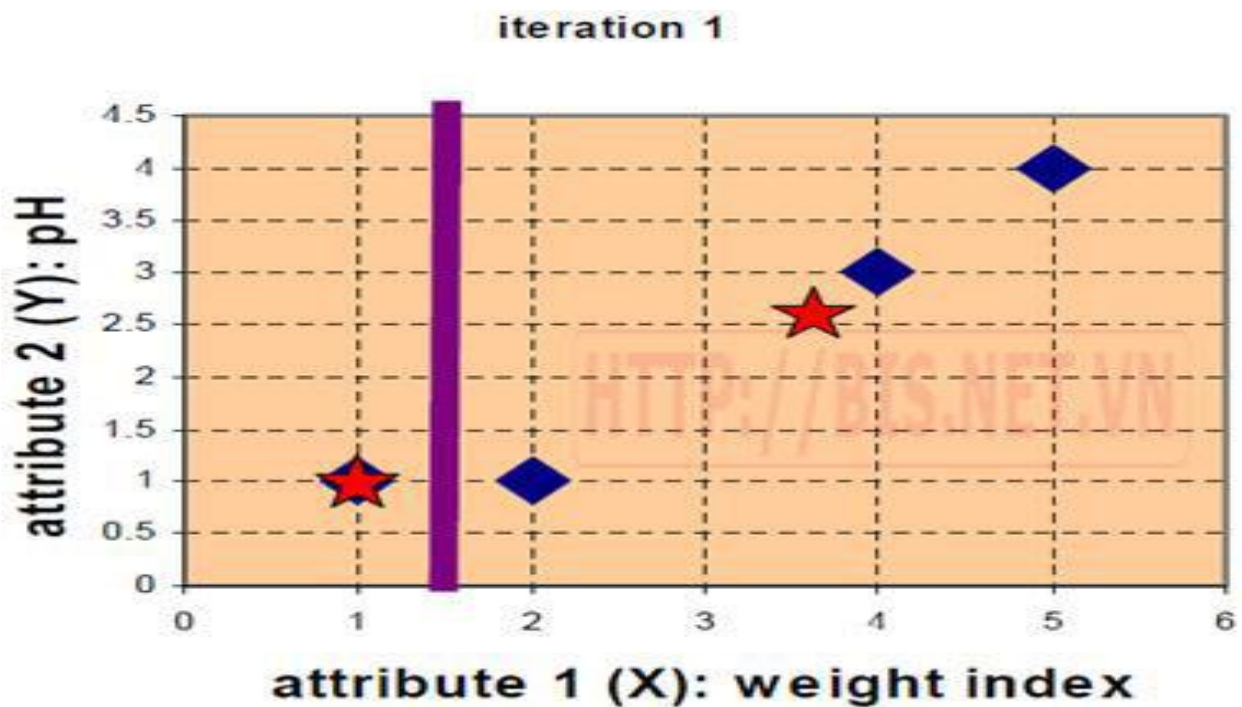
$$c_2 = (2,1) \quad \sqrt{(4-2)^2 + (3-1)^2} = 2.83$$

Bước 3. Nhóm các đối tượng vào nhóm gần nhất

$$G^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{matrix} \text{group} - 1 \\ \text{group} - 2 \end{matrix}$$

A B C D

Ta thấy rằng nhóm 1 sau vòng lặp thứ nhất gồm có 1 đối tượng A và nhóm 2 gồm các đối tượng còn lại B,C,D.



Hình 2.17 Vị trí cập nhật lần một các viên thuốc trên tọa độ

Bước 4. Tính lại tọa độ các tâm cho các nhóm mới dựa vào tọa độ của các đối tượng trong nhóm. Nhóm 1 chỉ có 1 đối tượng A nên tâm nhóm 1 vẫn không đổi, $c_1(1,1)$. Tâm nhóm 2 được tính như sau:

$$c_2 = \left(\frac{2 + 4 + 5}{3}, \frac{1 + 3 + 4}{3} \right) = \left(\frac{11}{3}, \frac{8}{3} \right)$$

Bước 5. Tính lại khoảng cách từ các đối tượng đến tâm mới

$$D^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{array}{l} c_1 = (1,1) \text{ group } - 1 \\ c_2 = \left(\frac{11}{3}, \frac{8}{3}\right) \text{ group } - 2 \end{array}$$

A B C D

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{array}{l} X \\ Y \end{array}$$

Bước 6. Nhóm các đối tượng vào nhóm

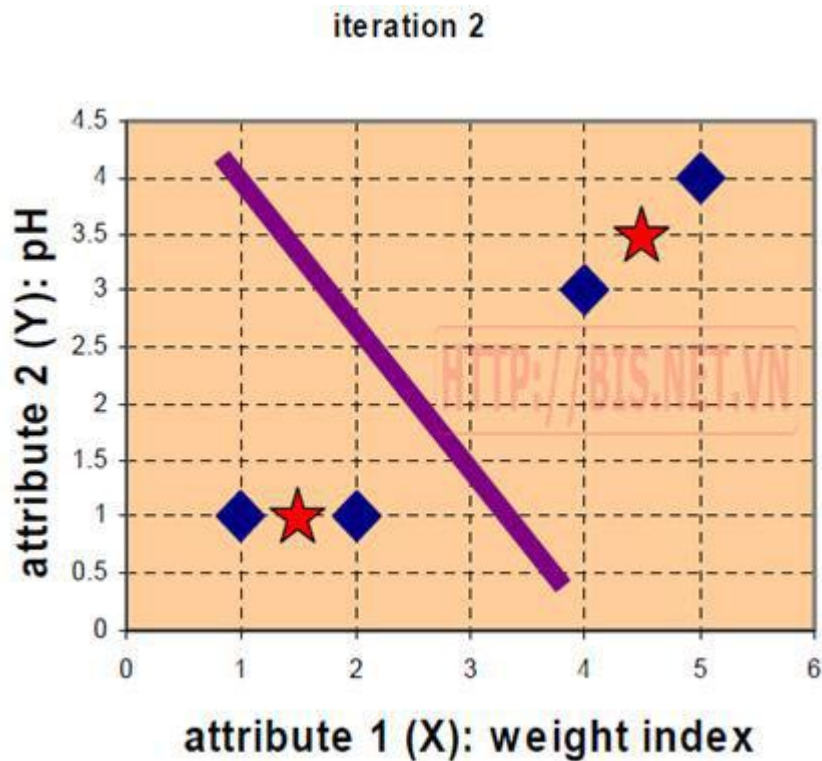
$$G^1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{array}{l} \text{group } - 1 \\ \text{group } - 2 \end{array}$$

A B C D

Bước 7. Tính lại tâm cho nhóm mới

$$c_1 = \left(\frac{1+2}{2}, \frac{1+1}{2} \right) = (1.5, 1)$$

$$c_2 = \left(\frac{4+5}{2}, \frac{3+4}{2} \right) = (4.5, 3.5)$$



Hình 2.18 Vị trí cập nhật lần hai các viên thuốc trên toạ độ

Bước 8. Tính lại khoảng cách từ các đối tượng đến tâm mới

$$D^2 = \begin{bmatrix} 0.5 & 0.5 & 3.2 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{matrix} c_1 = (1_2^1, 1) \text{ group} - 1 \\ c_2 = (4_2^1, 3_2^1) \text{ group} - 2 \end{matrix}$$

A B C D

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

Bước 9. Nhóm các đối tượng vào nhóm

$$G^2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} \text{group} - 1 \\ \text{group} - 2 \end{matrix}$$

A B C D

Ta thấy $G^2 = G^1$ (Không có sự thay đổi nhóm nào của các đối tượng) nên thuật toán dừng và kết quả phân nhóm như sau:

Object	Feature (X): Weight index	Feature 2 (Y): pH	Group (result)
Medicine A	1	1	1
Medicine B	2	1	1
Medicine C	4	3	2
Medicine D	5	4	2

Bảng 2.5 Kết quả phân nhóm

Thuật toán K-Means có ưu điểm là đơn giản, dễ hiểu và cài đặt. Tuy nhiên, một số hạn chế của K-Means là hiệu quả của thuật toán phụ thuộc vào việc chọn số nhóm K (phải xác định trước) và chi phí cho thực hiện vòng lặp tính toán khoảng cách lớn khi số cụm K và dữ liệu phân cụm lớn.

3. Mô tả phân công công việc

Tên SV	Đánh giá chung phần trăm đóng góp	Mô tả khái quát mảng công việc SV thực hiện trong đề án.
Võ Minh Huy	60 %	Xây dựng ý tưởng, viết code, góp ý báo cáo.
Võ Gia Huy	40 %	Xây dựng ý tưởng giao diện hiển thị, viết báo cáo.

Bảng 3.1 Mô tả phân công công việc

4. Kết luận

Đề án đã được hoàn thành được 90% mục tiêu đề ra.

Khó khăn: Do đề tài còn khá mới mẻ nên việc tìm hiểu gặp nhiều khó khăn

Ưu điểm: Hiểu được cách thức hoạt động một cách rõ ràng, chạy được code minh họa đơn giản của k-means

Khuyết điểm: Chưa làm được các dạng bài ứng dụng vào thực tế vì khá phức tạp

5. Tài liệu tham khảo

Tham khảo từ tài liệu thầy đưa

1. https://en.wikipedia.org/wiki/K-means_clustering

Tham khảo từ internet

1. https://machinelearningcoban.com/2017/01/01/kmeans/?fbclid=IwAR05JuwHWCiQp9CAya61ErTNmA_QF4oyqUsmpIOPUW_MJWOn6ETMhQSmqVc

2. <https://kipalog.com/posts/Thuat-toan-Kmean-va-ung-dung>
3. <http://bis.net.vn/forums/t/374.aspx>