

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN 2

-----[?][?][?][?][?]-



BÁO CÁO ĐỒ ÁN CUỐI KỲ

MÔN: PHÁT TRIỂN HỆ THỐNG THÔNG MINH

XÂY DỰNG WEB BÁN THUỐC TÂY

GIẢNG VIÊN HƯỚNG DẪN: NGUYỄN NGỌC DUY

NHÓM 21

N20DCCN040 – Phùng Đức Mạnh

N20DCCN079 - ĐẶNG ĐỨC TRỌNG

Thành phố Hồ Chí Minh, 4/11/2023

Mục lục

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI.....	2
1.1. Lý do chọn đề tài.....	2
1.2. Mục tiêu nghiên cứu	3
1.3. Đối tượng và phạm vi nghiên cứu.....	3
1.4. Phương pháp nghiên cứu	3
1.5. Một số hạn chế dự kiến.....	4
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT.....	4
2.1. Giới thiệu về học máy.....	4
2.2. Giới thiệu về bài toán phân cụm.....	4
2.3. Thuật toán Hierarchical Divisive Clustering	5
2.3.1. Mã giả:	5
2.3.2. Công thức toán học trong Hierarchical Divisive Clustering:	6
2.4. Hình ảnh minh họa sau khi phân cụm	6
2.5. Dữ liệu	7
2.5.1. Nghiệp vụ.....	7
2.5.2. Xác định thuộc tính và mô tả dữ liệu.....	8
CHƯƠNG 3: XÂY DỰNG CẤU HÌNH.....	9
3.1. Đọc dữ liệu từ bảng Thuốc trong database và xử lý dữ liệu.....	9
3.2. Áp dụng thuật toán phân cụm cho dữ liệu.....	11
3.3. Chèn dữ liệu là kết quả của thuật toán vào lại bảng Thuốc liên quan trong database.....	16
CHƯƠNG 4: THIẾT KẾ HỆ THỐNG	19
4.1. Thiết kế cơ sở dữ liệu.....	19
4.2. Thiết kế giao diện.....	21

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

1.1. Lý do chọn đề tài

- Trong thời đại ngày nay, sử dụng tri thức đã trở thành động lực chủ chốt cho tăng trưởng kinh tế quốc gia, cho tăng cường năng lực cạnh tranh của doanh nghiệp. Đồng thời, dung lượng dữ liệu số tăng rất nhanh chóng, chúng đã trở thành nguồn tài nguyên ẩn thông tin và tri thức có tiềm năng hữu ích cho phát triển kinh tế và tăng cường năng lực cạnh tranh. Nghiên cứu và triển khai các phương pháp tự động phát hiện các mẫu

mới, có giá trị, hữu ích tiềm năng và hiểu được trong khối dữ liệu đồ sộ, khắc phục hiện tượng giàu về dữ liệu mà nghèo về thông tin, hướng tới mục tiêu tăng cường tài nguyên tri thức là hết sức cần thiết và có ý nghĩa. Theo đó, việc định nghĩa khai phá dữ liệu được hiểu theo nghĩa đơn giản chính là việc rút trích thông tin hay tri thức mới và có ích từ nguồn dữ liệu khổng lồ.

- Trong lĩnh vực thương mại, tính cạnh tranh kinh doanh là rất cao, cho nên việc phân tích dữ liệu để đưa đến người dùng những dịch vụ tốt hơn, có nhiều tiện ích cho khách hàng hơn là điều cần thiết. Cụ thể trong chuyện mua bán điện thoại, khi đưa vào thông tin của một chiếc máy nào đó, người ta đồng thời phải đưa ra quyết định để biết loại máy đó thuộc nhóm nhu cầu nào. Tuy nhiên, với mỗi thông tin được đưa vào, người đánh giá, xếp loại chúng vào những nhóm nhu cầu phải có kiến thức, phải là chuyên gia am hiểu về các dòng máy, am hiểu về các thông tin chi tiết của máy. Như vậy nếu ta đưa vào rất nhiều thông tin, người này sẽ phải tốn rất nhiều thời gian để xem xét và đưa ra quyết định, như vậy sẽ dẫn đến nguy cơ mất cơ hội trong kinh doanh. Do đó, việc khai phá dữ liệu nhằm giúp đỡ con người trong công việc luôn được thúc đẩy mạnh mẽ, làm sao với các dữ liệu đã thu thập, những hành động tương ứng từ dữ liệu đó có thể mang lại lợi ích kịp thời, tối đa cho doanh nghiệp.

- Với những lý do trên cùng với sự cho phép của thầy Nguyễn Ngọc Duy, nhóm chúng em quyết định chọn đề tài **“Xây dựng web bán thuốc tây”** với chức năng thông minh: **gợi ý thuốc liên quan với thuốc mà người dùng đang xem**

1.2. Mục tiêu nghiên cứu

- Nghiên cứu mô hình thuật toán phân cụm: **Hierarchical Divisive Clustering**
- Ứng dụng của thuật toán phân cụm Hierarchical Divisive Clustering vào việc gợi ý những thuốc liên quan (chung cụm) với thuốc đang xem

1.3. Đối tượng và phạm vi nghiên cứu

- Đối tượng nghiên cứu: dữ liệu về điện thoại
- Phạm vi nghiên cứu: Thị trường điện thoại di động trên toàn thế giới

1.4. Phương pháp nghiên cứu

- Phương pháp thực nghiệm khoa học: Dựa vào tập dữ liệu đã nhận được, tiến hành

tiền xử lý và sử dụng các thuật toán phân cụm Hierarchical Divisive Clustering, đánh giá tính hiệu quả của thuật toán dựa trên tập dữ liệu kiểm thử

1.5. Một số hạn chế dự kiến

- Số lượng mẫu dữ liệu càng nhiều thì thời gian tính toán của thuật toán càng lâu

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Giới thiệu về học máy

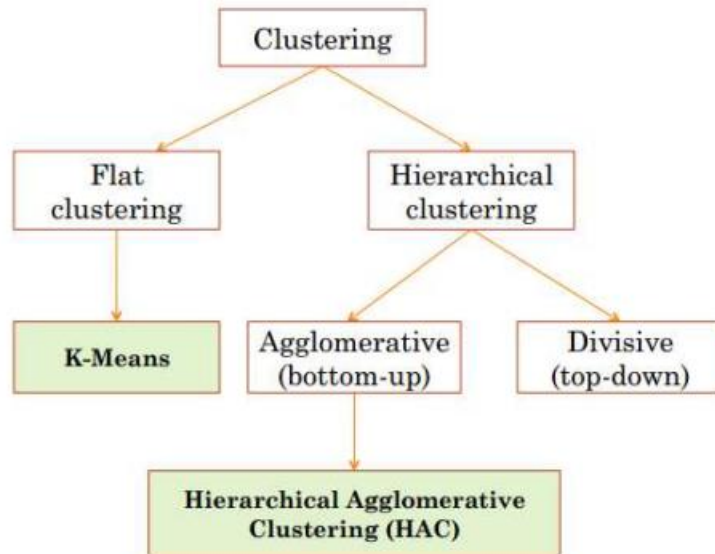
- **Học máy** (machine learning) là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kĩ thuật cho phép các hệ thống "học" tự động từ dữ liệu để giải quyết những vấn đề cụ thể.

- **Học máy** được chia thành 2 loại chính bao gồm: học có giám sát và học không giám sát.

- Học có giám sát là phương pháp sử dụng những dữ liệu được gán nhãn sẵn để suy luận ra quan hệ giữa đầu vào và đầu ra. Sau khi tìm hiểu cách tốt nhất để mô hình hóa các mối quan hệ cho dữ liệu được gán nhãn, thuật toán huấn luyện sẽ được sử dụng cho các bộ dữ liệu mới. Học tập có giám sát có thể được nhóm lại thành các vấn đề về phân loại và hồi quy.
- Học không giám sát sử dụng những dữ liệu chưa được gán nhãn sẵn để suy luận và tìm cách để mô tả dữ liệu cùng cấu trúc của chúng. Ứng dụng của học không giám sát đó là hỗ trợ phân loại thành các nhóm có đặc điểm tương đồng

2.2. Giới thiệu về bài toán phân cụm

- Khái niệm: **Phân cụm (Clustering)** thuộc loại học không giám sát (Unsupervised learning) là một dữ liệu là bài toán gom nhóm các đối tượng dữ liệu vào thành từng cụm (cluster) sao cho các đối tượng trong cùng một cụm có sự tương đồng theo một tiêu chí nào đó
 - Bài toán theo đề tài: phân nhóm các thuốc liên quan với nhau dựa theo 4 đặc trưng: tên thuốc, công dụng, số lượng dược chất, tổng trọng số dược chất
- Phân cụm chia thành các loại:



Hình 2.2.1. Các loại phân cụm

- Ở đề tài này, ta sẽ tìm hiểu về Phân cụm theo cấp bậc
- Phân cụm theo cấp bậc là mô hình phân cụm dựa trên kết nối, nhóm các điểm dữ liệu gần nhau dựa trên thước đo độ tương tự hoặc khoảng cách.
 - Giả định là các điểm dữ liệu gần nhau thì giống nhau hoặc có liên quan hơn so với các điểm dữ liệu ở xa nhau hơn.
 - **Agglomerative Clustering** (phân cụm kết tụ): Nó còn được gọi là cách tiếp cận từ dưới lên hoặc phân cụm kết tụ phân cấp (HAC). Một cấu trúc có nhiều thông tin hơn tập hợp các cụm không có cấu trúc được trả về bởi phân cụm phẳng. Thuật toán phân cụm này không yêu cầu chúng ta xác định trước số lượng cụm. Các thuật toán từ dưới lên xử lý mỗi dữ liệu dưới dạng một cụm đơn ngay từ đầu và sau đó liên tục kết hợp các cặp cụm cho đến khi tất cả các cụm được hợp nhất thành một cụm duy nhất chứa tất cả dữ liệu.
 - **Divisive Clustering** (Phân cụm phân chia) ngược lại với phân cụm HAC với cách tiếp cận từ trên xuống. Nó bắt đầu với tất cả các điểm dữ liệu trong một cụm duy nhất, sau đó phân chia đệ quy các cụm thành các cụm con nhỏ hơn dựa trên sự khác biệt của chúng
- Lý do chọn thuật toán Divisive Clustering:
 - Phân cụm tổng hợp đưa ra quyết định bằng cách xem xét các mô hình cục bộ hoặc các điểm lân cận mà không tính đến việc phân phối dữ liệu toàn cầu ngay từ đầu. Những quyết định ban đầu này không thể được hoàn tác. Trong khi đó, phân cụm phân chia sẽ xem xét phân phối dữ liệu toàn cầu khi đưa ra quyết định phân vùng cấp cao nhất.

2.3. Thuật toán Hierarchical Divisive Clustering

2.3.1. Mã giả:

Mã giả thuật toán hierarchical divisive cluster áp dụng cho đề tài

B1: tiền xử lý dữ liệu: chuyển dữ liệu text sang số, tính khoảng cách giữa các điểm dữ liệu với nhau và lưu vào matrix

B2: chọn một cụm có đường kính lớn nhất giữa 2 điểm trong cụm đó ra để phân cụm (nếu phân lần đầu thì ta lấy cụm gốc)

B3: lấy mỗi điểm trong cụm cần phân và tính toán khoảng cách trung bình tới các điểm còn lại trong cùng cụm

B4: lấy mỗi điểm trong cụm cần phân và tính toán khoảng cách trung bình tới các điểm còn lại khác cụm

B5: lấy hiệu 2 khoảng cách trung bình tính ở B3 và B4, nếu hiệu nào lớn nhất thì ta lấy điểm đó ra khỏi cụm và thêm tạm thời vào cụm mới

B6: quay lại B3, đến khi nào hiệu ≤ 0 thì quay lại bước 2, khi nào đường kính lớn nhất ≤ 0 thì chuyển sang bước 7

B7: trả về danh sách cụm được phân theo sơ đồ cây. Kết thúc thuật toán

2.3.2. Công thức toán học trong Hierarchical Divisive Clustering:

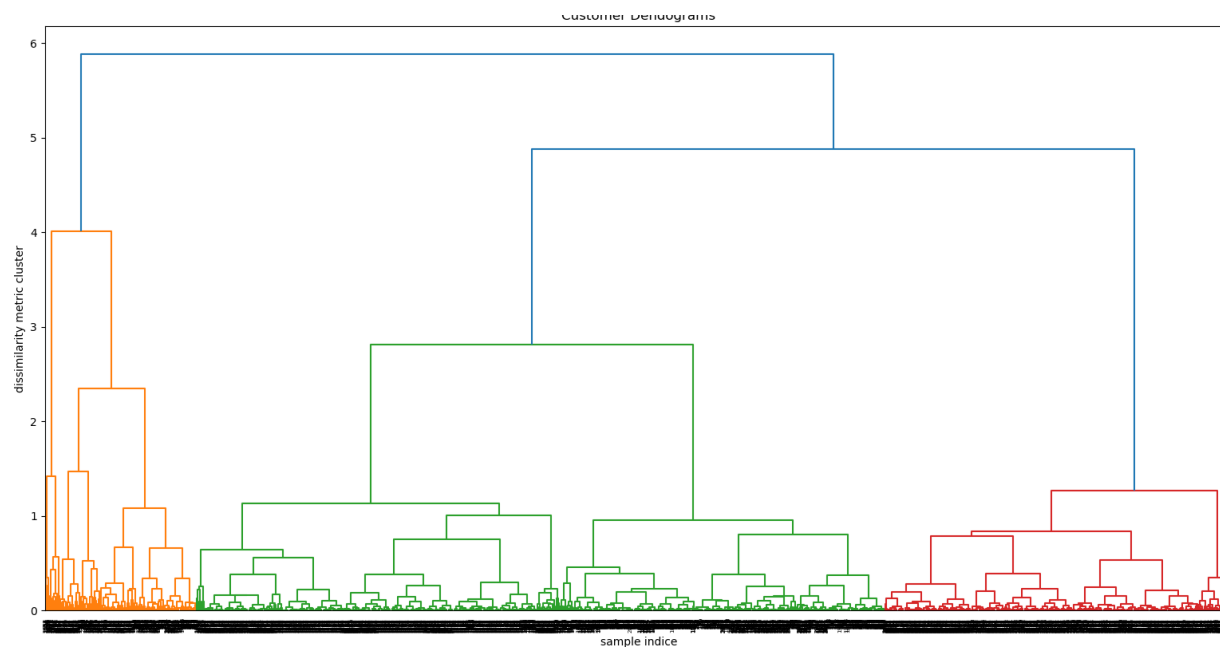
- Ở bước tiền xử lý dữ liệu, sử dụng công thức khoảng cách **Euclidean** để tính khoảng cách giữa các mẫu thuốc với nhau. Sau khi tính toán xong sẽ lưu lại dưới dạng ma trận kích thước $N \times N$ (với N : số lượng mẫu thuốc).

$$d(r_i, r_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \dots + (x_{in} - x_{jn})^2}$$

Hình 2.3.2.1 Công thức khoảng cách **Euclidean**

- Trong thuật toán sẽ có những công thức:
 - o Tính khoảng cách trung bình từ một mẫu thuốc với các thuốc khác nằm chung cụm.
 - o Tính khoảng cách trung bình từ một mẫu thuốc với các thuốc khác nằm khác cụm.

2.4. Hình ảnh minh họa sau khi phân cụm

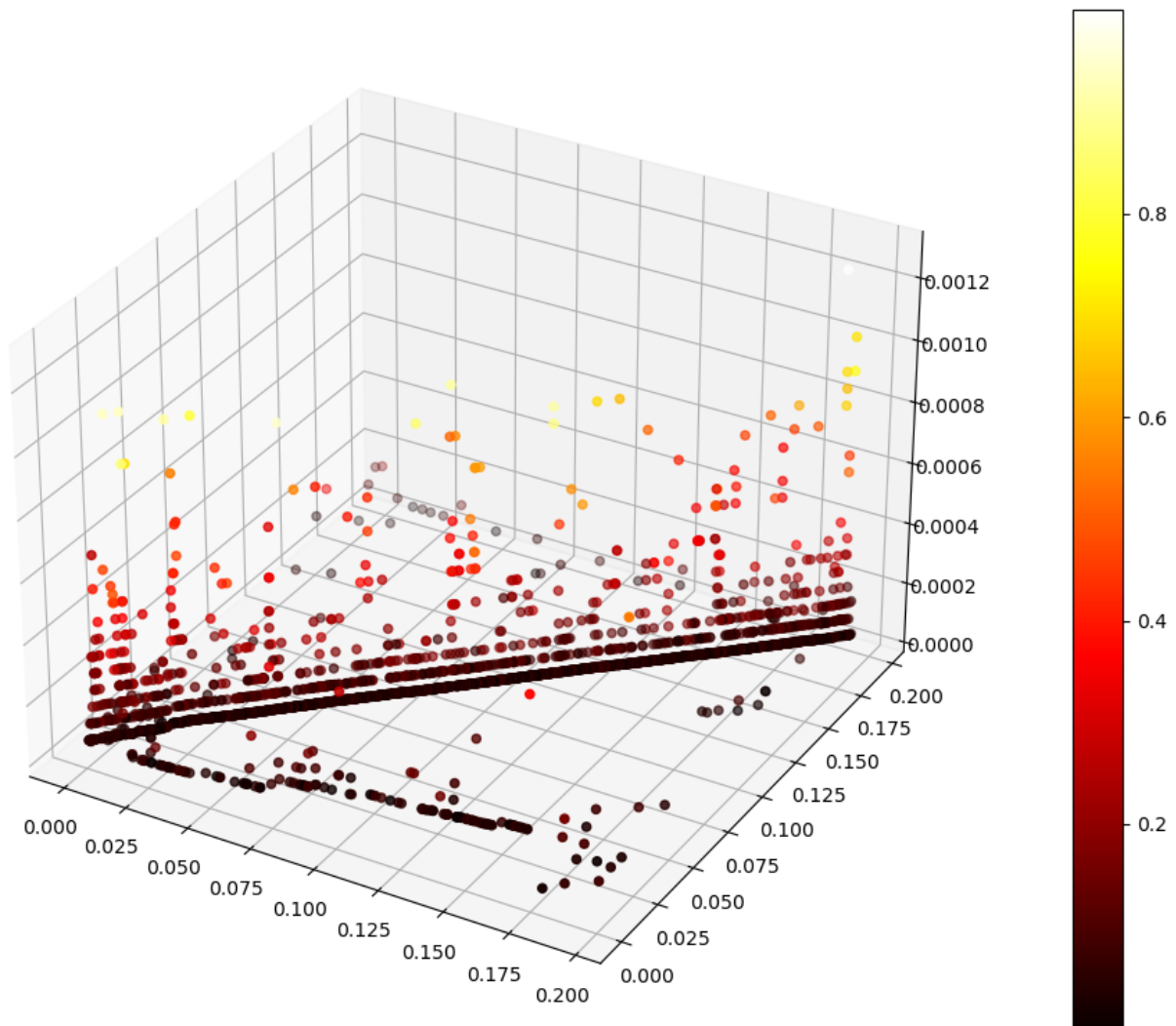


Hình 2.4.1. Hình ảnh các cụm được phân

2.5. Dữ liệu

2.5.1. Nghiệp vụ

Hình ảnh mô phỏng dữ liệu không gian 4 chiều do mỗi thuốc có 4 thuộc tính:



Hình 2.5.1.1 Hình ảnh mô phỏng các điểm dữ liệu thuốc

2.5.2. Xác định thuộc tính và mô tả dữ liệu

Dữ liệu lấy từ database có dạng sau:

id	ten_thuoc	cong_dung	soLuongDuocChat	tongTrongSoDuocChat
1	Thuốc Actadol 500mg Medipharco giảm nhanh các chứng sốt, đau nhức (10 vỉ x 10 viên)	Công dụng của Thuốc Actadol 500mgChỉ định Paraceta...	1	1436
2	Thuốc Acetab Extra Agimexpharm giúp giảm đau, hạ sốt (10 vỉ x 10 viên)	Công dụng của Thuốc Acetab ExtraChỉ định Acetab Extra ...	2	1869
3	Thuốc Allerphast 180mg Mepiphar điều trị viêm mũi dị ứng theo mùa	Công dụng của Thuốc Allerphast 180mgChỉ định Allerph...	1	684
4	Thuốc Acefalgan 500mg Euvipharm hỗ trợ hạ sốt và giảm đau (10 vỉ x 10 viên)	Công dụng của Thuốc Acefalgan 500mgChỉ định Thuốc A...	1	1436
5	Thuốc A.T Zinc 10mg An Thiên bù nước và điện giải trong phác đồ điều trị tiêu chảy kéo dài (100 viên)	Công dụng của Thuốc A.T Zinc 10mgChỉ định Thuốc A.T Z...	1	805
6	Thuốc Auclanityl 500/62.5mg Tipharco điều trị nhiễm khuẩn (12 gói)	Công dụng của Thuốc Auclanityl 500/62.5mgChỉ định Thu...	2	990
7	Thuốc Acetylcystein 200mg Khapharco tiêu nhầy trong bệnh nhầy nhớt (190 viên)	Công dụng của Thuốc AcetylcysteinChỉ định Thuốc Acetyl...	5	2696
8	Thuốc Auclatyl 500mg/125mg Tipharco điều trị viêm đường hô hấp (2 vỉ x 10 viên)	Công dụng của Thuốc Auclatyl 500mg/125mgChỉ định Th...	2	1415
9	Thuốc Amlacor - 5 Torrent hỗ trợ chống đau thắt ngực (3 vỉ x 10 viên)	Công dụng của Thuốc Amlacor - 5 TorrentTăng huyết áp ...	1	538
10	Thuốc Auclanityl Tipharco điều trị các bệnh nhiễm khuẩn (2 vỉ x 7 viên)	Công dụng của Thuốc AuclanitylChỉ địnhĐiều trị các nhiễ...	2	1415
11	Thuốc Augtipha 1g Tipharco điều trị bệnh lý nhiễm khuẩn (2 vỉ x 10 viên)	Công dụng của Thuốc Augtipha 1gChỉ định Thuốc Augtip...	2	1415
12	Thuốc Augtipha 625mg Tipharco điều trị nhiễm khuẩn (2 vỉ x 10 viên)	Công dụng của Thuốc Augtipha 625mg TipharcoChỉ định...	2	1415
13	Thuốc Auclanityl 500/125mg Tipharco hỗ trợ điều trị viêm phế quản, viêm phổi (20 viên)	Công dụng của Thuốc Auclanityl 500/125mgChỉ địnhThu...	2	1415
14	Thuốc Azibiotic 500 Medipharco điều trị nhiễm khuẩn (2 vỉ x 3 viên)	Công dụng của Thuốc Azibiotic 500Chỉ địnhThuốc Azibiot...	1	253
15	Thuốc Aciclovir 800mg Meyer điều trị nhiễm Herpes Zoster cấp tính (3 vỉ x 10 viên)	Công dụng của Thuốc Aciclovir 800mgChỉ định Thuốc Aci...	1	1086
16	Thuốc Auclanityl 250/31.25mg Tipharco điều trị nhiễm khuẩn nặng đường hô hấp (12 gói)	Công dụng của Thuốc Auclanityl 250/31.25mgChỉ định T...	2	1415
17	Thuốc Agimdogyl Agimexpharm điều trị viêm ruột, viêm miệng, viêm nha chu (2 vỉ x 10 viên)	Công dụng của Thuốc AgimdogylChỉ địnhThuốc Agimdog...	2	858
18	Siro Ambroxol 15mg/5ml Danapha điều trị các rối loạn về sự bài tiết ở phế quản (60ml)	Công dụng của Siro Ambroxol 15mg/5mlChỉ địnhThuốc ...	1	270
19	Thuốc Alpha-Chymotrypsin 4200lu Euvipharm điều trị phù nề (2 vỉ x 10 viên)	Công dụng của Alpha-Chymotrypsin 4200luChỉ định Thuố...	1	230
20	Thuốc Ammg 3B Trường Thọ hỗ trợ điều trị bệnh lý thần kinh (50 viên)	Công dụng của Thuốc Ammg 3BChỉ định Thuốc AMMG-3...	3	3260
21	Thuốc Amlodipin 5mg Trường Thọ điều trị tăng huyết áp, đau thắt ngực ổn định (30 viên)	Công dụng của Thuốc Amlodipin 5mgChỉ định Thuốc Am...	1	538
24	Thuốc An Thảo Nam Dược hỗ trợ điều trị hồi miêng, viêm chân răng (5 vỉ x 10 viên)	Công dụng của Thuốc An ThảoChỉ định Thuốc Thảo Dươ...	6	5000

Hình 2.5.2.1. Dữ liệu thuốc lấy từ database

- soLuongDuocChat: một thuốc sẽ chứa nhiều dược chất, thì thuộc tính này sẽ lưu tính toán tất cả dược chất có trong thuốc đó.
- tongTrongSoDuocChat: mỗi dược chất sẽ có một id riêng biệt, với thuộc tính này sẽ lưu kết quả tính tổng các id của các dược chất có trong thuốc đó. Vẫn có hạn chế chưa khắc phục được: có trường hợp số lượng dược chất giống nhau, trọng số bằng nhau nhưng dược chất trong thuốc đó khác nhau.

Với lệnh truy vấn sql:

```
select t.id, ten_thuoc, cong_dung, count(ctdc.id_duoc_chat) as soLuongDuocChat, sum(ctdc.id_duoc_chat) as tongTrongSoDuocChat from
(select id_duoc_chat, id_thuoc from ct_duoc_chat) ctdc,
(select id, ten_thuoc, cong_dung from thuoc) t
where ctdc.id_thuoc = t.id
group by t.id, ten_thuoc, cong_dung
```

Hình 2.5.2.2. Lệnh truy vấn để lấy dữ liệu từ database

CHƯƠNG 3: XÂY DỰNG CẤU HÌNH

3.1. Đọc dữ liệu từ bảng Thuốc trong database và xử lý dữ liệu

- Lấy dữ liệu từ database

```
2 usages 2 Duc trong
def read_data_from_sql():
    sql_query = '''select t.id, ten_thuoc, cong_dung, count(ctdc.id_duoc_chat) as soLuongDuocChat, sum(ctdc.id_duoc_chat) as tongTrongSoDuocChat from
    (select id_duoc_chat, id_thuoc from ct_duoc_chat) ctdc,
    (select id, ten_thuoc, cong_dung from thuoc) t
    where ctdc.id_thuoc = t.id
    group by t.id, ten_thuoc, cong_dung '''
    data = pd.read_sql(sql_query, cnxn)
    #print(data)
    return data, data.iloc[:,0]
```

Hình 3.1.1. Lấy dữ liệu từ database và lưu vào biến (có kiểu pandas) trong python

- Chia cắt dữ liệu: tách id thuốc vào mảng 1 chiều khác, 4 thuộc tính còn lại cắt vào mảng hai chiều

```

__name__ == '__main__':
    data, id_thuoc = read_data_from_sql()
    #print(data)
    data = data.iloc[:,1:]

```

Hình 3.1.2. Tách dữ liệu đọc được từ database

- Chuyển đổi dữ liệu trong cột tenThuoc (tên thuốc) và congDung (công dụng) từ văn bản sang số, sau đó chia toàn bộ dữ liệu cho giá trị lớn nhất trong tất cả dữ liệu để chuẩn hóa dữ liệu sang số thập phân có phạm vi < 1

```

data = convert_text_to_number(data)
# print(data)

```

```

1 usage  DucTrong
def convert_text_to_number(data):
    le = LabelEncoder()
    label = ["tenThuoc", "congDung", "soLuongDuocChat", "tongTrongSoDuocChat"]

    data.iloc[:, 0] = le.fit_transform(data.iloc[:, 0])
    data.iloc[:, 1] = le.fit_transform(data.iloc[:, 1])
    print("max value: ", max(data.max(axis=0)))
    data = data / (max(data.max(axis=0))+1)
    # print(data.iloc[0])
    # print(data.iloc[number_drug-1])

    return data

```

Hình 3.1.3. Chuyển dữ liệu từ văn bản sang số

```

warnings.warn(
max value: 18260

```

	ten_thuoc	cong_dung	soLuongDuocChat	tongTrongSoDuocChat
0	0.024971	0.031050	0.000055	0.078638
1	0.024095	0.030174	0.000110	0.102349
2	0.029133	0.034938	0.000055	0.037457
3	0.023438	0.029626	0.000055	0.078638
4	0.023109	0.029243	0.000055	0.044083
...
3559	0.172937	0.172444	0.000055	0.059635
3560	0.172827	0.172334	0.000055	0.035759
3561	0.172773	0.172280	0.000055	0.020371
3562	0.172718	0.172225	0.000055	0.059854
3563	0.172663	0.172170	0.000055	0.075407

Hình 3.1.4. Chia toàn bộ cho giá trị max là 18260 để có dữ liệu mới là các dữ liệu thập phân

- Tính toán khoảng cách **Euclidean** giữa các điểm dữ liệu (thuốc)

```

distance_matrix = cal_distance_matrix(data)

```

```

def cal_distance_matrix(data):
    number_drug = len(data)
    distance_matrix = np.zeros(shape=(number_drug, number_drug))
    for i in range(len(data)):
        for j in range(0, i):
            if i != j:
                distance_ecl = np.power((data.iloc[i] - data.iloc[j]), 2)
                distance_ecl = np.sqrt(distance_ecl.sum())
                distance_matrix[i][j] = distance_ecl
                distance_matrix[j][i] = distance_matrix[i][j]
    #print(distance_matrix)
    return distance_matrix

```

Hình 3.1.4. Hàm tính toán khoảng cách **Euclidean**

```

np.save( file: "distance_matrix2.npy", distance_matrix)

```

Hình 3.1.5. Lưu kết quả ma trận khoảng cách vào file nhị phân distance_matrix2.npy

```

np.save( file: "id_thuoc.npy", id_thuoc)

```

Hình 3.1.6. Lưu kết quả mảng một chiều chứa id thuốc vào file nhị phân id_thuoc.npy

3.2. Áp dụng thuật toán phân cụm cho dữ liệu

- Đọc dữ liệu từ 2 file nhị phân distance_matrix2.npy, d_thuoc.npy

```

mat = np.load("distance_matrix2.npy")
all_elements = np.load("id_thuoc.npy")

```

- Tạo một biến kiểu pandas index cột và hàng là id_thuoc, giá trị của các hàng và cột là dữ liệu từ ma trận khoảng cách

```

dissimilarity_matrix = pd.DataFrame(mat, index=all_elements, columns=all_elements)
# dissimilarity_matrix = pd.DataFrame(mat)

```

- Tiến hành phân cụm:
 - o Level: cấp độ cụm, level = 0 là cụm ban đầu chứa tất cả thuốc, giá trị level tăng dần đến khi level = <giá trị lớn nhất> là cụm cuối cùng mỗi thuốc là một cụm
 - o Current_clusters: danh sách cụm sau mỗi level (mỗi lần phân cụm)

```

# 1 cụm ban đầu gồm tất cả element:
current_clusters = [all_elements]

level = 1
index = 0
all_of_clusters = {}

while index != -1:
    print(level, current_clusters)
    # dùng deepcopy cho list nested nest
    # nếu không dùng copy thì sau khi rời vòng lặp thì biến current_clusters sẽ nhận giá trị là [all_elements]
    all_of_clusters[str(level)] = copy.deepcopy(current_clusters)

    # chọn cụm current_clusters[index] để phân,
    (a_clstr, b_clstr) = split(current_clusters[index])
    # delete objects, variables, lists, or parts of a list etc.
    # sau khi phân cụm này thành 2 cụm con thì xóa cụm này khỏi list cụm (current_clusters) và thêm 2 cụm con vào list cụm
    del current_clusters[index]
    current_clusters.append(a_clstr)
    current_clusters.append(b_clstr)
    print(current_clusters)
    np.save(file="current_clusters.npy", current_clusters)
    index = max_diameter(current_clusters)
    level += 1

all_of_clusters[str(level)] = copy.deepcopy(current_clusters)
print(all_of_clusters)

return all_of_clusters, level

```

- Hàm split(cụm cần phân): thực hiện phân một cụm thành 2 cụm con
 - Main_list: các id thuộc của cụm được phân, sẽ thay đổi khi id thuộc này được phân sang cụm splinter_group
 - splinter_group: cụm mới chứa các id thuộc được phân từ cụm main_list sang
 - flag (cờ hiệu): đánh dấu cụm main_list có thể phân được nữa không. Flag = 0: ngừng phân cụm và trả về kết quả 2 cụm con main_list, splinter_group, Flag = 1: tiếp tục phân cụm

```

# chọn cụm current_clusters[index] để phân,
(a_clstr, b_clstr) = split(current_clusters[index])
# delete objects, variables, lists, or parts of a list etc.
# sau khi phân cụm này thành 2 cụm con thì xóa cụm này khỏi list cụm (current_clusters) và thêm 2 cụm con vào list cụm

```

```

# thuật toán phân cụm
1 usage  Duc Trong
def split(element_list):
    # giả sử cụm ban đầu cluster_list = ['a','b','c','d','e']
    main_list = list(np.load("main_list.npy"))
    splinter_group = list(np.load("splinter_group.npy"))
    # main_list = element_list
    # splinter_group = []
    print("main_list",main_list)
    print("splinter_group",splinter_group)
    (most_dissm_object_index, flag) = splinter(main_list, splinter_group)
    # print("most_dissm_object_index", most_dissm_object_index)
    # print("flag", flag)
    # de qui
    while (flag > 0):
        main_list.remove(most_dissm_object_index)
        splinter_group.append(most_dissm_object_index)

        print("main_list", main_list)
        print("splinter_group", splinter_group)
        np.save( file: "main_list.npy",main_list)
        np.save( file: "splinter_group.npy",splinter_group)
        (most_dissm_object_index, flag) = splinter(main_list, splinter_group)
        print("flag", flag)
        # print("main list", main_list)
        # print("splinter_group", splinter_group)

    return (main_list, splinter_group)

```

- Hàm splinter:

- most_dissm_object_value: giá trị hiệu của 2 khoảng cách trung bình:
 - từ một thuốc này tới các thuốc cùng cụm
 - từ một thuốc này tới các thuốc khác cụm
 - ➔ sao cho giá trị hiệu này lớn nhất
 - ➔ mục đích: giá trị hiệu lớn nhất tức khoảng cách thuốc này tới các thuốc trong cụm xa hơn các thuốc khác cụm nên chia thuốc này sang cụm khác
 - ➔ nếu hiệu này nhỏ hơn không thì ngược lại: nên không cần chia thuốc này sang cụm khác
- most_dissm_object_index: id thuốc có giá trị hiệu lớn nhất ở trên

```

(most_dissm_object_index, flag) = splinter(main_list, splinter_group)

```

2 usages Duc Trong

```
def splinter(main_list, splinter_group):
    # giả sử cụm ban đầu main_list = ['a', 'b', 'c', 'd', 'e']
    # splinter_group = []

    # giá trị đối tượng khác biệt
    most_dissim_object_value = -np.inf
    most_dissim_object_index = None
    #print(len(main_list))
    for ele in main_list:
        if int(ele) % 500 == 0:
            print("ele", ele)

        # giả sử ele = 'a', main_list = ['a', 'b', 'c', 'd', 'e']
        x = avg_dissim_within_group_element(ele, main_list)
        y = avg_dissim_across_group_element(ele, main_list, splinter_group)
        # print("x", x)
        # print("y", y)
        diff = x - y
        # print("diff", diff)
        if diff > most_dissim_object_value:
            most_dissim_object_value = diff
            most_dissim_object_index = ele
    print("most_dissim_object_value", most_dissim_object_value)
    print("most_dissim_object_index", most_dissim_object_index)
    if most_dissim_object_value > 0:
        # return 1 ở đây là giá trị flag
        return (most_dissim_object_index, 1)
    # nếu giá trị most_dissim_object_value < 0: nghĩa là từng element trong
    else:
        return (-1, -1)
```

- Hàm avg_dissim_within_group_element: tính khoảng cách trung bình sự khác biệt với phần tử trong cùng cụm

```
x = avg_dissim_within_group_element(ele, main_list)
y = avg_dissim_across_group_element(ele, main_list, splinter_group)
# print("x", x)
# print("y", y)
diff = x - y
# print("diff", diff)
```

```

# trung bình sự khác biệt với phần tử trong cùng cụm
1 usage  Duc Trong
def avg_dissim_within_group_element(ele, element_list):
    max_diameter = -np.inf
    sum_dissm = 0
    for i in element_list:
        # giả sử tính tổng khoảng cách từ ele = 'a' đến các phần tử i trong main_list = ['a','b','c','d','e']
        sum_dissm += dissimilarity_matrix[ele][i]
        if (dissimilarity_matrix[ele][i] > max_diameter):
            max_diameter = dissimilarity_matrix[ele][i]
    if (len(element_list) > 1):
        # giả sử tính tb khoảng cách từ a đến các phần tử trong main_list = ['a','b','c','d','e']
        # sum_dissm: tổng của 4 đoạn: ab, ac, ad, ae nên phải chia cho len(main_list) - 1
        avg = sum_dissm / (len(element_list) - 1)
        #print("sum_dissm", sum_dissm)
    else:
        # khi main_list = ['a'] khoảng cách từ a -> chính nó = 0
        avg = 0
    return avg

```

- avg_dissim_across_group_element: tính khoảng cách trung bình sự khác biệt với phần tử trong khác cụm

```

# trung bình sự khác biệt với phần tử ngoài cụm
# tính khoảng cách 1 phần tử ele tới các phần tử nằm trong cụm khác splinter_list
1 usage  Duc Trong
def avg_dissim_across_group_element(ele, main_list, splinter_list):
    if len(splinter_list) == 0:
        return 0
    sum_dissm = 0
    for j in splinter_list:
        sum_dissm = sum_dissm + dissimilarity_matrix[ele][j]
    avg = sum_dissm / (len(splinter_list))
    #print("sum_dissm accross", sum_dissm)
    return avg

```

- Hàm max_diameter: Thuật toán sẽ dừng nếu như việc gộp cụm tạo thành những cụm có độ gắn kết (cohesion) thấp hơn. Độ gắn kết là một tiêu chuẩn để đo chất lượng cụm được tạo thành. Thông thường chúng ta có thể đo lường độ gắn kết dựa trên đường kính (diameter) của cụm sau gộp, đường kính được tính bằng khoảng cách lớn nhất giữa hai điểm trong cụm

```

def max_diameter(cluster_list):

    # lấy dc index của cụm có diameter lớn nhất -> mục đích lấy cụm này để phân cụm tiếp
    max_diameter_cluster_index = None
    # tìm đường kính lớn nhất trong các cụm
    max_diameter_cluster_value = -np.inf
    index = 0

    for element_list in cluster_list:
        for i in element_list:
            for j in element_list:
                if dissimilarity_matrix[i][j] > max_diameter_cluster_value:
                    max_diameter_cluster_value = dissimilarity_matrix[i][j]
                    max_diameter_cluster_index = index

            index += 1

    # khi mỗi object một cụm thì ta dừng phân cụm
    if (max_diameter_cluster_value <= 0):
        return -1

    return max_diameter_cluster_index

```

3.3. Chèn dữ liệu là kết quả của thuật toán vào lại bảng Thuốc liên quan trong database

- Hàm `get_data_insert_sql()`:
 - o Sau mỗi lần phân cụm: ta lưu vào biến `all_of_clusters` có kiểu dictionary (từ điển) với keys: level, value: danh sách cụm của level đó
 - o Ta khởi tạo biến `data_insert_sql` kiểu dictionary (từ điển) với keys: id thuốc đang xem, value: list n thuốc liên quan (n ở đây chọn cụ thể là 10).
 - o Từ biến `all_of_clusters`, ta duyệt ngược từ dưới lên trên (level max tới level = 0): để lấy được thuốc liên quan của từng thuốc và sau mỗi lần tìm thấy thuốc cùng cụm hoặc là cụm cha (level thấp hơn) thì chèn vào value trong biến `data_insert_sql`


```

def get_data_insert_sql():
    start = time.time()
    (all_of_clusters, level) = get_all_clusters()
    end = time.time()
    print("thời gian phân cụm: ", end - start)

    start = time.time()
    data_insert_sql = {}
    print("all_elements", all_elements_copy)
    for i in all_elements_copy:
        number_product = 0
        level_max = level-1
        data_insert_sql[i] = []
        #print("i: ", i)
        while level_max >= 1 and number_product < 10:
            for cluster in all_of_clusters[str(level_max)]:
                #print(cluster)
                if i in cluster:
                    for j in cluster:
                        if j != i and j not in data_insert_sql[i]:
                            data_insert_sql[i].append(j)
                            #print("data_insert_sql", data_insert_sql)
                            number_product += 1
                            if number_product >= 10:
                                break
                            #print("number_product: ", number_product)
                    break
            level_max -= 1

    end = time.time()
    #print(data_insert_sql)

```



```

def write_data_to_sql():
    data = get_data_insert_sql()
    # data = np.load("data_insert_sql_200_records.npy", allow_pickle=True)
    data = pd.DataFrame(data)

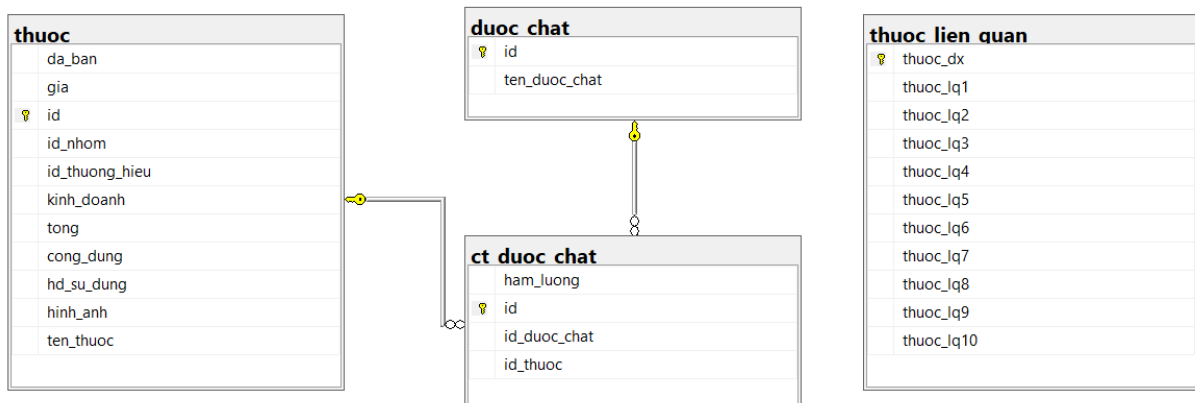
    data = data.transpose()
    data.columns = ['thuoc_lq1', 'thuoc_lq2', 'thuoc_lq3', 'thuoc_lq4', 'thuoc_lq5', 'thuoc_lq6', 'thuoc_lq7',
                    'thuoc_lq8', 'thuoc_lq9', 'thuoc_lq10']
    data['thuoc_dx'] = np.load("id_thuoc.npy")
    # print(data)
    # Insert Dataframe into SQL Server:

    for index, row in data.iterrows():
        #row = row.astype(int)
        cursor.execute("INSERT INTO thuoc_lien_quan values(?,?,?,?,?,?,?,?,?,?)",
                       int(row.thuoc_dx), int(row.thuoc_lq1), int(row.thuoc_lq2), int(row.thuoc_lq3), int(row.thuoc_lq4), int(row.thuoc_lq5), int(row.thuoc_lq6), :
    cnxn.commit()
    cursor.close()

```

CHƯƠNG 4: THIẾT KẾ HỆ THỐNG

4.1. Thiết kế cơ sở dữ liệu



- Kiểu dữ liệu trong bảng Thuốc:

thuoc			
	Column Name	Data Type	Allow Nulls
	da_ban	int	<input type="checkbox"/>
	gia	numeric(38, 2)	<input checked="" type="checkbox"/>
?	id	int	<input type="checkbox"/>
	id_nhom	int	<input checked="" type="checkbox"/>
	id_thuong_hieu	int	<input checked="" type="checkbox"/>
	kinh_doanh	bit	<input checked="" type="checkbox"/>
	tong	int	<input type="checkbox"/>
	cong_dung	nvarchar(MAX)	<input checked="" type="checkbox"/>
	hd_su_dung	nvarchar(MAX)	<input checked="" type="checkbox"/>
	hinh_anh	nvarchar(255)	<input checked="" type="checkbox"/>
	ten_thuoc	nvarchar(255)	<input type="checkbox"/>
			<input type="checkbox"/>


- Kiểu dữ liệu trong bảng dược chất

duoc chat			
	Column Name	Data Type	Allow Nulls
?	id	int	<input type="checkbox"/>
	ten_duoc_chat	nvarchar(100)	<input type="checkbox"/>
			<input type="checkbox"/>

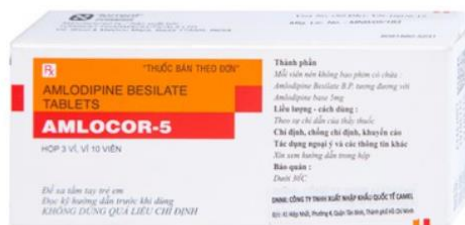
- Kiểu dữ liệu trong bảng chi tiết dược chất:

ct duoc chat			
	Column Name	Data Type	Allow Nulls
	ham_luong	nvarchar(10)	<input type="checkbox"/>
?	id	int	<input type="checkbox"/>
	id_duoc_chat	int	<input type="checkbox"/>
	id_thuoc	int	<input type="checkbox"/>
			<input type="checkbox"/>

- Kiểu dữ liệu trong bảng thuốc liên quan:

thuoc lien quan			
	Column Name	Data Type	Allow Nulls
	thuoc_dx	int	<input type="checkbox"/>
	thuoc_lq1	int	<input checked="" type="checkbox"/>
	thuoc_lq2	int	<input checked="" type="checkbox"/>
	thuoc_lq3	int	<input checked="" type="checkbox"/>
	thuoc_lq4	int	<input checked="" type="checkbox"/>
	thuoc_lq5	int	<input checked="" type="checkbox"/>
	thuoc_lq6	int	<input checked="" type="checkbox"/>
	thuoc_lq7	int	<input checked="" type="checkbox"/>
	thuoc_lq8	int	<input checked="" type="checkbox"/>
	thuoc_lq9	int	<input checked="" type="checkbox"/>
	thuoc_lq10	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

4.2. Thiết kế giao diện



Thuốc Amlocor - 5 Torrent hỗ trợ chống đau thắt ngực (3 vỉ x 10 viên)

266.409 đ

Thêm vào giỏ hàng

Công dụng

Công dụng của Thuốc Amlocor - 5 Torrent Tăng huyết áp & thiếu máu cơ tim kèm đau thắt ngực ổn định.

Hướng dẫn sử dụng

Cách dùng Thuốc Amlocor - 5 Torrent Người lớn: 5 mg x 1 lần/ngày, có thể tăng liều 10 mg/ngày nếu không đáp ứng điều trị sau 2 tuần. Người già & bệnh nhân xơ gan: cần chỉnh liều.

Các sản phẩm liên quan



Thuốc Amlodipine Stada 5mg CAP điều trị tăng...

495.200 đ

Không kinh doanh



Thuốc Amlodipin 5mg Domesco hỗ trợ điều trị tăng...

375.291 đ

Thêm vào giỏ hàng



Thuốc Amlodipine Stada 5mg Tab điều trị tăng...

193.907 đ

Không kinh doanh



Thuốc Amlodipin 5mg Truong Tho điều trị tăng...

157.477 đ

Không kinh doanh



Thuốc Amlodipine Stada



Thuốc Amlor 5mg Pfizer



Thuốc Amlor 5mg Pfizer



Thuốc Amlodipin 5mg

