BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

LUẬN VĂN THẠC SĨ KHOA HỌC

NGÀNH: CÔNG NGHỆ THÔNG TIN

NGHIÊN CỨU VÀ CÀI ĐẶT MỘT SỐ GIẢI THUẬT PHÂN CỤM, PHÂN LỚP

VŨ LAN PHƯƠNG

HÀ NỘI 2006

MỤC LỤC

MỞ ĐẦU 3
MỘT SỐ TỪ VIẾT TẮT VÀ THUẬT NGỮ THƯỜNG DÙNG 5
DANH MỤC BẢNG 6
DANH MỤC HÌNH 7
CHƯƠNG 1: TỔNG QUAN PHÁT HIỆN TRI THỨC VÀ KHAI PHÁ DỮ LIỆU 8
1.1 Giới thiệu chung 8
1.2 Các kỹ thuật khai phá dữ liệu 10
1.3 Lợi thế của khai phá dữ liệu so với các phương pháp khác 13
1.4 Các ứng dụng của KDD và những thách thức đối với KDD 15
1.5 Kết luận 17
CHƯƠNG 2: KỸ THUẬT PHÂN LOẠI TRONG KHAI PHÁ DỮ LIỆU 18
2.1 Phân loại là gì? 18
2.2 Các vấn đề quan tâm của phân loại 20
2.3 Phân loại bằng cây quyết định quy nạp 22
2.4 Phân loại Bayesian 30
2.5 Phân loại bằng lan truyền ngược 37
2.6 Phân loại dựa trên sự kết hợp 48
2.7 Các phương pháp phân loại khác 50
2.8 Độ chính xác classifier 56
2.9 Kết luận 59
CHƯƠNG 3: KỸ THUẬT PHÂN CỤM TRONG KHAI PHÁ DỮ LIỆU 60
3.1 Phân cụm là gì 60
3.2 Các kiểu dữ liệu trong phép phân cụm 64
3.3 Phân loại các phương pháp phân cụm chính 74
3.4 Các phương pháp phân chia 77
3.5 Các phương pháp phân cấp 84
3.6 Các phương pháp phân cụm dựa trên mật độ 94
3.7 Các phương pháp phân cụm dựa trên lưới 101
3.8 Kết luận 107
CHƯƠNG 4: CÀI ĐẶT THỬ NGHIỆM 108
4.1 Thiết kế tổng thể 108
4.2 Chuẩn bị dữ liệu 108
4.3 Thiết kế chương trình 109
4.4 Kết quả thực nghiệm và đánh giá 110
4.5 Kết luận 114
KẾT LUẬN 116
TÀI LIỆU THAM KHẢO 118

LỜI CẢM ƠN

Trước tiên em xin chân thành cảm ơn thầy giáo PGS.TS Nguyễn Ngọc Bình đã tận tình hướng dẫn, chỉ bảo em trong thời gian qua.

Em xin bày tỏ lòng biết ơn tới các thầy cô giáo trong khoa Công nghệ Thông tin nói riêng và trường Đại học Bách Khoa Hà Nội nói chung đã dạy bảo, cung cấp những kiến thức quý báu cho em trong suốt quá trình học tập và nghiên cứu tại trường.

Em cũng xin gửi lời cảm ơn tới gia đình, bạn bè, những người luôn cổ vũ, quan tâm và giúp đỡ em trong suốt thời gian học tập cũng như làm luận văn.

Do thời gian và kiến thức có hạn nên luận văn chắc không tránh khỏi những thiếu sót nhất định. Em rất mong nhận được những sự góp ý quý báu của thầy cô và các bạn.

Hà Nội, 11-2006

Vũ Lan Phương

MỞ ĐẦU

- **Giới thiệu**

Sự phát triển của công nghệ thông tin và việc ứng dụng công nghệ thông tin trong nhiều lĩnh vực của đời sống, kinh tế xã hội trong nhiều năm qua cũng đồng nghĩa với lượng dữ liệu đã được các cơ quan thu thập và lưu trữ ngày một tích luỹ nhiều lên. Họ lưu trữ các dữ liệu này vì cho rằng trong nó ẩn chứa những giá trị nhất định nào đó. Tuy nhiên, theo thống kê thì chỉ có một lượng nhỏ của những dữ liệu này (khoảng từ 5% đến 10%) là luôn được phân tích, số còn lại họ không biết sẽ phải làm gì hoặc có thể làm gì với chúng nhưng họ vẫn tiếp tục thu thập rất tốn kém với ý nghĩ lo sợ rằng sẽ có cái gì đó quan trọng đã bị bỏ qua sau này có lúc cần đến nó. Mặt khác, trong môi trường cạnh tranh, người ta ngày càng cần có nhiều thông tin với tốc độ nhanh để trợ giúp việc ra quyết định và ngày càng có nhiều câu hỏi mang tính chất định tính cần phải trả lời dựa trên một khối lượng dữ liệu khổng lồ đã có. Với những lý do như vậy, các phương pháp quản trị và khai thác cơ sở dữ liệu truyền thống ngày càng không đáp ứng được thực tế đã làm phát triển một khuynh hướng kỹ thuật mới đó là Kỹ thuật phát hiện tri thức và khai phá dữ liệu (KDD - Knowledge Discovery and Data Mining).

Kỹ thuật phát hiện tri thức và khai phá dữ liệu đã và đang được nghiên cứu, ứng dụng trong nhiều lĩnh vực khác nhau ở các nước trên thế giới, tại Việt Nam kỹ thuật này tương đối còn mới mẻ tuy nhiên cũng đang được nghiên cứu và dần đưa vào ứng dụng. Bước quan trọng nhất của quá trình này là Khai phá dữ liệu (Data Mining - DM), giúp người sử dụng thu được những tri thức hữu ích từ những CSDL hoặc các nguồn dữ liệu khổng lồ khác. Rất nhiều doanh nghiệp và tổ chức trên thế giới đã ứng dụng kĩ thuật khai phá dữ liệu vào hoạt động sản xuất kinh doanh của mình và đã thu được những lợi ích to lớn. Nhưng để làm được điều đó, sự phát triển của các mô hình toán học và các giải thuật hiệu quả là chìa khoá quan trọng. Vì vậy, trong luận văn này, tác giả sẽ đề cập tới hai kỹ

thuật thường dùng trong Khai phá dữ liệu, đó là Phân loại (Classification) và Phân cụm (Clustering hay Cluster Analyse).

- **Bố cục luận văn**

Ngoài các phần Mở đầu, Mục lục, Danh mục hình, Danh mục bảng, Kết luận, Tài liệu tham khảo, luận văn được chia làm 4 phần:

Phần I: Tổng quan về Phát hiện tri thức và Khai phá dữ liệu

Phần này giới thiệu một cách tổng quát về quá trình phát hiện tri thức nói chung và khai phá dữ liệu nói riêng. Đặc biệt nhấn mạnh về hai kỹ thuật chính được nghiên cứu trong luận văn đó là Kỹ thuật phân loại và Kỹ thuật phân cụm.

Phần II: Kỹ thuật phân loại (Classification)

Trong phần này, kỹ thuật phân loại được giới thiệu một cách chi tiết. Có nhiều kiểu phân loại như phân loại bằng cây quyết định quy nạp, phân loại Bayesian, phân loại bằng mạng lan truyền ngược, phân loại dựa trên sự kết hợp và các phương pháp phân loại khác. Ngoài ra còn đánh giá độ chính xác của phân loại thông qua các classifier - người phân loại.

Phần III: Kỹ thuật phân cụm (Clustering)

Kỹ thuật phân cụm cũng được chia làm nhiều kiểu: phân cụm phân chia, phân cụm phân cấp, phân cụm dựa trên mật độ và phân cụm dựa trên lưới.

Phần IV: Cài đặt thử nghiệm

Phần này trình bày một số kết quả đã đạt được khi tiến hành áp dụng các giải thuật khai phá dữ liệu để khai thác thông tin dữ liệu mẫu.

MỘT SỐ TỪ VIẾT TẮT VÀ THUẬT NGỮ THƯỜNG DÙNG

KDD	Phát hiện tri thức
DM	Khai phá dữ liệu
Classification	Phân loại
Clustering	Phân cụm
CSDL	Cơ sở dữ liệu

DANH MỤC BẢNG

Bảng 2.1: Các bộ dữ liệu huấn luyện từ cơ sở dữ liệu khách hàng *AllElectronics* 25
Bảng 2.2: Dữ liệu mẫu cho lớp *mua máy tính* 30
Bảng 2.3: Các giá trị đầu vào, trọng số và bias khởi đầu 45
Bảng 2.4: Các tính toán mạng đầu vào và đầu ra 45
Bảng 2.5: Tính toán sai số tại mỗi nút 45
Bảng 2.6: Tính toán việc cập nhật trọng số và bias 45
Bảng 3.1: Bảng ngẫu nhiên cho các biến nhị phân 69
Bảng 3.2: Bảng quan hệ chứa hầu hết các thuộc tính nhị phân 70
Bảng 4.1: Một ví dụ tệp định dạng dữ liệu *.names 109
Bảng 4.2: Một ví dụ tệp dữ liệu *.data 109
Bảng 4.3: Kết quả thí nghiệm phân lớp 111
Bảng 4.4: Kết quả cải thiện chất lượng phân lớp 112
Bảng 4.5: Kết quả thí nghiệm phân loại của Kmeans và Kmedoids 113
Bảng 4.6: Kết quả thí nghiệm phân loại của Kmedoids và See5 113

DANH MỤC HÌNH

Hình 1.1: Quá trình phát hiện tri thức 9
Hình 1.2: Tập dữ liệu với 2 lớp: có và không có khả năng trả nợ....11
Hình 1.3: Phân loại được học bằng mạng nơron cho tập dữ liệu cho vay 12
Hình 1.4: Phân cụm tập dữ liệu cho vay vào trong 3 cụm 13
Hình 2.1: Xử lý phân loại dữ liệu....19
Hình 2.2: Cây quyết định cho khái niệm *mua máy tính* 22
Hình 2.3: Giải thuật ID3 cho cây quyết định 23
Hình 2.4: Thuộc tính *tuổi* có thông tin thu được cao nhất 26
Hình 2.5: Các cấu trúc dữ liệu danh sách thuộc tính và danh sách lớp được dùng trong SLIQ cho dữ liệu mẫu trong bảng 2.2 30
Hình 2.6: a) Mạng belief Bayesian đơn giản, b) Bảng xác suất có điều kiện cho các giá trị của biến *LungCancer* (*LC*)....35
Hình 2.7: Một mạng nơron truyền thẳng đa mức 38
Hình 2.8: Giải thuật lan truyền ngược....41
Hình 2.9: Một unit lớp ẩn hay lớp đầu ra 42
Hình 2.10: Ví dụ một mạng nơron truyền thẳng đa mức 45
Hình 2.11: Các luật có thể được trích ra từ các mạng nơron huấn luyện....48
Hình 2.12: Một xấp xỉ tập thô của tập các mẫu thuộc lớp *C*....54
Hình 2.13: Các giá trị mờ đối với *thu nhập*....55
Hình 2.14: Đánh giá độ chính xác classifier với phương pháp holdout....56
Hình 2.15: Tăng độ chính xác classifier....58
Hình 3.1: Giải thuật *k-means*....79
Hình 3.2: Phân cụm một tập các điểm dựa trên phương pháp *k-means*....79
Hình 3.3: Giải thuật *k-medoids*....82
Hình 3.4: Phân cụm một tập các điểm dựa trên phương pháp *k-medoids*....82
Hình 3.5: Phân cụm một tập các điểm dựa trên phương pháp "Tích đống lồng" 86
Hình 3.6: Phân cụm một tập các điểm bằng CURE 91
Hình 3.7: CHAMELEON: Phân cụm phân cấp dựa trên k-láng giềng gần và mô hình hoá động 93
Hình 3.8: Mật độ tiến và mật độ liên kết trong phân cụm dựa trên mật độ 95
Hình 3.9: Sắp xếp cụm trong OPTICS....98
Hình 3.10: Hàm mật độ và attractor mật độ 99
Hình 3.11: Các cụm được định nghĩa trung tâm và các cụm có hình dạng tuỳ ý 100
Hình 3.12: Một cấu trúc phân cấp đối với phân cụm STING 101
Hình 3.13: Giải thuật phân cụm dựa trên wavelet....105
Hình 3.14: Một mẫu không gian đặc trưng 2 chiều....105
Hình 3.15: Đa phân giải của không gian đặc trưng trong hình 3.14. a) tỷ lệ 1; b) tỷ lệ 2; c) tỷ lệ 3 106
Hình 4.1: Thiết kế chương trình 110
Hình 4.2: Biểu đồ so sánh Kmeans và Kmedoids trong bài toán phân lớp với K=10 111
Hình 4.3: Biểu đồ so sánh Kmeans và Kmedoids trong bài toán phân loại 113
Hình 4.4: Biểu đồ so sánh Kmedoids và See5 trong bài toán phân loại 114

CHƯƠNG 1: TỔNG QUAN PHÁT HIỆN TRI THỨC VÀ KHAI PHÁ DỮ LIỆU

1.1 Giới thiệu chung

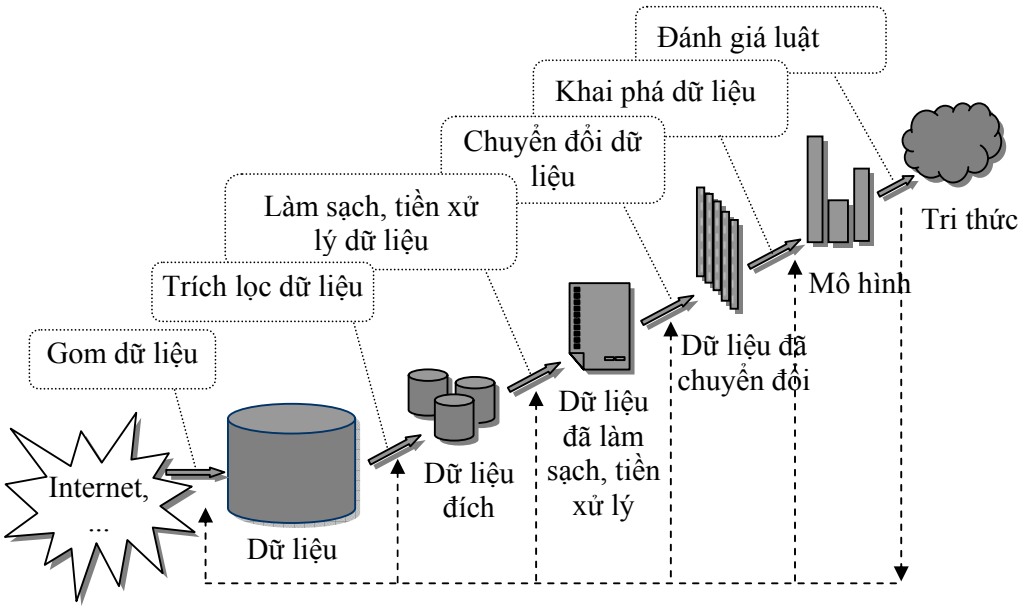
Trong những năm gần đây, sự phát triển mạnh mẽ của CNTT và ngành công nghiệp phần cứng đã làm cho khả năng thu thập và lưu trữ thông tin của các hệ thống thông tin tăng nhanh một cách chóng mặt. Bên cạnh đó việc tin học hoá một cách ồ ạt và nhanh chóng các hoạt động sản xuất, kinh doanh cũng như nhiều lĩnh vực hoạt động khác đã tạo ra cho chúng ta một lượng dữ liệu lưu trữ khổng lồ. Hàng triệu CSDL đã được sử dụng trong các hoạt động sản xuất, kinh doanh, quản lí..., trong đó có nhiều CSDL cực lớn cỡ Gigabyte, thậm chí là Terabyte. Sự bùng nổ này đã dẫn tới một yêu cầu cấp thiết là cần có những kĩ thuật và công cụ mới để tự động chuyển đổi lượng dữ liệu khổng lồ kia thành các tri thức có ích. Từ đó, các kĩ thuật khai phá dữ liệu đã trở thành một lĩnh vực thời sự của nền CNTT thế giới hiện nay.

1.1.1 Khái niệm khai phá dữ liệu

Khai phá dữ liệu (Data Mining) là một khái niệm ra đời vào những năm cuối của thập kỷ 1980. Nó là quá trình trích xuất các thông tin có giá trị tiềm ẩn bên trong lượng lớn dữ liệu được lưu trữ trong các CSDL, kho dữ liệu... Hiện nay, ngoài thuật ngữ khai phá dữ liệu, người ta còn dùng một số thuật ngữ khác có ý nghĩa tương tự như: khai phá tri thức từ CSDL, trích lọc dữ liệu, phân tích dữ liệu/mẫu, khảo cổ dữ liệu, nạo vét dữ liệu. Nhiều người coi Khai phá dữ liệu và một thuật ngữ thông dụng khác là Phát hiện tri thức trong CSDL (Knowlegde Discovery in Databases - KDD) là như nhau. Tuy nhiên trên thực tế, khai phá dữ liệu chỉ là một bước thiết yếu trong quá trình Phát hiện tri thức trong CSDL. Có thể nói Data Mining là giai đoạn quan trọng nhất trong tiến trình Phát hiện tri thức từ cơ sở dữ liệu, các tri thức này hỗ trợ trong việc ra quyết định trong khoa học và kinh doanh.

1.1.2 Các bước của quá trình phát hiện tri thức

Quá trình phát hiện tri thức tiến hành qua 6 giai đoạn như hình 1.1:



Hình 1.1: Quá trình phát hiện tri thức

Bắt đầu của quá trình là kho dữ liệu thô và kết thúc với tri thức được chiết xuất ra. Về lý thuyết thì có vẻ rất đơn giản nhưng thực sự đây là một quá trình rất khó khăn gặp phải rất nhiều vướng mắc như: quản lý các tập dữ liệu, phải lặp đi lặp lại toàn bộ quá trình, v.v...

(1) Gom dữ liệu: Tập hợp dữ liệu là bước đầu tiên trong quá trình khai phá dữ liệu. Đây là bước được khai thác trong một cơ sở dữ liệu, một kho dữ liệu và thậm chí các dữ liệu từ các nguồn ứng dụng Web.

(2) Trích lọc dữ liệu: Ở giai đoạn này dữ liệu được lựa chọn hoặc phân chia theo một số tiêu chuẩn nào đó phục vụ mục đích khai thác, ví dụ chọn tất cả những người có tuổi đời từ 25 - 35 và có trình độ đại học.

(3) Làm sạch, tiền xử lý và chuẩn bị trước dữ liệu: Giai đoạn thứ ba này là giai đoạn hay bị sao lãng, nhưng thực tế nó là một bước rất quan trọng trong quá trình khai phá dữ liệu. Một số lỗi thường mắc phải trong khi gom dữ liệu là tính không đủ chặt chẽ, logíc. Vì vậy, dữ liệu thường chứa các giá trị vô nghĩa và không có khả năng kết nối dữ liệu. Ví dụ: tuổi = 673. Giai đoạn này sẽ tiến hành xử lý những dạng dữ liệu không chặt chẽ nói trên. Những dữ liệu dạng này được xem như thông tin dư thừa, không có giá trị. Bởi vậy, đây là một quá trình rất

quan trọng vì dữ liệu này nếu không được "làm sạch - tiền xử lý - chuẩn bị trước" thì sẽ gây nên những kết quả sai lệch nghiêm trọng.

(4) Chuyển đổi dữ liệu: Tiếp theo là giai đoạn chuyển đổi dữ liệu, dữ liệu đưa ra có thể sử dụng và điều khiển được bởi việc tổ chức lại nó, tức là dữ liệu sẽ được chuyển đổi về dạng phù hợp cho việc khai phá bằng cách thực hiện các thao tác nhóm hoặc tập hợp.

(5) Khai phá dữ liệu: Đây là bước mang tính tư duy trong khai phá dữ liệu. Ở giai đoạn này nhiều thuật toán khác nhau đã được sử dụng để trích ra các mẫu từ dữ liệu. Thuật toán thường dùng là nguyên tắc phân loại, nguyên tắc kết, v.v...

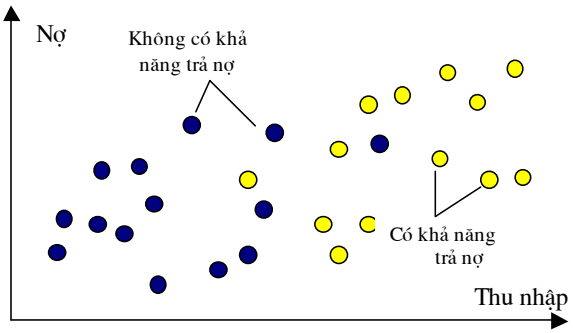
(6) Đánh giá các luật và biểu diễn tri thức: Ở giai đoạn này, các mẫu dữ liệu được chiết xuất ra bởi phần mềm khai phá dữ liệu. Không phải bất cứ mẫu dữ liệu nào cũng đều hữu ích, đôi khi nó còn bị sai lệch. Vì vậy, cần phải ưu tiên những tiêu chuẩn đánh giá để chiết xuất ra các tri thức (Knowlege) cần chiết xuất ra. Đánh giá sự hữu ích của các mẫu biểu diễn tri thức dựa trên một số phép đo. Sau đó sử dụng các kỹ thuật trình diễn và trực quan hoá dữ liệu để biểu diễn tri thức khai phá được cho người sử dụng.

Trên đây là 6 giai đoạn của quá trình phát hiện tri thức, trong đó giai đoạn 5 - khai phá dữ liệu (hay còn gọi đó là Data Mining) là giai đoạn được quan tâm nhiều nhất.

1.2 Các kỹ thuật khai phá dữ liệu

Hình 1.2 biểu diễn một tập dữ liệu giả hai chiều bao gồm 23 case (trường hợp). Mỗi một điểm trên hình đại diện cho một người vay tiền ngân hàng tại một số thời điểm trong quá khứ. Dữ liệu được phân loại vào hai lớp: những người không có khả năng trả nợ và những người tình trạng vay nợ đang ở trạng thái tốt (tức là tại thời điểm đó có khả năng trả nợ ngân hàng).

Hai mục đích chính của khai phá dữ liệu trong thực tế là dự đoán và mô tả.



Hình 1.2: Tập dữ liệu với 2 lớp: có và không có khả năng trả nợ

1.2.1 Khai phá dữ liệu dự đoán

Nhiệm vụ của khai phá dữ liệu dự đoán là đưa ra các dự đoán dựa vào các suy diễn trên dữ liệu hiện thời. Nó sử dụng các biến hay các trường trong cơ sở dữ liệu để dự đoán các giá trị không biết hay các giá trị tương lai. Bao gồm các kĩ thuật: phân loại (classification), hồi quy (regression)...

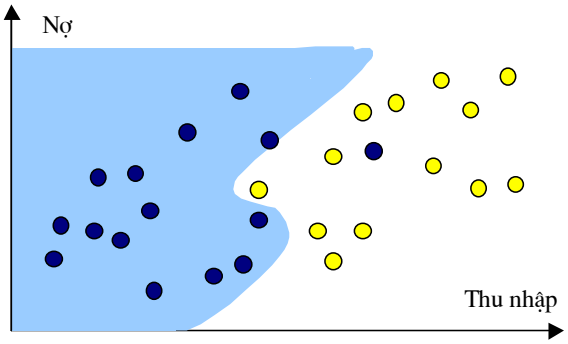
1.2.1.1 Phân loại

Mục tiêu của phương pháp phân loại dữ liệu là dự đoán nhãn lớp cho các mẫu dữ liệu. Quá trình phân loại dữ liệu thường gồm 2 bước: xây dựng mô hình và sử dụng mô hình để phân loại dữ liệu.

• Bước 1: Xây dựng mô hình dựa trên việc phân tích các mẫu dữ liệu cho trước. Mỗi mẫu thuộc về một lớp, được xác định bởi một thuộc tính gọi là thuộc tính lớp. Các mẫu dữ liệu này còn được gọi là tập dữ liệu huấn luyện. Các nhãn lớp của tập dữ liệu huấn luyện đều phải được xác định trước khi xây dựng mô hình, vì vậy phương pháp này còn được gọi là học có giám sát.

• Bước 2: Sử dụng mô hình để phân loại dữ liệu. Trước hết chúng ta phải tính độ chính xác của mô hình. Nếu độ chính xác là chấp nhận được, mô hình sẽ được sử dụng để dự đoán nhãn lớp cho các mẫu dữ liệu khác trong tương lai.

Hay nói cách khác, phân loại là học một hàm ánh xạ một mục dữ liệu vào một trong số các lớp cho trước. Hình 1.3 cho thấy sự phân loại của các dữ liệu vay nợ vào trong hai miền lớp. Ngân hàng có thể sử dụng các miền phân loại để tự động quyết định liệu những người vay nợ trong tương lai có nên cho vay hay không.



Hình 1.3: Phân loại được học bằng mạng nơron cho tập dữ liệu cho vay

1.2.1.2 Hồi quy

Phương pháp hồi qui khác với phân loại dữ liệu ở chỗ, hồi qui dùng để dự đoán về các giá trị liên tục còn phân loại dữ liệu thì chỉ dùng để dự đoán về các giá trị rời rạc.

Hồi quy là học một hàm ánh xạ một mục dữ liệu vào một biến dự báo giá trị thực. Các ứng dụng hồi quy có nhiều, ví dụ như đánh giá xác xuất một bệnh nhân sẽ chết dựa trên tập kết quả xét nghiệm chẩn đoán, dự báo nhu cầu của người tiêu dùng đối với một sản phẩn mới dựa trên hoạt động quảng cáo tiêu dùng.

1.2.2 Khai phá dữ liệu mô tả

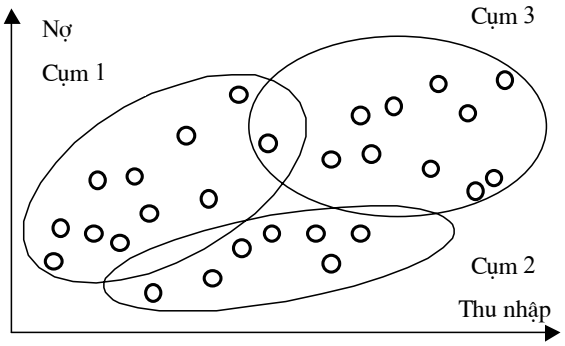
Kỹ thuật này có nhiệm vụ mô tả về các tính chất hoặc các đặc tính chung của dữ liệu trong CSDL hiện có. Bao gồm các kỹ thuật: phân cụm (clustering), phân tích luật kết hợp (association rules)...

1.2.2.1 Phân cụm

Mục tiêu chính của phương pháp phân cụm dữ liệu là nhóm các đối tượng tương tự nhau trong tập dữ liệu vào các cụm sao cho các đối tượng thuộc cùng một cụm là tương đồng còn các đối tượng thuộc các cụm khác nhau sẽ không tương đồng. Phân cụm dữ liệu là một ví dụ của phương pháp học không giám sát. Không giống như phân loại dữ liệu, phân cụm dữ liệu không đòi hỏi phải định nghĩa trước các mẫu dữ liệu huấn luyện. Vì thế, có thể coi phân cụm dữ liệu là một cách học bằng quan sát (learning by observation), trong khi phân loại dữ liệu là học bằng ví dụ (learning by example). Trong phương pháp này bạn sẽ

không thể biết kết quả các cụm thu được sẽ như thế nào khi bắt đầu quá trình. Vì vậy, thông thường cần có một chuyên gia về lĩnh vực đó để đánh giá các cụm thu được. Phân cụm dữ liệu được sử dụng nhiều trong các ứng dụng về phân đoạn thị trường, phân đoạn khách hàng, nhận dạng mẫu, phân loại trang Web… Ngoài ra phân cụm dữ liệu còn có thể được sử dụng như một bước tiền xử lí cho các thuật toán khai phá dữ liệu khác.

Hình 1.4 cho thấy sự phân cụm tập dữ liệu cho vay vào trong 3 cụm: lưu ý rằng các cụm chồng lên nhau cho phép các điểm dữ liệu thuộc về nhiều hơn một cụm.



Hình 1.4: Phân cụm tập dữ liệu cho vay vào trong 3 cụm

1.2.2.2 Luật kết hợp

Mục tiêu của phương pháp này là phát hiện và đưa ra các mối liên hệ giữa các giá trị dữ liệu trong CSDL. Mẫu đầu ra của giải thuật khai phá dữ liệu là tập luật kết hợp tìm được. Khai phá luật kết hợp được thực hiện qua 2 bước:

• Bước 1: tìm tất cả các tập mục phổ biến, một tập mục phổ biến được xác định qua tính độ hỗ trợ và thỏa mãn độ hỗ trợ cực tiểu.

• Bước 2: sinh ra các luật kết hợp mạnh từ tập mục phổ biến, các luật phải thỏa mãn độ hỗ trợ cực tiểu và độ tin cậy cực tiểu.

Phương pháp này được sử dụng rất hiệu quả trong các lĩnh vực như marketing có chủ đích, phân tích quyết định, quản lí kinh doanh,…

1.3 Lợi thế của khai phá dữ liệu so với các phương pháp khác

Khai phá dữ liệu là một lĩnh vực liên quan tới rất nhiều ngành học khác như: hệ CSDL, thống kê,... Hơn nữa, tuỳ vào cách tiếp cận được sử dụng, khai phá dữ liệu còn có thể áp dụng một số kĩ thuật như mạng nơ ron, lí thuyết tập thô hoặc tập mờ, biểu diễn tri thức… Như vậy, khai phá dữ liệu thực ra là dựa trên các phương pháp cơ bản đã biết. Tuy nhiên, sự khác biệt của khai phá dữ liệu so với các phương pháp đó là gì? Tại sao khai phá dữ liệu lại có ưu thế hơn hẳn các phương pháp cũ? Ta sẽ lần lượt xem xét và giải quyết các câu hỏi này.

1.3.1 Học máy (Machine Learning)

So với phương pháp học máy, khai phá dữ liệu có lợi thế hơn ở chỗ, khai phá dữ liệu có thể sử dụng với các cơ sở dữ liệu thường động, không đầy đủ, bị nhiễu và lớn hơn nhiều so với các tập dữ liệu học máy điển hình. Trong khi đó phương pháp học máy chủ yếu được áp dụng trong các CSDL đầy đủ, ít biến động và tập dữ liệu không quá lớn.

Thật vậy, trong học máy, thuật ngữ cơ sở dữ liệu chủ yếu đề cập tới một tập các mẫu được lưu trong tệp. Các mẫu thường là các vectơ với độ dài cố định, thông tin về đặc điểm, dãy các giá trị của chúng đôi khi cũng được lưu lại như trong từ điển dữ liệu. Một giải thuật học sử dụng tập dữ liệu và các thông tin kèm theo tập dữ liệu đó làm đầu vào và đầu ra biểu thị kết quả của việc học. Học máy có khả năng áp dụng cho cơ sở dữ liệu, lúc này, học máy sẽ không phải là học trên tập các mẫu nữa mà học trên tập các bản ghi của cơ sở dữ liệu. Tuy nhiên, trong thực tế, cơ sở dữ liệu thường động, không đầy đủ và bị nhiễu, lớn hơn nhiều so với các tập dữ liệu học máy điển hình. Các yếu tố này làm cho hầu hết các giải thuật học máy trở nên không hiệu quả. Khai phá dữ liệu lúc này sẽ xử lý các vấn đề vốn đã điển hình trong học máy và vượt quá khả năng của học máy, đó là sử dụng được các CSDL chứa nhiều nhiễu, dữ liệu không đầy đủ hoặc biến đổi liên tục.

1.3.2 Hệ chuyên gia (Expert Systems)

Các hệ chuyên gia nắm bắt các tri thức cần thiết cho một bài toán nào đó. Các kỹ thuật thu thập giúp cho việc lấy tri thức từ các chuyên gia con người.

Mỗi phương pháp hệ chuyên gia là một cách suy diễn các luật từ các ví dụ và giải pháp đối với bài toán chuyên gia đưa ra. Phương pháp hệ chuyên gia khác với khai phá dữ liệu ở chỗ các ví dụ của chuyên gia thường ở mức chất lượng cao hơn nhiều so với các dữ liệu trong CSDL, và chúng thường chỉ bao hàm được các trường quan trọng. Hơn nữa các chuyên gia sẽ xác nhận giá trị và tính hữu ích của các mẫu phát hiện được.

1.3.3 Thống kê (Statistics)

Mặc dù các phương pháp thống kê cung cấp một nền tảng lý thuyết vững chắc cho các bài toán phân tích dữ liệu nhưng chỉ có tiếp cận thống kê thuần tuý thôi chưa đủ bởi:

• Các phương pháp thống kê không phù hợp với các kiểu dữ liệu có cấu trúc trong rất nhiều các cơ sở dữ liệu

• Thống kê hoàn toàn tính toán trên dữ liệu, nó không sử dụng tri thức sẵn có về lĩnh vực quan tâm

• Các kết quả của phân tích thống kê có thể rất nhiều và khó có thể làm rõ được

• Các phương pháp thống kê cần có sự hướng dẫn của người dùng để xác định phân tích dữ liệu như thế nào và ở đâu.

Phương pháp thống kê là một trong những nền tảng lí thuyết của khai phá dữ liệu. Sự khác nhau cơ bản giữa khai phá dữ liệu và thống kê ở chỗ khai phá dữ liệu là một phương tiện được dùng bởi người sử dụng đầu cuối chứ không phải là các nhà thống kê. Khai phá dữ liệu đã khắc phục được các yếu điểm trên của thống kê, tự động quá trình thống kê một cách hiệu quả vì thế giảm bớt công việc của người dùng đầu cuối, tạo ra một công cụ dễ sử dụng hơn.

1.4 Các ứng dụng của KDD và những thách thức đối với KDD

1.4.1 Các ứng dụng của KDD

Các kỹ thuật KDD có thể được áp dụng vào trong nhiều lĩnh vực:

- Thông tin thương mại: Phân tích dữ liệu tiếp thị và bán hàng, phân tích vốn đầu tư, chấp thuận cho vay, phát hiện gian lận, ...

- Thông tin sản xuất: Điều khiển và lập lịch, quản lý mạng, phân tích kết quả thí nghiệm, ...
- Thông tin khoa học: Địa lý: Phát hiện động đất,...
- ...

1.4.2 Những thách thức đối với KDD

- *Các cơ sở dữ liệu lớn hơn rất nhiều:* cơ sở dữ liệu với hàng trăm trường và bảng, hàng triệu bản ghi và kích thước lên tới nhiều gigabyte là vấn đề hoàn toàn bình thường và cơ sở dữ liệu terabyte (10^{12} bytes) cũng đã bắt đầu xuất hiện.
- *Số chiều cao:* Không chỉ thường có một số lượng rất lớn các bản ghi trong cơ sở dữ liệu mà còn có một số lượng rất lớn các trường (các thuộc tính, các biến) làm cho số chiều của bài toán trở nên cao. Thêm vào đó, nó tăng thêm cơ hội cho một giải thuật khai phá dữ liệu tìm ra các mẫu không hợp lệ. Vậy nên cần giảm bớt hiệu quả kích thước của bài toán và tính hữu ích của tri thức cho trước để nhận biết các biến không hợp lệ.
- *Over-fitting (quá phù hợp):* Khi giải thuật tìm kiếm các tham số tốt nhất cho một mô hình đặc biệt sử dụng một tập hữu hạn dữ liệu, kết quả là mô hình biểu diễn nghèo nàn trên dữ liệu kiểm định. Các giải pháp có thể bao gồm hợp lệ chéo, làm theo quy tắc và các chiến lược thống kê tinh vi khác.
- *Thay đổi dữ liệu và tri thức:* Thay đổi nhanh chóng dữ liệu (động) có thể làm cho các mẫu được phát hiện trước đó không còn hợp lệ. Thêm vào đó, các biến đã đo trong một cơ sở dữ liệu ứng dụng cho trước có thể bị sửa đổi, xoá bỏ hay tăng thêm các phép đo mới. Các giải pháp hợp lý bao gồm các phương pháp tăng trưởng để cập nhật các mẫu và xử lý thay đổi.
- *Dữ liệu thiếu và bị nhiễu:* Bài toán này đặc biệt nhạy trong các cơ sở dữ liệu thương mại. Dữ liệu điều tra dân số U.S cho thấy tỷ lệ lỗi lên tới 20%. Các thuộc tính quan trọng có thể bị mất nếu cơ sở dữ liệu không được thiết kế với sự khám phá bằng trí tuệ. Các giải pháp có thể gồm nhiều chiến lược thống kê phức tạp để nhận biết các biến ẩn và các biến phụ thuộc.

- *Mối quan hệ phức tạp giữa các trường:* Các thuộc tính hay các giá trị có cấu trúc phân cấp, các quan hệ giữa các thuộc tính và các phương tiện tinh vi hơn cho việc biểu diễn tri thức về nội dung của một cơ sở dữ liệu sẽ đòi hỏi các giải thuật phải có khả năng sử dụng hiệu quả các thông tin này. Về mặt lịch sử, các giải thuật khai phá dữ liệu được phát triển cho các bản ghi có giá trị thuộc tính đơn giản, mặc dầu các kỹ thuật mới bắt nguồn từ mối quan hệ giữa các biến đang được phát triển.
- *Tính dễ hiểu của các mẫu:* Trong nhiều ứng dụng, điều quan trọng là những gì khai thác được phải càng dễ hiểu đối với con người thì càng tốt. Các giải pháp có thể thực hiện được bao gồm cả việc biểu diễn được minh hoạ bằng đồ thị, cấu trúc luật với các đồ thị có hướng, biểu diễn bằng ngôn ngữ tự nhiên và các kỹ thuật hình dung ra dữ liệu và tri thức.
- *Người dùng tương tác và tri thức sẵn có:* Nhiều phương pháp KDD hiện hành và các công cụ không tương tác thực sự với người dùng và không thể dễ dàng kết hợp chặt chẽ với tri thức có sẵn về một bài toán loại trừ theo các cách đơn giản. Việc sử dụng của miền tri thức là quan trọng trong toàn bộ các bước của xử lý KDD.
- *Tích hợp với các hệ thống khác:* Một hệ thống phát hiện đứng một mình có thể không hữu ích lắm. Các vấn đề tích hợp điển hình gồm có việc tích hợp với một DBMS (tức là qua một giao diện truy vấn), tích hợp với các bảng tính và các công cụ trực quan và điều tiết các dự đoán cảm biến thời gian thực.

1.5 Kết luận

Khai phá dữ liệu là lĩnh vực đã và đang trở thành một trong những hướng nghiên cứu thu hút được sự quan tâm của nhiều chuyên gia về CNTT trên thế giới. Trong những năm gần đây, rất nhiều các phương pháp và thuật toán mới liên tục được công bố. Điều này chứng tỏ những ưu thế, lợi ích và khả năng ứng dụng thực tế to lớn của khai phá dữ liệu. Phần này đã trình bày một số kiến thức tổng quan về khai phá dữ liệu, những kiến thức cơ bản nhất về các phương pháp phân cụm dữ liệu, phân loại dữ liệu và khai phá luật kết hợp.

CHƯƠNG 2: KỸ THUẬT PHÂN LOẠI TRONG KHAI PHÁ DỮ LIỆU

Các cơ sở dữ liệu với rất nhiều thông tin ẩn có thể được sử dụng để tạo nên các quyết định kinh doanh thông minh. Phân loại là một dạng của phân tích dữ liệu, nó dùng để trích ra các mô hình mô tả các lớp dữ liệu quan trọng hay để dự đoán các khuynh hướng dữ liệu tương lai. Phân loại dùng để dự đoán các nhãn xác thực (hay các giá trị rời rạc). Nhiều phương pháp phân loại được đề xuất bởi các nhà nghiên cứu các lĩnh vực như học máy, hệ chuyên gia, thống kê... Hầu hết các giải thuật dùng với giả thiết kích thước dữ liệu nhỏ. Các nghiên cứu khai phá cơ sở dữ liệu gần đây đã phát triển, xây dựng mở rộng các kỹ thuật phân loại có khả năng sử dụng dữ liệu thường trú trên đĩa lớn. Các kỹ thuật này thường được xem xét xử lý song song và phân tán.

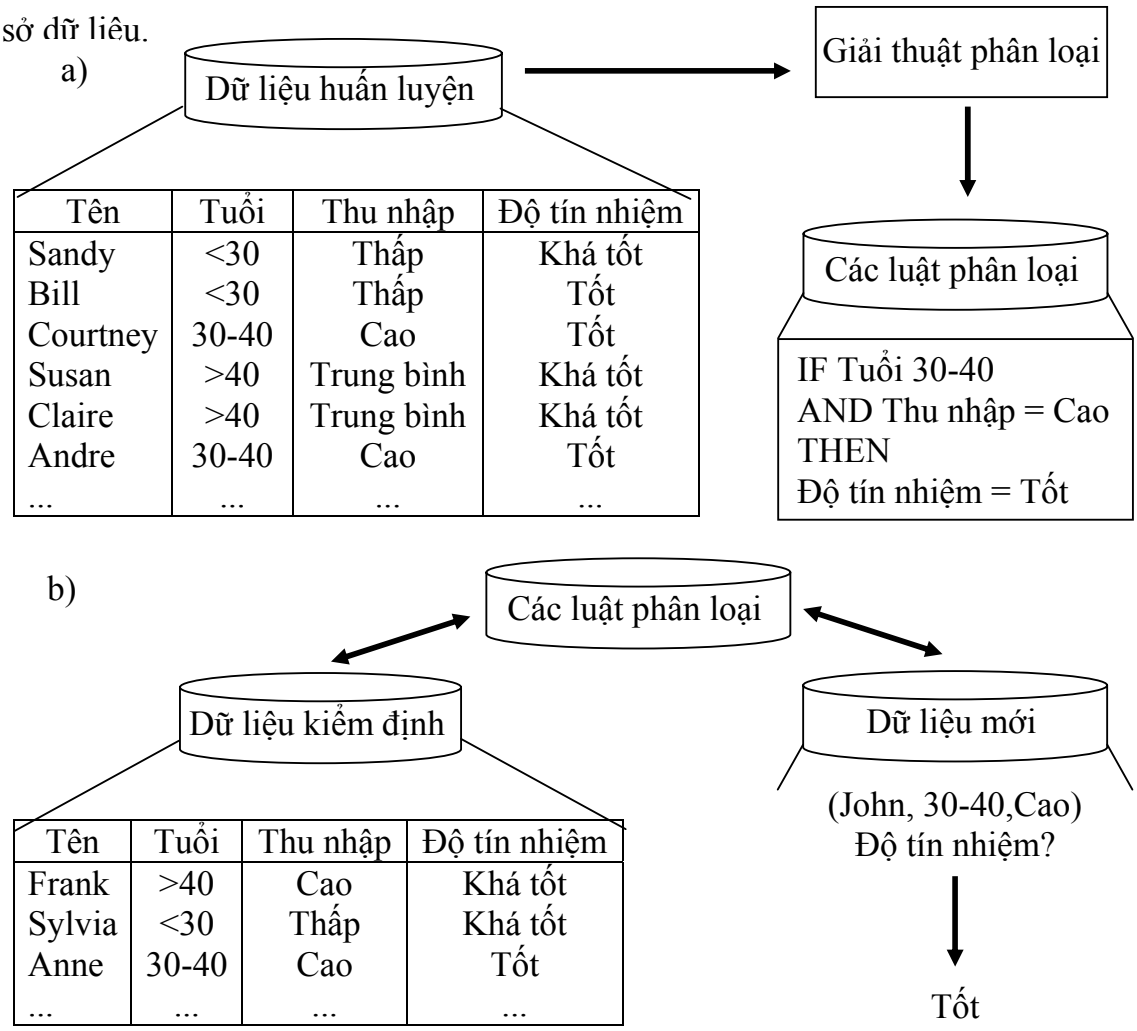
Trong chương này, ta sẽ xem xét các kỹ thuật cơ bản để phân loại dữ liệu như cây quyết định quy nạp, phân loại Bayesian, các mạng belief Bayesian, các mạng nơron và phân loại dựa trên sự kết hợp. Các tiếp cận khác của phân loại như các kỹ thuật classifier *k*-láng giềng gần nhất, lập luận dựa trên tình huống, giải thuật di truyền, tập thô và logic mờ cũng được đề cập.

2.1 Phân loại là gì?

Phân loại dữ liệu là một xử lý bao gồm hai bước (Hình 2.1). Ở bước đầu tiên, xây dựng mô hình mô tả một tập cho trước các lớp dữ liệu. Mô hình này có được bằng cách phân tích các bộ cơ sở dữ liệu. Mỗi bộ được giả định thuộc về một lớp cho trước, các lớp này chính là các giá trị của một thuộc tính được chỉ định, gọi là thuộc tính nhãn lớp. Các bộ dữ liệu để xây dựng mô hình gọi là tập dữ liệu huấn luyện. Do nhãn lớp của mỗi mẫu huấn luyện đã biết trước nên bước này cũng được biết đến như là học có giám sát. Điều này trái ngược với học không có giám sát, trong đó các mẫu huấn luyện chưa biết sẽ thuộc về nhãn lớp nào và số lượng hay tập các lớp được học chưa biết trước.

Mô hình học được biểu diễn dưới dạng các luật phân loại, cây quyết định hay công thức toán học. Ví dụ, cho trước một cơ sở dữ liệu thông tin về độ tín nhiệm của khách hàng, các luật phân loại được học để nhận biết các khách hàng

có *độ tín nhiệm* là *tốt* hay *khá tốt* (Hình 2.1a). Các luật được dùng để phân loại các mẫu dữ liệu tương lai cũng như cung cấp cách hiểu tốt hơn về nội dung cơ sở dữ liệu.



Tên	Tuổi	Thu nhập	Độ tín nhiệm
Sandy	<30	Thấp	Khá tốt
Bill	<30	Thấp	Tốt
Courtney	30-40	Cao	Tốt
Susan	>40	Trung bình	Khá tốt
Claire	>40	Trung bình	Khá tốt
Andre	30-40	Cao	Tốt
...	...	...	...

Tên	Tuổi	Thu nhập	Độ tín nhiệm
Frank	>40	Cao	Khá tốt
Sylvia	<30	Thấp	Khá tốt
Anne	30-40	Cao	Tốt
...	...	...	...

Hình 2.1: Xử lý phân loại dữ liệu

Trong bước thứ hai (hình 2.1b), mô hình được dùng để phân loại. Trước tiên, đánh giá độ chính xác dự đoán của mô hình (hay classifier). Phần 2.8 của chương này mô tả một số phương pháp đánh giá độ chính xác classifier. Phương pháp *holdout* là một kỹ thuật đơn giản sử dụng một tập kiểm định các mẫu đã được gắn nhãn lớp. Các mẫu này được chọn lựa ngẫu nhiên và độc lập với các mẫu huấn luyện. Độ chính xác của mô hình trên một tập kiểm định cho trước là phần trăm các mẫu của tập kiểm định được mô hình phân loại đúng. Đối với mỗi mẫu kiểm định, nhãn lớp đã biết được so sánh với dự đoán lớp của mô hình đã học cho mẫu đó. Nếu độ chính xác của mô hình được đánh giá dựa trên tập dữ

liệu huấn luyện, sự đánh giá này có thể là tối ưu, do vậy mô hình học có khuynh hướng quá phù hợp (*overfit*) dữ liệu. Bởi vậy, cần dùng một tập kiểm định.

Nếu độ chính xác của mô hình là chấp nhận được, mô hình có thể được sử dụng để phân loại các bộ hay các đối tượng dữ liệu tương lai mà chưa biết nhãn lớp. Ví dụ, các luật phân loại học trong hình 2.1a: việc phân tích dữ liệu khách hàng từ các khách hàng đã tồn tại có thể được dùng để dự đoán *độ tín nhiệm* của các khách hàng mới.

Ví dụ 2.1: Giả sử rằng ta có một cơ sở dữ liệu các khách hàng trên danh sách thư (mailing list) *AllElectronics*. Danh sách thư được dùng để gửi đi các tài liệu quảng cáo mô tả các sản phẩm mới và yết lên các sản phẩm hạ giá. Cơ sở dữ liệu mô tả các thuộc tính của khách hàng như *tên*, *tuổi*, *thu nhập*, *nghề nghiệp* và *độ tín nhiệm*. Khách hàng được phân loại vào nhóm người mua hay không mua máy tính tại *AllElectronics*. Giả sử rằng các khách hàng mới được thêm vào cơ sở dữ liệu và bạn sẽ thông báo cho những khách hàng này thông tin bán máy tính. Thay vì gửi tài liệu quảng cáo tới từng khách hàng mới, ta chỉ gửi tài liệu quảng cáo tới những người có khả năng muốn mua máy tính, như vậy chi phí sẽ hiệu quả hơn. Mô hình phân loại được xây dựng và sử dụng cho mục đích này.

2.2 Các vấn đề quan tâm của phân loại

2.2.1 Chuẩn bị dữ liệu để phân loại: Các bước tiền xử lý dữ liệu sau đây giúp cải thiện độ chính xác, hiệu suất và khả năng mở rộng của phân loại.

- *Làm sạch dữ liệu*: Đây là quá trình thuộc về tiền xử lý dữ liệu để gỡ bỏ hoặc làm giảm nhiễu và cách xử lý các giá trị khuyết. Bước này giúp làm giảm sự mập mờ khi học.

- *Phân tích sự thích hợp*: Nhiều thuộc tính trong dữ liệu có thể không thích hợp hay không cần thiết để phân loại. Vì vậy, phép phân tích sự thích hợp được thực hiện trên dữ liệu với mục đích gỡ bỏ bất kỳ những thuộc tính không thích hợp hay không cần thiết. Trong học máy, bước này gọi là trích chọn đặc trưng. Phép phân tích này giúp phân loại hiệu quả và nâng cao khả năng mở rộng.

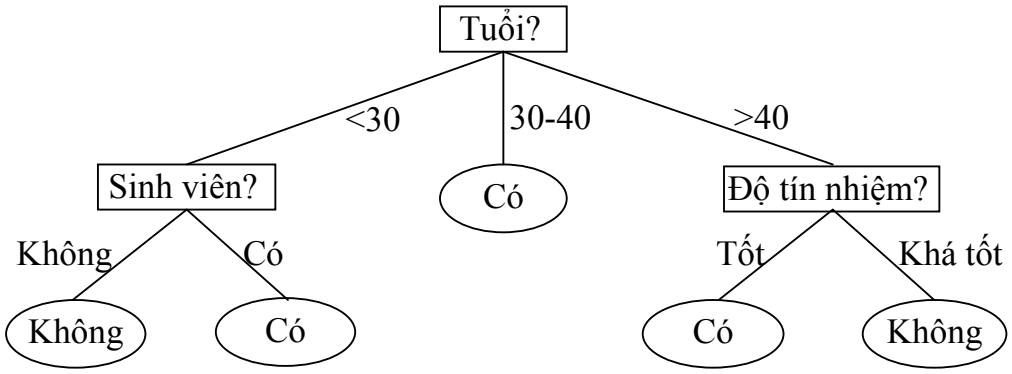
- *Biến đổi dữ liệu*: Dữ liệu có thể được tổng quát hoá tới các mức khái niệm cao hơn. Điều này rất hữu ích cho các thuộc tính có giá trị liên tục. Ví dụ, các giá trị số của thuộc tính *thu nhập* được tổng quát hoá sang các phạm vi rời rạc như *thấp*, *trung bình* và *cao*. Tương tự, các thuộc tính giá trị tên như *đường phố* được tổng quát hoá tới khái niệm mức cao hơn như *thành phố*. Nhờ đó các thao tác vào/ra trong quá trình học sẽ ít đi.

Dữ liệu cũng có thể được tiêu chuẩn hoá, đặc biệt khi các mạng nơron hay các phương pháp dùng phép đo khoảng cách trong bước học. Tiêu chuẩn hoá biến đổi theo tỷ lệ tất cả các giá trị của một thuộc tính cho trước để chúng rơi vào phạm vi chỉ định nhỏ như [-1.0,1.0] hay [0,1.0]. Tuy nhiên điều này sẽ cản trở các thuộc tính có phạm vi ban đầu lớn (như *thu nhập*) có nhiều ảnh hưởng hơn đối với các thuộc tính có phạm vi nhỏ hơn ban đầu (như các thuộc tính nhị phân).

2.2.2 So sánh các phương pháp phân loại: Các phương pháp phân loại có thể được so sánh và đánh giá theo các tiêu chí sau:

- *Độ chính xác dự đoán*: Dựa trên khả năng mô hình dự đoán đúng nhãn lớp của dữ liệu mới.
- *Tốc độ*: Dựa trên các chi phí tính toán. Chi phí này bao gồm sinh và sử dụng mô hình.
- *Sự tráng kiện*: Dựa trên khả năng mô hình đưa ra các dự đoán chính xác dữ liệu nhiễu hay dữ liệu với các giá trị khuyết cho trước.
- *Khả năng mở rộng*: Dựa trên khả năng trình diễn hiệu quả của mô hình đối với dữ liệu lớn.
- *Khả năng diễn dịch*: Dựa trên mức khả năng mà mô hình cung cấp để hiểu thấu đáo dữ liệu.

2.3 Phân loại bằng cây quyết định quy nạp



Hình 2.2: Cây quyết định cho khái niệm *mua máy tính*

"Cây quyết định là gì?"

Cây quyết định là cấu trúc cây có dạng biểu đồ luồng, mỗi nút trong là kiểm định trên một thuộc tính, mỗi nhánh đại diện cho một kết quả kiểm định, các nút lá đại diện cho các lớp. Nút cao nhất trên cây là nút gốc. Hình 2.2 thể hiện cây quyết định biểu diễn khái niệm *mua máy tính*, nó dự đoán liệu một khách hàng tại *AllElectronics* có mua máy tính hay không. Hình chữ nhật biểu thị các nút trong, hình elip biểu thị các nút lá.

Để phân loại một mẫu chưa biết, các giá trị thuộc tính của mẫu sẽ được kiểm định trên cây. Đường đi từ gốc tới một nút lá cho biết dự đoán lớp đối với mẫu đó. Cây quyết định có thể dễ dàng chuyển đổi thành các luật phân loại.

Mục 2.3.1 là giải thuật học cơ bản của cây quyết định. Khi cây quyết định được xây dựng, nhiều nhánh có thể phản ánh nhiễu hay các outlier trong dữ liệu huấn luyện. Việc cắt tỉa cây cố gắng nhận biết và gỡ bỏ các nhánh này. Cây cắt tỉa được mô tả trong mục 2.3.3. Cải tiến giải thuật cây quyết định cơ bản được đề cập tới trong mục 2.3.4. Các vấn đề về khả năng mở rộng cho cây quyết định quy nạp từ cơ sở dữ liệu lớn được đề cập trong mục 2.3.5.

2.3.1 Cây quyết định quy nạp

Giải thuật 2.3.1 Generate_decision_tree (Sinh cây quyết định): Xây dựng cây quyết định từ dữ liệu huấn luyện cho trước.

Đầu vào: Các mẫu huấn luyện *samples*, là các giá trị rời rạc của các thuộc tính;

---

tập các thuộc tính *attribute-list*.

Đầu ra: Cây quyết định.

Giải thuật:

1) **create** một nút *N*;

2) **if** tất cả các *samples* có cùng lớp *C* **then**

3) **return** *N* là một nút lá với nhãn lớp *C*;

4) **if** *attribute-list* là rỗng **then**

5) **return** *N* là một nút lá với nhãn là lớp phổ biến nhất trong *samples*;

6) **select** *test-attribute* - là thuộc tính có thông tin thu được cao nhất trong *attribute-list*;

7) Nhãn nút *N* là *test-attribute*;

8) **for** mỗi một giá trị a_i của *test-attribute*

9) Phát triển một nhánh từ nút *N* với điều kiện *test-attribute*= a_i;

10) Đặt s_i là tập các mẫu trong *samples* có *test-attribute*= a_i;

11) **if** s_i là rỗng **then**

12) gắn một lá với nhãn là lớp phổ biến nhất trong *samples*;

13) **else** gắn một nút được trả lại bởi Generate_decision_tree(s_i, *attribute-list* - *test-attribute*);

Hình 2.3: Giải thuật ID3 cho cây quyết định

Giải thuật nền tảng của cây quyết định quy nạp là ID3, một giải thuật cây quyết định quy nạp nổi tiếng. Mở rộng giải thuật được thảo luận trong mục 2.3.4 tới 2.3.6.

** Phép đo lựa chọn thuộc tính:*

Phép đo thông tin thu được (information gain) được dùng để lựa chọn thuộc tính kiểm định tại mỗi nút trên cây. Phép đo như vậy còn được gọi là *phép đo lựa chọn thuộc tính* hay *phép đo chất lượng phân chia*. Thuộc tính với thông tin thu được cao nhất (hay entropy lớn nhất) được chọn là thuộc tính kiểm định tại nút hiện thời. Thuộc tính này tối thiểu hoá thông tin cần thiết để phân loại các mẫu. Phép đo thông tin này sẽ tiến tới cực tiểu hoá số lượng các kiểm định cần

thiết để phân loại một đối tượng và đảm bảo rằng một cây đơn giản (nhưng không nhất thiết phải là đơn giản nhất) được tìm thấy.

Cho S là tập gồm s mẫu dữ liệu. Giả sử thuộc tính nhãn lớp có m giá trị riêng biệt định nghĩa m lớp riêng biệt *(với i = 1,...,m)*, s_i là số lượng các mẫu của S trong lớp C_i. Thông tin cần thiết để phân loại một mẫu cho trước được thể hiện trong phương trình (2.1):

$$I(s_1, s_2, ..., s_m) = -\sum_{i=1}^{m} p_i \log_2(p_i) \qquad (2.1)$$

với p_i là xác suất một mẫu tuỳ ý thuộc lớp C_i và bằng s_i/s.

Cho thuộc tính A có v giá trị riêng biệt, $\{a_1,a_2,...,a_v\}$. Thuộc tính A dùng để phân chia S vào trong v tập con $\{S_1,S_2,...,S_v\}$, S_i là các mẫu trong S có giá trị thuộc tính A là a_i. Nếu A được chọn là thuộc tính kiểm định (tức là thuộc tính tốt nhất để phân chia), thì các tập con này sẽ tương đương với các nhánh tăng trưởng từ nút chứa tập S. Cho s_{ij} là số các mẫu của lớp C_i trong tập con S_j. Entropy hay thông tin cần để phân chia s mẫu vào trong v tập con là:

$$E(A) = \sum_{j=1}^{v} \frac{s_{1j} + ... + s_{mj}}{s} I(s_{1j}, ..., s_{mj}) \qquad (2.2)$$

Mã hoá thông tin sẽ có được bằng cách phân nhánh trên A là:

$$Gain(A) = I(s_1,s_2,...,s_m) - E(A) \qquad (2.3)$$

Giải thuật tính toán thông tin thu được của từng thuộc tính. Thuộc tính với thông tin thu được cao nhất được lựa chọn là thuộc tính kiểm định cho tập S. Tạo một nút với nhãn là thuộc tính đó, các nhánh được tạo cho mỗi giá trị của thuộc tính này và các mẫu được phân chia phù hợp.

Ví dụ 2.2: *Quy nạp của một cây quyết định*: Bảng 2.1 miêu tả một tập huấn luyện các bộ dữ liệu lấy từ cơ sở dữ liệu khách hàng *AllElectronics*. Thuộc tính nhãn lớp *mua máy tính* có hai giá trị riêng biệt là {*Có,Không*}, do vậy có hai nhãn riêng biệt (m=2). Cho C_1 tương đương với lớp *Có* và nhãn C_2 tương đương với *Không*. Có 9 mẫu của lớp *Có* và 5 mẫu của lớp *Không*. Để tính toán thông

tin thu được của từng thuộc tính, trước tiên ta sử dụng phương trình (2.1) để tính toán thông tin cần phân loại một mẫu cho trước:

$$I(s_1, s_2) = I(9,5) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940$$

Tiếp theo ta cần tính entropy của từng thuộc tính. Bắt đầu với thuộc tính *tuổi*. Ta cần xem sự phân bổ của các mẫu *có* và *không* cho mỗi giá trị của *tuổi*. Ta tính thông tin trông chờ cho mỗi phân bổ này:

For *tuổi*="<30": $s_{11} = 2$ $s_{21} = 3$ $I(s_{11},s_{21}) = 0.971$

For *tuổi*="30-40": $s_{12} = 4$ $s_{22} = 0$ $I(s_{12},s_{22}) = 0$

For *tuổi*=">40": $s_{13} = 3$ $s_{23} = 2$ $I(s_{13},s_{23}) = 0.971$

Bảng 2.1: Các bộ dữ liệu huấn luyện từ cơ sở dữ liệu khách hàng *AllElectronics*

STT	Tuổi	Thu nhập	Sinh viên	Độ tín nhiệm	Lớp: mua máy tính
1	<30	Cao	Không	Khá tốt	Không
2	<30	Cao	Không	Tốt	Không
3	30-40	Cao	Không	Khá tốt	Có
4	>40	Trung bình	Không	Khá tốt	Có
5	>40	Thấp	Có	Khá tốt	Có
6	>40	Thấp	Có	Tốt	Không
7	30-40	Thấp	Có	Tốt	Có
8	<30	Trung bình	Không	Khá tốt	Không
9	<30	Thấp	Có	Khá tốt	Có
10	>40	Trung bình	Có	Khá tốt	Có
11	<30	Trung bình	Có	Tốt	Có
12	30-40	Trung bình	Không	Tốt	Có
13	30-40	Cao	Có	Khá tốt	Có
14	>40	Trung bình	Không	Tốt	Không

Sử dụng phương trình (2.2), thông tin trông chờ cần phân loại một mẫu cho trước nếu các mẫu này được phân chia theo *tuổi* là:

$$E(Tuoi) = \frac{5}{14}I(s_{11},s_{21}) + \frac{4}{14}I(s_{12},s_{22}) + \frac{5}{14}I(s_{13},s_{23}) = 0.694$$

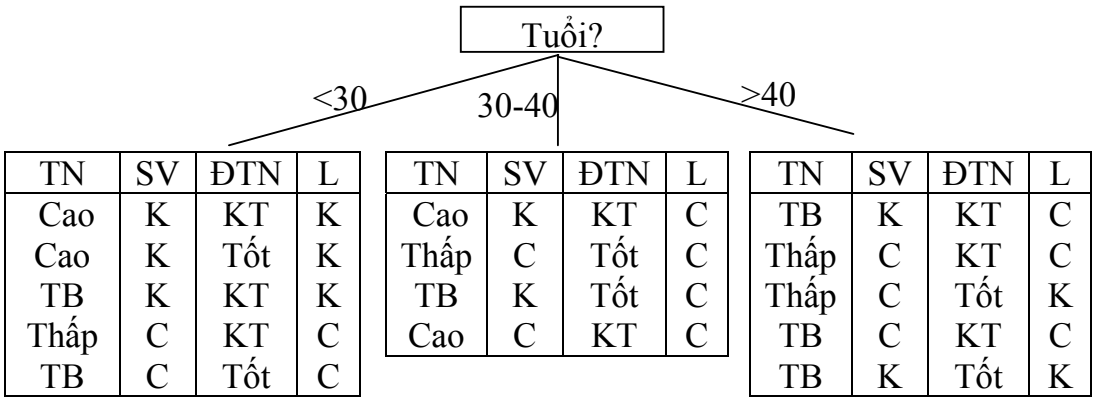
Do vậy thông tin thu được từ sự phân chia là:

$$Gain(tuổi) = I(s_1,s_2) - E(tuổi) = 0.246$$

Tương tự như vậy, ta có thể tính *Gain(thu nhập)* = 0.029, *Gain(sinh viên)* = 0.151, và *Gain(độ tín nhiệm)* = 0.048. Từ đó thuộc tính *tuổi* thu được thông

tin cao nhất, nó được chọn lựa là thuộc tính kiểm định. Một nút được tạo lập và gắn nhãn với *tuổi* và phân nhánh tăng trưởng đối với từng giá trị thuộc tính. Các mẫu sau đó được phân chia theo, như hình 2.4. Các mẫu rơi vào nhánh *tuổi* = 30-40 đều thuộc về lớp *Có*, do vậy một lá với nhãn *Có* được tạo lập tại đoạn cuối của nhánh này. Cây quyết định cuối cùng có được bởi thuật giải được thể hiện trong hình 2.2.

(Viết tắt trong hình 2.4: TN: Thu nhập; SV: Sinh viên; ĐTN: Độ tín nhiệm; TB: Trung bình; KT: Khá tốt; C: Có; K: Không; L:Lớp)



TN	SV	ĐTN	L
Cao	K	KT	K
Cao	K	Tốt	K
TB	K	KT	K
Thấp	C	KT	C
TB	C	Tốt	C

TN	SV	ĐTN	L
Cao	K	KT	C
Thấp	C	Tốt	C
TB	K	Tốt	C
Cao	C	KT	C

TN	SV	ĐTN	L
TB	K	KT	C
Thấp	C	KT	C
Thấp	C	Tốt	K
TB	C	KT	C
TB	K	Tốt	K

Hình 2.4: Thuộc tính *tuổi* có thông tin thu được cao nhất

Tuổi trở thành một thuộc tính kiểm định tại nút gốc của cây quyết định. Các nhánh được tăng trưởng theo từng giá trị của *tuổi*. Các mẫu được phân chia theo từng nhánh.

2.3.2 Cây cắt tỉa

Khi một cây quyết định được xây dựng, nhiều nhánh sẽ phản ánh sự bất bình thường trong dữ liệu huấn luyện bởi nhiễu hay các outlier. Các phương pháp cắt tỉa cây xử lý bài toán này. Các phương pháp này sử dụng các phép đo thống kê để gỡ bỏ tối thiểu các nhánh tin cậy, nhìn chung kết quả phân loại nhanh hơn, cải tiến khả năng phân loại phù hợp dữ liệu kiểm định độc lập.

Có hai tiếp cận phổ biến để cắt tỉa cây:

- Trong tiếp cận tiền cắt tỉa (*prepruning approach*), một cây được cắt tỉa bằng cách dừng sớm việc xây dựng nó (tức là bằng cách dừng hẳn sự phân chia hay sự phân chia tập con của các mẫu huấn luyện tại một nút cho trước). Như

vậy, nút sẽ trở thành một lá. Lá nắm giữ tần số lớp lớn nhất giữa các mẫu tập con.

Khi xây dựng một cây, các phép đo ví dụ như ý nghĩa thống kê χ^2, thông tin đạt được, v.v..., có thể được dùng để đánh giá chất lượng phân tách. Nếu phân chia các mẫu tại một nút cho kết quả phân tách dưới một ngưỡng chỉ định thì dừng việc phân chia tương lai của tập con cho trước. Có nhiều khó khăn trong việc lựa chọn một ngưỡng thích hợp.

- Tiếp cận hậu cắt tỉa (*postpruning*): gỡ bỏ các nhánh từ một cây "tăng trưởng đầy đủ". Một nút cây được tỉa bằng cách gỡ các nhánh của nó.

Tiền cắt tỉa cây và hậu cắt tỉa có thể được xen kẽ đối với một tiếp cận kết hợp. Hậu cắt tỉa yêu cầu tính toán nhiều hơn tiền cắt tỉa, nhìn chung sẽ dẫn tới một cây đáng tin cậy hơn.

2.3.3 Trích luật phân loại từ các cây quyết định

Tri thức trình bày trong các cây quyết định có thể được trích và trình bày dưới dạng các luật phân loại IF-THEN. Một luật tương ứng với một đường đi từ gốc tới một nút lá. Mỗi cặp *thuộc tính - giá trị* dọc theo đường đi tạo thành một liên kết trong tiền đề luật (phần "IF"). Nút lá là lớp dự đoán, thiết lập nên mệnh đề kết quả luật (phần "THEN"). Các luật IF-THEN giúp ta dễ hiểu hơn, đặc biệt nếu cây cho trước là rất lớn.

Ví dụ 2.3: *Sinh ra các luật phân loại từ một cây quyết định*: Cây quyết định như hình 2.2 có thể được chuyển đổi thành các luật phân loại "IF-THEN" bằng cách lần theo đường đi từ nút gốc tới từng nút lá trên cây.

Các luật trích được từ hình 2.2 là:

IF *tuổi* = "<30" AND *sinh viên = không* THEN *mua máy tính = không*

IF *tuổi* = "<30" AND *sinh viên = có* THEN *mua máy tính = có*

IF *tuổi* = "30-40" THEN *mua máy tính = có*

IF *tuổi* = ">40" AND *độ tín nhiệm = tốt* THEN *mua máy tính = có*

IF *tuổi* = ">40" AND *độ tín nhiệm = khá tốt* THEN *mua máy tính = không*

Một luật có thể được tỉa bớt bằng cách gỡ bỏ một số điều kiện trong tiền đề luật mà không làm ảnh hưởng lắm đến độ chính xác của luật. Đối với mỗi lớp, các luật trong phạm vi một lớp có thể được sắp xếp theo độ chính xác của chúng. Do đó rất dễ xảy ra hiện tượng là một mẫu kiểm định sẽ không thoả bất kỳ một tiền đề luật nào, một luật ngầm định ấn định lớp đa số (majority class) được thêm vào kết quả tập luật.

2.3.4 Cải tiến cây quyết định quy nạp cơ bản

Giải thuật cây quyết định quy nạp cơ bản ở mục 2.3.1 đòi hỏi tất các các thuộc tính là xác thực (categorical) hay rời rạc (discretized). Giải thuật có thể sửa đổi để cho phép các thuộc tính có giá trị liên tục. Kiểm định trên một thuộc tính A có giá trị liên tục cho kết quả vào hai nhánh, tương đương với hai điều kiện $A \leq V$ và $A > V$ cho các giá trị số (numeric) V của A. Nếu A có v giá trị thì có thể có v-1 phép phân tách được xem xét khi xác định V. Thông thường các điểm giữa mỗi cặp giá trị kề nhau được xem xét. Nếu các giá trị được sắp xếp trước thì chỉ cần một lần duyệt qua các giá trị.

Giải thuật cây quyết định quy nạp cơ bản tạo một nhánh cho mỗi giá trị của một thuộc tính kiểm định, sau đó phân phối các mẫu một cách phù hợp. Phân chia này có thể cho kết quả là một số lượng lớn các tập con nhỏ. Khi đó các tập con trở nên ngày càng nhỏ đi, xử lý phân chia có thể sử dụng mẫu có quy mô là thống kê không đầy đủ. Lúc này, việc tìm mẫu hữu ích trong các tập con sẽ trở nên không thích hợp bởi tính không đầy đủ của dữ liệu. Một cách khắc phục là nhóm các giá trị có thuộc tính xác thực hoặc tạo các cây quyết định nhị phân, tại đó mỗi nhánh là một kiểm định boolean trên một thuộc tính. Các cây nhị phân cho kết quả phân mảnh dữ liệu ít nhất. Nhiều nghiên cứu đã cho thấy các cây quyết định nhị phân có khuynh hướng chính xác hơn các cây truyền thống.

Nhiều phương pháp được đề xuất để xử lý các giá trị thuộc tính khuyết. Một giá trị bị khuyết của thuộc tính A có thể được thay thế bởi giá trị phổ biến nhất của A.

2.3.5 Khả năng mở rộng và cây quyết định quy nạp

Các giải thuật cây quyết định như ID3 và C4.5 được thiết lập cho các tập dữ liệu tương đối nhỏ. Hiệu quả và khả năng mở rộng là các vấn đề liên quan với nhau khi các giải thuật này được áp dụng vào việc khai phá các cơ sở dữ liệu rất lớn, thế giới thực. Hầu hết các giải thuật quyết định đều có hạn chế là các mẫu huấn luyện tập trung ở bộ nhớ chính. Trong các ứng dụng khai phá dữ liệu, các tập huấn luyện rất lớn của hàng triệu mẫu là phổ biến. Do vậy, hạn chế này giới hạn khả năng mở rộng của các giải thuật trên, tại đây cấu trúc cây quyết định có thể trở nên vô ích bởi việc trao đổi của các mẫu huấn luyện trong và ngoài các bộ nhớ chính và cache.

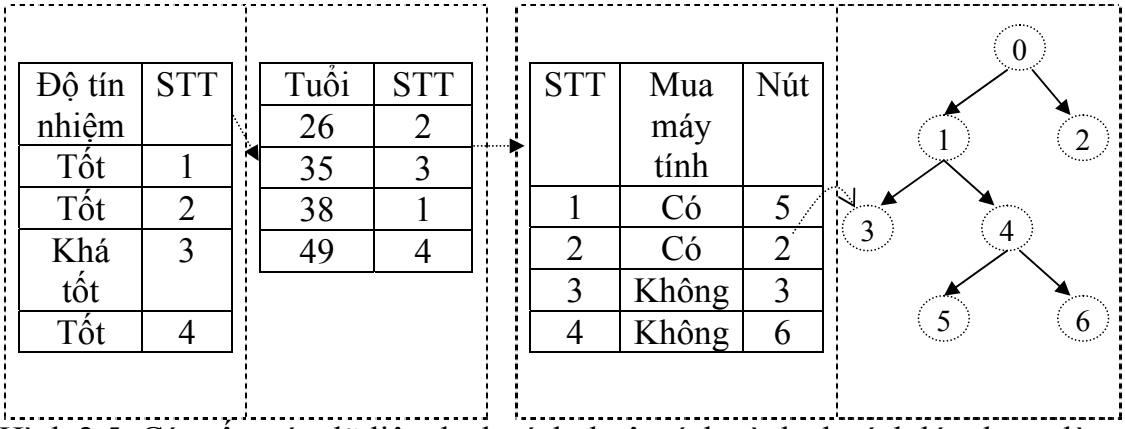
Lúc đầu, chiến lược cho cây quyết định quy nạp ở các cơ sở dữ liệu lớn có thể là rời rạc hoá các thuộc tính liên tục, giả định tập huấn luyện vừa đủ trong bộ nhớ. Để mở rộng, trước tiên phân chia dữ liệu vào trong các tập con một cách riêng biệt có thể vừa vào trong bộ nhớ và sau đó xây dựng một cây quyết định từ mỗi tập con. Classifier đầu ra cuối cùng là sự kết hợp của các classifier có được từ các tập con. Mặc dù phương pháp này cho phép phân loại các tập dữ liệu lớn, độ chính xác phân loại của nó không cao như chỉ có một classifier - nó được xây dựng bằng cách sử dụng tất cả dữ liệu cùng một lúc.

Một trong số các giải thuật cây quyết định gần đây được đề xuất để xử lý vấn đề khả năng mở rộng là SLIQ, nó có thể vận dụng các thuộc tính có giá trị xác thực và liên tục. Cả hai giải thuật đề xuất các kỹ thuật tiền sắp xếp trên đĩa - các tập dữ liệu thường trú là quá lớn để vừa trong bộ nhớ. Cả hai đều định nghĩa ích lợi của các cấu trúc dữ liệu mới giúp cho việc xây dựng cây trở nên thuận lợi. SLIQ dùng đĩa để lưu các danh sách thuộc tính và một bộ nhớ đơn lẻ để lưu danh sách lớp. Các danh sách thuộc tính và các danh sách lớp được sinh ra bởi SLIQ đối với dữ liệu mẫu ở bảng 2.2 được chỉ ra trên hình 2.5. Mỗi thuộc tính có một danh sách thuộc tính kết hợp, được đánh chỉ số bởi STT. Mỗi bộ được biểu diễn bởi liên kết của một mục (entry) từ mỗi danh sách thuộc tính sang một mục trong danh sách lớp, nó lần lượt được liên kết tới nút lá tương ứng trong cây quyết định. Danh sách lớp vẫn ở trong bộ nhớ vì nó thường được truy cập,

sửa đổi trong các pha xây dựng và cắt tỉa. Kích thước của danh sách lớp tăng trưởng cân xứng với số lượng các bộ trong tập huấn luyện. Khi một danh sách lớp không thể vừa vào trong bộ nhớ, việc biểu diễn của SLIQ suy giảm.

Bảng 2.2: Dữ liệu mẫu cho lớp *mua máy tính*

STT	Độ tín nhiệm	Tuổi	Mua máy tính
1	Tốt	38	Có
2	Tốt	26	Có
3	Khá tốt	35	Không
4	Tốt	49	Không



Hình 2.5: Các cấu trúc dữ liệu danh sách thuộc tính và danh sách lớp được dùng trong SLIQ cho dữ liệu mẫu trong bảng 2.2

2.4 Phân loại Bayesian

Classifier Bayesian là classifier thống kê. Phân loại Bayesian dựa trên định lý Bayes. Một classifier đơn giản của Bayesian đó là *Naive* Bayesian, so với việc thực thi của classifier cây quyết định và mạng nơron, classifier Bayesian đưa ra độ chính xác cao và nhanh khi áp dụng vào các cơ sở dữ liệu lớn.

Các classifier *Naive Bayesian* giả định rằng hiệu quả của một giá trị thuộc tính trên một lớp là độc lập so với giá trị của các thuộc tính khác. Giả định này được gọi là độc lập có điều kiện lớp. Như vậy sẽ đơn giản hoá các tính toán rắc rối, vì thế coi nó là "*naive-ngây thơ*". Các mạng *belief (dựa trên) Bayesian* là các mô hình đồ thị, nó không giống như classifier Bayesian ngây thơ, cho phép biểu diễn sự phụ thuộc giữa các tập con của các thuộc tính. Các mạng belief Bayesian cũng được dùng cho phân loại.

Mục 2.4.1 nói lại các khái niệm xác suất cơ bản và định lý Bayes. Sau đó ta sẽ xem phân loại Bayesian ngây thơ trong 2.4.2, các mạng belief Bayes được mô tả trong mục 2.4.3.

2.4.1 Định lý Bayes

Cho X là mẫu dữ liệu chưa biết nhãn lớp, H là giả thuyết ví dụ như mẫu dữ liệu X thuộc về lớp C. Đối với các bài toán phân loại, ta cần xác định $P(H|X)$ là xác suất xảy ra giả thuyết H trên mẫu dữ liệu X.

$P(H|X)$ là xác suất hậu nghiệm của H với điều kiện X. Ví dụ, giả sử các mẫu dữ liệu trong tập *hoa quả* được mô tả bởi *màu sắc* và *hình dạng* của chúng. Giả sử X là *đỏ* và *tròn*, H là giả thuyết X là *quả táo*. Thì $P(H|X)$ phản ánh độ tin cậy rằng X là một *quả táo* với việc đã nhìn thấy X là *đỏ* và *tròn*. Ngược lại, $P(H)$ là xác suất tiên nghiệm của H. Như ví dụ, đây là xác suất một mẫu dữ liệu bất kì cho trước là *quả táo* bất kể nó trông như thế nào. Xác suất hậu nghiệm $P(H|X)$ dựa trên nhiều thông tin (như nền tảng tri thức) hơn xác suất tiên nghiệm $P(H)$, nó độc lập với X.

Tương tự như vậy, $P(X|H)$ là xác suất hậu nghiệm của X với điều kiện H. Đó là xác suất để X là *đỏ* và *tròn*, ta đã biết sự thật là X là một *quả táo*. $P(X)$ là tiên nghiệm của X. Theo ví dụ trên, nó là xác suất để cho một mẫu dữ liệu từ tập *hoa quả* là *đỏ* và *tròn*.

$P(X)$, $P(H)$, $P(X|H)$ được đánh giá từ dữ liệu cho trước. Định lý Bayes thực sự có ích bởi nó cung cấp cách thức tính toán xác suất hậu nghiệm $P(H|X)$ từ $P(X)$, $P(H)$ và $P(X|H)$. Định lý Bayes như sau:

$$P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)} \qquad (2.4)$$

Trong mục tiếp theo ta sẽ xem định lý Bayes được dùng như thế nào trong classifier Bayesian ngây thơ.

2.4.2 Phân loại Bayesian ngây thơ

Classifier Bayesian ngây thơ hay classifier Bayessian đơn giản làm việc như sau:

1. Mỗi mẫu dữ liệu được đại diện bởi một vector đặc trưng n-chiều, $X=(x_1,x_2,...,x_n)$, mô tả n phép đo có được trên mẫu từ n thuộc tính tương ứng A_1, A_2,..., A_n.

2. Giả sử rằng có m lớp $C_1,C_2,...C_m$. Cho trước một mẫu dữ liệu chưa biết nhãn lớp X, classifier sẽ dự đoán X thuộc về lớp có xác suất hậu nghiệm cao nhất, với điều kiện trên X. Classifier Bayesian ngây thơ ấn định một mẫu không biết X vào một lớp C_i khi và chỉ khi:

$$P(C_i|X) > P(C_j|X) \text{ với } 1 \le j \le m, j \ne i$$

Do vậy cần tìm P($C_i|X$) lớn nhất. Theo định lý Bayes (Phương trình 2.4):

$$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)} \tag{2.5}$$

3. $P(X)$ không đổi với mọi lớp, $P(C_i)=s_i/s$ (s_i là số lượng các mẫu huấn luyện của lớp C_i và s là tổng số các mẫu huấn luyện), $P(X|C_i)P(C_i)$ cần được cực đại.

4. Cho trước các tập dữ liệu với nhiều thuộc tính, việc tính $P(X|C_i)$ sẽ rất tốn kém. Để giảm tính toán khi đánh giá $P(X|C_i)$, giả định *ngây thơ* của độc lập có điều kiện lớp được thiết lập. Điều này làm cho giá trị của các thuộc tính là độc lập có điều kiện với nhau, cho trước nhãn lớp của mẫu, tức là không có mối quan hệ độc lập giữa các thuộc tính. Vì thế,

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i) \tag{2.6}$$

$P(x_1|C_i)$, $P(x_2|C_i)$,..., $P(x_n|C_i)$ được đánh giá từ các mẫu huấn luyện với:

(a) Nếu A_k là xác thực thì $P(x_k|C_i)=s_{ik}/s_i$ với s_{ik} là số lượng các mẫu huấn luyện của lớp C_i có giá trị x_k tại A_k và s_i là số lượng các mẫu huấn luyện thuộc về C_i.

(b) Nếu A_k là giá trị liên tục thì thuộc tính được giả định có phân phối Gaussian. Bởi vậy,

$$P(x_k \mid C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}) = \frac{1}{\sqrt{2\pi}\sigma_{C_i}} e^{-\frac{(x-\mu_{C_i})^2}{2\sigma_{C_i}^2}} \tag{2.7}$$

với $g(x_k, \mu_{Ci}, \sigma_{Ci})$ là hàm mật độ (thông thường) Gaussian của thuộc tính A_k, với μ_{Ci}, σ_{Ci} đại diện cho các giá trị trung bình và độ lệch chuẩn của thuộc tính A_k đối với các mẫu huấn luyện của lớp C_i.

5. Để phân loại một mẫu chưa biết X, với $P(X|C_i)P(C_i)$ được đánh giá cho lớp C_i. Mẫu X được ấn định vào lớp C_i khi và chỉ khi:

$$P(X|C_i)P(C_i) > P(X|C_j)P(C_j) \text{ với } 1 \le j \le m, j \ne i$$

Hay nói cách khác, nó được ấn định tới lớp C_i mà tại đó $P(X|C_i)P(C_i)$ cực đại.

Ví dụ 2.4: *Dự đoán một nhãn lớp sử dụng phân loại Bayesian ngây thơ:* Ta cần dự đoán nhãn lớp của một mẫu chưa biết sử dụng phân loại Bayesian ngây thơ, với cùng dữ liệu huấn luyện đã có trong ví dụ 2.2 cho cây quyết định quy nạp. Dữ liệu huấn luyện trong bảng 2.1. Các mẫu dữ liệu được mô tả bởi các thuộc tính *tuổi, thu nhập, sinh viên* và *độ tín nhiệm*. Thuộc tính nhãn lớp *mua máy tính* có hai giá trị riêng biệt (tên là {*có* và *không*}). Cho C_1 tương đương với lớp *mua máy tính = có* và C_2 tương đương với lớp *mua máy tính = không*. Mẫu chưa biết ta sẽ phân loại chúng là:

X = (*tuổi* = "<30", *thu nhập=trung bình, sinh viên= có, độ tín nhiệm=khá tốt*)

Ta cần cực đại hoá $P(X|C_i)P(C_i)$ với i=1,2. $P(C_i)$ là xác suất tiên nghiệm của mỗi lớp có thể được tính toán dựa trên các mẫu huấn luyện:

P(*mua máy tính = có*) = 9/14 = 0.643

P(*mua máy tính = không*) = 5/14 = 0.357

Để tính P($X|C_i$) với i=1,2, ta tính các xác suất có điều kiện sau:

P(*tuổi = "<30" | mua máy tính = có*) = 2/9 = 0.222

P(*tuổi = "<30" | mua máy tính = không*) = 3/5 = 0.600

P(*thu nhập = trung bình | mua máy tính = có*) = 4/9 = 0.444

P(*thu nhập = trung bình | mua máy tính = không*) = 2/5 = 0.400

P(*sinh viên = có | mua máy tính = có*) = 6/9 = 0.667

P(*sinh viên = có | mua máy tính = không*) = 1/5 = 0.200

P(*độ tín nhiệm = khá tốt | mua máy tính = có*) = 6/9 = 0.667

P(*độ tín nhiệm = khá tốt | mua máy tính = không*) = 2/5 = 0.400

Sử dụng các xác suất ở trên ta có:

P(*X*|*mua máy tính = có*) = 0.222 x 0.444 x 0.667 x 0.667 = 0.044

P(*X*|*mua máy tính = không*) = 0.600 x 0.400 x 0.200 x 0.400 = 0.019

P(*Xjmua máy tính = có*)*P*(*mua máy tính = có*) = 0.044x 0.643 = 0.028

P(*Xjmua máy tính = không*)*P*(*mua máy tính = không*) = 0.019 x 0.357 = 0.007

Bởi vậy, classifier Bayesian ngây thơ dự đoán "*mua máy tính = có"* cho mẫu *X*.

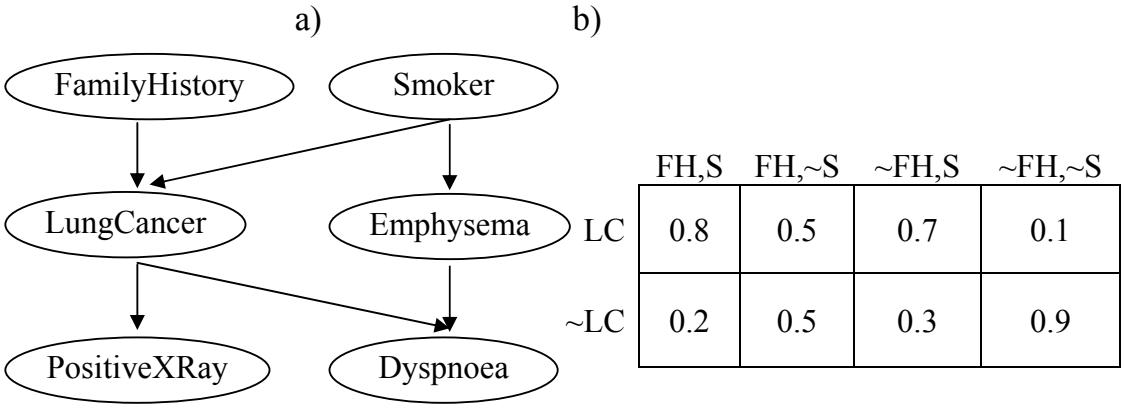
2.4.3 Các mạng belief Bayesian

Classifier Bayesian ngây thơ thực hiện trên giả định độc lập điều kiện lớp, tức là nhãn lớp của một mẫu là cho trước, giá trị của các thuộc tính độc lập có điều kiện với nhau. Giả định này làm đơn giản hoá việc tính toán. Khi giả định là đúng thì classifier Bayesian ngây thơ có độ chính xác cao nhất so với tất cả các classifier khác. Tuy nhiên trong thực tiễn, sự phụ thuộc có thể tồn tại giữa các biến. Các mạng belief Bayes định rõ phần chung các phân bố xác suất có điều kiện. Chúng cung cấp một mô hình đồ thị các mối quan hệ nhân quả, trên đó việc học được thực hiện.

Một mạng belief được định nghĩa bởi hai thành phần. Thứ nhất là một đồ thị không có chu trình và có hướng, tại đó mỗi nút đại diện cho một biến ngẫu nhiên và mỗi cung đại diện cho một phụ thuộc xác suất. Nếu một cung được vẽ từ một nút *Y* tới một nút *Z* thì *Y* là cha của *Z* hay tổ tiên gần nhất của *Z* và *Z* là con cháu của *Y*. Mỗi biến là độc lập có điều kiện với những nút không phải con cháu của nó trên đồ thị, cho trước các cha của chúng. Giá trị của các biến này có thể là rời rạc hay liên tục.

Ta có thể gọi chúng là *các mạng belief*, *các mạng Bayesian* hay *các mạng xác suất*. Một cách ngắn gọn, ta sẽ xem chúng như là các mạng belief.

(*FamilyHistory: tiền sử gia đình; LungCancer: ung thư phổi; Smoker: người hút thuốc; PositiveXRay: phim X quang; Emphysema: khí thũng; Dyspnoea: khó thở*)



Hình 2.6: a) Mạng belief Bayesian đơn giản, b) Bảng xác suất có điều kiện cho các giá trị của biến *LungCancer* (*LC*)

Hình 2.6a) cho thấy một mạng belief đơn giản lấy từ [Russell et al. 1995a] cho 6 biến Boolean. Các cung cho phép một biểu diễn tri thức nhân quả. Ví dụ, bệnh phổi một người bị ảnh hưởng bởi lịch sử bệnh phổi của gia đình anh ta, cũng như liệu người đó có nghiện thuốc lá hay không. Hơn nữa, các cung cũng chỉ ra rằng các biến *LungCancer* là độc lập có điều kiện với *Emphysema*, cho trước các cha của nó: *FamilyHistory* và *Smoker*. Điều này có nghĩa là một khi các giá trị của *FamilyHistory* và *Smoker* được biết thì biến *Emphysema* không cần cung cấp thêm bất kỳ một thông tin nào để đánh giá *LungCancer*.

Thành phần thứ hai định nghĩa mạng belief là một bảng xác suất có điều kiện (viết tắt: CPT - *conditional probability table*) cho mỗi biến. CPT cho một biến Z chỉ ra phân phối có điều kiện $P(Z|Parents(Z))$ với $Parents(Z)$ là các cha của Z. Hình 2.6b) cho thấy một CPT cho *LungCancer*. Xác suất có điều kiện cho mỗi giá trị của *LungCancer* cho trước đối với mỗi kết nối có thể có của các giá trị các cha của nó. Ví dụ, từ các mục phía trên trái nhất và phía dưới phải nhất tương ứng như sau:

$P(LungCancer = Có \mid FamilyHistory = Có, Smoker = Có) = 0.8$, và

$P(LungCancer = Không \mid FamilyHistory = Không, Smoker = Không) = 0.9$.

Xác suất chung của bất kỳ một bộ $(z_1,z_2,...,z_n)$ tương đương với các biến hay các thuộc tính $Z_1,Z_2,...,Z_n$ được tính toán bởi :

$$P(z_1,...,z_n) = \prod_{i=1}^{n} P(z_i \mid Parents(Z_i)) \qquad (2.8)$$

$P(z_i|Parents(Z_i))$ tương đương với các mục trong CPT cho Z_i.

Một nút trên mạng có thể được chọn như là nút "đầu ra", biểu diễn một thuộc tính nhãn lớp. Có thể có nhiều hơn một nút đầu ra. Các giải thuật suy diễn cho việc học cũng áp dụng được trên mạng này. Xử lý phân loại, có thể trả lại một nhãn lớp đơn lẻ, hay một phân phối xác suất cho thuộc tính nhãn lớp, tức là dự đoán xác suất của mỗi lớp.

2.4.4 Huấn luyện các mạng belief Bayesian

Trong việc học hay huấn luyện một mạng belief cấu trúc mạng có trước hay được suy diễn từ dữ liệu. Các biến mạng có thể quan sát được hay ẩn ở tất cả hoặc một số mẫu huấn luyện. Dữ liệu ẩn được xem là giá trị khuyết hay dữ liệu chưa đầy đủ.

Nếu cấu trúc mạng đã được biết và các biến là quan sát được thì việc học mạng là không phức tạp, chỉ cần tính các mục CPT, như đã làm với Bayesian ngây thơ.

Với cấu trúc mạng cho trước, một số bị biến ẩn thì dùng phương pháp gradient descent để huấn luyện mạng belief. Đối tượng này để học các giá trị cho các mục CPT. S là tập có s mẫu huấn luyện $X_1,X_2,...,X_s$. w_{ijk} là một mục CPT cho biến $Y_i=y_{ij}$ có các cha $U_i=u_{ik}$. Ví dụ, nếu w_{ijk} là mục CPT phía trên trái nhất của hình 2.6b) thì Y_i = *LungCancer;* giá trị của nó y_{ij} = *Có*; danh sách các nút cha của Y_i là U_i = {*FamilyHistory, Smoker*}; và danh sách giá trị của các nút cha u_{ik} = {*Có, Có*}. w_{ijk} được xem như là các trọng số, giống như các trọng số trong các unit ẩn của các mạng nơron (mục 2.5). Các trọng số, w_{ijk} ban đầu là các giá trị xác suất ngẫu nhiên. Chiến lược gradient descent biểu diễn leo đồi (hill-

climbing) tham. Tại mỗi lần lặp, các trọng số được cập nhật và cuối cùng sẽ hội tụ về một giải pháp tối ưu cục bộ.

Phương pháp nhằm mục đích cực đại hoá $P(S|H)$. Cho trước cấu trúc mạng và w_{ijk} khởi đầu, giải thuật xử lý như sau:

1. Tính các gradient: Cho *i, j, k* tính:

$$\frac{\partial \ln P(S \mid H)}{\partial w_{ijk}} = \sum_{d=1}^{s} \frac{P(Y_i = y_{ij}, U_i = u_{ik} \mid X_d)}{w_{ijk}} \qquad (2.9)$$

Xác suất bên phải của phương trình (2.9) được tính cho mỗi mẫu huấn luyện X_d trong S, xem nó là xác suất đơn giản p. Khi các biến được miêu tả bởi Y_i và U_i là ẩn đối với một vài X_d nào đó thì xác suất tương ứng p có thể được tính từ các biến quan sát được của mẫu sử dụng các giải thuật chuẩn cho suy diễn mạng Bayesian.

2. Lấy một bước nhỏ theo hướng của gradient: Các trọng số được cập nhật bởi

$$w_{ijk} \leftarrow w_{ijk} + (l) \frac{\partial \ln P(S \mid H)}{\partial w_{ijk}} \qquad (2.10)$$

với l là tỷ số học biểu diễn kích thước bước và $\frac{\partial \ln P(S \mid H)}{\partial w_{ijk}}$ được tính từ phương trình (2.9). Tỷ số học là một hằng số nhỏ.

3. Chuẩn hóa lại các trọng số: Vì các trọng số w_{ijk} là các giá trị xác suất, chúng phải giữa 0 và 1.0 và $\sum_j w_{ijk}$ phải bằng 1 với mọi *i, k*. Những tiêu chuẩn này có được bằng cách chuẩn hoá lại các trọng số sau khi chúng được cập nhật bởi phương trình (2.10).

2.5 Phân loại bằng lan truyền ngược

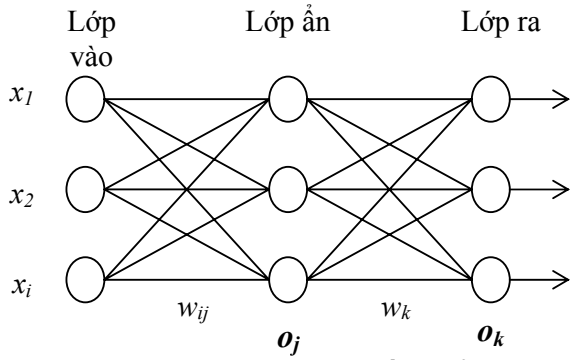
Lan truyền ngược là một giải thuật học mạng nơron. Nói một cách thô sơ, một mạng nơron là một tập các unit vào/ra có kết nối, tại đó, mỗi kết nối có một trọng số kết hợp với nó. Trong suốt pha học, mạng học bằng cách điều chỉnh các trọng số để có thể dự đoán nhãn lớp của các mẫu đầu vào một cách chính xác.

Các mạng nơron cần thời gian huấn luyện dài, do vậy các ứng dụng phù hợp thì sẽ khả thi hơn. Chúng yêu cầu một số lượng các tham số mà theo kinh nghiệm nó được xác định tốt nhất như cấu trúc liên kết mạng hay "cấu trúc" mạng. Khả năng diễn dịch của các mạng nơron nghèo nàn, do vậy việc hiểu được ý nghĩa biểu tượng đằng sau các trọng số được học là rất khó. Các đặc trưng này lúc đầu làm cho nhu cầu khai phá dữ liệu dùng mạng nơron ít đi.

Thuận lợi của các mạng nơron đó là độ cao dung sai của chúng đối với dữ liệu nhiễu cũng như khả năng phân loại các mẫu không được huấn luyện. Một số giải thuật gần đây được phát triển để trích lọc các luật từ các mạng nơron huấn luyện. Các yếu tố này góp phần làm cho các mạng nơron trở nên hữu ích hơn khi phân loại trong khai phá dữ liệu.

Giải thuật mạng nơron phổ biến nhất đó là giải thuật lan truyền ngược, được đề xuất năm những năm 1980. Mục 2.5.1 là các mạng truyền thẳng đa mức, đây là một kiểu mạng nơron biểu diễn bằng giải thuật lan truyền ngược. Mục 2.5.2 định nghĩa một cấu trúc liên kết mạng. Giải thuật lan truyền ngược được mô tả trong mục 2.5.3. Rút trích luật từ các mạng nơron huấn luyện trong mục 2.5.4.

2.5.1 Một mạng nơron truyền thẳng đa mức



Hình 2.7: Một mạng nơron truyền thẳng đa mức

Giải thuật lan truyền ngược biểu diễn việc học trên một mạng nơron truyền thẳng đa mức. Như thí dụ trên hình 2.7, các đầu vào tương ứng với các thuộc tính đo được đối với mỗi mẫu huấn luyện, cung cấp đồng thời vào một lớp các unit tạo thành lớp đầu vào. Đầu ra có trọng số của các unit này sau đó cung cấp

đồng thời tới lớp các unit thứ hai, ta gọi đó là lớp ẩn. Các đầu ra có trọng số của lớp ẩn có thể là đầu vào cho một lớp ẩn khác, v.v... Số lượng các lớp ẩn là tuỳ ý, mặc dầu trong thực tiễn thường xuyên chỉ có một lớp được sử dụng. Các đầu ra có trọng số của lớp ẩn cuối cùng là đầu vào cho các unit tạo nên lớp đầu ra, nó đưa ra dự đoán của mạng cho các mẫu cho trước.

Các unit trong các lớp ẩn và lớp đầu ra được coi là các unit đầu ra. Mạng nơron đa mức như biểu diễn trong hình 2.7 có 2 lớp unit đầu ra. Bởi vậy, ta nói rằng nó là một mạng nơron 2 lớp. Tương tự, một mạng chứa 2 lớp ẩn được gọi là một mạng nơron 3 lớp, v.v... Mạng được coi là truyền thẳng nếu như nó không có một trọng số nào quay lại một unit đầu vào hay tới một unit đầu ra của một lớp trước nó. Mạng được gọi là kết nối đầy đủ khi mà trong mạng, mỗi một unit cung cấp đầu vào cho từng unit ở lớp tiếp theo.

Với các unit ẩn đầy đủ cho trước, các mạng truyền thẳng đa mức của các hàm ngưỡng tuyến tính có thể xấp xỉ tới bất kỳ một hàm nào.

2.5.2 Định nghĩa cấu trúc liên kết mạng

"*Ta có thể thiết kế cấu trúc liên kết của một mạng nơron như thế nào?*"

- Trước khi huấn luyện bắt đầu, người dùng phải quyết định cấu trúc liên kết mạng bằng cách chỉ ra số lượng các unit trong lớp đầu vào, số lượng các lớp ẩn (nếu nhiều hơn 1), số lượng các unit trong mỗi lớp ẩn và số lượng các unit trong lớp đầu ra.

- Chuẩn hoá các giá trị đầu ra cho mỗi thuộc tính đã đo trong các mẫu huấn luyện sẽ giúp tăng tốc pha học. Các giá trị đầu vào được chuẩn hóa để nằm trong khoảng [0,1]. Các thuộc tính có giá trị rời rạc có thể được mã hoá để một unit đầu vào tương ứng với một giá trị miền. Ví dụ, nếu miền của một thuộc tính A là $\{a_0,a_1,a_2\}$ thì ta có thể ấn định 3 unit đầu vào cho A. Ta có I_0, I_1, I_2 là các unit đầu vào. Mỗi unit có giá trị ban đầu là 0. Nếu $A=a_0$ thì I_0 được đặt là 1, nếu $A=a_1$ thì I_1 được đặt là 1, v.v... Một unit đầu ra có thể được dùng để biểu diễn hai lớp (1 đại diện cho một lớp, 0 đại diện cho lớp khác). Nếu có nhiều hơn hai lớp thì unit đầu ra 1 tương ứng với lớp được sử dụng.

Không có luật rõ ràng về số lượng tốt nhất các unit lớp ẩn. Thử mạng thiết kế bằng phương pháp sai số, mạng ảnh hưởng đến độ chính xác kết quả mạng huấn luyện. Giá trị đầu tiên của các trọng số tác động tới kết quả độ chính xác. Khi một mạng được huấn luyện, độ chính xác không chấp nhận được thì xử lý huấn luyện thường lặp lại với một cấu trúc liên kết mạng khác hay một tập các trọng số khởi đầu khác.

2.5.3 Lan truyền ngược

Lan truyền ngược học bằng cách lặp đi lặp lại việc xử lý một tập các mẫu huấn luyện, so sánh dự đoán của mạng cho mỗi mẫu với nhãn lớp thực sự đã biết. Đối với mỗi mẫu huấn luyện, các trọng số được sửa đổi để cực tiểu hoá trung bình của bình phương sai số giữa dự đoán của mạng và lớp thực tế. Các sửa đổi này được làm theo hướng "ngược lại", tức là từ lớp đầu ra, xuyên qua mỗi lớp ẩn xuống tới lớp ẩn đầu tiên (do đó có tên là *lan truyền ngược*). Mặc dầu điều này không đảm bảo lắm, nhìn chung các trọng số cuối cùng sẽ hội tụ và xử lý việc học sẽ dừng. Giải thuật được tóm tắt trong hình 2.8.

Giải thuật 2.5.1 (Lan truyền ngược): Học mạng nơron để phân loại, sử dụng giải thuật lan truyền ngược.

Đầu vào: Các mẫu huấn luyện *samples*; tốc độ học *l*; một mạng truyền thẳng đa mức *network*.

Đầu ra: Một mạng nơron đã huấn luyện để phân loại các mẫu.

Giải thuật:

1) Khởi tạo giá trị ban đầu cho các trọng số và các bias trong *network*;
2) **while** điều kiện dừng chưa thỏa {
3) **for** mỗi mẫu huấn luyện *X* trong *samples* {
4) //Truyền đầu vào theo hướng tiến về phía trước
5) **for** mỗi unit *j* ở lớp ẩn hay lớp đầu ra
6) $I_j = \sum_i w_{ij} O_i + \theta_j$;//Tính đầu vào mạng của unit *j*
7) **for** mỗi unit *j* ở lớp ẩn hay lớp đầu ra

---

8) $O_j = \frac{1}{1+e^{-I_j}}$; //Tính đầu ra cho mỗi unit *j*

9) //Lan truyền ngược các sai số

10) **for** mỗi unit *j* ở lớp đầu ra

11) $Err_j = O_j(1 - O_j)(T_j - O_j)$; //Tính sai số

12) **for** mỗi unit *j* ở các lớp ẩn

13) $Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$; //Tính sai số

14) **for** mỗi trọng số w_{ij} trong *network*{

15) $\Delta w_{ij} = (l) Err_j O_i$; //Trọng số tăng dần

16) $w_{ij} = w_{ij} + \Delta w_{ij}$; } //Cập nhật trọng số

17) **for** mỗi bias θ_j trong *network*{

18) $\Delta\theta_j = (l)\ Err_j$; //bias tăng dần

19) $\theta_j = \theta_j + \Delta\theta_j$; } //Cập nhật bias

20) }}

Hình 2.8: Giải thuật lan truyền ngược

**** Giá trị khởi đầu các trọng số:*** Các trọng số trong mạng được thiết lập giá trị ban đầu là các số ngẫu nhiên nhỏ (phạm vi từ -1.0 tới 1.0 hay -0.5 tới 0.5). Mỗi nút kết hợp với một *bias*, các *bias* được gán giá trị ban đầu như nhau, đó là các số ngẫu nhiên nhỏ.

**** Mỗi mẫu huấn luyện X được xử lý theo các bước sau:***

- *Truyền các đầu vào về phía trước:* Ở bước này, mạng đầu vào và đầu ra của mỗi unit trong các lớp ẩn và lớp đầu ra được tính toán.

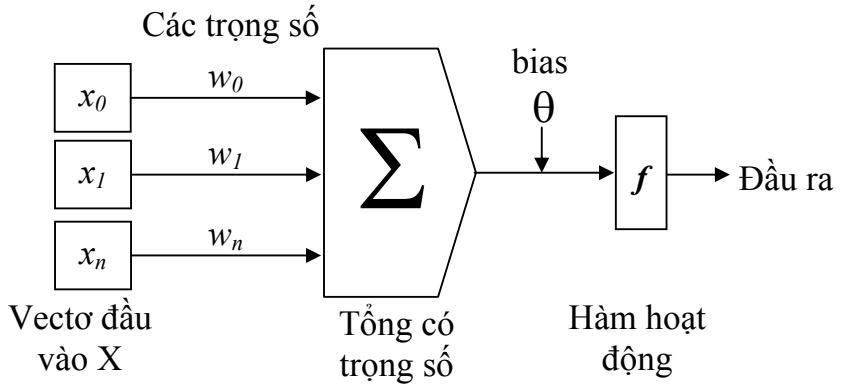
+ Cung cấp mẫu huấn luyện cho lớp đầu vào.

+ Tính mạng đầu vào cho mỗi unit ở các lớp ẩn và lớp đầu ra, đó là sự kết hợp tuyến tính các đầu vào của nó. Để minh họa điều này, một unit lớp ẩn hay lớp đầu ra được biểu diễn trên hình 2.9. Các đầu vào của unit trong thực tế là các đầu ra của các unit kết nối tới nó ở lớp trước. Để tính mạng đầu vào của unit, mỗi đầu vào được nhân với trọng số tương ứng sau đó

cộng lại. Cho trước một unit *j* thuộc lớp ẩn hay lớp đầu ra, mạng đầu vào I_j tới unit *j* là:

$$I_j = \sum_i w_{ij} O_i + \theta_j \tag{2.11}$$

với w_{ij} là trọng số kết nối từ unit *i* ở lớp trước tới unit *j*; O_i là đầu ra của unit *i*; θ_j là bias của unit *j*. Bias đóng vai trò là ngưỡng phục vụ cho mức hoạt động khác nhau của unit.



Hình 2.9: Một unit lớp ẩn hay lớp đầu ra

Ở hình 2.9, các đầu vào được nhân với các trọng số tương ứng để hình thành một tổng trọng số cộng với bias đã kết hợp với unit. Một hàm hoạt động không tuyến tính được gắn vào mạng đầu vào.

Mỗi unit trong lớp ẩn và lớp đầu ra có một mạng đầu vào, sau đó gắn vào một hàm hoạt động như minh hoạ ở hình 2.9. Hàm tượng trưng cho hoạt động của nơron được đại diện bởi unit, ví dụ như logistic hay simoid. Cho trước mạng đầu vào I_j của unit *j* thì O_j là đầu ra của unit *j*, được tính như sau:

$$O_j = \frac{1}{1 + e^{-I_j}} \tag{2.12}$$

Hàm này cũng được dựa trên một hàm nén (squashing), từ đó nó ánh xạ một miền đầu vào lớn lên trên phạm vi nhỏ hơn, đó là miền [0,1]. Hàm logistic là không tuyến tính và lấy vi phân, cho phép giải thuật lan truyền ngược mô hình hoá các bài toán phân loại, chúng không thể tách biệt một cách tuyến tính.

- *Sai số lan truyền ngược:* Sai số được truyền ngược bằng cách cập nhật trọng số và các bias để phản ánh sai số của dự đoán mạng. Đối với một unit *j* ở lớp đầu ra, sai số Err_j được tính như sau:

$$Err_j = O_j(1 - O_j)(T_j - O_j) \quad (2.13)$$

với O_j là đầu ra thực tế của unit *j* và T_j là đầu ra *true*, dựa trên nhãn lớp đã biết của mẫu huấn luyện cho trước. Lưu ý rằng $O_j(1 - O_j)$ là đạo hàm của hàm logistic.

Để tính sai số của một unit lớp ẩn *j* , tổng có trọng số của các sai số của các unit đã kết nối tới unit *j* ở lớp tiếp theo được xem xét. Sai số của một unit lớp ẩn *j* là:

$$Err_j = O_j(1 - O_j)\sum_k Err_k w_{ij} \quad (2.14)$$

với w_{jk} là trọng số của kết nối từ unit *j* tới một unit *k* trong lớp cao hơn tiếp theo và Err_j là sai số của unit *k*.

Các trọng số và bias được cập nhật để phản ánh các sai số truyền. Các trọng số được cập nhật bởi phương trình (2.15) và (2.16) dưới đây, với Δw_{ij} là thay đổi của trọng số w_{ij}.

$$\Delta w_{ij} = (l)\, Err_j O_i \quad (2.15)$$

$$w_{ij} = w_{ij} + \Delta w_{ij} \quad (2.16)$$

**** 'l' trong phương trình (2.15) là gì?:*** Biến *l* là tốc độ học, là hằng số thuộc khoảng [0,10]. Việc học của lan truyền ngược sử dụng phương pháp giảm độ dốc (gradient descent) để tìm kiếm một tập các trọng số có thể mô hình bài toán phân loại cho trước với mục tiêu tối thiểu hoá trung bình bình phương khoảng cách giữa các dự đoán lớp của mạng và nhãn lớp thực tế của các mẫu. Tốc độ học giúp tránh bị sa lầy tại một tối thiểu hoá cục bộ trong không gian quyết định (tức là tại chỗ các trọng số xuất hiện hội tụ nhưng không phải là giải pháp tối ưu) và giúp tìm ra tối thiểu hoá toàn cục. Nếu như tốc độ học quá nhỏ thì việc học sẽ rất chậm. Nếu tốc độ học quá lớn thì sự dao động giữa các giải pháp

không đầy đủ có thể xuất hiện. Luật thumb cho tốc độ học bằng 1/*t* với *t* là số lần lặp đối với tập huấn luyện.

Các bias được cập nhật bởi phương trình (2.17) và (2.18) dưới đây, $\Delta\theta_j$ thay cho bias θ_j .

$$\Delta\theta_j = (l)\, Err_j \tag{2.17}$$

$$\theta_j = \theta_j + \Delta\theta_j \tag{2.18}$$

Ở đây ta cập nhật các trọng số và các bias sau khi mỗi mẫu được đưa vào. Điều này được quy vào cập nhật trạng thái (case updating). Tiếp đến, trọng số và bias tăng dần có thể được chồng trong các biến để các trọng số và bias được cập nhật sau khi tất cả các mẫu trong tập huấn luyện được đưa ra. Chiến lược sau này được gọi là cập nhật epoch, tại đó một lần lặp hết tập huấn luyện là một epoch. Theo lý thuyết, nguồn gốc toán học của lan truyền ngược là dùng cập nhật epoch, nhưng trong thực tiễn, cập nhật trạng thái càng phổ biến thì độ chính xác các kết quả càng cao hơn.

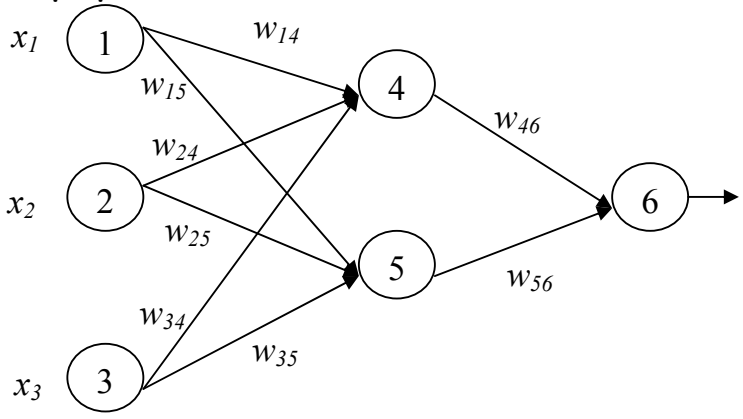
**** Điều kiện tới hạn:*** Huấn luyện dừng khi một trong những điều kiện sau xảy ra:

1. Tất cả Δw_{ij} ở epoch trước nhỏ hơn hoặc bằng ngưỡng được chỉ định.

2. Tỷ lệ phần trăm của mẫu phân loại sai trong epoch trước thấp hơn một ngưỡng nào đó.

3. Một số lượng các epoch được chỉ định từ trước đã kết thúc.

Trong thực tiễn, hàng trăm ngàn các epoch có thể được yêu cầu trước khi các trọng số hội tụ.



Hình 2.10: Ví dụ một mạng nơron truyền thẳng đa mức

Ví dụ 2.5: *Các tính toán mẫu đối với việc học bằng giải thuật lan truyền ngược*

Hình 2.10 cho thấy một mạng nơron truyền thẳng đa mức. Các giá trị trọng số khởi đầu và các giá trị bias của mạng được cho trong bảng 2.3 với mẫu huấn luyện đầu tiên $X = (1,0,1)$.

Bảng 2.3: Các giá trị đầu vào, trọng số và bias khởi đầu

x_1	x_2	x_3	w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_4	θ_4
1	0	1	0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	-0.4	0.2	0.1

Ví dụ này cho thấy các tính toán lan truyền ngược cho mẫu huấn luyện đầu tiên X cho trước. Mẫu được đưa vào trong mạng, mạng đầu vào và đầu ra của mỗi unit được tính toán. Kết quả như bảng 2.4.

Bảng 2.4: Các tính toán mạng đầu vào và đầu ra

Unit j	Mạng đầu vào I_j	Đầu ra O_j
4	0.2 + 0 - 0.5 - 0.4 = -0.7	$1/(1 + e^{0.7}) = 0.33$
5	-0.3 + 0 + 0.2 + 0.2 = 0.1	$1/(1 + e^{-0.1}) = 0.52$
6	(0.3)(0.33) - (0.2)(0.52) + 0.1 = 0.19	$1/(1 + e^{-0.19}) = 0.55$

Sai số của mỗi unit được tính và truyền ngược. Các giá trị sai số được chỉ ra trong bảng 2.5. Trọng số và các bias cập nhật được chỉ ra trong bảng 2.6.

Bảng 2.5: Tính toán sai số tại mỗi nút

Unit j	Err_j
6	(0.55)(1- 0.55)(1- 0.55) = 0.495
5	(0.52)(1- 0.52)(0.495)(-0.3) = 0.037
4	(0.33)(1- 0.33)(0.495)(-0.2) = -0.022

Bảng 2.6: Tính toán việc cập nhật trọng số và bias

Trọng số hay bias	Giá trị mới
w_{46}	-0.3 + (0.9)(0.495)(0.33) = - 0.153
w_{56}	-0.2 + (0.9)(0.495)(0.52) = -0.032
w_{14}	0.2 + (0.9)(-0.022)(1) = 0.180
w_{15}	-0.3 + (0.9)(0.037)(1) = -0.267
w_{24}	0.4 + (0.9)(-0.022)(0) = 0.4
w_{25}	0.1 + (0.9)(0.037)(0) = 0.1

w_{34}	-0.5 + (0.9)(-0.022)(1) = -0.520
w_{35}	0.2 + (0.9)(0.037)(1) = 0.233
θ_6	0.1 + (0.9)(0.495) = 0.546
θ_5	0.2 + (0.9)(0.037) = 0.233
θ_4	-0.4 + (0.9)(-0.022) = -0.420

Một số sự biến đổi và luân phiên ở giải thuật lan truyền ngược được đề xuất để phân loại trong các mạng nơron. Điều này bao gồm điều chỉnh động cấu trúc liên kết mạng và tốc độ học, các tham số khác hay sử dụng các hàm sai số khác nhau.

2.5.4 Lan truyền ngược và tính dễ hiểu

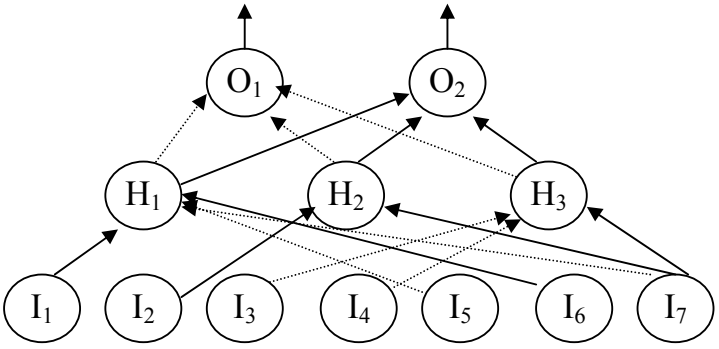
Bất lợi chủ yếu của các mạng nơron đó là khả năng biểu diễn tri thức của chúng. Tri thức có được dưới dạng một mạng các unit có kết nối với các liên kết có trọng số làm khó cho con người khi diễn dịch. Yếu tố này đã thúc đẩy nghiên cứu trích lọc tri thức đã nhúng trong các mạng nơron huấn luyện và trong việc biểu diễn tri thức một cách tượng trưng. Các phương pháp gồm các luật rút trích từ các mạng và phép phân tích độ nhạy.

Nhiều giải thuật khác nhau để rút trích các luật được đề xuất. Các phương pháp điển hình vẫn tồn tại những hạn chế khi đánh giá các thủ tục đã dùng để huấn luyện mạng nơron cho trước, cấu trúc liên kết mạng cho trước và sự rời rạc của các giá trị đầu vào cho trước.

Việc hiểu rõ ràng các mạng kết nối đầy đủ là khó. Do vậy, bước đầu tiên thường có khuynh hướng trích ra các luật từ các mạng nơron gọi là cắt tỉa mạng. Điều này bao gồm gỡ bỏ các liên kết có trọng số mà không làm suy giảm độ chính xác phân loại tại kết quả của mạng đã cho.

Mỗi khi mạng huấn luyện được cắt tỉa, nhiều tiếp cận sau đó sẽ thực hiện việc phân cụm giá trị liên kết, giá trị unit hay giá trị hoạt động. Theo phương pháp này, ví dụ, phân cụm được dùng để tìm tập các giá trị hoạt động thông dụng cho mỗi unit ẩn trong một mạng nơron 2 lớp được huấn luyện đã cho (hình 2.11). Sự kết hợp của các giá trị hoạt động này đối với mỗi unit ẩn được phân tích. Các luật nhận được từ sự kết hợp của các giá trị hoạt động với các giá trị

unit đầu ra tương ứng. Tương tự như vậy, các tập giá trị đầu vào và các giá trị hoạt động được học để đưa ra các luật mô tả mối quan hệ giữa các lớp unit đầu vào và ẩn. Cuối cùng, hai tập luật có thể được kết hợp dưới dạng các luật IF-THEN. Các giải thuật khác có thể nhận được các luật từ những hình thức khác, kể cả các luật *M x N* (với *M* không nằm trong *N* điều kiện cho trước trong tiền đề luật, phải là *true* để mệnh đề kết quả luật được áp dụng), các cây quyết định với các kiểm định *M x N*, các luật mờ và automata hữu hạn.



Nhận biết các tập giá trị hoạt động cho mỗi nút ẩn H_i: for H_1: (-1,0,1) for H_2: (0,1) for H_3: (-1,0.24,1)
Nhận được các luật liên hệ các giá trị hoạt động chung với các nút đầu ra O_j: IF (a_2 =0 AND a_3=-1) OR (a_1= -1 AND a_2=1 AND a_3=-1) OR (a_1=-1 AND a_2=0 AND a_3=0.24) THEN O_1=1, O_2=0 ELSE O_1=0, O_2=1

Nhận được các luật liên hệ các nút đầu vào I_i tới các nút đầu ra O_j:
IF(I_2=0 AND I_7=0) THEN a_2=0
IF(I_4=1 AND I_6=1) THEN a_3=-1
IF(I_5=0) THEN a_3=-1
...

Các luật thu được liên hệ các lớp đầu vào và đầu ra:

IF(I_2=0 AND I_7=0 AND I_4=1 AND I_6=1)THEN class=1

IF(I_2=0 AND I_7=0 AND I_5=0)THEN class=1

Hình 2.11: Các luật có thể được trích ra từ các mạng nơron huấn luyện

**** Phân tích độ nhạy:*** được dùng để đánh giá tác động một biến đầu vào cho trước trên một mạng đầu ra. Đầu vào biến bị biến đổi trong khi vẫn duy trì các biến đầu vào được ấn định tại một vài giá trị. Trong khi đó, các thay đổi ở mạng đầu ra bị giám sát. Tri thức thu được từ dạng phân tích này có thể được biểu diễn dưới dạng các luật như "IF *X giảm* 5% THEN *Y tăng* 8%".

2.6 Phân loại dựa trên sự kết hợp

"*Khai phá luật kết hợp có thể được sử dụng để phân loại không?*"

Khai phá luật kết hợp là một lĩnh vực quan trọng và có tính thiết thực cao của nghiên cứu khai phá dữ liệu. Các kỹ thuật khai phá dữ liệu áp dụng khai phá luật kết hợp cho các bài toán phân loại đã phát triển. Trong phần này, ta nghiên cứu phân loại dựa trên sự kết hợp.

Một phương pháp phân loại dựa trên sự kết hợp gọi là phân loại kết hợp, gồm có 2 bước. Bước đầu tiên, các luật kết hợp được sinh ra sử dụng một version đã sửa đổi của giải thuật khai phá luật kết hợp chuẩn đã biết như Apriori. Bước 2 xây dựng một classifier dựa trên các luật kết hợp đã phát hiện.

Cho D là dữ liệu huấn luyện và Y là tập tất cả các lớp trong D. Giải thuật ánh xạ các thuộc tính xác thực vào các giá trị nguyên dương liên tiếp. Các thuộc tính liên tục được rời rạc hoá và được ánh xạ. Mỗi mẫu dữ liệu d trong D sau đó được biểu diễn bởi một tập các cặp (thuộc tính, giá trị nguyên) gọi là các item và một nhãn lớp y. Cho I là tập tất cả các item trong D. Một luật kết hợp lớp (viết tắt: CAR - class association rule) có dạng $condset \Rightarrow y$, với *conset* là một tập các item ($conset \subseteq I$) và $y \in Y$. Các luật đó được biểu diễn bởi các ruleitem có dạng <*condset,y*>.

CAR có độ tin cậy c nếu c% các mẫu trong D chứa *condset* thuộc lớp y. CAR có hỗ trợ s nếu s% các mẫu trong D chứa *condset* và thuộc lớp y. Tổng hỗ trợ của một *condset* (**condsupCount**) là số lượng mẫu trong D chứa *condset*.

Tổng luật của một *ruleitem* (**rulesupCount**) là số lượng mẫu trong *D* có *condset* và được gán nhãn với lớp *y*. Các *ruleitem* thoả hỗ trợ cực tiểu là các *ruleitem* thường xuyên. Nếu một tập các *ruleitem* có cùng *condset* thì luật với độ tin cậy cao nhất được lựa chọn như một luật có thể (viết tắt: PR - Possible Rule) để miêu tả tập. Một luật thoả độ tin cậy cực tiểu được gọi là luật chính xác.

"*Phân loại kết hợp làm việc như thế nào?*"

Trước tiên, phương pháp phân loại kết hợp tìm tập tất cả các PR mà có cả tính thường xuyên và tính chính xác. Đó chính là các luật kết hợp lớp (viết tắt CARs - class association rules). Một *ruleitem* mà *condset* của nó chứa *k* item là một k-ruleitem. Giải thuật dùng một tiếp cận lặp, ở đây các *ruleitem* được xử lý tốt hơn các *itemset*. Giải thuật quét cơ sở dữ liệu, tìm kiếm k-ruleitems thường xuyên, với k = 1,2,... cho tới khi tất cả các k-ruleitems thường xuyên được tìm ra. Một lần quét được thực hiện đối với mỗi giá trị của *k*. k-ruleitems được dùng để khảo sát (k +1)-ruleitems. Khi quét cơ sở dữ liệu lần đầu tiên, tổng số hỗ trợ của 1-ruleitems được xác định và 1-ruleitems thường xuyên được giữ lại. 1-ruleitems thường xuyên còn gọi là tập F_1 được dùng để sinh ra ứng cử 2-ruleitems C_2. Tri thức của các đặc tính *ruleitem* thường xuyên được dùng để cắt tỉa các *ruleitem* ứng cử không phải là thường xuyên. Tri thức này cho thấy rằng *tất cả các tập con không rỗng của một ruleitems thường xuyên cũng phải là thường xuyên*. Cơ sở dữ liệu được quét lần thứ 2 để tính tổng số hỗ trợ của mỗi ứng cử, để 2-ruleitems thường xuyên (F_2) có thể được xác định. Xử lý này lặp lại với F_k được dùng để sinh ra C_{k+1}, cho tới khi không tìm thấy một ruleitems thường xuyên nào nữa. Các ruleitems thường xuyên mà thoả độ tin cậy cực tiểu hình thành nên tập các CAR. Việc cắt tỉa có thể được áp dụng cho tập luật này.

Bước thứ 2 của phương pháp phân loại kết hợp xử lý các CAR được phát sinh để xây dựng classifier. Vì tổng số lượng các tập con các luật được kiểm tra để xác định tập các luật chính xác nhất có thể là khổng lồ nên một phương pháp heuristic sẽ được dùng. Một thứ tự quyền ưu tiên giữa các luật được định nghĩa, tại đó một luật r_i có độ ưu tiên cao hơn các luật r_j (tức là $r_i \succ r_j$) nếu:

(1) Độ tin cậy của r_i lớn hơn của r_j, hay

(2) Các độ tin cậy là giống nhau nhưng r_i có hỗ trợ lớn hơn, hay

(3) Các độ tin cậy và hỗ trợ của r_i và r_j là như nhau nhưng r_i được sinh ra sớm hơn r_j.

Nhìn chung, giải thuật lựa chọn một tập các CAR quyền ưu tiên cao để phủ các mẫu trong *D*. Classifier duy trì các luật được chọn lựa từ thứ tự ưu tiên cao tới thấp. Khi phân loại một mẫu mới, luật đầu tiên thoả mẫu sẽ được dùng để phân loại nó. Classifier cũng chứa đựng một luật ngầm định, có thứ tự ưu tiên thấp nhất, nó định rõ một lớp ngầm định cho bất kỳ một mẫu mới nào mà không thoả bởi bất cứ một luật nào khác trong classifier.

Do vậy, khai phá luật kết hợp là một chiến lược quan trọng để sinh ra các classifier chính xác và có thể mở rộng.

2.7 Các phương pháp phân loại khác

Phần này, ta mô tả ngắn gọn một số phương pháp phân loại: *k*-láng giềng gần nhất, lập luận dựa trên tình huống, các giải thuật di truyền, tập thô và tập mờ. Trong các hệ thống khai phá dữ liệu thương mại, so với các phương pháp đã mô tả ở các mục trên, các phương pháp này nhìn chung ít được dùng để phân loại hơn. Ví dụ, phân loại láng giềng gần nhất lưu trữ tất cả các mẫu huấn luyện, như vậy sẽ gặp khó khăn khi học từ các tập dữ liệu rất lớn; nhiều ứng dụng của lập luận dựa trên tình huống, các giải thuật di truyền và các tập thô cho phân loại vẫn trong pha nguyên mẫu. Tuy vậy các phương pháp này có mức độ phổ biến ngày càng tăng và sau đây ta sẽ lần lượt xem xét chúng.

2.7.1 Các classifier k-láng giềng gần nhất

Các classifier láng giềng gần nhất dựa trên việc học bằng sự giống nhau. Các mẫu huấn luyện được mô tả bởi các thuộc tính số *n* - chiều. Mỗi mẫu đại diện cho một điểm trong một không gian *n* - chiều. Vì vậy tất cả các mẫu huấn luyện được lưu trữ trong không gian mẫu *n* - chiều. Khi có một mẫu chưa biết cho trước thì classifier *k*-láng giềng gần sẽ tìm kiếm trong không gian mẫu *k* mẫu huấn luyện gần mẫu chưa biết đó nhất. *k* mẫu huấn luyện này là *k* "láng

giềng gần nhất" của mẫu chưa biết. "Độ gần" được định nghĩa dưới dạng khoảng cách Euclidean, tại đó khoảng cách Euclidean giữa hai điểm $X = (x_1,x_2,...,x_n)$ và $Y = (y_1,y_2,...,y_n)$ là:

$$d(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \qquad (2.19)$$

Mẫu chưa biết được phân vào lớp phổ biến nhất trong số k láng giềng gần nhất của nó. Khi $k = 1$ thì mẫu chưa biết được ấn định lớp của mẫu huấn luyện gần nhất với nó trong không gian mẫu.

Các classifier láng giềng gần nhất dựa trên khoảng cách, từ đó chúng lưu trữ tất cả các mẫu huấn luyện. Các kỹ thuật đánh chỉ số hiệu quả được dùng khi số lượng các mẫu huấn luyện là rất lớn. Không giống như cây quyết định quy nạp và lan truyền ngược, các classifier láng giềng gần nhất ấn định các trọng số bằng nhau cho từng thuộc tính. Điều này có thể là nguyên nhân gây nhập nhằng khi có nhiều thuộc tính không thích hợp trong dữ liệu.

Các classifier láng giềng gần nhất cũng được dùng để dự đoán, tức là trả lại một dự đoán giá trị thực cho một mẫu chưa biết cho trước. Lúc này, classifier trả lại giá trị trung bình của các nhãn giá trị thực kết hợp với k-láng giềng gần nhất của mẫu chưa biết đó.

2.7.2 Lập luận dựa trên tình huống

Các classifier lập luận dựa trên tình huống (CBR: Case-based reasoning) là dựa trên khoảng cách. Không giống như các classifier k-láng giềng gần nhất lưu trữ các mẫu huấn luyện như là các điểm trong không gian Euclidean, các mẫu hay "các tình huống" được lưu trữ bởi CRB là các mô tả biểu tượng phức tạp. Các ứng dụng thương mại của CBR gồm bài toán giải quyết dịch vụ khách hàng trợ giúp tại chỗ, ví dụ, tại đó các tình huống mô tả các bài toán chẩn đoán có liên quan tới sản phẩm. CBR cũng được áp dụng cho nhiều lĩnh vực như công trình và pháp luật, tại đó các tình huống hoặc là các thiết kế kỹ thuật, hoặc là các quyết định pháp lý tương ứng.

Khi có một tình huống mới cho trước cần phân loại, một reasoner dựa trên tình huống trước tiên sẽ kiểm tra xem liệu một tình huống huấn luyện đồng nhất tồn tại hay không. Nếu nó được tìm thấy thì giải pháp đi kèm tình huống đó được trả lại. Nếu tình huống đồng nhất không tìm thấy thì reasoner dựa trên tình huống sẽ kiểm tra các tình huống huấn luyện có các thành phần giống các thành phần của tình huống mới. Theo quan niệm, các tình huống huấn luyện này có thể được xem xét như là các láng giềng của tình huống mới. Nếu các tình huống được biểu diễn như các đồ thị, điều này bao gồm cả việc tìm kiếm các đồ thị con giống với các đồ thị con nằm trong phạm vi tình huống mới. Reasoner dựa trên tình huống thử kết hợp giải pháp của các tình huống huấn luyện láng giềng để đề ra một giải pháp cho tình huống mới. Nếu xảy ra hiện tượng không tương hợp giữa các giải pháp riêng biệt thì quay lui để tìm kiếm các giải pháp cần thiết khác. reasoner dựa trên tình huống có thể dùng nền tảng tri thức và các chiến lược giải quyết bài toán để đề xuất một giải pháp kết hợp khả thi.

Những thách thức trong lập luận dựa trên tình huống đó là tìm một metric tương tự tốt (ví dụ, đối với các đồ thị con đối sánh), phát triển các kỹ thuật hiệu quả để đánh chỉ số các tình huống huấn luyện và các phương pháp cho các giải pháp kết hợp.

2.7.3 Các giải thuật di truyền

Các giải thuật di truyền cố gắng kết hợp chặt chẽ các ý tưởng phát triển tự nhiên. Việc học di truyền nhìn chung sẽ được bắt đầu như sau: Một quần thể (*population*) ban đầu được tạo gồm các luật được sinh ra ngẫu nhiên. Mỗi luật được biểu diễn bởi một dãy các bit. Ví dụ, giả sử rằng các mẫu trong một tập huấn luyện cho trước được mô tả bởi hai thuộc tính Boolean A_1 và A_2, có hai lớp C_1 và C_2. Luật "*IF* A_1 *and not* A_2 *THEN* C_2" được mã hoá thành dãy bit "100", với 2 bit trái nhất đại diện cho các thuộc tính A_1 và A_2 và bit phải nhất đại diện cho lớp. Tương tự, luật "*IF not* A_1 *and not* A_2 *THEN* C_1" được mã hoá thành "001". Nếu một thuộc tính có giá trị k với $k > 2$ thì k bit được dùng để mã hoá các giá trị thuộc tính. Các lớp có thể được mã hoá theo cách tương tự.

Dựa trên khái niệm về sự tồn tại của kiểm định phù hợp, một quần thể mới được thiết lập bao gồm các luật kiểm định phù hợp trong quần thể hiện thời, cũng như con cháu (*offspring*) của các luật này. Sự phù hợp của một luật được đánh giá bởi độ chính xác phân loại của nó trên một tập các mẫu huấn luyện.

Con cháu được tạo bằng cách áp dụng các phép di truyền như lai nhau và đột biến. Trong phép toán lai nhau, các chuỗi con từ các cặp luật được trao đổi để thiết lập các cặp luật mới. Trong phép toán đột biến, các bit được lựa chọn ngẫu nhiên trong chuỗi luật đã đảo ngược.

Xử lý việc sinh ra các quần thể mới dựa trên các quần thể trước của các luật tiếp tục cho tới khi một quần thể *P* "tiến hoá" tại đó mỗi luật trong *P* thoả một ngưỡng phù hợp được chỉ định trước.

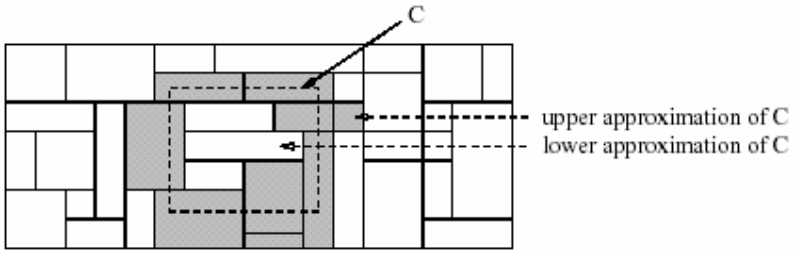
Các giải thuật di truyền dễ xử lý song song và được sử dụng cho phân loại cũng như các bài toán tối ưu khác. Trong khai phá dữ liệu, chúng có thể được dùng để đánh giá độ phù hợp của các giải thuật khác.

2.7.4 Lý thuyết tập thô

Lý thuyết tập thô được dùng cho phân loại để phát hiện ra các mối quan hệ có cấu trúc trong phạm vi dữ liệu không chính xác hay dữ liệu nhiễu. Nó áp dụng cho các thuộc tính có giá trị rời rạc. Các thuộc tính có giá trị liên tục do vậy phải được rời rạc hoá trước khi sử dụng.

Lý thuyết tập thô dựa trên sự thiết lập các lớp tương đương trong phạm vi dữ liệu huấn luyện. Tất cả các mẫu dữ liệu tạo thành một lớp tương đương không phân biệt được, đó là các mẫu đồng nhất về phương diện các thuộc tính mô tả dữ liệu. Trong dữ liệu thế giới thực cho trước, thông thường là các lớp không thể được phân biệt dưới dạng của các thuộc tính có sẵn. Các tập thô được dùng để xấp xỉ hay "làm thô" định nghĩa các lớp như vậy. Định nghĩa tập thô cho một lớp *C* cho trước được xấp xỉ bởi hai tập - một xấp xỉ thấp hơn *C* và một xấp xỉ cao hơn *C*. Xấp xỉ thấp hơn *C* gồm tất cả các mẫu dữ liệu dựa trên tri thức của các thuộc tính, tất nhiên thuộc về *C* mà không mập mờ. Xấp xỉ cao hơn *C* gồm tất cả các mẫu dữ liệu được dựa trên tri thức của các thuộc tính, không

được mô tả như không thuộc về *C*. Các xấp xỉ thấp hơn và cao hơn của lớp *C* như biểu diễn ở hình 2.12, tại đó miền mỗi hình chữ nhật đại diện cho một lớp tương đương. Các luật quyết định có thể được sinh ra cho mỗi lớp, một bảng quyết định được dùng để miêu tả các luật.



Hình 2.12: Một xấp xỉ tập thô của tập các mẫu thuộc lớp *C*

Các tập thô cũng được dùng để giảm bớt đặc trưng (các thuộc tính không góp phần vào việc phân loại dữ liệu huấn luyện cho trước, chúng có thể được nhận biết và gỡ bỏ) và phép phân tích sự thích hợp (sự đóng góp hay ý nghĩa của mỗi thuộc tính được đánh giá dưới phương diện là tác vụ phân loại). Bài toán tìm kiếm các tập con tối thiểu (các reduct) của các thuộc tính có thể mô tả tất cả các khái niệm trong tập dữ liệu đã cho là NP-khó. Tuy nhiên, các giải thuật để giảm mức độ tính toán được đã đề xuất. Ví dụ, dùng một ma trận nhận thức (discernibility matrix) lưu trữ các khác biệt của các giá trị thuộc tính đối với mỗi cặp mẫu dữ liệu. Hơn nữa, ma trận này thay cho việc tìm kiếm để dò các thuộc tính dư thừa trên toàn bộ tập huấn luyện.

2.7.5 Các tiếp cận tập mờ

Các hệ thống dựa trên luật cho phân loại có điểm bất lợi đó là chúng đòi hỏi các ngưỡng rõ ràng cho các thuộc tính liên tục. Ví dụ, xem luật (2.20) dưới đây để thấy chấp thuận yêu cầu cho khách hàng vay. Về cơ bản luật cho biết các yêu cầu đối với khách hàng: phải là những người đã có việc làm ít nhất trong hai năm và thu nhập tối thiểu \$50K thì mới được chấp thuận.

IF (*năm công tác*≥*2*)∧(*thu nhập*> 50*K*) *THEN quyết định=chấp thuận* (2.20)

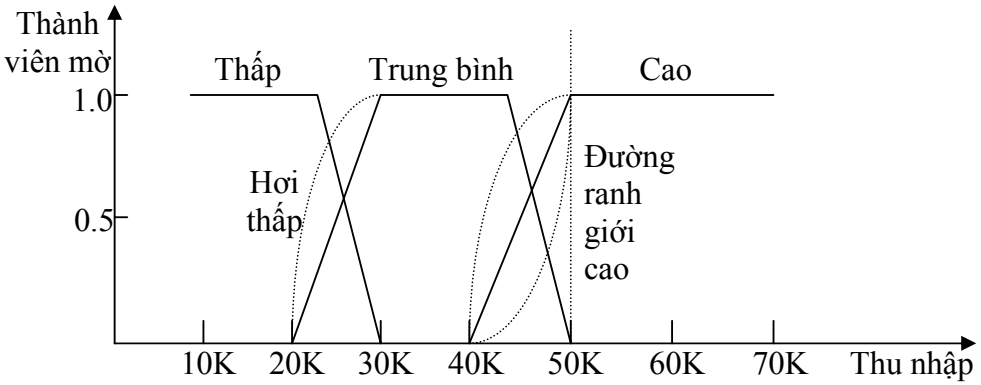
Với luật (2.20), một khách hàng - người mà đã làm việc ít nhất là 2 năm sẽ được cho vay nếu thu nhập của cô ta là \$51K, nhưng không nhận được nếu là

$50K. Đặt ngưỡng thô như vậy có vẻ không thuận lợi lắm. Logic mờ sẽ khắc phục được nhược điểm này bằng cách định nghĩa các ngưỡng mờ hay các đường biên "mờ". Không cần một ngưỡng rõ ràng giữa các tập hay các loại, logic mờ sử dụng các giá trị chân lý giữa 0.0 và 1.0 để biểu diễn mức độ thành viên của một giá trị nào đó vào một loại cho trước. Do vậy, với logic mờ, ta có được khái niệm *thu nhập*=$50K ở một mức độ nào đó là cao mặc dầu không cao như *thu nhập*= $51K.

Logic mờ hữu ích cho các hệ thống khai phá dữ liệu biểu diễn phân loại. Nó cung cấp thuận lợi khi làm việc tại một mức trừu tượng cao. Nhìn chung, tính hữu ích của logic mờ trong các hệ thống dựa trên luật bao gồm:

- Các giá trị thuộc tính được chuyển đổi sang các giá trị mờ. Hình 2.13 cho thấy các giá trị cho thuộc tính liên tục *thu nhập* được ánh xạ vào trong các loại rời rạc {*thấp, trung bình, cao*}, cũng như các giá trị thành viên mờ hay chân lý được tính toán như thế nào. Các hệ thống logic mờ cung cấp các công cụ đồ thị để trợ giúp các user trong bước này.

- Đối với một mẫu mới cho trước, có thể áp dụng nhiều hơn một luật mờ. Mỗi một luật thích hợp xây dựng một biểu quyết thành viên trong các loại, điển hình, các giá trị chân lý cho mỗi loại đã dự đoán được tính tổng.



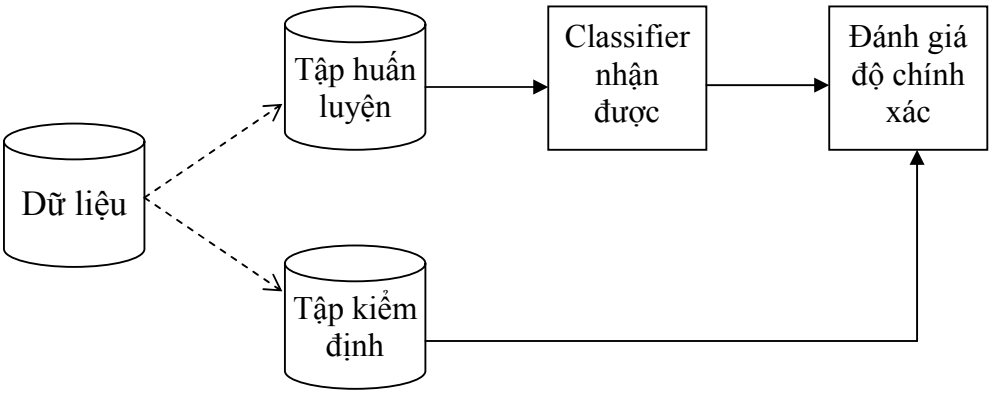
Hình 2.13: Các giá trị mờ đôi với *thu nhập*

- Các tổng có được ở trên được kết hợp vào trong một giá trị mà hệ thống cấp. Xử lý này có thể được làm bằng cách đánh trọng số mỗi loại bằng tổng chân lý của nó và nhân với giá trị chân lý trung bình của mỗi loại. Các phép tính

này có thể là phức tạp hơn, tuỳ thuộc vào độ phức tạp của các đồ thị thành viên mờ.

Các hệ thống logic mờ được dùng để phân loại trong nhiều lĩnh vực như chăm sóc sức khoẻ, tài chính.

2.8 Độ chính xác classifier



Hình 2.14: Đánh giá độ chính xác classifier với phương pháp holdout

Đánh giá độ chính xác classifier là việc quan trọng. Dữ liệu để đánh giá là dữ liệu không dùng để huấn luyện classifier, độ chính xác một classifier là độ phù hợp của nhãn dữ liệu tương lai. Ví dụ, huấn luyện một classifier từ dữ liệu bán hàng để dự đoán thói quen mua sắm của khách hàng, ta cần đánh giá độ chính xác classifier có thể dự đoán thói quen mua sắm của các khách hàng tương lai như thế nào. Độ chính xác đánh giá này sẽ trợ giúp cho việc so sánh các classifier khác nhau. Phần 2.9.1 nói về các kỹ thuật để đánh giá độ chính xác classifier như phương pháp holdout và hợp lệ chéo k-fold. Trong mục 2.9.2 mô tả hai chiến lược để tăng độ chính xác classifier: bagging và boosting. Mục 2.9.3 là các vấn đề có liên quan đến việc lựa chọn classifier.

2.8.1 Đánh giá độ chính xác classifier

Holdout và hợp lệ chéo là hai kỹ thuật phổ biến để đánh giá độ chính xác classifier dựa trên các phân chia lấy mẫu ngẫu nhiên từ dữ liệu cho trước.

Trong phương pháp holdout, dữ liệu đã cho được phân chia ngẫu nhiên vào trong hai tập độc lập: một tập huấn luyện và một tập kiểm định. Hai phần ba dữ liệu được chỉ định là tập huấn luyện và còn lại một phần ba được chỉ định là tập

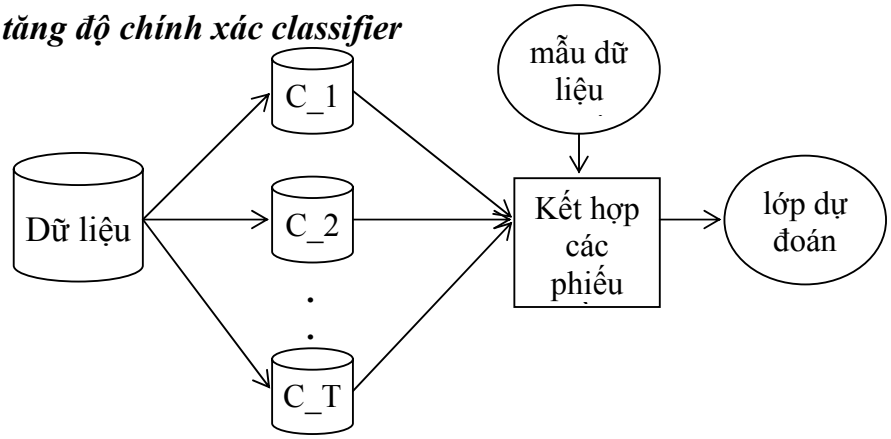
kiểm định. Tập huấn luyện được dùng để thu classifier, độ chính xác của nó được đánh giá với tập kiểm định (Hình 2.14). Việc đánh giá này là lạc quan bởi chỉ một phần dữ liệu ban đầu được dùng để thu classifier. Lấy mẫu con ngẫu nhiên là một sự thay đổi của phương pháp holdout trong đó phương pháp holdout được lặp lại *k* lần. Độ chính xác classifier bằng giá trị trung bình của các độ chính xác có được từ mỗi lần lặp.

Trong hợp lệ chéo *k*-fold, dữ liệu ban đầu được phân chia ngẫu nhiên vào trong *k* tập con riêng biệt ("các fold") $S_1,S_2,...,S_k$, chúng có kích thước xấp xỉ bằng nhau. Huấn luyện và kiểm định được thực hiện *k* lần. Trong lần lặp thứ *i*, tập con S_i đóng vai trò như một tập kiểm định và các tập con còn lại được dùng chung để huấn luyện classifier. Tức là classifier của lần lặp đầu tiên được huấn luyện trên các tập con $S_2,S_3,...,S_k$ và được kiểm định trên S_1; classifier của lần lặp thứ 2 được huấn luyện trên các tập con $S_1,S_3,...,S_k$ và được kiểm định trên S_2, v.v... Độ chính xác classifier là toàn bộ số lượng các phân loại chính xác từ *k* lần lặp chia cho tổng số lượng các mẫu trong dữ liệu ban đầu. Trong hợp lệ chéo phân tầng, các fold được phân tầng để sự phân bố lớp của các mẫu trong mỗi fold xấp xỉ như sự phân bố lớp trong dữ liệu ban đầu.

Nhìn chung, phân tầng hợp lệ chéo 10-fold được đề nghị để đánh giá độ chính xác classifier (thậm chí nếu khả năng tính toán cho phép thì có thể sử dụng nhiều fold hơn).

Sử dụng các kỹ thuật này để đánh giá độ chính xác classifier, làm tăng tổng số lần tính toán, tuy nhiên nó lại hữu ích cho việc lựa chọn classifier.

2.8.2 Gia tăng độ chính xác classifier



Hình 2.15: Tăng độ chính xác classifier

Trong mục trước, ta đã nghiên cứu các phương pháp đánh giá độ chính xác classifier. Trong mục 2.3.2 ta đã thấy cắt tỉa có thể được áp dụng vào cây quyết định để giúp cải thiện độ chính xác của kết quả các cây quyết định. Bagging (hay boostrap aggregation) và boosting là hai kỹ thuật (như hình 2.15). Mỗi khi kết hợp một loạt T classifier đã học $C_1,C_2,...,C_T$ sẽ tạo ra một classifier hỗn hợp được cải tiến C_*.

"*Các phương pháp này làm việc như thế nào?*" Giả sử rằng bạn là một bệnh nhân và bạn cần có một chẩn đoán được làm dựa trên các triệu chứng của bạn. Thay vì hỏi bác sỹ, bạn có thể tự lựa chọn. Nếu một chẩn đoán nào đó chuẩn hơn những cái khác, bạn sẽ lựa chọn là chẩn đoán cuối cùng hay chẩn đoán tốt nhất. Bây giờ thay thế mỗi bác sỹ bằng một classifier và bạn có khả năng trực giác đằng sau bagging. Bạn ấn định các trọng số bằng giá trị hay "trị giá" mỗi chẩn đoán của bác sỹ dựa trên độ chính xác của các chẩn đoán trước đó chúng đã làm. Chẩn đoán cuối cùng là sự kết hợp của các chẩn đoán có trọng số. Đây là bản chất của boosting. Ta sẽ có một cái nhìn gần hơn ở 2 kỹ thuật này:

Cho trước một tập S có s mẫu, bagging làm việc như sau. Tại lần lặp t (t = 1,2,...,T), một tập huấn luyện S_t được lấy mẫu, thay thế tập các mẫu gốc S. Từ khi sử dụng việc lấy mẫu với thay thế, một vài trong số các mẫu của S có thể không có mặt trong S_t, trong khi các mẫu khác có thể xuất hiện nhiều hơn một lần. Một classifier C_t được học cho mỗi tập huấn luyện S_t. Để phân loại một mẫu không biết X, mỗi classifier C_t phải trả lại dự đoán lớp cho nó, nó đếm như một phiếu bầu. Classifier thu được C_* đếm các phiếu bầu và các ấn định lớp với số phiếu bầu nhiều nhất cho X. Bagging có thể được áp dụng để dự đoán các giá trị liên tục bằng cách lấy giá trị trung bình của các phiếu bầu, hơn là lấy theo số đông giá trị.

Trong boosting, các trọng số được ấn định cho từng mẫu huấn luyện. Một loạt các classifier được học. Sau khi một classifier C_t được học, các trọng số được cập nhật để cho phép classifier tiếp theo C_{t+1} "chú ý nhiều hơn" tới các sai

số phân loại sai đã có với C_t. Classifier đã boost cuối cùng C_* kết hợp các phiếu bầu của mỗi classifier riêng lẻ, tại đó trọng số của mỗi phiếu bầu của classifier có chức năng là độ chính xác của nó. Giải thuật boosting có thể được mở rộng để dự đoán các giá trị liên tục.

2.8.3 Độ chính xác có đủ để đánh giá một classifier hay không?

Thêm vào độ chính xác, các classifier có thể được so dưới phương diện tốc độ và sự tráng kiện của chúng (ví dụ, độ chính xác trên dữ liệu nhiễu), khả năng mở rộng, và khả năng diễn dịch. Khả năng mở rộng có thể được ước lượng bằng cách đánh giá số lượng các thao tác I/O cần có cho một giải thuật phân loại cho trước trên các tập dữ liệu với kích thước tăng dần.

Trong các bài toán phân loại, giả sử rằng tất cả các đối tượng được phân loại duy nhất, tức là mỗi mẫu huấn luyện thuộc về chỉ một lớp. Như ta thảo luận ở trên, các giải thuật phân loại sau đó có thể được so sánh theo độ chính xác của chúng. Tuy nhiên, bởi tính đa dạng của dữ liệu trong các cơ sở dữ liệu lớn, việc giả sử rằng tất cả các đối tượng được phân loại được duy nhất không phải luôn hợp lý. Hơn nữa, giả định mỗi đối tượng thuộc về nhiều hơn một lớp có khả năng xảy ra nhiều hơn.

Việc trả lại một xác suất phân bố lớp hữu ích hơn việc trả lại một nhãn lớp. Các phép đo độ chính xác sau đó có thể sử dụng một heuristic dự đoán lần hai nhờ đó một dự đoán lớp được đánh giá chính xác nếu nó thích hợp với lớp có khả năng thứ nhất hay thứ hai. Mặc dầu điều này không được nghiên cứu, nhưng một mức độ nào đó sự phân lớp các đối tượng là không duy nhất. Đây không phải là một giải pháp đầy đủ.

2.9 Kết luận

Như vậy chương 2 đã trình bày khái niệm, các bước và các phương pháp phân loại, phương pháp đánh giá độ chính xác phân loại và gia tăng độ chính xác này.

CHƯƠNG 3: KỸ THUẬT PHÂN CỤM TRONG KHAI PHÁ DỮ LIỆU

3.1 Phân cụm là gì

Xử lý nhóm một tập các đối tượng vào trong các lớp các đối tượng giống nhau được gọi là phân cụm. Một cụm là một tập hợp các đối tượng dữ liệu giống nhau trong phạm vi cùng một cụm và không giống nhau với các đối tượng trong các cụm khác.

Phép phân tích cụm là một hoạt động quan trọng. Thời kì đầu, nó học làm thế nào để phân biệt giữa mèo và chó hay giữa động vật và thực vật, bằng cách trau dồi liên tục tiềm thức các lược đồ phân loại. Phép phân tích cụm được dùng rộng rãi trong nhiều ứng dụng, bao gồm nhận dạng, phép phân tích dữ liệu, xử lý ảnh, nghiên cứu thị trường, v.v... Bằng phân cụm, ta có thể nhận biết các vùng đông đúc và thưa thớt, bởi vậy tìm ra toàn bộ các mẫu phân bố và các tương quan thú vị giữa các thuộc tính dữ liệu. Trong kinh doanh, phân cụm có thể giúp cho các nhà nghiên cứu thị trường tìm ra các nhóm riêng biệt dựa trên khách hàng của họ và mô tả các nhóm khách hàng dựa trên các mẫu mua sắm. Trong sinh vật học, nó có thể được dùng để có được các nguyên tắc phân loại thực vật và động vật, phân loại gien theo chức năng giống nhau và có được sự hiểu biết thấu đáo các cấu trúc kế thừa trong các mẫu. Phân cụm cũng có thể được dùng để nhận biết các vùng đất giống nhau dùng trong cơ sở dữ liệu quan sát trái đất và nhận biết các nhóm có hợp đồng bảo hiểm ô tô với mức chi phí trung bình cao, cũng như nhận biết các nhóm nhà trong thành phố theo kiểu nhà, giá trị và khu vực địa lý. Nó có thể cũng giúp cho việc phân loại dữ liệu trên WWW để khai thác thông tin. Như một hàm khai phá dữ liệu, phép phân tích cụm được dùng như là một công cụ độc lập để có thể nhìn thấu được bên trong sự phân bố dữ liệu, để quan sát các đặc điểm của mỗi cụm và tập trung trên một tập đặc biệt các cụm cho phép phân tích xa hơn. Tiếp theo, nó phục vụ như là một bước tiền xử lý cho các giải thuật khác như phân loại và mô tả, thao tác trên các cụm đã dò được.

Phân cụm dữ liệu là một môn khoa học trẻ đang phát triển mạnh mẽ. Có một số lượng lớn các bài báo nghiên cứu trong nhiều hội nghị, hầu hết trong các lĩnh vực của khai phá dữ liệu: thống kê, học máy, cơ sở dữ liệu không gian, sinh vật học, kinh doanh, v.v...với tầm quan trọng và các kỹ thuật khác nhau. Do số lượng lớn các dữ liệu đã thu thập trong cơ sở dữ liệu nên phép phân tích cụm gần đây trở thành một chủ đề tích cực cao trong nghiên cứu khai phá dữ liệu.

Như là một nhánh của thống kê, phép phân tích cụm được nghiên cứu mở rộng đã nhiều năm, tập trung chính trên phép phân tích cụm dựa trên khoảng cách. Các công cụ phân tích cụm dựa trên *k-means*, *k-medoids* và một số các phương pháp khác cũng được xây dựng trong nhiều gói phần mềm hay hệ thống phân tích thống kê như S-Plus, SPSS và SAS.

Trong học máy, phép phân tích cụm thường dựa trên học không giám sát. Không giống như phân loại, phân cụm không dựa trên các lớp đã định nghĩa trước và các mẫu dữ liệu huấn luyện đã gắn nhãn lớp. Bởi lý do này nên nó có dạng là học bằng sự quan sát, hơn là học bằng các mẫu. Trong phân cụm khái niệm, một nhóm đối tượng hình thành nên một lớp chỉ khi nào nó được mô tả bởi một khái niệm. Điều này không giống với phân cụm theo cách truyền thống - cách mà đo tính giống nhau dựa trên khoảng cách hình học. Phân cụm truyền thống bao gồm hai thành phần: (1) Nó khám phá các lớp thích hợp. (2) Nó thiết lập các mô tả cho mỗi lớp như trong phân loại. Nguyên tắc chỉ đạo vẫn là làm sao cho độ giống nhau trong cùng một lớp là cao và độ giống nhau giữa các lớp là thấp.

Trong khai phá dữ liệu, người ta thường nghiên cứu các phương pháp để phép phân cụm ngày càng hiệu quả trong các cơ sở dữ liệu lớn. Các chủ đề tích cực của nghiên cứu tập trung trên khả năng mở rộng của các phương pháp phân cụm, hiệu quả của các phương pháp phân cụm dữ liệu có hình dạng và kiểu phức tạp, các kỹ thuật phân cụm cho dữ liệu với số chiều cao và các phương pháp phân cụm có sự pha trộn của dữ liệu số và dữ liệu xác thực trong các cơ sở dữ liệu lớn.

Phân cụm là một lĩnh vực nghiên cứu có nhiều thách thức, tại đó các ứng dụng tiềm năng của nó đưa ra các yêu cầu đặc biệt. Sau đây là các yêu cầu điển hình của phân cụm trong khai phá dữ liệu:

1. *Khả năng mở rộng:* Nhiều giải thuật phân cụm làm việc tốt trong các tập dữ liệu nhỏ chứa ít hơn 200 đối tượng dữ liệu, tuy nhiên một cơ sở dữ liệu lớn có thể chứa hàng triệu đối tượng. Phân cụm cho một mẫu của một tập dữ liệu lớn cho trước có thể dẫn tới các kết quả bị lệch. Ta có thể phát triển các giải thuật phân cụm có khả năng mở rộng cao trong các cơ sở dữ liệu lớn như thế nào?

2. *Khả năng giải quyết các kiểu khác nhau của các thuộc tính:* Nhiều giải thuật được thiết kế để phân cụm dữ liệu số dựa trên khoảng cách. Tuy nhiên, nhiều ứng dụng có thể yêu cầu phân cụm các kiểu khác nhau của dữ liệu như nhị phân, xác thực (tên) và dữ liệu có thứ tự hay sự pha trộn các kiểu dữ liệu này.

3. *Phát hiện ra các cụm với hình dạng tuỳ ý:* Nhiều giải thuật phân cụm định rõ các cụm dựa trên các phép đo khoảng cách Euclidean và Manhattan. Các giải thuật dựa trên các phép đo khoảng cách như thế này có khuynh hướng tìm các cụm hình cầu với kích thước và mật độ giống nhau. Tuy nhiên, một cụm có thể có hình dạng bất kỳ. Điều này rất quan trọng để phát triển các giải thuật - các giải thuật này có thể phát hiện ra các cụm có hình dạng tuỳ ý.

4. *Các yêu cầu tối thiểu cho miền tri thức để xác định rõ các tham số đầu vào:* Nhiều giải thuật phân cụm yêu cầu người dùng nhập vào các tham số nào đó trong phép phân tích cụm (như số lượng các cụm đã đề nghị). Kết quả phân cụm thường rất nhạy cảm với các tham số đầu vào. Nhiều tham số khó xác định, đặc biệt đối với các tập dữ liệu chứa các đối tượng số chiều cao. Điều này không chỉ là gánh nặng cho các user mà còn làm cho chất lượng phân cụm khó điều khiển.

5. *Khả năng giải quyết dữ liệu nhiễu:* Hầu hết các cơ sở dữ liệu thế giới thực chứa các outlier hay các dữ liệu khuyết, dữ liệu không biết hay dữ liệu sai.

Nhiều giải thuật phân cụm nhạy cảm với dữ liệu như thế này và có thể dẫn tới chất lượng các cụm kém.

6. *Sự không nhạy cảm khi sắp xếp các bản ghi đầu vào:* Nhiều giải thuật phân cụm nhạy cảm với trật tự của dữ liệu đầu vào, ví dụ: cùng một tập dữ liệu, khi trình diễn với các trật tự khác nhau trong cùng một giải thuật, có thể phát sinh đột xuất các cụm khác nhau. Do vậy, việc phát triển các giải thuật nhạy cảm với trật tự đầu vào thực sự quan trọng.

7. *Số chiều cao:* Một cơ sở dữ liệu hay một kho dữ liệu có thể chứa các chiều hay thuộc tính khác nhau. Nhiều giải thuật phân cụm có chất lượng rất tốt khi vận dụng dữ liệu với số chiều thấp, khoảng hai tới ba chiều. Mắt người rất giỏi xét đoán chất lượng phân cụm cho tới ba chiều. Thách thức đang đặt ra đối với việc phân cụm các đối tượng dữ liệu trong không gian có số chiều cao, đặc biệt lưu ý đến dữ liệu trong không gian số chiều cao có thể rất thưa thớt và bị lệch nhiều.

3. *Phân cụm dựa trên ràng buộc:* Các ứng dụng thế giới thực có thể cần thực hiện phân cụm dưới rất nhiều loại ràng buộc. Giả sử công việc của bạn là lựa chọn vị trí để đặt một số lượng cho trước các trạm tiền trả tiền tự động (ATMs) mới trong thành phố. Để giải quyết điều này, bạn có thể phân cụm các hộ gia đình trong khi xem xét các con sông và mạng lưới đường quốc lộ của thành phố và các yêu cầu khách hàng trên từng vùng như là các ràng buộc. Một nhiệm vụ đặt ra đó là tìm các nhóm dữ liệu với chất lượng phân cụm tốt và thoả rất nhiều ràng buộc khác nhau.

9. *Khả năng diễn dịch và tính tiện lợi:* Các user có thể trông chờ các kết quả phân cụm ở khả năng diễn dịch, tính toàn diện và tiện lợi. Phân cụm có thể cần được liên kết với các cách hiểu ngữ nghĩa cụ thể và các ứng dụng cụ thể. Việc nghiên cứu mục đích của ứng dụng ảnh hưởng như thế nào đến việc lựa chọn các phương pháp phân cụm là thực sự quan trọng.

Với các yêu cầu này, ta sẽ lần lượt nghiên cứu các xử lý phép phân tích cụm như sau: Trước tiên ta nghiên cứu các kiểu khác nhau của dữ liệu và chúng

có ảnh hưởng tới các phương pháp phân cụm như thế nào. Thứ hai, ta đưa ra một phân loại tổng quát các phương pháp phân cụm. Sau đó ta nghiên cứu mỗi phương pháp phân cụm một cách chi tiết, bao gồm các phương pháp phân chia, các phương pháp phân cấp, các phương pháp dựa trên mật độ, các phương pháp dựa trên lưới và các phương pháp dựa trên mô hình. Ta cũng kiểm tra phân cụm trong không gian có số chiều cao và thảo luận sự khác nhau của các phương pháp khác nhau.

3.2 Các kiểu dữ liệu trong phép phân cụm

Trong phần này, ta nghiên cứu các kiểu dữ liệu thường xuất hiện trong các phép phân cụm và tiền xử lý chúng như thế nào cho phép phân tích này. Giả sử rằng một tập dữ liệu được phân cụm chứa *n* đối tượng, nó có thể đại diện cho *người*, *nhà*, *văn bản*, *đất nước*, v.v... Các giải thuật phân cụm dựa trên bộ nhớ chính thao tác trên một trong hai cấu trúc dữ liệu sau:

1. *Ma trận dữ liệu* (hay cấu trúc: *đối tượng* x *biến*): Được đại diện bởi *n đối tượng*, ví dụ như *người* với *p biến* (còn được gọi là các *phép đo* hay các *thuộc tính*) như *tuổi*, *chiều cao*, *giới tính*, v.v... Cấu trúc có dạng bảng quan hệ, hay ma trận *n* x *p* (*n đối tượng* x *p biến*) như trong (3.1)

$$\begin{bmatrix} x_{11} & \dots & x_{1f} & \dots & x_{1p} \\ \dots & \dots & \dots & \dots & \dots \\ x_{i1} & \dots & x_{if} & \dots & x_{ip} \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1} & \dots & x_{nf} & \dots & x_{np} \end{bmatrix} \tag{3.1}$$

2. *Ma trận không tương đồng* (hay cấu trúc *đối tượng* x *đối tượng*): Nó lưu trữ một tập hợp các trạng thái (về mặt không gian, thời gian,...) cho tất cả *n* cặp *đối tượng*. Nó thường được biểu diễn bởi bảng *n* x *n* như hình (3.2)

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{bmatrix} \tag{3.2}$$

với $d(i,j)$ được đo bởi sự khác nhau hay không tương đồng giữa các đối tượng i và j. Do vậy $d(i,j) = d(j,i)$ và $d(i,i) = 0$, ta có ma trận trong hình (3.2). Các phép đo không tương đồng được thảo luận trong suốt phần này.

Ma trận dữ liệu thường được gọi là ma trận 2-mode (2 chế độ), trong khi đó ma trận không tương đồng được gọi là ma trận 1-mode (1 chế độ). Nhiều giải thuật phân cụm thao tác trên ma trận không tương đồng. Nếu dữ liệu được đưa ra dưới dạng ma trận dữ liệu thì nó có thể được chuyển đổi sang ma trận không tương đồng trước khi áp dụng các giải thuật phân cụm.

Cụm các đối tượng được tính toán dựa trên sự tương đồng hay không tương đồng của chúng. Trong phần này, trước tiên ta thảo luận chất lượng phân cụm có thể được đánh giá dựa trên các hệ số tương quan - có thể chuyển đổi thành các hệ số không tương đồng hay tương đồng. Sau đó ta thảo luận làm thế nào để tính độ không tương đồng của các đối tượng được mô tả bởi các biến dựa trên khoảng cách, các biến nhị phân, các biến dựa trên tên, có thứ tự và tỷ lệ (ratio) hay sự kết hợp của các kiểu biến này.

3.2.1 Độ không tương đồng và tương đồng: Đo chất lượng phân cụm

Phép đo của các hệ số không tương đồng hay tương đồng được dùng để đo chất lượng phân cụm. Độ không tương đồng $d(i,j)$ là một số không âm, nó gần bằng 0 khi i, j gần nhau và sẽ lớn hơn khi chúng khác biệt nhau nhiều hơn.

Không tương đồng có được bằng các đánh giá chủ quan đơn giản bởi một tập các observer (quan sát viên) hay các chuyên gia trên các đối tượng khác nhau nào đó. Sự không tương đồng được tính toán từ các hệ số tương quan. Cho trước n đối tượng để phân cụm, tương quan *Pearson product-moment* giữa hai biến f và g được định nghĩa trong (3.3), tại đó f và g là các biến mô tả các đối tượng, m_f và m_g là các giá trị trung bình của f và g và x_{if} là giá trị của f cho đối tượng thứ i, x_{ig} là giá trị của g cho đối tượng thứ i.

$$R(f,g) = \frac{\sum_{i=1}^{n}(x_{if} - m_f)(x_{ig} - m_g)}{\sqrt{\sum_{i=1}^{n}(x_{if} - m_f)^2}\sqrt{\sum_{i=1}^{n}(x_{ig} - m_g)^2}} \qquad (3.3)$$

Công thức chuyển đổi (3.4) được dùng để tính hệ số không tương quan $d(f,g)$ từ các hệ số tương quan $R(f,g)$:

$$d(f,g) = (1 - R(f,g))/2 \quad (3.4)$$

Các biến với một tương quan dương cao sẽ ấn định hệ số không tương đồng gần bằng 0. Các biến với một tương quan âm mạnh sẽ ấn định hệ số không tương đồng gần bằng 1 (nghĩa là các biến rất khác nhau).

Trong nhiều ứng dụng, người dùng thích dùng công thức chuyển đổi (3.5) hơn, tại đó các biến với tương quan âm hay dương cao ấn định cùng một giá trị tương đồng cao.

$$d(f,g) = 1 - |R(f,g)| \quad (3.5)$$

Người dùng có thể sử dụng hệ số tương đồng $s(i,j)$ thay cho hệ số không tương đồng. Công thức (3.6) được dùng để chuyển đổi giữa hai hệ số.

$$s(i,j) = 1 - d(i,j) \quad (3.6)$$

Lưu ý rằng không phải tất cả các biến đều cần trong phép phân tích cụm. Một biến là vô nghĩa với một phân cụm cho trước thì tính hữu ích sẽ ít hơn, do vậy nó ẩn đi thông tin hữu ích đã cung cấp bởi các biến khác. Ví dụ, số điện thoại của một người thường vô ích trong phân cụm người theo mô tả về họ như *tuổi*, *chiều cao*, *cân nặng*, v.v...Kiểu biến "rác" như vậy nên có trọng số 0, trừ khi nó được phép phân cụm xử lý.

3.2.2 Các biến tỷ lệ khoảng cách

Phần này thảo luận các biến tỷ lệ khoảng cách và chuẩn hoá chúng. Sau đó mô tả các phép đo khoảng cách phổ biến được dùng trong tính toán độ không tương đồng của các đối tượng được mô tả bởi các biến tỷ lệ khoảng cách. Các phép đo này bao gồm các khoảng cách Euclidean, Mahattan và Minkowski.

Các biến tỷ lệ khoảng cách là các phép đo liên tục của một tỷ lệ tuyến tính thô. Các mẫu điển hình như trọng lượng và chiều cao, sự kết hợp vĩ độ và kinh độ (ví dụ khi phân cụm nhà) và nhiệt độ khí hậu.

Đơn vị phép đo đã dùng có thể ảnh hưởng đến phép phân cụm. Ví dụ, thay đổi các đơn vị đo, như thay đổi từ meter tới inche cho chiều cao hay từ kilogram

tới pound cho trọng lượng, có thể dẫn tới một cấu trúc phân cụm rất khác biệt. Nhìn chung, biểu diễn một biến dưới các đơn vị nhỏ hơn sẽ dẫn tới một phạm vi lớn hơn cho biến đó và do vậy một hiệu ứng lớn hơn trên kết quả cấu trúc phân cụm. Để tránh sự phụ thuộc vào việc lựa chọn đơn vị đo, dữ liệu nên được chuẩn hoá. Chuẩn hoá các phép đo cố gắng mang lại cho tất cả các biến một trọng số như nhau. Tuy nhiên, trong nhiều ứng dụng, người ta có thể cố ý muốn mang tới trọng số lớn hơn cho một tập các biến nào đó so với các biến khác. Ví dụ, khi phân cụm các cầu thủ chơi bóng rổ, người ta có thể thích mang tới trọng số hơn cho biến chiều cao.

Để chuẩn hoá các phép đo, một lựa chọn đó là chuyển đổi các phép đo gốc sang các biến không đơn vị (unitless). Cho trước các phép đo đối với biến *f*. Điều này có thể được biểu diễn như sau:

1. Tính trung bình độ lệch tuyệt đối s_f

$$s_f = \frac{1}{n}\left(\left|x_{1f} - m_f\right| + \left|x_{2f} - m_f\right| + \cdots + \left|x_{nf} - m_f\right|\right) \tag{3.7}$$

với x_{1f} ,..., x_{nf} là n phép đo của f, m_f là giá trị trung bình của f, tức là $m_f = \frac{1}{n}\left(x_{1f} + x_{2f} + \cdots + x_{nf}\right)$

2. Tính phép đo chuẩn hoá, gọi là z-score như sau:

$$z_{if} = \frac{x_{if} - m_f}{s_f} \tag{3.8}$$

Thuận lợi của việc sử dụng độ lệch tuyệt đối trung bình đó là z-scores của các outlier không trở nên quá nhỏ, do vậy các outlier vẫn dễ nhận thấy. Tuy nhiên lựa chọn việc chuẩn hoá và biểu diễn chuẩn hoá như thế nào là thuộc về phía người dùng.

Sau khi chuẩn hoá hay không cần chuẩn hoá trong một số ứng dụng nào đó, ta tính độ không tương đồng (hay tương đồng) giữa các đối tượng. Cho trước các biến tỷ lệ khoảng cách, dựa trên khoảng cách giữa từng cặp đối tượng. Có một số tiếp cận để định nghĩa khoảng cách giữa các đối tượng. Phép đo khoảng cách phổ biến nhất là khoảng cách Euclidean, nó được định nghĩa như sau:

$$d(i,j) = \sqrt{\left|x_{i1} - x_{j1}\right|^2 + \left|x_{i2} - x_{j2}\right|^2 + ... + \left|x_{ip} - x_{jp}\right|^2} \quad (3.9)$$

với $i = (x_{i1}, x_{i2},..., x_{ip})$ và $j = (x_{j1}, x_{j2},..., x_{jp})$ là hai đối tượng dữ liệu p chiều.

Một metric nổi tiếng khác là khoảng cách Mahattan (hay city block) được định nghĩa bởi:

$$d(i,j) = \left|x_{i1} - x_{j1}\right| + \left|x_{i2} - x_{j2}\right| + ... + \left|x_{ip} + x_{jp}\right| \quad (3.10)$$

Cả khoảng cách Euclidean và khoảng cách Mahattan thoả các yêu cầu toán học của một hàm khoảng cách:

1. $d(i,j) \geq 0$ cho biết khoảng cách là một số không âm

2. $d(i,i) = 0$ cho biết khoảng cách của một đối tượng tới chính nó thì bằng 0

3. $d(i,j) = d(j,i)$ cho biết khoảng cách là một hàm đối xứng

4. $d(i,j) \leq d(i,h) + d(h,j)$ bất đẳng thức tam giác này cho biết khoảng cách trực tiếp từ i tới j không lớn hơn khoảng cách đi theo đường vòng qua bất kỳ một điểm h nào.

Khoảng cách Minkowski là tổng quát hoá của cả hai khoảng cách Euclidean và Mahattan. Nó được định nghĩa như sau:

$$d(i,j) = (\left|x_{i1} - x_{j1}\right|^q + \left|x_{i2} - x_{j2}\right|^q + ... + \left|x_{ip} + x_{jp}\right|^q)^{1/q} \quad (3.11)$$

với q là một số nguyên dương, nó đại diện cho khoảng cách Mahattan khi $q = 1$ và Euclidean khi $q = 2$.

Nếu mỗi biến được ấn định một trọng số theo độ quan trọng nhận biết của nó, khoảng cách Euclidean được đánh trọng số có thể được tính như sau:

$$d(i,j) = \sqrt{w_1\left|x_{i1} - x_{j1}\right|^2 + w_2\left|x_{i2} - x_{j2}\right|^2 + ... + w_p\left|x_{ip} - x_{jp}\right|^2} \quad (3.12)$$

Đánh trọng số cũng được áp dụng cho khoảng cách Mahattan và Minkowski.

3.2.3 Các biến nhị phân

Phần này mô tả làm thế nào để tính toán độ không tương đồng giữa các đối tượng được mô tả bởi các biến nhị phân đối xứng hoặc không đối xứng.

Một biến nhị phân chỉ có hai trạng thái 0 hay 1, với 0 là biến vắng mặt, 1 là biến có mặt. Cho trước biến *hút thuốc* mô tả một bệnh nhân, ví dụ, 1 chỉ rằng bệnh nhân hút thuốc, 0 cho biết bệnh nhân không hút thuốc. Xử lý các biến nhị phân giống như các biến tỷ lệ khoảng cách có thể dẫn tới lạc lối các kết quả phân cụm. Bởi vậy, các phương pháp chỉ định cho dữ liệu nhị phân cần phải tính toán độ không tương đồng.

Một tiếp cận để tính toán ma trận không tương đồng từ dữ liệu nhị phân đã cho. Nếu tất cả các biến nhị phân được xem như là có cùng trọng số, ta có bảng ngẫu nhiên 2 x 2, bảng 3.1, với a là số các biến bằng 1 cho cả hai đối tượng i và j, b là số các biến bằng *1* cho đối tượng i và 0 cho đối tượng j, c là số các biến bằng 0 cho đối tượng i và 1 cho đối tượng j, d là số các biến bằng 0 cho cả đối tượng i và j. Tổng số lượng của các biến là p, $p = a + b + c + d$.

Bảng 3.1: Bảng ngẫu nhiên cho các biến nhị phân

đối tượng *i* \ đối tượng *j*	1	0	tổng
1	a	b	$a + b$
0	c	d	$c + d$
tổng	$a + c$	$b + d$	p

Một biến nhị phân là đối xứng nếu như cả hai trạng thái của nó có cùng trị giá và mang cùng trọng số, do vậy không có sự ưu tiên nên kết quả mã hoá là 0 hay 1. Ví dụ, *giới tính* có thể là *nam* hay *nữ*. Độ tương đồng dựa trên các biến nhị phân đối xứng được gọi là độ tương đồng bất biến trong đó kết quả không thay đổi khi một số hay tất cả các biến nhị phân được mã hoá khác nhau. Đối với các độ đo tương đồng bất biến, hệ số được biết đến nhiều nhất là hệ số đối sánh đơn giản (simple matching coefficient) được định nghĩa trong (3.13).

$$d(i, j) = \frac{b + c}{a + b + c + d} \tag{3.13}$$

Một biến nhị phân là không đối xứng nếu như kết quả của các trạng thái quan trọng không bằng nhau. Ta sẽ mã hoá như sau: kết quả có tầm quan trọng nhất là 1 (ví dụ *dương tính* HIV) và những cái còn lại bằng 0 (ví dụ như *âm tính*

HIV). Một biến nhị phân như vậy được xem như là "biến unary". Độ tương đồng dựa trên các biến đó được gọi là độ tương đồng không bất biến. Đối với các độ tương đồng không bất biến, hệ số được biết đến nhiều nhất là hệ số Jaccard, được định nghĩa trong (3.14), tại đó các đối sánh âm *d* được xem là không quan trọng và do vậy đã bị lờ đi khi tính toán.

$$d(i,j) = \frac{b+c}{a+b+c} \tag{3.14}$$

Khi cả biến nhị phân đối xứng và không đối xứng xuất hiện trong cùng tập dữ liệu, tiếp cận các biến pha trộn được mô tả trong mục 3.2.5 có thể được áp dụng.

Ví dụ 3.1 *Độ không tương đồng giữa các biến nhị phân:* Giả sử rằng một bảng các bản ghi bệnh nhân, bảng 3.2 chứa các thuộc tính *tên*, *giới tính*, *sốt*, *ho*, test-1,test-2,test-3 và test-4 (test: xét nghiệm), với *tên* là một object-id, *giới tính* là một thuộc tính đối xứng và các thuộc tính còn lại là không đối xứng.

Bảng 3.2: Bảng quan hệ chứa hầu hết các thuộc tính nhị phân

Tên	Giới tính	Sốt	Ho	test-1	test-2	test-3	test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Đối với các giá trị thuộc tính không đối xứng, cho các giá trị Y và P là 1; N là 0. Giả sử rằng khoảng cách giữa các đối tượng (bệnh nhân) được tính toán dựa trên chỉ các biến không đối xứng. Theo công thức hệ số Jaccard (3.14), khoảng cách giữa mỗi cặp 3 bệnh nhân, Jack, Mary và Jim sẽ là:

$$d(jack, mary) = \frac{b+c}{a+b+c} = \frac{0+1}{2+0+1} = 0.33 \tag{3.15}$$

$$d(jack, jim) = \frac{b+c}{a+b+c} = \frac{1+1}{1+1+1} = 0.67 \tag{3.16}$$

$$d(jim, mary) = \frac{b+c}{a+b+c} = \frac{1+2}{1+1+2} = 0.75 \tag{3.17}$$

Các phép đo này cho thấy Jim và Mary không có hứa hẹn là có bệnh giống nhau. Trong 3 bệnh nhân này, Jack và Mary có thể có bệnh giống nhau nhất.

3.2.4 Các biến tên, có thứ tự và dựa trên tỷ lệ

Phần này thảo luận làm thế nào để tính độ không tương đồng giữa các đối tượng được mô tả bởi các biến tên, có thứ tự và dựa trên tỷ lệ.

- *Các biến tên:*

Biến *tên* là sự suy rộng của biến nhị phân, trong đó nó có thể mang nhiều hơn hai trạng thái. Ví dụ, *bản đồ màu* là một biến tên có thể có 5 trạng thái: *đỏ*, *vàng*, *xanh lá cây*, *hồng* và *xanh da trời*.

Cho số các trạng thái của một biến *tên* là *M*. Các trạng thái có thể được chỉ ra bởi các ký tự, các biểu tượng hay một tập các số nguyên như 1,2,...,*M*. Lưu ý rằng các số nguyên như thế này chỉ được dùng cho dữ liệu điều khiển và không đại diện cho bất kỳ một trật tự cụ thể nào.

Độ không tương đồng giữa hai đối tượng *i* và *j* có thể được tính bằng cách sử dụng tiếp cận đối sánh đơn giản như trong (3.18):

$$d(i,j) = \frac{p-m}{p} \tag{3.18}$$

với *m* là số lượng các đối sánh (tức là số lượng các biến mà *i* và *j* có cùng trạng thái) và *p* là tổng số của các biến. Các trọng số có thể được ấn định để làm tăng hiệu quả của *m*, hay ấn định trọng số lớn hơn cho các đối sánh trong các biến có số lượng các trạng thái lớn hơn.

Các biến tên có thể được mã hoá bởi một số lượng lớn các biến nhị phân không đối xứng bằng cách tạo một biến nhị phân mới cho mỗi trạng thái *tên*. Đối với một đối tượng với giá trị trạng thái cho trước, biến nhị phân miêu tả trạng thái đó đặt là 1, trong khi các biến nhị phân còn lại đặt là 0. Ví dụ, để mã hoá biến tên *bản đồ màu*, một biến nhị phân có thể được tạo lập cho từng màu trong danh sách 5 màu trên. Cho một đối tượng có màu *vàng*, biến *vàng* đặt là 1, trong khi bốn biến còn lại đặt là 0. Hệ số không tương đồng cho dạng này khi mã hoá được tính như các phương pháp trong mục 3.2.3.

- *Các biến có thứ tự:*

Biến có thứ tự rời rạc tương tự như một biến tên, loại trừ M trạng thái của giá trị có thứ tự được sắp xếp theo một trật tự có nghĩa. Các biến có thứ tự rất hữu ích cho việc thể hiện các đánh giá chất lượng một cách chủ quan mà không thể đo được bằng cách khách quan. Một biến có thứ tự liên tục trông giống như một tập dữ liệu liên tục với một tỷ lệ chưa biết, đó là mối quan hệ có thứ tự của các giá trị, là yếu tố cần thiết nhưng không phải là tính chất trọng yếu thực sự của chúng. Ví dụ, sắp xếp quan hệ trong một môn thể thao đặc thù thường cần thiết hơn các giá trị thực tế của một độ đo đặc thù. Các biến có thứ tự có thể cũng đạt được từ việc rời rạc hoá các con số tỷ lệ khoảng cách bằng cách chia phạm vi giá trị vào trong một số các lớp hữu hạn. Các giá trị của một biến có thứ tự có thể được ánh xạ tới các hạng (rank). Giả sử rằng một biến có thứ tự f có M_f trạng thái. Các trạng thái được sắp xếp định nghĩa có thứ tự là $1,...,M_f$.

Nghiên cứu các biến tên hoàn toàn giống với nghiên cứu các biến tỷ lệ khoảng cách khi tính toán độ không tương đồng giữa các đối tượng. Giả sử f là một biến trong tập các biến có thứ tự mô tả n đối tượng. Độ không tương đồng tính toán đối với f bao gồm các bước sau:

1. Giá trị của f cho đối tượng thứ i là x_{if} và f có M_f trạng thái đã được sắp xếp, miêu tả bởi thứ tự $1,...,M_f$. Thay thế mỗi x_{if} bởi hạng (rank) tương ứng của nó $r_{if} \in \{1,...,M_f\}$.

2. Từ đó mỗi một biến có thứ tự có một số lượng các trạng thái khác nhau, ánh xạ phạm vi của mỗi biến lên trên [0-1] bằng cách thay thế hạng r_{if} của đối tượng thứ i trong biến thứ f bởi

$$z_{if} = \frac{r_{if} - 1}{M_f - 1} \tag{3.19}$$

3. Tính độ không tương đồng, sử dụng bất kỳ độ đo khoảng cách nào đã mô tả trong mục 3.2.2, sử dụng z_{if} đại diện cho giá trị f cho đối tượng thứ i.

- *Các biến dựa trên tỷ lệ:*

Một biến dựa trên tỷ lệ làm một phép đo dương trên một tỷ lệ không tuyến tính, như tỷ lệ số mũ, xấp xỉ công thức dưới đây:

$$Ae^{Bt} \text{ hay } Ae^{-Bt} \quad (3.20)$$

với A và B là các hằng số dương.

Có ba phương pháp sử dụng các biến dựa trên tỷ lệ để việc tính độ không tương đồng giữa các đối tượng.

1. Xử lý các biến dựa trên tỷ lệ giống như các biến tỷ lệ khoảng cách. Tuy nhiên điều này không phải luôn là lựa chọn tốt bởi tỷ lệ có thể bị bóp méo.

2. Áp dụng phép biến đổi loga cho một biến dựa trên tỷ lệ f có giá trị x_{if} cho đối tượng i bằng cách sử dụng công thức $y_{if} = \log(x_{if})$. Các giá trị y_{if} được xử lý như giá trị tỷ lệ khoảng cách trong mục 3.2.2. Lưu ý rằng đối với nhiều biến dựa trên tỷ lệ, ta cũng có thể áp dụng phép biến đổi log hay các phép biến đổi khác, tuỳ thuộc vào định nghĩa và ứng dụng.

3. Xử lý x_{if} như dữ liệu có thứ tự liên tục và xử lý các hạng của chúng như giá trị tỷ lệ khoảng cách.

Hai phương pháp sau có hiệu quả nhất, mặc dầu việc lựa chọn phương pháp để dùng còn phụ thuộc vào ứng dụng cho trước.

3.2.5 Các biến có sự pha trộn của các kiểu

Mục 3.2.2 tới 3.2.4 đã đưa ra cách tính độ không tương đồng giữa các đối tượng được mô tả bởi các biến cùng kiểu, tại đó, các kiểu này có thể là tỷ lệ khoảng cách, nhị phân đối xứng, nhị phân không đối xứng, tên, có thứ tự hay dựa trên tỷ lệ. Tuy nhiên, trong nhiều cơ sở dữ liệu thực, các đối tượng được mô tả bởi một sự pha trộn các kiểu biến. Nhìn chung, một cơ sở dữ liệu có thể chứa tất cả 6 kiểu biến trong danh sách trên. Ta cần một phương pháp để tính độ không tương đồng giữa các đối tượng của các kiểu biến hỗn hợp.

Một tiếp cận là nhóm mỗi loại biến với nhau, thực hiện một phép phân tích cụm riêng biệt cho mỗi kiểu biến. Điều này là khả thi nếu như các phép phân tích này nhận được các kết quả thích hợp. Tuy nhiên, trong các ứng dụng thực,

thường không thể xảy ra một phép phân tích cụm tách biệt cho mỗi kiểu biến sẽ sinh ra các kết quả thích hợp.

Một tiếp cận được ưa thích hơn là xử lý tất cả các kiểu biến với nhau, thực hiện một phép phân cụm đơn. Một kỹ thuật như vậy được đề xuất bởi (Ducker et al. 1965) và mở rộng bởi (Kaufman and Rousseeuw 1990) kết hợp các biến khác nhau vào trong một ma trận không tương đồng và mang tất cả các biến có nghĩa lên trên một tỷ lệ chung trong khoảng [0,1].

Giả sử rằng tập dữ liệu chứa *p* biến kiểu hỗn hợp. Độ không tương đồng $d(i,j)$ giữa đối tượng *i* và *j* được định nghĩa:

$$d(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}} \qquad (3.21)$$

với indicator $\delta_{ij}^{(f)} = 0$ nếu x_{if} hoặc x_{jf} khuyết (tức là không có phép đo của biến *f* cho đối tượng *i* hay đối tượng *j*) hoặc (2) $x_{if} = x_{jf} = 0$ và biến *f* là nhị phân không đối xứng, các trường hợp còn lại $\delta_{ij}^{(f)} = 1$. $d_{ij}^{(f)}$ được tính toán tuỳ thuộc vào kiểu của nó:

1. Nếu *f* là nhị phân hay tên: $d_{ij}^{(f)} = 0$ nếu $x_{if} = x_{jf}$, các trường hợp còn lại $d_{ij}^{(f)} = 1$

2. Nếu *f* là tỷ lệ khoảng cách: $d_{ij}^{(f)} = \frac{|x_{if} - x_{jf}|}{\max_h x_{hf} - \min_h x_{hf}}$ với *h* chạy quá tất cả các đối tượng không khuyết đối với biến *f*.

3. Nếu *f* là có thứ tự hay dựa trên tỷ lệ: tính toán các hạng r_{if} và $z_{if} = \frac{r_{if} - 1}{M_f - 1}$ và xem xét z_{if} như tỷ lệ khoảng cách.

Do đó, độ không tương đồng giữa các đối tượng được tính ngay cả khi các biến mô tả các đối tượng có kiểu khác nhau.

3.3 Phân loại các phương pháp phân cụm chính

Hiện có một số lượng lớn các giải thuật phân cụm trong các tài liệu. Việc lựa chọn giải thuật phân cụm tuỳ thuộc vào kiểu dữ liệu cho sẵn, mục đích riêng

và ứng dụng. Nếu như phép phân tích cụm được dùng như một công cụ mô tả hay thăm dò thì có thể thử một vài giải thuật trên cùng dữ liệu để xem xem dữ liệu có thể thể hiện được điều gì.

Nhìn chung, các phương pháp phân cụm chính được phân thành các loại sau:

1. Các phương pháp phân chia:

Cho trước một cơ sở dữ liệu với n đối tượng hay các bộ dữ liệu, một phương pháp phân chia được xây dựng để chia dữ liệu thành k phần, mỗi phần đại diện cho một cụm, $k \leq n$. Đó là phân loại dữ liệu vào trong k nhóm, chúng thoả các yêu cầu sau: (1) Mỗi nhóm phải chứa ít nhất một đối tượng, (2) Mỗi đối tượng phải thuộc về chính xác một nhóm. Lưu ý rằng yêu cầu thứ 2 được nới lỏng trong nhiều kỹ thuật phân chia mờ sẽ được thảo luận ngắn gọn trong chương này.

Cho trước k là số lượng các phần chia cần xây dựng, phương pháp phân chia tạo lập phép phân chia ban đầu. Sau đó nó dùng kỹ thuật lặp lại việc định vị, kỹ thuật này cố gắng cải thiện sự phân chia bằng cách gỡ bỏ các đối tượng từ nhóm này sang nhóm khác. Tiêu chuẩn chung của một phân chia tốt là các đối tượng trong cùng cụm là "gần" hay có quan hệ với nhau, ngược lại, các đối tượng của các cụm khác nhau lại "tách xa" hay rất khác nhau. Có nhiều tiêu chuẩn khác nhau để đánh giá chất lượng các phép phân chia.

Trong phân cụm dựa trên phép phân chia, hầu hết các ứng dụng làm theo một trong hai phương pháp heuristic phổ biến: (1) Giải thuật *k-means* với mỗi cụm được đại diện bởi giá trị trung bình của các đối tượng trong cụm; (2) Giải thuật *k-medoids* với mỗi cụm được đại diện bởi một trong số các đối tượng định vị gần tâm của cụm. Các phương pháp phân cụm heuristic này làm việc tốt khi tìm kiếm các cụm có hình cầu trong các cơ sở dữ liệu có kích thước từ nhỏ tới trung bình. Để tìm ra các cụm với các hình dạng phức tạp và phân cụm cho các tập dữ liệu rất lớn, các phương pháp dựa trên phân chia cần được mở rộng. Các

phương pháp phân cụm dựa trên phân chia được nghiên cứu sâu hơn trong mục 3.4.

2. Các phương pháp phân cấp:

Một phương pháp phân cấp tạo một phân tích phân cấp tập các đối tượng dữ liệu đã cho. Một phương pháp phân cấp có thể được phân loại như tích đống hay phân chia, dựa trên việc phân ly phân cấp được hình thành như thế nào. Tiếp cận tích đống còn được gọi là tiếp cận "bottom - up", lúc đầu mỗi đối tượng lập thành một nhóm riêng biệt. Nó hoà nhập lần lượt các đối tượng hay các nhóm gần nhau với nhau cho tới khi tất cả các nhóm được hoà nhập thành một (mức cao nhất của hệ thống phân cấp), hay cho tới khi một gặp một điều kiện kết thúc. Tiếp cận phân ly còn được gọi là tiếp cận "top - down", lúc đầu tất cả các đối tượng trong cùng một cụm. Trong mỗi lần lặp kế tiếp, một cụm được chia vào trong các cụm nhỏ hơn cho tới khi cuối cùng mỗi đối tượng trong một cụm hay cho tới khi gặp một điều kiện kết thúc.

Sự kết hợp của việc lặp lại việc định vị và phân ly phân cấp sẽ thuận lợi bởi trước tiên sử dụng giải thuật phân ly phân cấp và sau đó cải tiến kết quả sử dụng định vị lặp. Nhiều giải thuật phân cụm mở rộng như BIRCH và CURE được phát triển dựa trên một tiếp cận tích hợp như vậy. Các phương pháp phân cụm phân cấp được nghiên cứu trong mục 3.5.

3. Các phương pháp dựa trên mật độ:

Hầu hết các phương pháp phân chia cụm các đối tượng dựa trên khoảng cách giữa các đối tượng. Các phương pháp như vậy có thể chỉ tìm được các cụm có hình cầu và sẽ gặp khó khăn khi các cụm đang khám phá lại có hình dạng tuỳ ý. Các phương pháp phân cụm được phát triển dựa trên khái niệm mật độ. Ý tưởng chung đó là tiếp tục phát triển cụm cho trước với điều kiện là mật độ (số các đối tượng hay các điểm dữ liệu) trong "lân cận" vượt quá ngưỡng, tức là đối với mỗi điểm dữ liệu trong phạm vi một cụm cho trước thì lân cận trong vòng bán kính đã cho chứa ít nhất một số lượng điểm tối thiểu. Một phương pháp như

vậy có thể được dùng để lọc ra nhiễu (các outlier) và khám phá ra các cụm có hình dạng bất kỳ.

DBSCAN là một phương pháp dựa trên mật độ điển hình, nó tăng trưởng các cụm theo một ngưỡng mật độ. OPTICS là một phương pháp dựa trên mật độ, nó tính toán một thứ tự phân cụm tăng dần cho phép phân tích cụm tự động và tương tác. Các phương pháp phân cụm dựa trên mật độ được nghiên cứu trong mục 3.6.

4. Các phương pháp dựa trên lưới:

Một phương pháp dựa trên lưới lượng tử hoá không gian đối tượng vào trong một số hữu hạn các ô hình thành nên một cấu trúc lưới. Sau đó nó thực hiện tất cả các thao tác phân cụm trên cấu trúc lưới (tức là trên không gian đã lượng tử hoá). Thuận lợi chính của tiếp cận này là thời gian xử lý nhanh chóng của nó độc lập với số các đối tượng dữ liệu và chỉ tuỳ thuộc vào số lượng các ô trong mỗi chiều của không gian lượng tử.

STING là một ví dụ điển hình của phương pháp dựa trên lưới. WaveCluster và CLIQUE là hai giải thuật phân cụm dựa trên cả lưới và mật độ. Các phương pháp phân cụm dựa trên lưới được nghiên cứu trong mục 3.7.

Nhiều giải thuật phân cụm tích hợp các ý tưởng của một vài phương pháp phân cụm, bởi vậy việc phân loại giải thuật đó không dễ như loại giải thuật chỉ phụ thuộc vào duy nhất một loại phương pháp phân cụm. Hơn nữa, nhiều ứng dụng có thể có giới hạn phân cụm với yêu cầu tích hợp một số kỹ thuật phân cụm.

Trong mục dưới đây ta xem xét từng phương pháp phân cụm trên một cách chi tiết. Các giải thuật tích hợp các ý tưởng của một số phương pháp phân cụm cũng được giới thiệu.

3.4 Các phương pháp phân chia

Cho trước một cơ sở dữ liệu với n đối tượng, k là số các cụm cần thiết lập, một giải thuật phân chia tổ chức các đối tượng vào trong k phần phân chia ($k \leq n$), với mỗi một phần phân chia đại diện cho một cụm. Các cụm được thiết lập

theo một tiêu chuẩn phân chia khách quan, thường được gọi là một hàm tương đồng, như khoảng cách, để các đối tượng trong phạm vi một cụm là "giống nhau", ngược lại, các đối tượng của các cụm khác nhau là "không giống nhau" về mặt các thuộc tính cơ sở dữ liệu.

3.4.1 Các phương pháp phân chia kinh điển: k-means và k-medoids

Các phương pháp phân chia nổi tiếng và thường được dùng nhất là *k-means* (MacQueen 1967), *k-medoids* (Kaufman và Rousseew 1987) và các dạng biến đổi của chúng.

3.4.1.1 Kỹ thuật dựa trên trọng tâm: phương pháp k-means

Giải thuật *k-means* lấy tham số đầu vào *k* và phân chia một tập *n* đối tượng vào trong *k* cụm để cho kết quả độ tương đồng trong cụm là cao trong khi độ tương đồng ngoài cụm là thấp. Độ tương đồng cụm được đo khi đánh giá giá trị trung bình của các đối tượng trong cụm, nó có thể được quan sát như là "trọng tâm" của cụm.

Giải thuật xử lý như sau: trước tiên nó lựa chọn ngẫu nhiên *k* đối tượng, mỗi đối tượng đại diện cho một trung bình cụm hay tâm cụm. Đối với những đối tượng còn lại, một đối tượng được ấn định vào một cụm mà nó giống nhất dựa trên khoảng cách giữa đối tượng và trung bình cụm. Sau đó cần tính giá trị trung bình mới cho mỗi cụm. Xử lý này được lặp lại cho tới khi hàm tiêu chuẩn hội tụ. Bình phương sai số tiêu chuẩn thường được dùng, định nghĩa như sau:

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} |x - m_i|^2 \qquad (3.22)$$

với x là điểm trong không gian, đại diện cho đối tượng cho trước, m_i là trung bình cụm C_i (cả x và m_i đều là nhiều chiều). Tiêu chuẩn này cố gắng cho kết quả k cụm càng đặc, càng riêng biệt càng tốt. Thủ tục *k-means* được tóm tắt trong hình 3.1.

Giải thuật xác định *k* phần phân chia thoả mãn tối thiểu hoá bình phương hàm sai số. Nó làm việc tốt khi các cụm là các đám mây đặc tách biệt so với những cụm khác. Phương pháp này có thể mở rộng có hiệu quả khi xử lý các tập dữ liệu lớn bởi độ phức tạp tính toán của giải thuật là $O(nkt)$, với n là số đối

tượng, k là số cụm, t là số lần lặp. Thông thường $k << n$ và $t << n$. Phương pháp thường kết thúc tại một điểm tối ưu cục bộ.

Giải thuật 3.4.1 (*k-means*) Giải thuật *k-means* đối với việc phân chia dựa trên giá trị trung bình của các đối tượng trong cụm.

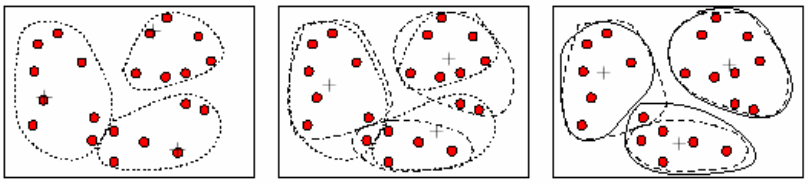
Đầu vào: Số cụm k và một cơ sở dữ liệu chứa n đối tượng.

Đầu ra: Một tập k cụm - cụm tối thiểu hoá bình phương sai số tiêu chuẩn.

Giải thuật:

1) Chọn tuỳ ý k đối tượng với tư cách là các tâm cụm ban đầu
2) **repeat**
3) Ấn định (lại) mỗi đối tượng về một cụm mà đối tượng đó giống nhất, dựa trên giá trị trung bình của các đối tượng trong cụm;
4) Cập nhật các trung bình cụm, tức là tính giá trị trung bình của các đối tượng trong cụm đó;
5) **Until** không có sự thay đổi nào;

Hình 3.1: Giải thuật *k-means*



Hình 3.2: Phân cụm một tập các điểm dựa trên phương pháp *k-means*

Tuy nhiên, phương pháp *k-means* chỉ áp dụng khi trung bình của một cụm được xác định. Không phải ứng dụng nào cũng có thể áp dụng kỹ thuật này, ví dụ những dữ liệu bao hàm các thuộc tính xác thực. Về phía các user, họ phải chỉ rõ k - số cụm, cần sớm phát hiện ra sự bất lợi. Phương pháp *k-means* không thích hợp với việc tìm các cụm có hình dáng không lồi hay các cụm có kích thước khác xa nhau. Hơn nữa, nó nhạy cảm với các điểm dữ liệu nhiễu và outlier, một số lượng nhỏ dữ liệu như vậy về căn bản có ảnh hưởng tới giá trị trung bình.

Ví dụ 3.2: Giả sử có một tập đối tượng được định vị trong một hình chữ nhật như hình 3.2. Cho $k = 3$, người dùng cần phải phân cụm các đối tượng vào trong 3 cụm.

Theo giải thuật 3.4.1, ta chọn 3 đối tượng tuỳ ý (đánh dấu là "+") với vai trò là 3 tâm cụm đầu tiên. Sau đó, mỗi đối tượng được phân vào trong các cụm đã chọn dựa trên tâm cụm gần nhất. Mỗi phân bố hình thành nên một hình chiếu được bao quanh bởi đường cong nét chấm, hình 3.2 a).

Cập nhật lại các tâm cụm. Đó là giá trị trung bình của mỗi cụm được tính toán lại dựa trên các đối tượng trong cụm. Tuỳ theo các tâm mới này, các đối tượng được phân bố lại vào trong các cụm đã lựa chọn dựa trên tâm cụm gần nhất. Mỗi phân bố lại hình thành nên một hình chiếu được bao quanh bởi đường cong nét gạch, hình 3.2 b).

Xử lý này lặp lại dẫn tới hình 3.2 c). Cuối cùng, không có sự phân bố lại các đối tượng vào trong bất kỳ cụm nào, và xử lý kết thúc. Các cụm cuối cùng là kết quả của xử lý phân cụm.

Một biến thể khác của *k-means* là phương pháp *k-modes* (Huang 1998) - mở rộng mô hình *k-means* - để phân cụm dữ liệu xác thực bằng cách thay giá trị trung bình các cụm bằng các *mode (chế độ hay kiểu)*, sử dụng độ đo không tương đồng mới để giải quyết đối tượng xác thực, sử dụng phương pháp dựa trên tần số để cập nhật các *mode* của các cụm. Phương pháp *k-means* và *k-modes* có thể được tích hợp để phân cụm dữ liệu với các giá trị hỗn hợp số và xác thực, người ta gọi đó là phương pháp *k-prototypes*.

Một biến thể khác của *k-means* đó là giải thuật EM (Expectation Maximization) (Lauritzen 1995), nó mở rộng mô hình *k-means* theo một cách khác: Thay vì ấn định mỗi điểm tới một cụm cho trước, nó ấn định mỗi điểm tới một cụm theo trọng số đại diện cho xác suất là thành viên. Hay nói một cách khác, không có các ranh giới tuyệt đối giữa các cụm. Bởi vậy, các giá trị trung bình mới sau đó được tính dựa trên các phép đo có trọng số.

3.4.1.2 Kỹ thuật dựa trên điểm đại diện: phương pháp k-medoids

Giải thuật *k-means* rất nhạy với các outlier, do vậy một đối tượng với giá trị cực lớn về cơ bản có thể bóp méo phân bố của dữ liệu. Thay vì lấy giá trị trung bình của các đối tượng trong một cụm như một điểm tham khảo, *k-medoids* lấy một đối tượng đại diện trong cụm, gọi là *medoid*, nó là điểm đại diện được định vị trung tâm nhất trong cụm. Do vậy, phương pháp phân chia vẫn được thực hiện dựa trên nguyên tắc tối thiểu hoá tổng của các độ không tương đồng giữa mỗi đối tượng với điểm tham khảo tương ứng của nó, điểm này thiết lập nên cơ sở của phương pháp *k-medoids*.

PAM (partition around medoids)- *phân chia xung quanh các medoid:*

Đây là một giải thuật phân cụm kiểu *k-medoids*. Nó tìm *k* cụm trong *n* đối tượng bằng cách trước tiên tìm một đối tượng đại diện (*medoid*) cho mỗi cụm. Tập các *medoid* ban đầu được lựa chọn tuỳ ý. Sau đó nó lặp lại các thay thế một trong số các *medoid* bằng một trong số những cái không phải *medoid* miễn là tổng khoảng cách của kết quả phân cụm được cải thiện.

Giải thuật chi tiết của PAM được trình bày trong hình 3.3. Giải thuật thử xác định *k* phần phân chia cho *n* đối tượng. Sau khi lựa chọn được *k-medoids* ban đầu, giải thuật lặp lại việc thử để có một sự lựa chọn các *medoid* tốt hơn bằng cách phân tích tất cả các cặp đối tượng có thể để một đối tượng là *medoid* và đối tượng kia thì không phải. Phép đo chất lượng phân cụm được tính cho mỗi sự kết hợp như vậy. Lựa chọn các điểm tốt nhất trong một lần lặp được chọn với tư cách là các *medoid* cho lần lặp tiếp theo. Chi phí của một lần lặp đơn là $O(k(n - k)^2)$. Đối với các giá trị *n* và *k* lớn, chi phí tính toán như vậy có thể là cao.

Giải thuật 3.4.2: Giải thuật *k-medoids* đối với việc phân chia dựa trên các đối tượng trung tâm

Đầu vào: Số cụm *k* và một cơ sở dữ liệu chứa *n* đối tượng

Đầu ra: Một tập *k* cụm đã tối thiểu hoá tổng các độ đo không tương đồng của tất cả các đối tượng tới *medoid* gần nhất của chúng.

Giải thuật:

1) Chọn tuỳ ý *k* đối tượng giữ vai trò là các *medoid* ban đầu;

2) **repeat**

3) Ấn định mỗi đối tượng vào cụm có *medoid* gần nó nhất;

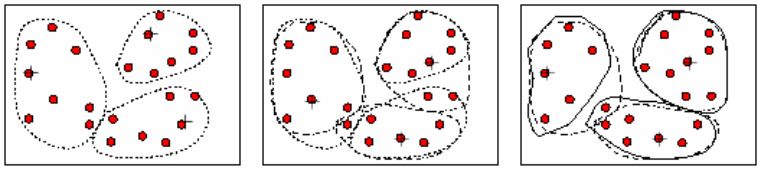
4) Tính hàm mục tiêu - là tổng các độ đo không tương đồng của tất cả các đối tượng tới *medoid* gần nhất của chúng;

5) Đổi *medoid x* bằng một đối tượng *y* nếu như việc thay đổi này làm giảm hàm mục tiêu;

6) **until** không có sự thay đổi nào;

Hình 3.3: Giải thuật *k-medoids*

Ví dụ 3.3: Giả sử có một tập đối tượng được định vị trong một hình chữ nhật được biểu diễn như hình 3.4. Cho $k = 3$, tức là người dùng cần phân các đối tượng vào trong 3 cụm.



Hình 3.4: Phân cụm một tập các điểm dựa trên phương pháp *k-medoids*

Theo giải thuật 3.4.2, ta chọn 3 đối tượng tuỳ ý (đánh dấu "+") với vai trò là 3 tâm cụm ban đầu. Sau đó mỗi đối tượng được phân bố vào các cụm đã chọn dựa trên tâm cụm gần nó nhất. Một phân bố như vậy hình thành nên một hình chiếu được bao quanh bởi đường cong nét chấm, hình 3.2 a).

Kiểu nhóm này sẽ cập nhật các tâm cụm. Đó là *medoid* của mỗi cụm được tính lại dựa trên các đối tượng trong cụm. Với các tâm mới, các đối tượng được phân bố lại tới các cụm đã chọn dựa trên tâm cụm gần nhất. Sự phân bố lại này thiết lập một hình chiếu mới bởi đường cong nét đứt, hình 3.4 b).

Lặp lại việc xử lý này để dẫn tới hình 3.4 c). Cuối cùng, không xảy ra sự phân bố lại các đối tượng trong bất kì cụm nào và xử lý kết thúc. Các cụm cuối cùng là kết quả của xử lý phân cụm.

Khi có sự hiện diện của nhiễu và các outlier, phương pháp *k-medoids* mạnh hơn *k-means* bởi so với giá trị trung bình (*mean*), *medoid* ít bị ảnh hưởng hơn bởi các outlier hay các giá trị ở rất xa khác nữa. Tuy nhiên, xử lý của nó có chi phí tốn kém hơn phương pháp *k-means* và nó cũng cần người dùng chỉ ra *k* - số cụm.

3.4.2 Các phương pháp phân chia trong các cơ sở dữ liệu lớn: từ k-medoids tới CLARANS

Giải thuật phân chia *k-medoids* điển hình như PAM làm việc hiệu quả đối với các tập dữ liệu nhỏ nhưng không có khả năng mở rộng tốt đối với các tập dữ liệu lớn. Để giải quyết với các tập dữ liệu lớn, một phương pháp dựa trên việc lấy mẫu gọi là CLARA (Clustering large applications) đã được phát triển bởi Kaufman và Rousseeuw, 1990.

Ý tưởng của CLARA như sau: thay vì lấy toàn bộ tập dữ liệu vào xem xét, chỉ một phần nhỏ dữ liệu thực được chọn với vai trò là một đại diện của dữ liệu, và các *medoid* được chọn từ mẫu này bằng cách sử dụng PAM. Nếu như mẫu được chọn lựa khá ngẫu nhiên, nó đại diện phù hợp cho toàn bộ tập dữ liệu, và các đối tượng đại diện (các *medoid*) được chọn do vậy sẽ giống với những cái được chọn lựa từ toàn bộ tập dữ liệu. CLARA đưa ra nhiều mẫu của tập dữ liệu, áp dụng PAM trên từng mẫu, và mang lại phân cụm tốt nhất cho đầu ra. Đúng như trông chờ, CLARA có thể giải quyết với các tập dữ liệu lớn hơn PAM. Độ phức tạp của mỗi lần lặp bây giờ trở thành $O(kS^2+k(n - k))$, với S là kích thước mẫu, k là số cụm, n là tổng số các điểm.

Hiệu quả của CLARA tuỳ thuộc vào kích thước mẫu. Lưu ý rằng PAM tìm kiếm cho *k medoids* tốt nhất giữa một tập dữ liệu cho trước, trong khi đó CLARA tìm kiếm cho *k medoids* tốt nhất giữa các mẫu đã lựa chọn của tập dữ liệu. CLARA không thể tìm được phân cụm tốt nhất nếu như bất kỳ một *medoid* được lấy mẫu không nằm trong *k medoids* tốt nhất. Ví dụ, nếu như một đối tượng O_i là một trong số các *medoid* trong *k medoids* tốt nhất nhưng nó không được chọn trong suốt quá trình lấy mẫu, CLARA sẽ không bao giờ tìm thấy

phân cụm tốt nhất. Một phân cụm tốt dựa trên các mẫu chưa chắc đã đại diện cho một phân cụm tốt cho toàn bộ tập dữ liệu nếu mẫu bị lệch (bias).

Để cải thiện chất lượng và khả năng mở rộng của CLARA, một giải thuật phân cụm khác gọi là CLARANS (Clustering Large Applications based upon RANdomized Search) được giới thiệu bởi Ng và Han, 1994. Nó cũng là một giải thuật kiểu *k-medoids* và kết hợp kỹ thuật lấy mẫu với PAM. Tuy vậy, không giống như CLARA, CLARANS không hạn chế bản thân nó cho bất kỳ một mẫu nào tại bất kỳ thời điểm nào cho trước. Trong khi đó CLARA lại có một mẫu được ấn định tại mọi giai đoạn tìm kiếm, CLARANS đưa ra một mẫu một cách ngẫu nhiên trong mỗi bước tìm kiếm. Xử lý phân cụm được thực hiện như tìm kiếm một đồ thị tại mọi nút là giải pháp tiềm năng, tức là một tập *k-medoids*. Phân cụm có được sau khi thay thế một *medoid* được gọi là láng giềng của phân cụm hiện thời. Số lượng các láng giềng được thử ngẫu nhiên bị hạn chế bởi một tham số. Nếu như một láng giềng tốt hơn được tìm thấy, CLARANS di chuyển tới láng giềng đó và xử lý lại bắt đầu lại; ngược lại, phân cụm hiện thời đưa ra một tối ưu cục bộ. Nếu như tối ưu cục bộ được tìm thấy, CLARANS bắt đầu với các nút được chọn lựa ngẫu nhiên mới để tìm kiếm một tối ưu cục bộ mới. Bằng thực nghiệm, CLARANS đã chỉ ra là hiệu quả hơn PAM và CLARA. Độ phức tạp tính toán của mỗi lần lặp trong CLARANS tỷ lệ tuyến tính với số lượng các đối tượng. CLARANS có thể được dùng để tìm số lượng lớn nhất các cụm tự nhiên sử dụng hệ số hình chiếu - đây là một đặc tính của các outlier, tức là các điểm mà không thuộc về bất kỳ cụm nào. Việc biểu diễn của giải thuật CLARANS có thể được cải thiện xa hơn nữa bằng cách khảo sát các cấu trúc dữ liệu không gian, như R*-trees, và nhiều kỹ thuật tập trung được có mặt trong các bài báo của Ester, Kriegel và Xu 1995.

3.5 Các phương pháp phân cấp

Phương pháp phân cụm phân cấp làm việc bằng cách nhóm các đối tượng dữ liệu vào trong một cây các cụm. Các phương pháp phân cụm phân cấp có thể được phân loại xa hơn trong phân cụm phân cấp tích đống và phân ly, tuỳ thuộc

vào sự phân ly phân cấp được thiết lập theo cách bottom-up hay top-down. Các nghiên cứu gần đây thường đề cập tới sự tích hợp của tích đống phân cấp với các phương pháp lặp lại việc định vị.

3.5.1 Phân cụm phân cấp tích đống và phân ly

Nhìn chung có hai kiểu phương pháp phân cụm phân cấp:

1. Phân cụm phân cấp tích đống:

Nó bắt đầu bằng cách đặt mỗi đối tượng vào trong cụm của bản thân nó và sau đó hoà nhập các cụm nguyên tử này vào trong các cụm càng ngày càng lớn hơn cho tới khi tất cả các đối tượng nằm trong một cụm đơn hay cho tới khi thoả điều kiện dừng cho trước. Hầu hết các phương pháp phân cụm phân cấp thuộc về loại này. Chúng chỉ khác nhau trong định nghĩa độ tương đồng giữa các cụm của chúng.

Ví dụ, phương pháp AGNES (Agglomerative Nesting) - *tích đống lồng* (Kaufman và Rousseeuw 1990). Phương pháp này sử dụng phương pháp kết nối đơn, tại đó mỗi cụm được đại diện bởi tất cả các điểm dữ liệu trong cụm, và độ tương đồng giữa hai cụm được đo bởi độ tương đồng của cặp điểm dữ liệu gần nhất thuộc về các cụm khác nhau. AGNES hoà nhập các nút (tức là các đối tượng hay các cụm riêng lẻ) có độ không tương đồng ít nhất, cứ thể cho tới khi hoà nhập thành một cụm duy nhất.

2. Phân cụm phân cấp phân ly:

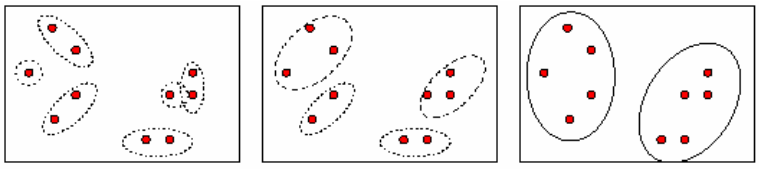
Nó ngược lại bằng cách bắt đầu với tất cả các đối tượng trong một cụm, chia nhỏ nó vào trong các phần ngày càng nhỏ hơn cho tới khi mỗi một đối tượng hình thành nên một cụm hay cho tới khi thoả một điều kiện dừng cho trước, ví dụ như số lượng các cụm được yêu cầu cần phải có hay khoảng cách giữa hai cụm gần nhất phải thoả một ngưỡng cho trước. Các phương pháp phân ly nhìn chung không nhiều và hiếm khi được áp dụng bởi khó đưa ra một quyết định đúng của việc phân chia ở một mức cao. Phương pháp phân cụm phân cấp phân ly như DIANA (Divisia Analysis) - *Phép phân tích phân ly* (Kaufman và Rousseeuw 1990).

Hoà nhập các cụm thường dựa trên khoảng cách giữa các cụm. Các phép đo được dùng rộng rãi cho khoảng cách giữa các cụm như sau, với m_i là giá trị trung bình cho cụm C_i, n_i là số lượng các điểm trong C_i, và $|p\text{-}p'|$ là khoảng cách giữa hai điểm p và p'.

$$d_{\min}(C_i, C_j) = \min_{p \in C_i, p' \in C_j} |p - p'|$$
$$d_{mean}(C_i, C_j) = |m_i - m_j|$$
$$d_{avg}(C_i, C_j) = 1/(n_i n_j) \sum_{p \in C_i} \sum_{p' \in C_j} |p - p'|$$
$$d_{\max}(C_i, C_j) = \max_{p \in C_i, p' \in C_j} |p - p'|$$

Ví dụ 3.4: Giả sử có một tập đối tượng được định vị trong một hình chữ nhật như hình 3.5.

Phương pháp phân cụm phân cấp tích đống AGNES làm việc như sau: Ban đầu mọi đối tượng được đặt vào trong một cụm của bản thân nó. Sau đó các cụm này được hoà nhập từng bước theo một số nguyên tắc như hoà nhập các cụm với khoảng cách Euclidean tối thiểu giữa các đối tượng gần nhất trong cụm. Hình 3.5 a) chỉ ra rằng các cụm đối tượng đơn gần nhất (tức là với khoảng cách Euclidean tối thiểu) trước tiên được hoà nhập vào trong hai cụm đối tượng. Xử lý hoà nhập cụm này được lặp lại và các cụm gần nhất lại được hoà nhập sau đó, như hình 3.5 b) và c). Cuối cùng, tất cả các đối tượng được hoà nhập vào trong một cụm lớn.



Hình 3.5: Phân cụm một tập các điểm dựa trên phương pháp "Tích đống lồng"

Phương pháp phân cụm phân cấp phân ly DIANA làm việc theo trật tự ngược lại. Đó là, trước tiên tất cả các đối tượng được đặt vào trong một cụm. Sau đó cụm được chia theo một số nguyên tắc, như là chia các cụm theo khoảng cách Euclidean cực đại giữa các đối tượng láng giềng gần nhất trong cụm. Hình 3.5 c) có thể được quan sát như là kết quả của phép phân chia đầu tiên. Xử lý phân chia cụm này được lặp lại và mỗi cụm lại tiếp tục được chia theo cùng tiêu

chuẩn. Hình 3.5 b) và a) có thể được quan sát như là snapshot của phân chia. Cuối cùng mỗi cụm sẽ chứa chỉ một đối tượng đơn.

Trong phân cụm phân cấp tích đống hay phân ly, ta có thể chỉ định số lượng các cụm cần có như một điều kiện kết thúc để xử lý phân cụm phân cấp dừng khi xử lý tiến đến số lượng cụm cần thiết.

Phương pháp phân cụm phân cấp mặc dầu đơn giản nhưng thường gặp khó khăn khi ra các quyết định tới hạn cho việc lựa chọn của các điểm hoà nhập hay phân chia một cách chính xác. Quyết định như vậy gọi là tới hạn bởi một khi một nhóm các đối tượng được hoà nhập hay chia, xử lý tại bước tiếp theo sẽ làm việc trên các cụm mới được sinh ra. Nó sẽ không bao giờ huỷ những gì đã làm trước đó và cũng không thực hiện chuyển đổi đối tượng giữa các cụm. Do vậy các quyết định hoà nhập hay phân chia nếu không đủ sáng suốt ở mỗi bước thì có thể dẫn tới chất lượng của các cụm sẽ kém. Hơn nữa, phương pháp này khả năng mở rộng không được tốt nên quyết định hoà nhập hay phân chia cần kiểm định và đánh giá một số lượng tốt các đối tượng hay các cụm.

Một hướng hứa hẹn để cải thiện chất lượng phân cụm của phương pháp phân cấp là tích hợp phân cụm phân cấp với các kỹ thuật phân cụm khác để có phân cụm nhiều pha. Một vài phương pháp như vậy được giới thiệu trong các mục con dưới đây. Thứ nhất là BIRCH, trước tiên sử dụng cấu trúc cây để phân chia phân cấp các đối tượng, sau đó áp dụng các giải thuật phân cụm khác để hình thành nên các cụm cải tiến. Thứ hai là CURE, đại diện cho mỗi cụm là một số lượng nào đó các điểm đại diện đã được ấn định, sau đó co chúng lại về phía tâm cụm bởi một phân số đã chỉ định. Thứ ba là ROCK, hoà nhập các cụm dựa trên liên kết nối của chúng. Thứ tư là CHAMELEON, khảo sát mô hình hoá động trong phân cụm phân cấp.

3.5.2 BIRCH: Dùng các cấp, cân bằng giữa giảm số lần lặp và phân cụm

Một phương pháp phân cụm phân cấp được tích hợp thú vị gọi là BIRCH (Balanced Iterative Reducing and Clustering using Hierachies) (Zhang, Ramakrishnan và Livny 1996). Nó đưa ra hai khái niệm: đặc trưng phân cụm

(CF - Clustering Feature) và cây CF (Clustering Feature tree), sử dụng cây CF đại diện một cụm tóm tắt để có được tốc độ và khả năng mở rộng phân cụm tốt trong các cơ sở dữ liệu lớn. Nó cũng tốt đối với phân cụm tăng trưởng động của các điểm dữ liệu đầu vào.

Một đặc trưng phân cụm CF là một bộ ba thông tin tóm tắt về cụm con các điểm. Cho trước N điểm có hướng $\{X_i\}$ trong một cụm con, CF được định nghĩa như sau:

$$CF = (N, \overrightarrow{LS}, SS) \tag{3.23}$$

với N là số các điểm trong cụm con, $\overrightarrow{LS}$ là tổng tuyến tính trên N điểm $\sum_{i=1}^{N} \vec{X}_i$ và SS là tổng bình phương của các điểm dữ liệu $\sum_{i=1}^{N} \vec{X}_i^{\,2}$.

Một cây CF là một cây cân bằng chiều cao, nó lưu trữ các đặc trưng phân cụm. Nó có hai tham số: hệ số phân nhánh B và ngưỡng T. Hệ số phân nhánh chỉ rõ số lượng tối đa các con. Tham số ngưỡng chỉ rõ đường kính tối đa của các cụm con được lưu trữ tại các nút lá. Bằng cách thay đổi giá trị ngưỡng, nó có thể thay đổi kích thước của cây. Các nút không phải là lá lưu trữ tổng các CFs của các nút con, do vậy, tóm tắt thông tin về các con của chúng.

Giải thuật BIRCH có hai pha sau đây:

- Pha 1: Quét cơ sở dữ liệu để xây dựng một cây CF bộ nhớ trong ban đầu, nó có thể được xem như là nén đa mức của dữ liệu mà nó cố gắng bảo toàn cấu trúc phân cụm vốn có của dữ liệu.
- Pha 2: Áp dụng một giải thuật phân cụm (đã lựa chọn) để phân cụm các nút lá của cây CF.

Trong pha 1, cây CF được xây dựng động khi các điểm dữ liệu được chèn vào. Do vậy, phương pháp này là một phương pháp tăng trưởng. Một điểm được chèn vào tới entry (cụm con) lá gần nhất. Nếu như đường kính của cụm con đã lưu trữ nút lá sau khi chèn lớn hơn giá trị ngưỡng, thì nút lá và các nút có thể khác bị chia. Sau khi chèn một điểm mới, thông tin về nó được đưa qua theo hướng gốc của cây. Ta có thể thay đổi kích thước cây CF bằng cách thay đổi

ngưỡng. Nếu như kích thước bộ nhớ cần thiết để lưu trữ cây CF là lớn hơn kích thước bộ nhớ chính thì một giá trị nhỏ hơn của ngưỡng được chỉ định và cây CF được xây dựng lại. Xử lý xây dựng lại này được biểu diễn bằng cách xây dựng một cây mới từ các nút lá của cây cũ. Do vậy, xử lý xây dựng lại cây được làm mà không cần đọc lại tất cả các điểm. Bởi vậy, để xây dựng cây, dữ liệu chỉ phải đọc một lần. Nhiều heuristic và các phương pháp cũng được giới thiệu để giải quyết các outlier và cải thiện chất lượng cây CF bởi các lần quét thêm vào của dữ liệu.

Sau khi cây CF được xây dựng, bất kỳ một giải thuật phân cụm nào, ví dụ như giải thuật phân chia điển hình có thể được dùng với cây CF trong pha 2.

BIRCH cố gắng đưa ra các cụm tốt nhất với các tài nguyên có sẵn. Với số lượng giới hạn của bộ nhớ chính, một xem xét quan trọng là cần tối thiểu hoá thời gian yêu cầu đối với I/O. Nó áp dụng kỹ thuật phân cụm nhiều pha: quét đơn tập dữ liệu mang lại một cơ sở phân cụm tốt, và một hay nhiều lần quét thêm vào (tuỳ ý) được dùng để cải thiện xa hơn chất lượng. Bởi vậy độ phức tạp tính toán của giải thuật là $O(N)$, với N là số các đối tượng được phân cụm.

Bằng các thử nghiệm đã thấy được khả năng mở rộng tuyến tính của giải thuật về mặt số lượng các điểm và chất lượng tốt của phân cụm dữ liệu. Tuy nhiên, mỗi nút trong cây CF có thể chỉ nắm giữ một số lượng giới hạn các entry bởi kích thước của nó, một nút cây CF không phải luôn luôn tương đương với một cụm tự nhiên. Hơn nữa, nếu các cụm không phải có hình cầu, BIRCH sẽ không thực hiện tốt bởi nó sử dụng khái niệm bán kính hay đường kính để điều khiển đường bao một cụm.

3.5.3 CURE: Phân cụm sử dụng các đại diện

Hầu hết các giải thuật phân cụm hoặc là có ưu đãi các cụm có dạng hình cầu và kích thước giống nhau, hoặc là rất mong manh với sự hiện diện của các outlier. Một phương pháp thú vị gọi là CURE (Clustering Using REpresentatives) (Guha, Rastogi và Shim 1998), tích hợp các giải thuật phân

chia và phân cấp, khắc phục vấn đề ưu đãi các cụm có dạng hình cầu và kích thước giống nhau.

CURE cung cấp một giải thuật phân cụm phân cấp mới lạ theo vị trí giữa (middle ground) giữa việc dựa trên trọng tâm và tất cả các cực điểm. Thay vì sử dụng một trọng tâm đơn đại diện một cụm, CURE ấn định một số lượng các điểm đại diện được lựa chọn để miêu tả một cụm. Các điểm đại diện này được sinh ra bằng cách trước tiên lựa chọn các điểm rải rác đều trong cụm, sau đó co chúng lại về phía tâm cụm bởi một phân số (hệ số co). Các cụm với cặp các điểm đại diện gần nhất sẽ được hoà nhập tại mỗi bước của giải thuật.

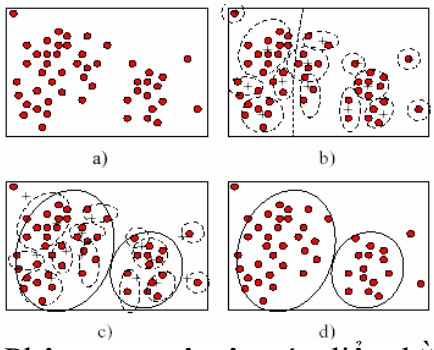
Mỗi cụm có hơn một điểm đại diện cho phép CURE điều chỉnh tốt hình học của các hình không phải hình cầu. Việc co lại giúp làm giảm đi hiệu quả của các outlier. Bởi vậy, CURE thực sự mạnh hơn đối với các outlier và nhận biết các cụm không có dạng hình cầu với kích thước khác nhau nhiều.

Để vận dụng các cơ sở dữ liệu lớn, CURE dùng kết hợp lấy mẫu và phân chia ngẫu nhiên: Một mẫu ngẫu nhiên trước tiên được phân chia và mỗi phân chia được phân cụm cục bộ. Các cụm cục bộ sau đó được phân cụm lần thứ hai để có được các cụm mong muốn.

Các bước chính của giải thuật CURE được phác hoạ vắn tắt như sau: (1) Lấy một mẫu ngẫu nhiên s; (2) Phân chia mẫu s thành p phần, mỗi phần có kích thước s/p; (3) Cụm cục bộ phân chia thành s/pq cụm $q>1$; (4) Khử các outlier bằng cách lấy mẫu ngẫu nhiên: Nếu một cụm tăng trưởng quá chậm, loại bỏ nó; (5) Phân cụm các cụm cục bộ, một xử lý co nhiều điểm đại diện về phía trọng tâm bằng một phân số α được chỉ định bởi người dùng, tại đó các đại diện có được hình dạng của cụm; (6) Đánh dấu dữ liệu với nhãn cụm tương ứng.

Sau đây ta biểu diễn một ví dụ để thấy cách làm việc của CURE.

Ví dụ 3.5: Giả sử có một tập các đối tượng được định vị trong một hình chữ nhật. Cho $p = 2$, người dùng cần phân cụm các đối tượng vào trong hai cụm.



Hình 3.6: Phân cụm một tập các điểm bằng CURE

Trước tiên, 50 đối tượng được lấy mẫu như hình 3.6 *a*). Sau đó, các đối tượng này được phân chia ban đầu vào trong hai cụm, mỗi cụm chứa 50 điểm. Ta phân cụm cục bộ các phần chia này thành 10 cụm con dựa trên khoảng cách trung bình tối thiểu. Mỗi đại diện cụm được đánh dấu bởi một chữ thập nhỏ, như hình 3.6 *b*). Các đại diện này được di chuyển về phía trọng tâm bởi một phân số α, như hình 3.6 *c*).Ta có được hình dạng của cụm và thiết lập thành 2 cụm. Do vậy, các đối tượng được phân chia vào trong hai cụm với các outlier được gỡ bỏ như biểu diễn ở hình 3.6 *d*).

CURE đưa ra các cụm chất lượng cao với sự hiện hữu của các outlier, các hình dạng phức tạp của các cụm với các kích thước khác nhau. Nó có khả năng mở rộng tốt cho các cơ sở dữ liệu lớn mà không cần hy sinh chất lượng phân cụm. CURE cần một ít các tham số được chỉ định bởi người dùng, như kích thước của mẫu ngẫu nhiên, số lượng các cụm mong muốn và hệ số co α. Độ nhạy một phép phân cụm được cung cấp dựa trên kết quả của việc thay đổi các tham số. Mặc dầu nhiều tham số bị thay đổi mà không ảnh hưởng tới chất lượng phân cụm nhưng tham số thiết lập nhìn chung có ảnh hưởng đáng kể.

Một giải thuật phân cụm phân cấp tích đống khác được phát triển bởi (Guha, Rastogi và Shim 1999) gọi là ROCK, nó phù hợp cho việc phân cụm các thuộc tính xác thực. Nó đo độ tương đồng của 2 cụm bằng cách so sánh toàn bộ liên kết nối của 2 cụm dựa trên mô hình liên kết nối tĩnh được chỉ định bởi người dùng, tại đó liên kết nối của hai cụm C1 và C2 được định nghĩa bởi số

lượng các liên kết chéo giữa hai cụm và liên kết *link*(p_i, p_j) là số lượng các láng giềng chung giữa hai điểm p_i và p_j.

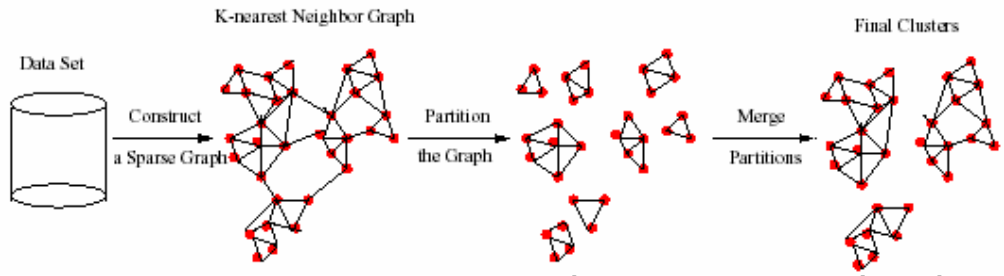
ROCK trước tiên xây dựng đồ thị thưa từ một ma trận tương đồng dữ liệu cho trước, sử dụng một ngưỡng tương đồng và khái niệm các láng giềng chia sẻ, và sau đó biểu diễn một giải thuật phân cụm phân cấp trên đồ thị thưa.

3.5.4 CHAMELEON: Một giải thuật phân cụm phân cấp sử dụng mô hình động

Một giải thuật phân cụm thú vị khác gọi là CHAMELEON, nó khảo sát mô hình hoá động trong phân cụm phân cấp, được phát triển bởi Karypis, Han và Kumar (1999). Khi xử lý phân cụm, 2 cụm được hoà nhập nếu liên kết nối và độ chặt (độ gần) giữa hai cụm được liên kết cao với liên kết nối và độ chặt nội tại của các đối tượng nằm trong phạm vi các cụm. Xử lý hoà nhập dựa trên mô hình động tạo điều kiện thuận lợi cho sự khám phá ra các cụm tự nhiên và đồng nhất, nó áp dụng cho tất cả các kiểu dữ liệu miễn là hàm tương đồng được chỉ định.

CHAMELEON có được dựa trên quan sát các yếu điểm của hai giải thuật phân cụm phân cấp: CURE và ROCK. CURE và các lược đồ quan hệ bỏ qua thông tin về liên kết nối tổng thể của các đối tượng trong 2 cụm; ngược lại, ở ROCK, các lược đồ quan hệ lờ đi thông tin về độ chặt của 2 cụm trong khi nhấn mạnh liên kết nối của chúng.

CHAMELEON trước tiên sử dụng một giải thuật phân chia đồ thị để phân cụm các mục dữ liệu vào trong một số lượng lớn các cụm con tương đối nhỏ. Sau đó dùng giải thuật phân cụm phân cấp tập hợp để tìm ra các cụm xác thực bằng cách lặp lại việc kết hợp các cụm này với nhau. Để xác định các cặp cụm con giống nhau nhất, cần đánh giá cả liên kết nối cũng như độ chặt của các cụm, đặc biệt là các đặc tính nội tại của bản thân các cụm. Do vậy nó không tuỳ thuộc vào một mô hình tĩnh được cung cấp bởi người dùng và có thể tự động thích ứng với các đặc tính nội tại của các cụm đang được hoà nhập.



Hình 3.7: CHAMELEON: Phân cụm phân cấp dựa trên k-láng giềng gần và mô hình hoá động

Như hình 3.7, CHAMELEON miêu tả các đối tượng dựa trên tiếp cận đồ thị được dùng phổ biến: k-láng giềng gần nhất. Mỗi đỉnh của đồ thị k-láng giềng gần nhất đại diện cho một đối tượng dữ liệu, tại đó tồn tại một cạnh giữa hai đỉnh (đối tượng), nếu một đối tượng là giữa *k* đối tượng giống nhau so với các đối tượng khác. Đồ thị k-láng giềng gần nhất G_k có được khái niệm láng giềng động: Bán kính láng giềng của một điểm dữ liệu được xác định bởi mật độ của miền mà trong đó các đối tượng cư trú. Trong một miền dày đặc, láng giềng được định nghĩa hẹp, và trong một miền thưa thớt, láng giềng được định rộng hơn. So sánh với mô hình định nghĩa bởi phương pháp dựa trên mật độ như DBSCAN (giới thiệu ở mục sau), DBSCAN dùng mật độ láng giềng toàn cục, G_k có được láng giềng tự nhiên hơn. Hơn nữa, mật độ miền được ghi như trọng số của các cạnh. Cạnh của một miền dày đặc theo trọng số lớn hơn so với của một miền thưa thớt.

CHAMELEON chỉ rõ sự tương đồng giữa mỗi cặp các cụm C_i và C_j theo liên kết nối tương đối $RI(C_i,C_j)$ và độ chặt tương đối $RC(C_i,C_j)$ của chúng.

Liên kết nối tương đối $RI(C_i,C_j)$ giữa hai cụm C_i và C_j được định nghĩa như liên kết nối tuyệt đối giữa C_i và C_j đã tiêu chuẩn hoá đối với liên kết nối nội tại của hai cụm C_i và C_j. Đó là:

$$RI(C_i,C_j)=\frac{\left|EC_{\{C_i,C_j\}}\right|}{\frac{1}{2}\left(\left|EC_{C_i}\right|+\left|EC_{C_j}\right|\right)} \qquad (3.24)$$

với $EC_{\{C_i,C_j\}}$ là cạnh cắt (*edge-cut*) của cụm chứa cả C_i và C_j để cụm này được rơi vào trong C_i và C_j, và tương tự như vậy, ECC_i (hay ECC_j) là kích thước

của *min-cut bisector* (tức là tổng trọng số của các cạnh mà chia đồ thị thành hai phần thô bằng nhau).

Độ chặt tương đối giữa một cặp các cụm C_i và C_j là $RC(C_i,C_j)$ được định nghĩa như là độ chặt tuyệt đối giữa C_i và C_j được tiêu chuẩn hoá đối với liên kết nối nội tại của hai cụm C_i và C_j. Đó là:

$$RC(C_i,C_j)=\frac{\overline{S}_{EC\{C_i,C_j\}}}{\frac{|C_i|}{|C_i|+|C_j|}\overline{S}_{EC_{C_i}}+\frac{|C_j|}{|C_i|+|C_j|}\overline{S}_{EC_{C_j}}} \qquad (3.25)$$

với $\overline{S}_{EC_{\{C_i,C_j\}}}$ là trọng số trung bình của các cạnh kết nối các đỉnh trong C_i tới các đỉnh C_j và $\overline{S}_{EC_{C_i}}$ (hay $\overline{S}_{EC_{C_j}}$) là trọng số trung bình của các cạnh thuộc về *min-cut bisecter* của cụm C_i (hay C_j).

Như vậy, CHAMELEON có nhiều khả năng khám phá ra các cụm có hình dạng tuỳ ý với chất lượng cao hơn so với DBSCAN và CURE. Tuy vậy, thời gian chi phí xử lý cho dữ liệu có chiều cao có thể là $O(n^2)$ cho n đối tượng trong tình huống xấu nhất.

3.6 Các phương pháp phân cụm dựa trên mật độ

Để tìm ra các cụm với hình dạng tuỳ ý, các phương pháp phân cụm dựa trên mật độ đã được phát triển, nó kết nối các miền với mật độ đủ cao vào trong các cụm hay phân cụm các đối tượng dựa trên phân bố hàm mật độ.

3.6.1 DBSCAN: Phương pháp phân cụm dựa trên mật độ trên các miền có kết nối với mật độ đủ cao

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) là một giải thuật phân cụm dựa trên mật độ, được phát triển bởi Ester, Kriegel, Sander và Xu (1996). Giải thuật này tăng trưởng các miền với mật độ đủ cao vào trong các cụm và tìm ra các cụm với hình dạng tuỳ ý trong cơ sở dữ liệu không gian có nhiễu. Một cụm được định nghĩa như là một tập cực đại các điểm có kết nối dựa trên mật độ.

Ý tưởng cơ bản của phân cụm dựa trên mật độ như sau: Đối với mỗi đối tượng của một cụm, láng giếng trong một bán kính cho trước (ε) (gọi là ε -láng giềng) phải chứa chứa ít nhất một số lượng tối thiểu các đối tượng (*MinPts*).

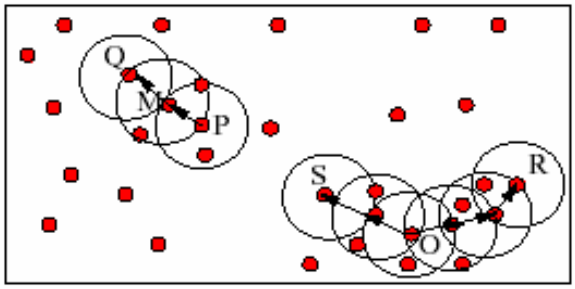
Một đối tượng nằm trong một bán kính cho trước (ε) chứa không ít hơn một số lượng tối thiểu các đối tượng láng giềng (*MinPts*), được gọi là đối tượng nòng cốt (*core object*) (đối với bán kính ε và số lượng tối thiểu các điểm *MinPts*).

Một đối tượng *p* là mật độ trực tiếp tiến (*directly density-reachable*) từ đối tượng *q* với bán kính ε và số lượng tối thiểu các điểm *MinPts* trong một tập các đối tượng *D* nếu *p* trong phạm vi ε -láng giềng của *q* với *q* chứa ít nhất một số lượng tối thiểu các điểm *MinPts*.

Một đối tượng *p* là mật độ tiến (*density-reachable*) từ đối tượng *q* với bán kính ε và *MinPts* trong một tập các đối tượng *D* nếu như có một chuỗi đối tượng $p_1,p_2,...,p_n$, $p_1=q$ và $p_n=p$ với $1 \le i \le n$, $p_i \in D$ và p_{i+1} là mật độ trực tiếp tiến từ p_i đối với ε và *MinPts*.

Một đối tượng *p* là mật độ liên kết với đối tượng *q* đối với ε và *MinPts* trong một tập đối tượng *D* nếu như có một đối tượng $o \in D$ để cả *p* và *q* là mật độ tiến từ *o* đối với ε và *MinPts*.

Ví dụ 3.6: Trong hình 3.8, ε cho trước đại diện cho bán kính các đường tròn, cho *MinPts*=3, *M* là mật độ trực tiếp tiến từ *P*; *Q* là mật độ (không trực tiếp) tiến từ *P*. Tuy nhiên *P* không phải là mật độ tiến từ *Q*. Tương tự như vậy, *R* và *S* là mật độ tiến từ *O*; và *O*, *R* và *S* tất cả là mật độ liên kết.



Hình 3.8: Mật độ tiến và mật độ liên kết trong phân cụm dựa trên mật độ

Lưu ý rằng mật độ tiến là bắc cầu đóng (transitive closure) của mật độ trực tiếp tiến, và quan hệ này là không đối xứng. Chỉ các đối tượng nòng cốt là mật độ tiến lẫn nhau (giao hoán). Mật độ liên kết là một quan hệ đối xứng.

Một cụm dựa trên mật độ là một tập các đối tượng mật độ liên kết là tối đa đối với mật độ tiến; mọi đối tượng không chứa trong bất kỳ một cụm nào là nhiễu.

Dựa trên khái niệm mật độ tiến, giải thuật phân cụm dựa trên mật độ DBSCAN được phát triển để phân cụm dữ liệu trong cơ sở dữ liệu. Nó kiểm soát ε-láng giềng của mỗi điểm trong cơ sở dữ liệu. Nếu như ε-láng giềng của một điểm *p* chứa nhiều hơn *MinPts*, một cụm mới với *p* là đối tượng nòng cốt được thiết lập. Sau đó lặp lại việc tập hợp các đối tượng trực tiếp từ các đối tượng nòng cốt này, nó có thể bao gồm việc hoà nhập một vài cụm mật độ tiến. Xử lý này dừng khi không có điểm mới nào được thêm vào ở bất kỳ cụm nào.

3.6.2 OPTICS: Sắp xếp các điểm để nhận biết cấu trúc phân cụm

Mặc dầu giải thuật phân cụm dựa trên mật độ DBSCAN có thể tìm ra cụm các đối tượng với việc lựa chọn các tham số đầu vào như ε và *MinPts*, người dùng vẫn chịu trách nhiệm lựa chọn các giá trị tham số tốt để tìm ra các cụm chính xác. Trên thực tế, đây là bài toán có sự kết hợp của nhiều giải thuật phân cụm khác. Các thiết lập tham số như vậy thường khá khó để xác định, đặc biệt trong thế giới thực, các tập dữ liệu số chiều cao. Hầu hết các giải thuật rất nhạy với các giá trị tham số: các thiết lập có sự khác biệt nhỏ có thể dẫn tới các phân chia dữ liệu rất khác nhau. Hơn nữa, các tập dữ liệu thực số chiều cao thường có phân bố rất lệch, thậm chí ở đó không tồn tại một thiết lập tham số toàn cục cho đầu vào, kết quả của một giải thuật phân cụm có thể mô tả bản chất cấu trúc phân cụm một cách chính xác.

Để khắc phục khó khăn này, một phương pháp sắp xếp cụm gọi là OPTICS (Ordering Points To Identify the Clustering Structure) được phát triển bởi (Ankerst, Breunig, Kriegel và Sander 1999). Nó tính một sắp xếp phân cụm tăng dần cho phép phân tích cụm tự động và tương tác. Sắp xếp phân cụm này chứa

đựng thông tin tương đương với phân cụm dựa trên mật độ phù hợp với một phạm vi rộng các thiết lập tham số.

Bằng cách khảo sát giải thuật phân cụm dựa trên mật độ, DBSCAN có thể dễ dàng thấy rằng đối với một giá trị hằng số *MinPts*, các cụm dựa trên mật độ đối với mật độ cao hơn (tức là một giá trị ε thấp hơn) được chứa hoàn toàn trong các tập mật độ liên kết đối với một mật độ thấp hơn. Bởi vậy, để đưa ra các cụm dựa trên mật độ với một tập các tham số khoảng cách, giải thuật cần lựa chọn các đối tượng để xử lý theo một trật tự cụ thể để đối tượng là mật độ tiến đối với giá trị ε thấp nhất được kết thúc trước tiên.

Dựa trên ý tưởng này, hai giá trị cần được lưu trữ đối với mỗi đối tượng: khoảng cách nòng cốt (*core-distance*) và khoảng cách tiến (*reachability-distance*).

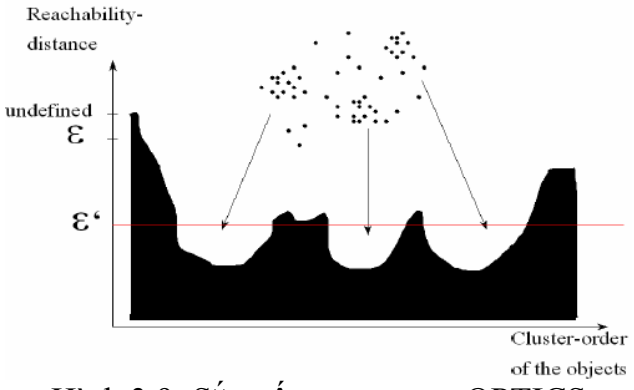
Khoảng cách nòng cốt của một đối tượng p là khoảng cách nhỏ nhất ε' giữa p và một đối tượng trong ε - láng giềng của nó để p sẽ là một đối tượng nòng cốt đối với ε' nếu như láng giềng này được chứa trong ε - láng giềng của p. Nếu không thì khoảng cách nòng cốt là không xác định.

Khoảng cách tiến của một đối tượng p đối với một đối tượng o khác là khoảng cách nhỏ nhất để p là mật độ trực tiếp tiến từ o nếu o là một đối tượng nòng cốt. Nếu o không phải là một đối tượng nòng cốt, ngay cả tại khoảng cách phát sinh ε, khoảng cách tiến của một đối tượng p đối với o là không xác định.

Giải thuật OPTICS tạo lập trật tự của một cơ sở dữ liệu, thêm vào đó là lưu trữ khoảng cách nòng cốt và một khoảng cách tiến phù hợp với mỗi đối tượng. Thông tin như vậy là đủ cho sự rút trích của tất cả các phân cụm dựa trên mật độ đối với bất kỳ một khoảng cách ε' nhỏ hơn khoảng cách phát sinh ε từ trật tự này.

Sắp xếp cụm của một tập dữ liệu có thể được trình bày và hiểu bằng đồ thị. Ví dụ, hình 3.9 là một biểu đồ tiến cho một tập dữ liệu hai chiều đơn giản, nó biểu diễn một cái nhìn tổng quát về dữ liệu được cấu trúc và phân cụm như thế

nào. Các phương pháp cũng được phát triển để quan sát các cấu trúc phân cụm cho dữ liệu số chiều cao.



Hình 3.9: Sắp xếp cụm trong OPTICS

Bởi tương đương cấu trúc của giải thuật OPTICS tới DBSCAN, giải thuật OPTICS có cùng độ phức tạp thời gian chạy như của DBSCAN. Các cấu trúc đánh chỉ số không gian có thể được dùng để nâng cao khả năng biểu diễn của nó.

3.6.3 DENCLUE: Phân cụm dựa trên các hàm phân bố mật độ

DENCLUE (DENsity -based CLUstEring - phân cụm dựa trên mật độ) (Hinneburg và Keim 1998) là phương pháp phân cụm dựa trên một tập các hàm phân bố mật độ.

Phương pháp được dựa trên ý tưởng sau: (1) Tác động của mỗi điểm dữ liệu có thể được làm mô hình chính thức sử dụng một hàm toán học gọi là hàm tác động, hàm tác động được xem như là một hàm mô tả tác động của một điểm dữ liệu trong phạm vi láng giềng của nó; (2) Toàn bộ mật độ của không gian dữ liệu có thể được làm mô hình theo phép phân tích tổng các hàm tác động của tất cả các điểm dữ liệu; (3) Các cụm sau đó có thể được xác định chính xác bằng cách nhận biết các attractor mật độ, tại đó các attractor mật độ cực đại cục bộ của toàn bộ hàm mật độ.

Hàm tác động của một điểm dữ liệu $y \in F^d$, với F^d là một không gian đặc trưng d chiều, là một hàm cơ bản $f_B^y : F^d \to R_0^+$, được định nghĩa dưới dạng một hàm tác động cơ bản f_B:

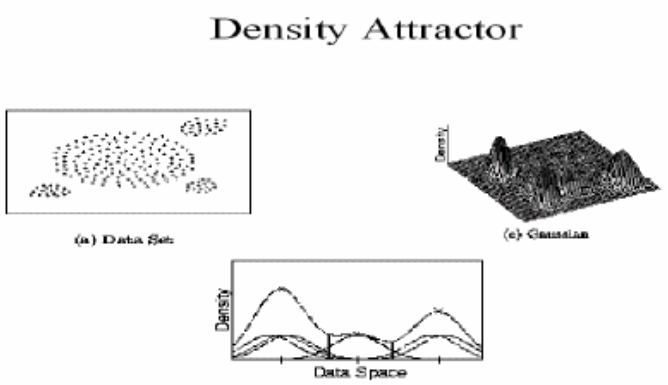
$$f_B^y = f_B(x,y) \tag{3.26}$$

Theo nguyên tắc, hàm tác động có thể là một hàm tuỳ ý nhưng nó nên là phản xạ và đối xứng. Nó có thể là một hàm khoảng cách Euclidean, một hàm tác động wave bình phương:

$$f_{Square}(x,y) = \begin{cases} 0 & if\ d(x,y) > \sigma \\ 1 & otherwise \end{cases} \tag{3.27}$$

hay một hàm tác động Gaussian:

$$f_{Gause}(x,y) = e^{-\frac{d(x,y)^2}{2\sigma^2}} \tag{3.28}$$



Hình 3.10: Hàm mật độ và attractor mật độ

Một hàm mật độ được định nghĩa là tổng các hàm tác động của tất cả các điểm dữ liệu. Cho trước N đối tượng dữ liệu được mô tả bởi một tập các vectơ đặc trưng $D = \{x_1,...,x_N\} \subset F^D$, hàm mật độ được định nghĩa như sau:

$$f_B^D = \sum_{i=1}^{N} f_B^{x_i}(x) \tag{3.29}$$

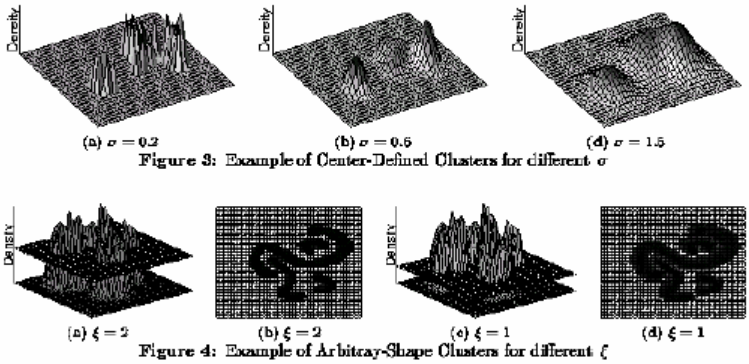
Ví dụ, hàm mật độ cho kết quả từ hàm tác động Gaussian (3.28) là:

$$f_{Gaussian}^D(x) = \sum_{i=1}^{N} e^{-\frac{d(x,y)^2}{2\sigma^2}} \tag{3.30}$$

Từ hàm mật độ, ta có thể định nghĩa độ dốc (gradient) của một hàm và attractor mật độ (attractor mật độ là cực đại cục bộ của toàn bộ hàm mật độ). Đối với một hàm tác động liên tục và phân biệt, một giải thuật leo đồi (hill climbing), được chỉ ra bởi độ dốc (gradient), có thể được dùng để xác định attractor mật độ của một tập các điểm dữ liệu.

Dựa trên các khái niệm này, cả cụm được định nghĩa trung tâm và cụm hình dạng tuỳ ý có thể được định nghĩa chính thức. Một cụm có định nghĩa trung tâm là một tập con *C* đang là mật độ được rút trích, với hàm mật độ không ít hơn một ngưỡng ξ, ngược lại (tức là nếu hàm mật độ nhỏ hơn ngưỡng ξ) thì nó là một outlier. Một cụm hình dạng tuỳ ý là một tập của tập con của *C*, mỗi tập đang là mật độ được rút trích, với hàm mật độ không ít hơn một ngưỡng ξ, và tồn tại một đường đi *P* từ mỗi miền tới những miền khác và hàm mật độ cho mỗi điểm dọc theo đường đi không ít hơn ξ.

DENCLUE có các thuận lợi chính sau đây khi so sánh với các giải thuật phân cụm khác: (1) Nó có một nền tảng toán học vững chắc, tổng quát hoá các phương pháp phân cụm khác, bao gồm các phương pháp dựa trên phân chia, phân cấp và dựa trên vị trí; (2) Nó có các đặc tính phân cụm tốt đối với các tập dữ liệu với số lượng nhiễu lớn; (3) Nó cho phép một mô tả toán học cô đọng của các cụm có hình dạng tuỳ ý trong các tập dữ liệu số chiều cao; (4) Nó sử dụng các ô lưới nhưng chỉ giữ thông tin về các ô lưới mà thực sự chứa đựng các điểm dữ liệu và quản lý các ô này trong một cấu trúc truy cập dựa trên cây và do vậy nó nhanh hơn đáng kể so với các giải thuật tác động, như nó nhanh hơn DBSCAN tới 45 lần. Tuy vậy, phương pháp cần sự chọn lựa cẩn thận các tham số, tham số mật độ σ và ngưỡng nhiễu ξ, việc lựa chọn các tham số như vậy có ảnh hưởng đáng kể chất lượng của các kết quả phân cụm.



Hình 3.11: Các cụm được định nghĩa trung tâm và các cụm có hình dạng tuỳ ý

3.7 Các phương pháp phân cụm dựa trên lưới

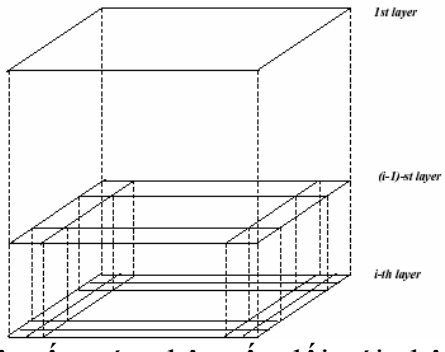
Một tiếp cận dựa trên lưới dùng cấu trúc dữ liệu lưới đa phân giải. Trước tiên nó lượng tử hoá không gian vào trong một số hữu hạn các ô mà đã hình thành nên cấu trúc lưới, sau đó thực hiện tất cả các thao tác trong cấu trúc lưới đó. Thuận lợi chính của tiếp cận này là thời gian xử lý nhanh, điển hình là độc lập của số lượng các đối tượng dữ liệu nhưng độc lập chỉ trên số lượng các ô trong mỗi chiều trong không gian lượng tử hóa.

Các ví dụ điển hình của tiếp cận dựa trên lưới bao gồm STING - khảo sát thông tin thống kê được lưu trữ trong các ô lưới; WaveCluster - các cụm đối tượng sử dụng phương pháp biến đổi wavelet; CLIQUE - miêu tả một tiếp cận dựa trên lưới và mật độ cho phân cụm trong không gian dữ liệu số chiều cao.

3.7.1 STING: Một tiếp cận lưới thông tin thống kê

STING (STatistical INformation Grid) (Wang, Yang và Munz 1997) là một tiếp cận đa phân giải dựa trên lưới. Trong tiếp cận này, miền không gian được chia thành các ô hình chữ nhật. Thường có một vài mức các ô hình chữ nhật tương ứng với các mức khác nhau của phân giải và các ô này thiết lập nên một cấu trúc phân cấp: mỗi ô tại một mức cao được phân chia để hình thành nên một số lượng các ô tại mức thấp hơn tiếp theo. Hơn nữa, các phần quan trọng của thông tin thống kê như *mean*, *max*, *min*, *count*, độ lệch chuẩn (*standard deviation*), v.v... đã kết hợp với các giá trị thuộc tính trong mỗi ô lưới được tính toán trước và được lưu trữ trước khi một truy vấn được submit tới một hệ thống.

Hình 3.12 cho thấy một cấu trúc phân cấp đối với phân cụm STING.



Hình 3.12: Một cấu trúc phân cấp đối với phân cụm STING

Tập các tham số dựa trên thống kê bao gồm: - tham số độc lập với thuộc tính *n* (*count*) và các tham số phụ thuộc thuộc tính *m* (*mean*), *s* (độ lệch chuẩn), *min* (minimum), *max* (maximum), và kiểu của phân bố mà giá trị thuộc tính trong ô tiếp theo như *normal- bình thường, uniform-đồng nhất, exponential- số mũ,* hay *none* (nếu phân bố không được biết). Khi dữ liệu được tải vào trong cơ sở dữ liệu, tập các tham số *n, m, s, min, max* của các ô mức đáy được tính toán trực tiếp từ dữ liệu. Giá trị của phân bố có thể được ấn định bởi người dùng nếu như kiểu phân bố không được biết trước hay có được bởi các kiểm định giả thuyết như kiểm định $\chi 2$. Các tham số của các ô mức cao hơn có thể dễ dàng được tính từ các tham số ở các ô mức thấp hơn. Kiểu phân bố của các ô mức cao hơn có thể được tính toán dựa trên các kiểu phân bố theo số đông của các ô tương đương mức thấp hơn của nó cộng với một ngưỡng xử lý lọc. Nếu như các phân bố của ô mức thấp hơn không giống nhau và thiếu ngưỡng kiểm định, kiểu phân bố của ô mức cao được đặt là "*none*".

Thông tin thống kê có được sẽ rất hữu ích khi trả lời các truy vấn. Top-down là phương pháp trả lời truy vấn dựa trên lưới thông tin thống kê có thể khái quát như sau: Trước tiên nó có thể xác định một lớp để bắt đầu, nó thường bao gồm một số lượng nhỏ các ô. Đối với mỗi ô trong lớp hiện thời, ta tính toán khoảng tin cậy (hay phạm vi được đánh giá) khả năng mà ô này có liên quan tới truy vấn. Các ô không liên quan sẽ được gỡ bỏ khỏi xem xét sau này, và xử lý ở mức sâu hơn sẽ chỉ xem xét các ô liên quan. Xử lý này được lặp lại cho tới khi nó tiến đến lớp đáy. Tại thời điểm này, nếu đạt được truy vấn chỉ định thì sẽ trả lại các miền các ô liên quan đáp ứng yêu cầu của truy vấn; mặt khác, lấy ra dữ liệu nằm trong các ô liên quan, tiếp tục xử lý; và trả lại các kết quả thoả mãn yêu cầu của truy vấn.

Tiếp cận này đưa ra một số thuận lợi so với các phương pháp phân cụm khác: (1) Tính toán dựa trên lưới là truy vấn độc lập, từ đó thông tin thống kê được lưu trữ trong mỗi ô đại diện cho thông tin tóm tắt của dữ liệu trong ô lưới, độc lập với truy vấn; (2) Cấu trúc lưới làm cho xử lý song song và cập nhật tăng

trưởng được thuận lợi; (3) Thuận lợi chủ yếu của phương pháp này hiệu quả của phương pháp: STING xuyên suốt dữ liệu một lần để tính toán các tham số thống kê của các ô, và do vậy độ phức tạp thời gian phát sinh các cụm là $O(N)$, với N là tổng số các đối tượng. Sau khi phát sinh cấu trúc phân cấp này, thời gian xử lý truy vấn là $O(G)$, với G là tổng số các ô lưới tại mức thấp nhất, nó thường nhỏ hơn nhiều so với N - tổng số các đối tượng.

Tuy vậy, từ khi STING sử dụng tiếp cận đa phân giải để thực hiện phép phân tích cụm, chất lượng của phân cụm STING sẽ tuỳ thuộc vào độ sần (granularity) của mức thấp nhất của cấu trúc lưới. Nếu độ sần là rất tốt, chi phí xử lý về cơ bản sẽ tăng lên; tuy nhiên nếu như mức đáy của cấu trúc lưới quá thô, nó có thể giảm chất lượng tốt (độ mịn) của phép phân cụm. Hơn nữa, STING không xem xét mối quan hệ không gian giữa các ô con và các ô láng giềng của chúng để xây dựng các ô cha. Kết quả là hình dạng của các cụm kết quả là nhất quán (*isothetic*), tất cả các đường bao cụm theo chiều ngang hoặc theo chiều dọc, không có chiều chéo nào được dò thấy. Điều này có thể dẫn tới chất lượng và độ chính xác các cụm thấp hơn nhưng có thời gian xử lý nhanh hơn.

3.7.2 WaveCluster: Phân cụm sử dụng phép biến đổi wavelet

WaveCluster (Sheikholeslami, Chatterjee và Zhang 1998) là một tiếp cận phân cụm đa phân giải, trước tiên tóm tắt dữ liệu bằng cách lợi dụng cấu trúc lưới đa phân giải trên không gian dữ liệu, sau đó biến đổi không gian đặc trưng gốc bằng phép biến đối wavelet và tìm các miền đông đúc trong không gian đã biến đổi.

Trong tiếp cận này, mỗi ô lưới tóm tắt thông tin của một nhóm các điểm, thông tin tóm tắt này vừa đủ để đưa vào trong bộ nhớ chính cho phép biến đổi wavelet đa phân giải và phép phân tích cụm sau đó. Trong cấu trúc lưới, các thuộc tính số của một đối tượng không gian có thể được đại diện bởi một vectơ đặc trưng, tại đó mỗi phần tử của vectơ tương đương với một thuộc tính số, hay

đặc trưng. Cho một đối tượng với *n* thuộc tính số, vectơ đặc trưng sẽ là một điểm trong không gian đặc trưng *n* chiều.

Phép biến đổi wavelet là một kỹ thuật xử lý tín hiệu, nó phân tích một tín hiệu vào trong các dải tần số con. Mô hình wavelet cũng làm việc trên các tín hiệu *n* chiều bằng cách áp dụng phép biến đổi 1 chiều *n* lần.

Trong phép biến đổi wavelet, dữ liệu không gian được chuyển đổi vào trong miền tần số. Kết hợp với một hàm nòng cốt thích hợp cho kết quả trong một không gian biến đổi, tại đó các cụm tự nhiên trong dữ liệu trở nên dễ phân biệt hơn. Các cụm sau đó có thể được nhận biết bằng cách tìm ra các miền đông đúc trong vùng biến đổi.

Phép biến đổi wavelet cung cấp các đặc trưng thú vị sau: Trước tiên nó cung cấp phân cụm không giám sát. Các lọc dạng nón làm nổi bật các miền mà tại đó các điểm phân cụm, nhưng đồng thời cũng có khuynh hướng ngăn chặn các thông tin yếu hơn trong đường bao của chúng. Do vậy, các miền đông đúc trong không gian đặc trưng gốc đóng vai trò như là các miền thu hút (attractor) đối với các điểm gần đó và như là miền hạn chế (inhibitor) đối với các điểm không đủ gần. Điều này nghĩa là các cụm trong dữ liệu tự động nổi bật lên và làm sạch các miền xung quanh chúng. Thứ hai, các lọc thông thấp được dùng trong phép biến đổi wavelet sẽ tự động loại bỏ các outlier. Hơn nữa, đặc tính đa phân giải của phép biến đổi wavelet có thể giúp dò các cụm tại các độ chính xác khác nhau. Cuối cùng, ứng dụng phép biến đổi wavelet là rất nhanh và việc xử lý như vậy có thể cũng được thực hiện song song.

Giải thuật phân cụm dựa trên wavelet phác thảo như sau:

Giải thuật 3.7.1: Giải thuật phân cụm dựa trên wavelet đối với phân cụm đa phân giải bằng phép biến đổi wavelet.

Đầu vào: Các vectơ đặc trưng của các đối tượng dữ liệu đa chiều

Đầu ra: Các đối tượng đã phân cụm

Giải thuật:

1) Lượng tử hoá không gian đặc trưng, sau đó phân các đối tượng vào các

unit;

2) Áp dụng phép biến đổi wavelet trong không gian đặc trưng;

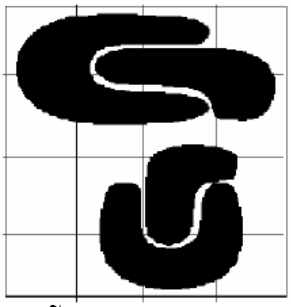
3) Tìm các phần hợp thành đã kết nối (các cụm) trong các dải con của không gian đặc trưng đã biến đổi tại các mức khác nhau;

4) Gắn các nhãn vào các unit;

5) Làm các bảng tra cứu và ánh xạ các đối tượng vào các cụm.

Hình 3.13: Giải thuật phân cụm dựa trên wavelet

Độ phức tạp tính toán của giải thuật này là $O(N)$ với N là số các đối tượng trong cơ sở dữ liệu.

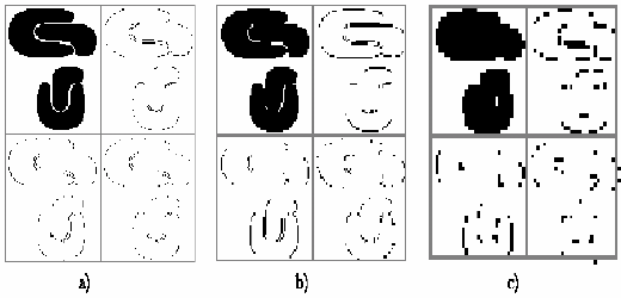


Hình 3.14: Một mẫu không gian đặc trưng 2 chiều

Ví dụ: Hình 3.14 (lấy từ Sheikholeslami, Chatterjee và Zhang (1998)) cho thấy một mẫu không gian đặc trưng 2 chiều, tại đó, mỗi điểm trong ảnh đại diện cho các giá trị đặc trưng của một đối tượng trong các tập dữ liệu không gian. Hình 3.15 (lấy từ Sheikholeslami, Chatterjee và Zhang (1998)) cho thấy kết quả của các phép biến đổi wavelet tại các tỷ lệ khác nhau, từ mịn (tỷ lệ 1) cho tới thô (tỷ lệ 3). Tại mỗi mức, dải con LL (bình thường) chỉ ra tại cung phần tư phía trên bên trái, dải con LH (các cạnh nằm ngang) chỉ ra tại cung phần tư phía trên bên phải và dải con HL (các cạnh nằm dọc) chỉ ra tại cung phần tư phía dưới bên trái và dải con HH (các góc) chỉ ra tại cung phần tư phía dưới bên phải.

WaveCluster là một giải thuật dựa trên mật độ và lưới. WaveCluster thích hợp với tất cả các yêu cầu của các giải thuật phân cụm tốt: nó xử lý các tập dữ liệu lớn một cách hiệu quả, tìm ra các cụm với hình dạng tuỳ ý, thành công trong việc xử lý các outlier, và không nhạy cảm đối với trật tự đầu vào. So với

BIRCH, CLARANS và DBSCAN, WaveCluster làm tốt hơn các phương pháp này ở cả hiệu suất và chất lượng phân cụm.



Hình 3.15: Đa phân giải của không gian đặc trưng trong hình 3.14. a) tỷ lệ 1; b) tỷ lệ 2; c) tỷ lệ 3

3.7.3 CLIQUE: Phân cụm không gian số chiều cao

Một giải thuật phân cụm khác, CLIQUE, Agrawal et al. (1998), tích hợp phương pháp phân cụm dựa trên lưới và mật độ theo một cách khác. Nó rất hữu ích cho phân cụm dữ liệu với số chiều cao trong các cơ sở dữ liệu lớn.

Cho trước một tập lớn các điểm dữ liệu đa chiều, các điểm dữ liệu này thường nằm không đồng nhất trong không gian dữ liệu. Phân cụm dữ liệu nhận biết các vị trí thưa thớt hay đông đúc, do vậy tìm ra toàn bộ các mẫu phân bố của tập dữ liệu.

Một unit là dày đặc nếu như phần nhỏ của các điểm dữ liệu chứa trong unit vượt quá một tham số mô hình đầu vào. Một cụm là một tập lớn nhất các unit dày đặc có kết nối.

CLIQUE phân chia không gian dữ liệu *m* chiều thành các unit hình chữ nhật không chồng lên nhau, nhận biết các unit dày đặc, và tìm ra các cụm trong toàn bộ các không gian con của không gian dữ liệu gốc, sử dụng phương pháp phát sinh candidate (ứng cử) giống với giải thuật Apriori cho khai phá các luật kết hợp.

CLIQUE thực hiện phân cụm đa chiều theo hai bước:

Trước tiên, CLIQUE nhận biết các cụm bằng cách xác định các unit dày đặc trong toàn bộ các không gian con của các interest và sau đó xác định các unit dày đặc có kết nối trong toàn bộ các không gian con của các interest.

Một heuristic quan trọng mà CLIQUE thông qua đó là nguyên lý Apriori trong phân cụm số chiều cao: Nếu một unit *k* chiều là dày đặc thì các hình chiếu (project) của nó trong không gian (*k*-1) chiều cũng vậy. Đó là nếu bất kỳ unit thứ (*k*-1) không phải là dày đặc, thì unit thứ *k* tương ứng của nó không phải là một unit ứng cử dày đặc (candidate dense). Bởi vậy, tất cả các unit dày đặc *k* chiều ứng cử có thể được sinh từ các unit dày đặc (*k*-1) chiều.

Thứ hai, CLIQUE sinh ra mô tả tối thiểu cho các cụm như sau: Trước tiên nó xác định các miền tối đa phủ một cụm các unit dày đặc có kết nối cho mỗi cụm và sau đó xác định phủ tối thiểu cho mỗi cụm.

CLIQUE tự động tìm các không gian con số chiều cao nhất để các cụm mật độ cao tồn tại trong các không gian con này. Nó không nhạy cảm với trật tự các bản ghi trong đầu vào và không đoán được phân bố dữ liệu tiêu chuẩn. Nó tỷ lệ tuyến tính với kích thước của đầu vào và có một khả năng mở rộng tốt như số các chiều trong dữ liệu được tăng lên. Tuy nhiên, độ chính xác của kết quả phân cụm có thể bị suy giảm tại phụ phí bởi tính đơn giản của phương pháp.

3.8 Kết luận

Chương này đề cập tới các phương pháp phân cụm truyền thống và các cải tiến phương pháp phân cụm truyền thống. Ngoài ra chương này còn đề cập tới khái niệm độ không tương đồng (hay tương đồng) của các đối tượng. Qua đó ta có thể thấy được khả năng phân cụm của từng phương pháp, khả năng áp dụng vào các bài toán thực tiễn.

CHƯƠNG 4: CÀI ĐẶT THỬ NGHIỆM

Chương này đưa ra kết quả cài đặt thử nghiệm bằng các giải thuật Kmeans và Kmedoids trên các bộ dữ liệu của UCI và đánh giá kết quả thực nghiệm.

4.1 Thiết kế tổng thể

Chương trình gồm các khối chức năng chính sau:

- Khối chức năng tiền xử lý
- Khối chức năng phân cụm

4.1.1 Khối chức năng tiền xử lý

Nhiệm vụ của khối chức năng này là đọc dữ liệu, xác định số mẫu, số thuộc tính, số lớp, các giá trị thuộc tính của từng mẫu dữ liệu.

4.1.2 Khối chức năng phân cụm

Khối chức năng này tiến hành phân cụm các mẫu dữ liệu. Dữ liệu được học không giám sát (unsupervised learning) theo hai giải thuật khác nhau: Kmeans và Kmedoids. Cuối cùng gắn nhãn lớp cho các cụm. Sau khi gắn nhãn lớp cho các cụm sẽ tiến hành xác định hiệu quả phân lớp, phân loại.

4.2 Chuẩn bị dữ liệu

Dữ liệu đầu vào chương trình là các tệp văn bản và được chia thành hai loại:

- Tệp định dạng dữ liệu (*.names): Định nghĩa tên các lớp, tên các thuộc tính, các giá trị của từng thuộc tính, kiểu thuộc tính.

- Tệp mẫu dữ liệu (*.data): Gồm các mẫu dữ liệu chứa đầy đủ thông tin giá trị các thuộc tính và giá trị lớp.

4.2.1 Tệp định dạng dữ liệu

- Dòng 1: liệt kê các giá trị lớp. Các giá trị này cách nhau bởi dấu phẩy "," và kết thúc bằng dấu chấm ".".

- Từ dòng 2:

+ Mỗi mẫu một dòng

+ Bắt đầu bằng tên một thuộc tính, dấu ":", sau đó là các giá trị rời rạc của thuộc tính (nếu thuộc tính là xác thực hay nhị phân) hoặc kiểu thuộc tính (nếu thuộc tính có kiểu liên tục).

- Tất cả các phần chú thích được đặt sau dấu "|"

Bảng 4.1: Một ví dụ tệp định dạng dữ liệu *.names

```
1, 2, 3.
1: continuous.
2: 1, 2, 3, 4.          |categorical
3: continuous.
4: 0, 1.                |binary
```

4.2.2 Tệp mẫu dữ liệu

Mỗi mẫu một dòng. Các giá trị thuộc tính của mẫu ghi trước, cuối cùng là giá trị lớp. Mỗi một giá trị này cách nhau bởi dấu ",".

Bảng 4.2: Một ví dụ tệp dữ liệu *.data

```
0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,1,4
0,0,0,1,0,1,1,1,1,1,0,1,0,1,0,1,1
0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0,2
0,1,1,0,1,1,0,0,1,1,0,0,2,1,0,0,2
1,0,0,1,0,0,0,1,1,1,0,0,4,1,0,1,1
0,1,1,0,1,0,0,0,1,1,0,0,2,1,0,1,2
```

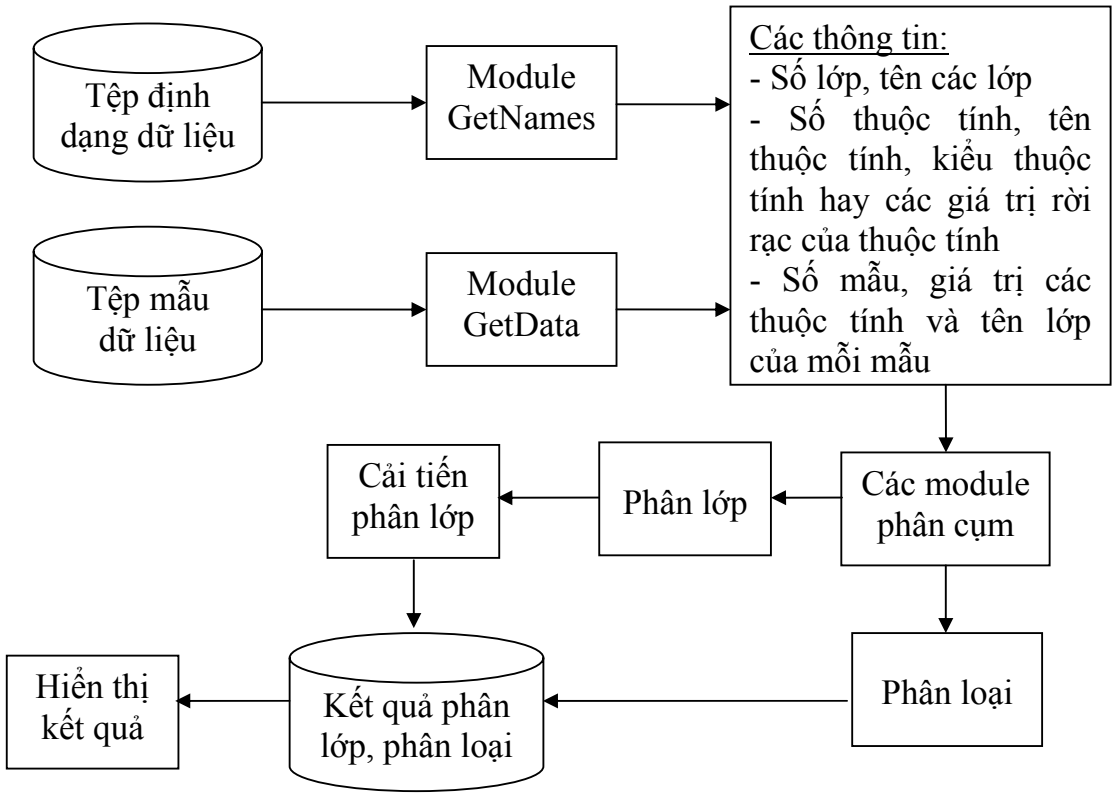
4.2.3 Nguồn dữ liệu

Trong khuôn khổ luận văn, dữ liệu được lấy từ địa chỉ web site:

- ftp://ftp.ics.uci.edu/pub/

4.3 Thiết kế chương trình

Với các khối chức năng và dữ liệu ở trên, chương trình được thiết kế như sau:



Hình 4.1: Thiết kế chương trình

4.4 Kết quả thực nghiệm và đánh giá

4.4.1 Các bước tiến hành thực nghiệm

- Phân cụm dữ liệu bằng giải thuật Kmeans và Kmedoids

- Gắn nhãn cho các cụm, đánh giá, so sánh hiệu quả gắn nhãn giữa hai giải thuật trên cho các bộ số liệu UCI (chỉ dùng các dữ liệu có thuộc tính liên tục).

- Gắn nhãn cho các cụm, đánh giá hiệu quả gắn nhãn cho dữ liệu có thuộc tính hỗn hợp

- Cải tiến hiệu quả phân lớp

- So sánh chất lượng phân loại với chương trình See5.

Chương trình See5 (phiên bản 2.03) là công cụ sử dụng kỹ thuật cây quyết định với giải thuật C5.0 dùng để phân loại dữ liệu được viết bởi Ross Quinlan. Tính hiệu quả của chương trình này đã được nhiều người công nhận. Vì thế, luận văn đã sử dụng nó làm công cụ để so sánh với các kết quả phân loại đã thực hiện. Hạn chế của See5 (phiên bản 2.03) chỉ dùng được tối đa 400 mẫu dữ liệu.

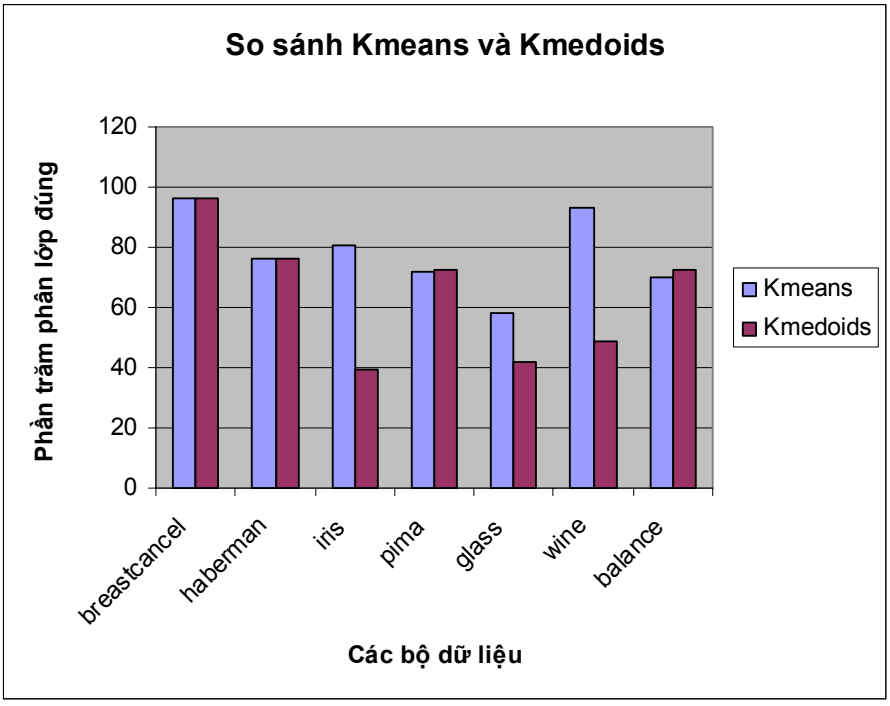
4.4.2 Thực nghiệm

Dưới đây là các kết quả đạt được:

4.4.2.1 Bài toán phân lớp: được thực hiện với số lượng các cụm là K = 2, 4, 6,8,10, 16. (Kmeans: ma; Kmedoids: md)

Bảng 4.3: Kết quả thí nghiệm phân lớp

Tên DL	Số mẫu	Số mẫu phân lớp đúng											
		K=2		K=4		K=6		K=8		K=10		K=16	
		ma	md	ma	md	ma	md	ma	md	ma	md	ma	md
Brea	500	328	480	485	481	484	481	481	482	481	482	481	482
Haber	306	225	225	229	226	230	231	228	232	233	234	237	240
Iris	150	100	51	126	53	126	55	125	57	121	59	142	65
Pima	768	532	504	539	537	541	528	525	554	554	558	558	561
Glass	214	78	82	105	84	117	86	117	88	125	90	140	96
Wine	178	107	72	173	80	169	85	168	84	166	87	173	93
Balan	625	293	369	407	423	448	441	503	451	438	453	483	459



Hình 4.2: Biểu đồ so sánh Kmeans và Kmedoids trong bài toán phân lớp với K=10

Biểu đồ trên cho thấy với dữ liệu kiểu liên tục khả năng phân lớp của Kmedoids trong bộ dữ liệu UCI thường thấp hơn so với Kmeans bởi điểm đại diện trong Kmedoids là một điểm đối tượng gần tâm cụm, tâm cụm trong

Kmeans là giá trị trung bình của các phần tử trong cụm. Nếu như dữ liệu ít nhiễu thì Kmeans sẽ cho kết quả hiệu quả hơn Kmedoids, trong trường hợp ngược lại, nếu một nhiễu với giá trị cực lớn, về cơ bản nó sẽ bóp méo phân bố dữ liệu nếu như dùng Kmeans, lúc này dùng Kmeadoids sẽ hiệu quả hơn. Theo biểu đồ so sánh ở trên ta nhận thấy dữ liệu ít nhiễu. Tuy nhiên, phép đo độ tương đồng của các đối tượng trong Kmedoids dường như chưa được hiệu quả lắm, do vậy phần trăm phân lớp đúng chưa được cao. Để cải thiện độ chính xác phân lớp, luận văn đưa ra phương pháp sau:

Với mỗi mẫu bị phân lớp sai trong mỗi cụm, ta sẽ đưa nó vào cụm thích hợp (giả sử là cụm A) nếu thoả mãn điều kiện:

+ Khoảng cách từ nó tới cụm hiện thời bằng khoảng cách tới cụm A

+ Nhãn lớp cụm A giống nhãn lớp của mẫu đó

+ Nếu thêm mẫu này vào cụm A, tâm cụm không thay đổi (hoặc thay đổi một khoảng cách epsilon đủ bé cho trước).

Thực nghiệm cho thấy độ chính xác phân lớp đã được tăng lên. Ví dụ ở một số bộ dữ liệu sau: (Cũ: C; Mới: M)

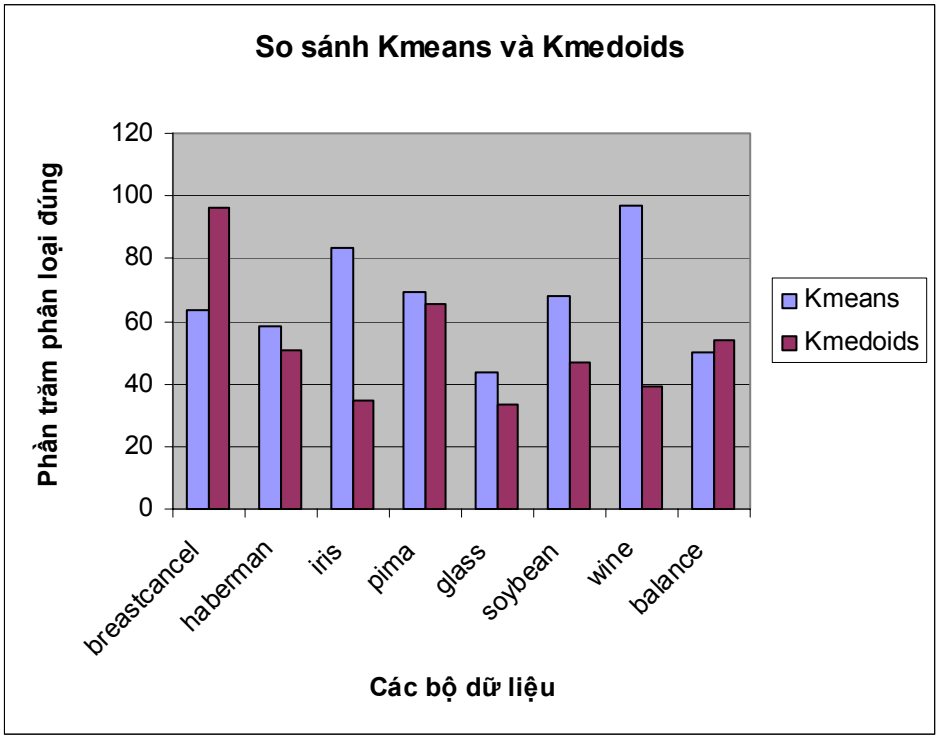
Bảng 4.4: Kết quả cải thiện chất lượng phân lớp

Tên DL		Iris	Wine	Balance	Haberman
K=4	C	53	80	423	226
	M	54	89	447	226
K=6	C	55	85	441	231
	M	57	85	459	231
K=8	C	57	84	451	232
	M	61	86	475	233
K=10	C	59	87	453	234
	M	65	90	477	237
K=16	C	65	93	459	240
	M	77	102	483	249
K=20	C	69	97	463	244
	M	85	110	487	257
K=25	C	74	102	468	249
	M	95	120	492	267

4.4.2.2 Bài toán phân loại

Bảng 4.5: Kết quả thí nghiệm phân loại của Kmeans và Kmedoids

Tên dữ liệu	Số mẫu	Số mẫu phân loại đúng		Tỷ lệ phân loại đúng (%)	
		ma	md	ma	md
Breastcancel	500	318	480	63.6	96
Haberman	306	179	115	58.4967	50.6536
Iris	150	125	52	83.3333	34.6667
Pima	768	532	504	69.2708	65.625
Glass	214	93	72	43.4579	33.6449
Soybean	47	32	22	68.0851	46.8085
Wine	178	172	70	96.6292	39.3258
Balance	625	313	336	50.08	53.76

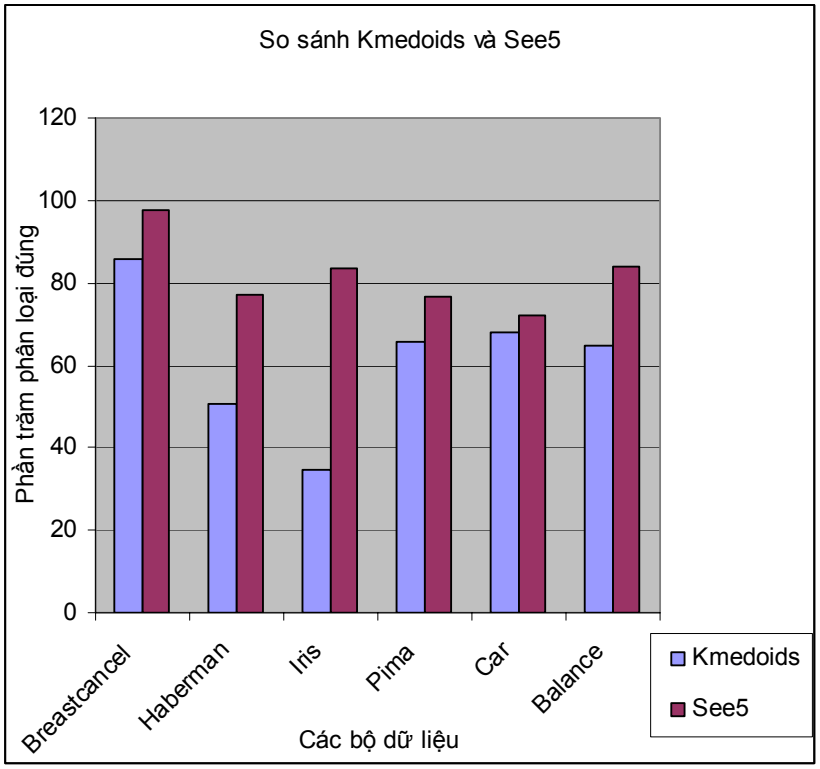


Hình 4.3: Biểu đồ so sánh Kmeans và Kmedoids trong bài toán phân loại

Bảng 4.6: Kết quả thí nghiệm phân loại của Kmedoids và See5

Tên dữ liệu	Số mẫu	Số mẫu phân loại đúng		Tỷ lệ phân loại đúng (%)	
		See5	md	See5	md
Breastcancel	400	391	344	97.75	86
Haberman	306	236	115	77.12418	50.6536

Iris	150	106	52	83.3333	34.6667
Pima	400	307	262	76.75	65.5
Car	298	289	202	72.25	67.7852
Balance	400	336	238	84	64.8501



Hình 4.4: Biểu đồ so sánh Kmedoids và See5 trong bài toán phân loại

Theo biểu đồ trên ta nhận thấy hiệu quả phân loại của See5 tốt hơn bởi nó có một mô hình phân loại dạng cây thực sự hiệu quả, mô hình này đã hạn chế được những nhánh phản ánh nhiễu nên chất lượng phân loại cao. Còn Kmedoids tuy đã xử lý được dữ liệu kiểu hỗn hợp nhưng chất lượng tính độ tương đồng của các đối tượng chưa cao nên khả năng phân loại kém hơn See5.

4.5 Kết luận

Như vậy, sau khi tiến hành thực nghiệm trên một số bộ dữ liệu của UCI ta nhận thấy kết quả phân lớp, phân loại các dữ liệu có thuộc tính liên tục của Kmeans tốt hơn so với Kmedoids. Với dữ liệu có thuộc tính hỗn hợp, Kmeans không xử lý được. Kmedoids với phương pháp tính độ tương đồng giữa hai mẫu do Ducker (1965) đề xuất, Kaufman và Rousseeuw cải tiến (1990) đã xử lý được dữ liệu này với độ chính xác trên trung bình và chi phí tính toán là $O(k(n\text{-}k)^2)$.

Đối với các giá trị n và k lớn, chi phí như vậy sẽ cao. Vậy nên việc cải tiến độ chính xác và tốc độ tính toán là hướng phát triển sau này.

KẾT LUẬN

Luận văn tập trung nghiên cứu lý thuyết và áp dụng một số kỹ thuật khai phá dữ liệu trên bộ dữ liệu của UCI. Đây là bước khởi đầu trong quá trình tìm hiểu những vấn đề cần quan tâm khi giải quyết các bài toán khai phá dữ liệu trong thực tế.

Trong khuôn khổ luận văn chưa áp dụng cụ thể vào một CSDL thực tế nào, mới chỉ dừng lại trên bộ dữ liệu UCI nên kết quả thực nghiệm chưa mang ý nghĩa thực tế. Tuy nhiên cũng có một số kết quả ban đầu là phát hiện tri thức từ bộ dữ liệu này.

Những kết quả mà luận văn đã thực hiện:

+ Về lý thuyết, luận văn tập trung tìm hiểu các kỹ thuật phân loại, phân cụm truyền thống và các phương pháp cải tiến chúng.

+ Về thực tiễn, luận văn đã đưa ra các kết quả cài đặt thử nghiệm trên bộ dữ liệu UCI bao gồm các kết quả phân loại, phân lớp, cải tiến chất lượng phân lớp.

Qua quá trình thực nghiệm và nghiên cứu lý thuyết có thể đưa ra một số kết luận như sau:

- Mỗi một giải thuật phân loại, phân cụm áp dụng cho một số mục tiêu và kiểu dữ liệu nhất định.
- Mỗi giải thuật có một mức độ chính xác riêng và khả năng thực hiện trên từng kích thước dữ liệu là khác nhau. Điều này còn tuỳ thuộc vào cách thức tổ chức dữ liệu ở bộ nhớ chính, bộ nhớ ngoài... của các giải thuật.
- Khai phá dữ liệu sẽ hiệu quả hơn khi bước tiền xử lý, lựa chọn thuộc tính, mô hình được giải quyết tốt.

Với những gì mà luận văn đã thực hiện, các hướng phát triển sau này của luận văn như sau:

- Độ chính xác phân lớp, phân loại phụ thuộc vào nhiều yếu tố như chất lượng dữ liệu, thuật toán cài đặt, phương pháp tính độ tương đồng của các đối tượng dữ liệu. Ngoài ra, các giá trị khuyết hay các thuộc tính dư thừa cũng phần nào làm ảnh hưởng đến chúng. Vì vậy hướng phát triển sau này là xử lý các giá trị khuyết, phát hiện và loại bỏ các thuộc tính dư thừa, cải tiến phương pháp tính độ tương đồng,... nhằm nâng cao chất lượng và tốc độ phân lớp, phân loại.
- Tiến hành cài đặt và tiếp tục nghiên cứu nhiều kỹ thuật khai phá dữ liệu hơn nữa, đặc biệt là triển khai giải quyết các bài toán cụ thể trong thực tế.

TÀI LIỆU THAM KHẢO

1. Anil K. Jain and Richard C. Dubes (1988), *Algorithms for clustering data*, Prentice-Hall, Inc., USA.
2. Ho Tu Bao (1998), *Introduction to knowledge discovery and data mining*.
3. Jiawei Han and Micheline Kambel (2000), *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers.
4. Joydeep Ghosh (2003), *Scalable Clustering*, Chapter 10, pp. 247-278, Formal version appears in: The Handbook of Data Mining, Nong Ye (Ed).
5. J.Ross Quinlan (1993), *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers.
6. Mercer (2003), *Clustering large datasets*, Linacre College.
7. Pavel Berkhin, *Survey of Clustering Data Mining Techniques*. Accrue Software, Inc., San Jose.