BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

PHAN ĐỨC GIẢI

KHẢO SÁT ẢNH HƯỞNG CỦA CÁC ĐỘ ĐO LỢI ÍCH LÊN ĐỘ CHÍNH XÁC TRONG BÀI TOÁN PHÂN LỚP DỰA TRÊN LUẬT KẾT HỢP

LUẬN VĂN THẠC SĨ

Chuyên ngành: Công Nghệ Thông Tin

Mã ngành: 60480201

TP. HỒ CHÍ MINH, tháng 09 năm 2015

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

PHAN ĐỨC GIẢI

KHẢO SÁT ẢNH HƯỞNG CỦA CÁC ĐỘ ĐO LỢI ÍCH LÊN ĐỘ CHÍNH XÁC TRONG BÀI TOÁN PHÂN LỚP DỰA TRÊN LUẬT KẾT HỢP

LUẬN VĂN THẠC SĨ

Chuyên ngành: Công Nghệ Thông Tin

Mã ngành: 60480201

HƯỚNG DẪN KHOA HỌC: TS. VÕ ĐÌNH BẢY

TP. HỒ CHÍ MINH, tháng 09 năm 2015

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH TẠI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP.HCM

Cán bộ hướng dẫn khoa học: **TS. Võ Đình Bảy**

Luận văn Thạc sĩ được bảo vệ tại Trường Đại học Công nghệ TP. HCM ngày 17 tháng 10 năm 2015.

Thành phần Hội đồng đánh giá Luận văn Thạc sĩ gồm:

(Ghi rõ họ, tên, học hàm, học vị của Hội đồng chấm bảo vệ Luận văn Thạc sĩ)

TT	Họ và tên	Chức danh Hội đồng
1	PGS.TS. Lê Hoài Bắc	Chủ tịch
2	GS.TSKH. Hoàng Văn Kiếm	Phản biện 1
3	TS. Cao Tùng Anh	Phản biện 2
4	TS. Hồ Đắc Nghĩa	Ủy viên
5	TS. Vũ Thanh Hiền	Ủy viên, Thư ký

Xác nhận của Chủ tịch Hội đồng đánh giá luận văn sau khi luận văn đã được sửa chữa (nếu có).

Chủ tịch Hội đồng đánh giá luận văn

TRƯỜNG ĐH CÔNG NGHỆ TP. HCM
PHÒNG QLKH – ĐTSĐH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

TP. HCM, ngày 08 tháng 03 năm 2015

NHIỆM VỤ LUẬN VĂN THẠC SĨ

Họ tên học viên:	Phan Đức Giải	Giới tính:	Nam
Ngày, tháng, năm sinh:	25/12/1988	Nơi sinh:	Đồng Nai
Chuyên ngành:	Công nghệ thông tin	MSHV:	1441860008

I- Tên đề tài:

Khảo sát ảnh hưởng của các độ đo lợi ích lên độ chính xác trong bài toán phân lớp dựa trên luật kết hợp.

II- Nhiệm vụ và nội dung:

- Nghiên cứu thuật toán CAR-Miner và thuật toán CARIM.
- Tìm hiểu về các độ đo lợi ích và kỹ thuật kiểm tra chéo (k-fold cross-validation).
- Nghiên cứu cách thức áp dụng các độ đo lợi ích để khai thác CARs.
- Thực nghiệm khảo sát các độ đo lợi ích lên độ chính xác trong khai thác CARs.

III- Ngày giao nhiệm vụ: 08/03/2015

IV- Ngày hoàn thành nhiệm vụ: 17/09/2015

V- Cán bộ hướng dẫn: *(Ghi rõ học hàm, học vị, họ, tên)* **TS. VÕ ĐÌNH BẢY**

CÁN BỘ HƯỚNG DẪN
(Họ tên và chữ ký)

KHOA QUẢN LÝ CHUYÊN NGÀNH
(Họ tên và chữ ký)

TS. VÕ ĐÌNH BẢY

LỜI CAM ĐOAN

Công trình nghiên cứu đề tài luận văn này là do chính tôi thực hiện, tôi cam đoan không sao chép bất kỳ dữ liệu nào từ các công trình nghiên cứu khác. Tất cả những tham khảo từ các nghiên cứu có liên quan đều được nêu rõ nguồn gốc sử dụng, danh mục các tài liệu tham khảo có nêu rõ trong luận văn.

Nội dung luận văn có tham khảo và sử dụng các tài liệu, thông tin được đăng tải trên các tác phẩm, tạp chí và các trang web theo danh mục tài liệu của luận văn.

Tác giả luận văn

Phan Đức Giải

LỜI CẢM ƠN

Lời đầu tiên, em xin bày tỏ lòng biết ơn sâu sắc đến Thầy, TS. Võ Đình Bảy bởi nhờ sự động viên, chỉ bảo tận tình, truyền đạt những kiến thức mới cũng như tạo mọi điều kiện tốt nhất để em có thể hoàn thành luận văn này.

Em cũng xin gửi lời cảm ơn đến quý Thầy/Cô trong khoa Công nghệ Thông tin trường Đại học Công Nghệ Tp. HCM đã động viên và hỗ trợ em rất nhiều kiến thức quý báu giúp em hoàn thành tốt luận văn.

Em cũng xin cảm ơn quý Thầy/Cô, Anh/Chị làm việc tại Phòng Sau Đại học đã hỗ trợ em rất nhiều về các thủ tục văn bản, giấy tờ liên quan đến luận văn.

Xin cảm ơn gia đình, đồng nghiệp, bạn bè đã động viên em trong suốt thời gian thực hiện luận văn này.

Tp. Hồ Chí Minh, ngày 17 tháng 09 năm 2015

Học viên Phan Đức Giải

TÓM TẮT

Đề tài "Khảo sát ảnh hưởng của các độ đo lợi ích lên độ chính xác trong bài toán phân lớp dựa trên luật kết hợp" nhằm khảo sát độ chính xác trong bài toán khai thác CARs với các độ đo lợi ích khác nhau.

Đề tài sử dụng kỹ thuật kiểm tra chéo (k-fold-cross-validation) để tính độ chính xác phân lớp của các, các mẫu ban đầu được chia thành k fold với kích thước bằng nhau. Trong k fold, một fold duy nhất được giữ lại như là dữ liệu xác nhận để thử nghiệm, và k - 1 fold còn lại được sử dụng như dữ liệu huấn luyện. Quá trình kiểm tra được lặp lại k lần, với mỗi k fold được dùng duy nhất một lần như dữ liệu xác nhận, tập dữ liệu huấn luyện k – 1 fold sẽ dùng thuật toán CARIM và áp dụng các độ đo lợi ích để tạo ra tập luật phân lớp, dùng tập luật được tạo ra từ dữ liệu huấn luyện k – 1 để kiểm tra mẫu thử nghiệm có được phân lớp đúng. Cuối cùng ta có được số mẫu phân lớp đúng và tính độ chính xác.

ABSTRACT

The research topic "The survey effect of Interestingness Measures to the accuracy of classification problem based on association rules" survey accuracy for the CARs with interestingness measures.

This study using k-fold cross-validation to calculate accuracy classification of database, the original sample is randomly partitioned into k equal sized fold. In k fold, a single fold is retained as the validation data for testing the model, and the remaining k − 1 fold are used as training data. The cross-validation process is then repeated k times, with each of the k fold used exactly once as the validation data, with k-1 training data used algorithm CARIM and apply interestingness measures to generate classification rules which for check validation data. Finally we get some samples correctly classified and accuracy.

MỤC LỤC

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH

DANH MỤC CÁC TỪ VIẾT TẮT

STT	Tên viết tắt	Nghĩa tiếng việt	Tên đầy đủ
1	CAR	Luật phân lớp kết hợp	Class Association Rules
2	CBA	Phân lớp dựa trên sự kết hợp	Classification Based on Associations
3	ILA	Thuật toán học quy nạp	Inductive Learning Algorithm
4	CMAR	Phương pháp không sinh thêm ứng viên	Classification based on Multiple Association Rules
5	MMAC	Phương pháp tổ chức phân loại đa lớp	Multi-class Multi-label Associative Classification
6	CSDL	Cơ sở dữ liệu	Database
7	CPAR	Phân lớp dựa trên luật kết hợp dự đoán	Classification based on predictive association rules
8	MCAR	Phân lớp dựa trên luật kết hợp đa lớp	Multi-class classification based on association rule
9	ECR	Luật lớp tương đương	Equivalence Class Rule
10	MECR	Luật lớp tương đương chỉnh sửa	Modification Equivalence Class Rule
11	cross-validation	Kiểm tra chéo	
12	MinSup	Ngưỡng hỗ trợ	Minimum support
13	MinConf	Ngưỡng tin cậy	Minimum confidence

CHƯƠNG 1: MỞ ĐẦU

1.1 Đặt vấn đề

Trong quá trình hoạt động, con người tạo ra nhiều dữ liệu nghiệp vụ. Các tập dữ liệu được tích lũy có kích thước ngày càng lớn, và có thể chứa nhiều thông tin ẩn dạng những quy luật chưa được khám phá. Chính vì vậy, một nhu cầu đặt ra là cần tìm cách trích rút từ tập dữ liệu đó các luật về phân lớp dữ liệu hay dự đoán những xu hướng dữ liệu tương lai. Những quy tắc nghiệp vụ thông minh được tạo ra sẽ phục vụ đắc lực cho các hoạt động thực tiễn, cũng như phục vụ đắc lực cho quá trình nghiên cứu khoa học. Công nghệ phân lớp và dự đoán dữ liệu ra đời để đáp ứng mong muốn đó.

Công nghệ phân lớp dữ liệu đã, đang và sẽ phát triển mạnh mẽ trước những khao khát tri thức của con người. Trong những năm qua, phân lớp dữ liệu đã thu hút sự quan tâm các nhà nghiên cứu trong nhiều lĩnh vực khác nhau như học máy, hệ chuyên gia, thống kê, v.v. Công nghệ này cũng ứng dụng trong nhiều lĩnh vực thực tế như: thương mại, nhà băng, maketing, nghiên cứu thị trường, bảo hiểm, y tế, giáo dục, v.v.

Một trong những chức năng được đề cập rất nhiều trong khai thác dữ liệu là khai thác sự kết hợp giữa các mẫu trong dữ liệu hay còn gọi là luật kết hợp. Trong thời kỳ đầu luật kết hợp chỉ đơn giản là khai thác sự hiện diện của một mẫu A thì dẫn đến sự xuất hiện mẫu B. Sau đó, luật kết hợp được phát triển để khai thác quan hệ có thuộc tính số lượng giữa các mẫu và được gọi là luật kết hợp số lượng. Một số khái niệm được bổ sung vào dữ liệu để khai thác luật kết hợp ở mức tổng quát, v.v. Khai thác luật kết hợp là một trong những phương pháp khai thác tri thức từ cơ sở dữ liệu (CSDL) và đã nhận được nhiều sự quan tâm trong giới khoa học máy tính và công nghệ tri thức. Thuật toán nổi tiếng là Apriori do tác giả Agrawal cùng các đồng sự đề xuất năm 1994 [4], ban đầu nó được ứng dụng vào việc khai thác luật kết hợp trong lĩnh vực thương mại. Luật kết hợp không chỉ dừng lại những ứng

dụng trong thương mại mà đã có những ứng dụng rộng rãi trong các lĩnh vực khác như trong y khoa, quản lý, thương mại và công nghiệp, v.v.

1.2 Tính cấp thiết của đề tài

Gần đây, một phương pháp mới về phân lớp trong khai thác dữ liệu được gọi là phân lớp dựa trên sự kết hợp (CBA [6]), được đưa ra để khai thác luật phân lớp kết hợp (CARs). Phương pháp này thường có độ chính xác cao hơn so với phương pháp C4.5 [3] và ILA [7]. Vì vậy một số thuật toán để khai thác luật phân lớp dựa trên khai thác luật kết hợp được phát triển trong những năm gần đây như : phân lớp dựa trên luật kết hợp đoán trước [12], phân lớp dựa trên nhiều luật kết hợp [10], phân lớp dựa trên sự kết hợp [6], phân lớp đa lớp dựa trên luật kết hợp [17], v.v.

Tuy nhiên những phương pháp trên chỉ tập trung chủ yếu trong việc xây dựng thuật toán phân lớp dựa trên luật kết hợp hoặc xây dựng luật phân lớp mà không thảo luận nhiều về vấn đề thời gian thực thi (khai thác) của các thuật toán. Hơn thế nữa, khai thác phân lớp dựa trên luật kết hợp (CARs) tiêu tốn rất nhiều thời gian bởi vì nó khai thác một tập đầy đủ các luật thỏa ngưỡng. Vì thế, cải thiện thời gian khai thác phân lớp dựa trên luật kết hợp là một trong những vấn đề chính cần được giải quyết.

Năm 2013, Nguyen và các đồng sự đã đề xuất thuật toán CAR-Miner nhằm cải tiến thời gian khai thác phân lớp dựa trên luật kết hợp [29]. Thuật toán xây dựng cấu trúc cây và sử dụng một số lý thuyết để cắt tỉa các node nhằm giảm thiểu thời gian xử lý trong quá trình duyệt các tập dữ liệu.

Hiện tại, các thuật toán khai thác luật phân lớp kết hợp hầu hết tập trung vào độ phổ biến và độ tin cậy của luật. Một số thuật toán cũng cải tiến độ chính xác bằng cách đưa ra các độ đo. Tuy nhiên, chưa có công trình nào nghiên cứu sự ảnh hưởng của các độ đo lợi ích lên độ chính xác của bộ phân lớp. Số lượng các độ đo hiện nay lên đến hơn ba mươi và mỗi độ đo có một số điểm mạnh riêng. Chính vì vậy, việc khảo sát sự ảnh hưởng của các độ đo lợi ích lên độ chính xác phân lớp rất cấp thiết giúp cho việc chọn lựa độ đo phù hợp đối với các CSDL.

1.3. Mục tiêu của đề tài

- Đề tài nghiên cứu thuật toán CAR-Miner [29], thuật toán CARIM [32], các độ đo lợi ích và khảo sát ảnh hưởng của các độ đo lợi ích lên độ chính xác trong bài toán phân lớp dựa vào luật kết hợp sử dụng kỹ thuật k-fold cross-validation.

1.4. Nội dung nghiên cứu

- Nghiên cứu bài toán phân lớp dựa vào luật kết hợp dùng thuật toán CAR-Miner [29], thuật toán CARIM [32] và áp dụng các độ đo lợi ích để tạo ra các tập luật.
- Khảo sát ảnh hưởng của các độ đo lợi ích lên độ chính xác trong bài toán phân lớp dựa vào luật kết hợp sử dụng kỹ thuật k-fold cross-validation.

1.5. Phương pháp luận và phương pháp nghiên cứu

- Tìm hiểu các tài liệu trong và ngoài nước về khai thác luật phân lớp và luật kết hợp. Nghiên cứu thuật toán CAR-Miner [29] trong bài toán phân lớp dựa vào luật kết hợp và áp dụng các độ đo lợi ích đẻ tạo ra các tập luật trong thuật toán CARIM [32].
- Tìm hiểu các độ đo lợi ích và khảo sát ảnh hưởng của các độ đo lợi ích lên độ chính xác trong bài toán phân lớp dựa vào luật kết hợp sử dụng kỹ thuật k-fold cross-validation.

CHƯƠNG 2: TỔNG QUAN VỀ LĨNH VỰC NGHIÊN CỨU

2.1. Khai thác luật phân lớp

Bài toán khai thác luật phân lớp kết hợp có thể được phát biểu tóm tắt như sau:

Cho cơ sở dữ liệu D với các thuộc tính là $\{A_1, A_2, \ldots, A_n\}$ và thuộc tính phân lớp C, trong đó A_1 chứa các giá trị $\{a_{i1}, a_{i2}, \ldots, a_{im}\}$, $C = \{c_1, c_2, \ldots, c_k\}$ (k lớp) là các nhãn lớp. Dựa vào tập dữ liệu đã cho, thuật toán tìm luật phân lớp sẽ tìm ra các luật của dữ liệu từ đó hình thành được bộ phân lớp và dựa vào bộ phân lớp đó ta có thể dự đoán được lớp của mẫu mới.

Đã có nhiều phương pháp được phát triển để phân lớp dữ liệu như cây quyết định (ID3 [1], C4.5 [3]), học quy nạp (ILA [7]), v.v. Cách tiếp cận này có ưu điểm là số luật sinh ra ít, đơn giản, dễ hiểu nhưng độ chính xác thường không cao. Năm 1998, khai thác luật phân lớp kết hợp được đề nghị, đây là phương pháp kết hợp giữa phân lớp và luật kết hợp [10]. Phương pháp này được đánh giá là có độ chính xác cao hơn so với các phương pháp phân lớp dựa trên luật trước đó. Một số thuật toán điển hình như CBA [10], CMAR [10], MMAC [15], ECR-CARM [21], CAR-Miner [29], CARIM[32].

2.2. Khai thác luật kết hợp

Luật kết hợp là dạng luật biểu diễn tri thức ở dạng tương đối đơn giản. Mục tiêu của phương pháp này là phát hiện và đưa ra các mối liên hệ giữa các giá trị dữ liệu trong cơ sở dữ liệu. Mẫu đầu ra của giải thuật khai thác dữ liệu là tập luật kết hợp.

Tuy luật kết hợp là một dạng luật khá đơn giản nhưng lại mang rất nhiều ý nghĩa. Thông tin mà luật này đem lại rất có lợi trong các hệ hỗ trợ ra quyết định giúp ta tìm được những luật kết hợp đặc trưng và mang nhiều thông tin từ cơ sở dữ

liệu tác nghiệp là một trong những hướng tiếp cận chính của lĩnh vực khai thác dữ liệu.

Bảng 2.1. mô tả một ví dụ minh họa về bài toán khai thác luật kết hợp. Một tập dữ liệu gồm 5 giao dịch chứa các mặt hàng {Bánh mì, Bơ, Sữa, Trứng, Bia} và bên phải là ví dụ về hai luật kết hợp được sinh ra từ tập dữ liệu bên trái.

Bảng 2.1. Khai thác luật kết hợp.

STT	Mặt hàng		Luật kết hợp
1	Bánh mì, Bơ, Sữa		
2	Bia, Bánh mì		
3	Bia, Bơ, Trứng, Sữa	➡	{Bơ} → { Bánh mì} {Bánh mì, Sữa} → {Bia}
4	Bia, Bánh mì, Trứng, Sữa		
5	Bơ, Trứng, Sữa		

2.3. Khai thác luật phân lớp dựa vào khai thác luật kết hợp

Khai thác luật phân lớp dựa vào khai thác luật kết hợp là tìm một tập con của các luật kết hợp có trong cơ sở dữ liệu. Tập con này chứa vế phải là giá trị của thuộc tính lớp. Bài toán được phát biểu như sau:

Cho một CSDL gồm N mẫu và L thuộc tính phân biệt đã được chuẩn hóa, nghĩa là các giá trị đã được biến đổi và rời rạc hóa theo một cách nào đó cho nhất quán. Ban đầu, một thuộc tính có thể là rời rạc hay liên tục và giả sử rằng tất cả các thuộc tính đều cùng kiểu. Đối với thuộc tính rời rạc, tất cả các giá trị được ánh xạ thành một tập các số nguyên dương liên tiếp nhau. Đối với thuộc tính liên tục, giá trị của nó được chia thành các khoảng rời rạc, và các khoảng này được ánh xạ

thành các số nguyên dương liên tiếp. Với sự ánh xạ này, đề tài có thể xem dữ liệu như một cặp các thuộc tính hoặc giá trị nguyên và một nhãn lớp. Cặp thuộc tính, giá trị nguyên được gọi là một item.

Cho cơ sở dữ liệu D, I là tập tất cả các item trong D và C là tập các nhãn lớp

Do đó mẫu d $\in$ D chứa tập X $\subseteq$ I.

Luật phân lớp kết hợp là phát biểu có dạng X $\rightarrow$ y trong đó X $\subseteq$ I và y $\in$ C. Độ tin cậy của luật là c nếu c% mẫu trong D chứa X được gán nhãn là lớp y. Độ phổ biến của luật là s nếu có s% mẫu trong D chứa X được gán nhãn là lớp y.

Mục tiêu của khai thác luật phân lớp dựa vào khai thác luật kết hợp là:

- Sinh ra tập CARs đầy đủ thỏa ngưỡng độ phổ biến tối thiểu và ngưỡng độ tin cậy tối thiểu.
- Xây dựng bộ phân lớp từ CARs.

2.4. Độ đo lợi ích

Tạo ra các luật trong luật kết hợp hoặc với phân lớp kết hợp có thể dẫn đến một tập hợp rất lớn các quy tắc mà làm cho nó không thể, cho dù các miền chuyên gia, để nghiên cứu. Có hàng ngàn hoặc thậm chí hàng triệu các luật, không tránh khỏi có một số không liên quan, không thực tế. Để giải quyết vấn đề này, các độ đo lợi ích có thể được sử dụng để lọc hoặc xếp hạng các luật.

Có nhiều độ đo lợi ích khác nhau được dùng rộng rãi trong học máy, khai thác dữ liệu và thống kê. Năm 1991, Piatetsky–Shapiro đề xuất thống kê độc lập các luật, là độ đo lợi ích [2]. Sau đó, nhiều độ đo được đề xuất. Năm 1994, Agrawal và Srikant đề xuất độ hỗ trợ và độ tin cậy cho khai thác luật kết hợp [4]. Thuật toán Apriori khai thác các luật cũng được thảo luận [4]. Lift và χ^2 như độ đo tương quan được đề xuất [5]. Hilderman và Hamilton so sánh sự khác nhau của các độ đo lợi ích và giải quyết các khái niệm của giao dịch null [30]. Omiecinski [14], Lee và các cộng sự [13] giải quyết tất cả độ tin cậy, tính mạch lạc, và họ có được độ đo tốt cho

khai thác các luật tương quan trong cơ sở dữ liệu giao dịch. Tan và các cộng sự thảo luận thuộc tính của 21 độ đo lợi ích và phân tích tác động của các ứng cử viên cắt tỉa dựa trên ngưỡng hỗ trợ [11]. Shekar và Natarajan đề xuất 3 độ đo cho việc nhận được các mối quan hệ giữa cặp item [16]. Bên cạnh, việc tạo ra nhiều độ đo, một số nghiên cứu được đề xuất làm thế nào để chọn ra độ đo cho một cơ sở dữ liệu [27, 28, 25].

Nhiều độ đo được đề xuất như support, confidence, cosine, lift, chi-square, gini-index, Laplace, phi-coefficient (khoảng 35 độ đo [20]). Mặc dù chúng khác nhau từ công thức, họ đều dùng bốn yếu tố để tính toán giá trị độ đo của luật $X \rightarrow Y$: n; n_X; n_Y; n_{XY}, trong đó n là tổng số giao dịch, n_X là số giao dịch chứa X, n_Y là số giao dịch chứa Y, n_{XY} là số giao dịch chứa cả X và Y. Một số yếu tố cho việc tính toán giá trị độ đo được xác định thông qua n, n_X, n_Y, n_{XY} như sau: $n_{\overline{X}} = n - n_X$, $n_{\overline{Y}} = n - n_Y$, $n_{X\overline{Y}} = n_X - n_{XY}$, $n_{\overline{XY}} = n_Y - n_{XY}$, $n_{\overline{XY}} = n - n_{XY}$.

Bảng 2.2: Một cơ sở dữ liệu mẫu để tính độ đo lợi ích.

TID	Item bough
1	A,C,T,W
2	C,D,W
3	A,C,T,W
4	A,C,D,W
5	A,C,D,T,W
6	C,D,T

Chúng ta có thể tính toán giá trị độ đo như sau: đặt vm(n, n_X, n_Y, n_{XY}) là giá trị độ đo của luật $X \rightarrow Y$, giá trị vm có thể được tính toán khi chúng ta biết độ đo cần tính toán dựa trên (n, n_X, n_Y, n_{XY}).

Ví dụ: xem xét cơ sỡ dữ liệu mẫu từ bảng 2.2 với X = AC, Y = TW $\Rightarrow$ n = 6, n_X = 4, n_Y = 3, n_{XY} = 3 $\Rightarrow$ $n_{\overline{X}} = 2$, $n_{\overline{Y}} = 3$. Chúng ta có giá trị một số độ đo trong bảng 2.3.

Bảng 2.3: Một số giá trị độ đo với luật $AC \rightarrow TW$.

Độ đo	Miền giá trị	Công thức	n	n_X	n_Y	n_{XY}	Ví dụ
Confidence	0..1	$\frac{n_{XY}}{n_X}$	6	4	3	3	$\frac{3}{4}$
Cosine	0..1	$\frac{n_{XY}}{\sqrt{n_X n_Y}}$	6	4	3	3	$\frac{3}{\sqrt{4\times 3}}$
Lift	0..∞	$\frac{n_{XY} n}{n_X n_Y}$	6	4	3	3	$\frac{3\times 6}{4\times 3} = \frac{3}{2}$
Rule interest	0..∞	$n_{XY} - \frac{n_X n_Y}{n}$	6	4	3	3	$3 - \frac{4\times 3}{6} = 1$
Laplace	0..1	$\frac{n_{XY}+1}{n_X+2}$	6	4	3	3	$\frac{4}{6}$
Jaccard	0..1	$\frac{n_{XY}}{n_X + n_Y - n_{XY}}$	6	4	3	3	$\frac{3}{6+3-3} = \frac{3}{4}$

CHƯƠNG 3: THUẬT TOÁN CAR-Miner và CARIM

3.1. Giới thiệu tổng quan

Luật phân lớp đóng vai trò quan trọng trong các hệ thống ra quyết định, chính vì vậy, có rất nhiều phương pháp được phát triển như C4.5 [3], ILA [7]. Các phương pháp này dựa trên kỹ thuật heuristic/tham lam nên độ chính xác thường chưa cao. Chính vì vậy năm 1998, Liu và các đồng sự đề xuất phương pháp phân lớp dựa vào khai thác luật kết hợp, được gọi là phân lớp kết hợp (CBA [10]). Phương pháp này thường có độ chính xác cao hơn C4.5 [3], ILA [7]. Lý do chính là nhờ nó khai thác tập luật đầy đủ hơn C4.5 [3], ILA [7], có thể sử dụng đa luật để dự đoán nhãn của mẫu mới. Một số phương pháp nhằm nâng cao hiệu quả khai thác được đề nghị về sau như phân lớp dựa trên luật kết hợp dự đoán (CPAR [12]), phân lớp dựa trên luật kết hợp đa nhãn (CMAR [10]), phân lớp dựa trên luật kết hợp đa lớp, đa nhãn (MMAC [15]), phân lớp dựa trên luật kết hợp đa lớp (MCAR [17]), khai thác luật phân lớp kết hợp dựa trên lớp tương đương và cây ECR (ECR-CARM [21]) và khai thác luật phân lớp kết hợp dựa trên cây MECR (CAR-Miner [29]). Một số nghiên cứu chỉ ra bộ phân lớp được tạo ra từ luật phân lớp kết hợp thường có độ chính xác cao hơn các phương pháp truyền thống như C4.5 [3], ILA [7] cả về lý thuyết lẫn thực nghiệm. Khai thác CARs dựa trên các độ đo lợi ích [32].

Năm 2008, Vo và Le đã trình bày một cách tiếp cận để khai thác CARs với chỉ một lần quét CSDL. Các tác giả đã phát triển cây có tên là ECR-tree và dựa vào đó, thuật toán ECR-CARM [21] cũng đã được phát triển. Năm 2013, MECR-tree, một cấu trúc cây mở rộng từ ECR-tree được phát triển[29], các tác giả dựa vào MECR-tree để phát triển thuật toán CAR-Miner [29],thuật toán CARIM [32] và áp dụng các độ đo lợi ích để khai thác nhanh CARs.

3.2. Các định nghĩa và mệnh đề

Gọi D là tập các dữ liệu huấn luyện với n thuộc tính $A_1, A_2, \ldots, A_n$ và |D| đối tượng (mẫu). Gọi $C = \{c_1, c_2, \ldots, c_k\}$ là tập các nhãn lớp. Mỗi giá trị của thuộc tính A_i và thuộc tính lớp C được ký hiệu bởi các ký tự thường a và c tương ứng.

+ Định nghĩa 3.1: Một itemset là một tập các cặp (thuộc tính, giá trị), được ký hiệu bởi $\{(A_{i1}, a_{i1}), (A_{i2}, a_{i2}), \ldots, (A_{im}, a_{im})\}$.

+ Định nghĩa 3.2: Một luật phân lớp kết hợp r có dạng $\{(A_{i1}, a_{i1}), (A_{i2}, a_{i2}), \ldots, (A_{im}, a_{im})\} \rightarrow c$, trong đó $\{(A_{i1}, a_{i1}), (A_{i2}, a_{i2}), \ldots, (A_{im}, a_{im})\}$ là một itemset và $c \in C$ là một nhãn lớp.

+ Định nghĩa 3.3: Khả năng xảy ra của luật r, kí hiệu ActOcc(r), là số dòng trên D chứa vế trái của r.

+ Định nghĩa 3.4: Đếm độ hỗ trợ của luật r, kí hiệu Sup(r), là số dòng trên D chứa vế trái và vế phải của r.

+ Định nghĩa 3.5: Độ tin cậy của luật r là tỉ số giữa Sup(r) chia cho ActOcc(r), nghĩa là $Conf(r) = \frac{Sup(r)}{Act0cc(r)}$.

Chẳng hạn, xét luật r: $\{(A, a_1)\} \rightarrow y$ từ CSDL ở bảng 3.1, ta có ActOcc(r) = 3 và sup(r) = 2. Do có 3 đối tượng có giá trị $A = a_1$, trong đó hai đối tượng có lớp là $y \Rightarrow$ Độ tin cậy của luật là conf (r) = 2/3.

+ Mệnh đề 3.1. Cho trước hai nút $\frac{att_1 \times value_1}{Obidset_1(c_{11},\ldots,c_{1k})}$ và $\frac{att_2 \times value_2}{Obidset_2(c_{21},\ldots,c_{2k})}$, nếu $att_1 = att_2$ và $value_1 \# value_2$, thì $Obidset_1 \cap Obidset_2 = \emptyset$.

Dựa vào mệnh đề 3.1, thuật toán không cần kết hợp hai itemset X và Y nếu chúng có cùng tập thuộc tính do sup(XY) = 0. Chẳng hạn, xét hai nút $\frac{1 \times a_1}{127(2,1)}$ và

$\begin{matrix}1\times a_2\\ 38(1,1)\end{matrix}$, trong đó Obidset ({(A, a_1)}) = 127 và Obidset ({(A, a_2)}) = 38 $\Rightarrow$ Obidset ({(A, a_1),(A, a_2)}) = Obidset ({(A, a_1)}) $\cap$ Obidset ({(A, a_2)}) = $\emptyset$. Tương tự ta có Obidset ({(A, a_1), (B, b_1)}) = 1 và Obidset ({(A, a_1); (B, b_2)}) = 2 $\Rightarrow$ Obidset ({(A, a_1), (B, b_1)}) $\cap$ Obidset ({(A, a_1); (B,b_2)}) = $\emptyset$.

+ **Mệnh đề 3.2.** Cho trước 2 nút $\begin{matrix}itemset_1\\ Obidset_1(c_{11},...,c_{1k})\end{matrix}$ và $\begin{matrix}itemset_2\\ Obidset_2(c_{21},...,c_{2k})\end{matrix}$, nếu $itemset_1 \subset itemset_2$ và $|Obidset_1| = |Obidset_2|$ thì $\forall i \in [1, k] : c_{1i} = c_{2i}$.

Từ mệnh đề 3.2, khi kết hai nút cha thành một nút con, có thể kiểm tra số phần tử của Obidset kết quả, nếu có kết quả bằng với một trong hai nút cha thì chép các thông tin của nút cha tương ứng cho nút con.

3.3. Cấu trúc cây MECR

Với cây MECR, mỗi nút trên cây chứa duy nhất một itemset với các thông tin nhưsau:

(a) Obitset: là tập các dòng dữ liệu chứa các itemset.

(b) (#c_1,#c_2,…,#c_k) là một danh sách các số nguyên, trong đó c_i là số các dòng trong Obidset thuộc về lớp c_i.

(c) pos - là một số nguyên dương lưu vị trí của lớp có số lần xuất hiện nhiều nhất. Nghĩa là $pos = \arg\max_{1\in[1,k]}\{\#c_i\}$.

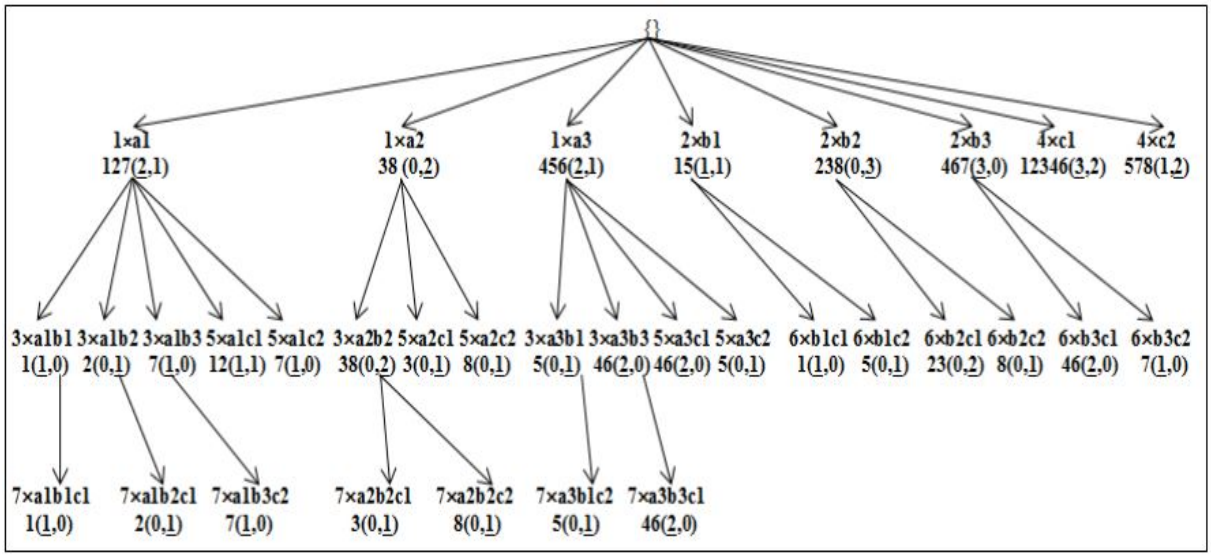
Chẳng hạn, xét nút chứa itemset X = {(A, a_3), (B, b_3)} từ bảng 3.1. Do X chứa trong các đối tượng 4, 6 và tất cả đều thuộc về lớp y. Vì vậy, nút $\begin{Bmatrix}(A,a_3),(B,b_3)\\ 46(2,0)\end{Bmatrix}$ (hay viết đơn giản là $\begin{matrix}3\times a_3b_3\\ 46(2,0)\end{matrix}$) được tạo ra trên cây, pos = 1 (được gạch chân tại vị trí 1 của nút này) do số đếm thuộc về lớp y là lớn nhất (2 so với 0). Biểu diễn sau khác cách biểu diễn đầu ở chỗ lưu trữ tập thuộc tính. Biểu diễn đầu

lưu đầy đủ tập thuộc tính trong khi biểu diễn sao lưu tập thuộc tính dưới dạng bit cho tiết kiệm bộ nhớ (sử dụng cách biểu diễn bit như sau: A có giá trị là $2^0 = 1$; B có giá trị là $2^1 = 2$; C có giá trị là $2^2 = 4$ nên AB = $2^0|2^1 = 3$; tương tự ABC = $2^0|2^1|2^2 = 7$). Với cách biểu diễn sau, itemset được chia thành 2 thành phần att × value.

Cung nối từ nút X đến nút Y sao cho X.itemset ⊂ Y.itemset và |X.itemset| = |Y.itemset| − 1. Hình 3.1 [29] mô tả cây tìm kiếm để khai thác luật phân lớp kết hợp. Đầu tiên, mức 1 của cây chứa các nút với 1-itemset, sau đó mỗi nút trên cây sẽ được kết hợp với các nút đứng sau nó có cùng nút cha để tạo ra một tập các nút con của nút hiện hành. Quá trình này được thực hiện một cách đệ qui cho các nút con của nút hiện hành cho đến khi không còn nút nào được tạo ra.

Bảng 3.1: Một cơ sở dữ liệu mẫu huấn luyện

OID	A	B	C	Lớp
1	a_1	b_1	c_1	y
2	a_1	b_2	c_1	n
3	a_2	b_2	c_1	n
4	a_3	b_3	c_1	y
5	a_3	b_1	c_2	n
6	a_3	b_3	c_1	y
7	a_1	b_3	c_2	y
8	a_2	b_2	c_2	n



Hình 3.1: Cây MECR-tree từ tập dữ liệu của bảng 3.1 với thuật toán CAR- Miner [29].

3.4. Thuật toán CAR-Miner

3.4.1. Thuật toán

Đầu vào: Cho bộ dữ liệu D, minSup và minConf.

Đầu ra: Tất cả CARs thỏa minSup and minConf.

Phương pháp thực hiện:

CAR-Miner (L_r, minSup, minConf)

1 CARs = ∅;

2 for all $l_i \in l_r$.children do

3 ENUMERATE-CAR (l_i, minconf)

4 P_i = ∅;

5 For all $l_j \in l_r$.children do, with j>i do

6 If l_i.att # l_j.att then

7 O.att = l_i.att ∪ l_j.att;

8 O.values = l_i.values ∪ l_j.values;

9 O.Obidset = l_i.Obidset ∩ l_j.Obidset;

10 If |O.Obidset| = |l_i.Obidset| then

11 O.count = l_i.count;

12 O.pos = l_i.pos;

13 else if |O.Obidset| = |l_j.Obidset| then

14 O.count = l_j.count;

15 O.pos = l_j.pos;

16 else

17 O.count = {count(x∈ O.Obidset | class (x) = c_j, ∀i∈ [1,k]};

18 O.pos = arg max{l.count$_i$}; $i \in [1,k]$

19 if O.count[O.pos] ≥ minSup then

20 Pi = P_i ∪ O;

21 CAR-Miner (P_i, minSup, minConf)

ENUMERATE-CAR (l, minConf)

22 Conf = l.count [l.pos]/ |l.Obidset|;

23 If conf ≥ minConf then

24 CARs = CARs ∪ {l.itemset → c.pos (l.count[l.pos], conf)}

Xét mỗi nút l_i với tất cả các nút l_j khác trong cây L_r, với j>i (dòng 2 và 5) để sinh ra một nút ứng viên con O. Với mỗi cặp (l_i,l_j) thuật toán kiểm tra l_i.att ≠ l_j.att hay không (dòng 6, sử dụng định lý 2.1). Nếu chúng khác nhau, thuật toán tính ba yếu tố att, values, Obitset cho nút O mới (dòng 7-9). Dòng 10 kiểm tra nếu số dòng chứa l_i là bằng số dòng chứa O (dùng định lý 2.2). Nếu đúng thì dùng định lý 2.2, thuật toán sao chép tất cả thông tin từ nút l_i sang nút O (dòng 11-12). Tương tự như vậy, nếu kết quả ở dòng 10 là sai, thuật toán kiểm tra l_j với O, nếu số dòng chứa chúng là giống nhau (dòng 13), thuật toán sao chép tất cả thông tin từ l_j sang O (dòng 14-15). Ngược lại, thuật toán tính O.count bằng cách dùng O.Obitset và O.pos (dòng 17-18). Sau khi tính tất cả thông tin cho node O, nếu O.count[O.pos] ≥

minSup (dòng 19-20) thuật toán sẽ đưa nút O vào P_i (P_i đã được khởi tạo rỗng ở dòng 4).

Cuối cùng, CAR-Miner sẽ gọi đệ quy với một tập mới P_i (P_i như là tham số đầu vào) (dòng 21).

Thủ tục ENUMERATE-CAR (l, minConf) sinh ra một luật từ nút l. Thủ tục trước tiên tính độ tin cậy của luật (dòng 22), nếu độ tin cậy của luật thỏa mãn minConf (dòng 23) thì nó thêm luật này vào tập CARs (dòng 24).

3.4.2. Ví dụ minh họa

Chúng ta dùng tập dữ liệu trong bảng 3.1 để mô tả tiến trình của thuật toán CAR-Miner với độ hỗ trợ (minSup) = 10% và độ tin cậy (mincof) = 60%. Hình 3.1 thể hiện kết quả của tiến trình này.

Cây MECR được xây dựng từ tập dữ liệu bảng 3.1 như sau:

Đầu tiên, nút gốc L_r chứa các 1-itemset phổ biến gồm

$$\left\{\begin{matrix} 1\times a_1 & 1\times a_2 & 1\times a_3 & 2\times b_1 & 2\times b_2 & 2\times b_3 & 4\times c_1 & 4\times c_2 \\ 127(2,1) & 38(0,2) & 456(2,1) & 15(1,1) & 238(0,3) & 467(3,0) & 12346(3,2) & 578(1,2) \end{matrix}\right\}$$

Sau đó, thủ tục CAR-Miner sẽ được thực thi với tham số L_r. Nút $l_i = \begin{matrix} 1\times a_2 \\ 38(0,2) \end{matrix}$ được dùng như ví dụ minh họa tiến trình CAR-Miner, nút l_i kết hợp với tất cả các nút thể hiện trong L_r:

- Với nút $l_j = \begin{matrix} 1\times a_3 \\ 456(2,1) \end{matrix}$: chúng ta có nút l_i và l_j có cùng thuộc tính là 1 và khác giá trị. Dựa vào mệnh đề 3.1 chúng ta sẽ không tính giá trị kết hợp của 2 nút này.
- Với nút $l_j = \frac{2\times b_1}{15(1,1)}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau O.att = l_i.att $\cup$ l_j.att = 1 | 2 = 3 là các biểu diễn bit; O.values = l_i.values $\cup$ l_j.values = a_2 $\cup$ b_1 = a_2b_1 , và

O.Obidset = l_i. Obidset $\cap$ l_j. Obidset = {3,8} $\cap$ {1,1} = {$\varnothing$}. Bởi vì O.count[O.pos] = 0 < minSup, nên nút O không được thêm vào tập P_i.

- Với nút l_j = $\begin{matrix} 2\times b_2 \\ 238(0,3) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau O.att = l_i.att $\cup$ l_j.att = 1 | 2 = 3 là các biểu diễn bit; O.values = l_i.values $\cup$ l_j.values = a_2 $\cup$ b_2 = a_2b_2 , và O.Obidset = l_i. Obidset $\cap$ l_j. Obidset = {3,8} $\cap$ {2,3,8} = {3,8}. Bởi vì | l_i. Obidset | = | O. Obidset |, thuật toán sẽ lấy toàn bộ nội dung của l_j tới O. Điều này có nghĩa rằng O.count = l_i.count = (0,2) và O.pos = 2. Vì O.count[O.pos] = 2 > minSup, O được thêm vào P_i $\Rightarrow$ P_i = $\begin{Bmatrix} 3\times a_2b_2 \\ 38(0,2) \end{Bmatrix}$.

- Với nút l_j = $\begin{matrix} 2\times b_3 \\ 467(3,0) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau O.att = l_i.att $\cup$ l_j.att = 1 | 2 = 3 là các biểu diễn bit; O.values = l_i.values $\cup$ l_j.values = a_2 $\cup$ b_3 = a_2b_3 , và O.Obidset = l_i. Obidset $\cap$ l_j. Obidset = {3,8} $\cap$ {4,6,7} = {$\varnothing$}. Bởi vì O.count[O.pos] = 0 < minSup, nên nút O không được thêm vào tập P_i.

- Với nút l_j = $\begin{matrix} 4\times c_1 \\ 12346(3,2) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau O.att = l_i.att $\cup$ l_j.att = 1 | 4 = 5 là các biểu diễn bit; O.values = l_i.values $\cup$ l_j.values = a_2 $\cup$ c_1 = a_2c_1, và O.Obidset = l_i. Obidset $\cap$ l_j. Obidset = {3,8} $\cap$ {1,2,3,4,6} = {3}. Thuật toán tính toán bổ sung nội dung gồm O.count = {0,1} và O.pos = 2. Bởi vì O.count[O.pos] = 1 $\geq$ minSup, O được thêm vào P_i $\Rightarrow$ P_i = $\begin{Bmatrix} 3\times a_2b_2 \; 5\times a_2c_1 \\ 38(0,2)\; 3(0,1) \end{Bmatrix}$.

- Với nút $l_j = \begin{matrix} 4\times c_2 \\ 578(1,2) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau O.att = l_i.att ∪ l_j.att = 1 | 4 = 5 là các biểu diễn bit; O.values = l_i.values ∪ l_j.values = a_2 ∪ c_2 = a_2c_2 , và O.Obidset = l_i. Obidset ∩ l_j. Obidset = {3,8} ∩ {5,7,8} = {8}. Thuật toán tính toán bổ sung nội dung gồm O.count = {0,1} và O.pos = 2. Bởi vì O.count[O.pos] = 1 ≥ minSup, O được thêm vào $P_i \Rightarrow P_i = \left\{ \begin{matrix} 3\times a_2b_2 & 5\times a_2c_1 & 5\times a_2c_2 \\ 38(0,2) & 3(0,1) & 8(0,1) \end{matrix} \right\}$.
- Sau khi P_i được tạo, thuật toán CAR-Miner được gọi đệ quy với tham số P_i, minSup, và minCof để tạo tất cả các nút con của P_i. Xem xét tiến trình tạo các nút con của nút $l_i = \begin{matrix} 3\times a_2b_2 \\ 38(0,2) \end{matrix}$:
  - Với nút $l_j = \begin{matrix} 5\times a_2c_1 \\ 3(0,1) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau O.att = l_i.att ∪ l_j.att = 3 | 5 = 7 hoặc 111 trong biểu diễn bit; O.values = l_i.values ∪ l_j.values = a_2b_2 ∪ a_2c_1 = a_2b_2c1, và O.Obidset = l_i. Obidset ∩ l_j. Obidset = {3,8} ∩ {3} = {3}. Thuật toán sẽ chép toàn bộ nội dung của l_j cho O. Điều này có nghĩa rằng O.count = l_i.count = (0,1) và O.pos = 2. Vì O.count[O.pos] = 1 > minSup, O được thêm vào $P_i \Rightarrow P_i = \left\{ \begin{matrix} 7\times a_2b_2c_1 \\ 3(0,1) \end{matrix} \right\}$.
  - Dùng tiến trình tương tự cho nút $l_j = \begin{matrix} 5\times a_2c_2 \\ 8(0,1) \end{matrix}$, chúng ta có được kết quả $P_i = \left\{ \begin{matrix} 7\times a_2b_2c_1 & 7\times a_2b_2c_2 \\ 3(0,1) & 8(0,1) \end{matrix} \right\}$.

Các luật được dễ dàng tạo ra trong các bước duyệt qua nút l_i (dòng 3) bằng cách gọi thủ tục ENUMERATE-CAR(l_i, minConf).

- Với nút l_j = $\begin{matrix}1\times a_2\\38(0,2)\end{matrix}$: thủ tục sẽ tính toán độ tin cậy của nút $\begin{matrix}1\times a_2\\38(0,2)\end{matrix}$, chúng ta có conf = l_i.count[l_i.pos] / |l_i.Obidset| = 2/2 = 1 > 0, Bởi vì conf ≥ minCof (60%), thêm luật $\{(A,a_2)\}\to n(2,1)$ vào tập luật CARs. Ý nghĩa của luật này là "nếu A = a_2 thì lớp là n" (độ hỗ trợ = 2 và độ tin cậy = 100%).
- Với nút l_j = $\begin{matrix}1\times a_3\\456(2,1)\end{matrix}$: thủ tục sẽ tính toán độ tin cậy của nút $\begin{matrix}1\times a_3\\456(2,1)\end{matrix}$, chúng ta có conf = l_i.count[l_i.pos] / |l_i.Obidset| = 2/3 = 0.6 > 0, Bởi vì conf ≥ minCof (60%), thêm luật $\{(A,a_3)\}\to n(2,0.6)$ vào tập luật CARs. Ý nghĩa của luật này là "nếu A = a_3 thì lớp là n" (độ hỗ trợ = 2 và độ tin cậy = 60%).
- Với nút l_j = $\begin{matrix}2\times b_1\\15(1,1)\end{matrix}$: thủ tục sẽ tính toán độ tin cậy của nút $\begin{matrix}2\times b_1\\15(1,1)\end{matrix}$, do số đếm giá trị ở mỗi lớp bằng nhau nên ta sẽ tính độ tin cậy của nút trên từng lớp, chúng ta có conf = l_i.count[1] / |l_i.Obidset| = 1/2 = 0.5 > 0, Bởi vì conf ≥ minCof (60%), thêm luật $\{(B,b_1)\}\to y(1,0.5)$ vào tập luật CARs. Ý nghĩa của luật này là "nếu B = b_1 thì lớp là y" (độ hỗ trợ = 1 và độ tin cậy = 50%). Chúng ta có conf = l_i.count[2] / |l_i.Obidset| = 1/2 = 0.5 > 0, Bởi vì conf ≥ minCof (60%), thêm luật $\{(B,b_1)\}\to n(1,0.5)$ tới tập luật CARs. Ý nghĩa của luật này là "nếu B = b_1 thì lớp là n" (độ hỗ trợ = 1 và độ tin cậy = 50%).
- Với nút l_j = $\begin{matrix}2\times b_2\\238(0,3)\end{matrix}$: thủ tục sẽ tính toán độ tin cậy của nút $\begin{matrix}2\times b_2\\238(0,3)\end{matrix}$, chúng ta có conf = l_i.count[l_i.pos] / |l_i.Obidset| = 3/3 = 1 > 0, Bởi vì conf ≥ minCof (60%), thêm luật $\{(B,b_2)\}\to n(3,1)$ vào tập luật

CARs. Ý nghĩa của luật này là "nếu B = b_2 thì lớp là n" (độ hỗ trợ = 3 và độ tin cậy = 100%).

- Với nút $l_j = \overset{2\times b_3}{467(3,0)}$: thủ tục sẽ tính toán độ tin cậy của nút $\overset{2\times b_3}{467(3,0)}$, chúng ta có conf = l_i.count[l_i.pos] / |l_i.Obidset| = 3/3 = 1 > 0, Bởi vì conf ≥ minCof (60%), thêm luật $\{(B,b_3)\} \to y(3,1)$ vào tập luật CARs. Ý nghĩa của luật này là "nếu B = b_3 thì lớp là n" (độ hỗ trợ = 3 và độ tin cậy = 100%).
- Với nút $l_j = \overset{4\times c_1}{12346(3,2)}$: thủ tục sẽ tính toán độ tin cậy của nút $\overset{4\times c_1}{12346(3,2)}$, chúng ta có conf = l_i.count[l_i.pos] / |l_i.Obidset| = 3/5 = 0.6 > 0, Bởi vì conf ≥ minCof (60%), thêm luật $\{(C,c_1)\} \to y(3,0.6)$ vào tập luật CARs. Ý nghĩa của luật này là "nếu C = c_1 thì lớp là y" (độ hỗ trợ = 3 và độ tin cậy = 60%).
- Với nút $l_j = \overset{4\times c_2}{578(1,2)}$: thủ tục sẽ tính toán độ tin cậy của nút $\overset{4\times c_2}{578(1,2)}$, chúng ta có conf = l_i.count[l_i.pos] / |l_i.Obidset| = 2/3 = 0.6 > 0, Bởi vì conf ≥ minCof (60%), thêm luật $\{(C,c_2)\} \to n(2,0.6)$ vào tập luật CARs. Ý nghĩa của luật này là "nếu C = c_2 thì lớp là n" (độ hỗ trợ = 2 và độ tin cậy = 60%).

Dựa vào mệnh đề 3.2, chúng ta sẽ không cần tính toán nội dung một số nút như {$3\times a_2b_2$, $7\times a_1b_1\ c_1$, $7\times a_1b_2\ c_1$, $7\times a_1b_3\ c_2$, $7\times a_2b_2\ c_1$, $7\times a_2b_2\ c_2$, $7\times a_2b_2\ c_2$, $7\times a_3b_1\ c_2$, $7\times a_3b_3\ c_1$}. Bảng 3.2 thể hiện các luật được tạo ra từ các nút trong hình 3.1.

Bảng 3.2: Tiến trình khai thác CARs của thuật toán CAR- Miner [29].

ID	Node	Luật được tạo	n_x	n_y	n_{xy}	vm
1	$1\times a_1$	Nếu A = a_1 thì lớp = y	3	4	2	0.7
2	$1\times a_2$	Nếu A = a_2 thì lớp = n	2	4	2	1
3	$1\times a_3$	Nếu A = a_3 thì lớp = y	3	4	2	0.7
4	$2\times b_2$	Nếu B = b_2 thì lớp = n	3	4	3	1
5	$2\times b_3$	Nếu B = b_3 thì lớp = y	3	4	3	1
6	$4\times c_1$	Nếu C = c_1 thì lớp = y	5	4	3	0.6
7	$4\times c_2$	Nếu C = c_2 thì lớp = n	3	4	2	0.7
8	$3\times a_1b_1$	Nếu A = a_1 và B = b_1 thì lớp = y	1	4	1	1
9	$3\times a_1b_2$	Nếu A = a_1 và B = b_2 thì lớp = n	1	4	1	1
10	$3\times a_1b_3$	Nếu A = a_1 và B = b_3 thì lớp = y	1	4	1	1
11	$5\times a_1c_1$	Nếu A = a_1 và C = c_1 thì lớp = n	2	4	1	1
12	$5\times a_1c_2$	Nếu A = a_1 và C = c_2 thì lớp = y	1	4	1	1
13	$3\times a_2b_2$	Nếu A = a_2 và B = b_2 thì lớp = n	2	4	2	1
14	$5\times a_2c_1$	Nếu A = a_2 và C = c_1 thì lớp = n	1	4	1	1
15	$5\times a_2c_2$	Nếu A = a_2 và C = c_2 thì lớp = n	1	4	1	1
16	$3\times a_3b_1$	Nếu A = a_3 và B = b_1 thì lớp = n	1	4	1	1
17	$3\times a_3b_3$	Nếu A = a_3 và B = b_3 thì lớp = y	2	4	2	1
18	$5\times a_3c_1$	Nếu A = a_3 và C = c_1 thì lớp = y	2	3	2	1
19	$5\times a_3c_2$	Nếu A = a_3 và C = c_2 thì lớp = n	1	4	1	1
20	$6\times b_1c_1$	Nếu B = b_1 và C = c_1 thì lớp = y	1	4	1	1

21	$6\times b_1c_2$	Nếu B = b_1 và C = c_2 thì lớp = n	1	4	1	1
22	$6\times b_2c_1$	Nếu B = b_2 và C = c_1 thì lớp = n	2	4	2	1
23	$6\times b_2c_2$	Nếu B = b_2 và C = c_2 thì lớp = n	1	1	1	1
24	$6\times b_3c_1$	Nếu B = b_3 và C = c_1 thì lớp = y	2	4	2	1
25	$6\times b_3c_2$	Nếu B = b_3 và C = c2 thì lớp = y	1	4	1	1
26	$7\times a_1b_1c_1$	Nếu A = a_1 và B = b_1 và C = c_1 thì lớp = y	1	4	1	1
27	$7\times a_1b_2c_1$	Nếu A = a_1 và B = b_2 và C = c_1 thì lớp = n	1	4	1	1
28	$7\times a_1b_3c_2$	Nếu A = a_1 và B = b_3 và C = c_2 thì lớp = y	1	4	1	1
29	$7\times a_2b_2c_1$	Nếu A = a_2 và B = b_2 và C = c_1 thì lớp = n	1	4	1	1
30	$7\times a_2b_2c_2$	Nếu A = a_2 và B = b_2 và C = c_2 thì lớp = n	1	4	1	1
31	$7\times a_3b_1c_2$	Nếu A = a_3 và B = b_1 và C = c_2 thì lớp = n	1	4	1	1
32	$7\times a_3b_3c_1$	Nếu A = a_3 và B = b_3 và C = c_1 thì lớp = y	2	4	2	1

3.5. Thuật toán CARIM

3.5.1. Thuật toán

Đầu vào: Cho bộ dữ liệu và một độ đo lợi ích vm.

Đầu ra: Tất cả CARs và độ đo lợi ích.

Phương pháp thực hiện:

CARIM(P, minSup)

```
1   CAR = ∅;
2   for all l_i ∈ P do
3      CAR  =  CAR ∪ ENUMERATE_RULE_IM(l_i);
4      P_i = ∅;
5      for all l_j ∈ P, with j > i do
6         if l_i.atts ≠ l_j.atts then
7            l.atts = l_i.atts ∪ l_j.atts;
8            l.vals = l_i.vals ∪ l_j.vals;
9            l.Obidset = l_i.Obidset ∩ l_j.Obidset;
10           if |l.Obidset| > 0 then
11              for all ob ∈ O.Obidset do
12                  O.count[ob]++;
13           P_i = P_i ∪ l;
14           CARIM(P_i, minSup);

ENUMERATE_RULE_IM(l)
15  CAR_l= ∅;
16  for i∈[1,k] do
17     if l.count[i] > 0 then
18        n_X= |l.Obidset|;
19        n_XY= l.count[i];
20        n_Y= Count[i];
21        CAR_l=CAR_l ∪{l.itemset → c_i(l.count[i], vm(n, n_X, n_Y,
      n_XY)};
22  return The rule with highest information from CAR_l;
```

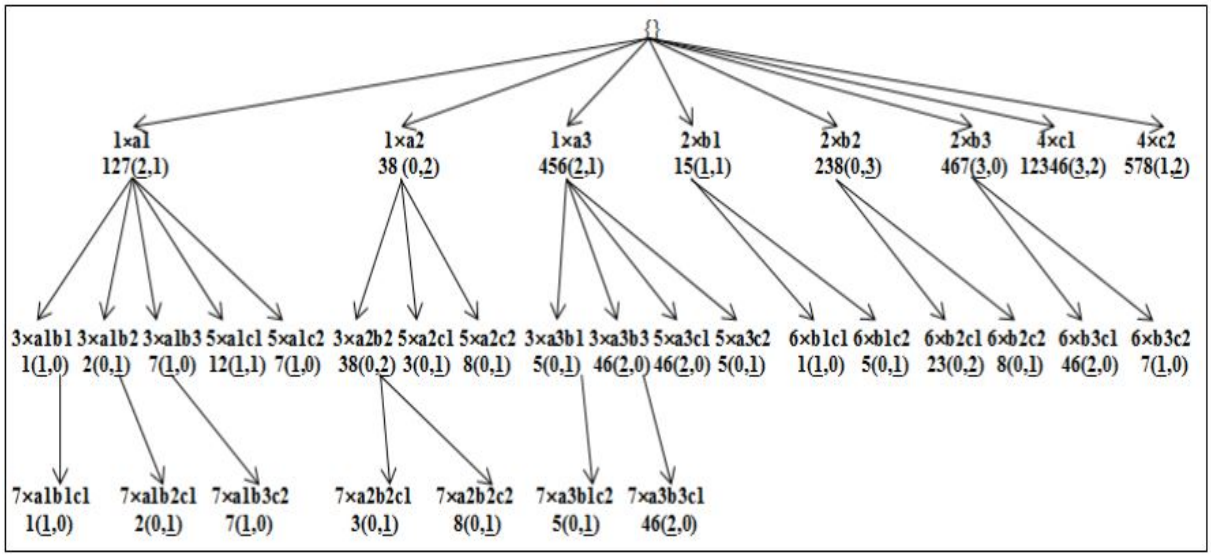
Xét mỗi nút l_i với tất cả các nút l_j khác trong cây L_r, với j>i (dòng 2 và 5) để sinh ra một nút ứng viên con l. Với mỗi cặp (l_i,l_j) thuật toán kiểm tra l_i.att ≠ l_j.att hay không (dòng 6). Nếu chúng khác nhau, thuật toán tính ba yếu tố att, values, Obitset

cho nút O mới (dòng 7-9). Nếu số đối tượng định danh lớn hơn không (dòng 10), thuật toán sẽ đếm đối tượng trong mỗi lớp mà chứa l.itemset và thêm nút này tới P_i (P_i được khởi tạo là rỗng trong dòng 4). Cuối cùng , thủ tục CARIM được gọi đệ quy với một tập mới P_i như một tham số (dòng 14).

Thủ tục ENUMERATE_RULE_IM(l) sinh ra một luật từ nút l. Trước tiên sẽ duyệt qua các lớp (dòng 16) để tạo ra luật. Nếu đếm của lớp này lớn hơn không (dòng 17), có nghĩa rằng nút l có thể tạo ra một luật từ l.itemset $\rightarrow c_i$. Thủ tục sau đó sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và n_{XY} (dòng 18-20), trong đó X là l.itemset và Y là lớp c_i. Để lấy độ hỗ trợ của X, các yếu tố trong Obibset sẽ được đếm. Độ hỗ trợ của Y (l.count[i]) và n (số các đối tượng) có thể thu được khi duyệt dữ liệu. Sau khi bốn yếu tố thu được, giá trị của bất kỳ độ được áp dụng có thể dễ dàng tính toán (dòng 21). Cuối cùng, thủ tục trả về luật với độ đo cao nhất từ tập luật CAR_l (dòng 22).

3.5.2. Ví dụ minh họa

Ví dụ trong bảng 3.1 sẽ được dùng để mô tả tiến trình của thuật toán CARIM với độ đo Jaccard (bảng 2.3). Hình 3.2 [32] thể hiện cấu trúc cây MECR từ dữ liệu trong bảng 3.1, trong đó số trước ký tự 'x' là biểu diện dưới dạng bit của các thuộc tính.



Hình 3.2: Cây MECR-tree từ tập dữ liệu của bảng 3.1 với thuật toán CARIM .

Cây MECR được xây dựng từ tập dữ liệu bảng 3.1 như sau

Đầu tiên, nút gốc L_r chứa các 1-itemset phổ biến gồm

$$\left\{\begin{matrix} 1\times a_1 & 1\times a_2 & 1\times a_3 & 2\times b_1 & 2\times b_2 & 2\times b_3 & 4\times c_1 & 4\times c_2 \\ 127(2,1) & 38(0,2) & 456(2,1) & 15(1,1) & 238(0,3) & 467(3,0) & 12346(3,2) & 578(1,2) \end{matrix}\right\}$$

Sau đó, thủ tục CARIM sẽ được thực thi với tham số L_r. Chúng ta dùng nút $l_i = \begin{matrix} 1\times a_2 \\ 38(0,2) \end{matrix}$ như ví dụ minh họa tiến trình CARIM, nút l_i kết hợp với tất cả các nút thể hiện trong L_r:

- Với nút $l_j = \begin{matrix} 1\times a_3 \\ 456(2,1) \end{matrix}$: chúng ta có nút l_i và l_j có cùng thuộc tính là 1 và khác giá trị. Dựa vào mệnh đề 3.1 chúng ta sẽ không tính giá trị kết hợp của 2 nút này.
- Với nút $l_j = \begin{matrix} 2\times b_1 \\ 15(1,1) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau l.atts = l_i.atts $\cup$ l_j.atts = 1 | 2 = 3 là các biểu diễn bit; l.vals = l_i.vals $\cup$ l_j.vals = a_2 $\cup$ b_1 = a_2b_1 , và l.Obidset = l_i. Obidset $\cap$ l_j. Obidset = {3,8} $\cap$ {1,1} = {$\varnothing$}. Bởi vì | l. Obidset | = 0, nên nút l không được thêm vào tập P_i.
- Với nút $l_j = \begin{matrix} 2\times b_2 \\ 238(0,3) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau l.atts = l_i.atts $\cup$ l_j.atts = 1 | 2 = 3 là các biểu diễn bit; l.vals = l_i.vals $\cup$ l_j.vals = a_2 $\cup$ b_2 = a_2b_2 , và l.Obidset = l_i. Obidset $\cap$ l_j. Obidset = {3,8} $\cap$ {2,3,8} = {3,8}. Bởi vì | l. Obidset | > 0, thuật toán sẽ tính toán giá trị O.count. Điều này có nghĩa rằng O.count = (0,2). Nút l được thêm vào $P_i \Rightarrow P_i = \left\{\begin{matrix} 3\times a_2b_2 \\ 38(0,2) \end{matrix}\right\}$

- Với nút $l_j = \begin{matrix} 2\times b_3 \\ 467(3,0) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau l.atts = l_i.atts ∪ l_j.atts = 1 | 2 = 3 là các biểu diễn bit; l.vals = l_i.vals ∪ l_j.vals = a_2 ∪ b_3 = a_2b_3 , và l.Obidset = l_i. Obidset ∩ l_j. Obidset = {3,8} ∩ {4,6,7} = {∅}. Bởi vì | l. Obidset | = 0, nên nút l không được thêm vào tập P_i.
- Với nút $l_j = \begin{matrix} 4\times c_1 \\ 12346(3,2) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau l.atts = l_i.atts ∪ l_j.atts = 1 | 4 = 5 là các biểu diễn bit; l.vals = l_i.vals ∪ l_j.vals = a_2 ∪ c_1 = a_2c_1 , và l.Obidset = l_i. Obidset ∩ l_j. Obidset = {3,8} ∩ {1,2,3,4,6} = {3}. Bởi vì | l. Obidset | > 0, thuật toán sẽ tính toán giá trị O.count. Điều này có nghĩa rằng O.count = (0,1). Nút l được thêm vào P_i ⇒ P_i = $\begin{Bmatrix} 3\times a_2b_2 & 5\times a_2c_1 \\ 38(0,2) & 3(0,1) \end{Bmatrix}$.
- Với nút $l_j = \begin{matrix} 4\times c_2 \\ 578(1,2) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau l.atts = l_i.atts ∪ l_j.atts = 1 | 4 = 5 là các biểu diễn bit; l.vals = l_i.vals ∪ l_j.vals = a_2 ∪ c_2 = a_2c_2 , và l.Obidset = l_i. Obidset ∩ l_j. Obidset = {3,8} ∩ {5,7,8} = {8}. Bởi vì | l. Obidset | > 0, thuật toán sẽ tính toán giá trị O.count. Điều này có nghĩa rằng O.count = (0,1). Nút l được thêm vào P_i ⇒ P_i = $\begin{Bmatrix} 3\times a_2b_2 & 5\times a_2c_1 & 5\times a_2c_2 \\ 38(0,2) & 3(0,1) & 8(0,1) \end{Bmatrix}$.
- Sau khi P_i được tạo, thuật toán CARIM được gọi đệ quy với tham số P_i, minSup, để tạo tất cả các nút con của P_i. Xem xét tiến trình tạo các nút con của nút $l_i = \begin{matrix} 3\times a_2b_2 \\ 38(0,2) \end{matrix}$:

- Với nút $l_j = \begin{matrix} 5\times a_2c_1 \\ 3(0,1) \end{matrix}$: vì nút l_i và l_j khác thuộc tính, chúng ta sẽ tính toán giá trị cho 3 yếu tố sau l.atts = l_i.atts ∪ l_j.atts = 3 | 5 = 7 hoặc 111 trong biểu diễn bit; l.vals = l_i.vals ∪ l_j.vals = a_2b_2 ∪ a_2c_1 = a_2b_2c1, và l.Obidset = l_i. Obidset ∩ l_j. Obidset = {3,8} ∩ {3} = {3}. Bởi vì | l. Obidset | > 0, thuật toán sẽ tính toán giá trị O.count. Điều này có nghĩa rằng O.count = (0,1). Nút l được thêm vào $P_i \Rightarrow P_i = \left\{ \begin{matrix} 7\times a_2b_2c_1 \\ 3(0,1) \end{matrix} \right\}$.
- Dùng tiến trình tương tự cho nút $l_j = \begin{matrix} 5\times a_2c_2 \\ 8(0,1) \end{matrix}$, chúng ta có được kết quả $P_i = \left\{ \begin{matrix} 7\times a_2b_2c_1 & 7\times a_2b_2c_2 \\ 3(0,1) & 8(0,1) \end{matrix} \right\}$.

Các luật được dễ dàng tạo ra trong các bước duyệt qua nút l_i (dòng 3) bằng cách gọi thủ tục ENUMERATE_RULE_IM(l_i).

- Với nút $l_j = \begin{matrix} 1\times a_2 \\ 38(0,2) \end{matrix}$: chúng ta có l.count[2] = 2 > 0, Thủ tục sau đó sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và $n_{XY,}$ với n_X = |l.Obidset| = 2, n_{XY} = l.count[2] = 2, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X + n_Y - n_{XY}}$ = $\frac{2}{2+4-2} = \frac{1}{2}$. Thêm luật $\{(A, a_2)\} \rightarrow n(2, \frac{1}{2})$ vào tập luật CARs.

- Với nút $l_j = \begin{matrix} 1\times a_3 \\ 456(2,1) \end{matrix}$: chúng ta có l.count[1] = 2 > 0, thủ tục sau đó sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và $n_{XY,}$ với n_X = |l.Obidset| = 3, n_{XY} = l.count[1] = 2, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X + n_Y - n_{XY}}$ =

$\frac{2}{3+4-2}=\frac{2}{5}$. Thêm luật $\{(A,a_3)\}\to y(2,\frac{2}{5})$ vào tập luật CARs. Chúng ta có l.count[2] = 1 > 0, thủ tục sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và n_{XY}, với n_X = |l.Obidset| = 3, n_{XY} = l.count[2] = 1, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X+n_Y-n_{XY}}$ = $\frac{1}{3+4-1}=\frac{1}{6}$. Thêm luật $\{(A,a_3)\}\to n(1,\frac{1}{6})$ vào tập luật CARs.

- Với nút l_j = $\frac{2\times b_1}{15(1,1)}$: chúng ta có l.count[1] = 1 > 0, thủ tục sau đó sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và $n_{XY,}$ với n_X = |l.Obidset| = 2, n_{XY} = l.count[1] = 1, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X+n_Y-n_{XY}}$ = $\frac{1}{2+4-1}=\frac{1}{5}$. Thêm luật $\{(B,b_1)\}\to y(1,\frac{1}{5})$ vào tập luật CARs. Chúng ta có l.count[2] = 1 > 0, thủ tục sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và n_{XY}, với n_X = |l.Obidset| = 2, n_{XY} = l.count[2] = 1, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X+n_Y-n_{XY}}$ = $\frac{1}{2+4-1}=\frac{1}{5}$. Thêm luật $\{(B,b_1)\}\to n(1,\frac{1}{5})$ vào tập luật CARs.

- Với nút l_j = $\frac{2\times b_2}{238(0,3)}$: chúng ta có l.count[2] = 3 > 0, Thủ tục sau đó sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và $n_{XY,}$ với n_X = |l.Obidset| = 3, n_{XY} = l.count[2] = 3, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X+n_Y-n_{XY}}$ = $\frac{3}{3+4-3}=\frac{3}{4}$. Thêm luật $\{(B,b_2)\}\to n(3,\frac{3}{4})$ vào tập luật CARs.

- Với nút l_j = $\frac{2\times b_3}{467(3,0)}$: chúng ta có l.count[1] = 3 > 0, Thủ tục sau đó sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và $n_{XY,}$ với n_X = |l.Obidset| = 3, n_{XY} = l.count[1] = 3, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X + n_Y - n_{XY}}$ = $\frac{3}{3+4-3} = \frac{3}{4}$. Thêm luật $\{(B, b_3)\} \to y(3, \frac{3}{4})$ vào tập luật CARs.

- Với nút l_j = $\frac{4\times c_1}{12346(3,2)}$: chúng ta có l.count[1] = 3 > 0, thủ tục sau đó sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và $n_{XY,}$ với n_X = |l.Obidset| = 5, n_{XY} = l.count[1] = 3, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X + n_Y - n_{XY}}$ = $\frac{3}{5+4-3} = \frac{1}{2}$. Thêm luật $\{(C, c_1)\} \to y(3, \frac{1}{2})$ vào tập luật CARs. Chúng ta có l.count[2] = 2 > 0, thủ tục sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và n_{XY}, với n_X = |l.Obidset| = 5, n_{XY} = l.count[2] = 2, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X + n_Y - n_{XY}}$ = $\frac{2}{5+4-2} = \frac{2}{7}$. Thêm luật $\{(C, c_1)\} \to n(2, \frac{2}{7})$ vào tập luật CARs.

- Với nút l_j = $\frac{4\times c_2}{578(1,2)}$: chúng ta có l.count[1] = 1 > 0, thủ tục sau đó sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và $n_{XY,}$ với n_X = |l.Obidset| = 3, n_{XY} = l.count[1] = 1, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X + n_Y - n_{XY}}$ = $\frac{1}{3+4-1} = \frac{1}{6}$. Thêm luật $\{(C, c_2)\} \to y(1, \frac{1}{6})$ vào tập luật CARs. Chúng ta có l.count[2] = 2 > 0, thủ tục sẽ tính toán giá trị tham số của luật này, bao gồm n_X, n_Y và n_{XY}, với n_X = |l.Obidset| = 3, n_{XY} = l.count[2]

= 2, n_Y = 4 $\Rightarrow$ vm = $\frac{n_{XY}}{n_X + n_Y - n_{XY}}$ = $\frac{2}{3+4-2} = \frac{2}{5}$. Thêm luật $\{(C, c_2)\} \rightarrow n(2, \frac{2}{5})$ vào tập luật CARs.

Bảng 3.3 thể hiện các luật được tạo ra từ các nút trong hình 3.2. Các quy tắc trong in đậm là mạnh nhất trong số những luật được tạo ra từ các nút trong cây tương ứng.

Bảng 3.3: Tiến trình khai thác CARs của thuật toán CARIM với độ đo Jaccard.

ID	**Node**	**Luật được tạo**	$\mathbf{n_X}$	$\mathbf{n_Y}$	$\mathbf{n_{XY}}$	**Độ đo Jaccard**
1	$1 \times a_1$	**Nếu A = a_1 thì lớp = y**	3	4	2	2/5
		Nếu A = a_1 thì lớp = n	3	4	1	1/6
2	$1 \times a_2$	Nếu A = a_2 thì lớp = n	2	4	2	2/4
3	$1 \times a_3$	**Nếu A = a_3 thì lớp = y**	3	4	2	2/5
		Nếu A = a_3 thì lớp = n	3	4	1	1/6
4	$2 \times b_1$	Nếu A = b_1 thì lớp = y	2	4	1	1/4
		Nếu A = b_1 thì lớp = n	2	4	1	1/4
5	$2 \times b_2$	Nếu B = b_2 thì lớp = n	3	4	3	3/4
6	$2 \times b_3$	Nếu B = b_3 thì lớp = y	3	4	3	3/4
7	$4 \times c_1$	**Nếu C = c_1 thì lớp = y**	5	4	3	3/6
		Nếu C = c_1 thì lớp = n	5	4	2	2/7
8	$4 \times c_2$	Nếu C = c_2 thì lớp = y	3	4	1	1/6
		Nếu C = c_2 thì lớp = n	3	4	2	2/5

9	$3\times a_1b_1$	Nếu A = a_1 và B = b_1 thì lớp = y	1	4	1	1/4
10	$3\times a_1b_2$	Nếu A = a_1 và B = b_2 thì lớp = n	1	4	1	1/4
11	$3\times a_1b_3$	Nếu A = a_1 và B = b_3 thì lớp = y	1	4	1	1/4
12	$5\times a_1c_1$	Nếu A = a_1 và C = c_1 thì lớp = y	2	4	1	1/4
		Nếu A = a_1 và C = c_1 thì lớp = n	2	4	1	1/4
13	$5\times a_1c_2$	Nếu A = a_1 và C = c_2 thì lớp = y	1	4	1	1/4
14	$3\times a_2b_2$	Nếu A = a_2 và B = b_2 thì lớp = n	2	4	2	2/4
15	$5\times a_2c_1$	Nếu A = a_2 và C = c_1 thì lớp = n	1	4	1	1/4
16	$5\times a_2c_2$	Nếu A = a_2 và C = c_2 thì lớp = n	1	4	1	1/4
17	$3\times a_3b_1$	Nếu A = a_3 và B = b_1 thì lớp = n	1	4	1	1/4
18	$3\times a_3b_3$	Nếu A = a_3 và B = b_3 thì lớp = y	2	4	2	2/4
19	$5\times a_3c_1$	Nếu A = a_3 và C = c_1 thì lớp = y	2	3	2	2/4
20	$5\times a_3c_2$	Nếu A = a_3 và C = c_2 thì lớp = n	1	4	1	1/4*
21	$6\times b_1c_1$	Nếu B = b_1 và C = c_1 thì lớp = y	1	4	1	1/4*
22	$6\times b_1c_2$	Nếu B = b_1 và C = c_2 thì lớp = n	1	4	1	1/4*
23	$6\times b_2c_1$	Nếu B = b_2 và C = c_1 thì lớp = n	2	4	2	2/4*
24	$6\times b_2c_2$	Nếu B = b_2 và C = c_2 thì lớp = n	1	1	1	1/4*
25	$6\times b_3c_1$	Nếu B = b_3 và C = c_1 thì lớp = y	2	4	2	2/4*
26	$6\times b_3c_2$	Nếu B = b_3 và C = c2 thì lớp = y	1	4	1	1/4*
27	$7\times a_1b_1c_1$	Nếu A = a_1 và B = b_1 và C = c_1 thì lớp = y	1	4	1	1/4*
28	$7\times a_1b_2c_1$	Nếu A = a_1 và B = b_2 và C = c_1 thì lớp = n	1	4	1	1/4*

29	$7 \times a_1 b_3 c_2$	Nếu A = a_1 và B = b_3 và C = c_2 thì lớp = y	1	4	1	1/4*
30	$7 \times a_2 b_2 c_1$	Nếu A = a_2 và B = b_2 và C = c_1 thì lớp = n	1	4	1	1/4*
31	$7 \times a_2 b_2 c_2$	Nếu A = a_2 và B = b_2 và C = c_2 thì lớp = n	1	4	1	1/4*
32	$7 \times a_3 b_1 c_2$	Nếu A = a_3 và B = b_1 và C = c_2 thì lớp = n	1	4	1	1/4*
33	$7 \times a_3 b_3 c_1$	Nếu A = a_3 và B = b_3 và C = c_1 thì lớp = y	2	4	2	2/4*

CHƯƠNG 4: KHẢO SÁT ẢNH HƯỞNG CỦA CÁC ĐỘ ĐO LỢI ÍCH LÊN ĐỘ CHÍNH XÁC

4.1 . k-fold cross-validation

Cross-validation (hay còn gọi là kiểm tra chéo) là một kỹ thuật mô hình xác nhận để đánh giá như thế nào về các kết quả của một phân tích thống kê và sẽ khái quát đến một bộ dữ liệu độc lập. Nó chủ yếu được sử dụng trong cài đặt nơi mà mục tiêu là dự đoán, để ước tính như thế nào về độ chính xác một mô hình dự đoán sẽ thực hiện trong thực tế. Trong vấn đề dự đoán, một mô hình thường được đưa ra một bộ dữ liệu của dữ liệu được biết trên đó huấn luyện được chạy (bộ dữ liệu huấn luyện), và một bộ dữ liệu của dữ liệu không rõ (hoặc dữ liệu đầu tiên nhìn thấy) dựa vào đó các mô hình được kiểm tra (dữ liệu kiểm tra). Mục tiêu của cross-validation là để xác định một tập dữ liệu để "thử nghiệm" các mô hình trong giai đoạn đào tạo (tức là, xác nhận bộ dữ liệu), cung cấp cho một cái nhìn sâu sắc về các mô hình này sẽ được phổ biến cho một tập dữ liệu độc lập (tức là, một bộ dữ liệu không rõ ràng, cho các trường hợp từ một vấn đề thực tế).

Một vòng của **cross-validation** liên quan đến việc phân vùng một mẫu dữ liệu thành các tập con bổ sung, thực hiện các phân tích trên một tập con (gọi là tập huấn luyện), và xác nhận các phân tích trên các tập con khác (gọi là tập xác nhận hoặc kiểm định). Để giảm độ đa dạng, nhiều vòng qua xác nhận được thực hiện bằng cách sử dụng phân vùng khác nhau, và các kết quả xác nhận được tính trung bình qua các vòng.

Đánh giá độ chính xác của bộ phân lớp rất quan trọng, bởi vì nó cho phép dự đoán được độ chính xác của các kết quả phân lớp những dữ liệu tương lai. Độ chính xác còn giúp so sánh các mô hình phân lớp khác nhau. Một số phương pháp đánh giá phổ biến bao gồm: Holdout (Splitting), k-fold cross validation, Leave-one-out cross validation.

Trong **k-fold cross-validation**, các mẫu ban đầu được phân chia ngẫu nhiên thành k fold kích thước bằng nhau. Trong số k fold, một fold duy nhất được giữ lại như là dữ liệu xác nhận để thử nghiệm các mô hình, và k - 1 fold còn lại được sử dụng như dữ liệu huấn luyện. Sau đó quá trình k-fold cross-validation được lặp đi lặp lại k lần, với mỗi k fold được sử dụng đúng một lần như các dữ liệu xác nhận. Các kết quả từ những fold sau đó có thể được lấy trung bình (hoặc kết hợp) để cung cấp một dự toán duy nhất. Ưu điểm của phương pháp lặp đi lặp lại ngẫu nhiên trên các mẫu con là tất cả các quan sát này được sử dụng cho cả dữ liệu đào tạo và xác nhận, và mỗi quan sát được sử dụng để xác nhận đúng một lần. 10-fold cross-validation thường được sử dụng, nhưng nói chung k-fold vẫn là một tham số không cố định. Trong phân tầng k-fold cross-validation, các fold được lựa chọn sao cho giá trị trung bình hồi đáp xấp xỉ bằng tất cả những fold. Trong trường hợp của một phân loại nhị phân, điều này có nghĩa là mỗi fold có chứa xấp xỉ tỷ lệ giống nhau của hai loại nhãn lớp.

k-fold cross validation là một kỹ thuật chung để tính hiệu suất của một phân lớp. Cho một tập m mẫu huấn luyện, chạy duy nhất k-fold cross validation thu được như sau:

- Đặt các mẫu huấn luyện theo một thứ tự ngẫu nhiên.
- Chia mẫu huấn luyện thành k fold.
- Cho i chạy từ 1 tới k
    - Huấn luyện phân lớp dùng cho tất cả các mẫu mà không chứa fold i.
    - Kiểm tra phân lớp trên tất cả các mẫu trong fold i.
    - Tính n_i , số mẫu trong fold i mà được phân lớp đúng.
- Trả về độ chính xác phân lớp đúng như sau: $E = \frac{\sum_{i=1}^{k} n_i}{m}$.

4.2. Độ chính xác

Chúng ta dùng kỹ thuật kiểm tra chéo (k-fold cross-validation) để tính độ chính xác và phân lớp đúng của các mẫu, tiến trình được tiến hành như sau:

- Dùng kỹ thuật kiểm tra chéo (k-fold cross-validation), phân chia thành các fold và ngẫu nhiên các mẫu ban đầu thành k fold.
- Cho i chạy từ 1 tới k, với mỗi bước i
  - Một fold duy nhất được giữ lại như là dữ liệu xác nhận để thử nghiệm, và k - 1 fold còn lại được sử dụng như dữ liệu huấn luyện.
  - Với tập dữ liệu huấn luyện k – 1, ta sẽ dùng thuật toán CARMIN và áp dụng các độ đo lợi ích (bảng 2.3) để tạo ra tập luật phân lớp.
  - Ta dùng tập luật được tạo ra từ dữ liệu huấn luyện k – 1 để kiểm tra mẫu thử nghiệm có được phân lớp đúng. Nếu mẫu thử nghiệm được phân lớp đúng, ta tính $n_i = n_i + 1$
- Cuối cùng ta có n_i số mẫu phân lớp đúng, độ chính xác được tính như sau: $E = \frac{\sum_{i=1}^{k} n_i}{m}$.

Ví dụ: Ta có một tập dữ liệu mẫu huấn luyện (bảng 3.1), với 8 mẫu huấn luyện và được phân theo từng lớp. Với kỹ thuật k-fold cross-validation, các mẫu huấn luyện sẽ được chia thành 8 fold.

Cho i chạy từ 1 tới 8:

- Với i=1, ta lấy fold 1 làm mẫu thử nghiệm, 7 fold còn lại làm tập mẫu huấn luyện (bảng 4.1).
- Từ tập mẫu huấn luyện này ta sẽ dùng thuật toán CARIM và áp dụng các độ đo lợi ích (bảng 2.3) để tạo ra tập luật phân lớp.

- Ta dùng tập luật được tạo ra để kiểm tra mẫu thử nghiệm có được phân lớp đúng. Nếu mẫu thử nghiệm được phân lớp đúng, ta tính $n_i = n_i + 1$.

Bảng 4.1 mô tả tiến trình k-fold cross-validation với i = 1, lấy fold 1 là mẫu thử nghiệm và lấy các fold 2,3,4,5,6,7,8 làm tập mẫu huấn luyện.

Bảng 4.1: Mô tả tiến trình k-fold cross-validation với i = 1.

OID	A	B	C	Class
1	a_1	b_1	c_1	y
2	a_1	b_2	c_1	n
3	a_2	b_2	c_1	n
4	a_3	b_3	c_1	y
5	a_3	b_1	c_2	n
6	a_3	b_3	c_1	y
7	a_1	b_3	c_2	y
8	a_2	b_2	c_2	n

→ Mẫu thử nghiệm

Tập mẫu huấn luyện

Bảng 4.2 mô tả tiến trình k-fold cross-validation với i = 2, lấy fold 2 là mẫu thử nghiệm và lấy các fold 1,3,4,5,6,7,8 làm tập mẫu huấn luyện.

Bảng 4.2: Mô tả tiến trình k-fold cross-validation với i = 2.

OID	A	B	C	Class
1	a_1	b_1	c_1	y
2	a_1	b_2	c_1	n
3	a_2	b_2	c_1	n
4	a_3	b_3	c_1	y
5	a_3	b_1	c_2	n
6	a_3	b_3	c_1	y
7	a_1	b_3	c_2	y
8	a_2	b_2	c_2	n

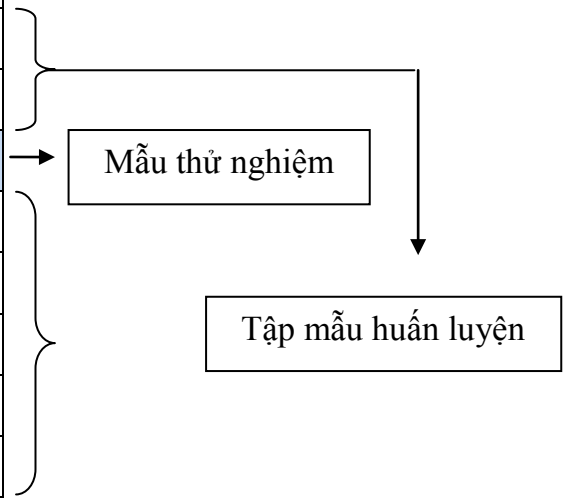
→ Mẫu thử nghiệm

Tập mẫu huấn luyện

Bảng 4.3 mô tả tiến trình k-fold cross-validation với i = 3, lấy fold 3 là mẫu thử nghiệm và lấy các fold 1,2,4,5,6,7,8 làm tập mẫu huấn luyện.

Bảng 4.3: Mô tả tiến trình k-fold cross-validation với i = 3.

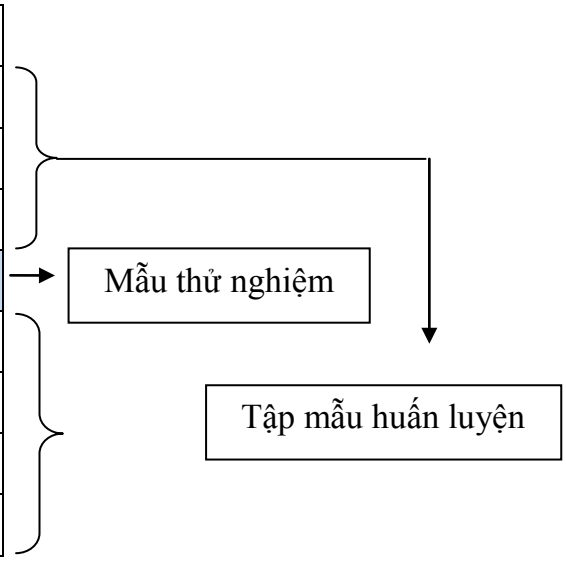
OID	A	B	C	Class
1	a_1	b_1	c_1	y
2	a_1	b_2	c_1	n
3	a_2	b_2	c_1	n
4	a_3	b_3	c_1	y
5	a_3	b_1	c_2	n
6	a_3	b_3	c_1	y
7	a_1	b_3	c_2	y
8	a_2	b_2	c_2	n



Bảng 4.4 mô tả tiến trình k-fold cross-validation với i = 4, lấy fold 4 là mẫu thử nghiệm và lấy các fold 1,2,3,5,6,7,8 làm tập mẫu huấn luyện.

Bảng 4.4: Mô tả tiến trình k-fold cross-validation với i = 4.

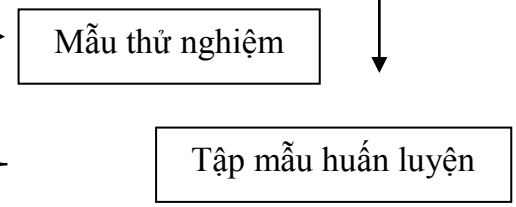
OID	A	B	C	Class
1	a_1	b_1	c_1	y
2	a_1	b_2	c_1	n
3	a_2	b_2	c_1	n
4	a_3	b_3	c_1	y
5	a_3	b_1	c_2	n
6	a_3	b_3	c_1	y
7	a_1	b_3	c_2	y
8	a_2	b_2	c_2	n



Bảng 4.5 mô tả tiến trình k-fold cross-validation với i = 5, lấy fold 5 là mẫu thử nghiệm và lấy các fold 1,2,3,4,6,7,8 làm tập mẫu huấn luyện.

Bảng 4.5: Mô tả tiến trình k-fold cross-validation với i = 5.

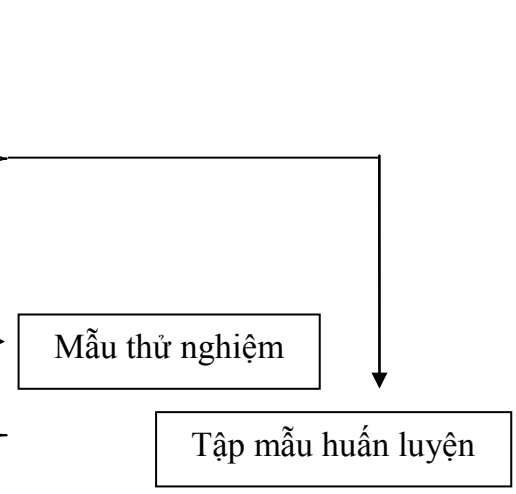
OID	A	B	C	Class
1	a_1	b_1	c_1	y
2	a_1	b_2	c_1	n
3	a_2	b_2	c_1	n
4	a_3	b_3	c_1	y
5	a_3	b_1	c_2	n
6	a_3	b_3	c_1	y
7	a_1	b_3	c_2	y
8	a_2	b_2	c_2	n



Bảng 4.6 mô tả tiến trình k-fold-cross-validation với i = 6, lấy fold 6 là mẫu thử nghiệm và lấy các fold 1,2,3,4,5,7,8 làm tập mẫu huấn luyện.

Bảng 4.6: Mô tả tiến trình k-fold cross-validation với i = 6.

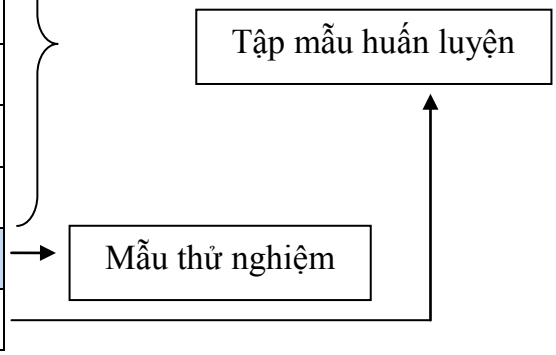
OID	A	B	C	Class
1	a_1	b_1	c_1	y
2	a_1	b_2	c_1	n
3	a_2	b_2	c_1	n
4	a_3	b_3	c_1	y
5	a_3	b_1	c_2	n
6	a_3	b_3	c_1	y
7	a_1	b_3	c_2	y
8	a_2	b_2	c_2	n



Bảng 4.7 mô tả tiến trình k-fold cross-validation với i = 7, lấy fold 7 là mẫu thử nghiệm và lấy các fold 1,2,3,4,5,6,8 làm tập mẫu huấn luyện.

Bảng 4.7: Mô tả tiến trình k-fold cross-validation với i = 7.

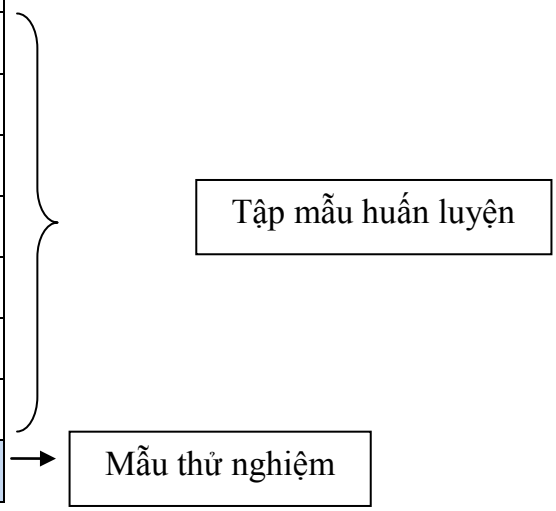
OID	A	B	C	Class
1	a_1	b_1	c_1	y
2	a_1	b_2	c_1	n
3	a_2	b_2	c_1	n
4	a_3	b_3	c_1	y
5	a_3	b_1	c_2	n
6	a_3	b_3	c_1	y
7	a_1	b_3	c_2	y
8	a_2	b_2	c_2	n



Bảng 4.8 mô tả tiến trình k-fold cross-validation với i = 8, lấy fold 8 là mẫu thử nghiệm và lấy các fold 1,2,3,4,5,6,7 làm tập mẫu huấn luyện.

Bảng 4.8: Mô tả tiến trình k-fold cross-validation với i = 8.

OID	A	B	C	Class
1	a_1	b_1	c_1	y
2	a_1	b_2	c_1	n
3	a_2	b_2	c_1	n
4	a_3	b_3	c_1	y
5	a_3	b_1	c_2	n
6	a_3	b_3	c_1	y
7	a_1	b_3	c_2	y
8	a_2	b_2	c_2	n



Khi duyệt hết các mẫu huyến luyện ta sẽ có n_i số mẫu phân lớp đúng trong 8 mẫu huấn luyện. Ta tính được độ chính xác như sau: $E = \frac{n_i}{8}$.

4.3. Kết quả thực nghiệm

Các thuật toán được sử dụng trong các thử nghiệm đã được chạy trên máy tính cá nhân có cài các phần mềm Microsoft Visual Studio 2010, Windows 7 (64 bit), với cấu hình máy Core i5 x 3.20 GHz, 16 GB bộ nhớ RAM.

Các kết quả thực nghiệm được thực nghiệm trong các tập dữ liệu thu được từ UCI Repository tại địa chỉ (http: // mlearn.ics.uci.edu). Bảng 4.9 cho thấy các đặc tính của các bộ dữ liệu thực nghiệm, các bộ dữ liệu thử nghiệm có các đặc tính khác nhau. Ví dụ bộ dữ liệu breast-cancer có ít thuộc tính khác nhau nhưng có nhiều mẫu, lymph chứa nhiều thuộc tính nhưng lại có ít mẫu.

Bảng 4.9. Đặc tính của tập dữ liệu thực nghiệm

Tập dữ liệu	Số thuộc tính	Số lớp	Số mẫu
breast-cancer	9	2	286
lymph	18	4	148
Vehicle	19	4	846

Kết quả thực nghiệm trên 3 tập dữ liệu từ bảng 4.9, dùng thuật toán CARIM với minSup = 1 và áp dụng các độ đo lợi ích để tìm các tập luật phân lớp, sử dụng kỹ thuật k-fold cross-validation để tính độ chính xác của tập dữ liệu. Kết quả thực nghiệm như sau:

- **Độ đo Confidence:**

Bảng 4.10 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *breast-cancer* với độ đo confidence. Với độ đo confidence có miền giá trị từ 0 tới 1, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-conf =0.3 thì số mẫu phân lớp đúng là 271, độ chính xác là 94%.
- Với min-conf =0.4 thì số mẫu phân lớp đúng là 230, độ chình xác là 80%.
- Với min-conf =0.5 thì số mẫu phân lớp đúng là 186, độ chính xác là 65%.
- Với min-conf =0.6 thì số mẫu phân lớp đúng là 42, độ chính xác là 15%.

Bảng 4.10. Kết quả thực nghiệm trên tập dữ liệu breast-cancer với độ đo Confidence

Breast	Min-conf = 0.3	Min-conf = 0.4	Min-conf = 0.5	Min-conf = 0.6
Phân lớp đúng	271	230	186	42
Độ chính xác (%)	94%	80%	65%	15%

Bảng 4.11 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *lymph* với độ đo confidence. Với độ do confidence có miền giá trị từ 0 tới 1, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-conf =0.3 thì số mẫu phân lớp đúng là 140, độ chình xác là 94%.
- Với min-conf =0.4 thì số mẫu phân lớp đúng là 136, độ chình xác là 92%.
- Với min-conf =0.5 thì số mẫu phân lớp đúng là 122, độ chính xác là 82%.
- Với min-conf =0.6 thì số mẫu phân lớp đúng là 32, độ chính xác là 22%.

Bảng 4.11. Kết quả thực nghiệm trên tập dữ liệu lymph với độ đo Confidence

Lymph	Min-conf = 0.3	Min-conf = 0.4	Min-conf = 0.5	Min-conf = 0.6
Phân lớp đúng	140	136	122	32
Độ chính xác (%)	94%	92%	82%	22%

Bảng 4.12 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *Vehicle* với độ đo confidence. Với độ do confidence có miền giá trị từ 0 tới 1, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-conf =0.3 thì số mẫu phân lớp đúng là 743, độ chình xác là 88%.
- Với min-conf =0.4 thì số mẫu phân lớp đúng là 255, độ chình xác là 32%.
- Với min-conf =0.5 thì số mẫu phân lớp đúng là 161, độ chính xác là 19%.
- Với min-conf =0.6 thì số mẫu phân lớp đúng là 21, độ chính xác là 2%.

Bảng 4.12. Kết quả thực nghiệm trên tập dữ liệu Vehicle với độ đo Confidence

Vehicle	Min-conf = 0.3	Min-conf = 0.4	Min-conf = 0.5	Min-conf = 0.6
Phân lớp đúng	743	255	161	21
Độ chính xác (%)	88%	32%	19%	2%

Bảng 4.13. So sánh độ chính xác trên các tập dữ liệu với độ đo Confidence

DATASET		**Breast**	**Lymph**	**Vehicle**
Confidence	***0.3***	94%	94%	88%
	0.4	80%	92%	32%
	0.5	65%	82%	19%
	0.6	15%	22%	2%

Kết quả so sánh từ bảng 4.13, ta thấy độ chính xác càng giảm khi các ngưỡng tối thiểu càng lớn, với 2 tập dữ liệu nhỏ (breast, lymph) thì có độ chính xác cao, trong khi đó với tập dữ liệu lớn hơn lại có độ chính xác thấp hơn.

- **Độ đo Cosine:**

Bảng 4.14 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *breast-cancer* với độ đo cosine. Với độ do cosine có miền giá trị từ 0 tới 1, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-cosine =0.3 thì số mẫu phân lớp đúng là 246, độ chính xác là 86%.
- Với min-cosine =0.4 thì số mẫu phân lớp đúng là 178, độ chính xác là 62%.
- Với min-cosine =0.5 thì số mẫu phân lớp đúng là 160, độ chính xác là 56%.
- Với min-cosine =0.6 thì số mẫu phân lớp đúng là 160, độ chính xác là 56%.

Bảng 4.14. Kết quả thực nghiệm trên tập dữ liệu breast-cancer với độ đo Cosine.

Breast	Min-cosine = 0.3	Min-cosine = 0.4	Min-cosine = 0.5	Min-cosine = 0.6
Phân lớp đúng	246	178	160	160
Độ chính xác (%)	86%	62%	56%	56%

Bảng 4.15 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *lymph* với độ đo cosine. Với độ đo cosine có miền giá trị từ 0 tới 1, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-cosine =0.3 thì số mẫu phân lớp đúng là 145, độ chính xác là 98%.
- Với min-cosine =0.4 thì số mẫu phân lớp đúng là 144, độ chính xác là 97%.
- Với min-cosine =0.5 thì số mẫu phân lớp đúng là 78, độ chính xác là 53%.
- Với min-cosine =0.6 thì số mẫu phân lớp đúng là 71, độ chính xác là 48%.

Bảng 4.15. Kết quả thực nghiệm trên tập dữ liệu lymph với độ đo Cosine.

Lymph	Min-cosine = 0.3	Min-cosine = 0.4	Min-cosine = 0.5	Min-cosine = 0.6
Phân lớp đúng	145	144	78	71
Độ chính xác (%)	98%	97%	53%	48%

Bảng 4.16 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *Vehicle* với độ đo confidence. Với độ do confidence có miền giá trị từ 0 tới 1, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-conf =0.3 thì số mẫu phân lớp đúng là 1, độ chình xác là 1%.
- Với min-conf =0.4 thì số mẫu phân lớp đúng là 0, độ chình xác là 0%.
- Với min-conf =0.5 thì số mẫu phân lớp đúng là 0, độ chính xác là 0%.
- Với min-conf =0.6 thì số mẫu phân lớp đúng là 0, độ chính xác là 0%.

Bảng 4.16. Kết quả thực nghiệm trên tập dữ liệu Vehicle với độ đo Cosine

Vehicle	Min-cosine = 0.3	Min-cosine = 0.4	Min-cosine = 0.5	Min-cosine = 0.6
Phân lớp đúng	1	0	0	0
Độ chính xác (%)	1%	0%	0%	0%

Bảng 4.17. So sánh độ chính xác trên các tập dữ liệu với độ đo Cosine

DATASET		**Breast**	**Lymph**	**Vehicle**
Cosine	***0.3***	86%	98%	1%
	0.4	62%	97%	0%
	0.5	56%	53%	0%
	0.6	56%	48%	0%

Kết quả so sánh từ bảng 4.17, ta thấy độ chính xác càng giảm khi các ngưỡng tối thiểu càng lớn, với 2 tập dữ liệu nhỏ (breast, lymph) thì có độ chính xác cao, trong khi đó với tập dữ liệu lớn hơn lại có độ chính xác 1% với ngưỡng tối thiểu 0.3 và độ chính xác 0% với các ngưỡng tối thiểu khác.

- **Độ đo Lift:**

Bảng 4.18 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *breast-cancer* với độ đo lift. Với độ do lift có miền giá trị từ 0 tới ∞, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-lift =0.7 thì số mẫu phân lớp đúng là 219, độ chính xác là 77%.
- Với min-lift =0.8 thì số mẫu phân lớp đúng là 109, độ chính xác là 38%.
- Với min-lift =0.9 thì số mẫu phân lớp đúng là 99, độ chính xác là 34%.
- Với min-lift =1 thì số mẫu phân lớp đúng là 90, độ chính xác là 31%.

Bảng 4.18. Kết quả thực nghiệm trên tập dữ liệu breast-cancer với độ đo Lift.

Breast	Min-lift = 0.7	Min-lift = 0.8	Min-lift = 0.9	Min-lift = 1
Phân lớp đúng	219	109	99	90
Độ chính xác (%)	77%	38%	34%	31%

Bảng 4.19 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *lymph* với độ đo lift. Với độ đo lift có miền giá trị từ 0 tới ∞, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-lift =0.7 thì số mẫu phân lớp đúng là 136, độ chính xác là 92%.
- Với min-lift =0.8 thì số mẫu phân lớp đúng là 123, độ chính xác là 83%.
- Với min-lift =0.9 thì số mẫu phân lớp đúng là 112, độ chính xác là 75%.
- Với min-lift =1 thì số mẫu phân lớp đúng là 56, độ chính xác là 37%.

Bảng 4.19. Kết quả thực nghiệm trên tập dữ liệu lymph với độ đo Lift.

Lymph	Min-lift = 0.7	Min-lift = 0.8	Min-lift = 0.9	Min-lift = 1
Phân lớp đúng	136	123	112	56
Độ chính xác (%)	92%	83%	75%	37%

Bảng 4.20 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *Vehicle* với độ đo lift. Với độ đo lift có miền giá trị từ 0 tới ∞, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-lift =0.7 thì số mẫu phân lớp đúng là 791, độ chính xác là 93%.
- Với min-lift =0.8 thì số mẫu phân lớp đúng là 780, độ chính xác là 92%.
- Với min-lift =0.9 thì số mẫu phân lớp đúng là 764, độ chính xác là 90%.
- Với min-lift =1 thì số mẫu phân lớp đúng là 760, độ chính xác là 89%.

Bảng 4.20. Kết quả thực nghiệm trên tập dữ liệu Vehicle với độ đo Lift

Vehicle	Min-lift = 0.7	Min-lift = 0.8	Min-lift = 0.9	Min-lift = 1
Phân lớp đúng	791	780	764	760
Độ chính xác (%)	93%	92%	90%	89%

Bảng 4.21. So sánh độ chính xác trên các tập dữ liệu với độ đo Lift

DATASET		**Breast**	**Lymph**	**Vehicle**
Lift	***0.7***	77%	92%	93%
	0.8	38%	83%	92%
	0.9	34%	75%	90%
	1	31%	37%	89%

Kết quả so sánh từ bảng 4.21, ta thấy độ chính xác giảm khi các ngưỡng tối thiểu càng lớn, với tập dữ liệu lớn (vehicle) có độ chính xác cao hơn so với 2 tập dữ liệu (breast, lymph) và tỷ lệ giảm độ chính xác cũng thấp hơn.

- **Độ đo Rule interest:**

Bảng 4.22 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *breast-cancer* với độ đo rule interest. Với độ do rule interest có miền giá trị từ 0 tới ∞, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-interest =0.7 thì số mẫu phân lớp đúng là 44, độ chính xác là 15%.
- Với min-interest =0.8 thì số mẫu phân lớp đúng là 40, độ chính xác là 13%.
- Với min-interest =0.9 thì số mẫu phân lớp đúng là 32, độ chính xác là 11%.
- Với min-interest =1 thì số mẫu phân lớp đúng là 32, độ chính xác là 11%.

Bảng 4.22. Kết quả thực nghiệm trên tập dữ liệu breast-cancer với độ đo Rule interest.

Breast	Min-interest = 0.7	Min-interest = 0.8	Min-interest = 0.9	Min-interest = 1
Phân lớp đúng	44	40	32	32
Độ chính xác (%)	15%	13%	11%	11%

Bảng 4.23 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *lymph* với độ đo rule interest. Với độ do rule interest có miền giá trị từ 0 tới ∞, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-interest =0.7 thì số mẫu phân lớp đúng là 53, độ chính xác là 35%.
- Với min-interest =0.8 thì số mẫu phân lớp đúng là 53, độ chính xác là 35%.
- Với min-interest =0.9 thì số mẫu phân lớp đúng là 47, độ chính xác là 31%.
- Với min-interest =1 thì số mẫu phân lớp đúng là 45, độ chính xác là 30%.

Bảng 4.23. Kết quả thực nghiệm trên tập dữ liệu lymph với độ đo Rule interest.

Lymph	Min-interest = 0.7	Min-interest = 0.8	Min-interest = 0.9	Min-interest = 1
Phân lớp đúng	53	53	47	45
Độ chính xác (%)	35%	35%	31%	30%

Bảng 4.24 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *Vehicle* với độ đo rule interest. Với độ do rule interest có miền giá trị từ 0 tới ∞, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-interest =0.7 thì số mẫu phân lớp đúng là 426, độ chình xác là 50%.
- Với min-interest =0.8 thì số mẫu phân lớp đúng là 368, độ chình xác là 43%.
- Với min-interest =0.9 thì số mẫu phân lớp đúng là 353, độ chính xác là 41%.
- Với min-interest =1 thì số mẫu phân lớp đúng là 323, độ chính xác là 38%.

Bảng 4.24. Kết quả thực nghiệm trên tập dữ liệu Vehicle với độ đo Rule interest

Vehicle	Min-interest = 0.7	Min-interest = 0.8	Min-interest = 0.9	Min-interest = 1
Phân lớp đúng	426	368	353	323
Độ chính xác (%)	50%	43%	41%	38%

Bảng 4.25. So sánh độ chính xác trên các tập dữ liệu với độ đo Rule interest

DATASET		**Breast**	**Lymph**	**Vehicle**
Interest	***0.7***	15%	35%	50%
	0.8	13%	35%	43%
	0.9	11%	31%	41%
	1	11%	30%	38%

Kết quả so sánh từ bảng 4.25, ta thấy độ chính xác giảm khi các ngưỡng tối thiểu càng lớn nhưng tỷ lệ giảm độ chính xác không nhiều chỉ chênh lệch từ 1% đến 2%, với tập dữ liệu lớn (vehicle) có độ chính xác cao hơn so với 2 tập dữ liệu (breast, lymph).

- **Độ đo Laplace:**

Bảng 4.26 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *breast-cancer* với độ đo laplace. Với độ do laplace có miền giá trị từ 0 tới 1, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-laplace =0.3 thì số mẫu phân lớp đúng là 276, độ chình xác là 96%.
- Với min-laplace =0.4 thì số mẫu phân lớp đúng là 262, độ chình xác là 92%.
- Với min-laplace =0.5 thì số mẫu phân lớp đúng là 186, độ chính xác là 65%.
- Với min-laplace =0.6 thì số mẫu phân lớp đúng là 35, độ chính xác là 12%.

Bảng 4.26. Kết quả thực nghiệm trên tập dữ liệu breast-cancer với độ đo Laplace.

Breast	Min-laplace = 0.3	Min-laplace = 0.4	Min-laplace = 0.5	Min-laplace = 0.6
Phân lớp đúng	276	262	186	35
Độ chính xác (%)	96%	92%	65%	12%

Bảng 4.27 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *lymph* với độ đo laplace. Với độ do laplace có miền giá trị từ 0 tới 1, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-laplace =0.3 thì số mẫu phân lớp đúng là 142, độ chình xác là 95%.
- Với min-laplace =0.4 thì số mẫu phân lớp đúng là 137, độ chình xác là 93%.
- Với min-laplace =0.5 thì số mẫu phân lớp đúng là 122, độ chính xác là 82%.
- Với min-laplace =0.6 thì số mẫu phân lớp đúng là 30, độ chính xác là 20%.

Bảng 4.27. Kết quả thực nghiệm trên tập dữ liệu lymph với độ đo Laplace.

Lymph	Min-laplace = 0.3	Min-laplace = 0.4	Min-laplace = 0.5	Min-laplace = 0.6
Phân lớp đúng	142	137	122	30
Độ chính xác (%)	95%	93%	82%	20%

Bảng 4.28 thể hiện kết quả thực nghiệm tính độ chính xác trên tập dữ liệu *Vehicle* với độ đo laplace. Với độ do laplace có miền giá trị từ 0 tới 1, nên ta chọn một số ngưỡng tối thiểu nằm trong miền giá trị để thực nghiệm:

- Với min-laplace =0.3 thì số mẫu phân lớp đúng là 756, độ chình xác là 89%.
- Với min-laplace =0.4 thì số mẫu phân lớp đúng là 717, độ chình xác là 85%.
- Với min-laplace =0.5 thì số mẫu phân lớp đúng là 161, độ chính xác là 19%.
- Với min-laplace =0.6 thì số mẫu phân lớp đúng là 21, độ chính xác là 2%.

Bảng 4.28. Kết quả thực nghiệm trên tập dữ liệu Vehicle với độ đo Laplace.

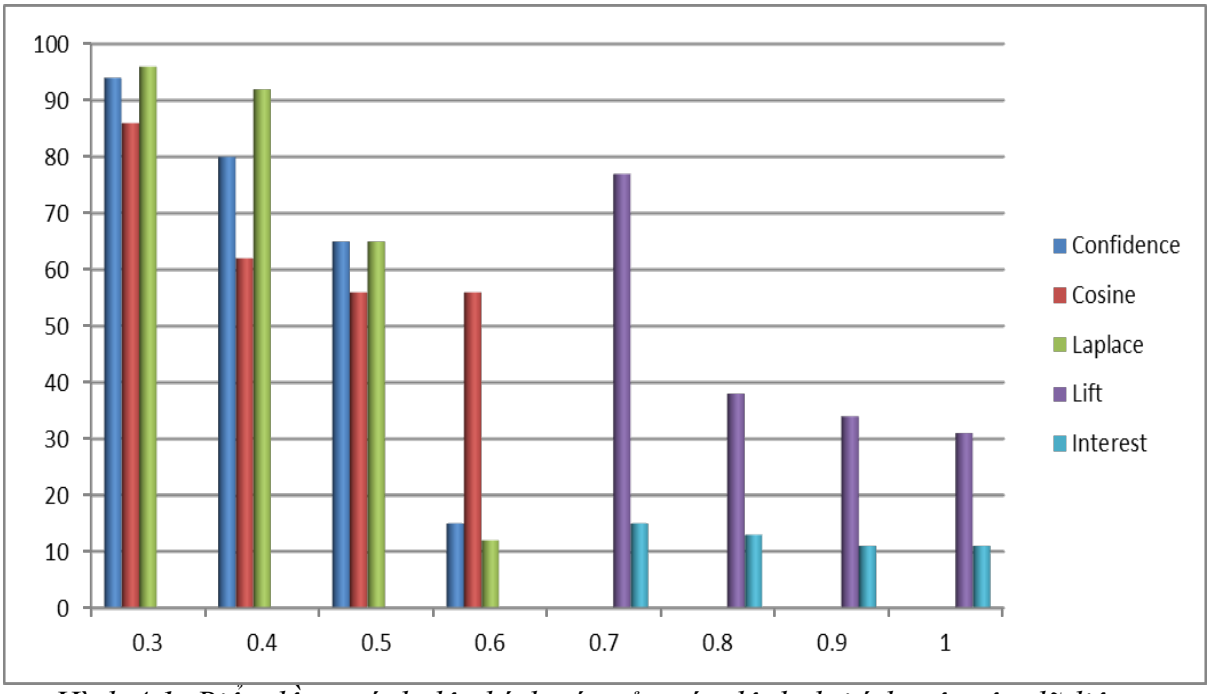
Vehicle	Min-laplace = 0.3	Min-laplace = 0.4	Min-laplace = 0.5	Min-laplace = 0.6
Phân lớp đúng	756	717	161	21
Độ chính xác (%)	89%	85%	19%	2%

Bảng 4.29. So sánh độ chính xác trên các tập dữ liệu với độ đo Laplace

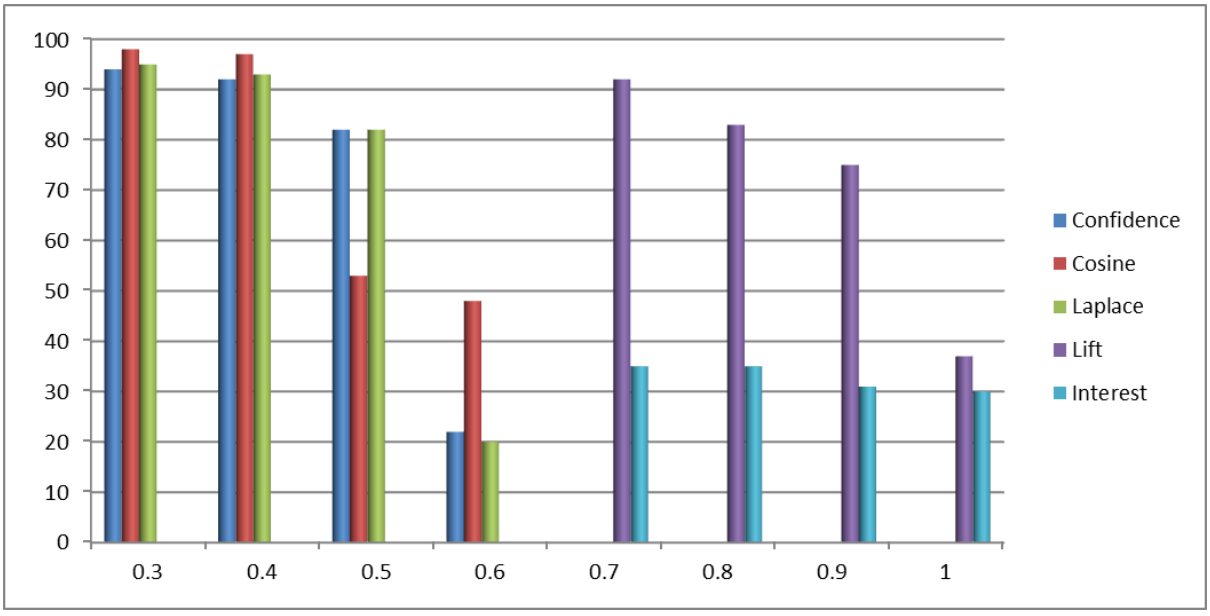
DATASET		**Breast**	**Lymph**	**Vehicle**
Laplace	***0.3***	96%	95%	89%
	0.4	92%	93%	85%
	0.5	65%	82%	19%
	0.6	12%	20%	2%

Kết quả so sánh từ bảng 4.29, ta thấy độ chính xác càng giảm khi các ngưỡng tối thiểu càng lớn, với 2 tập dữ liệu nhỏ (breast, lymph) thì có độ chính xác cao, trong khi đó với tập dữ liệu lớn hơn lại có độ chính xác thấp hơn.

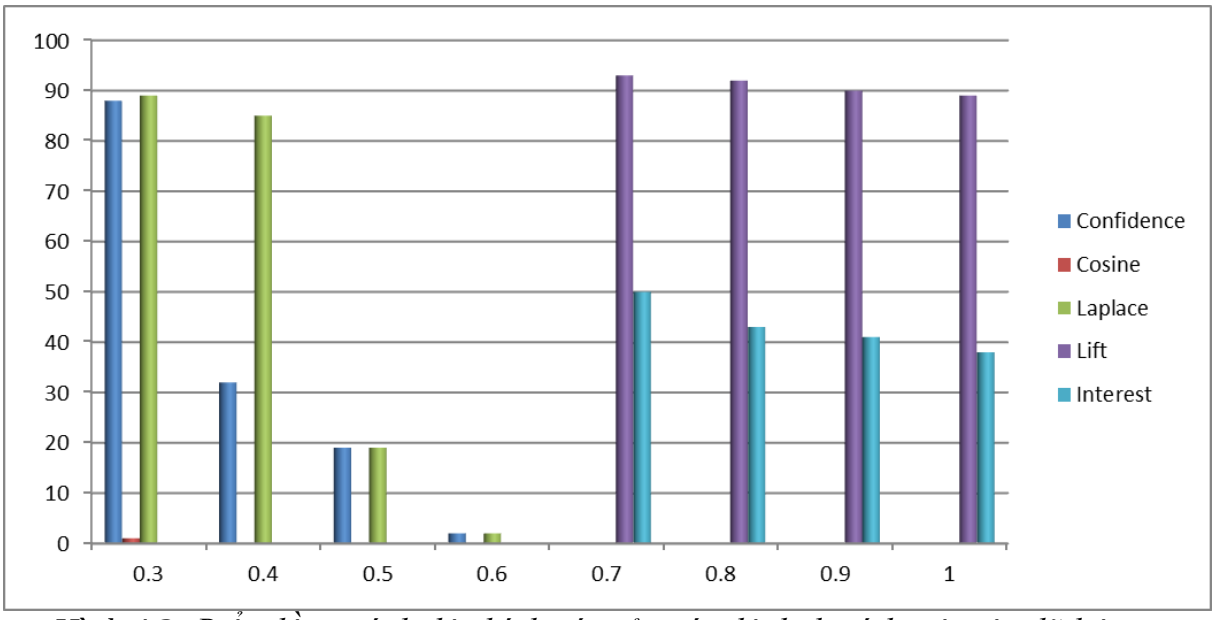
Từ những kết quả thực nghiệm trên, với từng độ đo ta sẽ lấy ngưỡng tối thiểu có độ chính xác để so sánh với các độ đo khác.



Hình 4.1: Biểu đồ so sánh độ chính xác của các độ đo lợi ích trên tập dữ liệu breast-cancer



Hình 4.2: Biểu đồ so sánh độ chính xác của các độ đo lợi ích trên tập dữ liệu Lymph



Hình 4.3: Biểu đồ so sánh độ chính xác của các độ đo lợi ích trên tập dữ liệu Vehicle

Hình 4.1 là biểu đồ so sánh độ chính xác theo từng độ đo trên tập dữ liệu *breast-cancer*, hình 4.2 là biểu đồ so sánh độ chính xác theo từng độ đo trên tập dữ liệu *lymph* và hình 4.3 là biểu đồ so sánh độ chính xác theo từng độ đo trên tập dữ liệu *Vehicle*.

- ✓ Với tập dữ liệu nhỏ (breast, lymph) ứng với các độ đo thì có độ chính xác tương đối cao và tỷ lệ giảm độ chính xác không nhiều, riêng với độ đo interest lại có độ chính xác thấp hơn nhiều so với các độ đo khác.
- ✓ Với tập dữ liệu lớn hơn (vehicle) ta thấy độ đo confidence, laplace, lift có độ chính xác cao, nhưng độ đo confidence, laplace thì tỷ lệ giảm độ chính xác chênh lệch nhiều ở ngưỡng tối thiểu từ 0.4 đến 0.5, trong khi đó độ đo lift, interest có tỷ lệ giảm độ chính xác không nhiều.
- ✓ Từ kết quả so sánh trên ta thấy độ đo lift luôn có độ chính xác cao trên cả tập dữ liệu nhỏ và lớn, tỷ lệ giảm độ chính xác cũng không nhiều. Vì thế độ đo lift được xem là độ đo tốt nhất trong các độ đo được khảo sát.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết luận

Luận văn đã khảo sát ảnh hưởng của các độ đo lợi ích lên độ chính xác trong bài toán phân lớp dựa trên luất kết hợp. Thực tế, các khái niệm này đã được sử dụng riêng rẻ trong các công trình trước đây nhưng lại chưa có khảo sát liên quan đến ảnh hưởng của các độ đo lợi ích lên độ chính xác trong bài toán phân lớp dựa trên luất kết hợp. Một số đóng góp cụ thể như sau:

- Nghiên cứu thuật toán CAR-Miner.
- Nghiên cứu thuật toán CARIM.
- Tìm hiểu về các độ đo lợi ích.
- Tìm hiểu về kỹ thuật kiểm tra chéo (k-fold cross-validation).
- Nghiên cứu cách thức áp dụng các độ đo lợi ích để khai thác CARs.
- Thực nghiệm khảo sát các độ đo lợi ích lên độ chính xác trong khai thác CARs.

5.2. Nhận xét

❖ **Ưu điểm:**

- Luận văn đã trình bày chi tiết cách tính độ chính xác trên các tập dữ liệu với các độ đo lợi ích khác nhau.
- Khảo sát cho thấy sự thay đổi độ chính xác trong việc áp dụng các độ đo lợi ích khác nhau để khai thác CARs. Từ đó có thể chọn ra độ đo thích hợp để khai thác CARs.
- Có thể áp dụng trên các CSDL gốc với số dòng dữ liệu lớn.

❖ **Hạn chế:**

- Luận văn đã trình bày chi tiết cách tính độ chính xác trên các tập dữ liệu với các độ đo lợi ích khác nhau nhưng lại chưa quan tâm đến thời gian thực thi. Do đó với các CSDL lớn sẽ tốn nhiều thời gian để tính độ chính xác.

5.3. Hướng phát triển

Nghiên cứu cải tiến thời gian khai thác CARs và tính độ chính xác trên các CSDL lớn với các độ đo lợi ích khác nhau.

Dựa vào kết quả của luận văn để tìm ra độ đo tốt nhất, có thể làm giảm đáng kể số lượng các luật cho gần như tất cả các bộ dữ liệu, trong khi độ chính xác hầu như không giảm hoặc thậm chí cải thiện hơn. Chọn lựa độ đo lợi ích cho từng CSDL.

TÀI LIỆU THAM KHẢO

[1] Ross Quinlan (1986): "Induction of Decision Trees", Machine Learning 1(1), (pp. 81-106).

[2] Gregory Piatetsky-Shapiro (1991): "Discovery, analysis, and presentation of strong rules", Knowledge Discovery in Databases, (pp. 229–248).

[3] Ross Quinlan (1992): "C4.5: programs for machine learning", Machine Learning 16, (pp. 235-240).

[4] Rakesh Agrawal, Ramakrishnan Srikant (1994): "Fast algorithms for mining association rules", in VLDB’94, (pp. 487–499).

[5] Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, Shalom Tsur (1997): "Dynamic itemset counting and implication rules for market basket analysis", in Proceedings of the 1997 ACM-SIGMOD International conference on management of Data (SIGMOD’97), (pp. 255–264).

[6] Bing Liu, Wynne Hsu, Yiming Ma (1998): "Integrating classification and association rule mining", in 4th International conference on knowledge discovery and Data mining, (pp. 80–86).

[7] Mehmet R. Tolun, Saleh M. Abu-Soud (1998): "ILA: an inductive learning algorithm for rule extraction", Expert Systems With Applications 14(3), (pp. 361–370).

[8] Mehmet R. Tolun, Hayri Sever, Mahmut Uludağ, Saleh M. Abu-Soud (1999): "ILA-2 an inductive learning algorithm for knowledge discovery", Cybernetics and Systems 30(7), (pp . 609–628).

[9] Giovanni Giurida, Wesley W. Chu, Dominique M. Hanssens (2000): "Mining classification rules from datasets with large number of many-valued attributes", in 7th International conference on extending database technology: advances in database technology (EDBT’00), (pp. 335–349).

[10] Wenmin Li, Jiawei Han, Jian Pei (2001): "CMAR: Accurate and efficient classification based on multiple class-association rules ", in 1st IEEE international conference on Data mining, (pp. 369–376).

[11] Pang-Ning Tan, Vipin Kumar, Jaideep Srivastava (2002): "Selecting the right interestingness measure for association patterns", in proceeding of the ACM SIGKDD international conference on knowledge discovery in databases (KDD’02), (pp. 32–41).

[12] Xiaoxin Yin, Jiawei Han (2003): "CPAR: Classification based on predictive association rules", in SIAM international conference on Data mining (SDM’03), (pp. 331–335).

[13] Young-Koo Lee, Won-Young Kim, Y.Dora Cai, Jiawei Han (2003): "CoMine: Efficient mining of correlated patterns", in proceeding of ICDM’03, (pp. 581–584).

[14] Edward R. Omiecinski (2003): "Alternative Interest Measures for Mining Associations in Databases", IEEE Transactions on Knowledge and Data Engineering, (pp. 57–69).

[15] Fadi A. Thabtah, Peter Cowling, Yonghong Peng (2004): "MMAC: A new multi-class, multi-label associative classification approach", the 4th IEEE International Conference on Data mining, (pp. 217-224).

[16] B. Shekar, Rajesh Natarajan (2004): "A transaction-based neighborhood-driven approach to quantifying interestingness of association rules", in proceedings of ICDM’04, (pp. 194-201).

[17] Fadi Thabtah, Peter Cowling, Yonghong Peng (2005): "MCAR: Multi-class classification based on association rule", in 3rd ACS/IEEE international conference on computer systems and applications, (pp. 33–39).

[18] Risi Thonangi, Vikram Pudi (2005): "ACME: An associative classifier based on maximum entropy principle", in 16th International conference algorithmic learning theory, (pp. 122–134).

[19] Adriano Veloso, Wagner Meira Jr, Mohammed J. Zaki (2006): "Lazy associative classification", in 2006 IEEE international conference on Data mining (ICDM'06), (pp. 645–654).

[20] Xuan-Hiep Huynh, Fabrice Guillet, Julien Blanchard, Pascale Kuntz, Henri Briand, Régis Gras (2007): "A graphbased clustering approach to evaluate interestingness measures: A tool and a comparative study", Quality Measures in Data mining. Springer-Verlag, (pp. 25–50).

[21] Bay Vo, Bac Le (2008): "A novel classification algorithm based on association rule mining", PKAW 2008, (pp. 61-75).

[22] Waleed A. Aljandal, William H. Hsu, Vikas Bahirwani, Doina Caragea, Tim Weninger (2008): "Validation-based normalization and selection of interestingness measures for association rules", in proceedings of the 18th international conference on artificial neural networks in engineering (ANNIE 2008), (pp. 1–8).

[23] Philippe Lenca, Patrick Meyer, Benoît Vaillant, Stéphane Lallich (2008): "On selecting interestingness measures for association rules: User oriented description and multiple criteria decision aid", European Journal of Operational Research, (pp. 610–626).

[24] Ya-Wen Chang Chien, Yen-Liang Chen (2010): "Mining associative classification rules with stock trading data – A GA-based method", Knowledge-Based Systems 23(6), (pp. 605–614).

[25] Mehmet Kaya (2010): "Autonomous classifiers with understandable rule using multiobjective genetic algorithms", Expert Systems With Applications 37(4), (pp. 3489–3494).

[26] Hamid Reza Qodmanan, Mahdi Nasiri, Behrouz Minaei-Bidgoli (2011): "Multi objective association rule mining with genetic algorithm without specifying minimum support and minimum confidence ", Expert Systems With Applications 38(1), (pp. 288–298).

[27] Bay Vo, Bac Le (2011): "Interestingness measures for association rules: Combination between lattice and hash tables", Expert Systems With Applications 38(9), (pp. 11630-11640).

[28] Guangfei Yang, Shingo Mabu, Kaoru Shimada, Kotaro Hirasawa (2011): "An evolutionary approach to rank class association rules with feedback mechanism", Expert Systems With Applications 38(12), (pp. 15040–15048).

[29] Loan Nguyen, Bay Vo, Tzung-Pei Hong, Hoang Chi Thanh (2013): "CAR-Miner: An efficient algorithm for mining class-association rules", Expert Systems With Applications 40(6), (pp. 2305-2311).

[30] Robert J. Hilderman, Howard J. Hamilton (2013): "Knowledge discovery and measures of interest", department of Computer Science.

[31] Dang Nguyen, Bay Vo, Bac Le (2014): "Efficient strategies for parallel mining class association rules", Expert Systems with Applications 41(10), (pp. 4716-4729).

[32] Loan Nguyen, Bay Vo, Tzung-Pei Hong (2015): "CARIM: An Efficient Algorithm for Mining Class-association Rules with Interestingness Measures", The international Arab Journal of Information Technology, 12(6A), (pp. 627-634).