|  |  |
| --- | --- |
|  |  |
| TRẦN HỮU MINH | **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**  **---------------------------------------** |
|  |
|  |
|  |
|  |
| **Trần Hữu Minh** |
|  |
|  |
|  |
|  |
|  |
| KỸ THUẬT MÁY TÍNH |  |
|  |
| **KỸ THUẬT PHÂN LOẠI DỮ LIỆU SỬ DỤNG THUẬT TOÁN CMAR TRONG DATA MINING** |
|  |
|  |
|  |
|  |
|  |
|  |
|  |  |
|  | LUẬN VĂN THẠC SĨ KỸ THUẬT  KỸ THUẬT MÁY TÍNH |
|  |  |
|  |  |
| 2015B |  |
|  |
|  |
|  |
| Hà Nội – Năm 2018 |
|  | |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**  **---------------------------------------** | |
|  | |
| **Trần Hữu Minh** | |
|  | |
|  | |
|  | |
|  | |
|  | |
| **KỸ THUẬT PHÂN LOẠI DỮ LIỆU SỬ DỤNG THUẬT TOÁN CMAR TRONG DATA MINING** | |
|  | |
|  | |
|  | |
|  | |
| Chuyên ngành : Kỹ thuật máy tính | |
|  | |
|  | |
|  | |
|  | |
|  | |
| LUẬN VĂN THẠC SĨ KỸ THUẬT  KỸ THUẬT MÁY TÍNH. | |
|  | |
|  | |
|  | |
|  | |
|  | |
| NGƯỜI HƯỚNG DẪN KHOA HỌC : | |
| 1. PGS.TS. Phạm Văn Hải | |
|  | |
|  | |
|  | |
|  | |
|  | |
| Hà Nội – Năm 2018 | |

# LỜI CAM ĐOAN

Những kiến thức trình bày trong luận văn là do tôi tìm hiểu, nghiên cứu và trình bày theo những kiến thức tổng hợp của cá nhân. Kết quả nghiên cứu trong luận văn này chưa từng được công bố tại bất kỳ công trình nào khác. Trong quá trình làm luận văn, tôi có tham khảo các tài liệu có liên quan và đã ghi rõ nguồn tài liệu tham khảo. Tôi xin cam đoan đây là công trình nghiên cứu của tôi và không sao chép của bất kỳ ai.

Tôi xin chịu hoàn toàn trách nhiệm, nếu sai, tôi xin chịu mọi hình thức kỷ luật theo quy định.

Hà Nội, ngày 24 tháng 9 năm 2018

Học viên

Trần Hữu Minh

# LỜI CẢM ƠN

            Trước tiên, tôi xin bày tỏ lòng biết ơn sâu sắc tới TS. Phạm Văn Hải và các thầy cô Viện CNTT-TT, Trường Đại học Bách Khoa Hà Nội và TS đã nhiệt tình hướng dẫn và đào tọa cho tôi để tạo điều kiện thuận lợi cho tôi nghiên cứu khoa học, và giúp tôi có thể hoàn thành luận văn một cách tốt nhất.

            Cuối cùng tôi xin gửi lời cám ơn đến gia đình, bạn bè, những người đã luôn bên tôi, động viên và khuyến khích tôi trong quá trình thực hiện đề tài nghiên cứu của mình.

Học viên

Trần Hữu Minh

BOOKS

# **TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP**

Luận văn tác giả trình bày nghiên cứu ứng luật phân loại dùng kỹ thuật phân loại dữ liệu CMAR để giải quyết bài toán phân lớp tốt hơn. Sau đấy áp dụng phương pháp đề xuất này giải quyết các bài toán chuẩn đoán người bị bệnh.

**Bố cục đồ án gồm 6 chương :**

CHƯƠNG 1. GIỚI THIỆU. Chương này trình bày tổng quan, mục đích và các

nhiệm vụ cần giải quyết trong đồ án.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT. Chương này trình bày tổng quan về phân

lớp dữ liệu và giải thuật CMAR.

CHƯƠNG 3. MÔ HÌNH GIẢI QUYẾT BÀI TOÁN. Chương này trình bày mô

hình tích giải thuật CMAR để chuẩn đoán bệnh .

CHƯƠNG 4. TRIỂN KHAI CHƯƠNG TRÌNH. Chương này mô tả về dữ liệu

thử nghiệm và phân tích chức năng của Chương trình.

CHƯƠNG 5. CÀI ĐẶT VÀ THỬ NGHIỆM CHƯƠNG TRÌNH. Chương này

trình bày các kết quả cài đặt và thử nghiệm Chương trình, đồng thời đánh giá độ

chính xác của mô hình tích hợp so với một số phương pháp khác.

CHƯƠNG 6. KẾT LUẬT VÀ HƯỚNG PHÁT TRIỂN. Chương này tổng kết

các kết quả đạt được trong đồ án và một số hạn chế cần khắc phục, đề xuất hướng

nghiên cứu tiếp theo.

# MỤC LỤC

[LỜI CAM ĐOAN 3](#_Toc531036995)

[LỜI CẢM ƠN 4](#_Toc531036996)

[TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP 5](#_Toc531036997)

[MỤC LỤC 6](#_Toc531036998)

[DANH MỤC HÌNH VẼ 8](#_Toc531036999)

[DANH MỤC BẢNG 10](#_Toc531037000)

[THUẬT NGỮ VÀ TỪ VIẾT TẮT 11](#_Toc531037001)

[MỞ ĐẦU 12](#_Toc531037002)

[1. Lý do chọn đề tài 12](#_Toc531037003)

[2. Mục đích nghiên cứu của luận văn 12](#_Toc531037004)

[3. Phương pháp nghiên cứu 12](#_Toc531037005)

[4. Đối tượng, phạm vi nghiên cứu 12](#_Toc531037006)

[CHƯƠNG 1. GIỚI THIỆU 13](#_Toc531037007)

[1.1 Tổng quan 13](#_Toc531037008)

[1.2 Mục đích bài toán 13](#_Toc531037009)

[1.3 Định hướng giải quết bài toán 13](#_Toc531037010)

[1.4 Nhiệm vụ của đồ án 14](#_Toc531037011)

[1.5. Nội dung 14](#_Toc531037012)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 16](#_Toc531037013)

[2.1 Tổng quan về phân lớp dữ liệu 16](#_Toc531037014)

[2.1.1 Bài toán phân lớp dữ liệu 16](#_Toc531037015)

[2.1.2 Quá trình phân lớp dữ liệu 16](#_Toc531037016)

[2.2. Giới thiệu về mô hình CMAR 17](#_Toc531037017)

[2.2.1. Các khái niệm cơ bản 18](#_Toc531037018)

[2.2.2. Phân loại kết hợp 20](#_Toc531037019)

[2.2.3. Tạo luật cho phân loại 22](#_Toc531037020)

[2.2.4. Phân loại dựa trên nhiều quy tắc 33](#_Toc531037021)

[2.3. Giải thuật Apriori-TFP 37](#_Toc531037022)

[2.3.1. Tổng quan 37](#_Toc531037023)

[2.3.2. Cây tổng hỗ trợ (T-tree) 37](#_Toc531037024)

[2.3.3 Cây hỗ trợ một phần P-Tree 41](#_Toc531037025)

[2.3.4 Giải thuật Apriori-TFP 43](#_Toc531037026)

[CHƯƠNG 3. MÔ HÌNH GIẢI QUYẾT BÀI TOÁN 47](#_Toc531037027)

[3.1. Mô hình triển khai giải thuật CMAR để chuẩn đoán bệnh và dự báo khả năng nhiễm bệnh 47](#_Toc531037028)

[3.2. Các bước thực hiện 47](#_Toc531037029)

[CHƯƠNG 4. TRIỂN KHAI CHƯƠNG TRÌNH 56](#_Toc531037030)

[4.1. Bộ dữ liệu cài đặt thử nghiệm 56](#_Toc531037031)

[4.2 Phân tích chức năng 59](#_Toc531037032)

[CHƯƠNG 5. CÀI ĐẶT THỬ NGHIỆM CHƯƠNG TRÌNH 62](#_Toc531037033)

[5.1. Môi trường cài đặt 62](#_Toc531037034)

[5.2 Thử nghiệm chương trình 62](#_Toc531037035)

[5.2.1 Thử nghiệm thuật toán cây quyết định và thuật toán kết hợp JCBA 62](#_Toc531037036)

[5.2.2 Thử nghiệm giải thuật CMAR 69](#_Toc531037037)

[5.3. Đánh giá độ chính xác của thuật toán 77](#_Toc531037038)

[CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 80](#_Toc531037039)

[6.1 Kết luận 80](#_Toc531037040)

[6.2. Hướng phát triển 80](#_Toc531037041)

# DANH MỤC HÌNH VẼ

Hình 1. Cây FP trong Ví dụ 1……………………………………………...………......24

Hình 2. Cây FP…………………………………………………………………………27

Hình 3. Một cây CR có một nút gốc ……………………………………………...….. 30

Hình 4. Kết quả dự đoán và thực tế của tập luật …………………………………….…35

Hình 5. Cây T-tree (để dễ hiểu, các item/thuộc tính được liệt kê bắt đầu từ 1) …………39

Hình 6. Các bước triển khai giải thuật apriori TFP……………………………………. 43

Hình 7. Mô hình triển khai giải thuật CMAR để chẩn đoán bệnh và dự báo khả năng nhiễm bệnh…………………………………………………………………………….47

Hình 8. Mô hình xây dựng cây P-tree (PartialSupportTree)…………………………...52

Hình 9. Mô hình xây dựng cây T-tree (TotalSupportTree)……………………..….….54

Hình 10. Biểu đồ phân rã chức năng chương trình…………………………………….59

Hình 11. Giao diện chính chương trình Weka …………………………………………62

Hình 12. Giao diện Explorer của chương trình Weka………………………………….63

Hình 13. Mở file dữ liệu và hiện thị trong Weka………………………………………63

Hình 14. Cấu hình thuật toán J48 trong Weka………………………………………….64

Hình 15. Kết quả chạy của thuật toán J48 với bộ dữ liệu vô sinh trong Weka…………65

Hình 16. Cài đặt thuật toán JCBA trong Weka…………………………………………66

Hình 17. Kết quả chạy của thuật toán JCBA với bộ dữ liệu vô sinh trong Weka……….67

Hình 18. Kết quả chạy thuật toán J48 với bộ dữ liệu tuyến giáp trong Weka…………68

Hình 19. Kết quả chạy thuật toán JCBA với bộ dữ liệu tuyến giáp trong Weka………68

Hình 20. Giao diện chính chương trình………………………………………………...69

Hình 21. Giao diện chương trình sau khi load file dữ liệu thô…………………………70

Hình 22. Giao diện chương trình sau khi rời rạc hóa…………………………………..70

Hình 23. Giao diện chương trình sau khi chuẩn hóa dữ liệu……………………………71

Hình 24. Giao diện chương trình ở tab “Tạo các luật”…………………………………72

Hình 25. Cài đặt ở tab tạo các luật……………………………………………………..72

Hình 26. Kết quả tạo luật của thuật toán CMAR trên dữ liệu bệnh vô sinh……………73

Hình 27. Tab nhận dạng để kiểm thử dữ liệu test bệnh vô sinh………………………..74

Hình 28. Kết quả chạy giải thuật CMAR với bộ dữ liệu test bệnh vô sinh…………….74

Hình 29. Giao diện chuẩn đoán bệnh…………………………………………………..75

Hình 30. Kết quả chuẩn đoán bệnh…………………………………………………….76

Hình 31. Kết quả tạo luật của thuật toán CMAR trên dữ liệu bệnh tuyến giáp………..76

Hình 32. Kết quả chạy giải thuật CMAR với bộ dữ liệu test bệnh tuyến giáp…………77

Hình 33. Kết quả không phân loại được khi ngưỡng tin tưởng là 80…………………..78

Hình 34. Kết quả không phân loại được khi ngưỡng tin tưởng là 70…………………..79

# DANH MỤC BẢNG

Bảng 1. Một tập dữ liệu huấn luyện……………………………………………………24

Bảng 2. Các quy tắc được tìm thấy trong tập dữ liệu huấn luyện………………………29

Bảng 3. Tập dữ liệu ví dụ 4……………………………………………………………38

Bảng 4. Cấu trúc 1 node của bảng P-tree……………………………………………….45

Bảng 5. Mức chỉ số TSH cho trẻ em…………………………………………………..48

Bảng 6. Mức chỉ số TSH cho đàn ông…………………………………………………49

Bảng 7. Mức chỉ số TSH cho phụ nữ………………………………………………….49

Bảng 8. Mức chỉ số TT4 theo độ tuổi…………………………………………………49

Bảng 9. Mức chỉ số T4U……………………………………………………………....49

Bảng 10. Mức chỉ số T3 theo độ tuổi…………………………………………………..49

Bảng 11. Mức chỉ số FTI theo độ tuổi…………………………………………………50

Bảng 12. Mức chỉ số TBG theo độ tuổi………………………………………………..50

Bảng 13. Cấu trúc 1 node của bảng P-tree……………………………………………..52

Bảng 14. Các thuộc tính trong bộ dữ liệu bệnh vô sinh nam…………………………..57

Bảng 15. Các thuộc tính trong bộ dữ liệu bệnh tuyến giáp……………………………..58

Bảng 16. Kết quả chuẩn đoán bệnh……………………………………………………77

# THUẬT NGỮ VÀ TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| Từ viết tắt | Tên tiếng Anh | Thuật ngữ tiếng Việt |
| CMAR | Classification based on multiple association rules | Phân loại dựa trên nhiều luật kết hợp |
| FP-tree | Frequent Pattern Tree | Cây mẫu phổ biến |
| P-tree | Partial Support Tree | Cây hỗ trợ một phần |
| T-tree | Total Support Tree | Cây tổng hỗ trợ |
| CARs | Class Association Rules | Tập con của các luật kết hợp trong CSDL |
| ARM |  | Khai thác quy tắc kết hợp |
| Rule |  | Quy tắc (luật) |

# MỞ ĐẦU

## 1. Lý do chọn đề tài

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 lớn đã được các cơ quan tổ chức thu thập và lưu trữ ngày một nhiều lên. Họ lưu trữ các dữ liệu này vì cho rằng nó ẩn chứa những giá trị nhất định nào đó. Và việc khai thác những giá trị này là cần thiết. Vì vậy đồ án này đưa ra một phương pháp khai phá những giá trị hữu ích ẩn sau những bộ dữ liệu lớn một cách hiệu quả.

## 2. Mục đích nghiên cứu của luận văn

Đồ án tập trung nghiên cứu mô hình phân lớp dựa trên giải thuật Phân loại dựa trên nhiều luật kết hợp - CMAR nhằm tạo ra một mô hình phân lớp tốt hơn so với các luật liên kết thông thường. Sau đó áp dụng mô hình phân lớp này để giải quyết bài toán chuẩn đoán bệnh, cụ thể tác giả chạy thử nghiệm với 2 tập dữ liệu sẵn có trong lĩnh vực chuẩn đoán bệnh là bệnh vô sinh và bệnh suy tuyến giáp.

## 3. Phương pháp nghiên cứu

Tiếp cận các bài báo đã nghiên cứu về phân loại và dữ liệu mở của các bệnh nhân , tác giả đề xuất xây dựng mô hình phân loại dựa trên thuật toán CMAR để ứng dụng cho chuẩn đoán xác suất mắc bệnh của người khám thông qua các cơ sở dữ liệu đã lưu lại của các đối tượng bệnh nhân khác trước đó.

Bên cạnh đó tác giả kết hợp và phân tích kết quả của một số phương pháp phân loại để đánh giá kết quả của đề tài.

## 4. Đối tượng, phạm vi nghiên cứu

Đối tượng nghiên cứu là tập dữ liệu về bệnh như bệnh vô sinh ở nam giới và bộ dữ liệu tuyến giáp. Phạm vi nghiên cứu tập trung vào xây dựng mô hình phân lớp khai phá thuộc tính bệnh có giá trị và ứng dụng nó trong chuẩn đoán bệnh.

# CHƯƠNG 1. GIỚI THIỆU

## 1.1 Tổng quan

Phân lớp dữ liệu là một trong những hướng nghiên cứu chính của khai phá dữ liệu. Công nghệ này đã, đang và sẽ có nhiều ứng dụng trong các lĩnh vực thương mại, ngân hàng, y tế, giáo dục…Có rất nhiều mô hình được đề xuất để giải quyết bài toán phân lớp như: Luật phân loại, mạng noron, cây quyết định, máy hỗ trợ vector (Support Vector Machine - SVM),…  
Xu hướng hiện nay là tích hợp nhiều thuật toán để tăng cường độ chính xác khi phân lớp. Cũng theo cách tiếp cận như vậy trong đồ án tác giả đề xuất một mô hình dùng nhiều luật liên kết để phân loại vớ độ chính xác cao. Sau đó tác giả áp dụng mô hình này để giải quyết bài toán chuẩn đoán bệnh. Cả hai bài toán này đều có thể giải quyết bằng cách áp dụng thuật toán phân lớp. Dưới đây là mô tả chi tiết về hai bài toán này :

**Bài toán chuẩn đoán bệnh** :

* Đầu vào: Một véc tơ trong đó các thuộc tính là nguyên nhân gây bệnh
* Đầu ra : Phân loại đánh giá mắc bệnh hay không

## 1.2 Mục đích bài toán

Đồ án tập trung nghiên cứu mô hình phân lớp dự trên thuật toán CMAR kết hợp nhiều luật phân loại nhằm tạo ra một mô hình phân lớp tốt hơn so với các luật liên kết thông thường. Theo đó áp dụng mô hình phân lớp này để giải quyết bài toán chuẩn đoán bệnh.

## 1.3 Định hướng giải quết bài toán

Để giải quyết bài toán này ta cần phải thực hiện các công việc sau :

* Thu thập dữ liệu của nhiều người bao gồm: Các yếu tố có thể là nguyên nhân gây bệnh và tình trạng bệnh của họ (mắc bệnh hay bình thường).
* Xây dựng thuật toán triển khai CMAR để giải quyết bài toán
* Giải quyết bài toán chuẩn đoán bệnh dùng module thuật toán vừa xây dựng

## 1.4 Nhiệm vụ của đồ án

* Nghiên cứu về luật liên kết cho phân loại dữ liệu.
* Nghiên cứu thuật toán CMAR: Phân loại dựa trên nhiều luật liên kết.
* Áp dụng mô hình này để giải quyết bài toán chuẩn đoán bệnh.
* Thử nghiệm đánh giá độ chính xác của mô hình dùng CMAR với một số mô hình phân lớp riêng lẻ.

## 1.5. Nội dung

Khai thác luật phân lớp kết hợp được đề xuất bởi Liu và các đồng sự vào năm 1998. Thuật toán CBA cũng đã được đề xuất trong công trình này. Phương pháp này thường cho độ chính xác cao hơn so với các phương pháp phân lớp dựa trên luật khác như cây quyết định, ILA, v.v. Hiện nay đã có nhiều thuật toán được phát triển nhằm làm tăng độ chính xác, giảm thời gian khai thác như CMAR. CMAR đề xuất phương pháp dự đoán lớp của mẫu mới dựa vào đa luật nên thường có độ chính xác cao hơn so với CBA.

- Trình bảy giải thuật CMAR: xây dựng cây CR-tree dựa trên tập các mẫu lớn hơn một giá trị ngưỡng hỗ trợ, dựa trên đó cắt tỉa tạo ra cây FP-tree và khai thác được các luật phân loại.

- Kết hợp các luật phân loại phù hợp để xác định nhãn của mẫu test.

- Trình bày giải thuật Apriori-TFP dựa trên cơ sở giải thuật CMAR và lợi ích vượt trội hơn của giải thuật này so với sử dụng cây FP-tree.

- Xây dựng chương trình dự đoán bệnh dựa trên nguồn dữ liệu “Bệnh vô sinh ở nam giới”

Tóm tắt qua nội dung như sau :

MỞ ĐẦU

- Lý do chọn đề tài

- Mục đích nghiên cứu của luận văn, đối tượng, phạm vi nghiên cứu.

- Phương pháp nghiên cứu.

CHƯƠNG 1: GIỚI THIỆU

- Giới thiệu về phân loại dữ liệu

- Phát biểu bài toán

- Định hướng giải quyết bài toán

- Nhiệm vụ đồ án

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

- Tổng quan về phân lớp dữ liệu

- Giới thiệu cây phân loại

- Giới thiệu về mô hình CMAR

- Giải thuật Apriori-TFP

CHƯƠNG 3: MÔ HÌNH GIẢI QUYẾT BÀI TOÁN

- Mô hình ứng dụng CMAR để chuẩn đoán bệnh

- Các bước thực hiện

CHƯƠNG 4: TRIỂN KHAI CHƯƠNG TRÌNH

- Giới thiệu bộ dữ liệu cài đặt thử nghiệm

- Phân tích chức năng

CHƯƠNG 5: CÀI ĐẶT VÀ THỬ NGHIỆM CHƯƠNG TRÌNH

- Cài đặt chương trình

- Thử nghiệm chương trình

- Đánh giá độ chính xác phô hình

CHƯƠNG 6: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

- Kết luận

- Định hướng phát triển

# CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

## 2.1 Tổng quan về phân lớp dữ liệu

### 2.1.1 Bài toán phân lớp dữ liệu

Phân lớp là quá trình xử lý sắp xếp một đối tượng dữ liệu vào một hay nhiều lớp cho trước, hay được gọi là quá trình gán nhãn cho các đối tượng dữ liệu. Nhiệm vụ của bài toán phân lớp dữ liệu là xây dựng mô hình phân lớp để khi có dữ liệu mới vào thì mô hình này sẽ cho biết dữ liệu đấy thuộc lớp nào.

Luận văn đề cập đến hai bài toán phân lớp sau:

* *Phân lớp nhị phân* là quá trình phân lớp dữ liệu đầu vào thành một trong hai nhãn lớp khác nhau.
* *Phân lớp đa lớp* là quá trình phân lớp với đầu ra số nhãn lớp lớn hơn hai. Theo đó tập hợp dữ liệu trong miền xử lý được phân chia thành nhiều lớp chứ không phải chỉ đơn thuần là hai lớp như trong bài toán *phân lớp nhị phân*.

Thực chất, bài toán phân lớp nhị phân là trường hợp riêng của bài toán phân lớp đa lớp. Trong phân lớp đa lớp, mỗi đối tượng dữ liệu trong tập huấn luyện cũng như các đối tượng mới sau khi được phân lớp có thể thuộc từ hai nhãn trở lên. Ví dụ một chiếc đồng hồ thông minh có thể được phân loại vào hai nhãn: thuộc lĩnh vực thời trang và thuộc lĩnh vực công nghệ.

### 2.1.2 Quá trình phân lớp dữ liệu

Quá trình phân lớp dữ liệu thường gồm hai bước: xây dựng mô hình ( bộ phân lớp) và sử dụng mô hình này để phân lớp dữ liệu.

*Bước 1*: Xây dựng mô hình được dựa trên các đối tượng dữ liệu đã được gán nhãn từ trước. Tập các mẫu dữ liệu này thường được gọi là tập dữ liệu huấn luyện( trainning set). Các nhãn lớp của tập dữ liệu huấn luyện được xác định trước, vì thế phương pháp này còn được gọi là phương pháp học có giám sát ( surpervised learning).

Trong bước này , chúng ta còn phải tính độ chính xác của mô hình vì vậy cần phải sử dụng một tập dữ liệu kiểm tra ( testing set). Nếu độ chính xác là chấp nhận được thì mô hình này sẽ được sử dụng để xác định nhãn lớp cho các dữ liệu mới trong tương lai.

Tồn tại nhiều phương pháp phân lớp dữ liệu như phương pháp Bayes, cây quyết định , mọng nơ ron, các luật phân loại,…

Bước 2: Sử dụng mô hình đã được xây dựng ở bước 1 để phân lớp dữ liệu mới.

## 2.2. Giới thiệu về mô hình CMAR

Các nghiên cứu trước đây đề xuất rằng phân loại kết hợp có độ chính xác phân loại cao và tính linh hoạt cao trong việc xử lý dữ liệu phi cấu trúc. Tuy nhiên, nó vẫn bị ảnh hưởng bởi các quy tắc khai thác lớn và đôi khi phân loại bị lệch ​​hoặc quá đúng do phân loại dựa trên quy tắc duy nhất có độ tin cậy cao [1].

Trong luận văn, tác giả trình bày nghiên cứu phương pháp phân loại kết hợp CMAR (phân loại dựa trên nhiều luật liên kết) của *các tác giả Wenmin Li, Jiawei Han, Jian Pei (School of Computing Science, Simon Fraser University).*

1. Thay vì dựa vào một luật duy nhất để phân loại, CMAR xác định nhãn lớp theo một tập luật. Với một dữ liệu mới đưa vào phân loại, CMAR chọn một tập thuộc tính nhỏ nhưng có độ tin cậy cao, các luật liên quan chặt chẽ và phân tích mối tương quan giữa các luật đó.

Để tránh giá trị có độ lệch lớn, CMAR sử dụng giá trị trọng số làm thước đo tiêu chuẩn xác định một luật mạnh thông qua hỗ trợ điều kiện và phân bố nhãn lớp.

1. Để cải thiện tính chính xác và tính hiệu quả, CMAR sử dụng cấu trúc dữ liệu mới CR-tree, để lưu trữ và truy xuất được số lượng lớn các luật phân loại.

CR-tree là cấu trúc cây tiền tố để khám phá sự phân chia giữa các luật, đạt được mức độ liên kết cao. Bản thân CR-tree cũng là một cấu trúc chỉ mục cho các luật và có thể thực hiện truy xuất luật một cách hiệu quả.

1. Để tăng tốc độ khai thác các luật hoàn chỉnh, CMAR phát triển cây tăng trưởng FP-growth. FP-growth nhanh hơn nhiều so với các phương pháp được ưu tiên sử dụng trong phân loại dựa trên kết hợp trước đó, chẳng hạn như [9, 3, 11].

Đặc biệt là khi tồn tại số lượng lớn các luật, tập dữ liệu huấn luyện lớn và luật mẫu dài.

### 2.2.1. Các khái niệm cơ bản

#### 2.2.1.1 Khái niệm

Giả sử một đối tượng dữ liệu obj = (a1, …, an) theo mô hình thuộc tính (A1, ….,An), với A1,….,An là các thuộc tính. Các thuộc tính có thể rời rạc hoặc liên tục:

* Đối với các thuộc tính rời rạc, giả định rằng tất cả các giá trị có thể được ánh xạ tới một tập các số nguyên dương liên tiếp.
* Đối với các thuộc tính liên tục, giả định rằng phạm vi giá trị được phân loại theo khoảng, và các khoảng này được ánh xạ tới các số nguyên dương liên tiếp.

Cho C = {c1 ,…, cm} là tập các nhãn hữu hạn.

Một tập dữ liệu huấn luyện là một tập các đối tượng dữ liệu sao cho: với mỗi đối tượng obj, tồn tại một nhãn cobj C liên kết với nó.

Một bộ phân loại là một hàm của tập thuộc tính (A1,….,An) và (obj) C trả về một nhãn lớp.

Nhiệm vụ phân loại là xây dựng một bộ phân loại từ tập dữ liệu huấn luyện sao cho nó có thể được sử dụng để dự đoán nhãn lớp của các đối tượng chưa xác định với độ chính xác cao.

Bên cạnh nhiều cách tiếp cận khác nhau để phân loại, như cách tiếp cận cây quyết định, Naive Bayes, k lân cận gần nhất, mạng nơ-ron thì có một phương pháp mới là khám phá mối quan hệ liên kết giữa các điều kiện của đối tượng và nhãn [4]. Nó sử dụng các “tập phổ biến” và mối quan hệ liên kết giữa các trường hợp và các nhãn lớp trong tập dữ liệu huấn luyện để phân loại. Nếu các mối liên hệ chặt chẽ giữa một số “tập phổ biến” và nhãn lớp có thể được phát hiện trong tập dữ liệu huấn luyện, thì đối tượng mới trong tương lai của các mẫu tương tự cũng phân loại được.

Giả sử cho một mẫu P = ai1 … aik là tập các giá trị thuộc tính với 1 ≤ j ≤ k, aij Aij, và ij ≠ ij’ với j’ ≠ j . Một đối tượng dữ liệu obj được cho phù hợp với mẫu P = ai1 … aik khi và chỉ khi (1 ≤ j ≤ k) obj có giá trị aij nằm trong thuộc tính Aij.

Luật kết hợp là một mối liên hệ điều kiện giữa hai tập các hạng mục dữ liệu P và c theo dạng sau:

|  |
| --- |
| Cho một tập huấn luyện T và c là một nhãn.  Luật R : P 🡪 c  Số các đối tượng trong T phù hợp với mẫu P và có nhãn lớp c được gọi là **độ** **hỗ trợ** của R, ký hiệu là ***sup(R)****.*  Tỉ số giữa số lượng đối tượng phù hợp với mẫu P và có nhãn lớp c so với tổng số đối tượng phù hợp với mẫu P, được gọi là **độ tin cậy** của R, ký hiệu là ***conf(R)***. |

#### 2.2.1.2. Độ hỗ trợ (Support)

Thể hiện phạm vi ảnh hưởng của luật trên toàn bộ cơ sở dữ liệu.

Luật P🡪c có độ support là s nếu s% số bản ghi trong tập dữ liệu (N bản ghi) có chứa P∪c. Hay là:

***Support(P***🡪***c) = Support(P∪c) = P∪ c / N = s%***

Với:

P∪c tập các bản ghi trên cơ sở dữ liệu có chứa cả vế trái lẫn vế phải của luật.

N : Tập tất cả các dòng trên cơ sở dữ liệu.

#### 2.2.1.3. Độ tin cậy (Confidence)

Thể hiện độ chính xác, tính đúng đắn, khả năng tin cậy của luật trong phạm vi ảnh hưởng của luật (xác định bởi độ đo support).

Giả sử :

***P*** : Tổng số dòng chứa vế trái của luật trên cơ sở dữ liệu.

***P∪ c*** : tập các bản ghi trên cơ sở dữ liệu có chứa cả vế trái lẫn vế phải của luật.

Luật “***P🡪c***” có độ tin cậy là ***c*** (confidence) nếu có c% số bản ghi (N bản ghi) chứa ***P∪c*** so với tổng số bản ghi chỉ chứa ***P***. Khi đó ta có :

***Confidence(P***🡪***c) = P∪ c / P = c%***

Tập các hạng mục dữ liệu gọi là ItemSet có độ support lớn hơn hay bằng giá trị ngưỡng nhỏ nhất (gọi là minsup) được gọi là Large ItemSet. Các ItemSet còn lại được gọi là các Small ItemSet.

Với mỗi một Large ItemSet - L, và A là một tập con khác rỗng của L, nếu tỉ lệ phần trăm giữa support của L so với support của A lớn hơn hay bằng độ tin cậy nhỏ nhất (gọi là minconf) thì ta có luật kết hợp A ⇒ (L\A).

Tóm lại tìm luật kết hợp là đi tìm những luật P🡪c trong cơ sở dữ liệu sao cho luật thỏa mãn những giới hạn tối thiểu support và confidence cho trước.

### 2.2.2. Phân loại kết hợp

Để nghiên cứu phân loại kết hợp, ta tìm hiểu bài toán sau:

Có 95% khách hàng không có việc làm không thể nhận được hạn mức tín dụng nhiều hơn 3000$, tức là, độ tin cậy của luật R: không việc làm 🡪 hạn mức tín dụng dưới 3000 là 95%, sau đó chúng ta có thể sử dụng luật R phân loại các đối tượng dữ liệu trong tương lai.

* Để tránh nhiễu dữ liệu, luật chỉ được sử dụng để phân loại nếu nó đặt tối thiểu bằng ***mức hỗ trợ***. Với ngưỡng hỗ trợ và ngưỡng tin cậy, phương pháp phân loại kết hợp tìm thấy tập luật liên kết lớp (CAR) vượt qua các ngưỡng.
* Khi một đối tượng mới xuất hiện, bộ phân loại chọn luật tương thích với đối tượng dữ liệu và có độ tin cậy cao nhất, sử dụng luật này để dự đoán nhãn của đối tượng mới.
* Tuy nhiên, quyết định dựa trên độ tin cậy cao nhất không phải lúc nào cũng đúng.

Ví dụ: khi xác định hạn mức tín dụng của khách hàng, với các giá trị thuộc tính: không có công việc, người nhập cư đầu tư, tài sản nước ngoài trên 500.000.

Top 3 các luật có độ tin cậy cao nhất phù hợp với khách hàng như sau :

* Luật R1: không việc 🡪 hạn mức tín dụng dưới 3000 (hỗ trợ: 3000, độ tin cậy: 95%)
* Luật R2: người nhập cư đầu tư 🡪 hạn mức tín dụng trên 3000 (hỗ trợ: 5000, độ tin cậy: 95%)
* Luật R3: tài sản nước ngoài trên 500.000 🡪 hạn mức tín dụng trên 3000 (hỗ trợ: 8000, độ tin cậy: 91%)

Vì vậy, với một khách hàng, chúng ta nên dự đoán nhãn lớp nào chính xác nhất.

Với phương pháp phân loại liên kết thông thường (như CBA) có thể dự đoán giới hạn tín dụng dưới 3000 chỉ theo luật R1 vì nó có độ tin cậy cao nhất. Tuy nhiên, việc xem xét kỹ hơn các luật R2 và R3 gợi ý nên xem xét lại quyết định một cách nghiêm túc. Ba luật trên có độ tin cậy tương tự nhau, nhưng R2 và R3 có độ hỗ trợ cao hơn. Quyết định dựa trên các luật R2 và R3 có vẻ đáng tin cậy hơn.

Ví dụ trên chỉ ra rằng để đưa ra dự đoán chính xác và đáng tin cậy, luật có độ tin cậy cao nhất không phải lúc nào cũng là lựa chọn tốt nhất và cần phân tích kỹ lưỡng, chi tiết và toàn diện dựa trên nhiều luật sẽ dẫn đến dự đoán chính xác hơn.

### 2.2.3. Tạo luật cho phân loại

CMAR bao gồm hai giai đoạn: tạo luật và phân loại.

* Giai đoạn tạo luật:

CMAR tính toán tập luật hoàn chỉnh dưới dạng R : P 🡪 c, với P là một mẫu trong tập dữ liệu huấn luyện và c là nhãn lớp sao cho sup(R) và conf(R) thoả mãn ngưỡng hỗ trợ và ngưỡng tin cậy.

Trong khi đó CMAR cắt tỉa một số luật và chỉ chọn một tập con các luật chất lượng để phân loại.

* Giai đoạn phân loại:

CMAR trích xuất một tập hợp con các luật khớp với đối tượng và dự đoán nhãn lớp của đối tượng bằng cách phân tích tập con của các luật này.

Có hai sự khác biệt lớn trong khai phá luật trong CMAR và thuật toán tăng trưởng FP chuẩn.

Một mặt, CMAR tìm thấy các tập phổ biến và tạo ra các luật trong một bước.

Thông thường, các luật liên kết phải được khai thác theo hai bước. Đây cũng là trường hợp của các phương pháp phân loại kết hợp truyền thống. Đầu tiên, tất cả các tập phổ biến (tức là, các mẫu vượt qua ngưỡng hỗ trợ) được tìm thấy. Sau đó, tất cả các luật kết hợp thỏa mãn ngưỡng tin cậy được tạo dựa trên các tập phổ biến được khai thác.

Sự khác biệt của CMAR từ các phương pháp phân loại liên kết khác là đối với mọi mẫu, CMAR duy trì sự phân bố của các nhãn lớp khác nhau giữa các đối tượng dữ liệu khớp với mẫu. Điều này được thực hiện mà không có bất kỳ chi phí nào trong quy trình đếm (điều kiện) cơ sở dữ liệu. Do đó, khi một tập phổ biến (tức là, mẫu vượt qua ngưỡng hỗ trợ) được tìm thấy, các luật về mẫu có thể được tạo ngay lập tức. Do đó, CMAR không có bước tạo luật riêng.

#### 2.2.3.1. Khai phá các luật liên kết

Để tìm tập luật để phân loại, CMAR đầu tiên khai phá bộ dữ liệu huấn luyện để tìm tập hoàn chỉnh các luật vượt qua ngưỡng hỗ trợ và ngưỡng tin cậy.

Để khai thác có khả năng mở rộng và hiệu quả cao, CMAR sử dụng phương pháp cây tăng trưởng FP (FP-growth). Cây FP-growth là một thuật toán khai thác tập phổ biến nhanh hơn các phương pháp thông thường như phương pháp Apriori, đặc biệt trong các tình huống có tồn tại các tập dữ liệu lớn, ngưỡng hỗ trợ thấp và các mẫu dài.

##### 2.2.3.1.1. Xây dựng cây FP-tree

Cây FP là cây tiền tố tương ứng với danh sách F. Đối với mỗi bộ dữ liệu trong tập dữ liệu huấn luyện, các giá trị thuộc tính xuất hiện trong danh sách F được trích xuất và sắp xếp theo thứ tự giảm dần.

**Bước 1**: Duyệt cơ sở dữ liệu lần thứ nhất, lấy ra tập F1 các phần tử phổ biến và độ support của chúng. Sắp theo thứ tự giảm dần độ support các phần tử trong tập F, ta được tập F1. Bỏ qua các phần tử không phổ biến.

**Bước 2**: Tạo một nút gốc cho cây T và gán nhãn là Null. Duyệt CSDL lần thứ hai, với mỗi giao tác trong cơ sở dữ liệu, thực hiện hai việc sau:

- Chọn các phần tử phổ biến trong mỗi giao tác sắp xếp support giảm dần theo thứ tự trong tập F1. Ký hiệu danh sách các phần tử đã sắp xếp là [p|P], trong đó p là phần tử đầu tiên của danh sách, P là các phần tử còn lại.

- Gọi hàm **Insert\_tree([p|P],T)** để đưa các phần tử trong danh sách vào cây T.

* ***Ví dụ minh hoạ 1:***

Một tập dữ liệu huấn luyện D được hiển thị trong bảng. Đặt ngưỡng hỗ trợ là 2 (minSup = 2) và ngưỡng tin cậy là 50% (minConf = 50).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id hàng | A | B | C | D | Nhãn | Tập phổ biến |
| 1 | a1 | b1 | c1 | d1 | A | a1, c1 |
| 2 | a1 | b2 | c1 | d2 | B | a1, b2, c1 |
| 3 | a2 | b3 | c2 | d3 | A | d3 |
| 4 | a1 | b2 | c3 | d3 | C | a1, b2, d3 |
| 5 | a1 | b2 | c1 | d3 | C | a1, b2, c1, d3 |

***a) dữ liệu đầu vào***

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Thuộc tính | **a1** | a2 | b1 | **b2** | b3 | **c1** | c2 | c3 | d1 | d2 | **d3** |
| Tần suất | **4** | 1 | 1 | **3** | 1 | **3** | 1 | 1 | 1 | 1 | **3** |

**b) tần suất của các giá trị**

**Bảng 1. Một tập dữ liệu huấn luyện**

**Lần duyệt thứ nhất*:*** Tìm tập hợp các giá trị thuộc tính xảy ra ít nhất hai lần trong T. Bộ này là F = {a1 , b2 , c1, d3 } và được gọi là các tập phổ biến (1-frequent itemset). Tất cả các giá trị thuộc tính khác, không đạt ngưỡng hỗ trợ sẽ được cắt bớt.

**Duyệt lần hai** để xây dựng FP-Tree, chú ý sắp xếp các giá trị thuộc tính trong F theo thứ tự giảm dần, tức là danh sách F = a1 - b2 - c1 - d3.

a1

b2

c1

d3

d3

(A:1)

Table

Header Table

root

Table

a1

Table

b2

Table

c1

(A:1)

Table

c1

(B:1)

Table

d3

(B:1)

Table

d3

(C:1)

Table

Item Link

Table

**Hình 1. Cây FP trong Ví dụ 1**

|  |
| --- |
| 1. Khởi tạo: **Root** 2. Chèn đối tượng đầu tiên: ID = 1 với P: (a1, c1)   Chèn vào trong cây T ở nhánh bên trái nhất.   1. Chèn đối tượng thứ hai: ID = 2 với P: (a1, b2, c1)   Thực hiện **Insert\_tree([p|P],T): nếu** T đã có cây mà nơi nhãn bằng A thì tăng nhãn lên 1. Nếu khác nhãn với A tạo nút mới n với n.count = 1  Tiếp tục thực hiện như thế.   1. Chèn đối tượng thứ ba: ID = 3 với P: (d3) 2. Chèn đối tượng thứ tư: ID = 4 với P: (a1, b2, d3) 3. Chèn đối tượng thứ năm: ID = 5 với P: (a1, b2, c1, d3) |

* **Thuật toán xây dựng FP-Tree**
* **Input:**

- Cơ sở dữ liệu giao tác D

- Ngưỡng hỗ trợ min-sup

* **Output:**

- Cây FP-Tree

* **Phương pháp**

a) *Duyệt D lần đầu để thu được tập F gồm các frequent item và số support của chúng. Sắp xếp các item trong F theo trật tự giảm dần của số support ta được danh sách F1.*

*b) Tạo nút gốc R và gán nhãn “null”*

*Tạo bảng Header có |F| dòng và đặt tất cả các node–link chỉ đến null*

*c) For each giao tác T D {*

*// Duyệt D lần 2*

*Chọn các item phổ biến của T đưa vào P;*

*Sắp các item trong P theo trật tự F1;*

*Call Insert\_Tree(P, R);*

*}*

*d) Procedure Insert\_Tree(P, R)* {

*Đặt P=[p|P-p], với p là phần tử đầu tiên và P-p là phần còn lại của danh sách if R có một con N sao cho N.item-name = p then*

*N.count ++*

*else {*

*Tạo nút mới N;*

*N.count = 1;*

*N.item-name = p*

*N. parent = R*

*// Tạo node-link chỉ đến item, H là bảng Header*

*N.node-link = H[p].head*

*H[p].head = N*

*}*

*// Tăng biến count của p trong bảng header thêm 1*

*H[p].count ++;*

*If (P-p) != null then*

*Call Insert\_Tree(P-p, N)*

*}*

*e) Tất cả các nút có cùng giá trị thuộc tính được liên kết với nhau như thành hàng đợi được bắt đầu từ bảng Header.*

##### 2.2.3.1.2. Xây dựng cây FP-growth

***Tính chất 1: Node-link property***

Với mỗi phần tử phổ biến ai, tất cả các tập phổ biến có chứa phần tử này có thể tìm được bằng cách dò theo các nút liên kết bắt đầu từ ai. Tính chất này có cơ sở trực tiếp từ cách xây dựng FP-tree. Điều này cho phép dễ dàng tìm ra tất cả các tập phổ biến có chứa ai bằng cách duyệt FP-tree một lần theo các nút liên kết với ai.

**Ví dụ:** Tính chất Node-link property khám phá các tập phổ biến dựa trên FP-tree

Trên cơ sở tính chất 1, thu thập tất cả các tập phổ biến mà có chứa nút ai bắt đầu từ nút đầu ai trong header table và các nút liên kết tiếp theo. Xem xét bắt đầu từ nút cuối cùng trong header table.

* Với nút d3, chúng ta thu được mẫu phổ biến (d3:3) và 3 đường đi trên FP-tree (a1 , b2 , c1, d3 ): C; (a1 , b2 , d3 ): C và d3: A. Vấn đề của việc tìm kiếm tất cả các mẫu phổ biến có d3 trong toàn bộ tập huấn luyện có thể được giảm xuống để khai phá các mẫu phổ biến trong cơ sở dữ liệu của d3.
* Sử dụng đệ quy: “trong cơ sở điều kiện của d3” a1 và b2 là các giá trị thuộc tính phổ biến, tức là chúng vượt qua ngưỡng hỗ trợ. Trong cơ sở dữ liệu d3 xảy ra trong mọi bộ dữ liệu và do đó được coi là phổ biến. Ta có thể khai thác cơ sở dữ liệu điều kiện đệ quy bằng cách xây dựng cây FP và cơ sở điều kiện.
* Trong cơ sở điều kiện của d3, a1 và b2 luôn luôn xuất hiện cùng nhau và do đó a1b2 là một mẫu phổ biến. a1 và b2 là hai mẫu con của a1b2 và có cùng hỗ trợ như a1b2 .

Để tránh kết quả tầm thường, chỉ nên áp dụng mẫu phổ biến a1b2d3. Dựa trên thông tin phân phối nhãn lớp, ta tạo luật a1b2d3 🡪 C với hỗ trợ 2 và mức tin cậy 100%.

a1

b2

c1

d3

d3

(A:1)

Table

Header Table

root

Table

a1

Table

b2

Table

c1

(A:1)

Table

c1

(B:1)

Table

d3

(B:1)

Table

d3

(C:1)

Table

Item Link

Table

a) P-Tree

Table

a1

b2

c1

Header Table

root

Table

a1

Table

b2

(C:1)

c1

(A:1)

Table

c1

(B:1, C:1)

Table

Item Link

Table

b) P-Tree sau khi ghép node d3

**Hình 2. Cây P-tree**

* Sau khi tìm kiếm các luật có d3, tất cả các nút của d3 được hợp nhất vào các nút cha của chúng tương ứng. Tức là, thông tin nhãn lớp đã đăng ký trong nút d3 được đăng ký trong nút cha của nó. Cây FP bị thu hẹp như trong Hình 2(b). Hoạt động thu hẹp cây này được thực hiện tại cùng một lần quét thu thập trong cơ sở điều kiện của d3.

***Tính chất 2: Prefix path property***

Để tìm ra các tập phổ biến từ một nút ai trên đường đi P, chỉ cần đến đường đi con trước nút ai trên P. Và số lần phổ biến của mỗi nút trên đường đi con này phải bằng với số lần phổ biến của nút ai.

Dựa trên các tính chất trên của thuật toán tìm các tập phổ biến trên cây FP-tree.

* **Thuật toán xây dựng FP-Growth**
* **Input**: cây FP-Tree của CSDL D, ngưỡng min\_sup
* **Output**: Một tập đầy đủ các mẫu phổ biến F
* **Phương pháp:** gọi FP-growth(FP-tree, null)

*Procedure* ***FP-growth*** *(Tree, α){*

***If*** *Tree chỉ chứa một đường dẫn đơn P* ***then*** *{*

***for each*** *tổ hợp của các nút trong P* ***do*** *{*

*phát sinh mẫu p = ;*

*support\_count(p) = min\_sup các nút trong*

*F = F*

*}*

*}*

***Else*** {

***For each*** *ai trong bảng header của Tree {*

*Phát sinh mẫu = ai­ ;*

*support\_count () = ai.support\_count*

*F = F*

*Xây dựng cơ sở có điều kiện của*

*Xây dựng FP-Tree có điều kiện ­của*

***if*** *!=*

*call FP\_growth(, );*

*}*

#### 2.2.3.2. Lưu trữ các quy tắc trong cây CR

Khi một quy tắc được tạo ra, nó được lưu trữ trong một cây CR, là một cấu trúc cây tiền tố. Đồ án trình bày ý tưởng chung của cây CR trong ví dụ sau [1].

**Ví dụ 2.** Sau khi khai thác một tập dữ liệu huấn luyện, bốn quy tắc được tìm thấy như trong Bảng 2.

|  |  |  |  |
| --- | --- | --- | --- |
| RuleId | Quy tắc | Hỗ trợ | Mức tin cậy |
| 1 | abc 🡪 A | 80 | 80% |
| 2 | abcd 🡪 A | 63 | 90% |
| 3 | abd 🡪 B | 36 | 60% |
| 4 | bcd 🡪C | 201 | 70% |

**Bảng 2. Các quy tắc được tìm thấy trong tập dữ liệu huấn luyện**

Một cây CR được xây dựng cho bộ quy tắc như trong Hình 2, trong khi quá trình xây dựng được giải thích như sau:

Header Table

Item head of node-links

a

b

c

d

e

root

Table

a

b

b

c

c (A, 80, 80%)

Table

e (B, 36, 60%)

Table

d (C, 210, 70%)

Table

d (A, 63, 90%)

Table

**Hình 3. Một cây CR có một nút gốc**

Một cây CR có một nút gốc. Tất cả các giá trị thuộc tính xuất hiện ở phía bên trái của luật đều được sắp xếp theo tần suất của chúng, nghĩa là giá trị thuộc tính phổ biến xuất hiện đầu tiên.

* Luật đầu tiên, abc 🡪 A được đưa vào cây bắt đầu từ nút gốc.

Nhãn lớp, mức hỗ trợ và mức tin cậy của luật, ký hiệu theo thứ tự là (A, 80, 80%) và được đánh dấu vào nút cuối cùng trong nhánh, nghĩa là nút c trong luật này.

* Luật thứ hai, abcd 🡪 A, chia sẻ tiền tố abc với luật đầu tiên. Do đó, nó được đưa vào cây bằng cách thêm chỉ một nút mới trên nhánh được hình thành bởi luật đầu tiên. Sau đó, nhãn lớp, mức hỗ trợ và mức tin cậy của luật được đánh dấu vào nút cuối cùng.
* Các luật thứ ba và thứ tư được đưa vào tương tự. Tất cả các nút có cùng giá trị thuộc tính được liên kết với nhau bằng một hàng đợi. Phần đầu của mỗi hàng đợi được lưu trữ trong một bảng *Header*.

Để lưu trữ tập luật cây CR chỉ cần 9 nút trong khi cây FP-tree lưu 13 nút.

Như có thể thấy từ ví dụ trên, cấu trúc cây CR có một số ưu điểm như sau.

* Cây CR là một cấu trúc nhỏ gọn, xác định khả năng chia sẻ giữa các luật và do đó tiết kiệm rất nhiều không gian lưu trữ.
* Cây CR chính bản thân nó là một chỉ số cho các luật. Khi chúng ta muốn lấy tất cả các luật có giá trị thuộc tính b và d trong tập hợp các luật trong *Ví dụ 2*, chúng ta chỉ cần duyệt qua các liên kết nút của d, bắt đầu từ bảng tiêu đề và tiếp tục tìm kiếm lên trên cho b.

- Khi một cây CR được xây dựng, việc truy xuất luật trở nên hiệu quả. Điều đó tạo điều kiện cho việc cắt tỉa các luật và sử dụng các luật để phân loại.

#### 2.2.3.3. Cắt tỉa các quy tắc

Số lượng các luật được tạo ra bởi khai phá các luật liên kết có thể rất lớn. Để làm cho việc phân loại có hiệu quả, chúng ta cần phải tỉa bớt các luật để xóa thông tin dư thừa và thông tin gây nhiễu.

Theo cơ sở của các luật về phân loại, cho hai luật R1 và R2, R1 được gọi là mạnh hơn R2, được biểu thị là R1 > R2, khi và chỉ khi xảy ra một trong ba trường hợp sau:

* *conf(R1) > conf(R2)*
* *conf(R1) = conf(R2) và sup(R1) > sup(R2)*
* *conf(R1) = conf(R2) và sup(R1) = sup(R2)* *nhưng số lượng thuộc tính trong vế điều kiện của R1 ít hơn so với R2.*

Ngoài ra, luật R1: P 🡪 c được gọi là một luật khái quát hơn luật R2: P’ 🡪 c’, khi và chỉ khi P là tập con của P’.

*CMAR sử dụng các phương pháp cắt tỉa luật như sau:*

*Đầu tiên*, sử dụng luật khái quát hơn và mức tin cậy cao để cắt giảm các luật cụ thể (có độ bao phủ thuộc tính nhỏ) và mức tin cậy thấp hơn. Với hai luật R1 và R2, trong đó R1 là một luật phổ biến hơn R2. CMAR cắt tỉa R2 nếu R1 mạnh hơn R2. Lý do là chỉ cần xem xét các luật *khái quát và mức tin cậy cao*, do đó các luật cụ thể hơn và mức tin cậy thấp nên được cắt tỉa.

Việc cắt tỉa này được thực hiện khi luật được chèn vào cây CR. Khi một luật được chèn vào cây, việc truy xuất trên cây được kích hoạt để kiểm tra xem luật có thể cắt tỉa hay không. Kết quả thử nghiệm của nghiên cứu cho thấy việc cắt tỉa này có hiệu quả.

*Thứ hai*, chỉ chọn các luật tương quan tích cực. Đối với mỗi luật R : P 🡪 c, kiểm tra xem liệu P có tương quan với c bởi . Chỉ các luật có tương quan thuận, tức là những luật có giá trị vượt qua ngưỡng mới được sử dụng để phân loại sau này. Tất cả các luật không thoả mãn còn lại sẽ được cắt tỉa.

Lý do của việc cắt tỉa này là ta chỉ sử dụng các luật phản ánh được những tác động mạnh mẽ để phân loại. Bằng cách loại bỏ những luật không tương quan tích cực, ta sẽ cắt giảm nhiễu.

*Thứ ba*, luật cắt tỉa dựa trên mức độ bao phủ của cơ sở dữ liệu. CMAR chọn một tập hợp con các luật chất lượng cao để phân loại. Điều đó đạt được bằng cách cắt tỉa các luật dựa trên độ bao phủ cơ sở dữ liệu. CMAR sử dụng một ngưỡng bao phủ để chọn vùng bao phủ của cơ sở dữ liệu, được trình bày trong *Thuật toán 1**(thuật toán cắt tỉa).*

Phương pháp bao phủ cơ sở dữ liệu được sử dụng trong CMAR tương tự như trong CBA. Sự khác biệt chính là, thay vì xóa một đối tượng dữ liệu khỏi tập dữ liệu huấn luyện ngay lập tức sau khi nó được bao phủ bởi một số luật đã chọn, ta để nó ở đó cho đến khi nó được bao phủ bởi ít nhất **luật**. Điều đó cho phép nhiều luật được chọn hơn. Khi phân loại một đối tượng dữ liệu mới, nó có thể có nhiều luật hơn để tham khảo và có thể có cơ hội tốt hơn để được dự đoán chính xác.

Việc cắt tỉa này được thực hiện khi quá trình khai phá luật kết thúc. Đây là lần cắt tỉa cuối cùng của các luật, được trình bày như sau:

* **Thuật toán cắt tỉa** (Chọn các luật dựa trên vùng bao phủ của cơ sở dữ liệu)
* **Đầu vào:** một tập luật và ngưỡng bao phủ
* **Đầu ra:** một tập con của các luật để phân loại
* **Phương pháp:**

1. *Sắp xếp các luật theo thứ tự giảm dần;*
2. *Đối với mỗi đối tượng dữ liệu trong tập dữ liệu huấn luyện, đặt biến đếm của nó bằng 0;*
3. *Trong khi cả tập dữ liệu huấn luyện và tập hợp luật đều không rỗng, đối với mỗi luật R theo thứ tự giảm dần, tìm tất cả các đối tượng dữ liệu khớp với luật R.*

*Nếu R có thể phân loại chính xác ít nhất một đối tượng thì hãy chọn R và tăng số lượng của các đối tượng phù hợp với R lên 1.*

*Một đối tượng dữ liệu được loại bỏ nếu số lượng của nó không vượt qua ngưỡng .*

### 2.2.4. Phân loại dựa trên nhiều quy tắc

Sau khi một bộ quy tắc được chọn để phân loại CMAR đã sẵn sàng để phân loại các đối tượng mới. Với một đối tượng dữ liệu mới, CMAR thu thập tập hợp con các quy tắc phù hợp với đối tượng mới từ tập hợp các quy tắc để phân loại. Trong phần này, phải tìm cách xác định nhãn lớp dựa trên tập con của các quy tắc [1, 11]. Có hai trường hợp xảy ra:

* Nếu tất cả các luật phù hợp với đối tượng mới có cùng nhãn lớp, chỉ cần gán nhãn đó cho đối tượng mới.
* Nếu các luật không thống nhất trong các nhãn lớp, CMAR sẽ chia tập luật thành các nhóm theo các nhãn lớp. Tất cả các luật trong một nhóm có cùng một nhãn lớp và mỗi nhóm có một nhãn riêng biệt. CMAR so sánh hiệu quả của các nhóm và chọn nhóm mạnh nhất.

Để so sánh độ mạnh của các nhóm, chúng ta cần tính “hiệu ứng kết hợp” của mỗi nhóm. Theo trực giác, nếu các luật trong một nhóm có mối tương quan dương và có mức hỗ trợ tốt, nhóm sẽ có tác động mạnh mẽ.

Có nhiều cách có thể để tính toán hiệu quả kết hợp của một tập luật. CMAR sử dụng luật mạnh nhất làm đại diện, đó là luật có giá trị cao nhất được chọn. Tuy nhiên, chỉ cần chọn luật có giá trị cao nhất có thể thuận lợi cho các lớp có kích thước nhỏ, như được minh họa trong *ví dụ 3*.

*Ví dụ 3: Trong trường hợp phê duyệt đơn xin thẻ tín dụng, có hai luật:*

R1 : công việc = không 🡪 từ chối (hỗ trợ = 30 , mức tin cậy = 60%)

R2 : giáo dục = đại học 🡪 chấp nhận (hỗ trợ = 200 , mức tin cậy = 99.5%)

Kết quả dự đoán và thực tế được thể hiện trong *Hình 4*.

|  |  |  |  |
| --- | --- | --- | --- |
| R1 | Chấp nhận | Từ chối | Tổng |
| Công việc = có | 438 | 32 | 470 |
| Công việc = không | 12 | 18 | 30 |
| Tổng | 450 | 50 | 500 |

*Dự đoán của luật R1*

|  |  |  |  |
| --- | --- | --- | --- |
| R2 | Chấp nhận | Từ chối | Tổng |
| Trình độ đại học = có | 199 | 1 | 200 |
| Trình độ đại học = không | 251 | 49 | 300 |
| Tổng | 450 | 50 | 500 |

*Dự đoán của luật R2*

|  |  |  |  |
| --- | --- | --- | --- |
| R1 | Chấp nhận | Từ chối | Tổng |
| Công việc = có | 423 | 47 | 470 |
| Công việc = không | 27 | 3 | 30 |
| Tổng | 450 | 50 | 500 |

*Kết quả thực tế của luật R1*

|  |  |  |  |
| --- | --- | --- | --- |
| R2 | Chấp nhận | Từ chối | Tổng |
| Trình độ đại học = có | 180 | 20 | 200 |
| Trình độ đại học = không | 270 | 30 | 300 |
| Tổng | 450 | 50 | 500 |

*Kết quả thực tế của luật R2*

***Hình 4. Kết quả dự đoán và thực tế của tập luật***

Dựa trên các giá trị dự đoán và thực tế, giá trị cho R1 và R2 là 88.4 và 33.6. Đối với khách hàng “không có việc làm” và với trình độ “giáo dục đại học” có thể dự đoán rằng đơn đăng ký của cô ấy bị từ chối bằng luật R1, nếu lựa chọn luật chỉ dựa trên giá trị .

Tuy nhiên, luật R2 có cảm giác tốt hơn nhiều so với R1 vì R2 có mức hỗ trợ và mức tin cậy cao hơn nhiều R1.

Một cách tốt hơn là tích hợp cả thông tin về sự tương quan và mức độ phổ biến vào thước đo. Sau khi xác minh theo kinh nghiệm, CMAR áp dụng một biện pháp có trọng số được xác định như sau.

Đối với mỗi luật R : P 🡪 c, để *sup(c)* là số đối tượng trong tập dữ liệu huấn luyện được liên kết với nhãn lớp c và |T| số lượng đối tượng trong tập dữ liệu huấn luyện, mô tả bằng các công thức sau:

Trong đó:

tính giới hạn trên của giá trị . Sau đó, đối với mỗi tập luật, trọng số của nhóm được định nghĩa là .

Theo đó, ta sử dụng giá trị so với giới hạn trên của nó (tức là max ) để vượt qua xu hướng ưu tiên lớp có kích thước nhỏ. Về mặt lý thuyết, rất khó để xác minh tính chính xác và hiệu quả của các phép tính toán dựa trên sức mạnh của các tập luật. Thay vào đó, ta đánh giá hiệu quả của các biện pháp theo kinh nghiệm và theo kết quả thử nghiệm, giá trị trọng số là tốt nhất trong số các công thức đo lường hiện nay.

## 2.3. Giải thuật Apriori-TFP

### 2.3.1. Tổng quan

Khai thác luật kết hợp (ARM) thu được từ một tập dữ liệu, một tập luật cho thấy hiệu quả khi một luật trong đó được áp dụng nếu tiền tố của luật này tương thích [8]. Để tạo các luật như vậy, bước đầu tiên là xác định mức hỗ trợ cho các tập các item phổ biến I. I có thể được sử dụng để sinh ra luật kết hợp của mẫu A 🡪 B, trong đó A và B là các tập con rời rạc của I. Các luật kết hợp được tạo ra thường được cắt tỉa theo một số khái niệm về mức tin cậy trong mỗi luật. Tuy nhiên khi việc cắt tỉa này đạt được, nó luôn luôn cần xác định tập phổ biến I chứa trong dữ liệu đầu vào. Điều này yêu cầu một cấu trúc lưu trữ dữ liệu hiệu quả.

Dựa trên điều này, một cơ chế lưu trữ dữ liệu hiệu quả cho các mục lưu trữ T-tree được tạo ra. Khi xem xét việc tiền xử lý dữ liệu cấu trúc P-tree được sinh ra, P-tree được sử dụng để thực hiện tính toán một phần của tổng hỗ trợ. Sau đó tiếp tục cho thấy việc sử dụng các cấu trúc này mang lại những lợi thế đáng kể dựa trên việc tôn trọng các kỹ thuật ARM hiện có [6].

### 2.3.2. Cây tổng hỗ trợ (T-tree)

Chi phí quan trọng nhất khi đánh giá cấu trúc dữ liệu ARM là số lượng các kết hợp tạo bởi các thuộc tính trong dữ liệu đầu vào sẽ tăng theo cấp số nhân với kích thước của bản ghi.

Giải pháp một phần là chỉ lưu trữ những kết hợp thực sự xuất hiện trong tập dữ liệu. Một cơ chế nữa là sử dụng *“downward closure property*” hay còn gọi là *tính chất phản đơn điệu* của tập bản ghi - “nếu bất kỳ tập item I nào không phổ biến, thì bất kỳ tập con nào của I cũng không phổ biến”. Điều này có thể được sử dụng hiệu quả để tránh việc tạo và tính độ hỗ trợ cho tất cả các kết hợp trong dữ liệu đầu vào. Tuy nhiên, cách tiếp cận này đòi hỏi:

1) Một số đường đi của tập dữ liệu

2) Việc xây dựng các tập ứng viên được tính trong lần vượt qua tiếp theo.

Thuật toán ARM nổi tiếng nhất sử dụng *tính chất phản đơn điệu* (*downward closure property*) là thuật toán Apriori của Agrawal và Srikant [8]. Agrawal và Srikant sử dụng cấu trúc dữ liệu *hash-tree*, tuy nhiên Apriori có thể được thực hiện tốt bằng cách sử dụng các cấu trúc thay thế như các cây điều tra [12]. Thiết lập các cây điều tra đặt ra một thứ tự trên các item và sau đó liệt kê các tập hợp theo thứ tự này.

***Ví dụ 4:*** Cho tập dữ liệu mẫu như sau:

|  |  |
| --- | --- |
| Số dòng | Ví trí cột |
| 1 | 1 3 4 |
| 2 | 2 4 5 |
| 3 | 2 4 6 |

**Bảng 3. Tập dữ liệu ví dụ 4**

Tập dữ liệu trên bao gồm ba bản ghi với sự kết hợp của sáu mục: {1; 3; 4}, {2; 4; 5} và {2; 4; 6} (và ngưỡng hỗ trợ rất thấp), sau đó cây sẽ bao gồm một nút cho mỗi tập phổ biến I (với số hỗ trợ của nó). Bậc thứ nhất của cây ghi lại sự hỗ trợ cho 1-itemsets, bậc thứ hai cho 2-itemsets, và tiếp tục như vậy.

1

2

3

4

5

6

1

1

1

2

3

2

4

2

2

4

2

1 2 1 3 1 1

1 1 2 1 1 1 1 1

1 1 1

a) Cây hỗ trợ toàn phần (T-tree) được lưu theo thứ tự ngược lại

3

1

1

1

1

1

2

1

null

2

null

2

null

1

null

1

null

1

null

1

null

1

null

1

null

1

null

**null**

**null**

**null null null**

**null null null null**

**null null**

**null null**

**null null null**

**null null null**

**0 1 2 3 4 5 6**

**0 1 2 3**

**0 1 2 3**

**0 1 2 3 3**

**0 1 2**

**0 1 2**

**0 1 2 3 4**

**0 1 2 3 4 5**

b) biểu diễn bên trong của T-treebằng cách sử dụng bảng chỉ mục (mảng)

1, 3, 4

4,6

4, 5

2

1, 3, 4

1, 3, 4

2, 4, 5

i) thêm bản ghi {1,3,4}

ii) thêm bản ghi {2,4,5}

ii) thêm bản ghi {2,4,6}

1

1 1 1

1 2 1

1 1 1

c) Tạo liên kết cây hỗ trợ một phần (P-tree)

**Hình 5. Cây T-tree (để dễ hiểu, các item/thuộc tính được liệt kê bắt đầu từ 1)**

Việc thực hiện cấu trúc này có thể được tối ưu hóa bằng cách lưu trữ các mức trong cây ở dạng mảng, do đó giảm số lượng các liên kết cần thiết và cung cấp trực tiếp chỉ mục. Với mục đích thứ hai, thuận tiện hơn khi xây dựng phiên bản cây "đảo ngược", như trong Hình 5a. Giải thuật đề cập đến hình thức của cây điều tra thiết lập nén như một cây T (Tổng cây hỗ trợ). Việc thực hiện cấu trúc này được minh họa trong Hình 5b, trong đó mỗi nút trong cây T là một đối tượng (TtreeNode) bao gồm một giá trị hỗ trợ (sup) và một tham chiếu (chldRef) đến một mảng các nút T-tree con.

Thuật toán T-tree Apriori được trình bày như dưới đây, điểm bắt đầu là tham chiếu đến sự bắt đầu của mảng cấp đầu tiên.

R là tập dữ liệu đầu vào, N số thuộc tính (cột), D số lượng bản ghi và K một mức trong cây T (biến boolean NewLevel là một trường trong lớp được khởi tạo là false).

Phương thức **TtreeNode()** là một hàm tạo để xây dựng một đối tượng TtreeNode mới, mô tả như sau [6]:

***Tạo cây T-tree cấp độ 1:***

*Hàm tạo cấp độ 1: createTtreeTopLevel(){*

*Duyệt một vòng lần lượt thuộc tính của mỗi đối tượng, khởi tạo mảng start[]:*

*start[i] = new* ***TtreeNode****();*

*Đếm số thuộc tính có giá trị tương đồng:*

*start[sj].sup ++;*

*}*

***Tạo cây T-tree với cấp độ 2,3… đồng thời áp dụng cắt tỉa***

*Hàm tạo cây T-tree: createTtree(){*

*Tạo cấp độ 1 của cây;*

*Cắt tỉa mảng start;*

*Phương pháp cắt tỉa cây dựa trên việc loại bỏ nút có giá trị hỗ trợ (support) nhỏ hơn ngưỡng hỗ trợ*

*Sinh ra các nút ở cấp độ N:* ***genLevelN****(start, 1, 1, null) bằng việc thêm nút mới vào để ghép cặp*

*K=2;*

*while (isNewLevel){*

*Thêm giá trị hỗ trợ cho nút*

*Cắt tỉa các nút ở cấp độ mới sinh ra*

*isNewLevel = false;*

***genLevelN****(start,1,K,{});*

*K++*

*}*

### 2.3.3 Cây hỗ trợ một phần P-Tree

Một bất lợi của Apriori là các nút tương tự bị lặp lại nhiều lần. Trong phần này sẽ giới thiệu khái niệm đếm hỗ trợ từng phần bằng cách sử dụng cây hỗ trợ một phần “P-tree” [6]. Cây P-tree sao chép dữ liệu đầu vào (trong một đường đi) vào một cấu trúc dữ liệu, trong đó duy trì tất cả các thuộc tính có liên quan của đầu vào, và sau đó khai thác cấu trúc này. Về mặt này, P-tree cung cấp hai ưu điểm:

1. Kết hợp các nút trùng lặp với các nút phổ biến, do đó giảm yêu cầu lưu trữ và xử lý.
2. Cho phép đếm một phần hỗ trợ cho các nút riêng lẻ trong cây được lưu trữ hiệu quả khi cây được xây dựng.

Cấu trúc tổng thể của P-tree là một cây cấu trúc nén. Bậc thứ nhất bao gồm các nút (các cá thể của lớp PtNodeTop), mỗi chỉ mục mô tả itemset 1, với tham chiếu con đến các nút P-tree chính (các thể hiện của lớp PtNode). Các thể hiện PtNodeTop bao gồm:

1) Một trường (sup) cho giá trị hỗ trợ

2) Một liên kết (chdRef) đến một đối tượng PtNode.

Các cá thể của lớp PtNode có:

1) Một trường hỗ trợ (sup)

2) Một mảng các số nguyên nhỏ (I) cho các mục mà nút đại diện

3) Liên kết con và họ hàng (chdRef và sibRef) để tiếp tục các nút P-tree.

Để xây dựng một cây P, duyệt qua các bản ghi dữ liệu đầu vào theo từng hàng. Khi hoàn thành, P-tree sẽ chứa tất cả các itemsets là ánh xạ các bản dữ liệu đầu vào. Sup được lưu trữ ở mỗi nút là một tổng hỗ trợ không đầy đủ, bao gồm tổng của các hỗ trợ được lưu trữ trong cây con của nút.

Khi các nút được chèn vào P-tree để duy trì tổ chức tổng thể của cây, lí do vì:

1) tạo ra một nút “giả” đại diện cho một chuỗi con

2) con trỏ “di chuyển” từ nút hiện tại để liên kết anh em của nút mới.

Một ví dụ về việc xây dựng cây P, sử dụng cùng một dữ liệu được trình bày trong Phần 2, được đưa ra trong Hình 5c. K hoàn thành cây bao gồm số lượng đầy đủ cho các khoản mục {2} và đếm từng phần cho các tập dữ liệu {1,2,3}và {4,6}. Vì lý do tính hiệu quả tính toán, cây P trong thực tế được khởi tạo là một tập hợp các mục được biểu diễn dưới dạng một mảng (xem ở trên và hàm createPtree() ở dưới [6]).

***Tạo cây P-tree: createPtree****(){*

*Duyệt một lượt tất cả các đối tượng (ri), đưa vào cây P:*

*Tạo cấp độ 1 của cây P:* ***addToPtreeTopLevel****(ri)*

*}*

***Tạo cấp độ 1: addToPtreeTopLevel****(r){*

*Khởi tạo mảng start, nếu một giá trị chưa tồn tại: start[r0] = new* ***PtNodeTop****();*

*Nếu đã tồn tại, tăng giá trị hỗ trợ: start[r0].sup++;*

*Thêm node r0 vào cây P-tree:* ***addToPtree****(0,start[r0].chdRef,del1(ri),null);*

*}*

### 2.3.4 Giải thuật Apriori-TFP

TFP (Total From Partial) là một giải thuật khai phá luật kết hợp được xây dựng bởi nhóm LUCS-KDD. Đây là giải thuật mở rộng của giải thuật Apriori-T (Aprori Total). Apriori-T được thiết kế để xử lý tập dữ liệu nhị phân đầu vào để xác định các tập thường xuyên và lưu trữ thông tin các tập thường xuyên đầu ra trong cây T-tree. Cây T-tree này được xử lý để xác định các luật kết hợp.

TFP xử lý theo cách tương tự như Apriori-T ngoại trừ việc thay vì xử lý với dữ liệu đầu vào thô trực tiếp, TFP sẽ tiền xử lý dữ liệu và lưu trong cây P-tree. Như vậy TFP có tất cả các chức năng của Apriori-T ngoại trừ việc nó hiệu quả hơn nhiều do kết quả của việc tiền xử lý trước sử dụng cấu trúc dữ liệu P-tree. Lợi thế này đặc biệt hiệu quả lúc xử lý dữ liệu có số lượng nhiều bản ghi trùng lặp hoặc các chuỗi con có đầu hàng đầu trùng lặp.

Sau đây là 3 bước chính để triển khai giải thuật apriori-TFP :

Tạo cây tăng trưởng P-tree

Tạo và Lưu trữ Rule bằng cây tiền tố T-tree

Tiền xử lý dữ liệu

**Hình 6. Các bước triển khai giải thuật apriori TFP**

***Đầu vào*** :

Mảng dữ liệu đầu vào ***dataArray***[rowIndex][colIndex]

Đặt ngưỡng hỗ trợ, ngưỡng tin cậy.

***Đầu ra*** : Tập các luật thỏa mãn ngưỡng hỗ trợ và ngưỡng tin cậy đã đặt

***Bước 1: Tiền xử lý dữ liệu***

* Tạo mảng ***countArr***[maxValue + 1][2] với chiều thứ nhất là giá trị, chiều thứ 2 là tần số xuất hiện của giá trị,để chứa giá trị giảm dần theo tần số xuất hiện của giá trị trong đầu vào với chỉ số bắt đầu từ 1 ( trừ các giá trị output)
* Tạo mảng ***conversionArray***[maxValue + 1][2] với chiều thứ nhất là index của giá trị trong mảng countArr[maxValue + 1], chiều thứ hai là tần số xuất hiện của giá trị.
* Chuyển đổi giá trị của từng hàng theo công thức :

attribute = ***dataArray***[rowIndex][colIndex];

itemSet[colIndex] = (short)***conversionArray***[attribute][0];

sortItemSet(itemSet);

***dataArray***[rowIndex] = itemSet;

Sau chuyển đổi thu được hàng dữ liệu mới theo thứ tự tăng dần index của các giá trị trong hàng theo mảng countArr[maxValue + 1]

***Bước 2: Tạo cây P-tree***

* Duyệt từng hàng dữ liệu và cập nhật support cho cấp độ đầu tiên (cột đầu tiên của mỗi hàng sẽ được lấy làm “top level” cho P-tree lưu ở mảng ***startPtreeRef***[maxValue + 1]) và các node con của nó.
* Sau khi duyệt hết các hàng ta có mảng P-tree hoàn chỉnh với các chuỗi link list bắt đầu từ top level
* Cuối cùng chuyển đổi sang dạng bảng với các mức tổ hợp: 1 node, 2 node, 3 node, .. từ mảng P-tree đã tạo. Mỗi nốt được lưu dưới dạng:

|  |
| --- |
| Tổ hợp node |
| Tổ hợp node cha |
| Support |

**Bảng 4. Cấu trúc 1 node của bảng P-tree**

***Bước 3: Tạo cây T-tree***

* Tạo cây T-tree level 1 bắt đầu từ cấp 1 trong bảng P-tree ở bước 2, cập nhật support cho top level của T-tree dựa vào bảng P-tree từ index bắt đầu từ 1 của bảng P-tree.
* Cắt tỉa những nốt có ngưỡng hỗ trợ bé hơn ngưỡng hỗ trợ đã cài đặt cho cây level 1
* Tạo cây T-tree level N bắt đầu với cấp N trong bảng P-tree, bắt đầu từ cấp N vì tổ hợp N nốt sẽ được tạo dựa vào nốt hiện tại và nốt trước đó. Cập nhật giá trị hỗ trợ cho các nốt theo ngưỡng hỗ trợ trong bảng P-tree ( chỉ cập nhật những nốt trong T-tree mà giá trị nốt không quá giá trị index của T-tree đang xét (N) )
* Cắt tỉa những nốt có ngưỡng hỗ trợ bé hơn ngưỡng hỗ trợ đã cài đặt
* Tạo luật với level N (>= 2) của T-tree , nếu độ tin cậy của luật lớn hơn độ tin cậy đã cài đặt và giá trị trọng số phải lớn hơn ngưỡng đã chọn, và thỏa mãn không có luật nào chung hơn có cùng độ tin cậy thì lưu luật lại.

# CHƯƠNG 3. MÔ HÌNH GIẢI QUYẾT BÀI TOÁN

## 3.1. Mô hình triển khai giải thuật CMAR để chuẩn đoán bệnh và dự báo khả năng nhiễm bệnh

Từ yêu cầu của bài toàn và từ những lý thuyết tìm hiểu được, tôi xin đề xuất mô hình giải quyết bài toán như sau:

Dự báo tỷ lệ mắc bệnh

Xây dựng tập các rule

Phân loại

Tạo cây tăng trưởng P-tree

-> Khai phá các quy tắc (luật)

Dữ liệu

huấn luyện

Lưu trữ Rule bằng cây tiền tố T-tree

Cơ chế cắt tỉa

-Chọn tất cả quy tắc tương quan với đối tượng

-Dựa trên độ phổ biến các lớp dựa trên trọng số  2

Kết quả

Người

bệnh

Giao diện

Chuyên gia

Giao diện

Chuẩn hóa dữ liệu

## 3.2. Các bước thực hiện

**Hình 7. Mô hình triển khai giải thuật CMAR để chẩn đoán bệnh và dự báo khả năng nhiễm bệnh**

Trong bài toán này, tôi sử dụng 3 bộ dữ liệu:

* Bộ dữ liệu bệnh vô sinh nam
* Bộ dữ liệu bệnh suy tuyến giáp

Mô hình đã trình bày gồm có 4 bước chính sau đây:

***Bước 1. Chuẩn hóa dữ liệu***

***Rời rạc hóa dữ liệu :***

Vì một số trường đầu vào có các khoảng giá trị cùng chung một tính chất nhất định nên ta sẽ tiến hành rời rạc hóa theo các nhóm để tăng độ cover của dữ liệu.

**Với bộ dữ liệu vô sinh**, ta sẽ rời rạc hóa theo khoảng cách cho trường tuổi và trường thời gian ngồi. Qua quá trình thực nghiệm tác giả chọn mức rời rạc hóa theo độ tuổi là 3 và mức rời rạc hóa theo thời gian ngồi là 1,6 giờ thì kết quả đạt được là tối ưu nhất cho bộ dữ liệu này.

**Với bộ dữ liệu tuyến giáp**, ta sẽ rời rạc hóa theo khoảng cách cho trường tuổi. Còn các trường chỉ số TSH, T3, TT4, FTI, T4U và TBG ta sẽ rời rạc hóa thành 3 mức với các mức 1: thấp ; 2: bình thường; 3: cao theo các bảng sau đây [19, 20, 21, 22] :

|  |  |  |  |
| --- | --- | --- | --- |
| **Độ tuổi** | **Bình thường** | **Thấp** | **Cao** |
| Mới sinh | 0.7 – 27 mU/L | < 0.7 mU/L | > 28 mU/L |
| 0 - 4 ngày | 1 – 29 mU/L | < 1 mU/L | > 30 mU/L |
| 2 – 20 tuần | 1.7 – 9.1 mU/L | < 1.7 mU/L | > 9.2 mU/L |
| 20 tuần – 18 tuổi | 0.7 – 64 mU/L | < 0.7 mU/L | > 64 mU/L |

**Bảng 5. Mức chỉ số TSH cho trẻ em**

|  |  |  |  |
| --- | --- | --- | --- |
| **Độ tuổi** | **Bình thường** | **Thấp** | **Cao** |
| 18 – 30 tuổi | 0.5 – 4.15 mU/L | < 0.5 mU/L | > 4.5 mU/L |
| 31 – 50 tuổi | 0.5 – 4.15 mU/L | < 0.5 mU/L | > 4.15 mU/L |
| 51 – 70 tuổi | 0.5 – 4.59 mU/L | < 0.5 mU/L | > 4.6 mU/L |
| 71 – 90 tuổi | 0.4 – 5.49 mU/L | < 0.4 mU/L | > 5.5 mU/L |

**Bảng 6. Mức chỉ số TSH cho đàn ông**

|  |  |  |  |
| --- | --- | --- | --- |
| **Độ tuổi** | **Bình thường** | **Thấp** | **Cao** |
| 18 – 29 tuổi | 0.4 – 2.34 mU/L | < 0.4 mU/L | > 4.5 mU/L |
| 30 – 49 tuổi | 0.4 – 4.04 mU/L | < 0.4 mU/L | > 4.1 mU/L |
| 50 – 79 tuổi | 0.46 – 4.68 mU/L | < 0.46 mU/L | > 4.7 mU/L |

**Bảng 7. Mức chỉ số TSH cho phụ nữ**

|  |  |
| --- | --- |
| **Độ tuổi** | **Bình thường** |
| < 2 tuần tuổi | 11.8 – 22.6 mcg/dL ( 152-292 nmol/L) |
| 2 tuần tuổi – 18 tuổi | 6.4 – 13.3 mcg/dL ( 83-172 nmol/L) |
| > 18 tuổi | 5.4 – 11.5 mcg/dL ( 57-148 nmol/L) |

**Bảng 8. Mức chỉ số TT4 theo độ tuổi**

|  |  |
| --- | --- |
| **Loại chỉ số** | **Bình thường** |
| T4U | 0.8 – 1.20 |

**Bảng 9. Mức chỉ số T4U**

|  |  |
| --- | --- |
| **Độ tuổi** | **Bình thường** |
| < 2 tuổi | 3.3-5.2 pg/mL |
| 2 – 12 tuổi | 3.3-4.8 pg/mL |
| 13 – 20 tuổi | 3.0-4.7 pg/mL |
| > 20 tuổi | 2.3 -4.2 pg/mL |

**Bảng 10. Mức chỉ số T3 theo độ tuổi**

|  |  |
| --- | --- |
| **Độ tuổi** | **Bình thường** |
| 1 - 5 tuổi | 5.9-15.0 mcg/dL |
| 6 – 10 tuổi | 6.0-13.9 mcg/dL |
| 11 – 19 tuổi | 5.9-13.2 mcg/dL |
| >= 20 tuổi | 4.8-12.7 mcg/dL |

**Bảng 11. Mức chỉ số FTI theo độ tuổi**

|  |  |
| --- | --- |
| **Độ tuổi** | **Bình thường** |
| 4 – 6 tuổi | 14.8-32.9 mcg/dL |
| 7 - 8 tuổi | 16.3-30.7 mcg/dL |
| 9 - 10 tuổi | 15.8-27.4 mcg/dL |
| 11 tuổi | 15.5-27.4 mcg/dL |
| 12 tuổi | 14.8-26.2 mcg/dL |
| 13 tuổi | 13.8-25.2 mcg/dL |
| 14 tuổi | 12.2-25.2 mcg/dL |
| 15 tuổi | 10.8-23.8 mcg/dL |
| 16 tuổi | 10.0-23.8 mcg/dL |
| 17 tuổi | 8.5 -23.1 mcg/dL |
| Đàn ông > 17 tuổi | 12.7 -25.1 mcg/dL |
| Phụ nữ > 17 tuổi | 13.5 -30.9 mcg/dL |

**Bảng 12. Mức chỉ số TBG theo độ tuổi**

***Chuẩn hóa theo dạng tăng dần:***

Việc chuẩn hóa dữ liệu để sắp xếp dữ liệu theo thứ tự tăng dần giúp cho việc thực thi hiệu quả hơn thuật toán tạo cây T-tree (và cây P-tree).

Giải thuật chuẩn hóa :

|  |
| --- |
| Số bản ghi : N  Số thuộc tính : M  Giá trị cộng thêm cho cột : Count = 1;  For( i:= 0 🡪 M -1){  If ( cột i chứ các giá trị là string) {  Cập nhật giá trị mới cho cột i hàng j = ( thứ tự tăng dần các string của cột) + Count ; (với 0 <= j < N)  If ( cột i có số lượng giá trị khác nhau < 5) {  Cập nhật giá trị mới cho cột i hàng j = (thứ tự tăng dần các giá trị của cột) + Count; (với 0 <= j < N)  Count += số lượng giá trị khác nhau của cột i + 2;  } else If (cột i chứa các giá trị chỉ có 1 chữ số sau dấu phẩy){  Cập nhật giá trị mới cho cột i hàng j = (giá trị cũ) x (10)x(2) + Count; (với 0 <= j < N)  Count += 20;  } else If (cột i chứa các giá trị có 2 chữ số sau dấu phẩy){  Cập nhật giá trị mới cho cột i hàng j = (giá trị cũ)x(100)x2 + Count; (với 0 <= j < N)  Count += 200;  }  } |

***Bước 2. Xây dựng cây P-tree (PartialSupportTree)***

1. Duyệt từng hàng dữ liệu và cập nhật support cho cấp độ đầu tiên (cột đầu tiên của mỗi hàng sẽ được lấy làm “top level” cho P-tree lưu ở mảng ***startPtreeRef***[maxValue + 1]) và các node con của nó.
2. Sau khi duyệt hết các hàng ta có mảng P-tree hoàn chỉnh với các chuỗi link list bắt đầu từ top level
3. Cuối cùng chuyển đổi sang dạng bảng với các mức tổ hợp: 1 node, 2 node, 3 node, .. từ mảng P-tree đã tạo. Mỗi nốt được lưu dưới dạng:

|  |
| --- |
| Tổ hợp node |
| Tổ hợp node cha |
| Support |

**Bảng 13. Cấu trúc 1 node của bảng P-tree**

Cập nhật dữ liệu, cắt bỏ bớt thông tin ở mỗi node trên cây.

Khai phá các rule

Dữ liệu

huấn luyện

Sử dụng thuật toán AprioriTFP thiết lập cây P-tree

Xác định thuộc tính có tần suất xuất hiện lớn hơn hoặc bằng giá trị ngưỡng hỗ trợ, giảm tải những đối tượng có giá trị trùng lặp

**Hình 8. Mô hình xây dựng cây P-tree (PartialSupportTree)**

***Bước 3. Xây dựng cây T-tree (TotalSupportTree) và lưu trữ quy tắc trong T-tree***

1. Bắt đầu với một T-tree trống.
2. Đi qua từng hàng của bảng P-tree theo cấp độ, bắt đầu với cấp 1, cập nhật cấp cao nhất của cây T theo tính chất của mỗi bản ghi trong bảng P-tree. Như sau:
3. Bản ghi đầu tiên biểu thị nút {1} và do đó cập nhật mức hỗ trợ cho phần tử 1 của hàng đầu tiên của cây T.
4. Bản ghi thứ hai trong bảng P-tree được coi là nút {2} và do đó cập nhật phần tử hỗ trợ 2 ở hàng đầu tiên của cây T -> sự hỗ trợ cho nút {2} là 2.
5. Chuyển đến hàng thứ hai trong bảng P-tree (chỉ số thứ 2) đại diện cho tập 2 nút.
6. Tiếp tục chuyển đến hàng thứ 3,4 cho đến khi hàng cuối trong bảng P-tree và tăng mức hỗ trợ cho các nút

Khi đã hoàn thành một vòng của cây P và chúng ta có thể loại bỏ các nút đơn từ bảng P-tree, cắt bỏ các nút không được hỗ trợ đầy đủ trên cây T, và tạo cấp độ T-tree tiếp theo (cấp độ 2, cấp độ 3).

1. Cấp độ hai của T-tree, bắt đầu với nhóm giá trị thứ 2 đặt trong bảng P-tree:
2. Tiếp tục tìm kiếm nhánh con của mỗi nút, cập nhật lại cây nếu nhánh con không còn tồn tại trong cấp 1 của T-tree [6].

Quy tắc cắt tỉa:

1. Cho 2 luật R1 và R2, R1 được quy ước có xếp hạng cao hơn R2 (R1>R2) khi và chỉ khi:
2. conf(R1) > conf(R2)
3. conf(R1) = conf(R2) nhưng sup(R1) > sup(R2)
4. conf(R1) = conf(R2) nhưng R1 có ít giá trị thuộc tính mà nó bao phủ hơn R2
5. Dựa trên quy tắc này, cho 2 luật R1 và R2, CMAR loại bỏ R2 nếu R1 xếp hạng cao hơn R2.
6. Chỉ lựa chọn luật có độ tương quan hơn, so sánh với giá trị trọng số **.**
7. Lựa chọn rule dựa trên dữ liệu mà nó bao phủ được.

|  |
| --- |
| ***Tạo cây T-tree cấp độ 1:***  *Hàm tạo cấp độ 1: createTtreeTopLevel(){*  Duyệt một vòng lần lượt thuộc tính của mỗi đối tượng, khởi tạo mảng start[]:  start[i] = new **TtreeNode**();  Đếm số thuộc tính có giá trị tương đồng:  start[sj].sup ++;  }  ***Tạo cây T-tree với cấp độ 2,3… đồng thời áp dụng cắt tỉa***  *Hàm tạo cây T-tree: createTtree(){*  Tạo cấp độ 1 của cây;  Cắt tỉa mảng start;  *Cắt tỉa cây dựa trên việc loại bỏ nút có giá trị hỗ trợ (support) nhỏ hơn ngưỡng hỗ trợ*  Sinh ra các nút ở cấp độ N: **genLevelN**(start, 1, 1, null) *bằng việc thêm nút mới vào để ghép cặp*  K=2;  while (isNewLevel){  Thêm giá trị hỗ trợ cho nút  Cắt tỉa các nút ở cấp độ mới sinh ra  isNewLevel = false;  **genLevelN**(start,1,K,{});  K++  } |

Cây T-tree cấp độ 2

Cập nhật các nút trên P-tree bằng việc cắt tỉa nút không xuất hiện trên hàng 1, mức hỗ trợ thấp

Cây T-tree cấp độ 1

Cây P-tree

**Hình 9. Mô hình xây dựng cây T-tree (TotalSupportTree)**

***Bước 4. Phân loại dựa trên bộ quy tắc***

Sau khi có một tập các quy tắc được lựa chọn cho phân loại. CMAR lựa chọn các quy tắc phù hợp với một đối tượng cần phân loại.

*Có 2 trường hợp:*

1. Nếu tất cả các quy tắc phù hợp với đối tượng mới (đang cần phân loại) khi có cùng nhãn lớp thì nhãn lớp đó được gán cho đối tượng cần phân loại.
2. Nếu các quy tắc không cùng một nhãn lớp thì chia nhãn lớp thành các nhóm cùng nhãn với nhau, CMAR so sánh hiệu quả và hiệu suất của nhóm để tìm nhóm tin tưởng nhất.
3. Mỗi luật tính giá trị trọng số của nó, giá trị lớn nhất tương ứng với luật tốt nhất
4. Giá trị được tính toán như sau:

Với mỗi một luật P->c, *sup(c)* là số lượng các đối tượng có nhãn là c, và |T| là số lượng đối tượng trong tập dữ liệu huấn luyện.

Trong đó:

Sau đó, đối với mỗi nhóm luật, giá trị của nhóm được định nghĩa là .

# CHƯƠNG 4. TRIỂN KHAI CHƯƠNG TRÌNH

## 4.1. Bộ dữ liệu cài đặt thử nghiệm

Mô hình để xuất thử nghiệm trên bộ dữ liệu bệnh vô sinh nam giới và bộ dữ liệu tuyến giáp [3, 17].

Các thuộc tính trong bộ dữ liệu bệnh vô sinh nam giới (Bảng 14).

|  |  |  |
| --- | --- | --- |
|  | Mô tả các thuộc tính | Giá trị |
| Input | Tuổi | 18-36 |
| Tiền sử mắc các bệnh thủy đậu, bại liệt | 1: có  2: không |
| Tai nạn nghiêm trọng | 1: có  2: không |
| Can thiệp phẫu thuật | 1: có  2: không |
| Sốt cao trong năm ngoái | 1: nhỏ hơn 3 tháng  2: lớn hơn 3 tháng  3: không sốt cao trong năm ngoái |
| Tần suất tiêu thụ rượu | 1: nhiều lần trong ngày  2: mỗi ngày  3: vài lần một tuần  4: một lần một tuần  5: hầu như không bao giờ và không bao giờ |
| Thói quen hút thuốc | 1: không bao giờ  2: thỉnh thoảng  3: hàng ngày |
| Số giờ ngồi mỗi ngày | Tối đa 16 giờ |
| Output | Tình trạng bệnh | 0: mắc bệnh  1: bình thường |

**Bảng 14. Các thuộc tính trong bộ dữ liệu bệnh vô sinh nam**

Các thuộc tính trong bộ dữ liệu bệnh tuyến giáp ( Bảng 15)

|  |  |  |
| --- | --- | --- |
|  | Mô tả các thuộc tính | Giá trị |
| Input | Tuổi | Số nguyên |
| Giới tính | F: Nữ  M: Nam |
| Hormone thyroxine | t : có  f : không |
| Sử dụng thyroxine | t : có sử dụng  f : không sử dụng |
| Thuốc ức chế tuyến giáp | t : có sử dụng  f : không sử dụng |
| Phẫu thuật tuyến giáp | t : có  f : không |
| Tiền sử bị suy tuyến giáp | t : có  f : không |
| Tiền sử bị cường giáp | t : có  f : không |
| Mang thai | t : có  f : không |
| Bệnh tật | t : có  f : không |
| Khối u | t : có  f : không |
| Tâm thần | t : bị  f : không bị |
| Bướu cổ | t : bị  f : không bị |
| Chỉ số TSH | y : có  n : không |
| TSH | Số thực |
| Chỉ số T3 | y : có  n : không |
| T3 | Số thực |
| Chỉ số TT4 | y : có  n : không |
| TT4 | Số thực |
| Chỉ số T4U | y : có  n : không |
| T4U | Số thực |
| Chỉ số FTI (Free Thyroxine Index) | y : có  n : không |
| FTI | Số thực |
| Chỉ số TBG (Thyroxine Binding Globulin) | y : có  n : không |
| TBG | Số thực |
| Output | Tình trạng bệnh | hypothyroid: mắc bệnh tuyến giáp  negative: âm tính |

**Bảng 15. Các thuộc tính trong bộ dữ liệu bệnh tuyến giáp**

## 4.2 Phân tích chức năng

Hệ thống gồm những chức năng như sau:

Chương trình chính

Chuẩn hoá dữ liệu

Khai phá dữ liệu

Chẩn đoán bệnh

Thiết lập cây P-Tree, khai phá bộ luật

Xây dựng cây T-tree, cắt tỉa dữ liệu, lưu trữ các luật

Rời rạc hóa và Chuấn hoá dữ liệu về thứ tự tăng dần

Chia dữ liệu thành 2 phần training và testing

Khởi tạo tham số phục vụ tạo cơ sở để tiến hành giải thuật sinh cây P-tree

**Hình 10. Biểu đồ phân rã chức năng chương trình**

**a. Chuẩn hóa dữ liệu  
*Chuẩn hóa dữ liệu***

Chuẩn hóa dữ liệu được mô tả ở bước 1 của phần 3.2✓ Đầu vào: Dữ liệu thô.  
✓ Chức năng: Rời rạc hóa dữ liệu và chuyển đổi giá trị các cột sang miền số nguyên sao cho giá trị cột sau luôn lớn hơn giá trị cột trước.  
✓ Đầu ra: Dữ liệu đã được chuẩn hóa.  
***Chia dữ liệu thành 2 phần training và testing***✓ Đầu vào: Dữ liệu đã chuẩn hóa.  
✓ Chức năng: Chia dữ liệu đã cho thành 2 phần : training và testing. Phần  
training dùng để huấn luyện, phần testing dùng để kiểm tra. Thông thường tỷ  
lệ phân chia là 70-30.  
✓ Đầu ra: 2 tập dữ liệu training và testing.  
***Khởi tạo tham số phục vụ tạo cơ sở để tiến hành giải thuật sinh cây P-tree***✓ Đầu vào: Nhập vào các thông số như: ngưỡng hỗ trợ, ngưỡng tin cậy

✓ Chức năng: Nhận những tham số do người dùng nhập vào để chuẩn bị cho  
việc huấn luyện.  
✓ Đầu ra: Rỗng.

**b. Khai phá dữ liệu  
*Thiết lập cây P-tree***✓ Đầu vào: tập training.  
✓ Chức năng: xác định giá trị thuộc tính có tần suất xuất hiện ít nhất bằng ngưỡng hỗ trợ  
✓ Đầu ra: bộ quy tắc (luật) được hỗ trợ 1 phần  
***Thiết lập cây T-tree***✓ Đầu vào: dữ liệu lưu trữ trên cây P-tree.  
✓ Chức năng: cắt tỉa  
✓ Đầu ra: cây T-tree (phiên bản rút gọn của P-tree và được mức hỗ trợ đầy đủ)  
***Hiển thị kết quả khai phá dữ liệu***✓ Đầu vào: rỗng.  
✓ Chức năng: hiển thị tập các quy tắc đã khai phá được  
✓ Đầu ra: hiển thị dạng bảng

**c. Chuẩn đoán bệnh**   
✓Chức năng: hiển thị form cho người dùng nhập vào những thông số là nguyên nhân gây bệnh.  
✓Đầu vào: các thông số là nguyên nhân gây bệnh.  
✓Đầu ra: chuẩn đoán mắc bệnh hay không.

# CHƯƠNG 5. CÀI ĐẶT THỬ NGHIỆM CHƯƠNG TRÌNH

## 5.1. Môi trường cài đặt

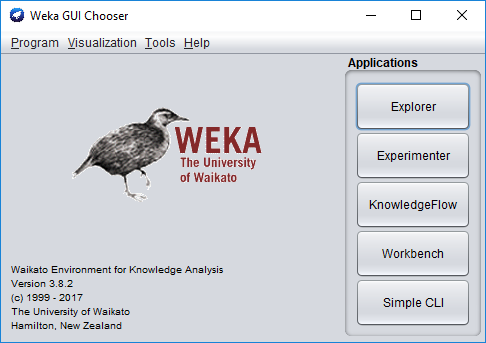
* Phần mền được viết bằng ngôn ngữ C# sử dụng IDE Visual Studio 2017
* Yêu cầu hệ điều hành windows (Windows7 hoặc Windows 10)
* .Net Framework 4.6.1 hoặc cao hơn
* Cài đặt phần mềm Weka [18] ( phiên bản hiện tại là 3.8) để chạy thuật toán cây quyết định (C4.5) và một thuật toán kết hợp (JCBA)

## 5.2 Thử nghiệm chương trình

Sau đây là minh họa chương trình bằng tập dữ liệu vô sinh nam giới :

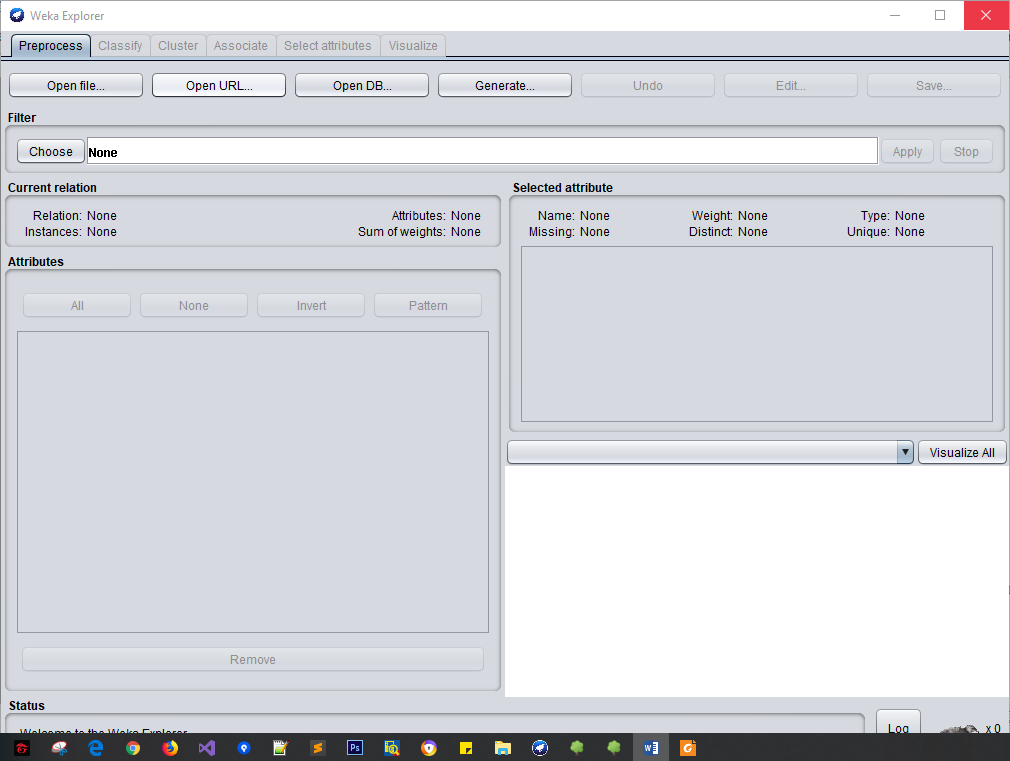
### 5.2.1 Thử nghiệm thuật toán cây quyết định và thuật toán kết hợp JCBA

Trước hết chạy chương trình Weka:

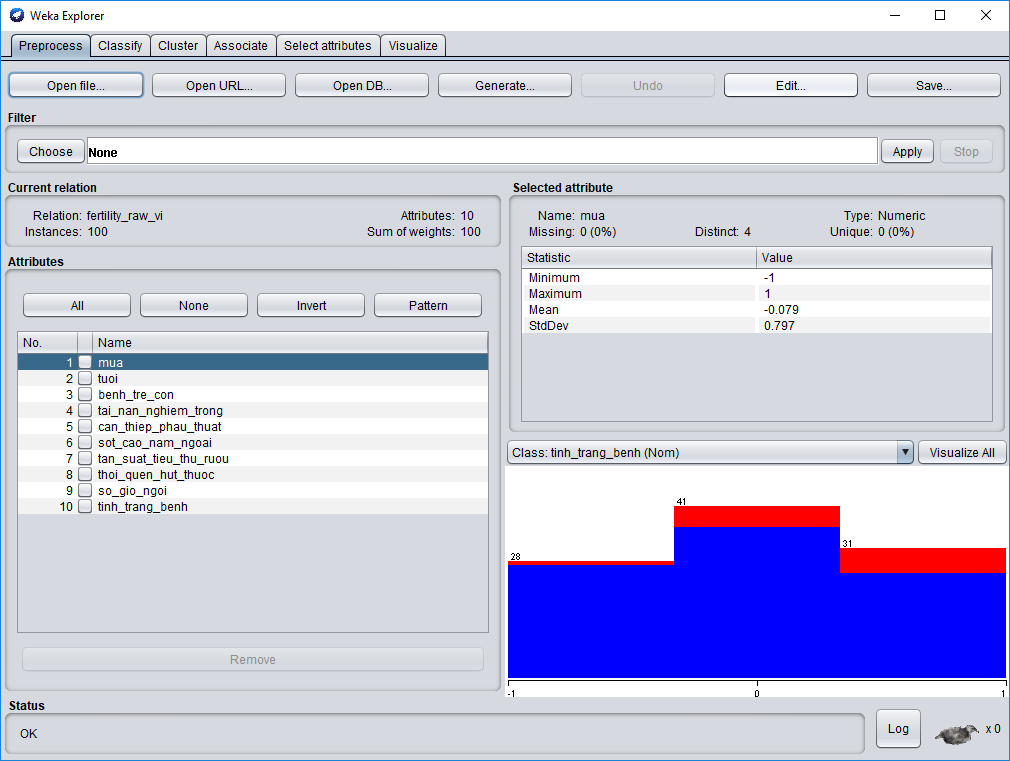


***Hình 11. Giao diện chính chương trình Weka***

Sau đó vào tab “Explorer” open file dữ liệu để chuẩn bị cho bước chạy thử nghiệm :

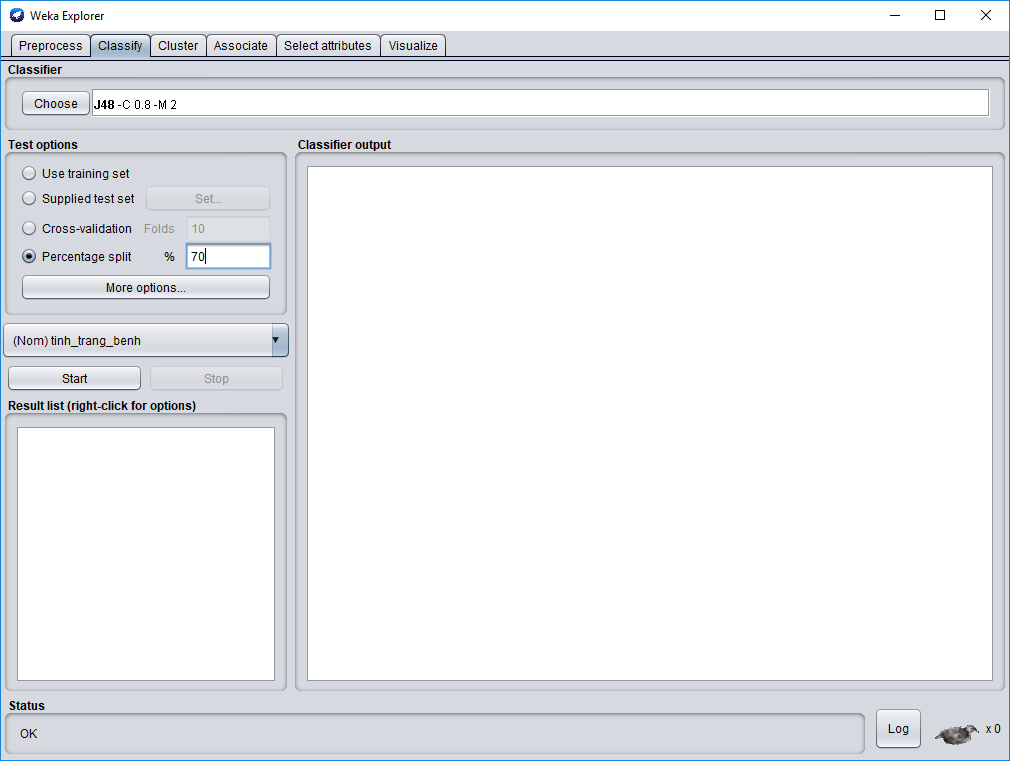


***Hình 12. Giao diện Explorer của chương trình Weka***



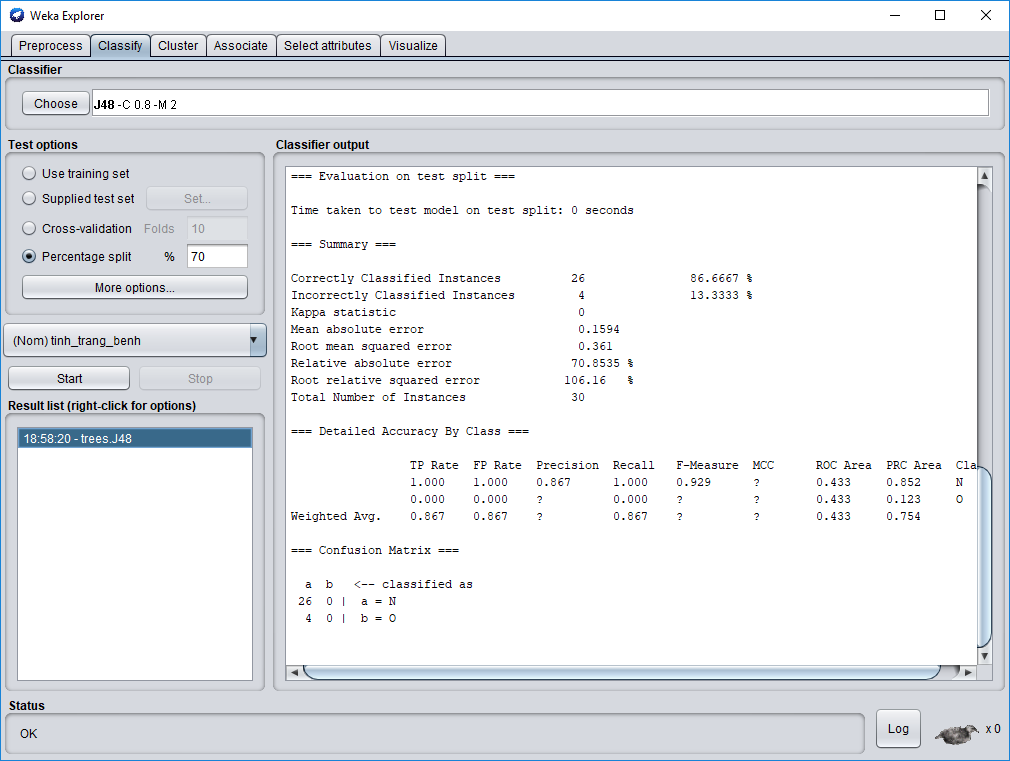
***Hình 13. Mở file dữ liệu và hiện thị trong Weka***

Chuyển sang tab “Classify” và cấu hình chọn thuật toán là J48 ( J48 là triển khai của giải thuật C4.5) , cài đặt 70% dữ liệu là huấn luyện và độ tin cậy là 80%:



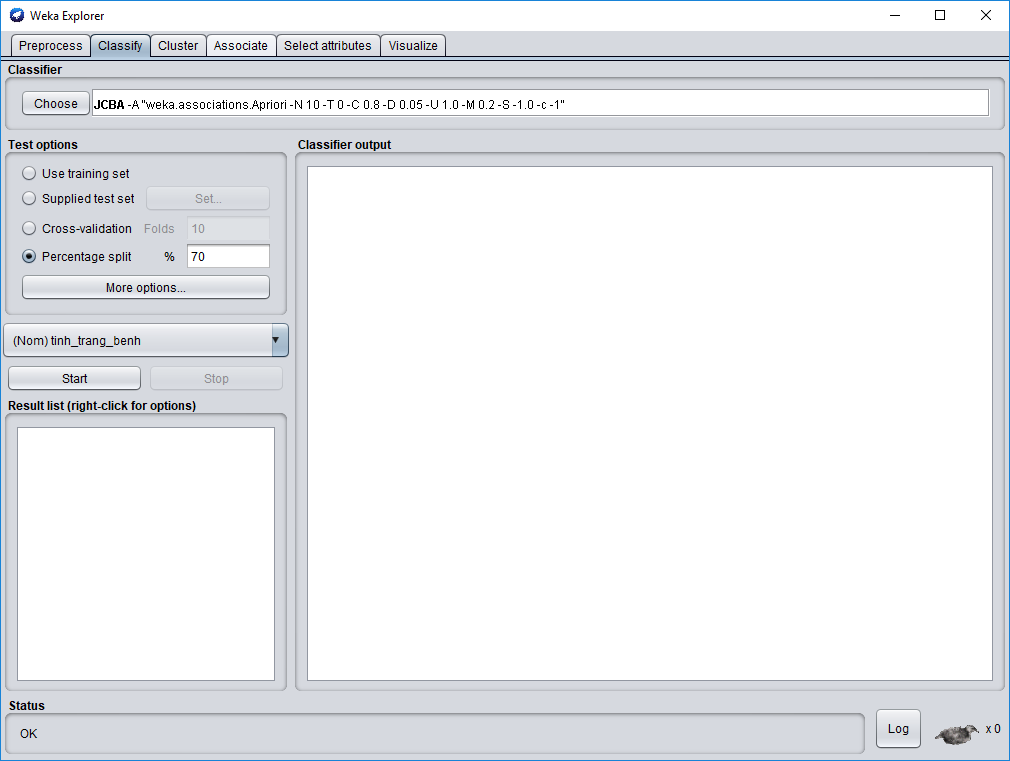
***Hình 14. Cấu hình thuật toán J48 trong Weka***

Nhấn nút “Start” để thực hiện chạy thử với bộ dữ liệu đã tải ở trên với thuật toán C4.5:



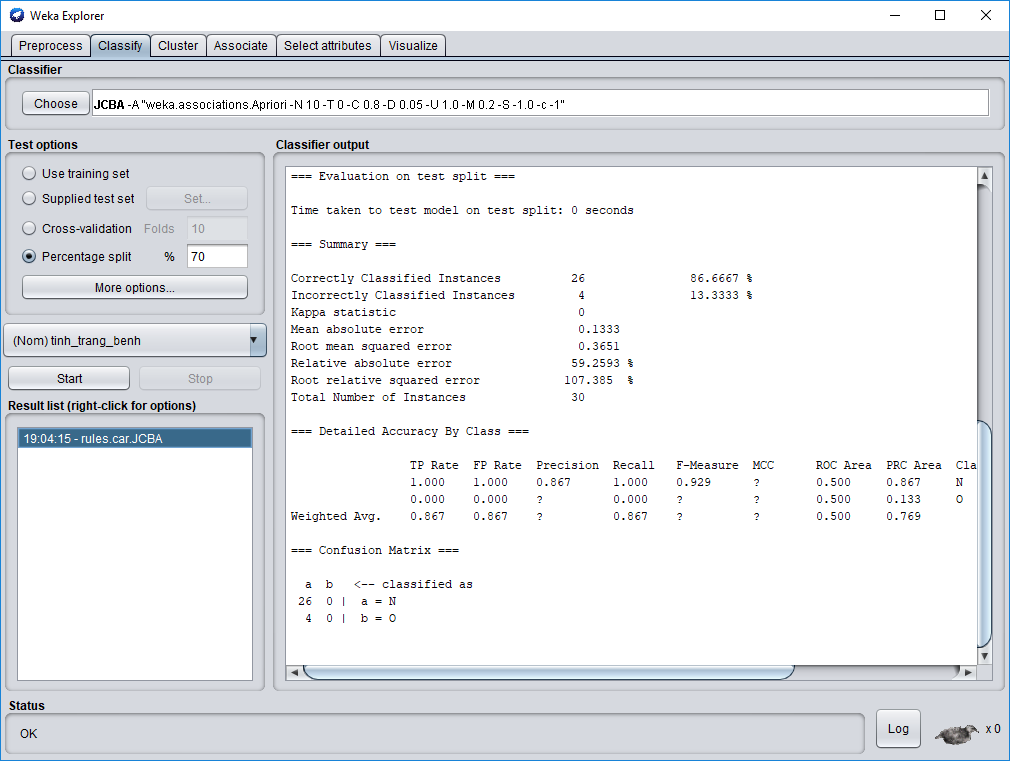
***Hình 15. Kết quả chạy của thuật toán J48 với bộ dữ liệu vô sinh trong Weka***

Tiếp theo, cũng ở tab “Classify” ta cấu hình thuật toán JCBA với độ tin cậy là 80%, độ hỗ trợ là 20% và 70% dữ liệu là huấn luyện :



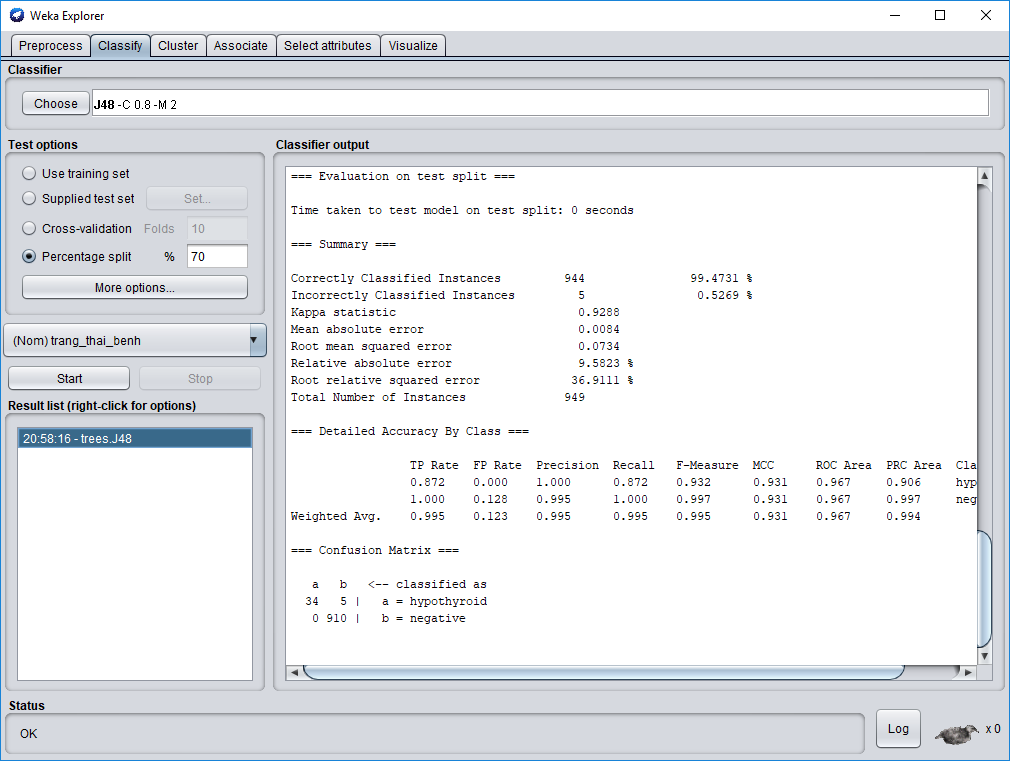
***Hình 16. Cài đặt thuật toán JCBA trong Weka***

Nhấn nút “Start” để thực hiện chạy thử với bộ dữ liệu đã tải ở trên với thuật toán JCBA:

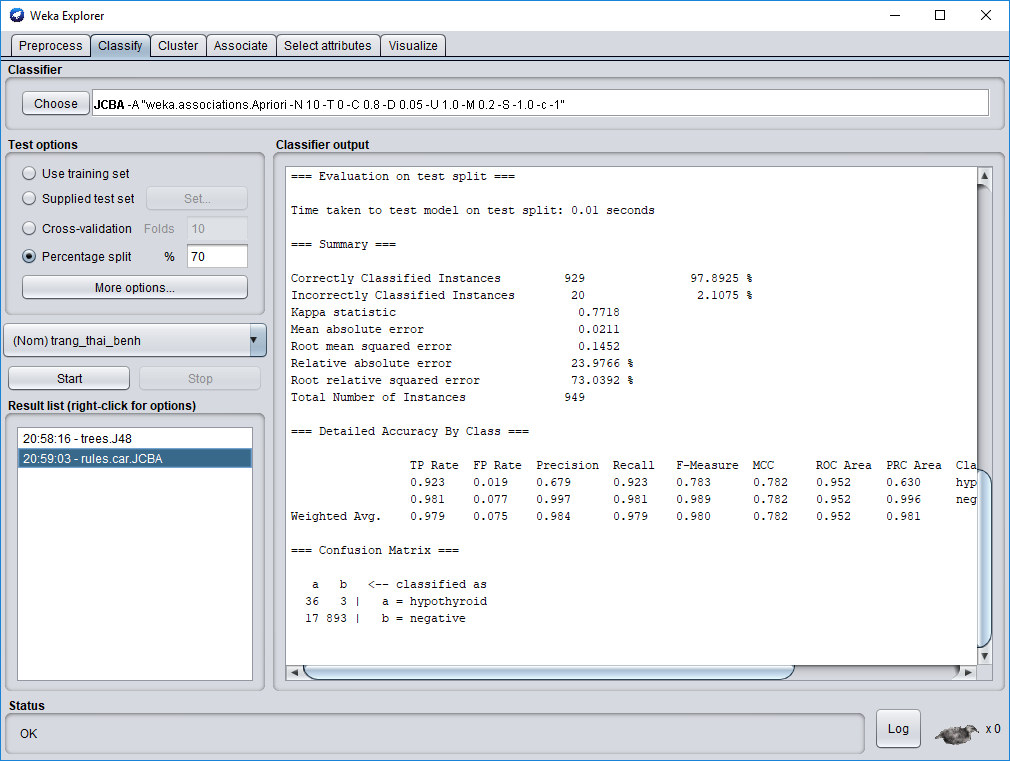


***Hình 17. Kết quả chạy của thuật toán JCBA với bộ dữ liệu vô sinh trong Weka***

Tương tự các bước trên ta có kết quả chạy thực nghiệm của giải thuật J48 (C.45) và giải thuật JCBA đối với tập dữ liệu bệnh tuyến giáp như sau :



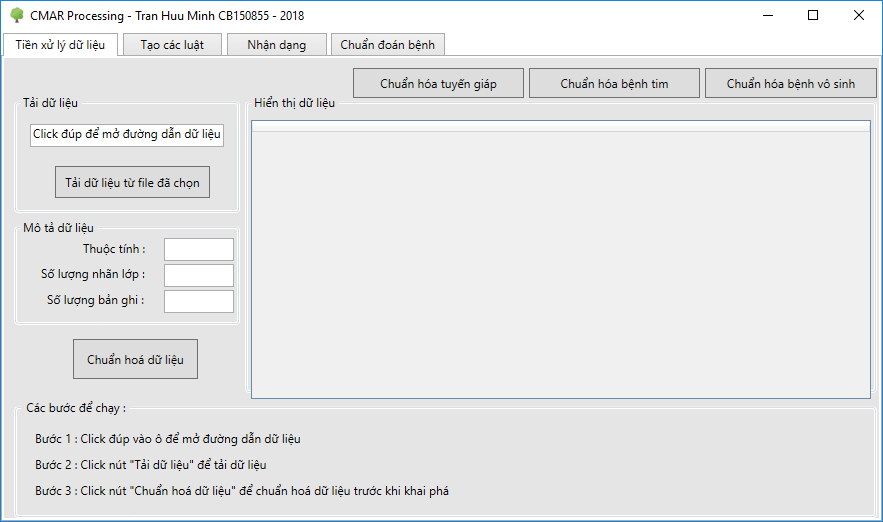
***Hình 18. Kết quả chạy thuật toán J48 với bộ dữ liệu tuyến giáp trong Weka***



***Hình 19. Kết quả chạy thuật toán JCBA với bộ dữ liệu tuyến giáp trong Weka***

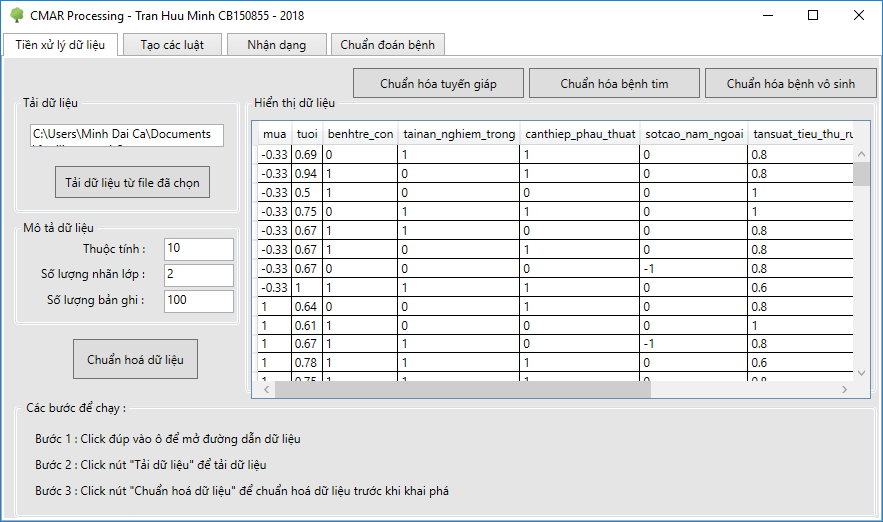
### 5.2.2 Thử nghiệm giải thuật CMAR

Ta mở chương trình CMAR lên:



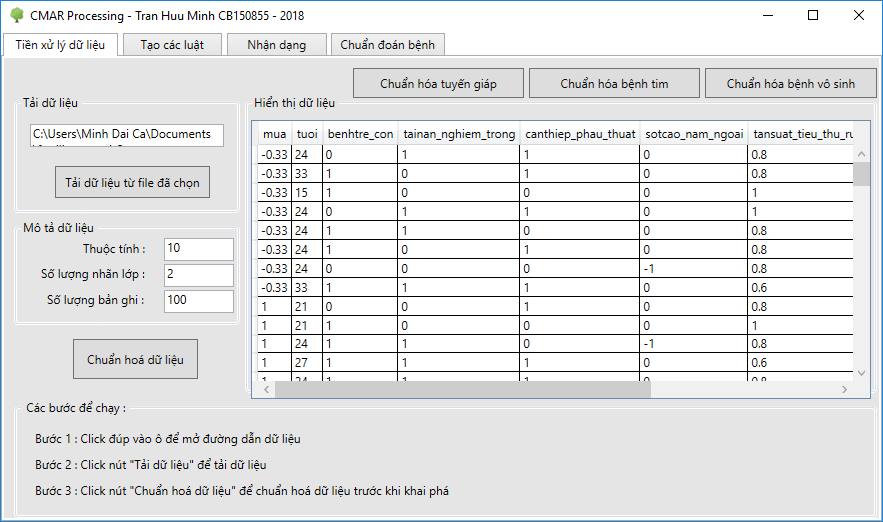
***Hình 20. Giao diện chính chương trình***

Ở Tab “Tiền xử lý dữ liệu”, người dùng sẽ chọn mở file dữ liệu thô từ file csv, các thông số về thuộc tính , số lớp đầu ra và số bản ghi sẽ được tự động cập nhật. Sau đó ấn nút “Tải từ file đã chọn” để load dữ liệu vào ô “Hiển thị dữ liệu”



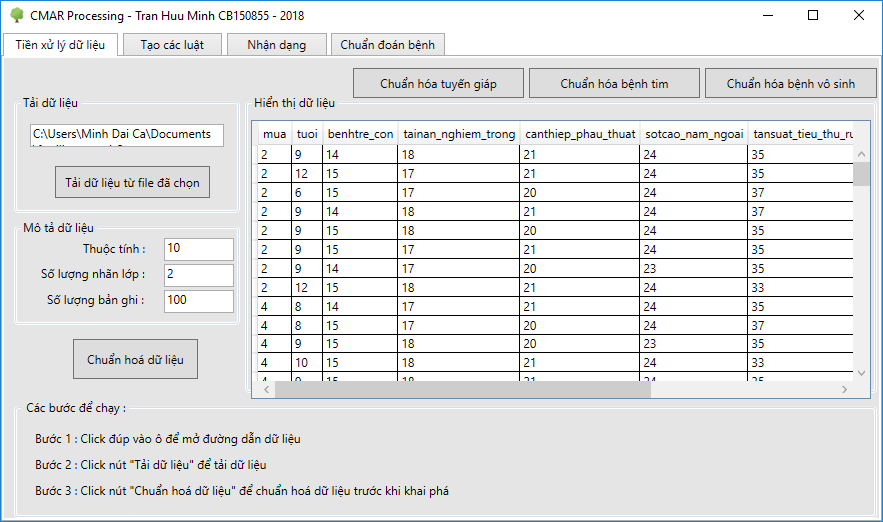
**Hình 21. Giao diện chương trình sau khi load file dữ liệu thô**

Tiếp theo ta rời rạc hóa bộ dữ liệu bệnh vô sinh này bằng cách ấn nút “Chuẩn hóa bệnh vô sinh” :



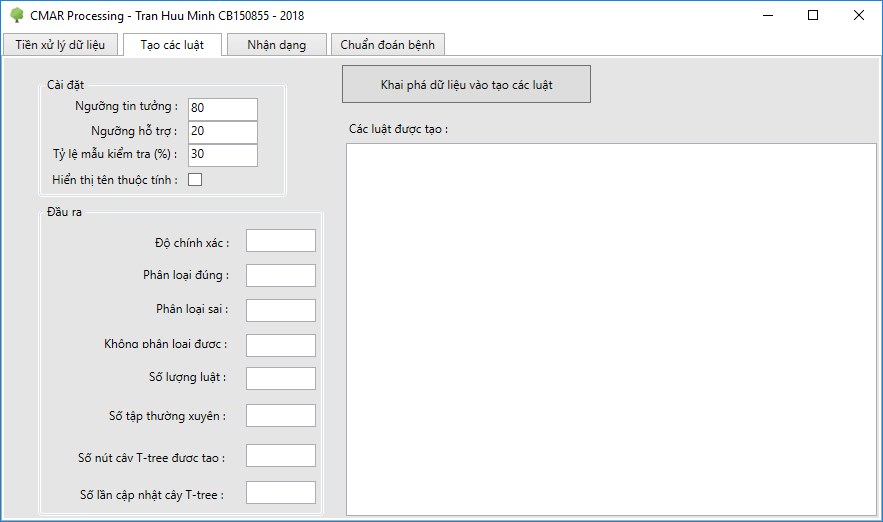
**Hình 22. Giao diện chương trình sau khi rời rạc hóa**

Sau đó, ta chuẩn hóa dữ liệu đầu vào bằng cách ấn nút “Chuẩn hóa dữ liệu” để thực hiện chuẩn hóa. Kết quả như hình sau:



**Hình 23. Giao diện chương trình sau khi chuẩn hóa dữ liệu**

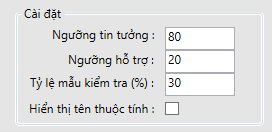
Tiếp theo, ta sang tab “Tạo các luật “ để thực hiện tạo rule và đánh giá kết quả.



**Hình 24. Giao diện chương trình ở tab “Tạo các luật”**

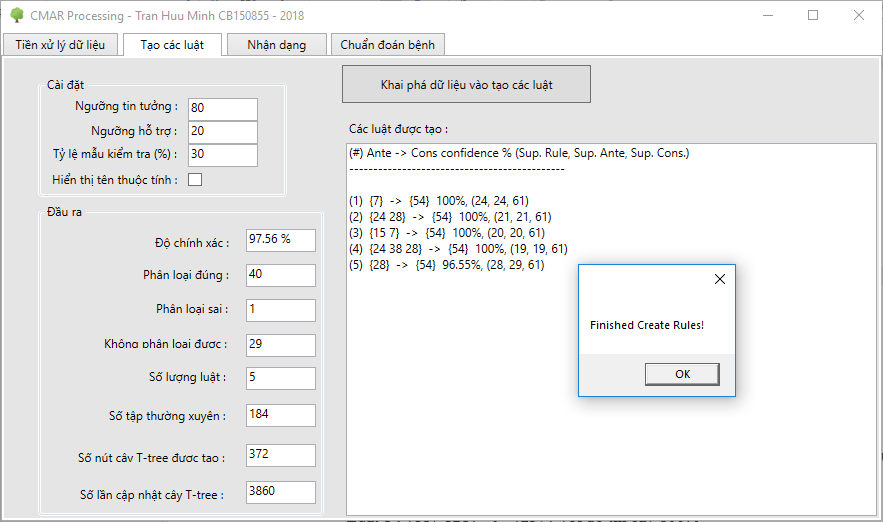
Ở tab này, ta sẽ cài đặt ngưỡng tin tưởng, ngưỡng hỗ trợ, tỉ lệ mẫu test .

Để hiển thị tên thuộc tính ở đầu ra ta có thể tích vào “Hiển thị thuộc tính”.



**Hình 25. Cài đặt ở tab tạo các luật**

Sau khi thiết lập xong cài đặt ta ấn nút “Khai phá dữ liệu và tạo các luật” để thực hiện tạo luật. Kết quả các luật sẽ được hiển thị ở vùng “Các luật được tạo”, cùng với đấy là các tham số đầu ra như độ chính xác, số luật, …

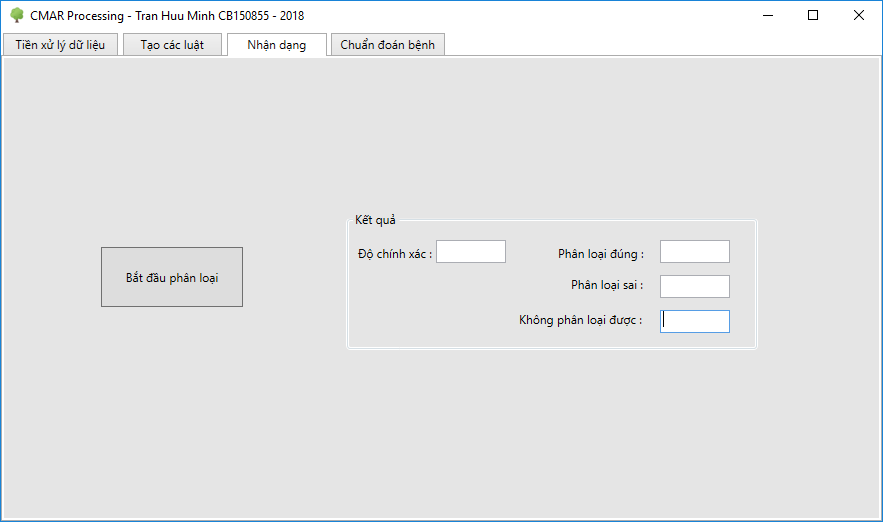


**Hình 26. Kết quả tạo luật của thuật toán CMAR trên dữ liệu bệnh vô sinh**

Ta thu được tập các luật CMAR được tạo. Ví dụ như kết quả trên ta có 5 luật được tạo ra :

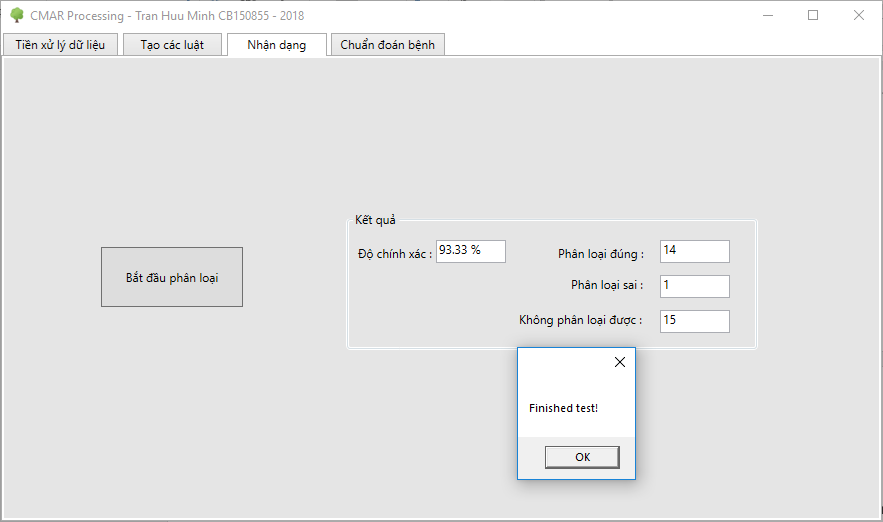
* Luật 1 : {7 } 🡪 {54} với độ tin cậy 100%
* Luật 2 : {24 28 } 🡪 {54} với độ tin cậy 100%
* Luật 3 : {15 7} 🡪 {54} với độ tin cậy 100%
* Luật 4 : {24 38 28} 🡪 {54} với độ tin cậy 100%
* Luật 5 : {28} 🡪 {54} với độ tin cậy 96.55%

Tiếp theo ta sẽ chạy thử để kiểm tra trên 30% dữ liệu test còn lại. Chuyển sang tab “Nhận dạng”.



**Hình 27. Tab nhận dạng để kiểm thử dữ liệu test bệnh vô sinh**

Ấn nút “Bắt đầu phân loại” để thực hiện việc kiểm tra. Kết quả kiểm tra sẽ được hiển thị như sau :

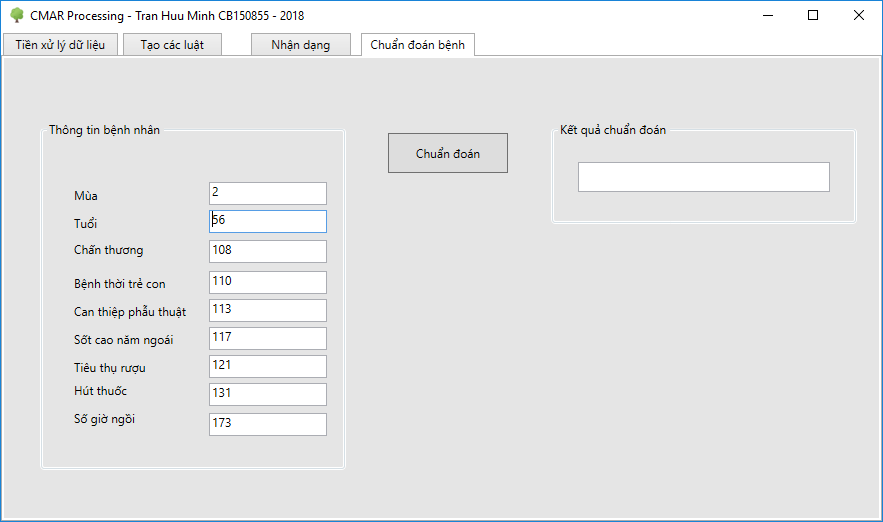


**Hình 28. Kết quả chạy giải thuật CMAR với bộ dữ liệu test bệnh vô sinh**

Số mẫu test là 30, số lớp phân loại đúng là 14, số lớp phân loại sai là 1, số lớp ko được phân loại là 15.

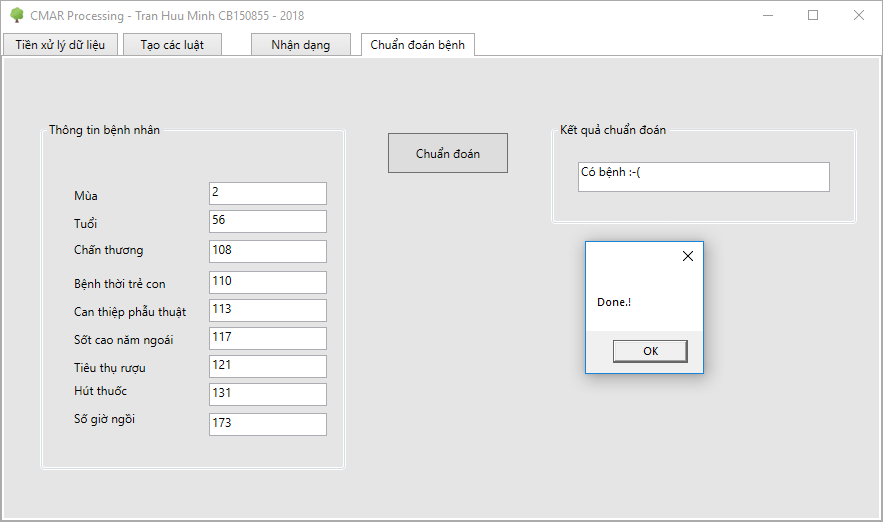
Độ chính xác phân loại của thuật toán là 93.33 %.

Cuối cùng sang tab “Chuẩn đoán bệnh” để thực hiện việc chuẩn đoán cho bệnh nhân.



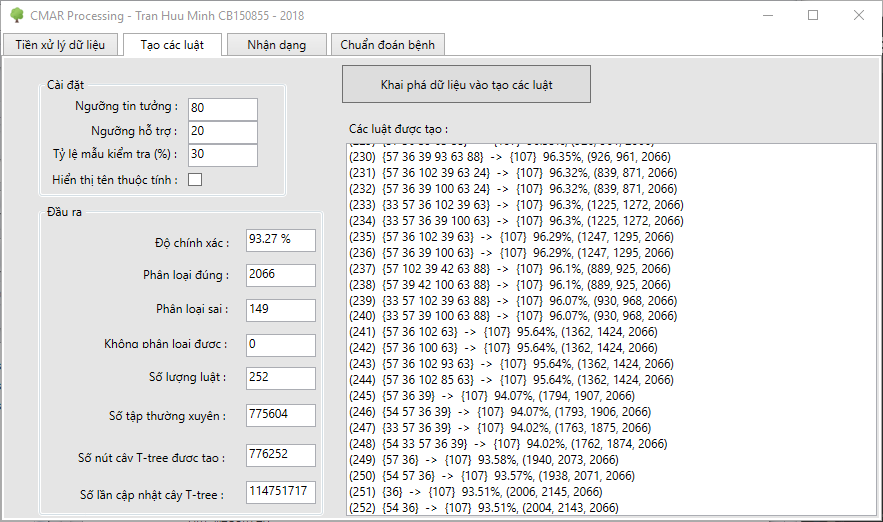
**Hình 29. Giao diện chuẩn đoán bệnh**

Ở tab này ta nhập các thông số của người bệnh ( tham chiếu từ dữ liệu chuẩn hóa). Sau đó chuẩn đoán bằng cách ấn nút “Chuẩn đoán”. Kết quả thu được sẽ là “Có bệnh”, “Bình thường” và “Không thể dự đoán”.

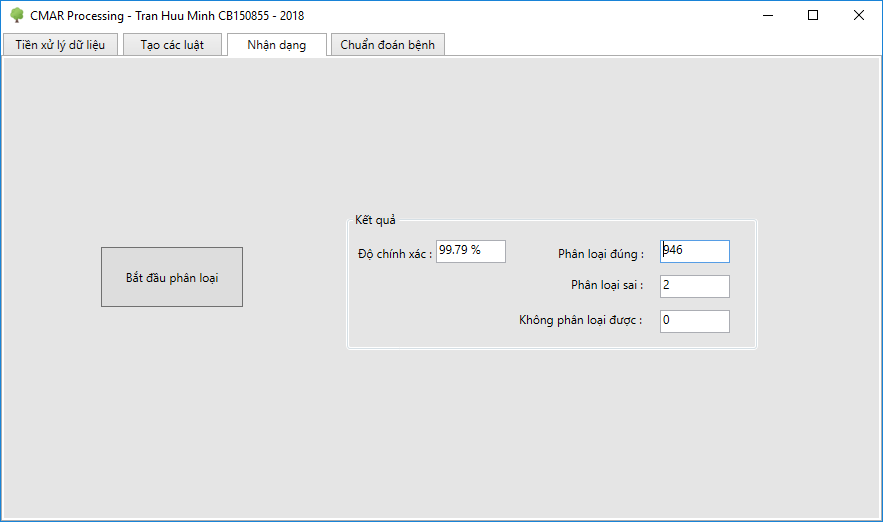


**Hình 30. Kết quả chuẩn đoán bệnh**

Tương tự kết quả khi chạy giải thuật CMAR cho bộ dữ liệu tuyến giáp như sau :



***Hình 31. Kết quả tạo luật của thuật toán CMAR trên dữ liệu bệnh tuyến giáp***



**Hình 32. Kết quả chạy giải thuật CMAR với bộ dữ liệu test bệnh tuyến giáp**

## 5.3. Đánh giá độ chính xác của thuật toán

Kết quả của thuật toán (mô hình đề xuất ) so với với cây quyết định (C4.5) và thuật toán phân loại luật liên kết khác (CBA) :

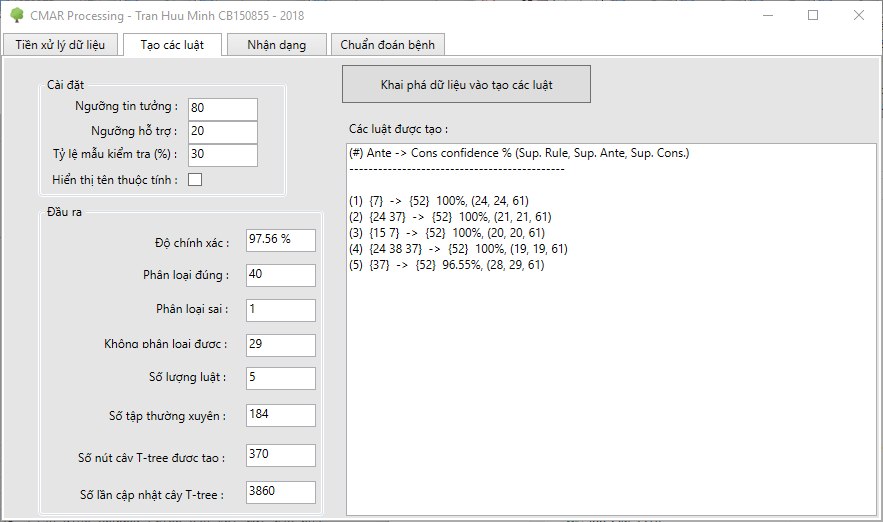
Công thức đo độ chính xác sử dụng:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tập dữ liệu** | **Số thuộc tính** | **Số lớp** | **C4.5** | **CBA** | **CMAR TFP** |
| Vô sinh nam giới | 10 | 2 | 86.67 | 86.67 | 93.33 |
| Bệnh tuyến giáp | 26 | 2 | 99.47 | 97.89 | 99.79 |

**Bảng 16. Kết quả chuẩn đoán bệnh**

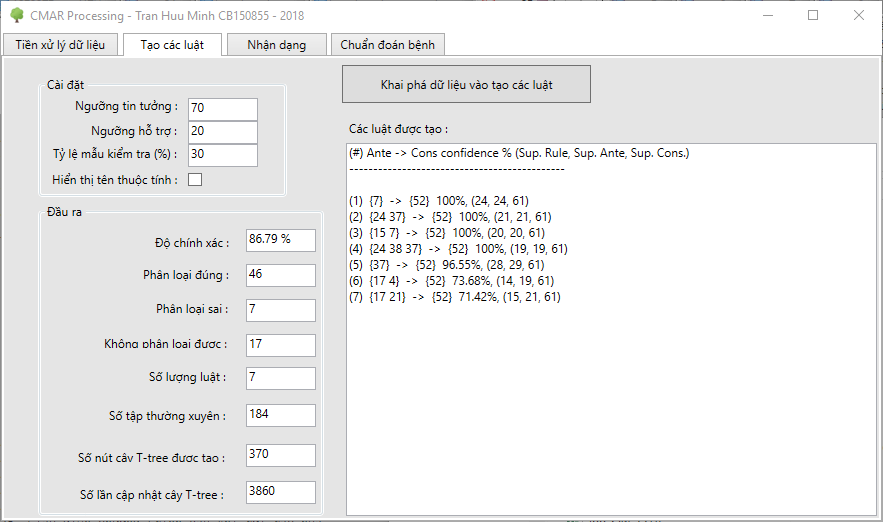
Kết quả, từ các số liệu trên, ta có thể thấy CMAR dùng Apriori-TFP có kết quả là 93,33 % đối với bộ vô sinh nam giới và 99.79% đối với bộ dữ liệu bệnh tuyến giáp, có kết quả tốt hơn các phương pháp phân loại khác về độ chính xác phân loại, nhất là trong nhưng bộ dữ liệu lớn và nhiều thuộc tính.

Ngoài ra, trong quá trình nghiên cứu tôi có nhận xét: đối với bộ dữ liệu nhỏ có thể xuất hiện một số bản ghi không được phân loại. Điều này xảy ra có 2 nguyên nhân: giá trị thuộc tính của đối tượng này chưa từng xuất hiện trong bộ huấn luyện, hoặc do giá trị thuộc tính không phổ biến (không vượt qua được ngưỡng hỗ trợ đặt ra). Và khi dữ liệu cho vào mô hình test không tìm thấy luật nào thỏa mãn nó sẽ được xếp vào loại là “không phân lớp được”. Ta có thể giảm số lượng không phân lớp bằng cách giảm ngưỡng tin cậy hoặc ngưỡng hỗ trợ. Ví dụ như ở bộ dữ liệu bệnh vô sinh, với ngường tin cậy là 80 và ngưỡng hỗ trợ là 20 thì số lượng không phân lớp được là 29.



**Hình 33. Kết quả không phân loại được khi ngưỡng tin tưởng là 80**

Nhưng khi ta giảm ngưỡng tin cậy xuống 70 thì số lượng không phân loại được sẽ là 17.



**Hình 34. Kết quả không phân loại được khi ngưỡng tin tưởng là 70**

Từ đây ta thấy, thuật toán CMAR dùng Apriori-TFP có ưu điểm hơn các thuật toán phân loại thông thường ở chỗ, nó sẽ thông báo cho người dùng biết được số lượng luật hiện có chưa đủ để phân loại dữ liệu theo tiêu chí đã đặt trước.

# CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 6.1 Kết luận

Luận văn đã xây dựng được mô hình phân lớp sử dụng thuật toán CMAR kết hợp nhiều luật phân loại nhằm tạo ra một mô hình phân lớp tốt hơn so với các luật liên kết thông thường. Sau đó áp dụng mô hình phân lớp này để giải quyết bài toán chuẩn đoán bệnh.

Đồ án đã hoàn thành các nhiệm vụ đặt ra khi thực hiện:

* Tìm hiểu về luật liên kết.
* Tìm hiểu thuật toán CMAR: Phân loại dựa trên nhiều luật liên kết.
* Sau đó áp dụng mô hình này để giải quyết bài toán chuẩn đoán bệnh.
* Thử nghiệm đánh giá độ chính xác của mô hình dùng CMAR với một số mô hình phân lớp riêng lẻ

## 6.2. Hướng phát triển

Với các hạn chế của luận văn, tác giả xin đề xuất một số hướng phát triển như sau :

1. Tìm và chạy thử trên nhiều bộ dữ liệu hơn đồng thời so sánh với nhiều phương pháp để tìm ra điểm mạnh và yếu của luật để có ứng dụng hợp lý cho nhiều trường hợp.
2. Tìm hiểu thêm các phương pháp để tối ưu và năng cao chất lượng thuật toán, nhất là phần tiền xử lý dữ liệu.
3. Mở rộng phạm vi ứng dụng phương pháp cho nhiều lĩnh vực khác trong cuộc sống.

# TÀI LIỆU THAM KHẢO

[1]. CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules - Wenmin Li Jiawei Han Jian Pe . Published in: Proceedings 2001 IEEE International Conference on Data Mining , 07 August 2002

[2]. Nguyễn Thị Thùy Linh, "Nghiên cứu các thuật toán phân lớp dữ liệu dựa trên cây quyết định," Đồ án tốt nghiệp, Đại học quốc gia Hà Nội, 2005.

[3]. Blake, C.L. and Merz, C.J. (1998). UCI Repository of machine learning databases. http://www.ics.uci.edu/~mlearn/MLRepository.html, Irvine, CA: University of California, Department of Information and Computer Science.

[4]. B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In KDD’98, New York, NY, Aug. 1998.

[5]. G. Dong, X. Zhang, L. Wong, and J. Li. Caep: Classification by aggregating emerging patterns. In DS’99 (LNCS1721), Japan, Dec. 1999.

[6]. Coenen and Leng (2004). Data Structures for Association Rule Mining: T-trees and P-trees To appear in IEEE Transaction in Knowledge and Data Engineering.

[7]. Han, J., Pei, J. and Yiwen, Y. (2000). Mining Frequent Patterns Without Candidate Generation. Proceedings ACM-SIGMOD International Conference on Management of Data, ACM Press, pp1-12.

[8]. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In VLDB’94, Chile, Sept. 1994.

[9]. B. Lent, A. Swami, and J. Widom. Clustering association rules. In ICDE’97, England, April 1997.

[10]. Liu, B. Hsu, W. and Ma, Y (1998). Integrating Classification and Association Rule Mining. Proceedings KDD-98, New York, 27-31 August. AAAI. pp80-86.

[11]. W. Li. Classification based on multiple association rules. M.Sc. Thesis, Simon Fraser University, April 2001.

[12]. R. Rymon, “Search Through Systematic Set Enumeration,” Proc. Third Int’l Conf. Principles of Knowledge and Reasoning, pp. 539-550, 1992

[13]. R. Duda and P. Hart. Pattern Classification and Scene Analysis. John Wiley & Sons, .1973.

[14]. (2018, October) Blood Pressure Chart. [Online].

https://healthiack.com/blood-pressure-chart

[15]. (2017, February) What should my cholesterol level be at my age? [Online].

https://www.medicalnewstoday.com/articles/315900.php

[16]. (2018) Aerobic Heart Rate Chart. [Online].

https://www.frontrower.com/aerobic-heart-rate-chart.html

[17]. (2013, May) Statlog Project Data Set. [Online]. http://archive.ics.uci.edu/ml/datasets/Statlog+Project

[18]. (2013, May) Data Mining Software in Java. [Online]. <http://www.cs.waikato.ac.nz/ml/weka/>

[19]. (2018, Nov) Test Center, For Physicians & Hospitals. [Online].

<https://www.questdiagnostics.com>

[20]. (2018, Nov) Free Thyroxine Index (FTI), Serum

https://www.mayomedicallaboratories.com/test-catalog/Clinical+and+Interpretive/62583

[21]. James K. Stoller, “The Cleveland Clinic Foundation Intensive Review of Internal Medicine”, sixth edition, 2014.

[22]. (2018, August) All About Standard TSH Ranges by Age and Life Stage [Online].

https://www.healthline.com/health/tsh-normal-range-by-age#women