

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC LẠC HỒNG**

PHẠM VĂN LONG

**KHAI PHÁ DỮ LIỆU THEO TIẾP CẬN TẬP
THỎ
VÀ CÂY QUYẾT ĐỊNH - ỨNG DỤNG TRONG
PHÂN LỚP NĂNG KHIẾU HỌC SINH**

**Chuyên Ngành: CNTT
Mã số: 60.48.02.01**

**LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG
TIN**

**NGƯỜI HƯỚNG DẪN KHOA HỌC
TS. HOÀNG THỊ LAN GIAO**

Đồng Nai, Năm 2012
LỜI CAM ĐOAN

Tôi xin cam đoan những kết quả được trình bày trong luận văn này là của riêng tôi, không sao chép từ bất kỳ một công trình nào khác. Nếu có điều gì không trung thực, tôi xin chịu hoàn toàn trách nhiệm.

Học viên

Phạm văn Long

LỜI CẢM ƠN

Trước tiên, em xin chân thành cảm ơn cô Hoàng Thị Lan Giao, mặc dù rất bận rộn trong công việc nhưng Cô luôn quan tâm giúp đỡ, sự chỉ bảo kịp thời và sự tận tình hướng dẫn em trong việc hoàn thành luận văn này.

Em xin cảm ơn Quý Thầy Cô trong khoa Công nghệ thông tin trường Đại học Lạc Hồng, em xin chân thành cảm ơn Thầy Cô giảng viên vì kiến thức mà Quý Thầy Cô truyền đạt cho em trong suốt quá trình học tập tại trường.

Xin được cảm ơn Sở Giáo dục và đào tạo Đồng Nai đã tạo mọi điều kiện để tôi được đi học và hoàn thành tốt khoá học.

Xin chân thành cảm ơn các anh chị em lớp cao học công nghệ thông tin khoá 2 trường Đại Học Lạc Hồng và các bạn đồng nghiệp đã luôn bên cạnh, động viên, khuyến khích tôi trong suốt thời gian học tập và thực hiện đề tài.

Xin chân thành cảm ơn!

Đồng Nai, ngày 28 tháng 7 năm 2012

Học viên Phạm Văn Long

MỤC LỤC

LỜI CẢM ƠN	i
MỤC LỤC.....	iv
DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT.....	iii
DANH MỤC CÁC BẢNG.....	vii
DANH MỤC CÁC HÌNH VẼ.....	viii
MỞ ĐẦU.....	1
CHƯƠNG 1: KHAI PHÁ DỮ LIỆU THEO TIẾP CẬN TẬP THÔ.....	4
1.1 Giới thiệu.....	4
1.2 Các khái niệm cơ bản.....	4
1.2.1 Hệ thống thông tin.....	4
1.2.2 Bảng quyết định	6
1.2.3 Quan hệ không phân biệt được	7
1.2.4 Xấp xỉ tập hợp trong tập thô	8
1.2.5 Sự phụ thuộc của các thuộc tính	11
1.2.6 Rút gọn các thuộc tính trong hệ thống thông tin.....	12
1.2.7 Ma trận phân biệt	14
1.3 Rút gọn dữ liệu trong hệ thống thông tin	16
1.4 Thuật toán tìm tập rút gọn của một bảng quyết định dựa vào ma trận phân biệt được.....	16
1.5 Tập thô với các công cụ khai phá dữ liệu	21
1.5.1 Khám phá tri thức trong cơ sở dữ liệu	21
1.5.2 Tập thô trong khai phá dữ liệu.....	22
1.5.3 Một số ứng dụng quan trọng của lý thuyết tập thô	23
1.6 Kết luận chương 1	25
CHƯƠNG 2: CÁC PHƯƠNG PHÁP XÂY DỰNG CÂY QUYẾT ĐỊNH	26
2.1 Khai phá dữ liệu với cây quyết định	26
2.1.1 Khái niệm.....	26
2.1.2 Thiết kế cây quyết định.....	26
2.2 Phương pháp tổng quát xây dựng cây quyết định.....	28
2.3. Phương pháp xây dựng cây quyết định ID3.....	30

2.3.1 Tiêu chí lựa chọn thuộc tính để phân lớp.....	30
2.3.2 Thuật toán ID3	31
2.3.3 Độ phức tạp tính toán.....	37
2.4 Phương pháp xây dựng cây quyết định C4.5	38
2.4.1 Giới thiệu	38
2.4.2 Xác định điểm chia tốt nhất	38
2.4.3 Một số vấn đề với thuộc tính	38
2.4.4 Thuật toán C4.5	43
2.5 Phương pháp xây dựng cây quyết định FID3	52
2.5.1 Xác định điểm chia tốt nhất	52
2.5.2. Thuật toán FID3	53
2.6 Kết luận chương 2	58
CHƯƠNG 3: MÔ PHỎNG CHƯƠNG TRÌNH PHÂN LỚP NĂNG KHIẾU HỌC SINH.....	59
3.1. Giới thiệu bài toán.....	59
3.2. Cài đặt ứng dụng	60
3.2.1. Giới thiệu về cơ sở dữ liệu	61
3.2.2 Màn hình giao diện của chương trình	62
3.2.3 Chức năng mở dữ liệu	63
3.2.4 Chức năng tìm tập rút gọn.....	64
3.2.5 Chức năng tạo và hiển thị cây quyết định	65
3.2.6 Chức năng phân lớp năng khiếu học sinh	65
3.2.7 Luật quyết định tương ứng với cơ sở dữ liệu.....	66
3.3. Kết luận chương 3	67
KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	68
Tài liệu tham khảo	
Phụ lục	

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT

CÁC KÝ HIỆU:

$S = (U, A)$	Hệ thống thông tin
V_a	Tập các giá trị của thuộc tính a
$IND(B)$	Quan hệ tương đương của tập thuộc tính B
$[x]_B$	Lớp tương đương chứa x của quan hệ không phân biệt được trên B
$DT=(U,C\cup D)$	Bảng quyết định
$\underline{B}(X)$	B-Xấp xỉ dưới của X
$\overline{B}(X)$	B-xấp xỉ trên của X
$BN_B(X)$	B -biên của tập X
$NEG_B(X)$	B -ngoài của tập
$POS_C(D)$	Miền C -khả định của D
$ POS_C(D) $	Lực lượng của tập $POS_C(D)$
$ U $	Lực lượng của tập U
$ \overline{B}(X) $	Lực lượng của B-xấp xỉ trên của X
$ \underline{B}(X) $	Lực lượng của B-Xấp xỉ dưới của X

CÁC CHỮ VIẾT TẮT

ID3: Iterative Dichotomiser 3

IG: Information Gain

DANH MỤC CÁC BẢNG

Số hiệu bảng	Tên bảng	Trang
1.1	Ví dụ về hệ thông tin	5
1.2	Ví dụ một bảng quyết định	6
1.3	Hệ thông tin minh họa sự phụ thuộc của thuộc tính	12
1.4	Rút gọn các thuộc tính trong hệ thống thông tin	14
1.5	Bảng quyết định minh họa ma trận phân biệt được	15
1.6	Ma trận phân biệt của hệ thông tin trong Bảng 1.4	15
1.7	Bảng quyết định minh họa ví dụ 1.11	19
2.1	Bảng quyết định minh họa Ví dụ 2.1	29
2.2	Bảng quyết định minh họa thuật toán ID3.	34
2.3	Tập dữ liệu có giá trị liên tục	39
2.4	Dữ liệu chứa thuộc tính thiếu giá trị	41
3.1	Danh sách các thuộc tính của bảng điểm tổng hợp	61

DANH MỤC CÁC HÌNH VẼ

Số hiệu	Tên hình vẽ	Trang
1.1	Tập xấp xỉ và miền	9
1.2	Minh họa chạy thuật toán tìm tập rút gọn cho ví dụ trên từ chương trình	22
1.3	Xử lý khám phá tri thức trong cơ sở dữ liệu	22
2.1	Ví dụ cây quyết định ứng với bảng quyết định 2.1	29
2.2	Cây quyết định bước đầu ví dụ 2.	35
2.3	Cây quyết định được xây dựng theo thuật toán ID3 ứng với bảng quyết định 2.2	37
2.4	Minh họa phân chia thuộc tính liên tục	40
2.5	Minh họa phân chia thuộc tính nhiều giá trị	42
2.6	Cây quyết định bước đầu được xây dựng theo thuật toán C4.5 ứng với Bảng quyết định 2.4	48
2.7	Cây quyết định được xây dựng theo thuật toán C4.5 nhánh “Quang cảnh” =Nắng	50
2.8	Cây quyết định được xây dựng theo thuật toán C4.5 ứng với Bảng quyết định 2.4	52
2.9	Cây quyết định bước đầu được xây dựng theo thuật toán FID3 ứng với Bảng quyết định 2.2	56
2.10	Cây quyết định được xây dựng theo thuật toán FID3 ứng với Bảng quyết định 2.2	58
3.1	Minh họa của bảng điểm tổng hợp	62
3.2	Minh họa màn hình giao diện của chương trình	62
3.3	Minh họa màn hình giao diện chức năng mở dữ liệu của chương trình	63
3.4	Minh họa màn hình giao diện chức năng tìm tập rút gọn của chương trình	64
3.5	Minh họa màn hình giao diện chức năng tạo và hiển thị cây quyết định của chương trình	64
3.6	Minh họa màn hình giao diện chức năng phân lớp năng khiếu học sinh của chương trình	65

MỞ ĐẦU

Sự phát triển mạnh mẽ và những tiến bộ vượt bậc của công nghệ thông tin trong thời gian gần đây đã góp phần làm bùng nổ thông tin. Trong giao dịch các thông tin đang dần được số hóa do nhiều tính năng vượt trội mà phương thức này đạt được như là có thể lưu trữ lâu dài, cập nhật, sửa đổi, tìm kiếm một cách nhanh chóng. Đó là lý do khiến cho số lượng thông tin số hóa ngày nay đang tăng dần theo cấp số nhân. Hơn nữa hiện nay trong tất cả các lĩnh vực của đời sống như là kinh doanh, thương mại, y tế, giáo dục, văn hoá, xã hội,...không một lĩnh vực nào lại không cần đến sự hỗ trợ của công nghệ thông tin. Các công cụ thu thập dữ liệu tự động và các công nghệ cơ sở dữ liệu được phát triển dẫn đến vấn đề một lượng dữ liệu khổng lồ được lưu trữ trong cơ sở dữ liệu và trong các kho thông tin của các tổ chức, cá nhân. Việc nắm bắt thông tin một cách nhạy bén, nhanh chóng và hữu ích đã mang lại rất nhiều sự thành công của các lĩnh vực đó. Do vậy việc khai phá tri thức từ dữ liệu trong các tập tài liệu lớn chứa đựng thông tin phục vụ nhu cầu nắm bắt thông tin có vai trò hết sức to lớn và được rất nhiều nhà nghiên cứu và ứng dụng tin học đặc biệt quan tâm. Việc nghiên cứu những phương pháp có thể tự động phát hiện những tri thức mới trong cơ sở dữ liệu trên máy tính đã tỏ ra thực sự hữu ích trong việc hỗ trợ quyết định cho con người.

Hiện nay có rất nhiều thuật toán khai phá tri thức bằng cách phân lớp và rời rạc dữ liệu như: Sử dụng cây quyết định, phương pháp thống kê, các mạng nơ ron, thuật toán di truyền,...Trong những năm gần đây, lý thuyết tập thô được nhiều nhóm nghiên cứu hoạt động trong lĩnh vực tin học nói chung và khai phá tri thức nói riêng nghiên cứu và áp dụng trong thực tế. Lý thuyết tập thô được xây dựng trên nền tảng toán học vững chắc giúp cung cấp những công cụ hữu ích để giải quyết những bài toán phân lớp dữ liệu và khai phá luật,...Với đặc tính có thể xử lý được những dữ liệu mơ hồ, không chắc chắn tập thô tỏ ra rất hữu ích trong việc giải quyết những bài toán thực tế. Từ những bảng dữ liệu lớn với dữ liệu dư thừa, không hoàn hảo, dữ liệu liên tục, hay dữ liệu dưới dạng ký hiệu lý thuyết tập thô cho phép khai phá tri thức từ những khối dữ liệu này nhằm phát hiện những luật tiềm ẩn từ khối dữ liệu này. Một trong những công cụ phân lớp hiệu quả nhất hiện nay là sử dụng cây quyết định. Sử dụng cây quyết định dựa trên Entropy và tập thô thật hiệu quả đối với những tập dữ liệu lớn, dữ liệu đầy đủ, không đầy đủ, không chắc chắn,

dữ liệu liên tục ... Thuật toán xây dựng cây quyết định có những ưu và khuyết điểm riêng việc kết hợp ưu điểm của các phương pháp với nhau để làm tăng hiệu quả cũng đang được quan tâm và phát triển. Vì những lý do trên nên luận văn chọn đề tài “KHAİ PHÁ DỮ LIỆU THEO TIẾP CẬN TẬP THƠ VÀ CÂY QUYẾT ĐỊNH - ỨNG DỤNG TRONG PHÂN LỚP NĂNG KHIẾU HỌC SINH”.

Mục đích nghiên cứu.

Nghiên cứu lý thuyết tập thơ, phương pháp phân lớp cây quyết định theo thuật toán ID3 đưa vào cài đặt chương trình ứng dụng phân lớp năng khiếu học sinh.

Đối tượng và phạm vi nghiên cứu.

Nghiên cứu về cơ sở khai phá dữ liệu dựa trên tiếp cận tập thơ.

Nghiên cứu cơ sở lý thuyết về phương pháp phân lớp dữ liệu và xây dựng cây quyết định ID3 trên hệ thống thông tin đầy đủ.

Phương pháp nghiên cứu

Nghiên cứu lý thuyết, phân tích, tổng hợp, cài đặt, khái quát rút ra những vấn đề cần thiết cho đề tài.

Ý nghĩa khoa học và thực tiễn của đề tài.

Khai phá dữ liệu, là sự khám phá hiệu quả những tri thức từ cơ sở dữ liệu lớn, và nó trở thành một vấn đề nóng cho việc đưa ra những quyết định. Một vấn đề quan trọng và phổ biến trong kỹ thuật khai phá dữ liệu là phân lớp và đã được ứng dụng rộng rãi trong thương mại, y tế, công nghiệp...

Trong những năm trước đây, phương pháp phân lớp đã được đề xuất, nhưng không có phương pháp tiếp cận phân loại nào là cao hơn và chính xác hơn hẳn những phương pháp khác. Tuy nhiên với mỗi phương pháp có một lợi thế và bất lợi riêng khi sử dụng. Vì vậy nó rất dễ hiểu và dễ sử dụng nhưng kết quả thì chưa được thoả đáng.

Phân loại sử dụng lý thuyết tập thơ, đã được nghiên cứu rộng rãi trong những năm gần đây. Lý thuyết tập thơ cung cấp cho nhiều nhà nghiên cứu và phân tích dữ liệu với nhiều kỹ thuật trong khai phá dữ liệu như là các khái niệm đặc trưng bằng cách sử dụng một số dữ kiện. Nhiều nhà nghiên cứu đã sử dụng lý thuyết tập thơ trong các ứng dụng như phân biệt thuộc tính, giảm số chiều, khám phá tri thức, và phân tích dữ liệu thời gian, ... Xây dựng cây quyết định bằng thuật toán ID3 dựa

trên lượng thông tin thu thêm IG (Information Gain) giảm thiểu số lần cần so sánh. Ý tưởng cơ bản của thuật toán là thuộc tính có giá trị IG lớn nhất sẽ được chọn để phân nhánh như là một giải pháp “heuristic” trong việc chọn lựa thuộc tính phân lớp. Tuy nhiên, một vấn đề của các thuật toán trên là một cây con có thể lặp lại nhiều lần trong cây quyết định. Bên cạnh đó, một thuộc tính có thể được dùng nhiều lần trên một đường đi cụ thể của cây. Điều đó làm giảm hiệu quả quá trình phân cấp. Do đó lựa chọn thuộc tính để phân nhánh là vấn đề rất quan trọng được nhiều nhà khoa học nghiên cứu, và có rất nhiều công trình được công bố trong những năm gần đây. Lý thuyết tập thô đã chứng minh được tiềm năng lớn trong suy diễn, do đó luận văn nghiên cứu thuật toán tìm tập rút gọn của một bảng quyết định từ đó chọn được các thuộc tính cần thiết đưa vào xây dựng cấu trúc cây quyết định để chọn thuộc tính phân nhánh tối ưu, làm cho cây có chiều cao nhỏ nhất.

Cấu trúc của luận văn chia làm ba chương:

Chương 1: Khai phá dữ liệu theo tiếp cận tập thô

Trong chương này trình bày tổng quan về khai phá dữ liệu và lý thuyết tập thô, ví dụ minh họa cụ thể trên từng khái niệm.

Chương 2: Các phương pháp xây dựng cây quyết định

Trong chương này trình bày một số phương pháp tổng quát xây dựng cây quyết định.

Chương 3: Mô phỏng chương trình phân lớp năng khiếu học sinh

Giới thiệu bài toán. Phát biểu bài toán, cài đặt kiểm chứng thuật toán tìm tập rút gọn của một bảng quyết định dựa vào ma trận phân biệt được và xây dựng cây quyết định ID3 trên tập dữ liệu mẫu.

CHƯƠNG 1

KHAI PHÁ DỮ LIỆU THEO TIẾP CẬN TẬP THÔ

1.1 Giới thiệu

Lý thuyết tập thô (Rough set) được đề xuất vào năm 1982 bởi Z.Pawlak. Lý thuyết này xây dựng phương pháp luận liên quan đến sự phân loại và phân tích không chắc chắn, thông tin và tri thức không đầy đủ và được coi là một trong những phương pháp tiếp cận đầu tiên không dựa trên thống kê trong phân tích dữ liệu [6]

Khái niệm cơ bản của lý thuyết tập thô là xấp xỉ dưới và trên của một tập, sự xấp xỉ của không gian là hình thức phân loại tri thức liên quan đến miền quan tâm. Tập con được tạo ra bởi xấp xỉ dưới mô tả bởi các đối tượng là những thành phần chắc chắn của một tập, trong khi xấp xỉ trên được đặc trưng bởi các đối tượng có khả năng thuộc tập quan tâm. Mỗi tập con xác định thông qua xấp xỉ dưới và xấp xỉ trên được gọi là tập thô.

Gần đây, lý thuyết tập thô trở thành một công cụ đánh giá trong xử lý các vấn đề khác nhau như trình bày tri thức không chắc chắn hoặc không chính xác, phân tích tri thức, đánh giá chất lượng và tính khả dụng của thông tin đối với tính nhất quán và sự có mặt các mẫu không theo thời gian, nhận dạng và đánh giá sự phụ thuộc thời gian, suy luận dựa trên sự không chắc chắn và thiếu thông tin dữ liệu.

1.2 Các khái niệm cơ bản

1.2.1 Hệ thống thông tin

Trong hầu hết các hệ quản trị cơ sở dữ liệu thông thường thì thông tin thường được biểu diễn dưới dạng các bảng, trong đó mỗi hàng biểu diễn thông tin về một đối tượng, mỗi cột biểu diễn thông tin về một thuộc tính của đối tượng. Từ đầu những năm 80 Z. Pawlak đã định nghĩa một khái niệm mới là hệ thống tin (infomation system) dựa trên khái niệm bảng truyền thống như sau:

Định nghĩa 1.1 [1],[3]: Hệ thống thông tin là một cặp $S = (U, A)$

Trong đó:

U : là một tập hữu hạn khác rỗng các đối tượng gọi là tập vũ trụ hay là tập phổ dụng

A : là một tập hữu hạn khác rỗng các thuộc tính.

Với mỗi phần tử $u \in U$ và $a \in A$ ta kí hiệu $u(a)$ là giá trị của thuộc tính a tại đối tượng u . kí hiệu V_a là tập giá trị của thuộc tính $a \in A$. Nếu $B \subseteq A$ là

một tập các thuộc tính ta kí hiệu $u(B)$ là một bộ gồm các giá trị $u(a)$ với $a \in B$. Vậy nếu u và v là hai đối tượng thuộc U , ta sẽ nói $u(B)=v(B)$ nếu $u(a)=v(a)$ với mọi thuộc tính $a \in B$

Ví dụ 1.1: Bảng 1.1 dưới đây biểu diễn về một hệ thống thông tin của 16 đối tượng với 5 thuộc tính.

Bảng 1.1 Ví dụ về hệ thống thông tin

		Thuộc tính				
		To	Ly	Ho	Nv	Av
Đối tượng	x1	K2	K2	K2	K2	K1
	x2	K2	K2	K2	K2	K1
	x3	SX	G	K2	K2	K1
	x4	G	K2	K2	K2	K1
	x5	G	K2	K2	K2	K1
	x6	K1	K2	K2	K2	K1
	x7	K1	K2	K2	K2	K1
	x8	K2	K2	K2	TB	K2
	x9	K2	K2	K2	TB	K2
	x10	K2	K2	TB	K2	G
	x11	K2	K2	K2	TB	K2
	x12	K1	K2	K2	TB	K2
	x13	K1	K1	K2	K2	K2
	x14	K1	K2	K2	TB	K2
	x15	K1	K2	TB	K2	K2
	x16	K1	K2	K2	TB	K2

Ta có một hệ thống tin $S = (U, A)$.

$$U = \{x_1, x_2, \dots, x_{16}\}$$

$$A = \{To, Ly, Ho, Nv, Av\}$$

$$V_{To} = \{SX, G, K1, K2\};$$

$$V_{Ly} = \{G, K1, K2\};$$

$$V_{Ho} = \{K2, TB\};$$

$$V_{Nv} = \{ K2, TB \};$$

$$V_{Av} = \{ G, K1, K2 \};$$

1.2.2 Bảng quyết định

Để có thể biểu diễn một dữ liệu thực tế, trong đó có những thuộc tính quyết định, chúng ta xét một trường hợp đặc biệt của hệ thông tin được gọi là bảng quyết định được định nghĩa như sau

Định nghĩa 1.2. [1], [2] : Bảng quyết định là một hệ thông tin có dạng

$$DT = (U, A \cup \{d\})$$

Trong đó: $d \notin A$ là thuộc tính phân biệt, được gọi là thuộc tính quyết định. Các thành phần của A được gọi là các thuộc tính điều kiện.

Ví dụ 1.2: Mô tả một bảng quyết định, với các thuộc tính điều kiện lấy ở Bảng 1.1 và thêm vào thuộc tính quyết định “Tc”

Bảng 1.2 Ví dụ một bảng quyết định

	Thuộc tính điều kiện					Thuộc tính quyết định
	To	Ly	Ho	Nv	Av	Tc
x ₁	K2	K2	K2	K2	K1	A
x ₂	K2	K2	K2	K2	K1	A
x ₃	SX	G	K2	K2	K1	T
x ₄	G	K2	K2	K2	K1	T
x ₅	G	K2	K2	K2	K1	T
x ₆	K1	K2	K2	K2	K1	T
x ₇	K1	K2	K2	K2	K1	T
x ₈	K2	K2	K2	TB	K2	T
x ₉	K2	K2	K2	TB	K2	T
x ₁₀	K2	K2	K2	TB	G	A
x ₁₁	K2	K2	K2	TB	K2	T
x ₁₂	K1	K2	K2	TB	K2	T
x ₁₃	K1	K1	K2	K2	K2	A
x ₁₄	K1	K2	K2	TB	K2	T
x ₁₅	K1	K2	TB	K2	K2	A
x ₁₆	K1	K2	K2	TB	K2	T

Chúng ta giả sử rằng tập các giá trị của giá trị quyết định d tương đương với tập $\{1, \dots, r(d)\}$ là các số nguyên dương từ 1 đến $r(d)$, tập này được gọi là phạm vi của thuộc tính quyết định d .

Lớp quyết định thứ k (ký hiệu là C_k) là một tập các đối tượng thoả mãn: $C_k = \{u \in U: d(u)=k\}$. Trong đó $1 \leq k \leq r(d)$.

Khi đó giá trị quyết định d sẽ chia tập các đối tượng thành $r(d)$ lớp quyết định: $\{C_1, \dots, C_{r(d)}\}$.

Trong trường hợp tổng quát thì có thể có nhiều thuộc tính quyết định, khi đó bảng quyết định có dạng $DT=(U, C \cup D)$, trong đó:

$$A = C \cup D$$

C : gọi là tập thuộc tính điều kiện.

D : được gọi là tập thuộc tính quyết định.

Bảng quyết định được gọi là nhất quán nếu với mọi $u, v \in U$, $u(C)=v(C)$ kéo theo $u(D)=v(D)$. Ngược lại, gọi là bảng không nhất quán.

1.2.3 Quan hệ không phân biệt được

Một trong những đặc điểm cơ bản của lý thuyết tập thô là dùng để lưu giữ và xử lý các dữ liệu không phân biệt được. Trong một hệ thông tin theo định nghĩa trên cũng có thể có những đối tượng không phân biệt được. Trước tiên ta nhắc lại định nghĩa quan hệ tương đương như sau:

Định nghĩa 1.5 [3] Một quan hệ hai ngôi (quan hệ nhị phân) $R \subseteq U \times U$ trên U là một quan hệ tương đương khi nó có cả 3 tính chất:

- Phản xạ: Mọi đối tượng đều quan hệ với chính nó.
- Đối xứng: Nếu xRy thì yRx .
- bắc cầu: Nếu xRy và yRz thì xRz .

Quan hệ tương đương R sẽ chia tập các đối tượng U thành các lớp tương đương. Lớp tương đương của phần tử $x \in U$, ký hiệu là $[x]_R$, chứa tất cả các đối tượng y mà xRy .

Bây giờ bắt đầu định nghĩa một quan hệ tương đương trên hệ thông tin. Quan hệ này sau này được sử dụng để biểu diễn những thông tin không phân biệt được.

Định nghĩa 1.6 [1],[3] cho tập con các thuộc tính $B \subset A$ trong hệ thống thông tin (U, A) . Quan hệ B-không phân biệt được (ký hiệu là $IND_A(B)$), được định nghĩa như sau:

$$IND_A(B) = \{(x, x') \in U^2 \mid \forall a \in B, a(x) = a(x')\}$$

Khi đó $IND_A(B)$ là một quan hệ tương đương trên U .

Lớp tương đương chứa x của quan hệ không phân biệt được trên B được ký hiệu là $[x]_B$.

Hai đối tượng x, x' , mà $(x, x') \in IND_A(B)$ được gọi là không phân biệt được bởi các thuộc tính trong B .

Khi xét trên một hệ thống tin xác định ta sẽ viết $IND(B)$ thay cho $IND_A(B)$.

Ví dụ 1.3: Xét hệ thống tin cho ở Bảng 1.1, phân hoạch của tập U sinh bởi quan hệ tương đương $IND(B)$:

- Với $B = \{To\}$ ta có $IND(B) = \{\{x_1, x_2, x_8, x_9, x_{10}, x_{11}\}, \{x_3\}, \{x_4, x_5\}, \{x_6, x_7, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}\}\}$. Lúc này ta nói x_1 và x_2 là không phân biệt được.

- Với $B = \{To, Ly, Ho, Nv, Av\}$ ta có $IND(B) = \{\{x_1, x_2\}, \{x_3\}, \{x_4, x_5\}, \{x_6, x_7\}, \{x_8, x_9, x_{10}, x_{11}\}, \{x_{12}, x_{14}, x_{15}, x_{16}\}, \{x_{13}\}\}$.

1.2.4 Các khái niệm xấp xỉ trong tập thô

a) *Xấp xỉ dưới, xấp xỉ trên*

Định nghĩa 1.7: [1],[3] Cho bảng quyết định $DT = (U, C \cup D)$ và tập thuộc tính $B \subset C$, $X \subseteq U$. Xấp xỉ trên và xấp xỉ dưới của tập X tương ứng với B , ký hiệu theo thứ tự là $\underline{B}X$ và $\overline{B}X$ được định nghĩa như sau:

$$\underline{B}X = \{x \in U : [x]_B \subset X\},$$

$$\overline{B}X = \{x \in U : [x]_B \cap X \neq \emptyset\}.$$

Tập hợp $\underline{B}X$ là tập các đối tượng trong U mà sử dụng các thuộc tính trong B ta có thể biết chắc chắn chúng là phần tử của X .

Tập hợp $\overline{B}X$ là tập các đối tượng trong U mà sử dụng các thuộc tính trong B ta chỉ có thể nói rằng chúng có thể là các phần tử của X .

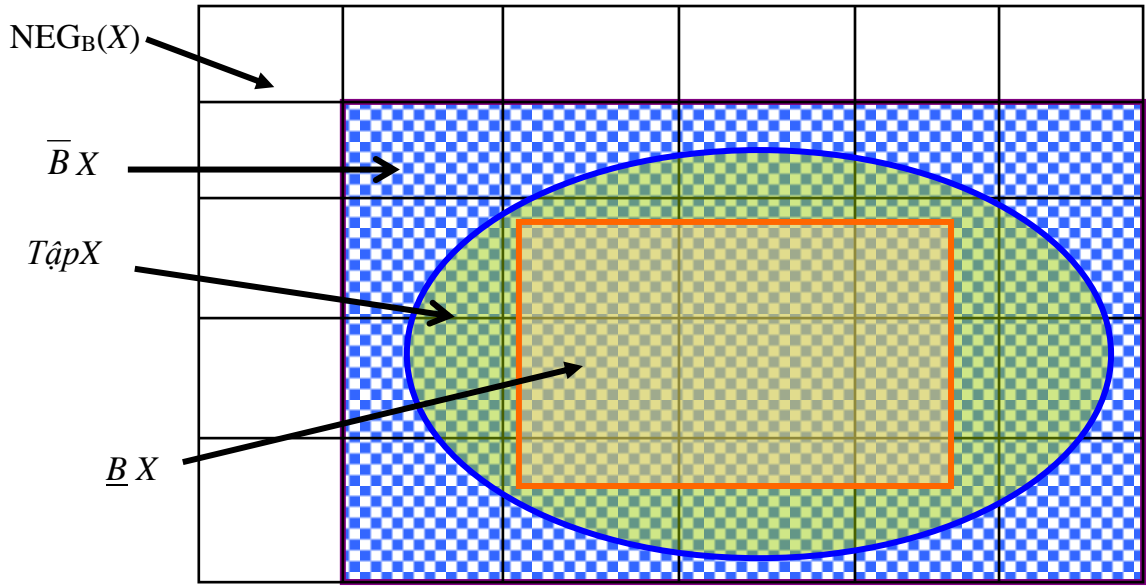
b) *Miền biên, Miền ngoài*[3]

- B -biên của tập X , ký hiệu $BN_B(X)$, được định nghĩa $BN_B(X) = \overline{B}X \setminus \underline{B}X$

$BN_B(X)$ chứa những đối tượng mà sử dụng các thuộc tính trong B ta không thể xác định được chúng có thuộc X hay không.

- B -ngoài của tập X , ký hiệu $NEG_B(X)$ được định nghĩa $NEG_B(X) = U \setminus \overline{B}X$
 $NEG_B(X)$ chứa những đối tượng mà sử dụng các thuộc tính trong B ta biết chắc chắn chúng không thuộc X .

Hình sau trình bày sự mô tả về tập xấp xỉ và miền



Hình 1.1: Mô tả về tập xấp xỉ và miền

Ví dụ 1.4: Trong Bảng 1.2 Với $U = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}\}$. Chọn thuộc tính điều kiện $B = \{Ly, Nv, Av\}$ và thuộc tính quyết định ta có: $D = \{Tc\}$ ta có:

Các lớp tương đương ứng với quan hệ $IND(B)$ là:

$IND(B) = \{E_1, E_2, E_3, E_4, E_5\}$, Trong đó:

$$E_1 = \{x_1, x_2, x_4, x_5, x_6, x_7\};$$

$$E_2 = \{x_3\};$$

$$E_3 = \{x_8, x_9, x_{11}, x_{12}, x_{14}, x_{16}\};$$

$$E_4 = \{x_{13}\};$$

$$E_5 = \{x_{10}, x_{15}\}.$$

Xấp xỉ trên và dưới của $D_T = \{x | Tc(x) = T\}$

$$\underline{B}D_T = \{E_2, E_3\} = \{x_3, x_8, x_9, x_{11}, x_{12}, x_{14}, x_{16}\}$$

$$\overline{B}D_T = \{E_1, E_2, E_3\} = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{11}, x_{12}, x_{14}, x_{16}\}$$

Miền biên, miền ngoài của $D_T = \{ x \mid Tc(x) = T \}$

$$BR_B(D_T) = \overline{B} D_T \setminus \underline{B} D_T = \{ x_1, x_2, x_4, x_5, x_6, x_7 \}$$

$$NEG_B(D_T) = U \setminus \overline{B} (D_T) = \{ x_{10}, x_{13}, x_{15} \}$$

Xấp xỉ trên và dưới của $D_T = \{ x \mid Tc(x) = A \}$

$$\underline{B} D_A = \{ E_4, E_5 \} = \{ x_{10}, x_{13}, x_{15} \}$$

$$\overline{B} D_A = \{ E_1, E_4, E_5 \} = \{ x_1, x_2, x_4, x_5, x_6, x_7, x_{10}, x_{13}, x_{15} \}$$

Miền biên, miền ngoài của $D_T = \{ x \mid Tc(x) = A \}$

$$BR_B(D_A) = \overline{B} D_A \setminus \underline{B} D_A = \{ x_1, x_2, x_4, x_5, x_6, x_7 \}$$

$$NEG_B(D_A) = U \setminus \overline{B} (D_A) = \{ x_3, x_8, x_9, x_{11}, x_{12}, x_{14}, x_{16} \}$$

c) Đặc trưng xấp xỉ [3]

Hệ số được dùng để đo lường đặc trưng được biểu diễn bởi

$$\alpha_B(X) = \frac{|\underline{B}(X)|}{|\overline{B}(X)|}$$

Trong đó $|\overline{B}(X)|$ và $|\underline{B}(X)|$ là lực lượng của xấp xỉ trên và dưới và các xấp xỉ là tập khác rỗng đó đó $0 \leq \alpha_B(X) \leq 1$.

Nếu $\alpha_B(X)=1$, X là một tập định nghĩa được theo thuộc tính B do đó X là tập cổ điển.

Nếu $\alpha_B(X) < 1$, X là tập thô theo thuộc tính B .

Ví dụ 1.5: Áp dụng công thức trên cho Bảng 1.2 ta được:

$$\alpha_B(D_T) = \frac{|\underline{B}(D_T)|}{|\overline{B}(D_T)|} = \frac{7}{13}; \quad \alpha_B(D_A) = \frac{|\underline{B}(D_A)|}{|\overline{B}(D_A)|} = \frac{6}{7}$$

d) Một số tính chất của các tập hợp xấp xỉ [3]

1. $\underline{B}(X) \subseteq X \subseteq \overline{B}(X)$
2. $\underline{B}(\emptyset) = \overline{B}(\emptyset) = \emptyset, \underline{B}(U) = \overline{B}(U) = U$
3. $\overline{B}(X \cup Y) = \overline{B}(X) \cup \overline{B}(Y)$
4. $\underline{B}(X \cap Y) = \underline{B}(X) \cap \underline{B}(Y)$
5. Nếu $X \subseteq Y$ thì $\underline{B}(X) \subseteq \underline{B}(Y), \overline{B}(X) \subseteq \overline{B}(Y)$
6. $\underline{B}(X \cup Y) \supseteq \underline{B}(X) \cup \underline{B}(Y)$
7. $\overline{B}(X \cap Y) \subseteq \overline{B}(X) \cap \overline{B}(Y)$

$$8. \underline{B}(U \setminus X) = U \setminus \overline{B}(X)$$

$$9. \overline{B}(U \setminus X) = U \setminus \underline{B}(X)$$

$$10. \underline{B}(\underline{B}(X)) = \overline{B}(\overline{B}(X)) = \underline{B}(X)$$

$$11. \overline{B}(\overline{B}(X)) = \underline{B}(\underline{B}(X)) = \overline{B}(X)$$

Người ta phân tập thô thành 4 loại [3]:

- X là xác định thô thực sự theo B nếu $\underline{B}X \neq \emptyset$ và $\overline{B}X \neq U$
- X là không xác định bên trong theo B nếu $\underline{B}X = \emptyset$ và $\overline{B}X \neq U$
- X là không xác định bên ngoài theo B nếu $\underline{B}X \neq \emptyset$ và $\overline{B}X = U$
- X là không xác định thực sự theo B nếu $\underline{B}X = \emptyset$ và $\overline{B}X = U$

1.2.5 Sự phụ thuộc của các thuộc tính

Trong phân tích dữ liệu, điều quan trọng là khám phá sự phụ thuộc giữa các thuộc tính. Một cách trực giác, một tập thuộc tính D phụ thuộc hoàn toàn trên tập thuộc tính C , kí hiệu $C \Rightarrow D$ nếu tất cả các giá trị của thuộc tính D xác định duy nhất bởi các giá trị của thuộc tính trong C . nói cách khác D phụ thuộc hoàn toàn trên C , nếu tồn tại một phụ thuộc hàm giữa các giá trị của D và C .

Khái niệm sự phụ thuộc của các thuộc tính được thể hiện dưới dạng hình thức như sau [3]:

Cho C và D là các tập con của tập thuộc tính A . Ta nói D phụ thuộc C với độ phụ thuộc k ($0 \leq k \leq 1$), ký hiệu $C \Rightarrow_k D$

$$k = \gamma(C, D) = \frac{|POS_C(D)|}{|U|}$$

trong đó: $POS_C(D) = \bigcup_{x \in D} \underline{C}(x)$

Tập $POS_C(D)$ được gọi là C -miền khẳng định của D . Nói cách khác $u \in POS_C(D)$ nếu và chỉ nếu $u(C) = v(C)$ kéo theo $u(D) = v(D)$ với mọi $v \in U$.

Đây là tập các đối tượng của U mà bằng cách sử dụng tập thuộc tính C ta có thể phân chúng một cách duy nhất vào phân hoạch của U theo tập thuộc tính D .

Nếu $k=1$ ta nói D phụ thuộc hoàn toàn vào C ;

Nếu $k < 1$ ta nói D phụ thuộc một phần vào C .

Có thể dễ dàng nhìn thấy rằng nếu D phụ thuộc hoàn toàn trên C thì $IND(C) \subseteq IND(D)$, nghĩa là sự phân chia đã tạo ra bởi C mịn hơn sự phân chia tạo ra

bởi D và khái niệm về sự phụ thuộc đã trình bày trong phần này tương ứng với các vấn đề đã quan tâm trong CSDL quan hệ.

Ví dụ 1.6 Sự phụ thuộc của thuộc tính:

Bảng 1.3. Hệ thống tin minh họa sự phụ thuộc của thuộc tính

	To	Ly	Ho	Nv	Av	Tc
x_1	K2	K2	K2	K2	K2	A
x_2	K2	K2	K2	K2	K2	A
x_3	K2	G	K2	K2	K1	T
x_4	G	K2	K2	K2	K1	T
x_5	G	K2	K2	K2	K1	T
x_6	K1	K2	K2	K2	K1	T
x_7	K1	K2	K2	K2	K1	T
x_8	K1	K2	K2	TB	K2	A
x_9	TB	K2	K2	TB	K2	A
x_{10}	TB	K2	K2	TB	K2	A

- Ta có một phụ thuộc hoàn toàn là: $\{Av\} \Rightarrow \{Tc\}$, bởi vì mỗi giá trị của thuộc tính Av sẽ tương ứng với một giá trị duy nhất của thuộc tính “Tc”.

- Ta có phụ thuộc một phần là: thuộc tính To xác định một vài giá trị duy nhất của thuộc tính Av. Đó là $\{To\} = 'G' \Rightarrow \{Tc\} = 'T'$, tương tự $\{To\} = 'TB' \Rightarrow \{Tc\} = 'A'$, nhưng $\{To\} = 'K1'$ hoặc $\{To\} = 'K2' \Rightarrow \{Tc\} = 'T'$ hoặc $\{Tc\} = 'A'$.

1.2.6 Rút gọn các thuộc tính trong hệ thống thông tin

Thông tin trong các hệ thống có thể dư thừa, các dư thừa có thể xảy ra:

Trường hợp 1: Các đối tượng giống nhau theo một tập thuộc tính đang quan tâm được lặp lại nhiều lần.

Trường hợp 2: Một số thuộc tính có thể bỏ đi mà thông tin chúng ta đang quan tâm do bảng quyết định cung cấp vẫn không bị mất mát.

- Với trường hợp 1: khái niệm lớp tương đương cho ta tiếp cận tinh giảm thông tin cần lưu trữ trong một hệ thống tin. Ta chỉ cần sử dụng một đối tượng để đại diện cho mỗi lớp tương đương.

- Với trường hợp 2: Chỉ giữ lại những thuộc tính bảo toàn quan hệ bất khả phân biệt, do đó bảo toàn khả năng xấp xỉ tập hợp trong một hệ thông tin.

Quá trình rút gọn một hệ thống thông tin mà tập các thuộc tính của hệ thống thông tin đã được rút gọn là độc lập và không còn thuộc tính nào có thể bị loại bỏ hơn nữa mà không làm mất thông tin từ hệ thống, kết quả được biết đến như là tập rút gọn. Nếu một thuộc tính từ tập con $B \subseteq A$ duy trì mối quan hệ không phân biệt được $IND(A)$ thì các thuộc tính $A \setminus B$ là không cần thiết. Các tập rút gọn cũng là tập con tối thiểu, nghĩa là không chứa các thuộc tính không cần thiết. Do đó việc rút gọn có khả năng phân loại các đối tượng mà không làm thay đổi hình thức của việc diễn tả tri thức

Thuộc tính cần thiết và không cần thiết. [1],[3]

Xét bảng quyết định $DT = (U, C \cup D)$.

Thuộc tính $c \in C$ được gọi là không cần thiết trong DT nếu $POS_C(D) = POS_{(C-\{c\})}(D)$. Ngược lại ta nói c là cần thiết trong DT .

Rõ ràng thuộc tính không cần thiết không làm tăng hay giảm khả năng phân loại khi có hoặc không có mặt thuộc tính đó trong C .

Khi loại khỏi C một số thuộc tính có thể bỏ được thì ta được một tập rút gọn của C .

Ta nói bảng quyết định $DT = (U, C \cup D)$ là độc lập nếu tất cả các thuộc tính $c \in C$ đều cần thiết trong DT ;

Rút gọn và lõi: [1],[3]

Tập thuộc tính $R \subseteq C$ được gọi là một rút gọn của C nếu $DT' = (U, R \cup D)$ là độc lập và $POS_R(D) = POS_C(D)$.

Một tập rút gọn là một tập con các thuộc tính duy trì các đặc tính cơ bản của tập dữ liệu gốc; do đó các thuộc tính không thuộc về một tập rút gọn là không cần thiết đối với sự phân loại các phần tử của tập vũ trụ.

Tập tất cả các thuộc tính cần thiết trong DT kí hiệu: $CORE(C)$. Khi đó,

$CORE(C) = \cap RED(C)$ Với $RED(C)$: Là tập tất cả các rút gọn của C .

Ví dụ 1.7: Rút gọn các thuộc tính trong hệ thống thông tin

Bảng 1.4 Rút gọn các thuộc tính trong hệ thống thông tin

	Av	Nv	Ly	Tc
x ₁	G	K1	K2	V
x ₂	G	K1	K1	L
x ₃	G	K1	G	L
x ₄	K2	K1	K2	V
x ₅	K2	K2	K1	V
x ₆	K2	K1	G	L

	Av	Ly	Tc
x ₁	G	K2	V
x ₂	G	K1	L
x ₃	G	G	L
x ₄	K2	K2	V
x ₅	K2	K1	V
x ₆	K2	G	L

	Nv	Ly	Tc
x ₁ , x ₄	K1	K2	V
x ₂	K1	K1	L
x ₃ , x ₆	K1	G	L
x ₅	K2	K1	V

- Rút gọn R₁= {Av, Ly,}
- Rút gọn R₂= {Nv, Ly,}
- Ta có: CORE = R₁ ∩ R₂ = {Ly}

1.2.7 Ma trận phân biệt

Phần trên cung cấp khái niệm về rút gọn thuộc tính trong một hệ thống tin, tuy nhiên chúng chưa thực sự rõ nét. Trong phần này chúng ta sẽ tìm hiểu bản chất của một rút gọn của tập thuộc tính và là cơ sở để hiểu được các thuật toán tìm tập rút gọn trong một hệ thống tin.

Định nghĩa 1.8. [1,3] Cho bảng quyết định $DT = (U, C \cup D)$ và tập đối tượng $U = \{u_1, u_2, \dots, u_n\}$. Ma trận phân biệt được của DT , kí hiệu: $M(DT) = (m_{ij})_{n \times n}$, là một ma trận đối xứng mà mỗi phần tử của nó là một tập hợp các thuộc tính, được xác định như sau:

$$m_{ij} = \begin{cases} \lambda, & u_i(D) = u_j(D) \\ \{c \in C \mid u_i(c) \neq u_j(c)\}, & u_i(D) \neq u_j(D) \end{cases}$$

Như vậy m_{ij} là tập hợp gồm tất cả các thuộc tính điều kiện có thể xếp các đối tượng u_i và u_j vào các lớp tương đương khác nhau.

Giá trị λ hàm ý cặp đối tượng u_i và u_j không phân biệt trên tập thuộc tính quyết định D

Ví dụ 1.8: Xét bảng quyết định sau

Bảng 1.5 Bảng quyết định minh họa ma trận phân biệt được

U	To	Ly	Av	Nv	Tc
u_1	G	K2	K1	G	T
u_2	G	K2	K1	K2	T
u_3	G	K1	K2	K2	A
u_4	G	K1	K1	G	L
u_5	K1	G	K2	K2	A
u_6	K1	G	G	K2	A
u_7	K1	G	K1	G	T

Trong đó tập thuộc tính điều kiện $C = \{To, Ly, Av, Nv\}$ và tập thuộc tính quyết định $D = \{Tc\}$.

Ta có ma trận phân biệt được tương ứng (là ma trận đối xứng nên ta chỉ cần xác định nửa ma trận dưới):

Bảng 1.6 Ma trận phân biệt của hệ thống tin trong Bảng 1.4

U	u_1	u_2	u_3	u_4	u_5	u_6	u_7
u_1	λ						
u_2	λ	λ					
u_3	$\{Ly, Av, Nv\}$	$\{Ly, Av\}$	λ				
u_4	$\{Ly\}$	$\{Ly, Nv\}$	$\{Av, Nv\}$	λ			
u_5	$\{To, Ly, Av, Nv\}$	$\{To, Ly, Av\}$	λ	$\{To, Ly, Av, Nv\}$	λ		
u_6	$\{To, Ly, Av, Nv\}$	$\{To, Ly, Av\}$	λ	$\{To, Ly, Av, Nv\}$	λ	λ	
u_7	λ	λ	$\{To, Ly, Av, Nv\}$	$\{To, Ly\}$	$\{Av, Nv\}$	$\{Av, Nv\}$	λ

1.3 Rút gọn dữ liệu trong hệ thống thông tin

Hình thức mà dữ liệu được biểu diễn trong một hệ thống thông tin phải đảm bảo không có sự dư thừa dữ liệu, ngụ ý rằng việc tối thiểu hóa các phép tính toán phức tạp trong quan hệ với việc tạo ra các luật trợ giúp việc trích xuất tri thức. Tuy nhiên, khi hệ thống thông tin sở hữu tình huống dư thừa dữ liệu, thì cần phải đối xử với nó. Một trong các cách để thực hiện việc này là sử dụng khái niệm rút gọn, mà không cần thay đổi các quan hệ không phân biệt được.

Một rút gọn là một tập các dữ liệu tối thiểu cần thiết, vì các thuộc tính gốc của hệ thống hay bảng thông tin là được duy trì. Vì vậy, tập rút gọn phải có khả năng phân lớp các đối tượng, mà không làm thay đổi hình thức biểu diễn tri thức.

1.4 Thuật toán tìm tập rút gọn của một bảng quyết định dựa vào ma trận phân biệt được [1]

Nói chung mọi thuật toán xác định các đối tượng của tập thô đều có thể dựa vào ma trận phân biệt được. Tuy vậy, các thuật toán này thường đòi hỏi một độ phức tạp rất lớn về thời gian và không gian lưu trữ. Để khắc phục nhược điểm đó, thuật toán đề cập ở đây cũng dựa vào ý nghĩa của ma trận phân biệt được nhưng không cần phải lưu trữ ma trận. Thay vào đó, thuật toán xác định số cặp đối tượng phân biệt được đối với từng thuộc tính điều kiện.

Cho $B \subset C$, $c_j \in C \setminus B$ và $X \subset U$. Ta kí hiệu $w_B^X(c_j)$ là số cặp đối tượng của X bằng nhau trên B nhưng khác nhau tại thuộc tính c_j . Tức là

$$w_B^X(c_j) = \text{Card}(\{(u,v) \in X^2 \mid u(B)=v(B) \text{ và } u(c_j) \neq v(c_j)\})$$

Tương tự

$$w_B^X(D) = \text{Card}(\{(u,v) \in X^2 \mid u(B)=v(B) \text{ và } u(D) \neq v(D)\})$$

Khi $B=\emptyset$ hai đại lượng trên được viết một cách đơn giản là $w^X(c_j)$ và $w^X(D)$. Chẳng hạn $w^X(c_j) = \text{Card}(\{(u,v) \in X^2 \mid u(c_j) \neq v(c_j)\})$

Khi $X=U$ ta viết các kí hiệu trên lần lượt là $w_B(c_j)$ và $w_B(D)$, còn khi $X=U$ và $B=\emptyset$ ta viết các kí hiệu trên lần lượt là $w(c_j)$ và $w(D)$

Nếu $R \subseteq C$ là một rút gọn của C thì mọi cặp đối tượng bằng nhau trên R cũng bằng nhau trên D , hay nói cách khác $w_R(D)=0$.

Tính hợp lý của thuật toán này dựa trên cơ sở khẳng định sau.

Mệnh đề 1.1[1] Cho $X \subseteq U$, giả sử $\text{IND}_X(D) = \{X_1, X_2, \dots, X_m\}$ với $\text{Card}(X) = x$, $\text{Card}(X_i) = x_i$

Khi đó:

$$x = \sum_{i=1}^m x_i$$

$$\text{và } w^X(D) = \sum_{i < j} x_i x_j = \frac{1}{2} \left(x^2 - \sum_{i=1}^m x_i^2 \right)$$

Mệnh đề 1.2 [1] Giả sử $X \subseteq U$, $R \subseteq C$ và $\text{IND}_X(R) = \{X_1, X_2, \dots, X_k\}$. Khi đó

$$(a) \quad w_B^X(D) = w_B^{X_1}(D) + w_B^{X_2}(D) + \dots + w_B^{X_k}(D)$$

(b) Với $c_j \in C \setminus R$, ta có

$$[\text{IND}_X(R \cup \{c\})] = \text{IND}_{X_1}(D) + \text{IND}_{X_2}(D) + \dots + \text{IND}_{X_k}(D)$$

(c) Nếu $c_j \in C \setminus R$ và $[\text{IND}_{X_i}(c_j)] = \{Y_1^i, Y_2^i, \dots, Y_p^i\}$ thì

$$w_{R \cup \{c_j\}}^{X_i}(D) = w^{Y_1^i}(D) + w^{Y_2^i}(D) + \dots + w^{Y_p^i}(D)$$

Mệnh đề 1.3 [1] R là một rút gọn của tập thuộc tính điều kiện C khi và chỉ khi R là tập tối thiểu thỏa $w_R(D) = 0$

Chứng minh:

Rõ ràng theo nhận xét trong phần trên, nếu R là một rút gọn của C thì R là tập tối thiểu thỏa tính chất: mọi cặp đối tượng bằng nhau trên R cũng bằng nhau trên D hay $w_R(D) = 0$.

Ngược lại, nếu R là tập tối thiểu thỏa $w_R(D) = 0$ có nghĩa là R xác định D hay $POS_R(D) = POS_C(D)$ và mọi tập con thực sự của R không thỏa tính chất này, do đó $DT' = (U, R \cup D)$ là độc lập. Vậy R là một rút gọn của C .

Vấn đề đặt ra là tại mỗi bước chọn lựa thuộc tính nào sẽ được đưa vào R . Một cách tự nhiên ta chọn thuộc tính mà khi tham gia vào tập rút gọn sẽ làm số cặp đối tượng bằng nhau trên R nhưng khác nhau trên D là ít nhất. Với cách chọn lựa heuristic này thuật toán có khả năng cho ta một tập rút gọn với số thuộc tính tối thiểu.

Thoạt tiên, ta chọn $R = \emptyset$ và sẽ bổ xung dần các thuộc tính vào R . tại mỗi bước, ta luôn kí hiệu $L = [IND(R)]$. Ban đầu $R = \emptyset$ nên $L = \{U\}$

Thuật toán

Vào: $DT = \{U, C \cup D\}$

Ra: Tập rút gọn R

Phương pháp:

$R = \emptyset; L = \{U\};$

Repeat

For $c_j \in C \setminus R$ do

Begin

For $X_i \in L$ do

Begin

Tìm $[IND_{X_i}(c_j)] = \{Y_1, Y_2, \dots, Y_m\}$

For $l=1$ to m do

Begin

Tìm $[IND_{Y_l}(D)] = \{Y_1^l, Y_2^l, \dots, Y_k^l\}$

$$w^{Y_l}(D) = \frac{1}{2} ((x^l)^2 - \sum_{i=1}^k (x_i^l)^2)$$

(Trong đó $x^l = \text{Card}(Y_1)$ và $x_i^l = \text{Card}(Y_i^l)$)

End

$$\gamma_j^i = w_{R \cup \{c_j\}}^{X_i}(D) = w^{Y_1}(D) + w^{Y_2}(D) + \dots + w^{Y_m}(D)$$

End

$$\alpha_j = w_{R \cup \{c_j\}}^{X_i}(D) = \sum_{X \in L_i} \gamma_j^i$$

End

Chọn thuộc tính c_r sao cho α_r bé nhất

$R = R \cup \{c_r\}$

$L = \bigcup_{X_i \in L} [IND_{X_i}(c_r)] \quad (= [IND(R)])$

Until $((\alpha_r = 0) \text{ hoặc } (R = C))$

Ví dụ 1.11 Xét bảng quyết định

Bảng 1.7 bảng quyết định minh họa ví dụ 1.11

U	To	Av	Nv	Tc
u ₁	G	K2	K2	T
u ₂	K2	K1	G	A
u ₃	K1	K2	TB	T
u ₄	G	TB	K1	A
u ₅	G	TB	K2	T
u ₆	G	K1	TB	A
u ₇	G	K2	K1	A

Thực hiện thuật giải trên ta nhận được kết quả từng bước như sau:

$$R = \emptyset; L = \{U\} \quad (X_1 = U)$$

$$[C_1^{X_1}] = \{Y_1 = \{u_1, u_4, u_5, u_6, u_7\}, Y_2 = \{u_2\}, Y_3 = \{u_3\}\}$$

$$\omega^{Y_1}(D) = \frac{1}{2}(5^2 - 2^2 - 3^2) = 6$$

$$\omega^{Y_2}(D) = \frac{1}{2}(1^2 - 1^2) = 0$$

$$\omega^{Y_3}(D) = \frac{1}{2}(1^2 - 1^2) = 0$$

$$\gamma_1^1 = \omega^{Y_1}(D) + \omega^{Y_2}(D) + \omega^{Y_3}(D) = 6$$

$$[C_2^{X_1}] = \{Y_1 = \{u_1, u_3, u_7\}, Y_2 = \{u_2, u_6\}, Y_3 = \{u_4, u_5\}\}$$

$$\omega^{Y_1}(D) = \frac{1}{2}(3^2 - 2^2 - 1^2) = 2$$

$$\omega^{Y_2}(D) = \frac{1}{2}(2^2 - 2^2) = 0$$

$$\omega^{Y_3}(D) = \frac{1}{2}(2^2 - 1^2 - 1^2) = 1$$

$$\gamma_2^1 = \omega^{Y_1}(D) + \omega^{Y_2}(D) + \omega^{Y_3}(D) = 3$$

$$[C_3^{X_1}] = \{Y_1 = \{u_1, u_5\}, Y_2 = \{u_2\}, Y_3 = \{u_3, u_6\}, Y_4 = \{u_4, u_7\}\}$$

$$\omega^{Y_1}(D) = \frac{1}{2}(2^2 - 2^2) = 0$$

$$\omega^{Y_2}(D) = 0$$

$$\omega^{Y_4}(D) = 0$$

$$\gamma_3^1 = \omega^{Y_1}(D) + \omega^{Y_2}(D) + \omega^{Y_3}(D) + \omega^{Y_4}(D) = 1$$

$$\left. \begin{array}{l} \alpha_1 = \gamma_1^1 = 6; \\ \alpha_2 = \gamma_2^1 = 3; \\ \alpha_3 = \gamma_3^1 = 1 \end{array} \right\} \Rightarrow r = 3$$

$$R = R \cup \{c_3\} = \{c_3\}$$

$$L = \{X_1 = \{u_1, u_5\}, X_2 = \{u_2\}, X_3 = \{u_3, u_6\}, X_4 = \{u_4, u_7\}\}$$

$$[C_1^{X_1}] = \{Y_1 = \{u_1, u_5\}\} \Rightarrow \gamma_1^1 = 0$$

$$[C_1^{X_2}] = \{Y_1 = \{u_2\}\} \Rightarrow \gamma_1^2 = 0$$

$$[C_1^{X_3}] = \{Y_1 = \{u_3\}, Y_2 = \{u_6\}\} \Rightarrow \gamma_1^3 = 0$$

$$[C_1^{X_4}] = \{Y_1 = \{u_4, u_7\}\} \Rightarrow \gamma_1^4 = 0$$

$$[C_2^{X_1}] = \{Y_1 = \{u_1\}, Y_2 = \{u_5\}\} \Rightarrow \gamma_2^1 = 0$$

$$[C_2^{X_2}] = \{Y_1 = \{u_2\}\} \Rightarrow \gamma_2^2 = 0$$

$$[C_2^{X_3}] = \{Y_1 = \{u_3\}, Y_2 = \{u_6\}\} \Rightarrow \gamma_2^3 = 0$$

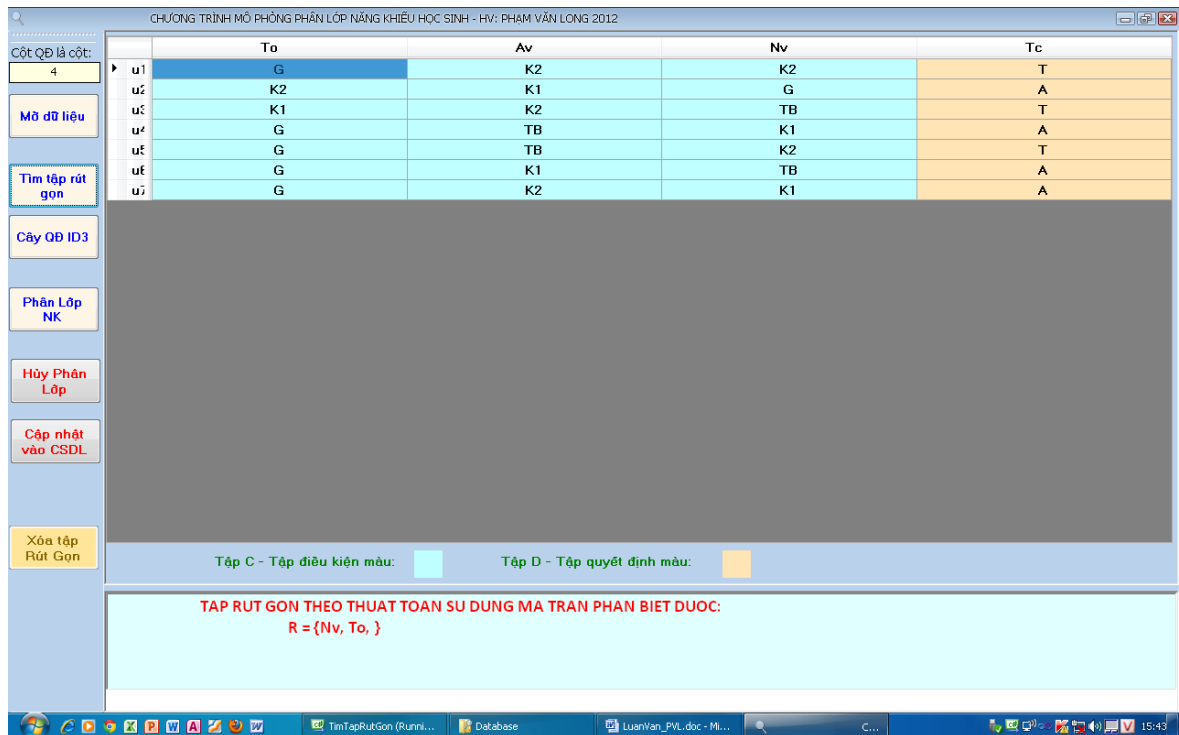
$$[C_2^{X_4}] = \{Y_1 = \{u_4\}, Y_2 = \{u_7\}\} \Rightarrow \gamma_2^4 = 0$$

$$\alpha_1 = \gamma_1^1 + \gamma_1^2 + \gamma_1^3 + \gamma_1^4 = 0$$

$$\alpha_2 = \gamma_2^1 + \gamma_2^2 + \gamma_2^3 + \gamma_2^4 = 0$$

Đến đây ta có thể chọn c_1 hoặc c_2 . Thuật toán dừng và ta nhận được hai rút gọn tương ứng:

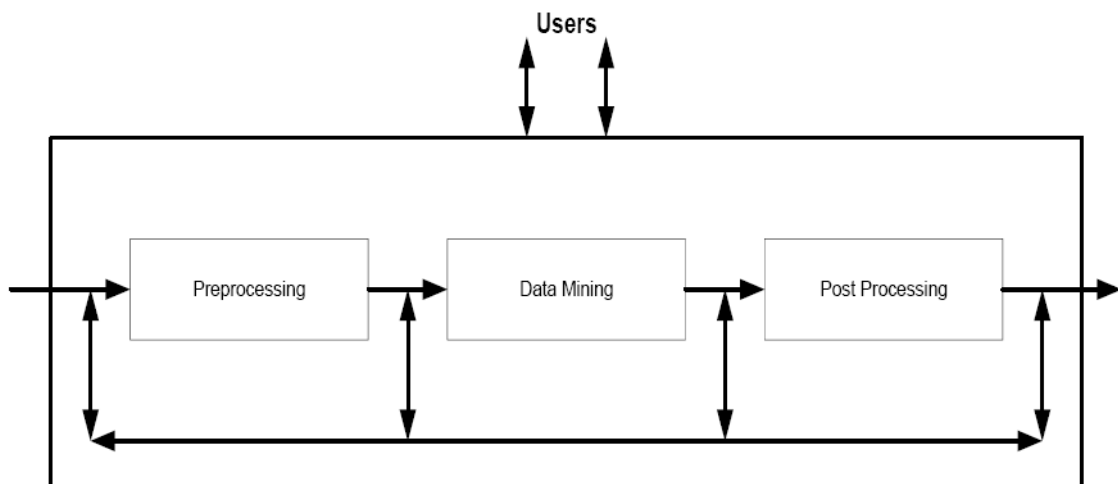
$$R = \{c_1, c_3\} = \{To, Nv\}; \quad R = \{c_2, c_3\} = \{Av, Nv\}$$

Hình 1.2: minh họa chạy thuật toán tìm tập rút gọn cho ví dụ trên từ chương trình

1.5 Tập thô với các công cụ khai phá dữ liệu

1.5.1 Khám phá tri thức trong cơ sở dữ liệu

Khám phá tri thức trong cơ sở dữ liệu là xử lý với các giai đoạn không tầm thường, tương tác và lặp lại cho sự nhận dạng của các mẫu có khả năng hiểu được, hợp lệ, mới và hữu ích tiềm tàng bắt đầu từ các nhóm dữ liệu lớn. Khám phá tri thức trong cơ sở dữ liệu được mô tả như một tiến trình bao gồm một vài giai đoạn thực hiện: tiền xử lý, khai phá dữ liệu và hậu xử lý.

Hình 1.3 Xử lý khám phá tri thức trong cơ sở dữ liệu

Giai đoạn tiền xử lý

Giai đoạn tiền xử lý hiểu được các chức năng liên quan đến việc tiếp nhận, tổ chức và xử lý dữ liệu, giai đoạn này được xem là giai đoạn chuẩn bị dữ liệu cho giai đoạn tiếp theo của khai phá dữ liệu.

Giai đoạn khai phá dữ liệu

Giai đoạn khai phá dữ liệu định nghĩa các kỹ thuật và thuật toán được sử dụng cho các vấn đề hỏi, ví dụ các kỹ thuật có thể được sử dụng trong giai đoạn này như mạng nơtron, tập thô, thuật toán di truyền, các mô hình thống kê và xác suất. Sự lựa chọn các kỹ thuật đáng tin cậy, trong nhiều trường hợp, trên từng kiểu công việc đã được phát triển.

Trong suốt giai đoạn khai phá dữ liệu, nhiều kiến thức hữu ích đã thu được và đã được đánh giá cao trong các ứng dụng. Nhiều tác giả xem xét việc khai phá dữ liệu đồng nghĩa với khám phá tri thức trong cơ sở dữ liệu, trong ngữ cảnh của giai đoạn này, quá trình khám phá tri thức trong cơ sở dữ liệu thường được biết đến là Khai phá dữ liệu, trong nghiên cứu này, nó là Khai phá dữ liệu, phần con của khám phá tri thức trong cơ sở dữ liệu

Khai phá dữ liệu đã trở thành lĩnh vực quan trọng được nghiên cứu ngày càng tăng, và nó cũng được gọi là phát hiện các tri thức trong cơ sở dữ liệu, vì vậy kết quả thu được trong một quá trình trích rút các thông tin tiềm ẩn bên trong, không tầm thường mà trước đây chưa biết và có khả năng là thông tin hữu ích, chẳng hạn như quy luật tri thức, các ràng buộc, các quy luật từ dữ liệu trong cơ sở dữ liệu.

Giai đoạn hậu xử lý

Trong giai đoạn hậu xử lý, tri thức thu được trong giai đoạn khai phá dữ liệu sẽ được xử lý. Giai đoạn này không phải luôn luôn cần thiết, tuy nhiên khả năng hợp lệ hữu ích của tri thức được khám phá.

1.5.2 Tập thô trong khai phá dữ liệu

Tập thô đã làm sáng tỏ nhiều lĩnh vực nghiên cứu, nhưng hiếm khi tìm thấy phương pháp ứng dụng cho thế giới thực. Khai phá dữ liệu với tập thô là một quá trình đa giai đoạn bao gồm chủ yếu là: rời rạc hóa; rút gọn và sinh ra các luật quyết định trên tập huấn luyện; phân lớp trên tập mẫu. Lý thuyết tập thô, từ khi ra đời đã

được sử dụng rộng rãi trong khai phá dữ liệu, và có chức năng quan trọng trong việc biểu diễn, nghiên cứu và kết luận các tri thức không chắc chắn, đó là một công cụ mạnh với thiết lập hệ thống quyết định thông minh. Mục tiêu chính là làm xuất hiện các kỹ thuật tập thô như thế nào để có thể được sử dụng như là một cách tiếp cận vấn đề khai phá dữ liệu và trích rút tri thức.

1.5.3 Một số ứng dụng quan trọng của lý thuyết tập thô

Lý thuyết tập thô cung cấp phương pháp có hiệu quả được áp dụng trong nhiều ngành của trí tuệ nhân tạo, một trong những ưu điểm của lý thuyết tập thô là chương trình triển khai thực hiện phương pháp này dễ dàng có thể chạy trên các máy tính song song, nhưng vẫn còn một số vấn đề cần giải quyết. Gần đây, rất nhiều nghiên cứu đã được thực hiện trong tập thô kết hợp với các phương pháp trí tuệ nhân tạo như logic mờ, Mạng nơtron, và hệ chuyên gia và một số kết quả quan trọng đã được tìm thấy. Lý thuyết tập thô cho phép mô tả đặc tính của một tập các đối tượng trong nhóm các giá trị của thuộc tính; tìm ra toàn bộ hoặc một phần phụ thuộc giữa các thuộc tính; giảm thuộc tính thừa; tìm thấy các thuộc tính có ý nghĩa và sinh ra các luật quyết định.

Các ứng dụng của tập thô đã giải quyết những vấn đề phức tạp, và do đó đã tạo nên sự hấp dẫn cho các nhà nghiên cứu trong những năm gần đây và đã được áp dụng thành công trong một số lĩnh vực đầy thách thức như phương pháp tính toán mềm. Phần này cung cấp một tổng quan ngắn gọn của một số các ứng dụng của tập thô. Một số thuộc tính của tập thô đã làm cho lý thuyết này là một sự lựa chọn hiển nhiên cho sử dụng trong các xử lý đối với những vấn đề thực tế:

Nhận dạng mẫu

Nhận dạng mẫu sử dụng tập thô là một trong những lĩnh vực ứng dụng thành công. Năm 2001 A. Mrozek và K. Cyran (2001) đề xuất một phương pháp lai của nhận dạng mẫu nhiễu xạ tự động dựa trên lý thuyết tập thô và mạng Nơtron. Trong phương pháp mới này, tập thô được sử dụng để xác định hàm mục tiêu và giải thuật tiến hóa ngẫu nhiên cho tìm kiếm không gian của trích rút đặc trưng, còn mạng nơtron được sử dụng cho mô hình hệ thống chưa chắc chắn. Các đặc trưng thu được cuối cùng là mẫu tối ưu từ các mẫu nhiễu xạ được nhập vào để phân loại theo ngữ nghĩa và thuật toán nhận dạng mẫu được thực hiện với các tiêu chuẩn tối ưu và tính toán chuẩn - tạo ra kỹ thuật tạo ảnh ba chiều (Holograms).

Phân tích âm thanh

Tập thô đã được áp dụng cho việc đánh giá âm thanh cho các phòng hoà nhạc. Thuật toán tập thô được áp dụng với bảng quyết định có chứa các thông số chất lượng chủ quan và các kết quả trên toàn bộ sở thích chủ quan của các đối tượng âm thanh được mô tả bởi các tham số. Hàm thành viên mờ vạch ra bản đồ kết quả kiểm tra đến gần đúng các tham số phân phối thử nghiệm, được xác định trên cơ sở xem xét thực nghiệm chủ quan riêng của tham số cá nhân tiềm ẩn trên toàn bộ sở thích. Một hệ thống nguyên mẫu dựa trên lý thuyết tập thô được sử dụng để tạo ra các quy tắc tổng quát mô tả mối quan hệ giữa các thông số âm thanh của các phòng hoà nhạc và các thuật toán xử lý âm thanh (Kotek, 1999).

Phân tích sức mạnh của hệ thống an ninh

Tập thô là một phương pháp tiếp cận sử dụng để giúp đỡ các kỹ sư kiến thức trong quá trình trích rút các sự kiện và các quy tắc của một tập các mẫu về những vấn đề sức mạnh hoạt động của hệ thống. Cách tiếp cận này mô tả việc giảm số lượng các mẫu, cung cấp một tập các mẫu nhỏ gọn hơn cho người dùng (Lambert-Torres et al., 1999).

Phân tích mẫu khí tượng và không gian

Một số chuyên mục của nhóm Vết đen của Mặt Trời (sunspots) có liên quan với năng lượng mặt trời. Đài thiên văn xung quanh trái đất theo dõi tất cả vết đen của mặt Trời không thể nhìn thấy để phát hiện sớm các tia sáng, việc nhận dạng các Vết đen Mặt Trời và phân loại được xử lý tại phòng thí nghiệm một cách vất vả, nó có thể được tự động nếu có máy học thành công. Việc sử dụng một phương pháp tiếp cận theo cấp bậc thô dựa trên phương pháp học để phân loại Vết đen Mặt Trời. Nó cố gắng học lược đồ phân loại Zurich dựa trên tập thô – cây quyết định. Hệ thống kết quả đã được đánh giá trên sự trích rút sunspots từ các hình ảnh vệ tinh, với kết quả đầy hứa hẹn (Nguyễn et al, 2005.).

Một ứng dụng mới của lý thuyết tập thô để phân loại dữ liệu radar về khí tượng đã được giới thiệu. Dữ liệu dung tích radar được sử dụng để phát hiện các sự kiện bão, nguyên nhân của thời tiết khắc nghiệt. Phân loại các tế bào bão là một vấn đề khó khăn khi nó tiến triển phức tạp trong suốt tuổi thọ của chúng. Ngoài ra, chiều cao và tính không chính xác của dữ liệu có thể được ngăn ngừa. phương pháp tập

thô sử dụng để phân loại một số sự kiện khí tượng của cơn bão (Shen & Jensen, 2007).

Hệ thống điều khiển thông minh

Một lĩnh vực ứng dụng quan trọng của lý thuyết tập thô là hệ thống điều khiển thông minh đặc biệt là khi kết hợp với lý thuyết mờ (Xie et al., 2004).

Đo lường chất lượng của một tập con riêng lẻ

Thuật toán Ant Colony System và lý thuyết tập thô được đề xuất một cách tiếp cận lai để lựa chọn các đặc trưng, lý thuyết tập thô cung cấp một hàm heuristic để đo lường chất lượng của một tập hợp riêng lẻ. Nó đã được nghiên cứu ảnh hưởng của các tham số thiết lập cho vấn đề này, đặc biệt giảm việc tìm kiếm. Kết quả thử nghiệm cho thấy cách tiếp cận này theo phương pháp lai có khả năng lựa chọn các đặc trưng (Anh et al., 2007).

Có nhiều khả năng cho sự phát triển của các phương pháp dựa trên lý thuyết tập thô như phân tích bất chuẩn, thống kê không tham số và định tính.

1.6 Kết luận chương 1

Trong chương này đã trình bày về lý thuyết tập thô, được đề xuất năm 1982 bởi Z. Pawlak, hệ thống hóa các kiến thức cơ bản của lý thuyết tập thô đã được trình bày trên từng ví dụ minh họa cụ thể. Trình bày về Thuật toán tìm tập rút gọn của một bảng quyết định dựa vào ma trận phân biệt được; các ví dụ cụ thể để minh họa từng bước thuật toán trên.

Lý thuyết tập thô đã tỏ ra thực sự hiệu quả hiệu quả trong lĩnh vực khai phá tri thức, những bài toán thực tế có dữ liệu ở dạng thô chưa qua xử lý, trong dữ liệu có nhiều thông tin dư thừa.

CHƯƠNG 2

CÁC PHƯƠNG PHÁP XÂY DỰNG CÂY QUYẾT ĐỊNH

2.1. Khai phá dữ liệu với cây quyết định

2.1.1 Khái niệm

Trong lĩnh vực học máy, cây quyết định là một kiểu mô hình dự báo, nghĩa là một ánh xạ từ các quan sát về một sự vật/hiện tượng tới các kết luận về giá trị mục tiêu của sự vật/hiện tượng. Mỗi một nút trong tương ứng với một biến; đường nối giữa nó với nút con của nó thể hiện một giá trị cụ thể cho biến đó. Mỗi nút lá đại diện cho giá trị dự đoán của biến mục tiêu, cho trước các giá trị của các biến được biểu diễn bởi đường đi từ nút gốc tới nút lá đó. Kỹ thuật học máy dùng trong cây quyết định được gọi là học bằng cây quyết định, hay chỉ gọi với cái tên ngắn gọn là cây quyết định.

Học bằng cây quyết định cũng là một phương pháp thông dụng trong khai phá dữ liệu. Khi đó, cây quyết định mô tả một cấu trúc cây, trong đó, các lá đại diện cho các phân loại còn cành đại diện cho các kết hợp của các thuộc tính dẫn tới phân loại đó. Một cây quyết định có thể được học bằng cách chia tập hợp nguồn thành các tập con dựa theo một kiểm tra giá trị thuộc tính. Quá trình này được lặp lại một cách đệ quy cho mỗi tập con dẫn xuất. Quá trình đệ quy hoàn thành khi không thể tiếp tục thực hiện việc chia tách được nữa, hay khi một phân loại đơn có thể áp dụng cho từng phần tử của tập con dẫn xuất.

Một bộ phân loại rừng ngẫu nhiên (random forest) sử dụng một số cây quyết định để có thể cải thiện tỉ lệ phân loại.

Cây quyết định cũng là một phương tiện có tính mô tả dành cho việc tính toán các xác suất có điều kiện.

Cây quyết định có thể được mô tả như là sự kết hợp của các kỹ thuật toán học và tính toán nhằm hỗ trợ việc mô tả, phân loại và tổng quát hóa một tập dữ liệu cho trước.

2.1.2 Thiết kế cây quyết định

- **Xử lý dữ liệu**

Một tập dữ liệu thô bao gồm các mẫu dữ liệu ban đầu hay chưa biến đổi từ tổng thể. Hầu hết dữ liệu thô hữu ích biểu diễn một cách chính xác. Một kết hợp của

các mẫu thống kê và sự điều chỉnh của chuyên gia.

Trong thế giới thực, nói chung dữ liệu thô chắc chắn có mức độ nhiễu. Điều này có các nguyên nhân khác nhau như là dữ liệu lỗi, dữ liệu có đại lượng không chính xác, Do đó, chúng ta thường tiền xử lý (nghĩa là, “làm sạch”) để cực tiểu hoá hay huỷ bỏ tất cả dữ liệu thô bị nhiễu. Các giai đoạn tiền xử lý này cũng có thể biến đổi dữ liệu thô hiển thị hữu ích hơn, như hệ thống thông tin. Khi nhiều bước tiền xử lý ứng dụng hiệu quả, nó sẽ giúp cải tiến hiệu quả phân lớp.

- **Tạo cây**

Cây quyết định được tạo thành bằng cách lần lượt chia (đệ quy) một tập dữ liệu thành các tập dữ liệu con, mỗi tập con được tạo thành chủ yếu từ các phần tử của cùng một lớp.

Các nút (không phải là nút lá) là các điểm phân nhánh của cây. Việc phân nhánh tại các nút có thể dựa trên việc kiểm tra một hay nhiều thuộc tính để xác định việc phân chia dữ liệu.

- **Tiêu chuẩn tách**

Việc lựa chọn chủ yếu trong các thuật toán phân lớp dựa vào cây quyết định là chọn thuộc tính nào để kiểm tra tại mỗi nút của cây. Chúng ta mong muốn chọn thuộc tính sao cho việc phân lớp tập mẫu là tốt nhất. Như vậy chúng ta cần phải có một tiêu chuẩn để đánh giá vấn đề này. Có rất nhiều tiêu chuẩn được đánh giá được sử dụng đó là:

- + Lượng thông tin thu thêm IG (Information Gain, thuật toán ID3 của John Ross Quilan [5]).

- + Đánh giá thay đổi các giá trị của thuộc tính RatioGain (RatioGain, thuật toán C4.5).

Các tiêu chuẩn trên sẽ được trình bày trong các thuật toán xây dựng cây quyết định.

- **Tiêu chuẩn dừng**

Đây là phần quan trọng trong cấu trúc phân lớp của cây quyết định nhằm chia một nút thành các nút con.

Chúng ta tập trung một số tiêu chuẩn dừng chung nhất được sử dụng trong cây quyết định. Tiêu chuẩn dừng truyền thống sử dụng các tập kiểm tra. Chúng ta

kiểm tra cây quyết định trong suốt quá trình xây dựng cây với tập kiểm tra và dùng thuật toán khi xảy ra lỗi. Một phương pháp khác sử dụng giá trị ngưỡng cho trước để dừng chia nút. Chúng ta có thể thay ngưỡng như là giảm nhiều, số các mẫu trong một nút, tỉ lệ các mẫu trong nút, hay chiều sâu của cây, ...

- **Tỉa cây**

Trong giai đoạn tạo cây chúng ta có thể giới hạn việc phát triển của cây bằng số bản tin tối thiểu tại mỗi nút, độ sâu tối đa của cây hay giá trị tối thiểu của lượng thông tin thu thêm.

Sau giai đoạn tạo cây chúng ta có thể dùng phương pháp “Độ dài mô tả ngắn nhất” (Minimum Description Length) hay giá trị tối thiểu của IG để tỉa cây (chúng ta có thể chọn giá trị tối thiểu của IG trong giai đoạn tạo cây đủ nhỏ để cho cây phát triển tương đối sâu, sau đó lại nâng giá trị này lên để tỉa cây).

2.2. Phương pháp tổng quát xây dựng cây quyết định

Quá trình xây dựng một cây quyết định cụ thể bắt đầu bằng một nút rỗng bao gồm toàn bộ các đối tượng huấn luyện và làm như sau [2].

1. Nếu tại nút hiện thời, tất cả các đối tượng huấn luyện đều thuộc vào một lớp nào đó thì cho nút này thành nút lá có tên là nhãn lớp chung của các đối tượng.
2. Trường hợp ngược lại, sử dụng một độ đo, chọn thuộc tính điều kiện phân chia tốt nhất tập mẫu huấn luyện có tại nút.
3. Tạo một lượng nút con của nút hiện thời bằng số các giá trị khác nhau của thuộc tính được chọn. Gán cho mỗi nhánh từ nút cha đến nút con một giá trị của thuộc tính rồi phân chia các đối tượng huấn luyện vào các nút con tương ứng.
4. Nút con t được gọi là thuần nhất, trở thành lá, nếu tất cả các đối tượng mẫu tại đó đều thuộc vào cùng một lớp. Lặp lại các bước 1-3 đối với mỗi nút chưa thuần nhất.

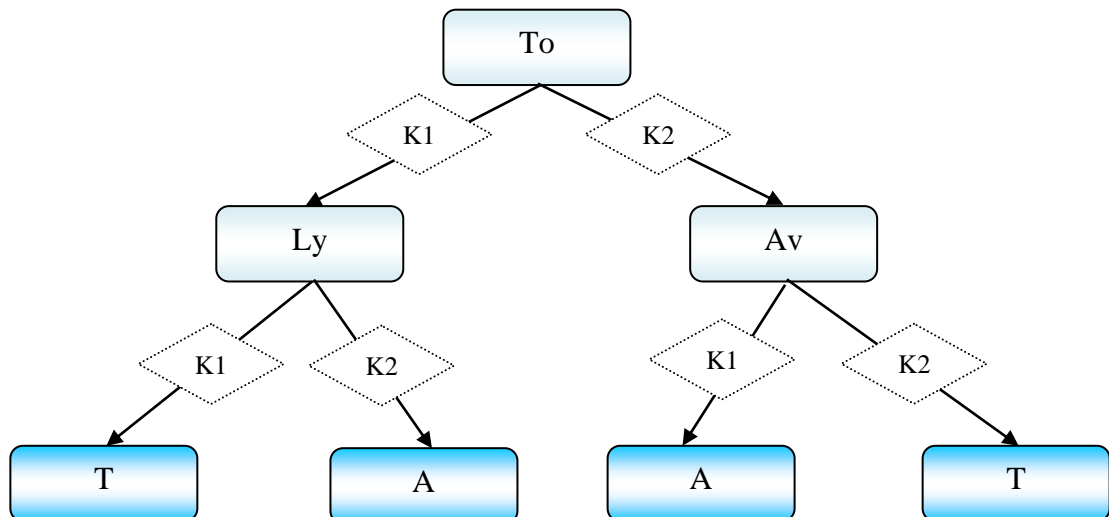
Ví dụ 2.1 Xây dựng một cây quyết định như sau:

Cho Bảng 2.1 biểu diễn thông tin về 7 đối tượng u_1, \dots, u_7 . Bảng 2.1 là một bảng quyết định với tập thuộc tính điều kiện $C = \{Ly, To, Nv, Av\}$ và thuộc tính quyết định là $d = \{Tc\}$.

Bảng 2.1 Bảng quyết định minh họa Ví dụ 2.1

U	Ly	To	Nv	Av	Tc
u ₁	K2	K1	XS	K1	A
u ₂	K2	K2	G	K2	T
u ₃	K2	K2	TB	K2	T
u ₄	K1	K1	K1	K1	T
u ₅	K1	K1	TB2	K2	T
u ₆	K2	K1	TB	K2	A
u ₇	K1	K2	K2	K1	A

Từ Bảng 2.1 ta có thể vẽ cây quyết định như Hình 2.1.

**Hình 2.1** Ví dụ cây quyết định ứng với bảng quyết định 2.1

Cây quyết định của ví dụ trên có thể được giải thích như sau: Các nút lá chứa các giá trị của thuộc tính quyết định hay thuộc tính phân lớp (thuộc tính “Tc”). Các nút con tương ứng với các thuộc tính khác thuộc tính điều kiện hay thuộc tính phân lớp; nút gốc cũng được xem như một nút con đặc biệt, ở đây chính là thuộc tính “To”. Các nhánh của cây từ một nút bất kỳ tương ứng với một giá trị của thuộc tính điều kiện được chọn. Lưu ý cây quyết định trên không có sự tham gia của thuộc tính “Nv” trong thành phần cây, các thuộc tính như vậy được gọi chung là các thuộc tính dư thừa bởi vì các thuộc tính này không ảnh hưởng đến quá trình xây dựng mô hình của cây.

Trong các thuật toán cơ sở xây dựng cây quyết định chỉ chấp nhận các thuộc tính tham gia vào quá trình phân lớp có giá trị rời rạc, bao gồm cả thuộc tính được dùng để dự đoán trong quá trình học cũng như các thuộc tính được sử dụng để kiểm tra tại mỗi nút của cây. Do đó trong trường hợp các thuộc tính có giá trị liên tục có thể dễ dàng loại bỏ bằng cách phân mảnh tập giá trị liên tục của thuộc tính thành một tập rời các khoảng.

Việc xây dựng cây quyết định được tiến hành một cách đệ quy, lần lượt từ nút gốc xuống tới tận các nút lá. Tại mỗi nút hiện hành đang xét, nếu kiểm tra thấy thỏa điều kiện dừng: thuật toán sẽ tạo nút lá. Nút này được gán một giá trị của nhãn lớp tùy điều kiện dừng được thỏa. Ngược lại, thuật toán tiến hành chọn điểm chia tốt nhất theo một tiêu chí cho trước, phân chia dữ liệu hiện hành theo điều kiện chia này.

Sau bước phân chia trên, thuật toán sẽ lặp qua tất cả các tập con (đã được chia) và tiến hành gọi đệ quy như bước đầu tiên với dữ liệu chính là các tập con này.

Trong bước 3, tiêu chuẩn sử dụng lựa chọn thuộc tính được hiểu là một số đo độ phù hợp, một số đo đánh giá độ thuần nhất, hay một quy tắc phân chia tập mẫu huấn luyện.

Vấn đề then chốt trong quá trình xây dựng cây quyết định là việc lựa chọn thuộc tính điều kiện kiểm tra tại mỗi nút (gọi tắt là chọn nút). Có nhiều phương pháp chọn nút dựa trên những tiêu chuẩn khác nhau đánh giá độ quan trọng của các thuộc tính. Có rất nhiều tiêu chuẩn thường được sử dụng để xây dựng cây quyết định, nhưng trong luận văn đề cập đến là dựa vào Entropy và tập thô, các tiêu chuẩn này được đề cập cụ thể trong từng thuật toán ở bên dưới.

2.3. Phương pháp xây dựng cây quyết định ID3

2.3.1. Tiêu chí lựa chọn thuộc tính để phân lớp

Như phần trên đã phân tích thì các tiêu chí để đánh giá tìm điểm chia là rất quan trọng, chúng được xem là một tiêu chuẩn “heuristic” để phân chia dữ liệu. Ý tưởng chính trong việc đưa ra các tiêu chí trên là làm sao cho các tập con được phân chia càng trở nên “trong suốt” (tất cả các bộ thuộc về cùng một nhãn) càng tốt.

Thuật toán dùng độ đo lượng thông tin thu thêm (information gain - IG) để xác định điểm chia [5]. Độ đo này dựa trên cơ sở lý thuyết thông tin của nhà toán học Claude Shannon, độ đo này được xác như sau:

Xét bảng quyết định $DT = (U, C \cup \{d\})$, số giá trị (nhãn lớp) có thể của d là k . Khi đó Entropy của tập các đối tượng trong T được định nghĩa bởi:

$$Entropy(DT) = -\sum_{i=1}^k p_i \log_2 p_i$$

Trong đó p_i là tỉ lệ các đối tượng trong DT mang nhãn lớp i .

Lượng thông tin thu thêm (IG) là lượng Entropy còn lại khi tập các đối tượng trong T được phân hoạch theo một thuộc tính điều kiện c nào đó. IG xác định theo công thức sau:

$$IG(DT, c) = Entropy(DT) - \sum_{v \in V_c} \frac{|DT_v|}{|DT|} Entropy(DT_v)$$

Trong đó V_c là tập các giá trị của thuộc tính c , DT_v là tập các đối tượng trong DT có giá trị thuộc tính c bằng v . $IG(DT, c)$ được John Ross Quinlan [5] sử dụng làm độ đo lựa chọn thuộc tính phân chia dữ liệu tại mỗi nút trong thuật toán xây dựng cây quyết định ID3. Thuộc tính được chọn là thuộc tính cho lượng thông tin thu thêm lớn nhất.

2.3.2. Thuật toán ID3

Thuật toán ID3 – Iterative Dichotomiser 3 là thuật toán dùng để xây dựng cây quyết định được John Ross Quinlan trình bày. Ý tưởng chính của thuật toán ID3 là để xây dựng cây quyết định bằng cách ứng dụng từ trên xuống chiến lược tham lam thông qua các tập đã cho để kiểm tra từng thuộc tính ở mọi nút của cây. Để chọn thuộc tính "tốt nhất" (để có cây tối ưu – có độ sâu nhỏ nhất), người ta phải tính IG thông qua Entropy của các thuộc tính điều kiện.

Dữ liệu vào: Bảng quyết định $DT = (U, C \cup \{d\})$

Dữ liệu ra: Mô hình cây quyết định

Thuật toán ID3 [5]

```

1.  TreeNode CreateTree(DT, C, {d})
2.  {
3.      if ( Nếu tất cả các mẫu cùng nhãn lớp  $d_i$  ) or (  $C == \text{null}$  )
4.          return (TreeNode( $d_i$ ));
5.      bestAttribute = getBestAttribute(DT,C);
6.      Root = TreeNode(bestAttribute);
7.      foreach (  $v$  in bestAttribute )
8.      {
9.           $DT_v = [DT]_v$ ;
10.          $C = C - \{\text{bestAttribute}\}$ ;
11.         ChildNode = CreateTree( $DT_v$ ,  $C$ , { $d$ });
12.         Root.AddTreeNode(ChildNode, $v$ );
13.     }
14.     return Root;
15. }
```

Giải thích thuật toán ID3.

+ Xét dòng thứ 3 nếu tất cả các mẫu huấn luyện U cùng lớp, tức có các mẫu này đều có giá trị giống nhau là d_i trên thuộc tính quyết định, thì thuật toán trả về nút lá có nhãn là d_i (dòng thứ 4).

+ Xét dòng thứ 5 tập thuộc tính điều kiện C là rỗng, thì thuật toán trả về nút lá có nhãn d_i là lớp phổ biến nhất trong DT (dòng thứ 6), tức nhãn lớp xuất hiện nhiều nhất trong tổng của từng giá trị nhãn lớp riêng biệt trong DT .

+ Xét dòng 7, nếu thuật toán chưa thỏa mãn điều kiện để dừng, tiếp tục xét bằng cách tìm kiếm thuộc tính điều kiện để phân chia tốt nhất. Để tìm thuộc tính điều kiện tốt nhất cần sử dụng một hàm getBestAttribute, kết quả của hàm này sẽ trả về thuộc tính điều kiện được chọn tương ứng có tên bestAttribute. Hàm getBestAttribute trả về thuộc tính điều kiện có giá trị IG lớn nhất trong DT .

+ Xét dòng 8, sau khi đã chọn được thuộc tính điều kiện phân chia tốt nhất `bestAttribute` thì gán nhãn cho nút gốc là `bestAttribute`.

+ Xét dòng 9, ứng với mỗi giá trị v của thuộc tính `bestAttribute`:

- Tiến hành phân chia bảng quyết định T thành các bảng quyết định DT_v (dòng thứ 11). DT_v là phân hoạch của DT theo thuộc tính điều kiện `bestAttribute` có giá trị là v .
- Cập nhật lại tập thuộc tính điều kiện C (dòng thứ 10).
- Xét dòng thứ 11 tạo nút con `ChildNode` của `Root` bằng cách gọi đệ quy hàm `CreateTree()` với các tham số tương ứng.
- Dòng 12 gắn nút con này vào gốc `Root` của cây tương ứng với giá trị của thuộc tính điều kiện `bestAttribute` bằng v .

Mã giả của hàm `getBestAttribute` như sau:

Dữ liệu vào: Bảng quyết định $DT = (U, C \cup \{d\})$

Dữ liệu ra: Thuộc tính điều kiện tốt nhất.

```
getBestAttribute(DT,C)
{
    maxIG = 0;
    foreach ( $c_i$  in  $C$ )
    {
        temp = IG(DT, $c_i$ );
        //Trả về lượng thông tin thu thêm IG(DT, $c_i$ )
        if (temp > maxIG)
        {
            maxIG= temp;
            result =  $c_i$ ;
        }
    }
    return result;
}
```

Ví dụ 2.2: Xét bảng quyết định $DT = \{U, C \cup \{d\}\}$ cho trong bảng 2.2.

Bảng 2.2 Bảng quyết định minh họa thuật toán ID3.

	To	Ly	Nv	Av	Tc
u_1	K2	G	K2	K2	A
u_2	K2	G	K2	K1	A
u_3	K1	G	K2	K2	T
u_4	G	K1	K2	K2	T
u_5	G	K2	K1	K2	T
u_6	G	K2	K1	K1	A
u_7	K1	K2	K1	K1	T
u_8	K2	K1	K2	K2	A
u_9	K2	K2	K1	K2	T
u_{10}	G	K1	K1	K2	T
u_{11}	K2	K1	K1	K1	T
u_{12}	K1	K1	K2	K1	T
u_{13}	K1	G	K1	K2	T
u_{14}	G	K1	K2	K1	A

Giải thích cơ sở dữ liệu Bảng 2.2: Để tiện lợi ta xem tất cả các thuộc tính đều có kiểu dữ liệu rời rạc. Thuộc tính nhãn lớp tức thuộc tính “Tc” chỉ có hai giá trị là “T” và “A”, như vậy có chín bộ dữ liệu có nhãn lớp là giá trị “T” và năm bộ giá trị “A”.

Thuật toán xây dựng cây quyết định như sau:

- Đầu tiên nút gốc được khởi tạo gồm các mẫu từ u_1 đến u_{14} .

Để tìm điểm chia tốt nhất, phải tính toán chỉ số IG của tất cả các thuộc tính trên. Đầu tiên sẽ tính Entropy cho toàn bộ tập huấn luyện U gồm chín bộ $\{u_3, u_4, u_5, u_7, u_9, u_{10}, u_{11}, u_{12}, u_{13}\}$ có giá trị thuộc tính nhãn là “T” và năm bộ $\{u_1, u_2, u_6, u_8, u_{14}\}$ có thuộc tính nhãn là “A”:

$$\text{Entropy}(DT_G) = \frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14} = 0.940$$

Kế tiếp tính IG cho từng thuộc tính, bắt đầu với thuộc tính “To”. Thuộc tính này có ba giá trị là “G”, “K1” và “K2”. Nhìn vào bảng dữ liệu 2.2, với giá trị “K2” có hai bộ $\{u_9, u_{11}\}$ có giá trị thuộc tính nhãn là “T” và ba bộ $\{u_1, u_2, u_8\}$ giá trị thuộc tính nhãn là “A”. Tương tự giá trị “K1” có bốn bộ $\{u_3, u_7, u_{12}, u_{13}\}$ có nhãn lớp là “T” và không có bộ nào có nhãn lớp là “A”; với giá trị “G” có ba bộ $\{u_4, u_5, u_{10}\}$ nhãn lớp “T” và hai bộ $\{u_6, u_{14}\}$ có nhãn lớp “A”. Theo công thức trên, độ đo lượng thông tin thu thêm của thuộc tính “To” xét trên DT là:

$$IG(DT, To) = Entropy(DT) - \sum_{v \in V_{To}} \frac{|DT_v|}{|DT|} Entropy(DT_v)$$

$$= 0.940 - \left[\frac{5}{14} \left(-\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} \right) + \frac{4}{14} \left(-\frac{4}{4} \log_2 \frac{4}{4} \right) + \frac{5}{14} \left(-\frac{3}{5} \log_2 \frac{3}{5} - \frac{2}{5} \log_2 \frac{2}{5} \right) \right] = 0.247$$

Theo cách tính tương tự như trên, tính chỉ số IG cho lần lượt các thuộc tính “Ly”, “Nv” và “Av” Kết quả sẽ là:

$$IG(DT, Ly) = 0.029;$$

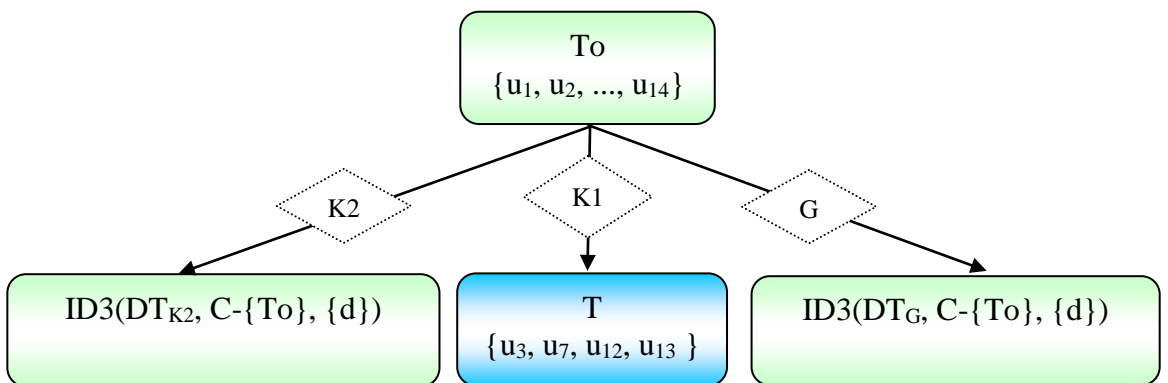
$$IG(DT, Nv) = 0.152;$$

$$IG(DT, Av) = 0.048;$$

Như vậy, thuộc tính “To” là thuộc tính có chỉ số IG lớn nhất nên sẽ được chọn là thuộc tính phân chia. Vì thế thuộc tính “To” được chọn làm nhãn cho nút gốc, ba nhánh được tạo ra lần lượt với tên là “G”, “K1”, “K2”.

Hơn nữa nhánh “K1” có các mẫu $\{u_3, u_7, u_{12}, u_{13}\}$ cùng thuộc một lớp “T” nên nút lá được tạo ra với nhãn là “T”.

Kết quả phân chia sẽ là cây quyết định như sau:



Hình 2.2 cây quyết định bước đầu ví dụ 2.2

- Bước tiếp theo gọi thuật toán đệ quy: $ID3(DT_{K2}, C-\{To\}, \{d\})$

Ta có DT_{K2} gồm có các mẫu $\{u_1, u_2, u_8, u_9, u_{11}\}$

Tương tự để tìm điểm chia tốt nhất tại thuật toán này, phải tính toán chỉ số IG của các thuộc tính “Ly”, “Nv” và “Av”. Đầu tiên ta cũng tính Entropy cho toàn bộ tập huấn luyện trong DT_{K2} gồm hai bộ $\{u_9, u_{11}\}$ có thuộc tính nhãn là “T” và ba bộ $\{u_1, u_2, u_8\}$ có thuộc tính nhãn là “A”:

$$Entropy(T_{K2}) = -\frac{2}{5}\log_2 \frac{2}{5} - \frac{3}{5}\log_2 \frac{3}{5} = 0.971$$

Tiếp theo tính IG cho thuộc tính “Ly” thuộc tính này có ba giá trị “G”, “K1” và “K2”. Nhìn vào bảng dữ liệu trên, với giá trị “K2” có một bộ $\{u_9\}$ có giá trị thuộc tính nhãn là “T” và không bộ có giá trị thuộc tính nhãn là “A”. Tương tự giá trị “K1” có một bộ $\{u_{11}\}$ có nhãn lớp là “T” và một bộ $\{u_8\}$ có nhãn lớp là “A”; với giá trị “G” có không bộ có nhãn lớp “T” và hai bộ $\{u_1, u_2\}$ có nhãn lớp “A”. Theo công thức trên, độ đo lượng thông tin thu thêm của thuộc tính “Ly” xét trên DT_{K2} là:

$$\begin{aligned} IG(DT_{K2}, Ly) &= Entropy(DT_{K2}) - \sum_{v \in V_{Ly}} \frac{|DT_{K2_v}|}{|DT_{K2}|} Entropy(DT_{K2_v}) \\ &= 0.971 - \left[\frac{1}{5} \left(-\frac{1}{1} \log_2 \frac{1}{1} \right) + \frac{2}{5} \left(-\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{2} \log_2 \frac{1}{2} \right) + \frac{2}{5} \left(-\frac{2}{2} \log_2 \frac{2}{2} \right) \right] = 0.571 \end{aligned}$$

Tính tương tự ta cũng có:

$$IG(DT_{K2}, Nv) = 0.971, \quad IG(DT_{K2}, Av) = 0.020.$$

Vì vậy, ta chọn thuộc tính “Nv” làm nhãn cho nút bên trái nối với nhánh “K2”. Với thuộc tính này có hai giá trị “T”, “A” nên ta tiếp tục ta tạo thành hai nhánh mới là “T” và “A”. Ứng với nhánh “T” gồm các mẫu $\{u_9, u_{11}\}$ cùng có giá trị quyết định là “T” nên tạo nút lá là “T”. Tương tự với nhánh “A” gồm các mẫu $\{u_1, u_2, u_8\}$ nên tạo thêm nút lá là “A”.

- Đối với nút nối với nhánh “G”, ta gọi thuật toán đệ quy $ID3(T_G, C-\{To\}, \{d\})$

Tương tự, ta có:

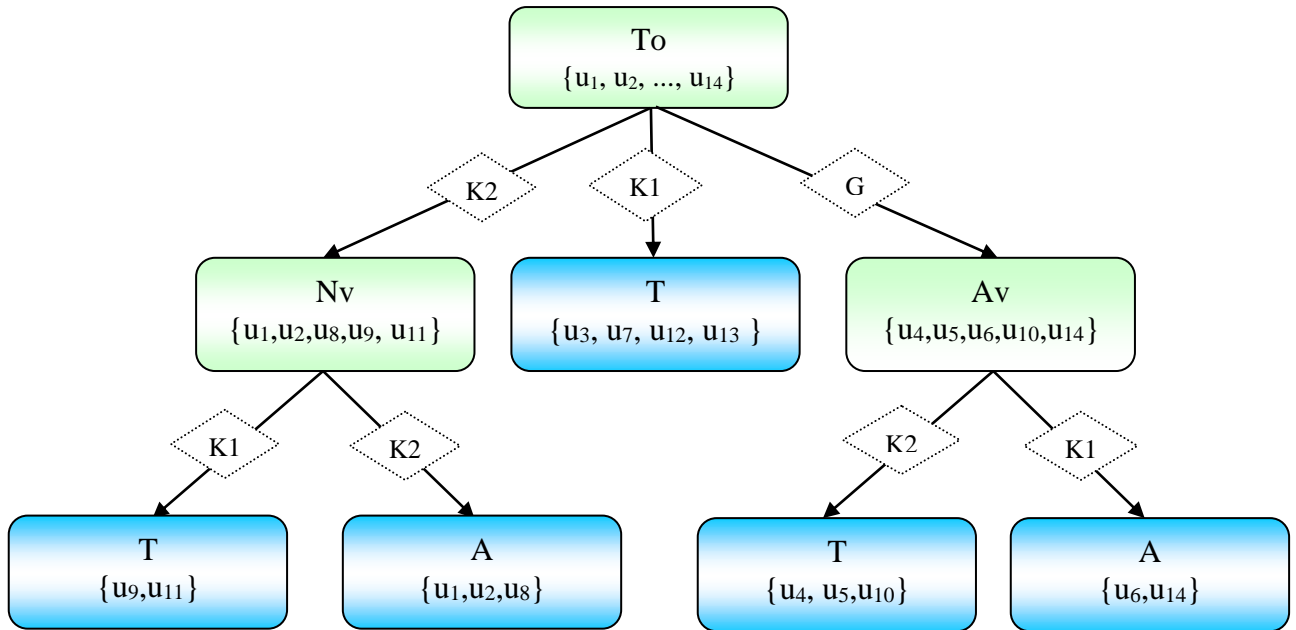
$$Entropy(DT_G) = 0.971$$

$$IG(DT_G, Ly) = 0.02$$

$$IG(DT_G, Nv) = 0.02$$

$$IG(DT_G, Av) = 0.971$$

Do đó thuộc tính “Av” có độ đo lượng thông tin thu thêm IG lớn nhất, vì thế ta chọn “Av” làm nhãn cho nút này. Kết quả cuối cùng ta có cây quyết định như Hình 2.2.



Hình 2.3: Cây quyết định được xây dựng theo thuật toán ID3 ứng với Bảng quyết định 2.2

2.3.3 Độ phức tạp tính toán

Giả sử tổng số mẫu là n và tổng số thuộc tính là a . Trong trường hợp xấu nhất, chiều cao tối đa của cây quyết định từ gốc đến mỗi nút lá là a , do đó tổng số nút của cây quyết định ít hơn $a*n$. Tại nút gốc, thuật toán yêu cầu bằng việc chia mỗi mẫu đối với mỗi thuộc tính c để có $IG(DT, c)$, thời gian của quá trình xử lý là $a*n$ và thời gian tại các nút khác không ít hơn nút gốc. Vì thế trong trường hợp xấu nhất độ phức tạp tính toán của thuật toán là $O(a*n*a*n)$. Do đó độ phức tạp của thuật toán là $T(n) = O(a^2*n^2)$.

2.4 Phương pháp xây dựng cây quyết định C4.5

2.4.1 Giới thiệu

- Trong các thuật toán học cây quyết định thì ID3 là thuật toán phổ dụng nhất. Nhưng thuật toán ID3 còn có các hạn chế sau đây:

- + Trong thuật toán ID3, giá trị thuộc tính là rời rạc, trong khi thế giới thực còn tồn tại cả thuộc tính có giá trị liên tục.

- + Trong thuật toán ID3, nếu các thuộc tính có nhiều giá trị mà mỗi giá trị lại duy nhất, sẽ dẫn tới tạo cây phức tạp, không đưa ra được quyết định cho các trường hợp trong thực tế.

- Thuật toán C4.5 là sự mở rộng của giải thuật ID3 trên một số khía cạnh sau đây:

- + Cho phép dữ liệu đầu vào của các thuộc tính là liên tục
- + Cho phép thao tác với các thuộc tính có dữ liệu không xác định
- + Đưa ra phương pháp cắt tỉa cây và giảm lược các luật để phù hợp với những bộ dữ liệu lớn

2.4.2 Xác định điểm chia tốt nhất

Ngoài việc sử dụng Entropy và IG thuật toán C4.5 còn sử dụng độ đo thông tin tiềm ẩn (*SplitInformation*) được tạo ra bằng cách chia tập dữ liệu trong một số tập con nào đó và độ đo đánh giá sự thay đổi các giá trị của thuộc tính (*RatioGain*)

$$\text{SplitInformation}(DT, c) = - \sum_{i=1}^k \frac{|DT_v|}{|DT|} \log_2 \frac{|DT_v|}{|DT|}$$

$$\text{RatioGain}(DT, c) = \frac{IG(DT, c)}{\text{SplitInformation}(DT, c)}$$

Tất cả các thuộc tính sẽ được tính toán độ đo *RatioGain*, thuộc tính nào có độ đo *RatioGain* lớn nhất sẽ được chọn làm thuộc tính phân chia.

2.4.3 Một số vấn đề với thuộc tính

Thuộc tính liên tục:

Thuật toán ID3 bị giới hạn bởi việc liên quan đến tập những giá trị rời rạc. Trong thuật toán C4.5 sẽ mở rộng phạm vi hoạt động cho những thuộc tính có giá trị liên tục để phù hợp với yêu cầu thực tế.

Bảng 2.3 Tập dữ liệu có giá trị liên tục

Quang cảnh	Nhiệt độ	Độ ẩm	Gió	Chơi tennis
Nắng	Nóng	85	Nhẹ	Không
Nắng	Nóng	90	Mạnh	Không
Âm u	Nóng	78	Nhẹ	Có
Mưa	Ấm áp	96	Nhẹ	Có
Mưa	Mát	80	Nhẹ	Có
Mưa	Mát	70	Mạnh	Không
Âm u	Mát	65	Mạnh	Có
Nắng	Ấm áp	95	Nhẹ	Không
Nắng	Mát	70	Nhẹ	Có
Mưa	Ấm áp	80	Nhẹ	Có
Nắng	Ấm áp	70	Mạnh	Có
Âm u	Ấm áp	90	Mạnh	Có
Âm u	Nóng	75	Nhẹ	Có
Mưa	Ấm áp	80	Mạnh	Không

Thuật toán C4.5 đưa ra định nghĩa những giá trị rời rạc mới để phân những giá trị liên tục thành những thuộc tính tượng trưng theo qui tắc sau:

- Dựa trên một giá trị nếu muốn phân chia nhị phân.
- Dựa trên vài giá trị nếu muốn có nhiều nhánh.
- Với mỗi giá trị các mẫu thuộc một lớp theo dạng $C \leq v$ và $C > v$.
- Cách chọn giá trị v hiệu quả:
 - + Chọn giá trị trung bình từng cặp giá trị của thuộc tính để phân chia và tính chỉ số gia lượng thông tin
 - + Chọn giá trị phân chia có chỉ số IG cao nhất

Ví dụ 2.3 Chọn giá trị trung bình từng cặp giá trị của thuộc tính để phân chia và tính chỉ số gia lượng thông tin. Từ Bảng 2.3 ta tính độ đo lượng thông tin thu thêm $IG(DT_{\text{độ ẩm}}, \text{độ ẩm} = 67.5)$ như sau:

$$Entropy(DT) = -\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14} = 0.940$$

$$Entropy_{độ ẩm=67.5}(DT_{độ ẩm}) = \frac{1}{14} Entropy(DT_{độ ẩm \leq 67.5}) + \frac{13}{14} Entropy(DT_{độ ẩm > 67.5})$$

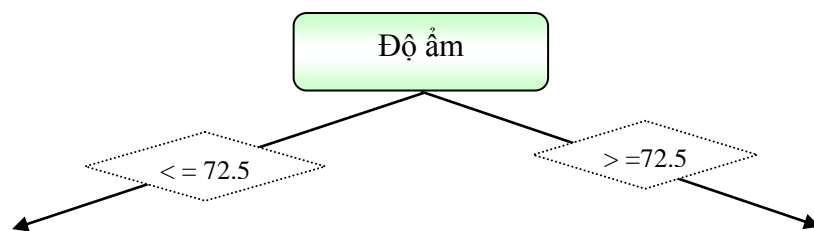
$$= \frac{1}{14} \cdot 0 + \frac{13}{14} \left(-\frac{8}{13} \log_2 \frac{8}{13} - \frac{5}{13} \log_2 \frac{5}{13} \right) = 0.839$$

$$IG(DT_{độ ẩm}, độ ẩm = 67.5) = 0.940 - 0.839 = 0.101$$

Tính tương tự cho các giá trị còn lại

Độ ẩm																
	65		70		75		78		80		85		90		95	
	67.5		72.5		76.5		79.5		82.5		87.5		92.5		95.5	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Có	1	8	3	6	4	5	5	4	7	2	3	2	8	1	8	1
Không	0	5	1	4	1	4	1	4	2	3	7	2	4	1	5	0
IG	0.047		0.646		0.045		0.090		0.102		0.025		0.010		0.047	

Như vậy giá trị chọn để phân chia là 72.5



Hình 2.4 Minh họa phân chia thuộc tính liên tục

Thuộc tính nhiều giá trị:

Thuật toán ID3 bị giới hạn bởi việc liên quan đến những thuộc tính có nhiều giá trị, mà các giá trị này lại duy nhất. Khi đó, việc chia một tập dữ liệu thành quá

nhieu các tập con dẫn đến số lượng các lớp tại mỗi nút giảm và do đó Entropy trên thuộc tính đó cũng giảm theo, nên IG sẽ cao hơn các thuộc tính khác. Vì vậy thuộc tính này sẽ được chọn thường xuyên để tách, dẫn đến độ phân nhánh lớp, cây sẽ rất lớn và phức tạp.

Ví dụ 2.3 ta thêm thuộc tính ngày vào Bảng 2.3 ta được bảng sau:

Bảng 2.4 Dữ liệu chứa thuộc tính có nhiều giá trị

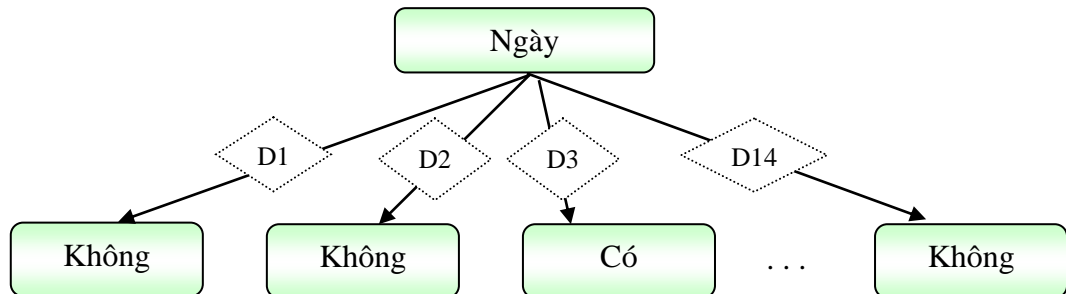
Ngày	Quang cảnh	Nhiệt độ	Độ ẩm	Gió	Chơi tennis
D1	Nắng	Nóng	85	Nhẹ	Không
D2	Nắng	Nóng	90	Mạnh	Không
D3	Âm u	Nóng	78	Nhẹ	Có
D4	Mưa	Ấm áp	96	Nhẹ	Có
D5	Mưa	Mát	80	Nhẹ	Có
D6	Mưa	Mát	70	Mạnh	Không
D7	Âm u	Mát	65	Mạnh	Có
D8	Nắng	Ấm áp	95	Nhẹ	Không
D9	Nắng	Mát	70	Nhẹ	Có
D10	Mưa	Ấm áp	80	Nhẹ	Có
D11	Nắng	Ấm áp	70	Mạnh	Có
D12	Âm u	Ấm áp	90	Mạnh	Có
D13	Âm u	Nóng	75	Nhẹ	Có
D14	Mưa	Ấm áp	80	Mạnh	Không

$$\text{Entropy}_{\text{Ngày}}(DT) = \frac{1}{14} \text{Entropy}(DT_{D1}) + \frac{1}{14} \text{Entropy}(DT_{D2}) + \dots + \frac{1}{14} \text{Entropy}(DT_{D14})$$

$$\text{Entropy}(DT_{D1}) = \text{Entropy}(DT_{D2}) = \dots = \text{Entropy}(DT_{D14}) = 0 \rightarrow \text{Entropy}_{\text{Ngày}}(DT) = 0$$

$$\text{IG}(DT, \text{Ngày}) = \text{Entropy}(DT) - \text{Entropy}_{\text{Ngày}}(DT) = 0.940$$

Lúc này thuộc tính ngày có độ đo IG cao nhất so với các thuộc tính khác trong tập dữ liệu. Nó sẽ được chọn làm thuộc tính phân tách. Kết quả phép tách trên thuộc tính “Ngày”



Hình 2.5 Minh họa phân chia thuộc tính nhiều giá trị

Vấn đề của thuộc tính “Ngày”:

Thuộc tính “Ngày” có nhiều nhất những giá trị trong việc phân chia tập dữ liệu huấn luyện thành những tập nhỏ. Cũng chính vì điều này nó sẽ có thu thập thông tin rất cao liên quan đến tập dữ liệu huấn luyện. Tuy nhiên nó lại là một công cụ tiên đoán tồi của hàm mục tiêu.

Giải quyết vấn đề của thuộc tính “Ngày”: Lựa chọn thuộc tính để phân tách theo nguyên tắc :

- Thuộc tính có tỉ lệ gia lượng thông tin RatioGain cao.
- Entropy của thuộc tính lớn hơn Entropy trung bình của tất cả các thuộc tính.

Thuộc tính thiếu giá trị

Nếu giá trị của thuộc tính C_i bị mất trên một số bộ dữ liệu, hướng giải quyết. Giả sử rằng $(x, C(x))$ là một trong những tập huấn luyện trong DT và giá trị $C(x)$ là không được biết đến. Giải pháp là thay bằng giá trị xuất hiện nhiều nhất của thuộc tính C_i

Sử dụng Bảng 2.3 và thêm mẫu dữ liệu mới có thuộc tính “Gió” thiếu giá trị ta được:

Bảng 2.4 Dữ liệu chứa thuộc tính thiếu giá trị

Quang cảnh	Nhiệt độ	Độ ẩm	Gió	Chơi tennis
Nắng	Nóng	85	Nhẹ	Không
Nắng	Nóng	90	Mạnh	Không
Âm u	Nóng	78	Nhẹ	Có
Mưa	Ấm áp	96	Nhẹ	Có
Mưa	Mát	80	Nhẹ	Có
Mưa	Mát	70	Mạnh	Không
Âm u	Mát	65	Mạnh	Có
Nắng	Ấm áp	95	Nhẹ	Không
Nắng	Mát	70	Nhẹ	Có
Mưa	Ấm áp	80	Nhẹ	Có
Nắng	Ấm áp	70	Mạnh	Có
Âm u	Ấm áp	90	Mạnh	Có
Âm u	Nóng	75	Nhẹ	Có
Mưa	Ấm áp	80	Mạnh	Không
Nắng	Nóng	80	???	Không

Vì thuộc tính “Gió” có: (8 Nhẹ, 6 Mạnh) → giá trị “ ??? ” sẽ là “Nhẹ”

2.4.4 Thuật toán C4.5

Dữ liệu vào: Tập E, tập danh sách thuộc tính, tập nhãn lớp

Dữ liệu ra; Mô hình cây quyết định

Tại hàm chính, gọi đệ qui CreateTree () với ba tham số vào là tập dữ liệu DT, tập danh sách thuộc tính của C và tập nhãn lớp {d}. Thuật toán làm việc bằng cách đệ qui chọn giá trị thuộc tính tốt nhất để chia (bước 7), tiến hành mở rộng nút con bằng cách gọi đệ qui cho đến khi điều kiện dừng (bước 1) được thỏa mãn.

Chi tiết thuật toán C4.5

TreeNode CreateTree($DT, C, \{d\}$)

1. if ĐiềuKiệnDừng(DT, C) = đúng
2. Nút lá = CreateNode()
3. Nút lá.NhãnLớp = PhânLớp(DT)
4. Return Nút lá
5. Else
6. Nút gốc = CreateNode()
7. Nút gốc. ĐiềuKiệnKiểmTra = TìmĐiểmChiaTốtNhất(DT, C)
bestAttribute = getBestAttribute(DT, C);
8. Đặt $C = C \setminus \{\text{Nút chọn phân chia}\}$
9. Đặt $V = \{v \mid v \text{ thỏa điều kiện là phần phân chia xuất phát từ Nút gốc} \}$
10. Lặp qua từng tập phân chia $v \in V$
11. Đặt $DT_v = \{e \mid \text{Nút gốc.ĐiềuKiệnKiểmTra}(e) = v \text{ và } e \in DT \}$
12. Nút con = CreateTree($DT_v, C, \{d\}$)
13. Dừng lặp
14. End if
15. Return Nút gốc

Giải thích:

+ Dòng đầu tiên sẽ kiểm tra điều kiện dừng, nếu được thỏa mãn nghĩa là đệ qui để tạo ra được đến nút lá. Điều kiện dừng chỉ xảy ra khi:

\Rightarrow Tất cả các dòng trong tập dữ liệu DT thuộc về cùng một lớp duy nhất (1).

\Rightarrow Không có bất cứ dòng nào trong tập DT , điều này có thể xảy ra khi tập con được tạo ở bước phân chia các tập con là rỗng (2)

\Rightarrow Trong trường hợp (1) chỉ việc tiến hành tạo nút lá bằng hàm CreateNode() và tiến hành gán nhãn cho nút lá này bằng cách gán nhãn duy nhất cho thuộc tính nhãn của nút vừa được tạo này.

⇒ Trường hợp (2) sẽ trả về nút lá bằng rỗng và tiến hành gán nhãn cho nút cha là nhãn lớp xuất hiện nhiều nhất như sau:

⇒ Nhãn lớp = $\max(\text{tổng của từng giá trị nhãn lớp riêng biệt trong } DT)$

⇒ Hàm PhânLớp(DT) thực hiện việc xác định nhãn cho một tập dữ liệu DT , nó tự động xác định và trả về đúng giá trị nhãn cho cả hai trường hợp trên.

+ Xét Dòng 3 và 4 xảy ra khi chỉ còn một thuộc tính trong nút cha (nút cha là nút sau khi đã phân chia tạo ra tập dữ liệu D). Nếu sau khi phân chia trên nút cha mà tập D không còn thuộc tính để phân chia, trả về nút lá là giá trị nhãn xuất hiện nhiều nhất trong D .

+ Xét dòng 5, nếu thuật toán chưa thỏa mãn điều kiện dừng, tiếp tục xét bằng cách tìm kiếm điểm chia tốt nhất. Để tìm điểm chia tốt nhất cần sử dụng một hàm đánh giá, kết quả của hàm này sẽ trả về thuộc tính được chọn tương ứng. Về các tiêu chuẩn đánh giá cũng như chọn điểm chia sẽ được giải thích rõ hơn trong các phần bên dưới.

+ Xét dòng 7 và 8, sau khi đã chọn được điểm chia tốt nhất, tiến hành phân tập D thành các tập con D_i , cập nhật lại danh sách các thuộc tính.

+ Xét dòng 9 và 10, lặp qua danh sách các tập con D_i và tiến hành gọi đệ qui hàm CreateTree () với tham số mới tương ứng.

Ví dụ 2.4 minh họa thuật toán C4.5: Xét Bảng dữ liệu 2.4

Thuật toán xây dựng cây quyết định như sau:

- Tập dữ liệu vào:
 - Tập dữ liệu thời tiết
 - Tập danh sách thuộc tính: Ngày, Quang cảnh, Nhiệt độ, Độ ẩm, Gió.
 - Tập nhãn lớp: có, không
- Dữ liệu ra: Mô hình cây quyết định chơi tennis

Thuật toán xây dựng cây quyết định như sau:

Lần tạo cây đầu tiên

TìmĐiểmChiaTốtNhất(DT, C) với DT là Tập dữ liệu thời tiết và C là Tập danh sách thuộc tính: Ngày, Quang cảnh, Nhiệt độ, Độ ẩm, Gió.

Thuộc tính quyết định “Chơi tennis” chỉ có hai giá trị “Có” – “Không”

Như vậy có 9 bộ dữ liệu có nhãn lớp là giá trị “Có” và 5 bộ dữ liệu có nhãn lớp là giá trị “Không”

$$\text{Tính Entropy}(DT) = -\frac{9}{14}\log_2 \frac{9}{14} - \frac{5}{14}\log_2 \frac{5}{14} = 0.940$$

* *Độ đo RatioGain cho thuộc tính “Quang Cảnh” (QC):*

Tập Giá Trị (QC) = {Nắng, Âm u, Mưa}, khi đó:

$DT_{\text{Nắng}}(2\text{Có}, 3\text{Không})$

$$\Rightarrow \text{Entropy}(DT_{\text{Nắng}}) = \frac{5}{14} \left(-\frac{2}{5}\log_2 \frac{2}{5} - \frac{3}{5}\log_2 \frac{3}{5} \right) = 0.347$$

$$DT_{\text{Âm u}}(4\text{Có}, 0\text{Không}) \Rightarrow \text{Entropy}(DT_{\text{Âm u}}) = \frac{4}{14} \cdot 0 = 0$$

$DT_{\text{Mưa}}(3\text{Có}, 2\text{Không})$

$$\Rightarrow \text{Entropy}(DT_{\text{Mưa}}) = \frac{5}{14} \left(-\frac{3}{5}\log_2 \frac{3}{5} - \frac{2}{5}\log_2 \frac{2}{5} \right) = 0.347$$

$$\begin{aligned} \text{Entropy}_{\text{QC}}(DT) &= \frac{5}{14} \text{Entropy}(S_{\text{Nắng}}) + \frac{4}{14} \text{Entropy}(DT_{\text{Âm u}}) + \\ &+ \frac{5}{14} \text{Entropy}(DT_{\text{Mưa}}) = 0.347 + 0 + 0.347 = 0.694 \end{aligned}$$

$$\text{IG}(DT, \text{QC}) = \text{Entropy}(DT) - \text{Entropy}_{\text{QC}}(DT) = 0.940 - 0.694 = 0.246$$

$$\text{SplitInformation}(DT, \text{QC}) = -\frac{5}{14}\log_2 \frac{5}{14} - \frac{4}{14}\log_2 \frac{4}{14} - \frac{5}{14}\log_2 \frac{5}{14} = 1.577$$

$$\text{RatioGain}(DT, \text{QC}) = 0.246 / 1.577 = 0.156$$

* *Độ đo RatioGain cho thuộc tính “Gió”:*

Tập Giá Trị (Gió) = {Mạnh, Nhẹ}, khi đó:

$DT_{\text{Mạnh}}(3\text{Có}, 3\text{Không})$

$$\Rightarrow \text{Entropy}(DT_{\text{Mạnh}}) = \frac{6}{14} \left(-\frac{3}{6}\log_2 \frac{3}{6} - \frac{3}{6}\log_2 \frac{3}{6} \right) = 0.429$$

$DT_{\text{Nhẹ}}(6\text{Có}, 2\text{Không})$

$$\Rightarrow \text{Entropy}(DT_{\text{Nhẹ}}) = \frac{8}{14} \left(-\frac{6}{8}\log_2 \frac{6}{8} - \frac{2}{8}\log_2 \frac{2}{8} \right) = 0.464$$

$$\text{Entropy}_{\text{Gió}}(DT) = \frac{6}{14} \text{Entropy}(DT_{\text{Mạnh}}) + \frac{8}{14} \text{Entropy}(DT_{\text{Nhẹ}}) = 0.429 + 0.464 = 0.893$$

$$IG(DT, \text{Gió}) = \text{Entropy}(DT) - \text{Entropy}_{\text{Gió}}(DT) = 0.940 - 0.893 = 0.047$$

$$\text{SplitInformation}(DT, \text{Gió}) = -\frac{6}{14} \log_2 \frac{6}{14} - \frac{8}{14} \log_2 \frac{8}{14} = 0.985$$

$$\text{RatioGain}(DT, \text{Gió}) = 0.047/0.985 = 0.048$$

* *Độ đo RatioGain cho thuộc tính “Độ ẩm”:*

Tập Giá Trị (Độ ẩm) = $\{ \leq 72.5, > 72.5 \}$, khi đó:

$DT_{\leq 72.5}(3\text{Có}, 1\text{Không})$

$$\Rightarrow \text{Entropy}(DT_{\leq 72.5}) = \frac{4}{14} \left(-\frac{3}{4} \log_2 \frac{3}{4} - \frac{1}{4} \log_2 \frac{1}{4} \right) = 0.231$$

$DT_{> 72.5}(6\text{Có}, 4\text{Không})$

$$\Rightarrow \text{Entropy}(DT_{> 72.5}) = \frac{10}{14} \left(-\frac{6}{10} \log_2 \frac{6}{10} - \frac{4}{10} \log_2 \frac{4}{10} \right) = 0.694$$

$$\text{Entropy}_{\text{Độ ẩm}}(DT) = \frac{4}{14} \text{Entropy}(DT_{\leq 72.5}) + \frac{10}{14} \text{Entropy}(DT_{> 72.5}) = 0.925$$

$$IG(DT, \text{Độ ẩm}) = \text{Entropy}(DT) - \text{Entropy}_{\text{Độ ẩm}}(DT) = 0.940 - 0.925 = 0.015$$

$$\text{SplitInformation}(DT, \text{Độ ẩm}) = -\frac{4}{14} \log_2 \frac{4}{14} - \frac{10}{14} \log_2 \frac{10}{14} = 0.863$$

$$\text{RatioGain}(DT, \text{Độ ẩm}) = 0.015/0.863 = 0.017$$

* *Độ đo RatioGain cho thuộc tính “Nhiệt độ”:*

Tập Giá Trị (Nhiệt độ) = {Nóng, Ấm áp, Mát}, khi đó:

$DT_{\text{Nóng}}(2\text{Có}, 2\text{Không})$

$$\Rightarrow \text{Entropy}(DT_{\text{Nóng}}) = \frac{4}{14} \left(-\frac{2}{4} \log_2 \frac{2}{4} - \frac{2}{4} \log_2 \frac{2}{4} \right) = 0.286$$

$DT_{\text{Ấm áp}}(4\text{Có}, 2\text{Không})$

$$\Rightarrow \text{Entropy}(DT_{\text{Ấm áp}}) = \frac{6}{14} \left(-\frac{4}{6} \log_2 \frac{4}{6} - \frac{2}{6} \log_2 \frac{2}{6} \right) = 0.394$$

$DT_{\text{Mát}}(3\text{Có}, 1\text{Không})$

$$\Rightarrow \text{Entropy}(DT_{\text{Mát}}) = \frac{4}{14} \left(-\frac{3}{4} \log_2 \frac{3}{4} - \frac{1}{4} \log_2 \frac{1}{4} \right) = 0.321$$

$$\text{Entropy}_{\text{Nhiệt độ}}(DT) = \frac{4}{14} \text{Entropy}(DT_{\text{Nóng}}) + \frac{6}{14} \text{Entropy}(DT_{\text{Ấm áp}}) + \frac{4}{14} \text{Entropy}(DT_{\text{Mát}})$$

$$= 0.286 + 0.394 + 0.321 = 0.911$$

$$IG(DT, \text{Nhiệt độ}) = \text{Entropy}(DT) - \text{Entropy}_{\text{Nhiệt độ}}(DT) = 0.940 - 0.911 = 0.029$$

$$\text{SplitInformation}(DT, \text{Nhiệt độ}) = -\frac{4}{14} \log_2 \frac{4}{14} - \frac{6}{14} \log_2 \frac{6}{14} - \frac{4}{14} \log_2 \frac{4}{14} = 1.557$$

$$\text{RatioGain}(DT, \text{Nhiệt độ}) = 0.029/1.557 = 0.019$$

* Độ đo RatioGain cho thuộc tính “Ngày”:

$$\text{Entropy}(DT_{D1}) = \text{Entropy}(DT_{D2}) = \dots = \text{Entropy}(DT_{D14}) = 0$$

$$\text{Entropy}_{\text{Ngày}}(DT) = \frac{1}{14} \text{Entropy}(DT_{D1}) + \frac{1}{14} \text{Entropy}(DT_{D2}) + \dots + \frac{1}{14} \text{Entropy}(DT_{D14})$$

$$= 14 \cdot \frac{1}{14} \cdot 0 = 0$$

$$\text{Entropy}(DT_{D1}) = \text{Entropy}(DT_{D2}) = \dots = \text{Entropy}(DT_{D14}) = 0$$

$$IG(DT, \text{Ngày}) = \text{Entropy}(DT) - \text{Entropy}_{\text{Ngày}}(DT) = 0.940 - 0 = 0.940$$

$$\text{SplitInformation}(DT, \text{Ngày}) = \left(-\frac{4}{14} \log_2 \frac{4}{14} \right) \cdot 14 = 3.807$$

$$\text{RatioGain}(DT, \text{Ngày}) = 0.940/3.837 = 0.246$$

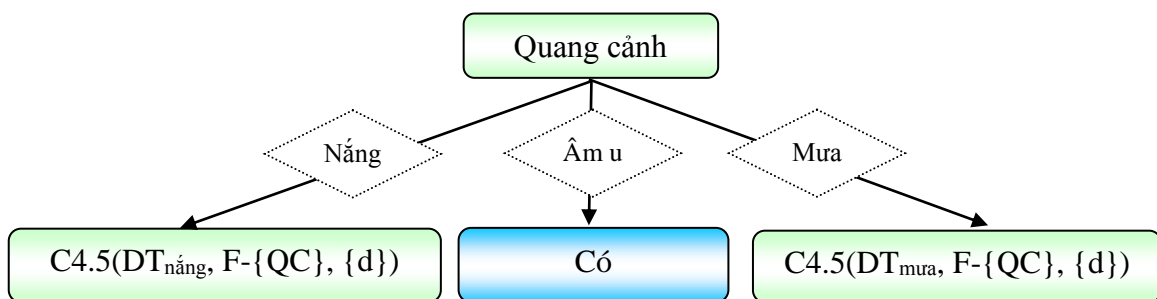
Chọn thuộc tính tốt nhất để phân chia

$$\text{Entropy trung bình của các thuộc tính} = (0.694 + 0.892 + 0.925 + 0.911 + 0)/5 = 0.684$$

$$\text{Entropy}_{\text{QC}}(DT) = 0.694 > 0.684$$

$$\text{RatioGain}(DT, \text{QC}) = 0.246/1.577 = 0.156$$

→ “Quang cảnh” là thuộc tính được chọn để phân chia



Hình 2.6: Cây quyết định bước đầu được xây dựng theo thuật toán C4.5 ứng với Bảng quyết định 2.4

Lần tạo cây thứ hai:

Xét nhánh “Quang cảnh” = Nắng

$$DT_{\text{Nắng}}(2\text{Có}, 3\text{Không}) \Rightarrow \text{Entropy}(DT_{\text{Nắng}}) = -\frac{2}{5} \log_2 \frac{5}{5} - \frac{3}{5} \log_2 \frac{3}{5} = 0.971$$

* Độ đo RatioGain cho thuộc tính “Nhiệt độ”:

$$DT_{\text{Nóng}}(0\text{Có}, 2\text{Không}) \Rightarrow \text{Entropy}(DT_{\text{Nóng}}) = 0$$

$$DT_{\text{Ám áp}}(1\text{Có}, 1\text{Không}) \Rightarrow \text{Entropy}(DT_{\text{Ám áp}}) = \frac{2}{5} \left(-\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{2} \log_2 \frac{1}{2} \right) = 0.4$$

$$DT_{\text{Mát}}(1\text{Có}, 0\text{Không}) \Rightarrow \text{Entropy}(DT_{\text{Mát}}) = 0$$

$$\begin{aligned} \text{Entropy}_{\text{Nhiệt độ}}(DT_{\text{Nắng}}) &= \frac{2}{5} \text{Entropy}(DT_{\text{Nóng}}) + \frac{2}{5} \text{Entropy}(DT_{\text{Ám áp}}) + \frac{1}{5} \text{Entropy}(DT_{\text{Mát}}) \\ &= 0.4 \end{aligned}$$

$$\text{IG}(DT_{\text{nắng}}, \text{Nhiệt độ}) = 0.971 - 0.4 = 0.571$$

$$\text{SplitInformation}(DT_{\text{Nắng}}, \text{Nhiệt độ}) = -\frac{2}{5} \log_2 \frac{2}{5} - \frac{2}{5} \log_2 \frac{2}{5} - \frac{1}{5} \log_2 \frac{1}{5} = 1.522$$

$$\text{RatioGain}(DT, \text{Nhiệt độ}) = 0.571 / 1.522 = 0.375$$

* Độ đo RatioGain cho thuộc tính “Độ ẩm”:

Độ ẩm						
	70		85		90	
	77.5		87.5		92.5	
	<=	>	<=	>	<=	>
Có	2	0	2	0	2	0
Không	0	3	1	2	2	1
IG	0.971		0.420		0.171	

Tập Giá Trị (Độ ẩm) = $\{<=77.5, >77.5\}$, khi đó:

$$\text{Entropy}_{\text{Độ ẩm}}(DT) = -\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} = 0.971$$

$$\text{Entropy}_{\text{Độ ẩm}}(DT_{\text{Nắng}}) = \frac{2}{5} \text{Entropy}(DT_{<=77.5}) + \frac{3}{5} \text{Entropy}(DT_{>77.5}) = 0$$

$$\text{IG}(DT_{\text{Nắng}}, \text{Độ ẩm}) = 0.971 - 0 = 0.971$$

$$\text{SplitInformation}(DT_{\text{Nắng}}, \text{Độ ẩm}) = -\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} = 0.971$$

$$\text{RatioGain}(DT, \text{Độ ẩm}) = 0.971 / 0.971 = 1$$

* Độ đo RatioGain cho thuộc tính “Gió”:

$$DT_{\text{Mạnh}}(1\text{Có}, 2\text{Không}) \Rightarrow \text{Entropy}(DT_{\text{Mạnh}}) = 0.551$$

$$DT_{\text{Nhẹ}}(1\text{Có}, 1\text{Không}) \Rightarrow \text{Entropy}(DT_{\text{Nhẹ}}) = 0.4$$

$$\text{Entropy}_{\text{Gió}}(DT_{\text{Nắng}}) = \frac{3}{5} \text{Entropy}(DT_{\text{Mạnh}}) + \frac{2}{5} \text{Entropy}(DT_{\text{Nhẹ}}) = 0.951$$

$$\text{IG}(DT_{\text{Nắng}}, \text{Gió}) = 0.971 - 0.951 = 0.02$$

$$\text{SplitInformation}(DT_{\text{Nắng}}, \text{Gió}) = -\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} = 0.971$$

$$\text{RatioGain}(DT, \text{Gió}) = 0.02/0.971 = 0.021$$

* Độ đo RatioGain cho thuộc tính “Ngày”:

Tập Giá Trị (Ngày) = {D1, D2, D8, D9, D11}, khi đó:

$$\text{Entropy}(DT_{D1}) = \text{Entropy}(DT_{D2}) = \text{Entropy}(DT_{D8}) = \text{Entropy}(DT_{D9}) = \text{Entropy}(DT_{D11}) = 0$$

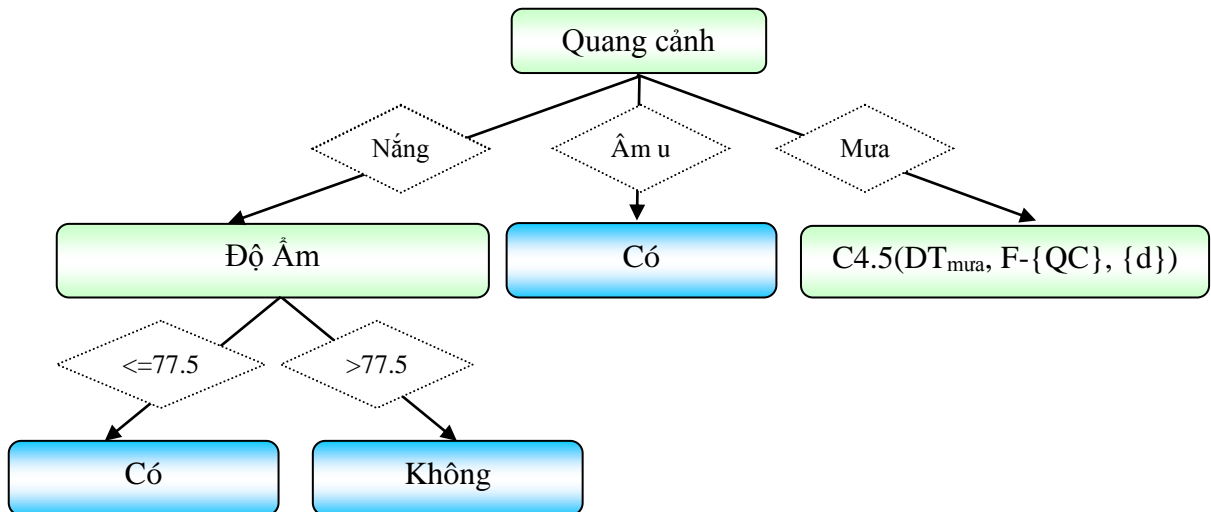
$$\text{Entropy}_{\text{Ngày}}(DT_{\text{Nắng}}) = 5 \cdot (1/5) \cdot 0 = 0$$

$$\text{IG}(DT_{\text{Nắng}}, \text{Ngày}) = 0.971 - 0 = 0.971$$

$$\text{SplitInformation}(DT_{\text{Nắng}}, \text{Ngày}) = \left(-\frac{1}{5} \log_2 \frac{1}{5} \right) \cdot 5 = 2.322$$

$$\text{RatioGain}(DT_{\text{Nắng}}, \text{Ngày}) = 0.971/2.322 = 0.418$$

➔ “Độ ẩm” là thuộc tính được chọn để phân chia



Hình 2.7: Cây quyết định được xây dựng theo thuật toán C4.5
nhánh “Quang cảnh” = Nắng

Xét nhánh “Quang cảnh” = Mưa

* Độ đo RatioGain cho thuộc tính “Nhiệt độ”:

Tập Giá Trị (Nhiệt độ) = {Ấm áp, Mát}, khi đó:

$$DT_{\text{Ấm áp}}(2\text{Có}, (1\text{Không}))$$

$$\Rightarrow \text{Entropy}(DT_{\text{Ám áp}}) = \frac{3}{5} \left(-\frac{2}{3} \log_2 \frac{2}{3} - \frac{1}{3} \log_2 \frac{1}{3} \right) = 0.551$$

$$DT_{\text{Mát}}(1\text{Có}, 1\text{Không}) \Rightarrow \text{Entropy}(DT_{\text{Mát}}) = 0.4$$

$$\text{Entropy}_{\text{Nhiệt độ}}(DT_{\text{Mưa}}) = \frac{2}{5} \text{Entropy}(DT_{\text{Mát}}) + \frac{3}{5} \text{Entropy}(DT_{\text{Ám áp}}) = 0.551 + 0.4 = 0.951$$

$$\text{IG}(DT_{\text{Mưa}}, \text{Nhiệt độ}) = 0.971 - 0.951 = 0.02$$

$$\text{SplitInformation}(DT_{\text{Mưa}}, \text{Nhiệt độ}) = -\frac{3}{5} \log_2 \frac{3}{5} - \frac{2}{5} \log_2 \frac{2}{5} = 0.971$$

$$\text{RatioGain}(DT_{\text{Mưa}}, \text{Nhiệt độ}) = 0.02/0.971 = 0.021$$

* *Độ đo RatioGain cho thuộc tính “Gió”:*

$$DT_{\text{Mạnh}}(0\text{Có}, 2\text{Không}) \Rightarrow \text{Entropy}(DT_{\text{Mạnh}}) = 0$$

$$DT_{\text{Nhẹ}}(3\text{Có}, 0\text{Không}) \Rightarrow \text{Entropy}(DT_{\text{Nhẹ}}) = 0$$

$$\text{Entropy}_{\text{Gió}}(DT_{\text{Mưa}}) = 0$$

$$\text{IG}(DT_{\text{Mưa}}, \text{Gió}) = 0.971 - 0 = 0.971$$

$$\text{SplitInformation}(DT_{\text{Mưa}}, \text{Gió}) = -\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} = 0.971$$

$$\text{RatioGain}(DT_{\text{Mưa}}, \text{Gió}) = 0.971/0.971 = 1$$

* *Độ đo RatioGain cho thuộc tính “Ngày”:*

Tập Giá Trị (Ngày) = {D4, D5, D6, D10, D14}, khi đó:

$$\text{Entropy}(DT_{\text{D4}}) = \text{Entropy}(DT_{\text{D5}}) = \text{Entropy}(DT_{\text{D6}}) = \text{Entropy}(DT_{\text{D10}}) = \text{Entropy}(DT_{\text{D14}}) = 0$$

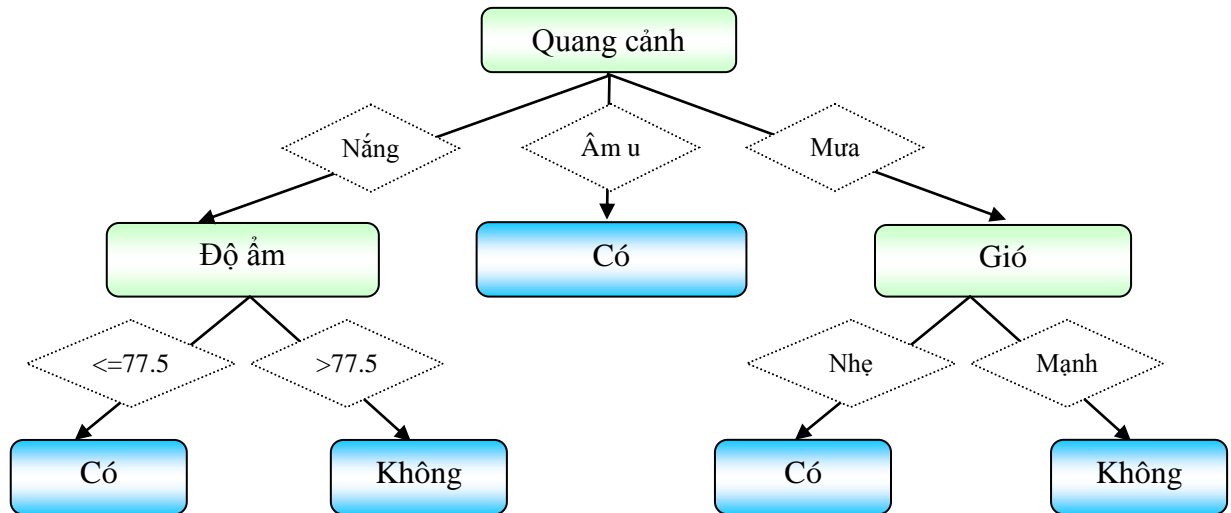
$$\text{Entropy}_{\text{Ngày}}(DT_{\text{Mưa}}) = 5 \cdot \left(\frac{1}{5} \right) \cdot 0 = 0$$

$$\text{IG}(DT_{\text{Mưa}}, \text{Ngày}) = 0.971 - 0 = 0.971$$

$$\text{SplitInformation}(DT_{\text{Mưa}}, \text{Ngày}) = \left(-\frac{1}{5} \log_2 \frac{1}{5} \right) \cdot 5 = 2.322$$

$$\text{RatioGain}(DT_{\text{Mưa}}, \text{Ngày}) = 0.971/2.322 = 0.418$$

➔ “Gió” là thuộc tính được chọn để phân chia



Hình 2.8: Cây quyết định được xây dựng theo thuật toán C4.5
ứng với Bảng quyết định 2.4

2.5 Phương pháp xây dựng cây quyết định FID3

2.5.1 Xác định điểm chia tốt nhất

Xét bảng quyết định $DT = (U, C \cup \{d\})$.

Lượng thông tin thu thêm ổn định IG_{fix} - Fixed Information Gain [5] là tiêu chuẩn mới cho chọn thuộc tính thuộc tính điều kiện c nào đó để phân chia. IG_{fix} được xác định theo công thức sau:

$$IG_{fix}(DT, c) = \sqrt{\gamma(d, c) * \frac{IG(DT, c)}{|c|}}$$

Trong đó:

- + $|c|$ là số các giá trị khác nhau của thuộc tính điều kiện c
- + $\gamma(d, c)$ là độ phụ thuộc d vào c
- + $IG(DT, c)$ là lượng thông tin thu thêm

Lượng thông tin thu thêm ổn định của thuộc tính được sử dụng như một tiêu chuẩn cho việc chọn thuộc tính kiểm tra tại mỗi nút trong cây quyết định. Thuộc tính điều kiện với giá trị lượng thông tin thu thêm ổn định lớn nhất được chọn từ tập rút gọn thuộc tính và được sử dụng làm nút gốc của cây.

2.5.2. Thuật toán FID3

Dữ liệu vào: Bảng quyết định $DT = (U, C \cup \{d\})$

Dữ liệu ra: Mô hình cây quyết định

Thuật toán FID3 - Fixed Iterative Dichotomiser 3 [5]

```

TreeNode CreateTree(DT, C, {d})
{
    if ( Nếu tất cả các mẫu cùng nhãn lớp  $d_i$  )
        return (TreeNode( $d_i$ ));
    if ( C == null )
        return (TreeNode( $d_j$ ));
    bestAttribute = getBestAttribute(DT,C);
    Root = TreeNode(bestAttribute);
    foreach (v in bestAttribute)
    {
        DTv = [DT]v;
        C = C - {bestAttribute};
        if ( $|DT_v| == 0$ )
            Root.AddTreeNode(TreeNode( $d_i$ ),v);
        else
        {
            ChildNode = CreateTree(DTv, C, {d});
            Root.AddTreeNode(ChildNode,v);
        }
    }
    return Root;
}

```

Giải thích thuật toán FID3.

Các dòng của thuật toán được giải thích tương tự như thuật toán ID3, chỉ khác nhau ở dòng gọi hàm getBestAttribute, tức chọn thuộc tính điều kiện phân nhánh tại mỗi nút của cây. Hàm getBestAttribute trả về thuộc tính điều kiện tốt nhất làm thuộc tính phân nhánh. Giả mã của hàm getBestAttribute như sau:

Dữ liệu vào: Bảng quyết định $DT = (U, C \cup \{d\})$

Dữ liệu ra: Thuộc tính điều kiện tốt nhất.

```

getBestAttribute(DT,C)
{
    C' = C;
    foreach (ci in C)
    {
        if (DependencyAttribute(DT, ci) == 0)
            //Tính độ phụ thuộc của thuộc tính
            C' = C' - ci;
    }
    maxIGfix = 0;
    foreach (ci in C')
    {
        temp = IGfix(DT,ci);
        //Tính lượng thông tin thu thêm ổn định
        if (temp >= maxIGfix)
        {
            maxIGfix = temp;
            result = ci;
        }
    }
    return result;
}

```

Ví dụ, Xét bảng quyết định $DT = \{U, C \cup \{d\}\}$ cho trong Bảng 2.2.

Trong đó tập thuộc tính điều kiện $C = \{To, Ly, Nv, Av\}$ và thuộc tính quyết định $d = \{Tc\}$

- Trước tiên để xây dựng cây, một nút gốc được khởi tạo, và tính độ phụ thuộc d vào tất cả các thuộc tính điều kiện.

Thuộc tính d gồm hai giá trị : “T” (9 mẫu) và “A” (5 mẫu) nên :

$$[U]_d = \{\{u_3, u_4, u_5, u_7, u_9, u_{10}, u_{11}, u_{12}, u_{13}\}, \{u_1, u_2, u_6, u_8, u_{14}\}\}$$

+ Với thuộc tính “To” gồm ba giá trị: “K2” (5 mẫu), “K1” (4 mẫu) và “G” (5 mẫu) nên:

$$[U]_{To} = \{\{u_1, u_2, u_8, u_9, u_{11}\}, \{u_3, u_7, u_{12}, u_{13}\}, \{u_4, u_5, u_6, u_{10}, u_{14}\}\}$$

$$\text{Do đó } \gamma(d, To) = \frac{|pos_{To}(d)|}{|U|} = \frac{|\{u_3, u_7, u_{12}, u_{13}\}|}{|U|} = \frac{4}{14} = 0.286$$

+ Với thuộc tính “Ly” gồm ba giá trị: “K2” (4 mẫu), “K1” (6 mẫu) và “G” (4 mẫu) nên:

$$[U]_{Ly} = \{\{u_5, u_6, u_7, u_9\}, \{u_4, u_8, u_{10}, u_{11}, u_{12}, u_{14}\}, \{u_1, u_2, u_3, u_{13}\}\}$$

$$\gamma(d, Ly) = \frac{|pos_{Ly}(d)|}{|U|} = \frac{0}{12} = 0$$

+ Với thuộc tính “Nv” gồm hai giá trị: “K1” (bảy mẫu), “K2” (bảy mẫu).

Nên

$$[U]_{Nv} = \{\{u_5, u_6, u_7, u_9, u_{10}, u_{11}, u_{13}\}, \{u_1, u_2, u_3, u_4, u_8, u_{12}, u_{14}\}\}$$

$$\gamma(d, Nv) = \frac{|pos_{Nv}(d)|}{|U|} = 0$$

+ Với thuộc tính “Av” gồm hai giá trị: “K2” (tám mẫu), “K1” (sáu mẫu).

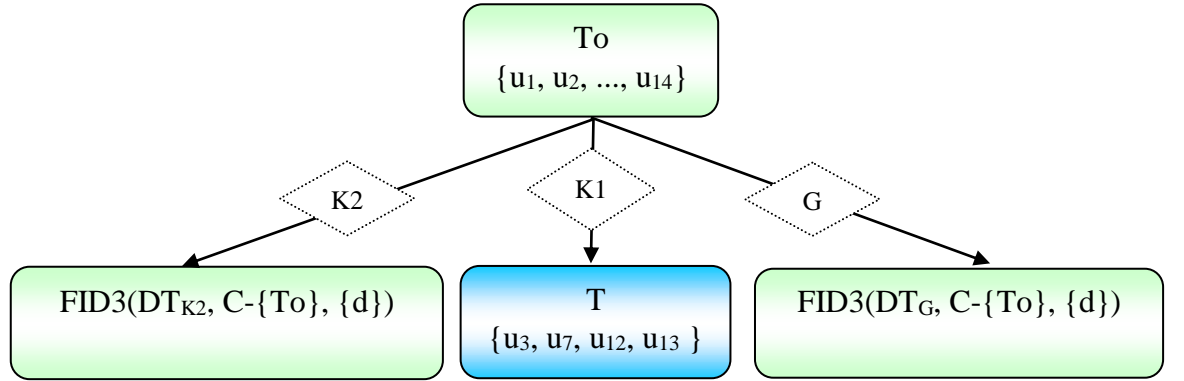
Nên

$$[U]_{Av} = \{\{u_1, u_3, u_4, u_5, u_8, u_9, u_{10}, u_{13}\}, \{u_2, u_6, u_7, u_{11}, u_{12}, u_{14}\}\}$$

$$\gamma(d, Av) = \frac{|pos_{Av}(d)|}{|U|} = 0$$

Nên tập thuộc tính mới $C' = \{To\}$. Vì $|C'| = 1$ nên ta không cần tính

$IG_{fix}(DT, Tuổi)$ và chọn thuộc tính này làm nhãn cho nút gốc, nên nút gốc có nhãn là “To” và ba nhánh được tạo ra lần lượt với tên là “K1”, “K2”, “G”. Nhận thấy rằng tất cả các mẫu ứng với giá trị khi “To” bằng “G” đều thuộc vào cùng một lớp “T”, do đó một nút lá được tạo ra ở cuối của nhánh với nhãn là “T”. Nên cây quyết định có dạng:



Hình 2.9: Cây quyết định bước đầu được xây dựng theo thuật toán FID3 ứng với Bảng quyết định 2.2

- Bước tiếp theo gọi thuật toán đệ quy: $FID3(DT_{K2}, C-\{To\}, \{d\})$

Ta có DT_{K2} gồm có các mẫu $\{u_1, u_2, u_8, u_9, u_{11}\}$ với hai mẫu $\{u_9, u_{11}\}$ và ba mẫu $\{u_1, u_2, u_8\}$

$$[U_{K2}]_d = \{\{u_9, u_{11}\}, \{u_1, u_2, u_8\}\}$$

Tương tự để tìm điểm chia tốt nhất tại thuật toán này, phải tính toán độ phụ thuộc của các thuộc tính

+ Với thuộc tính “Ly”, thuộc tính này có ba giá trị “K2”, “K1”, “G” Vì vậy

$$[U_{K2}]_{Ly} = \{\{u_9\}, \{u_8, u_{11}\}, \{u_1, u_2\}\}$$

$$\gamma(d, Ly) = \frac{|pos_{Ly}(d)|}{|U_{K2}|} = \frac{|\{u_9, u_1, u_2\}|}{|U_{K2}|} = \frac{3}{5} = 0.600$$

+ Với thuộc tính “Nv” gồm hai giá trị: “K1” (hai mẫu), “K2” (ba mẫu). Nên

$$[U_{K2}]_{Nv} = \{\{u_9, u_{11}\}, \{u_1, u_2, u_8\}\}$$

$$\gamma(d, Nv) = \frac{|pos_{Nv}(d)|}{|U_{K2}|} = \frac{|\{u_9, u_{11}, u_1, u_2, u_8\}|}{|U_{K2}|} = \frac{5}{5} = 1$$

+ Với thuộc tính “Av” gồm hai giá trị: “K2” (ba mẫu), “K1” (hai mẫu). Nên

$$[U_{K2}]_{Av} = \{\{u_1, u_8, u_9\}, \{u_2, u_{11}\}\}$$

$$\gamma(d, Av) = \frac{|pos_{Av}(d)|}{|U_{K2}|} = \frac{0}{5} = 0$$

Nên thuộc tính mới $C'_{K2}=\{Ly, Nv\}$. Tiếp theo tính $IG_{fix}(DT_{K2}, Ly)$ và $IG_{fix}(DT_{K2}, Nv)$

$$Entropy(DT_{K2}) = -\frac{2}{5}\log_2 \frac{2}{5} - \frac{3}{5}\log_2 \frac{3}{5} = 0.971$$

Tiếp theo tính IG cho thuộc tính “Ly”, thuộc tính này có ba giá trị “K2”, “K1” và “G”. Nhìn vào Bảng quyết định 2.2, với giá trị “K2” có một bộ $\{u_9\}$ có giá trị thuộc tính nhãn là “T” và không bộ có giá trị thuộc tính nhãn là “A”. Tương tự giá trị “K1” có một bộ $\{u_{11}\}$ có nhãn lớp là “T” và một bộ $\{u_8\}$ có nhãn lớp là “A”; với giá trị “G” có không bộ có nhãn lớp “T” và hai bộ $\{u_1, u_2\}$ có nhãn lớp “A”. Nên độ đo lượng thông tin thu thêm của thuộc tính “Ly” xét trên DT_{K2} là:

$$IG(DT_{K2}, Ly) = Entropy(DT_{K2}) - \sum_{v \in V_{Ly}} \frac{|DT_{K2_v}|}{|DT_{K2}|} Entropy(DT_{K2_v})$$

$$= 0.971 - \left[\frac{1}{5} \left(-\frac{1}{1} \log_2 \frac{1}{1} \right) + \frac{2}{5} \left(-\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{2} \log_2 \frac{1}{2} \right) + \frac{2}{5} \left(-\frac{2}{2} \log_2 \frac{2}{2} \right) \right] = 0.571$$

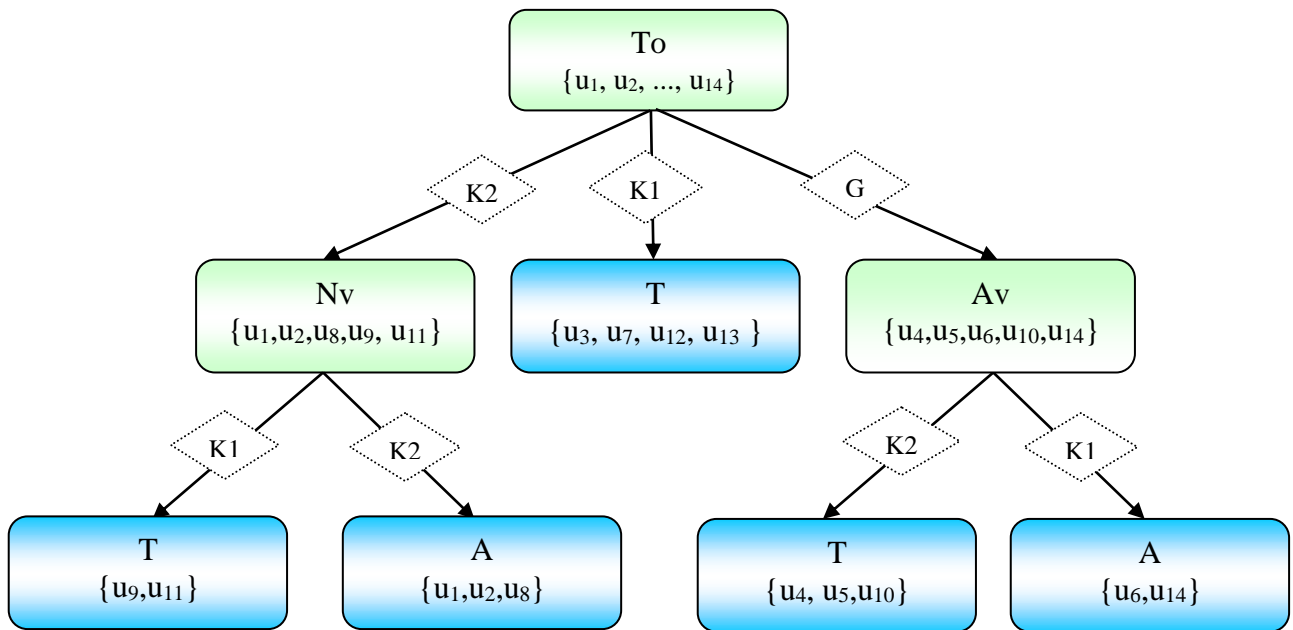
$$\text{Do đó } IG_{fix}(DT_{K2}, Ly) = \sqrt{\gamma(d, Ly) * \frac{IG(DT_{K2}, Ly)}{|Ly|}}$$

$$= \sqrt{0.600 * \frac{0.571}{3}} = 0.338$$

Tính tương tự ta cũng có $IG(DT_{K2}, Nv) = 0.971$. Nên

$$IG_{fix}(DT_{K2}, Nv) = \sqrt{1 * \frac{0.971}{3}} = 0.569$$

Ta thấy $IG_{fix}(DT_{K2}, Nv)$ có giá trị lớn nhất nên nút gốc có nhãn là “Nv” và có hai nhánh được tạo ra ứng với các giá trị của “Nv”. Nhận thấy rằng tất cả các mẫu ứng với giá trị khi “Nv” bằng “K1” đều thuộc vào cùng một lớp “T”, do đó một nút lá được tạo ra ở cuối của nhánh với nhãn là “T”; Tương tự tất cả các mẫu ứng với giá trị “Nv” bằng “K1” cùng thuộc lớp “A” nên nút lá được tạo có nhãn là “A”. Phần còn lại được tiếp tục phân lớp trên mỗi nhánh một cách tương tự. Cây quyết định cuối cùng khi thực hiện theo thuật toán FID3:



Hình 2.10: Cây quyết định được xây dựng theo thuật toán FID3 ứng với Bảng quyết định 2.2

2.6 Kết luận chương 2

Trong chương này đã trình bày phương pháp tổng quát xây dựng cây quyết định; Thuật toán xây dựng cây quyết định ID3, C4.5, FID3; Các ví dụ cụ thể để minh họa từng bước trên mỗi thuật toán. và đồng thời kiểm chứng từng thuật toán ở phụ lục 1 trên các bảng dữ liệu của các ví dụ minh họa.

CHƯƠNG 3: MÔ PHỎNG

CHƯƠNG TRÌNH PHÂN LỚP NĂNG KHIẾU HỌC SINH

Để minh họa cho phần lý thuyết ở chương 1 và chương 2, phần này sẽ mô tả một ứng dụng đơn giản cài đặt thuật toán tìm tập rút gọn và thuật toán xây dựng cây quyết định ID3

3.1. Giới thiệu bài toán

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

Việc đưa ứng dụng công nghệ thông tin vào giảng dạy trong các cấp bậc học đã và đang được quan tâm đầu tư khá lớn từ phía lãnh đạo ngành. Xuất phát từ các văn bản chỉ đạo của Đảng và nhà nước nhất là chỉ thị 58-CT/UW của Bộ Chính Trị ngày 07 tháng 10 năm 2000 về việc đẩy mạnh ứng dụng CNTT phục vụ sự nghiệp Công nghiệp hóa và Hiện đại hóa đã chỉ rõ trọng tâm của ngành giáo dục là đào tạo nguồn nhân lực về CNTT và đẩy mạnh ứng dụng CNTT trong công tác giáo dục và đào tạo, đây là nhiệm vụ mà Thủ tướng Chính phủ đã giao cho ngành giáo dục thông qua quyết định số 81/2001/QĐ-TTg;

Hiện nay các trường học đã và đang trang bị rất nhiều các thiết bị và phần mềm phục vụ việc giảng dạy cũng như quản lý một số lĩnh vực hoạt động của trường. Các giáo viên đã được đào tạo, tập huấn và tự học để đáp ứng việc vận dụng thiết bị và phần mềm trong giảng dạy và đã đầu tư khá nhiều vào việc ứng dụng công nghệ thông tin trong soạn giảng và kiểm tra đánh giá học sinh. Các cán bộ quản lý của trường cũng đã được tập huấn sử dụng nhiều phần mềm để đưa áp dụng công nghệ thông tin vào quản lý một số lĩnh vực hoạt động của trường như : quản lý nhân sự , quản lý học sinh, quản lý thư viện, quản lý sắp xếp thời khóa biểu, quản lý tài chính, quản lý tài sản công . . .

Theo chỉ đạo phân phối chuyên môn của Bộ Giáo dục và Đào tạo, các em học sinh bậc Trung học cơ sở và Trung học phổ thông có tiết qui định cho môn học ‘‘ Tự chọn’’ theo năng khiếu. Các tiết học này các em sẽ được đăng ký học môn mà các em có năng lực và hứng thú nhất. Với cách tiếp cận là làm thế nào để hỗ trợ Ban giám hiệu của trường có thể đưa ra gợi ý cho các em học sinh chọn môn học tự chọn phù hợp với năng khiếu của các em một cách khách quan dựa trên các cơ sở khoa học là hết sức có ý nghĩa và cần thiết.

Để xác định được năng khiếu của một học sinh phụ thuộc rất nhiều yếu tố ảnh hưởng như : Điểm các môn học, lứa tuổi, tâm lý, giới tính, khu vực sống, hoàn cảnh gia đình . . . Trong luận văn chỉ tập trung nghiên cứu việc đề xuất từ điểm trung bình các môn học lựa chọn ra các môn có ảnh hưởng đến phân lớp năng khiếu của học sinh thông qua cây quyết định đã nghiên cứu tìm ra các luật quyết định (phân lớp) năng khiếu của học sinh.

3.2. Cài đặt ứng dụng

Ứng dụng này được xây dựng bằng ngôn ngữ lập trình visual C# (trên nền tảng công nghệ Microsoft .Net 2010) chạy trên môi trường Window với cơ sở dữ liệu Microsoft Access 2010. Ứng dụng này tập trung vào xây dựng thuật toán tìm tập rút gọn dựa vào ma trận phân biệt được của lý thuyết tập thô để tìm ra các thuộc tính điểm có ảnh hưởng cao đến phân lớp năng khiếu của học sinh được trình bày ở chương 1. Thực hiện đưa các thuộc tính của tập rút gọn này vào thuật toán cây quyết định ID3 được trình bày ở chương 2. Từ cây quyết định này hay các luật quyết định trích ra từ các cây quyết định tiến hành việc phân lớp năng khiếu học sinh cho môn tự chọn để hỗ trợ Ban giám hiệu của trường có thể đưa ra gợi ý cho

các em học sinh chọn môn học tự chọn phù hợp với năng khiếu của các em một cách khách quan dựa trên các cơ sở khoa học

3.2.1. Giới thiệu về cơ sở dữ liệu

Trong quá trình thực nghiệm, luận văn sử dụng bảng dữ liệu là bảng điểm tổng hợp học kỳ, cả năm của học sinh trong năm 2011 và 2012 trích từ một số trường THCS của Tỉnh Đồng Nai (bao gồm trường nội ô thành phố, trường ở huyện, có trường điểm, trường đạt chuẩn và trường chưa đạt chuẩn)

Bảng điểm tổng hợp có các thuộc tính và giá trị của các thuộc tính được cho trong bảng sau :

Bảng 3.1: Danh sách các thuộc tính của bảng điểm tổng hợp

STT	Tên thuộc tính	Giá trị	Giải thích
1	TO	0.0 đến 10.0	Điểm trung bình môn Toán của học sinh
2	LY	0.0 đến 10.0	Điểm trung bình môn Lý của học sinh
3	HO	0.0 đến 10.0	Điểm trung bình môn Hóa của học sinh
4	SH	0.0 đến 10.0	Điểm trung bình môn Sinh của học sinh
5	NV	0.0 đến 10.0	Điểm trung bình môn Văn của học sinh
6	SU	0.0 đến 10.0	Điểm trung bình môn Sử của học sinh
7	DI	0.0 đến 10.0	Điểm trung bình môn Địa của học sinh
8	AV	0.0 đến 10.0	Điểm trung bình môn Anh văn của học sinh
9	CD	0.0 đến 10.0	Điểm trung bình môn Công dân của học sinh
10	CN	0.0 đến 10.0	Điểm trung bình môn Công nghệ của học sinh
11	AN	0.0 đến 10.0 Hoặc G,K,TB,Y	Điểm trung bình/đánh giá môn Âm nhạc của học sinh
12	MT	0.0 đến 10.0 Hoặc G,K,TB,Y	Điểm trung bình/đánh giá môn Mỹ thuật của học sinh
13	TD	0.0 đến 10.0 Hoặc G,K,TB,Y	Điểm trung bình/đánh giá môn Thể dục của học sinh
14	TBM	0.0 đến 10.0	Điểm trung bình các môn của học sinh
15	NangKhieu	Tên môn học (năng khiếu)	Môn năng khiếu của học sinh

DaTaHoc: Database (Access 2007 - 2010) - Microsoft Access

Table Tools: Fields, Table

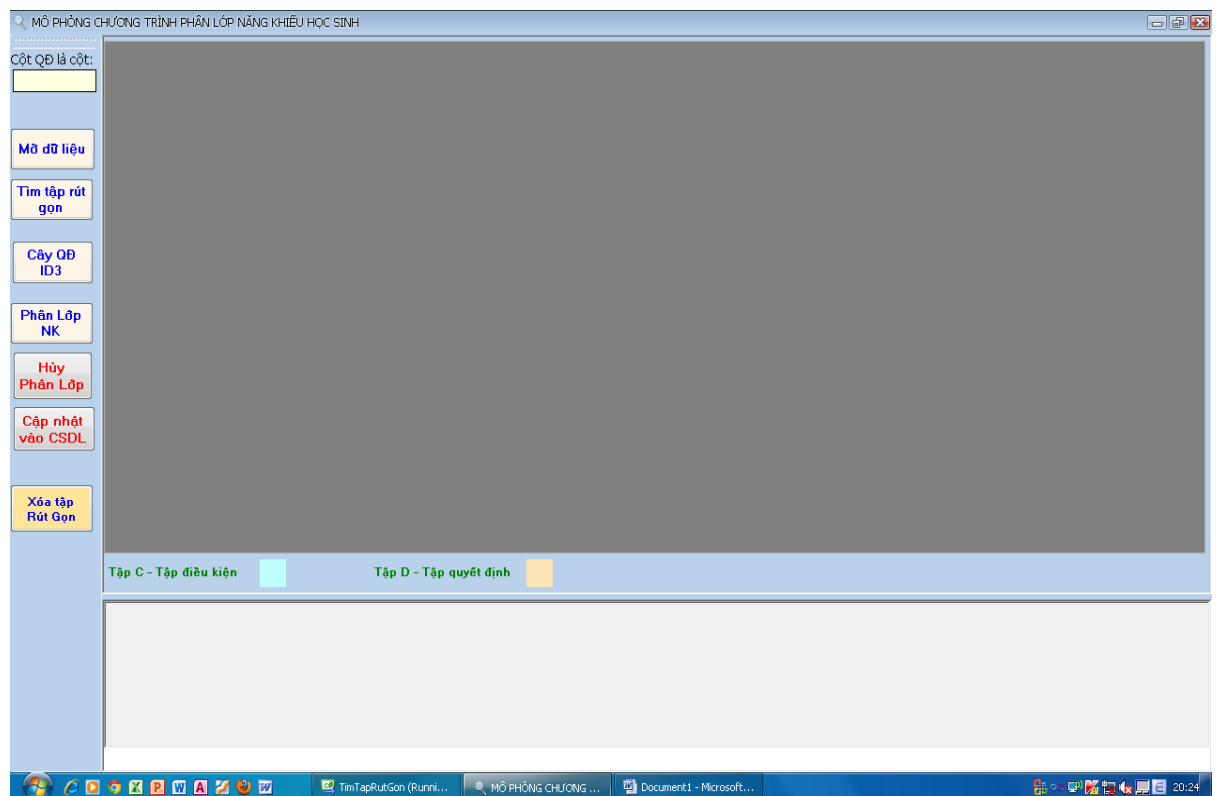
Views: Datasheet View

Record: 1 of 359

STT	TO	LY	HO	SH	NV	SU	DI	AV	CD	CN	AN	MT	TD	TBM	NangKhai	Click to Add
97	6.8	7.6	6.8	7.1	7.8	6.1	7	6.5	9	8.7	G	G	K	7.2	Văn	
98	6.7	5	5.5	3.9	7.1	5	6.9	6.3	8.4	7.5	K	G	K	6.2	Anh văn	
99	6.7	8.1	5.7	7.3	7.3	5.6	5.5	6.5	9.1	8.1	G	G	G	7	Lý	
100	6.7	7.7	7	6	7.6	4.4	7.7	5.9	8.3	9.7	K	K	G	7.2	Văn	
101	6.7	7.3	7.1	7.2	7.4	8.6	7.7	6.1	9.4	9.3	K	K	G	7.5	Anh văn	
102	6.6	8.1	6.6	6.1	7.1	6.1	6.6	6.3	8.7	9.4	G	G	G	7	Lý	
103	6.3	7.7	8	7.8	7.1	8.2	7.2	5.9	9.4	8.9	G	G	G	7.5	Lý	
104	6.2	8.4	5	6.8	7.4	5.3	5.2	6.3	7.8	7.8	G	G	K	6.6	Lý	
105	6.1	8.4	6.5	7.1	7.7	8.1	7.1	6.3	9.5	8.2	G	G	K	7.1	Lý	
106	6	8.1	5	6.7	7	8	6.8	5.6	9.5	8.8	K	G	G	7	Lý	
107	5.9	8.6	5.5	7.4	7.7	5.3	4.4	6.1	8.6	8.8	G	K	G	6.9	Lý	
108	5.8	8.2	5.9	4.4	7.5	7.4	8.1	6.4	7.6	8.6	K	G	K	6.9	Lý	
109	5.7	5.7	5.2	8.1	7.9	4.1	4.8	5.7	8.6	7.9	K	G	G	6.2	Văn	
110	5.7	8.1	6.6	8.4	7.6	9.1	8	7	9.7	9.5	G	K	G	7.9	Lý	
111	5.5	6.8	7.7	5.1	7.3	5.7	7.6	6.5	8.7	8.5	G	G	K	6.8	Hóa	
112	5.5	8.4	7.2	3.9	7.3	7.5	8	6.6	9.3	7.6	K	G	K	6.9	Lý	
113	5.4	8.4	8.8	5.9	7.9	6.1	6.8	6.5	8.9	7.6	K	G	K	7	Lý	
114	5.3	7.6	6.1	6.3	7	4.4	4.1	6	8.1	5.9	K	G	K	6	Lý	
115	5.2	6.8	5.9	6	7.1	8.4	8.1	6.4	8.6	7.7	K	G	K	6.8	Anh văn	
116	5	6.3	5.8	4.9	7.4	7	6.4	4.5	8.3	7.6	K	G	K	6.1	Anh văn	
117	5	6.8	5.6	8.1	7.6	4.4	6.1	6.1	7.9	7.7	K	G	K	6.3	Văn	
118	4.4	7.1	5.7	6.7	7.1	7.9	6.8	6.2	8.5	8	K	G	K	6.8	Anh văn	
119	9.4	9.4	9.3	7.5	7.6	9.6	9.2	9.1	9.9	9	G	G	G	8.9	Toán	
120	8	7.9	9.2	8.1	8	8.6	8.9	9.1	9.2	9.4	G	G	G	8.6	Anh văn	
121	8	8.8	9	8.2	7.7	9.3	8.8	9.1	9.5	9.8	G	G	G	8.6	Anh văn	
122	7.6	9.3	8.1	9.2	7.4	7.8	7.2	9	9.8	9.4	G	G	G	8.4	Lý	
123	7.2	9.3	6.7	5.9	7.8	8.8	7.5	9.6	8.6	9.3	G	G	K	7.9	Anh văn	

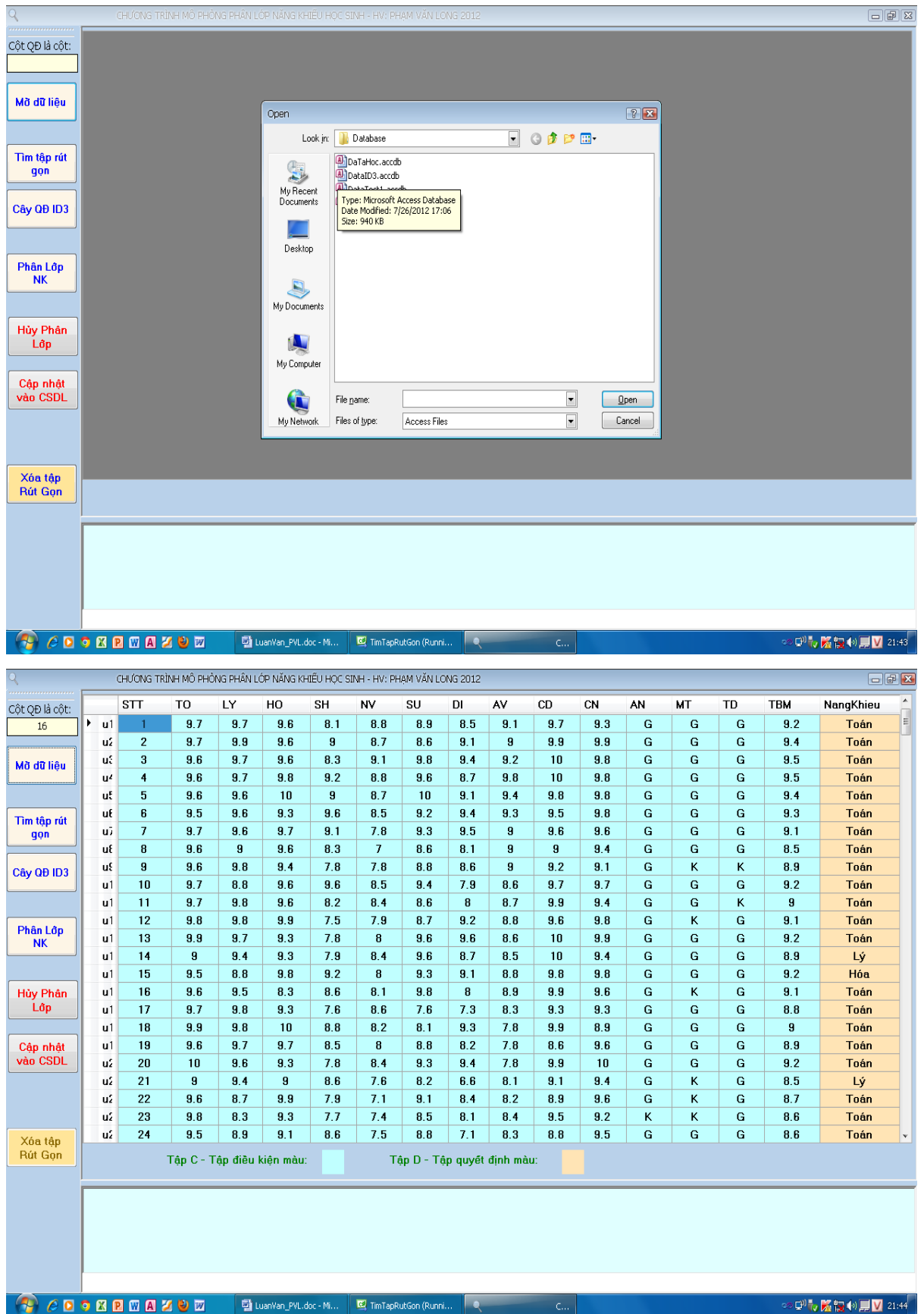
Hình 3.1 minh họa của bảng điểm tổng hợp

3.2.2 Màn hình giao diện của chương trình



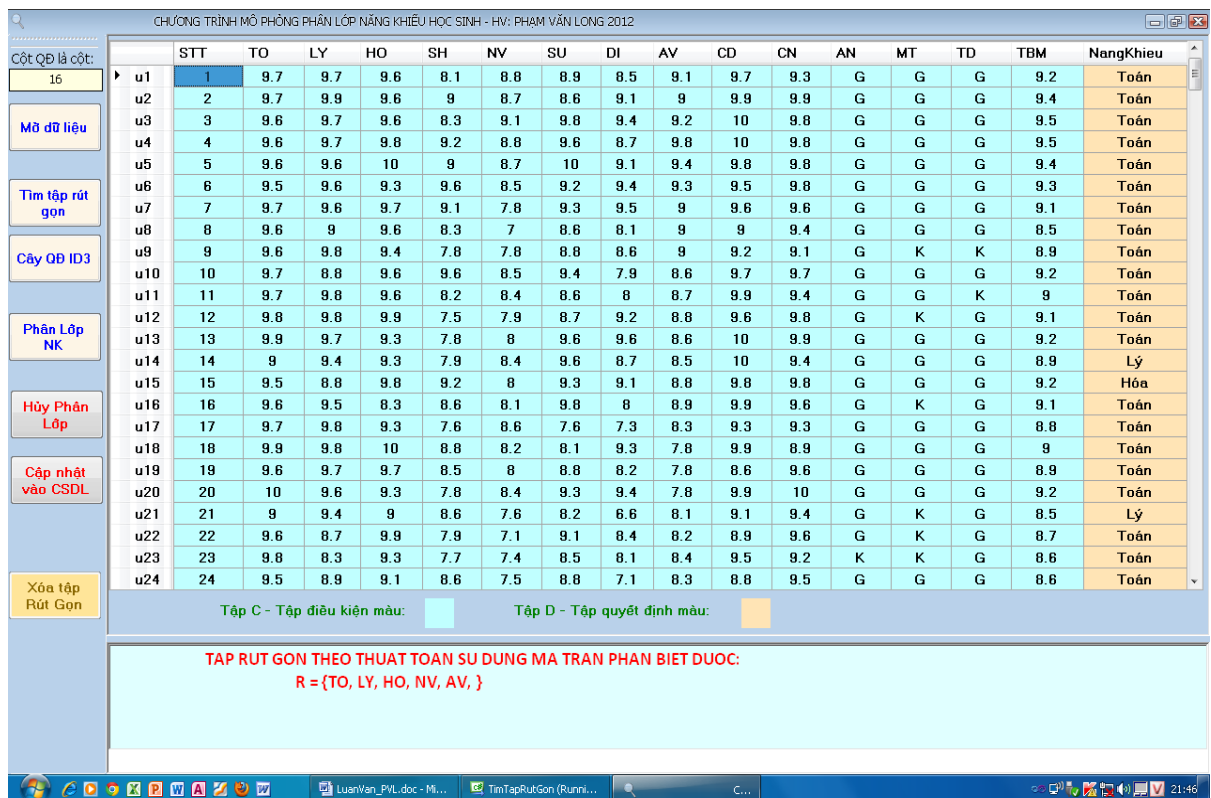
Hình 3.2 minh họa màn hình giao diện của chương trình

3.2.3 Chức năng mở dữ liệu



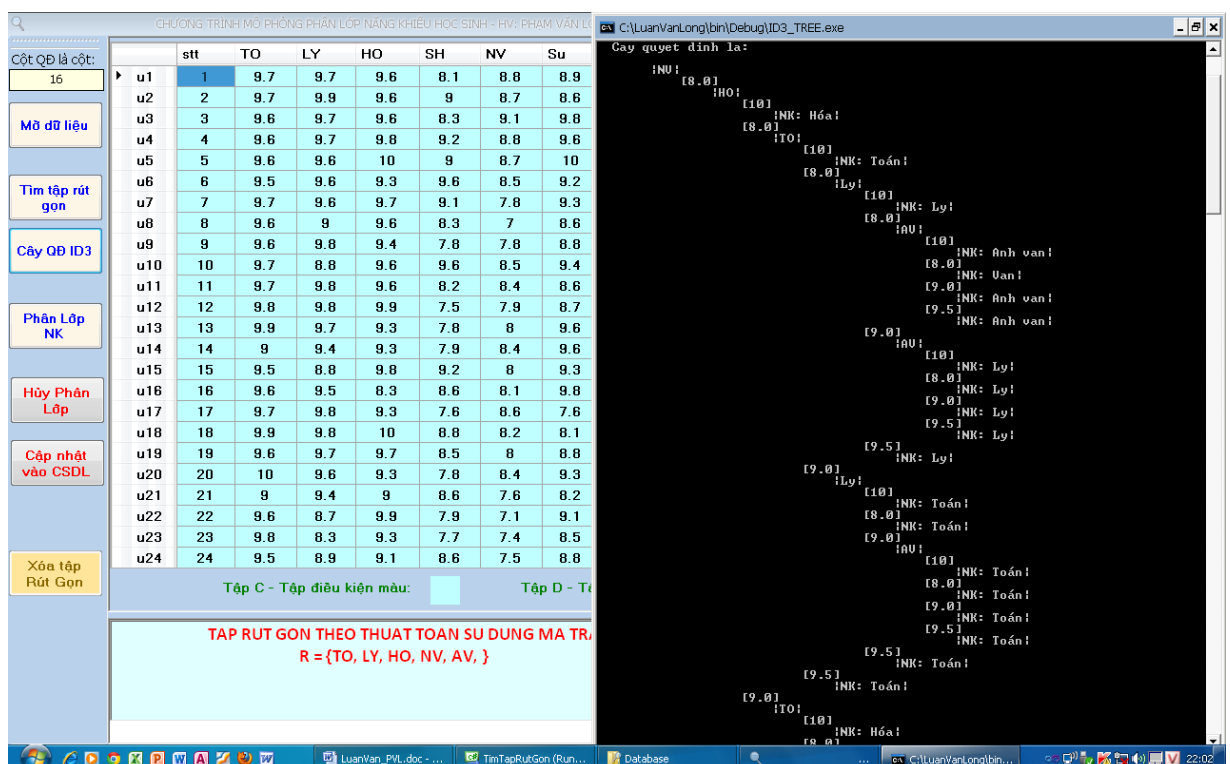
Hình 3.3 minh họa màn hình giao diện chức năng mở dữ liệu của chương trình

3.2.4 Chức năng tìm tập rút gọn



Hình 3.4 minh họa màn hình giao diện chức năng tìm tập rút gọn của chương trình

3.2.5 Chức năng tạo và hiển thị cây quyết định



Hình 3.5 minh họa màn hình giao diện chức năng tạo và hiển thị cây quyết định của chương trình

3.2.6 Chức năng phân lớp năng khiếu học sinh

CHƯƠNG TRÌNH MÔ PHỎNG PHÂN LỚP NĂNG KHIẾU HỌC SINH - HV: PHẠM VĂN LONG 2012

Cột QĐ là cột:

16

Mở dữ liệu

Tìm tập rút gọn

Cây QB ID3

Phân Lớp NK

Hủy Phân Lớp

Cập nhật vào CSDL

Xóa tập Rút Gọn

	stt	TO	LY	HO	SH	NV	Su	Di	AV	CD	CN	AN	MT	TD	TBM	NangKieu
u1	1	8.4	7.5	No	4.8	7.4	5.3	6.3	8.6	9.3	8.7	G	G	G	7.4	
u2	2	8	7.5	No	5.9	5.4	6.4	7.5	7.2	8.1	9.2	G	G	K	7.1	
u3	3	9	8	No	8.5	7.8	7.9	7.5	8.3	9.2	9.9	G	G	G	8.4	
u4	4	9.8	9.7	No	9.7	9.3	9.5	9.3	9.7	10	9.9	G	G	G	9.6	
u5	5	8.8	8.2	No	6.9	7.7	5.4	7.4	8.2	9.6	9.4	G	G	G	8	
u6	6	7	6.3	No	4.8	5.4	6.6	8.3	6.6	8.6	8.8	G	G	G	6.9	
u7	7	9	8.3	No	8.9	9	7.5	8.6	9.9	9.9	9.1	G	G	K	8.9	
u8	8	7.8	8.5	No	7.8	8	8.9	8.5	8.8	8	8.6	G	G	K	8.3	
u9	9	8.8	7.4	No	6.8	6.7	5.9	7.9	8	8.9	8.6	G	G	K	7.7	
u10	10	9.4	9.2	No	8.7	8.6	8.6	8.4	9.6	9.6	8.9	G	G	G	8.8	
u11	11	8.5	7	No	6.9	6	4.2	8.3	6.5	7.4	8.2	G	G	K	6.7	
u12	12	8.7	8.4	No	7.1	8	7.6	7.8	8.1	8.7	8.5	G	G	G	8.1	
u13	13	7.1	7.8	No	7.9	8	8.3	8.4	7.8	9.1	8.8	G	G	K	8	
u14	14	9.2	8.7	No	7	7.4	8.1	8.5	7.8	9.3	8.6	G	G	G	8.3	
u15	15	8.4	8.5	No	6.8	8.6	7.4	8.1	9	8.9	8.4	G	G	G	8.2	
u16	16	9.3	9.6	No	8.5	8.1	8.9	8.4	9	9.9	9.4	G	G	G	9	
u17	17	9.2	9	No	7.4	7.5	8.1	7.6	6.5	8.6	8.7	G	G	K	8.2	
u18	18	7.5	7	No	8.2	7.3	7.8	7.2	7	9.3	8.7	G	G	G	7.6	
u19	19	8.5	7.8	No	7.8	7.5	8.7	7.9	8.2	9.3	8.5	G	G	K	8.1	
u20	20	9.5	9.1	No	7.8	8.9	8.7	7.5	9.7	9.2	9	G	G	G	8.9	
u21	21	9.9	9.8	No	9.8	9.2	9.6	9.4	9.5	9.7	9.2	G	G	G	9.5	
u22	22	8.1	8.3	No	7.5	7.9	6.9	7.5	8.3	9.3	8.9	G	G	K	8.1	
u23	23	9.9	9.6	No	9.6	9.4	9.2	9.9	9.4	9.4	9.3	G	G	G	9.5	
u24	24	7.5	6	No	4.9	7.6	7.1	7.4	7.3	7.4	7.7	G	G	Tb	7	

Tập C - Tập điều kiện màu: Tập D - Tập quyết định màu:

21:55

Hình 3.6 minh họa màn hình giao diện chức năng phân lớp năng khiếu học sinh của chương trình

3.2.7 Luật quyết định tương ứng với cơ sở dữ liệu

Tri thức trình bày trong cây quyết định có thể rút ra và biểu diễn dưới dạng các luật IF...THEN. Mỗi đường đi từ nút gốc đến lá tạo nên một luật. Các cặp (thuộc tính, giá trị) dọc theo đường đi tạo thành một liên kết trong tiền đề luật phần IF. Nút lá với giá trị của thuộc tính dự đoán tạo nên phần THEN của luật. Các luật IF ... THEN giúp ta dễ hiểu hơn, đặc biệt nếu cây cho trước là rất lớn. Cụ thể là từ cây quyết định của thuật toán, chúng ta có thể rút ra được một số luật phổ biến như sau:

IF (NV, ≤ 8.5) AND (TO, ≥ 9.5) AND (LY, < 9.5) AND (HO, < 9.5) AND (AV, < 9.5) THEN (NangKhieu, Toán)

IF (NV, ≤ 8.5) AND (TO, < 9.5) AND (LY, ≥ 9.5) AND (HO, < 9.5) AND (AV, < 9.5) THEN (NangKhieu, Lý)

IF (NV, ≤ 8.5) AND (TO, < 9.5) AND (LY, < 9.5) AND (HO, < 9.5) AND (AV, > 9.5) THEN (NangKhieu, Anh Văn)

IF (NV, ≤ 8.5) AND (TO, < 9.5) AND (LY, < 9.5) AND (HO, ≥ 9.5) AND (AV, < 9.5) THEN (NangKhieu, Hóa)

IF (NV, ≤ 8.5) AND (TO, ≥ 9.0) AND (LY, < 9.0) AND (HO, < 9.0) AND (AV, < 9.0) THEN (NangKhieu, Toán)

IF (NV, ≤ 8.5) AND (TO, < 9.0) AND (LY, ≥ 9.0) AND (HO, < 9.0) AND (AV, < 9.0) THEN (NangKhieu, Lý)

IF (NV, ≤ 8.5) AND (TO, < 9.0) AND (LY, < 9.0) AND (HO, < 9.0) AND (AV, > 9.0) THEN (NangKhieu, Anh Văn)

IF (NV, ≤ 8.5) AND (TO, < 9.0) AND (LY, < 9.0) AND (HO, ≥ 9.0) AND (AV, < 9.0) THEN (NangKhieu, Hóa)

IF (NV, < 8.5) AND (NV, ≥ 8.0) AND (TO, ≥ 8.5) AND (LY, < 8.5) AND (HO, < 8.5) AND (AV, < 8.5) THEN (NangKhieu, Toán)

IF (NV, < 8.5) AND (NV, ≥ 8.0) AND (TO, < 8.5) AND (LY, ≥ 8.5) AND (HO, < 8.5) AND (AV, < 8.5) THEN (NangKhieu, Lý)

IF (NV, < 8.5) AND (NV, ≥ 8.0) AND (TO, < 8.5) AND (LY, < 8.5) AND (HO, ≥ 8.5) AND (AV, < 8.5) THEN (NangKhieu, Hóa)

IF (NV, < 8.5) AND (NV, ≥ 8.0) AND (TO, < 8.5) AND (LY, < 8.5) AND (HO, < 8.5) AND (AV, ≥ 8.5) THEN (NangKhieu, Anh Văn)

IF (NV, <8.5) AND (NV, >=8.0) AND (TO, < 8.5) AND (LY, < 8.5) AND
(HO, <8.5) AND (AV, < 8.5) THEN (NangKhieu, Vãn)

.....

Nhận xét : Các luật quyết định này có thể hỗ trợ phân lớp năng khiếu học sinh.

3.3. Kết luận chương 3

Trong chương này đã phát biểu bài toán để chọn môn học tự chọn phù hợp với năng khiếu của các em học sinh một cách khách quan dựa trên các cơ sở khoa học.

Tiến hành cài đặt và kiểm chứng chương trình mô phỏng phân lớp năng khiếu học sinh sử dụng thuật toán tìm tập rút gọn dựa trên ma trận phân biệt được dựa trên thuật toán xác định số cặp đối tượng phân biệt được đối với từng thuộc tính điều kiện kết hợp với thuật toán xây dựng cây quyết định ID3.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết luận :

Với mục tiêu đề ra ban đầu là “khai phá dữ liệu theo tiếp cận tập thô và cây quyết định - Ứng dụng trong phân lớp năng khiếu học sinh ”, luận văn đã giải quyết và đạt được một số kết quả như sau.

Hệ thống các kiến thức cơ bản về khai phá dữ liệu.

Hệ thống hóa các kiến thức cơ bản của lý thuyết tập thô, thuật toán tìm tập rút gọn dựa trên ma trận phân biệt được, trình bày trên từng ví dụ minh họa cụ thể để làm cơ sở cho các phương pháp xây dựng cây quyết định.

Luận văn đã trình bày các phương pháp xây dựng cây quyết định, và trình bày thuật toán xây dựng cây quyết định ID3, C4.5, FID3. Một số ví dụ minh họa cho các phương pháp xây dựng cây cũng được trình bày.

Luận văn đã cố gắng trình bày các thuật toán bằng ngôn ngữ lập trình C#, một tiến nhỏ được thêm vào nhằm làm cho các thuật toán đạt điều kiện dùng.

Hướng phát triển của đề tài:

Nghiên cứu các phương pháp xây dựng cây quyết định trên hệ thống thông tin không đầy đủ, dữ liệu không chắc chắn và cài đặt các phương pháp xây dựng cây quyết định trong trường hợp xử lý song song.

Tìm hiểu nhu cầu thực tế, cũng như tham khảo các ý kiến của chuyên gia để xây dựng chương trình áp dụng kỹ thuật đã nghiên cứu, bổ xung một số yếu tố khác ảnh hưởng đến môn năng khiếu của học sinh ngoài điểm tổng hợp trung bình của các môn từ có được sản phẩm thực sự hữu ích cho các trường bậc phổ thông.

TÀI LIỆU THAM KHẢO

Tiếng Việt

- [1]. Hồ Thuần, Hoàng Thị Lan Giao (2005), “Một thuật toán tìm tập rút gọn sử dụng ma trận phân biệt được”, Chuyên san các công trình nghiên cứu triển khai Viễn thông và CNTT, (15), tr. 83-87.
- [2]. Nguyễn Thanh Tùng (2009), “Một tiêu chuẩn mới chọn nút xây dựng cây quyết định”, Tạp chí Khoa học và Công nghệ, 47(2), tr. 15–25.

Tiếng Anh

- [3]. Andrzej Skowron, Ning Zong (2000). Rough Sets in KDD. Tutorial Notes.
- [4]. Ramadevi Yellasiri, C.R. Rao, Hara RamaKrishna, T. Prathima (2008), “Reduct based Decision Tree (RDT)”, International Journal of Soft Computing, 3(4), pp. 321-325.
- [5]. John Ross Quilan (1990), “Decision trees and decision making”, IEEE transactions on Man and Cybernetics, (20), pp. 339-346.
- [6]. Zdzisław Pawlak (1998) - Rough Set Theory and Its Application to Data Analysis, Cybernetics and Systems: An International Journal 29, pp. 661-688.
- [7]. Ho Tu Bao (1996). Introduction to Knowledge Discovery and Data mining. Institute of Information Technology National Center for Natural Science and Technology

PHỤ LỤC

Mã nguồn của thuật toán tìm tập rút gọn sử dụng ma trận phân biệt được dựa trên số cặp phân biệt được và thuật toán xây dựng cây quyết định ID3

```
using System;
using System.Collections.Generic;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.OleDb;

namespace TimTapRutGon
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();

            //Hàm này cho phép loại bỏ các giá trị trùng lặp (trùng thì chỉ chọn một giá trị)
            private List<string> SelectDistinct(Hashtable hash)
            {
                //Bảng hashtable chứa hai vùng giá trị: key-lưu chỉ số của cột; value-lưu giá trị trong cột
                List<string> temp = new List<string>();
                List<string> fromHash = new List<string>();
                //Chuyển từ các value trong bảng hashtable sang list để thuận tiện hơn cho việc xử lý bằng LINQ
                foreach (DictionaryEntry s in hash)
                {
                    fromHash.Add(s.Value.ToString());
                }
                //Xử lý bằng LINQ: chọn các giá trị trong fromHash không trùng lặp
                var x = (from p in fromHash
                    select p).Distinct();
                //Trả về danh sách các giá trị sau khi loại bỏ những phần trùng lặp
                foreach (string t in x)
                {
                    temp.Add(t);
                }
                return temp;
            }

            //Sự kiện khi Form chạy
            private void Form1_Load(object sender, EventArgs e)
            {
                //Đánh dấu tập D - tập decision set (tập quyết định)
                textBox1_TextChanged_1(sender, e);

                //Ghi vào tiêu đề mỗi dòng như trong ví dụ: u1, u2,...
                for (int i = 0; i < dataGridView1.Rows.Count; i++)
                {
                    dataGridView1.Rows[i].HeaderCell.Value = "u" + (i + 1).ToString();
                }

                //Phép hiệu của hai tập hợp. Ví dụ: M={1, 2, 3}, N={2, 4} thì M\N={1, 3}
                private List<int> Subtraction(List<int> M, List<int> N)
                {
                    /*
```

* Với mỗi phần tử i trong tập M , với mỗi phần tử j trong tập N ; nếu hai phần tử i và j khác nhau
 * thì ta bổ sung nó vào trong tập kết quả; ngược lại ta loại nó ra (dựa vào biến ok)

```

*/
List<int> temp = new List<int>();
foreach (int i in M)
{
    bool ok = false;
    foreach (int j in N)
        if (i == j)
        {
            ok = true;
            break;
        }
    if (!ok)
        temp.Add(i);
}
return temp;
}

//Phân hoạch các tập theo giá trị
private List<List<int>> Partition(Hashtable M)
{
    List<List<int>> t = new List<List<int>>();
    //lvalues lưu các giá trị không trùng lặp
    List<string> lvalues = SelectDistinct(M);
    foreach (string val in lvalues)
    {
        List<int> list = new List<int>();
        foreach (DictionaryEntry d in M)
        {
            if (d.Value.Equals(val))
                list.Add((int)d.Key);
        }
        t.Add(list);
    }
    return t;
}

//Đếm các giá trị theo nhóm đã select distinct
private int countValues(List<int> list, string s, Hashtable x)
{
    int count = 0;
    foreach (int t in list)
        if (x[t].Equals(s))
            count++;
    return count;
}

//Tìm giá trị nhỏ nhất trong một dãy các giá trị
private int min(List<double> list)
{
    double temp = list[0];
    int index = 0;
    for (int i = 1; i < list.Count; i++)
        if (list[i] < temp)
        {
            index = i;
            temp = list[i];
        }
    return index;
}

```

```

bool isDuplicate(int x, List<int> list)
{
    foreach (int y in list)
        if (x == y)
            return true; //Lặp
    return false; //Không lặp
}

private void TapRutGonTheoMaTranPhanBietDuoc()
{
    richTextBox1.Text += "\t\t TAP RUT GON THEO THUAT TOAN SU DUNG MA TRAN
    PHAN BIET DUOC:\n \t\t\t\t R = ";

    List<Hashtable> c = new List<Hashtable>();
    for (int i = 0; i < dataGridView1.Columns.Count; i++)
        c.Add(new Hashtable());
    for (int i = 1; i <= dataGridView1.Rows.Count; i++)
        for (int j = 0; j < c.Count; j++)
            c[j][i] = dataGridView1.Rows[i - 1].Cells[j].Value;

    List<int> E = new List<int>(); //Universe set
    for (int i = 1; i <= dataGridView1.Columns.Count; i++)
        E.Add(i);
    List<int> D = new List<int>(); //Decision set
    for (int i = 0; i < textBox1.Lines.Length; i++)
        D.Add(Convert.ToInt16(textBox1.Lines[i]));
    List<int> C = Subtraction(E, D); //Conditional set

    List<int> R = new List<int>();
    List<int> U = new List<int>();
    for (int i = 1; i <= dataGridView1.Rows.Count; i++)
        U.Add(i);

    List<List<int>>> L = new List<List<int>>>();
    L.Add(U);
    //-----
    do //--Tương đương repeat
    {
        int alphaIndex = 0;
        //List<List<double>>> mygama = new List<List<double>>>();
        //List<double> gama = new List<double>();
        List<double> alpha = new List<double>();
        List<List<List<int>>>> multibounds = new List<List<List<int>>>>();
        List<List<double>>> mygama = new List<List<double>>>();
        foreach (int cj in Subtraction(C, R)) //--Tương đương for cj in C\R
        {
            List<List<int>>> bound = new List<List<int>>>();
            List<double> gama = new List<double>();
            foreach (List<int> Xi in L) //--Tương đương Xi in L
            {
                List<double> listomega = new List<double>();
                Hashtable temp = new Hashtable();
                foreach (int index in Xi)
                    temp[index] = c[cj - 1][index];
                bound = Partition(temp);
                multibounds.Add(bound);
                double omega;
                for (int j = 0; j < bound.Count; j++)
                {
                    List<string> valD = SelectDistinct(c[D[0] - 1]);
                    //Have to change D[0] into other. In this case, D just
                    contain 1 column.
                    omega = 0;
                }
            }
        }
    }
}

```



```

1]), 2);

        foreach (string s in valD)
            omega += Math.Pow(countValues(bound[j], s, c[D[0] -

            omega = Math.Pow(bound[j].Count, 2) - omega;
            omega /= 2;
            listomega.Add(omega);
        }//--end of for j
        double tgama = 0;
        foreach (double xgama in listomega)
            tgama += xgama;

        gama.Add(tgama);
    }
    mygama.Add(gama);
    double a1 = 0;
    foreach (double a2 in gama)
        a1 += a2;
    alpha.Add(a1);
}

alphaindex = min(alpha);
while (isDuplicate(alphaindex + 1, R))
{
    alpha.RemoveAt(alphaindex);
    alphaindex = min(alpha);
}
R.Add(alphaindex + 1); //Vi chi so bat dau tu 0
L = multibounds[alphaindex];
if (alpha[alphaindex] == 0)
    break;
} while (true);
//----- In kết quả -----

//richTextBox1.Text += "Cac tap rut gọn la: \n\t{";

if (dataGridView1.Columns.Count < 15)
{
    richTextBox1.Text += "{";

    foreach (int x in R)
        richTextBox1.Text += dataGridView1.Columns[x - 1].HeaderText +

", ";
}
if (dataGridView1.Columns[1].HeaderText.ToString() == "TO" )
{
    List<int> lo = new List<int>();
    lo.Add(2); lo.Add(3); lo.Add(4); lo.Add(6); lo.Add(9);
    richTextBox1.Text += "{";
    foreach (int x in lo)
        richTextBox1.Text += dataGridView1.Columns[x - 1].HeaderText + ", ";
}

//Chỉ số x và y trừ đi 1 vì nó bắt đầu từ 1, nhưng GridView lại bắt đầu
từ 0 (Phần header không được tính vào chỉ số).
richTextBox1.Text += "}\n";
}

private List<int> Subtraction1(List<int> list, int c)
{
    List<int> temp = new List<int>();
    foreach (int cj in list)
        if (cj != c)
            temp.Add(cj);
    return temp;
}

```

```

    }

    private bool CompareList(List<string> listA, List<string> listB)
    {
        for (int i = 0; i < listA.Count; i++)
            if (!listA[i].Equals(listB[i]))
                return false;
        return true;
    }

    //=====CHẠY THUẬT TOÁN TÌM TẬP RÚT GỌN THEO MA TRẬN PHÂN BIỆT ĐƯỢC
    private void TimRutGon_Click(object sender, EventArgs e)
    {
        TapRutGonTheoMaTranPhanBietDuoc();
        CayQD.Enabled = true;
    }

    private void textBox1_TextChanged_1(object sender, EventArgs e)
    {
        try
        {
            for (int i = 0; i < dataGridView1.Columns.Count; i++)
                dataGridView1.Columns[i].DefaultCellStyle.BackColor =
panel1.BackColor;
            for (int i = 0; i < textBox1.Lines.Length; i++)
                dataGridView1.Columns[Convert.ToInt16(textBox1.Lines[i])
1].DefaultCellStyle.BackColor = panel2.BackColor;

        }
        catch (Exception) { };
    }
    private void button1_Click(object sender, EventArgs e)
    {
        System.Diagnostics.Process.Start(@"C:\LuanVanLong\bin\Debug\ID3_TREE.exe");
    }
    //OleDbConnection con;
    OleDbDataAdapter dAdapter;
    DataTable dTable;
    private void LoadDaTa_Click(object sender, EventArgs e)
    {
        if (openFileDialog1.ShowDialog() == DialogResult.OK)
        {
            //Tạo kết nối
            string connString = "Provider=Microsoft.ACE.OLEDB.12.0; Data
Source=" + openFileDialog1.FileName;
            OleDbConnection con = new OleDbConnection(connString);
            con.Open();

            //Tìm bảng dữ liệu trong file Access
            DataTable tables = con.GetOleDbSchemaTable(OleDbSchemaGuid.Tables,
new object[] { null, null, null, "TABLE" });
            string tbl_name = "";
            foreach (DataRow row in tables.Rows)
                tbl_name = row[2].ToString();

            //Truy vấn dữ liệu và hiển thị ra dataGridView
            string query = "SELECT * FROM " + tbl_name;
            dAdapter = new OleDbDataAdapter(query, connString);
            //OleDbDataAdapter dAdapter = new OleDbDataAdapter(query,
connString);

```

```

        OleDbCommandBuilder cBuilder = new OleDbCommandBuilder(dAdapter);
        dTable = new DataTable();
        dAdapter.Fill(dTable);
        BindingSource bSource = new BindingSource();
        bSource.DataSource = dTable;
        dataGridView1.DataSource = bSource;
        dataGridView1.AutoResizeColumns();

        for (int j = 0; j < Convert.ToInt16(dataGridView1.Columns.Count) ; j++)
        {
            dataGridView1.Columns[j].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleCenter;
        }

        Form1_Load(sender, e); //Hiển thị màu hồng cho tập D và tiêu đề dòng là u1,
u2,...

        con.Close();
        textBox1.Text = dataGridView1.Columns.Count.ToString();

        CayQD.Enabled = true;
        TimRutGon.Enabled = true;
        button4.Enabled = false;
        label4.Visible = true;
        label5.Visible = true;
        panel1.Visible = true;
        panel2.Visible = true;
    }
}

/*****
private void CacellDD_Click(object sender, EventArgs e)
{
    int D = Convert.ToInt16(dataGridView1.Columns.Count) -1;
    for (int i = 0; i < dataGridView1.Rows.Count; i++)
        dataGridView1.Rows[i].Cells[D].Value = "";
    OleDbCommandBuilder db = new OleDbCommandBuilder(dAdapter);
    dAdapter.MissingSchemaAction = MissingSchemaAction.AddWithKey;

    dAdapter.Update(dTable);
}

/*****
**/

private void UpDateCSDL_Click(object sender, EventArgs e)
{
    //DataTable dt = (DataTable)dataGridView1.DataSource;
    OleDbCommandBuilder db = new OleDbCommandBuilder(dAdapter);
    dAdapter.MissingSchemaAction = MissingSchemaAction.AddWithKey;

    dAdapter.Update(dTable);
}

private void button4_Click(object sender, EventArgs e)
{
    richTextBox1.Clear();
}

private void button5_Click(object sender, EventArgs e)
{
    double T, L, H, V, A;
    string D1 = "Văn", D2 = "Toán", D3 = "Lý", D4 = "Hóa", D5 = "Anh văn";
    int D = Convert.ToInt16(dataGridView1.Columns.Count) - 1;

```

```

for (int i = 0; i < dataGridView1.Rows.Count; i++)
{
    T = Convert.ToDouble(dataGridView1.Rows[i].Cells[1].Value);
    L = Convert.ToDouble(dataGridView1.Rows[i].Cells[2].Value);
    A = Convert.ToDouble(dataGridView1.Rows[i].Cells[8].Value);
    V = Convert.ToDouble(dataGridView1.Rows[i].Cells[5].Value);
    if (dataGridView1.Rows[i].Cells[3].Value.ToString() == "No")
        H = 0;
    else H = Convert.ToDouble(dataGridView1.Rows[i].Cells[3].Value);

    if (V >= 8.5 & T < 9.5 & L < 9.5 & H < 9.5 & A < 9.5)
        dataGridView1.Rows[i].Cells[D].Value = D1;
    else if (V < 8.5 & T > 9.5)
        dataGridView1.Rows[i].Cells[D].Value = D2;
    else if (V < 8.5 & L > 9.5)
        dataGridView1.Rows[i].Cells[D].Value = D3;
    else if (V < 8.5 & A > 9.5)
        dataGridView1.Rows[i].Cells[D].Value = D5;
    else if (V < 8.5 & H > 9.5)
        dataGridView1.Rows[i].Cells[D].Value = D4;
    else if (V < 8.5 & T > 9)
        dataGridView1.Rows[i].Cells[D].Value = D2;
    else if (V < 8.5 & L > 9)
        dataGridView1.Rows[i].Cells[D].Value = D3;
    else if (V < 8.5 & A > 9)
        dataGridView1.Rows[i].Cells[D].Value = D5;
    else if (V < 8.5 & H > 9)
        dataGridView1.Rows[i].Cells[D].Value = D4;
    else if (V < 8.5 & V >= 8 & T > 8.5)
        dataGridView1.Rows[i].Cells[D].Value = D2;
    else if (V < 8.5 & V >= 8 & L > 8.5)
        dataGridView1.Rows[i].Cells[D].Value = D3;
    else if (V < 8.5 & V >= 8 & A > 8.5)
        dataGridView1.Rows[i].Cells[D].Value = D5;
    else if (V < 8.5 & V >= 8 & H > 8.5)
        dataGridView1.Rows[i].Cells[D].Value = D4;
    else if (V < 8.5 & V >= 8)
        dataGridView1.Rows[i].Cells[D].Value = D1;
    else if (V < 8.5 & V >= 7.5 & T > 8)
        dataGridView1.Rows[i].Cells[D].Value = D2;
    else if (V < 8.5 & V >= 7.5 & L > 8)
        dataGridView1.Rows[i].Cells[D].Value = D3;
    else if (V < 8.5 & V >= 7.5 & A > 8)
        dataGridView1.Rows[i].Cells[D].Value = D5;
    else if (V < 8.5 & V >= 7.5 & H > 8)
        dataGridView1.Rows[i].Cells[D].Value = D4;
    else if (V < 8 & V >= 7.5)
        dataGridView1.Rows[i].Cells[D].Value = D1;
    else if (T >= 7.5)
        dataGridView1.Rows[i].Cells[D].Value = D2;
    else if (A >= 7.5)
        dataGridView1.Rows[i].Cells[D].Value = D5;
    else if (L >= 7.5)
        dataGridView1.Rows[i].Cells[D].Value = D3;
    else if (H >= 7.5)
        dataGridView1.Rows[i].Cells[D].Value = D4;
}

CayQD.Enabled = false;
TimRutGon.Enabled = false;
button4.Enabled = false;

```

```

}
#region Phần trang trí!!!!
private void toolStripButton2_Click(object sender, EventArgs e)
{
    richTextBox1.Clear();
}

private void toolStripButton1_MouseHover(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "Mở File dữ liệu Access";
}

private void toolStripButton1_MouseLeave(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "";
}

private void toolStripButton2_MouseHover(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "Xóa kết quả đã thực thi";
}

private void toolStripButton2_MouseLeave(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "";
}

private void textBox1_MouseHover(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "Tập D - Tập quyết định";
}

private void textBox1_MouseLeave(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "";
}

private void toolStripComboBox1_MouseHover(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "Chọn giải thuật để thực thi";
}

private void toolStripComboBox1_MouseLeave(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "";
}

private void panel1_MouseHover(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "Chọn màu cho tập điều kiện - Double Click
lên đây !";
}

private void panel1_MouseLeave(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "";
}

private void panel2_MouseHover(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "Chọn màu cho tập quyết định - Double Click
lên đây !";
}

```

```

private void panel2_MouseLeave(object sender, EventArgs e)
{
    toolStripStatusLabel1.Text = "";
}

private void panel1_DoubleClick_1(object sender, EventArgs e)
{
    if (colorDialog1.ShowDialog() == DialogResult.OK)
        panel1.BackColor = colorDialog1.Color;
    textBox1_TextChanged_1(sender, e);
}

private void panel2_DoubleClick(object sender, EventArgs e)
{
    if (colorDialog1.ShowDialog() == DialogResult.OK)
        panel2.BackColor = colorDialog1.Color;
    textBox1_TextChanged_1(sender, e);
}
#endregion
}
}

namespace ID3_DECISION_TREE
{
    /* Lớp Attribute: mName: Tên thuộc tính; mValues: Giá trị của thuộc tính; mLabel:
    Lưu nhãn lớp của thuộc tính, nếu nó là thuộc tính quyết định */

    public class toancuc
    {
        public static int pt;    // dung pt lam bien toan cuc
    }

    public class Attribute
    {
        ArrayList mValues;
        string mName;
        object mLabel;
        /* Khởi tạo một lớp Attribute có kiểu Object: mLabel = Label; Tên thuộc tính
        bằng rỗng; Giá trị bằng null */
        public Attribute(string name, string[] values)
        {
            mName = name;
            mValues = new ArrayList(values);
            mValues.Sort();
        }
        //Phương thức gán nhãn lớp, tức thuộc tính này là nút lá
        public Attribute(object Label)
        {
            mLabel = Label;
            mName = string.Empty;
            mValues = null;
        }
        //Phương thức AttributeName: Lấy tên của thuộc tính
        public string AttributeName
        {
            get { return mName; }
        }
        //Phương thức trả về mảng các giá trị của thuộc tính mValues
        public string[] values
        {
            get
            {
                if (mValues != null)
                    return (string[])mValues.ToArray(typeof(string));
            }
        }
    }
}

```

```

        else
            return null;
    }
}
//Phương thức trả về vị trí của value trong mValue.
public int indexValue(string value)
{
    if (mValues != null)
        return mValues.BinarySearch(value);
    else
        return -1;
}

//Phương thức trả về tên của thuộc tính
public override string ToString()
{
    if (mName != string.Empty)
        return mName;
    else return mLabel.ToString();
}
}
// *****Lớp xây dựng cây quyết định*****
public class TreeNode
{
    private ArrayList mChilds = null;
    private Attribute mAttribute;
    /* Khởi tạo cây quyết định. Thuộc tính với giá trị của nó được tạo. Nếu giá
    trị của thuộc tính khác null thì tạo các nhánh ứng với mỗi giá trị của thuộc tính
    có giá trị bằng null, Ngược lại một nút chưa có giá trị thì nhánh tạo ra có
    một nút có giá trị null*/

    public TreeNode(Attribute attribute)
    {
        if (attribute.values != null)
        {
            mChilds = new ArrayList(attribute.values.Length);
            for (int i = 0; i < attribute.values.Length; i++)
                mChilds.Add(null);
        }
        else
        {
            mChilds = new ArrayList(1);
            mChilds.Add(null);
        }
        mAttribute = attribute;
    }
    // Thêm một nút con vào nhánh có giá trị ValueName
    public void AddTreeNode(TreeNode treeNode, string ValueName)
    {
        int index = mAttribute.indexValue(ValueName);
        mChilds[index] = treeNode;
    }
    //Trả về tên thuộc tính
    public Attribute attribute
    {
        get { return mAttribute; }
    }
    //Trả về nút con của nút có tên là nhánh dẫn đến nó.
    public TreeNode getChildByBranchName(string branchName)
    {
        int index = mAttribute.indexValue(branchName);
        return (TreeNode)mChilds[index];
    }
}

```

```

/*Lớp xây dựng cây quyết định của thuật toán BuildDecisionTree*/
public class DecisionTree
{
    private DataTable mSamples;
    private int mTotalPositives = 0;
    private int mTotal = 0;
    private string mTargetAttribute = "NangKhieu";
    private double mEntropySet = 0.0;
    //Trả về số phần tử True trong bảng quyết định
    private int countTotalPositives(DataTable samples)
    {
        int result = 0;
        foreach (DataRow aRow in samples.Rows)
        {
            if ((string)aRow[mTargetAttribute] == "Co")
                result++;
        }
        return result;
    }
    /* Duyệt qua bảng và kiểm tra thuộc tính có giá trị là value và trả về số
    phần tử True và số phần tử âm. */
    private void getValuesToAttribute(DataTable samples, Attribute attribute,
    string value, out int positives, out int negatives)
    {
        positives = 0;
        negatives = 0;
        foreach (DataRow aRow in samples.Rows)
        {
            if (((string)aRow[attribute.AttributeName] == value))
            {
                if ((string)aRow[mTargetAttribute] == "Co")
                    positives++;
                else
                    negatives++;
            }
        }
    }
    //Phương thức tính entropy  $-p \log(p, 2) + p \log(p, 2)$ 
    private double calcEntropy(int positives, int negatives)
    {
        int total = positives + negatives;
        double ratioPositive = (double)positives / total;
        double ratioNegative = (double)negatives / total;
        /* Cây sẽ ngưng làm việc khi phát hiện root.Attribute.value chứa giá
    trị null*/
        if (total == 0)
            return 0;
        if (ratioPositive != 0)
            ratioPositive = -(ratioPositive) * System.Math.Log(ratioPositive,
    2);
        if (ratioNegative != 0)
            ratioNegative = -(ratioNegative) * System.Math.Log(ratioNegative,
    2);
        double result = ratioPositive + ratioNegative;
        return result;
    }
    //Phương thức tính lượng thông tin thu thêm (IG)
    private double gain(DataTable samples, Attribute attribute)
    {
        string[] values = attribute.values;
        double sum = 0.0;
        for (int i = 0; i < values.Length; i++)
        {
            int positives, negatives;
            positives = negatives = 0;

```



```

        getValuesToAttribute(samples, attribute, values[i], out positives,
out negatives);
        double entropy = calcEntropy(positives, negatives);
        sum += -(double)(positives + negatives) / mTotal * entropy;
    }
    return mEntropySet + sum;
}
//Chọn thuộc tính có IG lớn nhất, trả về thuộc tính tốt nhất để phân nhánh
cây quyết định
private Attribute getBestAttribute(DataTable samples, Attribute[]
attributes)
{
    double maxGain = 0.0;
    Attribute result = null;
    foreach (Attribute attribute in attributes)
    {
        double aux = gain(samples, attribute);
        if (aux > maxGain)
        {
            maxGain = aux;
            result = attribute;
        }
    }
    if (maxGain == 0)
        result = attributes[0];
    return result;
}
//Trả về True nếu tất cả các mẫu cùng lớp True
private bool allSamplesPositives(DataTable samples, string targetAttribute)
{
    foreach (DataRow row in samples.Rows)
    {
        if ((string)row[targetAttribute] != "Co")
            return false;
    }
    return true;
}
//Trả về False nếu tất cả các mẫu cùng lớp False
private bool allSamplesNegatives(DataTable samples, string targetAttribute)
{
    foreach (DataRow row in samples.Rows)
    {
        if ((string)row[targetAttribute] != "Khong")
            return false;
    }
    return true;
}
//Trả về nhãn lớp chiếm đa số trong samples
private ArrayList getDistinctValues(DataTable samples, string
targetAttribute)
{
    ArrayList distinctValues = new ArrayList(samples.Rows.Count);
    foreach (DataRow row in samples.Rows)
    {
        if (distinctValues.IndexOf(row[targetAttribute]) == -1)
            distinctValues.Add(row[targetAttribute]);
    }
    return distinctValues;
}
//Đếm số các mẫu thuộc chiếm đa số trong samples.
private object getMostCommonValue(DataTable samples, string targetAttribute)
{
    ArrayList distinctValues = getDistinctValues(samples, targetAttribute);
    int[] count = new int[distinctValues.Count];

```

```

foreach (DataRow row in samples.Rows)
{
    int index = distinctValues.IndexOf(row[targetAttribute]);
    count[index]++;
}
int MaxIndex = 0;
int MaxCount = 0;
for (int i = 0; i < count.Length; i++)
{
    if (count[i] > MaxCount)
    {
        MaxCount = count[i];
        MaxIndex = i;
    }
}
return distinctValues[MaxIndex];
}
public TreeNode mountTree(DataTable samples, string targetAttribute,
Attribute[] attributes)
{
    mSamples = samples;
    return internalMountTree(mSamples, targetAttribute, attributes);
}
//Phương thức lập xây dựng cây quyết định
private TreeNode internalMountTree(DataTable samples, string
targetAttribute, Attribute[] attributes)
{
    /*Nếu tất cả các mẫu cùng thuộc một lớp True thì tạo nút lá có nhãn
lớp là True*/
    if (allSamplesPositives(samples, targetAttribute) == true)
        return new TreeNode(new Attribute(true));
    /*Nếu tất cả các mẫu cùng thuộc một lớp False thì tạo nút lá có nhãn lớp
là False*/

    if (allSamplesNegatives(samples, targetAttribute) == true)
        return new TreeNode(new Attribute(false));

    /* Nếu không còn thuộc tính điều kiện nào thì tạo nhãn lớp có số lần
xuất hiện lớn nhất của thuộc tính quyết định */

    if (attributes.Length == 0)
        return new TreeNode(new Attribute(getMostCommonValue(samples,
targetAttribute)));
    mTotal = samples.Rows.Count;
    mTargetAttribute = targetAttribute;
    mTotalPositives = countTotalPositives(samples);
    mEntropySet = calcEntropy(mTotalPositives, mTotal - mTotalPositives);
    Attribute bestAttribute = getBestAttribute(samples, attributes);
    TreeNode root = new TreeNode(bestAttribute);
    DataTable aSample = samples.Clone();
    //Phương thức gắn nút vào cây Root
    foreach (string value in bestAttribute.values)
    { // Xóa tất cả các dòng trong bảng asample.
        aSample.Rows.Clear();
        DataRow[] rows = samples.Select(bestAttribute.AttributeName + " = "
+ "'" + value + "'");
        /*Chọn tất cả các dòng có điều kiện thuộc tính tốt nhất bằng giá trị
value của nó. Phân hoạch của thuộc tính điều kiện tốt nhất theo tập các giá trị của
nó.*/

        foreach (DataRow row in rows)
        {
            aSample.Rows.Add(row.ItemArray);
        }
    }
}

```

```

        /*Tạo danh sách các thuộc tính điều kiện loại thuộc tính tốt nhất
được chọn*/
        ArrayList aAttributes = new ArrayList(attributes.Length - 1);
        for (int i = 0; i < attributes.Length; i++)
        {
            if (attributes[i].AttributeName != bestAttribute.AttributeName)
                aAttributes.Add(attributes[i]);
        }
        /* Nếu không còn đối tượng nào thì một nút lá được khởi tạo với nhãn
lớp có giá trị chung nhất (nhiều nhất)*/
        if (aSample.Rows.Count == 0)
            root.AddTreeNode(new TreeNode(new
Attribute(getMostCommonValue(samples, targetAttribute)), value);
        else
        {
            DecisionTree dc3 = new DecisionTree();
            TreeNode ChildNode = dc3.mountTree(aSample, targetAttribute,
(Attribute[])aAttributes.ToArray(typeof(Attribute)));
            root.AddTreeNode(ChildNode, value);
        }
        return root;
    }
}
class AlgorithmSample
{
    public static int SoNut = 1;
    //Phương thức in các nút trong cây.

    public static void printNode(TreeNode root, string tabs)
    {
        if (root.attribute.ToString().ToLower() != "co" &&
root.attribute.ToString().ToLower() != "khong")
        {
            Console.WriteLine(tabs + '|' + root.attribute + '|');
        }
        if (root.attribute.values != null)
        {
            for (int i = 0; i < root.attribute.values.Length; i++)
            {
                // toancuc.KQT = toancuc.KQT + tabs + "\t" + "<" +
root.attribute.values[i] + ">";

                Console.WriteLine(tabs + "\t" + " [" +
root.attribute.values[i] + "]");
                TreeNode childNode =
root.GetChildByBranchName(root.attribute.values[i]);
                printNode(childNode, "\t" + tabs);
            }
        }
    }
    //Phương thức đếm số nút trong cây
    public static int CalCountNode(TreeNode root)
    {
        if (root.attribute.values != null)
        {
            for (int i = 0; i < root.attribute.values.Length; i++)
            {
                SoNut++;
            }
        }
    }
}

```

```

TreeNode childNode =
root.GetChildByBranchName(root.attribute.values[i]);
    CalCountNode(childNode);
    }
    }
    return SoNut;
}

// *****Phương thức kết nối cơ sở dữ liệu*****

static DataTable GetData()
{
    string connString;
    string query;
    Console.WriteLine("\n Chon tap tin du lieu");
    Console.WriteLine("\n 1. Nhap so 1 de chon CSDL minh hoa cho cay QD Vi
du 2.1");
    Console.WriteLine("\n 2. Nhap so 2 de chon CSDL Diem tong hop (DL
Hoc)");

    string chon = "";
    chon = Console.ReadLine();
    int ch = Int32.Parse(chon);
    if (ch == 1)
    {
        toancuc.pt = 1; //co 5 thuoc tinh quyet dinh la 5 mon nang khieu con
lai la thuoc tinh dieu kien

        connString = "Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=C:\\LuanVanLong\\Database\\Playteniss.accdb;Persist Security Info=False";
    }
    else
    {
        toancuc.pt = 1; //co 1 thuoc tinh la quyet dinh con lai la thuoc
tinh dieu kien
        connString = "Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=C:\\LuanVanLong\\Database\\DataID3.accdb;Persist Security Info=False";

    }

    //toancuc.pt = 1; //co 1 thuoc tinh la quyet dinh con lai la thuoc tinh
dieu kien
    // connString = "Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=C:\\LuanVanLong\\Database\\DataID3.accdb;Persist Security Info=False";
    OleDbConnection con = new OleDbConnection(connString);
    con.Open();

    //*****Tìm bảng dữ liệu trong file Access*****

    DataTable tables = con.GetOleDbSchemaTable(OleDbSchemaGuid.Tables,
new object[] { null, null, null, "TABLE" });
    string tbl_name = "";
    foreach (DataRow row in tables.Rows)
        tbl_name = row[2].ToString();

    //query = "SELECT T, L, H, V, NN, NK FROM " + tbl_name;

    //query = "SELECT TO, LY, HO, NV, AV, NangKhieu FROM " + tbl_name;
    query = "SELECT * FROM " + tbl_name;
    OleDbDataAdapter dAdapter = new OleDbDataAdapter(query, connString);
    OleDbCommandBuilder cBuilder = new OleDbCommandBuilder(dAdapter);

```

```

        DataTable result = new DataTable();
        dAdapter.Fill(result);
        con.Close();
        return result;
    }

    static ArrayList getDistinctValue(DataTable samples, string targetAttribute)
    {
        ArrayList distinctValues = new ArrayList(samples.Rows.Count);
        foreach (DataRow row in samples.Rows)
        {
            if (distinctValues.IndexOf(row[targetAttribute]) == -1)
                distinctValues.Add(row[targetAttribute]);
        }
        return distinctValues;
    }

    static void Main(string[] args)
    {
        DataTable samples = GetData();

        int numberColumn = samples.Columns.Count; //tong so thuoc tinh trong
        bang du lieu

        int sl; //sl chinh la so thuoc tinh lam thuoc tinh quyet dinh
        sl = toancuc.pt;

        Attribute[] attributes = new Attribute[numberColumn-sl];
        for (int i = 0; i < numberColumn -sl; i++)
        {
            ArrayList temp = getDistinctValue(samples,
            samples.Columns[i].ColumnName);
            attributes[i] = new Attribute(samples.Columns[i].ColumnName,
            (string[])temp.ToArray(typeof(string)));
        }

        for (int i = samples.Columns.Count - sl; i < numberColumn; i++)
        {
            //Console.WriteLine("\n Cay quyet dinh la | {0} |
            \n",samples.Columns[i].ColumnName);
            Console.WriteLine("\n Cay quyet dinh la: \n");

            DecisionTree BuildDecisionTree = new DecisionTree();

            TreeNode root = BuildDecisionTree.mountTree(samples,
            samples.Columns[i].ColumnName, attributes);
            printNode(root, " ");
            Console.WriteLine("\n Enter de tiep tục");
            Console.Read();
            Console.Read();
        }

        Console.Read();
    }
}

```

