การสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้

นางสาวฝนทิพย์ คูณแก้ว

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์
มหาวิทยาลัยเทคโนโลยีสุรนารี
ปีการศึกษา 2555

CLASSIFICATION MODEL INDUCTION BASED ON USER PREFERENCES

Fonthip Koongaew

A Thesis Submitted in Partial Fulfillment of the Requirements for the

Degree of Master of Engineering in Computer Engineering

Suranaree University of Technology

Academic Year 2012

การสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

คณะกรรมการสอบวิทยานิพนธ์

_____

(รศ. ดร.กิตติศักดิ์ เกิดประสพ)
ประธานกรรมการ

_____

(รศ. ดร.นิตยา เกิดประสพ)
กรรมการ (อาจารย์ที่ปรึกษาวิทยานิพนธ์)

_____

(อ. ดร.จิติมนต์ อั่งสกุล)
กรรมการ

_____

(ศ. ดร.ชูกิจ ลิมปิจำนงค์)
รองอธิการบดีฝ่ายวิชาการ

_____

(รศ. ร.อ. ดร.กนต์ธร ชำนิประศาสน์)
คณบดีสำนักวิชาวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีสุรนารี

ฝนทิพย์ คูณแก้ว : การสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้
(CLASSIFICATION MODEL INDUCTION BASED ON USER PREFERENCES)
อาจารย์ที่ปรึกษา : รองศาสตราจารย์ ดร.นิตยา เกิดประสพ, 167 หน้า.

การทำเหมืองข้อมูลมีบทบาทในการวิเคราะห์ข้อมูลและสารสนเทศขององค์กรและหน่วยงานทั้งภาคธุรกิจ การแพทย์ วิศวกรรมและอื่น ๆ จุดเด่นของการทำเหมืองข้อมูล คือ ช่วยสนับสนุนเทคนิคการวิเคราะห์ข้อมูลที่สามารถค้นพบความสัมพันธ์ภายในกลุ่มข้อมูล งานวิจัยนี้ได้เห็นจุดเด่นของการจำแนกหาลักษณะข้อมูลในแต่ละกลุ่ม เพื่องานทางด้านการแพทย์ที่มีประโยชน์ต่อแพทย์และผู้ป่วย ที่อาจจะสื่อสารกันได้ผ่านทางโมเดลที่จัดทำขึ้นมาจากการทำเหมืองข้อมูล โมเดลเพื่อการจำแนกนี้สามารถแสดงผลออกมาได้ในรูปแบบโครงสร้างต้นไม้ตัดสินใจ แต่เมื่อใดที่เราอยากรู้ข้อมูลเพียงบางส่วน ผลลัพธ์ของการทำเหมืองข้อมูลตามปกติจะไม่เอื้อต่อการตอบคำถามที่มีลักษณะเจาะจงเช่นนี้ ผู้วิจัยได้เห็นความสำคัญในจุดนี้ จึงพัฒนาวิธีการจำแนกข้อมูลตามข้อกำหนดที่ผู้ใช้ระบุ โดยงานวิจัยจะใช้วิธีการเขียนโปรแกรมเชิงตรรกะเพื่อพัฒนาฟังก์ชันการทำงานของการทำเหมืองข้อมูลแบบจำแนก เพื่อให้ได้ซึ่งแพทเทิร์นของข้อมูลที่นำมาจากแหล่งข้อมูลจริง และจะได้ผลในรูปแบบของกฎ ถ้า...แล้ว และเพิ่มการเขียนโปรแกรมเชิงตรรกะในรูปแบบที่สร้างเงื่อนไขให้กับขั้นตอนการแสดงผลลัพธ์ ซึ่งจะช่วยในการคัดเลือกเฉพาะบางกฎที่ผู้ใช้สนใจ เพื่อเพิ่มประสิทธิภาพในการค้นหาคำตอบที่ตรงกับความต้องการของผู้ใช้และได้ผลของแพทเทิร์นที่กระชับ ไม่มีข้อมูลที่เกินความจำเป็นที่ผู้ใช้ไม่ต้องการ อีกทั้งยังช่วยลดเวลาในการค้นหากฎทั้งหมด แทนที่การค้นหาจะต้องค้นหากฎทั้งหมดที่เป็นไปได้ เนื่องจากกฎที่ได้ทำการจำแนกข้อมูลแล้ว ได้ถูกบันทึกไว้ในหน่วยความจำ ซึ่งสามารถค้นหากฎในเงื่อนไขแบบใดก็ได้

สาขาวิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนักศึกษา_____

ปีการศึกษา 2555 ลายมือชื่ออาจารย์ที่ปรึกษา_____

FONTHIP KOONGAEW : CLASSIFICATION MODEL INDUCTION BASED ON USER PREFERENCES. THESIS ADVISOR : ASSOC. PROF. NITTAYA KERDPRASOP, Ph.D., 167 PP.

DATA MINING/CLASSIFICATION/ USER PREFERENCES

The data mining has been influential in analyzing information and data in the work of organizations and business, medical, engineering, etc. The strength of data mining is the help in the matter of technique in data analysis. This technique can be used to find relationships among data, along with other information. This thesis proposes highlight of the classification for the data in the field of medicine to be useful for clinicians and patients that may communicate through the model made from data mining. The complete model can be output in the form of decision trees. But when we want to know just some of the information, a whole decision tree is superfluous. This research has focused on this point of knowledge reduction. We use the method of logic programming to implement the functionality of data mining to extract patterns from data taken from real sources. The results of pattern extraction will be in the form of rules. We increase efficiency of knowledge navigation by allowing users to specify constraints or preferences, which will help in the selection of specific rules of interest. Our methodology can enhance the search for answers, as well as reduce the time to locate all the rules.

School of Computer Engineering Student's Signature_____

Academic Year 2012 Advisor's Signature_____

กิตติกรรมประกาศ

วิทยานิพนธ์นี้สำเร็จลุล่วงด้วยดี ผู้วิจัยขอกราบขอบพระคุณ บุคคล และกลุ่มบุคคลต่าง ๆ ที่ได้กรุณาให้คำปรึกษา แนะนำ ช่วยเหลืออย่างดียิ่ง ทั้งในด้านวิชาการ และด้านการดำเนินงานวิจัย ดังต่อไปนี้

รองศาสตราจารย์ ดร.นิตยา เกิดประสพ อาจารย์ที่ปรึกษาวิทยานิพนธ์ และรองศาสตราจารย์ ดร.กิตติศักดิ์ เกิดประสพ ที่ให้คำปรึกษาในการทำงานวิจัย การจัดการรูปแบบ และช่วยตรวจทานความถูกต้องของวิทยานิพนธ์

ผู้ช่วยศาสตราจารย์ ดร.พิชโยทัย มหัทธนาภิวัฒน์ ผู้ช่วยศาสตราจารย์ ดร.คะชา ชาญศิลป์ ผู้ช่วยศาสตราจารย์ สมพันธุ์ ชาญศิลป์ และผู้ช่วยศาสตราจารย์ ดร.ปรเมศวร์ ห่อแก้ว อาจารย์ประจำสาขาวิชาวิศวกรรมคอมพิวเตอร์ สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

คุณกัลญา พับโพธิ์ เลขานุการสาขาวิชาวิศวกรรมคอมพิวเตอร์ ที่ให้ความช่วยเหลือในการประสานงานด้านเอกสารระหว่างศึกษา

คุณวิทวัส วิทยาไกรเลิศ คุณไพชยนต์ คงไชย คุณสุรสิทธิ์ ท้าวกอกและนักศึกษาบัณฑิตศึกษา สาขาวิชาวิศวกรรมคอมพิวเตอร์ ทุกท่านที่ให้คำปรึกษาและช่วยเหลือด้วยดีมาโดยตลอด

นอกจากนี้ขอขอบคุณครู อาจารย์ทั้งในอดีตและปัจจุบันที่ให้ความรู้แก่ผู้วิจัยจนประสบความสำเร็จในชีวิต

ท้ายที่สุดที่จะลืมไม่ได้ ขอกราบขอบพระคุณ บิดา มารดา ที่ให้กำเนิด อบรม เลี้ยงดูด้วยความรัก และส่งเสริมการศึกษาเป็นอย่างดีโดยตลอด ทำให้ผู้วิจัยมีความรู้ ความสามารถ มีจิตใจที่เข้มแข็ง รวมทั้งเป็นกำลังใจที่ยิ่งใหญ่แก่ผู้วิจัย จนทำให้ผู้วิจัยประสบความสำเร็จในชีวิตเรื่อยมา

ฝนทิพย์ คูณแก้ว

สารบัญ

หน้า

บทคัดย่อ (ภาษาไทย) ..... ก

บทคัดย่อ (ภาษาอังกฤษ) ..... ข

กิตติกรรมประกาศ ..... ค

สารบัญ ..... ง

สารบัญตาราง ..... ช

สารบัญรูป ..... ญ

บทที่

1 บทนำ 1

1.1 ความสำคัญและที่มาของปัญหาการวิจัย 1

1.2 วัตถุประสงค์ของการวิจัย 4

1.3 ขอบเขตการวิจัย 4

1.4 ประโยชน์ที่คาดว่าจะได้รับ 5

2 ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง 6

2.1 การทำเหมืองข้อมูล (Data Mining) 6

2.2 ขั้นตอนของการค้นหาความรู้จากฐานข้อมูล 9

2.3 เทคนิคที่ใช้เพื่อการทำเหมืองข้อมูล (Data Mining Techniques) 12

2.4 การจำแนกประเภทข้อมูล (Classification) 15

2.5 อัลกอริทึม ID3 23

2.6 ภาษาเชิงตรรกะ 34

2.6.1 หลักการพื้นฐานของภาษาเชิงตรรกะ 35

2.6.2 แบบชนิดของข้อมูล 36

2.6.3 โครงสร้างของโปรแกรมและการดำเนินการ 37

2.7 หลักการของ User Preference 39

2.8 งานวิจัยที่เกี่ยวข้อง 40

สารบัญ (ต่อ)

หน้า

3 วิธีดำเนินการวิจัย 44
3.1 ขั้นตอนการวิจัย 44
3.2 โปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ 45
3.2.1 การจำแนกข้อมูลด้วยอัลกอริทึม ID3 และ ID3UP 45
3.2.2 การแสดงผลการจำแนกข้อมูลเมื่อใส่เควียรีด้วย
เพรดิเคตตรรกศาสตร์ 50
3.3 การทำงานของโปรแกรมสังเคราะห์โมเดลเพื่อการจำแนก
ตามข้อกำหนดของผู้ใช้ 59
3.3.1 การเตรียมข้อมูลสำหรับโปรแกรม ID3UP 59
3.3.2 การเรียกใช้โปรแกรม 61
3.3.3 การแสดงผลของกฎทั้งหมดด้วยเพรดิเคต findRule([]) 63
3.3.4 การแสดงผลของกฎตามเงื่อนไขของผู้ใช้ด้วยเพรดิเคต
findRule([X]) หรือ findRule([\+X]) หรือ findRule([X,\+X]) 63
3.3.5 การแสดงผลของกฎตามเงื่อนไขของผู้ใช้ด้วยเพรดิเคต
findRuleOr([[X],[X]]) หรือ findRuleOr([[\+X],[\+X]])
หรือ findRuleOr([[X,\+X],[\+X,X]]) 65
4 การทดสอบและอภิปรายผล 67
4.1 ข้อมูลที่ใช้ในการทดสอบ 67
4.2 ผลของการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP 72
4.2.1 ผลการทดสอบกับข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain) 72
4.2.2 ผลการทดสอบกับข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer) 74
4.2.3 ผลการทดสอบกับข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease) 76

สารบัญ (ต่อ)

หน้า

4.3 ผลของการค้นหากฎด้วยเพรดิเคตตรรกศาสตร์ 78
4.3.1 การค้นหากฎจำแนกข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain) 78
4.3.2 การค้นหากฎจำแนกข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer) 94
4.3.3 การค้นหากฎจำแนกข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease) 107
4.4 ผลของการเปรียบเทียบเวลาและจำนวนกฎสำหรับการเควียรีด้วยคำสั่งที่กำหนด 128
4.4.1 เปรียบเทียบจำนวนกฎ 128
4.4.2 เปรียบเทียบเวลา 130
4.4.3 เปรียบเทียบความถูกต้อง 134
4.5 การอภิปรายผล 141
5 สรุปผลการวิจัยและข้อเสนอแนะ 142
5.1 สรุปผลการวิจัย 143
5.2 การประยุกต์งานวิจัย 143
5.3 ปัญหาและข้อเสนอแนะ 144
รายการอ้างอิง 145
ภาคผนวก
ภาคผนวก ก. บทความวิชาการที่ได้รับการตีพิมพ์เผยแพร่ 149
ภาคผนวก ข. รหัสต้นฉบับโปรแกรม ID3UP for findRule() Predicate & findRuleOr() Predicate , ID3UP for findRule() Predicate & findRuleOr() Predicate (ปรับปรุง) 157
ประวัติผู้เขียน 167

มหาวิทยาลัยเทคโนโลยีสุรนารี

สารบัญตาราง

ตารางที่ **หน้า**

2.1 ข้อมูลการซื้อขายรถ 18
2.2 Train Data ของข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ 20
2.3 Test Data ของข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ 21
2.4 ตารางแสดง Detailed Accuracy By Class ของโมเดล Decision Tree 21
2.5 ตารางแสดง Detailed Accuracy By Class ของโมเดล Neural Network 21
2.6 ตารางแสดง Detailed Accuracy By Class ของโมเดล Logistic Regression 21
2.7 Train Data ของข้อมูลการดำรงตำแหน่งของอาจารย์มหาวิทยาลัยแห่งหนึ่ง 22
2.8 Test Data ของข้อมูลการดำรงตำแหน่งของอาจารย์มหาวิทยาลัยแห่งหนึ่ง 22
2.9 ข้อมูลผลการทำนายของข้อมูล Unseen Data 23
2.10 ข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ 24
2.11 สรุปเปรียบเทียบงานวิจัยที่เกี่ยวข้องกับการจำแนกข้อมูลในรูปแบบต่าง ๆ เพื่อสร้างกฎและเควียรีกฎที่ตรงกับเงื่อนไขที่กำหนดไว้ 42
3.1 ตารางแสดงข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ 59
4.1 รายละเอียดแอททริบิวต์ข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain) 68
4.2 รายละเอียดแอททริบิวต์ข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer) 70
4.3 รายละเอียดแอททริบิวต์ข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease) 71
4.4 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคตับอักเสบ 79
4.5 กฎที่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 อยู่ในกฎ 82
4.6 กฎที่ไม่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 อยู่ในกฎ 83
4.7 กฎที่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 และมีแอททริบิวต์ class มีค่าเป็น live 86
4.8 กฎที่ไม่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 และมีแอททริบิวต์ class มีค่าเป็น live 87

สารบัญตาราง (ต่อ)

ตารางที่ **หน้า**

4.9 กฎที่มีแอททริบิวต์ ascites มีค่าเป็น yes รวมกับ กฎที่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 และมีแอททริบิวต์ age มีค่าเป็น 31-40 88

4.10 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน 92

4.11 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยมะเร็งเต้านม 94

4.12 กฎการจำแนกที่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up อยู่ในกฎ 98

4.13 กฎที่ไม่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up อยู่ในกฎ 99

4.14 กฎที่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up และมีแอททริบิวต์ class มีค่าเป็น no_recurrence_events อยู่ในกฎ 102

4.15 กฎที่ไม่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up และไม่มีแอททริบิวต์ class มีค่าเป็น no_recurrence_events อยู่ในกฎ 103

4.16 กฎที่มีแอททริบิวต์ breast_quad มีค่าเป็น left_low และแอททริบิวต์ tumor_size มีค่าเป็น 30-34 รวมกับ ที่มีแอททริบิวต์ breast_quad มีค่าเป็น left_low และแอททริบิวต์ tumor_size มีค่าเป็น 30-34 และแอททริบิวต์ age มีค่าเป็น 50-59 105

4.17 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน 107

4.18 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคไทรอยด์ 108

4.19 กฎที่มีแอททริบิวต์ pregnant มีค่าเป็น false อยู่ในกฎ 113

4.20 กฎที่ไม่มีแอททริบิวต์ tumor มีค่าเป็น false อยู่ในกฎ 114

4.21 กฎที่มีแอททริบิวต์ referral_source มีค่าเป็น other , แอททริบิวต์ query_hypothyroid มีค่าเป็น true และแอททริบิวต์ sex มีค่าเป็น female อยู่ในกฎ 118

4.22 กฎที่ไม่มีแอททริบิวต์ referral_source มีค่าเป็น svhc , แอททริบิวต์ referral_source มีค่าเป็น svi และแอททริบิวต์ referral_source มีค่าเป็น other อยู่ในกฎ 118

4.23 กฎที่ไม่มีแอททริบิวต์ referral_source มีค่าเป็น svhc และ query_hypothyroid มีค่าเป็น false และมีแอททริบิวต์ class มีค่าเป็น sick รวมกับกฎที่ไม่มีแอททริบิวต์ query_hypothyroid มีค่าเป็น false 120

4.24 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน 125

สารบัญรูป

รูปที่ **หน้า**

1.1 โครงสร้างการจำแนกประเภทข้อมูลและคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้ 3
2.1 กระบวนการของการค้นหาความรู้จากฐานข้อมูล 9
2.2 โครงสร้างต้นไม้ตัดสินใจ 13
2.3 กระบวนการจำแนกประเภทข้อมูล (Classification) 16
2.4 โครงสร้างทั่วไปของต้นไม้ตัดสินใจ 17
2.5 ต้นไม้ตัดสินใจจำแนกประเภทลูกค้าที่ซื้อ/ไม่ซื้อรถใหม่ 18
2.6 โครงสร้างนิวรอลเน็ต 19
2.7 โครงสร้างต้นไม้ขั้นที่ 1 29
2.8 โครงสร้างต้นไม้ขั้นที่ 2 31
2.9 โครงสร้างต้นไม้ขั้นที่ 3 33
2.10 โครงสร้างต้นไม้ของข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ 34
2.11 โมเดลการเควียรีกฎ 40
3.1 ผังงานการจำแนกข้อมูลด้วยอัลกอริทึม ID3 และประมวลผล
ด้วยโปรแกรมโปรล็อก 46
3.2 รายละเอียดขั้นตอนอัลกอริทึม ID3 47
3.3 รายละเอียดขั้นตอนอัลกอริทึม ID3UP 48
3.4 ขั้นตอนการเควียรีกฎทั้งหมดด้วยเพรดิเคต findRule([]) 51
3.5 ผังงานการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP เมื่อมีการเควียรีกฎ
บางส่วนตามเงื่อนไขของผู้ใช้ ภายใต้การดำเนินการของ AND 53
3.6 รายละเอียดขั้นตอนอัลกอริทึม findRule([Argument]) predicate 54
3.7 ผังงานการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP เมื่อมีการเควียรีกฎ
บางส่วนตามเงื่อนไขของผู้ใช้ ภายใต้การดำเนินการของ OR 57
3.8 รายละเอียดขั้นตอนอัลกอริทึม findRuleOr([Argument]) predicate 58

สารบัญตาราง (ต่อ)

ตารางที่ **หน้า**

4.25 การเปรียบเทียบเวลาของการทดสอบการเควียรีกฎด้วยข้อมูลผู้ป่วยตับอักเสบ 131
4.26 การเปรียบเทียบเวลาของการทดสอบการเควียรีกฎด้วยข้อมูลผู้ป่วยมะเร็งเต้านม 132
4.27 การเปรียบเทียบเวลาของการทดสอบการเควียรีกฎด้วยข้อมูลผู้ป่วยโรคไทรอยด์ 133
4.28 10 กฎแรกของข้อมูลผู้ป่วยตับอักเสบที่ถูกจำแนกข้อมูลด้วยโปรแกรม
การสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ (Prolog) 136
4.29 10 กฎแรกของข้อมูลผู้ป่วยมะเร็งเต้านมที่ถูกจำแนกข้อมูลด้วยโปรแกรม
การสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ (Prolog) 138
4.30 10 กฎแรกของข้อมูลผู้ป่วยโรคไทรอยด์ที่ถูกจำแนกข้อมูลด้วยโปรแกรม
การสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ (Prolog) 140

มหาวิทยาลัยเทคโนโลยีสุรนารี

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

3.9 ข้อมูลที่อยู่ในรูปแบบข้อความตรรกศาสตร์อันดับหนึ่ง
พร้อมทำการจำแนกข้อมูล 60
3.10 รูปแบบของแอททริบิวต์ของข้อมูล 61
3.11 ผลที่ได้หลังจากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP 62
3.12 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎทั้งหมด
เมื่อเควียรีด้วยเพรดิเคต findRule([]) 63
3.13 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข
เมื่อเควียรีด้วยเพรดิเคต findRule([X]) 64
3.14 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข
เมื่อเควียรีด้วยเพรดิเคต findRule([\+X]) 64
3.15 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข
เมื่อเควียรีด้วยเพรดิเคต findRule([X,\+X]) 64
3.16 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข
เมื่อเควียรีด้วยเพรดิเคต findRuleOr([[X],[X]]) 65
3.17 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข
เมื่อเควียรีด้วยเพรดิเคต findRuleOr([[\+X],[\+X]]) 66
3.18 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข
เมื่อเควียรีด้วยเพรดิเคต findRuleOr([[X,\+X],[\+X,X]]) 66
4.1 ตัวอย่างข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain) 69
4.2 ตัวอย่างข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer) 70
4.3 ตัวอย่างข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease) 71
4.4 ผลลัพธ์ของข้อมูลผู้ป่วยโรคตับอักเสบเมื่อถูกจำแนกด้วยโปรแกรม ID3UP 72
4.5 ผลการจำแนกข้อมูลผู้ป่วยโรคตับอักเสบด้วยโปรแกรม ID3UP
แสดงการสร้างกฎลำดับที่หนึ่ง 73
4.6 ผลการจำแนกข้อมูลผู้ป่วยโรคตับอักเสบด้วยโปรแกรม ID3UP
แสดงการสร้างกฎลำดับที่สอง 73

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

4.7 ผลลัพธ์ของข้อมูลผู้ป่วยมะเร็งเต้านมเมื่อถูกจำแนกด้วยโปรแกรม ID3UP 74

4.8 ผลการจำแนกข้อมูลผู้ป่วยมะเร็งเต้านมโปรแกรม ID3UP
แสดงการสร้างกฎลำดับที่หนึ่ง 75

4.9 ผลการจำแนกข้อมูลผู้ป่วยมะเร็งเต้านมโปรแกรม ID3UP
แสดงการสร้างกฎลำดับที่สอง 75

4.10 ผลลัพธ์ของข้อมูลผู้ป่วยโรคไทรอยด์เมื่อถูกจำแนกด้วยโปรแกรม ID3UP 76

4.11 ผลการจำแนกข้อมูลผู้ป่วยโรคไทรอยด์ด้วยโปรแกรม ID3UP
แสดงการสร้างกฎลำดับที่หนึ่ง 77

4.12 ผลการจำแนกข้อมูลผู้ป่วยโรคไทรอยด์ด้วยโปรแกรม ID3UP
แสดงการสร้างกฎลำดับที่สอง 77

4.13 ผลการทดสอบกับข้อมูลผู้ป่วยโรคตับอักเสบ เมื่อเควียรีกฎ
ทั้งหมดด้วย findRule([ ]) 78

4.14 กราฟแสดงจำนวนของกฎของข้อมูลผู้ป่วยตับอักเสบ
ที่ได้จากการเควียรีด้วยคำสั่งต่าง ๆ 129

4.15 กราฟแสดงจำนวนของกฎของข้อมูลผู้ป่วยมะเร็งเต้านม
ที่ได้จากการเควียรีด้วยคำสั่งต่าง ๆ 129

4.16 กราฟแสดงจำนวนของกฎของข้อมูลผู้ป่วยโรคไทรอยด์
ที่ได้จากการเควียรีด้วยคำสั่งต่าง ๆ 130

4.17 10 กฎแรกของข้อมูลผู้ป่วยตับอักเสบ
ที่ถูกจำแนกข้อมูลด้วยโปรแกรม Weka 3.6 135

4.18 10 กฎแรกของข้อมูลผู้ป่วยมะเร็งเต้านม
ที่ถูกจำแนกข้อมูลด้วยโปรแกรม Weka 3.6 137

4.19 10 กฎแรกของข้อมูลผู้ป่วยโรคไทรอยด์
ที่ถูกจำแนกข้อมูลด้วยโปรแกรม Weka 3.6 139

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มาของปัญหาการวิจัย

ปัจจุบันระบบสนับสนุนข้อมูลในการตัดสินใจได้เข้ามามีอิทธิพลในการ วางแผน และดำเนินงานขององค์กรต่าง ๆ ทั้งด้านธุรกิจ ด้านการแพทย์ ด้านวิศวกรรม และอื่น ๆ ซึ่งระบบสนับสนุนข้อมูลในการตัดสินใจนี้ มักจะทำในรูปแบบของการทำเหมืองข้อมูล (Data Mining) กับฐานข้อมูลขนาดใหญ่ (Adriaans and Zantinge, 1999) ผลลัพธ์ที่ได้จากการทำเหมืองข้อมูลสามารถบอกถึงข้อเท็จจริงในลักษณะของกฎความสัมพันธ์ (Association Rules) โมเดลเพื่อการจำแนกประเภทข้อมูล (Data Classification Model) เกณฑ์ในการแบ่งกลุ่มข้อมูล (Data Clustering Criteria) และจินตทัศน์เกี่ยวกับข้อมูล (Visualization) (วิกิพีเดีย สารานุกรมเสรี, 2554a) อีกทั้งยังมีเกณฑ์มาตรฐานการวัดความถูกต้อง ความแม่นยำ และประสิทธิภาพของการใช้งานที่มีผลต่อการนำข้อมูลไปใช้ประโยชน์ในด้านต่าง ๆ

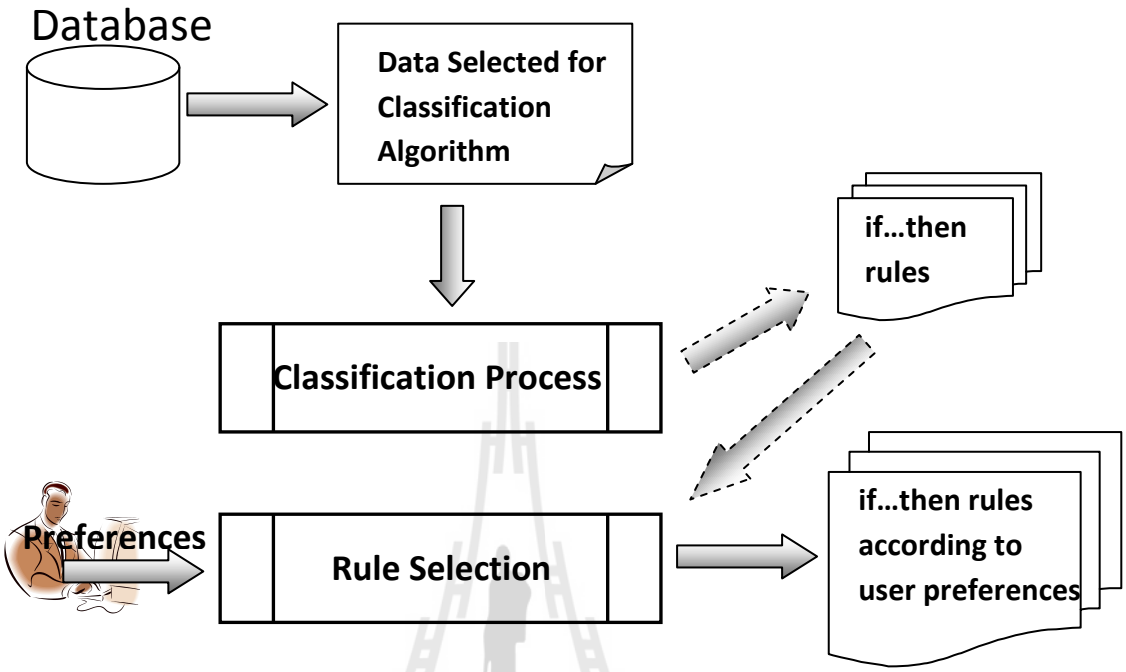
จุดเด่นของการทำเหมืองข้อมูล ที่จะมาช่วยในเรื่องของเทคนิคการจัดการข้อมูล และการทำนายผลความเป็นไปได้ของข้อมูล คือ สามารถนำข้อมูลในอดีตมาสนับสนุนในการจัดทำเหมืองข้อมูลได้ ด้วยเทคนิคนี้สามารถใช้ค้นข้อมูลสำคัญที่ปะปนกับข้อมูลอื่น ๆ ในฐานข้อมูลที่ไม่ใช่แค่การสุ่ม เรียกกระบวนการนี้ว่า KDD (Knowledge Discovery in Databases) หรือ การค้นหาความรู้จากฐานข้อมูล (Adriaans and Zantinge, 1999)

การใช้ประโยชน์เทคโนโลยีการทำเหมืองข้อมูล ได้มีการใช้อย่างแพร่หลายในงานหลากหลายลักษณะ โดยเฉพาะการจำแนกหาลักษณะข้อมูลในแต่ละกลุ่ม (Classification) ที่จะได้ผลลัพธ์เป็นแบบแผนและแพทเทิร์นที่ปรากฏในข้อมูล เรียกว่า โมเดลเพื่อการจำแนก (Classification Model) ที่สามารถใช้เป็นเครื่องมือในการคาดหมายประเภทหรือลักษณะของข้อมูลในอนาคตหรือใช้ประกอบการตัดสินใจ (นิตยา เกิดประสพ, 2552) ซึ่งโครงสร้างของ Classification Model สามารถรายงานผลออกมาในรูปแบบโครงสร้างต้นไม้ตัดสินใจ (Decision Tree) ซึ่งจะเอื้อ

ประโยชน์ให้ผู้ที่ไม่มีความรู้ด้านการทำเหมืองข้อมูล สามารถอ่านผลออกมาได้ว่าข้อมูลนี้มีทิศทางไปในทางใด

งานวิจัยนี้ได้เห็นจุดเด่นของการจำแนกหาลักษณะข้อมูลในแต่ละกลุ่มที่นิยมใช้กันมากในงานหลายประเภท เช่น งานทางด้านการแพทย์ ซึ่งการจำแนกข้อมูลจะมีประโยชน์ต่อแพทย์ บุคลากรด้านสาธารณสุข และผู้ป่วย ที่อาจจะสื่อสารกันได้ผ่านทางโมเดลที่จัดทำขึ้นมาจากการทำเหมืองข้อมูล เช่น ถ้ามีสมมติฐานที่ว่า “โรคติดเชื้อชนิดใดในโรงพยาบาลมีแนวโน้มลุกลามเป็นโรคระบาด” (ปัณณวิชญ์ วงศ์วิวัฒนานนท์, 2551) ซึ่งข้อมูลที่จะช่วยตัดสินใจในสมมติฐานนี้ อาจจะได้มาจากประวัติของผู้ป่วย สภาพแวดล้อม และปัจจัยอื่น ๆ หรืออาจจะเป็นสมมติฐานที่ว่า “ผู้ป่วยที่มีอาการอย่างไร ถึงจะมีความเสี่ยงต่อการเกิดเป็นโรคเบาหวาน” เมื่อมีการทำเหมืองข้อมูลและได้ผลลัพธ์ออกมาเป็นโมเดลที่เกี่ยวข้องกับสมมติฐานเหล่านี้ จะสามารถแสดงผลออกมาได้ในรูปแบบของโครงสร้างต้นไม้ตัดสินใจ แต่เมื่อใดที่เราอยากรู้ผลลัพธ์เพียงบางส่วน เช่น อยากรู้เฉพาะผลลัพธ์ที่เกี่ยวข้องกับน้ำตาลในเลือดและผู้ป่วยที่เสี่ยงต่อการเป็นโรคเบาหวาน ผลลัพธ์ของการทำเหมืองข้อมูลตามปกติแล้ว จะไม่เอื้อต่อการตอบคำถามที่มีลักษณะเจาะจงเช่นนี้

ผู้วิจัยได้เห็นความสำคัญในจุดนี้ จึงพัฒนาวิธีการจำแนกข้อมูลตามข้อกำหนดที่ผู้ใช้ระบุ โดยงานวิจัยจะใช้วิธีการเขียนโปรแกรมเชิงตรรกะ (Logic Programming) พัฒนาฟังก์ชันการทำงานของการทำเหมืองข้อมูล เพื่อให้ได้แพทเทิร์นของข้อมูลที่นำมาจากแหล่งข้อมูลจริง และจะได้ผลในรูปแบบของกฎ if…then และเพิ่มการเขียนโปรแกรมเชิงตรรกะในรูปแบบที่สร้างเงื่อนไขให้กับขั้นตอนการแสดงผลลัพธ์ ซึ่งจะช่วยในการคัดเลือกเฉพาะบางกฎที่ผู้ใช้สนใจ เพื่อเพิ่มประสิทธิภาพในการค้นหาคำตอบ สามารถรายงานผลลัพธ์ที่ตรงกับความต้องการของผู้ใช้ และได้ผลลัพธ์ที่กระชับ ไม่มีผลลัพธ์ที่เกินความจำเป็นที่ผู้ใช้ไม่ต้องการ อีกทั้งผู้ใช้ยังสามารถเรียกดูเฉพาะบางกฎด้วยเงื่อนไขที่สามารถสนับสนุนการแสดงผลได้หลากหลายและมีประสิทธิภาพ และเวลาในการค้นหากฎตามเงื่อนไขจะใช้เวลาน้อย เนื่องจากกฎที่ได้จากโมเดลการจำแนกข้อมูลนั้นได้ถูกบันทึกไว้ในหน่วยความจำ ทำให้สามารถค้นหากฎด้วยเงื่อนไขใดก็ได้ โครงสร้างของแนวคิดนี้แสดงได้ดังรูปที่ 1.1



รูปที่ 1.1 โครงสร้างการจำแนกประเภทข้อมูลและคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้

จากรูปที่ 1.1 เมื่อข้อมูลที่อยู่ในฐานข้อมูลถูกนำมาจัดรูปแบบเพื่อจำแนกข้อมูลด้วยกระบวนการ Classification ผลที่ได้จากการจำแนก คือ กฎ โดยผู้ใช้จะติดต่อกับกระบวนการ Rule Selection เพื่อคัดเลือกกฎที่ต้องการแสดงผล ทั้งการแสดงผลทั้งหมดหรือแสดงผลตามเงื่อนไขของผู้ใช้เอง

1.2 วัตถุประสงค์ของการวิจัย

จากแนวคิดในงานวิจัยการจำแนกประเภทข้อมูลด้วยวิธีการเขียนโปรแกรมเชิงตรรกะและคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้ ผู้วิจัยได้ตั้งวัตถุประสงค์ในการวิจัยไว้ ดังนี้

1.2.1 เพื่อศึกษาและพัฒนาการเขียนโปรแกรมเชิงตรรกะ สำหรับงานการทำเหมืองข้อมูลประเภทการจำแนก ซึ่งจะมีประสิทธิภาพและรวดเร็วต่อการพัฒนาโปรแกรมต้นแบบ

1.2.2 เพื่อจำแนกหาลักษณะข้อมูลในแต่ละกลุ่มของข้อมูลที่ยกตัวอย่างมาจากข้อมูลด้านการแพทย์ ซึ่งมีข้อมูลที่สำคัญในการจำแนก โดยจะรายงานผลต่อผู้ใช้ในรูปแบบของกฎ if…then

1.2.3 เพื่อเพิ่มประสิทธิภาพและความยืดหยุ่นในการรายงานผลต่อผู้ใช้ด้วยวิธีการเขียนโปรแกรมเชิงตรรกะและคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้ (User Preferences)

1.2.4 เพื่อหาข้อเปรียบเทียบระหว่างจำนวนของกฎที่แสดงผลออกมาด้วยวิธีการค้นหาทั้งหมด และวิธีการใช้เงื่อนไขคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้

1.2.5 เพื่อหาข้อเปรียบเทียบจำนวนของกฎที่แสดงผลออกมา พร้อมทั้งข้อดีข้อเสีย ด้วยวิธีของการใช้เงื่อนไขคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้ เมื่อมีการคัดเลือกกฎภายใต้การดำเนินการด้วยโอเปอเรเตอร์ AND และ OR

1.3 ขอบเขตการวิจัย

งานวิจัยนี้เป็นการศึกษาการจำแนกประเภทข้อมูลด้วยวิธีการเขียนโปรแกรมเชิงตรรกะและคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้ โดยมีขอบเขตของการวิจัยในระบบที่พัฒนาขึ้นดังต่อไปนี้

1.3.1 การจำแนกข้อมูลในแต่ละกลุ่มด้วยวิธีการเขียนโปรแกรมเชิงตรรกะและคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้ สามารถใช้ได้กับฐานความรู้ทุกรูปแบบ ที่อยู่ในรูปแบบพร้อมทำเหมืองข้อมูล ด้วยฟังก์ชันการทำงานประเภทการจำแนกข้อมูลและข้อมูลเป็นลักษณะข้อความในตรรกศาสตร์อันดับหนึ่ง (First-order Logic)

1.3.2 ระบบที่พัฒนาขึ้นนี้สามารถใช้ได้ดีกับข้อมูลทางด้านการแพทย์หรือข้อมูลอื่น ๆ ที่มักจะใช้ข้อมูลแค่บางส่วนมาใช้ประโยชน์ ซึ่งสามารถนำมาใช้ได้อย่างแท้จริง และมีประสิทธิภาพ

1.3.3 การจำแนกให้ได้แพทเทิร์นที่เป็นไปได้ ด้วยโปรแกรมเชิงตรรกะใช้ฟังก์ชันการทำเหมืองข้อมูลประเภทการจำแนกด้วยโครงสร้างต้นไม้ตัดสินใจ

1.3.4 สามารถระบุผลการรายงานของการจำแนกได้ว่าต้องการรายงานผลส่วนใดของข้อมูลและคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้ที่เขียนอยู่ในลักษณะของข้อความในรูปแบบตรรกศาสตร์อันดับหนึ่ง

1.3.5 จำแนกข้อมูลที่ได้ออกมาเป็นแพทเทิร์นในรูปแบบของกฎ if…then

1.4 ประโยชน์ที่คาดว่าจะได้รับ

งานวิจัยนี้เป็นการศึกษาการจำแนกประเภทข้อมูลด้วยวิธีการเขียนโปรแกรมเชิงตรรกะและเพิ่มขีดความสามารถด้านการคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้ โดยคาดว่าจะได้รับประโยชน์ดังต่อไปนี้

1.4.1 ระบบที่พัฒนาขึ้นเป็นการเขียนโปรแกรมเชิงตรรกะสามารถประมวลผลได้อย่างรวดเร็ว

1.4.2 ระบบที่พัฒนาขึ้นในส่วนของการรายงานผล ใช้วิธีการเขียนโปรแกรมเชิงตรรกะและคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้ สามารถตอบคำถามที่ตรงประเด็น และมีความยืดหยุ่นสำหรับผู้ใช้ในการกำหนดเงื่อนไขคัดเลือกกฎ ซึ่งผลที่ปรากฏนั้นจะสามารถให้คำตอบเดียวหรือหลายคำตอบได้

มหาวิทยาลัยเทคโนโลยีสุรนารี

บทที่ 2
ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

2.1 การทำเหมืองข้อมูล (Data Mining)

การทำเหมืองข้อมูลเป็นการค้นหาความสัมพันธ์และรูปแบบที่มีอยู่จริงในฐานข้อมูลและแฝงอยู่ภายในกลุ่มข้อมูล โดยความสัมพันธ์เหล่านี้แสดงให้เห็นถึงความรู้ต่าง ๆ ที่อาจนำมาใช้ประโยชน์ได้ เครื่องมือที่ใช้การทำเหมืองข้อมูลสามารถใช้วิธีการทางสถิติ ขั้นตอนวิธีการทางคณิตศาสตร์ และการเรียนรู้ของเครื่องจักรที่ใช้การทำงานแบบอัตโนมัติและกึ่งอัตโนมัติ (Adriaans and Zantinge, 1999)

พัฒนาการของการวิเคราะห์ข้อมูล เริ่มต้นจากช่วงปี ค.ศ.1960 เริ่มมีการทำ ***Data Collection*** คือ การรวบรวมเพื่อนำข้อมูลมาจัดเก็บอย่างเหมาะสมในอุปกรณ์ที่น่าเชื่อถือและป้องกันการสูญหายได้เป็นอย่างดี การเก็บรวบรวมข้อมูลเป็นขั้นตอนหนึ่งของกระบวนการวิจัยเพื่อให้ได้มาซึ่งข้อมูลที่ตอบสนองตามวัตถุประสงค์ที่วางไว้ การเก็บข้อมูลนั้น จะมีทั้งการเก็บข้อมูลขึ้นมาใหม่ และการรวมข้อมูล (Data Compilation) ซึ่งหมายถึง การนำเอาข้อมูลต่าง ๆ ที่ผู้อื่นได้เก็บไว้แล้ว หรือรายงานไว้ในเอกสารต่าง ๆ มาทำการศึกษาวิเคราะห์ต่อ (Vassiliadis, Quix, Vassiliou and Jarke, 2001)

วิธีการเก็บรวบรวมข้อมูล ประกอบด้วย

- การสำรวจ (Survey)
- การสังเกตการณ์ (Observation)
- การสัมภาษณ์ (Interview)
- การรวบรวมข้อมูลจากเอกสาร (Data Collection)

ช่วงปี ค.ศ.1980 ได้มีการพัฒนาเทคนิค ***Data Analysis*** คือ การนำข้อมูลที่จัดเก็บมาวิเคราะห์หาความสัมพันธ์ต่อกันในข้อมูล เพื่อประโยชน์ในการนำไปวิเคราะห์ขั้นสูงและการตัดสินใจ

ช่วงปี ค.ศ.1990 เกิด ***Data Warehouse & Decision Support System*** คือ การรวบรวมข้อมูลมาจัดเก็บลงในคลังข้อมูล โดยครอบคลุมข้อมูลจากหน่วยงานทั้งหมดขององค์กร เพื่อช่วยสนับสนุนการตัดสินใจ

คลังข้อมูล (Data Warehouse) เป็นฐานข้อมูลขนาดใหญ่ขององค์กรหรือหน่วยงาน โดยจะนิยมใช้ในรูปแบบการเก็บข้อมูลให้เหมาะสมกับการเรียกใช้ เพื่อสะดวกต่อองค์กรที่ใช้การจัดเก็บข้อมูลในรูปแบบของคลังข้อมูล อีกทั้งเหมาะสำหรับงานวิเคราะห์ข้อมูลและใช้ข้อมูลในการตัดสินใจทำงานต่าง ๆ และยังสามารถรวบรวมข้อมูลในอดีตและปัจจุบันเข้ากันเป็นฐานข้อมูลเดียวได้ (ทวีศักดิ์ นาคม่วง, 2547)

ระบบสนับสนุนการตัดสินใจ (Decision Support System) เป็นระบบย่อยหนึ่งในระบบสารสนเทศเพื่อการจัดการ โดยที่จะช่วยในเรื่องการตัดสินใจในเหตุการณ์ กิจกรรมทางธุรกิจ หรือการตัดสินใจเชิงกลยุทธ์ (ทวีศักดิ์ นาคม่วง, 2547)

ช่วงปี ค.ศ.2000 มีการทำ ***Data Mining*** คือ การนำข้อมูลจากฐานข้อมูลมาวิเคราะห์และประมวลผล โดยการสร้างแบบจำลองหรือโมเดล และความสัมพันธ์ทางสถิติหรือคณิตศาสตร์

การทำเหมืองข้อมูลเป็นการสังเคราะห์รูปแบบข้อมูลอย่างละเอียดจากฐานข้อมูลที่มีขนาดใหญ่ โดยใช้ขั้นตอนวิธีการเรียนรู้จากข้อมูลในอดีต ซึ่งผลลัพธ์ที่ได้จากการสังเคราะห์ จะเป็นรูปแบบข้อมูลที่ไม่เคยทราบมาก่อน (Unknown) มีความถูกต้อง (Valid) และเป็นความรู้ที่นำไปปฏิบัติได้ (Actionable) (Han, Kamber, and Pei, 2006 ; Zaïane, 1999) ความหมายของลักษณะสำคัญของรูปแบบข้อมูล อธิบายได้ดังนี้

รูปแบบข้อมูลที่ไม่เคยทราบมาก่อน (Unknown) เป็นความรู้ที่ผู้ใช้งานไม่เคยรู้มาก่อน โดยจะไม่สามารถตั้งสมมติฐานล่วงหน้าได้ว่ารูปแบบหรือกฎที่ได้จะเป็นรูปแบบใด

รูปแบบที่มีความถูกต้อง (Valid) เมื่อผู้ใช้เริ่มใช้เทคนิคของการทำเหมืองข้อมูล จะค้นพบสิ่งที่น่าสนใจตลอดเวลา แต่จะต้องพิจารณาด้วยว่าสิ่งนั้นถูกต้องหรือไม่ เช่น การทำเหมืองข้อมูลประเภทที่ให้ผลลัพธ์เป็นกฎ ความหมายของกฎจะต้องไม่ผิดจากข้อเท็จจริงที่ปรากฏในฐานข้อมูล

รูปแบบที่นำไปปฏิบัติได้ (Actionable) ข้อมูลจะต้องถูกแปลงออกมาและสามารถนำมาใช้ช่วยประกอบการตัดสินใจ เพื่อสร้างความได้เปรียบในเชิงธุรกิจ

วัตถุประสงค์ของการทำเหมืองข้อมูล

- เพื่อค้นหาความรู้ใหม่ในฐานข้อมูลที่มาจากแหล่งข้อมูลที่น่าเชื่อถือ (Knowledge Discovery in Databases)
- เพื่อช่วยในการวิเคราะห์หาความรู้ที่ซ่อนอยู่ภายในฐานข้อมูล (Knowledge Extraction)
- เพื่อจัดการกับข้อมูลในอดีตที่มีความสัมพันธ์กับข้อมูลปัจจุบัน ซึ่งจะเอื้อต่อการสร้างรูปแบบหรือกฎที่มีประโยชน์ต่อการทำเหมืองข้อมูล (Data Archeology)
- เพื่อสำรวจข้อมูลเพื่อใช้ประโยชน์ในการกำหนดเป้าหมายการทำเหมืองข้อมูล (Data Exploration)
- เพื่อค้นหารูปแบบ (Patterns) ของข้อมูลที่ซ่อนอยู่ในฐานข้อมูล ซึ่งจะเป็นประโยชน์ต่อการทำนายผลของฐานข้อมูลนั้น ๆ ว่ามีทิศทางไปทางไหน (Data Pattern Processing)
- เพื่อใช้ขุดเจาะข้อมูลในเชิงลึก อาจจะแสดงผลออกมาเป็นค่าทางสถิติ (Data Dredging)

ลักษณะงานประเภทต่าง ๆ ของการทำเหมืองข้อมูลนั้น จะใช้สำหรับค้นหารูปแบบ หรือกฎของข้อมูลที่ซ่อนอยู่ภายในฐานข้อมูลขนาดใหญ่ซึ่งมาจากข้อมูลจริง โดยมีวิธีการในการทำเหมืองข้อมูลในรูปแบบต่าง ๆ เช่น การสร้างกฎความสัมพันธ์ (Association Rules), การจำแนกประเภทข้อมูล (Data Classification), การแบ่งกลุ่มข้อมูล (Data Clustering), จินตทัศน์ (Visualization) (วิกิพีเดีย สารานุกรมเสรี, 2554a) ซึ่งสามารถแสดงผลแบบกราฟ, แผนภาพต้นไม้ และอื่น ๆ อีกทั้งผู้ใช้สามารถดูข้อมูลแบบเจาะลึกที่แสดงประกอบผลลัพธ์ได้ เช่น ตัวเลขของความถูกต้องและความแม่นยำในการทำนายผล อีกทั้งสามารถใช้เครื่องมือประกอบอื่นในการสอบถามข้อมูลได้

การทำเหมืองข้อมูลสามารถวิเคราะห์ข้อมูลได้หลากหลายลักษณะ โดยสามารถจำแนกตามฟังก์ชันการทำงานได้ดังต่อไปนี้

(1) การวิเคราะห์คุณสมบัติและการแยกแยะข้อมูล (Characterization and Discrimination)

(2) การหาความสัมพันธ์ของข้อมูล (Association)

(3) การจัดหมวดหมู่หรือการวิเคราะห์การถดถอย (Classification/ Regression)

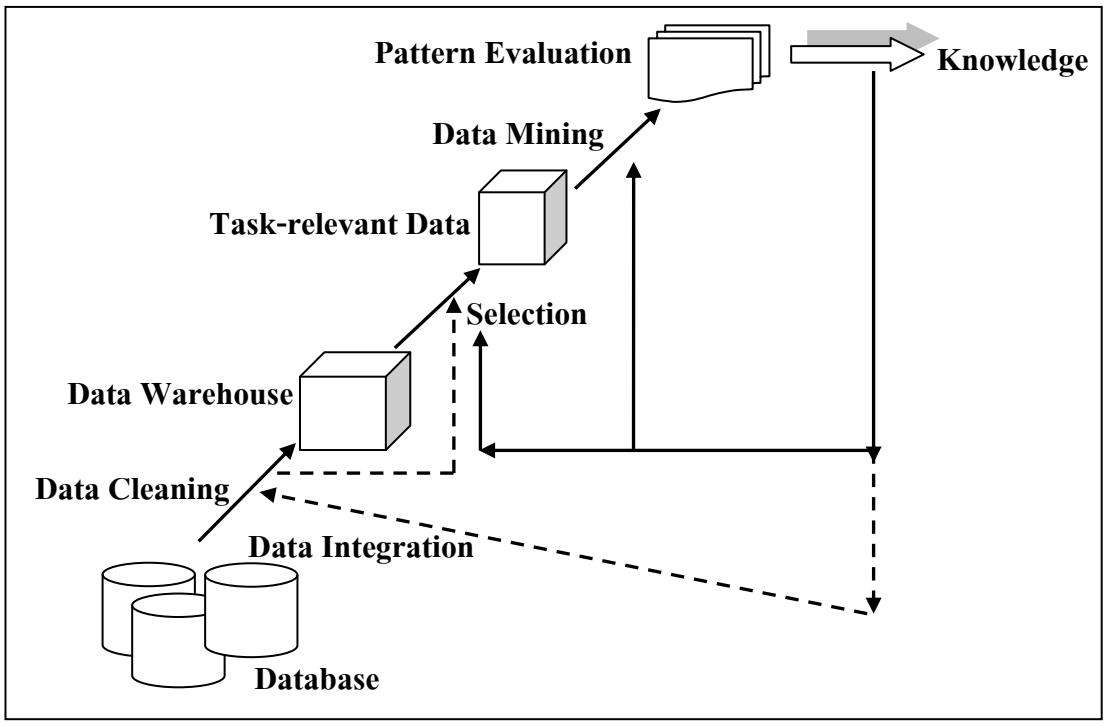
(4) การวิเคราะห์การรวมกลุ่ม หรือ การแบ่งแยกข้อมูล (Cluster Analysis/ Segmentation)

(5) การประเมินและการพยากรณ์ (Estimation and Prediction)

(6) การบรรยายและการแสดงภาพของข้อมูล (Description and Visualization)

2.2 ขั้นตอนของการค้นหาความรู้จากฐานข้อมูล

การค้นหาความรู้จากฐานข้อมูล หรือ KDD คือ กระบวนการค้นหาลักษณะแฝงของข้อมูลที่อยู่ในกลุ่มข้อมูลจำนวนมาก (Bodhisuwan, 2010) เป็นกระบวนการที่สำคัญในการค้นหาลักษณะที่น่าสนใจของข้อมูล เช่น รูปแบบ, กฎ, ความสัมพันธ์, โครงสร้าง หรือ ลักษณะที่ผิดปกติของข้อมูล จากข้อมูลจำนวนมากที่เก็บอยู่ในฐานข้อมูล



รูปที่ 2.1 กระบวนการของการค้นหาความรู้จากฐานข้อมูล (Osmar, 2011)

ขั้นตอนต่าง ๆ ของกระบวนการ KDD ที่แสดงดังรูปที่ 2.1 ประกอบด้วย

(1) ขั้นตอนการรวบรวมข้อมูล (Data Integration) ข้อมูลที่สามารถนำมาใช้ในการทำเหมืองข้อมูล ประกอบด้วย

- ข้อมูลจากฐานข้อมูลเชิงสัมพันธ์ (Relational Databases)
- ข้อมูลจากคลังข้อมูล (Data Warehouses)
- ข้อมูลจากฐานข้อมูลรายการปรับปรุง (Transactional Databases)
- ข้อมูลจากฐานข้อมูลพิเศษหรือที่เก็บข่าวสารพิเศษ ได้แก่
  - ฐานข้อมูลเชิงวัตถุ (Object-oriented Databases)
  - ฐานข้อมูลเกี่ยวกับเวลา (Temporal Databases)
  - ฐานข้อมูลข้อความ (Text Databases) และ ฐานข้อมูลมัลติมีเดีย (Databases)
  - ฐานข้อมูลแบบเก่าในอดีต (Archival Databases) หรือข้อมูลที่มาจากต่างฐานข้อมูลกัน (Multi- Databases)
  - ฐานข้อมูลจากแหล่งเว็บไซต์ (Web Databases)

(2) ขั้นตอนการคัดเลือกข้อมูล (Data Selection) คือ การคัดเลือกเฉพาะบางข้อมูลที่จะนำมาใช้ทำเหมืองข้อมูล เนื่องจากข้อมูลทั้งหมดในฐานข้อมูลเป็นการรวบรวมข้อมูลทั้งหมดของหน่วยงาน ทำให้ปรากฏข้อมูลหลายส่วนที่ไม่จำเป็นต่อการทำเหมืองข้อมูล เช่น รหัสพนักงาน ชื่อบิดา-มารดาของพนักงาน ขั้นตอนการคัดเลือกข้อมูลประกอบด้วยสองขั้นตอนย่อย คือ

(2.1) การกรองข้อมูลและประมวลผลก่อน (Data Cleaning and Preprocessing) คือ การคัดเลือกข้อมูลที่เก็บรวมรวมมาซึ่งมีจำนวนมาก โดยจะเลือกข้อมูลที่ตรงประเด็น และมีประโยชน์ในการทำเหมืองข้อมูล เพราะบางข้อมูลอาจจะไม่เป็นประโยชน์ เช่น ข้อมูลการซื้อขายสินค้า อาจจะไม่ใช้รหัสของลูกค้า เพราะลูกค้าแต่ละคน จะมีรหัสที่แตกต่างกัน ซึ่งจะไม่มีผลต่อการทำเหมืองข้อมูล ในขั้นตอนนี้เป็นขั้นตอนที่จะได้มาซึ่งคุณภาพของข้อมูลที่จะนำไปวิเคราะห์

(2.2) การแปลงรูปแบบข้อมูล (Data Reduction And Transformation) คือ การลดรูปและจัดข้อมูลให้อยู่ในรูปแบบเดียวกัน ซึ่งจะมีรูปแบบ (Format) ที่เป็นมาตรฐาน และเหมาะสมที่จะนำไปใช้กับฟังก์ชัน (Functions) ต่าง ๆ ในการทำเหมืองข้อมูล

(3) ขั้นตอนการทำเหมืองข้อมูล (Data Mining) เป็นการเลือกฟังก์ชันและอัลกอริทึมในการทำเหมืองข้อมูล การเลือกฟังก์ชันของการทำเหมืองข้อมูล เช่น Summarization, Classification, Regression, Association หรือ Clustering ซึ่งแต่ละฐานข้อมูลอาจจะใช้ได้หลายฟังก์ชัน แล้วแต่ความสะดวกและวัตถุประสงค์ของผู้ใช้งาน การทำเหมืองข้อมูลสามารถวิเคราะห์ข้อมูลได้หลากหลายลักษณะ โดยสามารถจำแนกตามฟังก์ชันการทำงานได้ดังต่อไปนี้

- การวิเคราะห์คุณสมบัติและการแยกแยะข้อมูล (Characterization and Discrimination)
- การหาความสัมพันธ์ของข้อมูล (Association)
- การจัดหมวดหมู่หรือการวิเคราะห์การถดถอย (Classification/ Regression)
- การวิเคราะห์การรวมกลุ่ม หรือ การแบ่งแยกข้อมูล (Cluster Analysis/ Segmentation)
- การประเมินและการพยากรณ์ (Estimation and Prediction)
- การบรรยายและการแสดงภาพของข้อมูล (Description and Visualization)

การเลือกอัลกอริทึมของการทำเหมืองข้อมูล เป็นการเลือกเทคนิคสำหรับการทำเหมืองข้อมูล เช่น เทคนิคการสร้างต้นไม้ตัดสินใจ เทคนิคโครงข่ายประสาทเทียม เพื่อทำการค้นหารูปแบบหรือกฎที่สนใจ

(4) ขั้นตอนการประเมินผลรูปแบบ (Pattern Evaluation) ในขั้นตอนนี้จะเป็นการวิเคราะห์ผลลัพธ์ที่ได้และแปลความหมาย ประเมินผลว่าผลลัพธ์นั้นเหมาะสมหรือตรงวัตถุประสงค์หรือไม่

(5) ขั้นตอนการประยุกต์ใช้องค์ความรู้ที่ค้นพบ

ชนิดขององค์ความรู้ที่ค้นพบ

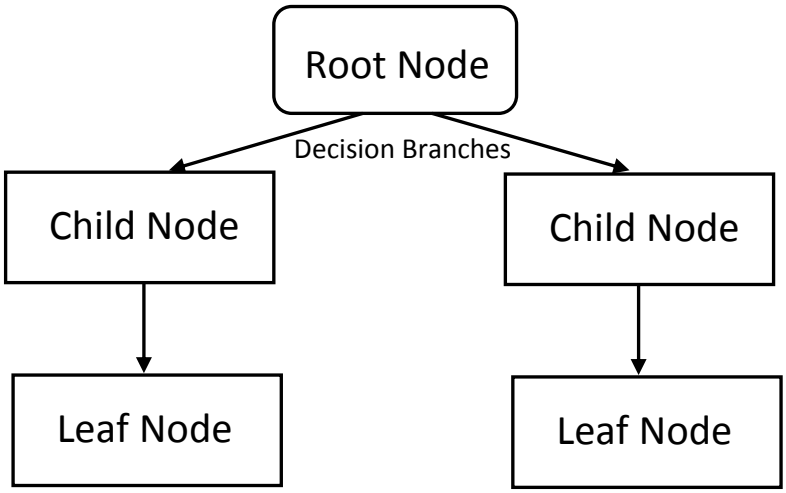
- องค์ความรู้เกี่ยวกับคุณลักษณะของข้อมูล (Characterization) เช่น สามารถทราบถึงคุณสมบัติของคนที่จบการศึกษาเกียรตินิยมอันดับ 1
- องค์ความรู้เกี่ยวกับการจำแนกข้อมูล (Discrimination) เช่น การจำแนกลักษณะของนักศึกษาที่เกรดเฉลี่ยสูงกว่า 3.00 เปรียบเทียบกับนักศึกษาที่เกรดเฉลี่ยต่ำกว่า 2.00
- องค์ความรู้เกี่ยวกับความสัมพันธ์ของข้อมูล (Association) เช่น ความสัมพันธ์ของการซื้อสินค้าพบว่า ถ้าลูกค้าซื้อ เบียร์ จะต้องซื้อ น้ำแข็ง ด้วย
- องค์ความรู้เกี่ยวกับการแยกประเภทข้อมูลและการพยากรณ์ (Classification and Prediction) เช่น การทำนายว่าผู้ป่วยที่เป็นโรคเบาหวาน จะมีระดับน้ำตาลมากกว่าคนปกติมากเท่าไหร่ ซึ่งอ่านได้จากรูปแบบหรือกฎที่ได้มาจากการทำเหมืองข้อมูลของฐานข้อมูลประวัติผู้ที่ป่วยเป็นโรคเบาหวาน

- องค์ความรู้เกี่ยวกับการจัดกลุ่มข้อมูล (Clustering) เช่น การจัดกลุ่มนักเรียนที่มีสถานภาพการเรียนปกติและสถานภาพตกออก ซึ่งแต่ละกลุ่มจะมีลักษณะที่เป็นเอกลักษณ์ และมีลักษณะคล้ายกันภายในกลุ่ม
- องค์ความรู้เกี่ยวกับการวิเคราะห์ข้อมูลที่แตกต่างจากข้อมูลส่วนใหญ่ (Outlier Analysis)
- องค์ความรู้เกี่ยวกับข้อมูลอื่น ๆ ในงานที่ค้นพบ (Other Data Mining Tasks)

2.3 เทคนิคที่ใช้เพื่อการทำเหมืองข้อมูล (Data Mining Techniques)

(1) Neural Network เป็นแนวคิดให้คอมพิวเตอร์ทำงานเลียนแบบการทำงานในสมองของมนุษย์และเปลี่ยนตัวเองจากการประมวลผลตามลำดับ (Sequential Processing) ให้เป็นการประมวลผลแบบคู่ขนานได้ (Parallel Processing) มีลักษณะการทำงานโดย Process จะรับ Input เข้าไปคำนวณ และสร้าง Output ออกมาในลักษณะที่ไม่ใช่การทำงานแบบเชิงเส้นตรง (Yuan and Shaw, 1995) เพราะ Input แต่ละตัวจะถูกให้ลำดับความสำคัญของค่าไม่เท่ากัน ค่าของ Output ที่ได้จากการเชื่อมโยงกันนี้ จะถูกนำมาเปรียบเทียบกับ Output ที่ได้ตั้งเอาไว้ ถ้าค่าที่ออกมาเกิดความคลาดเคลื่อนจะนำไปสู่การปรับค่าหรือน้ำหนักของค่าที่ใส่ไว้ให้แต่ละ Input

(2) Decision Tree เป็นการนำข้อมูลมาสร้างแบบจำลองการพยากรณ์ในรูปแบบโครงสร้างต้นไม้ ซึ่งจะมีการทำงานแบบ Supervised Learning (การเรียนรู้แบบมีผู้สอน) สามารถสร้างแบบจำลองการจัดหมวดหมู่ได้จากกลุ่มตัวอย่างข้อมูลที่กำหนดไว้ก่อนล่วงหน้า เรียกว่า Training Set ได้อัตโนมัติ และพยากรณ์กลุ่มของรายการที่ยังไม่เคยนำมาจัดหมวดหมู่ได้โครงสร้างต้นไม้ตัดสินใจ ประกอบด้วย Root Node, Child และ Leaf Node (Mitchell, 1997) ดังแสดงในรูปที่ 2.2



รูปที่ 2.2 โครงสร้างต้นไม้ตัดสินใจ

(3) Memory Based Reasoning (MBR) การนำประสบการณ์การเรียนรู้ของมนุษย์โดยอาศัยการสังเกตที่เกิดขึ้นและสร้างรูปแบบของสิ่งนั้นขึ้นมา สามารถใช้ MBR เพื่อวิเคราะห์ฐานข้อมูลที่มีอยู่ และกำหนดลักษณะพิเศษของข้อมูลที่อยู่ในนั้น ซึ่งข้อมูลจะต้องมีลักษณะที่สมบูรณ์เพื่อการทำนายที่แม่นยำและมีประสิทธิภาพ (Air Force Research Laboratory Information Directorate, 2004)

(4) Cluster Detection จะแบ่งฐานข้อมูลออกเป็นส่วน ๆ เรียกว่า Segment (กลุ่มเรคคอร์ดที่มีลักษณะคล้ายกัน) ส่วนเรคคอร์ดที่ต่างกันจะอยู่นอกเซกเมนต์ ซึ่งจะถูกใช้เพื่อค้นหากลุ่มย่อย (Sub Group) ที่เหมือน ๆ กันในฐานข้อมูล โดยจะเพิ่มความถูกต้องในการวิเคราะห์ และสามารถเข้าถึงยังกลุ่มเป้าหมายได้อย่างถูกต้อง (Brezany and Fur, 2006)

(5) Link Analysis มุ่งเน้นทำงานบนเรคคอร์ดที่มีความสัมพันธ์กัน หรือเรียกว่า Association (Air Force Research Laboratory Information Directorate, 2004) โดยเทคนิคนี้จะเน้นไปที่รูปแบบการซื้อสินค้าหรือเหตุการณ์ที่เกิดขึ้นเป็นลำดับ มีอยู่ 3 เทคนิค คือ

- **Association Discovery** ใช้วิเคราะห์การซื้อขายสินค้าในรายการเดียวกัน ศึกษาความสัมพันธ์ของการซื้อสินค้า ซึ่งสินค้าเหล่านั้นอาจมีแนวโน้มที่จะถูกซื้อควบคู่กันไป ซึ่งการวิเคราะห์แบบนี้เรียกว่า ***Market Basket Analysis*** (Gutierrez, 2006)

- **Sequential Pattern Discovery** ถูกใช้ระบุความเกี่ยวเนื่องกันของการซื้อสินค้าของลูกค้า

- **Similar Time Sequence Discovery** ค้นหาความเกี่ยวเนื่องกันระหว่างข้อมูล 2 กลุ่ม ซึ่งขึ้นต่อกันทางด้านเวลา

(6) Genetic Algorithm คล้ายกับการสร้างพันธุกรรมที่ดีที่สุดบนขั้นตอนของวิวัฒนาการทางชีวภาพ แนวคิดหลัก คือ เมื่อเวลาผ่านไป วิวัฒนาการของเซลล์สิ่งมีชีวิตจะเลือกสายพันธุ์ที่ดีที่สุด ***"Fittest Species"*** (Goldberg, 1989)

(7) Rule Induction ดึงเอาชุดกฎเกณฑ์ต่าง ๆ มาสร้างเป็นเงื่อนไขหรือกรณี โดยมีวิธีการสร้างชุดของกฎที่เป็นอิสระ ซึ่งไม่จำเป็นต้องอยู่ในรูปแบบของโครงสร้างต้นไม้ (Quinlan, 1987)

(8) K-nearest Neighbor ใช้วิธีการจัดแบ่งคลาส (Class) โดยตัดสินใจว่าคลาสใดที่จะแทนเงื่อนไขหรือกรณีใหม่ ๆ ได้บ้าง โดยการตรวจสอบจำนวนบางจำนวนของกรณีหรือเงื่อนไขที่เหมือนกันหรือใกล้เคียงกันมากที่สุด ซึ่งจะหาผลรวมของจำนวนเงื่อนไขหรือกรณีต่าง ๆ สำหรับแต่ละคลาส และกำหนดเงื่อนไขใหม่ให้คลาสที่เหมือนกันกับคลาสที่ใกล้เคียงมากที่สุด (Coomans and Massart, 1982)

(9) Logistic Regression ซึ่งเป็นการวิเคราะห์ความถดถอยเชิงเส้น โดยใช้ในการพยากรณ์ผลลัพธ์ของ 2 ตัวแปร เช่น Yes/No , 0/1

สมการ Logistic Regression (Peduzzi, Concato, Kemper, Holford, and Feinstein, 1996) คำนวณจากอัตราส่วนความน่าจะเป็น

$$\text{อัตราส่วนความน่าจะเป็น} = \frac{\text{ความน่าจะเป็นที่จะเกิดเหตุการณ์}}{\text{ความน่าจะเป็นที่จะไม่เกิดเหตุการณ์}}$$

(10) Discriminant Analysis วิธีการทางคณิตศาสตร์ใช้ในการจำแนกและวิเคราะห์ วิธีนี้ได้รับการเผยแพร่ครั้งแรกในปี 1936 โดย R.A Fisher เพื่อแยกต้น Iris ออกเป็น 3 สายพันธุ์ วิธีการนี้ทำให้ค้นพบต้นไม้ประเภทอื่น ๆ อีกมาก ปัจจุบันถูกนำมาใช้ในงานด้านต่าง ๆ เช่น ทางการแพทย์ สังคมวิทยา และชีววิทยา (สมประสงค์ เสนารัตน์, 2553)

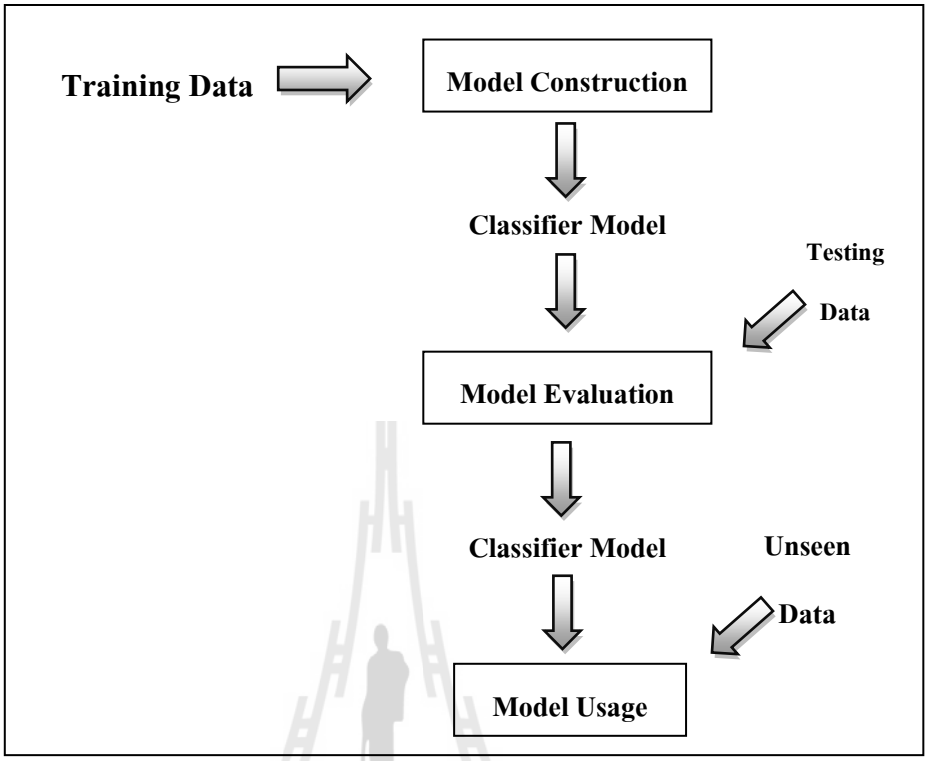
(11) Generalized Additive Models พัฒนามาจาก Linear Regression และ Logistic Regression สามารถใช้ได้กับปัญหาแบบ Regression และ Classification (Hastie and Tibshirani, 1990)

(12) Multivariate Adaptive Regression Splits สามารถค้นหาและแสดงรายการตัวแปรอิสระที่มีความสำคัญสูงสุดและกำหนดจุดแสดงความเป็นอิสระแต่ละตัวแปร (Hastie, Tibshirani, and Friedman, 2009)

2.4 การจำแนกประเภทข้อมูล (Classification)

เป็นกระบวนการสร้างโมเดลจัดการข้อมูลให้อยู่ในกลุ่มที่กำหนดมาให้ ตัวอย่างเช่น คัดเลือกและจัดกลุ่มนักเรียนที่มีผลการเรียน ดีมาก ดี ปานกลาง ไม่ดี โดยพิจารณาจากประวัติและผลการเรียน หรือแบ่งประเภทของลูกค้าว่าเชื่อถือได้หรือไม่ โดยพิจารณาจากข้อมูลประวัติของลูกค้าที่มีอยู่ (วิกิพีเดีย สารานุกรม, 2554b) กระบวนการจำแนกประเภทข้อมูล แสดงได้ดังรูปที่ 2.3

มหาวิทยาลัยเทคโนโลยีสุรนารี



รูปที่ 2.3 กระบวนการจำแนกประเภทข้อมูล (Classification)

กระบวนการทำงานของการจำแนกประเภทข้อมูล ประกอบด้วย 3 ขั้นตอนหลัก คือ การสร้างโมเดล (Model Construction), การประเมินโมเดล (Model Evaluation) และการใช้โมเดล (Model Usage)

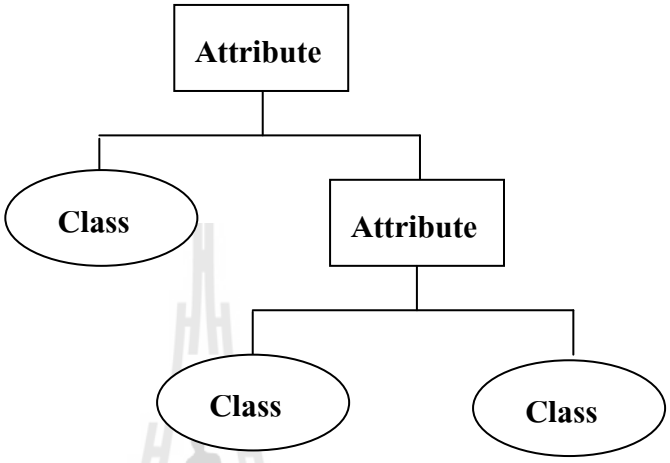
ขั้นตอนที่ 1 Model Construction (Learning)

เป็นขั้นตอนการสร้างโมเดลโดยการเรียนรู้จากข้อมูลที่ได้กำหนดคลาสไว้เรียบร้อยแล้ว ข้อมูลนี้เรียกว่า ข้อมูลฝึก (Training Data) ซึ่งโมเดลที่ได้อาจแสดงในรูปของต้นไม้ตัดสินใจ นิวรอลเน็ต สมการคณิตศาสตร์ หรือรูปแบบอื่น ๆ

ต้นไม้ตัดสินใจ (Decision Tree)

มีลักษณะคล้ายโครงสร้างต้นไม้ (รูปที่ 2.4) ซึ่งประกอบด้วย โหนดภายใน (Internal Node) ซึ่งจะแสดงคุณลักษณะของข้อมูล โดยที่จุดเริ่มต้นของต้นไม้ เรียกว่า โหนดราก (Root Node)

และกิ่งแสดงค่าของคุณลักษณะของแต่ละโหนด และลีฟโหนดหรือโหนดใบแสดงกลุ่ม ซึ่งเป็นผลลัพธ์ที่สามารถแยกแยะประเภทของข้อมูลได้



รูปที่ 2.4 โครงสร้างทั่วไปของต้นไม้ตัดสินใจ

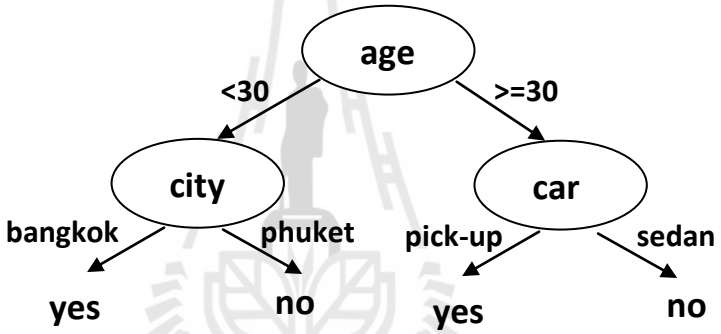
<u>ส่วนประกอบของต้นไม้ตัดสินใจ</u>

- โหนดภายใน (Internal Node) คือ คุณลักษณะต่าง ๆ ของข้อมูล ซึ่งเมื่อข้อมูลใด ๆ ตกลงมาที่โหนด จะใช้คุณลักษณะนี้เป็นตัวตัดสินใจว่าการจำแนกประเภทควรจะดำเนินการต่อไปในทิศทางใด โดยโหนดภายในที่เป็นจุดเริ่มต้นของต้นไม้ เรียกว่า โหนดราก (Root Node)
- กิ่ง (Branch) คือ ค่าของคุณลักษณะในโหนดภายในที่แตกกิ่งออกมา ซึ่งโหนดภายในจะแตกกิ่งเป็นจำนวนเท่ากับจำนวนค่าของคุณลักษณะในโหนดภายในนั้น
- โหนดใบ (Leaf Node) คือ กลุ่มต่าง ๆ ซึ่งเป็นผลลัพธ์ในการจำแนกประเภทข้อมูล

สมมุติว่ามีฐานข้อมูลซื้อขายรถ ดังตารางที่ 2.1 (นิตยา เกิดประสพ, 2552) โดยบันทึกประวัติการซื้อรถของลูกค้า โดยแต่ละแอททริบิวต์มีความสัมพันธ์เชื่อมโยงกัน ซึ่งสามารถเขียนโครงสร้างต้นไม้ตัดสินใจได้ดังรูปที่ 2.5

ตารางที่ 2.1 ข้อมูลการซื้อขายรถ

customerId	car	age	city	buyNewCar
c1	sedan	27	bangkok	yes
c2	pick-up	35	phuket	yes
c3	pick-up	40	bangkok	yes
c4	sedan	22	bangkok	yes
c5	pick-up	50	phuket	no
c6	sedan	25	phuket	no



รูปที่ 2.5 ต้นไม้ตัดสินใจจำแนกประเภทลูกค้าที่ซื้อ/ไม่ซื้อรถใหม่

มหาวิทยาลัยเทคโนโลยีสุรนารี

ซึ่งสามารถเขียน Classification Rules จากโครงสร้างต้นไม้ตัดสินใจ ได้เป็นดังนี้

IF (age < 30) AND (city = bangkok) THEN (buyNewCar = Yes)

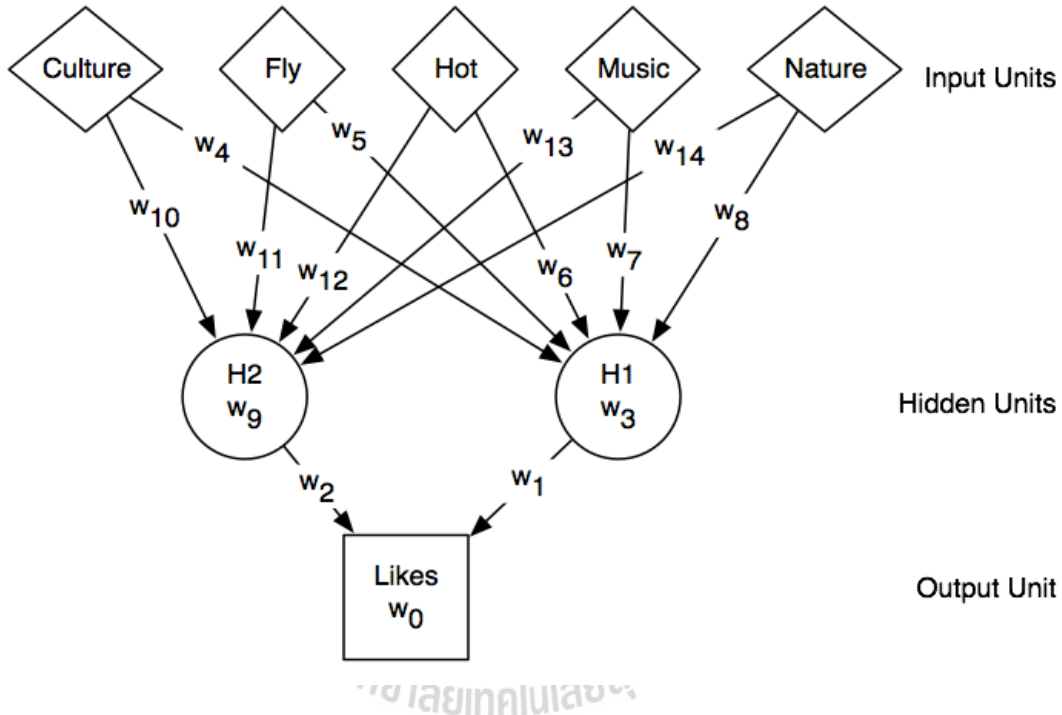
IF (age < 30) AND (city = phuket) THEN (buyNewCar = No)

IF (age > = 30) AND (car = sedan) THEN (buynew Car = No)

IF (age > = 30) AND (car = pick-up) THEN (buynew Car = Yes)

นิวรอลเน็ต (Neural Net)

เป็นเทคโนโลยีมาจากงานวิจัยด้านปัญญาประดิษฐ์ (Artificial Intelligence) เพื่อใช้ในการคำนวณค่าฟังก์ชันจากกลุ่มข้อมูล วิธีการ คือ ให้เครื่องเรียนรู้จากตัวอย่างต้นแบบและฝึก (Train) ให้ระบบได้รู้จักที่จะคิดแก้ปัญหาที่กว้างขึ้นได้ ในโครงสร้างของนิวรอลเน็ตจะประกอบด้วยโหนด (Node) สำหรับ Input และ Output โดยการประมวลผล จะคำนวณค่าน้ำหนักที่เชื่อมระหว่างโหนดที่กระจายอยู่ในโครงสร้างเป็นชั้น ๆ ดังตัวอย่างในรูปที่ 2.6



รูปที่ 2.6 โครงสร้างนิวรอลเน็ต (David and Alan, 2010)

ขั้นตอนที่ 2 Model Evaluation (Testing for Accuracy)

เป็นขั้นตอนการประเมินความถูกต้องโดยอาศัยข้อมูลที่ใช้ทดสอบ (Test Data) ซึ่งคลาสที่แท้จริงของข้อมูลที่ใช้ทดสอบนี้จะถูกนำมาเปรียบเทียบกับคลาสที่หามาได้จากรูปแบบหรือกฎ เพื่อทดสอบความถูกต้อง

ตัวอย่างผลการทดสอบกับโมเดล 3 รูปแบบ ได้แก่ Decistion Tree, Neural Network และ Logistic Regression โดยใช้ข้อมูล Train Data ซึ่งอยู่ในตารางที่ 2.2 มีข้อมูลทั้งหมด 14 เรคคอร์ด และข้อมูล Test Data ในตารางที่ 2.3 โดยมีข้อมูลทั้งหมด 5 เรคคอร์ด ผลการทดสอบโมเดลทั้ง 3 รูปแบบแสดงได้ดังตารางที่ 2.4 - 2.6 โดยจากผลการทดสอบสรุปได้ว่าโมเดลทั้ง 3 รูปแบบที่สร้างจากเทคนิคที่แตกต่างกัน มีความสามารถในการจำแนกประเภทข้อมูลเท่าเทียมกัน

ตารางที่ 2.2 Train Data ของข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ (Quinlan, 1992)

No.	Attributes				Class
	Outlook	Temperature	Humidity	Windy	
1	Sunny	Hot	High	False	No
2	Sunny	Hot	High	True	No
3	Overcast	Hot	High	False	Play
4	Rain	Mild	High	False	Play
5	Rain	Cool	Normal	False	Play
6	Rain	Cool	Normal	True	No
7	Overcast	Cool	Normal	True	Play
8	Sunny	Mild	High	False	No
9	Sunny	Cool	Normal	False	Play
10	Rain	Mild	Normal	False	Play
11	sunny	Mild	Normal	True	Play
12	Overcast	Mild	High	True	Play
13	Overcast	Hot	Normal	False	Play
14	rain	Mild	High	True	No

ตารางที่ 2.3 Test Data ของข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ (Quinlan, 1992)

No.	Attributes				Class
	Outlook	Temperature	Humidity	Windy	
1	Sunny	Hot	High	False	No
2	Sunny	Hot	High	True	No
3	Overcast	Hot	High	False	Play
4	Rain	Mild	High	False	Play
5	Rain	Cool	Normal	False	Play

ตารางที่ 2.4 ตารางแสดง Detailed Accuracy By Class ของโมเดล Decision Tree

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	1	0	1	1	1	1	Play
	1	0	1	1	1	1	No
Weighted Avg.	1	0	1	1	1	1	

ตารางที่ 2.5 ตารางแสดง Detailed Accuracy By Class ของโมเดล Neural Network

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	1	0	1	1	1	1	Play
	1	0	1	1	1	1	No
Weighted Avg.	1	0	1	1	1	1	

ตารางที่ 2.6 ตารางแสดง Detailed Accuracy By Class ของโมเดล Logistic Regression

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	1	0	1	1	1	1	Play
	1	0	1	1	1	1	No
Weighted Avg.	1	0	1	1	1	1	

ขั้นตอนที่ 3 Model Usage (Classifying New Data)

เป็นการประยุกต์ใช้โมเดลเพื่อจำแนกประเภทข้อมูลที่ไม่เคยเห็นมาก่อน (Unseen Data) และไม่ทราบประเภทของข้อมูลใหม่นี้ โดยจะทำการกำหนดคลาสให้กับ Objects ใหม่ที่ได้มา หรือ ทำนายค่าออกมาตามที่ต้องการ

ตัวอย่าง การใช้งานโมเดล เริ่มต้นจากขั้นตอนการสร้างโมเดล เพื่อจำแนกคลาส Tenured ของข้อมูลฝึกตามตารางที่ 2.7

ตารางที่ 2.7 Train Data ของข้อมูลการดำรงตำแหน่งของอาจารย์มหาวิทยาลัยแห่งหนึ่ง

NAME	RANK	YEARS	TENURED
Somchai	Assistant Prof	3	no
Somying	Assistant Prof	7	yes
Somsak	Professor	2	yes
Somjai	Associate Prof	7	yes
Somsri	Assistant Prof	6	no
Sompasong	Associate Prof	3	no

เมื่อข้อมูลตามตารางที่ 2.7 ถูกใช้เป็นข้อมูลฝึกเพื่อสร้างโมเดลจำแนกประเภทข้อมูล โดยจะต้องมีการประมาณค่าความแม่นยำ (Accuracy) ของแบบจำลองที่ได้เสียก่อนการนำไปใช้ ข้อมูลทดสอบ แสดงได้ดังตารางที่ 2.8

ตารางที่ 2.8 Test Data ของข้อมูลการดำรงตำแหน่งของอาจารย์มหาวิทยาลัยแห่งหนึ่ง

NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Jimmy	Assistant Prof	7	no
Mike	Professor	5	yes
Anna	Assistant Prof	7	yes

โมเดลของการจำแนกข้อมูลนั้น จะได้ผลลัพธ์ของกฎออกมา โดยมีกฎ

IF rank = 'professor' OR (rank = 'Assistant Prof'and years > 6) THEN tenured = 'yes' เป็นหนึ่งในผลลัพธ์ ซึ่งสามารถทำนายผลของข้อมูล Unseen Data ได้ ดังตารางที่ 2.9

Unseen Data → (Jeff, Professor, 4)

ตารางที่ 2.9 ข้อมูลผลการทำนายของข้อมูล Unseen Data

NAME	RANK	YEARS	TENURED
Holly	Professor	4	**yes**

2.5 อัลกอริทึม ID3

ID3 ย่อมาจาก Iterative Dichotomiser 3 เป็นอัลกอริทึมที่สร้างขึ้นในปี 1986 โดยนักคณิตศาสตร์และนักคอมพิวเตอร์ ชื่อ Ross Quinlan เพื่อใช้ประโยชน์ในการสร้างต้นไม้ประกอบการตัดสินใจ โดยอัลกอริทึมนี้จะใช้การจัดแบ่งประเภทข้อมูล ซึ่งสามารถทำได้หลายแบบ แต่ ID3 จะทำการค้นหาเพื่อเลือกรูปแบบการแบ่งประเภทที่ดีที่สุดมาใช้ (Bahety, 2006)

วิธีการของอัลกอริทึม ID3 จะใช้การเลือกคุณลักษณะ เพื่อเป็นตัวจัดแบ่งประเภทข้อมูล ซึ่งจะเห็นได้ว่าสามารถสร้างเป็นโครงสร้างต้นไม้ตัดสินใจได้มากกว่าหนึ่งแบบ หลักการทำงานสำคัญของ ID3 เพื่อเลือกรูปแบบการสร้างต้นไม้ที่ดีที่สุดนั้น คือ การเลือกตัวจัดแบ่งข้อมูลโดยใช้ค่าทางสถิติที่เรียกว่า "การเพิ่มคุณค่าข้อมูล (Information Gain)" แล้วนำมาทดสอบแต่ละโหนดของต้นไม้ตัดสินใจ

วิธีการวัดการเพิ่มคุณค่าข้อมูล คือ การวัดอัตราการลดลงของค่า Entropy หรือค่าความไม่บริสุทธิ์ของข้อมูลในโหนด เนื่องจากความบริสุทธิ์ของข้อมูล แปรผกผันกับค่า Entropy ดังนั้นการลดค่า Entropy โดยการหาตัวจัดแบ่งข้อมูลที่ดีกว่า ย่อมนำไปสู่การสร้างต้นไม้ที่ให้ค่าการเพิ่มของคุณค่าข้อมูลที่สูงขึ้น

การวัดค่า Entropy คือ การวัดความแตกต่างของกลุ่มของข้อมูลที่ใช้กันอย่างแพร่หลาย ในกรณีที่มีข้อมูลหลายประเภท สามารถวัดค่า Entropy โดยพยายามหาค่าที่น้อยที่สุด คือ

- กลุ่มข้อมูลที่เหมือนกันหมด จะได้ค่า Entropy เป็น 0
- กลุ่มข้อมูลที่แตกต่างกัน เท่ากันเป็นสองกลุ่ม เช่น กลุ่มตัวเลขบวก และลบ มีจำนวนข้อมูลในสองกลุ่มเท่ากัน จะได้ค่า Entropy เป็น 1

กฎของ Entropy คือ ให้ชุดของข้อมูล M ประกอบด้วยค่าที่เป็นไปได้ คือ { m_1, m_2, …, m_n } และให้ความน่าจะเป็นที่จะเกิดค่า m_i มีค่าเท่ากับ $P(m_i)$ จะได้ว่าค่า Entropy ของ M ซึ่งใช้วัดค่าสารสนเทศโดยเฉลี่ยเพื่อระบุประเภทของข้อมูล เขียนแทนด้วย I(M) คำนวณได้จากสูตร (นิตยา เกิดประสพ, 2552)

$$I(M) = \sum_{i}^{n} - P(m_i)\log_2 P(m_i)$$

ค่า Gain ของคุณสมบัติ X ที่ใช้แบ่งข้อมูลที่โหนดหนึ่ง สามารถคำนวณได้จากการลบค่า Entropy ทั้งหมดที่โหนดนี้กับค่า Entropy ที่ได้หลังจากแบ่งด้วยคุณสมบัติ X ดังนี้

$$Gain(X) = I(T) - I_x(T)$$

ตัวอย่างการหากฎจากอัลกอริทึม ID 3 แสดงได้ด้วยตัวอย่างข้อมูลจากตารางที่ 2.10

ตารางที่ 2.10 ข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ (Quinlan, 1992)

No.	Attributes				Class
	Outlook	Temperature	Humidity	Windy	
1	Sunny	Hot	High	False	No
2	Sunny	Hot	High	True	No
3	Overcast	Hot	High	False	Play
4	Rain	Mild	High	False	Play
5	Rain	Cool	Normal	False	Play
6	Rain	Cool	Normal	True	No
7	Overcast	Cool	Normal	True	Play
8	Sunny	Mild	High	False	No
9	Sunny	Cool	Normal	False	Play
10	Rain	Mild	Normal	False	Play
11	sunny	Mild	Normal	True	Play
12	Overcast	Mild	High	True	Play
13	Overcast	Hot	Normal	False	Play
14	rain	Mild	High	True	No

คำนวณหาค่า Gain จากข้อมูลตารางที่ 2.10 มีขั้นตอนโดยทั่วไป ดังนี้

- นำแอททริบิวต์แต่ละแอททริบิวต์มาคำนวณค่า จากสูตร $Gain(X) = I(T) - I_x(T)$
- จะได้ค่า Gain (X) ของแต่ละแอททริบิวต์ โดยเลือกแอททริบิวต์ ที่มีค่า Gain สูงสุดมาเป็นโหนดแม่ ส่วนชั้นถัดลงมาจะใช้ Gain จากแอททริบิวต์ที่เหลือ ซึ่งก่อนเลือกแอททริบิวต์ที่จะใช้วัดค่า Gain จะต้องทำการคำนวณหาค่า Entropy ก่อน
- หา Gain ของแอททริบิวต์แต่ละแอททริบิวต์ที่เหลืออยู่ ถ้ายังมีข้อมูลที่เป็นคนละคลาสปะปนกันอยู่ ให้ทำการวนซ้ำไปเรื่อย ๆ จนข้อมูลหมด หรือแยกได้คลาสที่มีสมาชิกเป็นไปในรูปแบบเดียวกัน

คำนวณค่า Gain ของแอททริบิวต์ outlook

จากสูตรหาค่า Entropy

$$I(T) = \sum_{i=1}^{n} -P(Ti)\log P(Ti)$$

คำนวณค่า Entropy ของข้อมูลทั้งหมด คือ $I(T) = [-\frac{5}{14}\log_2\frac{5}{14} - \frac{9}{14}\log_2\frac{9}{14}]$

$$= 0.94$$

คำนวณค่า Entropy เมื่อจำแนกข้อมูลด้วยแอททริบิวต์ outlook

$$I(outlook - sunny) = \frac{5}{14}(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5})$$

$$= \frac{5}{14}(0.41 + 0.53) \quad = 0.35$$

$$I(outlook - overcast) = \frac{4}{14}(-\frac{4}{4}\log_2\frac{4}{4})$$

$$= 0$$

$$I(outlook - rain) = \frac{5}{14}(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5})$$

$$= \frac{5}{14}(0.53 + 0.44) \quad = 0.35$$

ดังนั้นค่าที่ได้จากสูตร $Gain\ (X) = I(T) - I_X(T)$

$$Gain(outlook) = I(T) - I_{outlook}(sunny, overcast, rain)$$

จะได้ = 0.94 – (0.35+0+0.35)

= 0.94 – 0.70

= 0.24

คำนวณค่า Gain ของแอททริบิวต์ temperature

จากสูตรหาค่า Entropy

$$I\ (T) = \sum_{i=1}^{n} - P(Ti) \log P(Ti)$$

จะได้ค่า Entropy ของข้อมูลทั้งหมด คือ $I\ (T) = [-\frac{5}{14}\log_2\frac{5}{14} - \frac{9}{14}\log_2\frac{9}{14}]$

$= 0.94$

คำนวณค่า Entropy เมื่อจำแนกข้อมูลด้วยแอททริบิวต์ temperature

$$I(temperature - hot) = \frac{4}{14}(-\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4})$$

$$= \frac{4}{14}(1) \quad = 0.29$$

$$I(temperature - mild) = \frac{6}{14}(-\frac{2}{6}\log_2\frac{2}{6} - \frac{4}{6}\log_2\frac{4}{6})$$

$$= \frac{6}{14}(0.92) \quad = 0.39$$

$$I(temperature - cool) = \frac{4}{14}(-\frac{1}{4}\log_2\frac{1}{4} - \frac{3}{4}\log_2\frac{3}{4})$$

$$= \frac{4}{14}(0.81) \quad = 0.23$$

ดังนั้นค่าที่ได้จากสูตร $Gain\ (X) = I(T) - I_X(T)$

$$Gain(temperature) = I(T) - I_{temperature}(hot, mild, cool)$$

จะได้ = 0.94 – (0.29 + 0.39 + 0.23)

= 0.94 – (0.91)

= 0.03

คำนวณค่า Gain ของแอททริบิวต์ humidity

จากสูตรหาค่า Entropy

$$I\ (T) \quad = \sum_{i=1}^{n} \ - P(Ti) \log\ P(Ti)$$

จะได้ค่า Entropy ของข้อมูลทั้งหมด คือ $I\ (T) = [-\frac{5}{14}\log_2\frac{5}{14} - \frac{9}{14}\log_2\frac{9}{14}]$

$$= 0.94$$

คำนวณค่า Entropy เมื่อจำแนกข้อมูลด้วยแอททริบิวต์ humidity

$$I(humidity - high) = \frac{7}{14}(-\frac{4}{7}\log_2\frac{4}{7} - \frac{3}{7}\log_2\frac{3}{7})$$

$$= \frac{7}{14}(0.99) \quad = 0.50$$

$$I(humidity - normal) = \frac{7}{14}(-\frac{1}{7}\log_2\frac{1}{7} - \frac{6}{7}\log_2\frac{6}{7})$$

$$= \frac{7}{14}(0.59) \quad = 0.30$$

ดังนั้นค่าที่ได้จากสูตร $Gain\ (X) = I(T) - I_X(T)$

$$Gain(humidity) = I(T) - I_{humidity}(high, normal)$$

จะได้ = 0.94 – (0.50 + 0.30)

= 0.94 – 0.80

= 0.14

คำนวณค่า Gain ของแอททริบิวต์ windy

จากสูตรหาค่า Entropy

$$I\ (T) \quad = \sum_{i=1}^{n} \ - P(Ti) \log\ P(Ti)$$

จะได้ค่า Entropy ของข้อมูลทั้งหมด คือ $I\ (T) = [-\frac{5}{14}\log_2\frac{5}{14} - \frac{9}{14}\log_2\frac{9}{14}]$

$$= 0.94$$

คำนวณค่า Entropy เมื่อจำแนกข้อมูลด้วยแอททริบิวต์ windy

$$I(windy-false)=\frac{8}{14}(-\frac{2}{8}\log_2\frac{2}{8}-\frac{6}{8}\log_2\frac{6}{8})$$

$$=\frac{8}{14}(0.82) \quad =0.47$$

$$I(windy-true)=\frac{6}{14}(-\frac{3}{6}\log_2\frac{3}{6}-\frac{3}{6}\log_2\frac{3}{6})$$

$$=\frac{6}{14}(1) \quad =0.43$$

ดังนั้นค่าที่ได้จากสูตร $Gain\ (X) = I(T) - I_X(T)$

$$Gain(windy) = I(T) - I_{windy}(false, true)$$

จะได้ = 0.94 – (0.47 + 0.43)

= 0.94 – 0.90

= 0.04

สรุปค่า Gain แต่ละแอททริบิวต์ มีดังนี้

Gain(outlook) = 0.24

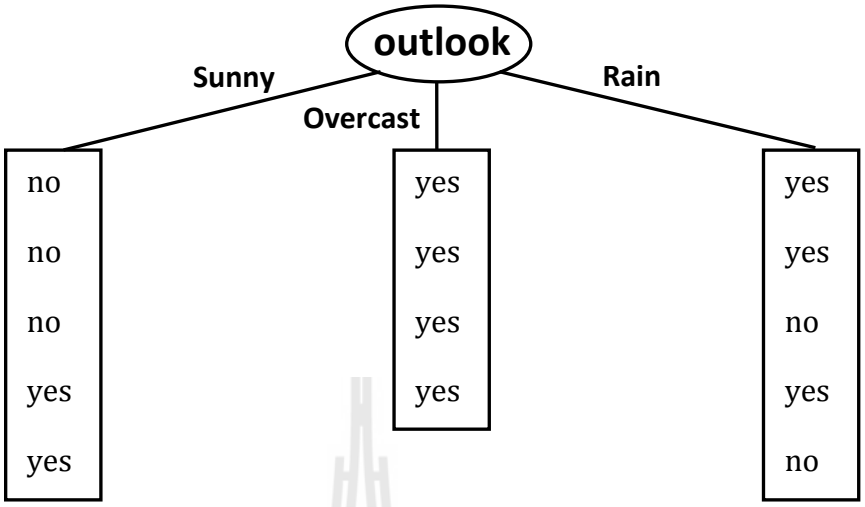
Gain(temperature) = 0.03

Gain(humidity) = 0.14

Gain(windy) = 0.04

จะเห็นได้ว่า gain(outlook) มีค่ามากที่สุด จึงเลือกเป็นโหนดแม่ (แสดงดังรูปที่ 2.7) และทำการหาแอททริบิวต์ที่ต่อจาก outlook – sunny และ outlook – rain ต่อไปอีกชั้นหนึ่ง เนื่องจากยังมีข้อมูลที่รวมกันอยู่ ส่วน outlook – overcast ไม่ต้องหาแอททริบิวต์ต่อไป เนื่องจากสามารถคัดแยกประเภทข้อมูลได้สำเร็จแล้ว คือ มีค่าของ class – yes ทั้งหมด

มหาวิทยาลัยเทคโนโลยีสุรนารี



รูปที่ 2.7 โครงสร้างต้นไม้ขั้นที่ 1

คำนวณค่า Gain กรณีแอททริบิวต์ outlook - sunny

จากสูตรหาค่า Entropy

$$I(T) = \sum_{i=1}^{n} -P(Ti)\log P(Ti)$$

จะได้ค่า Entropy ทั้งหมด คือ $I(outlook-sunny) = [-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}]$

$$= 0.97$$

หาค่า Gain ของแอททริบิวต์ *temperature* เพื่อใช้แยกกลุ่มข้อมูลที่มีค่า *outlook - sunny*

จะได้ค่าของแอททริบิวต์ที่เหลือ คือ $I(temperature-hot) = \frac{2}{5}(-\frac{2}{2}\log_2\frac{2}{2} - 0)$

$$= 0$$

$$I(temperature-mild) = \frac{2}{5}(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2})$$

$$= 0.4$$

$$I(temperature-cool) = \frac{1}{5}(0 - \frac{1}{1}\log_2\frac{1}{1})$$

$$= 0$$

ดังนั้นค่าที่ได้จากสูตร $Gain(X) = I(T) - I(X)(T)$

$Gain(outlook\text{-}sunny\&temperature) = I(outlook\text{-}sunny) - I_{outlook\text{-}sunny\&temperature}(hot,mild,cool)$

จะได้ = 0.97 – 0.4

= 0.57

หาค่า Gain ของแอททริบิวต์ *humidity* เพื่อใช้แยกกลุ่มข้อมูลที่มีค่า *outlook - sunny*

จะได้ค่าของแอททริบิวต์ที่เหลือ คือ $I(humidity - high) = \frac{3}{5}(-\frac{3}{3}\log_2\frac{3}{3} - 0)$

$= 0$

$I(humidity - normal) = \frac{2}{5}(0 - \frac{2}{2}\log_2\frac{2}{2})$

$= 0$

$Gain(outlook\text{-}sunny\&humidity) = I(outlook\text{-}sunny) - I_{outlook\text{-}sunny\&humidity}(hot,mild,cool)$

จะได้ = 0.97 – 0

= 0.97

หาค่า Gain ของแอททริบิวต์ *windy* เพื่อใช้แยกกลุ่มข้อมูลที่มีค่า *outlook - sunny*

จะได้ค่าของแอททริบิวต์ที่เหลือ คือ $I(windy - false) = \frac{3}{5}(-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3})$

$= 0.56$

$I(windy - true) = \frac{2}{5}(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2})$

$= 0.4$

$Gain(outlook\text{-}sunny\&windy) = I(outlook\text{-}sunny) - I_{outlook\text{-}sunny\&windy}(hot,mild,cool)$

จะได้ = 0.97 – 0.96

= 0.01

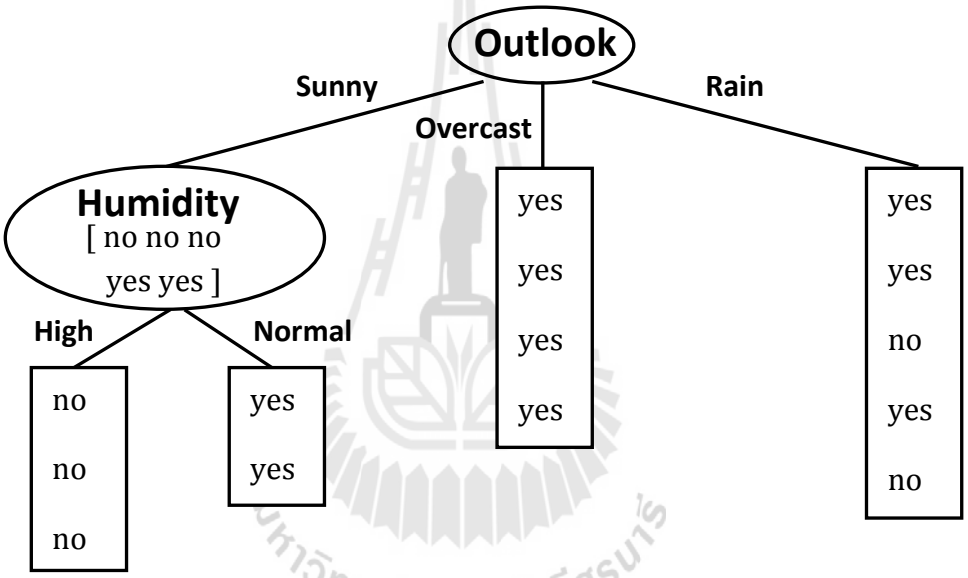
สรุปค่า Gain แต่ละแอททริบิวต์ ที่มีค่า *outlook – sunny* มีดังนี้

Gain(temperature) = 0.57

Gain(humidity) = 0.97

Gain(windy) = 0.01

ค่า Gain ของแอททริบิวต์ humidity มีค่ามากที่สุด จึงเลือกเป็นโหนดที่ต่อจากแอททริบิวต์ *outlook* ที่มีค่าเท่ากับ *sunny* ดังรูปที่ 2.8



รูปที่ 2.8 โครงสร้างต้นไม้ขั้นที่ 2

คำนวณค่า Gain กรณีแอททริบิวต์ outlook - rain

จากสูตรหาค่า Entropy

$$I(T) = \sum_{i=1}^{n} -P(Ti)\log P(Ti)$$

จะได้ค่า Entropy ทั้งหมด คือ $I(outlook - rain) = [-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}]$

$$= 0.97$$

หาค่า Gain ของแอททริบิวต์ *temperature* เพื่อใช้แยกกลุ่มข้อมูลที่มีค่า *outlook - rain*

จะได้ค่าของแอททริบิวต์ที่เหลือ คือ

$$I(hot) = 0$$

$$I(mild) = \frac{3}{5}(-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3})$$

$$= 0.56$$

$$I(cool) = \frac{2}{5}(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2})$$

$$= 0.4$$

$$Gain(outlook\text{-}rain\&temperature) = I(outlook\text{-}rain) - I_{outlook\text{-}rain\&temperature}\ (hot,mild,cool)$$

จะได้ $= 0.97 - 0.96$

$= 0.01$

หาค่า Gain ของแอททริบิวต์ *humidity* เพื่อใช้แยกกลุ่มข้อมูลที่มีค่า *outlook - rain*

จะได้ค่าของแอททริบิวต์ที่เหลือ คือ

$$I(high) = \frac{2}{5}(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2})$$

$$= 0.4$$

$$I(normal) = \frac{3}{5}(-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3})$$

$$= 0.56$$

$$Gain(outlook\text{-}rain\&humidity) = I(outlook\text{-}rain) - I_{outlook\text{-}rain\&humidity}\ (high,normal)$$

จะได้ $= 0.97 - 0.96$

$= 0.01$

หาค่า Gain ของแอททริบิวต์ *windy* เพื่อใช้แยกกลุ่มข้อมูลที่มีค่า *outlook - rain*

จะได้ค่าของแอททริบิวต์ที่เหลือ คือ

$$I(false) = \frac{3}{5}(0 - \frac{3}{3}\log_2\frac{3}{3})$$

$$= 0$$

$$I(true) = \frac{2}{5}(-\frac{2}{2}\log_2\frac{2}{2} - 0)$$

$$= 0$$

$$Gain(outlook\text{-}rain\&windy) = I(outlook\text{-}rain) - I_{outlook\text{-}rain\&windy}\ (high,normal)$$

จะได้ $= 0.97 - 0$

$= 0.97$

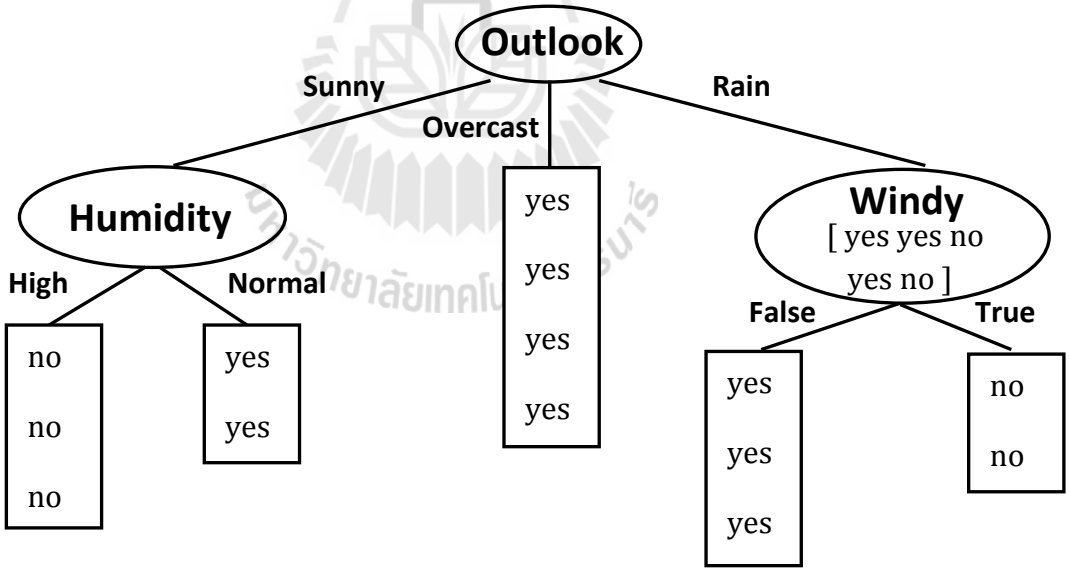
สรุปค่า Gain แต่ละแอททริบิวต์ ที่มีค่า *outlook – rain* มีดังนี้

gain(temperature) = 0.01

gain(humidity) = 0.01

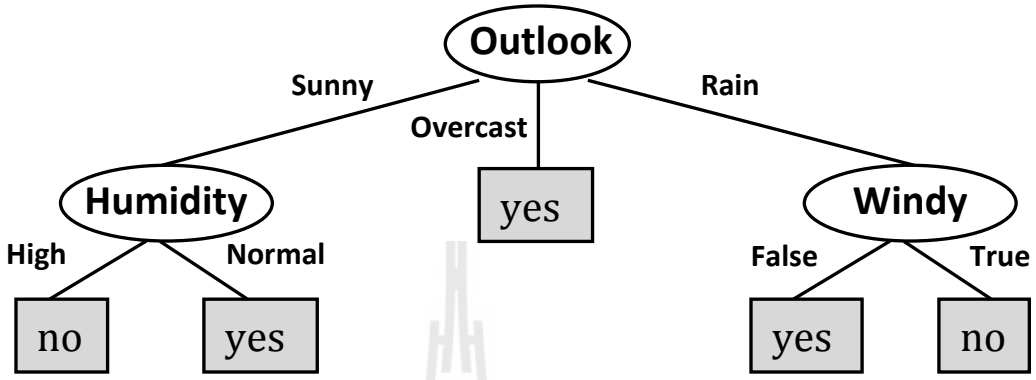
gain(windy) = 0.97

ค่า Gain ของแอททริบิวต์ windy มีค่ามากที่สุด จึงเลือกเป็นโหนดที่ต่อจากแอททริบิวต์ *outlook* ที่มีค่าเท่ากับ *rain* แสดงโครงสร้างต้นไม้ได้ดังรูปที่ 2.9



รูปที่ 2.9 โครงสร้างต้นไม้ขั้นที่ 3

ดังนั้นแผนภาพต้นไม้ที่ได้จากข้อมูลนี้ คือ แผนภาพที่มีแอททริบิวต์ outlook เป็นโหนดราก และแอททริบิวต์ humidity และ windy เป็นโหนดลูก ผลของการตัดสินใจ yes/no เป็นโหนดใบ (ดังรูปที่ 2.10)



รูปที่ 2.10 โครงสร้างต้นไม้ของข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ

โมเดลจากแผนภาพโครงสร้างต้นไม้สามารถเขียนเป็นกฎได้ดังนี้

rule 1 if outlook = sunny and humidity = high then don't play

rule 2 if outlook = sunny and humidity = normal then play

rule 2 if outlook = overcast then play

rule 3 if outlook = rain and windy = false then play

rule 4 if outlook = rain and windy = true then don't play

2.6 ภาษาเชิงตรรกะ

การพัฒนางานวิจัยนี้ใช้วิธีการเขียนโปรแกรมเชิงตรรกะ ภาษาที่นิยมใช้ในการเขียนโปรแกรมเชิงตรรกะ คือ ภาษาโปรล็อก (Prolog) ย่อมาจาก PROgrammation en LOGique (Logic Programming) สร้างขึ้นโดย Alain Colmerauer ราว ค.ศ. 1972 ภาษาโปรล็อกเกิดจากความพยายามที่จะสร้างภาษาที่อาศัยวิธีการทางตรรกศาสตร์แทนที่จะกำหนดคำสั่งเป็นขั้นตอนอย่างละเอียดให้กับคอมพิวเตอร์ (Hodgson, 1999)

ภาษาโปรล็อกถูกนำไปใช้ในการทำโปรแกรมสำหรับปัญญาประดิษฐ์ และในงานภาษาศาสตร์เชิงคำนวณ โดยเฉพาะการประมวลผลภาษาธรรมชาติ จุดเด่นของภาษานี้ คือ ไวยากรณ์

และความหมายของภาษาชัดเจน สามารถเขียนโปรแกรมให้ทำงานได้จริงภายในไม่กี่บรรทัด งานวิจัยจำนวนมากที่ทำให้เกิดการพัฒนาภาษาโปรล็อก ในปัจจุบันนั้นเป็นผลมาจากโครงการระบบคอมพิวเตอร์ยุคที่ห้าของรัฐบาลญี่ปุ่นที่เลือกภาษาโปรล็อกเป็นภาษาแก่น (Kernel Language) ของระบบปฏิบัติการ (Visual Prolog, 2011)

2.6.1 หลักการพื้นฐานของภาษาเชิงตรรกะ

ภาษาโปรล็อกมีพื้นฐานมาจากแคลคูลัสภาคแสดง (Predicate Calculus) แนวคิดพื้นฐานที่สำคัญ ได้แก่

การทำให้เท่ากัน (Unification) คือ การทำ pattern matching เพื่อทำให้สองเทอมมีค่าที่ตรงกันและการแทนค่าในตัวแปร ซึ่งการกำหนดตัวแปรนั้นสามารถกำหนดเป็น ตัวอักษร ตัวเลข และเส้นใต้อักษร โดยตัวแปรนั้นจะต้องขึ้นต้นด้วยตัวพิมพ์ใหญ่ เช่น เมื่อมีค่าความจริงเป็น eat(fish). เมื่อสอบถามด้วย ?-eat(X). โปรแกรมจะทำ การแทนค่า fish ในตัวแปร X เพื่อให้ตรงกับค่าความจริง ซึ่งเป็นหลักการ Unification ดังนั้นผลลัพธ์ที่ได้ คือ X = fish.

การเรียกซ้ำจากส่วนท้าย (Tail Recursion) ภาษาโปรล็อกมักใช้ลิสต์เป็นส่วนสำคัญในการดำเนินการ เมื่อคำสั่งใด ๆ ที่ทำงานร่วมกับลิสต์ มักจะนิยมใช้การดำเนินการแบบ Tail Recursion ตัวอย่างเช่น

```
last([E], E).
last([_|T], E) :- last(T, E).
```

เป็นคำสั่งที่หาค่าตัวสุดท้ายภายในลิสต์ที่กำหนดให้ ซึ่งการใช้คำสั่ง last(T,E) เป็นการดำเนินการแบบ Tail Recursion ซึ่งเป็นการส่งค่าที่เหลือกลับไปดำเนินการใหม่ จนกว่าจะได้ค่าตัวสุดท้ายของลิสต์

การย้อนรอย (Backtracking) ภาษาโปรล็อกมักนิยมใช้การย้อนรอยเพื่อค้นหาคำตอบการค้นหาจะกระทำในแนวลึกก่อน (depth-first search) เมื่อยังไม่พบคำตอบก็จะย้อนกลับไปค้นหาคำตอบจากจุดเริ่มต้น ตัวอย่างเช่น

```
animal(lion).
animal(sparrow).
has_feathers(sparrow).

bird(X) :- animal(X) , has_feathers(X).
```

เมื่อสอบถามว่า bird(B) คำตอบที่ได้ คือ B = sparrow

?- bird(B).

B = sparrow

วิธีการดำเนินการเริ่มจาก bird(X) :- animal(X) , has_feathers(X) ซึ่งจะแทนค่าที่ animal(X) โดยที่ X แทนค่าได้ lion แต่เมื่อคิด has_feathers(lion) จะไม่เป็นความจริง ดังนั้น X เป็น lion ไม่ได้ โปรล็อกจะกลับไปคิดบรรทัดบนใหม่อีกครั้งที่ animal(X) และ X แทนค่าได้ sparrow เมื่อแทนค่าใน has_feathers(sparrow) เป็นจริง ดังนั้น คำตอบ B = sparrow การย้อนกลับไปคิดนี้เรียกว่า การย้อนรอย (Backtracking)

2.6.2 แบบชนิดของข้อมูล

อะตอม

เป็นค่าคงที่ซึ่งเขียนแทนด้วยข้อความ โดยอะตอมสามารถประกอบด้วย ตัวอักษร ตัวเลข เส้นใต้อักษร (Underscore) และจะต้องขึ้นต้นด้วย ตัวพิมพ์เล็ก โดยปกติแล้วถ้าต้องการอะตอมที่ใช้เครื่องหมายพิเศษ จะเขียนเครื่องหมาย (') กำกับไว้ เช่น '+' เป็น อะตอม แต่ + เป็นตัวดำเนินการ

ตัวเลข

ภาษาโปรล็อกส่วนใหญ่จะไม่แบ่งแยกระหว่างเลขจำนวนเต็ม กับเลขจำนวนจริง

ตัวแปร

ตัวแปรจะสามารถประกอบด้วย ตัวอักษร ตัวเลข และเส้นใต้อักษร โดยจะต้องขึ้นต้นด้วย ตัวพิมพ์ใหญ่ ตัวแปรในภาษาโปรล็อกไม่ใช่ที่เก็บข้อมูล แต่จะมีลักษณะคล้ายรูปแบบ ซึ่งกำหนดไว้เพื่อการทำให้เท่ากัน

ตัวแปรนิรนาม (Anonymous Variable) จะเขียนโดยใช้ เครื่องหมายเส้นใต้อักษรเพียงตัวเดียว (_) หมายความว่า ไม่สนใจว่าตัวแปรนั้นจะมีค่าเป็นอย่างไร ซึ่ง Input เข้ามาเป็นแบบใดก็ได้

พจน์

พจน์ (Term) ใช้แทนข้อมูลที่มีความซับซ้อน ประกอบด้วย ส่วนหัว (Head) เป็นอะตอม เรียกว่า ฟังก์เตอร์ (Functor) และพารามิเตอร์ต่าง ๆ (ไม่กำหนดประเภท) จำนวนพารามิเตอร์จะเรียกว่า Arity พจน์สามารถเขียนแทนโดยใช้เพียง Head และ Arity โดยเขียนเป็น Functor/Arity

ลิสต์

ลิสต์ไม่ใช่ข้อมูลแบบเดี่ยว แต่เป็นโครงสร้างที่ใช้เก็บกลุ่มของข้อมูลมีการนิยามแบบเรียกซ้ำ คือ

- อะตอม [ ] ใช้แทนลิสต์ว่าง
- ถ้า T เป็นลิสต์ และ H เป็นส่วนย่อย จะใช้พจน์ [H|T]

ส่วนย่อยแรก เรียกว่าส่วนหัว (H หรือ Head) จะตามด้วยส่วนที่เหลือของลิสต์ ที่เรียกว่าส่วนหาง (T หรือ Tail) เช่น ลิสต์ [a, b, c] สามารถเขียนในแบบนิยามเรียกซ้ำเป็น [a | [b, c]] หรือ [a, b| [c]] หรือ [a, b, c | []] การประมวลผลข้อมูลในลิสต์ จะทำโดยประมวลผลข้อมูลส่วนหัวก่อน แล้วค่อยทำส่วนที่เหลือ โดยใช้การเรียกซ้ำ

ลิสต์สามารถเขียนได้หลายแบบ ตามความสะดวกของโปรแกรมเมอร์

- เขียนส่วนย่อยทุกตัว: [a, 1, X, f(x), sum(H,T)]
- เขียนส่วนแรกตัวเดียว: [a | T]
- เขียนส่วนแรกหลายตัว: [a, b, f(x) | T]

สายอักขระ

สายอักขระจะเขียนอยู่ในเครื่องหมายอัญประกาศคู่ เช่น “Hello” , “Hello world”

2.6.3 โครงสร้างของโปรแกรมและการดำเนินการ

ข้อเท็จจริง

การเขียนโปรแกรมภาษาโปรล็อกจะแตกต่างจากการใช้ภาษาเชิงกระบวนคำสั่ง (Procedural Langugage) โดยเริ่มจากการสร้างฐานข้อมูลข้อเท็จจริงและกฎ จากนั้นจึงใช้การสอบถาม (Queries) เพื่อหาคำตอบ หน่วยพื้นฐานของภาษาโปรล็อกคือ เพรดิเคต (Predicate) ซึ่งใช้นิยามความจริง เพรดิเคตจะเขียนอยู่ในรูปพจน์ ซึ่งประกอบด้วยส่วนหัว และอาร์กิวเมนต์ เช่น

eat(banana). % eat คือ เพรดิเคต ซึ่งมีอาร์กิวเมนต์หนึ่งตัว คือ banana

brother(june,jack). % brother คือ เพรดิเคต ซึ่งมีอาร์กิวเมนต์สองตัว คือ june และ jack

มีเพรดิเคตหลายตัว ที่กำหนดไว้ในตัวภาษา เพื่อทำให้โปรแกรมเมอร์สามารถเรียกใช้ได้สะดวก เช่น เพรดิเคต “writeln”

writeln(‘Hello’). %จะแสดงผลออกหน้าจอเว้นบรรทัด 1 บรรทัด

กฎ

listens2music(yolanda) :- happy(yolanda).

เครื่องหมาย " **:-** " แปลว่า " **ถ้า** " กฎนี้หมายความว่า listens2music(yolanda) เป็นจริง ถ้า happy(yolanda) เป็นจริง

การประเมินค่า

เมื่อส่วนแปลคำสั่งของภาษาโปรล็อกได้รับการสอบถาม จะค้นหาข้อเท็จจริงที่เข้ากันได้กับการสอบถามนั้น ถ้าไม่มีข้อเท็จจริงอยู่ จะตรวจสอบกฎที่ทำให้ได้ข้อเท็จจริงไปจนจบโปรแกรม เช่น

happy(yolanda).

listens2music(mia).

listens2music(yolanda):- happy(yolanda).

playsAirGuitar(mia):- listens2music(mia).

playsAirGuitar(yolanda):- listens2music(yolanda).

เมื่อสอบถามตามคำถามข้างล่าง จะได้คำตอบว่า yes. เนื่องจากมีข้อเท็จจริงอยู่ในฐานความรู้ (Knowledge Base)

?- playsAirGuitar(mia).

yes.

นิเสธ

การสอบถามจะเป็นเท็จ เมื่อไม่สามารถหาข้อเท็จจริงหรือกฎที่สนับสนุนการสอบถามนั้นได้ คือ ถ้าทุกสิ่งที่ควรรู้เก็บไว้ในฐานข้อมูลแล้ว ดังนั้นจึงไม่มีสิ่งใดที่อยู่ภายนอกขอบเขตนี้รวมถึงสิ่งที่ไม่รู้ หรืออีกนัยหนึ่ง ข้อเท็จจริงที่ยังไม่รู้ว่าเป็นจริง (หรือเท็จ) จะสมมุติว่าเป็นเท็จ เพื่อค้นหาความจริงในฐานความรู้เพื่อให้ได้คำตอบที่เป็นจริง

enjoys(vincent,X):- burger(X), **NOT** bigKahunaBurger(X).

enjoys(vincent,X):- burger(X), \+ bigKahunaBurger(X).

NOT และ \+ มีความหมายเป็นเท็จ ถ้าอยู่หน้าเพรดิเคตใด เพรดิเคตนั้นจะเป็นเท็จ (fail) ทันที

การดำเนินการ Cut

การดำเนินการ Cut ที่ใช้เครื่องหมาย " ! " หมายถึง เพรดิเคตที่มี 0 อาร์กิวเมนต์ (!/0) โดยเป็นคำสั่งที่บอกให้ตัวแปลคำสั่งหยุดกระบวนการย้อนรอยเพื่อหาตัวเลือกอื่นหลังเครื่องหมาย ! เช่น

max(X,Y,Y):- X =< Y, !.

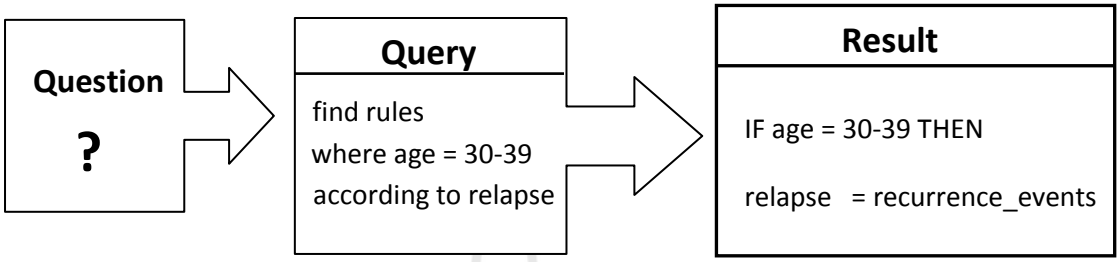
max(X,Y,X):- X>Y.

เป็นการหาค่าสูงสุดภายในเพรดิเคต max ซึ่งมี 3 อาร์กิวเมนต์ โดยจะรับค่ามาเปรียบเทียบ 2 ตัว ถ้าเข้ากฎที่ 1 จะเช็คว่า X=<Y ใช่หรือไม่ ถ้าเป็นจริง จะดำเนินการต่อไปที่เครื่องหมาย " ! " ซึ่งจะตัดการค้นหากฎต่อไป คือ เพรดิเคต max ที่ประกาศไว้ในบรรทัดต่อมา ทำให้ไม่ต้องเสียเวลาในการค้นหาคำตอบ ซึ่งจะเกินความจำเป็น เนื่องจากถ้าเข้ากฎที่ 1 แล้ว กฎที่ 2 จะไม่เป็นจริง เนื่องจาก X น้อยกว่า Y แล้ว (เงื่อนไขของกฎที่ 1) จะเป็นไปไม่ได้ว่า X มากกว่า Y (เงื่อนไขของกฎที่ 2)

2.7 หลักการของ User Preference

งานวิจัยนี้ได้เน้นการสร้างเงื่อนไขเพื่อรองรับการแสดงผลของกฎเฉพาะที่ตรงตามเงื่อนไขหรือข้อกำหนดของผู้ใช้ โดยผู้ใช้สามารถระบุให้ทำการแสดงหรือไม่แสดงบางกฎ การระบุเงื่อนไขนี้จะใช้เพรดิเคตที่สร้างขึ้นด้วยภาษาโปรล็อก คือ เพรดิเคต findRule([X]) เพื่อการแสดงกฎที่มีข้อความ X หรือ findRule([\+X]) เพื่อการแสดงกฎที่ไม่มี X หรือ findRuleOr([[X1,X2],[\+X3,\+X4]]) เพื่อการระบุหลายเงื่อนไขประกอบกัน เป็นต้น ซึ่งรายละเอียดการใช้เพรดิเคตเหล่านี้จะกล่าวถึงในบทที่ 3 โดยที่การสร้างเพรดิเคตโดยกำหนดเงื่อนไขนี้ ได้แนวคิดมาจากฟังก์ชันการทำงานของภาษา SQL คือ การสอบถามด้วยคำสั่งของภาษา SQL ที่เรียกว่า การเควียรี (query) เพื่อให้ได้ข้อมูลที่เป็นตาราง แพทเทิร์นหรือกฎของฐานข้อมูล คำสั่งการสอบถามที่นำมาใช้เป็นแนวคิดในงานวิจัยนี้ คือ คำสั่ง WHERE ซึ่งในระบบฐานข้อมูลหมายความว่า การระบุเงื่อนไขหลังคำสั่ง WHERE ในการเลือกให้แสดงข้อมูล แพทเทิร์น หรือกฎ ที่ตรงกับเงื่อนไขที่กำหนด

จากงานวิจัยที่เกี่ยวข้อง ได้พูดถึง Intelligent Data Mining คือ การทำเหมืองข้อมูลเพื่อสร้างกฎสำหรับ Trigger ไว้ในฐานข้อมูลทางการแพทย์ เพื่อสะดวกต่อการค้นหา บ่งบอกถึงปัจจัยต่าง ๆ เพื่อวินิจฉัยโรคเบื้องต้น โดยส่วนใหญ่จะแสดงผลออกมาเป็นกฎ if…then (Stühlinger, Hogl, Stoyan, and Müller, 2000) ซึ่งงานวิจัยนี้ได้นำแนวคิดนี้มาสร้างโมเดลการใช้ข้อกำหนดของ User สำหรับเควียรีกฎในเหมืองข้อมูล ดังรูปที่ 2.11



รูปที่ 2.11 โมเดลการเควียรีกฎ

การพัฒนางานวิจัยนี้ใช้ซอฟต์แวร์ ECLiPSe ซึ่งเป็นซอฟต์แวร์แบบ Open-Source ดาวน์โหลดได้จาก http://www.eclipseclp.org ซอฟต์แวร์นี้สนับสนุนการเขียนโปรแกรมเชิงตรรกะ (Logic Programming) และการเขียนโปรแกรมแบบมีเงื่อนไขบังคับ (Constraint Programming) ซึ่งจัดทำขึ้นเพื่อตอบสนองความต้องการของผู้ใช้ในด้านการจัดสรรทรัพยากร การจัดสรรเวลาที่รวดเร็ว มักจะใช้เทคนิคเงื่อนไขบังคับเชิงคณิตศาสตร์ และการค้นหาที่รวดเร็ว โดยโปรแกรม ECLiPSe มี Libraries ที่สามารถเรียกใช้โมดูลหรือฟังก์ชันการทำงานที่เป็นรูปแบบเฉพาะได้ โดยผู้ใช้ไม่ต้องเขียนฟังก์ชันขึ้นมาใหม่ ทำให้สะดวกและรวดเร็วต่อการทำงาน (Wikipedia The Free Encyclopedia, 2011a)

2.8 งานวิจัยที่เกี่ยวข้อง

ในการวิจัยเรื่องการจำแนกประเภทข้อมูลด้วยวิธีการเขียนโปรแกรมเชิงตรรกะด้วยเงื่อนไขบังคับให้ตรงตามข้อกำหนดที่ผู้ใช้ระบุ ผู้วิจัยได้ทำการศึกษาค้นคว้างานวิจัยในอดีตที่มีความเกี่ยวข้องกับการออกแบบแนวคิดการจำแนกประเภทข้อมูลด้วยวิธีการเขียนโปรแกรมเชิงตรรกะด้วยเงื่อนไขบังคับ ดังสรุปในตารางที่ 2.11 และมีรายละเอียดดังนี้

Han, Fu, Wang, Koperski และ Zaiane (1996) ได้นำเสนอเกี่ยวกับ การใช้เครื่องมือการทำเหมืองข้อมูลเพื่อประโยชน์สำหรับฐานข้อมูลเชิงสัมพันธ์ (Relational Database) เพื่อประโยชน์ในการใช้ Data Mining Query Language (DMQL) ซึ่งเป็นวิธีการเควียรีกฎจากการทำเหมืองข้อมูลในลักษณะต่าง ๆ เช่น Data Generalization, Data Classification, Mining Association Rules เพื่อตอบสนองการร้องขอรูปแบบของกฎที่ได้จากการทำเหมืองข้อมูลซึ่งมีความสัมพันธ์กับฐานข้อมูล เช่น กฎจากการจำแนกข้อมูล (Classification) ของฐานข้อมูลประวัตินักเรียนคณะ Computing Science ในประเทศแคนาดา ซึ่งจะดูว่ากฎหรือแพทเทิร์นที่มี GPA นักเรียนเป็นปัจจัยสำคัญมีกฎใดบ้าง

Stühlinger, Hogl, Stoyan และ Müller (2000) ได้นำเสนอเกี่ยวกับ การทำเหมืองข้อมูลจากการรวบรวมข้อมูลทางด้านการแพทย์ เพื่อประโยชน์ในด้านธุรกิจ คือ ให้มีความรวดเร็วในการค้นหาประวัติคนไข้ เพราะการทำเหมืองข้อมูล มีการจำแนกประเภทข้อมูล (Data Classification) ที่มีลักษณะเหมือนกันอยู่ในพวกเดียวกัน ส่วนด้านการแพทย์ คือ มีความถูกต้องในการวินิจฉัยโรคเบื้องต้นจากประวัติคนไข้ที่มีลักษณะอาการที่เป็นเฉพาะหรือช่วงอายุที่จะก่อให้เกิดโรคนั้น ซึ่งจะทำออกมาในรูปแบบของกฎ If…Then ซึ่งกฎเหล่านี้จะนำไปใช้ในการค้นหาข้อมูลในฐานข้อมูลต่อไป

Elfeky, Saad และ Fouad (2001) ได้นำเสนอเกี่ยวกับ การประยุกต์ใช้ Data Mining Query Language (DMQL) ซึ่งเป็นการเควียรีกฎที่ได้จากการทำเหมืองข้อมูล เพื่อค้นหากฎหรือแพทเทิร์นที่มีความสัมพันธ์กับฐานข้อมูลประเภทวัตถุ โดยเป็นฐานข้อมูลที่ผู้ใช้ไม่ต้องสนใจว่าในฐานข้อมูลจะมีฟิลด์หรือเรคคอร์ดเป็นอย่างไร แต่จะสนใจกับ Objects ซึ่งเป็นข้อมูลจริงที่ถูกสร้างขึ้น โดยมี Class เป็นตัวกำหนดรายละเอียดต่าง ๆ ของข้อมูล เช่น การค้นหากฎโดยแสดง ID ของนักเรียนที่จบการศึกษาแล้ว ซึ่งจะใช้ s.into.id และ s.status = "Graduate" ประกาศตัวแปรสำหรับเงื่อนไขที่ต้องการค้นหา ซึ่งเป็นวิธีการประกาศตัวแปรในรูปแบบของ Object-Oriented Language

Ogata (2001) ได้นำเสนอเกี่ยวกับ การค้นหาข้อมูลของสาเหตุการเกิดแผ่นดิวไหว โดยใช้หลักการการจำแนกประเภทข้อมูล (Data Classification) ผสมกับ Constraints ซึ่งข้อมูลการเกิดแผ่นดินไหวจะได้มาจากการรวบรวมผลการประมวลผลภาษาธรรมชาติ (Natural Language), การเรียนรู้จากเหตุการณ์หรือสิ่งที่สนใจ โดยทราบข้อมูลเพียงบางส่วน (Inductive Learning) และการสกัดข้อมูล (Information Extraction) ซึ่งการทำงานของงานวิจัยนี้ คือ จะนำข้อมูลที่บันทึกไว้เกี่ยวกับเหตุการณ์แผ่นดินไหวในรูปแบบของ Texts และมองว่าข้อมูลที่สนใจขณะนั้น ได้ถูกนำมาจำแนกประเภทข้อมูล (Data Classification) แล้ว และมาสกัดข้อความด้วยเทคโนโลยี Ontology ซึ่ง

นิยมใช้ในมาตรฐานของการออกแบบจำลองโครงสร้างของ eXtensible Markup Language (XML) ซึ่งงานวิจัยนี้จะเก็บข้อมูลไว้ในรูปแบบของ XML และนำตัวแปรที่อยู่ในแท็กของ XML มาเป็นเพรดิเคต และข้อความที่อยู่ในแท็กนั้นจะเป็นค่าของอาร์กิวเมนต์แต่ละเพรดิเคต และนำไปเขียนด้วยภาษาโปรแกรมเชิงตรรกะ (Logic Programming) และนำ Constraints มาใช้ร่วมกันเพื่อกำหนดหาสาเหตุที่สำคัญ ๆ ของเหตุการณ์แผ่นดินไหว โดยการหาค่า Support และ Confidence มาสนับสนุนแต่ละเหตุการณ์

Malerba, Appice และ Ceci (2004) ได้นำเสนอเกี่ยวกับ การประยุกต์ใช้ Data Mining Query Language (DMQL) สำหรับ Spatial Data Mining ซึ่งเป็นการทำเหมืองข้อมูลรูปแบบหนึ่งที่นิยมใช้ในงานด้านระบบสารสนเทศภูมิศาสตร์ (Geographic Information Systems) การกำหนดเงื่อนไขต่าง ๆ ของการเควียรีในงานวิจัยนี้ คือ

<Query_Statement> ::= SELECT <Object> {, <Object>}

FROM <Class> {, <Class>}

[WHERE <Conditions>]

โดย <Object> คือ สิ่งที่ต้องการให้แสดงผล

<Class> คือ Class ของ Object

<Conditions> คือ เงื่อนไขที่เกี่ยวข้องกับการเควียรี Object นั้น

ตารางที่ 2.11 สรุปเปรียบเทียบงานวิจัยที่เกี่ยวข้องกับการจำแนกข้อมูลในรูปแบบต่าง ๆ เพื่อสร้างกฎและเควียรีกฎที่ตรงกับเงื่อนไขที่กำหนดไว้

บทความวิจัยที่เกี่ยวข้องประกอบด้วย 1 = Han, Fu, Wang, Koperski และ Zaiane (1996), 2 = Stühlinger, Hogl, Stoyan และ Müller (2000), 3 = Elfeky, Saad และ Fouad (2001), 4 = Ogata (2001), 5 = Malerba, Appice และ Ceci (2004), * = การจำแนกข้อมูลเพื่อการแสดงผลตามข้อกำหนดของผู้ใช้ (งานวิจัยของวิทยานิพนธ์ฉบับนี้)

กระบวนการทำงาน	งานวิจัยที่เกี่ยวข้อง					
	1	2	3	4	5	*
ฐานความรู้ของระบบ						
ประเภทของฐานความรู้						
ฐานความรู้ที่ได้จาก Relational Database	✓	✓		✓		
ฐานความรู้ที่ได้จาก Knowledge Base						✓
ฐานความรู้ที่ได้จาก Object-Oriented Database			✓		✓	
การพัฒนา						
พัฒนาโดยใช้รูปแบบของภาษาโปรล็อก				✓		✓
พัฒนาโดยใช้รูปแบบของภาษา XML				✓		
พัฒนาโดยการทำเหมืองข้อมูล (Data Mining)	✓	✓	✓	✓	✓	✓
พัฒนาโดยการจำแนกข้อมูล (Classification)				✓		✓
ระบบแสดงผล						
รูปแบบของกฎ (Rules)						
if..then rules		✓				✓
ตีความหมายจากกฎ เป็นข้อมูล	✓		✓		✓	
ไม่แสดงกฎ				✓		
เทคนิคการเควียรีกฎ						
เขียนเพรดิเคตด้วยภาษาโปรล็อก						✓
SQL Query	✓	✓	✓		✓	
ไม่มีเทคนิคการเควียรีกฎ				✓		
เทคนิคการสร้างเงื่อนไข						
พัฒนาการสร้างเงื่อนไขด้วยภาษาโปรล็อกในโปรแกรม				✓		✓
พัฒนาการสร้างเงื่อนไขด้วยเทคนิค DMQL ในฐานข้อมูล	✓	✓				
พัฒนาการสร้างเงื่อนไขด้วยเทคนิค ODMQL ในฐานข้อมูล			✓		✓	
การประยุกต์ใช้ระบบ						
วิจัยเพื่อทดสอบความถูกต้อง		✓	✓	✓	✓	✓
วิจัยเพื่อประยุกต์ใช้กับข้อมูลจริง	✓	✓	✓	✓	✓	✓

บทที่ 3
วิธีดำเนินการวิจัย

งานวิจัยการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ มีจุดมุ่งหมายเพื่อพัฒนาการค้นหาและแสดงผลของกฎที่ได้จากการจำแนกข้อมูลตามเงื่อนไขที่กำหนด โดยใช้เทคนิคการจำแนกข้อมูลด้วยอัลกอริทึม ID3 และพัฒนาโปรแกรมตามอัลกอริทึม ID3 ด้วยภาษาโปรล็อก รวมถึงใช้ภาษาโปรล็อกในการพัฒนาคำสั่งการค้นหากฎเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ รายละเอียดเนื้อหาในบทนี้ประกอบด้วย ขั้นตอนการวิจัย ในหัวข้อ 3.1 โปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ อธิบายในหัวข้อ 3.2 และการทำงานของโปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ แสดงไว้ในหัวข้อ 3.3

3.1 ขั้นตอนการวิจัย

แนวทางการวิจัยของงานวิจัยนี้ประกอบด้วยการออกแบบและพัฒนาขั้นตอนวิธีการเขียนโปรแกรมเชิงตรรกะและคัดเลือกกฎด้วยข้อกำหนดของผู้ใช้ โดยมีรายละเอียดขั้นตอนวิธีดังต่อไปนี้

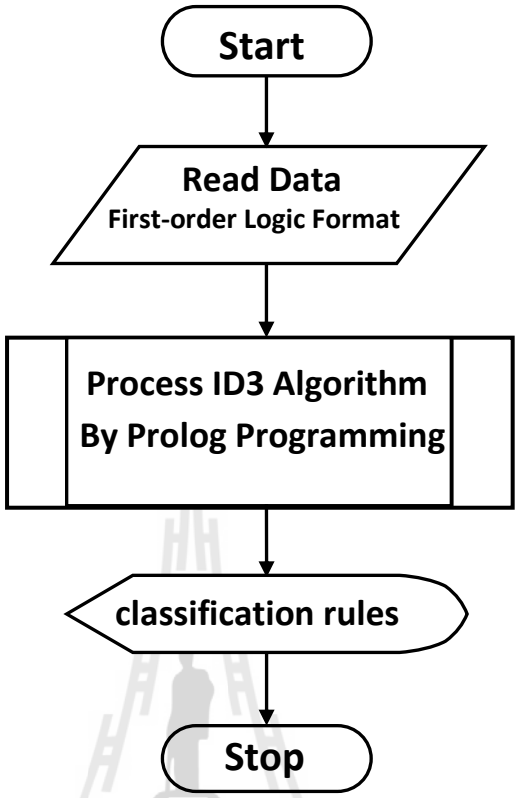
1. ศึกษาและรวบรวมงานวิจัยที่เกี่ยวข้อง
2. ศึกษาทฤษฎีและขั้นตอนการทำงานของอัลกอริทึม ID3
3. ศึกษาขั้นตอนวิธีและกระบวนการเขียนโปรแกรมด้วยภาษาโปรล็อก
4. ออกแบบโครงสร้างข้อมูลทดสอบที่เหมาะสมกับรูปแบบของการทำงานของอัลกอริทึม ID3 ด้วยภาษาโปรล็อก
5. ออกแบบอัลกอริทึมสำหรับการจำแนกข้อมูลตามขั้นตอนวิธี ID3 ด้วยภาษาโปรล็อก
6. ออกแบบอัลกอริทึมสำหรับการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ เพื่อค้นหาหรือแสดงบางกฎ
7. พัฒนาโปรแกรมที่ได้ออกแบบอัลกอริทึมไว้และทดสอบโปรแกรมกับข้อมูลที่ออกแบบไว้
8. วิเคราะห์และสรุปผลการวิจัย

3.2 โปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้

เนื้อหาในส่วนนี้จะกล่าวถึงแผนผังการทำงานการจำแนกข้อมูลและการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ด้วยวิธีการเควียรีในรูปแบบต่าง ๆ ที่ผู้วิจัยได้ออกแบบไว้ เพื่อพัฒนาโปรแกรมโดยใช้ภาษาโปรล็อกสำหรับจำแนกข้อมูล และคัดเลือกบางกฎที่ตรงกับคำสั่งการเควียรีของผู้ใช้

3.2.1 การจำแนกข้อมูลด้วยอัลกอริทึม ID3 และ ID3UP

การจำแนกข้อมูลด้วยอัลกอริทึม ID3 (Quinlan, 1986) เป็นกระบวนการสร้างโมเดลจัดการข้อมูลให้อยู่ในกลุ่มที่กำหนด โดยจะจัดการจำแนกข้อมูลเป็นต้นไม้ตัดสินใจ โดยอัลกอริทึม ID3 เป็นต้นไม้ตัดสินใจที่ทำงานจากบนลงล่าง โดยในการเลือกว่าลักษณะใดดีที่สุดนั้นดูจากค่าของลักษณะ ซึ่งเรียกว่า การเพิ่มคุณค่าข้อมูล (Information Gain) ซึ่งต้นไม้ตัดสินใจที่ใช้ในงานวิจัยนี้จะไม่มีการตัดกิ่ง (Pruning) เพื่อสะดวกต่อการใช้ข้อมูลทั้งหมดสำหรับตอบคำถามที่ตรงต่อความต้องการของผู้ใช้ เนื่องจากบางอัลกอริทึมที่มีการตัดกิ่ง คำตอบบางส่วนอาจจะถูกตัดออกไป ซึ่งคำตอบนั้นอาจจะมีประโยชน์ต่อผู้ใช้หรือมีส่วนที่ผู้ใช้สนใจ จากแผนผังการทำงานดังรูปที่ 3.1 และรายละเอียดขั้นตอนอัลกอริทึม ID3 ในรูปที่ 3.2 ได้อธิบายวิธีการทำงานของอัลกอริทึม ID3 เมื่อข้อมูลที่จะทำการจำแนกประเภทต้องอยู่ในรูปแบบที่กำหนดโดยให้เป็นข้อความตรรกศาสตร์อันดับหนึ่ง และกระบวนการอัลกอริทึม ID3 จะหาค่า Entropy และค่า Gain เพื่อทำการสร้างเซตของกฎเป็นเอาท์พุตต่อไป โดยที่โปรแกรมของงานวิจัยนี้ได้ออกแบบและพัฒนาโดยเลือกใช้ Decision Tree Induction เป็นเทคนิคหลัก ซึ่งเป็นหนึ่งในการดำเนินงานของอัลกอริทึม ID3 และต่อยอดการทำงานของอัลกอริทึมด้วยฟังก์ชัน findRule และ findRuleOr เพื่อสะดวกต่อการค้นหาคำตอบ โดยอัลกอริทึมที่สร้างขึ้นมาใหม่นี้ เรียกชื่อว่า อัลกอริทึม ID3UP (Induction of Decision Tree based on User Preferences) แสดงอัลกอริทึม ID3UP ได้ดังรูปที่ 3.3



รูปที่ 3.1 ผังงานการจำแนกข้อมูลด้วยอัลกอริทึม ID3 และประมวลผลด้วยโปรแกรมโปรล็อก

มหาวิทยาลัยเทคโนโลยีสุรนารี

Algorithm 1 ID3 Algorithm

Input: data

Output: a decision tree with node and edge structures

(1) Read data

(2) Traverse tree from a root node to each leaf node

(2.1) node = DecisionTreeNode(examples)

handle target attributes with arbitrary labels

(2.2) dictionary = summarizeExamples(examples, targetAttribute)

for key in dictionary:

if dictionary[key] == total number of examples

node.label = key

return node

(2.3) if attributes is empty or number of examples < minimum allowed per branch:

node.label = most common value in examples

return node

(2.4) bestA = the attribute with the most information gain

node.decision = bestA

for each possible value v of bestA :

subset = the subset of examples that have value v for bestA

if subset is not empty:

node.addBranch(id3(subset, targetAttribute, attributes-bestA))

return node

รูปที่ 3.2 รายละเอียดขั้นตอนอัลกอริทึม ID3 (Swarthmore Department of Computer Science, 2011)

Algorithm 2 ID3UP Algorithm

Input: training data in first-order logic format

Output: constrained rules

(1) Get training data in first-order logic format

(2) Traverse tree from a root node to each leaf node

(2.1) Create a root node and branches

Calculate I_{all}(node,branch)

(I_{all}(node,branch) = an entropy value of each attributes)

Sum total of I_{all}(node,branch) values

Choose sum total of entropy values which are minimum

Return a root node and branch

(2.2) Create leaf node and branches

Calculate I_{re}(node,branch)

(I_{re}(node,branch) = an entropy value of attributes remain which relate before attribute)

Sum total of I_{re}(node,branch) values

Choose sum total of entropy values which are minimum

Check class of goal attribute

Count the amount of data for each class

Save to List

(List = $[a_1,a_2,\ldots,a_n]$)

Check values in List must counting number is one value

(List = [1,0] or [1,0,0])

รูปที่ 3.3 รายละเอียดขั้นตอนอัลกอริทึม ID3UP

if List is true

Stop create leaf node and branches

Save all decision tree in lists

else List is fail

Return to create leaf node and branches

(3) Convert decision tree to rules set

Split values in each list

(V = value in each list) and

(V = node-branch)

if V is first value

Write " if " next to V

Return to split values in each list

else if V is remainder and V is not last value

Write V

Return to split values in each list

else

Write " then " next to V

Save in memory

(4) Get preference from user

(findRule([Argument]) predicate = preference of user) or

(findRuleOr([[Argument],[Argument],…,[Argument]] predicate)

= preference of user)

Write constrained rules according to condition of preference

รูปที่ 3.3 รายละเอียดขั้นตอนอัลกอริทึม ID3UP (ต่อ)

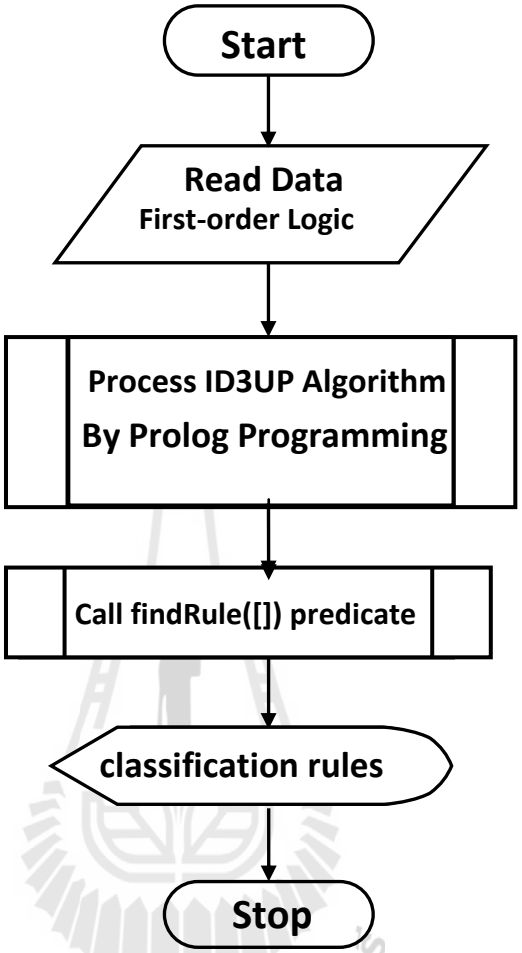
3.2.2 การแสดงผลการจำแนกข้อมูลเมื่อใส่เควียรีด้วยเพรดิเคตตรรกศาสตร์

การจำแนกข้อมูลด้วยอัลกอริทึม ID3UP เมื่อจบขั้นตอนการแปลงโครงสร้างต้นไม้เป็นเซตของกฎการตัดสินใจ จะได้ผลลัพธ์เป็นกฎออกมาทั้งหมด ซึ่งบางกฎอาจจะไม่เป็นประโยชน์ต่อผู้ใช้ ทำให้ต้องกำหนดความต้องการของผู้ใช้ เพื่อสะดวกต่อการค้นหา และลดจำนวนการแสดงผลของกฎที่อาจจะมีจำนวนมากเกินไป โดยอัลกอริทึม ID3UP ที่ออกแบบไว้ สามารถประมวลผลให้ได้ Rules มาทั้งหมด โดยไม่มีการตัดกิ่งและถูกเก็บไว้เป็นโมเดล เพื่อเอื้อประโยชน์ต่อการแสดงผลตามความต้องการของผู้ใช้ เมื่อต้องการค้นหาหรือไม่ค้นหาบางแอททริบิวต์ (ระบุด้วยรูปแบบ “Attribute”) หรือบางค่าของแอททริบิวต์ (ระบุด้วยรูปแบบ “Attribute-Value”) ที่อยู่ในส่วนลึกมากภายในต้นไม้ตัดสินใจ

(1) การเควียรีด้วยเพรดิเคตเพื่อแสดงกฎทั้งหมด

การแสดงกฎทั้งหมดที่ได้จากการจำแนกข้อมูล สามารถเลือกใช้เพรดิเคต **findRule([])** สำหรับเควียรีให้กับโปรแกรม ซึ่ง findRule เป็นชื่อเพรดิเคตสำหรับเรียกส่วนของโปรแกรมที่ต้องการแสดงผลของกฎที่ได้จากการจำแนกข้อมูลและอาร์กิวเมนต์ในเพรดิเคต findRule มี 1 อาร์กิวเมนต์ คือ ลิสต์ว่าง หรือ [] หมายความว่า ไม่มีการกำหนดเงื่อนไขในการแสดงผลของกฎ ทำให้กฎที่จะแสดงผลนั้นจะแสดงผลออกมาทั้งหมดทางส่วนแสดงผลของโปรแกรม ขั้นตอนการเควียรีกฎทั้งหมด แสดงดังรูปที่ 3.4

มหาวิทยาลัยเทคโนโลยีสุรนารี



รูปที่ 3.4 ขั้นตอนการเควียรีกฎทั้งหมดด้วยเพรดิเคต findRule([])

(2) การเควียรีด้วยเพรดิเคตเพื่อแสดงกฎบางส่วนตามเงื่อนไขของผู้ใช้และสามารถกำหนดกฎให้แสดงภายใต้การดำเนินการของ AND

การแสดงกฎบางส่วนตามเงื่อนไขของผู้ใช้และสามารถกำหนดกฎให้แสดงภายใต้การดำเนินการของโอเปอเรเตอร์ AND คือ เมื่อผู้ใช้ต้องการค้นหากฎบางส่วนตามเงื่อนไขที่กำหนด สามารถเลือกใช้เพรดิเคต **findRule([X])** หรือ **findRule([\+X])** หรือ **findRule(MultiX)** สำหรับเควียรีผลลัพธ์ในเซตของกฎการจำแนกข้อมูล ซึ่งการค้นหาใช้เพรดิเคตเดียวกับการค้นหากฎทั้งหมด แต่แตกต่างกันตรงที่อาร์กิวเมนต์ในเพรดิเคต findRule ที่มี 1 อาร์กิวเมนต์ แทนที่จะเป็น 0 อาร์กิวเมนต์ ขั้นตอนการเควียรีกฎบางส่วนตามเงื่อนไขของผู้ใช้และสามารถกำหนดกฎให้แสดง

ภายใต้การดำเนินการของ AND แสดงดังรูปที่ 3.5 และแสดงขั้นตอนในรูปแบบของซูโดโค้ดได้ดังรูปที่ 3.6

อธิบายการใช้งานเพรดิเคต findRule([Argument]) โดยกำหนดเงื่อนไขในส่วนอาร์กิวเมนต์ด้วยรูปแบบต่าง ๆ ได้ดังนี้

- **findRule([X])** หมายถึง การกำหนด X ให้เป็นเฉพาะชื่อของแอททริบิวต์หรือ ชื่อและค่าของแอททริบิวต์ (Attribute-Value) ที่ต้องการให้แสดงผลอยู่ใน Rules เช่น

findRule([outlook]) คือ แสดงกฎที่**มี**แอททริบิวต์ “outlook”

findRule([outlook - sunny] คือ แสดงกฎที่**มี**แอททริบิวต์ “outlook” และค่าของแอททริบิวต์นี้เป็น “sunny”

- **findRule([\+X])** เครื่องหมาย “\+” เป็นตัวแทนของคำสั่ง NOT ซึ่งหมายถึง กำหนดให้ \+X คือ ชื่อแอททริบิวต์ หรือ ชื่อแอททริบิวต์พร้อมค่าที่จะไม่แสดงผลอยู่ในกฎ เช่น

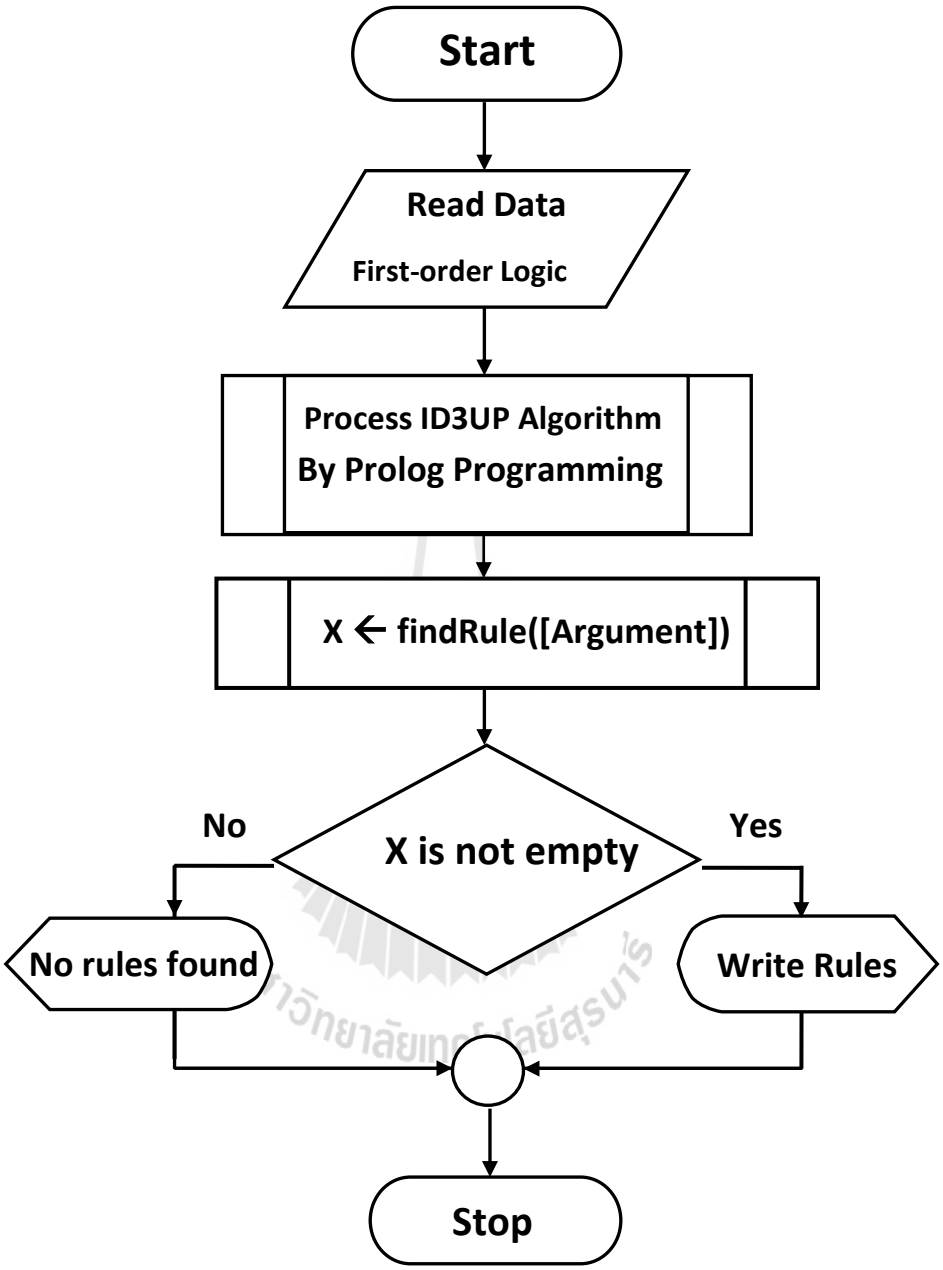
findRule([\+outlook]) คือ แสดงกฎที่**ไม่**แอททริบิวต์ “outlook”

findRule([\+outlook - sunny] คือ แสดงกฎที่**ไม่มี**แอททริบิวต์ “outlook” ที่ปรากฏค่าเป็น “sunny”

- **findRule([MultiX])** หมายถึง การกำหนดให้มีหลายเงื่อนไขโดยใช้ตัวแปร MultiX ทำหน้าที่รวบรวมเงื่อนไขเหล่านั้นให้อยู่ในโครงสร้างข้อมูลลิสต์ เงื่อนไขภายในลิสต์ใช้เครื่องหมาย “ , ” เป็นการดำเนินการของ **AND** และสามารถใช้เครื่องหมาย \+ ที่เป็นตัวแทนของ NOT ได้ โดยสามารถใส่ได้ทั้งชื่อแอททริบิวต์ หรือชื่อแอททริบิวต์พร้อมค่า ตัวอย่างของการเควียรีกรณีระบุหลายเงื่อนไข แสดงได้ดังนี้

findRule([outlook , humidity - high]) คือ แสดงกฎที่มีแอททริบิวต์ “outlook” **และ** มีแอททริบิวต์ “humidity” ที่มีค่าเป็น “high”

findRule([outlook , \+humidity - high]) คือ แสดงกฎที่มีแอททริบิวต์ “outlook” **และ** ไม่มีแอททริบิวต์ “humidity” ที่มีค่าเป็น “high”



รูปที่ 3.5 ผังงานการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP เมื่อมีการเควียรีกฎบางส่วนตามเงื่อนไขของผู้ใช้ ภายใต้การดำเนินการของ AND

Algorithm 3 findRule([Argument]) predicate

Input: condition for findRule predicate

Output: a set of rules

```
Call findRule( [Argument] ) predicate in Display GUI
    if Argument = []
        Write all rules
    else if Argument = X
        Check condition of X
        ( X = a set of attribute name ) or
        ( X = a set of attribute name and value )
        Write all rules that satisfy condition of X
    else if Argument = \+X
        Check condition of \+X
        /* NOT condition : exclude rules that contain attribute name or
        attribute name and value from result set */
        ( \+X = a set of attribute name ) or
        ( \+X = a set of attribute name and value )
        Write all rules that satisfy condition of \+X
    else if Argument = MultiX
        Check condition of  MultiX
        ( MultiX = sets of attribute names or attribute names and values
          under control AND condition )
        Write all rules that satisfy condition of MultiX
else    Result set is empty
```

รูปที่ 3.6 รายละเอียดขั้นตอนอัลกอริทึม findRule([Argument]) predicate

(3) การเควียรีด้วยเพรดิเคตเพื่อแสดงกฎบางส่วนตามเงื่อนไขของผู้ใช้ภายใต้การดำเนินการของ OR

การแสดงกฎบางส่วนตามเงื่อนไขของผู้ใช้ ภายใต้การดำเนินการของ OR คือ การกำหนดการค้นหากฎภายใต้หลายเงื่อนไขให้แสดงผลรวมกัน โดยจะแสดงผลตามเงื่อนไขที่กำหนดภายในลิสต์ย่อยหลายลิสต์ที่อยู่ในลิสต์ใหญ่ โดยกำหนดการใช้งานสำหรับการเควียรี คือ เพรดิเคต **findRuleOr([[Argument$_1$], [Argument$_2$], …, [Argument$_N$]])** โดยกฎที่ตรงตามเงื่อนไขภายในลิสต์ย่อยทุกลิสต์จะออกมาทั้งหมด ซึ่งตรงกับการดำเนินการของ OR คือ **การรวมกฎ** ตามเงื่อนไขที่เป็นไปได้ ซึ่งภายในลิสต์ย่อยสามารถใช้การดำเนินการของ AND ผสมอยู่ได้ และลิสต์ย่อยสามารถมีไม่จำกัดจำนวน เพื่อความหลากหลายในการเควียรีกฎ ขั้นตอนการเควียรีกฎบางส่วนตามเงื่อนไขภายใต้การดำเนินการของ OR แสดงดังรูปที่ 3.7 และแสดงซูโดโค้ดได้ดังรูปที่ 3.8

อธิบายการใช้งานเพรดิเคต findRuleOr([[Argument$_1$], [Argument$_2$], …, [Argument$_N$]]) โดยที่ลิสต์ย่อยของ Argument มีจำนวนได้ไม่จำกัด และรูปแบบของ Argument สามารถใส่เงื่อนไขเดียวหรือหลายเงื่อนไข ยกตัวอย่างเช่น

- **findRuleOr([[X$_1$], [X$_2$], …, [X$_N$]])** เป็นการกำหนดค่าของเพรดิเคต findRuleOr ด้วยลิสต์ย่อยซึ่งไม่จำกัดจำนวน โดยมีเงื่อนไขภายในลิสต์ย่อย 1 เงื่อนไข คือ กำหนด $X_1, X_2, \ldots, X_N$ ให้เป็นชื่อของแอททริบิวต์หรือชื่อแอททริบิวต์พร้อมค่าที่จะแสดงผลอยู่ในกฎ ซึ่งกฎที่ตรงตามเงื่อนไขของลิสต์ย่อยทั้งหมดนั้น จะ**รวม**กัน เพื่อการแสดงผลในแต่ละครั้งที่มีการเควียรี เช่น

findRuleOr([[outlook], [class - yes]]) คือ แสดงกฎที่มีแอททริบิวต์ “outlook” **รวม**กับกฎที่มีแอททริบิวต์ “class” และค่าของแอททริบิวต์เป็น “yes”

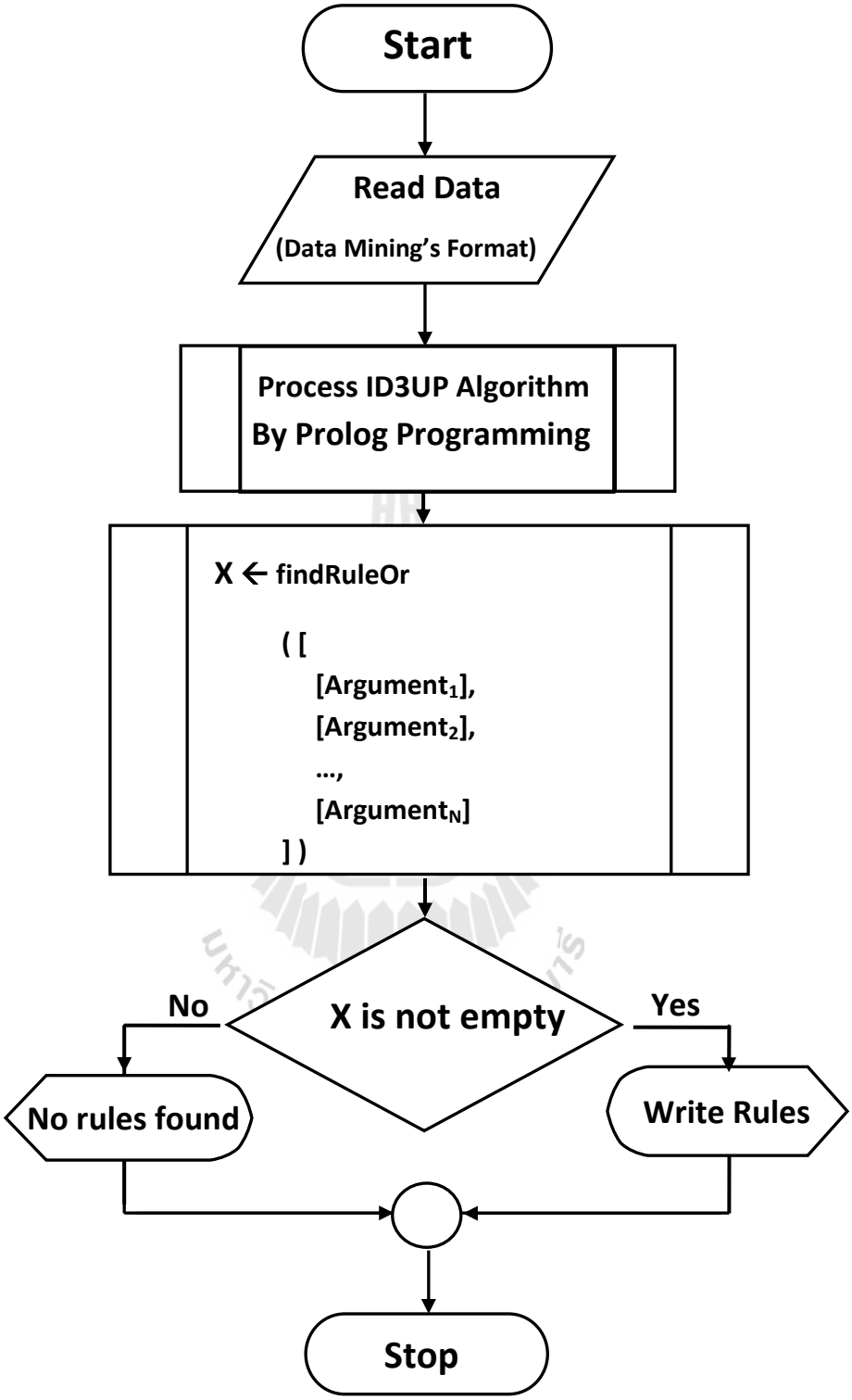
- **findRuleOr([[\+X$_1$], [\+X$_2$], …, [\+X$_N$]])** เป็นการกำหนดค่าของเพรดิเคต findRuleOr ด้วยลิสต์ย่อยจำนวนไม่จำกัด โดยมีเงื่อนไขภายในลิสต์ย่อย 1 เงื่อนไข โดยที่เครื่องหมาย “\+” คือ คำสั่ง NOT ซึ่งกำหนดให้ $\+X_1, \+X_2, \ldots, \+X_N$ คือ ชื่อแอททริบิวต์หรือ ชื่อแอททริบิวต์พร้อมค่าที่จะไม่แสดงผลอยู่ในกฎ ซึ่งกฎที่ตรงตามเงื่อนไขของลิสต์ทั้งหมดจะ**รวม**กันเพื่อการแสดงผลในแต่ละครั้งที่มีการเควียรี เช่น

findRuleOr([[\+outlook], [\+humidity], [\+class - yes]]) คือ แสดงกฎที่ไม่มีแอททริบิวต์ “outlook” **<u>รวม</u>**กับกฎที่ไม่มีแอททริบิวต์ “humidity” **<u>รวม</u>**กับกฎที่ไม่มีแอททริบิวต์ “class” ที่มีค่าเป็น “yes”

- **findRuleOr([[MultiX$_1$], [MultiX$_2$], …, [MultiX$_N$]])** หมายถึง การกำหนดค่าของ MultiX$_1$, MultiX$_2$, …, MultiX$_N$ ให้มีหลายเงื่อนไขภายในลิสต์ย่อยที่ไม่จำกัดจำนวน โดยใช้เครื่องหมาย “ , ” ซึ่งเป็นการดำเนินการของ AND คั่นแต่ละเงื่อนไขภายในลิสต์ย่อยแต่ละลิสต์ และเงื่อนไขนั้นสามารถใช้เครื่องหมาย \+ ที่เป็นตัวแทนการดำเนินการของ NOT ได้ ซึ่งแต่ละเงื่อนไขจะกำหนดเป็นชื่อแอททริบิวต์หรือชื่อแอททริบิวต์พร้อมค่า และลิสต์ย่อยแต่ละลิสต์ จะคั่นด้วยเครื่องหมาย “,” ซึ่งเป็นตัวแทนการดำเนินการของ OR โดยกฎที่ตรงตามเงื่อนไขของลิสต์ย่อยแต่ละลิสต์จะ**<u>รวม</u>**กัน เพื่อการแสดงผลในแต่ละครั้งที่มีการเควียรี เช่น

findRuleOr([[outlook , humitidy, \+class - yes] , [\+outlook - sunny], [humidity]]) คือ แสดงกฎที่มีแอททริบิวต์ “outlook” และแอททริบิวต์ humidity และไม่มีแอททริบิวต์ “class” ที่มีค่าเป็น “yes” **<u>รวม</u>**กับกฎที่ไม่มีแอททริบิวต์ “outlook” ที่มีค่าเป็น “sunny” และสุดท้าย**<u>รวม</u>**กับกฎที่มีแอททริบิวต์ “humidity”

มหาวิทยาลัยเทคโนโลยีสุรนารี



รูปที่ 3.7 ผังงานการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP เมื่อมีการเควียรีกฎบางส่วนตามเงื่อนไขของผู้ใช้ ภายใต้การดำเนินการของ OR

Algorithm 4 findRuleOr([[Argument$_1$], [Argument$_2$], …, [Argument$_N$]]) predicate

Input: condition for findRuleOr predicate

Output: a set of rules

Call findRuleOr([[Argument$_1$], [Argument$_2$], …, [Argument$_N$]]) predicate

in Display GUI

if Argument = [X_1], [X_2], …, [X_N]

Check conditions of [X1] OR [X2] OR [X_N] (OR condition)

[X_1], [X_2], …, [X_N] = a set of attribute names ;

[X_1], [X_2], …, [X_N] = a set of attribute names and values

Union setsof rules that satisfy conditions

Write all rules

else if Argument = [\+X_1], [\+X_2], …, [\+X_N]

Check conditions of [\+X1] OR [\+X2] OR [\+X_N] (OR condition)

/* " \+ " -> NOT condition : exclude rules that contain set of attribute names or a set of attribute names with values from result set */

[\+X1], [\+X2], …, [\+X_N] = a set of attribute names ;

[\+X1], [\+X2] , …, [\+X_N] = a set of attribute names and values

Union setsof rules that satisfy conditions

Write all rules

else if Argument = [MultiX_1], [MultiX_2], …, [MultiX_N]

Check conditions of [MultiX_1], OR [MultiX_2] OR [MultiX_N] (OR condition)

[MultiX_1], [MultiX_2], …, [MultiX_N] = a set of attribute names ;

[MultiX_1], [MultiX_2], …, [MultiX_N] = a set of attribute names and values

Union setsof rules that satisfy conditions

Write all rules

Else

Result set is empty

รูปที่ 3.8 รายละเอียดขั้นตอนอัลกอริทึม findRuleOr([Argument]) predicate

3.3 การทำงานของโปรแกรมสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้

ขั้นตอนการทำงานของโปรแกรมสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ (ID3UP) ที่ได้พัฒนาขึ้น สามารถอธิบายได้ดังต่อไปนี้

3.3.1 การเตรียมข้อมูลสำหรับโปรแกรม ID3UP

ข้อมูลตัวอย่างที่ใช้ประกอบการอธิบายเป็นข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ แสดงได้ดังตารางที่ 3.1

ตารางที่ 3.1 ตารางแสดงข้อมูลประกอบการตัดสินใจเล่นกอล์ฟ (Quinlan, 1992)

No.	Attributes				Class
	Outlook	Temperature	Humidity	Windy	
1	sunny	Hot	High	False	N
2	sunny	Hot	High	True	N
3	overcast	Hot	High	False	P
4	rain	Mild	High	False	P
5	rain	Cool	Normal	False	P
6	rain	Cool	Normal	True	N
7	overcast	Cool	Normal	True	P
8	sunny	Mild	High	False	N
9	sunny	Cool	Normal	False	P
10	rain	Mild	Normal	False	P
11	sunny	Mild	Normal	True	P
12	overcast	Mild	High	True	P
13	overcast	Hot	Normal	False	P
14	rain	Mild	High	True	N

เปลี่ยนรูปแบบของข้อมูลเดิมที่อยู่ในลักษณะของตารางข้อความ ให้เป็นรูปแบบข้อความของตรรกศาสตร์อันดับหนึ่งที่สามารถทำการจำแนกข้อมูลได้ด้วยโปรแกรม ID3UP ข้อมูลที่แปลงรูปแบบแล้วแสดงได้ดังรูปที่ 3.9

```
data([
[outlook-sunny,temerature-hot,humidity-high,windy-false,play-no],
[outlook-sunny,temerature-hot,humidity-high,windy-true,play-no],
[outlook-cloudy,temerature-hot,humidity-high,windy-false,play-yes],
[outlook-rainy,temerature-mild,humidity-high,windy-false,play-yes],
[outlook-rainy,temerature-cool,humidity-normal,windy-false,play-yes],
[outlook-rainy,temerature-cool,humidity-normal,windy-true,play-no],
[outlook-cloudy,temerature-cool,humidity-normal,windy-true,play-yes],
[outlook-sunny,temerature-mild,humidity-high,windy-false,play-no],
[outlook-sunny,temerature-cool,humidity-normal,windy-false,play-yes],
[outlook-rainy,temerature-mild,humidity-normal,windy-false,play-yes],
[outlook-sunny,temerature-mild,humidity-normal,windy-true,play-yes],
[outlook-cloudy,temerature-mild,humidity-high,windy-true,play-yes],
[outlook-cloudy,temerature-hot,humidity-normal,windy-false,play-yes],
[outlook-rainy,temerature-mild,humidity-high,windy-true,play-no]]
```

รูปที่ 3.9 ข้อมูลที่อยู่ในรูปแบบข้อความตรรกศาสตร์อันดับหนึ่งพร้อมทำการจำแนกข้อมูล

เมื่อเตรียมในส่วนข้อมูลแล้ว จะต้องเพิ่มส่วนนิยามข้อมูล เพื่อบอกขอบเขตค่าที่เป็นไปได้ทั้งหมดของแต่ละแอททริบิวต์ ข้อมูลแต่ละตัวจะต้องนิยามให้มีแอททริบิวต์ ซึ่งได้ดัดแปลงข้อมูลให้มีลักษณะดังรูปที่ 3.10 โดยรูปแบบ คือแอททริบิวต์ที่เป็นเป้าหมายในการจำแนกข้อมูล จะต้องถูกกำหนดไว้เป็นลิสต์แรก และรูปแบบของการกำหนด คือ ชื่อ *แอททริบิวต์ – ค่าของแอททริบิวต์*

```
+[
        [play-yes,play-no],
        [outlook-sunny,outlook-cloudy,outlook-rainy],
        [temerature-hot,temerature-mild,temerature-cool],
        [humidity-high,humidity-normal],
        [windy-true,windy-false]
]).
```

รูปที่ 3.10 รูปแบบของแอททริบิวต์ของข้อมูล

3.3.2 การเรียกใช้โปรแกรม

การเริ่มต้นการทำงานของโปรแกรม เริ่มจากการเรียกเพรดิเคต run เพื่อให้โปรแกรมจำแนกข้อมูลด้วยอัลกอริทึม ID3UP ซึ่งโปรแกรมจะเริ่มต้นจากการอ่านข้อมูลที่อยู่ในรูปแบบพร้อมทำเหมืองข้อมูล ทำการประมวลผลด้วยอัลกอริทึม ID3UP ที่ใช้ค่า Gain ในการคัดเลือกแอททริบิวต์ เพื่อสร้างเป็นโครงสร้างต้นไม้ และผลที่ได้ออกมาจากรายงานแอททริบิวต์พร้อมค่าของแอททริบิวต์ ที่เลือกในแต่ละโหนด และ Rules ที่ได้จากการจำแนกข้อมูล ดังรูปที่ 3.11

Choose : [outlook - sunny, outlook - cloudy, outlook - rainy]

At : outlook – sunny

Choose : [humidity - high, humidity - normal]

At : humidity – high

Ans: play – no

rule : if outlook - sunny humidity - high then play – no

At : humidity – normal

Ans: play – yes

rule :if outlook - sunny humidity - normal then play - yes

At : outlook – cloudy

Ans: play – yes

rule :if outlook - cloudy then play – yes

At : outlook – rainy

Choose : [windy - true, windy - false]

At : windy – true

Ans: play – no

rule :if outlook - rainy windy - true then play - no

At : windy – false

Ans: play – yes

rule :if outlook - rainy windy - false then play - yes

รูปที่ 3.11 ผลที่ได้หลังจากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP

3.3.3 การแสดงผลของกฎทั้งหมดด้วยเพรดิเคต findRule([])

เมื่อมีการเควียรีด้วยเพรดิเคต findRule([]) จากคำสั่งที่มีอยู่ในเพรดิเคตนี้ กำหนดให้ค้นหากฎตามเงื่อนไขที่กำหนดภายในลิสต์ หลังจากจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แล้ว ซึ่งกฎที่ได้จะถูกบันทึกไว้ในหน่วยความจำ เมื่อเรียกเพรดิเคตสำหรับการค้นหา กฎจะแสดงผลออกมาตามเงื่อนไขที่กำหนด ซึ่งการค้นหากฎทั้งหมด จะไม่ใส่เงื่อนไขใด ๆ กฎทั้งหมดที่ได้จากการเควียรีด้วยเพรดิเคต findRule([]) แสดงดังรูปที่ 3.12

```
if outlook - sunny  humidity - high then play - no

if outlook - sunny  humidity - normal then play - yes

if outlook - cloudy then play - yes

if outlook - rainy  windy - true then play - no

if outlook - rainy  windy - false then play - yes
```

รูปที่ 3.12 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎทั้งหมดเมื่อเควียรีด้วยเพรดิเคต findRule([])

3.3.4 การแสดงผลของกฎตามเงื่อนไขของผู้ใช้ด้วยเพรดิเคต findRule([X]) หรือ findRule([\+X]) หรือ findRule([X,\+X])

เมื่อมีการเควียรีด้วยเพรดิเคต findRule([]) จากคำสั่งที่มีอยู่ในเพรดิเคตนี้ คือ คำสั่งเดียวกันกับการเรียกเพรดิเคตที่ค้นหากฎทั้งหมด แต่แตกต่างกันตรงที่กำหนดเงื่อนไขภายในลิสต์ ซึ่งเป็นเงื่อนไขของชื่อแอททริบิวต์หรือแอททริบิวต์พร้อมค่า ที่จะปรากฏในกฎที่จะแสดงผลออกมา ซึ่งเงื่อนไขจะเก็บไว้ในตัวแปรหนึ่ง เพื่อใช้เป็นตัวแปรในการค้นหา ตัวอย่างเช่น

- findRule([humidity]) จะได้ผลลัพธ์ดังรูปที่ 3.13

```
if outlook - sunny  humidity - high then play – no
if outlook - sunny  humidity - normal then play - yes
```

รูปที่ 3.13 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข เมื่อเควียรีด้วยเพรดิเคต findRule([X])

- findRule([\+humidity-high]) จะได้ผลลัพธ์ดังรูปที่ 3.14

```
if outlook - sunny  humidity - normal then play – yes
if outlook - cloudy then play – yes
if outlook - rainy  windy - true then play – no
if outlook - rainy  windy - false then play - yes
```

รูปที่ 3.14 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข เมื่อเควียรีด้วยเพรดิเคต findRule([\+X])

- findRule([play - no,\+outlook - sunny]) จะได้ผลลัพธ์ดังรูปที่ 3.15

```
if outlook - rainy  windy - true then play - no
```

รูปที่ 3.15 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข เมื่อเควียรีด้วยเพรดิเคต findRule([X,\+X])

3.3.5 การแสดงผลของกฎตามเงื่อนไขของผู้ใช้ด้วยเพรดิเคต findRuleOr([[X],[X]]) หรือ findRuleOr([[\+X],[\+X]]) หรือ findRuleOr([[X,\+X],[\+X,X]])

เพรดิเคต findRuleOr เป็นเพรดิเคตที่ใช้คำสั่งของเพรดิเคต findRule ผสมอยู่ด้วย คือ ภายในลิสต์ย่อยของเพรดิเคต findRuleOr สามารถใส่ได้หลายเงื่อนไขภายใต้การดำเนินการของ AND และลิสต์ย่อยนี้จะถูกนำไปเช็คกับคำสั่งในเพรดิเคต findRule ซึ่งลิสต์ย่อยแต่ละลิสต์จะถูกเชื่อมด้วยการดำเนินการของ OR ซึ่งกฎที่ตรงกับเงื่อนไขของลิสย์ย่อย จะถูกนำมารวมกันเพื่อแสดงผล ตัวอย่างเช่น

- findRuleOr([[outlook - sunny],[outlook - rainy]]) จะได้ผลลัพธ์ดังรูปที่ 3.16

```
if outlook - rainy  windy - false then play – yes
if outlook - rainy  windy - true then play – no
if outlook - sunny  humidity - high then play – no
if outlook - sunny  humidity - normal then play - yes
```

รูปที่ 3.16 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข เมื่อเควียรีด้วยเพรดิเคต findRuleOr([[X],[X]])

- findRuleOr([[\+outlook - sunny],[\+humidity - high]]) จะได้ผลลัพธ์ดังรูปที่ 3.17

```
if outlook - cloudy then play – yes
if outlook - rainy  windy - false then play – yes
if outlook - rainy  windy - true then play – no
if outlook - sunny  humidity - normal then play - yes
```

รูปที่ 3.17 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข เมื่อเควียรีด้วยเพรดิเคต findRuleOr([[\+X],[\+X]])

- findRuleOr([[outlook - sunny , \+humidity - high],
  [\+outlook - sunny , humidity - high]]) จะได้ผลลัพธ์ดังรูปที่ 3.18

```
if outlook - cloudy then play – yes
if outlook - rainy  windy - false then play – yes
if outlook - rainy  windy - true then play – no
if outlook - sunny  humidity - normal then play - yes
```

รูปที่ 3.18 ผลลัพธ์ที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงกฎตามเงื่อนไข เมื่อเควียรีด้วยเพรดิเคต findRuleOr([[X,\+X],[\+X,X]])

บทที่ 4
การทดสอบและอภิปรายผล

การทดสอบประสิทธิภาพของโปรแกรม ID3UP ซึ่งเป็นการทดสอบประสิทธิภาพการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ การเปรียบเทียบประสิทธิภาพจะทำการเปรียบเทียบจำนวนกฎที่ได้จากการเควียรีที่ได้ออกแบบให้มีความสามารถในการใส่เงื่อนไขได้หลายรูปแบบ สำหรับการทดสอบระบบจะทำการทดสอบบนเครื่องคอมพิวเตอร์ Intel Core i5 ความเร็ว 2.4 GHz หน่วยความจำหลัก 4.00 GB ฮาร์ดดิสก์ความจุ 500 GB และทำการทดสอบกับข้อมูล 3 ชุด ปรากฏในรายละเอียดหัวข้อที่ 4.1 ผลของการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP ปรากฏในหัวข้อที่ 4.2 ผลของการค้นหากฎด้วยเพรดิเคตภาษาโปรล็อก ปรากฏในหัวข้อที่ 4.3 ผลของการเปรียบเทียบเวลาและจำนวนกฎสำหรับการเควียรีด้วยคำสั่งที่กำหนด ปรากฏในหัวข้อที่ 4.4 และการอภิปรายผล ปรากฏในหัวข้อที่ 4.5

4.1 ข้อมูลที่ใช้ในการทดสอบ

การทดสอบประสิทธิภาพการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ ได้ใช้ข้อมูลจาก UCI Machine Learning Repository (http://archive.ics.uci.edu/ml) ที่เป็นข้อมูลสำหรับการจำแนกข้อมูล (Classification) จำนวน 3 ชุด โดยรายละเอียดข้อมูลมีดังนี้

ข้อมูลชุดที่หนึ่ง เป็นข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain) โดยมีจำนวนข้อมูลทั้งหมด 155 แถว 15 แอททริบิวต์ ได้แก่ age, sex, steroid, antivirals, fatigue, malaise, anorexia, spleen_palpable, spiders, ascites, varices, bilirubin, sgot, albumin และ class โดยรายละเอียดของแอททริบิวต์ แสดงดังตารางที่ 4.1 และตัวอย่างข้อมูลแสดงดังรูปที่ 4.1

ข้อมูลชุดที่สอง เป็นข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer) โดยมีจำนวนข้อมูลทั้งหมด 285 แถว 10 แอททริบิวต์ ได้แก่ age, menopause, tumor_size, inv_nodes, node_caps, deg_malig, breast-left, breast_quad, irradiat และ class โดยรายละเอียดของแอททริบิวต์ แสดงดังตารางที่ 4.2 และตัวอย่างข้อมูลแสดงดังรูปที่ 4.2

ข้อมูลชุดที่สาม เป็นข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease) โดยมีจำนวนข้อมูลทั้งหมด 1062 แถว 16 แอททริบิวต์ ได้แก่ sex, on_thyroxine, query_on_thyroxine, on_antithyroid_medication, sick, pregnant, thyroid_surgery, query_hypothyroid, query_hyperthyroid, lithium, goiter, tumor, hypopituitary, psych, referral_source และ class โดยรายละเอียดของแอททริบิวต์ แสดงดังตารางที่ 4.3 และตัวอย่างข้อมูลแสดงดังรูปที่ 4.3

ตารางที่ 4.1 รายละเอียดแอททริบิวต์ข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain)

Attributes	All Possible Values
age	"1-10", "11-20", "21-30", "31-40", "41-50", "51-60", "61-70", "71-80"
sex	male, female
steroid	yes, no
antivirals	yes, no
fatigue	yes, no
malaise	yes, no
anorexia	yes, no
spleen_palpable	yes, no
spiders	yes, no
ascites	yes, no
varices	yes, no
bilirubin	"0.1-1.0", "1.1-2.0", "2.1-3.0", "3.1-4.0", "4.1-5.0", "5.1-6.0", "6.1-7.0", "7.1-8.0"

ตารางที่ 4.1 รายละเอียดแอททริบิวต์ข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain) (ต่อ)

Attributes	All Possible Values
sgot	"1-100", "101-200", "201-300", "301-400", "401-500", "501-600", "601-700"
albumin	"2.1-3", "3.1-4", "4.1-5", "5.1-6", "6.1-7"
class	die, live

[age-"21-30",sex-female,steroid-no,antivirals-yes,fatigue-yes,malaise-yes,anorexia-yes,spleen_palpable-yes,spiders-yes,ascites-yes,varices-yes,bilirubin-"0.1-1.0",sgot-"1-100",albumin-"3.1-4",class-live],

[age-"41-50",sex-male,steroid-no,antivirals-yes,fatigue-no,malaise-yes,anorexia-yes,spleen_palpable-yes,spiders-yes,ascites-yes,varices-yes,bilirubin-"0.1-1.0",sgot-"1-100",albumin-"3.1-4",class-live],

[age-"71-80",sex-male,steroid-yes,antivirals-yes,fatigue-no,malaise-yes,anorexia-yes,spleen_palpable-yes,spiders-yes,ascites-yes,varices-yes,bilirubin-"0.1-1.0",sgot-"1-100",albumin-"3.1-4",class-live],

[age-"31-40",sex-male,steroid-yes,antivirals-no,fatigue-yes,malaise-yes,anorexia-yes,spleen_palpable-yes,spiders-yes,ascites-yes,varices-yes,bilirubin-"0.1-1.0",sgot-"1-100",albumin-"3.1-4",class-live],

[age-"31-40",sex-male,steroid-yes,antivirals-yes,fatigue-yes,malaise-yes,anorexia-yes,spleen_palpable-yes,spiders-yes,ascites-yes,varices-yes,bilirubin-"0.1-1.0",sgot-"101-200",albumin-"3.1-4",class-live],

รูปที่ 4.1 ตัวอย่างข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain)

ตารางที่ 4.2 รายละเอียดแอททริบิวต์ข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer)

Attributes	All Possible Values
age	"10-19", "20-29", "30-39", "40-49", "50-59", "60-69", "70-79", "80-89", "90-99"
menopause	lt40, ge40, premeno
tumor_size	"0-4", "5-9", "10-14", "15-19", "20-24", "25-29", "30-34", "35-39", "40-44", "45-49", "50-54", "55-59"
inv_nodes	"0-2", "3-5", "6-8", "9-11", "12-14", "15-17", "18-20", "21-23", "24-26", "27-29", "30-32", "33-35", "36-39"
node_caps	yes, no
deg_malig	1, 2, 3
breast	left, right
breast_quad	left_up, left_low, right_up, right_low, central
irradiat	yes, no
class	no_recurrence_events, recurrence_events

[age-"30-39",menopause-premeno,tumor_size-"30-34",inv_nodes-"0-2",node_caps-no,deg_malig-3,breast-left,breast_quad-left_low,irradiat-no,class-no_recurrence_events],

[age-"40-49",menopause-premeno,tumor_size-"20-24",inv_nodes-"0-2",node_caps-no,deg_malig-2,breast-right,breast_quad-right_up,irradiat-no,class-no_recurrence_events],

[age-"40-49",menopause-premeno,tumor_size-"20-24",inv_nodes-"0-2",node_caps-no,deg_malig-2,breast-left,breast_quad-left_low,irradiat-no,class-no_recurrence_events],

[age-"60-69",menopause-ge40,tumor_size-"15-19",inv_nodes-"0-2",node_caps-no,deg_malig-2,breast-right,breast_quad-left_up,irradiat-no,no,class-recurrence_events],

รูปที่ 4.2 ตัวอย่างข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer)

ตารางที่ 4.3 รายละเอียดแอททริบิวต์ข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease)

Attributes	All Possible Values
sex	female, male
on_thyroxine	false, true
query_on_thyroxine	false, true
on_antithyroid_medication	false, true
sick	false, true
pregnant	false, true
thyroid_surgery	false, true
query_hypothyroid	false, true
query_hyperthyroid	false, true
lithium	false, true
goitre	false, true
tumor	false, true
hypopituitary	false, true
psych	false, true
referral_source	STMW, SVHC, SVI, SVHD, other
class	sick, negative

[sex-female,on_thyroxine-false,query_on_thyroxine-false,on_antithyroid_medication-false,sick-false,pregnant-false,thyroid_surgery-false,query_hypothyroid-false,query_hyperthyroid-f,lithium-false,goitre-false,tumor-false,hypopituitary-false,psych-false,referral_source-SVHC,class-negative],

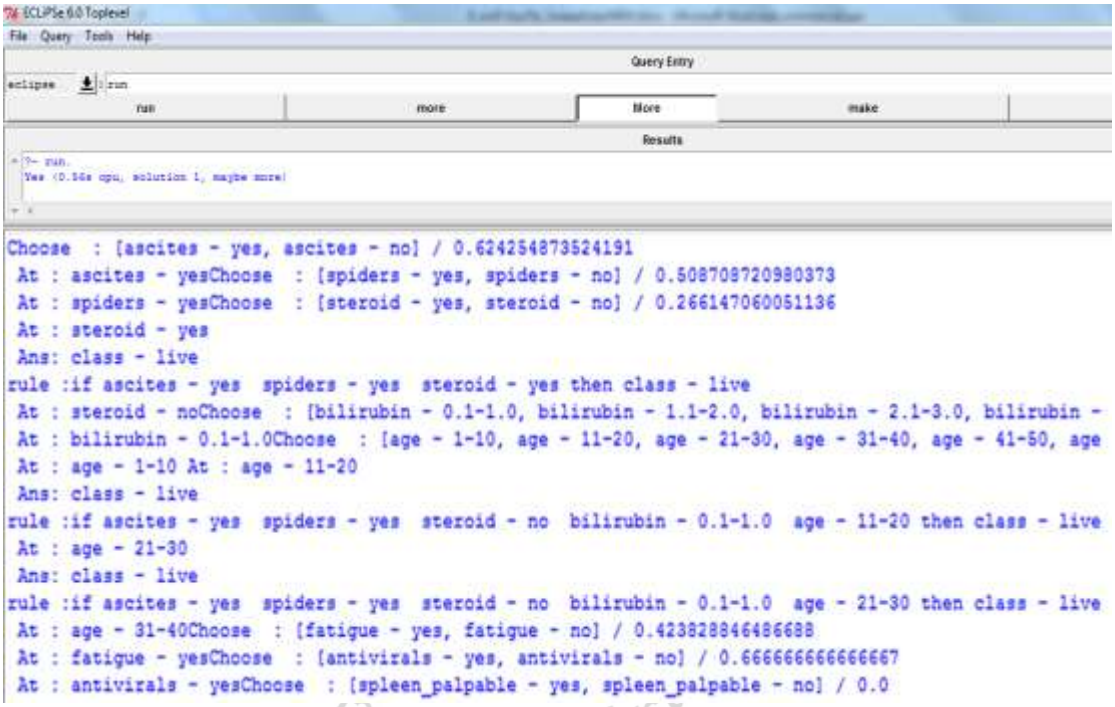
[sex-female,on_thyroxine-false,query_on_thyroxine-false,on_antithyroid_medication-false,sick-false,pregnant-false,thyroid_surgery-false,query_hypothyroid-false,query_hyperthyroid-f,lithium-false,goitre-false,tumor-false,hypopituitary-false,psych-false,referral_source-other,class-negative],

รูปที่ 4.3 ตัวอย่างข้อมูลผู้ป่วยไทรอยด์ (Thyroid Disease)

4.2 ผลของการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP

4.2.1 ผลการทดสอบกับข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain)

นำข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain) ทดสอบกับโปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ เริ่มจากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP ที่พัฒนาเป็นโปรแกรมด้วยภาษาโปรล็อกให้ผลการทดสอบ ดังรูปที่ 4.4



รูปที่ 4.4 ผลลัพธ์ของข้อมูลผู้ป่วยโรคตับอักเสบเมื่อถูกจำแนกด้วยโปรแกรม ID3UP

จากรูปที่ 4.4 โปรแกรมได้จำแนกข้อมูลแอททริบิวต์พร้อมค่า ที่เลือกในแต่ละโหนด พร้อมทั้งได้กฎของข้อมูลออกมา ตัวอย่างผลลัพธ์ของข้อมูลผู้ป่วยโรคตับอักเสบเมื่อถูกจำแนกข้อมูลด้วยอัลกอริทึม ID3UP แสดงดังรูปที่ 4.5

Choose : [ascites - yes, ascites - no] / 0.624254873524191

At : ascites - yesChoose : [spiders - yes, spiders - no] / 0.508708720980373

At : spiders - yesChoose : [steroid - yes, steroid - no] / 0.266147060051136

At : steroid - yes

Ans: class - live

rule :if ascites - yes spiders - yes steroid - yes then class - live

รูปที่ 4.5 ผลการจำแนกข้อมูลผู้ป่วยโรคตับอักเสบด้วยโปรแกรม ID3UP แสดงการสร้างกฎลำดับที่หนึ่ง

จากรูปที่ 4.5 ความหมายของขั้นตอนการจำแนกข้อมูล คือ เริ่มเลือกโหนดรากเป็น แอททริบิวต์ ascites มีค่าเป็น yes โหนดต่อไปเลือกแอททริบิวต์ spiders มีค่าเป็น yes โหนดต่อไปเลือกแอททริ-บิวต์ steroid มีค่าเป็น yes และแอททริบิวต์เป้าหมาย class มีค่าเป็น live ดังนั้นกฎที่ได้ คือ **if ascites - yes spiders - yes steroid - yes then class - live**

At : steroid - noChoose : [bilirubin - 0.1-1.0, bilirubin - 1.1-2.0, bilirubin - 2.1-3.0, bilirubin - 3.1-4.0, bilirubin - 4.1-5.0, bilirubin - 5.1-6.0, bilirubin - 6.1-7.0, bilirubin - 7.1-8.0] / 0.453477512857794

At : bilirubin - 0.1-1.0Choose : [age - 1-10, age - 11-20, age - 21-30, age - 31-40, age - 41-50, age - 51-60, age - 61-70, age - 71-80] / 0.364421968666466

At : age - 1-10 At : age - 11-20

Ans: class - live

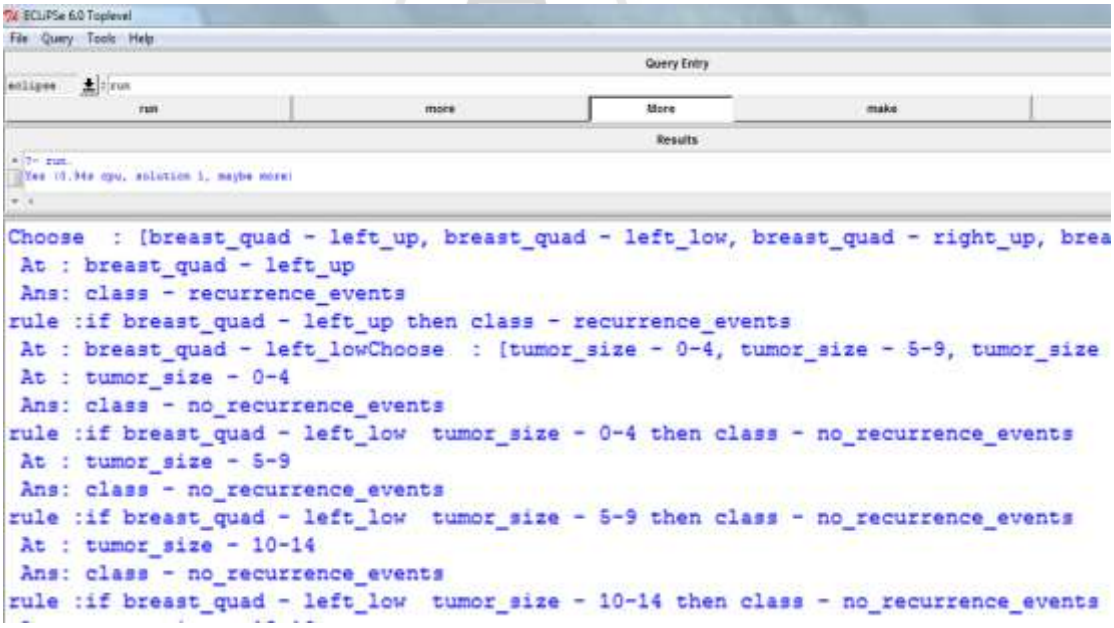
rule :if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 11-20 then class – live

รูปที่ 4.6 ผลการจำแนกข้อมูลผู้ป่วยโรคตับอักเสบด้วยโปรแกรม ID3UP แสดงการสร้างกฎลำดับที่สอง

จากรูปที่ 4.6 ข้อมูลที่ถูกจำแนกได้จำแนกต่อจากการเลือกโหนดราก คือ แอททริบิวต์ ascites มีค่าเป็น yes และโหนดต่อไป คือ แอททริบิวต์ spiders มีค่าเป็น yes (จากรูปที่ 4.5) โดยจะเลือกโหนดต่อไปเป็นแอททริบิวต์ steroid มีค่าเป็น no และโหนดต่อไปเป็นแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 โหนดระดับต่อจากนั้นเป็นแอททริบิวต์ age มีค่าเป็น 11-20 และที่โหนดใบเป็นแอททริบิวต์เป้าหมาย คือ class มีค่าเป็น live ดังนั้นกฎที่ได้ คือ **if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 11-20 then class – live**

4.2.2 ผลการทดสอบกับข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer)

นำข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer) ทดสอบกับโปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ เริ่มจากการจำแนกข้อมูลด้วยโปรแกรม ID3UP ให้ผลการทดสอบ ดังรูปที่ 4.7



รูปที่ 4.7 ผลลัพธ์ของข้อมูลผู้ป่วยมะเร็งเต้านมเมื่อถูกจำแนกด้วยโปรแกรม ID3UP

Choose : [breast_quad - left_up, breast_quad - left_low, breast_quad - right_up, breast_quad - right_low, breast_quad - central] / 0.531951311047556

At : breast_quad - left_up

Ans: class - recurrence_events

rule :if breast_quad - left_up then class - recurrence_events

รูปที่ 4.8 ผลการจำแนกข้อมูลผู้ป่วยมะเร็งเต้านมโปรแกรม ID3UP
แสดงการสร้างกฎลำดับที่หนึ่ง

จากรูปที่ 4.8 ความหมายของโมเดลที่ได้เมื่อโปรแกรมมีการจำแนกข้อมูล คือ เริ่มเลือกโหนดรากเป็น แอททริบิวต์ breast_quad มีค่าเป็น left_up โหนดต่อไป คือ แอททริบิวต์เป้าหมาย class มีค่าเป็น recurrence_events ดังนั้นกฎที่ได้ คือ **if breast_quad - left_up then class - recurrence_events**

At : breast_quad - left_low Choose : [inv_nodes - 0-2, inv_nodes - 3-5, inv_nodes - 6-8, inv_nodes - 9-11, inv_nodes - 12-14, inv_nodes - 15-17, inv_nodes - 18-20, inv_nodes - 21-23, inv_nodes - 24-26, inv_nodes - 27-29, inv_nodes - 30-32, inv_nodes - 33-35, inv_nodes - 36-39] / 0.781265510891497

At : inv_nodes - 0-2 Choose : [tumor_size - 0-4, tumor_size - 5-9, tumor_size - 10-14, tumor_size - 15-19, tumor_size - 20-24, tumor_size - 25-29, tumor_size - 30-34, tumor_size - 35-39, tumor_size - 40-44, tumor_size - 45-49, tumor_size - 50-54, tumor_size - 55-59] / 0.617466806814915

At : tumor_size - 0-4

Ans: class - no_recurrence_events

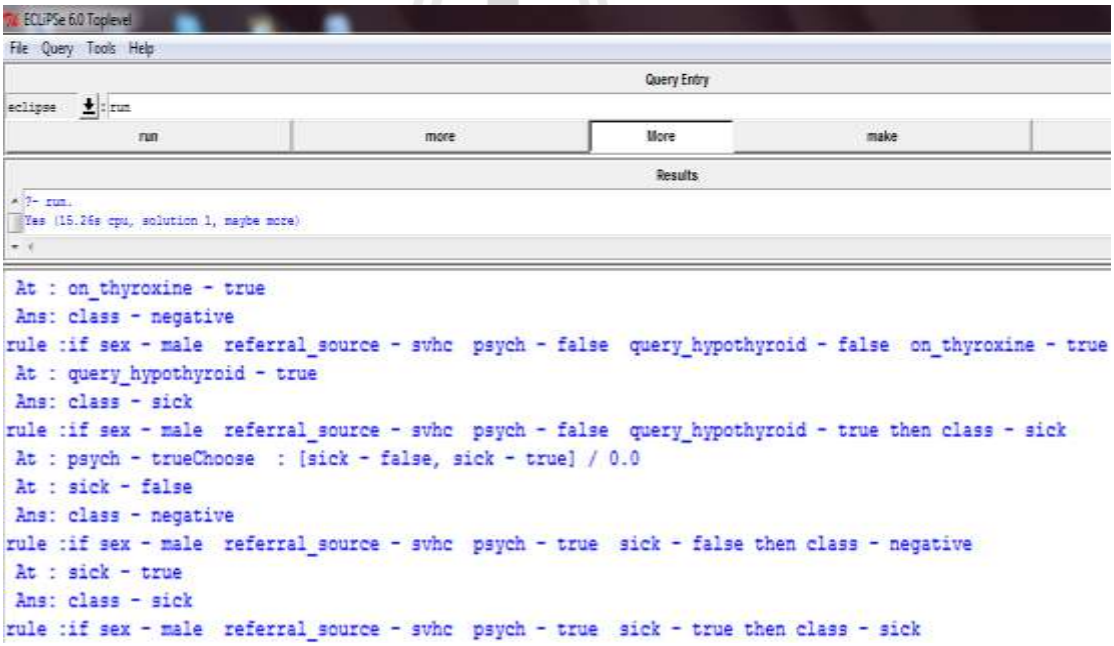
rule :if breast_quad - left_low inv_nodes - 0-2 tumor_size - 0-4 then class - no_recurrence_events

รูปที่ 4.9 ผลการจำแนกข้อมูลผู้ป่วยมะเร็งเต้านมโปรแกรม ID3UP
แสดงการสร้างกฎลำดับที่สอง

จากรูปที่ 4.9 โปรแกรมจำแนกข้อมูลต่อ (จากรูปที่ 4.8) โดยเลือกโหนดรากเป็นแอททริ-บิวต์ *breast_quad* เหมือนเดิม แต่เลือกค่าเป็น *left_low* โหนดต่อไปเลือก แอททริบิวต์ *inv_nodes* มีค่าเป็น *0-2* โหนดต่อไปเลือกแอททริบิวต์ *tumor_size* มีค่าเป็น *0-4* และแอททริบิวต์เป้าหมาย คือ class มีค่าเป็น *no_recurrence_events* ดังนั้นกฎที่ได้ คือ **if breast_quad - left_up then class - no_recurrence_events**

4.2.3 ผลการทดสอบกับข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease)

นำข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease) ทดสอบกับโปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ เริ่มจากการจำแนกข้อมูลด้วยโปรแกรม ID3UP ให้ผลการทดสอบ ดังรูปที่ 4.10



รูปที่ 4.10 ผลลัพธ์ของข้อมูลผู้ป่วยโรคไทรอยด์เมื่อถูกจำแนกด้วยโปรแกรม ID3UP

Choose : [sex - female, sex - male] / 0.403017820005403

At : sex - femaleChoose : [sick - false, sick - true] / 0.246427508015239

At : sick - falseChoose : [referral_source - stmw, referral_source - svhc, referral_source - svi, referral_source - svhd, referral_source - other] / 0.20330430468268

At : referral_source - stmw

Ans: class - negative

rule :if sex - female sick - false referral_source - stmw then class – negative

รูปที่ 4.11 ผลการจำแนกข้อมูลผู้ป่วยโรคไทรอยด์ด้วยโปรแกรม ID3UP แสดงการสร้างกฎลำดับที่หนึ่ง

จากรูปที่ 4.11 โปรแกรมเริ่มจำแนกข้อมูลโดยเลือกโหนดรากเป็นแอททริบิวต์ *sex* มีค่าเป็น *female* โหนดต่อไปเลือกแอททริบิวต์ *sick* มีค่าเป็น *false* โหนดต่อไปเลือกแอททริบิวต์ *referral_source* มีค่าเป็น *stmw* และแอททริบิวต์เป้าหมาย คือ *class* มีค่าเป็น *negative* ดังนั้นกฎที่ได้คือ **if sex - female sick - false referral_source - stmw then class – negative**

At : referral_source - svhc

Ans: class - negative

rule :if sex - female sick - false referral_source - svhc then class – negative

รูปที่ 4.12 ผลการจำแนกข้อมูลผู้ป่วยโรคไทรอยด์ด้วยโปรแกรม ID3UP แสดงการสร้างกฎลำดับที่สอง

จากรูปที่ 4.12 ข้อมูลถูกจำแนกต่อ (จากรูปที่ 4.11) โดยที่โหนดรากและโหนด sick - false ยังคงเป็นเหมือนเดิม แต่แตกต่างตรงที่โหนดที่เลือกแอททริบิวต์ *referral_source* จะให้ค่าเป็น svhc และโหนดถัดไปจะเป็นแอททริบิวต์เป้าหมาย *class* ซึ่งมีค่าเป็น *negative* ดังนั้นกฎที่ได้ คือ **if sex - female sick - false referral_source - svhc then class – negative**

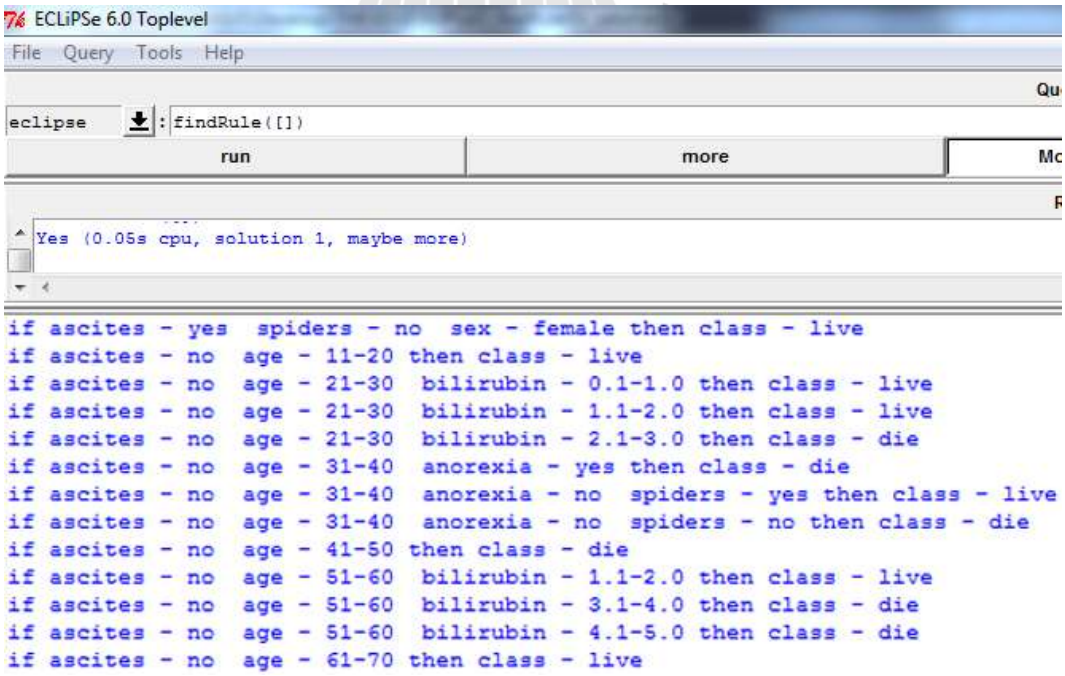
4.3 ผลของการค้นหากฎด้วยเพรดิเคตตรรกศาสตร์

กฎที่ได้จากการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP จะถูกบันทึกไว้ในหน่วยความจำเพื่อรอการค้นหากฎตามเงื่อนไขของผู้ใช้ ดังรายละเอียดต่อไปนี้

4.3.1 การค้นหากฎจำแนกข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain)

(1) ค้นหากฎทั้งหมดด้วยเพรดิเคต findRule() จากโมเดลข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain)

การค้นหากฎทั้งหมดที่ได้จากการจำแนกข้อมูล จะใช้เพรดิเคต findRule() ที่ใส่ค่าอาร์กิวเมนต์ของเพรดิเคตเป็นเซตว่าง ([ ]) แสดงดังรูปที่ 4.13 ซึ่งการค้นหากฎทั้งหมดของข้อมูลผู้ป่วยโรคตับอักเสบ ได้จำนวนกฎทั้งหมด 51 กฎ แสดงดังตารางที่ 4.4



รูปที่ 4.13 ผลการทดสอบกับข้อมูลผู้ป่วยโรคตับอักเสบ เมื่อเควียรีกฎทั้งหมดด้วย findRule([ ])

ตารางที่ 4.4 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคตับอักเสบ

No.	Rule
1	if ascites - yes spiders - yes steroid - yes then class - live
2	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 11-20 then class - live
3	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 21-30 then class - live
4	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - yes spleen_palpable - yes then class - die
5	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - yes spleen_palpable - no then class - live
6	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - no then class - live
7	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - no then class - live
8	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 41-50 varices - yes then class - live
9	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 41-50 varices - no then class - die
10	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 51-60 then class - live
11	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 61-70 then class - die
12	if ascites - yes spiders - yes steroid - no bilirubin - 1.1-2.0 then class - live
13	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 21-30 then class - live
14	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 31-40 then class - die
15	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 41-50 then class - live
16	if ascites - yes spiders - yes steroid - no bilirubin - 7.1-8.0 then class - die
17	if ascites - yes spiders - no sex - male sgot - 1-100 age - 21-30 then class - live
18	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - yes malaise - yes then class - die
19	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - yes malaise - no then class – live

ตารางที่ 4.4 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคตับอักเสบ (ต่อ)

No.	Rule
20	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - no then class - live
21	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 0.1-1.0 steroid - yes then class - live
22	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 0.1-1.0 steroid - no anorexia - yes then class - die
23	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 0.1-1.0 steroid - no anorexia - no then class - live
24	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 1.1-2.0 then class - die
25	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 2.1-3.0 then class - live
26	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 4.1-5.0 then class - die
27	if ascites - yes spiders - no sex - male sgot - 1-100 age - 51-60 then class - live
28	if ascites - yes spiders - no sex - male sgot - 1-100 age - 61-70 albumin - 3.1-4 then class - die
29	if ascites - yes spiders - no sex - male sgot - 1-100 age - 61-70 albumin - 4.1-5 then class - live
30	if ascites - yes spiders - no sex - male sgot - 101-200 age - 21-30 then class - live
31	if ascites - yes spiders - no sex - male sgot - 101-200 age - 31-40 steroid - yes then class - live
32	if ascites - yes spiders - no sex - male sgot - 101-200 age - 31-40 steroid - no then class - die
33	if ascites - yes spiders - no sex - male sgot - 101-200 age - 41-50 then class - die
34	if ascites - yes spiders - no sex - male sgot - 101-200 age - 51-60 then class - die

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.4 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคตับอักเสบ (ต่อ)

No.	Rule
35	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - yes then class - live
36	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - no steroid - yes then class - die
37	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - no steroid – no then class - live
38	if ascites - yes spiders - no sex - male sgot - 501-600 then class - die
39	if ascites - yes spiders - no sex - female then class - live
40	if ascites - no age - 11-20 then class - live
41	if ascites - no age - 21-30 bilirubin - 0.1-1.0 then class - live
42	if ascites - no age - 21-30 bilirubin - 1.1-2.0 then class - live
43	if ascites - no age - 21-30 bilirubin - 2.1-3.0 then class - die
44	if ascites - no age - 31-40 anorexia - yes then class - die
45	if ascites - no age - 31-40 anorexia - no spiders - yes then class - live
46	if ascites - no age - 31-40 anorexia - no spiders - no then class - die
47	if ascites - no age - 41-50 then class - die
48	if ascites - no age - 51-60 bilirubin - 1.1-2.0 then class - live
49	if ascites - no age - 51-60 bilirubin - 3.1-4.0 then class - die
50	if ascites - no age - 51-60 bilirubin - 4.1-5.0 then class - die
51	if ascites - no age - 61-70 then class - live

(2) ค้นหากฎด้วยเพรดิเคต findRule() เมื่อใส่เงื่อนไขให้กับอาร์กิวเมนต์ทดสอบด้วยข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain)

- กำหนดให้อาร์กิวเมนต์ ของการเควียรีมีจำนวน 1 ตัวแปร

กำหนด X เป็นตัวแปร โดยให้ค่าของ X เป็น **bilirubin - "0.1-1.0"** ดังนั้นคำสั่งการเควียรี คือ **findRule([bilirubin - "0.1-1.0"])** ผลลัพธ์ที่ได้ คือ จำนวนกฎมีจำนวนน้อยลงจากการค้นหากฎทั้งหมด คือ มีจำนวน 14 กฎ จากทั้งหมด 51 กฎ แสดงดังตารางที่ 4.5 ซึ่งทำให้เห็นว่าการค้นหากฎโดยใช้เงื่อนไขมีประสิทธิภาพที่ดี แสดงผลลัพธ์เฉพาะเงื่อนไขที่ผู้ใช้กำหนด ไม่แสดงกฎที่ไม่ตรงกับเงื่อนไข บางทีกฎบางกฎอาจจะไม่จำเป็นต่อผู้ใช้ อย่างเช่นตัวอย่าง แสดงเฉพาะกฎที่มีข้อมูล *ปริมาณน้ำดีในตับมีจำนวน 0.1-1.0 mg/dl (bilirubin-"0.1-1.0") จะมีผู้ป่วยที่มีชีวิตอยู่หรือเสียชีวิตด้วยเงื่อนไขใดบ้าง* (ดูได้จากเงื่อนไขของกฎที่ได้ถูกค้นหาออกมา)

ตารางที่ 4.5 กฎที่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 อยู่ในกฎ

No.	Rule
1	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 11-20 then class - live
2	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 21-30 then class - live
3	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 31-40 fatigue - yes antivirals - yes spleen_palpable - yes then class - die
4	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 31-40 fatigue - yes antivirals - yes spleen_palpable - no then class - live
5	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 31-40 fatigue - yes antivirals - no then class - live
6	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 31-40 fatigue - no then class - live
7	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 41-50 varices - yes then class - live
8	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 41-50 varices - no then class - die

ตารางที่ 4.5 กฎที่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 อยู่ในกฎ (ต่อ)

No.	Rule
9	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 51-60 then class - live
10	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 61-70 then class - die
11	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 **bilirubin - 0.1-1.0** steroid - yes then class - live
12	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 **bilirubin - 0.1-1.0** steroid - no anorexia - yes then class - die
13	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 **bilirubin - 0.1-1.0** steroid - no anorexia - no then class - live
14	if ascites - no age - 21-30 **bilirubin - 0.1-1.0** then class - live

การกำหนดค่าของ X ใส่ในรูปของเงื่อนไข NOT โดยกำหนดให้เป็น **\+bilirubin - "0.1-1.0"** ดังนั้นคำสั่งการเควียรี คือ **findRule([\+bilirubin - "0.1-1.0"])** ซึ่งจะแสดงกฎที่ไม่มี bilirubin - 0.1-1.0 อยู่ในกฎ ซึ่งผลลัพธ์ที่ได้มีจำนวน 37 กฎ ซึ่งจำนวนของกฎลดน้อยลงจากการค้นหากฎทั้งหมดโดยไม่ใส่เงื่อนไขเช่นกัน แสดงดังตารางที่ 4.6

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.6 กฎที่ไม่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 อยู่ในกฎ

No.	Rule
1	if ascites - yes spiders - yes steroid - yes then class – live
2	if ascites - yes spiders - yes steroid - no bilirubin - 1.1-2.0 then class - live
3	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 21-30 then class - live
4	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 31-40 then class - die
5	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 41-50 then class - live
6	if ascites - yes spiders - yes steroid - no bilirubin - 7.1-8.0 then class - die
7	if ascites - yes spiders - no sex - male sgot - 1-100 age - 21-30 then class - live

ตารางที่ 4.6 กฎที่ไม่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 อยู่ในกฎ (ต่อ)

No.	Rule
8	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - yes malaise - yes then class - die
9	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - yes malaise - no then class - live
10	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - no then class - live
11	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 1.1-2.0 then class - die
12	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 2.1-3.0 then class - live
13	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 4.1-5.0 then class - die
14	if ascites - yes spiders - no sex - male sgot - 1-100 age - 51-60 then class - live
15	if ascites - yes spiders - no sex - male sgot - 1-100 age - 61-70 albumin - 3.1-4 then class - die
16	if ascites - yes spiders - no sex - male sgot - 1-100 age - 61-70 albumin - 4.1-5 then class - live
17	if ascites - yes spiders - no sex - male sgot - 101-200 age - 21-30 then class - live
18	if ascites - yes spiders - no sex - male sgot - 101-200 age - 31-40 steroid - yes then class - live
19	if ascites - yes spiders - no sex - male sgot - 101-200 age - 31-40 steroid - no then class - die
20	if ascites - yes spiders - no sex - male sgot - 101-200 age - 41-50 then class - die
21	if ascites - yes spiders - no sex - male sgot - 101-200 age - 51-60 then class - die
22	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - yes then class - live

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.6 กฎที่ไม่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 อยู่ในกฎ (ต่อ)

No.	Rule
23	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - no steroid - yes then class - die
24	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - no steroid - no then class - live
25	if ascites - yes spiders - no sex - male sgot - 501-600 then class - die
26	if ascites - yes spiders - no sex - female then class - live
27	if ascites - no age - 11-20 then class - live
28	if ascites - no age - 21-30 bilirubin - 1.1-2.0 then class - live
29	if ascites - no age - 21-30 bilirubin - 2.1-3.0 then class - die
30	if ascites - no age - 31-40 anorexia - yes then class - die
31	if ascites - no age - 31-40 anorexia - no spiders - yes then class - live
32	if ascites - no age - 31-40 anorexia - no spiders - no then class - die
33	if ascites - no age - 41-50 then class - die
34	if ascites - no age - 51-60 bilirubin - 1.1-2.0 then class - live
35	if ascites - no age - 51-60 bilirubin - 3.1-4.0 then class - die
36	if ascites - no age - 51-60 bilirubin - 4.1-5.0 then class - die
37	if ascites - no age - 61-70 then class - live

- กำหนดให้อาร์กิวเมนต์ของการเควียรีมีจำนวน 2 ตัวแปรหรือมากกว่า

กำหนดตัวแปรตัวที่ 1 ให้เป็นเหมือนเดิม คือ X1 = **bilirubin - "0.1-1.0"** และเพิ่มเงื่อนไขให้กับตัวแปรตัวที่ 2 คือ X2 = **class – live** ดังนั้นคำสั่งในการเควียรี คือ **findRule([bilirubin - "0.1-1.0" , class - live])** กฎที่ได้จากการเควียรีเงื่อนไขนี้แสดงดังตารางที่ 4.7 ซึ่งจำนวนของกฎลดลงเหลือ 10 กฎ เมื่อเทียบกับผลจากการเควียรีด้วย 1 ตัวแปร จะได้กฎ 14 กฎ เนื่องจากเงื่อนไขที่กำหนดให้อาร์กิวเมนต์ของเพรดิเคต findRule() ภายในลิสต์จะคั่นด้วยเครื่องหมาย “ , ” คือการใส่เงื่อนไขของ AND ซึ่งเป็นการคัดเลือกกฎที่มีเงื่อนไขตรงกันทั้ง 2 เงื่อนไข ทำให้เงื่อนไขในการค้นหามีความจำเพาะมากขึ้น จึงทำให้กฎมีจำนวนลดลง ซึ่งการใส่เงื่อนไขจะไม่จำกัดจำนวนของตัวแปร ตัวอย่างเช่น ผลลัพธ์ของกฎที่ได้ถูกเพิ่มเงื่อนไขจาก 1 เงื่อนไข เป็น 2 เงื่อนไข มีความหมายว่า

ปริมาณน้ำดีในตับมีจำนวน 0.1-1.0 mg/dl (bilirubin-"0.1-1.0") จะมีผู้ป่วยที่มีชีวิต (class-live) เป็นจำนวนมากน้อยเท่าใด (ดูได้จากจำนวนกฎที่ได้ถูกค้นหาออกมา)

ตารางที่ 4.7 กฎที่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 และมีแอททริบิวต์ class มีค่าเป็น live

No.	Rule
1	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 11-20 then **class - live**
2	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 21-30 then **class - live**
3	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 31-40 fatigue - yes antivirals - yes spleen_palpable - no then **class - live**
4	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 31-40 fatigue - yes antivirals - no then **class - live**
5	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 31-40 fatigue - no then **class - live**
6	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 41-50 varices - yes then **class - live**
7	if ascites - yes spiders - yes steroid - no **bilirubin - 0.1-1.0** age - 51-60 then **class - live**
8	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 **bilirubin - 0.1-1.0** steroid - yes then **class - live**
9	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 **bilirubin - 0.1-1.0** steroid - no anorexia - no then **class - live**
10	if ascites - no age - 21-30 **bilirubin - 0.1-1.0** then **class - live**

เมื่อทดลองกำหนดค่าของ X1 หรือ X2 ใส่ในรูปเงื่อนไข NOT โดยผสมกันอยู่ภายในลิสต์ โดยกำหนดให้เป็น X1 = \+**bilirubin - "0.1-1.0"**และ X2 = **class – live** ดังนั้นคำสั่งในการเควียรีคือ **findRule([\+bilirubin - "0.1-1.0" , class - live])** ซึ่งจะแสดงกฎที่ไม่มี bilirubin - 0.1-1.0 แต่มี class-live อยู่ในกฎ ซึ่งผลลัพธ์ที่ได้มีจำนวน 20 กฎ (แสดงดังตารางที่ 4.8) ซึ่งจำนวนของกฎลดน้อยลงจากการค้นหากฎทั้งหมดโดยไม่ใส่เงื่อนไข

ตารางที่ 4.8 กฎที่ไม่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 และมีแอททริบิวต์ class มีค่าเป็น live

No.	Rule
1	if ascites - yes spiders - yes steroid - yes then **class - live**
2	if ascites - yes spiders - yes steroid - no bilirubin - 1.1-2.0 then **class - live**
3	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 21-30 then **class - live**
4	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 41-50 then **class - live**
5	if ascites - yes spiders - no sex - male sgot - 1-100 age - 21-30 then **class - live**
6	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - yes malaise - no then **class - live**
7	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - no then **class - live**
8	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 2.1-3.0 then **class - live**
9	if ascites - yes spiders - no sex - male sgot - 1-100 age - 51-60 then **class - live**
10	if ascites - yes spiders - no sex - male sgot - 1-100 age - 61-70 albumin - 4.1-5 then **class - live**
11	if ascites - yes spiders - no sex - male sgot - 101-200 age - 21-30 then **class - live**
12	if ascites - yes spiders - no sex - male sgot - 101-200 age - 31-40 steroid - yes then **class - live**
13	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - yes then **class - live**
14	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - no steroid - no then **class - live**
15	if ascites - yes spiders - no sex - female then **class - live**
16	if ascites - no age - 11-20 then **class - live**
17	if ascites - no age - 21-30 bilirubin - 1.1-2.0 then **class - live**
18	if ascites - no age - 31-40 anorexia - no spiders - yes then **class - live**
19	if ascites - no age - 51-60 bilirubin - 1.1-2.0 then **class - live**
20	if ascites - no age - 61-70 then **class - live**

(3) ค้นหากฎด้วยเพรดิเคต findRuleOr() เมื่อใส่เงื่อนไขให้กับอาร์กิวเมนต์ทดสอบด้วยข้อมูลผู้ป่วยโรคตับอักเสบ (Hepatitis Domain)

- กำหนดให้อาร์กิวเมนต์ของการเควียรีมีจำนวน 2 ตัวแปรหรือมากกว่า

กำหนดตัวแปรตัวที่ 1 คือ [X1] = **ascites – yes** ตัวแปรตัวที่ 2 คือ [X2] = **bilirubin - "0.1-1.0" , age - "31-40"** ดังนั้นคำสั่งของการเควียรี จะได้ **findRuleOr([[ascites - yes] , [bilirubin - "0.1-1.0" , age - "31-40"]])** ผลลัพธ์การแสดงผลของกฎมีจำนวนน้อยลงเช่นกัน คือ 43 กฎ แต่บางส่วนของกฎแสดงผลซ้ำกัน เนื่องจากการค้นหากฎของโปรแกรมจะทำเงื่อนไขของตัวแปรตัวที่ 1 ก่อน คือ ascites – yes เมื่อกฎใดที่เข้าเงื่อนไขนี้จะแสดงผลออกมา และตรวจสอบเงื่อนไขของตัวแปรตัวต่อไป คือ bilirubin - "0.1-1.0" , age - "31-40" เมื่อกฎที่ตรงกับเงื่อนไขนี้ก็จะแสดงผลออกมาเช่นกัน ทำให้การแสดงผลของกฎทั้งหมดมีโอกาสซ้ำกัน ดังตารางที่ 4.9 โดยกฎลำดับที่ 4-7 (เงื่อนไขของตัวแปรที่ 1) และกฎลำดับที่ 40-43 (เงื่อนไขของตัวแปรที่ 2) ซ้ำกัน

ตารางที่ 4.9 กฎที่มีแอททริบิวต์ ascites มีค่าเป็น yes รวมกับ กฎที่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 และมีแอททริบิวต์ age มีค่าเป็น 31-40

No.	Rule
1	if ascites - yes spiders - yes steroid - yes then class - live
2	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 11-20 then class - live
3	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 21-30 then class - live
4	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - yes spleen_palpable - yes then class - die
5	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - yes spleen_palpable - no then class - live
6	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - no then class - live
7	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - no then class - live

ตารางที่ 4.9 กฎที่มีแอททริบิวต์ ascites มีค่าเป็น yes รวมกับ กฎที่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 และมีแอททริบิวต์ age มีค่าเป็น 31-40 (ต่อ)

No.	Rule
8	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 41-50 varices - yes then class - live
9	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 41-50 varices - no then class - die
10	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 51-60 then class - live
11	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 61-70 then class - die
12	if ascites - yes spiders - yes steroid - no bilirubin - 1.1-2.0 then class - live
13	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 21-30 then class - live
14	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 31-40 then class - die
15	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 41-50 then class - live
16	if ascites - yes spiders - yes steroid - no bilirubin - 7.1-8.0 then class - die
17	if ascites - yes spiders - no sex - male sgot - 1-100 age - 21-30 then class - live
18	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - yes malaise - yes then class - die
19	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - yes malaise - no then class - live
20	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - no then class - live
21	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 0.1-1.0 steroid - yes then class - live
22	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 0.1-1.0 steroid - no anorexia - yes then class - die
23	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 0.1-1.0 steroid - no anorexia - no then class - live
24	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 1.1-2.0 then class - die

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.9 กฎที่มีแอททริบิวต์ ascites มีค่าเป็น yes รวมกับ กฎที่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 และมีแอททริบิวต์ age มีค่าเป็น 31-40 (ต่อ)

No.	Rule
25	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 2.1-3.0 then class - live
26	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 4.1-5.0 then class - die
27	if ascites - yes spiders - no sex - male sgot - 1-100 age - 51-60 then class - live
28	if ascites - yes spiders - no sex - male sgot - 1-100 age - 61-70 albumin - 3.1-4 then class - die
29	if ascites - yes spiders - no sex - male sgot - 1-100 age - 61-70 albumin - 4.1-5 then class - live
30	if ascites - yes spiders - no sex - male sgot - 101-200 age - 21-30 then class - live
31	if ascites - yes spiders - no sex - male sgot - 101-200 age - 31-40 steroid - yes then class - live
32	if ascites - yes spiders - no sex - male sgot - 101-200 age - 31-40 steroid - no then class - die
33	if ascites - yes spiders - no sex - male sgot - 101-200 age - 41-50 then class - die
34	if ascites - yes spiders - no sex - male sgot - 101-200 age - 51-60 then class - die
35	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - yes then class - live
36	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - no steroid - yes then class - die
37	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - no steroid - no then class - live
38	if ascites - yes spiders - no sex - male sgot - 501-600 then class - die
39	if ascites - yes spiders - no sex - female then class - live

ตารางที่ 4.9 กฎที่มีแอททริบิวต์ ascites มีค่าเป็น yes รวมกับ กฎที่มีแอททริบิวต์ bilirubin มีค่าเป็น 0.1-1.0 และมีแอททริบิวต์ age มีค่าเป็น 31-40 (ต่อ)

No.	Rule
40	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - yes spleen_palpable - yes then class - die
41	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - yes spleen_palpable - no then class - live
42	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - no then class - live
43	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - no then class - live

- ปรับปรุงโปรแกรมไม่ให้แสดงกฎซ้ำกัน เมื่อกำหนดให้อาร์กิวเมนต์ของการเควียรีมีจำนวน 2 ตัวแปรหรือมากกว่า

จากการเควียรีด้วยคำสั่ง **findRuleOr([[ascites - yes] , [bilirubin - "0.1-1.0" , age - "31-40"]])** ซึ่งได้ผลลัพธ์ของกฎที่ซ้ำกัน ทำให้ประสิทธิภาพการแสดงผลลดลงและจำนวนของกฎมีมากเกินความจำเป็น จึงปรับปรุงโปรแกรมด้วยการใช้คำสั่งการค้นหากฎที่จะตัดกฎที่ซ้ำกันออกและแสดงเพียงกฎเดียว ทำให้กฎที่ได้มีจำนวนลดลงจากเดิม มีประสิทธิภาพในการแสดงผลที่ดีขึ้น ดังตารางที่ 4.10 จะเห็นได้ว่าจำนวนกฎที่ได้มีจำนวนที่น้อยลงกว่าเดิม คือ 39 กฎ จาก 43 กฎ

ตารางที่ 4.10 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน

No.	Rule
1	if ascites - yes spiders - yes steroid - yes then class - live
2	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 11-20 then class - live
3	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 21-30 then class - live
4	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - yes spleen_palpable - yes then class - die
5	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - yes spleen_palpable - no then class - live
6	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - no then class - live
7	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - no then class - live
8	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 41-50 varices - yes then class - live
9	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 41-50 varices - no then class - die
10	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 51-60 then class - live
11	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 61-70 then class - die
12	if ascites - yes spiders - yes steroid - no bilirubin - 1.1-2.0 then class - live
13	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 21-30 then class - live
14	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 31-40 then class - die
15	if ascites - yes spiders - yes steroid - no bilirubin - 2.1-3.0 age - 41-50 then class - live
16	if ascites - yes spiders - yes steroid - no bilirubin - 7.1-8.0 then class - die
17	if ascites - yes spiders - no sex - male sgot - 1-100 age - 21-30 then class - live
18	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - yes malaise - yes then class - die
19	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - yes malaise - no then class - live

ตารางที่ 4.10 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน (ต่อ)

No.	Rule
20	if ascites - yes spiders - no sex - male sgot - 1-100 age - 31-40 steroid - no then class - live
21	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 0.1-1.0 steroid - yes then class - live
22	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 0.1-1.0 steroid - no anorexia - yes then class - die
23	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 0.1-1.0 steroid - no anorexia - no then class - live
24	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 1.1-2.0 then class - die
25	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 2.1-3.0 then class - live
26	if ascites - yes spiders - no sex - male sgot - 1-100 age - 41-50 bilirubin - 4.1-5.0 then class - die
27	if ascites - yes spiders - no sex - male sgot - 1-100 age - 51-60 then class - live
28	if ascites - yes spiders - no sex - male sgot - 1-100 age - 61-70 albumin - 3.1-4 then class - die
29	if ascites - yes spiders - no sex - male sgot - 1-100 age - 61-70 albumin - 4.1-5 then class - live
30	if ascites - yes spiders - no sex - male sgot - 101-200 age - 21-30 then class - live
31	if ascites - yes spiders - no sex - male sgot - 101-200 age - 31-40 steroid - yes then class - live
32	if ascites - yes spiders - no sex - male sgot - 101-200 age - 31-40 steroid - no then class - die
33	if ascites - yes spiders - no sex - male sgot - 101-200 age - 41-50 then class - die
34	if ascites - yes spiders - no sex - male sgot - 101-200 age - 51-60 then class - die

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.10 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน (ต่อ)

No.	Rule
35	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - yes then class - live
36	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - no steroid - yes then class - die
37	if ascites - yes spiders - no sex - male sgot - 201-300 spleen_palpable - no steroid - no then class - live
38	if ascites - yes spiders - no sex - male sgot - 501-600 then class - die
39	if ascites - yes spiders - no sex - female then class - live

4.3.2 การค้นหากฎจำแนกข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer)

(1) ค้นหากฎทั้งหมดด้วยเพรดิเคต findRule() จากโมเดลข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer)

ใช้เพรดิเคต findRule() ใส่ค่าอาร์กิวเมนต์เป็นเซตว่าง ([ ]) ซึ่งการค้นหากฎทั้งหมดของข้อมูลผู้ป่วยมะเร็งเต้านมได้จำนวนกฎทั้งหมด 48 กฎ แสดงดังตารางที่ 4.11

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.11 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยมะเร็งเต้านม

No.	Rule
1	if breast_quad - left_up then class - recurrence_events
2	if breast_quad - left_low tumor_size - 0-4 then class - no_recurrence_events
3	if breast_quad - left_low tumor_size - 5-9 then class - no_recurrence_events
4	if breast_quad - left_low tumor_size - 10-14 then class - no_recurrence_events
5	if breast_quad - left_low tumor_size - 15-19 then class - no_recurrence_events
6	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 40-49 then class - recurrence_events

ตารางที่ 4.11 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยมะเร็งเต้านม (ต่อ)

No.	Rule
7	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 50-59 menopause - lt40 then class - recurrence_events
8	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 50-59 menopause - premeno then class - no_recurrence_events
9	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 60-69 then class - recurrence_events
10	if breast_quad - left_low tumor_size - 20-24 node_caps - no then class - no_recurrence_events
11	if breast_quad - left_low tumor_size - 25-29 deg_malig - 1 then class - no_recurrence_events
12	if breast_quad - left_low tumor_size - 25-29 deg_malig - 2 irradiat - yes age - 40-49 then class - recurrence_events
13	if breast_quad - left_low tumor_size - 25-29 deg_malig - 2 irradiat - yes age - 50-59 then class - no_recurrence_events
14	if breast_quad - left_low tumor_size - 25-29 deg_malig - 2 irradiat - no then class - no_recurrence_events
15	if breast_quad - left_low tumor_size - 25-29 deg_malig - 3 then class - recurrence_events
16	if breast_quad - left_low tumor_size - 30-34 age - 30-39 then class - no_recurrence_events
17	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 1 irradiat - yes then class - recurrence_events
18	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 1 irradiat - no then class - no_recurrence_events
19	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 2 then class - no_recurrence_events
20	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 3 then class - recurrence_events

ตารางที่ 4.11 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยมะเร็งเต้านม (ต่อ)

No.	Rule
21	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 0-2 then class - no_recurrence_events
22	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - ge40 then class - recurrence_events
23	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - premeno then class - no_recurrence_events
24	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 6-8 then class - no_recurrence_events
25	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 9-11 then class - no_recurrence_events
26	if breast_quad - left_low tumor_size - 30-34 age - 60-69 then class - no_recurrence_events
27	if breast_quad - left_low tumor_size - 35-39 age - 30-39 then class - recurrence_events
28	if breast_quad - left_low tumor_size - 35-39 age - 40-49 then class - no_recurrence_events
29	if breast_quad - left_low tumor_size - 35-39 age - 50-59 then class - no_recurrence_events
30	if breast_quad - left_low tumor_size - 40-44 inv_nodes - 0-2 then class - no_recurrence_events
31	if breast_quad - left_low tumor_size - 40-44 inv_nodes - 3-5 then class - no_recurrence_events
32	if breast_quad - left_low tumor_size - 40-44 inv_nodes - 6-8 then class - recurrence_events
33	if breast_quad - left_low tumor_size - 45-49 then class - no_recurrence_events
34	if breast_quad - left_low tumor_size - 50-54 breast - left then class - no_recurrence_events
35	if breast_quad - left_low tumor_size - 50-54 breast - right then class - recurrence_events

ตารางที่ 4.11 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยมะเร็งเต้านม (ต่อ)

No.	Rule
36	if breast_quad - right_up node_caps - yes deg_malig - 2 inv_nodes - 0-2 age - 40-49 then class - no_recurrence_events
37	if breast_quad - right_up node_caps - yes deg_malig - 2 inv_nodes - 0-2 age - 60-69 then class - recurrence_events
38	if breast_quad - right_up node_caps - yes deg_malig - 2 inv_nodes - 3-5 then class - recurrence_events
39	if breast_quad - right_up node_caps - yes deg_malig - 2 inv_nodes - 6-8 age - 30-39 then class - no_recurrence_events
40	if breast_quad - right_up node_caps - yes deg_malig - 2 inv_nodes - 6-8 age - 50-59 then class - recurrence_events
41	if breast_quad - right_up node_caps - yes deg_malig - 2 inv_nodes - 6-8 age - 60-69 then class - no_recurrence_events
42	if breast_quad - right_up node_caps - yes deg_malig - 2 inv_nodes - 9-11 then class - no_recurrence_events
43	if breast_quad - right_up node_caps - yes deg_malig - 3 then class - recurrence_events
44	if breast_quad - right_up node_caps - no inv_nodes - 0-2 then class - no_recurrence_events
45	if breast_quad - right_up node_caps - no inv_nodes - 3-5 age - 30-39 then class - no_recurrence_events
46	if breast_quad - right_up node_caps - no inv_nodes - 3-5 age - 60-69 then class - recurrence_events
47	if breast_quad - right_up node_caps - no inv_nodes - 12-14 then class - no_recurrence_events
48	if breast_quad - central then class - recurrence_events

(2) ค้นหากฎด้วยเพรดิเคต findRule() เมื่อใส่เงื่อนไขให้กับอาร์กิวเมนต์ ทดสอบด้วยข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer)

- กำหนดให้อาร์กิวเมนต์ของการเควียรีมีจำนวน 1 ตัวแปร

กำหนด X เป็นตัวแปร โดยให้ค่าของ X เป็น **breast_quad - right_up** ดังนั้นคำสั่งการเควียรี คือ **findRule([breast_quad - right_up])** ผลลัพธ์ที่ได้ คือ จำนวนกฎมีจำนวนน้อยลงจากการค้นหากฎทั้งหมด คือ มีจำนวน 12 กฎ จากทั้งหมด 48 กฎ แสดงดังตารางที่ 4.12 จากการเควียรีด้วยเงื่อนไขนี้ มีความหมายว่า แสดงเฉพาะกฎที่มีข้อมูล *บริเวณทางด้านบนขวา (breast_quad - right_up) ของเต้านมที่เกิดก้อนเนื้อมะเร็ง ส่งผลให้เกิดปัจจัยใดบ้างที่จะทำให้ผู้ป่วยมีโอกาสหรือไม่มีโอกาสกลับมาเป็นมะเร็งเต้านม*

ตารางที่ 4.12 กฎการจำแนกที่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up อยู่ในกฎ

No.	Rule
1	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 0-2 age - 40-49 then class - no_recurrence_events
2	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 0-2 age - 60-69 then class - recurrence_events
3	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 3-5 then class - recurrence_events
4	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 6-8 age - 30-39 then class - no_recurrence_events
5	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 6-8 age - 50-59 then class - recurrence_events
6	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 6-8 age - 60-69 then class - no_recurrence_events
7	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 9-11 then class - no_recurrence_events
8	if **breast_quad - right_up** node_caps - yes deg_malig - 3 then class - recurrence_events

ตารางที่ 4.12 กฎการจำแนกที่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up อยู่ในกฎ (ต่อ)

No.	Rule
9	if **breast_quad - right_up** node_caps - no inv_nodes - 0-2 then class - no_recurrence_events
10	if **breast_quad - right_up** node_caps - no inv_nodes - 3-5 age - 30-39 then class - no_recurrence_events
11	if **breast_quad - right_up** node_caps - no inv_nodes - 3-5 age - 60-69 then class - recurrence_events
12	if **breast_quad - right_up** node_caps - no inv_nodes - 12-14 then class - no_recurrence_events

กำหนดค่าของ X ใส่ในรูปของเงื่อนไข NOT โดยกำหนดให้เป็น **\+breast_quad - right_up** ดังนั้นคำสั่งการเควียรี คือ **findRule([\+breast_quad - right_up])** ซึ่งจะแสดงกฎที่ไม่มี breast_quad - right_up อยู่ในกฎ ซึ่งผลลัพธ์ที่ได้มีจำนวน 36 กฎ ซึ่งจำนวนของกฎลดน้อยลงจากการค้นหากฎทั้งหมดโดยไม่ใส่เงื่อนไขเช่นกัน แสดงดังตารางที่ 4.13

ตารางที่ 4.13 กฎที่ไม่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up อยู่ในกฎ

No.	Rule
1	if breast_quad - left_up then class - recurrence_events
2	if breast_quad - left_low tumor_size - 0-4 then class - no_recurrence_events
3	if breast_quad - left_low tumor_size - 5-9 then class - no_recurrence_events
4	if breast_quad - left_low tumor_size - 10-14 then class - no_recurrence_events
5	if breast_quad - left_low tumor_size - 15-19 then class - no_recurrence_events
6	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 40-49 then class - recurrence_events

ตารางที่ 4.13 กฎที่ไม่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up อยู่ในกฎ (ต่อ)

No.	Rule
7	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 50-59 menopause - lt40 then class - recurrence_events
8	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 50-59 menopause - premeno then class - no_recurrence_events
9	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 60-69 then class - recurrence_events
10	if breast_quad - left_low tumor_size - 20-24 node_caps - no then class - no_recurrence_events
11	if breast_quad - left_low tumor_size - 25-29 deg_malig - 1 then class - no_recurrence_events
12	if breast_quad - left_low tumor_size - 25-29 deg_malig - 2 irradiat - yes age - 40-49 then class - recurrence_events
13	if breast_quad - left_low tumor_size - 25-29 deg_malig - 2 irradiat - yes age - 50-59 then class - no_recurrence_events
14	if breast_quad - left_low tumor_size - 25-29 deg_malig - 2 irradiat - no then class - no_recurrence_events
15	if breast_quad - left_low tumor_size - 25-29 deg_malig - 3 then class - recurrence_events
16	if breast_quad - left_low tumor_size - 30-34 age - 30-39 then class - no_recurrence_events
17	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 1 irradiat - yes then class - recurrence_events
18	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 1 irradiat - no then class - no_recurrence_events
19	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 2 then class - no_recurrence_events
20	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 3 then class - recurrence_events

ตารางที่ 4.13 กฎที่ไม่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up อยู่ในกฎ (ต่อ)

No.	Rule
21	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 0-2 then class - no_recurrence_events
22	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - ge40 then class - recurrence_events
23	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - premeno then class - no_recurrence_events
24	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 6-8 then class - no_recurrence_events
25	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 9-11 then class - no_recurrence_events
26	if breast_quad - left_low tumor_size - 30-34 age - 60-69 then class - no_recurrence_events
27	if breast_quad - left_low tumor_size - 35-39 age - 30-39 then class - recurrence_events
28	if breast_quad - left_low tumor_size - 35-39 age - 40-49 then class - no_recurrence_events
29	if breast_quad - left_low tumor_size - 35-39 age - 50-59 then class - no_recurrence_events
30	if breast_quad - left_low tumor_size - 40-44 inv_nodes - 0-2 then class - no_recurrence_events
31	if breast_quad - left_low tumor_size - 40-44 inv_nodes - 3-5 then class - no_recurrence_events
32	if breast_quad - left_low tumor_size - 40-44 inv_nodes - 6-8 then class - recurrence_events

ตารางที่ 4.13 กฎที่ไม่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up อยู่ในกฎ (ต่อ)

No.	Rule
33	if breast_quad - left_low tumor_size - 45-49 then class - no_recurrence_events
34	if breast_quad - left_low tumor_size - 50-54 breast - left then class - no_recurrence_events
35	if breast_quad - left_low tumor_size - 50-54 breast - right then class - recurrence_events
36	if breast_quad - central then class - recurrence_events

- กำหนดให้อาร์กิวเมนต์ของการเควียรีมีจำนวน 2 ตัวแปรหรือมากกว่า

กำหนดตัวแปรตัวที่ 1 ให้เป็นเหมือนเดิม คือ X1 = **breast_quad - right_up** และเพิ่มเงื่อนไขให้กับตัวแปรตัวที่ 2 คือ X2 = **class - no_recurrence_events** ดังนั้นคำสั่งในการเควียรี คือ **findRule([breast_quad - right_up , class - no_recurrence_events])** กฎที่ได้จากการเควียรีเงื่อนไขนี้มีจำนวน 7 กฎ แสดงดังตารางที่ 4.14 ซึ่งจำนวนของกฎลดลงจากการเควียรีด้วย 1 ตัวแปร ลดลงจาก 48 กฎ ซึ่งความหมายของการเควียรีด้วยเงื่อนไขนี้ คือ *บริเวณทางด้านบนขวา (breast_quad - right_up) ของเต้านมที่เกิดก้อนเนื้อมะเร็ง ส่งผลให้เกิดปัจจัยใดบ้างที่จะทำให้ผู้ป่วยไม่มีโอกาสกลับมาเป็นมะเร็งเต้านม*

ตารางที่ 4.14 กฎที่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up และมีแอททริบิวต์ class มีค่าเป็น no_recurrence_events อยู่ในกฎ

No.	Rule
1	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 0-2 age - 40-49 then **class - no_recurrence_events**
2	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 6-8 age - 30-39 then **class - no_recurrence_events**
3	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 6-8 age - 60-69 then **class - no_recurrence_events**

ตารางที่ 4.14 กฎที่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up และมีแอททริบิวต์ class มีค่าเป็น no_recurrence_events อยู่ในกฎ (ต่อ)

No.	Rule
4	if **breast_quad - right_up** node_caps - yes deg_malig - 2 inv_nodes - 9-11 then **class - no_recurrence_events**
5	if **breast_quad - right_up** node_caps - no inv_nodes - 0-2 then **class - no_recurrence_events**
6	if **breast_quad - right_up** node_caps - no inv_nodes - 3-5 age - 30-39 then **class - no_recurrence_events**
7	if **breast_quad - right_up** node_caps - no inv_nodes - 12-14 then **class - no_recurrence_events**

กำหนดค่าของ X1 หรือ X2 ใส่ในรูปเงื่อนไข NOT กำหนดให้ X1 = **\+breast_quad - right_up** และ X2 = **\+class - no_recurrence_events** ดังนั้นคำสั่งในการเควียรี คือ **findRule([\+breast_quad - right_up , \+class - no_recurrence_events])** ซึ่งจะแสดงกฎที่ไม่มี breast_quad - right_up และ class - no_recurrence_events อยู่ในกฎ ซึ่งผลลัพธ์ที่ได้มีจำนวน 13 กฎ ซึ่งจำนวนของกฎลดน้อยลงจากการค้นหากฎทั้งหมดโดยไม่ใส่เงื่อนไขเช่นกัน แสดงดังตารางที่ 4.15

ตารางที่ 4.15 กฎที่ไม่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up และไม่มีแอททริบิวต์ class มีค่าเป็น no_recurrence_events อยู่ในกฎ

No.	Rule
1	if breast_quad - left_up then class - recurrence_events
2	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 40-49 then class - recurrence_events
3	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 50-59 menopause - lt40 then class - recurrence_events

ตารางที่ 4.15 กฎที่ไม่มีแอททริบิวต์ breast_quad มีค่าเป็น right_up และไม่มีแอททริบิวต์ class มีค่าเป็น no_recurrence_events อยู่ในกฎ (ต่อ)

No.	Rule
4	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 60-69 then class - recurrence_events
5	if breast_quad - left_low tumor_size - 25-29 deg_malig - 2 irradiat - yes age - 40-49 then class - recurrence_events
6	if breast_quad - left_low tumor_size - 25-29 deg_malig - 3 then class - recurrence_events
7	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 1 irradiat – yes then class - recurrence_events
8	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 3 then class - recurrence_events
9	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - ge40 then class - recurrence_events
10	if breast_quad - left_low tumor_size - 35-39 age - 30-39 then class - recurrence_events
11	if breast_quad - left_low tumor_size - 40-44 inv_nodes - 6-8 then class - recurrence_events
12	if breast_quad - left_low tumor_size - 50-54 breast - right then class - recurrence_events
13	if breast_quad - central then class - recurrence_events

(3) ค้นหากฎด้วยเพรดิเคต findRuleOr() เมื่อใส่เงื่อนไขให้กับอาร์กิวเมนต์ ทดสอบด้วยข้อมูลผู้ป่วยมะเร็งเต้านม (Breast Cancer)

- กำหนดให้อาร์กิวเมนต์ของการเควียรีมีจำนวน 2 ตัวแปรหรือมากกว่า

กำหนดตัวแปรตัวที่ 1 คือ [X1] = **breast_quad - left_low , tumor_size - "30-34"** ตัวแปรตัวที่ 2 คือ [X2] = **breast_quad - left_low , tumor_size - "30-34" , age - "50-59"** ดังนั้นคำสั่งของการเควียรี จะได้ **findRuleOr([[breast_quad - left_low , tumor_size - "30-34"],[breast_quad - left_low , tumor_size - "30-34" , age - "50-59"]])** ผลลัพธ์การแสดงผลของกฎมีจำนวนน้อยลงถ้าเทียบจากกฎทั้งหมด คือ 16 กฎ แต่บางส่วนของกฎแสดงผลซ้ำกันเหมือนกับ

ข้อมูลผู้ป่วยโรคตับอักเสบ ดังตารางที่ 4.16 โดยกฎลำดับที่ 6-10 (เงื่อนไขของตัวแปรที่1) และกฎลำดับที่ 12-16 (เงื่อนไขของตัวแปรที่ 2) ซ้ำกัน

ตารางที่ 4.16 กฎที่มีแอททริบิวต์ breast_quad มีค่าเป็น left_low และแอททริบิวต์ tumor_size มีค่าเป็น 30-34 รวมกับ ที่มีแอททริบิวต์ breast_quad มีค่าเป็น left_low และแอททริบิวต์ tumor_size มีค่าเป็น 30-34 และแอททริบิวต์ age มีค่าเป็น 50-59

No.	Rule
1	if breast_quad - left_low tumor_size - 30-34 age - 30-39 then class - no_recurrence_events
2	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 1 irradiat - yes then class - recurrence_events
3	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 1 irradiat - no then class - no_recurrence_events
4	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 2 then class - no_recurrence_events
5	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 3 then class - recurrence_events
6	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 0-2 then class - no_recurrence_events
7	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - ge40 then class - recurrence_events
8	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - premeno then class - no_recurrence_events
9	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 6-8 then class - no_recurrence_events
10	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 9-11 then class - no_recurrence_events
11	if breast_quad - left_low tumor_size - 30-34 age - 60-69 then class - no_recurrence_events

ตารางที่ 4.16 กฎที่มีแอททริบิวต์ breast_quad มีค่าเป็น left_low และแอททริบิวต์ tumor_size มีค่าเป็น 30-34 รวมกับ ที่มีแอททริบิวต์ breast_quad มีค่าเป็น left_low และแอททริบิวต์ tumor_size มีค่าเป็น 30-34 และแอททริบิวต์ age มีค่าเป็น 50-59 (ต่อ)

No.	Rule
12	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 0-2 then class - no_recurrence_events
13	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - ge40 then class - recurrence_events
14	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - premeno then class - no_recurrence_events
15	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 6-8 then class - no_recurrence_events
16	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 9-11 then class - no_recurrence_events

- ปรับปรุงโปรแกรมไม่ให้แสดงกฎซ้ำกัน เมื่อกำหนดให้อาร์กิวเมนต์ของการเควียรีมีจำนวน 2 ตัวแปรหรือมากกว่า

จากการเควียรีด้วยคำสั่ง **findRuleOr([[breast_quad - left_low , tumor_size - "30-34"],[breast_quad - left_low , tumor_size - "30-34" , age - "50-59"]])** ซึ่งได้ผลลัพธ์ของกฎที่ซ้ำกัน และปรับปรุงทำให้กฎที่ได้มีจำนวนลดลงจากเดิม ได้ผลลัพธ์ดังตารางที่ 4.17 จะเห็นได้ว่าจำนวนกฎที่ได้มีจำนวนที่น้อยลงกว่าเดิม คือ 11 กฎ จาก 16 กฎ

ตารางที่ 4.17 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน

No.	Rule
1	if breast_quad - left_low tumor_size - 30-34 age - 30-39 then class - no_recurrence_events
2	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 1 irradiat - yes then class - recurrence_events
3	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 1 irradiat - no then class - no_recurrence_events
4	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 2 then class - no_recurrence_events
5	if breast_quad - left_low tumor_size - 30-34 age - 40-49 deg_malig - 3 then class - recurrence_events
6	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 0-2 then class - no_recurrence_events
7	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - ge40 then class - recurrence_events
8	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 3-5 menopause - premeno then class - no_recurrence_events
9	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 6-8 then class - no_recurrence_events
10	if breast_quad - left_low tumor_size - 30-34 age - 50-59 inv_nodes - 9-11 then class - no_recurrence_events
11	if breast_quad - left_low tumor_size - 30-34 age - 60-69 then class - no_recurrence_events

4.3.3 การค้นหากฎจำแนกข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease)

(1) ค้นหากฎทั้งหมดด้วยเพรดิเคต findRule() ด้วยโมเดลข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease)

ใช้เพรดิเคต findRule() ใส่ค่าอาร์กิวเมนต์เป็นเซตว่าง ([ ]) ซึ่งการค้นหากฎทั้งหมดของข้อมูลผู้ป่วยโรคไทรอยด์ได้จำนวนกฎทั้งหมด 61 กฎ แสดงดังตารางที่ 4.18

ตารางที่ 4.18 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคไทรอยด์

No.	Rule
1	if sex - female sick - false referral_source - stmw then class - negative
2	if sex - female sick - false referral_source - svhc then class - negative
3	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - false psych - false then class - negative
4	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - false psych - true tumor - false then class - sick
5	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - false psych - true tumor - true then class - negative
6	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - true then class - sick
7	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - true thyroid_surgery - false then class - sick
8	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - true thyroid_surgery - true then class - negative
9	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - false then class - sick
10	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - true then class - negative
11	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - false then class - sick
12	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - true then class - negative
13	if sex - female sick - false referral_source - svhd on_thyroxine - true then class - negative

ตารางที่ 4.18 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคไทรอยด์ (ต่อ)

No.	Rule
14	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false pregnant - false lithium - false then class - sick
15	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false pregnant - false lithium - true then class - negative
16	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false pregnant - true then class - negative
17	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - true then class - negative
18	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - true then class - negative
19	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - true then class - negative
20	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - false pregnant - false query_on_thyroxine - false then class - sick

ตารางที่ 4.18 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคไทรอยด์ (ต่อ)

No.	Rule
21	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - false pregnant - false query_on_thyroxine - true then class - negative
22	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - false pregnant - true then class - negative
23	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - true then class - negative
24	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - true then class - negative
25	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - true then class - negative
26	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - true then class - negative
27	if sex - female sick - false referral_source - other query_hyperthyroid - true then class - negative
28	if sex - female sick - true query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false tumor - false query_on_thyroxine - false then class - sick
29	if sex - female sick - true query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false tumor - false query_on_thyroxine - true then class - negative
30	if sex - female sick - true query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false tumor - true then class - negative
31	if sex - female sick - true query_hypothyroid - false on_thyroxine - false query_hyperthyroid - true then class - negative

ตารางที่ 4.18 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคไทรอยด์ (ต่อ)

No.	Rule
32	if sex - female sick - true query_hypothyroid - false on_thyroxine - true then class - negative
33	if sex - female sick - true query_hypothyroid - true then class - negative
34	if sex - male referral_source - stmw then class - negative
35	if sex - male referral_source - svhc psych - false query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false lithium - false then class - sick
36	if sex - male referral_source - svhc psych - false query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false lithium - true then class - negative
37	if sex - male referral_source - svhc psych - false query_hypothyroid - false on_thyroxine - false query_hyperthyroid - true then class - negative
38	if sex - male referral_source - svhc psych - false query_hypothyroid - false on_thyroxine - true then class - negative
39	if sex - male referral_source - svhc psych - false query_hypothyroid - true then class - sick
40	if sex - male referral_source - svhc psych - true sick - false then class - negative
41	if sex - male referral_source - svhc psych - true sick - true then class - sick
42	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - false then class - sick
43	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - false hypopituitary - false then class - negative
44	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - false hypopituitary - true then class - sick
45	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - true then class - sick
46	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - true then class - negative
47	if sex - male referral_source - svi query_hyperthyroid - true then class - negative

ตารางที่ 4.18 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคไทรอยด์ (ต่อ)

No.	Rule
48	if sex - male referral_source - svhd query_hypothyroid - false then class - negative
49	if sex - male referral_source - svhd query_hypothyroid - true then class - sick
50	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - false on_thyroxine - false then class - sick
51	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - false then class - sick
52	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - true then class - negative
53	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - true then class - negative
54	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - true then class - negative
55	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - true then class - negative
56	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - true then class - negative
57	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - true then class - negative
58	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - false then class - negative

ตารางที่ 4.18 กฎการจำแนกทั้งหมดของข้อมูลผู้ป่วยโรคไทรอยด์ (ต่อ)

No.	Rule
59	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - true on_antithyroid_medication - false then class - sick
60	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - true on_antithyroid_medication - true then class - negative
61	if sex - male referral_source - other sick - true then class - negative

(2) ค้นหากฎด้วยเพรดิเคต findRule() เมื่อใส่เงื่อนไขให้กับอาร์กิวเมนต์ ทดสอบด้วยข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease)

- กำหนดให้อาร์กิวเมนต์ ของการเควียรีมีจำนวน 1 ตัวแปร

กำหนด X เป็นตัวแปร โดยให้ค่าของ X เป็น **pregnant – false** ดังนั้นคำสั่งการเควียรี คือ **findRule([pregnant - false])** ผลลัพธ์ที่ได้ คือ จำนวนกฎมีจำนวนน้อยลงจากการค้นหากฎทั้งหมด คือ มีจำนวน 4 กฎ จากทั้งหมด 61 กฎ แสดงดังตารางที่ 4.19 จากตัวอย่างแสดงเฉพาะกฎที่มีข้อมูล *ผู้ป่วยที่ไม่ได้ตั้งครรภ์ (pregnant - false) ส่งผลให้เกิดปัจจัยใดบ้างที่จะทำให้ผู้ป่วยมีโอกาสหรือไม่มีโอกาสเป็นโรคไทรอยด์*

ตารางที่ 4.19 กฎที่มีแอททริบิวต์ pregnant มีค่าเป็น false อยู่ในกฎ

No.	Rule
1	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false **pregnant - false** lithium - false then class - sick
2	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false **pregnant - false** lithium - true then class - negative

ตารางที่ 4.19 กฎที่มีแอททริบิวต์ pregnant มีค่าเป็น false อยู่ในกฎ (ต่อ)

No.	Rule
3	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - false **pregnant - false** query_on_thyroxine - false then class - sick
4	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - false **pregnant - false** query_on_thyroxine - true then class - negative

การกำหนดเงื่อนไขในรูปของ NOT ให้ค่าของ X เป็น \+ **tumor - false** ดังนั้นคำสั่งการเควียรี คือ **findRule([\+tumor - false])** ผลลัพธ์ที่ได้ คือ จำนวนกฎมีจำนวนน้อยลง คือ มีจำนวน 42 กฎ จากทั้งหมด 61 กฎ จากการเควียรีด้วยเงื่อนไขนี้ มีความหมายว่า แสดงเฉพาะกฎที่ไม่มีข้อมูลที่ปรากฏว่า *ผู้ป่วยที่ไม่ได้มีเนื้องอก (\+tumor - false) ส่งผลให้เกิดปัจจัยใดบ้างที่จะทำให้ผู้ป่วยมีโอกาสหรือไม่มีโอกาสเป็นโรคไทรอยด์* แสดงดังตารางที่ 4.20

ตารางที่ 4.20 กฎที่ไม่มีแอททริบิวต์ tumor มีค่าเป็น false อยู่ในกฎ

No.	Rule
1	if sex - female sick - false referral_source - stmw then class - negative
2	if sex - female sick - false referral_source - svhc then class - negative
3	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - false psych - false then class - negative
4	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - false psych - true tumor - true then class - negative

ตารางที่ 4.20 กฎที่ไม่มีแอททริบิวต์ tumor มีค่าเป็น false อยู่ในกฎ (ต่อ)

No.	Rule
5	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - true then class - sick
6	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - true thyroid_surgery - false then class - sick
7	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - true thyroid_surgery - true then class - negative
8	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - false then class - sick
9	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - true then class - negative
10	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - false then class - sick
11	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - true then class - negative
12	if sex - female sick - false referral_source - svhd on_thyroxine - true then class - negative
13	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - true then class - negative
14	if sex - female sick - false referral_source - other query_hyperthyroid - true then class - negative
15	if sex - female sick - true query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false tumor - true then class - negative
16	if sex - female sick - true query_hypothyroid - false on_thyroxine - false query_hyperthyroid - true then class - negative
17	if sex - female sick - true query_hypothyroid - false on_thyroxine - true then class - negative
18	if sex - female sick - true query_hypothyroid - true then class - negative
19	if sex - male referral_source - stmw then class - negative

ตารางที่ 4.20 กฎที่ไม่มีแอททริบิวต์ tumor มีค่าเป็น false อยู่ในกฎ (ต่อ)

No.	Rule
20	if sex - male referral_source - svhc psych - false query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false lithium - false then class - sick
21	if sex - male referral_source - svhc psych - false query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false lithium - true then class - negative
22	if sex - male referral_source - svhc psych - false query_hypothyroid - false on_thyroxine - false query_hyperthyroid - true then class - negative
23	if sex - male referral_source - svhc psych - false query_hypothyroid - false on_thyroxine - true then class - negative
24	if sex - male referral_source - svhc psych - false query_hypothyroid - true then class - sick
25	if sex - male referral_source - svhc psych - true sick - false then class - negative
26	if sex - male referral_source - svhc psych - true sick - true then class - sick
27	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - false then class - sick
28	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - false hypopituitary - false then class - negative
29	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - false hypopituitary - true then class - sick
30	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - true then class - sick
31	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - true then class - negative
32	if sex - male referral_source - svi query_hyperthyroid - true then class - negative
33	if sex - male referral_source - svhd query_hypothyroid - false then class - negative
34	if sex - male referral_source - svhd query_hypothyroid - true then class - sick
35	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - true then class - negative

ตารางที่ 4.20 กฎที่ไม่มีแอททริบิวต์ tumor มีค่าเป็น false อยู่ในกฎ (ต่อ)

No.	Rule
36	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - true then class - negative
37	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - true then class - negative
38	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - true then class - negative
39	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - false then class - negative
40	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - true on_antithyroid_medication - false then class - sick
41	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - true on_antithyroid_medication - true then class - negative
42	if sex - male referral_source - other sick - true then class - negative

- กำหนดให้อาร์กิวเมนต์ของการเควียรีมีจำนวน 2 ตัวแปรหรือมากกว่า

กำหนดตัวแปรทั้งหมด 4 ตัวแปร โดยตัวแปรตัวที่ 1 คือ X1 = **referral_source - other** ตัวแปรตัวที่ 2 คือ X2 = **query_hypothyroid - true** ตัวแปรตัวที่ 3 คือ X3 = **sex – female** ดังนั้น คำสั่งในการเควียรี คือ **findRule([referral_source - other,query_hypothyroid - true,sex - female])** กฎที่ได้จากการเควียรีเงื่อนไขนี้แสดงดังตารางที่ 4.21 โดยจำนวนของกฎลดลงคือ ได้กฎ 1 กฎ จากทั้งหมด 61 กฎ

ตารางที่ 4.21 กฎที่มีแอททริบิวต์ referral_source มีค่าเป็น other , แอททริบิวต์ query_hypothyroid มีค่าเป็น true และแอททริบิวต์ sex มีค่าเป็น female อยู่ในกฎ

No.	Rule
1	if **sex - female** sick - false **referral_source - other** query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - true **query_hypothyroid - true** then class - negative

กำหนดตัวแปรทั้งหมด 3 ตัว และกำหนดเงื่อนไขในรูปของ NOT โดยให้ค่าของ X1 = **\+referral_source - svhc** ตัวแปรตัวที่ 2 คือ X2 = **\+referral_source – svi** และตัวแปรตัวที่ 3 คือ X3 = **\+referral_source – other** ดังนั้นคำสั่งในการเควียรี คือ **findRule([\+referral_source - svhc , \+referral_source - svi , \+referral_source - other])** กฎที่ได้จากการเควียรีเงื่อนไขนี้แสดงดังตารางที่ 4.22 ซึ่งจำนวนของกฎลดลงคือ ได้ 13 กฎ จากทั้งหมด 61 กฎ

ตารางที่ 4.22 กฎที่ไม่มีแอททริบิวต์ referral_source มีค่าเป็น svhc , แอททริบิวต์ referral_source มีค่าเป็น svi และแอททริบิวต์ referral_source มีค่าเป็น other อยู่ในกฎ

No.	Rule
1	if sex - female sick - false referral_source - stmw then class - negative
2	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - false then class - sick
3	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - true then class - negative
4	if sex - female sick - false referral_source - svhd on_thyroxine - true then class - negative
5	if sex - female sick - true query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false tumor - false query_on_thyroxine - false then class - sick
6	if sex - female sick - true query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false tumor - false query_on_thyroxine - true then class - negative

ตารางที่ 4.22 กฎที่ไม่มีแอททริบิวต์ referral_source มีค่าเป็น svhc , แอททริบิวต์ referral_source มีค่าเป็น svi และแอททริบิวต์ referral_source มีค่าเป็น other อยู่ในกฎ (ต่อ)

No.	Rule
7	if sex - female sick - true query_hypothyroid - false on_thyroxine - false query_hyperthyroid - false tumor - true then class - negative
8	if sex - female sick - true query_hypothyroid - false on_thyroxine - false query_hyperthyroid - true then class - negative
9	if sex - female sick - true query_hypothyroid - false on_thyroxine - true then class - negative
10	if sex - female sick - true query_hypothyroid - true then class - negative
11	if sex - male referral_source - stmw then class - negative
12	if sex - male referral_source - svhd query_hypothyroid - false then class - negative
13	if sex - male referral_source - svhd query_hypothyroid - true then class - sick

(3) ค้นหากฎด้วยเพรดิเคต findRuleOr() เมื่อใส่เงื่อนไขให้กับอาร์กิวเมนต์ทดสอบด้วยข้อมูลผู้ป่วยโรคไทรอยด์ (Thyroid Disease)

- กำหนดให้อาร์กิวเมนต์ของการเควียรีมีจำนวน 2 ตัวแปรหรือมากกว่า

กำหนดตัวแปรโดยใส่เงื่อนไขในรูปของ NOT ผสมอยู่ด้วย โดยที่กำหนดตัวแปรตัวที่ 1 คือ [X1] = **\+referral_source - svhc , \+query_hypothyroid - false,class - sick** และตัวแปรตัวที่ 2 คือ [X2] = **\+query_hypothyroid - false** ดังนั้นคำสั่งของการเควียรีที่จะได้ **findRuleOr([[\+referral_source - svhc , \+query_hypothyroid - false,class - sick] , [\+query_hypothyroid - false]])** ผลลัพธ์การแสดงผลของกฎมีจำนวนมากกว่าการเควียรีแสดงผลกฎทั้งหมด คือ 49 กฎ และมีกฎที่ซ้ำกันเช่นกัน ดังตารางที่ 4.23 โดยกฎลำดับที่ 1-9 (เงื่อนไขของตัวแปรที่1) และกฎลำดับที่ 12,14,17,33,35-36,39-40,47 (เงื่อนไขของตัวแปรที่ 2) ซ้ำกัน

ตารางที่ 4.23 กฎที่ไม่มีแอททริบิวต์ referral_source มีค่าเป็น svhc และ query_hypothyroid มีค่าเป็น false และมีแอททริบิวต์ class มีค่าเป็น sick รวมกับกฎที่ไม่มีแอททริบิวต์ query_hypothyroid มีค่าเป็น false

No.	Rule
1	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - false then class - sick
2	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - false then class - sick
3	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false pregnant - false lithium - false then class - sick
4	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - false then class - sick
5	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - false hypopituitary - true then class - sick
6	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - true then class - sick
7	if sex - male referral_source - svhd query_hypothyroid - true then class - sick
8	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - false on_thyroxine - false then class - sick
9	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - true on_antithyroid_medication - false then class - sick
10	if sex - female sick - false referral_source - stmw then class - negative
11	if sex - female sick - false referral_source - svhc then class - negative
12	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - false then class - sick

ตารางที่ 4.23 กฎที่ไม่มีแอททริบิวต์ referral_source มีค่าเป็น svhc และ query_hypothyroid มีค่าเป็น false และมีแอททริบิวต์ class มีค่าเป็น sick รวมกับกฎที่ไม่มีแอททริบิวต์ query_hypothyroid มีค่าเป็น false (ต่อ)

No.	Rule
13	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - true then class - negative
14	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - false then class - sick
15	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - true then class - negative
16	if sex - female sick - false referral_source - svhd on_thyroxine - true then class - negative
17	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false pregnant - false lithium - false then class - sick
18	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false pregnant - false lithium - true then class - negative
19	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false pregnant - true then class - negative
20	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - true then class - negative

ตารางที่ 4.23 กฎที่ไม่มีแอททริบิวต์ referral_source มีค่าเป็น svhc และ query_hypothyroid มีค่าเป็น false และมีแอททริบิวต์ class มีค่าเป็น sick รวมกับกฎที่ไม่มีแอททริบิวต์ query_hypothyroid มีค่าเป็น false (ต่อ)

No.	Rule
21	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - true then class - negative
22	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - true then class - negative
23	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - true then class - negative
24	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - true then class - negative
25	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - true then class - negative
26	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - true then class - negative
27	if sex - female sick - false referral_source - other query_hyperthyroid - true then class - negative
28	if sex - female sick - true query_hypothyroid - true then class - negative
29	if sex - male referral_source - stmw then class - negative
30	if sex - male referral_source - svhc psych - false query_hypothyroid - true then class - sick
31	if sex - male referral_source - svhc psych - true sick - false then class - negative
32	if sex - male referral_source - svhc psych - true sick - true then class - sick

ตารางที่ 4.23 กฎที่ไม่มีแอททริบิวต์ referral_source มีค่าเป็น svhc และ query_hypothyroid มีค่าเป็น false และมีแอททริบิวต์ class มีค่าเป็น sick รวมกับกฎที่ไม่มีแอททริบิวต์ query_hypothyroid มีค่าเป็น false (ต่อ)

No.	Rule
33	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - false then class - sick
34	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - false hypopituitary - false then class - negative
35	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - false hypopituitary - true then class - sick
36	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - true then class - sick
37	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - true then class - negative
38	if sex - male referral_source - svi query_hyperthyroid - true then class - negative
39	if sex - male referral_source - svhd query_hypothyroid - true then class - sick
40	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - false on_thyroxine - false then class - sick
41	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - true then class - negative
42	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - true then class - negative
43	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - true then class - negative

ตารางที่ 4.23 กฎที่ไม่มีแอททริบิวต์ referral_source มีค่าเป็น svhc และ query_hypothyroid มีค่าเป็น false และมีแอททริบิวต์ class มีค่าเป็น sick รวมกับกฎที่ไม่มีแอททริบิวต์ query_hypothyroid มีค่าเป็น false (ต่อ)

No.	Rule
44	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - true then class - negative
45	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - true then class - negative
46	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - true then class - negative
47	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - true on_antithyroid_medication - false then class - sick
48	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - true on_antithyroid_medication - true then class - negative
49	if sex - male referral_source - other sick - true then class - negative

- ปรับปรุงโปรแกรมไม่ให้แสดงกฎซ้ำกัน เมื่อกำหนดให้อาร์กิวเมนต์ของการเควียรีมีจำนวน 2 ตัวแปรหรือมากกว่า

จากการเควียรีด้วยคำสั่ง **findRuleOr([[\+referral_source - svhc , \+query_hypothyroid - false,class - sick],[\+query_hypothyroid - false]])** ซึ่งได้ผลลัพธ์ของกฎที่ซ้ำกัน และปรับปรุงทำให้กฎที่ได้มีจำนวนลดลงจากเดิม ดังตารางที่ 4.24 จะเห็นได้ว่าจำนวนกฎที่ได้มีจำนวนที่น้อยลงกว่าเดิม คือ 40 กฎ จากเดิมที่มี 49 กฎ

ตารางที่ 4.24 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน

No.	Rule
1	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false pregnant - false lithium - false then class - sick
2	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false pregnant - false lithium - true then class – negative
3	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - false pregnant - true then class - negative
4	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - false psych - true then class - negative
5	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - false goitre - true then class - negative
6	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - false query_on_thyroxine - true then class - negative
7	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - true then class - negative
8	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - false on_antithyroid_medication - true then class - negative
9	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - false thyroid_surgery - true then class - negative

ตารางที่ 4.24 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน (ต่อ)

No.	Rule
10	if sex - female sick - false referral_source - other query_hyperthyroid - false tumor - true then class - negative
11	if sex - female sick - false referral_source - other query_hyperthyroid - true then class - negative
12	if sex - female sick - false referral_source - stmw then class - negative
13	if sex - female sick - false referral_source - svhc then class - negative
14	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - false then class - sick
15	if sex - female sick - false referral_source - svhd on_thyroxine - false query_hyperthyroid - true then class - negative
16	if sex - female sick - false referral_source - svhd on_thyroxine - true then class - negative
17	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - false then class - sick
18	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - true then class - negative
19	if sex - female sick - true query_hypothyroid - true then class - negative
20	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - false on_thyroxine - false then class - sick
21	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - false on_thyroxine - true query_hypothyroid - true then class - negative
22	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - false on_antithyroid_medication - true then class - negative

ตารางที่ 4.24 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน (ต่อ)

No.	Rule
23	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - false tumor - true then class - negative
24	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - false goitre - true then class - negative
25	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - false psych - true then class - negative
26	if sex - male referral_source - other sick - false query_hyperthyroid - false thyroid_surgery - true then class - negative
27	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - true on_antithyroid_medication - false then class - sick
28	if sex - male referral_source - other sick - false query_hyperthyroid - true query_hypothyroid - true on_antithyroid_medication - true then class - negative
29	if sex - male referral_source - other sick - true then class - negative
30	if sex - male referral_source - stmw then class - negative
31	if sex - male referral_source - svhc psych - false query_hypothyroid - true then class - sick
32	if sex - male referral_source - svhc psych - true sick - false then class - negative
33	if sex - male referral_source - svhc psych - true sick - true then class - sick
34	if sex - male referral_source - svhd query_hypothyroid - true then class - sick
35	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - false then class - sick
36	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - false hypopituitary - false then class - negative
37	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - false hypopituitary - true then class - sick

ตารางที่ 4.24 กฎเมื่อโปรแกรมถูกปรับปรุงไม่ให้แสดงกฎที่ซ้ำกัน (ต่อ)

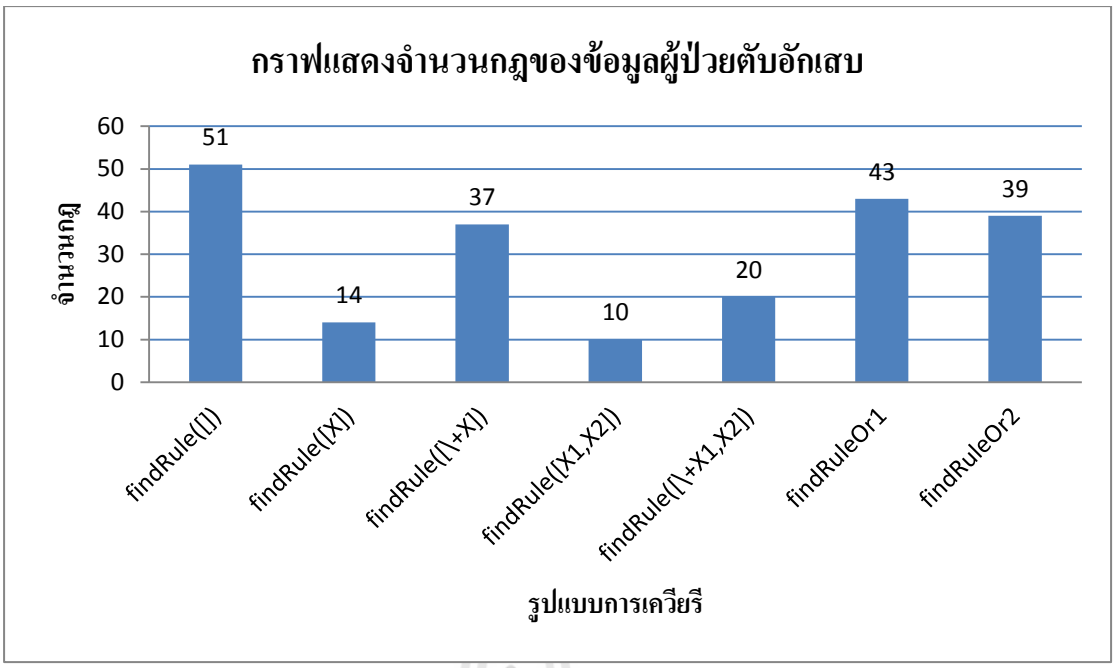
No.	Rule
38	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - false query_on_thyroxine - true goitre - true then class - sick
39	if sex - male referral_source - svi query_hyperthyroid - false on_thyroxine - true then class - negative
40	if sex - male referral_source - svi query_hyperthyroid - true then class - negative

4.4 ผลของการเปรียบเทียบเวลาและจำนวนกฎสำหรับการเควียรีด้วยคำสั่งที่กำหนด

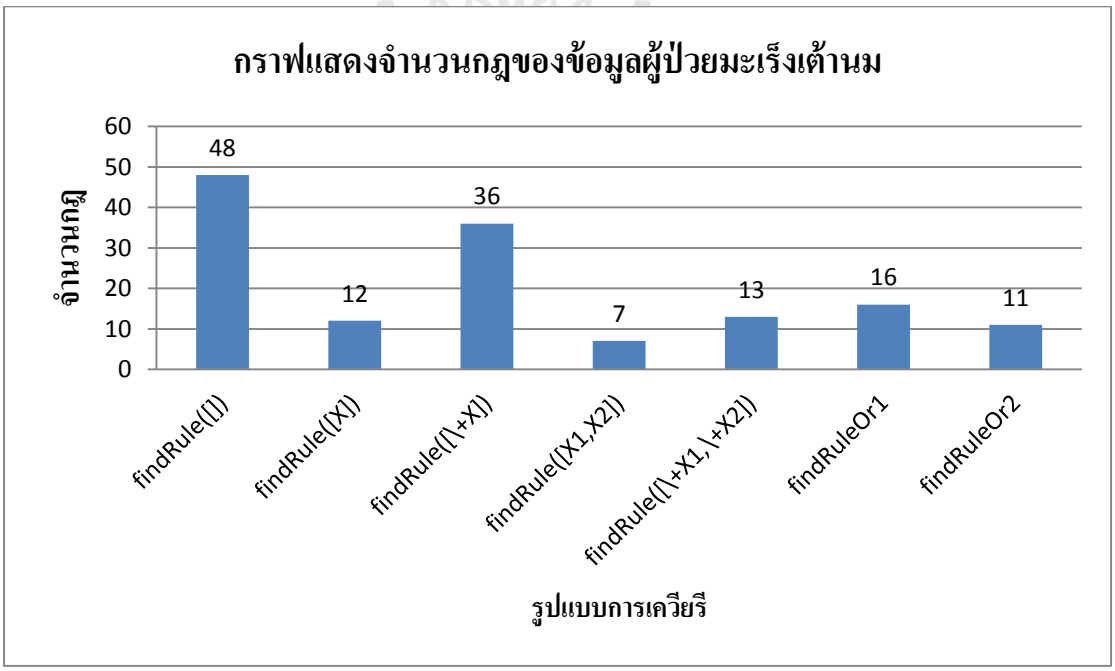
4.4.1 เปรียบเทียบจำนวนกฎ

จากการทดสอบกับข้อมูลสามชุด ด้วยคำสั่งการเควียรีในรูปแบบและเงื่อนไขที่กำหนด เมื่อนำจำนวนข้อมูลมาเปรียบเทียบกันแล้ว พบว่า การเควียรีไม่ว่าจะใช้เงื่อนไขในรูปแบบใด เช่น ใช้เพรดิเคต findRule หรือ findRuleOr ที่อาจจะมี 1 เงื่อนไขหรือมากกว่า 1 เงื่อนไขเป็นตัวกำหนด ก็จะได้จำนวนของกฎที่ลดลงทุกครั้งเมื่อเปรียบเทียบกับการเควียรีแสดงผลกฎทั้งหมด และจำนวนกฎที่ได้จากการเควียรี findRuleOr ในแบบที่ 1 จะมีจำนวนมากกว่าแบบที่ 2 ซึ่งเป็นรูปแบบที่ปรับปรุงโปรแกรมแล้ว แสดงดังรูปที่ 4.14 , 4.15 และ 4.16

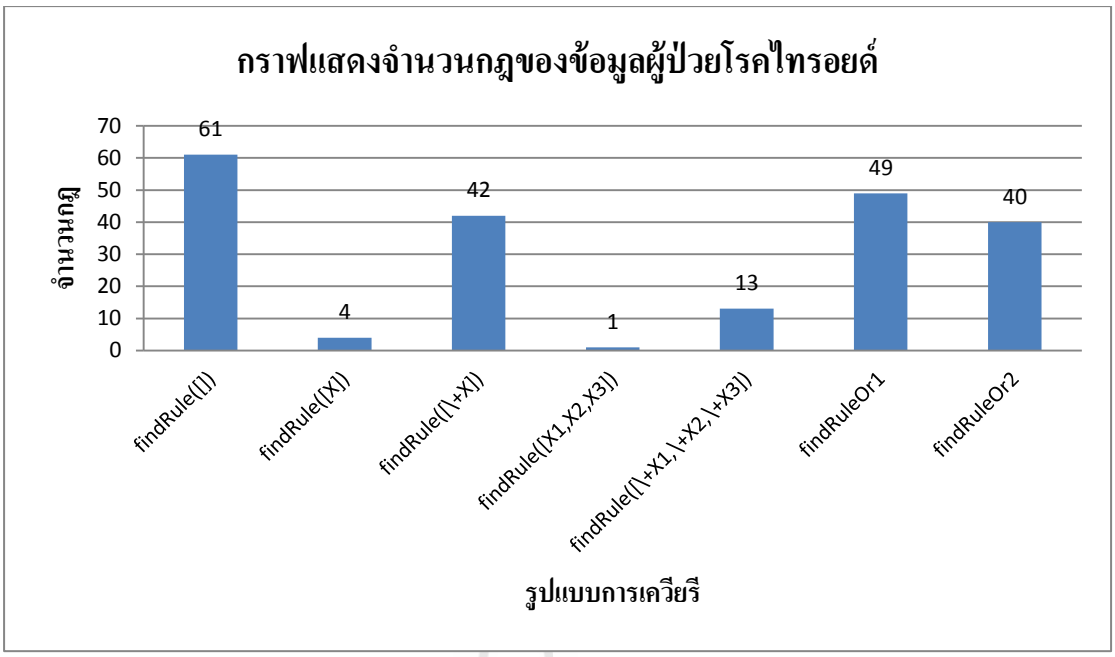
มหาวิทยาลัยเทคโนโลยีสุรนารี



รูปที่ 4.14 กราฟแสดงจำนวนของกฎของข้อมูลผู้ป่วยตับอักเสบที่ได้จากการเควียรีด้วยคำสั่งต่าง ๆ



รูปที่ 4.15 กราฟแสดงจำนวนของกฎของข้อมูลผู้ป่วยมะเร็งเต้านมที่ได้จากการเควียรีด้วยคำสั่งต่าง ๆ



รูปที่ 4.16 กราฟแสดงจำนวนของกฎของข้อมูลผู้ป่วยโรคไทรอยด์
ที่ได้จากการเควียรีด้วยคำสั่งต่าง ๆ

4.4.2 เปรียบเทียบเวลา

การเปรียบเทียบเวลาในการทดสอบกับทั้งสามข้อมูลนั้น จะทดลองการเควียรีด้วยรูปแบบคำสั่งและเงื่อนไขต่าง ๆ ด้วยจำนวนครั้งที่เท่ากัน คือ 10 ครั้ง จากนั้นจะทำการเฉลี่ยค่าเวลา (AVG) ของการทดลองทั้งสิบครั้ง ดังแสดงตารางที่ 4.25, 4.26 และ 4.27

จากตารางที่ 4.25, 4.26 และ 4.27 ผลของเวลาที่ได้นั้น จะเห็นว่ามีเวลาที่ทดสอบในแต่ละครั้งและเวลาที่เฉลี่ยแล้วเป็นเวลาที่น้อยมาก เนื่องจากโปรแกรมได้จำแนกข้อมูลและบันทึกไว้ในหน่วยความจำไว้แล้ว เพื่อรอคำสั่งการเควียรีตามเงื่อนไขที่ผู้ใช้กำหนด จึงไม่จำเป็นต้องเสียเวลาจำแนกข้อมูลใหม่ในแต่ละครั้งของการเควียรี ทำให้สะดวกและรวดเร็วในการแสดงผล แต่เวลานั้นจะไม่คงที่ เนื่องจากการทดสอบในแต่ละครั้งอาจจะมีผลจากโปรแกรมอื่น ๆ ที่ทำงานในขณะนั้น

4.4.3 เปรียบเทียบความถูกต้อง

การทดสอบเพื่อยืนยันความถูกต้องของการจำแนกข้อมูลด้วยอัลกอริทึม ID3UP ได้ทำการเปรียบเทียบผลลัพธ์ระหว่างโปรแกรม ID3 ของซอฟต์แวร์ Weka 3.6 และโปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ (ID3UP) กล่าวคือ การจำแนกข้อมูลทั้ง 3 ชุด ให้ผลลัพธ์ที่ตรงกันจึงมีความถูกต้อง 100 เปอร์เซ็นต์ และจำนวนกฎในการจำแนกข้อมูลมีจำนวนเท่ากัน ดังนี้

<u>ข้อมูลผู้ป่วยตับอักเสบ</u>

Weka 3.6 - ID3	โปรแกรม ID3UP
51 กฎ	51 กฎ

<u>ข้อมูลผู้ป่วยมะเร็งเต้านม</u>

Weka 3.6 - ID3	โปรแกรม ID3UP
48 กฎ	48 กฎ

สำนักวิทยาลัยเทคโนโลยีสุรนารี

ข้อมูลผู้ป่วยโรคไทรอยด์

Weka 3.6 - ID3	โปรแกรม ID3UP
61 กฎ	61 กฎ

เมื่อเปรียบเทียบความถูกต้องในการจำแนกข้อมูล ผลทดสอบมีความถูกต้องเป็นไปตามการจำแนกของอัลกอริทึม ID3 ในโปรแกรม Weka 3.6 โดยจะยกตัวอย่างเปรียบเทียบ 10 กฎแรกที่ได้จากการจำแนกข้อมูลระหว่างโปรแกรม Weka 3.6 และโปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ (Prolog) ดังนี้

ข้อมูลผู้ป่วยตับอักเสบ

```
ascites = yes
|  spiders = yes
|  |  steroid = yes: live
|  |  steroid = no
|  |  |  bilirubin = 0.1-1.0
|  |  |  |  age = 11-20: live
|  |  |  |  age = 21-30: live
|  |  |  |  age = 31-40
|  |  |  |  |  fatigue = yes
|  |  |  |  |  |  antivirals = yes
|  |  |  |  |  |  |  spleen_palpable = yes: die
|  |  |  |  |  |  |  spleen_palpable = no: live
|  |  |  |  |  |  antivirals = no: live
|  |  |  |  |  fatigue = no: live
|  |  |  |  age = 41-50
|  |  |  |  |  varices = yes: live
|  |  |  |  |  varices = no: die
|  |  |  |  age = 51-60: live
```

รูปที่ 4.17 10 กฎแรกของข้อมูลผู้ป่วยตับอักเสบที่ถูกจำแนกข้อมูลด้วยโปรแกรม Weka 3.6

ตารางที่ 4.28 10 กฎแรกของข้อมูลผู้ป่วยตับอักเสบที่ถูกจำแนกข้อมูลด้วยโปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ (Prolog)

No.	Rule
1	if ascites - yes spiders - yes steroid - yes then class - live
2	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 11-20 then class - live
3	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 21-30 then class - live
4	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - yes spleen_palpable - yes then class - die
5	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - yes spleen_palpable - no then class - live
6	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - yes antivirals - no then class - live
7	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 31-40 fatigue - no then class - live
8	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 41-50 varices - yes then class - live
9	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 41-50 varices - no then class - die
10	if ascites - yes spiders - yes steroid - no bilirubin - 0.1-1.0 age - 51-60 then class - live

ข้อมูลผู้ป่วยมะเร็งเต้านม

```
breast_quad = left_up: recurrence_events
breast_quad = left_low
|   tumor_size = 0-4: no_recurrence_events
|   tumor_size = 5-9: no_recurrence_events
|   tumor_size = 10-14: no_recurrence_events
|   tumor_size = 15-19: no_recurrence_events
|   tumor_size = 20-24
|   |   node_caps = yes
|   |   |   age = 40-49: recurrence_events
|   |   |   age = 50-59
|   |   |   |   menopause = lt40: recurrence_events
|   |   |   |   menopause = premeno: no_recurrence_events
|   |   |   age = 60-69: recurrence_events
|   |   node_caps = no: no_recurrence_events
```

รูปที่ 4.18 10 กฎแรกของข้อมูลผู้ป่วยมะเร็งเต้านมที่ถูกจำแนกข้อมูลด้วยโปรแกรม Weka 3.6

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.29 10 กฎแรกของข้อมูลผู้ป่วยมะเร็งเต้านมที่ถูกจำแนกข้อมูลด้วยโปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ (Prolog)

No.	Rule
1	if breast_quad - left_up then class - recurrence_events
2	if breast_quad - left_low tumor_size - 0-4 then class - no_recurrence_events
3	if breast_quad - left_low tumor_size - 5-9 then class - no_recurrence_events
4	if breast_quad - left_low tumor_size - 10-14 then class - no_recurrence_events
5	if breast_quad - left_low tumor_size - 15-19 then class - no_recurrence_events
6	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 40-49 then class - recurrence_events
7	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 50-59 menopause - lt40 then class - recurrence_events
8	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 50-59 menopause - premeno then class - no_recurrence_events
9	if breast_quad - left_low tumor_size - 20-24 node_caps - yes age - 60-69 then class - recurrence_events
10	if breast_quad - left_low tumor_size - 20-24 node_caps - no then class - no_recurrence_events

มหาวิทยาลัยเทคโนโลยีสุรนารี

ข้อมูลผู้ป่วยโรคไทรอยด์

```
sex = female
|  sick = false
|  |  referral_source = stmw: negative
|  |  referral_source = svhc: negative
|  |  referral_source = svi
|  |  |  query_hypothyroid = false
|  |  |  |  query_hyperthyroid = false
|  |  |  |  |  on_thyroxine = false
|  |  |  |  |  |  psych = false: negative
|  |  |  |  |  |  psych = true
|  |  |  |  |  |  |  tumor = false: sick
|  |  |  |  |  |  |  tumor = true: negative
|  |  |  |  |  on_thyroxine = true: sick
|  |  |  |  query_hyperthyroid = true
|  |  |  |  |  thyroid_surgery = false: sick
|  |  |  |  |  thyroid_surgery = true: negative
|  |  |  query_hypothyroid = true
|  |  |  |  on_thyroxine = false: sick
|  |  |  |  on_thyroxine = true: negative
```

รูปที่ 4.19 10 กฎแรกของข้อมูลผู้ป่วยโรคไทรอยด์ที่ถูกจำแนกข้อมูลด้วยโปรแกรม Weka 3.6

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.30 10 กฎแรกของข้อมูลผู้ป่วยโรคไทรอยด์ที่ถูกจำแนกข้อมูลด้วยโปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ (Prolog)

No.	Rule
1	if sex - female sick - false referral_source - stmw then class - negative
2	if sex - female sick - false referral_source - svhc then class - negative
3	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - false psych - false then class - negative
4	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - false psych - true tumor - false then class - sick
5	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - false psych - true tumor - true then class - negative
6	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - false on_thyroxine - true then class - sick
7	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - true thyroid_surgery - false then class - sick
8	if sex - female sick - false referral_source - svi query_hypothyroid - false query_hyperthyroid - true thyroid_surgery - true then class - negative
9	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - false then class - sick
10	if sex - female sick - false referral_source - svi query_hypothyroid - true on_thyroxine - true then class - negative

การคัดเลือกกฎเฉพาะเงื่อนไขของผู้ใช้ โปรแกรม ID3 ของซอฟต์แวร์ Weka 3.6 ไม่สามารถทำได้ แต่โปรแกรม ID3UP ที่พัฒนาขึ้น ทำได้อย่างมีประสิทธิภาพ แสดงผลออกมาตรงตามเงื่อนไขที่กำหนด

4.5 การอภิปรายผล

จากการทดสอบโปรแกรม ID3UP ด้วยข้อมูลสามชุด พบว่า โปรแกรมการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้หรือ ID3UP สามารถจำแนกข้อมูลออกมาเป็นกฎได้ถูกต้องและการเควียรีด้วยคำสั่งและเงื่อนไขที่กำหนดจากผู้ใช้ สามารถทำได้รวดเร็วและมีการแสดงผลของกฎที่ถูกต้อง อีกทั้งจำนวนกฎที่ได้นั้น มีจำนวนลดลงตรงกับจุดประสงค์ของโปรแกรม คือ ต้องการคัดเลือกเฉพาะกฎที่จำเป็นต่อผู้ใช้ โดยจะใส่เงื่อนไขที่คัดเลือกกฎออกมา แต่การเขียนโปรแกรมบางคำสั่งส่งผลให้เกิดข้อผิดพลาดเกี่ยวกับการแสดงผล คือ มีกฎที่ซ้ำกัน ทำให้จำนวนกฎมีมากขึ้น การแสดงผลจึงไม่มีประสิทธิภาพ และเมื่อปรับปรุงโปรแกรมแล้ว จะได้ผลออกมาที่ดีขึ้น อีกทั้งกฎที่ได้นั้นไม่ซ้ำกันและจำนวนกฎน้อยลง และเวลาในการเควียรีแต่ละครั้งสามารถทำได้รวดเร็วเนื่องจากโปรแกรมจะไปค้นหากฎที่ตรงกับเงื่อนไขจากกฎที่ได้จำแนกข้อมูลไว้แล้ว ซึ่งถูกบันทึกไว้ในหน่วยความจำ

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.25 การเปรียบเทียบเวลาของการทดสอบการเควียรีกฎด้วยข้อมูลผู้ป่วยตับอักเสบ

ครั้งที่ / รูปแบบการเควียรี	1	2	3	4	5	6	7	8	9	10	AVG
findRule([])	0.02	0.05	0.01	0.05	0.03	0.05	0.01	0.01	0.03	0.03	0.029
findRule([bilirubin - "0.1-1.0"])	0.01	0.02	0.01	0.01	0.03	0.01	0.02	0.02	0.01	0.01	0.015
findRule([\+bilirubin - "0.1-1.0"])	0.02	0.01	0.02	0.03	0.03	0.02	0.03	0.01	0.01	0.02	0.020
findRule([bilirubin - "0.1-1.0" , class - live])	0.02	0.02	0.02	0.01	0.02	0.01	0.01	0.01	0.01	0.02	0.015
findRule([\+bilirubin - "0.1-1.0" , class - live])	0.02	0.02	0.01	0.02	0.02	0.01	0.01	0.01	0.02	0.02	0.016
findRuleOr([[ascites - yes],[bilirubin - "0.1-1.0, age - "31-40"]]) → Version 1	0.02	0.02	0.00	0.03	0.03	0.05	0.03	0.05	0.03	0.06	0.032
findRuleOr([[ascites - yes],[bilirubin - "0.1-1.0", age - "31-40"]]) → Version 2	0.02	0.01	0.02	0.01	0.02	0.02	0.02	0.03	0.01	0.02	0.018

หมายเหตุ : หน่วยของเวลา คือ วินาที

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.26 การเปรียบเทียบเวลาของการทดสอบการเควียรีกฎด้วยข้อมูลผู้ป่วยมะเร็งเต้านม

ครั้งที่ / รูปแบบการเควียรี	1	2	3	4	5	6	7	8	9	10	AVG
findRule([])	0.06	0.03	0.05	0.02	0.02	0.03	0.01	0.02	0.03	0.03	0.030
findRule([breast_quad - right_up])	0.03	0.05	0.01	0.02	0.03	0.01	0.01	0.01	0.02	0.02	0.021
findRule([\+breast_quad - right_up])	0.02	0.02	0.03	0.02	0.03	0.03	0.03	0.01	0.01	0.02	0.022
findRule([breast_quad - right_up , class - no_recurrence_events])	0.03	0.01	0.03	0.02	0.01	0.01	0.02	0.01	0.01	0.02	0.017
findRule([\+breast_quad - right_up , \+class - no_recurrence_events])	0.02	0.01	0.01	0.03	0.03	0.01	0.01	0.01	0.01	0.01	0.015
findRuleOr([[breast_quad - left_low , inv_nodes - "6-8"] , [breast_quad - left_low, inv_nodes - "6-8", tumor_size - "15-19"]]) → Version 1	0.02	0.01	0.01	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.013
findRuleOr([[breast_quad - left_low , inv_nodes - "6-8"] , [breast_quad - left_low, inv_nodes - "6-8", tumor_size - "15-19"]]) → Version 2	0.02	0.01	0.02	0.02	0.01	0.01	0.02	0.01	0.01	0.01	0.014

หมายเหตุ : หน่วยของเวลา คือ วินาที

มหาวิทยาลัยเทคโนโลยีสุรนารี

ตารางที่ 4.27 การเปรียบเทียบเวลาของการทดสอบการเควียรีกฎด้วยข้อมูลผู้ป่วยโรคไทรอยด์

ครั้งที่ / รูปแบบการเควียรี	1	2	3	4	5	6	7	8	9	10	AVG
findRule([])	0.02	0.01	0.02	0.01	0.02	0.03	0.03	0.03	0.02	0.02	0.021
findRule([pregnant - false])	0.01	0.01	0.02	0.01	0.01	0.02	0.01	0.01	0.01	0.01	0.012
findRule([\+pregnant - false])	0.01	0.01	0.01	0.01	0.02	0.01	0.01	0.02	0.02	0.02	0.014
findRule([query_hyperthyroid - true , query_hypothyroid - true ,sex - male , on_antithyroid_medication - false])	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.010
findRule([\+query_on_thyroxine - false , \+on_antithyroid_medication , \+class - sick])	0.01	0.01	0.01	0.01	0.01	0.02	0.02	0.01	0.03	0.02	0.015
findRuleOr([[query_hyperthyroid - true , query_hypothyroid - false] , [\+query_hypothyroid - true] , [\+class - negative]]) → Version 1	0.01	0.03	0.03	0.03	0.03	0.05	0.03	0.02	0.05	0.03	0.031
findRuleOr([[query_hyperthyroid - true , query_hypothyroid - false] , [\+query_hypothyroid - true] , [\+class - negative]]) → Version 2	0.02	0.01	0.01	0.02	0.01	0.02	0.01	0.02	0.02	0.03	0.017

หมายเหตุ : หน่วยของเวลา คือ วินาที

มหาวิทยาลัยเทคโนโลยีสุรนารี

บทที่ 5
สรุปผลการวิจัยและข้อเสนอแนะ

การทำเหมืองข้อมูล (Data Mining) ในปัจจุบันมีความจำเป็นต่อการทำงานเกี่ยวกับข้อมูลและการวิเคราะห์ข้อมูล เพื่อค้นหารูปแบบหรือโมเดลของข้อมูล และอัลกอริทึมที่ใช้ในการทำเหมืองข้อมูลมีอยู่มากมายให้ได้ใช้ประโยชน์ โดยผู้ใช้สามารถเลือกให้เหมาะสมกับข้อมูลและเหมาะสมกับงาน งานการวิเคราะห์ข้อมูลที่นิยมใช้มากที่สุด คือ การจำแนกข้อมูล (Classification) อัลกอริทึมที่นิยมใช้มากกับงานประเภทนี้ คือ อัลกอริทึมสังเคราะห์ต้นไม้ตัดสินใจ (Decision Tree Induction) ซึ่งเป็นอัลกอริทึมที่มีประโยชน์กับการสร้างโมเดลข้อมูลที่ต้องการแสดงผลลัพธ์ออกมาเป็นแผนภาพที่สะดวกต่อการแปลงกฎ (Rules) เพื่อง่ายต่อการนำไปใช้ทำนายหรือดูแนวโน้มของข้อมูลว่าเป็นไปในลักษณะแบบใด ซึ่งการแสดงผลลัพธ์ที่ได้จากการจำแนกข้อมูลนั้น ถ้าข้อมูลมีจำนวนมาก จะทำให้ผลที่ได้เป็นต้นไม้ที่มีจำนวนโหนดมากตามไปด้วย โดยบางส่วนของโมเดลอาจจะไม่จำเป็นในการใช้งานสำหรับผู้ใช้ การคัดเลือกกฎหรือรูปแบบในส่วนที่สำคัญต่อการใช้งานหรือตรงความต้องการของผู้ใช้จะมีประโยชน์มากสำหรับการทำงาน

งานวิจัยนี้จึงมุ่งเน้นการค้นหาเฉพาะกฎการจำแนกข้อมูลที่ตรงตามเงื่อนไขของผู้ใช้ โดยพื้นฐานการจำแนกข้อมูล ใช้อัลกอริทึม ID3 พัฒนาและประมวลผลด้วยภาษาโปรล็อกซึ่งเป็นภาษาเชิงตรรกะที่สามารถพัฒนาโปรแกรมได้รวดเร็ว เนื่องจากเป็นภาษาระดับสูงและใช้การเขียนคำสั่งเชิงประกาศ อัลกอริทึม ID3 ที่ใช้เป็นพื้นฐานเป็นอัลกอริทึมสำหรับการจำแนกข้อมูลที่ไม่มีการตัดกิ่ง (Pruning) ทำให้กฎการจำแนกจะแสดงผลออกมาทั้งหมด ซึ่งจะทำให้การคัดเลือกกฎเฉพาะเงื่อนไขที่กำหนดสามารถค้นหาจากกฎทั้งหมดได้อย่างมีประสิทธิภาพ

ขั้นตอนการวิจัยแบ่งออกเป็น การศึกษาขั้นตอนวิธีการทำงานการจำแนกข้อมูลด้วยอัลกอริทึม ID3 โดยใช้ภาษาโปรล็อกโดยใช้รูปแบบไวยากรณ์ของโปรแกรม Eclipse ซึ่งเป็นโปรแกรมสำเร็จรูปที่สามารถประมวลผลภาษาโปรล็อกโดยมี Module ให้สามารถเรียกใช้ได้อย่างอัตโนมัติ โดยไม่ต้องเขียนคำสั่งเพิ่มเติมมาก ช่วยลดเวลาและจำนวนบรรทัดของโค้ดได้เป็นอย่างดี และในงานวิจัยนี้ได้เพิ่มคำสั่งการค้นหากฎที่ตรงตามเงื่อนไขที่กำหนดของผู้ใช้ โดยผ่านเพรดิเคตของภาษาโปรล็อก เพื่อให้ได้กฎที่ต้องการ ทำให้กฎมีจำนวนลดลงและแสดงผลได้อย่างถูกต้อง

5.1 สรุปผลการวิจัย

การทดสอบประสิทธิภาพของการสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ ได้ใช้ข้อมูลจริงจาก UCI Machine Learning Repository (http://archive.ics.uci.edu/ml) ที่เป็นข้อมูลสำหรับการจำแนกข้อมูลจำนวน 3 ชุด ได้แก้ ข้อมูลผู้ป่วยโรคตับอักเสบ, ข้อมูลผู้ป่วยมะเร็งเต้านม, ข้อมูลผู้ป่วยโรคไทรอยด์ และแปลงข้อมูลอยู่ในรูปแบบที่สามารถจำแนกข้อมูลด้วยโปรแกรมของงานวิจัยนี้ได้ การค้นหากฎตามเงื่อนไขโดยใช้เพรดิเคตที่กำหนดนั้น เงื่อนไขที่จะค้นหาจำเป็นจะต้องทำให้อยู่ในรูปแบบที่กำหนด คือ Attribute – Value และความหลากหลายในการค้นหาก็ต้องกำหนดเป็นไปตามคำสั่งที่สร้างไว้ คือ ถ้าเป็นคำสั่ง NOT ให้ใช้เครื่องหมาย \+ ทำให้การค้นหามีความชัดเจนและมีประสิทธิภาพมากยิ่งขึ้น

มหาวิทยาลัยเทคโนโลยีสุรนารี

5.2 การประยุกต์งานวิจัย

งานวิจัยนี้มีส่วนสำคัญสำหรับการแสดงผลของกฎที่ได้จากการจำแนกข้อมูล เหมาะสมกับการประยุกต์ใช้กับข้อมูลที่มีจำนวนข้อมูลและแอททริบิวต์จำนวนมาก ที่จะส่งผลให้กฎที่ได้จากการทำเหมืองข้อมูลมีจำนวนมากไปด้วย การประยุกต์ใช้โปรแกรม ID3UP ของงานวิจัยนี้ ทำให้ลดจำนวนการแสดงผลของกฎได้ดี สามารถประยุกต์ใช้การงานที่ต้องการจำแนกข้อมูล แต่ไม่ต้องการแสดงผลของกฎทั้งหมด เพียงแต่แสดงออกมาเฉพาะเงื่อนไขที่ผู้ใช้กำหนด กล่าวคือ แสดงบางแอททริบิวต์หรือแสดงบางค่าที่มีอยู่ในกฎ เช่น งานของข้อมูลทางการแพทย์ที่อาจจะดูเฉพาะสาเหตุบางอย่างที่มีผลทำให้เกิดโรคได้ ซึ่งงานวิจัยนี้สามารถใช้เป็นแนวทางในการพัฒนาโปรแกรมที่ต้องการคัดเลือกข้อมูลที่ได้จากการทำเหมืองข้อมูล และเป็นงานวิจัยในอนาคตที่สามารถปรับปรุงให้การแสดงผลเฉพาะมีความหลากหลายมากกว่าเดิม

5.3 ปัญหาและข้อเสนอแนะ

การสังเคราะห์โมเดลเพื่อการจำแนกตามข้อกำหนดของผู้ใช้ จำเป็นต้องทำข้อมูลให้อยู่ในรูปแบบของข้อความตรรกศาสตร์อันดับหนึ่งที่สามารถสนับสนุนกับคำสั่งที่สร้างขึ้นได้ ทำให้ในการใช้งานผู้ใช้อาจจะเสียเวลาในการจัดการข้อมูลก่อนนำมาจำแนกข้อมูล นอกจากนี้การค้นหากฎตามเงื่อนไขที่ผู้ใช้ระบุนั้นสามารถทำได้แค่เฉพาะเงื่อนไขที่ตรงกับรูปแบบและคำสั่งที่กำหนดไว้ อาจจะทำให้การสนับสนุนงานบางอย่างไม่มีประสิทธิภาพที่ดีมากนัก หากจะต้องพัฒนาโปรแกรมให้มีประสิทธิภาพให้ดียิ่งขึ้นสามารถทำได้หลายแนวทาง เช่น

- การจัดรูปแบบของข้อมูลที่สนับสนุนการทำงานของโปรแกรม สามารถทำได้โดยเพิ่มโมดูลเพื่อการแปลงข้อมูลอัตโนมัติ
- การค้นหากฎเฉพาะเงื่อนไข ทำให้มีความหลากหลายยิ่งขึ้นอาจปรับปรุงโปรแกรมภาษาโปรล็อกที่จะเพิ่มเพรดิเคตเฉพาะการค้นหา

มหาวิทยาลัยเทคโนโลยีสุรนารี

รายการอ้างอิง

ทวีศักดิ์ นาคม่วง. (2547). **ระบบสนับสนุนการตัดสินใจ (Decision Support Systems) [ออนไลน์].** ที่มา : http://www.sirikitdam.egat.com/WEB_MIS/107/index.html

นิตยา เกิดประสพ. (2552). **การค้นหาความรู้และการขุดค้นข้อมูล (Knowledge Discovery and Data Mining)**. สาขาวิชาวิศวกรรมคอมพิวเตอร์, มหาวิทยาลัยเทคโนโลยีสุรนารี , นครราชสีมา.

ปัณณวิชญ์ วงศ์วิวัฒนานนท์. (2551). **การทำเหมืองข้อมูลกับสารสนเทศทางการแพทย์ (Data mining in medical informatics).** กลุ่มงานเวชศาสตร์ฟื้นฟู, โรงพยาบาลราชวิถี, กรุงเทพมหานคร.

วิกิพีเดีย สารานุกรมเสรี. (2554a). **การทำเหมืองข้อมูล [ออนไลน์].** ที่มา : http://th.wikipedia.org/wiki/การทำเหมืองข้อมูล

วิกิพีเดีย สารานุกรมเสรี. (2554b). **การแบ่งประเภทข้อมูล [ออนไลน์].** ที่มา : http://th.wikipedia.org/wiki/การแบ่งประเภทข้อมูล

สมประสงค์ เสนารัตน์. (2553). **การจำแนกกลุ่มด้วยเทคนิค Discriminant Analysis.** สาขาวิชาวิจัยและประเมินผล, มหาวิทยาลัยมหาสารคาม, มหาสารคาม.

Adriaans, P., and Zantinge, D. (1999). **Two Crows Corporation, Introduction to Data Mining and Knowledge Discovery.** New York, U.S.A.

Air Force Research Laboratory Information Directorate (2004). **Link Analysis Workbench**. Air Force Research Laboratory Information Directorate, Rome Research Site, New York, U.S.A.

Bahety, A. (2006). **Extension and Evaluation of ID3 – Decision Tree Algorithm**. Department of Computer Science, University of Maryland, College Park, U.S.A.

Bodhisuwan, W. (2011). **Data Collection and Sampling Techniques [Online].** Available URL : http://suanpalm3.kmutnb.ac.th/teacher/winai/

Brezany, P., and Fur, I. (2006). **Automatic Cluster Detection.** Softwarewissenschaft Universitat Wien, Austria.

Coomans, D., and Massart, D.L. (1982). **Alternative k-nearest neighbour rules in supervised pattern recognition : Part 1. k-Nearest neighbour classification by using alternative voting rules**. Analytica Chimica Acta, U.S.A.

David, P., and Alan, M. (2010). **Artificial Intelligence Foundations of Computational Agents [Online].** Available URL : http://artint.info/html/ArtInt_183.html

Elfeky, M.G., Saad, A.A., and Fouad, S.A. (2001). **ODMQL: Object Data Mining Query Language.** Computer Science and Automatic Control Department, Faculty of Engineering, Alexandria University, Egypt.

Goldberg, D.E. (1989). **Genetic Algorithms in Search Optimization and Machine Learning,** p.41. Addison-Wesley Professional.

Gutierrez, N. (2006). **Demystifying Market Basket Analysis.** Information Management Special Reports.

Han, J., and Kamber, M. (2001). **Data Mining Concepts and Techniques,** p.29. Morgan Kaufmann, U.S.A.

Han, J., Fu, Y., Wang, W., Koperski, K., and Zaiane, O. (1996). **DMQL: A Data Mining Query Language for Relational Database.** Database System Research Laboratory, School of Computing Science, Simon Fraser University, British Columbia, Canada.

Han, J., Kamber, M., and Pei, J. (2006). **Data Mining: Concepts and Techniques**. Morgan Kaufmann, U.S.A.

Hastie, T.J., and Tibshirani, R.J. (1990). **Generalized Additive Models.** Chapman & Hall/CRC.

Hastie, T.J., Tibshirani, R.J., and Friedman, J.H. (2009). **The Elements of Statistical Learning, 2nd edition.** Springer.

Hodgson, J. (1999). **Exception Handling in ISO Prolog.** Saint Joseph's University, Philadelphia, Rensylvania, U.S.A.

Malerba, D., Appice, A., and Ceci, M. (2004). **A Data Mining Query Language for Knowledge Discovery in a Geographical Information System.** Dipartimento di Informatica, Università degli Studi di Bari, Italy.

Mitchell, T. (1997). **Machine Learning**, pp.57-78, McGraw-Hill.

Ogata, N. (2001). **Discovery of Event Classifications and Causal Constraints from Earthquake Articles.** Faculty of Language and Culture, Osaka University, Japan.

Peduzzi, P., Concato, J., Kemper, E., Holford, T.R., and Feinstein, A.R. (1996). A simulation study of the number of events per variable in logistic regression analysis. **Journal of Clinical Epidemiology,** vol.49, no.12, pp.1373-1379.

Quinlan, J.R. (1986). **Induction of Decision Trees.** Centre for Advanced Computing Sciences, New South Wales Institute of Technology , Sydney.

Quinlan, J.R. (1987). Generating production rules from decision trees. In McDermott, John. **Proceedings of the Tenth International Joint Conference on Artificial Intelligence (IJCAI-87),** pp.304–307, Milan, Italy.

Quinlan, J.R. (1992). **C4.5: Programs for Machine Learning.** Los Altos, California: Morgan Kaufmann.

Stanfill, C.W. (1987). **Memory-Based Reasoning.** Thinking Machines Corporation, Cambridge, MA, U.S.A.

Stühlinger, W., Hogl, O., Stoyan, H., and Müller, M. (2000). Intelligent Data Mining for Medical Quality Management. **Proceedings Fifth Workshop on Intelligent Data Analysis in Medicine and Pharmacology,** pp.55-57. Workshop Notes of the 14th European Conference on Artificial Intelligence.

Swarthmore Department of Computer Science (2011). **ID3 Pseudocode [Online].** Available URL : http://web.cs.swarthmore.edu/~meeden/cs63/f05/id3.html

Thorpe, S. J. (1990). Spike arrival times: A highly efficient coding scheme for neural networks. **Parallel Processing in Neural Systems,** pp.91-94, Elsevier, Amsterdam, Netherlands.

Vassiliadis, P., Quix, C., Vassiliou, Y., and Jarke, M. (2001). **Data Warehouse Process Management.** Department of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece.

Visual Prolog (2011). **An Evolution of Turbo Prolog Developed [Online].** Available URL : http://www.visual-prolog.com

WIKIPEDIA The Free Encyclopedia (2011a). **ECLiPSe [Online].** Available URL : http://en.wikipedia.org/wiki/ECLiPSe

WIKIPEDIA The Free Encyclopedia (2011b). **Prolog [Online].** Available URL : http://en.wikipedia.org/wiki/Prolog

Yuan, Y., and Shaw, M.J. (1995). Induction of fuzzy decision trees, **Fuzzy Sets and Systems,** vol. 69, pp.125-139.

Zaïane, O.R. (1999). **Principles of Knowledge Discovery in Databases**. University of Alberta, Canada.

ภาคผนวก ก

บทความวิชาการที่ได้รับการตีพิมพ์เผยแพร่

มหาวิทยาลัยเทคโนโลยีสุรนารี

รายชื่อบทความวิชาการที่ได้รับการตีพิมพ์เผยแพร่

ฝนทิพย์ คูณแก้ว, กิตติศักดิ์ เกิดประสพ และนิตยา เกิดประสพ. 2555. **Classification Model Induction Based on User Preferences**. ในการประชุมวิชาการ CIMMACS'12 The 11th International Conference on Computational Intelligence, Man-Machine System., and Cybernetics, ประเทศสิงคโปร์. 11 – 13 พฤษภาคม 2555.

Classification Model Induction Based on User Preferences

FONTHIP KOONGAEW, NITTAYA KERDPRASOP, and KITTISAK KERDPRASOP
Data Engineering Research Unit, School of Computer Engineering
Suranaree University of Technology
111 University Avenue, Nakhon Ratchasima 30000 THAILAND
gocjune29@hotmail.com, nittaya@sut.ac.th, kerdpras@sut.ac.th

Abstract: - The data mining has been influential in gathering information and data in the work of organizations and business, medical, engineering etc. The strength of data mining is the help in the matter of technique in information management. This technique can be used to find important information along with other information. This paper proposes highlight of the classification for the data in each group to work in the field of medicine that is useful for clinicians and patients that may communicate through the model made from data mining, such as "How much the amount of sugar in the blood of patients to be a risk of diabetes?". It can be output in the form of decision trees. But when we want to know just some of the information, a whole decision tree is superfluous. This research has focused on this point of knowledge reduction. We will use the method of logic programming to imitate the functionality of data mining to extract patterns from data taken from real sources. The results of pattern extraction will be in the form of rules. We increase efficiency of knowledge navigation by allowing users to specify constraints or preferences, which will help in the selection of specific rules of interest. Our methodology can enhance the search for answers, as well as reduce the time to locate all the rules.

Key-Words: - Data mining , Classification , Logic Programming , Prolog , ID3 Algorithm , User Constraint

1 Introduction

Data mining is to find relationships and patterns that exist in the database but can be hidden within large amounts of data. In this volume of information will be provided in the form of the meaning and the rules. The relationship of these to demonstrate knowledge. Useful in the database. It also has a more sophisticated analysis that contribute to the data in various formats. By using Data mining tools that can be compiled in the form of statistical, mathematical algorithms. And the process of learning. (To improve performance, it is automatic and semi-automatic). [1]

Data mining is a synthesis of detailed information from the database is large, based on information learned from the past or present. The results of the synthesis of Data mining are 3 types 1) Unknow 2) Valid and 3) Actionable. [2]

Classification is the process model data in the specified group. It is convenient to have a lot of data processing. Classification process is used to display the model with Decision Tree. Decision Tree is like tree structure. This includes an internal node that shows the feature (Attribute) of the data. The beginning is called Root Node and branches represent the attributes of each node. And Leaf node represents a group (Class) which is a results can be discerned. [3]

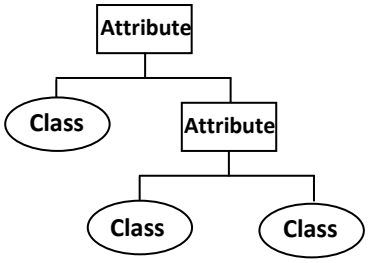


Fig.1 Decision Trees

Prolog is a Logic Programming Language for the efforts to create a logical way of living rather than the detailed instructions to the computer. [4] Prolog is used in the program for Artificial Intelligence and Computational Linguistics, especially for natural language processing. The syntax and semantics of the language is simple and clear. The first goal of the language is a tool for linguists. [5]

The paper is organized as follows: Section 2 describes the related work. Concerning the classification of information. And the use of query data. Section 3 explains the meaning and the theory of ID3 algorithm used in data classification, data

preparation, data classification is designed by Prolog. And the User Constraint for the selection rules. Section 4 compares the accuracy of the rules on the classification algorithm using Weka. Together with the efficiency and speed of search rules using User Constraint. Section 5 summarizes the results of the identification information with the ID3 algorithm by using the User Constraint and future work.

2 Related Works

Data mining has been influential in various fields such as business, medicine, engineering, etc. which often come from large databases. Most of the information collected from real data [6]. This research was in part Postprocessing is to find the information that the user's purposes. The research was described. Postprocessing is a process that knowledge to interpret the Data Mining and processing techniques will show the result. [7]

In the major part of the KDD process is to select the algorithm to be used. [8] Many research use technical Classification with information. This research was conducted using the ID3 algorithm to classify data. The input data of ID3 is known as sets of “training” or “learning” data instances, which will be used by the algorithm to generate the Decision Tree. The machine is “learning” from this set of preliminary data. [9]

Wolf, Oliver, Herbert and Michael [10] has presented about data mining of medical data. For the purpose of business is very fast in the search history. Because data mining is to classify data types that look like they are in the same. The field of medicine is to be accurate in the diagnosis of disease, the symptoms of patients with a history of a specific age range that would cause it. And reports in the form of If ... Then rules and these rules will be used to find information in the database (Database)

Fahad, Rao, Olof, and Göran [11] have taken the oral medicine in large number of EMRs (Electronic Medical records) to perform a search pattern of data for to find a possible factor’s diagnosis of oral diseases. Using C4.5 algorithm which the algorithm’s ability to handle data with missing values. It also avoids overfitting the data and reduce error pruning.

SHELLY, DHARMINDER and ANAND recommend the use of classified information. This is based on the Data Mining algorithms such as Decision Trees for the analysis to obtain information on the diagnosis of breast cancer which use of patient information in the database. It can identified that Factors and trends in breast cancer patients. [12]

This research is part of the query rules from data mining. The specific conditions to show results as needed. This will help reduce the number of rules is displayed to avoid the unnecessary rules. And J. Han and team [13] have presented a data mining tool for Relational Database. For the purpose of Data Mining Query Language (DMQL) such as classification of the student of Computing Science in Canada which will see the rules or patterns have GPA of students is an important factor such as classification of the student of Computing Science in Canada which will see the rules or patterns that have a GPA of students is an important factor which has brought this concept to various research applications such as Elfeky M.G. and team [14] to the search rules or patterns that are correlated with the type of Object Data Mining Query Language (ODMQL) as well as Donato Malerba and team [15] for Spatial Data Mining, which is a data mining model. one Commonly used in Geographic Information Systems (GIS) to create and analyze models of building conditions.

3 Experiments And Results

3.1 The Classification Using ID3 Algorithm With Prolog Programming

ID3 is an algorithm for use in building a decision tree. The algorithm is used to classify data. This can be done several ways, but ID3 is a search to select the best classification was used. [16]

The classification using ID3 algorithm with Prolog programming. The data given in the form of *if* ... *then* rules by ID3 algorithm — a decision tree from top to bottom. In choosing the best way to see the value of the Information Gain. Pruning to avoid at all easy to read. The program successfully used by Prolog is a logic operation. The problem with structured uncertainties. It is also the language of the summary of responses to complex diagnostic and review of the facts derived from actual data in the form to do Data mining.

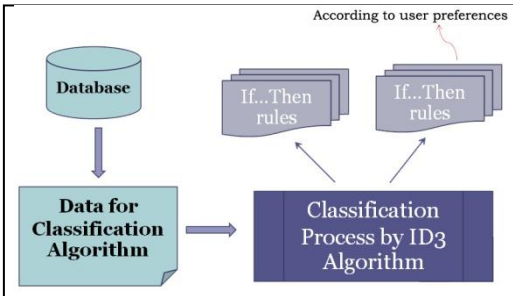


Fig. 2 Architecture of framework

3.2 Designed And Developed The Classification System Using ID3 Algorithm With Prolog Programming

The researcher has to do with the actual data. This is a medical history of breast cancer patients, which is taken from the UCI Machine Learning Repository and changed the format of the original data. Into a format that can be classified information (Fig.3). The results of the classification of algorithm data by ID3. Information Gain is selected on the basis of the most valuable attributes. When preparing the data. Each data must be defined to have attributes that can modify the data (Fig. 4). The format of attributes — goal attribute is to be determined at the first. And the form of is *Name of attribute – Values of attribute*.

```
data([
[age-30-39,     menopause-premeno,      tumorsize-30-34,        inv_nodes-0-2,
[age-40-49,     menopause-premeno,      tumorsize-20-24,        inv_nodes-0-2,
[age-40-49,     menopause-premeno,      tumorsize-20-24,        inv_nodes-0-2,
[age-60-69,     menopause-ge40,         tumorsize-15-19,        inv_nodes-0-2,
[age-40-49,     menopause-premeno,      tumorsize-0-4,          inv_nodes-0-2,
[age-60-69,     menopause-ge40,         tumorsize-15-19,        inv_nodes-0-2,
[age-50-59,     menopause-premeno,      tumorsize-25-29,        inv_nodes-0-2,
[age-60-69,     menopause-ge40,         tumorsize-20-24,        inv_nodes-0-2,
[age-40-49,     menopause-premeno,      tumorsize-50-54,        inv_nodes-0-2,
[age-40-49,     menopause-premeno,      tumorsize-20-24,        inv_nodes-0-2,
[age-40-49,     menopause-premeno,      tumorsize-0-4,          inv_nodes-0-2,
[age-50-59,     menopause-ge40,         tumorsize-25-29,        inv_nodes-0-2,
[age-60-69,     menopause-lt40,         tumorsize-10-14,        inv_nodes-0-2,
[age-50-59,     menopause-ge40,         tumorsize-25-29,        inv_nodes-0-2,
[age-40-49,     menopause-premeno,      tumorsize-30-34,        inv_nodes-0-2,
[age-60-69,     menopause-lt40,         tumorsize-30-34,        inv_nodes-0-2,
[age-40-49,     menopause-premeno,      tumorsize-15-19,        inv_nodes-0-2,
[age-50-59,     menopause-premeno,      tumorsize-30-34,        inv_nodes-0-2,
[age-60-69,     menopause-ge40,         tumorsize-30-34,        inv_nodes-0-2,
[age-50-59,     menopause-ge40,         tumorsize-30-34,        inv_nodes-0-2,
```

Fig. 3 The format of Breast Cancer data used for classification.

```
 +
[
[class-no_recurrence_events, class-recurrence_events],
[age-10-19,age-20-29,age-30-39,age-40-49,age-50-59,age-60-69,
[menopause-lt40,menopause-ge40,menopause-premeno],
[tumorsize-0-4,tumorsize-5-9,tumorsize-10-14,tumorsize-15-19,
[inv_nodes-0-2,inv_nodes-3-5,inv_nodes-6-8,inv_nodes-9-11,inv
[node_caps-yes,node_caps-no],
[deg_malig-1,deg_malig-2,deg_malig-3],
[breast-left,breast-right],
[breast_quad-left_up,breast_quad-left_low,breast_quad-right_u
[irradiat-yes,irradiat-no]
]
).
```

Fig. 4 The format of attributes.

The principle of the ID3 algorithm refers to the Entropy of M data set — contains the possible values is {m1, m2,..., mn} and the probability that the value of mi is equal to P (mi) is that the Entropy of the M data set that measures the average value of information technology to identify the type of information, denoted by I (M) calculated from the formula. [17]

$$I(M) = \sum_{i}^{n} -P(m_i)\log_2 P(m_i)$$

Entropy values for each data set. Then calculate Gain value of each attribute. The principle of ID3 algorithm processed by Prolog Programming. The first step is to choose an attribute node. Entropy is the value of at least minus Attribute with the Entropy of goal class. Using minimum Entropy. It has the most Gain. Also node is an attribute which to calculates Gain of label which will be similar to Entropy of label is a minimum value determined by the formula Information Gain.

$$Gain(X) = I(T) - I_x(T)$$

This research was ID3 algorithm is used to classify data in order to display the rules. And conditions (User Constraint) which satisfy the selection rule.

Classification using ID3 algorithm with Logic Programming was started read the information on the Data mining format. Processing by ID3 algorithm and the results obtained from the selected attributes from each node. The final rules will be possible (Fig. 5).

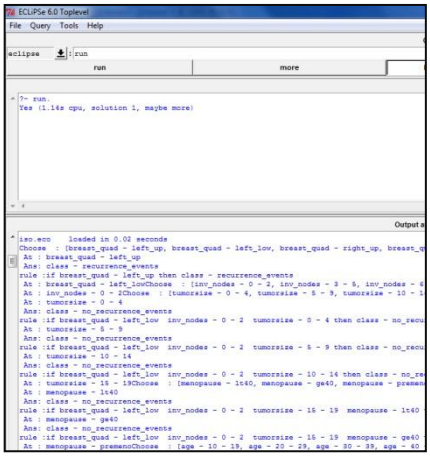


Fig. 5 The program displays the data classification using ID3 algorithm.

3.3 Designed And Developed The Classification System Using ID3 Algorithm With Prolog Programming – Rules Are Selected In The User Constraint

This research has focused on creating conditions to support the display of the rule to show and not only rules that are invoked as a way to query which uses predicates that make up the predicate of findRule ([X]). or findRule ([\X]) is a predicate of the conditions from the functionality of the SQL language is a method of query terms in the database. Query command is a concept used in the WHERE clause of the database means. Terms and Conditions and select one or more of the conditions.

Research related to Intelligent Data Mining Data mining is to establish rules for the Trigger in the clinical data. Purpose to find and identify the factors to diagnose disease. The majority of output as "if…then" rules [10] that this research has brought this concept to model the User Constraint for query rules in data mining (Fig. 6).

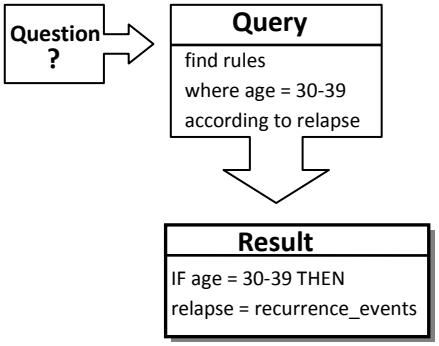


Fig. 6 The Model of query rules

The classification of data using ID3 algorithm would generate all of rules , some rule may not be useful to users. User Constraint must be so easy to find. And reduce the visibility of the rule that might be excessively. The ID3 algorithm designed to process all the rule without Pruning process and stored as a model. Order to contribute to search or not search Attributes, Attributes and Label is very deep in the decision tree.

The system is designed to *findRule ([])* Predicate that can find rule to display the results. Which is defined as follows.

3.3.1 Find all rules

findRule ([]) : display of all the rules that the system will classify the data.

3.3.2 Find attributes have in Rules

findRule ([X]) : display of rule by the User Constraint requires that *X* is an attribute that is contained in the rule or X is an attribute and the label (such as *irradiat – yes*) the existing rules will be displayed. It can put as many attributes.

3.3.2 Find attributes not have in Rules

findRule ([\+X]) : display of rules by the Using Constraint specify that *X* is an attribute that does not exist in the rules that will be displayed. It can put as many attributes. The symbol \+ of the command is *NOT* or *X* is an attribute and label (such as *\+inv_nodes-"0-2"*) that is not in rules. It can put as many attributes.

3.4 Results

The result of the query patterns or rules from data mining type of classification. Depending on the condition that the user specified in *findRule* predicate. The rule output is dependent on certain conditions. If the number of rules are less. Thus, the query rules is a condition that can cut a lot of rules. Method of query the rules in this manner. The principle is similar to DMQL[13] as a result of information from large databases. By the way, this type of query rules. Similar to the DMQL [13] as a result of information from large databases for data mining, the patterns or rules that are hidden in those data and the pattern or rule that has been recorded into the database. And rules query by relational query language, SQL.

This paper introduced the concept of how to query the rules of the principle of DMQL to create a predicate by Prolog Programming for query the specific rules as prescribed. The system is capable of data mining and query patterns or rules which does not require recorded in the database. Also rules do not need to query the SQL language that defines the conditions that are too long and must have knowledge about how to query the database.

The system was tested with Breast Cancer Data Set from UCI Machine Learning Repository [18], number of instances = 286 and number of Attributes = 9. This data set useful for the selection of the rule. In fact, if data about medicine. Some attributes may not be necessary to diagnose the

disease may be cut off. After classifying all the rules of the query by Prolog Programming. A total of all rules is 102 rules (Fig. 7).

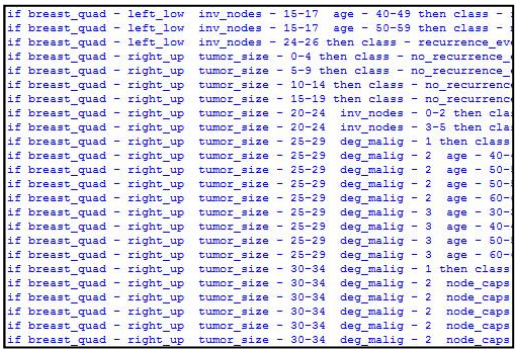


Fig. 7 To query all rules.

System query rules by the user constraint for to display only rules that are needed. We tested to query rules which a *class-recurrence_events* query only. This means that there are any rules that would breast cancer patients relapse. The specific conditions of the query *findRule ([class - recurrence_events])* which there are 43 rules. It is fewer all rules. And speed of the query rule is faster. Since all the rules of classification of data set stored in memory before the query rule (Fig. 8).

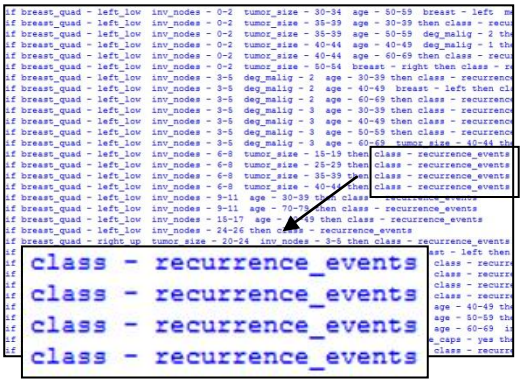


Fig. 8 To query rules have condition.

Adding query conditions to rules. The system can include an unlimited number of conditions. As well as insert conditions want and do not want in *findRule* predicate. If the conditions of a query is *findRule([breast_quad - right_up , \+age , class - recurrence_events]).* The results show two rules. This condition means that Breast Cancer patients with a history of Breast Cancer in the upper right chest. Without condition of age is a factor of relapse (Fig. 9).

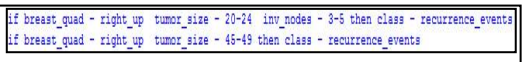


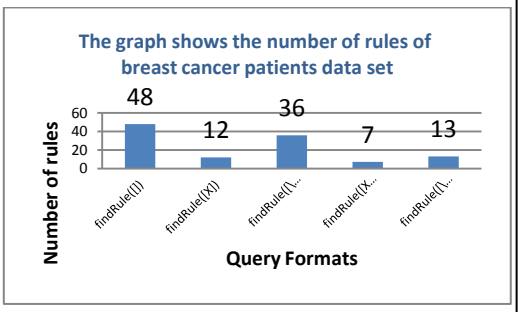
Fig. 9 To query rules have conditions.

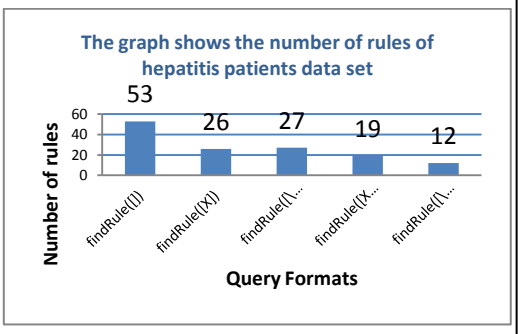
4 Evaluation

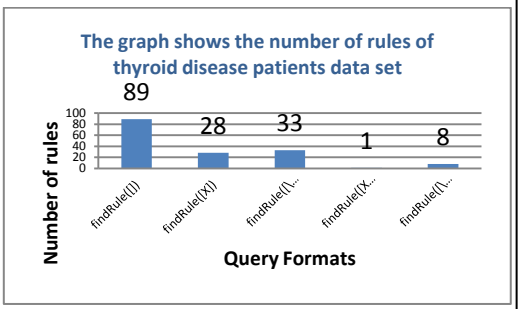
Our research compared with its efficiency of framework. The comparison to classification by ID3 algorithm with other algorithms which cannot query in this research. The comparison of 2 types.

4.1 Comparison of the number of rules

Tested with 3 data sets of medical data. Each query condition of all graphs. There are fewer rules which are more specific. It has effective report.







4.2 Comparison of time.

Testing times of each data set which query condition 5 times. And each times are averaged. All times to experiment is less. The rules of classification is saved before to query conditions by users. For example, hepatitis patients data set (Fig. 10).

findRule([]) → 0.03 s

findRule([X]) → 0.01 s

findRule([\+X]) → 0.02 s

findRule([X1,X2]) → 0.016 s

findRule([\+X1,X2]) → 0.018 s

Fig. 10 The testing of time of hepatitis patients data set

5 Conclusion

Data Mining of Classification has been used with various databases for to create patterns or rules to be utilized in a variety of ways to be kind of database. This research illustrates the importance of classification of medical data set. For doctor or person concerned without knowledge of data mining can be seen as factors for disease. When importing data set for data mining by using Prolog Programming. There are lot of rules. The predicate generated by the Prolog Programming for rules to a query by the user constraint. To query the specific rules defining the conditions that can quickly and accurately. Moreover, the query conditions, the number of rules that show a decline. The selection rules that match the criteria. It is useful to users who can understand the rules that match the requirements specified in the conditions. It is simple and readable. This concept can be utilized for database which to classify and to query patterns or rules specific conditions. You do not need to bring the rules into the database. And do not use SQL to query the rule's conditions.

Future work to extend the database with a large and complex. The rules in each time to query may be stored in memory. For the purpose of the rule in the next query. The rule that was previously. In order to query rules each time associated with it. And reduce the number of rules to keep.

References:

[1] Anand Bahety, *Extension and Evaluation of ID3 – Decision Tree Algorithm*, 2006, Department of Computer Science University of Maryland, College Park.

[2] Donato Malerba, Annalisa Appice and Michelangelo Ceci, *A Data Mining Query Language for Knowledge Discovery in a Geographical Information System*, 2004.

[3] Elfeky M.G., Saad A.A. and Fouad S.A., *ODMQL: OBJECT DATA MINING QUERY LANGUAGE*, 2001.

[4] Fahad Shahbaz Khan, Rao Muhammad Anwer, Olof Torgersson and Göran Falkman, *Data Mining in Oral Medicine Using Decision Trees*, 2008.

[5] Goldberg and David E., *Genetic Algorithms in Search Optimization and Machine Learning*, 1989, pp.41, Addison-Wesley Professional.

[6] Ivan Bruha and A. (Fazel) Famili, *Postprocessing in Machine Learning and Data Mining*, 2000.

[7] J. Han, Y. Fu, W. Wang, K. Koperski, and O. Zaiane., *DMQL: A Data Mining Query Language for Relational Database*, 1996.

[8] J.P.E. Hodgson, *Exception Handling in ISO Prolog*, 1999.

[9] J.R, QUINLAN, *Induction of Decision Trees*, 1986

[10] Jiawei Han, Micheline Kamber and Jian Pei, *Data Mining: Concepts and Techniques*, 2006.

[11] Karina Gibert, Miquel Sànchez-Marrè and Víctor Codinaa, *Choosing the Right Data Mining Technique: Classification of Methods and Intelligent Recommendation*, 2010.

[12] Associate Professor Nittaya Kerdprasop, *Post-data mining processing*, 2009

[13] Pieter Adriaans and Dolf Zantinge, *Introduction to Data Mining and Knowledge Discovery, Third Edition (Potomac, MD: Two Crows Corporation, 1999)*, 1996, Addison Wesley.

[14] Shelly Gupta, Dharminder Kumar and Anand Sharma, *Data Mining Classification Techniques Applied For Breast Cancer Diagnosis And Prognosis*, 2011.

[15] Thair Nu Phyu, *Survey of Classification Techniques in Data Mining*, 2009.

[16] UCI Machine Learning Repository, http://archive.ics.uci.edu/ml, 2011.

[17] Visual Prolog, *History*, http://www.visual-prolog.com, 2011.

[18] Wolf Stühlinger, Oliver Hogl, Herbert Stoyan and Michael Müller, *Intelligent Data Mining for Medical Quality Management*, 2000.

ภาคผนวก ข

รหัสต้นฉบับของโปรแกรม

ID3UP for findRule() Predicate & findRuleOr() Predicate

ID3UP for findRule() Predicate & findRuleOr() Predicate (ปรับปรุง)

มหาวิทยาลัยเทคโนโลยีสุรนารี

โปรแกรม ID3UP for findRule() Predicate & findRuleOr() Predicate

```
% Dicision Tree : id3
:-lib(listut). %can use only 1 library
:- lib(sd).
:-dynamic rule_me/1.
:-dynamic allrule/1.

append_me([H|T],L,[H|RT])  :-  append_me(T,L,RT).
append_me([],L,L).

findRuleOr([H|T])  :-  ( findRule(H) -> true;! ) , findRuleOr(T).
findRuleOr([]).

findRule(X)  :-  allrule(L) , findRule(X,L).
findRule(_,[]).
findRule(X,[H|T])  :-  (findQuery(X,H) -> split_rule(H) ; true) , findRule(X,T).

findQuery([\+X|TX],H)   :-        ( findAtt(X,H) -> false ;
                                        ( findAttLabel(X,H) -> false ; true )
                                  ) , findQuery(TX,H).
findQuery([X|TX],H)  :-           ( findAtt(X,H) -> true ;
                                        (findAttLabel(X,H) -> true ; false )
                                  ) , findQuery(TX,H).
findQuery([],_).

findAtt(X,[X-_|_]).
findAtt(X,[_|T])  :-  findAtt(X,T).
findAtt(_,[])  :-  false.
```

```
findAttLabel(X,[X|_]).
findAttLabel(X,[_|T]) :- findAttLabel(X,T).
findAttLabel(_,[]) :- false.
run :- retractall(rule_me(_)),retractall(allrule(_)),compile("File.txt"),data(Data+Attrs),
       main(Data,Attrs,[]),retractall(rule_me([_])),findall(X,rule_me(X),R),assert(allrule(R)).
main([],_,_).
main(_,[],_).
main(Data, Attrs,OldAttr) :- all_info(Data+Attrs, R1) ,
                             (\+hasOneClass(R1) -> (maplist(avg_info,R1,Out) ,
                             chooseMin(Out, nil/11,OO), OO=O/_,writeln('Choose ':OO),
                             [listut]:delete(Attrs,O,NewAttrs),
                             (foreach(X,O),param(Data,NewAttrs,OldAttr) do
                             (filterData([X],Data,NewData),
                             write(' At':X),append_me(OldAttr,[X],OAL),
                             main(NewData,NewAttrs,OAL))
                             )); writeln(''),getlast_goal(Data,Att-Ans),
                             write(" Ans: "),writeln(Att-Ans),
                             append_me(OldAttr,[Att-Ans],NOAL),
                             write('rule :'),split_rule(NOAL)),assert(rule_me(NOAL)).

split_rule(NOAL) :-  mem_last(NOAL,L,RL),write('if'),
                     (foreach(RLL,RL) do write(' '),write(RLL),write(' ')),
                     write('then'),write(' '),writeln(L).

%last member in list
%mem_last([1,2,3,5,6],L,RL).
mem_last([H],H,[]).
mem_last([H|T],L,[H|RL]) :- mem_last(T,L,RL).

%mem(Data+Attrs),all_info(Data+Attrs,R).
all_info(Data+[P|Attr],R) :- maplist(info(Data,P), Attr, R).
```

```
%mem(Data+_), info(Data,[p-y,p-n],[o-s,o-c,o-r], R). %R=[[2,3], [4,0], [3,2]]
info(Data, P, O, O-R)  :-  maplist(info1(Data,P),O,R).
info1(Data, P, O, R)  :-  maplist(mcount(Data,O),P,R).
mcount(Data,O,P,Sum)  :-   foreach(L,Data),fromto(0,I,R,Sum),param(O,P) do
                           ((member(O,L), member(P,L))->R is I+1; R is I).

%avg_info(o-[[2,3],[4,0],[3,2]],R).
avg_info(O-LL,O/Info)  :-  flatten(LL,L), sumlist(L, Sum),
                           (foreach(X,LL), fromto(0,I,N,Info),param(Sum) do
                           (sumlist(X,SumX),
                            (Sum>0 -> Ratio is SumX/Sum,!; Ratio is 0),
                            logInfo(X,InfoSub),
                            N is I+(Ratio*InfoSub)
                           )).

%hasOneClass([]):-!.
%hasOneClass([_-VL|T])  :-  (foreach(XL,VL) do memberchk(0,XL)), hasOneClass(T).

sumlist(L,[],L):-!.
sumlist([],[],[]):-!.
sumlist([H1|T1],[H2|T2],[HR|TR])  :-  HR is H1+H2,sumlist(T1,T2,TR).

sumlists([],R,R)  :-  !.
sumlists([H|T],PR,NR)  :-  sumlist(H,PR,R),sumlists(T,R,NR).

hasOneClass([])  :-  !.
hasOneClass([_-VL|T])  :-  sumlists(VL,[],NR),hasOneClass(T,NR).
hasOneClass([],[])  :-  !.
hasOneClass([_-VL|T],PR)  :-  sumlists(VL,PR,NR),hasOneClass(T,NR).
hasOneClass([],[H|T])  :-  (H=0->hasOneClass([],T);find(T)).
```

```
find([]):-!.
find([H|T])  :-  (H=0->find(T);false).

% get the Last data in Fist of list in list
getlast_goal([H|_],R)  :-  getlast(H,R).
getlast([H],H).
getlast([_|T],R)  :-  getlast(T,R).

%filterData
filterData(_,[],[]).
filterData(L,[H|Data],[H|R])  :-  msubset(L,H),!,filterData(L,Data,R).
filterData(L,[H|Data],R)  :-  \+msubset(L,H),!,filterData(L,Data,R).

msubset(S1,S2)  :-  foreach(X,S1), param(S2) do member(X,S2),!.

allmem([H],L)  :-  member(H,L),!.
allmem([H|T],L)  :-  member(H,L), allmem(T,L).

logInfo(XL, R)  :-  sumlist(XL,Sum), Sum==0, R=99,!.
logInfo(XL, R)  :-  sumlist(XL,Sum),
                    (foreach(X,XL), fromto(0,S,N,R), param(Sum) do
                    ( Ratio is X/Sum,
                     (Ratio>0->[iso]:log(Ratio,Log);Log is 1), %log(0) is undefined
                    [iso]:log(2,Base2),
                    N is S-(Ratio*(Log/Base2))
                    )).

chooseMin([],O/Tmp,O/Tmp).
chooseMin([A/H|T],O/Tmp,Min)  :-  (H<Tmp -> NextMin = A/H ; NextMin = O/Tmp),
chooseMin(T,NextMin,Min).
```

โปรแกรม ID3UP for findRule() Predicate & findRuleOr() Predicate (ปรับปรุง)

```
% Dicision Tree : id3
:-lib(listut). %can use only 1 library
:- lib(sd).
:-dynamic rule_me/1.
:-dynamic allrule/1.

append_me([H|T],L,[H|RT])  :-  append_me(T,L,RT).
append_me([],L,L).

findRuleOr([H|T])  :-  findRule2(H) , findRuleOr(T).
findRuleOr([])  :-  setof(H,r(H),Z) , split(Z).

split([]).
split([H|T])  :-  split_rule(H) , split(T).

findRule2(X)  :-  allrule(L) , findRule2(X,L).

findRule2(_,[]).
findRule2(X,[H|T])  :-  (findQuery(X,H)-> assert(r(H));true),
          findRule2(X,T).

findRule(X)  :-  allrule(L) , findRule(X,L).
findRule(_,[]).
findRule(X,[H|T])  :-  (findQuery(X,H) -> split_rule(H) ; true) , findRule(X,T).

findQuery([\+X|TX],H)   :-          ( findAtt(X,H) -> false ;
                                        ( findAttLabel(X,H) -> false ; true )
                                    ) , findQuery(TX,H).
```

```
findQuery([],_).


findAtt(X,[X-_|_]).
findAtt(X,[_|T]) :- findAtt(X,T).
findAtt(_,[]) :- false.


findAttLabel(X,[X|_]).
findAttLabel(X,[_|T]) :- findAttLabel(X,T).
findAttLabel(_,[]) :- false.
run :- retractall(rule_me(_)),retractall(allrule(_)),compile("File.txt"),data(Data+Attrs),
        main(Data,Attrs,[]),retractall(rule_me([_])),findall(X,rule_me(X),R),assert(allrule(R)).
main([],_,_).
main(_,[],_).
main(Data, Attrs,OldAttr) :- all_info(Data+Attrs, R1) ,
                             (\+hasOneClass(R1) -> (maplist(avg_info,R1,Out) ,
                             chooseMin(Out, nil/11,OO), OO=O/_,writeln('Choose ':OO),
                             [listut]:delete(Attrs,O,NewAttrs),
                             (foreach(X,O),param(Data,NewAttrs,OldAttr) do
                             (filterData([X],Data,NewData),
                             write(' At':X),append_me(OldAttr,[X],OAL),
                             main(NewData,NewAttrs,OAL))
                             write(' At':X) , append_me(OldAttr,[X],OAL),
                             main(NewData,NewAttrs,OAL)))) ; writeln(''),
                             getlast_goal(Data,Att-Ans),
                             write(" Ans: "),writeln(Att-Ans),
                             append_me(OldAttr,[Att-Ans],NOAL) , write('rule :'),
                             split_rule(NOAL)) , assert(rule_me(NOAL)).
```

```
split_rule(NOAL)  :-    mem_last(NOAL,L,RL),write('if'),
                        (foreach(RLL,RL) do write(' '),write(RLL),write(' ')),
                        write('then'),write(' '),writeln(L).

%last member in list
%mem_last([1,2,3,5,6],L,RL).
mem_last([H],H,[]).
mem_last([H|T],L,[H|RL])  :-  mem_last(T,L,RL).

%mem(Data+Attrs),all_info(Data+Attrs,R).
all_info(Data+[P|Attr],R)  :-  maplist(info(Data,P), Attr, R).

%mem(Data+_), info(Data,[p-y,p-n],[o-s,o-c,o-r], R). %R=[[2,3], [4,0], [3,2]]
info(Data, P, O, O-R)  :-  maplist(info1(Data,P),O,R).
info1(Data, P, O, R)  :-  maplist(mcount(Data,O),P,R).
mcount(Data,O,P,Sum)  :-    foreach(L,Data),fromto(0,I,R,Sum),param(O,P) do
                            ((member(O,L), member(P,L))->R is I+1; R is I).

info1(Data, P, O, R)  :-  maplist(mcount(Data,O),P,R).
                          mcount(Data,O,P,Sum)  :-  foreach(L,Data),fromto(0,I,R,Sum),
                          param(O,P) do ((member(O,L), member(P,L))->R is I+1; R is I).

%avg_info(o-[[2,3],[4,0],[3,2]],R).
avg_info(O-LL,O/Info)  :-   flatten(LL,L), sumlist(L, Sum),
                            (foreach(X,LL), fromto(0,I,N,Info),param(Sum) do
                            (sumlist(X,SumX),
                             (Sum>0 -> Ratio is SumX/Sum,!; Ratio is 0),
                             logInfo(X,InfoSub),
                             N is I+(Ratio*InfoSub)
                            )).
```

```
%hasOneClass([]):-!.
%hasOneClass([_-VL|T]) :- (foreach(XL,VL) do memberchk(0,XL)), hasOneClass(T).

sumlist(L,[],L):-!.
sumlist([],[],[]):-!.
sumlist([H1|T1],[H2|T2],[HR|TR]) :- HR is H1+H2,sumlist(T1,T2,TR).

sumlists([],R,R):-!.
sumlists([H|T],PR,NR) :- sumlist(H,PR,R) , sumlists(T,R,NR).

hasOneClass([]):-!.
hasOneClass([_-VL|T]) :- sumlists(VL,[],NR) , hasOneClass(T,NR).

hasOneClass([],[]) :- !.
hasOneClass([_-VL|T],PR) :- sumlists(VL,PR,NR),hasOneClass(T,NR).
hasOneClass([],[H|T]) :- (H=0->hasOneClass([],T);find(T)).

find([]):-!.
find([H|T]) :- (H=0->find(T);false).

% get the Last data in Fist of list in list
getlast_goal([H|_],R) :- getlast(H,R).
getlast([H],H).
getlast([_|T],R) :- getlast(T,R).

%filterData
filterData(_,[],[]).
filterData(L,[H|Data],[H|R]) :- msubset(L,H) , ! , filterData(L,Data,R).
filterData(L,[H|Data],R) :- \+msubset(L,H) , ! , filterData(L,Data,R).
```

```
msubset(S1,S2)  :-  foreach(X,S1) , param(S2) do member(X,S2) , ! .


allmem([H],L)  :-  member(H,L) , ! .
allmem([H|T],L)  :-  member(H,L) , allmem(T,L).


logInfo(XL, R)  :-  sumlist(XL,Sum), Sum==0, R=99,!.
logInfo(XL, R)  :-  sumlist(XL,Sum),
                    (foreach(X,XL), fromto(0,S,N,R), param(Sum) do
                    ( Ratio is X/Sum,
                     (Ratio>0->[iso]:log(Ratio,Log);Log is 1), %log(0) is undefined
                    [iso]:log(2,Base2),
                    N is S-(Ratio*(Log/Base2)))).


chooseMin([],O/Tmp,O/Tmp).
chooseMin([A/H|T],O/Tmp,Min)  :-  (H<Tmp -> NextMin = A/H ; NextMin = O/Tmp),
chooseMin(T,NextMin,Min).
```

ประวัติผู้เขียน

นางสาวฝนทิพย์ คูณแก้ว เกิดเมื่อวันที่ 29 มิถุนายน พ.ศ. 2531 ที่จังหวัดกรุงเทพมหานคร เริ่มเข้าศึกษาระดับชั้นอนุบาล 1 จนถึงชั้นประถมศึกษาปีที่ 6 ที่โรงเรียนอุดมวิทยา อำเภอบึงสามพัน จังหวัดเพชรบูรณ์ จากนั้นได้เข้าศึกษาต่อในระดับมัธยมศึกษาตอนต้นและตอนปลาย ที่โรงเรียนบึงสามพันวิทยาคม อำเภอบึงสามพัน จังหวัดเพชรบูรณ์ ปีการศึกษา 2550 ได้เข้าศึกษาต่อระดับปริญญาตรีในสาขาวิชาวิศวกรรมคอมพิวเตอร์ สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี และสำเร็จการศึกษาเมื่อปี พ.ศ. 2553 ภายหลังสำเร็จการศึกษาในระดับปริญญาตรี ได้เข้าศึกษาในระดับปริญญาโท สาขาวิชาวิศวกรรมคอมพิวเตอร์ สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี ในปีการศึกษา 2554

ในระหว่างการศึกษาได้รับความอนุเคราะห์อย่างดียิ่งจากอาจารย์ประจำวิชา Database System ได้รับความไว้วางใจให้เป็นผู้ช่วยสอนปฏิบัติการ ผลงานวิจัยจากการทำวิทยานิพนธ์ได้รับการตีพิมพ์เผยแพร่ในลักษณะของบทความวิจัยที่มีการนำเสนอในที่ประชุมวิชาการ และบทความฉบับเต็มตีพิมพ์ในเอกสารการประชุม (Proceedings) ซึ่งรายละเอียดสามารถดูได้ที่ภาคผนวก ก

มหาวิทยาลัยเทคโนโลยีสุรนารี