ตารางที่ 3.5 เปรียบเทียบผลการจัดกลุ่มข้อมูล breast cancer ด้วยวิธีจัดกลุ่มแบบ batch และแบบ incremental

จำนวนกลุ่ม (K)	การจัดกลุ่มแบบ batch			การจัดกลุ่มแบบ incremental			ความแตกต่างของการจัดข้อมูลเข้ากลุ่ม
	จำนวนข้อมูลในกลุ่ม	เวลาที่ใช้ (วินาที)	ค่า SSE	จำนวนข้อมูลในกลุ่ม	เวลาที่ใช้ (วินาที)	ค่า SSE	
2	Cluster1=123 Cluster2=68	1.54	2013.33	Cluster1=131 Cluster2=80	1.37	1967.67	11.52%
3	Cluster1=81 Cluster2=63 Cluster3=47	2.13	1846.78	Cluster1=79 Cluster2=65 Cluster3=47	1.43	1731.11	3.66%
4	Cluster1=51 Cluster2=46 Cluster3=46 Cluster4=48	2.09	1701.26	Cluster1=49 Cluster2=47 Cluster3=49 Cluster4=46	1.51	1691.98	5.76%
5	Cluster1=43 Cluster2=38 Cluster3=35 Cluster4=37 Cluster5=38	1.47	1568.14	Cluster1=45 Cluster2=32 Cluster3=34 Cluster4=39 Cluster5=41	1.39	1446.72	7.33%
6	Cluster1=34 Cluster2=29 Cluster3=30 Cluster4=27 Cluster5=37 Cluster6=34	1.56	1432.59	Cluster1=31 Cluster2=31 Cluster3=32 Cluster4=30 Cluster5=35 Cluster6=32	1.48	1303.33	6.28%
7	Cluster1=28 Cluster2=28 Cluster3=24 Cluster4=23 Cluster5=25 Cluster6=33 Cluster7=30	2.17	1173.79	Cluster1=29 Cluster2=29 Cluster3=28 Cluster4=29 Cluster5=28 Cluster6=24 Cluster7=24	1.56	1072.13	8.37%

ตารางที่ 3.6 ค่า SSE ของการจัดกลุ่มข้อมูลตามความหนาแน่นแบบ batch และแบบ incremental ของข้อมูล post-operative

K	Clustering method	ค่า SSE ตามความหนาแน่นของข้อมูล											
		[1,0.1]	[1,0.15]	[1,0.2]	[2,0.1]	[2,0.15]	[2,0.2]	[3,0.1]	[3,0.15]	[3,0.2]	[4,0.1]	[4,0.15]	[4,0.2]
2	Batch	1739.1	1740.6	1749.3	1702.3	1716.9	1741.1	1713.3	1701.1	1722.4	1756.3	1754.8	1769.5
	Incremental	1670.3	1687.3	1675.1	1689.0	1691.8	1703.7	1643.4	1629.7	1707.4	1775.6	1684.6	1795.3
3	Batch	1445.6	1462.9	1508.9	1677.5	1544.2	1493.2	1389.9	1401.2	1599.7	1611.0	1543.5	1679.9
	Incremental	1395.5	1401.2	1434.7	1504.1	1400.2	1542.7	1358.2	1341.4	1477.8	1489.5	1531.2	1622.1
4	Batch	1372.2	1387.1	1401.0	1512.9	1488.7	1395.5	1321.1	1356.7	1407.9	1417.3	1399.0	1542.3
	Incremental	1311.3	1402.6	1398.7	1476.5	1359.2	1344.6	1305.2	1296.1	1307.7	1369.0	1304.4	1411.8
5	Batch	1196.7	1209.3	1136.5	1227.6	1235.5	1409.9	1287.6	1012.5	1293.7	1785.2	1303.3	1224.7
	Incremental	1112.2	1301.0	1198.6	1211.1	1109.5	1245.0	1193.3	1074.6	1249.0	1329.5	1268.7	1143.6
6	Batch	1071.9	1192.7	1068.5	1201.9	1137.8	1224.3	1125.6	1016.5	1298.0	1173.4	1243.0	1195.2
	Incremental	1132.4	1098.3	1163.4	1255.7	1190.4	1207.3	1114.8	1079.8	1212.5	1249.0	1300.7	1211.1
7	Batch	978.3	895.6	913.2	965.1	970.2	899.0	901.7	816.5	893.2	907.6	954.5	972.3
	Incremental	913.3	941.6	932.8	899.0	928.4	919.6	829.5	807.7	901.3	916.5	907.4	932.6
8	Batch	965.4	1008.2	974.7	966.2	1105.7	989.3	891.4	902.6	972.4	1123.7	1065.9	993.7
	Incremental	891.2	903.4	876.2	1025.3	998.6	916.7	875.3	892.0	913.4	936.5	997.8	1012.1
9	Batch	901.3	932.7	890.5	912.7	934.1	899.6	813.2	829.4	937.6	929.4	956.5	948.2
	Incremental	876.3	832.7	890.1	885.9	906.7	813.2	836.7	891.0	903.7	919.5	896.0	932.4
10	Batch	896.5	942.1	937.6	881.3	920.5	874.3	832.6	879.5	916.7	903.2	924.0	996.4
	Incremental	901.2	928.7	936.5	890.1	879.0	892.6	851.1	900.3	899.6	917.2	906.1	932.4
11	Batch	872.4	884.5	832.1	864.3	901.1	856.2	813.4	876.7	913.2	892.0	936.5	912.7
	Incremental	906.3	872.5	843.1	892.7	819.9	769.5	793.3	812.7	939.6	895.4	901.8	879.2
12	Batch	883.1	872.9	853.1	902.4	875.9	836.2	813.7	859.0	913.4	898.2	913.2	906.5
	Incremental	792.8	813.5	826.7	913.8	885.4	763.2	799.0	834.2	865.9	913.4	920.6	895.4

ตารางที่ 3.7 ค่า SSE ของการจัดกลุ่มข้อมูลตามความหนาแน่นแบบ batch และแบบ incremental ของข้อมูล breast cancer

K	Clustering method	ค่า SSE ตามความหนาแน่นของข้อมูล											
		[1,0.1]	[1,0.15]	[1,0.2]	[2,0.1]	[2,0.15]	[2,0.2]	[3,0.1]	[3,0.15]	[3,0.2]	[4,0.1]	[4,0.15]	[4
2	Batch	2046.3	2137.0	2246.9	2012.5	2354.1	2168.4	2079.3	1958.2	2143.2	2013.4	2019.9	2
	Incremental	1975.7	1983.2	2034.1	1989.7	2154.3	2011.9	1964.3	1941.7	1985.0	2054.2	2098.7	2
3	Batch	1982.3	2013.5	2163.4	2098.2	2175.9	1972.6	2001.8	1935.1	1972.7	1908.3	2085.2	2
	Incremental	1763.4	1865.4	1795.0	1932.4	1955.7	1843.9	1932.3	1832.7	1954.0	1885.9	1932.7	1
4	Batch	1712.6	1698.7	1734.6	1800.3	1763.9	1756.4	1831.2	1749.2	1654.1	1693.2	1834.9	1
	Incremental	1678.3	1732.1	1688.7	1709.7	1789.5	1703.2	1688.4	1603.0	1642.5	1659.2	1701.3	1
5	Batch	1543.1	1612.3	1593.8	1600.1	1572.8	1559.6	1482.0	1501.3	1459.7	1532.7	1468.9	1
	Incremental	1437.3	1389.3	1372.7	1401.3	1388.2	1369.4	1419.5	1344.2	1465.1	1370.6	1458.7	1
6	Batch	1402.9	1472.5	1439.0	1372.6	1385.0	1409.2	1376.9	1312.1	1308.9	1427.1	1365.7	1
	Incremental	1371.7	1362.8	1398.4	1432.5	1411.0	1349.6	1451.0	1327.8	1309.5	1420.7	1372.6	1
7	Batch	1372.5	1419.2	1352.6	1275.7	1293.0	1321.7	1286.5	1203.4	1384.2	1358.7	1402.6	1
	Incremental	1218.9	1239.0	1311.2	1168.7	1203.4	1195.9	1202.7	1134.9	1310.2	1287.4	1197.3	1
8	Batch	1132.7	1068.2	1143.7	1079.2	1159.0	1086.5	1034.2	1007.8	1013.4	1179.0	1092.3	1
	Incremental	1107.6	1195.7	1183.2	1096.7	1132.4	1097.5	1106.3	1002.4	1146.9	1093.7	1089.0	1
9	Batch	893.6	912.4	895.6	903.1	943.5	879.6	907.4	859.6	935.2	911.7	909.3	8
	Incremental	932.7	942.7	939.5	891.2	903.4	950.2	875.1	832.7	913.2	894.7	953.0	9
10	Batch	793.9	765.9	801.4	865.9	793.2	786.4	713.0	759.2	803.4	860.9	903.1	8
	Incremental	684.9	713.2	695.7	801.5	738.2	679.8	695.0	713.2	699.3	715.4	769.2	8
11	Batch	714.5	726.9	774.0	693.1	732.9	680.2	613.4	695.2	787.0	695.3	761.2	7
	Incremental	732.2	695.3	689.4	719.0	677.0	613.5	589.0	632.4	715.6	709.8	674.9	6
12	Batch	632.8	579.2	684.3	588.7	516.5	543.2	508.7	640.1	632.2	619.5	593.2	5
	Incremental	588.7	535.2	645.9	607.1	580.3	511.7	573.2	546.0	532.9	611.4	590.2	5

อภิปรายผล

การจัดกลุ่มข้อมูลแบบ batch และแบบ incremental เมื่อไม่ต้องคำนึงถึงความหนาแน่นของข้อมูล (ผลการทดสอบตามตารางที่ 3.1 และ 3.2) พบว่าการจัดกลุ่มข้อมูลแบบ incremental ใช้เวลาโดยรวมน้อยกว่าแบบ batch และข้อมูลที่ถูกจัดเข้ากลุ่มในขั้นสุดท้ายของทั้งสองวิธีการมีความแตกต่างกันไม่เกิน 12% เมื่อพิจารณาคุณภาพของกลุ่มข้อมูลจากค่า SSE พบว่าวิธีจัดกลุ่มข้อมูลแบบ incremental ให้ผลลัพธ์เป็นกลุ่มข้อมูลที่เกาะกลุ่มกันค่อนข้างดี และการเกาะกลุ่มจะดีขึ้นเมื่อเพิ่มจำนวนกลุ่ม

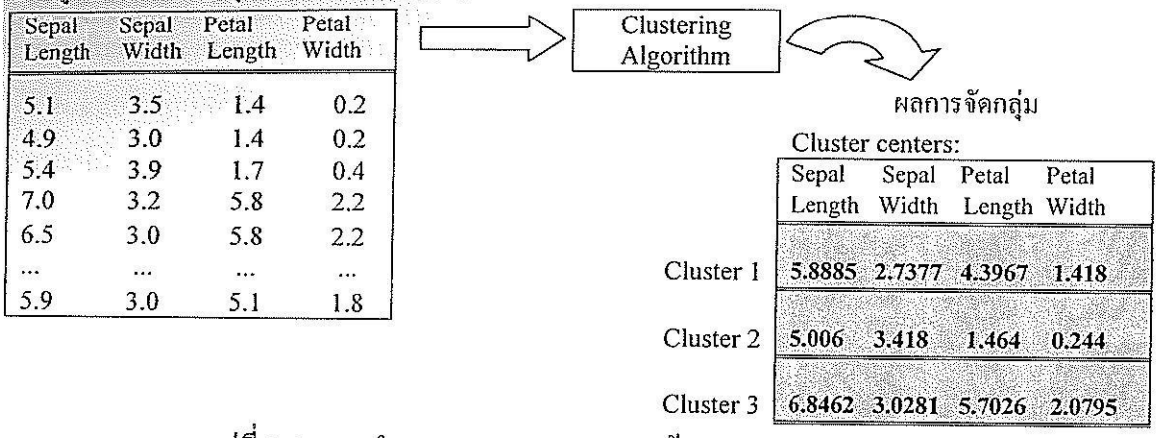
เมื่อทดลองคัดเลือกข้อมูลตัวแทนตามค่าความหนาแน่น เพื่อนำมาใช้จัดกลุ่มและหาค่า mean ของกลุ่ม พบว่า (ตามผลที่ได้ในตารางที่ 3.4 และ 3.5) วิธีการคัดเลือกข้อมูลตัวแทน จะให้ผลการจัดกลุ่มที่ดีกว่าการใช้ข้อมูลทั้งหมดมาคำนวณหาค่า mean ของกลุ่ม และเกณฑ์ขั้นต่ำในการคัดเลือกข้อมูลตัวแทนอยู่ที่ประมาณ [3,0.1], [3,0.15] และ [4,0.1] นั่นคือควรจะพิจารณาคัดเลือกข้อมูลตัวแทนที่มีค่าความหนาแน่นอยู่ระหว่าง 0.1 ถึง 0.15 (หรือเป็นข้อมูลที่มีข้อมูลอื่นอยู่ใกล้เคียงอย่างต่ำ 10-15%) และการพิจารณาความใกล้เคียงใช้การเปรียบเทียบค่าของแอททริบิวต์อย่างต่ำ 3 หรือ 4 แอททริบิวต์

3.3 สรุปผลโครงการวิจัยที่ 2

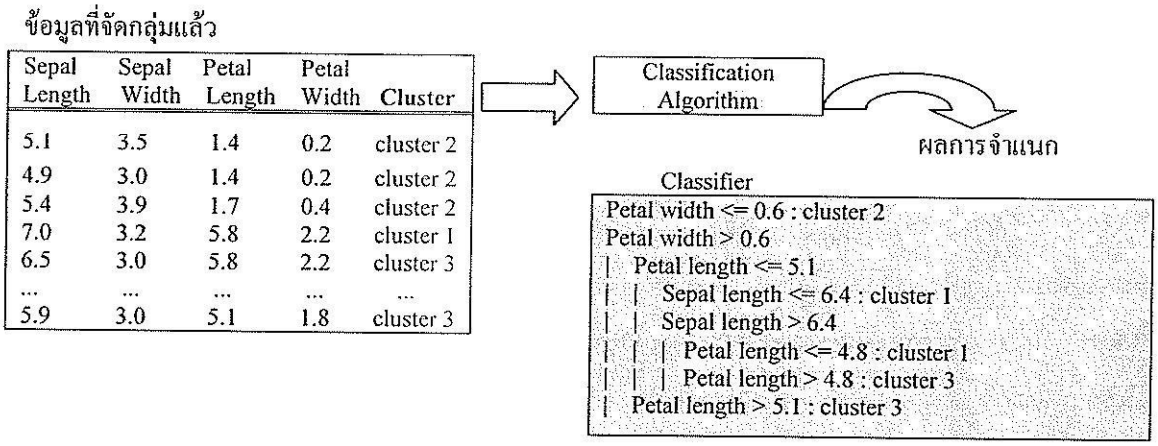
การวิเคราะห์ข้อมูลด้วยวัตถุประสงค์ที่จะค้นหารูปแบบ หรือลักษณะเด่นภายในกลุ่มข้อมูลเพื่อที่จะนำความรู้ที่ได้ไปใช้ประโยชน์ มักจะเริ่มต้นกระบวนการวิเคราะห์ด้วยการแยกข้อมูลออกเป็นกลุ่ม ภายในแต่ละกลุ่มจะประกอบด้วยข้อมูลที่คล้ายกัน ต่อจากนั้นจึงจะเป็นการวิเคราะห์โดยละเอียดกับข้อมูลแต่ละกลุ่มเพื่อหาลักษณะร่วม, แนวโน้มการเปลี่ยนแปลง หรือลักษณะอื่นๆ ตามที่นักวิเคราะห์สนใจ การจัดกลุ่มจึงมักจะเป็นขั้นตอนแรกของกระบวนการวิเคราะห์ข้อมูล และเป็นขั้นตอนการเรียนรู้ที่เรียกว่า unsupervised learning เนื่องจากไม่มีการชี้แนะว่าควรจะจัดข้อมูลเป็นกลุ่มโดยพิจารณาจากลักษณะใดของข้อมูล การจัดกลุ่มจึงต้องพิจารณาทุกลักษณะในข้อมูลแต่ละตัว ซึ่งต่างจากขั้นตอนหลังจากนี้ที่ข้อมูลได้รับการกำหนดกลุ่มแล้ว การวิเคราะห์ข้อมูลจึงจะมีลักษณะเจาะจงมากขึ้น เช่น กำหนดให้ศึกษาลักษณะของข้อมูลในกลุ่มที่ 1 ว่าแตกต่างจากข้อมูลในกลุ่มที่ 2 อย่างไร การศึกษาลักษณะภายในข้อมูลที่เจาะจงมากขึ้นนี้เรียกว่า supervised learning

ตัวอย่างในรูปที่ 3.12 แสดงการจัดกลุ่มข้อมูล iris ด้วยโปรแกรม k-means ข้อมูลประกอบด้วยความยาวและความกว้างของกลีบดอกไอริส 3 สายพันธุ์ ผลลัพธ์ของการจัดกลุ่มจะได้เป็นค่า mean ของความยาวและความกว้างของกลีบดอกไอริสในแต่ละกลุ่ม เมื่อจัดกลุ่มข้อมูลได้แล้วจึงจะนำข้อมูลที่มีการระบุกลุ่ม (รูปที่ 3.13) ไปวิเคราะห์ด้วยอัลกอริทึม classification ซึ่งจัดเป็นอัลกอริทึมประเภท supervised learning เนื่องจากได้รับการชี้แนะว่าให้จำแนกข้อมูลตามกลุ่ม

(cluster) ผลลัพธ์ที่ได้จึงเป็นการจำแนกลักษณะของข้อมูลในกลุ่มที่ 1, 2 และ 3 หรืออาจพิจารณาได้ว่าเป็นการบรรยาย (describe) ลักษณะของข้อมูลในแต่ละกลุ่ม



รูปที่ 3.12 การทำ unsupervised learning ด้วย clustering algorithm



รูปที่ 3.13 การทำ supervised learning ด้วย classification algorithm

การจัดกลุ่มข้อมูลจึงเป็นขั้นตอนที่สำคัญของงานวิเคราะห์ข้อมูล และจากลักษณะของการเรียนรู้แบบ unsupervised learning เวลาที่ใช้ในการทำงานจึงมักจะสูงกว่าการเรียนรู้ในแบบ supervised learning เวลาที่ใช้นี้จะผันแปรตามจำนวนมิติของข้อมูลและปริมาณของข้อมูล เมื่อข้อมูลมีปริมาณสูงมากอัลกอริทึมการจัดกลุ่มข้อมูลจะทำงานได้ช้าลง หรือถ้าข้อมูลมีปริมาณมากจนเกินไปจะไม่สามารถทำงานได้เนื่องจากปริมาณข้อมูลมีมากเกินกว่าความจุของหน่วยความจำ ถ้าต้องการปรับปรุงอัลกอริทึมจัดกลุ่มข้อมูลให้สามารถทำงานกับข้อมูลปริมาณสูงมากได้จะต้องใช้วิธีการจัดกลุ่มข้อมูลแบบเพิ่มพูนและใช้ข้อมูลตัวแทนในการพิจารณาค่า mean ของกลุ่ม

โครงการวิจัยนี้มีวัตถุประสงค์หลักคือ การออกแบบและพัฒนาเทคนิคการจัดกลุ่มข้อมูล โดยในโครงการวิจัยนี้ได้นำเสนอวิธีการจัดกลุ่มข้อมูลในแบบปกติ เรียกว่าการจัดกลุ่มข้อมูล

แบบ batch นั่นคือใช้ข้อมูลทั้งหมดในคราวเดียวเพื่อการจัดกลุ่มข้อมูล และนำเสนอวิธีการจัดกลุ่มข้อมูลแบบเพิ่มพูน หรือ incremental clustering ที่ใช้การแบ่งข้อมูลที่จะจัดกลุ่มออกเป็นส่วนย่อย จากนั้นจัดกลุ่มข้อมูลในส่วนย่อย แล้วจึงนำค่า mean ของกลุ่มข้อมูลซึ่งเป็นผลลัพธ์ที่ได้จากการจัดกลุ่มข้อมูลย่อย มา merge เพื่อค้นหาค่า mean ใหม่ที่เหมาะสมจะเป็นลักษณะตัวแทนของกลุ่ม ผลการทดลองพบว่า วิธีการจัดกลุ่มข้อมูลแบบ batch และแบบ incremental ให้ผลลัพธ์สุดท้ายเป็นกลุ่มข้อมูลที่มีสมาชิกในกลุ่มไม่แตกต่างกันมาก แต่วิธี incremental clustering จะใช้เวลาในการประมวลผลน้อยกว่า

นอกจากเทคนิค incremental clustering แล้ว โครงการวิจัยนี้ยังได้นำเสนอเทคนิคการจัดกลุ่มข้อมูลตามความหนาแน่น โดยใช้วิธีการคัดเลือกข้อมูลที่มีความหนาแน่นตรงตามเกณฑ์ที่กำหนด มาพิจารณาค่า mean ของกลุ่ม เพื่อที่ในขั้นตอนสุดท้ายจะใช้ค่า mean ที่ได้เพื่อการพิจารณาจัดข้อมูลเข้ากลุ่ม ผลการทดลองพบว่าวิธีการจัดกลุ่มข้อมูลตามความหนาแน่น สามารถให้ผลการจัดกลุ่มที่ดี โดยมีค่า sum of squared error รวมในทุกกลุ่มต่ำกว่าวิธีจัดกลุ่มข้อมูลที่ใช้ข้อมูลทุกตัวเพื่อคำนวณค่า mean

ข้อเสนอแนะ

การทดลองในโครงการวิจัยนี้ ได้ทดสอบประสิทธิภาพของเทคนิคการจัดกลุ่มข้อมูลกับข้อมูลขนาดเล็ก (14 เรคคอร์ด) และข้อมูลขนาดกลาง (81 และ 191 เรคคอร์ด) ถึงแม้ว่าผลการทดลองจะยืนยันว่าเทคนิค density-biased clustering และวิธีการจัดกลุ่มข้อมูลแบบ incremental ให้ผลลัพธ์ที่ดีกว่าการจัดกลุ่มข้อมูลตามปกติ แต่การนำซอฟต์แวร์ที่พัฒนาขึ้นไปใช้งานจริง ยังต้องการการทดสอบกับข้อมูลจำนวนมากกว่านี้ และควรจะทดสอบเพิ่มเติมกับข้อมูลขนาดใหญ่ที่มีจำนวนมากกว่า 10,000 เรคคอร์ด

ซอฟต์แวร์ที่พัฒนาขึ้นนี้เน้นการจัดกลุ่มข้อมูลที่มีชนิดข้อมูลเป็นข้อความ (nominal, categorical) การจัดกลุ่มข้อมูลที่แอททริบิวต์มีค่าเป็นจำนวนเลข สามารถทำได้โดยปรับปรุงซอฟต์แวร์นี้เพียงเล็กน้อย แต่ถ้าค่าตัวเลขในแอททริบิวต์มีช่วงของค่าที่เป็นไปได้กว้างมาก เช่นในกรณีแอททริบิวต์อายุ ที่มีค่าอยู่ระหว่าง 0-120 ควรเพิ่มฟังก์ชัน normalization เช่นใช้เทคนิค max-min normalization หรือ z-score standardization เพื่อปรับลดช่วงกว้างของค่าสูงสุด-ต่ำสุด ทั้งนี้เพื่อลด bias ของการคำนวณค่า mean ของกลุ่มข้อมูล

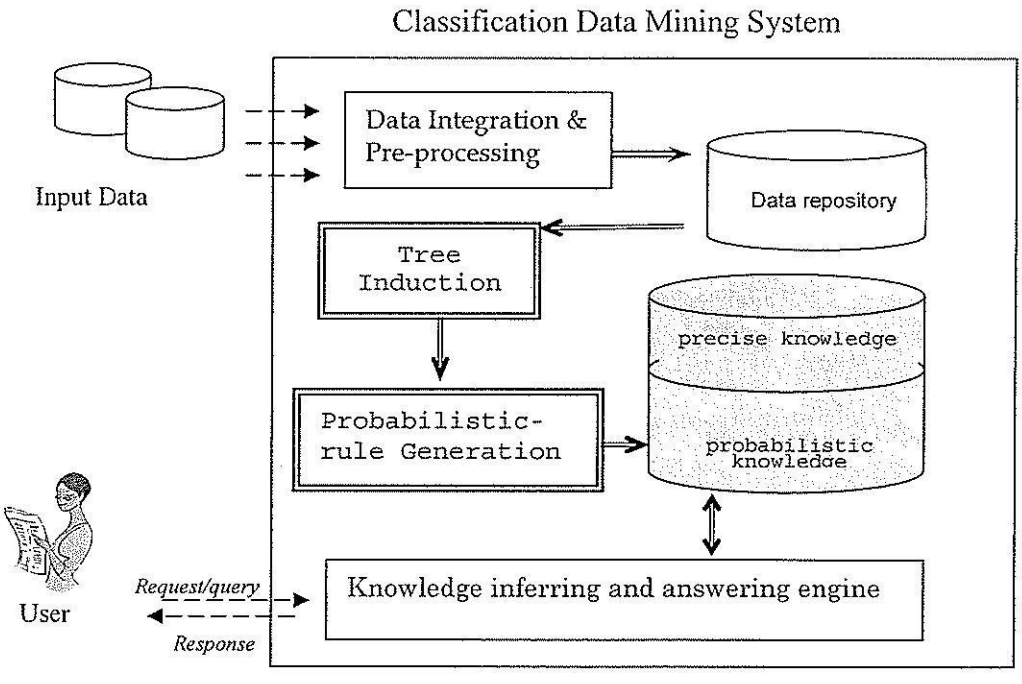
บทที่ 4

การพัฒนาการทำเหมืองข้อมูลแบบจำแนก (โครงการวิจัยที่ 3)

4.1 วิธีดำเนินการวิจัยโครงการวิจัยที่ 3

4.1.1 กรอบแนวคิดของงานวิจัย

วัตถุประสงค์หลักของโครงการวิจัยการพัฒนาการทำเหมืองข้อมูลแบบจำแนก คือ การพัฒนาซอฟต์แวร์เพื่อสร้าง classifier หรือตัวจำแนก ซึ่งจะเป็นโมเดลที่สามารถนำไปใช้จำแนกประเภทข้อมูลหรือจำแนกค่าที่สนใจในข้อมูลที่จะเกิดขึ้นใหม่ในอนาคต การสร้างโมเดลใช้เทคนิคของต้นไม้ตัดสินใจเชิงอุปนัย และเพื่อป้องกันความบกพร่องของโมเดลในกรณีที่มีข้อมูลรบกวนปรากฏอยู่ในชุดข้อมูลฝึก งานวิจัยนี้จึงใช้การคำนวณค่าความน่าจะเป็นของการปรากฏข้อมูลในแต่ละกิ่งของโครงสร้างต้นไม้ตัดสินใจ เพื่อเรียงลำดับกฎการตัดสินใจ หรือ decision rules ที่แปลงมาจากต้นไม้ตัดสินใจตามสัดส่วนที่แต่ละกฎสามารถอธิบายข้อมูลได้ (หรือเรียกว่าค่า coverage ของกฎการตัดสินใจ) ผลลัพธ์สุดท้ายที่ได้จะเป็นเซตของ probabilistic decision rules ที่สามารถคัดเลือกไปเก็บไว้ในฐานความรู้เพื่อการใช้งานต่อไป แนวคิดโดยรวมของงานวิจัยนี้แสดงได้ดังรูปที่ 4.1



รูปที่ 4.1 กรอบแนวคิดของการออกแบบและพัฒนาซอฟต์แวร์เพื่อการทำเหมืองข้อมูลแบบจำแนก

ในกรอบเชิงแนวคิดของระบบเหมืองข้อมูลเพื่อการจำแนก (Classification data mining system) การทำงานของระบบจะต้องใช้ส่วนประกอบแรกคือ Data integration and Pre-processing ทำหน้าที่รวมข้อมูลที่อาจจะถูกส่งมาจากหลายแหล่ง และจัดรูปแบบข้อมูลที่อาจแตกต่างกันให้อยู่ในมาตรฐานที่กำหนดไว้ (ในงานวิจัยนี้ใช้มาตรฐาน Horn clause ของภาษา Prolog นั่นคือข้อมูลแต่ละรายการหรือแต่ละเรคคอร์ดจะเป็นหนึ่งคลอสหรือหนึ่งข้อความตามรูปแบบ fact ของ Prolog) และถ้าข้อมูลมีขนาดใหญ่มากผู้ใช้งานระบบสามารถเรียกใช้เครื่องมือ Sampling ในส่วน Pre-processing เพื่อคัดเลือกข้อมูลตัวแทนมาใช้ในขั้นตอนการเรียนรู้เพื่อสร้างโมเดลของข้อมูล ในกรณีที่ข้อมูลมีแอททริบิวต์หรือ feature มากเกินจำเป็น ผู้ใช้งานระบบสามารถสั่งผ่านฟังก์ชัน Feature selection ให้คัดเลือกเฉพาะ feature ที่คาดว่าจะเป็นประโยชน์ในการเรียนรู้เพื่อสร้างโมเดล ข้อมูลที่ผ่านการทำงานของส่วน Data integration and Pre-processing จะถูกเก็บไว้ในฐานข้อมูลในลักษณะของ Prolog data file (หมายถึงไฟล์ข้อมูลที่มีส่วนขยายเป็น .pl)

ส่วนประกอบส่วนที่สองคือ Tree induction ซึ่งเป็นส่วน mining engine ที่ทำหน้าที่อ่านข้อมูลและสร้างโมเดลข้อมูลในลักษณะของต้นไม้ตัดสินใจ เมื่ออ่านข้อมูลจนจบไฟล์และการสร้างต้นไม้ตัดสินใจเสร็จสิ้น โครงสร้างต้นไม้ทั้งต้นที่แทนด้วยโหนดและกิ่งต่างๆของโหนดตั้งแต่โหนดรากจนกระทั่งถึงโหนดใบ จะถูกส่งต่อไปยังส่วนที่สามที่ทำหน้าที่แปลงจากต้นไม้ตัดสินใจให้เป็นกฎการตัดสินใจ

ส่วนของ Probabilistic-rule generation จะทำหน้าที่ traverse ไปตามกิ่งต่างๆของต้นไม้พร้อมทั้งคำนวณสัดส่วนของข้อมูลที่ปรากฏในแต่ละกิ่ง เพื่อกำหนดเป็นค่าความน่าจะเป็นในการครอบคลุมข้อมูลของแต่ละกฎการตัดสินใจ ในขั้นตอนนี้ผู้ใช้สามารถกำหนดค่าเกณฑ์ขั้นต่ำของค่าความน่าจะเป็นเพื่อคัดเลือกเฉพาะกฎที่อธิบาย (หรือ cover) ข้อมูลได้มาก กฎการตัดสินใจที่ผ่านเกณฑ์คัดเลือกจะถูกบันทึกไว้ในฐานความรู้ เพื่อการใช้งานต่อไปในกระบวนการสอบถามความรู้ของระบบผู้เชี่ยวชาญหรือระบบฐานความรู้ แต่เนื่องจากความรู้ที่ได้จากการทำเหมืองข้อมูลเป็นความรู้ที่สังเคราะห์จากข้อมูลที่เกิดขึ้นจริงในอดีต ข้อมูลอาจจะมีข้อผิดพลาดและข้อมูลเหล่านี้อาจไม่แสดงรูปแบบหรือโมเดลที่ถูกต้องครบถ้วนได้ทั้งหมด จึงต้องแยกความรู้ที่ได้จากการทำเหมืองข้อมูลไว้ในฐานความรู้ส่วนที่เรียกว่า probabilistic knowledge ในส่วนของความรู้ที่ได้จากการสัมภาษณ์ผู้เชี่ยวชาญซึ่งมักจะเป็นความรู้ที่มีความถูกต้องสูงจะแยกไว้เป็นส่วน precise knowledge

เมื่อมีการสอบถามหรือปรึกษาระบบผู้เชี่ยวชาญ ส่วนประกอบที่เรียกว่า Knowledge inferring and answering engine จะทำหน้าที่ติดต่อกับฐานความรู้ทั้งในส่วน precise knowledge และ probabilistic knowledge เพื่ออนุมานคำตอบที่ใกล้เคียงกับข้อสอบถามของผู้ใช้ให้มากที่สุด

การออกแบบและพัฒนาซอฟต์แวร์ในระบบทำเหมืองข้อมูลเพื่อการจำแนกนี้ จะเน้นที่ส่วนประกอบหลักสองส่วนคือ Tree induction และ Probabilistic-rule generation ทั้งนี้เนื่องจากใน

ส่วนของ Data integration and Pre-processing และ Knowledge inferring and answering engine จะปรากฏรายละเอียดในบทที่ 2 และ 5 เรื่อง "วิธการเตรียมข้อมูลอัตโนมัติก่อนการทำเหมืองข้อมูล" และ "การประมวลผลหลังกระบวนการทำเหมืองข้อมูล" ตามลำดับ

4.1.2 การออกแบบอัลกอริทึมเพื่อการทำเหมืองข้อมูลแบบจำแนก

ในการออกแบบอัลกอริทึมที่เป็นโมดูลหลักของระบบเหมืองข้อมูลเพื่อการจำแนก จะออกแบบสอดคล้องกับกรอบแนวคิดของระบบที่แสดงดังรูปที่ 4.1 ดังนั้นระบบนี้จะประกอบด้วยสองโมดูลคือ Tree-induction และ Probabilistic-rule generation รายละเอียดของแต่ละอัลกอริทึมแสดงได้ดังต่อไปนี้

โมดูล Tree-induction

อัลกอริทึมในการสร้างต้นไม้ตัดสินใจประกอบด้วยสองขั้นตอนหลักคือการอ่านข้อมูลทั้งหมดพร้อมทั้งกำหนดค่าเริ่มต้นตัวนับหมายเลขโหนด และขั้นตอนการสร้างต้นไม้ โดยในขั้นตอนสร้างต้นไม้นี้จะวนทำซ้ำจนกระทั่งแยกข้อมูลได้สมบูรณ์

Algorithm 4.1 Tree-induction

Input: a data set formatted as Prolog clauses
Output: a decision tree with node/2 and edge/3 structures

(1) Initialization
  (1.1) Clear temporary knowledge base (KB) by removing all information regarding the predicates node/2, edge/3 and current_node/1
  (1.2) Set node counter = 0
  (1.3) Scan data set to get information about data attributes, positive instances, negative instances, total data instances

(2) Building tree
  (2.1) Increment node counter
  (2.2) Repeat steps 2.2.1-2.2.4 until there is no more attributes left for creating decision attributes
    (2.2.1) Compute Info value of each candidate attribute
    (2.2.2) Choose the attribute that yields minimum Info to be decision node at current tree level
    (2.2.3) Assert edge/3 and node/2 information into KB
    (2.2.4) Split data instances along node branches
  (2.3) Repeat steps 2.1 and 2.2 until the lists of positive and negative instances are empty
  (2.4) Output tree structure containing node/2 and edge/3 predicates

---

อัลกอริทึม Tree-induction ทำหน้าที่รับข้อมูลเข้าในลักษณะของ Prolog clauses จากนั้นใช้ค่าแอททริบิวต์ที่ปรากฏในข้อมูลแต่ละรายการ สร้างโครงสร้าง node และ edge ของ

ต้นไม้ตัดสินใจ เมื่ออัลกอริทึมอ่านข้อมูลเข้าครบหมดแล้วจะเริ่มกระบวนการสร้างโครงสร้างต้นไม้ (ขั้นตอนที่ 2 ของอัลกอริทึม Tree-induction) ในขั้นตอนย่อยที่ 2.2 เป็นการคัดเลือกแอททริบิวต์ที่ให้ค่า Gain สูงที่สุด โดยค่า Gain นี้จะคำนวณจากฟังก์ชัน Info ที่ทำหน้าที่คำนวณสัดส่วนของจำนวนข้อมูลที่เป็นคลาส positive -- P(p) และสัดส่วนของจำนวนข้อมูลที่เป็นคลาส negative -- P(n) ตัวแปร p หมายถึงจำนวน positive instances (เช่นข้อมูลที่มีค่า class = yes) และ n หมายถึงจำนวน negative instances (เช่นข้อมูลที่มีค่า class = no) สูตรคำนวณค่า Info และ Gain (Quinlan 1993) แสดงได้ดังนี้

$$Info(P(p),P(n)) = -P(p)\log_2 P(p) - P(n)\log_2 P(n)$$

$$Gain(Attribute) = Info(\frac{p}{p+n},\frac{n}{p+n}) - \sum_{i=1}^{v}\frac{p_i+n_i}{p+n} Info(\frac{p_i}{p_i+n_i},\frac{n_i}{p_i+n_i})$$

ฟังก์ชัน Info(P(p), P(n)) จะให้ค่าสัดส่วนการปะปนกันของข้อมูลทั้งที่เป็น positive instances และ negative instances แอททริบิวต์ที่ให้ค่า Gain สูงที่สุดหมายถึงแอททริบิวต์นั้นเมื่อนำมาใช้เป็นโหนดของต้นไม้ตัดสินใจสามารถแยก positive instances ออกจาก negative instances ได้ดีที่สุด แอททริบิวต์ที่ถูกเลือกจะถูกบันทึกค่าเป็น edge ของต้นไม้ โครงสร้างของ node และ edge กำหนดไว้ดังนี้

node(nodeID, [Positive_Instances]-[Negative_Instancess])

edge(ParentNode, EdgeLabel, ChildNode)

โหนดแต่ละโหนดจะมีหมายเลขกำกับพร้อมทั้งระบุข้อมูล (ข้อมูลแต่ละรายการจะมีหมายเลขกำกับเช่นกัน) ในโหนดนั้นทั้งในส่วนที่เป็น positive instances และ negative instances ในส่วนของโครงสร้างกิ่งหรือ edge จะระบุข้อมูลสามส่วนคือ หมายเลขโหนดที่เป็นโหนดแม่ของกิ่งนั้น, ชื่อของ edge (คือค่าของแอททริบิวต์ที่ถูกเลือกเป็นโหนดแม่ ดังนั้นจำนวน edge ที่แตกออกจากโหนดจะมีจำนวนเท่ากับจำนวนค่าที่เป็นไปได้ของแอททริบิวต์นั้น) และหมายเลขของโหนดที่เป็นโหนดลูก

กระบวนการสร้างโหนดและ edge จะดำเนินซ้ำไปเช่นนี้จนกว่าข้อมูลที่เป็น positive instances และ negative instances จะถูกแยกออกจากกันโดยสมบูรณ์ หรือจนกระทั่งไม่มีแอททริบิวต์เหลือให้ใช้สร้างต้นไม้ได้อีกต่อไป เมื่อเสร็จกระบวนการในขั้นตอนนี้ข้อมูล node และ edge ทั้งหมดที่บันทึกไว้จะถูกส่งต่อไปยังโมดูล Probabilistic-rule generation

โมดูล Probabilistic-rule generation

อัลกอริทึมในส่วนนี้ทำหน้าที่รับข้อมูล node และ edge จากโมดูลที่ทำหน้าที่สร้างต้นไม้ตัดสินใจ รวมทั้งรับค่าความน่าจะเป็นขั้นต่ำจากผู้ใช้ ผลลัพธ์สุดท้ายที่ได้จากอัลกอริทึมคือตัวจำแนกข้อมูล ที่แสดงในลักษณะของกฎการตัดสินใจ หรือ decision rules โดยกฎนี้จะเรียงลำดับตามค่าความน่าจะเป็นสูงสุดลดหลั่งลงมาจนกระทั่งถึงกฎการตัดสินใจที่มีค่าความน่าจะเป็นต่ำสุดตามที่ผู้ใช้กำหนด รายละเอียดขั้นตอนต่างๆในอัลกอริทึมแสดงได้ดังต่อไปนี้

Algorithm 4.2 Probabilistic-rule generation

Input: a decision tree with node/2 and edge/3 structures, and
a probability threshold

Output: a set of probabilistic decision rules ranking in descending order

(1) Traverse tree from a root node to each leaf node
  (1.1) Collect edge information and count number of data instances
  (1.2) Compute probability as a proportion
  (number of instances at leaf node) / (total data instances in a data set)
  (1.3) Assert a rule containing a triplet
  <attribute-value pair, class, probability value> into temporary KB
(2) Sort rules in the KB in descending order according to the rules' probability
(3) Remove rules that have probability less than the specified threshold
(4) Assert selected rules into the KB and return KB as an output

---

ขั้นตอนแรกของอัลกอริทึมเป็นการอ่านข้อมูลจาก node และ edge เรียงลำดับตามค่าหมายเลขโหนด โดยเริ่มจากโหนดรากที่เป็นโหนดหมายเลขศูนย์ไล่ตามแต่ละกิ่งหรือ edge ลงมาจนกระทั่งถึงโหนดใบ ในขณะที่อ่านข้อมูลในแต่ละกิ่งจะนับจำนวนข้อมูลที่ถูกแยกย่อยลงมาในแต่ละกิ่งเพื่อคำนวณค่าความน่าจะเป็นในการครอบคลุมข้อมูลของกฎการตัดสินใจ ค่าความน่าจะเป็นนี้เป็นสัดส่วนระหว่างจำนวนข้อมูลที่โหนดใบหารด้วยจำนวนข้อมูลทั้งหมด (นั่นคือหารด้วยข้อมูลที่โหนดราก) จากนั้นบันทึกกฎการตัดสินใจที่แต่ละกฎมีส่วนประกอบ 3 ส่วน คือ ค่าของแอททริบิวต์ที่เป็นปัจจัยประกอบการตัดสินใจ, คลาสของข้อมูลที่เป็นผลของการตัดสินใจ และ ค่าความน่าจะเป็นของกฎการตัดสินใจ

เมื่อแปลงโครงสร้างต้นไม้ตัดสินใจเป็นกฎการตัดสินใจได้ครบในทุกกิ่ง และทุกโหนดใบแล้ว จะมีการเรียงลำดับกฎตามค่าความน่าจะเป็น (ขั้นตอนที่ 2 ตามอัลกอริทึม) โดยเรียงจากค่ามากสุดลงมาหาค่าน้อยที่สุด ต่อจากนั้นจะตัดทิ้งกฎที่มีค่าความน่าจะเป็นต่ำกว่าเกณฑ์ที่ผู้ใช้กำหนด (ขั้นตอนที่ 3 ตามอัลกอริทึม) และในขั้นตอนสุดท้ายเป็นการบันทึกกฎการตัดสินใจลงในฐานความรู้

4.1.3 การพัฒนาโปรแกรมเพื่อการทำเหมืองข้อมูลแบบจำแนก

โปรแกรมเพื่อการทำเหมืองข้อมูลแบบจำแนกนี้พัฒนาขึ้นโดยใช้ภาษาโปรล็อก การอธิบายขั้นตอนพัฒนาโปรแกรมจะใช้ไวยากรณ์ของภาษาโปรล็อกตามมาตรฐานของ SWI Prolog เวอร์ชัน 5.6.55 (ดาวน์โหลดได้จากเว็บไซต์ www.swi-prolog.org) ลักษณะเด่นประการหนึ่งของภาษาโปรล็อกคือการใช้รูปแบบเดียวกันของทั้งข้อมูลและคำสั่งที่ทำงานกับข้อมูล

รูปแบบของข้อมูล

ข้อมูลฝึกที่จะเป็นอินพุทของโปรแกรม จะมีลักษณะเป็นข้อความที่อยู่ในรูปแบบของข้อความที่เป็นจริง หรือ fact ตัวอย่างของข้อมูลแสดงได้ดังรูปที่ 4.2

```
%% Data weather
%
%attributes: names and their possible values
%
attribute( outlook,      [sunny, overcast, rainy] ).
attribute( temperature, [hot, mild, cool]         ).
attribute( humidity,     [high, normal]           ).
attribute( windy,        [true, false]            ).
attribute( class,        [yes, no]                ).
%data
instance(1, class=no, [outlook=sunny, temperature=hot, humidity=high, windy=false]).
instance(2, class=no, [outlook=sunny, temperature=hot, humidity=high, windy=true]).
instance(3, class=yes, [outlook=overcast, temperature=hot, humidity=high, windy=false]).
instance(4, class=yes, [outlook=rainy, temperature=mild, humidity=high, windy=false]).
instance(5, class=yes, [outlook=rainy, temperature=cool, humidity=normal, windy=false]).
instance(6, class=no, [outlook=rainy, temperature=cool, humidity=normal, windy=true]).
instance(7, class=yes, [outlook=overcast, temperature=cool, humidity=normal, windy=true]).
instance(8, class=no, [outlook=sunny, temperature=mild, humidity=high, windy=false]).
instance(9, class=yes, [outlook=sunny, temperature=cool, humidity=normal, windy=false]).
instance(10, class=yes, [outlook=rainy, temperature=mild, humidity=normal, windy=false]).
instance(11, class=yes, [outlook=sunny, temperature=mild, humidity=normal, windy=true]).
instance(12, class=yes, [outlook=overcast, temperature=mild, humidity=high, windy=true]).
instance(13, class=yes, [outlook=overcast, temperature=hot, humidity=normal, windy=false]).
instance(14, class=no, [outlook=rainy, temperature=mild, humidity=high, windy=true]).
%
```

รูปที่ 4.2 ตัวอย่างไฟล์ข้อมูลที่จะนำเข้ายังโปรแกรมเพื่อสร้างกฎการตัดสินใจ

บรรทัดแรกของข้อมูลเริ่มต้นด้วยเครื่องหมาย % หมายถึง comment โครงสร้างของไฟล์ข้อมูลจะแยกส่วนประกอบออกเป็นสองส่วนคือ ส่วนคำอธิบายแอททริบิวต์ (ได้แก่ส่วนข้อความ attribute ...) และส่วนแสดงรายละเอียดของข้อมูลแต่ละเรคคอร์ด (ได้แก่ส่วนข้อความ instance ...) ในส่วนที่อธิบายแอททริบิวต์ ภายในจะประกอบด้วยสองอาร์กิวเม้นต์ อาร์กิวเม้นต์แรกจะบอกชื่อแอททริบิวต์ อาร์กิวเม้นต์ที่สองเป็นลิสต์ของค่าที่เป็นไปได้ทั้งหมดของแอททริบิวต์นั้น ในส่วนของข้อมูลหรือ instance จะประกอบด้วยสามอาร์กิวเม้นต์ คือ หมายเลขของข้อมูล, ค่าคลาส

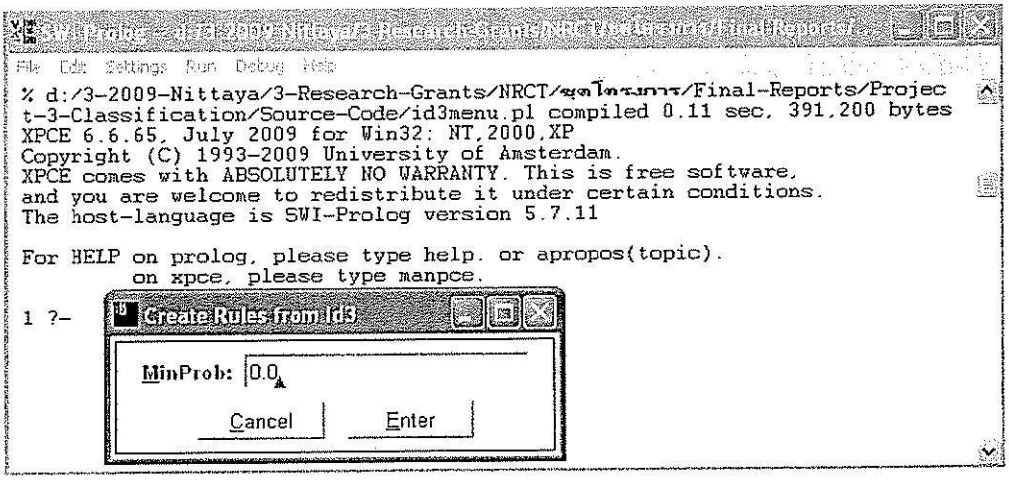
ของข้อมูล, ลิสต์ที่ระบุค่าของแต่ละแอททริบิวต์ โดยวิธีการระบุค่าจะใช้รูปแบบ attribute-value pair หรือ ชื่อแอททริบิวต์=ค่าของแอททริบิวต์ เมื่อสร้างข้อมูลในรูปแบบที่กำหนดเสร็จแล้วจะต้องบันทึกไฟล์ให้อยู่ในรูปแบบของโปรแกรม Prolog ที่มีส่วนขยาย (file extension) เป็น .pl เช่น ตัวอย่างไฟล์ข้อมูลในรูปที่ 4.2 บันทึกอยู่ในชื่อ weather.pl ข้อมูลนี้ (Quinlan 1993) เป็นข้อมูลที่รวบรวมการตัดสินใจของนักกอล์ฟในสิบสี่วันที่ผ่านมาว่าวันใดเขาตัดสินใจออกไปเล่นกอล์ฟ (class = yes) และวันใดที่ไม่ออกไปเล่น (class = no) แอททริบิวต์ที่ใช้ประกอบการตัดสินใจประกอบด้วยสภาพท้องฟ้า (outlook) อุณหภูมิ (temperature) ความชื้น (humidity) และสภาพลมแรง (windy)

โปรแกรม Tree-induction

ระบบเหมืองข้อมูลเพื่อการจำแนก เริ่มต้นทำงานจากโปรแกรมแรก (ชื่อ id3menu) ที่เป็นส่วนสร้าง GUI โต้ตอบกับผู้ใช้ (การพัฒนาส่วน GUI ใช้ซอฟต์แวร์ XPCE ที่ถูกผนวกอยู่ใน SWI Prolog) จากนั้นจะเริ่มการสร้างต้นไม้ตัดสินใจด้วยการเรียกคำสั่ง callId3 คำสั่ง callId3 จะทำหน้าที่เรียกคำสั่ง mainId3 โดยระบุอาร์กิวเม้นต์เป็นค่าความน่าจะเป็นขั้นต่ำ ถ้าผู้ใช้ไม่ระบุค่านี้ระบบจะกำหนดให้เป็น 0.0 ตัวอย่างการเรียกใช้ระบบด้วยคำสั่ง 'id3menu' แสดงได้ดังรูปที่ 4.3

```
id3menu:-
      new(Dialog,dialog('Create Rules from Id3')),
      send_list(Dialog, append,
      [ new(Per,text_item(minProb,'0.0')),
        button(cancel, message(Dialog, destroy)),
        button(enter, and(message(@prolog,callId3,Per?selection ),
        message(Dialog, destroy))) ]),
      send(Dialog, open).

callId3(Per):- term_to_atom(Per1,Per), mainId3(Per1).
```



รูปที่ 4.3 หน้าจอหลักของการเรียกใช้โปรแกรมสร้างต้นไม้ตัดสินใจ

คำสั่ง mainID3 ทำหน้าที่เรียกใช้ init ซึ่งเป็นส่วนสร้าง node และ edge รวมถึงตัวนับหมายเลขโหนด กระบวนการสร้างต้นไม้ตัดสินใจจะไปสิ้นสุดที่คำสั่ง addKnowledge ที่ทำหน้าที่บันทึกข้อมูล node และ edge ลงในฐานข้อมูลชั่วคราว เมื่อถึงขั้นตอนนี้ผู้ใช้สามารถตรวจสอบข้อมูลเกี่ยวกับ node และedge ได้โดยการสั่ง listing(node) และ listing(edge) ซึ่งจะได้รายละเอียดปรากฏดังรูปที่ 4.4

```
mainId3(Min) :-   init(AllAttr, EdgeList),
                  getnode(N),
                  create_edge_onelevel(N, AllAttr, EdgeList),
                  addKnowledge,
                  selectRule(Min, Res),
                  maplist(writeln, Res).
```

SWI-Prolog -- d:/3-2009-Nittaya/3-Research-Grants/NRCT/ชุดโครงการ/Final-Reports/...

File Edit Settings Run Debug Help

```
1 ?- listing(node).
:- dynamic node/2.

node(1, [3, 4, 5, 7, 9, 10, 11, 12, 13]-[1, 2, 6, 8, 14]).
node(2, [9, 11]-[1, 2, 8]).
node(3, []-[1, 2, 8]).
node(4, [9, 11]-[]).
node(5, [3, 7, 12, 13]-[]).
node(6, [4, 5, 10]-[6, 14]).
node(7, []-[6, 14]).
node(8, [4, 5, 10]-[]).

true.

2 ?- listing(edge).
:- dynamic edge/3.

edge(0, root=nil, 1).
edge(1, outlook=sunny, 2).
edge(2, humidity=high, 3).
edge(2, humidity=normal, 4).
edge(1, outlook=overcast, 5).
edge(1, outlook=rainy, 6).
edge(6, windy=true, 7).
edge(6, windy=false, 8).

true.
```

รูปที่ 4.4 รายละเอียดที่แสดงจากการใช้คำสั่ง listing(node) และ listing(edge)

ข้อมูล node หมายเลขหนึ่งระบุว่าที่โหนดเริ่มแรกนี้มีข้อมูลที่เป็นกลุ่ม positive คือข้อมูลหมายเลข [3,4,5,7,9,10,11,12,13] ปะปนอยู่กับข้อมูลที่เป็นกลุ่ม negative คือข้อมูลหมายเลข [1,2,6,8,14] เมื่อพิจารณาประกอบกับข้อมูล edge บรรทัดที่สอง, บรรทัดที่ห้า และบรรทัดที่หก ที่ระบุว่า

edge(1, outlook=sunny, 2)

edge(1, outlook=overcast, 5)

edge(1, outlook=rainy, 6)

ทำให้เราทราบว่าแอททริบิวต์ outlook ถูกเลือกใช้เป็นโหนดในระดับแรกสุด และมีกิ่งแตกออกไปสามกิ่ง คือ กรณีแอททริบิวต์นี้มีค่าเป็น sunny, overcast และ rainy โดยกรณี outlook=sunny จากโหนดแม่ที่เป็นหมายเลขหนึ่งจะมีเส้นเชื่อมไปยังโหนดลูกที่เป็นโหนดหมายเลขสอง ในทำนองเดียวกันกรณี outlook=overcast จากโหนดแม่ที่เป็นหมายเลขหนึ่งจะมีเส้นเชื่อมไปยังโหนดลูกที่เป็นโหนดหมายเลขห้า และสุดท้ายกรณี outlook=rainy จากโหนดแม่ที่เป็นหมายเลขหนึ่งจะมีเส้นเชื่อมไปยังโหนดลูกที่เป็นโหนดหมายเลขหก ถ้าพิจารณาที่ข้อมูลของโหนดหมายเลขห้า

node(5, [3, 7, 12, 13]-[ ])

ปรากฏข้อมูลในโหนดนี้สี่เรคคอร์ดและทุกรคคอร์ดอยู่ในกลุ่ม positive (นั่นคือ class = yes) ทำให้การสร้างต้นไม้ย่อยในกิ่งนี้สิ้นสุดโดยมีโหนดหมายเลขห้าเป็นโหนดใบ ในขณะที่โหนดหมายเลขสองและหมายเลขหกยังต้องมีการสร้างต้นไม้ย่อยต่อไป

จะเห็นได้ว่าโครงสร้างต้นไม้ตัดสินใจนี้สามารถใช้เป็นโมเดลอธิบายการตัดสินใจของนักกอล์ฟได้ว่าในสถานการณ์แบบใดที่เขาตัดสินใจออกไปเล่นกอล์ฟ และในสถานการณ์แบบใดที่เขาจะไม่ออกไปเล่น แต่ต้นไม้ตัดสินใจนี้จะแปลความได้ค่อนข้างยากสำหรับผู้ใช้ที่ไม่คุ้นเคยกับโครงสร้างข้อมูลชนิดนี้ การแปลความที่ง่ายกว่านี้คือการใช้กฎการตัดสินใจ ถ้า...แล้ว

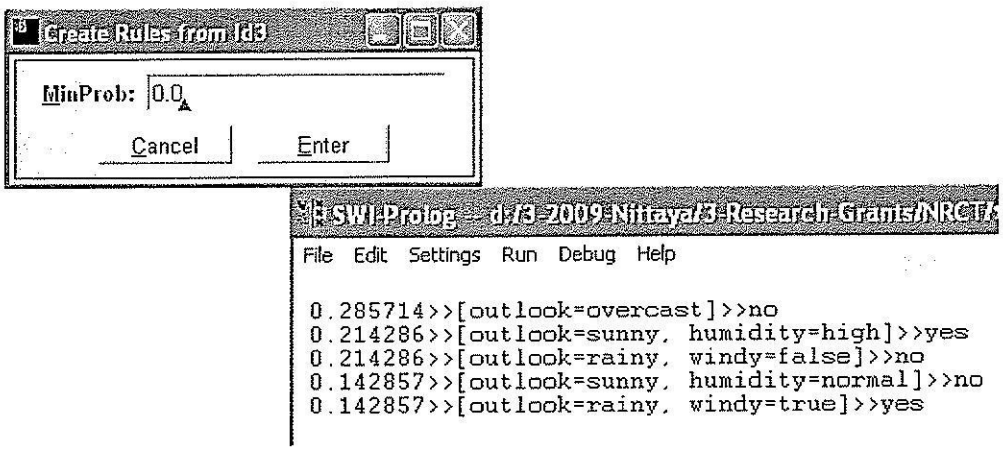
โปรแกรม Probabilistic-rule generation

การแปลงโครงสร้างต้นไม้ตัดสินใจที่ประกอบด้วย node และ edge ให้เป็นกฎการตัดสินใจ IF..THEN มีขั้นตอนการเขียนคำสั่งดังต่อไปนี้

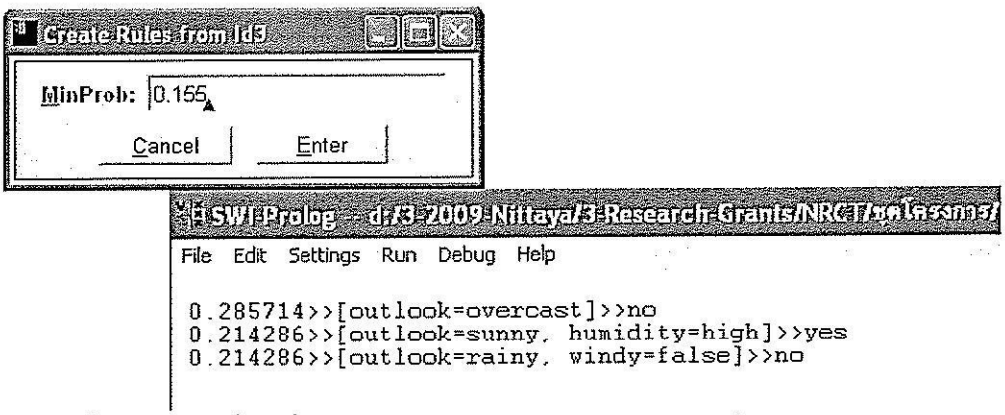
```
addKnowledge :-
        findall([A], pathFromRootToLeaf(A, _), Res),
        retractall(_>>_>>_),
        maplist(apply(assert), Res),
        writeln(addToKNB), nl.
selectRule(V,Res) :-
        findall(N>>X>>Class, (X>>Class>>N,N>=V), Res1),
        sort(Res1,Res2), reverse(Res2,Res).
path(A, [H| T], C) :- edge(A, H, B), path(B, T, C).
path(C, [], C) :- !.
```

```
pathFromRootToLeaf(V>>Class>>Num,C) :-
        path(1, V, C),
        node(C, Value1-Value2),
        (Value1=[]; Value2 =[]),
        (Value1=[]->length(Value2, Numb) ; length(Value1, Numb)),
        total+Total,
        Num is Numb/Total,
        (Value1=[]->Class=yes ; Class=no).
```

จากหน้าจอเริ่มต้นในรูปที่ 4.3 ถ้าผู้ใช้ไม่เปลี่ยนค่าความน่าจะเป็น (นั่นคือใช้ค่าที่โปรแกรมระบุไว้เป็น 0.0) เมื่อคลิกที่ปุ่ม Enter จะปรากฏผลลัพธ์เป็นกฎการตัดสินใจแสดงได้ดังรูปที่ 4.5 ส่วนในกรณีที่ผู้ใช้ระบุค่าความน่าจะเป็นเป็นค่าอื่นเช่น 0.155 เมื่อคลิกที่ปุ่ม Enter จะปรากฏผลลัพธ์เป็นกฎการตัดสินใจแสดงได้ดังรูปที่ 4.6



รูปที่ 4.5 โมเดลที่ได้ในลักษณะของกฎการตัดสินใจเมื่อเรียกใช้โปรแกรมด้วยค่าความน่าจะเป็น 0.0



รูปที่ 4.6 โมเดลที่ได้เมื่อเรียกใช้โปรแกรมด้วยค่าความน่าจะเป็น 0.155

การแสดงผลลัพธ์ในลักษณะของกฎการตัดสินใจ ข้อความแต่ละบรรทัดจะประกอบด้วยส่วนประกอบสามส่วนคือ ค่าความน่าจะเป็น, ปัจจัยที่ใช้ประกอบการตัดสินใจ และผลการตัดสินใจ ดังตัวอย่างกฎการตัดสินใจในรูปที่ 4.6

0.285714 >> [outlook=overcast] >> no

0.214286 >> [outlook=sunny, humidity=high] >> yes

0.214286 >> [outlook=rainy, windy=false] >> no

กฎแรกแปลความหมายได้ว่ากฎนี้มีความน่าจะเป็นในการครอบคลุมข้อมูล 0.285714 โดยมีปัจจัยประกอบการตัดสินใจคือ outlook=overcast และผลการตัดสินใจคือ no นั่นคือเมื่อท้องฟ้ามีเมฆมากนักกอล์ฟตัดสินใจที่จะไม่ออกไปเล่นกอล์ฟ

กฎในบรรทัดที่สองแปลความด้วยวิธีเดียวกันได้ว่า กฎนี้มีความน่าจะเป็นในการครอบคลุมข้อมูล 0.214286 โดยมีปัจจัยประกอบการตัดสินใจคือ outlook=sunny และ humidity=high ผลการตัดสินใจคือ yes (เครื่องหมาย , ในส่วนที่เป็นปัจจัยประกอบการตัดสินใจหมายถึง and ใช้ในกรณีที่มีปัจจัยมากกว่าหนึ่งปัจจัย)

กฎที่สามแปลความได้ว่า กฎนี้มีความน่าจะเป็นในการครอบคลุมข้อมูล 0.214286 โดยมีปัจจัยประกอบการตัดสินใจคือ outlook=rainy และ windy=false ผลการตัดสินใจคือ no

4.2 การทดสอบโปรแกรมการทำเหมืองข้อมูลแบบจำแนก

4.2.1 วิธีการทดสอบความถูกต้องและประสิทธิภาพของโปรแกรม

ระบบเหมืองข้อมูลเพื่อการจำแนก เป็นโปรแกรมที่พัฒนามาจากพื้นฐานของโปรแกรม ID3 (Quinlan 1993) ที่ใช้การสร้างต้นไม้ตัดสินใจเป็นเครื่องมือหลักในการแยกข้อมูลที่มีหลายคลาสปะปนกันให้เหลือเป็นกลุ่มข้อมูลย่อยที่เป็นคลาสเดียวกัน จากนั้นอ่านลักษณะของข้อมูลในแต่ละกลุ่มย่อยจากโครงสร้างโหนดและกิ่งของต้นไม้ตัดสินใจ วิธีการจำแนกข้อมูลเพื่อให้ได้โมเดลในลักษณะนี้ได้รับการยอมรับว่ามีความถูกต้องของโมเดลสูง และข้อเด่นของวิธีการนี้คือโมเดลเข้าใจได้ง่าย (เมื่อเปรียบเทียบกับวิธีการอื่น เช่น neural network) ดังนั้นงานวิจัยนี้จึงทดสอบความถูกต้องของโปรแกรมด้วยการทดสอบผลลัพธ์ที่ได้ (ได้แก่ โมเดลข้อมูล) กับผลลัพธ์ที่ได้จากโปรแกรม ID3 แต่เนื่องจากโมเดลที่เรียนรู้จากข้อมูลฝึกขนาดใหญ่จะมีความซับซ้อนมาก จึงไม่สะดวกที่จะทดสอบเปรียบเทียบตัวโครงสร้างของตัวโมเดลโดยตรง แต่จะใช้วิธีทดสอบความแม่นตรงของโมเดลเมื่อใช้ทำนายคลาสของข้อมูลชุดทดสอบ จากนั้นเปรียบเทียบความแม่นตรงในการทำนายคลาสโดยโมเดลที่สร้างจากโปรแกรมที่พัฒนาขึ้นเทียบกับโมเดลจากโปรแกรม ID3

ในส่วนของการทดสอบประสิทธิภาพเป็นการทดสอบการลดขนาดของโมเดล โดยใช้โมเดลทั้งหมดที่มีค่าความน่าจะเป็นตั้งแต่ 0.0 ถึง 1.0 ทดสอบความแม่นตรงในการทำนาย (predicting accuracy) เปรียบเทียบกับความแม่นตรงที่ได้เมื่อโมเดลมีขนาดเล็กลง การลดขนาดจะลดเป็นสัดส่วนจากค่าความน่าจะเป็นสูงสุด ใช้สัดส่วนตั้งแต่ 1/2(max.Prob), 1/3(max.Prob), 1/4(max.Prob), 1/5(max.Prob), 1/6(max.Prob), 1/7(max.Prob), 1/8(max.Prob), 1/9(max.Prob), จนถึงขนาด 1/10(max.Prob) ตัวอย่างเช่นจากข้อมูลนักกอล์ฟ สมมติให้โมเดลในรูปแบบของกฎการตัดสินใจที่มีค่าความน่าจะเป็นกำกับ (probabilistic decision rules) ทั้งหมดที่มีค่าความน่าจะเป็นภายในช่วง [0.0-1.0] ได้แก่

Rule 1: 0.285714>>[outlook=overcast]>>no

Rule 2: 0.214286>>[outlook=sunny, humidity=high]>>yes

Rule 3: 0.214286>>[outlook=rainy, windy=false]>>no

Rule 4: 0.142857>>[outlook=sunny, humidity=normal]>>no

Rule 5: 0.142857>>[outlook=rainy, windy=true]>>yes

Rule 6: 0.011326>>[outlook=rainy, humidity=high]>>no

ค่าความน่าจะเป็นสูงสุดของกฎการตัดสินใจชุดนี้คือ 0.285714 และเมื่อต้องการลดขนาดโมเดลลงด้วยเกณฑ์ 1/10(max.Prob) จะได้ 1/10 * (0.285714) = 0.02857 ดังนั้นที่ขนาดของการลดสัดส่วนลงหนึ่งในสิบของค่าความน่าจะเป็นจะได้โมเดลเป็น Rules 1-5 โดย Rule 6 จะถูกคัดทิ้งเนื่องจากค่าความน่าจะเป็นไม่ถึงเกณฑ์

ข้อมูลที่ใช้ในการทดสอบ

การทดสอบความถูกต้องของการสร้างโมเดล ความแม่นตรงของโมเดลในการทำนายคลาสของชุดข้อมูลทดสอบ รวมถึงการทดสอบประสิทธิภาพของโมเดลที่มีการลดขนาดลง จะใช้ชุดข้อมูลมาตรฐานจาก UCI Repository (www.ics.uci.edu/~mlearn/MLRepository.html) จำนวน 6 ชุดข้อมูล ประกอบด้วย ข้อมูล Monk, Post-operative, Breast cancer, Vote, Hepatitis, Mushroom ข้อมูลแต่ละชุดประกอบด้วย training data และ test data ดังนั้นวิธีการทดสอบความแม่นตรงจึงใช้วิธี hold out หรือวิธีที่แยกข้อมูลทดสอบออกจากข้อมูลฝึก เพื่อให้การทดสอบโมเดลเป็นการทดสอบกับข้อมูลใหม่โดยแท้จริง รายละเอียดของข้อมูลที่ใช้ทั้งหกชุดข้อมูลสรุปได้ดังตารางที่ 4.1

ตารางที่ 4.1 รายละเอียดของข้อมูลที่ใช้ทดสอบความถูกต้องและประสิทธิภาพของโปรแกรม

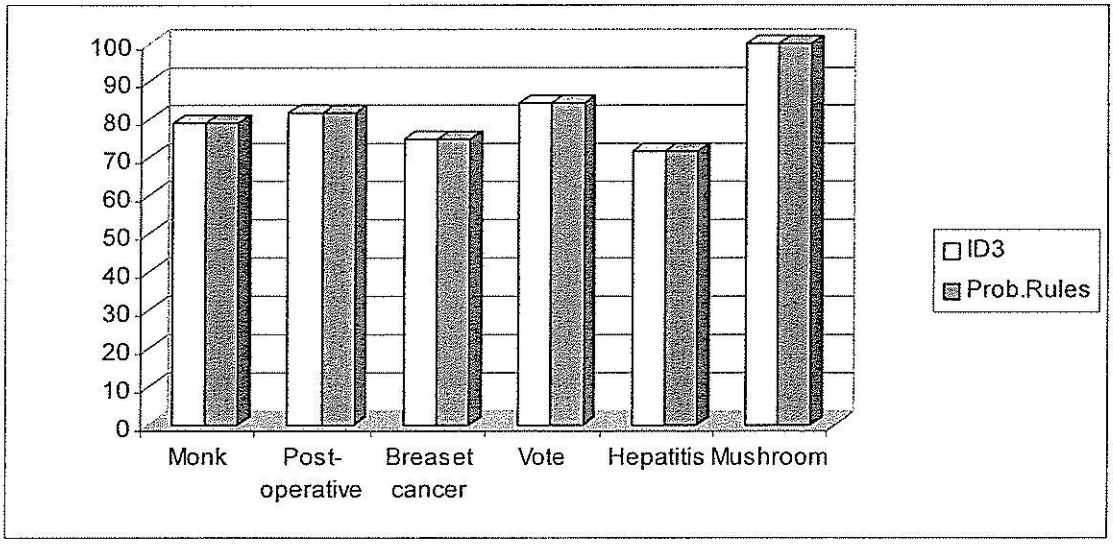
ชื่อชุดข้อมูล	จำนวนข้อมูลฝึก (instances)	จำนวนข้อมูลทดสอบ (instances)	จำนวน predicting attributes	จำนวนคลาสใน goal attribute
Monk	124	432	6	2
Post-operative	50	36	8	2
Breast cancer	191	95	9	2
Vote	300	135	16	2
Hepatitis	103	52	19	2
Mushroom	5416	2708	22	2

4.2.2 ผลการทดสอบ

การทดลองในขั้นตอนแรกเป็นการทดสอบความถูกต้องของโปรแกรมสร้างกฎการตัดสินใจ โดยเปรียบเทียบกับโปรแกรมID3 วิธีการเปรียบเทียบใช้ค่าความแม่นตรง (accuracy) ในการทำนายคลาสข้อมูลทดสอบของโมเดลที่ได้จากทั้งสองโปรแกรม ผลการทดสอบแสดงดังตารางที่ 4.2 และผลการทดสอบเดียวกันนี้แสดงเป็นภาพได้ดังรูปที่ 4.7

ตารางที่ 4.2 ผลการทดสอบความแม่นตรงของโมเดล Probabilistic decision rules เปรียบเทียบกับโมเดล ID3

ชื่อชุดข้อมูล	ความแม่นตรง (accuracy)	
	Probabilistic decision rules	ID3
Monk	79.03%	79.03%
Post-operative	81.65%	81.65%
Breast cancer	74.73%	74.73%
Vote	84.33%	84.33%
Hepatitis	71.42%	71.42%
Mushroom	100%	100%



รูปที่ 4.7 กราฟเปรียบเทียบความแม่นตรง (accuracy) ของ Probabilistic decision rules และ ID3

การทดสอบในขั้นตอนที่สอง เป็นการทดสอบประสิทธิภาพของโมเดลที่แสดงในลักษณะของกฎการตัดสินใจที่ค่าความน่าจะเป็นกำกับ หรือ Probabilistic decision rules ที่มีการลดขนาดของโมเดลลงมาตามสัดส่วนต่างๆกัน วิธีการวัดประสิทธิภาพจะพิจารณาจากขนาดของโมเดล (วัดจากจำนวนของกฎ) และความแม่นตรงของโมเดลในการทำนายคลาสของข้อมูล ผลการทดสอบการสร้างและทดสอบโมเดลกับชุดข้อมูลมาตรฐานทั้งหกชุดสรุปผลได้ดังตารางที่ 4.3 และแสดงผลการทดสอบกับชุดข้อมูลมาตรฐานแต่ละข้อมูลเป็นภาพกราฟได้ดังรูปที่ 4.8-4.13