การทำเหมืองข้อมูลเป็นกระบวนการที่อาจไม่เสร็จสิ้นในคราวเดียว ถ้าหากผลลัพธ์ที่ได้เป็นโมเดลที่ยังมีความถูกต้องต่ำเกินไป กระบวนการอาจจะต้องวนซ้ำ โดยย้อนกลับมาเริ่มต้นรวบรวมข้อมูล และคัดเลือกข้อมูลใหม่อีกครั้ง จากความสำคัญดังกล่าวของข้อมูลและกระบวนการเตรียมข้อมูล งานวิจัยนี้จึงมีวัตถุประสงค์ที่จะพัฒนาเทคนิคและโปรแกรมที่จะเป็นเครื่องมือช่วยงานให้กับนักวิเคราะห์ข้อมูลที่ต้องการใช้เทคโนโลยีการทำเหมืองข้อมูล

เครื่องมือช่วยเตรียมข้อมูลที่พัฒนาขึ้นนี้ประกอบด้วย

- *โปรแกรมแปลงรูปแบบข้อมูล* ทำหน้าที่รวมข้อมูลและเปลี่ยนรูปแบบข้อมูล ในงานวิจัยนี้ใช้ข้อมูลจาก UCI repository ที่เป็นแหล่งข้อมูลมาตรฐานสำหรับวิเคราะห์และทดสอบในงาน machine learning และ data mining ปัจจุบัน (ปี 2009) แหล่งข้อมูลนี้รวบรวมข้อมูลไว้ราว 187 datasets ส่วนใหญ่เป็นข้อมูลที่ใช้ในงานทำเหมืองข้อมูลประเภทการจำแนก (classification) ข้อมูลแต่ละ dataset จะประกอบด้วยสองไฟล์คือ data file (ไฟล์ที่มีส่วนขยายเป็น .data) เป็นรายละเอียดข้อมูล และ names file (ไฟล์ที่มีส่วนขยายเป็น .names) เป็นคำอธิบายแอททริบิวต์และคลาสของข้อมูล ข้อมูลในทั้งสองไฟล์จะปรากฏในรูปแบบข้อความ โปรแกรมแปลงรูปแบบข้อมูลที่พัฒนาขึ้น จะรวมทั้งสองไฟล์และแปลงข้อความให้อยู่ในรูปแบบ Horn clauses ที่โปรแกรมโปรล็อกสามารถประมวลผลได้
- *โปรแกรมปรับปรุงข้อมูล* ทำหน้าที่ตรวจสอบความครบถ้วนของค่าที่ปรากฏในแต่ละรายการข้อมูล ถ้ามีค่าใดสูญหายจะแสดงเมนูให้ผู้ใช้เลือกว่าจะเติมแทนค่าที่สูญหายนั้นด้วยค่าใด ผู้ใช้สามารถเลือกค่าคงที่เช่น '?' หรือ 'missing' เติมลงในตำแหน่งที่มีค่าสูญหาย หรือเลือกให้เติมค่าที่เป็นค่าส่วนใหญ่ของข้อมูลในแอททริบิวต์นั้น ในกรณีที่มีค่าสูญหายเป็นปริมาณมากผู้ใช้อาจเลือกที่จะตัดทิ้งแอททริบิวต์นั้นได้
- *โปรแกรมคัดเลือกข้อมูล* การคัดเลือกข้อมูลในงานวิจัยนี้เป็นการคัดเลือกแอททริบิวต์ หรือ feature selection โปรแกรมจะมีฟังก์ชันแสดงการกระจายของข้อมูลในแต่ละแอททริบิวต์ในลักษณะฮิสโตแกรม เพื่อให้ผู้ใช้พิจารณาว่าแอททริบิวต์ใดสมควรถูกคัดเลือกไว้ และแอททริบิวต์ใดสมควรถูกตัดทิ้ง ผลลัพธ์ที่ได้จากโปรแกรมนี้คือไฟล์ข้อมูลที่ผ่านการคัดเลือกแอททริบิวต์แล้ว ไฟล์นี้จะถูกบันทึกเพื่อนำไปใช้งานต่อในขั้นการลดขนาดข้อมูล หรืออาจนำไปใช้ในการทำเหมืองข้อมูลได้ทันที

- *โปรแกรมลดขนาดข้อมูล* ในกรณีไฟล์ข้อมูลมีขนาดใหญ่มากโปรแกรมนี้จะช่วยลดปริมาณข้อมูลด้วยเทคนิคการสุ่ม วิธีการสุ่มจะมีให้ผู้ใช้เลือก 3 วิธีคือ การสุ่มแบบปกติและไม่ใส่ค่าคืนกลับ (random sampling without replacement), การสุ่มแบบปกติและใส่ค่าคืนกลับ (random sampling with replacement), และการสุ่มตามความหนาแน่น (density-biased sampling) นอกจากเลือกวิธีการสุ่มแล้วผู้ใช้จะเลือกขนาดของข้อมูลสุ่มว่าต้องการสุ่มเลือกข้อมูลเป็นปริมาณกี่เปอร์เซนต์ ในกรณีของการสุ่มตามความหนาแน่นผู้ใช้จะสามารถเลือกขนาดของความหนาแน่นที่ต้องการ ข้อมูลที่มีความหนาแน่นไม่ถึงเกณฑ์จะไม่ถูกสุ่มเลือก ข้อมูลที่ได้จากการสุ่มจะบันทึกไว้ในไฟล์ให้โปรแกรมทำเหมืองข้อมูลสามารถประมวลผลต่อได้

ข้อเสนอแนะ

งานวิจัยนี้จำกัดขอบเขตของการพัฒนาโปรแกรมเพื่อการเตรียมข้อมูลก่อนการทำเหมืองข้อมูล ให้รับข้อมูลในรูปแบบ UCI repository จากนั้นแปลงรูปแบบให้เป็น Horn clauses ส่งต่อให้กับส่วนที่ทำหน้าที่ปรับปรุงข้อมูล คัดเลือกข้อมูล และลดขนาดของข้อมูล การทดสอบโปรแกรมให้ผลที่ตรงตามวัตถุประสงค์ แต่โปรแกรมนี้ยังสามารถพัฒนาให้มีความสามารถสูงขึ้นในด้านต่างๆ ดังนี้

(1) การกำหนดรูปแบบข้อมูล สามารถขยายขอบเขตให้รวบรวมข้อมูลจากฐานข้อมูลที่อาจจะอยู่ในหลายฐานข้อมูลให้เป็นไฟล์เดียว หรืออาจจะพัฒนาต่อไปให้สามารถรวบรวมข้อมูลจากคลังข้อมูลทั้งจาก fact table และ dimension table

(2) การปรับปรุงข้อมูลใช้การพิจารณาค่าที่สูญหาย หรือ missing values โดยยังไม่พิจารณากรณีข้อมูลผิดพลาด (random error, noise) ดังนั้นถ้าต้องการเพิ่มความสามารถของโปรแกรม อาจพิจารณาเพิ่มฟังก์ชันการตรวจสอบข้อมูลผิดพลาดและพัฒนาแนวทางการจัดการกับข้อมูลผิดพลาด นอกจากนี้ในส่วนของการจัดการกับข้อมูลสูญหายอาจเพิ่มเทคนิคการจัดการแบบอื่นๆ เช่นใช้การเติมค่าโดยพิจารณาจากค่าใกล้เคียง หรือทำนายค่าจากแอททริบิวต์อื่น

(3) การคัดเลือกแอททริบิวต์ข้อมูลในงานวิจัยนี้ ใช้วิธีแสดงการกระจายค่าของข้อมูลในลักษณะของฮิสโตแกรม จากนั้นให้ผู้ใช้กำหนดว่าจะคัดเลือกแอททริบิวต์ใดบ้าง ในส่วนนี้สามารถพัฒนาให้ดีขึ้นได้โดยการเพิ่มเทคนิคการ

พิจารณาความเหมาะสมของแอททริบิวต์ (เช่น คำนวณ gain value) จากนั้นแสดงคำแนะนำให้ผู้ใช้ตัดสินใจว่าควรจะเลือกแอททริบิวต์ใด

(4) การลดขนาดข้อมูลในงานวิจัยนี้ใช้วิธีการสุ่มข้อมูล โดยสร้างวิธีการสุ่มไว้ให้ผู้ใช้เลือกสามวิธี ฟังก์ชันในการสุ่มหรือการคัดเลือกรายการข้อมูลอาจเพิ่มเติมมากขึ้นกว่านี้เช่น พิจารณาวิธีการสุ่มแบบ stratified sampling หรืออาจใช้การทำ clustering ก่อนที่จะสุ่มข้อมูลจากแต่ละคลัสเตอร์

บทที่ 3

การพัฒนาการทำเหมืองข้อมูลแบบจัดกลุ่ม (โครงการวิจัยที่ 2)

3.1 วิธีดำเนินการวิจัยโครงการวิจัยที่ 2

3.1.1 กรอบแนวคิดของโครงการวิจัยที่ 2

ขั้นตอนการทำงานของโปรแกรมจัดกลุ่มข้อมูลอัตโนมัติ โดยทั่วไปจะประกอบด้วย 5 ขั้นตอนหลัก คือ

(1) กำหนดจำนวนกลุ่มและกำหนดลักษณะที่จะใช้จัดข้อมูลเข้ากลุ่ม งานที่ต้องทำประกอบด้วย pattern representation, feature selection, feature extraction

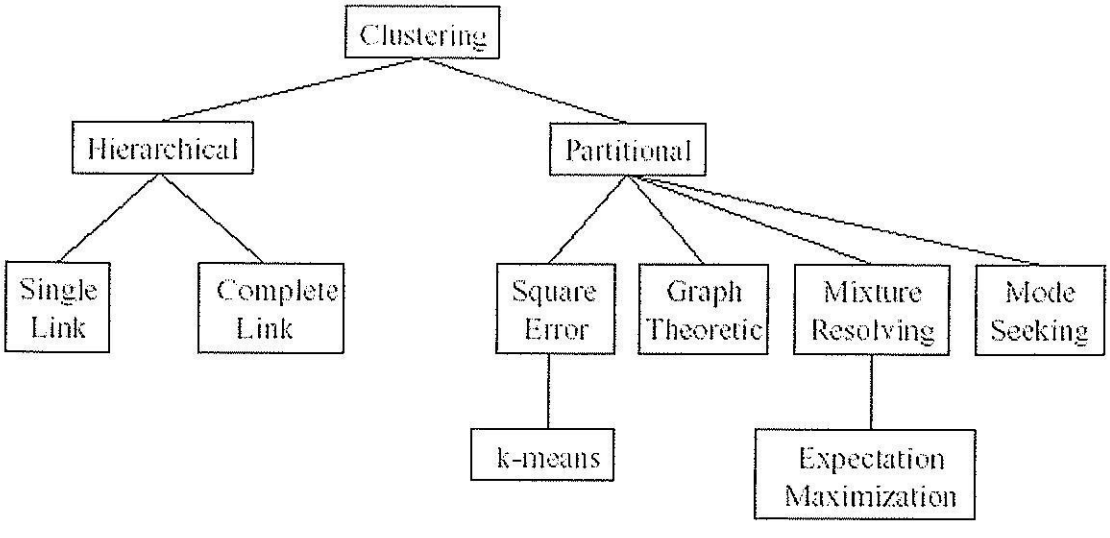
(2) กำหนดฟังก์ชันที่จะใช้วัดความคล้ายคลึงของข้อมูล งานที่ต้องทำประกอบด้วย pattern similarity measure

(3) จัดกลุ่มข้อมูล งานที่ต้องทำประกอบด้วย grouping

(4) กำหนดรูปแบบการแสดงกลุ่ม งานที่ต้องทำประกอบด้วย data abstraction

(5) วิเคราะห์ผลการจัดกลุ่ม งานที่ต้องทำประกอบด้วย cluster validation

การจัดกลุ่มข้อมูลอัตโนมัติ จะได้ผลลัพธ์ที่เป็นประโยชน์และมีประสิทธิภาพมากน้อยเพียงใดจะขึ้นอยู่กับเทคนิคที่ใช้ในขั้นตอนการจัดกลุ่มข้อมูล (grouping) ซึ่งโดยทั่วไปเทคนิค หรืออัลกอริทึมที่ใช้จะจัดอยู่ในสองกลุ่มใหญ่ คือ partitioning methods และ hierarchical methods ซึ่งในแต่ละกลุ่มยังสามารถแยกย่อยเป็นอีกหลายเทคนิค ดังแสดงด้วยแผนภาพในรูปที่ 3.1



รูปที่ 3.1 แสดงการจัดหมวดหมู่เทคนิคที่ใช้ในการทำ clustering

Partitioning methods

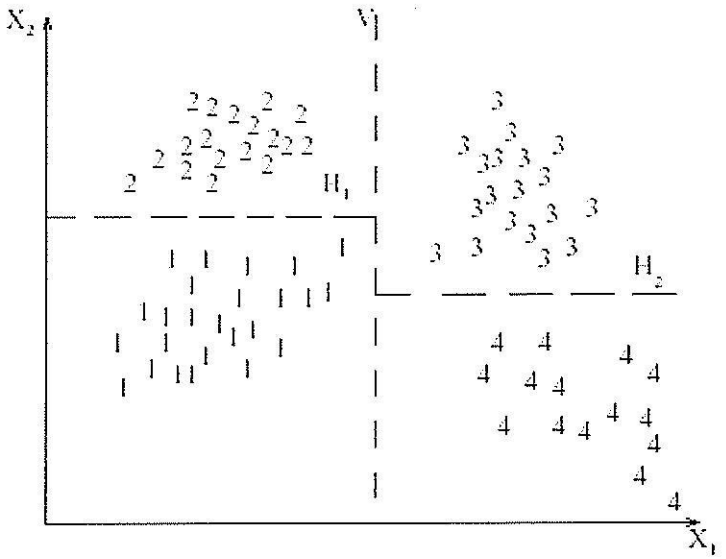
ข้อมูล n ตัวจะถูกแบ่งออกเป็น k กลุ่ม โดยที่ $k \leq n$ และมีข้อกำหนดว่าแต่ละกลุ่มจะต้องมีข้อมูลอยู่อย่างน้อยหนึ่งตัว และข้อมูลหนึ่งตัวจะต้องอยู่ในกลุ่มเดียวเท่านั้น (ข้อกำหนดประการหลังนี้จะไม่มีในกรณีของ fuzzy clustering ที่ข้อมูลสามารถถูกพิจารณาให้อยู่ในหลายกลุ่มได้) ตัวอย่างอัลกอริทึมในกลุ่มนี้ได้แก่ k-means, k-medoids, k-modes, k-prototypes, EM (Expectation Maximization), CLARANS ตัวอย่างการทำงานของ partitioning methods แสดงได้ดังรูปที่ 3.2 จากรูปแกน x_1 และ x_2 คือ feature หรือคุณลักษณะของข้อมูลที่ใช้ในการพิจารณาจัดกลุ่ม การจัดกลุ่มครั้งแรกพิจารณาที่ feature x_1 ทำให้ได้เส้น V ที่ใช้แบ่งกลุ่มข้อมูลส่วนแรก จากนั้นใช้ feature x_2 พิจารณาแบ่งกลุ่มต่อไปทำให้ได้เส้นแบ่งกลุ่มในแนวนอน คือ H_1 และ H_2 จากตัวอย่างนี้จะจัดข้อมูลได้ 4 กลุ่ม และลักษณะข้อมูลในแต่ละกลุ่มคือ

Cluster 1 : $[x_1 < V]$ and $[x_2 < H_1]$

Cluster 2 : $[x_1 < V]$ and $[x_2 > H_1]$

Cluster 3 : $[x_1 > V]$ and $[x_2 > H_2]$

Cluster 4 : $[x_1 > V]$ and $[x_2 < H_2]$

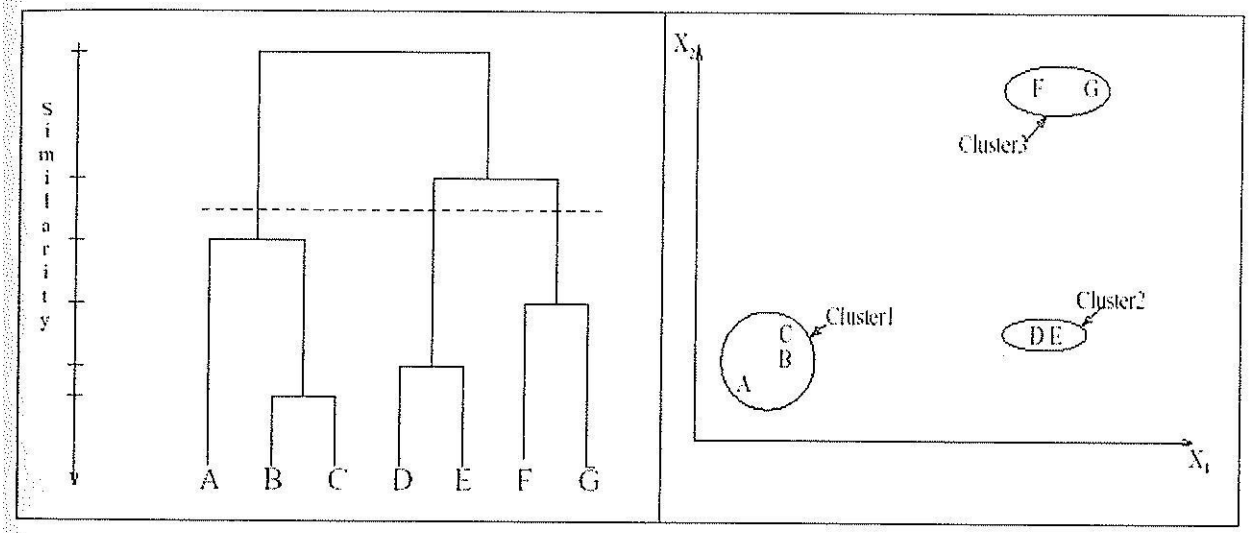


รูปที่ 3.2 การจัดกลุ่มข้อมูลด้วยเทคนิค partitioning

อัลกอริทึมในกลุ่ม partitioning จะใช้ได้ดีกับกรณีข้อมูลจำนวนน้อยและมี feature ไม่มาก ถ้าข้อมูลมี feature มากจะทำให้ได้จำนวนกลุ่มที่มากเกินไป นอกจากนี้เกณฑ์ในการจัดกลุ่ม เช่น squared error จะทำงานได้ดีกับข้อมูลที่รวมกลุ่มหนาแน่น

Hierarchical methods

เทคนิคในกลุ่มนี้แบ่งย่อยออกเป็น agglomerative และ divisive เทคนิค agglomerative จะเริ่มทำงานด้วยการจัดข้อมูลหนึ่งตัวเป็นหนึ่งกลุ่ม จากนั้นพิจารณารวมกลุ่มข้อมูลที่คล้ายคลึงกันเข้าด้วยกัน ทำในลักษณะนี้ไปจนกระทั่งไม่สามารถรวมกลุ่มข้อมูลต่อไปได้ เทคนิค divisive มีหลักการทำงานแบบเดียวกันแต่ทำในทิศทางตรงกันข้าม โดยเริ่มจากข้อมูลทั้งหมดจัดเป็นกลุ่มเดียวแล้วกระจายกลุ่มที่แตกต่างกันออกตามลำดับจนกระทั่งไม่สามารถกระจายต่อไปได้ ภาพในรูปที่ 3.3 แสดงการจัดข้อมูล 7 ตัว (ได้แก่ข้อมูล A, B, C, D, E, F และ G) เข้าเป็น 3 กลุ่มด้วยเทคนิค hierarchical



รูปที่ 3.3 การจัดกลุ่มข้อมูลด้วยเทคนิค hierarchical

อัลกอริทึมในกลุ่ม hierarchical ประกอบด้วย CURE, Chamelon, BIRCH ข้อดีของอัลกอริทึมในกลุ่มนี้คือ ทำงานกับข้อมูลที่จัดกลุ่มได้หลากหลายลักษณะกว่าวิธีการ partitioning แต่ข้อเสียคือใช้เวลาในการทำงานนานกว่าและใช้เนื้อที่หน่วยความจำมากกว่า

อัลกอริทึมในการจัดกลุ่มข้อมูลทั้งในแบบ partitioning และ hierarchical ทำงานได้ดีกับข้อมูลขนาดเล็ก แต่เมื่อข้อมูลมีจำนวนมากขึ้นอัลกอริทึมจะเริ่มด้อยประสิทธิภาพ ดังแสดงด้วยผลการวิเคราะห์เปรียบเทียบเวลาและหน่วยความจำที่ใช้สำหรับแต่ละอัลกอริทึม ในตารางที่ 3.1 โดย n คือ จำนวนข้อมูล k คือ จำนวนกลุ่มของข้อมูล และ l คือ จำนวนรอบที่อัลกอริทึมใช้ในการทำงาน

ตารางที่ 3.1 เปรียบเทียบเวลาและหน่วยความจำที่ใช้ในแต่ละอัลกอริทึม

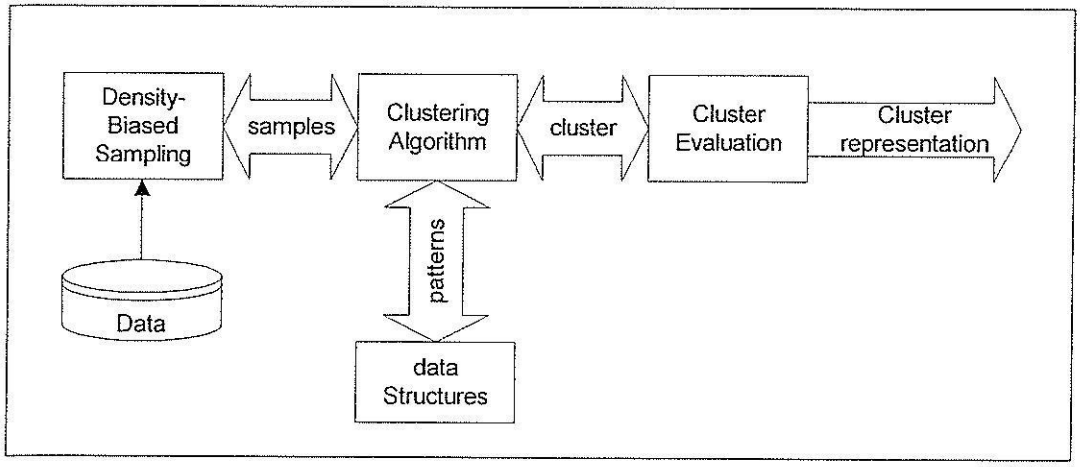
Clustering Algorithm	Time Complexity	Space Complexity
leader	$O(kn)$	$O(k)$
k-means	$O(nkl)$	$O(k)$
ISODATA	$O(nkl)$	$O(k)$
shortest spanning path	$O(n^2)$	$O(n)$
single-line	$O(n^2 \log n)$	$O(n^2)$
complete-line	$O(n^2 \log n)$	$O(n^2)$

ในกรณีของการจัดกลุ่มในงานทำเหมืองข้อมูลซึ่งมีปริมาณข้อมูลจำนวนมาก ได้มีการพัฒนาอัลกอริทึมที่สามารถรองรับข้อมูลขนาดใหญ่ได้ ในกลุ่มที่ใช้เทคนิค partitioning ได้มีการพัฒนาอัลกอริทึม CLARANS (Clustering Large Applications based on RANdom Sampling) โดยทีมนักวิจัย R.Ng และ J.Han ในปี 1994 ในกลุ่มเทคนิค hierarchical ได้มีการพัฒนาอัลกอริทึม BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies) โดยทีมนักวิจัย T.Zhang,R. Ramakrishnan และ M.Livny ในปี 1996

ถึงแม้อัลกอริทึม CLARANS และ BIRCH จะสามารถทำงานกับข้อมูลขนาดใหญ่ในเวลาที่เร็วขึ้นได้ (O(n)) แต่ข้อจำกัดที่สำคัญของอัลกอริทึมคือ ข้อมูลเริ่มต้นทั้งหมดจะต้องสามารถบรรจุอยู่ในหน่วยความจำหลักได้ ข้อจำกัดนี้เป็นอุปสรรคที่สำคัญในการนำอัลกอริทึมดังกล่าวมาใช้ในงานทำเหมืองข้อมูล เนื่องจากข้อมูลที่เกิดขึ้นจริงมักจะมีปริมาณมากเกินกว่าจะบรรจุข้อมูลทั้งหมดไว้ในหน่วยความจำหลักได้ แนวทางจัดการกับปัญหานี้เป็นได้ 2 แนวทาง คือ (1) ใช้วิธีการประมวลผลแบบขนาน (parallel clustering) หรือ (2) ใช้วิธีนำข้อมูลเพิ่มเข้ามาทีละส่วนเพื่อจัดกลุ่ม หรือเรียกว่าการจัดกลุ่มแบบเพิ่มพูน (incremental clustering)

โครงการวิจัยนี้จะมุ่งเน้นไปที่แนวทาง incremental clustering เนื่องจากนำไปใช้ประโยชน์ได้กว้างขวาง และมีข้อกำหนดน้อยกว่าแนวทางการประมวลผลแบบขนาน โดยจะพิจารณานำเทคนิคการสุ่มมาใช้ช่วยเพื่อใช้ข้อมูลที่เป็นตัวแทน แทนที่จะใช้ข้อมูลทุกตัวซึ่งคาดว่าจะช่วยเพิ่มความเร็วในการทำงานของอัลกอริทึม โดยที่การสุ่มจะพิจารณาให้น้ำหนักตามความหนาแน่นของข้อมูล (density-biased sampling) แทนที่จะเป็นแบบ random sampling ทั้งนี้เพื่อป้องกันข้อมูลกลุ่มเล็กถูกลดความสำคัญหรือถูกกำจัดไปในระหว่างการจัดกลุ่ม เนื่องจากความเข้าใจผิดว่าเป็นข้อมูลรบกวน (noise)

แนวทางการวิจัยที่เสนอขึ้นนี้ จะประกอบด้วยการออกแบบและพัฒนาโครงสร้างของส่วนประกอบหลักสองส่วน (ดังแสดงในรูปที่ 3.4) ที่จะต้องใช้ในการทำ incremental clustering ได้แก่ density-biased sampling technique และ incremental clustering algorithm



รูปที่ 3.4 กรอบแนวคิดของงานออกแบบและพัฒนาการจัดกลุ่มข้อมูลตามความหนาแน่น

งานวิจัยตลอดทั้งโครงการประกอบด้วยขั้นตอนหลัก ดังต่อไปนี้

ขั้นตอนที่ 1 ออกแบบและพัฒนาเทคนิคการสุ่มตามความหนาแน่น

ขั้นตอนการทำงานในส่วนนี้จะประกอบด้วย การกำหนดเกณฑ์การวิเคราะห์ความหนาแน่นของข้อมูล การกำหนดฟังก์ชันเพื่อรักษาคุณสมบัติความหนาแน่นของข้อมูล และการกำหนดน้ำหนักที่ควรจะให้กับข้อมูลแต่ละตัว จากนั้นจะเป็นการออกแบบอัลกอริทึม density-biased sampling

ขั้นตอนที่ 2 วิเคราะห์และออกแบบโครงสร้างข้อมูลที่ใช้ช่วยในการทำ clustering

โครงสร้างข้อมูลที่นิยมใช้ช่วยในกระบวนการทำ clustering มีได้หลากหลาย เช่น ใช้โครงสร้างแฮช ใช้โครงสร้างต้นไม้สมดุล งานในขั้นตอนนี้จึงเป็นการศึกษาแนวทางที่มีผู้เสนอไว้เพื่อวิเคราะห์จุดเด่น-จุดด้อย ซึ่งจะเป็นประโยชน์ในการออกแบบโครงสร้างข้อมูลของอัลกอริทึม incremental clustering โครงสร้างข้อมูลที่มีประสิทธิภาพจะมีผลอย่างมากต่อความเร็วและความสามารถของอัลกอริทึม

ขั้นตอนที่ 3 ออกแบบอัลกอริทึมในการทำ incremental clustering

ในขั้นตอนนี้จะเป็นการกำหนดมาตรวัดที่จะใช้จัดข้อมูลเป็นกลุ่ม หรือ คลัสเตอร์ รวมทั้งกำหนดเกณฑ์ที่จะใช้พิจารณาปรับปรุงคลัสเตอร์ที่ได้ให้เหมาะสมยิ่งขึ้น จากนั้นกำหนดค่าที่จะใช้ในการปรับจุดกึ่งกลางของคลัสเตอร์ เทคนิคของการสุ่มตามความหนาแน่นจะถูกนำมาใช้ร่วมกับการออกแบบอัลกอริทึม incremental clustering โดยพยายามลดจำนวนครั้งของการ scan ข้อมูลในฐานข้อมูลเพื่อเพิ่มความเร็วของอัลกอริทึม

3.1.2 การออกแบบอัลกอริทึมเพื่อจัดกลุ่มข้อมูลตามความหนาแน่น

การทำงานของอัลกอริทึมจัดกลุ่มข้อมูลตามความหนาแน่น จะประกอบด้วยการทำงานสองขั้นตอนใหญ่คือ การวัดความหนาแน่นของข้อมูลเพื่อคัดเลือกข้อมูลเป็นตัวแทนในการจัดกลุ่ม และการจัดข้อมูลตัวแทนเข้ากลุ่มโดยใช้ค่า similarity ของแอททริบิวต์เป็นเกณฑ์ในการจัดเข้ากลุ่ม

Algorithm 3.1 Density-biased clustering

Input: a data file,
minimum number of matched attributes M,
minimum density D

Output: a set of clusters with cluster means information

Phase 1 Selecting samples from dense data

(1) Show the GUI of density-biased clustering component

(2) Get the user's response to obtain the data file name,
minimum number of matched attributes M, density threshold D, and
number of cluster K

/* Compute similar instances and their density values */

(3) Open data file and read data instance

(4) For each data instance do

(4.1) Scan data file to collect instances, Ins, with matched attributes ≥ M

(4.2) Compute density, Den, as proportion of Ins to total instances in data file

(4.3) If Den ≥ D, then record this instance in temporary file F

Phase 2 Grouping data from the dense area

(5) Taking the first K data instances in F as temporary cluster means

(6) Repeat

(6.1) Assign each data instance in F into closest cluster, based on similarity

(6.2) Compute new cluster means

(6.3) Until each data instance does not change its cluster

(7) Return the cluster means and cluster members

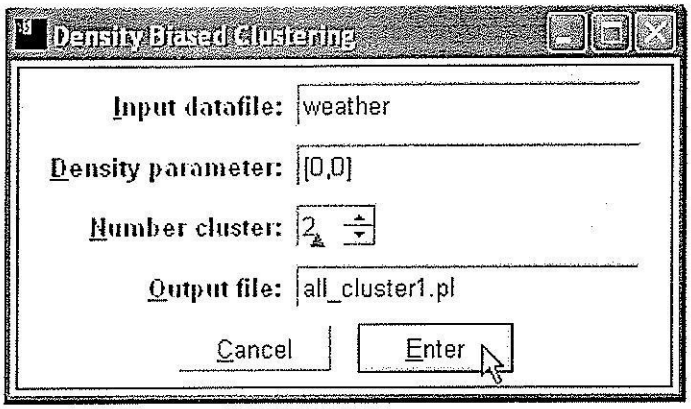
ในช่วงของการคัดเลือกข้อมูลตามความหนาแน่น จะให้ผู้ใช้กำหนดเกณฑ์ขั้นต่ำว่าจะพิจารณาความหนาแน่นด้วยค่าในกี่แอททริบิวต์ (M, Minimum number of attributes) และจะต้องการสุ่มข้อมูลที่มีความหนาแน่นขั้นต่ำเท่าไร (D, Density) โดยที่ $D \in [0..1]$ จากนั้นเริ่มต้นทำงานด้วยการเปิดไฟล์และอ่านข้อมูลทีละรายการเพื่อตรวจสอบข้อมูลที่อยู่ใกล้เคียง การวัดความใกล้เคียงใช้การเปรียบเทียบค่าในแต่ละแอททริบิวต์ ถ้ามีค่าคล้ายกันอย่างน้อย M แอททริบิวต์ถือว่า

เป็นข้อมูลที่เกาะกลุ่มอยู่ใกล้กัน ตรวจสอบข้อมูลใกล้เคียงเช่นนี้กับข้อมูลทุกรายการ จากนั้นนับจำนวนว่ามีข้อมูลกี่รายการที่จัดว่าอยู่ใกล้เคียง คำนวณค่าข้อมูลใกล้เคียงให้เป็นค่าสัดส่วนโดยหารด้วยจำนวนข้อมูลทั้งหมดในไฟล์ ทำการตรวจสอบเช่นนี้กับข้อมูลทุกรายการ จากนั้นคัดเลือกไว้เฉพาะข้อมูลที่มีค่า D ถึงเกณฑ์ที่ระบุ แล้วนำข้อมูลที่คัดเลือกไว้จัดกลุ่มเป็น K กลุ่ม โดยค่า K จะต้องมีค่าไม่มากกว่าจำนวนข้อมูลที่คัดเลือกไว้ ข้อมูลในรูปที่ 3.5 เป็นข้อมูลที่ใช้เป็นตัวอย่างเพื่อแสดงขั้นตอนการทำงานของโปรแกรมจัดกลุ่มข้อมูล

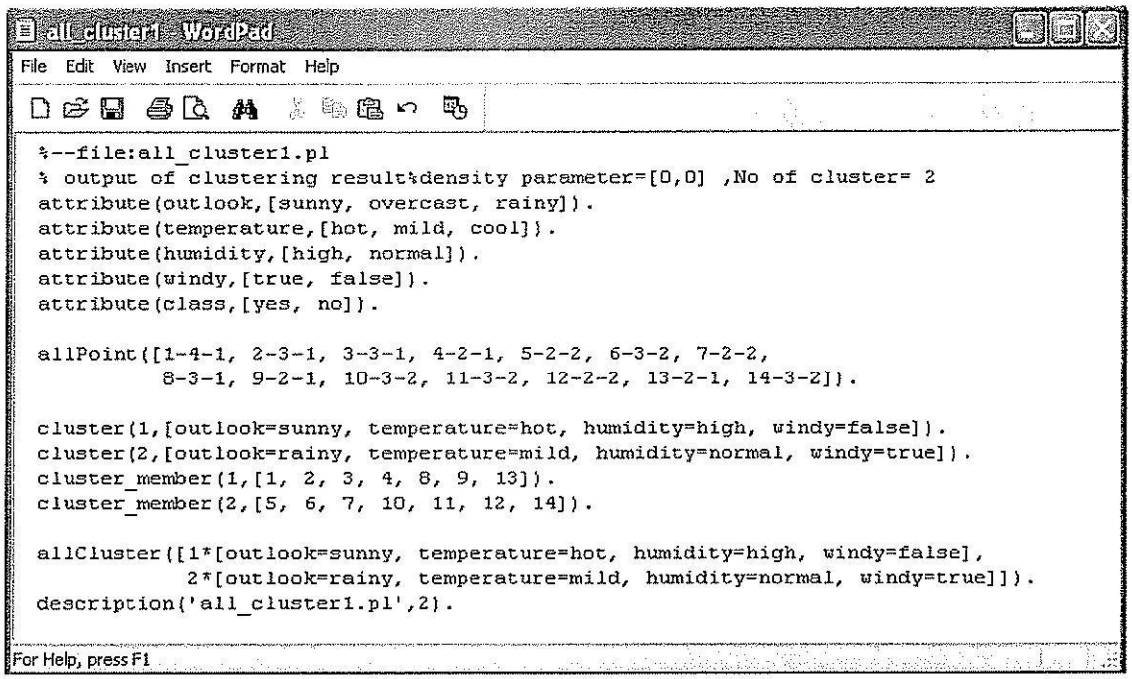
```
%% Data weather
%
%attributes: names and their possible values
%
attribute( outlook,      [sunny, overcast, rainy] ).
attribute( temperature, [hot, mild, cool]        ).
attribute( humidity,     [high, normal]          ).
attribute( windy,        [true, false]           ).
attribute( class,        [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 3.5 ตัวอย่างไฟล์ข้อมูลที่จะนำเข้ายังโปรแกรมจัดกลุ่มข้อมูลตามความหนาแน่น

โปรแกรมจัดกลุ่มข้อมูลตามความหนาแน่น เริ่มต้นทำงานโดยแสดงจอภาพดังรูปที่ 3.6 ให้ผู้ใช้ระบุชื่อแฟ้มข้อมูล จำนวนกลุ่ม (K) และพารามิเตอร์ที่ใช้ในการคำนวณความหนาแน่นของข้อมูล โดยพารามิเตอร์นี้จะประกอบด้วยจำนวนเลขสองจำนวน [M, D] ค่าM จะเป็นค่าจำนวนเต็มหมายถึงจำนวนแอททริบิวต์ของข้อมูลที่จะใช้คำนวณ similarity และค่า D จะเป็นค่าขั้นต่ำของความหนาแน่น (หมายถึง สัดส่วนข้อมูลที่อยู่ใกล้เคียงกับข้อมูลแต่ละเรคคอร์ด) โดย D จะมีค่าอยู่ระหว่าง 0.0 ถึง 1.0 การระบุค่า D เป็น 0 หรือ 0.0 หมายถึงให้จัดกลุ่มข้อมูลทุกเรคคอร์ดโดยไม่ต้องคำนึงถึงความหนาแน่นข้อมูล จากข้อมูลตัวอย่างในรูปข้างต้นเมื่อจัดกลุ่มโดยระบุความหนาแน่นเป็น 0 จะได้ผลลัพธ์ดังรูปที่ 3.7



รูปที่ 3.6 จอภาพเริ่มต้นของโปรแกรมจัดกลุ่มข้อมูลตามความหนาแน่น



รูปที่ 3.7 ผลลัพธ์ของการจัดกลุ่มข้อมูลเมื่อระบุค่าความหนาแน่นเป็นศูนย์

ผลลัพธ์ของการจัดกลุ่มเมื่อกำหนดพารามิเตอร์เป็น [0,0] และกำหนดให้จัดข้อมูลเป็นสองกลุ่ม ได้ลักษณะของกลุ่มเป็น

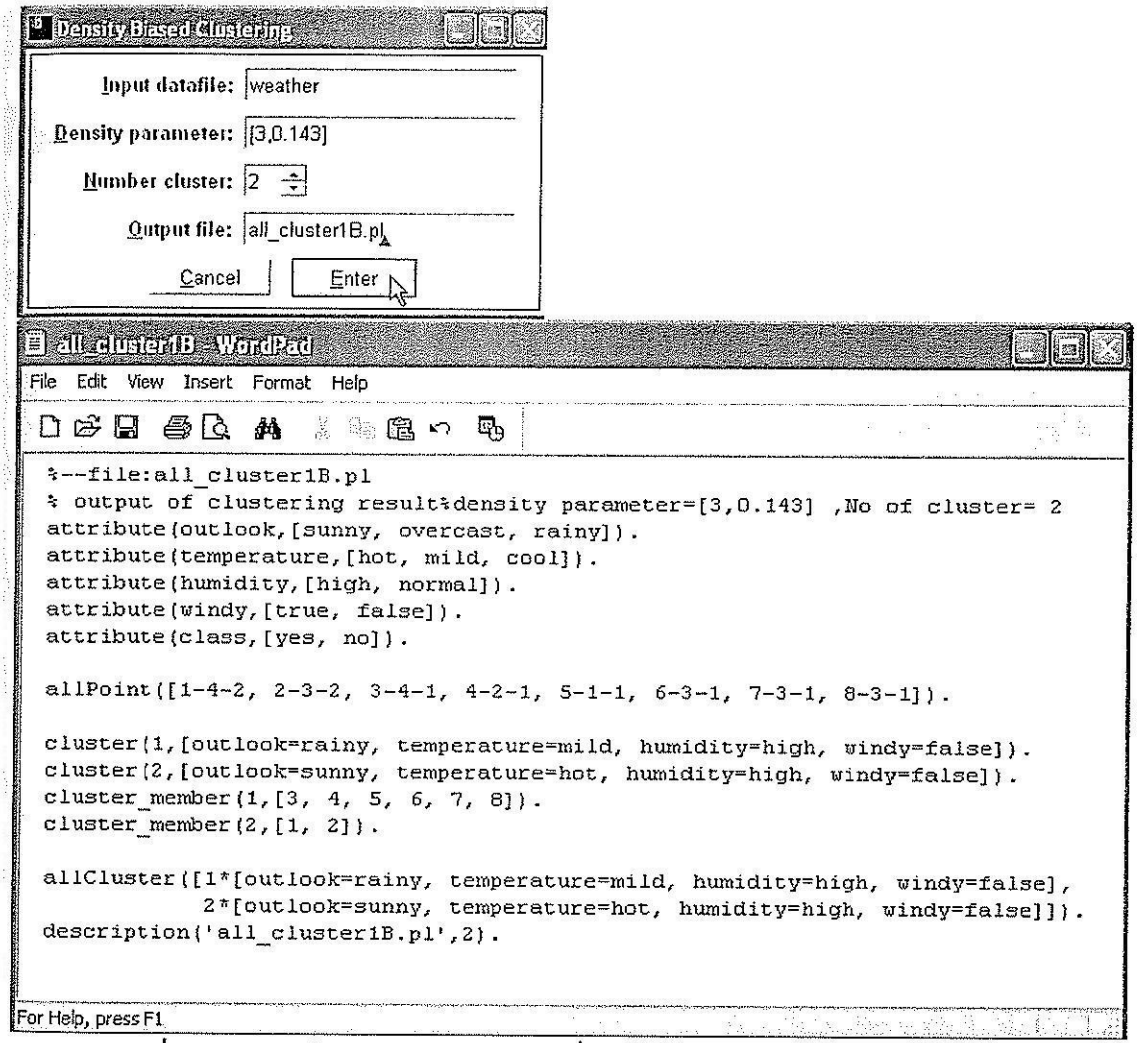
```
cluster(1,[outlook=sunny,temperature=hot,humidity=high,windy=false]).
cluster(2,[outlook=rainy,temperature=mild,humidity=normal,windy=true]).
```

และจัดข้อมูลทั้ง 14 เรคคอร์ดเข้ากลุ่มได้ดังนี้

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

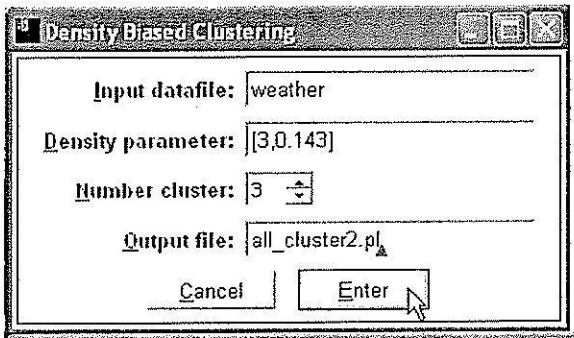
เมื่อกำหนดพารามิเตอร์ความหนาแน่นของข้อมูลเป็น [3, 0.143] ซึ่งหมายถึงวัดความใกล้เคียงของข้อมูลโดยใช้แอททริบิวต์อย่างต่ำ 3 แอททริบิวต์ และคัดเลือกข้อมูลที่มีข้อมูลอื่นอยู่

ใกล้เคียงคิดเป็นสัดส่วนอย่างน้อย 0.143 หรือ 14.3% (นั่นคือ ข้อมูลที่ถูกคัดเลือกจะต้องมีข้อมูลอื่นอยู่ใกล้เคียงคิดเป็น 0.143 × 14 เรคคอร์ด = 2 เรคคอร์ด) ข้อมูลที่มีค่าความหนาแน่นตรงตามเกณฑ์ดังกล่าวมีจำนวน 8 เรคคอร์ด และผลลัพธ์ของการจัดกลุ่มตามเกณฑ์ดังกล่าวแสดงดังรูปที่ 3.8



รูปที่ 3.8 ผลลัพธ์ของการจัดกลุ่มข้อมูลเมื่อระบุค่าความหนาแน่นเป็น [2, 0.143]

เมื่อพิจารณาลักษณะของข้อมูลในทั้งสองกลุ่ม จะเห็นว่าค่าของแอททริบิวต์ humidity และwindy ของทั้งสองกลุ่มจะเหมือนกัน ส่วนที่แตกต่างกันจะมีเพียงค่าของแอททริบิวต์ outlook และ temperature และเมื่อจัดกลุ่มข้อมูลด้วยเกณฑ์ความหนาแน่น [3, 0.143] และเปลี่ยนจำนวนกลุ่มของข้อมูลจากสองกลุ่มเป็นสามกลุ่ม จะได้ผลลัพธ์ดังรูปที่ 3.9



all_cluster2 - WordPad

File Edit View Insert Format Help

```
%--file:all_cluster2.pl
% output of clustering result%density parameter=[3,0.143] ,No of cluster= 3
attribute(outlook,[sunny, overcast, rainy]).
attribute(temperature,[hot, mild, cool]).
attribute(humidity,[high, normal]).
attribute(windy,[true, false]).
attribute(class,[yes, no]).

allPoint([1-4-1, 2-4-2, 3-3-3, 4-3-3, 5-2-3, 6-3-1, 7-4-3, 8-2-3]).

cluster(1,[outlook=sunny, temperature=hot, humidity=high, windy=false]).
cluster(2,[outlook=overcast, temperature=hot, humidity=high, windy=false]).
cluster(3,[outlook=rainy, temperature=mild, humidity=normal, windy=false]).
cluster_member(1,[1, 6]).
cluster_member(2,[2]).
cluster_member(3,[3, 4, 5, 7, 8]).

allCluster([1*[outlook=sunny, temperature=hot, humidity=high, windy=false],
            2*[outlook=overcast, temperature=hot, humidity=high, windy=false],
            3*[outlook=rainy, temperature=mild, humidity=normal, windy=false]]).
description('all_cluster2.pl',3).
```

รูปที่ 3.9 ผลลัพธ์ของการจัดข้อมูลเป็นสามกลุ่มและระบุค่าความหนาแน่นเป็น [3, 0.143]

3.1.3 การจัดกลุ่มข้อมูลแบบเพิ่มพูน

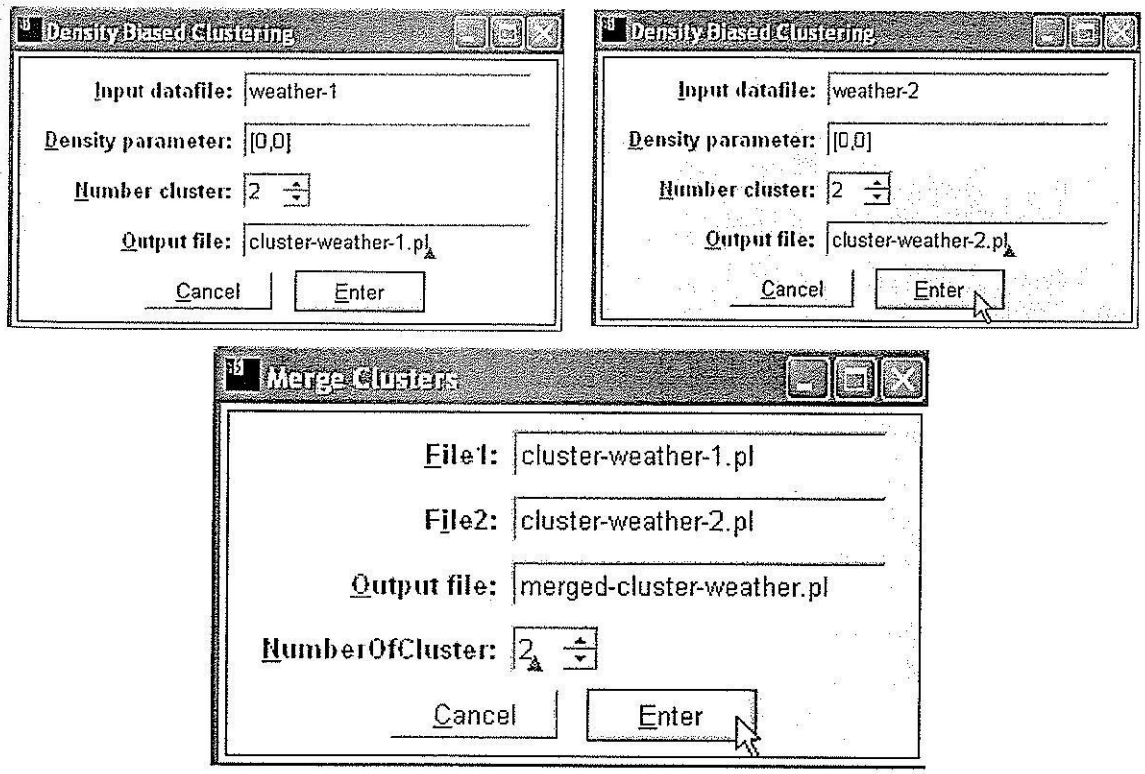
แนวคิดเกี่ยวกับการจัดกลุ่มข้อมูลแบบเพิ่มพูน หรือ incremental clustering เกิดจากการพยายามจัดกลุ่มกับข้อมูลที่มีขนาดใหญ่มาก ทำให้ต้องแบ่งจัดข้อมูลเป็นกลุ่มย่อยๆ จากนั้นจึงจะรวมข้อมูลในกลุ่มย่อย (merge clusters) ให้เป็นกลุ่มใหญ่ ในงานวิจัยนี้ใช้วิธีการรวมกลุ่มหรือคลัสเตอร์ โดยพิจารณาแต่ละ cluster mean ให้เป็นเสมือนหนึ่งรายการข้อมูล จากนั้น merge cluster means ให้ได้ค่า means ใหม่ ขั้นตอนการทำงานแสดงดังอัลกอริทึมต่อไปนี้

Algorithm 3.2 Incremental clustering

Input: cluster means from density-biased clustering
Output: a new set of merged cluster means

(1) Show GUI to obtain file names, F1 and F2, which are outputs from density-biased clustering
(2) Read number of desired clusters, K
(3) Read cluster descriptions from F1 and F2
(4) Treat cluster descriptions as data points and call the clustering process
(5) Output cluster means

จากข้อมูล weather ที่มีจำนวนข้อมูล 14 เรคคอร์ด เมื่อแบ่งข้อมูลออกเป็นไฟล์ย่อยสองไฟล์ โดยไฟล์แรก (weather-1) บันทึกข้อมูลเรคคอร์ที่ 1-7 และไฟล์ที่สอง (weather-2) บันทึกข้อมูลเรคคอร์ดที่ 8-14 จากนั้นทำ density-biased clustering กับข้อมูลแต่ละไฟล์ด้วยการกำหนดค่า K เป็น 2 และกำหนดพารามิเตอร์ความหนาแน่นของข้อมูลเป็น [0,0] บันทึกผลลัพธ์ของการทำ clustering กับข้อมูลทั้งสองชุดไว้ในไฟล์ cluster-weather-1.pl และ cluster-weather-2.pl ทำการรวมคลัสเตอร์ในทั้งสองไฟล์ด้วยการเรียกใช้โปรแกรม incremental clustering จะปรากฏจอภาพดังรูปที่ 3.10 เมื่อคลิกปุ่ม Enter จะได้ผลลัพธ์ของการรวมคลัสเตอร์ และปรากฏเป็นค่าคลัสเตอร์ใหม่ดังรูปที่ 3.11



รูปที่ 3.10 จอภาพของโปรแกรมการรวมคลัสเตอร์ในงานจัดกลุ่มข้อมูลแบบเพิ่มพูน

```
%--file:cluster-weather-1.pl
% output of clustering result
%density parameter=[0,0] ,No of cluster= 2
attribute(outlook,[sunny, overcast, rainy]).
attribute(temperature,[hot, mild, cool]).
attribute(humidity,[high, normal]).
attribute(windy,[true, false]).
attribute(class,[yes, no]).
allPoint([1-4-1, 2-3-1, 3-3-1, 4-2-1, 5-3-2, 6-4-2, 7-3-2]).

cluster(1,[outlook=sunny, temperature=hot,
        humidity=high, windy=false]).
cluster(2,[outlook=rainy, temperature=cool,
        humidity=normal, windy=true]).
cluster_member(1,[1, 2, 3, 4]).
cluster_member(2,[5, 6, 7]).

allCluster([1*[outlook=sunny, temperature=hot,
humidity=high, windy=false], 2*[outlook=rainy,
temperature=cool, humidity=normal, windy=true]]).
description('cluster-weather-1.pl',2).
```

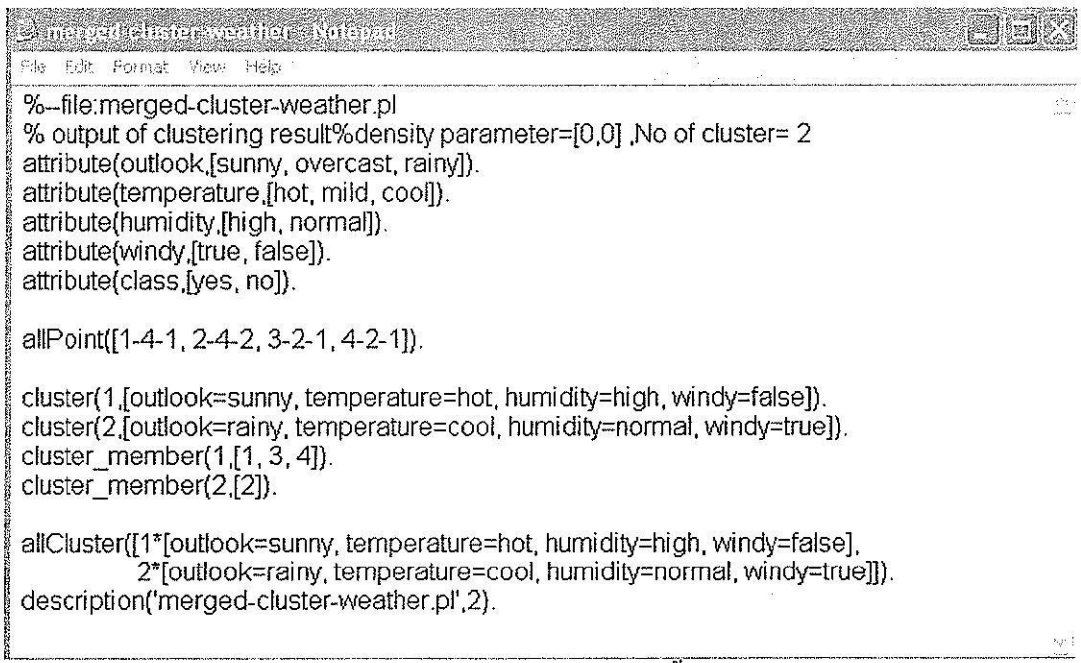
(a) ผลลัพธ์จากการจัดกลุ่มข้อมูลในไฟล์ weather-1

```
%--file:cluster-weather-2.pl
% output of clustering result
%density parameter=[0,0] ,No of cluster= 2
attribute(outlook,[sunny, overcast, rainy]).
attribute(temperature,[hot, mild, cool]).
attribute(humidity,[high, normal]).
attribute(windy,[true, false]).
attribute(class,[yes, no]).
allPoint([1-3-1, 2-4-2, 3-2-2, 4-3-1, 5-3-1, 6-2-2, 7-3-1]).

cluster(1,[outlook=sunny, temperature=mild,
        humidity=high, windy=true]).
cluster(2,[outlook=sunny, temperature=cool,
        humidity=normal, windy=false]).
cluster_member(1,[1, 4, 5, 7]).
cluster_member(2,[2, 3, 6]).

allCluster([1*[outlook=sunny, temperature=mild,
humidity=high, windy=true], 2*[outlook=sunny,
temperature=cool, humidity=normal, windy=false]]).
description('cluster-weather-2.pl',2).
```

(b) ผลลัพธ์จากการจัดกลุ่มข้อมูลในไฟล์ weather-2

```
%file tmp.pl
:-dynamic attribute/2, instance/3.
attribute(outlook,[sunny, overcast, rainy]).
attribute(temperature,[hot, mild, cool]).
attribute(humidity,[high, normal]).
attribute(windy,[true, false]).
attribute(class,[yes, no]).
instance(1,class=yes,[outlook=sunny, temperature=hot, humidity=high, windy=false]).
instance(2,class=yes,[outlook=rainy, temperature=cool, humidity=normal, windy=true]).
instance(3,class=no,[outlook=sunny, temperature=mild, humidity=high, windy=true]).
instance(4,class=no,[outlook=sunny, temperature=cool, humidity=normal, windy=false]).
```

(c) ข้อมูลในไฟล์ชั่วคราวที่เกิดจากการรวมผลลัพธ์ของการจัดกลุ่มข้อมูลในขั้นตอน (a) และ (b)



(d) ผลลัพธ์สุดท้ายของการรวมคลัสเตอร์ในขั้นตอน (a) และ (b)

รูปที่ 3.11 ผลลัพธ์ของการจัดกลุ่มข้อมูลแบบเพิ่มพูนกับข้อมูล weather

เมื่อพิจารณาผลของการจัดกลุ่มข้อมูลแบบปกติ (เรียกว่าแบบ batch) ที่รวมข้อมูลทั้งหมดแล้วจัดเข้ากลุ่มในคราวเดียว เปรียบเทียบกับวิธี incremental ที่มีการแบ่งข้อมูลให้เป็นไฟล์ขนาดเล็กหลายไฟล์แล้วแยกจัดกลุ่มข้อมูลในแต่ละไฟล์ จากนั้นนำคลัสเตอร์ที่ได้มารวมจัดกลุ่มใหม่จะได้ผลลัพธ์เป็นค่า means (หรือลักษณะส่วนใหญ่ของกลุ่ม) ที่ใกล้เคียงกันมาก มีเพียงลักษณะในแอททริบิวต์ temperature ของคลัสเตอร์ที่สองเท่านั้นที่ค่าแตกต่างกัน สรุปการเปรียบเทียบได้ดังตารางที่ 3.2

ถ้าหากใช้ลักษณะค่า means นี้ไปจัดข้อมูลแต่ละเรคคอร์ดเข้ากลุ่มจะได้ผลการจัดกลุ่มที่ค่อนข้างใกล้เคียงกัน ดังแสดงในตารางที่ 3.3 การจัดข้อมูลเข้ากลุ่มในกรณีที่ข้อมูลมีระยะห่างจากคลัสเตอร์ที่หนึ่งและสองเท่ากัน จะแสดงผลการจัดกลุ่มเป็น “กลุ่ม 1 หรือ 2” ซึ่งหมายถึงจะจัดข้อมูลเข้ากลุ่มใดก็ได้ ถ้าหากผลการจัดกลุ่มเป็น “กลุ่ม 1 หรือ 2” แล้วตัดสินให้จัดข้อมูลเข้ากลุ่ม 1 จะได้ผลสรุปว่าวิธีการจัดข้อมูลเข้ากลุ่มแบบ incremental ให้ผลแตกต่างจากวิธีแบบ batch เพียงหนึ่งเรคคอร์ดจากข้อมูลทั้งหมด 14 เรคคอร์ด หรือคิดเป็น 7.14% ซึ่งจากผลการเปรียบเทียบนี้ทำให้คาดหมายได้ว่าเมื่อข้อมูลมีปริมาณมากกว่านี้ ความแตกต่างในผลการจัดกลุ่มแบบ batch และแบบ incremental จะมีนัยสำคัญที่ลดลง

ตารางที่ 3.2 เปรียบเทียบค่า means ของการจัดกลุ่มแบบ batch และแบบ incremental

วิธีการจัดกลุ่มข้อมูล	**ค่า means**
แบบ batch	Cluster1:[outlook=sunny,temperature=hot,humidity=high,windy=false] Cluster2:[outlook=rainy,temperature=mild,humidity=normal,windy=true]
แบบ incremental	Cluster1:[outlook=sunny,temperature=hot,humidity=high,windy=false] Cluster2:[outlook=rainy,temperature=cool,humidity=normal,windy=true]

ตารางที่ 3.3 ผลการจัดข้อมูลเข้ากลุ่มของวิธีการจัดกลุ่มแบบ batch และแบบ incremental

ข้อมูลเรคคอร์ดที่	ผลการจัดกลุ่มแบบ batch	ผลการจัดกลุ่มแบบ incremental
1	กลุ่ม 1	กลุ่ม 1
2	กลุ่ม 1	กลุ่ม 1
3	กลุ่ม 1	กลุ่ม 1
4	กลุ่ม 1	กลุ่ม 1 หรือ 2
5	กลุ่ม 2	กลุ่ม 2
6	กลุ่ม 2	กลุ่ม 2
7	กลุ่ม 2	กลุ่ม 2
8	กลุ่ม 1	กลุ่ม 1
9	กลุ่ม 1 หรือ 2	กลุ่ม 1
10	กลุ่ม 2	กลุ่ม 2
11	กลุ่ม 2	กลุ่ม 2
12	กลุ่ม 1 หรือ 2	กลุ่ม 2
13	กลุ่ม 1	กลุ่ม 1
14	กลุ่ม 2	กลุ่ม 2

3.2 การทดสอบโปรแกรมการทำเหมืองข้อมูลแบบจัดกลุ่ม

3.2.1 วิธีการทดสอบประสิทธิภาพของการจัดกลุ่ม

โปรแกรมที่ใช้ในงานจัดกลุ่มข้อมูลประกอบด้วยโปรแกรม density-biased clustering ทำหน้าที่คัดเลือกข้อมูลตามความหนาแน่นเพื่อนำไปจัดกลุ่ม และโปรแกรม incremental clustering ทำหน้าที่จัดกลุ่มข้อมูลเช่นเดียวกัน แต่ใช้เทคนิคการแบ่งข้อมูลเป็นส่วนเล็กๆหลายส่วน เพื่อจัดข้อมูลในแต่ละกลุ่มเล็กให้เป็นคลัสเตอร์ จากนั้นขยายขอบเขตการจัดกลุ่มโดยรวมลักษณะเด่นในแต่ละกลุ่มย่อยของข้อมูล (merge clusters)โดยใช้ค่า means ของแต่ละกลุ่มย่อย มาพิจารณาเพื่อรวมเป็นกลุ่มใหญ่ ทำให้ได้ผลลัพธ์เป็นค่า means ค่าใหม่ที่เป็นตัวแทนข้อมูลของหลายกลุ่มย่อย จากนั้นรวมกลุ่มย่อยของข้อมูลเช่นนี้จนครบจำนวนข้อมูลทั้งหมด

การทดสอบประสิทธิภาพของโปรแกรม จะใช้วิธีทดสอบผลการรวมกลุ่มข้อมูลของวิธี incremental clustering ที่ทยอยจัดกลุ่มข้อมูลที่มีขนาดเล็กแล้วรวมค่า means ของข้อมูลกลุ่มย่อย ให้เป็นค่า means ของกลุ่มที่ใหญ่ขึ้น เปรียบเทียบกับวิธีการจัดกลุ่มข้อมูลแบบ batch ซึ่งจะต้อง

รวบรวมข้อมูลทั้งหมดให้เป็นไฟล์เดียวแล้วจึงทำการจัดกลุ่มข้อมูล การเปรียบเทียบผลการจัดกลุ่มจะใช้การพิจารณาข้อมูลที่เป็นสมาชิกในแต่ละกลุ่มของวิธีจัดกลุ่มแบบ incremental เปรียบเทียบกับสมาชิกของกลุ่มที่จัดแบบ batch และเปรียบเทียบคุณภาพการรวมกลุ่มของข้อมูลด้วยการคำนวณค่า sum of squared errors (SSE) ค่านี้จะเป็นค่าโดยอ้อมในการบ่งชี้ความหนาแน่นของกลุ่มข้อมูล ค่า SSE ที่ต่ำกว่าจะหมายถึงการเกาะกลุ่มของข้อมูลที่ดีกว่า

ข้อมูลที่ใช้ในการทดสอบได้แก่ข้อมูล post-operative (เป็นข้อมูลเกี่ยวกับการสังเกตอาการของคนไข้ภายหลังการผ่าตัด จำนวนข้อมูล 86 เรคคอร์ด ข้อมูลแต่ละเรคคอร์ดประกอบด้วย 8 แอททริบิวต์) และข้อมูล breast cancer (เป็นข้อมูลเกี่ยวกับการวินิจฉัยการเกิดซ้ำของมะเร็งในคนไข้ที่เคยเป็นมะเร็งเต้านม จำนวนข้อมูล 191 เรคคอร์ด ข้อมูลแต่ละเรคคอร์ดประกอบด้วย 9 แอททริบิวต์)

ขั้นตอนการทดลองจะเริ่มต้นด้วยการเตรียมข้อมูลเพื่อจะเป็นข้อมูลเข้าให้กับโปรแกรม density-biased clustering โดยโปรแกรมนี้สามารถจัดกลุ่มข้อมูลแบบ batch ได้โดยการกำหนดพารามิเตอร์ความหนาแน่นของข้อมูลเป็น [0,0] ซึ่งหมายถึงไม่ต้องมีการคัดเลือกข้อมูลที่หนาแน่นถึงเกณฑ์ แต่จะใช้ข้อมูลทั้งหมดในการจัดกลุ่มเพื่อหาลักษณะเด่นของกลุ่มข้อมูล จากนั้นใช้ลักษณะเด่นที่ได้นี้ไปเป็นเกณฑ์ในการจัดข้อมูลแต่ละเรคคอร์ดเข้ากลุ่ม

ในการเตรียมข้อมูลเพื่อทดสอบกับโปรแกรม incremental clustering จะแบ่งข้อมูลเป็นสี่ส่วนย่อย จากนั้นทำการจัดกลุ่มในแต่ละส่วนย่อย ผลลัพธ์ที่ได้จะเป็นลักษณะเด่นของกลุ่มข้อมูลในแต่ละส่วนย่อย (หรือค่า means) จากนั้นใช้ลักษณะเด่นนี้เป็นเสมือนข้อมูล เพื่อทำการ merge ค่า means ของแต่ละคลัสเตอร์ เมื่อรวมค่า means ของข้อมูลทั้งสี่ส่วนย่อยเสร็จ จะได้ค่า means สุดท้ายของข้อมูลทั้งหมด นำค่า means สุดท้ายนี้เป็นเกณฑ์ในการจัดข้อมูลแต่ละเรคคอร์ดเข้ากลุ่ม ตรวจสอบผลลัพธ์ของวิธีจัดกลุ่มแบบ incremental เปรียบเทียบกับแบบ batch โดยการพิจารณาความแตกต่างของการจัดข้อมูลเข้ากลุ่มว่ามีข้อมูลกี่เรคคอร์ดที่ถูกจัดเข้ากลุ่มแตกต่างกัน และพิจารณาการเกาะกลุ่มของข้อมูลที่ได้จากการจัดทั้งสองแบบด้วยค่า SSE รวมถึงพิจารณาเวลาที่ใช้ในกระบวนการจัดกลุ่มทั้งหมด

3.2.2 ผลการทดสอบและอภิปรายผล

การทดสอบประสิทธิภาพของวิธีการการจัดกลุ่มข้อมูลในแบบ batch เปรียบเทียบกับแบบ incremental ของข้อมูล post-operative แสดงผลการทดสอบได้ดังตารางที่ 3.4 และผลการทดสอบกับข้อมูล breast cancer แสดงได้ดังตารางที่ 3.5 ผลการทดสอบที่แสดงในตารางทั้งสองนี้เป็นขั้นตอนเริ่มต้นที่ยังไม่พิจารณาคัดเลือกข้อมูลตามความหนาแน่น โดยในการรันโปรแกรม density-biased clustering จะกำหนดพารามิเตอร์ค่าความหนาแน่นเป็น [0,0]

การทดสอบในขั้นตอนที่สอง เป็นการตรวจสอบผลของการคัดเลือกข้อมูลตามความหนาแน่นที่เกณฑ์ขั้นต่ำขนาดต่างๆเพื่อจัดกลุ่มข้อมูลทั้งในแบบ batch และแบบ incremental เกณฑ์ความหนาแน่นจะเริ่มทดสอบที่ [1,0.1] หมายถึงการคำนวณความหนาแน่นของข้อมูลจะพิจารณาค่า similarity ของหนึ่งแอททริบิวต์และจะต้องมีข้อมูลที่คล้ายคลึงกับข้อมูลนั้นอย่างต่ำ 0.1 หรือ 10% เกณฑ์ที่ใช้จะเพิ่มขึ้นเป็น [1,0.15], [1,0.2], ..., [4,0.2] ผลการทดสอบในขั้นตอนนี้จะแสดงเฉพาะค่า SSE ของผลการจัดกลุ่มข้อมูลที่แต่ละค่าความหนาแน่น และที่จำนวนกลุ่มข้อมูล (ค่า K) ที่ขนาดต่างๆกัน ผลการทดสอบกับข้อมูล post-operative แสดงดังตารางที่ 3.6 และผลการทดสอบกับข้อมูล breast cancer แสดงดังตารางที่ 3.7

ตารางที่ 3.4 เปรียบเทียบผลการจัดกลุ่มข้อมูล post-operative ด้วยวิธีจัดกลุ่มแบบ batch และแบบ incremental

จำนวนกลุ่ม (K)	การจัดกลุ่มแบบ batch			การจัดกลุ่มแบบ incremental			ความแตกต่างของการจัดข้อมูลเข้ากลุ่ม
	จำนวนข้อมูลในกลุ่ม	เวลาที่ใช้ (วินาที)	ค่า SSE	จำนวนข้อมูลในกลุ่ม	เวลาที่ใช้ (วินาที)	ค่า SSE	
2	Cluster1=40 Cluster2=46	1.23	1742.88	Cluster1=42 Cluster2=44	1.06	1695.53	6.97%
3	Cluster1=36 Cluster2=22 Cluster3=28	1.40	1401.66	Cluster1=33 Cluster2=26 Cluster3=27	1.18	1368.74	9.30%
4	Cluster1=26 Cluster2=21 Cluster3=21 Cluster4=18	1.44	1281.33	Cluster1=28 Cluster2=20 Cluster3=21 Cluster4=17	1.17	1195.67	8.14%
5	Cluster1=20 Cluster2=19 Cluster3=14 Cluster4=17 Cluster5=16	1.51	1017.34	Cluster1=23 Cluster2=20 Cluster3=16 Cluster4=17 Cluster5=10	1.33	1145.98	11.63%
6	Cluster1=17 Cluster2=14 Cluster3=16 Cluster4=14 Cluster5=15 Cluster6=10	1.49	967.85	Cluster1=18 Cluster2=15 Cluster3=15 Cluster4=14 Cluster5=14 Cluster6=10	1.40	913.33	6.97%
7	Cluster1=15 Cluster2=11 Cluster3=13 Cluster4=10 Cluster5=19 Cluster6=11 Cluster7=7	1.62	712.63	Cluster1=14 Cluster2=10 Cluster3=14 Cluster4=11 Cluster5=13 Cluster6=12 Cluster7=12	1.48	654.87	10.46%