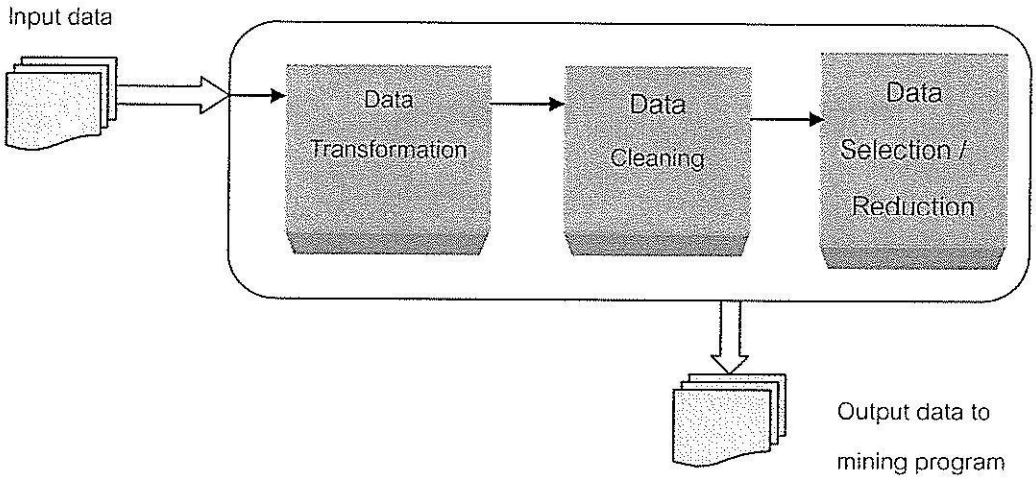
บทที่ 2

วิธีการเตรียมข้อมูลอัตโนมัติก่อนการทำเหมืองข้อมูล (โครงการวิจัยที่ 1)

2.1 วิธีดำเนินการวิจัยโครงการวิจัยที่ 1

2.1.1 กรอบแนวคิดของงานวิจัย

โครงการวิจัยนี้มีวัตถุประสงค์หลักในการออกแบบและพัฒนาซอฟต์แวร์ ที่ทำหน้าที่เตรียมข้อมูล ให้โปรแกรมทำเหมืองข้อมูลสามารถประมวลผลได้อย่างมีประสิทธิภาพ กรอบแนวคิดเชิงโครงสร้างของซอฟต์แวร์แสดงได้ดังแผนภาพต่อไปนี้



รูปที่ 2.1 โครงสร้างซอฟต์แวร์เตรียมข้อมูลก่อนการทำเหมืองข้อมูล

กระบวนการเตรียมข้อมูลจะประกอบด้วยการแปลงข้อมูล (data transformation) การปรับปรุงข้อมูล (data cleaning) และการคัดเลือกข้อมูล (data selection/reduction) ขั้นตอนต่างๆในการดำเนินงานเพื่อพัฒนาซอฟต์แวร์ มีดังต่อไปนี้

ขั้นตอนที่ 1 รวบรวมเอกสารอ้างอิงที่เกี่ยวข้อง และคัดเลือกข้อมูล

รวบรวมเอกสารเกี่ยวกับเทคนิคการจัดการกรณี missing values และเทคนิคการสุ่มข้อมูล (sampling techniques) รวมถึงคัดเลือกข้อมูลเพื่อใช้ทดสอบซอฟต์แวร์ที่สร้างขึ้น ข้อมูลที่ใช้จะต้องมีขนาดใหญ่เพียงพอที่จะเอื้อให้สามารถทดสอบประสิทธิภาพของส่วน reduction ได้ และเป็นข้อมูลที่มีบางส่วนขาดหายไปเพื่อให้สามารถทดสอบส่วน cleaning ได้ ข้อมูลที่จะคัดเลือกเพื่อใช้ในงานวิจัยนี้ส่วน

หนึ่งจะค้นหามาจาก UCI Repository (http://archive.ics.uci.edu/ml/datasets.html) ซึ่งเป็นแหล่งรวมข้อมูลที่นิยมใช้ในการทำ machine learning

ขั้นตอนที่ 2 ออกแบบโปรแกรม

วางแผนและออกแบบโปรแกรมทั้งสามส่วนของซอฟต์แวร์ และส่วนเชื่อมต่อที่จะประสานการทำงานของทุกส่วน รวมทั้งออกแบบและพัฒนาส่วนที่จะติดต่อกับผู้ใช้ (graphical user interface, GUI)

ขั้นตอนที่ 3 ออกแบบ พัฒนา และทดสอบโปรแกรมส่วน data transformation

โปรแกรมแปลงข้อมูลหรือ data transformation ทำหน้าที่แปลงไฟล์ข้อมูลให้ใช้ได้กับระบบการทำเหมืองข้อมูล ข้อมูลเริ่มต้นที่จะเข้ามาสู่ซอฟต์แวร์ทำเหมืองข้อมูลเป็นข้อมูลที่ประกอบด้วยสองไฟล์ย่อยคือ names file และ data file รูปแบบนี้มักจะปรากฏกับข้อมูลใน UCI repository โปรแกรมแปลงข้อมูลที่พัฒนาขึ้นจะสามารถรวม names file และ data file จากนั้นเปลี่ยนรูปแบบข้อมูลแต่ละรายการให้เป็น Horn clauses ที่ภาษาโปรล็อกสามารถประมวลผลได้

ขั้นตอนที่ 4 ออกแบบ พัฒนา และทดสอบโปรแกรมส่วน data cleaning

อัลกอริทึมสังเคราะห์ความรู้ (learning or mining algorithm) จำนวนมากไม่สามารถทำงานได้ถ้า input file มีข้อมูลไม่ครบ โปรแกรม data cleaning จึงถูกสร้างขึ้นเพื่อทำหน้าที่เติมข้อมูลที่ขาดหายไปเพื่อให้ input file สมบูรณ์ ดังนั้นโปรแกรม data cleaning จะต้องมีความสามารถตรวจหา missing values ใน input file ได้โดยอัตโนมัติ และบรรจุค่าทดแทนส่วนที่ขาดหายไป แนวทางที่ใช้ในงานวิจัยนี้มีได้ 3 แนวทางคือ

(1) ทดแทนด้วยค่าคงที่ที่สร้างขึ้นมาใหม่, หรือ

(2) ทดแทนด้วยค่าส่วนใหญ่ของแอททริบิวต์นั้นๆ, หรือ

(3) ผู้ใช้อาจเลือกตัดแอททริบิวต์ที่มี missing values ออกไป

ขั้นตอนที่ 5 ออกแบบ พัฒนา และทดสอบโปรแกรมส่วน data selection / reduction

โปรแกรม data reduction จะทำหน้าที่ลดขนาดข้อมูลให้สามารถประมวลผลได้ในขั้น mining การลดขนาดจะใช้เทคนิคการสุ่มข้อมูล (sampling) เป็นหลัก โดยผู้ใช้สามารถกำหนดวิธีการสุ่มข้อมูลได้หลายลักษณะ ได้แก่

- การสุ่มจากข้อมูลดิบโดยไม่ใส่ข้อมูลกลับคืน (random sampling without replacement)

- การสุ่มจากข้อมูลดิบโดยใส่ข้อมูลกลับคืน (random sampling with replacement) ซึ่งวิธีการนี้จะทำให้ข้อมูลที่ถูกสุ่มปรากฏซ้ำได้
- การสุ่มตามความหนาแน่นของข้อมูล (density-biased sampling) วิธีการนี้จะพิจารณาความหนาแน่นของกลุ่มข้อมูลประกอบการสุ่ม

ขั้นตอนที่ 6 เชื่อมต่อโปรแกรมทั้งสามส่วนของซอฟต์แวร์เตรียมข้อมูล

เมื่อโปรแกรมทั้งสามส่วนเสร็จสมบูรณ์ จะทดสอบความถูกต้องด้วยข้อมูลที่คัดเลือกไว้ในขั้นตอนแรก โดยใช้ข้อมูลจาก UCI repository เมื่อได้ข้อมูลที่ผ่านกระบวนการเตรียมข้อมูลแล้ว ข้อมูลนั้นจะถูกส่งไปทดสอบการใช้งานได้จริง ด้วยการรันบนระบบเหมืองข้อมูล

2.1.2 การออกแบบและพัฒนาวิธีการแปลงและปรับปรุงข้อมูล

การแปลงรูปแบบไฟล์ข้อมูล (data transformation) จำแนกเป็นสองงานย่อยคือ การรวมสองไฟล์คือ names file และ data file ให้เป็นข้อมูลไฟล์เดียว จากนั้นแปลงข้อมูลให้อยู่ในรูปแบบ Horn clauses ซึ่งเป็นรูปแบบข้อความในภาษาโปรล็อก

ตัวอย่างของข้อมูลในรูปแบบของ UCI repository แสดง names file และ data file ได้ดังรูปที่ 2.2 รายการข้อมูลจะปรากฏใน data file ข้อมูลหนึ่งบรรทัดคือหนึ่งเรคคอร์ด แต่ละแอททริบิวต์คั่นด้วยเครื่องหมาย ‘,’ หรือเรียกว่ารูปแบบ CSV (comma separated value) ส่วนที่เป็นคำอธิบายโครงสร้างข้อมูลจะปรากฏใน names file บรรทัดแรกของ names file จะเป็นค่าที่เป็นไปได้ทั้งหมดของคลาส บรรทัดอื่นต่อจากนั้นจะเป็นคำอธิบายชื่อแอททริบิวต์และค่าที่เป็นไปได้ทั้งหมดของแอททริบิวต์นั้น เมื่อข้อมูลเดียวกันนี้ถูกเปลี่ยนให้อยู่ในรูปแบบ Horn clauses จะมีลักษณะดังรูปที่ 2.3

```
yes, no.

outlook: sunny, overcast, rain.
temperature: hot, mild, cool.
humidity: high, normal.
windy: true, false.
```

names file

```
sunny, hot, high, false, no
sunny, hot, high, true, no
overcast, hot, high, false, yes
rain, mild, high, false, yes
rain, cool, normal, false, yes
rain, cool, normal, true, no
overcast, cool, normal, true, yes
sunny, mild, high, false, no
sunny, cool, normal, false, yes
rain, mild, normal, false, yes
sunny, mild, normal, true, yes
overcast, mild, high, true, yes
overcast, hot, normal, false, yes
rain, mild, high, true, no
```

data file

รูปที่ 2.2 ตัวอย่างไฟล์ข้อมูลในรูปแบบ UCI repository

```
% Golf data set
%
%       Header file for attribute declaration
attribute( outlook,     [sunny, overcast, rainy] ).
attribute( temperature, [hot, mild, cool]        ).
attribute( humidity,    [high, normal]           ).
attribute( windy,       [true, false]            ).
attribute( class,       [yes, no]                ).
%       Data instances
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 2.3 ตัวอย่างไฟล์ข้อมูลในรูปแบบ Horn clauses

ข้อมูลในรูปแบบ Horn clauses จะยังคงรักษาโครงสร้างของไฟล์ข้อมูลที่ประกอบด้วยส่วนอธิบายแอททริบิวต์ และส่วนรายละเอียดข้อมูล เพียงแต่ข้อมูลจะถูกเปลี่ยนจากข้อความปกติเป็นข้อความที่เรียกว่า clause ในภาษาโปรล็อกที่มีข้อกำหนดว่าแต่ละ clause จะสิ้นสุดด้วยเครื่องหมาย '.' และข้อมูลหนึ่งรายการจะถูกแปลงเป็น Horn clause หนึ่งคลอส ขั้นตอนในการแปลงรูปแบบข้อมูลจาก UCI repository เป็น Horn clauses จะเริ่มจากการแปลงโครงสร้างข้อมูลใน names file ให้เป็นคลอส attribute(AttributeName, [List-of-AttributeValues]) เมื่อแปลงส่วนโครงสร้างเสร็จแล้ว จึงจะเริ่มอ่าน data file และแปลงข้อมูลแต่ละรายการเป็นคลอส instance(ID, Class= Value, [AttributeName=Value]) รายละเอียดการทำงานแสดงได้ดังอัลกอริทึมที่ 2.1

Algorithm 2.1 Data transformation

Input: names file and data file
Output: a data file as Horn clauses

---

(1) Open names file and read the value list in the first line
(2) Write to the output file the clause 'attribute(class, [value list]).'
(3) While not end-of-file do
(3.1) Read attribute name
(3.2) Read attribute-value list
(3.3) Write to the output file the clause 'attribute(attribute name, [value list]).'
(4) Close names file
(5) Open data file
(6) Set instance counter I = 1
(7) While not end-of-file do
(7.1) Read each value that appears in CSV format
(7.2) Match attribute name with corresponding attribute value
(7.3) Construct attribute-value pair format (i.e. attributeName=value)
(7.4) Remove last attribute-value pair to be class-value
(7.5) Write a clause 'instance(I, class-value, [attribute-value list]).' to output file
(7.6) Increment I
(8) Close data file
(9) Return the output file

ข้อมูลที่ผ่านขั้นตอนการแปลงไฟล์ข้อมูลแล้ว จะถูกส่งต่อมายังส่วนปรับปรุงข้อมูล (data cleaning) การปรับปรุงข้อมูลจะเกิดขึ้นเมื่อข้อมูลปรากฏ missing values ซึ่งในรูปแบบของข้อมูล UCI repository เมื่อข้อมูลบางแอททริบิวต์หายไปหรือไม่ปรากฏค่า จะปรากฏสัญลักษณ์ '?' ในตำแหน่งของข้อมูลที่แอททริบิวต์นั้น ตัวอย่างของข้อมูลที่มี missing values แสดงได้ดังรูปที่ 2.4

```
attribute( outlook,      [sunny, overcast, rainy] ).
attribute( temperature, [hot, mild, cool]        ).
attribute( humidity,     [high, normal]          ).
attribute( windy,        [true, false]           ).
attribute( class,        [yes, no]               ).
%        Data instances
instance(1, class=no, [outlook=sunny, temperature=hot, humidity=?, windy=?]).
instance(2, class=no, [outlook=sunny, temperature=hot, humidity=high, windy=true]).
instance(3, class=yes, [outlook=overcast, temperature=hot, humidity=high, windy=?]).
instance(4, class=yes, [outlook=rainy, temperature=mild, humidity=high, windy=false]).
...
```

รูปที่ 2.4 ตัวอย่างไฟล์ข้อมูลที่ปรากฏ *missing values*

การปรากฏ missing values จะมีผลเสียต่อกระบวนการทำเหมืองข้อมูล ทำให้ได้โมเดลที่ความเชื่อมั่นลดลง ข้อมูลที่เป็นตัวแทนที่ดีจึงไม่ควรมีค่า missing วิธีจัดการกับกรณีข้อมูลหายไปในงานวิจัยนี้จะสร้างทางเลือกให้ผู้ใช้เลือกได้สามแนวทางคือ (1) ทดแทนค่าที่หายไปด้วยข้อความ 'missing' และเมื่อในโมเดลที่เป็นเอาท์พุทจากการทำเหมืองข้อมูล ปรากฏคำว่า 'missing' ผู้ใช้จะสามารถแปลความหมายได้ว่าหมายถึงการไม่ปรากฏค่าในแอททริบิวต์, (2) ทดแทนค่าที่หายไปด้วยค่าส่วนใหญ่ของแอททริบิวต์ ในกรณีนี้เมื่อผู้ใช้แปลผลโมเดลจะไม่ทราบว่าโมเดลนั้นได้จากการวิเคราะห์ข้อมูลที่มี missing values, และ (3) ผู้ใช้สามารถเลือกตัดทิ้งแอททริบิวต์ที่มี missing values ปรากฏอยู่มาก รายละเอียดการทำงานในส่วนปรับปรุงข้อมูลแสดงได้ดังอัลกอริทึมที่ 2.2

Algorithm 2.2 Data cleaning

Input: a data file containing Horn clauses with *some* missing values
Output: a data file that its contents had been cleaned

(1) Show the GUI of data cleaning component
(2) Get the response from user's choice
(3) If choice = 'replace with missing', then
(3.1) Read each data instance
(3.2) Replace attributeName=? with attributeName=missing
(4) If choice = 'replace with majority', then
(4.1) Scan the whole data set
(4.2) Count the majority value of each attribute
(4.3) Read each data instance
(4.3) Replace attributeName=? with attributeName=majority
(5) If choice = 'remove attribute with missing values', then
Call feature selection module to remove unwanted attributes
(6) Return the output file that all the missing cases have been cleaned

2.1.3 การออกแบบและพัฒนาวิธีการคัดเลือกข้อมูล

การคัดเลือกข้อมูลในงานวิจัยนี้หมายถึง การคัดเลือกแอททริบิวต์ (feature selection) และการคัดเลือกข้อมูลบางรายการด้วยวิธีการสุ่ม (random sampling) การคัดเลือกแอททริบิวต์จะมีประโยชน์ในการลดขนาดของโมเดล และการสุ่มจะช่วยลดขนาดของข้อมูล และอาจช่วยให้ได้ข้อมูลที่เป็นตัวแทนที่ดีเพื่อการสังเคราะห์โมเดลที่มีความแม่นตรงสูงขึ้น

การคัดเลือกแอททริบิวต์จะใช้วิธีวิเคราะห์การกระจายของข้อมูล แล้วแสดงเป็นภาพในลักษณะของฮิสโตแกรม (histogram) จากนั้นจะให้ผู้ใช้ระบุชื่อแอททริบิวต์ที่ต้องการคัดเลือกไว้ ขั้นตอนการคัดเลือกแอททริบิวต์แสดงได้ดังอัลกอริทึมที่ 2.3

Algorithm 2.3 Feature selection

Input: a data file that contains complete Horn clauses
Output: a data file with reduced features

(1) Show the GUI of feature selection component
(2) Get the user's response to obtain the desired attribute names
(3) Scan the input data file
(3.1) Remove unwanted attribute clauses from the header file
(3.2) Remove unwanted 'attributeName=value' from every instance clauses
(3.3) Write the new clauses to the output file
(4) Return the output file

ในส่วนของการคัดเลือกรายการข้อมูลด้วยวิธีการสุ่ม (random sampling) ผู้ใช้สามารถเลือกวิธีการสุ่มได้ว่าจะใช้การสุ่มแบบมีการใส่ข้อมูลคืนกลับ (random sampling with replacement) หรือการสุ่มแบบไม่ใส่ข้อมูลคืนกลับ (random sampling without replacement) การสุ่มแบบใส่ข้อมูลคืนกลับมักจะใช้ในกรณีที่ข้อมูลที่จะถูกสุ่มมีปริมาณไม่มากนัก และการใส่ข้อมูลคืนกลับมีผลให้ข้อมูลบางรายการถูกสุ่มซ้ำได้มากกว่าหนึ่งครั้ง ส่วนการสุ่มแบบไม่ใส่ข้อมูลคืนกลับจะใช้เมื่อชุดข้อมูลที่จะถูกสุ่มมีปริมาณมาก การสุ่มทั้งสองแบบนี้เป็นการสุ่มแบบอิสระนั่นคือข้อมูลทุกตัวมีโอกาสเท่ากันที่จะถูกเลือก วิธีการสุ่มข้อมูลแสดงได้ดังอัลกอริทึมที่ 2.4

Algorithm 2.4 Data sampling

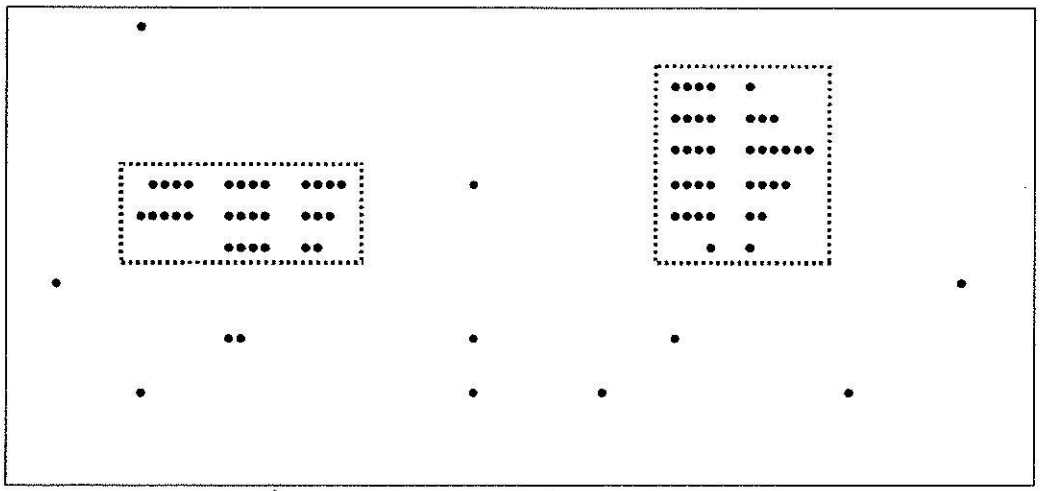
Input: a data file
Output: a data file with reduced instances

(1) Show the GUI of random sampling component
(2) Get the user's response to obtain the desired random sampling methods and the percentage of sample size
(3) Set the instance counter, I = 0
(4) Compute the number of instances to be sampled, S
(5) If choice = 'random sampling with replacement', then
(5.1) Generate random number, N, N ∈ [1..TotalInstances]
(5.2) Write the instance N to the output file
(5.3) Increment I
(5.4) If I<S, then repeat step (5)
(6) If choice = 'random sampling without replacement', then
(6.1) Generate random number, N, N ∈ [1..TotalInstances]
(6.2) If instance N does not appear in the input file,
(6.2.1 Then re-generate the random number N
(6.2.2) Otherwise write the instance N to the output file
(6.3) Delete instance N from the input file
(6.4) Increment I
(6.5) If I<S, then repeat step (6)
(7) Assert header (attribute clauses) to the output file
(8) Return the output file

2.1.3 การคัดเลือกข้อมูลตามความหนาแน่น

การคัดเลือกข้อมูลด้วยการสุ่มในแบบ random sampling จะใช้ลดขนาดข้อมูลได้ดีในกรณีที่ข้อมูลมีการการกระจายอย่างสม่ำเสมอ แต่ในกรณีที่ข้อมูลมีการกระจายไม่สม่ำเสมอ เช่นมีข้อมูลหนาแน่นในบางช่วง แต่ในช่วงอื่นๆมีข้อมูลเบาบางมาก ถ้าให้ใช้วิธี random sampling (ทั้งในแบบ with replacement และ without replacement) ที่ข้อมูลแต่ละตัวมีโอกาสถูกคัดเลือกเท่าๆกัน ข้อมูลที่ถูกสุ่มมาอาจจะไม่คงลักษณะการกระจายตัวเหมือนข้อมูลดั้งเดิมก่อนที่จะถูกสุ่ม ในงานวิจัยนี้จึงเสนอแนวทางการสุ่มข้อมูลแบบที่เรียกว่า การสุ่มตามความหนาแน่น (density-biased sampling) เพื่อประโยชน์ในการคงลักษณะการกระจายตัวของข้อมูล และจะทำให้ได้โมเดลที่สอดคล้องกับรูปแบบข้อมูล

การสุ่มข้อมูลตามความหนาแน่นมีแนวคิดที่แสดงเป็นแผนภาพได้ดังรูปที่ 2.5 ในรูปแสดงข้อมูลแต่ละรายการด้วยสัญลักษณ์ '•' จากรูปจะเห็นได้ว่าข้อมูลมีการเกาะกลุ่มเป็นสองกลุ่มใหญ่ (ดีกรอบกลุ่มข้อมูลด้วยเส้นประ) ในขณะที่ข้อมูลที่อยู่นอกกรอบกระจายตัวอย่างเบาบาง ดังนั้นถ้าต้องการสุ่มข้อมูลตัวแทนจากข้อมูลชุดนี้ ข้อมูลที่ถูกเลือกเป็นตัวแทน ควรจะเป็นข้อมูลจากสองกลุ่มใหญ่นี้ การสุ่มข้อมูลจากกลุ่มที่มีความหนาแน่นสูงจะมีประโยชน์ในการกำจัดข้อมูลที่ไม่เข้ากลุ่ม หรือ outliers และในบางครั้งสามารถกำจัดข้อมูลที่เป็นข้อมูลที่ผิดพลาด หรือ noisy data การกำจัด noisy data และ outliers จะช่วยให้ได้โมเดลที่มีความถูกต้องสูงและลดปัญหา overfitting



รูปที่ 2.5 แนวคิดของการสุ่มข้อมูลตามความหนาแน่น

วิธีการสุ่มข้อมูลตามความหนาแน่น จะให้ผู้ใช้กำหนดเกณฑ์ขั้นต่ำว่าจะพิจารณาความหนาแน่นด้วยค่าในกี่แอททริบิวต์ (M , Minimum number of attributes) และจะต้องการสุ่มข้อมูลที่มีความหนาแน่นขั้นต่ำเท่าไร (D, Density) โดยที่ D $\in$ [0..1] จากนั้นเริ่มต้นทำงานด้วยการเปิดไฟล์และอ่านข้อมูลทีละรายการเพื่อตรวจสอบข้อมูลที่อยู่ใกล้เคียง การวัดความใกล้เคียงใช้การเปรียบเทียบค่าในแต่ละแอททริบิวต์ ถ้ามีค่าคล้ายกันอย่างน้อย M แอททริบิวต์ถือว่าเป็นข้อมูลที่เกาะกลุ่มอยู่ใกล้กัน ตรวจสอบข้อมูลใกล้เคียงเช่นนี้กับข้อมูลทุกรายการ จากนั้นนับจำนวนว่ามีข้อมูลกี่รายการที่จัดว่าอยู่ใกล้เคียง คำนวณค่าข้อมูลใกล้เคียงให้เป็นค่าสัดส่วนโดยหารด้วยจำนวนข้อมูลทั้งหมดในไฟล์ ทำการตรวจสอบเช่นนี้กับข้อมูลทุกรายการ จากนั้นคัดเลือกไว้เฉพาะข้อมูลที่มีค่า D ถึงเกณฑ์ที่ระบุ แล้วสุ่มจากข้อมูลในกลุ่มนี้ให้ได้จำนวนข้อมูลตามที่ต้องการ ขั้นตอนเหล่านี้แสดงในอัลกอริทึมที่ 2.5

Algorithm 2.5 Density-biased sampling

Input: *a data file, sample size S,*
minimum number of matched attributes M, minimum density D

Output: a data file with reduced instances

(1) Show the GUI of density-biased sampling component
(2) Get the user's response to obtain the sample size S,
minimum number of matched attributes M, density threshold D
(3) Set the instance counter, I = 0
(4) Compute the number of instances to be sampled, SS

/* Compute similar instances and their density values */
(5) Open data file and read data instance
(6) For each data instance do
(6.1) Scan data file to collect instances, Ins, with matched attributes ≥ M
(6.2) Compute density, Den, as proportion of Ins to total instances in data file
(6.3) If Den ≥ D, then record this instance in temporary file F

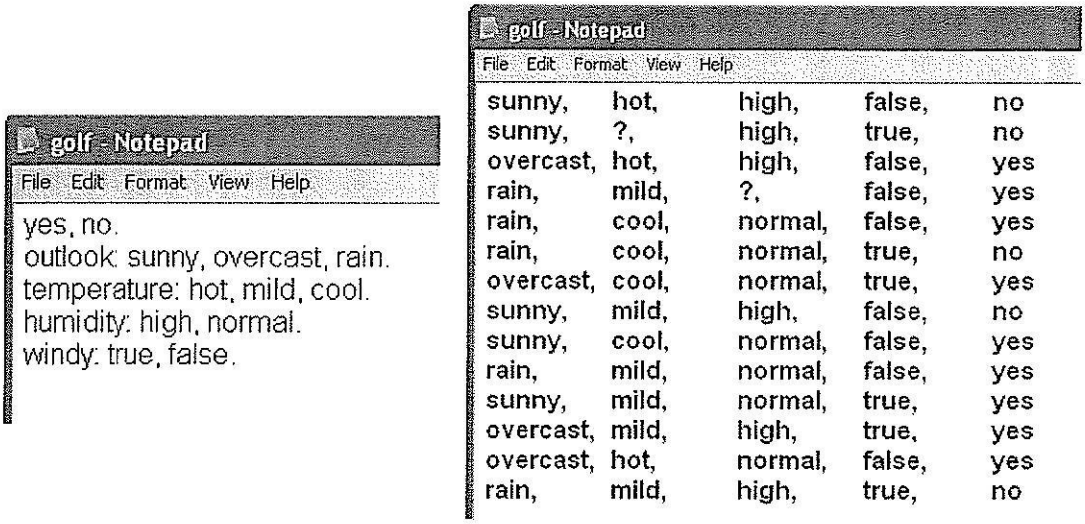
/* Sampling from the dense area */
(7) If instances in F ≤ SS, then output file is F
(8) Otherwise,
(8.1) Generate random number, N, N ∈ [1..TotalInstancesInF]
(8.2) Write the instance N to the output file
(8.3) Increment I
(8.4) If I<SS, then repeat step (8.1)
(9) Assert header (attribute clauses) to the output file
(10) Return the output file

2.2 การทดสอบโปรแกรมเตรียมข้อมูลอัตโนมัติก่อนการทำเหมืองข้อมูล

2.2.1 การทดสอบ Data transformation

การทดสอบความถูกต้องของโปรแกรมในการแปลงรูปแบบแฟ้มข้อมูล ได้ทดสอบกับข้อมูลจาก UCI repository จำนวนหลายชุดข้อมูล แต่ในรายงานนี้จะนำเสนอผลการทดสอบกับข้อมูลเพียงชุดเดียวเท่านั้นเนื่องจากผลการทดสอบเป็นเช่นเดียวกันทั้งหมด ชุดข้อมูลที่ใช้เป็นตัวอย่างคือข้อมูล golf (*Quinlan, 1993*) ที่เป็นข้อมูลการตัดสินใจของนักกอล์ฟว่าจะออกไปเล่น/ไม่ออกไปเล่นกอล์ฟ การตัดสินใจพิจารณาจากสภาพท้องฟ้า อุณหภูมิ ความชื้นในบรรยากาศ และ

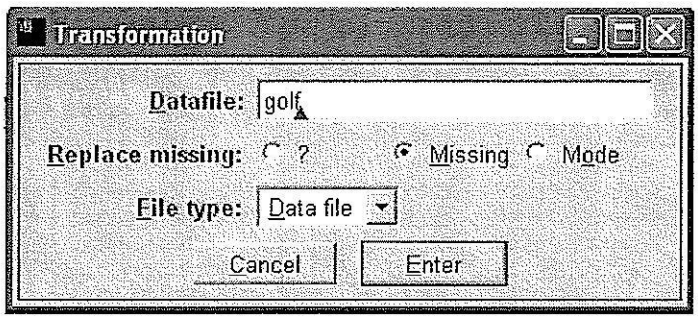
ความแรงของลม ข้อมูลที่ใช้มีจำนวน 14 รายการ เป็นข้อมูลที่เป็นข้อความ (categorical, nominal) ทั้งหมด และเพื่อให้สามารถทดสอบการจัดการกับข้อมูลสูญหายได้ กำหนดให้ข้อมูลสองรายการมีข้อมูลไม่ครบถ้วน (แทนด้วยสัญลักษณ์ '?') ได้แก่ข้อมูลรายการที่สองที่มีค่าในแอททริบิวต์ที่สองหายไป และข้อมูลรายการที่สี่ที่มีค่าในแอททริบิวต์ที่สามหายไป ข้อมูลในส่วน names file (golf.names) และ data file (golf.data) แสดงดังรูปที่ 2.6



golf.names golf.data

รูปที่ 2.6 ข้อมูลที่ใช้ทดสอบการทำงานของโปรแกรมเตรียมข้อมูล

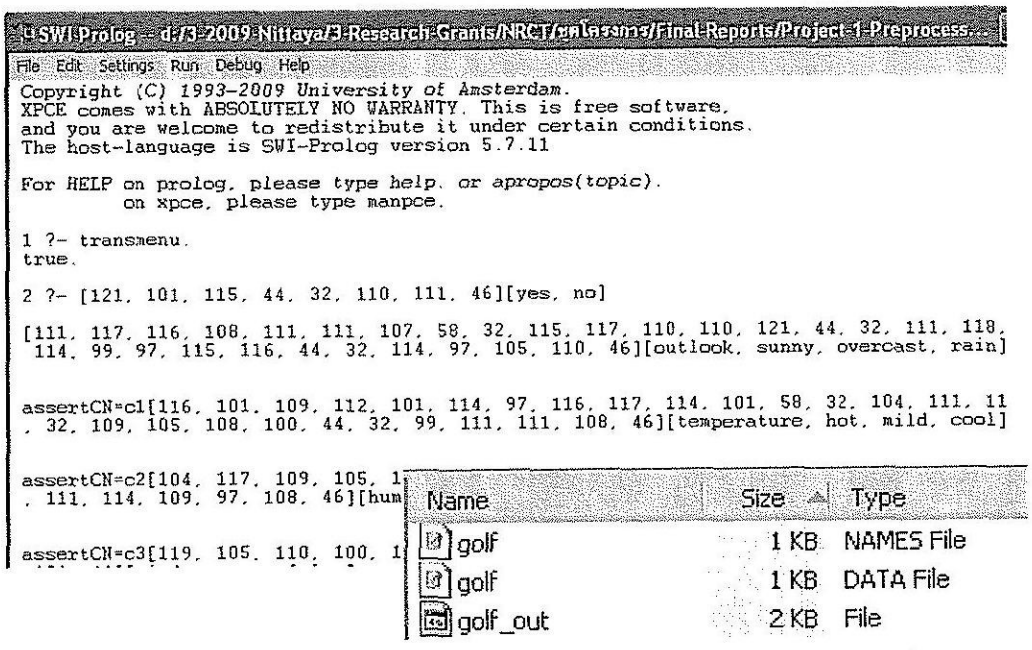
การเตรียมข้อมูลเริ่มต้นจากการเรียกใช้โปรแกรม data transformation ที่จะปรากฏหน้าจอโต้ตอบกับผู้ใช้ดังแสดงในรูปที่ 2.7 ผู้ใช้จะใส่ชื่อไฟล์โดยไม่ต้องระบุส่วนขยาย พร้อมกันนั้นจะสามารถเลือกวิธีการจัดการกับกรณีข้อมูลสูญหายได้สามรูปแบบ คือ แทนด้วย '?' หรือแทนด้วยข้อความ 'missing' หรือแทนด้วยค่าส่วนใหญ่ของข้อมูล (mode)



รูปที่ 2.7 จอภาพส่วน Data transformation

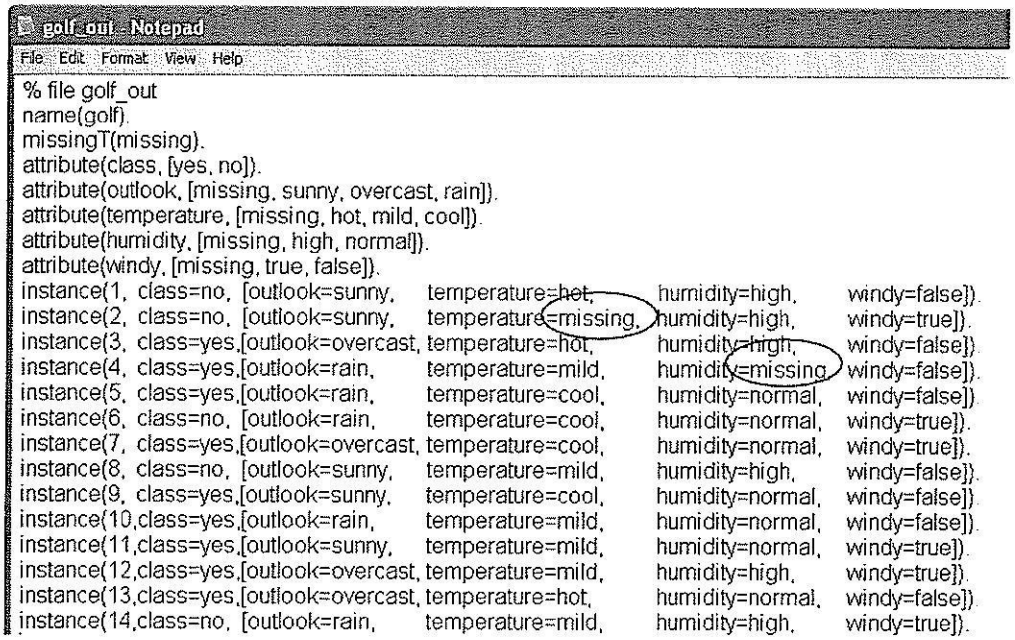
2.2.2 การทดสอบ Data cleaning

จากจอภาพเริ่มต้นดังรูปที่ 2.7 ที่ผู้ใช้เลือกที่จะแทนค่าที่หายไปด้วยข้อความ 'missing' เมื่อคลิกที่ปุ่ม Enter ในหน้าต่างของ SWI Prolog จะแสดงการทำงานในระหว่างการแปลงข้อมูลดังรูปที่ 2.8 และเมื่อการแปลงข้อมูลเสร็จสิ้น จะปรากฏไฟล์ข้อมูลชื่อ 'golf_out' ในไดเรกทอรีเดียวกับข้อมูล golf.names และ golf.data



รูปที่ 2.8 จอภาพของ SWI Prolog ขณะแปลงข้อมูล และไฟล์ golf_out ที่ได้

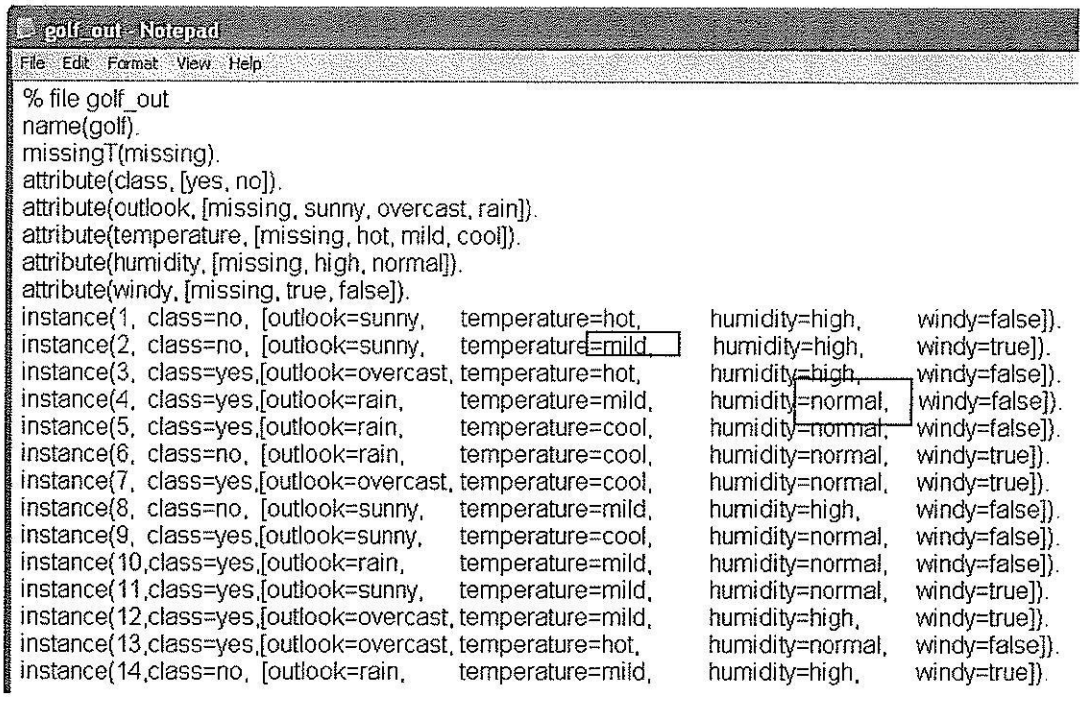
ข้อมูลในไฟล์ golf_out ที่ได้จากการรวมไฟล์ golf.names และ golf.data รวมทั้งได้มีการแทนที่ค่าสูญหายในข้อมูลรายการที่สองและรายการที่สี่ด้วยข้อความ 'missing' แสดงได้ดังรูปที่ 2.9 แต่ถ้าเปลี่ยนวิธีการจัดการกับข้อมูลสูญหายให้แทนที่ด้วยค่าส่วนใหญ่ (นั่นคือผู้ใช้เลือก mode) จะปรากฏข้อมูลใน golf_out ดังรูปที่ 2.10 ที่นำค่าส่วนใหญ่ในแอททริบิวต์นั้นมาแทนที่


golf_out - Notepad

File Edit Format View Help

```
% file golf_out
name(golf).
missingT(missing).
attribute(class, [yes, no]).
attribute(outlook, [missing, sunny, overcast, rain]).
attribute(temperature, [missing, hot, mild, cool]).
attribute(humidity, [missing, high, normal]).
attribute(windy, [missing, true, false]).
instance(1, class=no, [outlook=sunny, temperature=hot, humidity=high, windy=false]).
instance(2, class=no, [outlook=sunny, temperature=missing, humidity=high, windy=true]).
instance(3, class=yes,[outlook=overcast, temperature=hot, humidity=high, windy=false]).
instance(4, class=yes,[outlook=rain, temperature=mild, humidity=missing, windy=false]).
instance(5, class=yes,[outlook=rain, temperature=cool, humidity=normal, windy=false]).
instance(6, class=no, [outlook=rain, 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=rain, 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=rain, temperature=mild, humidity=high, windy=true]).
```

รูปที่ 2.9 ข้อมูลในไฟล์ golf_out ที่ค่าที่สูญหายถูกแทนที่ด้วยข้อความ 'missing'


golf_out - Notepad

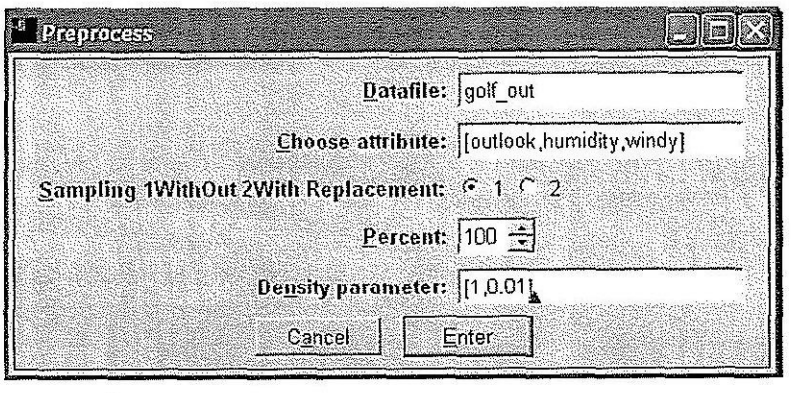
File Edit Format View Help

```
% file golf_out
name(golf).
missingT(missing).
attribute(class, [yes, no]).
attribute(outlook, [missing, sunny, overcast, rain]).
attribute(temperature, [missing, hot, mild, cool]).
attribute(humidity, [missing, high, normal]).
attribute(windy, [missing, true, false]).
instance(1, class=no, [outlook=sunny, temperature=hot, humidity=high, windy=false]).
instance(2, class=no, [outlook=sunny, temperature=mild, humidity=high, windy=true]).
instance(3, class=yes,[outlook=overcast, temperature=hot, humidity=high, windy=false]).
instance(4, class=yes,[outlook=rain, temperature=mild, humidity=normal, windy=false]).
instance(5, class=yes,[outlook=rain, temperature=cool, humidity=normal, windy=false]).
instance(6, class=no, [outlook=rain, 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=rain, 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=rain, temperature=mild, humidity=high, windy=true]).
```

รูปที่ 2.10 ข้อมูลในไฟล์ golf_out ที่ค่าที่สูญหายถูกแทนที่ด้วยค่าส่วนใหญ่

2.2.3 การทดสอบ Feature selection

เมื่อมีการเรียกใช้ฟังก์ชัน feature selection ในโมดูล preprocess ที่เป็นโมดูลสำหรับคัดเลือกข้อมูล จะปรากฏหน้าจอดังรูปที่ 2.11 ในหน้าจอโต้ตอบกับผู้ใช้จะปรากฏกรอบข้อความให้ผู้ใช้พิมพ์ชื่อไฟล์ที่มีรูปแบบข้อมูลเป็น Horn clauses ซึ่งจากตัวอย่างก่อนหน้าไฟล์ที่ใช้จะเป็นชื่อ golf_out ในบรรทัดต่อมาจะเป็นส่วนที่ให้ผู้ใช้ระบุชื่อแอททริบิวต์ที่ต้องการ รายชื่อแอททริบิวต์จะพิมพ์อยู่ภายในวงเล็บ [] ซึ่งแทนโครงสร้างลิสต์ในภาษาโปรล็อก จากภาพระบุชื่อแอททริบิวต์ [outlook, humidity, windy] โดยตัดทิ้งแอททริบิวต์ temperature ในบรรทัดที่สามและสี่เป็นการระบุวิธีสุ่มข้อมูลและปริมาณข้อมูลที่ต้องการ ถ้าผู้ใช้ต้องการทำ feature selection อย่างเดียวสามารถระบุค่าเปอร์เซนต์เป็น 100 ข้อมูลที่ได้หลังจากทำ feature selection แสดงได้ดังรูปที่ 2.12



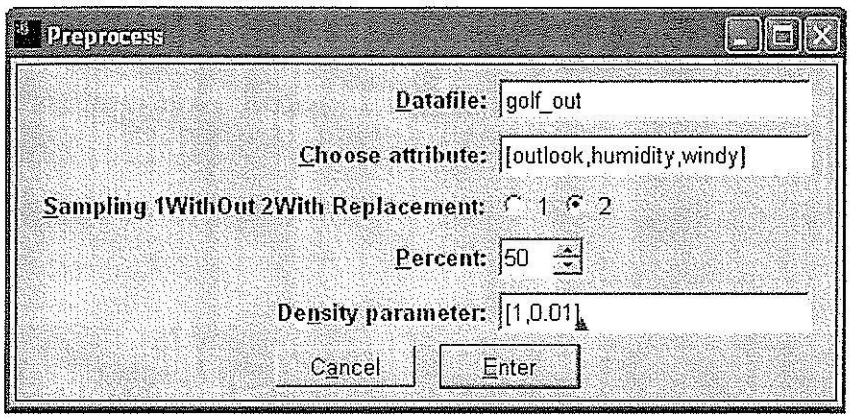
รูปที่ 2.11 จอภาพส่วน Feature selection และ Data sampling

```
File  Edit  Format  View  Help

%Density Parameter=[1,0.01] Sampling[Percent,Type]=[100,1]

%Want 14 records, but has 14 records
attribute(outlook, [missing, sunny, overcast, rain]).
attribute(humidity, [missing, high, normal]).
attribute(windy, [missing, true, false]).
attribute(class, [yes, no]).
instance(1,  class=yes,  [outlook=overcast,  humidity=high,     windy=true]).
instance(2,  class=yes,  [outlook=rain,      humidity=missing,  windy=false]).
instance(3,  class=no,   [outlook=sunny,     humidity=high,     windy=true]).
instance(4,  class=yes,  [outlook=sunny,     humidity=normal,   windy=true]).
instance(5,  class=yes,  [outlook=overcast,  humidity=normal,   windy=false]).
instance(6,  class=yes,  [outlook=overcast,  humidity=normal,   windy=true]).
instance(7,  class=no,   [outlook=sunny,     humidity=high,     windy=false]).
instance(8,  class=yes,  [outlook=rain,      humidity=normal,   windy=false]).
instance(9,  class=no,   [outlook=sunny,     humidity=high,     windy=false]).
instance(10, class=yes,  [outlook=sunny,     humidity=normal,   windy=false]).
instance(11, class=yes,  [outlook=rain,      humidity=normal,   windy=false]).
instance(12, class=yes,  [outlook=overcast,  humidity=high,     windy=false]).
instance(13, class=no,   [outlook=rain,      humidity=normal,   windy=true]).
instance(14, class=no,   [outlook=rain,      humidity=mild,     windy=true]).
```

รูปที่ 2.12 ผลลัพธ์ของ Feature selection โดยระบุแอททริบิวต์ [outlook, humidity, windy]

2.2.4 การทดสอบ Sampling with replacement

จากข้อมูลเริ่มต้นในรูปแบบ Horn clauses จำนวน 14 รายการที่มีการแทนค่าที่สูญหายในรายการที่สองและสี่ด้วยข้อความ 'missing' เมื่อนำมาทดสอบโปรแกรมสุ่มข้อมูลในรูปแบบการสุ่มแบบใส่ค่ากลับคืนด้วยเปอร์เซนต์การสุ่ม 50% ได้ผลลัพธ์เป็นข้อมูลที่ลดจำนวนลงครึ่งหนึ่ง (นั่นคือจะเหลือข้อมูลเพียง 7 รายการ) ดังรูปที่ 2.13 และเนื่องจากการสุ่มเป็นแบบที่มีการใส่ค่ากลับคืนในผลลัพธ์จึงปรากฏข้อมูลซ้ำคือข้อมูลในรายการที่ 1 และรายการที่ 5



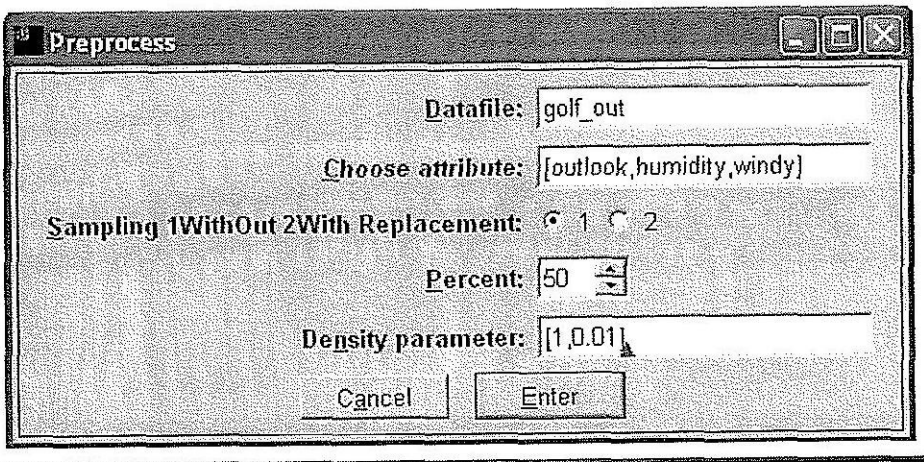
File Edit Format View Help

```
%Density Parameter=[1,0.01] Sampling[Percent,Type]=[50,2]

%Want 7 records, but has 14 records
attribute(outlook, [missing, sunny, overcast, rain]).
attribute(humidity, [missing, high, normal]).
attribute(windy, [missing, true, false]).
attribute(class, [yes, no]).
instance(7,  class=yes,  [outlook=overcast,  humidity=normal,  windy=true]).
instance(5,  class=yes,  [outlook=rain,      humidity=normal,  windy=false]).
instance(11, class=yes,  [outlook=sunny,     humidity=normal,  windy=true]).
instance(3,  class=yes,  [outlook=overcast,  humidity=high,    windy=false]).
instance(1,  class=no,   [outlook=sunny,     humidity=high,    windy=false]).
instance(1,  class=no,   [outlook=sunny,     humidity=high,    windy=false]).
instance(5,  class=yes,  [outlook=rain,      humidity=normal,  windy=false]).
```

รูปที่ 2.13 ผลลัพธ์ของการสุ่มข้อมูลแบบมีการใส่ค่ากลับคืน

2.2.5 การทดสอบ Sampling without replacement

การลดขนาดของข้อมูลด้วยการสุ่มแบบไม่ใส่ค่ากลับคืน ทดสอบกับข้อมูล golf_out และระบุขนาดของข้อมูลที่ต้องการสุ่มเป็น 50% แสดงผลทดสอบความถูกต้องของโปรแกรมได้ดังรูปที่ 2.14 จากรูปจะเห็นว่าข้อมูลที่ปรากฏในผลลัพธ์มีจำนวน 7 รายการซึ่งคิดเป็น 50% ของข้อมูลตั้งต้น และรายการข้อมูลที่ปรากฏจะไม่ซ้ำกัน



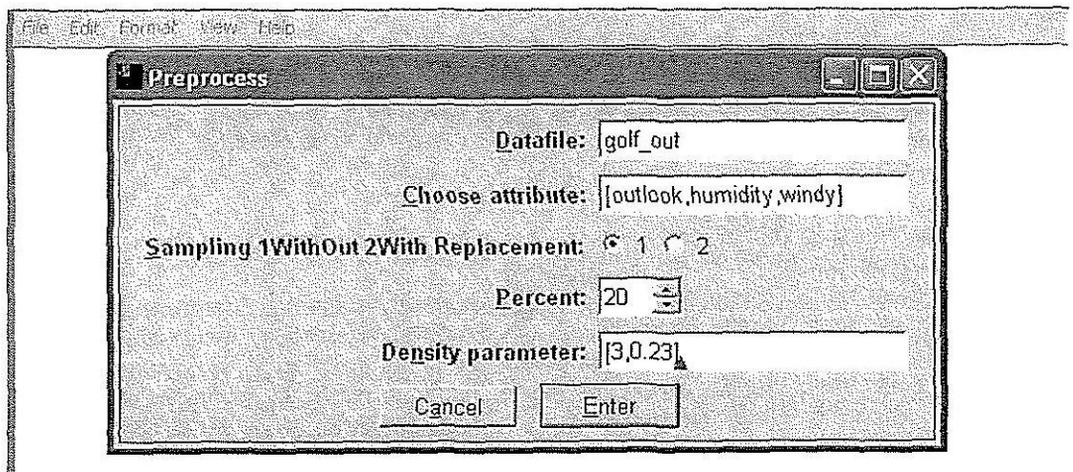
File Edit Format View Help

```
%Density Parameter=[1,0.01]  Sampling[Percent,Type]=[50,1]

%Want 7 records, but has 14 records
attribute(outlook, [missing, sunny, overcast, rain]).
attribute(humidity, [missing, high, normal]).
attribute(windy, [missing, true, false]).
attribute(class, [yes, no]).
instance(3,  class=yes,  [outlook=overcast,  humidity=high,    windy=false]).
instance(2,  class=no,   [outlook=sunny,     humidity=high,    windy=true]).
instance(5,  class=yes,  [outlook=rain,      humidity=normal,  windy=false]).
instance(11, class=yes,  [outlook=sunny,     humidity=normal,  windy=true]).
instance(1,  class=no,   [outlook=sunny,     humidity=high,    windy=false]).
instance(12, class=yes,  [outlook=overcast,  humidity=high,    windy=true]).
instance(8,  class=no,   [outlook=sunny,     humidity=high,    windy=false]).
```

รูปที่ 2.14 การสุ่มข้อมูลที่ขนาด 50% และไม่มีการใส่ค่ากลับคืน

2.2.6 การทดสอบ Density-biased sampling

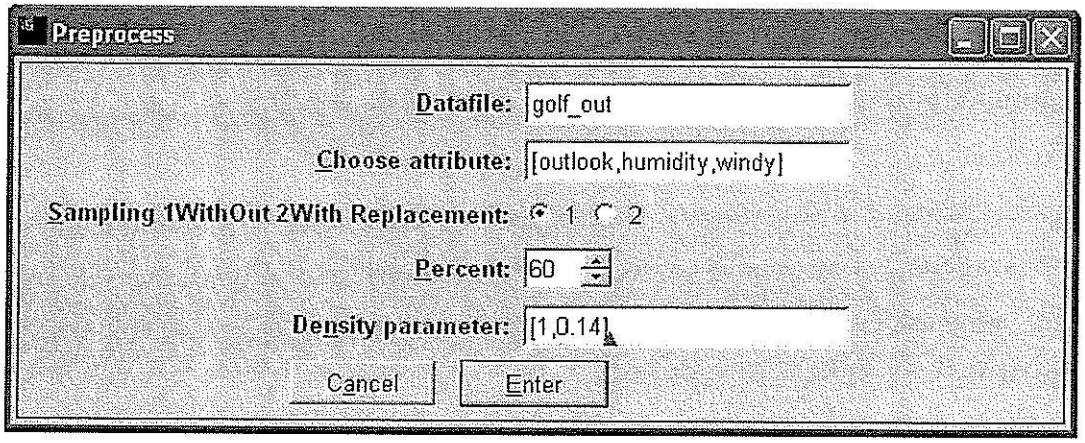
การสุ่มตามความหนาแน่นเป็นวิธีการสุ่มที่จะต้องมีเกณฑ์กำหนดสองอย่างคือ จำนวนแอททริบิวต์ที่จะใช้เปรียบเทียบข้อมูลแต่ละรายการกับข้อมูลอื่นๆ เพื่อคำนวณค่าความหนาแน่นของข้อมูลใกล้เคียง เกณฑ์ที่สองคือสัดส่วนขั้นต่ำของจำนวนข้อมูลที่อยู่ใกล้เคียง ค่าสัดส่วนนี้จะอยู่ระหว่าง [0.0-1.0] เช่นถ้าข้อมูลทั้งหมดมีจำนวน 10 รายการ สัดส่วนขั้นต่ำที่ 0.2 จะหมายถึงต้องมีจำนวนข้อมูลใกล้เคียงปรากฏอยู่อย่างน้อยสองรายการ การทดสอบความถูกต้องของโปรแกรมสุ่มข้อมูลตามความหนาแน่นตามที่ปรากฏในรูปที่ 2.15 เป็นการสุ่มข้อมูลด้วยปริมาณ 20% นั่นคือจะต้องสุ่มข้อมูลมา 3 รายการ มีการกำหนดเกณฑ์ความหนาแน่นเป็น [3, 0.23] หมายถึงพิจารณาสุ่มข้อมูลที่มีความหนาแน่นหรือมีข้อมูลอยู่ใกล้เคียงอยู่ในสัดส่วนขั้นต่ำ 0.23 หรือคิดเป็นข้อมูล 3 รายการจากทั้งหมด 14 รายการ เกณฑ์ความใกล้เคียงพิจารณาจากค่าที่เหมือนกันอย่างน้อย 3 แอททริบิวต์ จากเกณฑ์ความหนาแน่นที่กำหนดทำให้มีข้อมูลที่สุ่มได้เพียงหนึ่งรายการ จากที่ต้องการทั้งหมดสามรายการ ทั้งนี้เนื่องจากระบุเกณฑ์ความหนาแน่นสูงเกินไป




```
%Density Parameter=[3,0.23]  Sampling[Percent,Type]=[20,1]

%Want 3 records, but has 1 records
attribute(outlook, [missing, sunny, overcast, rain]).
attribute(humidity, [missing, high, normal]).
attribute(windy, [missing, true, false]).
attribute(class, [yes, no]).
instance(5,  class=yes,  [outlook=rain,  humidity=normal,  windy=false]).
```

รูปที่ 2.15 การสุ่มตามความหนาแน่นด้วยเกณฑ์ขั้นต่ำ [3, 0.23]

จากข้อมูล golf_out เมื่อเปลี่ยนลดเกณฑ์ค่าความหนาแน่นในการสุ่มข้อมูล โดยกำหนดให้พิจารณาค่าที่เหมือนกันจากจำนวนแอททริบิต์อย่างน้อยหนึ่งแอททริบิวต์ ความหนาแน่นขั้นต่ำของข้อมูลคือ 0.14 หรือคิดเป็นจำนวนข้อมูลอย่างน้อยสองรายการ ขนาดของข้อมูลสุ่มที่ต้องการคือ 60% จะได้ผลลัพธ์ดังรูปที่ 2.16



File Edit Format View Help

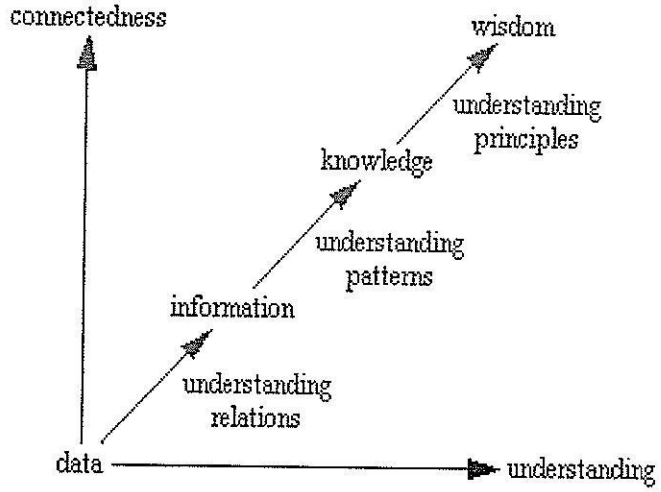
```
%Density Parameter=[1,0.14] Sampling[Percent,Type]=[60,1]

%Want 8 records, but has 14 records
attribute(outlook, [missing, sunny, overcast, rain]).
attribute(humidity, [missing, high, normal]).
attribute(windy, [missing, true, false]).
attribute(class, [yes, no]).
instance(4,   class=yes,  [outlook=rain,     humidity=missing,  windy=false]).
instance(5,   class=yes,  [outlook=rain,     humidity=normal,   windy=false]).
instance(11,  class=yes,  [outlook=sunny,    humidity=normal,   windy=true]).
instance(10,  class=yes,  [outlook=rain,     humidity=normal,   windy=false]).
instance(7,   class=yes,  [outlook=overcast, humidity=normal,   windy=true]).
instance(13,  class=yes,  [outlook=overcast, humidity=normal,   windy=false]).
instance(12,  class=yes,  [outlook=overcast, humidity=high,     windy=true]).
instance(1,   class=no,   [outlook=sunny,    humidity=high,     windy=false]).
```

รูปที่ 2.16 ผลลัพธ์ของการสุ่มตามความหนาแน่นด้วยเกณฑ์ขั้นต่ำ [1, 0.14]

2.3 สรุปผลโครงการวิจัยที่ 1

การทำเหมืองข้อมูลเป็นวิวัฒนาการของเทคโนโลยีฐานข้อมูล ที่ต้องการใช้ประโยชน์จากข้อมูลให้มากที่สุด การประมวลผลข้อมูลทำให้ทราบสารสนเทศที่ช่วยสร้างความเข้าใจเกี่ยวกับรายละเอียดข้อมูลได้มากขึ้น การทำเหมืองข้อมูลเป็นพัฒนาการของการประมวลผลอีกขั้นหนึ่งที่ช่วยให้ทราบรูปแบบหรือแพทเทิร์นของข้อมูลและสารสนเทศ การทราบรูปแบบจะช่วยให้เราคาดหมายหรือทำนายแนวโน้มการเปลี่ยนแปลงของข้อมูลที่จะเกิดขึ้นในอนาคตได้ วิวัฒนาการจากข้อมูลไปเป็นสารสนเทศและจากสารสนเทศไปเป็นความรู้แสดงได้ดังรูปที่ 2.17 (http://www.outsights .com/systems/dikw/dikw.htm)

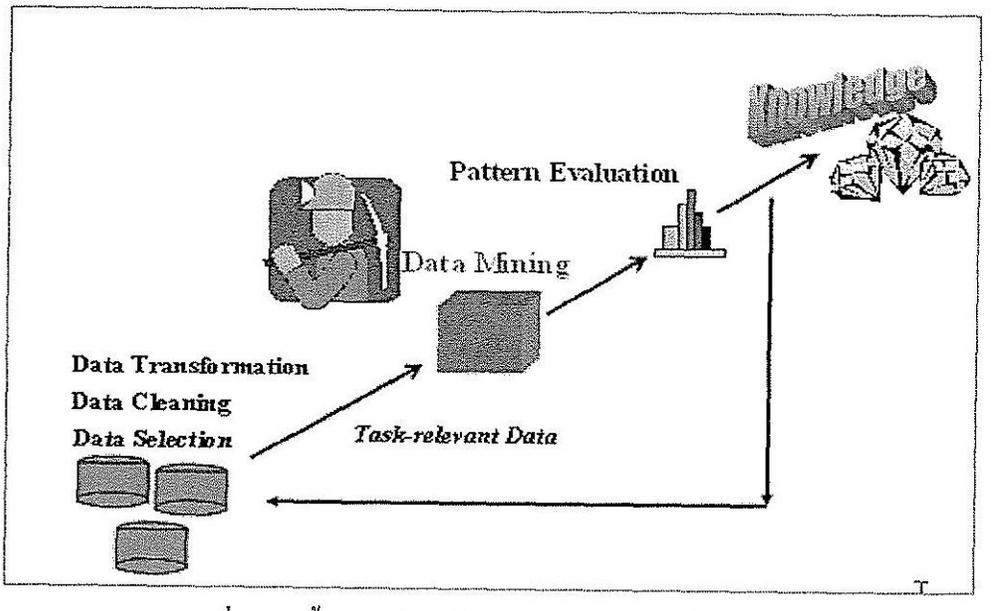
สารสนเทศ (information) เกิดจากการรวบรวมและประมวลข้อมูลเพื่อให้ข้อมูลที่อาจจะถูกเก็บกระจัดกระจาย แต่ข้อมูลเหล่านั้นมีความเกี่ยวข้องเชื่อมโยงกันจึงถูกนำมารวมเป็นกลุ่มเดียวกันและแสดงให้อยู่ในรูปแบบที่ง่ายต่อการทำความเข้าใจ ส่วนความรู้ (knowledge) เกิดจากนำข้อมูลที่มีการเชื่อมโยงนั้นมาวิเคราะห์หารูปแบบของการเชื่อมโยงหรือหาปัจจัยหลักที่ก่อให้เกิดการเชื่อมโยง เมื่อวิเคราะห์ทราบเหตุและปัจจัยแล้วประโยชน์สูงสุดของการประมวลข้อมูลคือช่วยให้เรามีความฉลาด (wisdom) เหตุแห่งความฉลาดคือการวิเคราะห์จากข้อมูลจนกระทั่งสามารถสร้างความเข้าใจในเชิงหลักการเกี่ยวกับข้อมูลนั้นได้



รูปที่ 2.17 พัฒนาการของข้อมูลที่ถูกเปลี่ยนเป็นสารสนเทศและความรู้

กิจกรรมวิเคราะห์ข้อมูลในลักษณะของการทำเหมืองข้อมูล จัดเป็นกระบวนการใหญ่ที่ประกอบด้วยขั้นตอนย่อยๆ หลายขั้นตอน (แสดงดังรูปที่ 2.18) โดยทั่วไปขั้นตอนเหล่านี้จะประกอบด้วย การรวบรวมข้อมูลที่อาจจะกระจายอยู่ในหลายแหล่ง หรือถูกบันทึกไว้ในหลายฐานข้อมูล ข้อมูลที่รวบรวมมานี้อาจจะบันทึกไว้ในคลังข้อมูลเพื่อความสะดวกในการเรียกใช้โดย

โปรแกรมทำเหมืองข้อมูล หรืออาจจะบันทึกไว้ในลักษณะของไฟล์ข้อมูลไฟล์เดียวก็ได้ โดยปกติแล้วข้อมูลเหล่านี้ยังไม่สามารถใช้ในการทำเหมืองข้อมูลได้โดยตรงเนื่องจากรูปแบบ คุณภาพ และขนาดของข้อมูลยังไม่เหมาะสม จึงต้องมีกระบวนการตรียมข้อมูลก่อนการทำเหมืองข้อมูล



รูปที่ 2.18 ขั้นตอนต่างๆ ในกระบวนการทำเหมืองข้อมูล

การเตรียมข้อมูลเป็นขั้นตอนที่สำคัญและมักจะใช้เวลามาก เนื่องจากต้องมีการเปลี่ยนรูปแบบแฟ้มข้อมูล (data transformation) มีการตรวจสอบรายการข้อมูลว่ามีความครบถ้วนสมบูรณ์ (data cleaning) เพราะถ้าข้อมูลไม่ครบถ้วน โปรแกรมทำเหมืองข้อมูลจะไม่สามารถประมวลผลได้ นอกจากนี้ยังต้องมีการคัดเลือกข้อมูล (data selection) โดยการเลือกเฉพาะแอททริบิวต์ หรือ feature ที่แสดงลักษณะเด่นของข้อมูลได้ค่อนข้างดี การคัดเลือกแอททริบิวต์ (feature selection) อย่างเหมาะสมจะช่วยให้โปรแกรมทำเหมืองข้อมูลสามารถแสดงโมเดลที่ดีมีความถูกต้องสูง และมีขนาดไม่ใหญ่จนเกินไป การคัดเลือกข้อมูลยังรวมถึงการคัดเลือกรายการข้อมูล (data instances) ที่เป็นตัวแทนที่ดีของข้อมูลกลุ่มใหญ่

การมีข้อมูลตัวแทนที่ดีจะช่วยให้การทำเหมืองข้อมูลแสดงผลลัพธ์เป็นโมเดลที่ถูกต้อง มีโครงสร้างไม่ซับซ้อนและได้ผลลัพธ์ในเวลาที่รวดเร็ว การคัดเลือกรายการข้อมูลนี้จะมีความสำคัญมากในกรณีที่ข้อมูลเป็นข้อมูลสตรีม เนื่องจากข้อมูลประเภทนี้จะเกิดขึ้นอย่างต่อเนื่องและมีปริมาณข้อมูลมาก การคัดเลือกเฉพาะข้อมูลตัวแทนมาวิเคราะห์หาโมเดลจึงมีความสำคัญต่อประสิทธิภาพการประมวลผลของโปรแกรมทำเหมืองข้อมูล