Báo Cáo

Môn Khai Phá Dữ Liệu

Đề tài: Nghiên cứu lớp bài toán luật kết hợp trong lãnh vực khai phá dữ liệu. Nghiên cứu cơ sở lý thuyết họ giải thuật Apriori. Viết chương trình Demo

GVHD: Trương Quang Hải

Nhóm thực hiện:

- Phạm Nhật Trí 51004200
- Nguyễn Bình Long 51004190
- Phạm Nguyễn Đức Dương 50900483

Mục lục Trang

Giới thiệu

Bài báo cáo này được soạn ra để tổng hợp lại những vấn đề mà nhóm chúng em đã tìm hiểu trong quá trình báo cáo hàng tuần. Về phần cấu trúc thì nhóm chúng em sẽ tái cấu trúc lại nội dung báo cáo, chúng em không soạn dựa theo nội dung báo cáo hàng tuần mà sẽ trình bày dựa theo nội dung ở dạng một khối thống nhất để vừa giúp chúng em ôn lại kiến thức một cách có hệ thống, bên cạnh đó còn giúp quá trình theo dõi về nội dung dễ dàng hơn.

Báo cáo sẽ giới thiệu về khâu tiền xử lý để làm sạch dữ liệu trước khi tiến hành khai phá, sau đó bốn giải thuật được học trong chương trình sẽ được giới thiệu, tiếp đến là giới thiệu một số ứng dụng mà khai phá dữ liệu có thể áp dụng trong thực tế, tiếp đến là giới thiệu về giải thuật Apriori và cải tiến của nó, và cuối cùng là giới thiệu về Demo giải thuật Apriori.

Trong quá trình soạn do giới hạn về thời gian và năng lực nên dù đã cố gắng hết sức thì cũng không tránh khỏi sai sót, mong thầy thông cảm.

Chương 1. Tổng Quan Về Khai Phá Dữ Liệu

Trong chương này, trình bày các khái niệm của khai phá dữ liệu, các bước của quá trình khám phá và ứng dụng của nó. Tiếp theo là thao tác đầu tiên với dữ liệu - Tiền xử lý dữ liệu. Sau đó là 4 phương pháp khai phá dữ liệu và các thuật giải của chúng.

1.1. Khai phá dữ liệu

1.1.1. Khái niệm

Khai phá dữ liệu (data mining) hay Khám phá tri thức từ dữ liệu (knowledge discovery from data) là việc trích rút ra được các mẫu hoặc tri thức quan trọng (không tầm thường, ẩn, chưa được biết đến và có thể hữu ích) thừ một lượng dữ liệu (rất) lớn.

Các tên gọi khác:

- Khám phá tri thức trong các cơ sở dữ liệu (Knowledge discovery in databases KDD).

- Trích rút tri thức (knowledge extraction).

- Phân tích mẫu/dữ liệu (data/pattern analysis)**.**

- …

1.1.2. Các bước trong quá trình khai phá

Quá trình được thực hiện qua 9 bước:

1- Tìm hiểu lĩnh vực của bài toán (ứng dụng): Các mục đích của bài toán, các tri thức cụ thể của lĩnh vực.

2- Tạo nên (thu thập) một tập dữ liệu phù hợp.

3- Làm sạch và tiền xử lý dữ liệu.

4- Giảm kích thức của dữ liệu, chuyển đổi dữ liệu: Xác định thuộc tính quan trọng, giảm số chiều (số thuộc tính), biểu diễn bất biến.

5- Lựa chọn chức năng khai phá dữ liệu: Phân loại, gom cụm, dự báo, sinh ra các luật kết hợp.

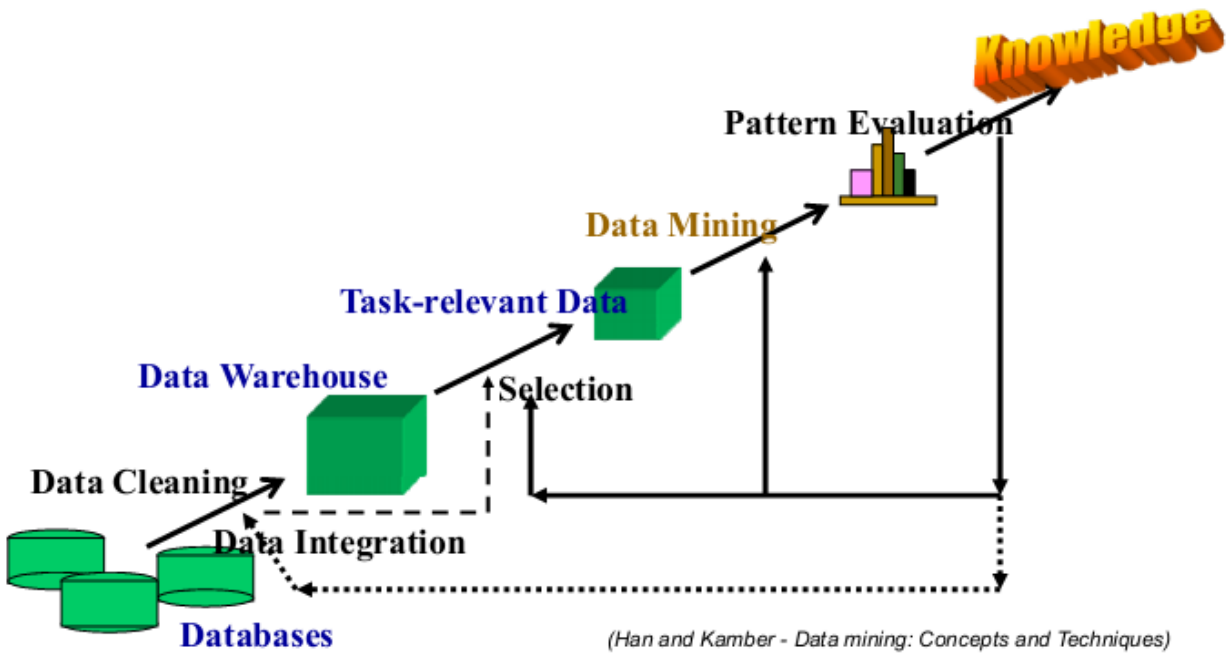
6- Lựa chọn/ Phát triển (các) giải thuật khai phá dữ liệu phù hợp.

7- Tiến hành khai phá dữ liệu.

8- Đánh giá mẫu thu được và biểu diễn tri thức: Hiển thị hóa, chuyển đổi, bỏ đi các mẫu dư thừa,…

9- Sữ dụng tri thức được khai phá.

Quá trình khám phá tri thức theo cách nhìn của giới nghiên cứu về các hệ thống dữ liệu và kho dữ liệu về quá trình khám phá tri thức.



Hình 1.1.2_Quá trình khai phá tri thức.

Chuẩn bị dữ liệu (*data preparation*), bao gồm các quá trình làm sạch dữ liệu (*data cleaning*), tích hợp dữ liệu (*data integration*), chọn dữ liệu (*data selection*), biến đổi dữ liệu (*data transformation*).

Khai thác dữ liệu (*data mining*): xác định nhiệm vụ khai thác dữ liệu và lựa chọn kỹ thuật khai thác dữ liệu. Kết quả cho ta một nguồn tri thức thô.

Đánh giá (*evaluation*): dựa trên một số tiêu chí tiến hành kiểm tra và lọc nguồn tri thức thu được.

Triển khai (*deployment*).

Quá trình khai thác tri thức không chỉ là một quá trình tuần tự từ bước đầu tiên đến bước cuối cùng mà là một quá trình lặp và có quay trở lại các bước đã qua.

1.1.3. Ứng dụng của khai phá dữ liệu

Kinh tế - ứng dụng trong kinh doanh, tài chính, tiếp thị bán hàng, bảo hiểm, thương mại, ngân hàng, … Đưa ra các bản báo cáo giàu thông tin; phân tích rủi ro trước khi đưa ra các chiến lược kinh doanh, sản xuất; phân loại khách hàng từ đó phân định thị trường, thị phần; …

Khoa học: Thiên văn học – dự đoán đường đi các thiên thể, hành tinh, …; Công nghệ sinh học – tìm ra các gen mới, cây con giống mới, …; …

Web: các công cụ tìm kiếm.

1.2. Tiền xử lý dữ liệu

Quá trình tiền xử lý dữ liệu, đầu tiên phải nắm được dạng dữ liệu, thuộc tính, mô tả của dữ liệu thao tác. Sau đó tiếp hành 4 giai đoạn chính: làm sạch, tích hợp, biến đổi, thu giảm dữ liệu.

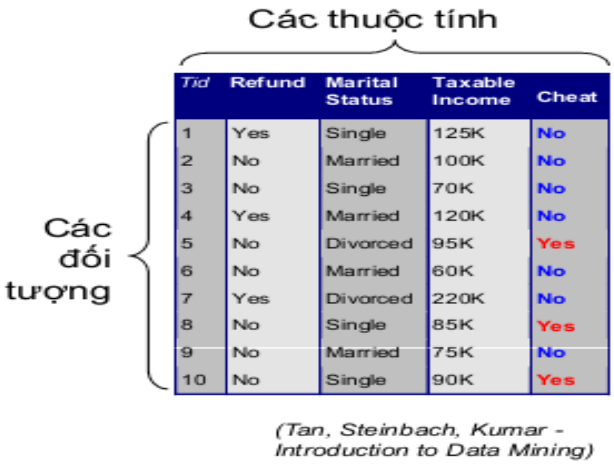
1.2.1. Dữ liệu

a) Tập dữ liệu

- Một tập dữ liệu (dataset) là một tập hợp các đối tượng (object) và các thuộc tính của chúng.

- Mỗi thuộc tính (attribute) mô tả một đặc điểm của một đối tượng.

‰



Hình 1.2.1_Ví dụ dataset

Ví dụ: Các thuộc tính Refund, Marital Status , Taxable Income, Cheat

b) Các kiểu tập dữ liệu

- Bản ghi (record): Các bản ghi trong cở sở dữ liệu quan hệ. Ma trận dữ liệu. Biểu diễn văn bản. Hay dữ liệu giao dịch.„

- Đồ thị (graph): World wide web. Mạng thông tin, hoặc mạng xã hội

- Dữ liệu có trật tự: Dữ liệu không gian (ví dụ: bản đồ). Dữ liệu thời gian (ví dụ: time-series data). Dữ liệu chuỗi (ví dụ: chuỗi giao dịch).

c) Các kiểu giá trị thuộc tính:

- Kiểu định danh/chuỗi (norminal): không có thứ tự. Ví dụ: Các thuộc tính như : Name, Profession, …

- Kiểu nhị phân (binary): là một trường hợp đăc biệt của kiểu định danh. Tập các giá trị chỉ gồm có 2 giá trị (Y/N, 0/1, T/F).

- Kiểu có thứ tự (ordinal): Integer, Real, … -lấy giá trị từ một tập có thứ tự giá trị. Ví dụ: Các thuộc tính lấy giá trị số như : Age, Height ,… Hay lấy một tập xác định, thuộc tính Income lấy giá trị từ tập {low, medium, high}.

Kiểu thuộc tính rời rạc (discrete-valued attributes): có thể là tập các giá trị của một tập hữu hạn. Bao gồm thuộc tính có kiểu giá trị là các số nguyên, nhị phân.

Kiểu thuộc tính liên tục (continuous-valued attributes):Các giá trị làsố thực.

d) Các đặc tính mô tả của dữ liệu:

- Giúp hiểu rõ về dữ liệu có được: chiều hướng chính/trung tâm, sự biến thiên, sự phân bố.

- Sự phân bố của dữ liệu (data dispersion):

+ Giá trị cực tiểu/cực đại (min/max).

+ Giá trị xuất hiện nhiều nhất (mode).

+ Giá trị trung bình (mean).

+ Giá trị trung vị (median).

+ Sự biến thiên (variance) và độ lệch chuẩn (standard deviation) .

+ Các ngoại lai (outliers).

1.2.2. Làm sạch dữ liệu (data cleaning)

Đối với dữ liệu thu thập được, cần xác định các vấn đề ảnh hưởng là cho nó không sạch. Bởi vì, dữ liệu không sạch (có chứa lỗi, nhiễu, không đầy đủ, có mâu thuẫn) thì các tri thức khám phá được sẽ bị ảnh hưởng và không đáng tin cậy, sẽ dẫn đến các quyết định không chính xác. Do đó, cần gán các giá trị thuộc tính còn thiếu; sửa chữa các dữ liệu nhiễu/lỗi; xác định hoặc loại bỏ các ngoại lai (outliers); giải quyết các mâu thuẫn dữ liệu.

a) Các vấn đề của dữ liệu

Trên thực thế dữ liệu thu có thể chứa nhiễu, lỗi, không hoàn chỉnh, có mâu thuẫn.

- Không hoàn chỉnh (incomplete): Thiếu các giá trị thuộc tính hoặc thiếu một số thuộc tính. Ví dụ: salary = <undefined>.

- Nhiễu/lỗi (noise/error): Chứa đựng những lỗi hoặc các mang các giá trị bất thường. Ví dụ: salary = "-525" , giá trị của thuộc tính không thể là một số âm.

- Mâu thuẫn (inconsistent): Chứa đựng các mâu thuẫn (không thống nhất). Ví dụ: salary = "abc" , không phù hợp với kiểu dữ liệu số của thuộc tính salary.

b) Nguồn gốc/lý do của dữ liệu không sạch

- Không hoàn chỉnh (incomplete): Do giá trị thuộc tính không có (not available) tại thời điểm được thu thập. Hoặc các vấn gây ra bởi phần cứng, phần mềm, hoặc người thu thập dữ liệu.

- Nhiễu/lỗi (noise/error): Do việc thu thập dữ liệu, hoăc việc nhập dữ liệu, hoặc việc truyền dữ liệu.

- Mâu thuẫn (inconsistent): Do dữ liệu được thu thập có nguồn gốc khác nhau. Hoặc vi phạm các ràng buộc (điều kiện) đối với các thuộc tính.

c) Giải pháp khi thiếu giá trị của thuộc tính

- Bỏ qua các bản ghi có các thuộc tính thiếu giá trị. Thường áp dụng trong các bài toán phân lớp. Hoặc khi tỷ lệ % các giá trị thiếu đối với các thuộc tính quá lớn.

- Một số người sẽ đảm nhiệm việc kiểm tra và gán các giá trị thuộc tính còn thiếu, nhưng đòi hỏi chi phí cao và rất tẻ nhạt.

- Gán giá trị tự động bởi máy tính:

+ Gán giá trị mặc định

+ Gán giá trị trung bình của thuộc tính đó.

+ Gán giá trị có thể xảy ra nhất – dựa theo phương pháp xác suất.

d) Giải pháp khi dữ liệu chứa nhiễu/lỗi

- Phân khoảng (binning): Sắp xếp dữ liệu và phân chia thành các khoảng (bins) có tần số xuất hiện giá trị như nhau. Sau đó, mỗi khoảng dữ liệu có thể được biểu diễn bằng trung bình, trung vị, hoặc các giới hạn … của các giá trị trong khoảng đó.

- Hồi quy (regression): Gắn dữ liệu với một hàm hồi quy.

- Phân cụm (clustering): Phát hiện và loại bỏ các ngoại lai (sau khi đã xác định các cụm).

- Kết hợp giữa máy tính và kiểm tra của con người: Máy tính sẽ tự động phát hiện ra các giá trị nghi ngờ. Các giá trị này sẽ được con người kiểm tra lại.

1.2.3. Tích hợp dữ liệu (data integration)

Tích hợp dữ liệu là quá trình trộn dữ liệu từ các nguồn khác nhau vào một kho dữ liệu có sẵn cho quá trình khai phá dữ liệu.

Khi tích hợp cần xác định thực thể từ nhiều nguồn dữ liệu để tránh dư thừa dữ liệu. Ví dụ: Bill Clinton ≡ B.Clinton.

Việc dư thừa dữ liệu là thường xuyên xảy ra, khi tích hợp nhiều nguồn. Bởi cùng một thuộc tính (hay cùng một đối tượng) có thể mang các tên khác nhau trong các nguồn (cơ sở dữ liệu) khác nhau. Hay các dữ liệu suy ra được như một thuộc tính trong một bảng có thể được suy ra từ các thuộc tính trong bảng khác. Hay sự trùng lắp các dữ liệu. Các thuộc tính dư thừa có thể bị phát hiện bằng phân tích tương quan giữa chúng.

Phát hiện và xử lý các mâu thuẫn đối với giá trị dữ liệu: Đối với cùng một thực thể trên thực tế, nhưng các giá trị thuộc tính từ nhiều nguồn khác nhau lại khác nhau. Có thể cách biểu diễn khác nhau, hay mức đánh giá, độ do khác nhau.

Yêu cầu chung đối với quá trình tích hợp là giảm thiểu (tránh được là tốt nhất) các dư thừa và các mâu thuẫn. Giúp cải thiện tốc độ của quá trình khai phá dữ liệu và nâng cao chất lượng của các kết quả tri thức thu được.

1.2.4. Biến đổi dữ liệu (data transformation)

Biến đổi dữ liệu là việc chuyển toàn bộ tập giá trị của một thuộc tính sang một tập các giá trị thay thế, sao cho mỗi giá trị cũ tương ứng với một trong các giá trị mới.

Các phương pháp biến đổi dữ liệu:

- Làm trơn (smoothing): Loại bỏ nhiễu/lỗi khỏi dữ liệu.
- Kết hợp (aggregation): Sự tóm tắt dữ liệu, xây dựng các khối dữ liệu.
- Khái quát hóa (generalization): Xây dựng các phân cấp khái niệm.
- Chuẩn hóa (normalization): Đưa các giá trị về một khoảng được chỉ định.

+ Chuẩn hóa min-max, giá trị mới nằm khoảng [new_min_i , new_max_i]

$$v^{new} = \frac{v^{old} - min_i}{max_i - min_i}(new\_max_i - new\_min_i) + new\_min_i$$

+ Chuẩn hóa z-score, với μ_i , σ_i : giá trị trung bình và độ lệch chuẩn của thuộc tính *i*

$$v^{new} = \frac{v^{old} - \mu_i}{\sigma_i}$$

+ Chuẩn hóa bởi thang chia 10, với j là giá trị số nguyên nhỏ nhất sao cho: $\max(\{v^{new}\}) < 1$

$$v^{new} = \frac{v^{old}}{10^j}$$

- Xây dựng các thuộc tính mới dựa trên các thuộc tính ban đầu.

1.2.5. Thu giảm dữ liệu (data reduction)

Một kho dữ liệu lớn có thể chứa lượng dữ liệu lên đến terabytes sẽ làm cho quá trình khai phá dữ liệu chạy rất mất thời gian, do đó nên thu giảm dữ liệu.

Việc thu giảm dữ liệu sẽ thu được một biểu diễn thu gọn, mà nó vẫn sinh ra cùng (hoặc xấp xỉ) các kết quả khai phá như tập dữ liệu ban đầu.

Các chiến lược thu giảm:

- Giảm số chiều (dimensionality reduction), loại bỏ bớt các thuộc tính không (ít) quan trọng.

- Giảm lượng dữ liệu (data/numberosity reduction)

+ Kết hợp khối dữ liệu.

+ Nén dữ liệu.

+ Hồi quy.

+ Rời rạc hóa.

1.3. Phương pháp Dự báo

1.3.1. Giới thiệu Dự báo

Bài toán dự báo dùng để dựa vào thông tin liên quan đến người mua hàng (thu nhập, trình độ…..), hay các mặt hàng mà khách hàng đã mua,…… để tiến hành đưa ra dự báo về những lựa chọn mà có khả năng cao sẽ xảy ra tiếp theo.

Vd: một khách hàng mua một chiếc máy tính xách tay, thì người bán hàng sẽ gợi ý về một số phiên bản hệ điều hành, phần mềm diệt virus, ứng dụng văn phòng….để cho khách hàng xem xét.

Cả 4 phương pháp được học trong chương trình (hồi qui, phân loại, gom cụm, khai phá luật kết hợp) để có thể dùng để dự báo được. Ở trong mục này chỉ giới thiệu về *hồi qui* dữ liệu, ba giải thuật còn lại (phân loại, gom cụm, khai phá luật kết hợp) sẽ được giới thiệu ở các mục sau.

Kỹ thuật dự báo khi dùng với hồi qui được dùng để dự báo các giá trị (số) *liên tục*. Còn 3 kỹ thuật còn lại được dùng để dự báo các giá trị (số) *rời rạc*.

1.3.2. Tổng quan Hồi qui

a) Khái niệm

* Hồi qui là kỹ thuật thống kê cho phép dự đoán các trị (số) liên tục. *J.Han et al(2001, 2006).*

* Hồi qui (Phân tích hồi quy – regression analysis) là kỹ thuật thống kê cho phép ước lượng các mối liên kết giữa các biến. *Wiki(2009)*

* Hồi qui (Phân tích hồi quy) là kỹ thuật thống kê trong lĩnh vực phân tích dữ liệu và xây dựng các mô hình từ thực nghiệm, cho phép mô hình hồi qui vừa được khám phá được dùng cho mục đích dự báo (prediction), điều khiển (control), hay học (learn) cơ chế đã tạo ra dữ liệu. *R.D.Snee(1977)*

b) Mô hình Hồi qui (regression model):

Mô hình mô tả mối liên kết (relationship) giữa một tập các biến dự báo (predictor variables/independent variables) và một hay nhiều đáp ứng (responses/dependent variables).

c) Phân loại:

- Hồi qui tuyến tính (linear) và phi tuyến (nonlinear).

- Hồi qui đơn biến (single) và đa biến (multiple).

- Hồi qui có thông số (parametric), phi thông số (nonparametric), và thông số kết hợp (semiparametric).

- Hồi qui đối xứng (symmetric) và bất đối xứng (asymmetric).

1.3.3. Hồi qui tuyến tính

Hồi qui tuyến tính gồm hồi qui tuyến tính đơn biến và hồi qui tuyến tính đa biến.

* Giới thiệu về hồi qui tuyến tính đơn biến

Dạng tổng quát: $y = w_0 + w_1 x$

Trong đó x là biến đoán trước (predictor variable), y là giá trị được đoán ra với giá trị x tương ứng (response variable). Để xác định giá trị w_0 và w_1, ta sử dụng

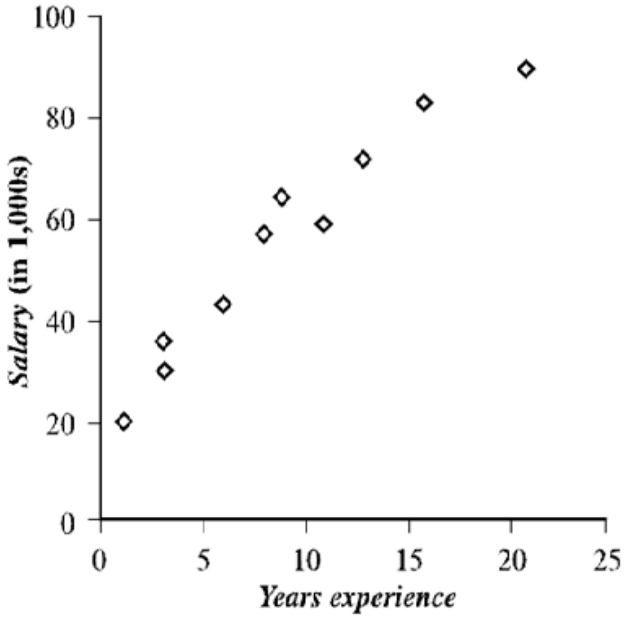
công thức bình phương tối thiểu để được một đường thẳng thích hợp nhất. Cách tính w_0 và w_1 như sau:

$$w_1 = \frac{\sum_{i=1}^{|D|}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{|D|}(x_i - \bar{x})^2} \qquad w_0 = \bar{y} - w_1\bar{x}$$

Ví dụ: Hồi qui tuyến tính đơn biến với data <salary>

Salary data.

x years experience	*y salary (in $1000s)*
3	30
8	57
9	64
13	72
3	36
6	43
11	59
21	90
1	20
16	83



Hình 1.3.2_Ví dụ hôi qui tuyến tính đơn biến

1.3.4. Hồi qui phi tuyến

Dạng tổng quát: $Y_i = b_0 + b_1X_{i1} + b_2X_{i2} + \ldots + b_kX_{ik}$

Trong đó: i = 1..n với n là số đối tượng đã quan sát
k = số biến độc lập (số thuộc tính/tiêu chí/yếu tố…)
Y = biến phụ thuộc
X = biến độc lập
$b_{0\ldots k}$ = trị của các hệ số hồi qui

1.4. Phương pháp Phân loai

1.4.1. Giới thiệu Phân loại

Phân loại dữ liệu là dạng phân tích dữ liệu nhằm rút trích các mô hình mô tả các lớp dữ liệu hoặc dự đoán xu hướng dữ liệu.

Quá trình gồm hai bước:

- Bước học (giai đoạn huấn luyện): xây dựng bộ phân loại (classifier) bằng việc phân tích/học tập huấn luyện.

- Bước phân loại (classification): phân loại dữ liệu/đối tượng mới nếu độ chính xác của bộ phân loại được đánh giá là có thể chấp nhận được (acceptable).

Các giải thuật phân loại dữ liệu:

- Phân loại dữ liệu với cây quyết định (decision tree).

- Phân loại dữ liệu với mạng Bayesian.

- Phân loại dữ liệu với mạng neural.

- Phân loại dữ liệu với k phần tử gần nhất (k-nearest neighbor).

- Phân loại dữ liệu với suy diễn dữa trên tình huống (case-based reasoning).

- Phân loại dữ liệu dựa trên tiến hóa gen (genetic algorithms).

- Phân loại dữ liệu với lý thuyết tập thô (rough sets).

- Phân loại dữ liệu với lý thuyết tập mờ (fuzzy sets).

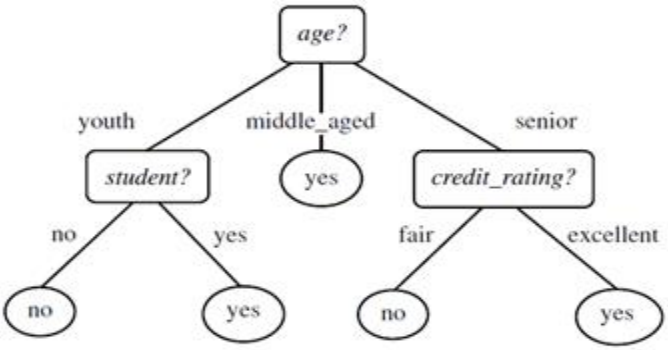
1.4.2. Phân loại dữ liệu với cây quyết định

Cây quyết định (decision tree) là một mô hình dùng để phân loại dữ liệu gồm có:

- Node nội: chứa giá trị trên một thuộc tính để cho quá trình thực hiện phép kiểm thử.

- Node lá: chứa nhãn (label) hoặc mô tả của một lớp (class label).

- Nhánh từ một node nội: kết quả của một phép thử trên thuộc tính tương ứng.



A decision tree for the concept *buys_computer*, indicating whether a customer at *AllElectronics* is likely to purchase a computer. Each internal (nonleaf) node represents a test on an attribute. Each leaf node represents a class (either *buys_computer = yes* or *buys_computer = no*).

Hình 1.4.2_Một ví dụ về cây quyết định

Giới thiệu một số độ đo:

- Information Gain (được dùng trong ID3)

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

Trong đó: *Info(D)*: Lượng thông tin cần để phân loại một phần tử D.

P_i: xác suất để một phần tử bất kỳ trong D thuộc về lớp C_i , với i = 1..m.

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

$$Gain(A) = Info(D) - Info_A(D)$$

- Gain Ratio (được dùng trong C4.5)

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$$

- Gini Index (được dùng trong CART)

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2$$

$$Gini_A(D) = \frac{|D_1|}{|D|}Gini(D_1) + \frac{|D_2|}{|D|}Gini(D_2)$$

$$\Delta Gini(A) = Gini(D) - Gini_A(D).$$

Giải thuật xây dựng cây quyết định: Một số giải thuật xây dựng cây quyết định nhu ID3, C4.5, CART (Classification and Regression Trees). Giải thuật tổng quát xây dựng cây quyết định từ Training Data

Input:

- Data partition, *D*, which is a set of training tuples and their associated class labels;
- *attribute_list*, the set of candidate attributes;
- *Attribute_selection_method*, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting_attribute* and, possibly, either a *split point* or *splitting subset*.

Output: A decision tree.

Method:

(1) create a node *N*;
(2) **if** tuples in *D* are all of the same class, *C* **then**
(3) return *N* as a leaf node labeled with the class *C*;
(4) **if** *attribute_list* is empty **then**
(5) return *N* as a leaf node labeled with the majority class in *D*; // majority voting
(6) apply **Attribute_selection_method**(*D*, *attribute_list*) to **find** the "best" *splitting_criterion*;
(7) label node *N* with *splitting_criterion*;
(8) **if** *splitting_attribute* is discrete-valued **and**
multiway splits allowed **then** // not restricted to binary trees
(9) *attribute_list* ← *attribute_list* − *splitting_attribute*; // remove *splitting_attribute*
(10) **for each** outcome *j* of *splitting_criterion*
// partition the tuples and grow subtrees for each partition
(11) let D_j be the set of data tuples in *D* satisfying outcome *j*; // a partition
(12) **if** D_j is empty **then**
(13) attach a leaf labeled with the majority class in *D* to node *N*;
(14) **else** attach the node returned by **Generate_decision_tree**(D_j, *attribute_list*) to node *N*;
endfor
(15) return *N*;

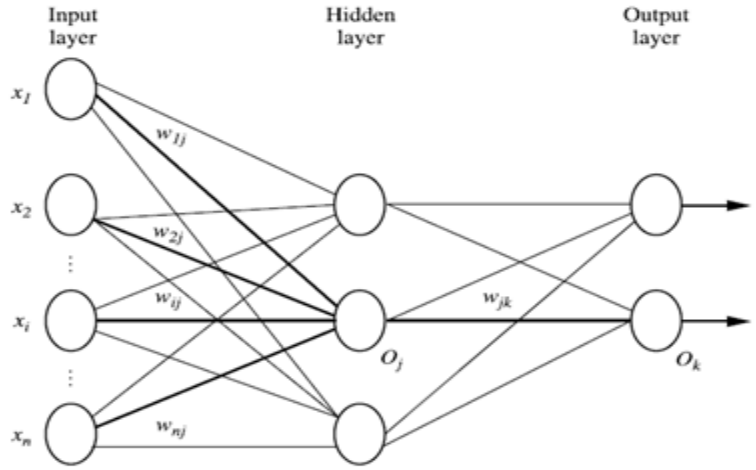
1.4.3. Phân loại dữ liệu với mạng Bayesian

Phân loại dữ liệu với mạng Bayes là việc sử dụng phân loại dựa trên xác suất có điều kiện do Bayes tìm ra. Công thức xác suất có điều kiện có dạng:

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)}$$

1.4.4. Phân loại dữ liệu với mạng Neural

Được mô phỏng dựa theo mạng Neural trong não bộ. Được xây dựng bằng cách lập lại việc học một tập hợp có trọng số các dự đoán về một lớp các nhãn dựa vào trọng số. Thường được hiện thực bằng giải thuật backpropagation. Gồm có input layer, một hoặc nhiều layers ẩn, và output layer. Dữ liệu được đưa vào input layer, dựa vào trọng số để di chuyển đến các neural thích hợp trong hidden layer và cuối cùng là ra output layer để trả về kết quả.



Hình 1.4.4_ Minh họa cho dạng tổng quát của mạng Neural

Giải thuật lan truyền ngược (backpropagation)

Algorithm: Backpropagation. Neural network learning for classification or prediction, using the backpropagation algorithm.

Input:

- D, a data set consisting of the training tuples and their associated target values;
- l, the learning rate;
- *network*, a multilayer feed-forward network.

Output: A trained neural network.

Method:

(1) Initialize all weights and biases in *network*;
(2) while terminating condition is not satisfied {
(3) for each training tuple X in D {
(4) // Propagate the inputs forward:
(5) for each input layer unit j {
(6) $O_j = I_j$; // output of an input unit is its actual input value
(7) for each hidden or output layer unit j {
(8) $I_j = \sum_i w_{ij} O_i + \theta_j$; //compute the net input of unit j with respect to the previous layer, i
(9) $O_j = \frac{1}{1+e^{-I_j}}$; } // compute the output of each unit j
(10) // Backpropagate the errors:
(11) for each unit j in the output layer
(12) $Err_j = O_j(1 - O_j)(T_j - O_j)$; // compute the error
(13) for each unit j in the hidden layers, from the last to the first hidden layer
(14) $Err_j = O_j(1 - O_j) \sum_k Err_k w_{jk}$; // compute the error with respect to the next higher layer, k
(15) for each weight w_{ij} in *network* {
(16) $\Delta w_{ij} = (l) Err_j O_i$; // weight increment
(17) $w_{ij} = w_{ij} + \Delta w_{ij}$; } // weight update
(18) for each bias θ_j in *network* {
(19) $\Delta \theta_j = (l) Err_j$; // bias increment
(20) $\theta_j = \theta_j + \Delta \theta_j$; } // bias update
(21) } }

1.5. Phương pháp Gom cụm

1.5.1. Giới thiệu Gom cụm

Gom cụm dữ liệu: Việc nhóm một tập các đối tượng có cùng đặc điểm giống nhau hay gần giống nhau vào cùng một nhóm.

Các đối tượng trong cùng một cụm tương tự với nhau hơn so với đối tượng ở cụm khác.

Phương pháp gom cụm hỗ trợ giai đoạn tiền xử lý dữ liệu, mô tả sự phân bố dữ liệu/đối tượng, …

Các phương pháp gom cụm tiêu biểu:

- Phân hoạch (partitioning): các phân hoạch được tạo ra và đánh giá theo một tiêu chí nào đó.

- Phân cấp (hierarchical): phân rã tập dữ liệu/đối tượng có thứ tự phân cấp theo một tiêu chí nào đó.

- Dựa trên mật độ (density-based): dựa trên connectivity and density functions.

- Dựa trên lưới (grid-based): dựa trên a multiple-level granularity structure.

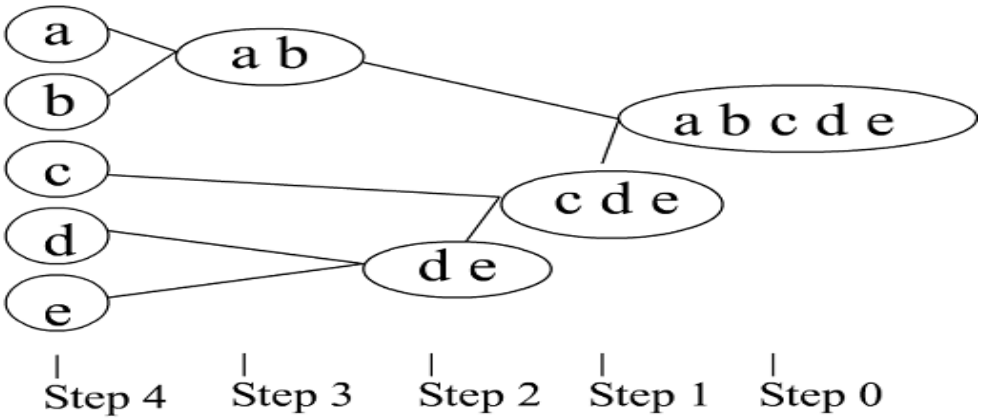
- Dựa trên mô hình (model-based): một mô hình giả thuyết được đưa ra cho mỗi cụm; sau đó hiệu chỉnh các thông số để mô hình phù hợp với cụm dữ liệu/đối tượng nhất.

- …

1.5.2. Phương pháp phân cấp

Cây các cụm: dùng biểu diễn phân cấp cụm . Với các lá của cây biểu diễn từng đối tượng và các nút trung gian và gốc biểu diễn các cụm.

Tạo cây phân cấp từ trên xuống: Từ cụm lớn nhất chứa tất cả đối tượng. Chia thành cụm nhỏ hơn, đến khi có n cụm thỏa mãn điều kiện dừng.



Hình 1.5.3_Tạo cây phân cấp từ trên xuống

Tạo cây phân cấp từ dưới lên:

- Tạo n nhóm, mỗi nhóm gồm một đối tượng và lập một ma trận khoảng cách cấp n.

- Tìm 2 nhóm u, v có khoảng cách nhỏ nhất.

- Gộp 2 nhóm u,v thành nhóm uv và lập ma trận khoảng cách mới cho uv

- Lặp lại quá trình đến khi còn 1 nhóm

1.5.3. Phương pháp phân hoạch

Với tập dữ liệu chứa n đối tượng, tạo phân hoạch thành tập có k cụm sao cho:

- Mỗi cụm có ít nhất 1 đối tượng.
- Mỗi đối tượng thuộc về 1 cụm duy nhất.
- Tìm phân hoạch có k cụm sao tối ưu hóa các tiêu chuẩn phân hoạc được chọn.

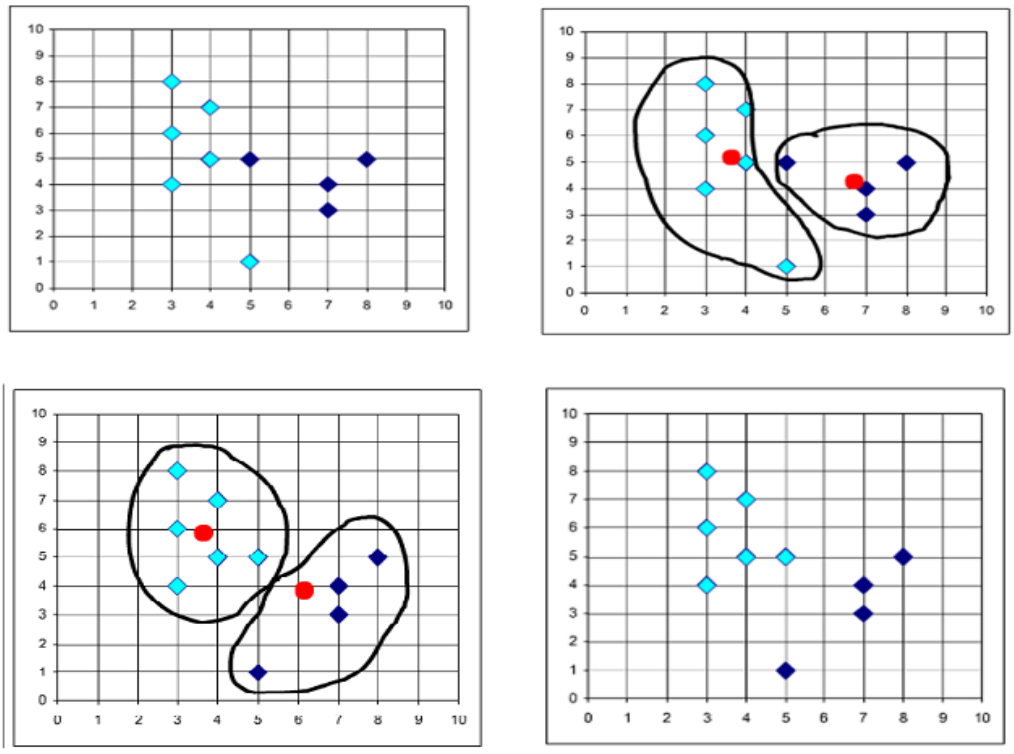
Thuật toán k-mean:

1- Phân hoạch đối tượng thành k cụm ngẫu nhiên.

2- Tính các tâm cho từng cụm trong phân hoạch hiện hành.

3- Gán mỗi đối tượng cho cụm tâm gần nhất

4- Nếu cụm không có sự thay đổi thì dừng lại, ngược lại quay lại bước 2



Hình 1.5.3_1.Giải thuật toán k-mean: với n = 10, k = 2

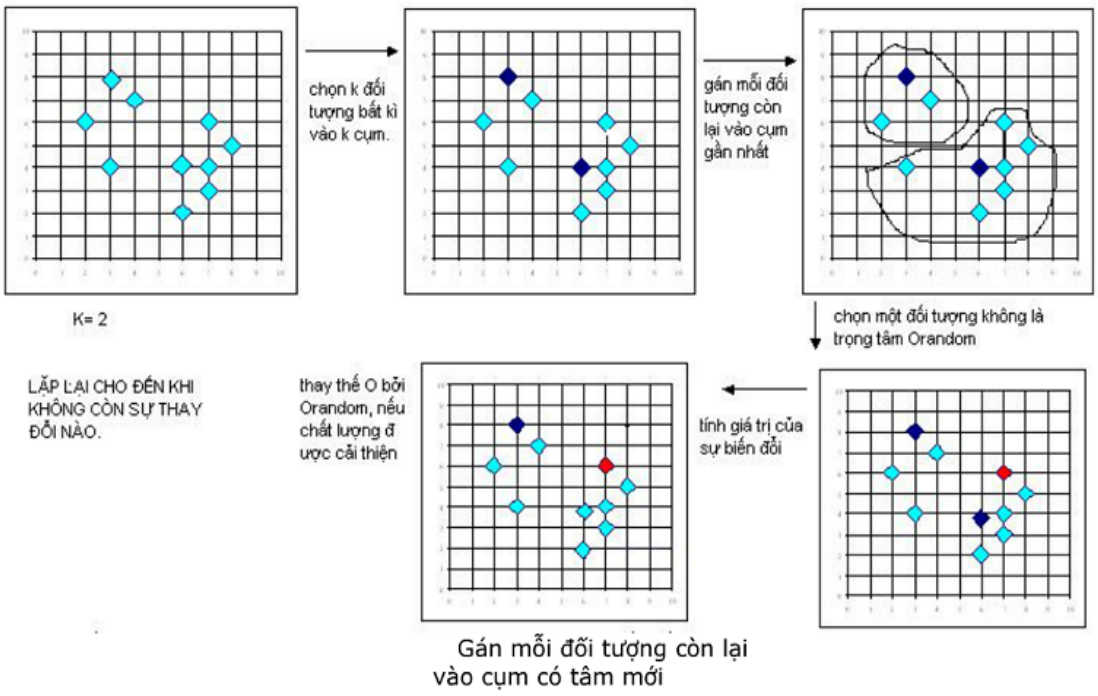
Thuật toán k-medold:

1- Chọn k đối tượng ngẫu nhiên làm tâm của nhóm.

2- Gán từng đối tượng còn lại vào cụm có tâm gần nhất.

3- Chọn ngẫu nhiên 1 đối tượng không là tâm, thay một trong các tâm là nó; nếu nó làm thay đổi các đối tượng trong cụm.

4- Nếu gán tâm mới thì quay lại bước 2, ngược lại thì dừng.



*Hình 1.5.3_2.*Giải thuật toán k-medold: với *n = 10, k = 2*

1.6. Phương pháp khai phá luật kết hợp

1.6.1. Giới thiệu luật kết hợp

Bài toán phát hiện luật kết hợp (association rule mining): với một tập hợp các giao dịch cho trước, cần tìm các luật dự đoán khả năng xuất hiện trong một giao dịch cảu các mục (items) này dựa trên việc xuất hiện của các mục khác.

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Các ví dụ của luật kết hợp:

{Diaper} → {Beer}

{Milk, Bread} → {Eggs, Coke}

{Beer, Bread} → {Milk}

Các định nghĩa cơ bản:

- Tập mục (itemset): là một tập hợp gồm một hoặc nhiều mục. Tâp mục mức k (k-itemset) có k mục. Ví dụ: 3-itemset là {Milk, Bread, Diaper}.

- Luật kết hợp – kí hiệu X -> Y, trong đó X, Y là các tập mục.

- Tổng số hỗ trợ (support count)- kí hiệu σ : là số lần xuất hiện của một tập mục. Ví dụ: σ({Milk, Bread, Diaper}) = 2.

- Độ hỗ trợ (support)- kí hiệu s: là tỷ lệ các giao dịch chứa cả X và Y đối với tất cả các giao dịch. Ví dụ: s({Milk, Diaper, Beer}) = 2/5.

- Độ tin cậy (confidence) – kí hiệu c: là tỷ lệ các giao dịch chứa cả X và Y đối với các giao dịch chứa X. Ví dụ: c({Milk, Diaper, Beer}) = 2/3.

- Tập mục thường xuyên (frequent/large itemset): là tập mục mà độ hỗ trợ lớn hơn hoặc bằng một giá trị ngưỡng minsup.

1.6.2. Phát hiện luật kết hợp

Với một tập các giao dịch T, mục đích của bài toán phát hiện luật kết hợp là tìm ra tất cả các luật có:

- Độ hỗ trợ s ≥ giá trị ngưỡng **minsup**, và

- Độ tin cậy ≥ giá trị ngưỡng **minconf**.

Cách tiếp cận vét cạn (Brute-force):

- Liệt kê tất cả các luật kết hợp có thể.

- Tính toán độ hỗ trợ và độ tin cậy cho mỗi luật.

- Loại bỏ đi các luật có độ hỗ trợ nhỏ hơn minsup hoặc có độ tin cậy nhỏ hơn minconf.

=> Phương pháp vét cạn này có chi phí tính toán quá lớn, không áp dụng được trong thực tế.

Xét tập mục: {Milk, Diaper, Beer}

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Các luật kết hợp:

{Milk, Diaper} → {Beer} (s=0.4,c=0.67)

{Milk, Beer} → {Diaper} (s=0.4, c=1.0)

{Diaper, Beer} → {Milk} (s=0.4,c=0.67)

{Beer} → {Milk, Diaper} (s=0.4, c=0.67)

{Diaper} → {Milk Beer} (s =04, c=0.5)

{Milk} → {Diaper, Beer} (s=0.4, c=0.5)

Ta thấy tất cả các luật trên đều là sự phân tách (thành 2 tập con) của cùng tập mục : {Milk, Diaper, Beer}.

Các luật sinh ra từ cùng một tập mục sẽ có cùng độ hỗ trợ, nhưng có thể khác về độ tin cậy.

Do đó, trong quá trình phát hiện luật kết hợp, chúng ta có thể tách riêng 2 yêu cầu về độ hỗ trợ và độ tin cậy.

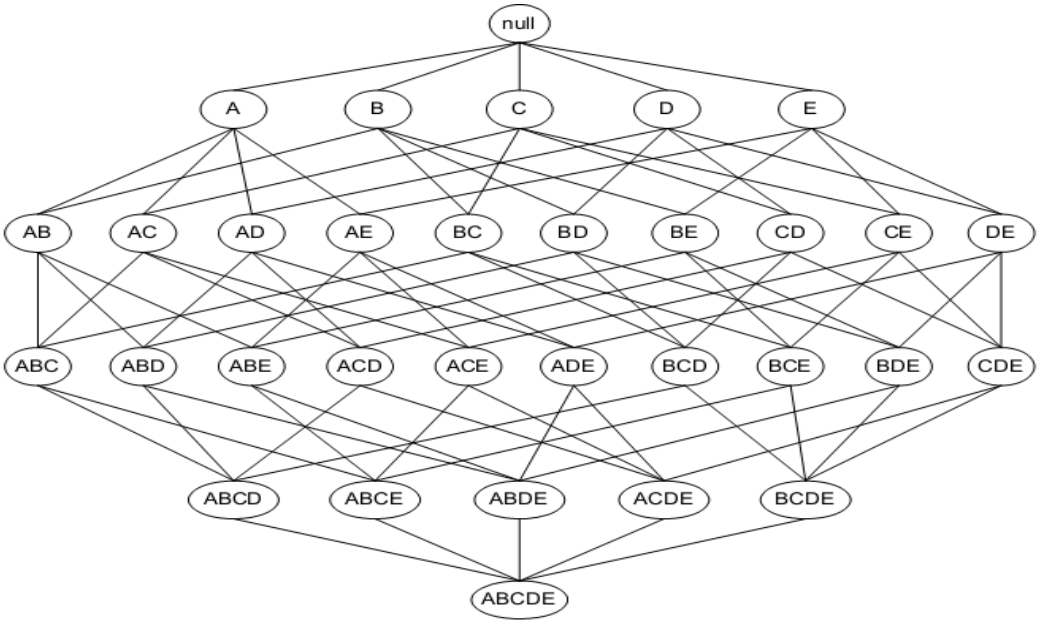
Vây nên quá trình phát hiện luật kết hợp sẽ phân gồm 2 bước (2 giai đoạn) quan trọng:

- Sinh ra các tập mục thường xuyên (frequent/large itemsets): Sinh ra tất cả các tập mục có độ hỗ trợ ≥ **minsup**.

- Sinh ra các luật kết hợp: Từ mỗi tập mục thường xuyên (thu được ở bước trên), sinh ra tất cả các luật có độ tin cậy cao(≥ **minconf**).

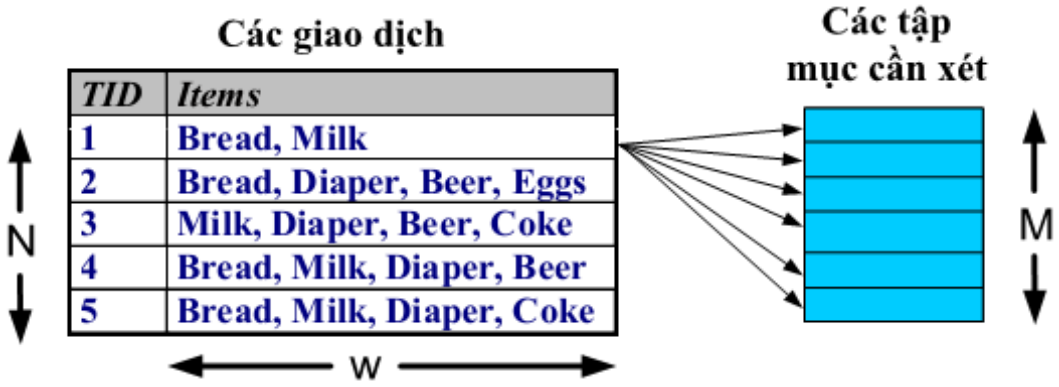
Tuy vậy, bước sinh ra các tập mục thường xuyên (bước 1) vẫn có chi phí tính toán quá cao.

Với *d* mục, thì phải xét đến 2^d các tập mục có thể.



Lược đồ biểu diễn các tập mục cần xét, với *d* = 5

Với phương pháp vét cạn(Brute-force) để sinh ra các tập mục thường xuyên (bước 1):



Hình 1.6.2_Sinh tập mục thường xuyên bằng phương pháp vét cạn

- Mỗi tập mục trong lược đồ đều được xét.

- Tính độ hỗ trợ của mỗi tập mục, bằng cách duyệt qua tất cả các giao dịch.

- Với mỗi giao dịch, so sánh nó với mỗi tập mục được xét.

- Độ phức tạp ~ O(N.M.w). Nếu M = 2^d thì độ phức tạp này là quá lớn.

1.6.3. Các chiến lược sinh tập thường xuyên

Dựa vào các phân tích ở mục 1.6.2, ta có các chiến lược:

- Giảm bớt số lượng các tập mục cần xét (M): Tìm kiếm (xét) đầy đủ $M = 2^d$. Sau đó, sử dụng các kỹ thuật cắt tỉa để giảm giá trị M.

- Giảm bớt số lượng các giao dịch cần xét (N): Giảm giá trị N, khi kích thước (số lượng các mục) của tập mục tăng lên.

- Giảm bớt số lượng các so sánh (matchings/comparisons) giữa các tập mục và các giao dịch (N.M): Sử dụng các cấu trúc dữ liệu phù hợp (hiệu quả) để lưu các tập mục cần xét hoặc các giao dịch. Không cần phải so sánh mỗi tập mục với mỗi giao dịch.

Từ các chiến lược ta xét 2 giải thuật cơ bản:

- Giải thuật Apriori (được trình bày ở mục 3.1).

- Giải thuật FP-Growth.

1.6.4. Giải thuật FP-Growth

FP-Growth biểu diễn dữ liệu của các giao dịch bằng một cấu trúc dữ liệu gọi là FP –tree.

FP-Growth sử dụng cấu trúc FP-tree để xác định trực tiếp các tập mục thường xuyên.

Biểu diễn bằng FP-tree:

- Với mỗi giao dịch, FP-tree xây dựng một đường đi (path) trong cây.

- Hai giao dịch có chứa cùng một số mục, thì đường đi của chúng sẽ có phần (đoạn) chung. Càng nhiều các đường đi có các phần chung, thì việc biểu diễn bằng FP-tree sẽ càng gọn.

- Nếu kích thước của FP-tree đủ nhỏ có thể lưu trữ trong bộ nhớ làm việc, thì giải thuật FP-Growth có thể xác định các tập thường xuyên trực tiếp từ FP-tree lưu trong bộ nhớ.

Xây dựng FP-tree:

- Ban đầu, FP-tree chỉ chứa duy nhất nút gốc (được biểu diễn bởi ký hiệu null).

- Cơ sở dữ liệu các giao dịch được duyệt lần thứ 1, để xác định (tính) độ hỗ trợ của mỗi mục.

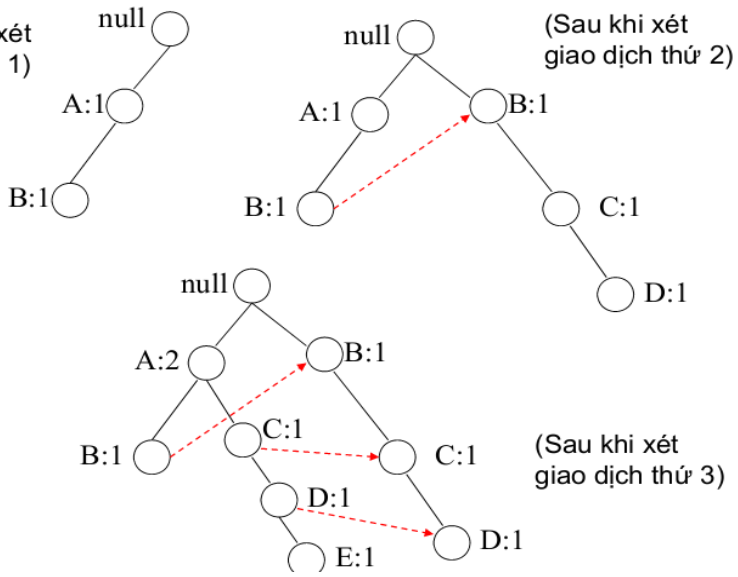
- Các mục không thường xuyên bị loại bỏ.

- Các mục thường xuyên được sắp xếp theo thứ tự giảm dần về độ hỗ trợ.

- Cơ sở dữ liệu các giao dịch được duyệt lần thứ 2, để xây dựng FP-tree.

Ví dụ: Xây dựng FP-tree

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{A}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}



Sinh các tập mục thường xuyên:

- FP-Growth sinh các tập mục thường xuyên trực tiếp từ FP-tree từ mức lá đến mức gốc (bottom-up).

- Vì mỗi giao dịch được biểu diễn bằng một đường đi trong FP -tree, chúng ta có thể xác định các tập mục trong FPtree, chúng ta có thể xác định các tập mục thường xuyên kết thúc bởi một mục (vd: E), bằng cách duyệt các đường đi chứa mục đó (E).

Chương 2. Ứng dụng của khai phá dữ liệu

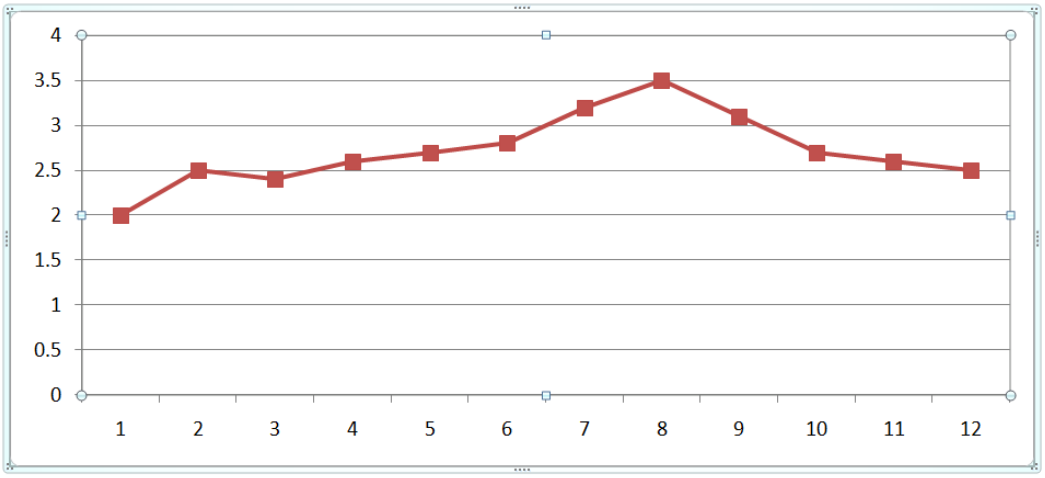
2.1. Hỗ trợ ra quyết định nhập kho trong siêu thị

2.1.1. Giới thiệu về bài toán

Bài toán này tiến hành sử dụng dữ liệu về mặt hàng quạt máy của một siêu thị sau một năm kinh doanh. Sau đó ta tiến hành đồ thị hóa lượng dữ liệu này với trục hoành là tháng, trục tung là sản lượng tương ứng của tháng. Từ đồ thị kinh doanh này ta sẽ xác định một dạng đồ thị thích hợp mà biểu diễn được (gần giống) sự biến thiên của đồ thị đó. Khi đó ứng với năm sau thì ta sẽ dùng lại dạng đồ thị này để dự đoán sức mua mặt hàng quạt của khách hàng trong năm sau vì chúng cũng sẽ diễn ra theo đúng định dạng đồ thị này. Do đó sẽ giúp người quản lý chủ động hơn trong việc nhập về số lượng quạt thích hợp để luôn đáp ứng tốt nhất nhu cầu của khách hàng.

Tháng	Sản Lượng
1	2
2	2.5
3	2.4
4	2.6
5	2.7
6	2.8
7	3.2
8	3.5
9	3.1
10	2.7
11	2.6
12	2.5

Bảng sản lượng kinh doanh mặt hàng quạt của siêu thị



Đồ thị về sức mua quạt máy của siêu thị trong 12 tháng

2.1.2. Đánh giá của thầy sau khi giới thiệu về bài toán

Sau khi tiến hành trình bày bằng slide và thuyết trình về nội dung của bài toán ứng dụng khai phá dữ liệu để hỗ trợ chủ động trong việc ra quyết định của quản kho để đáp ứng nhu cầu khách hàng thì thầy có cho nhận xét. Đối với việc sử dụng hồi quy dữ liệu để sử dụng tính chất *dự báo* để chủ động trong việc nhập mặt hàng quạt máy thì không thích hợp. Trong trường hợp này thì phần mà nhóm giới thiệu chính là cách thức của phương pháp lấy pattern từ một mô hình để khi có sự xuất hiện pattern đó một lần nữa thì ta có thể dự đoán cách thức mà pattern này diễn ra (thường là như các lần trước đây). Bên cạnh đó không nên thu giảm các field liên quan đến thời tiết ngày hôm đó để tăng tính chính xác cho giải thuật.

2.2.Tiếp thị chéo

2.2.1.Giới thiệu về bài toán

Trong một cửa hàng bán lẻ thì người quản lý có rất nhiều cách thức trong việc sắp xếp thứ tự vị trí của các sản phẩm mà họ đang kinh doanh. Trong thực tế thì khách hàng khi mua một sản phẩm A thì thường hay có xu hướng tiếp tục mua tiếp các sản phẩm B, C, D….. có liên quan đến sản phẩm A. Do đó người quản lý phải tìm hiểu về giỏ hàng mà khách hàng thường hay thực hiện trong các giao dịch để rút ra quy luật để từ đó tiến hành sắp xếp lại các mặt hàng thường được mua cùng nhau để đặt chúng cạnh nhau nhằm giúp giảm công sức đi lại cho khách hàng, gợi ý mua hàng để tăng doanh số bán hàng. Để có thể thực hiện được mục tiêu như trên thì ta có thể áp dụng khai phá luật kết hợp để rút ra các quy luật đó.

Ví dụ: Cửa hàng bán lẻ trích ra một số giao dịch mà khách hàng đã thực hiện trong lịch sử giao dịch trước đây của cửa hàng.

TID	*List of item_IDs*
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

Sau khi tiến hành áp dụng một giải thuật khai phá luật kết hợp (vd:Apriori) thì ta được bảng Frequent Itemset như sau:

Itemset	**Support count**
{I1, I2, I3}	2
{I1, I2, I5}	2

Từ bảng này thì người quản lý của cửa hàng bán lẻ sẽ yêu cầu nhân viên của mình tiến hành đặt các sản phẩm có trong Itemset trong cùng một hàng gần nhau (cùng một gian hàng) để giúp người mua hàng và kích thích người tiêu dùng.

Chương 3. Giải thuật Apriori

Trong chương này trình bày giải thuật Apriori của Luật kết hợp, đưa ra các đánh giá và biện pháp cải tiến cho giải thuật.

3.1. Giải thuật Apriori

Quá trình sinh ra luật kết hợp chia làm hai bước. Bước đầu tiên là sinh ra các tập thường xuyên. Bước thứ hai sinh ra các luật kết hợp. Ở mục 1.6.3 ta thấy được bước thứ nhất của quá trình rất phức tạp. Giải thuật Apriori là một phương pháp làm giảm độ phức tạp ở bước này.

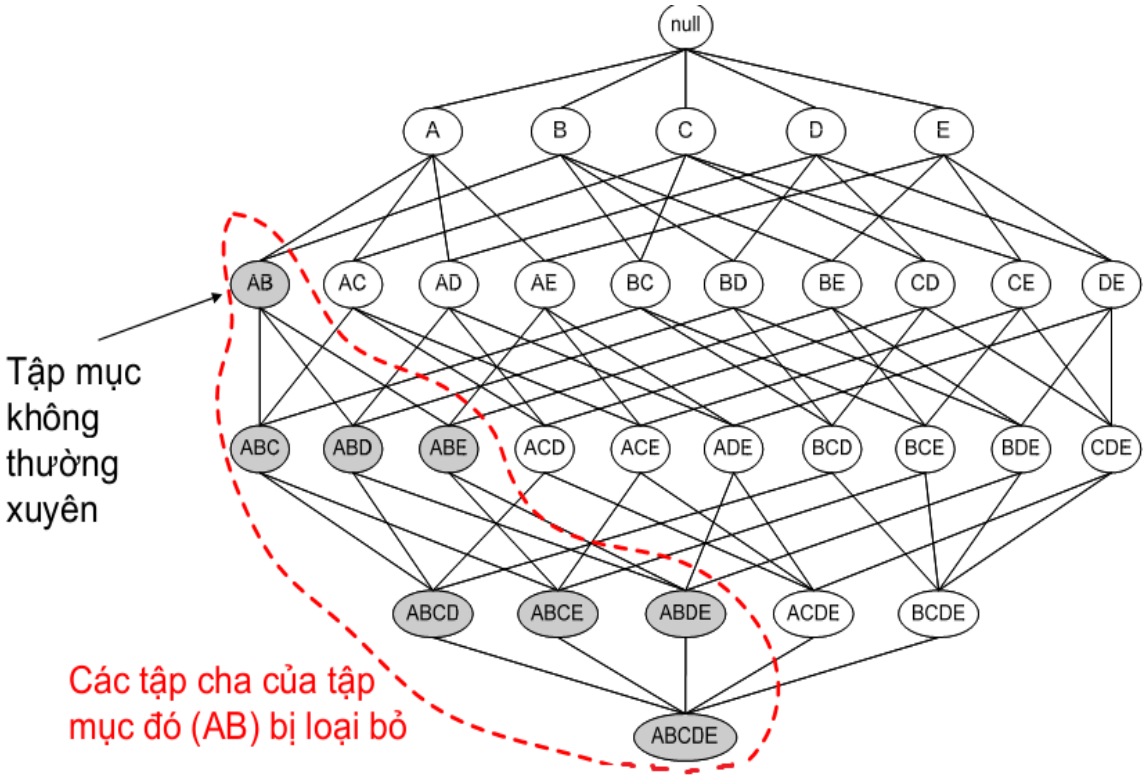
Nguyên tắc của giải thuật Apriori – Loại bỏ dựa trên độ hỗ trợ:

- Nếu một tập mục là thường xuyên, thì tất cả các tập con (subsets) của nó đều là các tập mục thường xuyên.

- Nếu một tập mục là không thường xuyên (not frequent) thì tất cả các tập cha (supersets) của nó đều là các tập mục không thường xuyên.

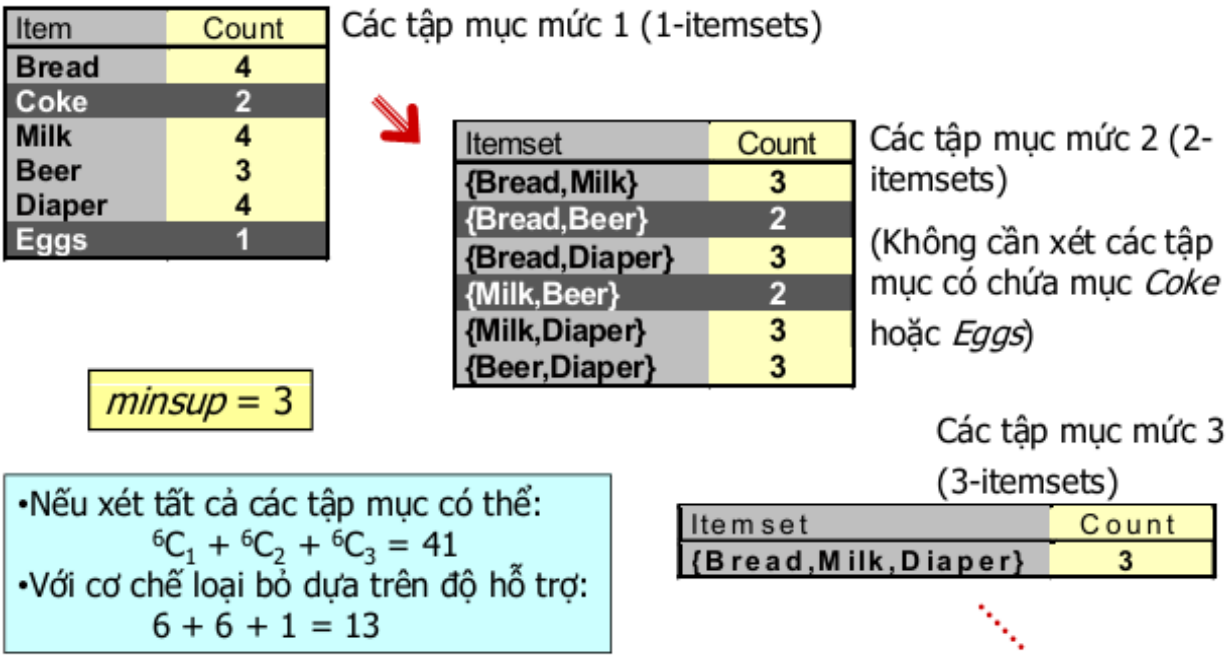
Nguyên tắc của giải thuật Apriori dựa trên đặc tính không đơn điệu (anti-monotone) của độ hỗ trợ:

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$



Lược đồ biểu diễn các tập mục cần xét được loại bỏ bớt theo độ hỗ trợ

Ví dụ: Loại bỏ dựa trên độ hỗ trợ *minsup = 3*



Giải thuật Apriori:

1- Sinh ra tất cả các tập mục thường xuyên mức 1(frequent 1-itemsets)

2- Gán k =1

3- Lặp lại, cho đến khi không có thêm bất kỳ tập mục thường xuyên nào mới.

3.1- Từ các tập mục thường xuyên mức k, sinh ra các tập mục mức (k+1) cần xét.

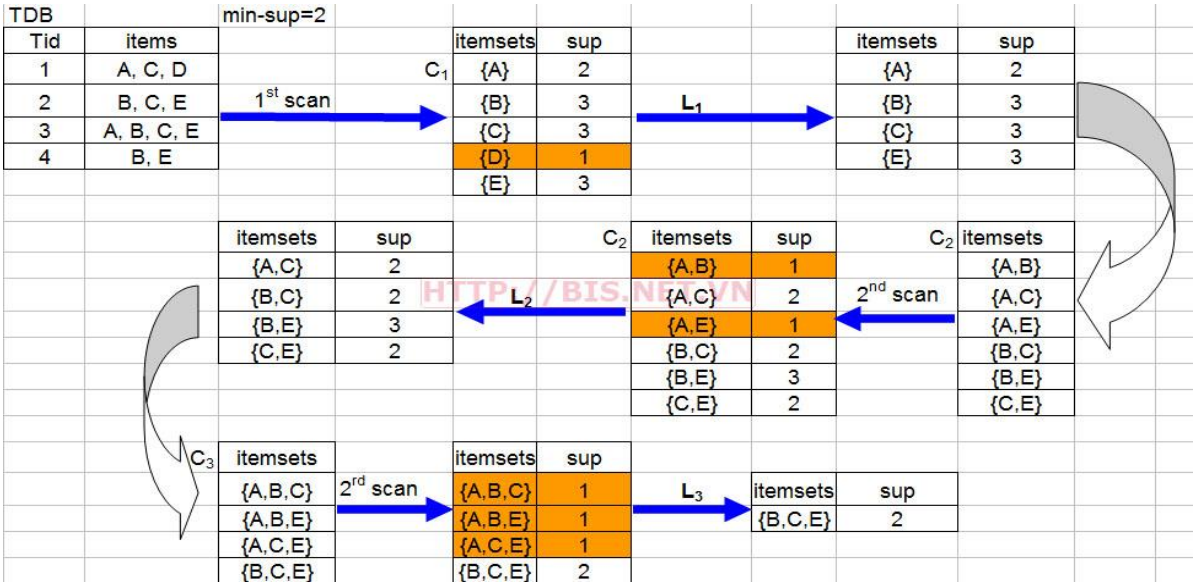
3.2- Loại bỏ các tập mục mức k+1 chứa các tập con là các tập mục không thường xuyên mức k.

3.3- Tính độ hỗ trợ của các tập mục mức k+1, bằng cách duyệt qua tất cả các giao dịch.

3.4- Loại bỏ các tập mục không thường xuyên mức k+1.

3.5- Thu được các tập mục thường xuyên mức k+1.

Ví dụ: Với minsup = 2 [1].



4- Với mỗi tập mục thường (I) thu được, sinh ra tất cả các tập con (B) không rỗng

5- Với mỗi tập B, sinh ra các luật kết hợp: B (I-B)

6- Với mỗi luật kết hợp, duyệt qua tất cả các giao dịch. Chọn các luật có độ tin cậy(c) ≥ minconf

Ví dụ: với I= {A1,A2,A5}

Các tập con của I: {A1}, {A2}, {A5}, {A1,A2},{A1,A5},{A2,A5}

Có các luật kết hợp sau:

{A1} => {A2,A5}; {A2} =>{A1,A5}; {A5} =>{A1,A2};

{A1,A2} =>{A5}; {A1,A5} =>{A2}; {A2,A5} => {A1}

Với frequent itemsets I ={B,C,E}, min_conf =80%. Ta có 2 luật kết hợp là:{B,C} =>{E}; {C,E} =>{B}.

Tid	items
1	A, C, D
2	B, C, E
3	A, B, C, E
4	B, E

min-conf = 80%

Association Rule	Confidence
{B} => {C,E}	67%
{C}=>{B,E}	67%
{E}=>{B,C}	67%
{B,C}=>{E}	100%
{B,E}=>{C}	67%
{C,E}=>{B}	100%

3.2. Đánh giá giải thuật Apriori

Các yếu tố ảnh hưởng:

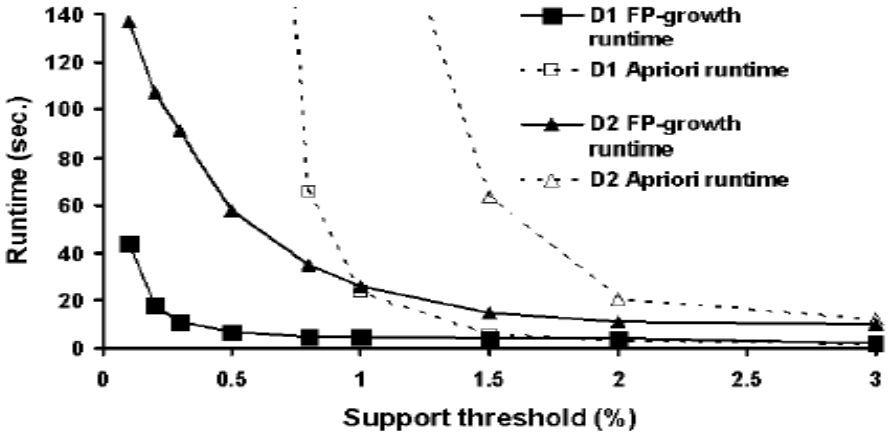
- Lựa chọn giá trị ngưỡng minsup: Giá trị minsup quá thấp sẽ sinh ra nhiều tập mục thường xuyên. Điều này sẽ làm tăng số lượng tập mục phải xét.

- Số lượng các mục trong cơ sở dữ liệu (các giao dịch): Cần thêm bộ nhớ để lưu giá trị độ hỗ trợ với mỗi mục. Nếu số lượng các mục(tập mục mức 1) thường xuyên tăng lên thì chi phí và chi phí I/O (duyệt các giao dịch) cũng tăng.

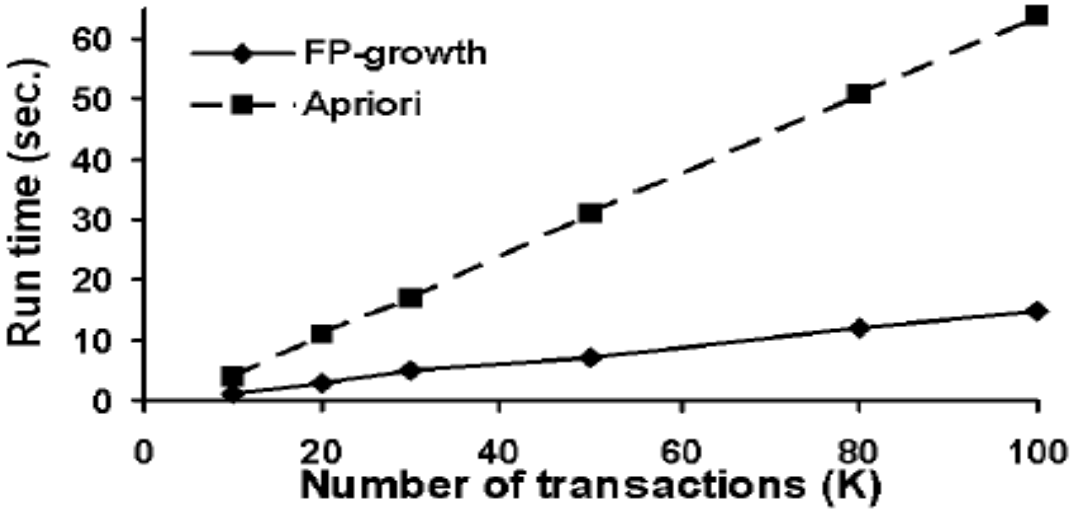
- Kích thước của cơ sở dữ liệu (các giao dịch): Giải thuật phải duyệt cơ sở dữ liệu nhiều lần, do đó chi phí tính toán của Apriori tăng lên khi số lượng các giao dịch tăng lên.

- Kích thước trung bình của các giao dịch: Khi kích thước (số lượng các mục) trung bình của các giao dịch tăng lên, thì độ dài tối đa của các tập mục thường xuyên cũng tăng.

So sánh giữa giải thuật Apriori và giải thuật FP-Growth



Biểu đồ: Độ hỗ trợ - Thời gian chạy



Biểu đồ: Số lượng giao dịch - Thời gian chạy

3.3. Các cải tiến của giải thuật Apriori

Kỹ thuật dựa trên bảng băm (hash-based technique): Một k-itemset ứng với hashing bucket count nhỏ hơn minimum support threshold không là một frequent itemset.

Giảm giao dịch (transaction reduction): Một giao dịch không chứa frequent k-itemset nào thì không cần kiểm tra ở các lần sau (cho k+1-itemset).

Phân hoạch (partitioning): Một itemset phải frequent trong ít nhất một phân hoạch thì mới có thể frequent trong toàn bộ tập dữ liệu.

Lấy mẫu (sampling): Khai phá chỉ tập con dữ liệu cho trước với một trị support threshold nhỏ hơn và cần một phương pháp để xác định tính toàn diện (completeness).

Đếm itemset động (dynamic itemset couting): Chỉ thêm các itemset dự tuyển khi tất cả các tập con của chúng được dự đoán là frequent.

Chương 4. Demo giải thuật Apriori

4.1. Hiện thực giải thuật Apriori

Hiện thực giải thuật Apriori cải tiến dựa trên cấu trúc dữ liệu bảng băm và giảm giao dịch . Tất cả dữ liệu được truy xuất trực tiếp từ database.

a) Ngôn ngữ và công cụ sử dụng

Database sử dụng Mysql.

Ngôn ngữ sử dụng : Java.

Kết nối database: sử dụng thư viện mysql-connector-java-5.1.14-bin.jar.

Giao diện: sử dụng thư viện Swing.

b) Giải Thích Code

Chương trình được viết với 4 class chính, sau đây là mô tả cho từng class:

1-Class "node": chứa các thông tin key, count, next được dùng để lưu các thông tin của các item hoặc 1 tập các item.

2-Class "pushData": đây là class có hàm: getInput với tham số đầu vào là username, password của database, và file dùng để import vào database.

3-Class "getHash": Đây là class chính, chứa các hàm con, thực thi giải thuật apriori cải tiến.

- Function getNameData:

+ Input:

Namedb: tên database cần lấy danh sách database.
User và password của database.

+ Ouput:

Trả về danh sách các database hiện có trong sever.

- Function getInput:

+ Input:

Data: tên của database cần lấy data.
Username và password của database.

+ Output:

Trả về một mảng node các item và count của mỗi item sau khi đã quét toàn bộ database.

- Function calculateInit:

+ Input:

C: là mảng node các key và count của key đó.
Minsub: độ hỗ trợ mà người dùng nhập.

+ Output:

Trả về một danh sách các node có độ hỗ trợ lớn hơn minSub.

- Function genrateTableHash:

+ Input:

L: là danh sách các node mà mỗi node với key là một item.
Count: thứ tự tập của của L_{k+1} sẽ được sinh ra tiếp theo từ tập C_k . Vì key là 1 item => count = 2;

+ Output:

Trả về một bảng băm với key là một tập các thường xuyên của các item được ngăn cách mới dấu ",", vd (123,987). Và node chứa key, và count của key đó, được khởi tạo là 0.

- Function GenrateTableHash_2

+ Input:

L: là danh sách các node mà mỗi node với key gồm nhiều item.
Count: thứ tự tập của của L_{k+1} sẽ được sinh ra tiếp theo từ tập C_k . Vì key gồm nhiều item => count > 2;

+ Output:

Trả về một bảng băm với key là một tập các thường xuyên của các item được ngăn cách mới dấu ",", vd (123,987). Và node chứa key, và count của key đó, được khởi tạo là 0.

- Function CompareData:

+ Input:

hasTable: bảng băm với key là một thường xuyên, và value là node.
Count: thứ tự của tập L.

+ Output:

Trả về một danh sách node từ bảng băm, sau khi đã quét qua database.

- Function GenrateListPartitions:

+ Input:

L: là danh sách các node, với mỗi node là một tập thường xuyên.

+ Output:

Trả về một danh sách của một danh sách, chứa các luật được sinh ra, nhưng chưa so sánh với độ hỗ trợ.

- Function calculateRule:

+ Input:

L: là danh sách của một danh sách, chứa các luật chưa được so sánh với độ hỗ trợ.

MinCon: độ hỗ trợ tối thiểu của các luật.

+ Output:

Trả về một danh sách của một sanh sách, với một danh sách là một luật thỏa mãn độ hỗ trợ tối thiểu.

4-Class "AppAprioriImprove": đây là class chứa giao diện đồ họa, để thao tác với người dùng, gồm các sự kiện:

- jButton4ActionPerformed:

+ Input:

username và password của database của người dùng.

+ Output:

Nếu kết nối thành công, hiện ra danh sách các database có trong server ở combobox của name Data.

Kết nối lỗi, hiện thông báo, và yêu cầu người dùng nhập lại.

- jButton2ActionPerformed:

+ Input:

Username và password của database.
getFile: tên file cần import vào database.
NameData: tên database cần import vào .

+ Output:

Import dữ liệu vào database thành công , xuất hiện thông báo import to database successful.

- jButton5ActionPerformed : tương tự như sự kiện jButton4ActionPerformed.

- jButton6ActionPerformed:

+ Input:

Username và password của người dùng.
nameData: cần khai phá.

Min Support và Min Confidence do người dùng nhập vào.

+ Output:

Xuất ra các tập thường xuyên thỏa độ MinSup, và các luật thỏa MinCon.

4.2. Hướng dẫn sữ dụng demo

4.2.1.Cài đặt môi trường

Cài đặt bộ jdk của java, có thể download từ : http://www.oracle.com/technetwork/java/javase/downloads/index.html

Cài đặt MySQL, ở đây giới thiệu dung bộ Wamp Server, có thể download từ : http://www.wampserver.com/en/

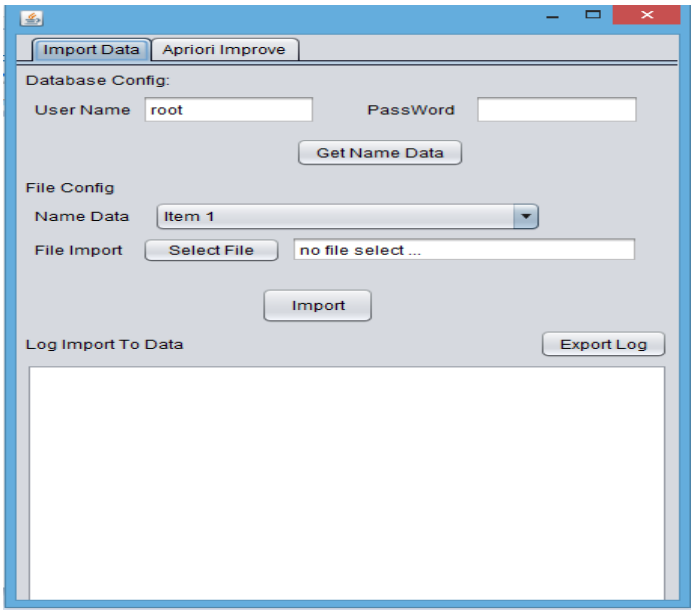
Sử dụng thư viện kết nối database MySQL: mysql-connector-java-5.1.14-bin.jar.

Sử dụng thư viện tổ hợp ra một tập từ một danh sách : combinatoricslib-2.0.jar

4.2.2. Ứng dụng

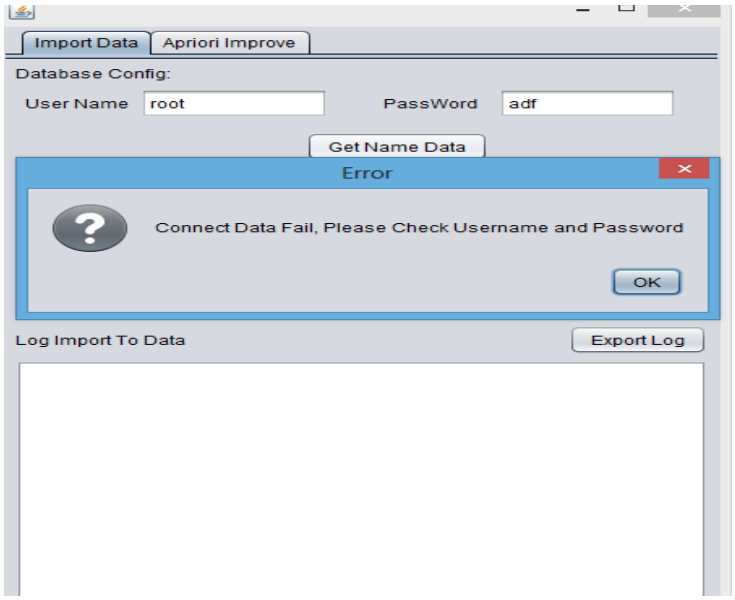
a) Import Data từ File dữ liệu.

Khởi chạy file AppAprioriImprove.jar, hiện thị giao diện:



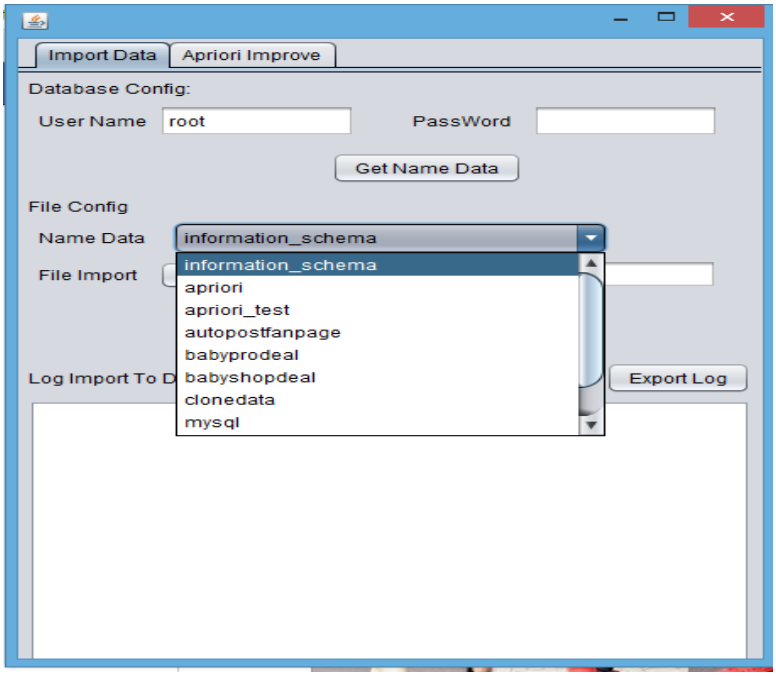
Hình 4.2.2_1

Chọn Tab "import Data" , nhập user name và password của Database. Nếu User name hoặc password sai, thì sẽ báo lỗi connect to database, và bắt người dùng nhập lại.



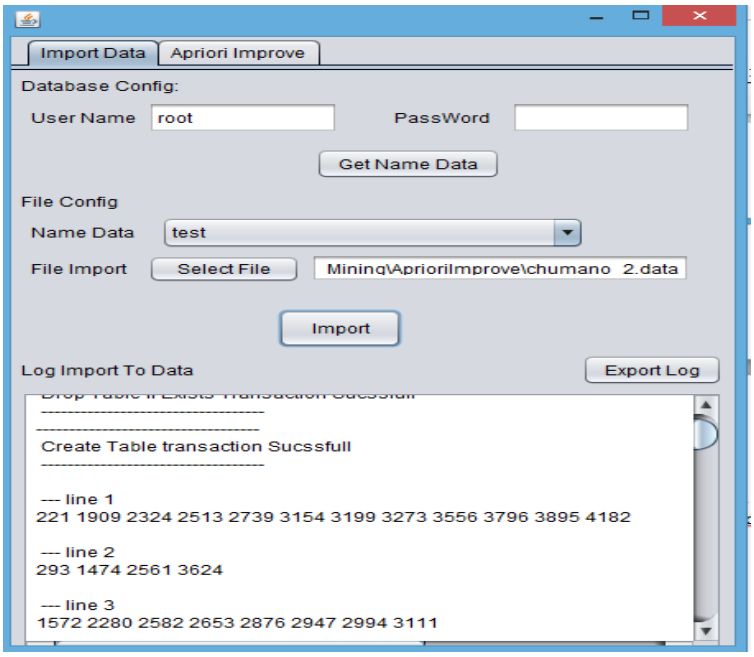
Hình 4.2.2_2

Nhập đúng Username và Password, sẽ hiện ra danh sách các database.



Hình 4.2.2_3

Chọn File, và sau đó nhấn Import, sẽ import file vào database, với bảng "Transaction", Có hiện File Log.



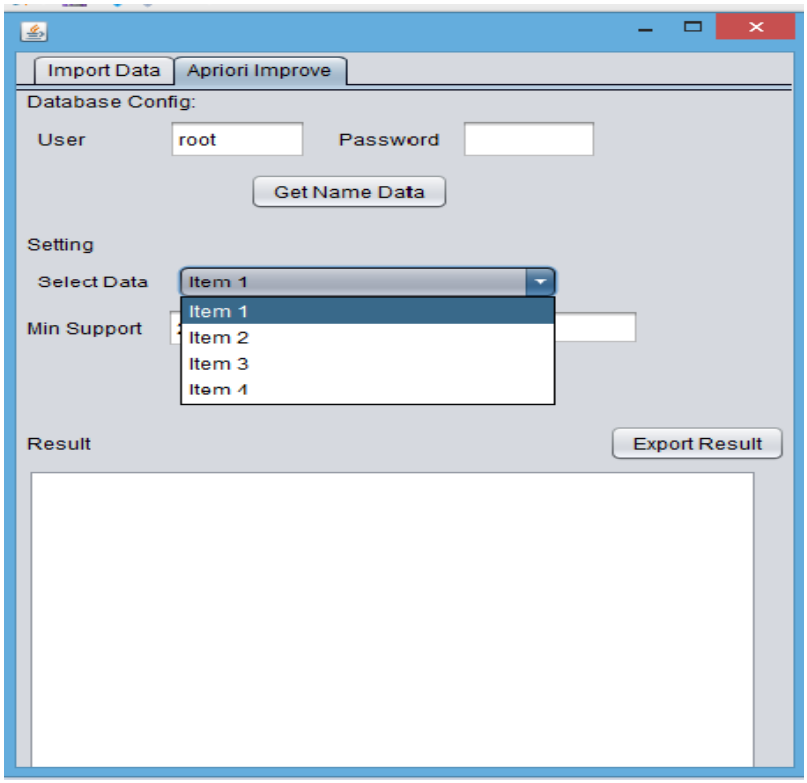
Hình 4.2.2_4

Export log ra file “E:\logPushToData.txt”. Đây là đường dẫn cố định, nếu máy không có ổ đĩa E:\, thì sẽ bị lỗi, xin vui long restart lại App.

b) Chạy Giải Thuật Apriori cải tiến, với Database vừa import.

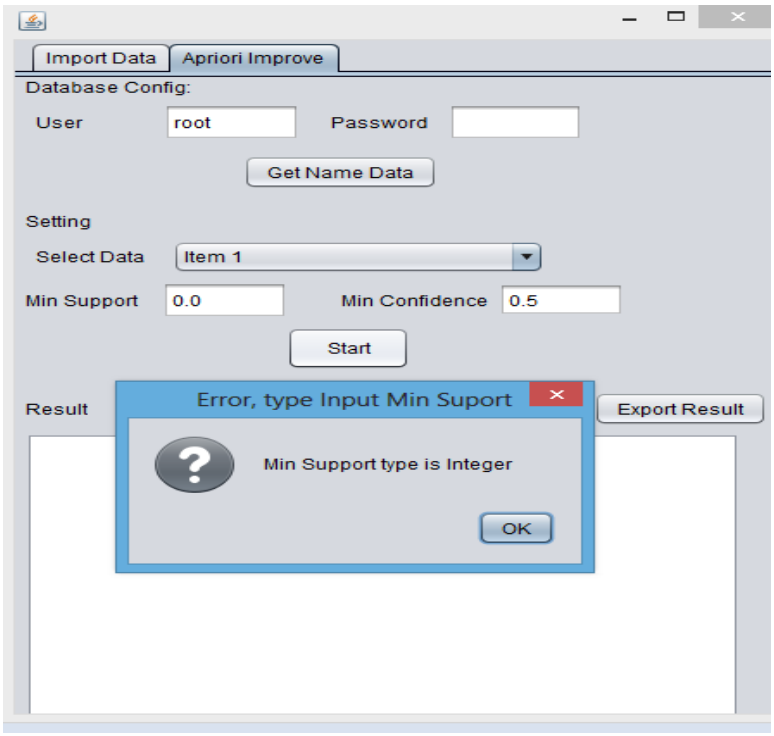
Chọn Tab Apriori Improve. Nhập User name và password tương tự như import data.

Chọn Data để chạy Giải thuật, lưu ý Data phải chứa bảng “Transaction”.

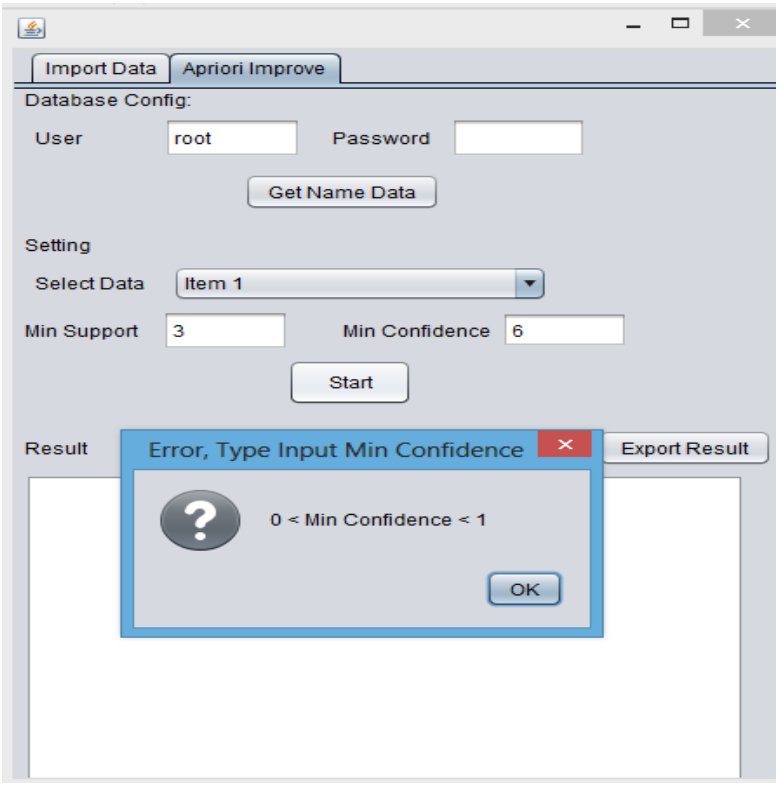


Hình 4.2.2_5

Nhập Min Support kiểu là số nguyên và lớn hơn 0, và Min Confidence là kiểu số thực với 0 < MinCon < 1. Nếu nhập sai, chương trình sẽ báo lỗi, và bắt bạn phải nhập lại.

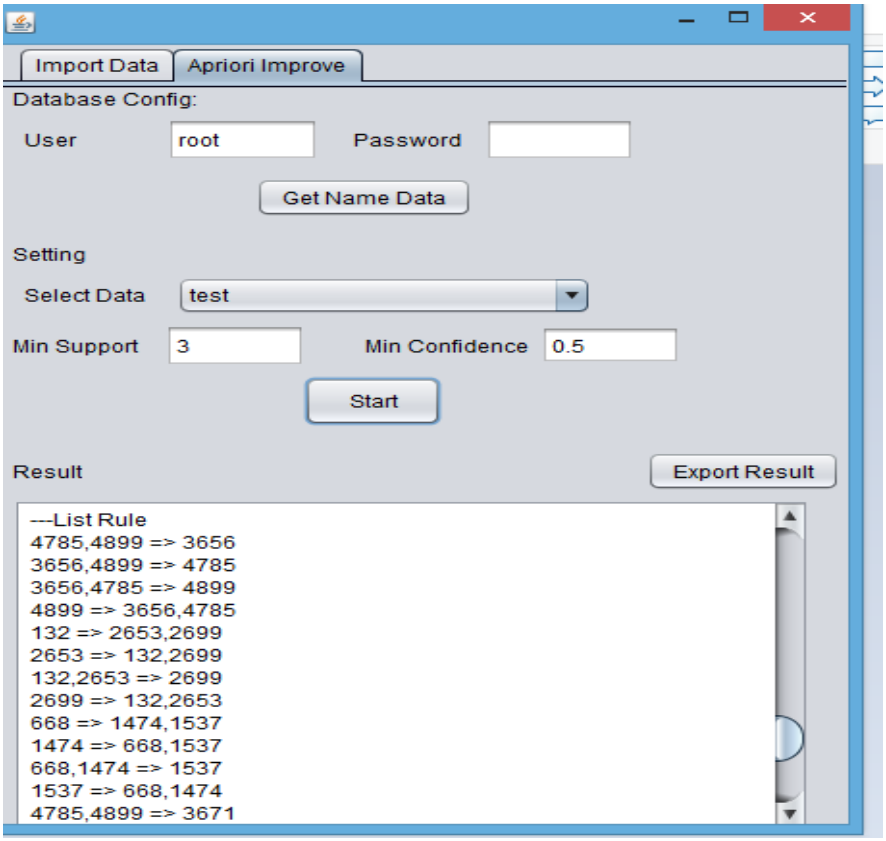


Hình 4.2.2_6



Hình 4.2.2_7

Sau nhập đúng input, chọn Start. Chương trình chạy và xuất ra kết quả.



Hình 4.2.2_8

Nếu muốn Export kết quả, chọn "Export Result", chương trình sẽ mặt định lưu file kết quả vào "E:\Result_GenRule.txt".

Chương 5. Đánh giá tổng kết

5.1. Ưu điểm

Hiểu và hiện thực thành công giải thuật Apriori .

Ứng dụng đã hiện thực được phần cải tiến của giải thuật bằng cấu trúc dữ liệu bảng băm và thực hiện giảm số giao dịch, làm giảm thời gian chạy của giải thuật.

Đồng thời để tăng tốc độ chạy đã lưu toàn bộ dữ liệu vào database. Mọi truy xuất điều thực hiện trực tiếp từ database.

Ứng dụng có giao diện trực quan, dễ sử dụng.

5.2. Nhược điểm

Chưa tìm được cách kiểm tra tính đúng đắn của các luật kết hợp, là kết quả của ứng dụng.

Chưa hiện thực việc loại bỏ các luật kết hợp khi không có tính khả dụng trong thực tế.

Ứng dụng chưa hiện thực được thanh trạng thái đang xử lý dữ liệu ở dạng thanh trạng thái trực quan cho người sử dụng.

Tài liệu tham khảo

- Tập slide bài giảng môn Data Mining , khoa Khoa học & Máy tính, trường Đh Bách Khoa TP Hồ Chí Minh.

- Tập slide bài giảng môn Khai phá Dữ liệu, Nguyễn Nhật Quang, trường Đh Bách Khoa Hà Nội.

- Ebook: Jiawei Han, Micheline Kamber, "Data Mining: Concepts and Techniques", Third Edition, Morgan Kaufmann Publishers.

- [1]: http://bis.net.vn/forums/t/389.aspx

- http://en.wikipedia.org/wiki/Apriori_algorithm