BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC HOA SEN
KHOA KHOA HỌC VÀ CÔNG NGHỆ

KHÓA LUẬN TỐT NGHIỆP

Tên đề tài:

NGHIÊN CỨU MỘT SỐ KỸ THUẬT KHAI THÁC DỮ LIỆU, ỨNG DỤNG TRONG THỊ TRƯỜNG CHỨNG KHOÁN

Giảng viên hướng dẫn : ThS. Phan Đình Thế Huân
Nhóm sinh viên thực hiện: Trương Tấn Đức
Trần Chí Lương
Nguyễn Huỳnh Phương Thảo
Lớp : QL071

Tháng 12 /năm 2010

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC HOA SEN
KHOA KHOA HỌC VÀ CÔNG NGHỆ

KHÓA LUẬN TỐT NGHIỆP

Tên đề tài:

NGHIÊN CỨU MỘT SỐ KỸ THUẬT KHAI THÁC DỮ LIỆU, ỨNG DỤNG TRONG THỊ TRƯỜNG CHỨNG KHOÁN

Giảng viên hướng dẫn : ThS. Phan Đình Thế Huân
Nhóm sinh viên thực hiện: Trương Tấn Đức
Trần Chí Lương
Nguyễn Huỳnh Phương Thảo
Lớp : QL071

Tháng 12 /năm 2010

PHIẾU GIAO ĐỀ TÀI KHÓA LUẬN TỐT NGHIỆP

1. **Mỗi sinh viên phải viết riêng một báo cáo**
2. **Phiếu này phải dán ở trang đầu tiên của báo cáo**

1. Họ và tên sinh viên/ nhóm sinh viên được giao đề tài (sĩ số trong nhóm: 03)

(1) Trương Tấn Đức MSSV: 070099 khóa: QL071

(2) Trần Chí Lương MSSV: 071176 khóa: QL071

(3) Nguyễn Huỳnh Phương Thảo MSSV: 070079 khóa: QL071

Chuyên ngành : Công nghệ thông tin Khoa : Khoa học và Công nghệ

2. Tên đề tài : Nghiên cứu một số kỹ thuật khai thác dữ liệu, ứng dụng trong thị trường chứng khoán.

3. Các dữ liệu ban đầu:

Đã tìm hiểu lĩnh vực khai thác dữ liệu (data mining).

4. Các yêu cầu đặc biệt:

Thu thập dữ liệu chứng khoán từ sàn HoSE.

5. Kết quả tối thiểu phải có:

1.Nắm bắt được một thuật toán trong hướng khai thác dữ liệu thời gian (time series data).

2.Nắm bắt được kỹ thuật plug-in thuật toán vào Data mining engine của MS SQL Server 2008.

3.Triển khai được thuật toán khai thác dữ liệu thời gian trên dữ liệu thu thập thực tế.

Ngày giao đề tài: 06/09/2010 Ngày nộp báo cáo: 24/12/2010

Họ tên GV hướng dẫn 1: Phan Đình Thế Huân Chữ ký: ……

Họ tên GV hướng dẫn 2: …… Chữ ký: …..……..

Ngày .... tháng ... năm

TRÍCH YẾU

Data mining hay còn được gọi là khai thác dữ liệu. Đây là lĩnh vực rộng lớn quan tâm đến việc tìm ra tri thức nằm trong kho dữ liệu lớn. Trong đề án chuyên ngành nhóm đã tìm hiểu về lĩnh vực khai thác dữ liệu, tìm hiểu chi tiết ba thuật toán Apriori, Prefixspan, Clospan của cách tiếp cận Sequential Pattern Mining và cài đặt Apriori trên dữ liệu chứng khoán. Tiếp tục phát triển đề tài về khai thác dữ liệu ở khóa luận tốt nghiệp, nhóm đã chốt lại những điểm quan trọng của khai thác dữ liệu, thực hiện nghiên cứu sâu về các thuật toán từ sau thuật toán Clospan đến thời điểm hiện tại, cùng với các thuật toán có sẵn trong Analysis Services của Microsoft SQL Server. Sau đó, nhóm đã tìm hiểu cách cài đặt thuật toán tích hợp vào Analysis Services. Đồng thời, nhóm tiến hành cài đặt thuật toán Apriori để có những thể nghiệm về việc cài đặt thuật toán như đã tìm hiểu. Sau cùng nhóm cài đặt thuật toán tích hợp Bide, một thuật toán được xem là bước tiến lớn từ sau thuật toán Clospan và xây dựng ứng dụng độc lập chạy thuật toán tích hợp trên dữ liệu chứng khoán. Nhằm đảm bảo dữ liệu đúng và đủ để thuật toán có thể khai thác và cho ra tri thức, nhóm đã phát triển ứng dụng tự động tải dữ liệu về từ trang web chứng khoán.

MỤC LỤC

LỜI CẢM ƠN

Nhóm xin gửi lời cảm ơn đến nhà trường, khoa Khoa học và Công nghệ đã tạo điều kiện cho nhóm hoàn thành khóa luận tốt nghiệp này.

Hơn nữa, nhóm xin chân thành cảm ơn giảng viên hướng dẫn là thầy Phan Đình Thế Huân. Thầy đã cung cấp các tài liệu cần thiết, liên kết các trang chuyên đề về lĩnh vực khai thác dữ liệu và tận tình theo dõi, cố vấn hướng đi cho nhóm trong thời gian thực hiện khóa luận.

Qua việc thực hiện khóa luận này, nhóm đã có cơ hội tìm hiểu về một lĩnh vực khai thác dữ liệu còn mới lạ và đầy tìm năng, cũng như có thêm kiến thức về Analysis Servives. Nhóm đã rút ra được những kinh nghiệm về nghiên cứu và phát triển sản phẩm cho riêng mình và học hỏi được cách thức làm việc nhóm một cách hiệu quả.

Chân thành cảm ơn.

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

Giảng viên hướng dẫn ký tên

DANH MỤC HÌNH

DANH MỤC BẢNG

TỪ ĐIỂN THUẬT NGỮ

Từ được dịch

Data mining	khai thác dữ liệu
Knowledge Discovery in Database (KDD)	khám phá tri thức từ dữ liệu
Knowlegde extraction	trích xuất dữ liệu
Data/pattern analysis	phân tích dữ liệu/mẫu
Data archaeology	khảo cổ dữ liệu
Data dredging	nạo vét dữ liệu
Descriptive	hướng mô tả
Predictive	hướng dự đoán
Data cleaning	làm sạch dữ liệu
Data integration	tích hợp dữ liệu
Data selection	lựa chọn dữ liệu
Data transformation	chuyển hoá dữ liệu
Pattern evaluation	đánh giá mẫu
Knowledge representation	biểu diễn tri thức
Classification	chức năng phân loại
Association	chức năng kết hợp
Regression	chức năng hồi quy
Clustering	chức năng gom nhóm, cụm
Sequence analysis	chức năng phân tích chuỗi
Deviation analysis	chức năng phân tích độ lệch
Time series data	dữ liệu chuỗi thời gian
Continuous data	dữ liệu có tính liên tục
Discrete data	dữ liệu có tính rời rạc
Database	cơ sở dữ liệu (CSDL)
Projected database	cơ sở dữ liệu chiếu
Sequential database	cơ sở dữ liệu dạng chuỗi
Candidate generation	cơ chế tạo chuỗi ứng viên
Closed sequence	chuỗi đóng

Super sequence	chuỗi cha
Subsequence	chuỗi con
Pseudo projection	chiếu giả
Support	độ phổ biến
Confidence	độ tin cậy
Importance	độ quan trọng
Itemset (of a sequence)	bộ dữ liệu
Item (of an itemset)	mục dữ liệu
Sequential Pattern Mining (SPAM)	hướng khai thác mẫu chuỗi
Trend analysis	hướng phân tích xu hướng
Similarity search	hướng tìm kiếm tương tự
Periodicity pattern	hướng tìm mẫu có tính chu kỳ
Matching	so khớp
Trend or long-term movements	dao động theo thời gian
Cyclic movements or cyclic variations	biến đổi theo chu kỳ
Seasonal movements or seasonal variations	dao động hợp lý
Irregular or random movements	dao động bất thường
Full periodicity pattern	mẫu chu kỳ chính xác
Partial periodicity pattern	mẫu nửa chu kỳ
Vertical	dữ liệu dạng dọc
Horizonal	dữ liệu dạng ngang
Stream data	dữ liệu dòng
Spatial data	dữ liệu không gian
Uncertain data	dữ liệu không chắc chắn
Biological data	dữ liệu trong sinh học
Association analysis	phân tích kết hợp
Tag	thẻ
Journal	tạp chí chuyên đề khoa học
Conference	hội nghị
Workshop	hội thảo
Symposium	tiểu luận

1. NHẬP ĐỀ

Khai thác dữ liệu được nghiên cứu và triển khai tại các nước trên thế giới cách đây khoảng ba mươi năm và hiện tại vẫn đang tiếp tục. Ở nước ta lĩnh vực này còn mới tuy nhiên cũng dần được đưa vào ứng dụng. Nhóm nhận thấy thị trường chứng khoán vào nước ta hơn mười năm trở lại đây nên khối dữ liệu đã tương đối nhiều. Nếu ta áp dụng các kỹ thuật khai thác nhất định trong lĩnh vực khai thác dữ liệu vào khối dữ liệu này thì đó là một cơ hội tiềm năng để ta có thể:

- Tìm ra được xu hướng tăng giảm của các mã chứng khoán (với tỷ lệ chính xác chấp nhận được).
- Sau khi tìm được càng nhiều xu hướng gần giống nhau thì có thể gọi đó là luật và dùng chúng để dự đoán đầu tư (mua, bán hoặc giữ), thành lập hệ chuyên gia tư vấn về thị trường chứng khoán.

Để có thể áp dụng được kiến thức về khai thác dữ liệu vào thực tiễn nhóm đã đặt ra những mục tiêu trước khi thực hiện khóa luận bao gồm:

1. Tìm hiểu các hướng chính để khai thác dữ liệu chứng khoán.
2. Xây dựng ứng dụng tự động tải dữ liệu chứng khoán.
3. Nghiên cứu về các thuật toán phát triển từ sau thuật toán Clospan.
4. Tìm hiểu Data mining engine trong SQL.
5. Tìm hiểu về chức năng, dữ liệu đầu vào và tri thức tìm được của các thuật toán được sử dụng trong Microsoft SQL Server Analysis Services.
6. Cấu hình và cài đặt thuật toán tích hợp vào Analysis Services.
7. Xây dựng ứng dụng độc lập sử dụng thuật toán tích hợp Analysis Services của Microsoft SQL Server để khai thác dữ liệu chứng khoán.

2. TỔNG QUAN

2.1. Phân tích vấn đề

Xét CSDL sau:

Ngay	Ma	GiaThChieu	GiaMo	GiaCao	GiaThap	GiaDong	TDGia	TLTDGia	KLGD	GTGD
2009-12-24	OPC	52	52	52.5	51	52.5	0.5	1	10230	532075000
2009-12-24	PAC	72	72	73	71	73	1	1.4	39200	2837670000
2009-12-24	PET	21	21	22	20.5	22	1	4.8	572590	12229031000
2009-12-24	PGC	18.8	18.3	19.4	18.3	19.4	0.6	3.2	120650	2289066000
2009-12-24	PGD	53	53	54	51.5	54	1	1.9	208970	11131775000
2009-12-24	PHR	34.5	34.3	35.8	33.5	35.5	1	2.9	89590	3126112000
2009-12-24	PHT	36.4	34.6	38	34.6	38	1.6	4.4	27400	1012103000
2009-12-24	PIT	13	12.8	13.3	12.7	13	0	0	40950	531178000
2009-12-24	PJT	10.8	10.6	11.2	10.6	11.1	0.3	2.8	8530	93572000
2009-12-24	PNC	9.1	9.2	9.2	8.8	9.2	0.1	1.1	20100	182287000
2009-12-24	PNJ	58	57	59	56.5	59	1	1.7	62940	3649470000
2009-12-24	PPC	17.8	17.8	18.1	17.4	18.1	0.3	1.7	1179300	20908062000

Hình 2.1 CSDL chứng khoán

Ta thấy các mã chứng khoán có các thuộc tính như giá tham chiếu, giá mở cửa, giá cao nhất, giá thấp nhất, giá đóng cửa, tỷ lệ thay đổi giá, khối lượng giao dịch… được tính theo từng ngày. Tổng cộng có gần 300 mã sẽ tạo nên 300 dòng. Như vậy, mỗi ngày cần phải cập nhật thêm 300 dòng dữ liệu. Điều này cho thấy khối lượng của CSDL sẽ tăng rất nhiều theo thời gian. Khi nhìn vào CSDL như vậy ta có thể thấy được chính xác thông tin của một mã bằng cách thống kê, tuy nhiên ta không thể xác định được gì thêm vì dữ liệu quá nhiều và gây rối tầm nhìn, ví dụ như:

- Có mối liên hệ giữa mã này và mã khác hay không, làm thế nào biết được?
- Liệu rằng sự dao động một loại giá nào đó của mã OPC có thể làm khối lượng giao dịch của mã PHR tăng hay không?
- Dự đoán giá mở cửa của một mã nào đó dựa vào một số thuộc tính trước đó.
- Có thông tin nào đáng giá bên dưới khối dữ liệu kia giúp ta dự đoán trước được xu hướng tăng giảm của các mã chứng khoán bằng một số phương pháp nhất định không?

Tất cả câu hỏi trên đều có liên quan đến xác suất, nghĩa là không thể đúng tuyệt đối. Tuy nhiên, nếu ta nắm được quy luật biến đổi thì cơ hội thành công sẽ cao. Một số ví dụ về quy luật đơn giản dễ thấy như: nếu trời mây đen mù mịt thì ắt hẳn sẽ dẫn đến mưa, trong số khách hàng mua Laptop thì 80% người sẽ mua thêm kệ nâng máy để giảm nhiệt, hay khi lướt web thì một số liên kết cụ thể sẽ được nhiều người truy cập vào nhất, từ đó dự đoán độ tuổi, chẳng hạn trong số những người vào trang web giải trí thì có 60% là thiếu niên truy cập liên kết trò chơi trực tuyến, 20% là trung niên truy cập liên kết phim…

Để giải quyết các câu hỏi trên nếu chỉ dùng phương pháp thống kê thì chưa đủ vì chúng còn mang tính dự đoán, vì vậy nhóm đã tìm hiểu và nghiên cứu lĩnh vực khai thác dữ liệu.

2.2. Khai thác dữ liệu

2.2.1. Khái niệm [3]

Khai thác dữ liệu nổi lên từ cuối những năm 1980 và có bước tiến vượt bậc vào những năm 1990. Năm 1989, Fayyad, Piatestsky Shapiro và Smyth đã dùng khái niệm "Khám phá tri thức trong CSDL" (KDD) để chỉ toàn bộ quá trình phát hiện các tri thức có ích từ kho chứa dữ liệu.

Về định nghĩa hình thức, khai thác dữ liệu được định nghĩa là "quá trình trích xuất hay khai thác tri thức từ một lượng lớn dữ liệu". Một số các tên gọi khác của khai thác dữ liệu như:

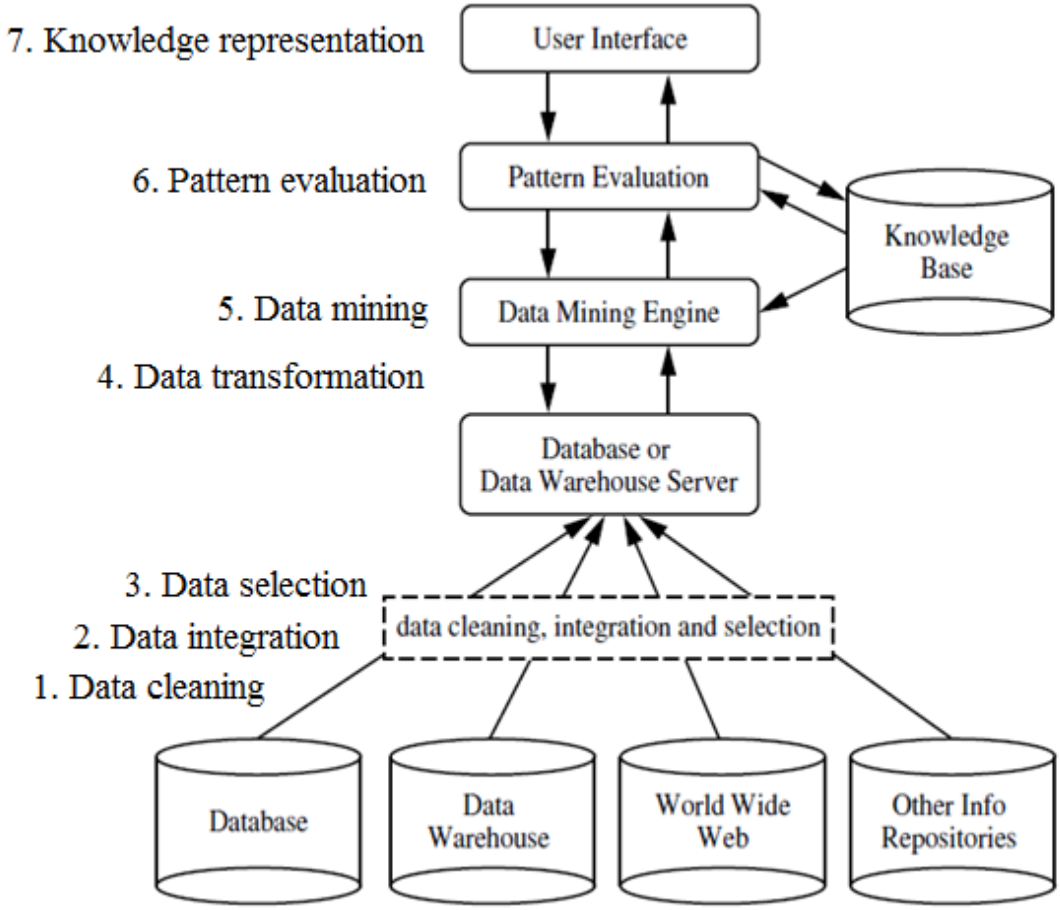
- Khai phá tri thức từ CSDL.
- Trích xuất dữ liệu.
- Phân tích dữ liệu/mẫu.
- Khảo cổ dữ liệu.
- Nạo vét dữ liệu.

Thực chất lĩnh vực này liên quan đến việc phân tích các dữ liệu và sử dụng các kỹ thuật để tìm ra các mẫu có tính lặp lại đều đặn trong tập dữ liệu để phục vụ hai hướng mục đích là mô tả và dự đoán. Dữ liệu ở đây có thể là văn bản, số, chuỗi DNA, tín hiệu âm thanh… và lưu tại CSDL như kho chứa, quan hệ… Hướng mô tả nhằm biểu hiện các đặc tính khái quát của loại dữ liệu đang phân tích, các chức năng phổ biến của hướng này là phân nhóm, kết hợp và phân tích chuỗi. Hướng dự đoán là dựa vào dữ liệu sẵn có ta dự đoán được xu hướng dao động của chúng và biểu diễn xu hướng đó bằng các cách thức khác nhau, các chức năng phổ biến của hướng này là phân loại, hồi quy và phân tích độ lệch. Trong mỗi chức năng của mỗi hướng sẽ bao gồm những thuật toán, kỹ thuật khai thác cụ thể và chúng có thể kết hợp với nhau trong các giai đoạn khai thác dữ liệu để cho ra kết quả có chất lượng.

Nhìn chung, khai thác dữ liệu có nhiều ưu điểm. So với học máy, khai thác dữ liệu có thể áp dụng với CSDL lớn, chứa nhiều dữ liệu nhiễu. So với hệ chuyên gia, khai thác dữ liệu không yêu cầu dữ liệu có chất lượng quá cao. Với phương pháp thống kê, đây là một trong các nền tảng lý thuyết mà khai thác dữ liệu đã kế thừa.

2.2.2. Quy trình [3]

Toàn bộ quy trình khai thác dữ liệu của một hệ thống khai thác dữ liệu đúng nghĩa được trình bày như hình 2.2.



Hình 2.2 Quy trình khai thác dữ liệu của một hệ thống khai thác dữ liệu

1) Làm sạch dữ liệu: xử lý các dữ liệu trùng nhau, bị nhiễu hoặc thiếu…

2) Tích hợp dữ liệu: gộp dữ liệu từ các nguồn khác nhau như: CSDL quan hệ, kho chứa, dữ liệu dạng text, excel, thu thập từ các địa điểm khác nhau (chi nhánh công ty, địa phương)...

3) Lựa chọn dữ liệu: lấy ra các dữ liệu cần quan tâm để khai thác trong CSDL đã gom, ví dụ như tạo view.

4) Chuyển hoá dữ liệu: chuyển dữ liệu thành đầu vào cho thuật toán khai thác, ví dụ như chuyển dữ liệu liên tục thành rời rạc.

5) Khai thác dữ liệu: dùng thuật toán cụ thể để khai thác dữ liệu và cho ra các mẫu đạt yêu cầu. Đây là giai đoạn quan trọng nhất bên cạnh việc thu thập và tiền xử lý dữ liệu.

6) Đánh giá mẫu: đánh giá chất lượng các mẫu bằng một số ràng buộc như các độ đo, giá trị độ phổ biến, độ tin cậy...

7) Biểu diễn tri thức: trình bày các mẫu đạt (tri thức) trên giao diện dễ hiểu, có ý nghĩa đối với người dùng và có thể cho phép người dùng thực hiện những thao tác đơn giản như xóa, lọc...

2.2.3. Ứng dụng

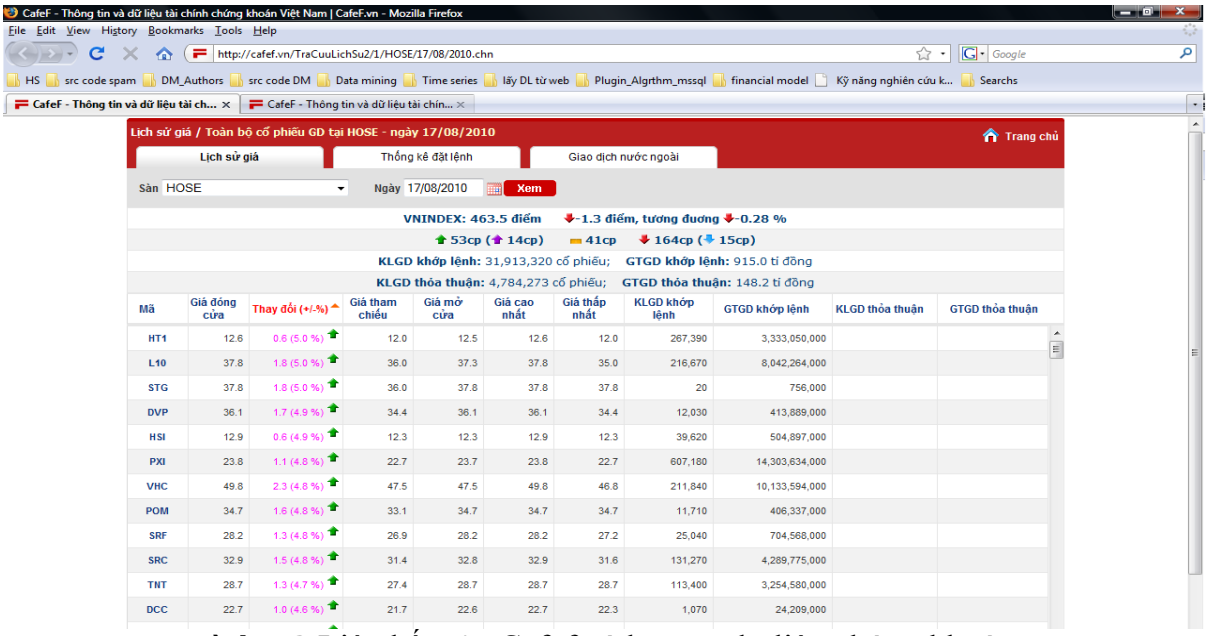
Khai thác dữ liệu được ứng dụng rất nhiều tại các nước trên thế giới về các lĩnh vực như phân tích DNA để chuẩn đoán ung thư, bào chế thuốc, máy tìm kiếm trên Web, dự đoán kết quả bầu cử, phát hiện các loại tội phạm gian lận, đặc biệt là các doanh nghiệp kinh doanh sử dụng để phân tích hành vi khách hàng, dữ liệu trang web…

Ví dụ: phần mềm IBM SPSS Modeler của SPSS (công ty IBM), có khả năng phân loại khách hàng tiềm năng, quản lý rủi ro kinh doanh, chiến lược tiếp thị... Sản phẩm thương mại có tại
http://www.spss.com/software/modeler/

Kdnuggets là trang web lớn về khai khác dữ liệu.
Liên kết http://www.kdnuggets.com/solutions/index.html có chứa nhiều phần mềm thương mại của các công ty lớn, sử dụng khai thác dữ liệu vào các lĩnh vực đa dạng như phát hiện tội phạm, CRM, thể thao, thương mại điện tử, sinh học…

2.3. Khai thác dữ liệu trong thị trường chứng khoán

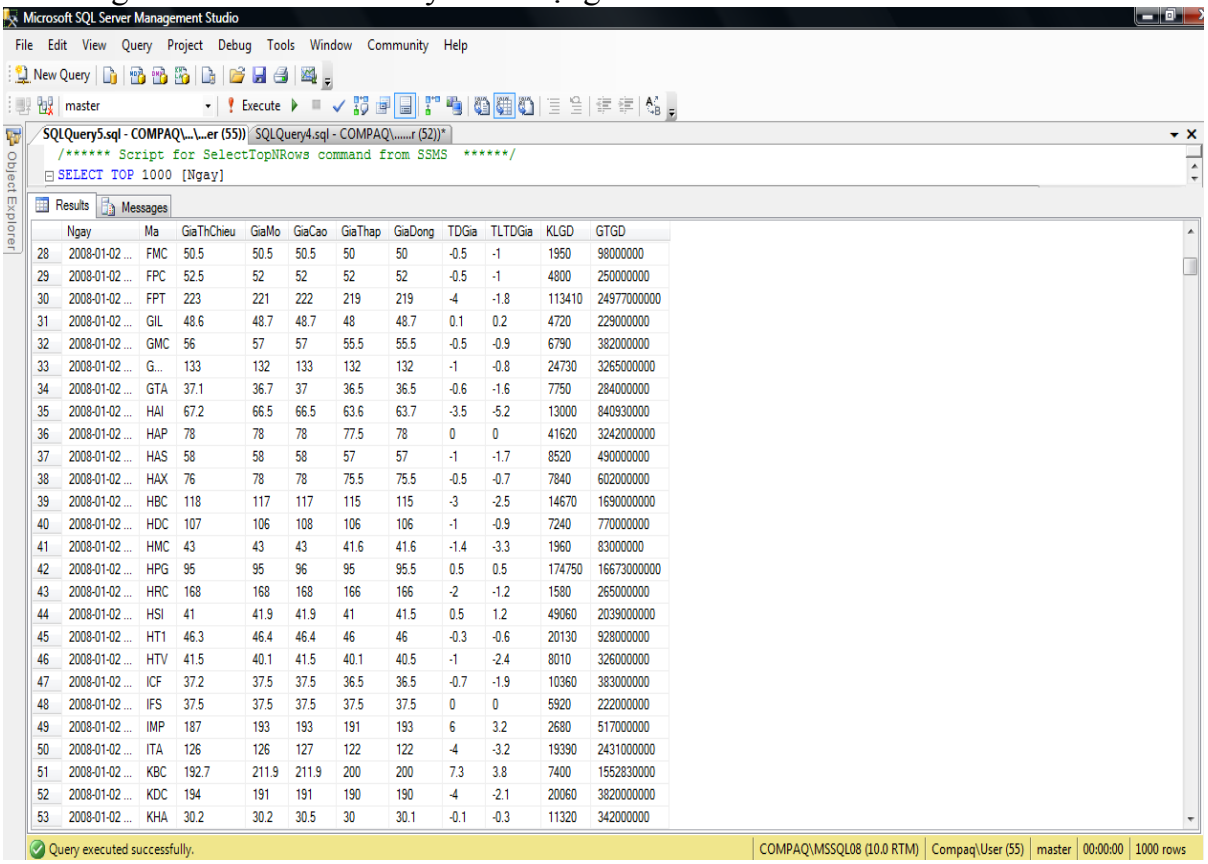
Dữ liệu chứng khoán của nhóm được lấy về tự động từ nguồn http://cafef.vn/.



Mã	Giá đóng cửa	Thay đổi (+/-%)	Giá tham chiếu	Giá mở cửa	Giá cao nhất	Giá thấp nhất	KLGD khớp lệnh	GTGD khớp lệnh	KLGD thỏa thuận	GTGD thỏa thuận
HT1	12.6	0.6 (5.0 %)	12.0	12.5	12.6	12.0	267,390	3,333,050,000		
L10	37.8	1.8 (5.0 %)	36.0	37.3	37.8	35.0	216,670	8,042,264,000		
STG	37.8	1.8 (5.0 %)	36.0	37.8	37.8	37.8	20	756,000		
DVP	36.1	1.7 (4.9 %)	34.4	36.1	36.1	34.4	12,030	413,889,000		
HSI	12.9	0.6 (4.9 %)	12.3	12.3	12.9	12.3	39,620	504,897,000		
PXI	23.8	1.1 (4.8 %)	22.7	23.7	23.8	22.7	607,180	14,303,634,000		
VHC	49.8	2.3 (4.8 %)	47.5	47.5	49.8	46.8	211,840	10,133,594,000		
POM	34.7	1.6 (4.8 %)	33.1	34.7	34.7	34.7	11,710	406,337,000		
SRF	28.2	1.3 (4.8 %)	26.9	28.2	28.2	27.2	25,040	704,568,000		
SRC	32.9	1.5 (4.8 %)	31.4	32.8	32.9	31.6	131,270	4,289,775,000		
TNT	28.7	1.3 (4.7 %)	27.4	28.7	28.7	28.7	113,400	3,254,580,000		
DCC	22.7	1.0 (4.6 %)	21.7	22.6	22.7	22.3	1,070	24,209,000		

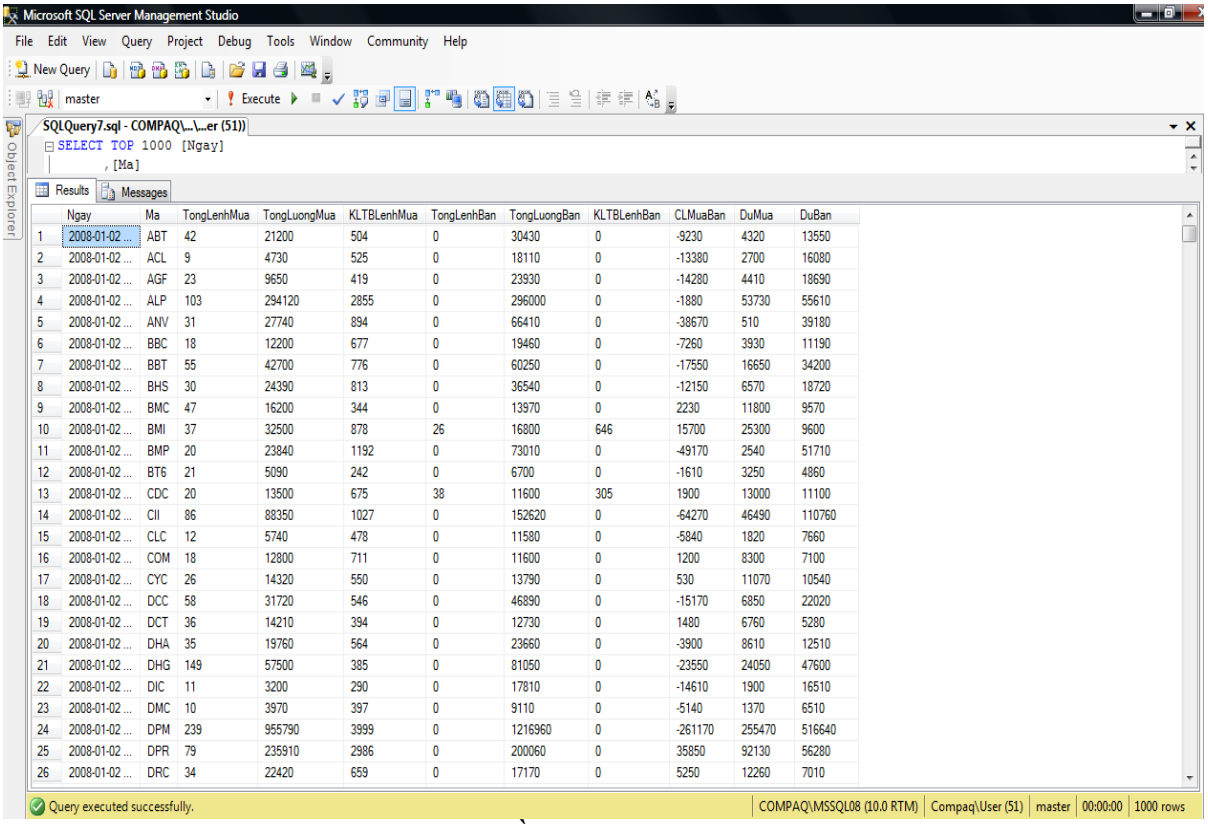
Hình 2.3 Liên kết của Cafef có lưu trữ dữ liệu chứng khoán

Các bảng của CSDL sau khi lấy về có dạng như hình 2.4 và hình 2.5.



	Ngay	Ma	GiaThChieu	GiaMo	GiaCao	GiaThap	GiaDong	TDGia	TLTDGia	KLGD	GTGD
28	2008-01-02 ...	FMC	50.5	50.5	50.5	50	50	-0.5	-1	1950	98000000
29	2008-01-02 ...	FPC	52.5	52	52	52	52	-0.5	-1	4800	250000000
30	2008-01-02 ...	FPT	223	221	222	219	219	-4	-1.8	113410	24977000000
31	2008-01-02 ...	GIL	48.6	48.7	48.7	48	48.7	0.1	0.2	4720	229000000
32	2008-01-02 ...	GMC	56	57	57	55.5	55.5	-0.5	-0.9	6790	382000000
33	2008-01-02 ...	G...	133	132	133	132	132	-1	-0.8	24730	3265000000
34	2008-01-02 ...	GTA	37.1	36.7	37	36.5	36.5	-0.6	-1.6	7750	284000000
35	2008-01-02 ...	HAI	67.2	66.5	66.5	63.6	63.7	-3.5	-5.2	13000	840930000
36	2008-01-02 ...	HAP	78	78	78	77.5	78	0	0	41620	3242000000
37	2008-01-02 ...	HAS	58	58	58	57	57	-1	-1.7	8520	490000000
38	2008-01-02 ...	HAX	76	78	78	75.5	75.5	-0.5	-0.7	7840	602000000
39	2008-01-02 ...	HBC	118	117	117	115	115	-3	-2.5	14670	1690000000
40	2008-01-02 ...	HDC	107	106	108	106	106	-1	-0.9	7240	770000000
41	2008-01-02 ...	HMC	43	43	43	41.6	41.6	-1.4	-3.3	1960	83000000
42	2008-01-02 ...	HPG	95	95	96	95	95.5	0.5	0.5	174750	16673000000
43	2008-01-02 ...	HRC	168	168	168	166	166	-2	-1.2	1580	265000000
44	2008-01-02 ...	HSI	41	41.9	41.9	41	41.5	0.5	1.2	49060	2039000000
45	2008-01-02 ...	HT1	46.3	46.4	46.4	46	46	-0.3	-0.6	20130	928000000
46	2008-01-02 ...	HTV	41.5	40.1	41.5	40.1	40.5	-1	-2.4	8010	326000000
47	2008-01-02 ...	ICF	37.2	37.5	37.5	36.5	36.5	-0.7	-1.9	10360	383000000
48	2008-01-02 ...	IFS	37.5	37.5	37.5	37.5	37.5	0	0	5920	222000000
49	2008-01-02 ...	IMP	187	193	193	191	193	6	3.2	2680	517000000
50	2008-01-02 ...	ITA	126	126	127	122	122	-4	-3.2	19390	2431000000
51	2008-01-02 ...	KBC	192.7	211.9	211.9	200	200	7.3	3.8	7400	1552830000
52	2008-01-02 ...	KDC	194	191	191	190	190	-4	-2.1	20060	3820000000
53	2008-01-02 ...	KHA	30.2	30.2	30.5	30	30.1	-0.1	-0.3	11320	342000000

Hình 2.4 CSDL chứng khoán của sàn HOSE



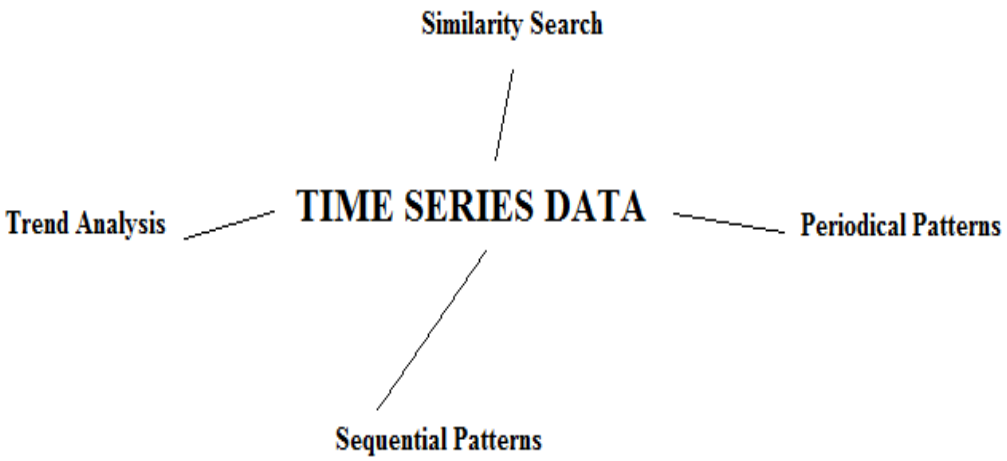
	Ngay	Ma	TongLenhMua	TongLuongMua	KLTBLenhMua	TongLenhBan	TongLuongBan	KLTBLenhBan	CLMuaBan	DuMua	DuBan
1	2008-01-02 ...	ABT	42	21200	504	0	30430	0	-9230	4320	13550
2	2008-01-02 ...	ACL	9	4730	525	0	18110	0	-13380	2700	16080
3	2008-01-02 ...	AGF	23	9650	419	0	23930	0	-14280	4410	18690
4	2008-01-02 ...	ALP	103	294120	2855	0	296000	0	-1880	53730	55610
5	2008-01-02 ...	ANV	31	27740	894	0	66410	0	-38670	510	39180
6	2008-01-02 ...	BBC	18	12200	677	0	19460	0	-7260	3930	11190
7	2008-01-02 ...	BBT	55	42700	776	0	60250	0	-17550	16650	34200
8	2008-01-02 ...	BHS	30	24390	813	0	36540	0	-12150	6570	18720
9	2008-01-02 ...	BMC	47	16200	344	0	13970	0	2230	11800	9570
10	2008-01-02 ...	BMI	37	32500	878	26	16800	646	15700	25300	9600
11	2008-01-02 ...	BMP	20	23840	1192	0	73010	0	-49170	2540	51710
12	2008-01-02 ...	BT6	21	5090	242	0	6700	0	-1610	3250	4860
13	2008-01-02 ...	CDC	20	13500	675	38	11600	305	1900	13000	11100
14	2008-01-02 ...	CII	86	88350	1027	0	152620	0	-64270	46490	110760
15	2008-01-02 ...	CLC	12	5740	478	0	11580	0	-5840	1820	7660
16	2008-01-02 ...	COM	18	12800	711	0	11600	0	1200	8300	7100
17	2008-01-02 ...	CYC	26	14320	550	0	13790	0	530	11070	10540
18	2008-01-02 ...	DCC	58	31720	546	0	46890	0	-15170	6850	22020
19	2008-01-02 ...	DCT	36	14210	394	0	12730	0	1480	6760	5280
20	2008-01-02 ...	DHA	35	19760	564	0	23660	0	-3900	8610	12510
21	2008-01-02 ...	DHG	149	57500	385	0	81050	0	-23550	24050	47600
22	2008-01-02 ...	DIC	11	3200	290	0	17810	0	-14610	1900	16510
23	2008-01-02 ...	DMC	10	3970	397	0	9110	0	-5140	1370	6510
24	2008-01-02 ...	DPM	239	955790	3999	0	1216960	0	-261170	255470	516640
25	2008-01-02 ...	DPR	79	235910	2986	0	200060	0	35850	92130	56280
26	2008-01-02 ...	DRC	34	22420	659	0	17170	0	5250	12260	7010

Hình 2.5 CSDL chứng khoán về đặt lệnh của sàn HOSE (HOSE_ORDER)

Mô tả chi tiết ứng dụng tải dữ liệu tự động và CSDL được trình bày trong phụ lục A.

Ta thấy hầu hết các thuộc tính trong CSDL chứng khoán là số, vì vậy có khả năng sâu chuỗi cao. Hơn nữa, chúng thay đổi liên tục hằng ngày, mỗi lần thay đổi là cả một "series" giá của một mã xác định và tất cả "series" của tất cả mã (toàn bộ bảng giao dịch trực tuyến). Chính vì vậy, dữ liệu chứng khoán được xác định là dữ liệu chuỗi thời gian. Dữ liệu chuỗi thời gian được định nghĩa là một chuỗi các điểm dữ liệu, được đo từng lần kế tiếp nhau tại các khoảng thời gian thống nhất. Nói nôm na, đây là dữ liệu dạng chuỗi các giá trị thu thập được trong một khoảng thời gian lặp xác định như hằng ngày, hằng tuần, ví dụ: dữ liệu chứng khoán, tiền lương, lượng nước chảy hằng năm của sông…

Nghiên cứu về dữ liệu chuỗi thời gian nằm trong 10 vấn đề lớn của khai thác dữ liệu. Về loại dữ liệu này, ta có bốn cách tiếp cận chính sau: [12]



Hình 2.6 Các hướng tiếp cận để khai thác dữ liệu chuỗi thời gian

Hướng tìm kiếm tương tự

Hướng này nhằm phát hiện ra các chuỗi có khác biệt rất ít so với các chuỗi khác. Cụ thể, quy trình tìm kiếm sẽ xóa mờ các khác biệt đó (không phải là xóa bỏ hoàn toàn mà là làm cho giống) trong một ngưỡng giới hạn cho phép. Có hai loại là: so khớp chuỗi con và so khớp toàn bộ chuỗi, tương ứng với chiều dài của chuỗi dữ liệu cần khai thác mà chọn phương pháp phù hợp. Tri thức tìm được của tìm kiếm tương tự, chẳng hạn từ dữ liệu đã có về giá tăng giảm của một mã nào đó ta vẽ được biểu đồ dao động, trong biểu đồ sẽ có tăng, giảm, tăng mạnh, hay ổn định… Kế đến phương pháp sẽ tìm kiếm xem liệu các mã khác có mô hình dao động giống hoặc gần giống biểu đồ đó trong tương lai gần hay không bằng cách tìm chuỗi và so khớp như trên. Như vậy từ một mã ta có thể dự đoán được nhiều mã khác.

Các phương pháp tìm kiếm tương tự:

Trước khi vào các phương pháp chính, có một số phương pháp phụ như:

- Làm giảm tải khối lượng CSDL (vẫn đảm bảo chất lượng dữ liệu):
  - "The Discrete Fourier Transform" (DFT).
  - "Discrete Wavelet Transforms" (DWT).

  - "Singular Value Decomposition (SVD) based on Principle Components Analysis" (PCA).
  - "Random Projection-based Sketch Techniques".
- Các phương pháp tạo chỉ mục cho CSDL dạng chuỗi để việc tìm kiếm nhanh hơn như R-tree, R*-tree…

Phương pháp chính:

- Qua ba bước: "Atomic Matching", "Window Stitching", và "Subsequence Ordering".
- Sử dụng độ đo (ngưỡng giới hạn cho phép) là "Euclidean Distance", "Normalization Transformation",…
- Phương pháp "Shape Definition Language" được dùng để tìm các chuỗi giống nhau.

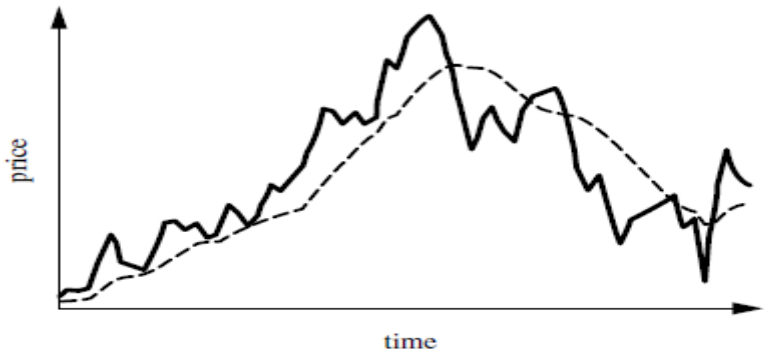
Hướng phân tích xu hướng

Hướng này nhằm tìm mẫu của một thuộc tính sẽ thay đổi như thế nào qua thời gian. Ví dụ: xét mã ABC, với tất cả dữ liệu về giá của mã này ta tạo được một hàm Y = F(t) có t là thời gian và y là một thuộc tính nào đó, và biểu diễn bằng biểu đồ. Chẳng hạn, ta có thể biết được mỗi thứ năm ABC sẽ tăng 3% (khớp với hàm), hay cứ thứ sáu là ABC giảm khoảng 2% (gần đúng hàm). Như vậy, ta có thể dự đoán thuộc tính bất kỳ của một mã. Hướng này có thể xử lý cả dữ liệu liên tục và dữ liệu rời rạc. Quá trình khai thác gồm hai giai đoạn:

a. Mô hình hoá dữ liệu chuỗi thời gian.
b. Dự báo dữ liệu.

Để mô hình hoá dữ liệu chuỗi thời gian, cần bốn loại mẫu:

- Dao động theo thời gian: dựa vào dữ liệu khai thác, cho thấy xu hướng dao động theo kỳ hạn dài.



Hình 2.7 Mô hình biểu hiện xu hướng dao động theo thời gian

- Biến đổi theo chu kỳ: cho thấy sự lặp lại của dao động theo một chu kỳ hoặc bị sai lệch một khoảng nào đó so với giá trị cũ khi đã hết một chu kỳ.

- Dao động hợp lý: cho thấy các hướng dao động có vẻ hợp lý (dựa vào dữ liệu đã khai thác), chẳng hạn bán đắt socola vào các ngày gần lễ valentine, hay bao lì xì vào lễ tết, giá của mã thường tăng khi nhiều người mua, giảm khi ngược lại…
- Dao động bất thường: cho thấy những dao động khách quan hiếm khi xảy ra, và xảy ra ngẫu nhiên như giá tăng hay giảm đột biến do các biến cố: lao động đình công, thiên tai, nhân viên thuyên chuyển,…

Các kỹ thuật để tìm, phân loại bốn mẫu trên và xây dựng mô hình là "Regression Analysis" (kỹ thuật hồi quy này thuộc chức năng phân lớp và dự đoán), "Weighted Moving Average", "Freehand", "Least Squares", "Autocorrelation Analysis"… Một số nghiên cứu mới dùng các kỹ thuật xử lý khác của chức năng gom cụm, phân nhóm dữ liệu theo tính chất nào đó của các loại mã, hoặc sử dụng Decision Tree, Neural Network thuộc chức năng phân loại… tuy nhiên không được phổ biến.

Mô hình chuỗi thời gian có cơ sở lập luận là khi quan sát trong khoảng thời gian gần, mô hình sử dụng một lối thứ tự theo thời gian vì thế giá trị được đưa ra trong một thời kỳ sẽ biểu hiện được lấy bằng cách nào đó từ giá trị cũ trước đó.

Sau khi có mô hình, bước sau sẽ dự báo dựa vào mô hình đó bằng cách tính toán từng điểm dữ liệu thông qua thời gian có các cấu trúc nội tại thuộc bốn loại mẫu trên, sử dụng mô hình để dự đoán sự kiện tương lai dựa vào sự kiện quá khứ: dự đoán điểm dữ liệu trước khi chúng được đo, chẳng hạn dự đoán giá mở cửa của chứng khoán dựa vào các đặc điểm trước đó. Phương pháp phổ biến để dự báo là ARIMA (Auto-Regressive Integrated Moving Average). ARIMA được phát triển bởi George Box và G.M. Jenkins, là kết hợp của hai phương pháp "Auto-regressive" và "Moving Average" nên còn được gọi là hệ phương pháp Box-Jenkins, đây được xem là nền tảng của phân tích chuỗi thời gian tĩnh. Có ba giai đoạn của hệ phương pháp trên là: định dạng mô hình, ước lượng và đánh giá.

Về sau mô hình chuỗi thời gian đã trở nên tinh vi hơn và trên thế giới đã nghiên cứu để cho ra mô hình điều kiện Heteroskedasticity bằng ARCH (Autoregressive Conditional Heteroskedasticity) và GARCH (Generalized Autoregressive Conditional Heteroskedasticity), thường dùng trong kiểu dữ liệu chuỗi thời gian liên quan đến tài chính. Bên cạnh đó, mô hình chuỗi thời gian cũng được dùng để tìm hiểu các mối quan hệ bên trong giữa các tham biến được biểu diễn bởi hệ thống các phương trình sử dụng VAR (Vector Autoregression) và mô hình cấu trúc VAR.

Phân tích xu hướng là hướng tiếp cận phổ biến nhất để khai thác dữ liệu chuỗi thời gian của các nghiên cứu trên thế giới.

Mẫu có tính chu kỳ

Hướng này tìm các mẫu có tính tái diễn. Ví dụ: nếu quán có nhiều khách hàng vào uống nước lúc 4 đến 5 giờ chiều, thì các món ăn của quán sẽ bán chạy vào lúc 6 đến 7 giờ tối. Có hai loại:

- Mẫu chu kỳ chính xác: là mẫu có tính tái diễn (chu kỳ) gần đúng toàn bộ. Ví dụ: bốn mùa trong một năm gần như diễn ra theo đúng thời gian và chưa bao

giờ sai (không chênh lệch thời gian nhiều hay thậm chí bị đảo thứ tự), sự dao động của mã (xem xét tất cả các ngày trong tuần).

- Mẫu nửa chu kỳ: là mẫu có tính tái diễn chỉ đúng khoảng một phần, tất nhiên mẫu này sẽ không có chất lượng cao như mẫu trên, tuy nhiên đây lại là dạng mẫu nhiều nhất trên thực tế. Ví dụ: ta thường uống cà phê vào buổi sáng, nhưng các hoạt động khác trong ngày lại diễn ra rất lộn xộn không theo giờ giấc như: xem ti vi, đi chơi, học bài…; sự dao động của mã vào mỗi thứ hai, hoặc vào chiều thứ năm...

Trong quá trình khai thác, ta phải xác định các đặc tính chính xác hoặc gần đúng của mỗi mẫu. Sau đó đối với mẫu chính xác, dùng phương pháp Fast Fourier Transformation và mẫu gần đúng dùng các tính chất của thuật toán Apriori và các biến thể của thuật toán này.

Mẫu chuỗi

Sequential **PA**ttern **M**ining (SPAM) được giới thiệu bởi Agrawal và Srikant vào 1995, đây là một trường hợp đặc biệt của dữ liệu chuỗi thời gian, cơ sở dữ liệu chứa loại dữ liệu này là dạng chuỗi bao gồm chuỗi các biến cố được sắp theo thứ tự xảy ra có thể theo một thời gian cụ thể hoặc không. Đây là chủ đề được rất nhiều người và nhóm chuyên gia nghiên cứu trong khoảng 10 năm trở lại và được xem là chủ đề nổi lên như một tiềm năng bên cạnh phân tích xu hướng. Mục đích của hướng này là tìm các mối quan hệ giữa những biến cố xảy ra để xác định thứ tự xảy ra của chúng, ta có thể phân tích các biến cố của một nhóm chuỗi cụ thể hoặc phân tích nhiều nhóm khác nhau, ví dụ như dữ liệu của một mã hoặc của tất cả các mã. Phương pháp khai thác mẫu chuỗi dùng để xem xét sự biến động của giá cổ phiếu là kết quả của sự kiện cổ phiếu trước đó (mua, bán, giữ). Các sự kiện khác nhau có thể dẫn đến giá cả khác nhau. Sau đó, ý tưởng là làm sao để dự đoán được những hành vi như vậy nhằm giúp các nhà đầu tư tối ưu hóa việc quản lý danh mục đầu tư. Một trong những vấn đề cơ bản của việc phân tích trình tự của sự kiện là tìm các chuỗi con của sự kiện xảy ra phổ biến, tức là các bộ sự kiện xảy ra thường xuyên với một trật tự nhất định và trong phạm vi thời gian cụ thể.

Các thuật toán khai thác mẫu chuỗi có thể trích xuất các mô hình thống kê quan trọng của mẫu A => B [t] (sự kiện B sau sự kiện A trong thời gian t) từ các chuỗi sự kiện.

Khai thác mẫu nửa chu kỳ có thể được xem là một trường hợp của SPAM khi xem các chuỗi chu kỳ là tập các chuỗi đầu vào của SPAM.

2.4. Hướng chọn của nhóm

Mỗi hướng cho ra tri thức khác nhau. Nhóm chọn khai thác mẫu chuỗi vì hướng này được nghiên cứu nhiều trên thế giới trong các năm gần đây. Hơn nữa, cơ chế của phương pháp có phần đơn giản hơn phân tích xu hướng, phù hợp với mục đích của nhóm là phát hiện các thành phần xuất hiện thường xuyên, phổ biến trong dữ liệu, và xây dựng luật kết hợp từ các thành phần đó vì chúng xuất hiện thường xuyên nên luật xây dựng từ chúng đáng để xem xét. Các luật này có tính linh động cao, phụ thuộc vào

dữ liệu đầu vào vế trái và vế phải của luật. Nếu luật đúng, ta sẽ nắm được nhiều thông tin mà người khác không nhận thấy được.

2.5. Khảo sát các thuật toán của phương pháp SPAM

Mô tả chi tiết về khảo sát và tài liệu được trình bày trong phần phụ lục B.

2.5.1. Các khái niệm cơ bản [12]

Giả sử ta muốn biết cách thức đầu tư của người chơi chứng khoán, họ thường hay mua các loại mã nào hoặc điều gì khiến cho đồng loạt người chơi bán ra hoặc mua vào một mã nào đó, để biết được ta cần phải phân tích CSDL có các thuộc tính như khối lượng giao dịch, giá mở cửa, giá đóng cửa của các mã… Kết quả phân tích được có thể cho ra một số mệnh đề với tỷ lệ chính xác tương đối đúng với đa số trường hợp. Các mệnh đề này chính là các luật kết hợp. Theo thời gian, các luật có thể sai và cần được cập nhật. Tập các luật đúng sẽ được lưu trữ. Xét một luật cụ thể:

Giá đóng cửa của mã ABC giảm 5% → khối lượng giao dịch mã DEF tăng 3% với độ phổ biến = 2% và độ tin cậy = 70%.

- Độ phổ biến = 2% có nghĩa là trong cơ sở dữ liệu mà ta đang xét, cần có 2% trong tất cả các dòng thì chứa cả 2 giá trị là giá đóng cửa mã ABC giảm 5% và khối lượng giao dịch mã DEF tăng 3% để thỏa điều kiện.
- Độ tin cậy = 70% có nghĩa là có 70% trong tất cả các dòng đều cho thấy khi giá đóng cửa của ABC giảm 5% thì khối lượng giao dịch DEF sẽ tăng 3%.

Về mặt hình thức, độ phổ biến được định nghĩa là số lần xuất hiện của dữ liệu đang xét trong cơ sở dữ liệu (tính theo từng dòng). Độ tin cậy là độ chính xác của vế phải trong điều kiện đã xảy ra vế trái.

Về công thức, gọi $I = \{i_1, i_2, \ldots, i_m\}$ là bộ dữ liệu. Cơ sở dữ liệu D chứa các giao dịch T là bộ dữ liệu ($T \subseteq I$). Mỗi giao dịch có TID để nhận dạng (khoá chính). Gọi A là một tập dữ liệu nào đó, ta có $T \in D$ tính độ phổ biến cho $A \in I$ nếu chứa tất cả các mục của A, ($A \subseteq T$).

Ta có A → B là dạng của luật kết hợp với $A \subset I$, $B \subset I$ và A không có liên quan gì đến B ($A \cap B = Ø$). Ta có công thức tính như sau:

- Độ phổ biến (A → B) = P ($A \cap B$).
- Độ tin cậy (A → B) = P(B|A) = P { (B) ∩ (A)} / P(A).

("$A \cap B$" trong hai công thức được hiểu là có cả A và B, không phải là phần chung của A và B)

Độ phổ biến nhỏ nhất (min_sup) và độ tin cậy nhỏ nhất (min_con) do người dùng nhập vào để tìm ra các luật có độ chính xác mà mình mong muốn.

Ví dụ: các bước thành lập cơ sở dữ liệu chuỗi và luật kết hợp:
Ta có cơ sở dữ liệu sau

Bảng 2.1 CSDL mẫu

Mã khách hàng	Ngày	ID mặt hàng được mua
1	1-1-2010	30
1	2-1-2010	90
2	3-1-2010	10, 20
2	5-1-2010	30
2	1-12-2009	40, 60, 70
3	4-2-2010	30, 50, 70
4	7-1-2010	30
4	3-3-2010	40, 70
4	12-5-2009	90
5	3-1-2009	90

Tiến hành sâu chuỗi dữ liệu theo mã khách hàng ta được

Bảng 2.2 CSDL dạng chuỗi

Mã khách hàng	Chuỗi
1	<(30)(90)>
2	<(10,20)(30)(40,60,70)>
3	<(30,50,70)>
4	<(30)(40,70)(90)>
5	<(90)>

Các bộ dữ liệu thỏa mức min_sup 40% là

Bảng 2.3 Bộ dữ liệu thỏa min_sup

Bộ dữ liệu	Độ phổ biến	Độ phổ biến (tính theo %)
(30)	4	80%
(40)	2	40%
(70)	3	60%
(40,70)	2	40%
(90)	3	60%

Xét luật kết hợp giữa (30) và (40) ta có:

Độ phổ biến (30 → 40) = 40%, độ tin cậy (30 → 40) = 50%

Vậy luật 30 → 40 có độ phổ biến = 40% và độ tin cậy = 50% có nghĩa sẽ có 50% khách hàng sau khi mua mặt hàng ID = 30 sẽ mua tiếp mặt hàng ID = 40.

Độ phổ biến (40 → 30) = 40%, độ tin cậy (40 → 30) = 100% có nghĩa toàn bộ khách hàng sau khi mua mặt hàng ID = 40 sẽ mua tiếp mặt hàng ID = 30.

Có nhiều loại luật kết hợp, loại ta đã xét ở trên là loại đơn chiều dạng "Boolean". Dạng này so khớp các mục dữ liệu trong chuỗi (True/False) và chỉ xét trên một khuynh hướng, ví dụ: sự tăng hay giảm của mã chứng khoán, thói quen mua hàng của khách hàng… Một số loại khác như luật kết hợp đa chiều định lượng, luật kết hợp phân cấp đơn, luật kết hợp đa phân cấp… Tuy nhiên, trong khóa luận nhóm chỉ trình bày luật kết hợp đơn chiều dạng "Boolean" như ở trên.

2.5.2. Khảo sát

Hướng khai thác mẫu chuỗi phổ biến được mở rộng từ việc khai thác bộ dữ liệu phổ biến. Dữ liệu sau khi sâu chuỗi lại sẽ có dạng <a,b,c,d,e,f> đối với chuỗi đơn giản chỉ có các bộ dữ liệu, khi bộ dữ liệu có thể chứa nhiều mục dữ liệu thì chuỗi có dạng phức tạp hơn như <(a)(b,d,c)(e)(f)>. Điểm khác biệt cơ bản giữa khai thác bộ dữ liệu và khai thác mẫu chuỗi là:

- Khai thác bộ dữ liệu dựa trên các dữ liệu không lồng nhau và không cần thứ tự giữa các bộ dữ liệu.
- Khai thác mẫu chuỗi dựa trên dữ liệu có thể lồng nhau, sắp xếp theo một thứ tự nhất định (tăng, giảm hoặc chữ cái hoặc theo một quy ước nào đó).

Ví dụ: khai thác bộ dữ liệu bao gồm các thuật toán như Apriori, FP-Growth, ECLAT…

<a,d,c,b> <=> <a,b,c,d> : mẫu <b,c> xuất hiện trong cả hai chuỗi.

Khai thác mẫu chuỗi bao gồm các thuật toán như GSP, Freespan, Prefixspan, Clospan…

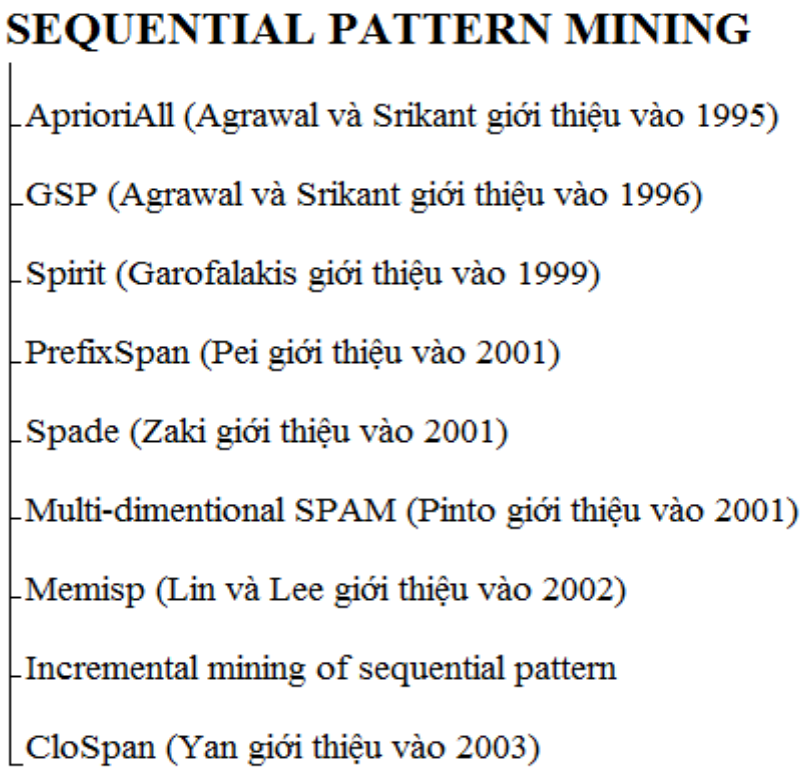
Trường hợp chỉ có chuỗi đơn: <a,b,c,d> != <a,d,c,b> : mẫu <b,c> chỉ xuất hiện một lần.

Trường hợp chuỗi lồng: <(a,b),b,c,d> : mục dữ liệu không có thứ tự, bộ dữ liệu có thứ tự.

Việc phân loại còn dựa trên cơ chế khai thác dữ liệu, ta có hai loại khai thác:

- Tạo ra các chuỗi ứng viên (theo hướng quét cạn): gồm có các thuật toán như Apriori, GSP, SPADE. Loại này nhằm ra tất cả ứng viên có thể có, sau đó kiểm tra độ phổ biến của chúng.
- Dùng cấu trúc cây để giảm bớt một số nhánh không cần thiết trong quá trình khai thác: gồm có các thuật toán như FP-Growth, Prefixspan, Clospan… Loại này đa số sử dụng tìm kiếm theo chiều sâu (DFS) để tối ưu việc giảm lược không gian tìm kiếm.

Quá trình phát triển của các thuật toán trong SPAM được thể hiện ở hình 2.8.



Hình 2.8 Quá trình phát triển của SPAM đến thời điểm năm 2003

AprioriAll là thuật toán đầu tiên để khai thác mẫu chuỗi. Dựa vào cách tiếp cận sơ khởi của luật kết hợp Apriori, thuật toán tạo ra chuỗi tất cả ứng viên và kiểm tra độ phổ biến của chúng. Chính vì vậy, thời gian để thực hiện phụ thuộc vào khối lượng CSDL và số lượng chuỗi dự kiến nên AprioriAll không thích hợp cho CSDL lớn, thuật toán chỉ là nền tảng để phát triển các thuật toán hiệu quả hơn.

Trước Clospan (2003) có nhiều dạng đã được phát triển như:

- Prefixspan: là thuật toán gốc của Clospan.
- GSP (cải tiến từ Apriori).
- SPADE khai thác dữ liệu chuỗi theo dạng dọc (dữ liệu dòng thành cột, cột thành dòng), trong khi các loại khác theo dạng ngang.
- "Constraint-based": dùng một số ràng buộc do người dùng tự định nghĩa để giảm bớt khối lượng dữ liệu cần khai thác và chỉ khai thác ra các mẫu mà người dùng đặc tả.
- "Multidimentional, Multilevel Sequential Pattern Mining": mở rộng các thuật toán khai thác cơ bản (Prefixspan, GSP…) bằng cách gắn thêm cột dữ liệu vào dữ liệu sequenceID và itemID một cách tách biệt, chẳng hạn gắn cột chuỗi các địa điểm bán vào seqID của một khách hàng, chuỗi các thông tin sản phẩm vào itemID.

- "Incremental Mining of Sequential Pattern": vì dữ liệu chuỗi thời gian thay đổi rất nhanh nên khi thay đổi thì tất cả các thuật toán phải khai thác lại. Phương pháp này giúp không phải khai thác lại toàn bộ CSDL khi không cần thiết. Gồm có:
  - GSP-based và MFS-based Incremental Mining (Zhang giới thiệu vào 2002).
  - SPADE-based Incremental Mining – ISM (Parthasarathy giới thiệu vào 1999).
  - ISE và IUS (MASSEGLIA giới thiệu vào 2000).
  - KISP (Lin và Lee giới thiệu vào 2003).

Jiawei Han là tác giả khá nổi tiếng sau hai tác giả kinh điển đặt ra vấn đề khai thác mẫu chuỗi là Agrawal & Srikant (1995), như site Microsoft Academic cũng đã sắp hạng tác giả này trên cả Srikant vì số lượng cũng như chất lượng bài viết khoa học đóng góp nhiều đáng kể so với các tác giả khác qua liên kết http://academic.research.microsoft.com/CSDirectory/Author_category_7.htm.

Vì vậy, phần từ năm 2003 trở đi nhóm sẽ phân thành hai mục: mục của tác giả Jiawei Han và mục của các tác giả khác.

Jiawei Han

Từ sau Clospan (2003), nảy sinh khái niệm "Maximal": chuỗi chứa chuỗi, tuy nhiên khái niệm "Close" bao gồm khái niệm này vì còn có thêm điều kiện là phải có cùng độ phổ biến. Vào thời điểm này có hai vấn đề được nêu ra: việc tìm các mẫu sẽ tạo ra rất nhiều mẫu con phổ biến nếu số lượng mẫu thỏa độ phổ biến nhiều, vấn đề này được giải quyết bằng Clospan: chỉ tìm các chuỗi đóng, với chuỗi đóng được định nghĩa là chuỗi không được chứa trong bất kỳ chuỗi nào khác có cùng độ phổ biến. Vấn đề thứ hai là chọn một độ phổ biến nhỏ nhất thích hợp, vì nếu để người dùng chọn, đòi hỏi phải có nhiều kinh nghiệm phân tích, chưa kể phải hiểu rõ về thuật toán, giá trị quá nhỏ sẽ tìm ra khối lượng mẫu khổng lồ, giá trị quá lớn sẽ ít các mẫu thỏa được độ phổ biến. Vì vậy, thuật toán TSP (ICDM 2003) đưa ra để giải quyết vấn đề thứ hai, thuật toán này còn được áp dụng kỹ thuật của Clospan để tìm các chuỗi đóng.

Đến 2004, thuật toán Bide đưa ra phương pháp mới hoàn toàn không cần dựa vào cấu trúc cây để giảm lược bước (Clospan đã dùng để tìm các chuỗi đóng) nên tiên tiến hơn Clospan, thuật toán có hai thiết kế phù hợp với hai đặc trưng loại dữ liệu là CSDL có chuỗi đơn (không có bộ dữ liệu) và CSDL có chuỗi lồng.

Tiếp đó, Par-CSP là thuật toán được giới thiệu trong "parallel Sequential Pattern Mining" (2005), chủ yếu dựa trên Bide nhưng được thiết kế tốt hơn (cách phân phối tài nguyên máy và giải phóng bộ nhớ), sao cho khai thác được phần cứng của máy hiệu quả và năm 2006 là một khung thiết kế được giới thiệu để giúp "Parallel Mining" nâng cao thêm hiệu năng.

Tiếp theo đó là cả một "series" các thuật toán giải quyết cho việc CSDL quá lớn (đơn vị hàng triệu dòng), chuỗi cực dài, làm sao để khai thác trong thời gian cho phép: tạo chỉ mục cho chuỗi trong CSDL (Seqindex - SDM 2005), khai thác các tập phổ biến đóng (VLDB 2005), và tạo ràng buộc nhất định (Constraint-based Pattern Growth - JIIS 2007). Ý tưởng tạo ra những chuỗi gần đúng trong trường hợp không thể khai

thác ra kết quả chính xác, ý tưởng này dùng "Core Pattern Fusion" (ICDE 2007), hay "Approximate Sequential Pattern" (ICDM 2007) cho trường hợp không phải khai thác lại CSDL lớn khi dữ liệu được cập nhật.

Một hướng phát triển khác, cũng nhằm mục đích khi CSDL cập nhật không cần khai thác lại là Incspan được giới thiệu tại KDD 2004, cải tiến của thuật toán này là Cispan (SDM 2008), tiếp đó là Flospan đang được phát triển.

Năm 2009, một khái niệm mới về mẫu được đưa ra là "Repetitive Gapped Subsequences", mẫu này không chỉ có thông tin về độ phổ biến đối với tất cả chuỗi trong CSDL mà còn có thêm thông tin về số lần xuất hiện trong một chuỗi cụ thể nào đó.

Qua các thuật toán của Jiawei Han về SPAM, có thể thấy hướng nghiên cứu chính là đề ra các giải pháp mới tổng quát về các vấn đề chưa giải quyết được như:

- CSDL quá lớn (hàng triệu dòng) và các thuật toán khai thác hiện tại không thể hoàn thành trong thời gian cho phép, hay khi dữ liệu thay đổi thì bất cập là phải khai thác lại toàn bộ CSDL vì đặc trưng của dữ liệu chuỗi thời gian là thay đổi rất nhanh, liên tục hằng ngày, hoặc hằng giờ và có thể nhanh hơn (dữ liệu chứng khoán, dữ liệu dòng…). Một trường hợp ngoại lệ nhỏ là khai thác chuỗi DNA để dự đoán các loại bệnh, đây cũng là dạng chuỗi nhưng các thành phần trong DNA có thể thay đổi chậm tùy bệnh.
- Ngoài các cách tiếp cận cũ để khai thác (đếm độ phổ biến, tạo chuỗi ứng viên như Apriori, GSP, hay hướng cấu trúc cây như FP-Growth, Prefixspan, Clospan) có thể tìm các tiếp cận khác hiệu quả hơn hay không?

Các giải pháp tổng quát này sau đó được nghiên cứu và ứng dụng vào những lĩnh vực cụ thể, cải tiến theo hướng riêng biệt đặc trưng của loại dữ liệu nhằm khai thác hiệu quả hơn như DNA, phân tích hành vi khách hàng, thị trường chứng khoán, dữ liệu dòng… Dữ liệu dòng là dữ liệu có vào có ra như dữ liệu ghi băng của vệ tinh, hay dữ liệu trong cảm biến của thiết bị.

Các tác giả khác

Rất nhiều ứng dụng của các thuật toán tổng quát vào từng lĩnh vực cụ thể như:

- Khai thác dữ liệu dòng: "Multiple Streams" (ICDM 2005), "A Centroid Approach" (JIIS 2006), "Efficient Frequent Pattern Mining Over Data Streams" (CIKM 2008).
- Dữ liệu không gian: "Mining of Positive and Negative Spatial Patterns" (SDM (SIAMDM) 2005), "Closed Partial Orders" (SDM (SIAMDM) 2005).
- Dữ liệu không chắc chắn: "Vertical Mining of Frequent Patterns from Uncertain Data" (JCIS 2009), "Frequent Pattern Mining with Uncertain Data" (ACM SIGKDD 2009), "Efficient algorithms for mining constrained frequent patterns from uncertain data" (ACM SIGKDD Workshop 2009).
- Dữ liệu trong sinh học: "Scalable Sequential Pattern Mining for Biological Sequences" *(CIKM 2004).*

- Dữ liệu trong giáo dục: "Using Sequential Pattern Mining for Links Recommendation in Adaptive Hypermedia Educational Systems" (Current Developments in Technology-Assisted Education 2006).
- Dữ liệu Web Log: "Mining Constraint-based Multidimensional Frequent Sequential Pattern in Web Logs" (European Journal of Scientific Research (EJSR) 2009 volume 36 issue 3).
- Khai thác dữ liệu trong kho chứa (OLAP): "HYPE: Mining Hierarchical Sequential Pattern Mining" (ACM international workshop on Data warehousing and OLAP 2006).

Tuy nhiên cũng có hướng cải tiến cho ra các thuật toán hiệu quả hơn từ các thuật toán tổng quát:

- Hai thuật toán Bitmap-based GSP và SM-tree cải tiến từ GSP (Efficient Sequential Pattern Mining Algorithms (4th WSEAS 2005)).
- Thuật toán LAP-IN được giới thiệu với hiệu năng cao hơn Prefixspan, SPADE, đặc biệt hiệu quả khi khai thác CSDL dày (hội thảo ICDE 2005), sau đó được trình bày lại tại DEWS 2006 và DASFAA 2007.
- COBRA: thuật toán này được đưa ra tại (DaWak 2006), chứng minh hiệu năng tốt hơn Bide.
- Thuật toán PAID được giới thiệu tại hội nghị IDEAS 2006, cho thấy hiệu năng cao hơn đối với Prefixspan, Bide và Lapin bằng phương pháp hạn chế quá trình đếm độ phổ biến càng nhiều càng tốt.
- CMDS: thuật toán kết hợp khai thác bộ dữ liệu phổ biến (mới nhất là DCI-CLOSE) và khai thác mẫu chuỗi phổ biến (mới nhất là Bide) giới thiệu tại ITNG 2006 (Closed Multidimensional Sequential Pattern Mining).
- IMCS: giới thiệu vào hội nghị "On Advances In Data and Web Management" (2007), hiệu năng cao hơn Prefixspan, Clospan, Bide và Incspan.
- PRISM: giới thiệu tại ICDM 2007 và hiệu năng cao hơn SPADE, Prefixspan. Thuật toán này được trình bày lại tại JCSS 2009 và 2010.
- Thuật toán GenMiner và GenMiner-EQ được giới thiệu tại SDM 2008, so sánh hiệu năng với Prefixspan, Clospan, Bide. Thuật toán này hơn Bide trong một số trường hợp, tuy nhiên kém hơn các trường hợp khác.
- "Margin-Closed Sequential Pattern Mining": được giới thiệu tại hội thảo KDD 2010-ACM SIGKDD 2010, sử dụng cơ chế của Bide nhưng sau đó sẽ hợp các mẫu lại theo cơ chế "Margin".
- Dùng lý thuyết tập thô để khai thác dữ liệu chuỗi, cách này có thể khai thác cả các mẫu chu kỳ (một hướng khác của khai thác dữ liệu chuỗi thời gian) và tìm các mẫu nửa chu kỳ ("A Sequential Pattern Mining Using Rough Set Theory 2010).

2.5.3. Thuật toán được trình bày

Trong khóa luận này, nhóm trình bày Apriori và Bide. Vì Bide có thể được gọi là "chuẩn" chung từ 2004. "Chuẩn" nghĩa là hầu như các thuật toán từ sau đó đến nay trong lĩnh vực liên quan đều được cải tiến từ Bide, và một số các hướng cải tiến này chưa được so sánh với nhau. Cũng tương tự như năm 1995 khi Agrawal & Srikant đề ra AprioriAll, đây là nền tảng để các thuật toán sau cải tiến.

Bide có hai hướng thiết kế, hướng chuỗi đơn và hướng chuỗi lồng. Hướng chuỗi đơn là hướng rất thích hợp với ứng dụng trong thị trường chứng khoán của nhóm vì CSDL sau khi chuyển hoá sang dạng chuỗi thì hầu hết các view mà nhóm dùng để khai thác chỉ chứa chuỗi đơn.

2.6. Thuật toán Apriori [3]

Mô tả thuật toán

- Tìm các bộ dữ liệu thỏa độ phổ biến.
- Sinh luật kết hợp từ các tập con của các bộ phổ biến đó và tính toán độ tin cậy.

Minh hoạ

Mã giả của thuật toán tại bước 1:
Dữ liệu đầu vào: cơ sở dữ liệu D đã chuyển thành chuỗi, độ phổ biến (min_sup).
Dữ liệu trả về: tập L các bộ dữ liệu phổ biến trong D.

Phương thức chính

```
1  L1 = tìm bộ dữ liệu thỏa min_sup (D);
2  for (k =2; Lk-1 ≠ Ø; k++) {
3    Ck = Apriori(Lk-1, min_sup);
4    for each giao dịch t  ∈ D { //quét D
5    for each ứng viên c  ∈ Ct
6              c.count++;
7    }
8    Lk = {c ∈ Ck | c.count ≥ min_sup}
9  }
10 return L = UkLk;
```

Phương thức Apriori(L_{k-1}: bộ dữ liệu thứ (k-1) thỏa min_sup, min_sup)

```
1  for each bộ dữ liệu l1 ∈ Lk-1
2    for each bộ dữ liệu l2 ∈ lk-1
3    if ((l1[1] = l2[1]) và (l1[2] = l2[2] và… và (l1[k-2] = l2[k-2] và (l1[k-1] < l2[k-1])) then {
4                              c = l1 kết l2;
5                              if tồn tại bộ con không thỏa min_sup (c, Lk-1) then
6                              delete c;
7                              else add c to Ck;
8                              }
9  return Ck;
```

Phương thức tồn tại bộ con không thỏa min_sup (c: ứng viên bộ dữ liệu thứ k, L_{k-1}: các bộ dữ liệu thứ (k-1) thỏa min_sup)

```
1  for each bộ con thứ (k-1) của c
2    ff s ∉ L(k-1) then
3              return TRUE;
4  return FALSE;
```

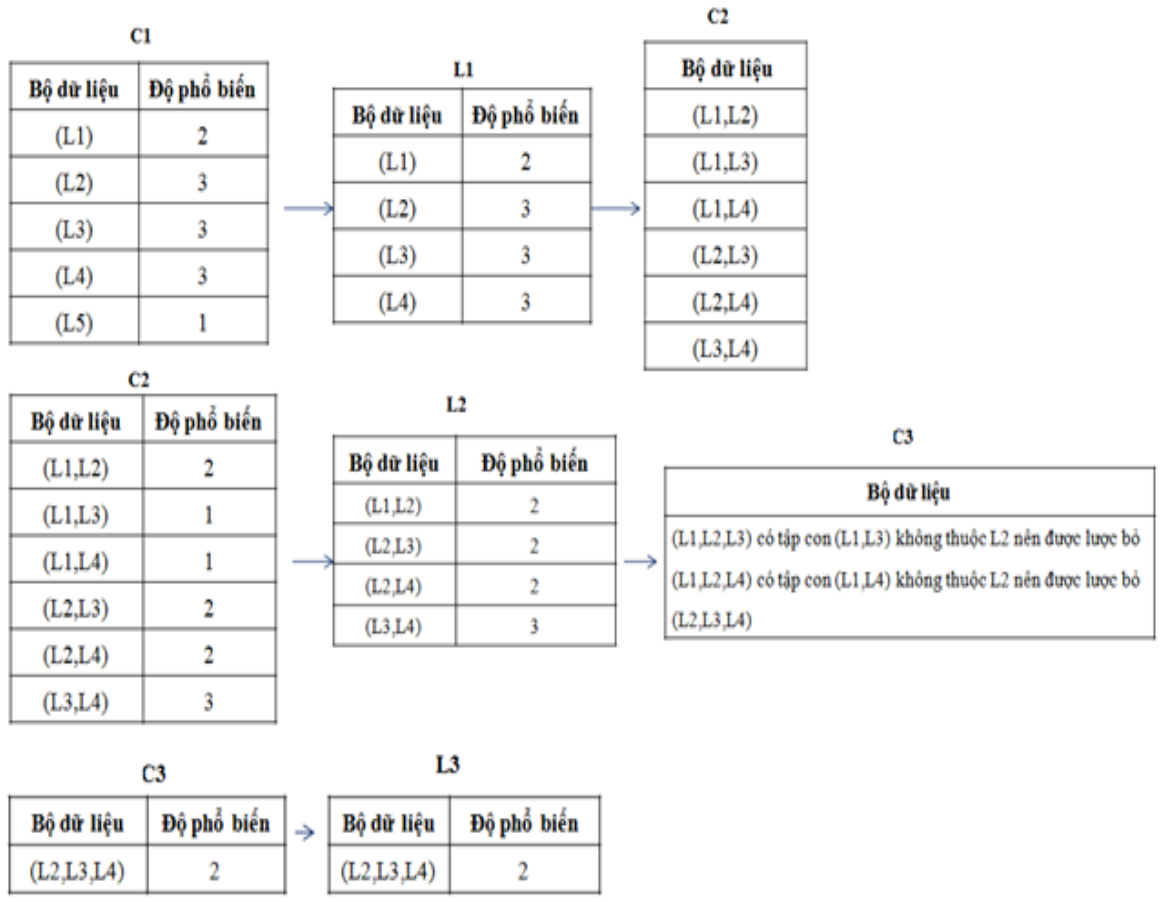
Ta có CSDL như bảng 2.4.

Bảng 2.4 CSDL chuỗi D

TID	Danh sách bộ dữ liệu
T1	L1, L2, L5
T2	L2, L3, L4
T3	L3, L4
T4	L1,L2, L3, L4

Với độ phổ biến nhỏ nhất là 50%. Kí hiệu "Cx" chỉ các chuỗi ứng viên có chiều dài x, ký hiệu "Lx" chỉ các bộ dữ liệu thỏa độ phổ biến có chiều dài x, với x là số nguyên và $x \geq 1$.

Bước tìm bộ dữ liệu được thể hiện ở hình 2.9.



Hình 2.9 Bước tìm bộ dữ liệu thỏa độ phổ biến

Bước tìm luật:

Các tập con s không rỗng của bộ dữ liệu S (L2,L3,L4) là:

{ (L2,L3), (L2,L4), (L3,L4), (L2), (L3), (L4)}

Ta có các luật s → (S – s) với độ tin cậy = độ phổ biến (S)/ độ phổ biến(s)

- (L2,L3) → L4 độ tin cậy = 2/2 = 100%
- (L2,L4) → L3 độ tin cậy = 2/2 = 100%
- (L3,L4) → L2 độ tin cậy = 2/3 = 67 %
- L2 → (L3,L4) độ tin cậy = 2/3 = 67%
- L3 → (L2,L4) độ tin cậy = 2/3 = 67%
- L4 → (L2,L3) độ tin cậy = 2/3 = 67%

2.7. Thuật toán Bide [10]

Mô tả thuật toán

Bide có dữ liệu đầu vào và bước tìm luật giống như Apriori, điểm khác biệt là Bide chỉ tìm các chuỗi đóng trong khi Apriori tìm cả chuỗi đóng và không đóng. Chuỗi đóng được định nghĩa là chuỗi không bị chuỗi nào khác chứa và có cùng độ phổ biến. Vì vậy Bide có hiệu năng cao hơn Apriori, các luật của Bide sẽ ít hơn của Apriori nhưng chất lượng cao hơn vì chúng bao gồm thông tin luật của các chuỗi không đóng.

Ta có chuỗi S chứa tiền tố e (e có thể có chiều dài-1: e; hoặc chiều dài-i: $e_1e_2…e_i$).

Định nghĩa 1: First instance of a prefix sequence (FI)

Được tính từ đầu chuỗi S đến lần xuất hiện đầu tiên của e.

Ví dụ:
C A B C có tiền tố A B, FI[A B] = C A B.
C B A A B có tiền tố A B, FI[A B] = C B A A B.
A B D có tiền tố A B, FI[A B] = A B.
A E F B A B A có tiền tố A B, FI[A B] = A E F B.

Định nghĩa 2: Projected sequence of a prefix sequence (PS)

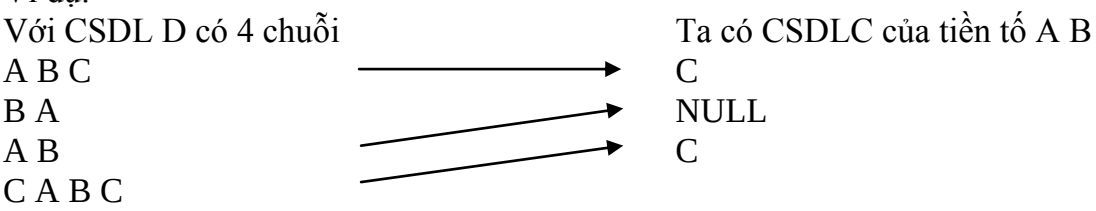
Được tính từ sau FI[e] đến hết chuỗi S. PS được dùng làm chuỗi trong CSDL chiếu.

Ví dụ:
A B D có tiền tố A B, PS = D
C A A B C có tiền tố A B, PS = C.
A E F B A B A có tiền tố A B, PS = A B A.
C A A B có tiền tố A B, PS = NULL.

Định nghĩa 3: Projected database of a prefix sequence

Tập hợp các PS lấy từ CSDL ban đầu ứng với từng tiền tố, được gọi là CSDL chiếu (CSDLC).

Ví dụ:



Định lý 1: BI-Directional Extension closure checking

Nếu một tiền tố e không có FEI và BEI thì e chính là chuỗi đóng, còn lại thì e là chuỗi không đóng.

Đây chính là bước BEI + FEI = 0 của mã giả.

Bổ đề 1: Forward-extension event checking (FEI)

Các mục dữ liệu trong tập FEI của một tiền tố e chính là các mục trong tập LFI của tiền tố e đó thỏa điều kiện độ phổ biến$^{CSDLC[e]}$[mục dữ liệu $\in$ LFI] = độ phổ biến [e]. Với LFI (Local Frequent Item) là những mục dữ liệu nằm trong CSDLC của e và có độ phổ biến $\geq$ độ phổ biến nhỏ nhất.

Ví dụ:
Độ phổ biến nhỏ nhất là 2.
Tiền tố A có độ phổ biến là 4. CSDLC A = {C F, C B, B D C, C B D }
LFI = {B:3, C:4, D:2} → FEI có một mục dữ liệu là {C:4}.

Định nghĩa 4: Last instance of a prefix sequence (LI)

Được tính từ đầu S đến lần xuất hiện sau cùng của phần tử cuối của e.

Ví dụ:
C A A B C có tiền tố A B, LI[A B] = C A A B.
A B A D có tiền tố A B, LI[A B] = A B.
B A B có tiền tố A B, LI[A B] = B A B.
A E F B A B A có tiền tố A B, LI[A B] = A E F B A B.

Định nghĩa 5: The i-th last-in-last appearance w.r.t. a prefix sequence (LL_i)

Với i = n: LL_i là lần xuất hiện sau cùng của e_i trong LI của tiền tố e.
Với 1<= i < n: LL_i là lần xuất hiện sau cùng của e_i trong LI của tiền tố e và LL_i phải xuất hiện trước LL_{i+1}.

Ví dụ:
Ta có LI[A B] = A B C B A B. Vậy LL_1 = A(thứ 2), LL_2 = B(thứ 3).
Ta có LI[A B C] = A B C B A C C. Vậy LL_1 = A(đầu tiên), LL_2 = B(thứ 2), LL_3 = C(thứ 3).
Ta có LI[A] = A B C B A. Vậy LL_1 = A(thứ 2).

Định nghĩa 6: The i-th maximum period of a prefix sequence (Max_i)

Với 1 < i <= n: Max_i là phần chuỗi nằm giữa $FI[e_{i-1}]$ và LL_i.
Với i = 1: Max_i là phần chuỗi S đứng trước LL_i.

Ví dụ:

LI[A B] = A B C B A B; LL_1 = A(thứ 2), Max_1 = A B C B;
LL_2 = B(thứ 3), FI[A] = A, Max_2 =B C B A.
LI[A B C] = A B C B A C C; LL_1 = A(đầu tiên), Max_1 = Ø;
LL_2 = B(thứ 2), FI[A] = A, Max_2 = B C;
LL_3 = C (thứ 3), FI[A B] = A B, Max_3 = C B A C.
LI[A] = A B C B A; LL_1 = A(thứ 2), Max_1 = A B C B.

Bổ đề 2: Backward-extension event checking (BEI)

Nếu tồn tại i (1 <= i <= n) mà có mục dữ liệu e' trong mỗi chuỗi của CSDLC e ứng với MAX_i, e' được ghi nhận là BEI của e.

Định nghĩa 7: The i-th last-in-first appearance w.r.t. a prefix sequence (LF_i)
(tương tự như LL_i, nhưng xét trong FI)

Với i = n: LF_i là lần xuất hiện sau cùng của e_i trong FI của tiền tố e.
Với 1<= i < n: LF_i là lần xuất hiện sau cùng của e_i trong FI của tiền tố e và LF_i phải xuất hiện trước LF_{i+1}.

Định nghĩa 8: The i-th semi-maximum period of a prefix sequence ($Semi_i$)
(tương tự như Max_i, nhưng xét với LF_i)

Với 1 < i <= n: Max_i là phần chuỗi nằm giữa $FI[e_{i-1}]$ và LF_i.
Với i = 1: Max_i là phần chuỗi S đứng trước LF_i.

Định lý 2: BackScan search space pruning

Nếu tồn tại i (1 <= i <= n) mà có mục dữ liệu e' trong mỗi chuỗi của CSDLC e ứng với $Semi_i$, dừng tìm kiếm tiền tố e_i và sang nhánh khác.

ScanSkip Technique

Áp dụng cho BEI và BackScan, trong quá trình tìm Semi (BackScan) hay Max(BEI), nếu xuất hiện Ø tại i-th nào đó của một FI(hay LI), thì toàn bộ i-th đó của các FI tiếp theo (hay LI tiếp theo) sẽ không cần tìm kiếm vì đã có Ø thì tất cả LI (hay FI) không thể có chung mục dữ liệu tại i-th đó được.

Minh họa

BIDE (CSDL, min_sup, chuoi_dong)
Đầu vào: CSDL và ngưỡng độ phổ biến nhỏ nhất (min_sup).
Đầu ra: tập các chuỗi đóng.

1: chuoi_dong = Ø;
2: F1 = chuoi_chieudai-1_phobien (CSDL, min_sup);
3: for (mỗi chuoi_chieudai-1 f1 trong F1)
4: $CSDLC^{f1}$ = chieu_gia (CSDL);
5: for (mỗi f1 trong F1)
6: if (!BackScan(f1, $CSDLC^{f1}$))
7: BEI = backward_extension_check (f1, $CSDLC^{f1}$);
8: gọi đệ quy bide ($CSDLC^{f1}$, f1, min_sup, BEI, chuoi_dong);
9: return chuoi_dong;

Bide ($CSDLC^{f1}$, f1, min_sup, BEI, chuoi_dong)

Đầu vào: CSDLC của f1, f1, ngưỡng min_sup, giá trị BEI.
Đầu ra: tập các chuỗi đóng.

1: LFI = mucdulieu_phobien_cucbo ($CSDLC^{f1}$);
2: FEI = { z trong LFI | z.đo_pho_bien = độ pho_bien^{CSDLC}(f1) };
3: if (BEI + FEI) == 0
4: chuoi_dong = chuoi_dong U f1;
5: for (mỗi i trong LFI)
6: $f1^i$ = <f1, i>;
7: $CSDLC^{f1i}$ = chieu_gia ($CSDL^{f1}$, $f1^i$);
8: for (mỗi i trong LFI)
9: if (!BackScan($f1^i$, $CSDLC^{f1i}$))
10: BEI = backward_extension_check ($f1^i$, $CSDLC^{f1i}$);
11: Gọi đệ quy bide ($CSDLC^{f1i}$, $f1^i$, min_sup, BEI, chuoi_dong);

Ta có CSDL D như bảng 2.5.

Bảng 2.5 CSDL D

SID	**S**
1	C A A B C
2	A B C B
3	C A B C
4	A B B C A

Min_sup = 2. Quét D ta có các chuỗi có chiều dài-1 thỏa min_sup:

Bảng 2.6 Các chuỗi có chiều dài-1 thỏa min_sup

A	4
B	4
C	4

Tạo CSDL chiếu cho A, B, C ta có:

Bảng 2.7 CSDLC của A, B và C

A:4
A B C
B C B
B C
B B C A

B:4
C
C B
C
B C A

C:4
A A B C
B
A B C
A

Xét A:4

Thực hiện BackScan:

Bảng 2.8 BackScan của A

FI	**LF_1**	**$Semi_1$**
C A	A	C
A	A	Ø → ScanSkip

Không thỏa BackScan, thực hiện BEI:

Bảng 2.9 BEI của A

LI	**LL_1**	**Max_1**
C A A	A(thứ 2)	C A
A	A	Ø → ScanSkip

BEI = 0.

Thực hiện LFI:

Bảng 2.10 LFI của A

A	2
B	4
C	4

Thực hiện FEI:

Bảng 2.11 FEI của A

B	4
C	4

FEI = 2.

BEI + FEI = 0 + 2 = 2.

Thực hiện CSDL chiếu cho AA:2, AB:4, AC:4.

Bảng 2.12 CSDLC của AA, AB và AC

AA:2
B C
NULL

AB:4
C
C B
C
B C A

AC:4
NULL
B
NULL
A

Xét AA:2

Thực hiện BackScan:

Bảng 2.13 BackScan của AA

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[A]**	**$Semi_2$**
C A A	A(đầu tiên)	C	A(thứ 2)	C A	Ø → ScanSkip
A B B C A	A(đầu tiên)	Ø → ScanSkip			

Không thỏa BackScan, thực hiện BEI:

Bảng 2.14 BEI của AA

LI	**LL_1**	**Max_1**	**LL_2**	**FI[A]**	**Max_2**
C A A	A(đầu tiên)	C	A(thứ 2)	C A	Ø → ScanSkip
A B B C A	A(đầu tiên)	Ø → ScanSkip			

BEI = 0.

Thực hiện LFI: LFI = Ø.

Thực hiện FEI: FEI = 0.

BEI + FEI = 0 → Xuất AA:2.

Xét AB:4

Thực hiện BackScan:

Bảng 2.15 BackScan của AB

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[A]**	**$Semi_2$**
C A A B	A(thứ 2)	C A	B	C A	A
A B	A	Ø → ScanSkip	B	A	Ø → ScanSkip

Không thỏa BackScan, thực hiện BEI:

Bảng 2.16 BEI của AB

LI	**LL_1**	**Max_1**	**LL_2**	**FI[A]**	**Max_2**
C A A B	A(thứ 2)	C A	B	C A	A
A B C B	A	Ø → ScanSkip	B(thứ 2)	A	B C
C A B			B	C A	Ø → ScanSkip

BEI = 0.

Thực hiện LFI:

Bảng 2.17 LFI của AB

B	2
C	4

Thực hiện FEI:

Bảng 2.18 FEI của AB

C	4

FEI = 1.

BEI + FEI = 0 + 1 = 1.

Thực hiện CSDL chiếu cho ABB: 2, ABC: 4.

Bảng 2.19 CSDLC của ABB, ABC

ABB:2

NULL
CA

ABC:4

NULL
B
NULL
A

Xét ABB:2

Thực hiện BackScan:

Bảng 2.20 BackScan của ABB

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[A]**	**$Semi_2$**	**LF_3**	**FI[AB]**	**$Semi_3$**
A B C B	A	Ø → ScanSkip	B(đầu tiên)	A	Ø → ScanSkip	B(thứ 2)	A B	C
A B B						B(thứ 2)	A B	Ø

Không thỏa BackScan, thực hiện BEI:

Bảng 2.21 BEI của ABB

LI	**LL_1**	**Max_1**	**LL_2**	**FI[A]**	**Max_2**	**LL_3**	**FI[AB]**	**Max_3**
A B C B	A	Ø → ScanSkip	B(đầu tiên)	A	Ø → ScanSkip	B(thứ 2)	A B	C
A B B						B(thứ 2)	A B	Ø

BEI = 0.

Thực hiện LFI: LFI = Ø.

Thực hiện FEI: FEI = 0.

BEI + FEI = 0 → xuất ABB:2.

Xét ABC:4

Thực hiện BackScan:

Bảng 2.22 BackScan của ABC

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[A]**	**$Semi_2$**	**LF_3**	**FI[AB]**	**$Semi_3$**
C A A B C	A(thứ 2)	C A	B	C A	A	C(thứ 2)	C A A B	Ø → ScanSkip
A B C	A	Ø → ScanSkip	B	A	Ø → ScanSkip			

Không thỏa BackScan, hiện BEI:

Bảng 2.23 BEI của ABC

LI	**LL_1**	**Max_1**	**LL_2**	**FI[A]**	**Max_2**	**LL_3**	**FI[AB]**	**Max_3**
C A A B C	A(thứ 2)	C A	B	C A	A	C(thứ 2)	C A A B	Ø → ScanSkip
A B C	A	Ø → ScanSkip	B	A	Ø → ScanSkip			

BEI = 0.

Thực hiện LFI: LFI = Ø.

Thực hiện FEI: FEI = 0.

BEI + FEI = 0 → xuất ABC:4.

Xét AC:4

Thực hiện BackScan:

Bảng 2.24 BackScan của AC

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[A]**	**$Semi_2$**
C A A B C	A(thứ 2)	C A	C(thứ 2)	C A	A B
A B C	A	Ø → ScanSkip	C	A	B
C A B C			C(thứ 2)	C A	B
A B B C			C	A	B B

Thỏa BackScan($Semi_2$ B).

Xét B:4

Thực hiện BackScan:

Bảng 2.25 BackScan của B

FI	**LF_1**	**$Semi_1$**
C A A B	B	C A A
A B	B	A
C A B	B	C A
A B	B	A

Thỏa BackScan($Semi_1$ A).

Xét C:4

Thực hiện BackScan:

Bảng 2.26 BackScan của C

FI	**LF_1**	**$Semi_1$**
C	C	Ø → ScanSkip

Không thỏa BackScan, thực hiện BEI:

Bảng 2.27 BEI của C

LI	**LL_1**	**Max_1**
C A A B C	C(thứ 2)	C A A B
A B C	C	A B
C A B C	C(thứ 2)	C A B
A B B C	C	A B B

BEI = 2.

Thực hiện LFI:

Bảng 2.28 LFI của C

A	3
B	3
C	2

Thực hiện FEI: FEI = 0.

BEI + FEI = 2 + 0 = 2.

Thực hiện CSDL chiếu cho CA:3, CB:3, CC:2.

Bảng 2.29 CSDLC của CA, CB, CC

CA:3

A B C
B C
NULL

CB:3

C
NULL
C

CC:2

NULL
NULL

Xét CA:3

Thực hiện BackScan:

Bảng 2.30 BackScan của CA

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[C]**	**$Semi_2$**
C A	C	Ø → ScanSkip	A	C	Ø → ScanSkip

Không thỏa BackScan, thực hiện BEI:

Bảng 2.31 BEI của CA

LI	**LL_1**	**Max_1**	**LL_2**	**FI[C]**	**Max_2**
C A A	C	Ø → ScanSkip	A(thứ 2)	C	A
C A			A	C	Ø → ScanSkip

BEI = 0.

Thực hiện LFI:

Bảng 2.32 LFI của CA

B	2
C	2

Thực hiện FEI: FEI = 0.

BEI + FEI = 0 → xuất CA:3.

Thực hiện CSDL chiếu cho CAB:2, CAC: 2.

Bảng 2.33 CSDLC của CAB, CAC

CAB:2
C
C

CAC:2
NULL
NULL

Xét CAB:2

Thực hiện BackScan:

Bảng 2.34 BackScan của CAB

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[C]**	**$Semi_2$**	**LF_3**	**FI[CA]**	**$Semi_3$**
C A A B	C	Ø → ScanSkip	A(thứ 2)	C	A	B	C A	A
C A B			A	C	Ø	B	C A	Ø

Không thỏa BackScan, thực hiện BEI:

Bảng 2.35 BEI của CAB

LI	**LL_1**	**Max_1**	**LL_2**	**FI[C]**	**Max_2**	**LL_3**	**FI[CA]**	**Max_3**
C A A B	C	Ø → ScanSkip	A(thứ 2)	C	A	B	C A	A
C A B			A	C	Ø	B	C A	Ø

BEI = 0

Thực hiện LFI:

Bảng 2.36 LFI của CAB

C	2

Thực hiện FEI: FEI = 1.

BEI + FEI = 0 + 1 = 1.

Thực hiện CSDL chiếu cho CABC:2.

Bảng 2.37 CSDLC của CABC

NULL
NULL

Xét CABC:2

Thực hiện BackScan:

Bảng 2.38 BackScan của CABC

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[C]**	**$Semi_2$**	**LF_3**	**FI[CA]**	**$Semi_3$**	**LF_4**	**FI[CAB]**	**$Semi_4$**
C A A B C	C(đầu tiên)	Ø → ScanSkip	A(thứ 2)	C(đầu tiên)	A(đầu tiên)	B	C A	A (thứ 2)	C(thứ 2)	C A A B	Ø → ScanSkip
C A B C			A	C(đầu tiên)	Ø	B	C A	Ø			

Không thỏa BackScan, thực hiện BEI:

Bảng 2.39 BEI của CABC

LI	**LL_1**	**Max_1**	**LL_2**	**FI[C]**	**MAX_2**	**L_{L_3}**	**FI[CA]**	**MAX_3**	**LL_4**	**FI[CAB]**	**MAX_4**
CAABC	C(đầu tiên)	Ø → ScanSkip	A(thứ 2)	C(đầu tiên)	A(đầu tiên)	B	C A	A (thứ 2)	C(thứ 2)	C A A B	Ø → Scan Skip
CABC			A	C(đầu tiên)	Ø	B	C A	Ø			

BEI = 0.

Thực hiện LFI: LFI = Ø.

Thực hiện FEI: FEI = 0.

BEI + FEI = 0 → xuất CABC:2.

Xét CAC:2

Thực hiện BackScan:

Bảng 2.40 BackScan của CAC

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[C]**	**$Semi_2$**	**LF_3**	**FI[CA]**	**$Semi_3$**
C A A B C	C(đầu tiên)	Ø → ScanSkip	A(thứ 2)	C	A(đầu tiên)	C(thứ 2)	C A	A B
C A B C			A	C	Ø	C(thứ 2)	C A	B

Thỏa BackScan($Semi_3$ B).

Xét CB:3

Thực hiện BackScan:

Bảng 2.41 BackScan của CB

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[C]**	**$Semi_2$**
C A A B	C	Ø → ScanSkip	B	C	A A
A B C B			B(thứ 2)	A B C	Ø → ScanSkip

Không thỏa BackScan, thực hiện BEI:

Bảng 2.42 BEI của CB

LI	**LL_1**	**Max_1**	**LL_2**	**FI[C]**	**Max_2**
C A A B	C	Ø → ScanSkip	B	C	A A
A B C B			B(thứ 2)	A B C	Ø → ScanSkip

BEI = 0.

Thực hiện LFI:

Bảng 2.43 LFI của CB

C	2

Thực hiện FEI: FEI = 0.

BEI + FEI = 0 → xuất CB:3.

Thực hiện CSDL chiếu cho CBC:2.

Bảng 2.44 CSDLC của CBC

NULL
NULL

Xét CBC:2

Thực hiện BackScan:

Bảng 2.45 BackScan của CBC

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[C]**	**$Semi_2$**	**LF_3**	**FI[CB]**	**$Semi_3$**
C A A B C	C(đầu tiên)	Ø → ScanSkip	B	C	A A	C(thứ 2)	C A A B	Ø → ScanSkip
C A B C			B	C	A			

Thỏa BackScan($Semi_2$ A).

Xét CC:2

Thực hiện BackScan:

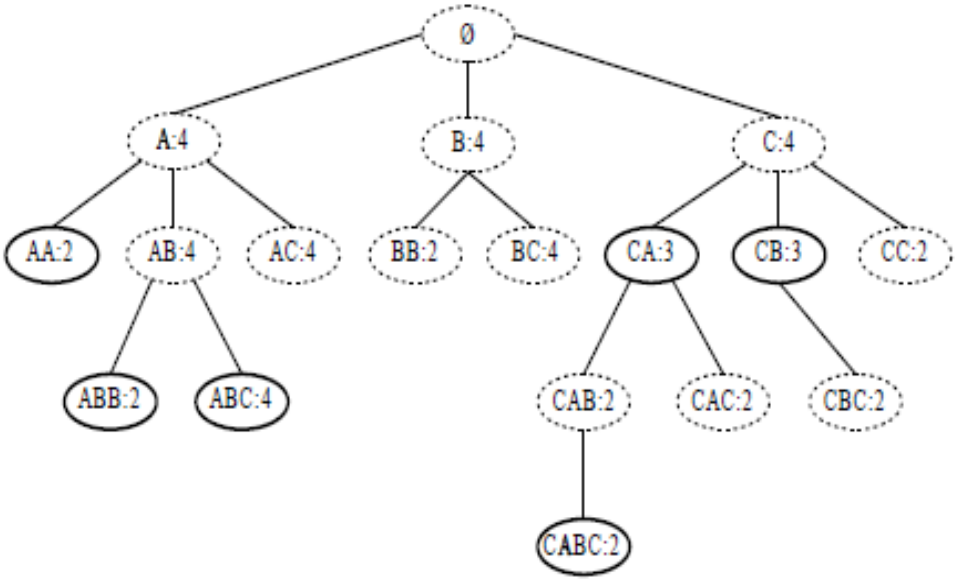
Bảng 2.46 BackScan của CC

FI	**LF_1**	**$Semi_1$**	**LF_2**	**FI[C]**	**$Semi_2$**
C A A B C	C(đầu tiên)	Ø → ScanSkip	C(thứ 2)	C	A A B
C A B C			C(thứ 2)	C	A B

Thỏa BackScan($Semi_2$ A, $Semi_2$ B).

Kết thúc.

Như vậy ta xuất được FCS = {AA:2, ABB:2, ABC:4, CA: 3, CABC:2, CB:3}.



Hình 2.10 Tập các chuỗi đóng sau khi sử dụng Bide

Như ta thấy:

ABC là mẫu cha của A, AB, AC, B, BC, C.

ABB là mẫu cha của BB.

CABC là mẫu cha của CC, CAB, CAC, CBC.

Ràng buộc của thuật toán

- BackScan chạy i-th từ 1 đến mục dữ liệu nào của tiền tố mà tất cả chuỗi trong CSDL chiếu đã có chung Semi thì sẽ dừng không cần tìm tiếp vì đã thỏa điều kiện BackScan.
- BackScan dùng để phát hiện chuỗi không đóng, vì vậy nếu thỏa BackScan thì không cần tìm tiếp chuỗi đó mà sẽ chuyển nhánh kế tiếp.
- BEI chạy i-th từ 1 đến mục dữ liệu nào của tiền tố mà tất cả chuỗi trong CSDL chiếu đã có chung Max thì sẽ dừng, không cần tìm tiếp vì BEI > 0, tiếp tục thực hiện LFI, không cần tìm FEI vì BEI + FEI > 0 và chuỗi đó là chuỗi không đóng.
- Chỉ chạy FEI khi BEI = 0 và khi phát hiện FEI > 0 thì dừng không cần tìm tiếp.
- Check ScanSkip cho BackScan và BEI.
- Tất cả CSDL chiếu của các tiền tố nên được xây dựng bằng chiếu giả (trỏ đến vị trí của tiền tố ứng với từng chuỗi, vậy một CSDL chiếu thay vì lưu 4 chuỗi thì sẽ lưu 4 trỏ).
- Chỉ cần thấy FI (hoặc LI) = Ø thì tất cả i-th của Semi (hoặc Max) đều là Ø. Trên thực tế, điều kiện này dường như là bất khả thi, vì một tiền tố có CSDL

chiếu thì sẽ có FI, và đã có FI thì chắc chắn có LI. Với trường hợp các chuỗi đơn giản ít lặp, LI chính là FI vì LI chỉ xuất hiện một lần và không lặp, dẫn đến BackScan và BEI giống nhau nên bị thừa.

- Trong quá trình BackScan và BEI đều dùng CSDL chiếu nhưng các chuỗi thì là chuỗi CSDL ban đầu, vì vậy giá trị NULL vẫn tính để dùng CSDL ban đầu, chỉ có Ø là bỏ qua.
- Với các tiền tố chiều dài-1 thì với (Semi, Max) i = 1 và với (LF, LL) i = n.
- Trong quá trình BackScan:
  1. Khi tìm các i-th LF, Semi thì các i-th độc lập với nhau nên ta có thể chạy i-th LF đến i-th Semi rồi (i+1)-th LF đến (i+1)-th Semi hoặc chạy hết i-th LF rồi chạy hết i-th Semi đều được (chạy lần lượt hoặc song song đều được).
  2. Khi tìm FI, ta phải làm lần lượt: FI của chuỗi thứ nhất trong CSDLC (sử dụng chuỗi CSDL gốc tương ứng), i-th LF, i-th Semi; đến FI của chuỗi thứ hai…. Không thể làm hết FI rồi mới tìm i-th LF, i-th Semi. Đây là ứng dụng của ScanSkip, nếu xuất hiện Ø tại i-th Semi của FI nào thì dừng ngay, không cần tìm FI kế tiếp.

  Tương tự cho BEI(với LI, i-th LL, Max).

2.8. Khai thác dữ liệu trong MSSQL Analysis Services

Các phiên bản mới đây khoảng từ năm 2005 trở lại của bộ công cụ Visual Studio và MSSQL do Microsoft phát triển hỗ trợ khai thác dữ liệu rất mạnh. Các thuật toán trong Analysis Services hầu hết đều dùng cơ chế của thuật toán nền tảng, kinh điển rải đều trong những chức năng của khai thác dữ liệu. Ví dụ như về lĩnh vực liên quan đến khóa luận của nhóm, Analysis Services có cơ chế thuật toán Apriori trong Microsoft Association Rule, cơ chế dự báo ARIMA trong Microsoft Time Series. Bên cạnh đó thì Microsoft còn điều chỉnh về kỹ thuật khai thác, cách lưu trữ để tối ưu thuật toán và cho phép người dùng có thể tích hợp thuật toán bất kỳ vào Analysis Services để sử dụng và so sánh hiệu năng. Ưu điểm của điều này là nếu ta tự xây dựng thuật toán sẽ cần nhiều kiến thức và mất nhiều thời gian về cách lưu trữ cấu trúc dữ liệu và khai báo thế nào cho hợp lý, tối ưu hoá cơ chế, phân phối tài nguyên máy tính, kết nối CSDL... trong khi Analysis Services đã cung cấp sẵn những điều đó bởi đội ngũ thiết kế của Microsoft, ta chỉ cần lập trình phần lõi thuật toán để khai thác dữ liệu và cho ra tri thức. Từ phần này trở đi, tư liệu nhóm trình bày dựa trên các phiên bản Visual Studio và MSSQL Server Management Studio 2005, 2008. Ngôn ngữ sử dụng là C#.

2.8.1. Mô tả các thuật toán trong Analysis Services [6]

Microsoft Naïve Bayes

Đây là thuật toán có chức năng phân loại. Thuật toán Bayes được giới thiệu vào khoảng thế kỷ 18, được xem là nền tảng của các thuật toán sau đó được ứng dụng vào khai thác dữ liệu. Tuy nhiên, hiện nay đã có nhiều thuật toán mới ra đời và tinh vi hơn như là Decision Tree và Neural Network. Mặc dù vậy, đặc trưng của thuật toán này là dễ dàng tìm được mối quan hệ giữa các thuộc tính với nhau nhanh hơn các phương

pháp khác. Thuật toán này cũng được dùng để tính xác suất và dự đoán.Ví dụ hình 2.11 là một trong các kết quả đã khai thác được.

Attributes	Values	Favors Republican	Favors Democrat
Class Action Fairness Act	Y		
Fed Up Higher Education Technical ...	N		
Fed Up Higher Education Technical ...	Y		
Class Action Fairness Act	N		
Help Efficient Accessible Low Cost Ti...	Y		
Help Efficient Accessible Low Cost Ti...	N		
Premanent Death Tax Repeal Act	N		
Pension Security Act	N		
Premanent Death Tax Repeal Act	Y		

Hình 2.11 Tri thức tìm được của Microsoft Naïve Bayes

Attributes là các thuộc tính (các đối tượng bỏ phiếu), Values là phiếu chống hay thuận, Republican: cộng hoà, Democrat: dân chủ.

Dữ liệu đầu vào là số lượng phiếu bầu của mỗi đối tượng và giá trị phiếu bầu (Yes/No).

	DEATH TAX		HOMELAND SECURITY		HELP AMERICA VOTE		CHILD ABDUCTION		PARTY	
	D	R	D	R	D	R	D	R	D	R
Yea	41	214	87	211	184	172	178	210	211	223
Nay	166	4	114	6	11	36	23	1		
Yea	20%	98%	43%	97%	94%	83%	89%	99.5%	49%	51%
Nay	80%	2%	57%	3%	6%	17%	11%	0.5%		

Hình 2.12 Dữ liệu đầu vào của Microsoft Naïve Bayes

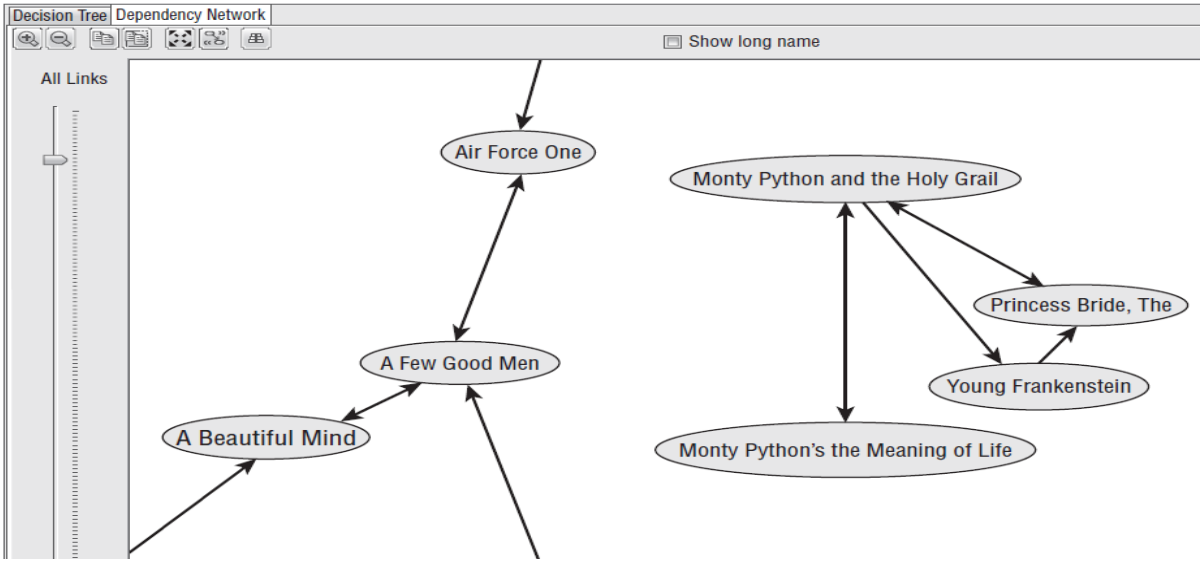
(D: dân chủ, R: cộng hoà, Yea: thuận (Yes), Nay: chống (No)).

Bayes không áp dụng được cho các dữ liệu mang tính liên tục và các lớp riêng biệt không tuyến tính, trường hợp này Decision Tree và Neural Network giải quyết tốt.

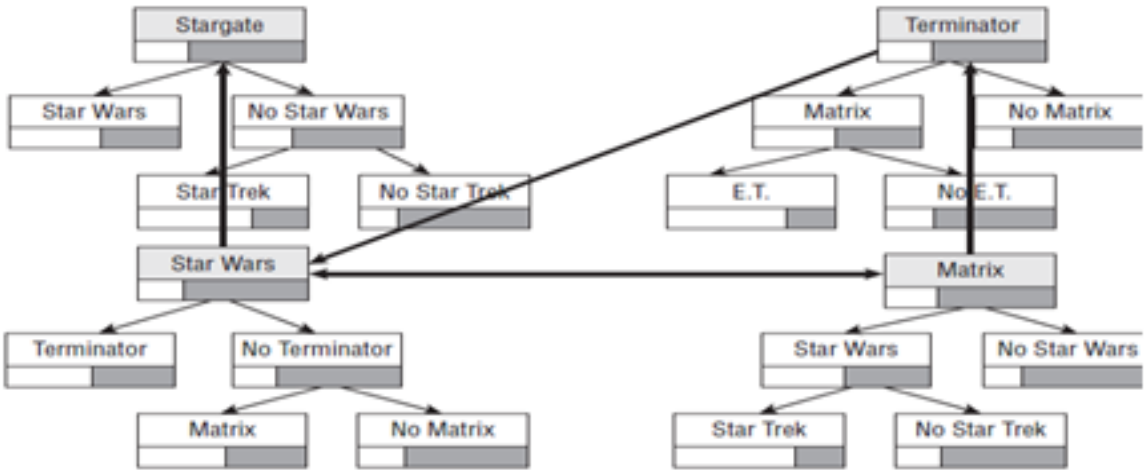
Microsoft Decision Trees

Có chức năng phân loại và kết hợp với hồi quy, đặc biệt là từ một mô hình dữ liệu có thể tạo nhiều cây để phân tích kết hợp. Mục đích của thuật toán là giúp đưa ra quyết định cho vấn đề bằng cách phân tích các trường hợp. Ví dụ: ta có cho một người bạn mượn số tiền lớn hay không? Các trường hợp của anh ta cần xem xét là: đang học đại

học (?), nhà nghèo (dấu hiệu xấu), đi xe tay ga (dấu hiệu tốt), bạn thân (tốt), học bình thường (?)… Một ví dụ khác của rạp chiếu phim: dữ liệu đầu vào với dòng là các ID khách hàng, thuộc tính là giới tính, tình trạng hôn nhân và dữ liệu các phim chọn để chiếu, dữ liệu đầu ra là các phim được chọn chiếu tiếp theo để phù hợp với đa số khách hàng hiện tại của mình. Quá trình khai thác là từ dữ liệu đầu vào sẽ thực hiện các truy vấn DMX (ngôn ngữ truy vấn hỗ trợ cho các thuật toán khai thác dữ liệu) để phân tích, cuối cùng là phân tích kết hợp giữa các trường hợp đó và cho ra danh sách các phim nên chiếu. Lúc truy vấn có thể phân tích kết hợp ở dưới dạng bảng truyền thống hoặc ở dạng cây được thể hiện ở hình 2.13 và 2.14 cho thấy thuật toán có góc nhìn thoáng.



Hình 2.13 Tab Dependency Network của Microsoft Decision Trees



Hình 2.14 Hình minh hoạ xử lý phân tích kết hợp của Microsoft Decision Trees

Và kết quả, ta thấy được các phim nên chiếu cho mỗi khách thể hiện ở hình 2.15.

CUSTOMERID	RECOMMENDATION
101	*Terminator 2: Judgment Day*
	Shawshank Redemption, The
	A Beautiful Mind
	Matrix, The
	Saving Private Ryan

Hình 2.15 Tri thức tìm được của Microsoft Decision Trees

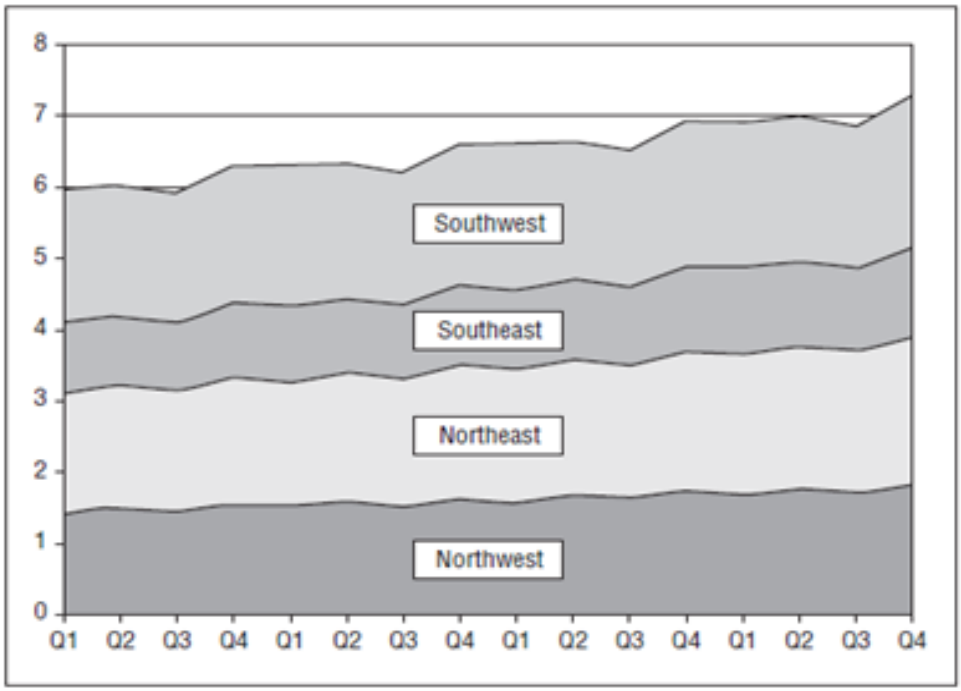
Có thể xem các phim nên chiếu cho đa số khách bằng câu truy vấn đơn giản.

Microsoft Clustering

Dùng kỹ thuật Expectation Maximum (kỹ thuật mềm, có thể thay đổi các tham số liên quan) hoặc Distance-based (K-mean) (kỹ thuật cứng, khó thay đổi tham số), tùy trường hợp sẽ chọn kỹ thuật thích hợp.

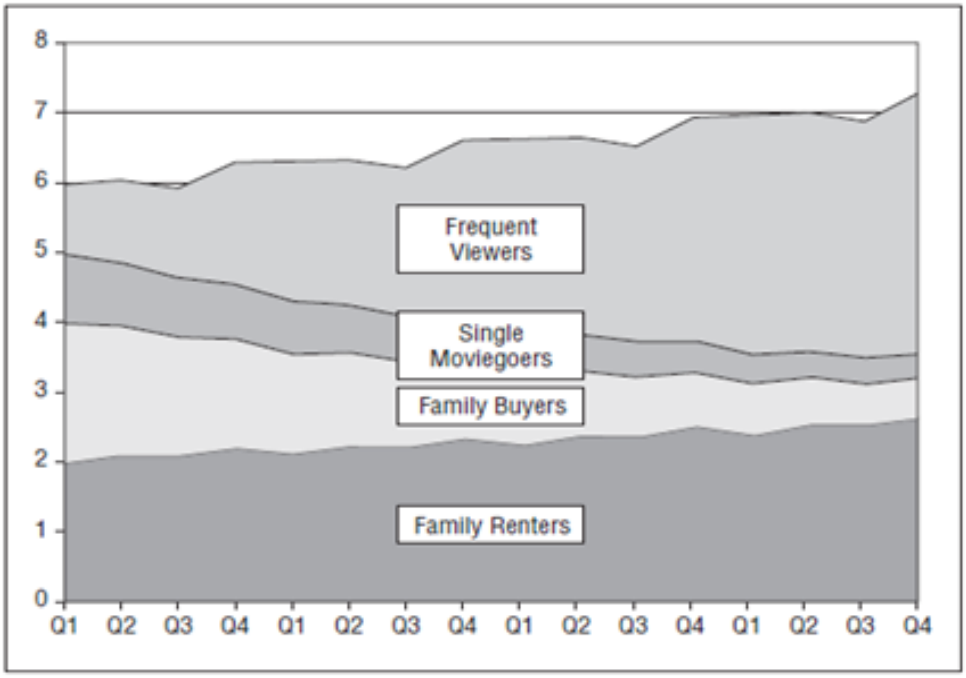
Mục đích là từ khối dữ liệu, phân dữ liệu vào cùng nhóm có tính chất tương tự nhau, mỗi nhóm dữ liệu có tính chất khác biệt nhau. Từ kết quả thu được ta có thể nhìn thấy được những thông tin quan trọng mà lúc chưa gom không thấy, từ đó đề ra chiến lược mới giúp cho công việc của mình tốt hơn.

Ví dụ: dữ liệu lưu trữ của các công ty đa quốc gia hay lưu ở dạng từng khu vực và báo cáo theo từng quý, quý này tại vùng này bán được bao nhiêu, vùng kia bán được bao nhiêu, quý sau bán được bao nhiêu…



Hình 2.16 Mô hình minh hoạ dữ liệu đầu vào của Microsoft Clustering

Nhìn vào sơ đồ trên, ta thấy được là kết quả kinh doanh có chiều hướng lời (tại bốn khu vực đều tăng), tuy nhiên khó tìm ra chiến lược để tăng tiếp doanh thu. Lấy dữ liệu dùng vẽ mô hình trên làm dữ liệu đầu vào của Clustering. Thuật toán sẽ khai thác và cho ra các hướng nhìn khác giúp ta có nhiều thông tin hơn để có thể đề ra chiến lược. Ví dụ như sau, khi khai thác ta gom nhóm thông tin theo hướng khác là loại khách hàng mua. Kết quả thu được thể hiện ở hình 2.17 hoàn toàn khác so với hướng nhìn theo khu vực.



Hình 2.17 Tri thức tìm được của Microsoft Clustering

Lúc này nhìn vào ta thấy rõ (dù doanh thu tất cả vùng đều tăng) nhưng trong đó, nhóm "Family Buyers" và "Single Moviegoers" giảm. Từ đó việc đề ra chiến lược tăng doanh thu dễ dàng hơn: làm thế nào tăng tiêu thụ của hai nhóm khách hàng đó? Sao họ lại không mua, có khuyến mãi đặc biệt cho nhóm đó để kéo khách hàng lại hay không?

Microsoft Sequence Clustering

Dùng kỹ thuật "Probabilistic Clustering" và kết hợp phân tích chuỗi. Mục đích chính cũng là gom nhóm các trường hợp tương tự nhau và dùng kỹ thuật Markov Chain Model. Thuật toán này được áp dụng cho phép khai thác hành vi của người lướt web, chẳng hạn trên web thương mại bán hàng, phân tích các Web-Click của người dùng, thấy được nhóm này đang ở thẻ nào, nhóm kia đang ở thẻ nào của trang web, sách và tạp chí hay dụng cụ thể thao… Để phân tích Web-Click thì cách xâu chuỗi là có hiệu quả nhất. Sau đó kết quả khai thác được cũng phục vụ tương tự như Microsoft Clustering. Thuật toán này có thể dùng khai thác dữ liệu dòng, dữ liệu sinh học.

Microsoft Association Rule

Dùng luật kết hợp, đây là thuật toán được dùng phổ biến nhất. Thuật toán này không khai thác được dữ liệu liên tục.

Quá trình khai thác có hai bước:

- Tìm các bộ dữ liệu phổ biến sử dụng thuật toán Apriori với ba ngưỡng giá trị là độ phổ biến, độ tin cậy, độ quan trọng.
- Phát sinh luật.

Microsoft Linear Regression

Phần lõi được thiết kế chung với phương pháp hồi quy và Microsoft Decision Trees. Có thể nói Microsoft Decision Trees bao gồm luôn thuật toán này, tuy nhiên thuật toán được đặt trong danh sách các thuật toán để giúp cho những người dùng muốn thực hiện phân tích hồi quy tuyến tính đơn giản mà không cần dùng đến Decision Tree.

Microsoft Neural Network và Microsoft Logistic Regression

Microsoft Logistic Regression có phần lõi được cài đặt giống với Microsoft Neural Network, dùng kỹ thuật hồi quy để khai thác dữ liệu, cả hai đầu dữ liệu vào và ra đều rời rạc, tuy nhiên đơn giản hơn Microsoft Neural Network vì không có lớp "Hidden Layer". Lớp này chuyển đổi dữ liệu đầu vào để tạo ra dữ liệu đầu vào mới và kết nối với dữ liệu đầu ra, áp dụng cho các trường hợp cấp hai để dự đoán mối quan hệ giữa các sự kiện. Vì vậy Microsoft Logistic Regression chỉ dự đoán mối quan hệ ở cấp một và là một trường hợp của Microsoft Neural Network. Thuật toán này được dùng để mô hình và dự đoán xác suất xảy ra của các sự kiện dựa vào dữ liệu đầu vào.

Microsoft Neural Network được áp dụng để tìm các mẫu không tuyến tính phức tạp hơn mà có thể Microsoft Logistic Regression hay Microsoft Decision Trees không phát hiện ra nên phức tạp và chạy chậm hơn các thuật toán trong chức năng phân loại và hồi quy của Microsoft. Được áp dụng khi kết quả khai thác của Microsoft Decision Trees, Microsoft Naïve Bayes, Microsoft Regression được đánh giá không chất lượng. Đối với các trường hợp đơn giản ta có thể dùng chúng mà không cần dùng Microsoft Neural Network.

Ví dụ: dùng Microsoft Neural Network khai thác trường hợp đơn giản ra kết quả cũng tương tự Microsoft Naïve Bayes ở hình 2.18.

Attribute	Value	Favors Own	Favors Rent
Age	38.003 - 54.813		
Age	20.000 - 28.252		
Marital Status	Never Married		
Education Level	Trade School		
Marital Status	Divorced		
Education Level	Grade School		
Marital Status	Married		
Education Level	High School		
Marital Status	Separated		
Education Level	Post-Doc		
Age	33.127 - 38.003		
Gender	Female		
Education Level	Associate's Degree		
Education Level	Some College		
Age	28.252 - 33.127		
Education Level	Master's Degree		
Marital Status	Other		

Hình 2.18 Tri thức tìm được của Microsoft Neural Network

Viewer thể hiện mức liên quan giữa các thuộc tính với nhau (độ tuổi, trạng thái hôn nhân, trình độ… với sở thích là sở hữu nhà hay thuê mướn nhà).

Text Mining

Đây không phải là thuật toán nằm trong danh sách. Tuy nhiên, ta có thể khai thác dữ liệu dạng text bằng cách dùng SQL Server Intergration Services (SSIS) chuyển dữ liệu không cấu trúc thành có cấu trúc. Sau đó có thể dùng các kỹ thuật khai thác trên dữ liệu đã chuyển đổi đó.

Microsoft Time Series

Sử dụng dữ liệu chuỗi tương tự Microsoft Sequence Clustering.

Mục đích của thuật toán là tạo mô hình và dự báo, thuật toán này sử dụng kết hợp hai thuật toán khai thác là Auto Regression và ARIMA (hệ phương pháp Box Jenkins) để dự đoán có độ chính xác nhất. Bên cạnh đó, thuật toán còn sử dụng kỹ thuật Fast Fourier Transform của hướng tìm mẫu có tính chu kỳ để tìm các mẫu tái diễn theo chu kỳ. Vậy Microsoft Time Series là sự kết hợp giữa hướng phân tích xu hướng và hướng tìm mẫu chu kỳ của các phương pháp khai thác dữ liệu chuỗi thời gian. Để áp dụng Microsoft Time Series, cần dùng cấu trúc OLAP cho dữ liệu đầu vào để khai thác có hiệu quả. Microsoft Time Series không áp dụng các kỹ thuật của SPAM.

Với Office 2007 (Excel, Visio)

Microsoft cung cấp Data Mining Add-Ins Tools miễn phí. Tuy nhiên, công cụ này chỉ hỗ trợ tốt đối với người dùng không chuyên như nhà phân tích tài chính hay khách hàng bình thường vì giao diện đơn giản, thân thiện. Với khả năng cho phép thêm thuật toán tích hợp của BI Dev Studio giúp các nhà nghiên cứu hay lập trình viên có thể tự

do thiết kế thuật toán của mình một cách toàn diện, khai thác tốt hơn các chức năng hỗ trợ của SQL Analysis Services.

2.8.2. Cấu hình thuật toán tích hợp vào MSSQL Analysis Services [1]

Yêu cầu

- Bộ Visual Studio (VS) và MSSQL có chức năng Data Mining Analysis Services. Điều này có thể kiểm tra trong lúc cài đặt sản phẩm. Nếu VS không có chức năng này cũng có thể dùng SQL Server Business Intelligence Development Studio (BI Dev Studio) để thay thế.
- Gói Data Mining Managed Plug-in API được cung cấp bởi Microsoft tại liên kết http://download.microsoft.com/download/f/7/4/f74cbdb1-87e2-4794-9186-e3ad6bd54b41/DMMgdPlugInAPI.msi.

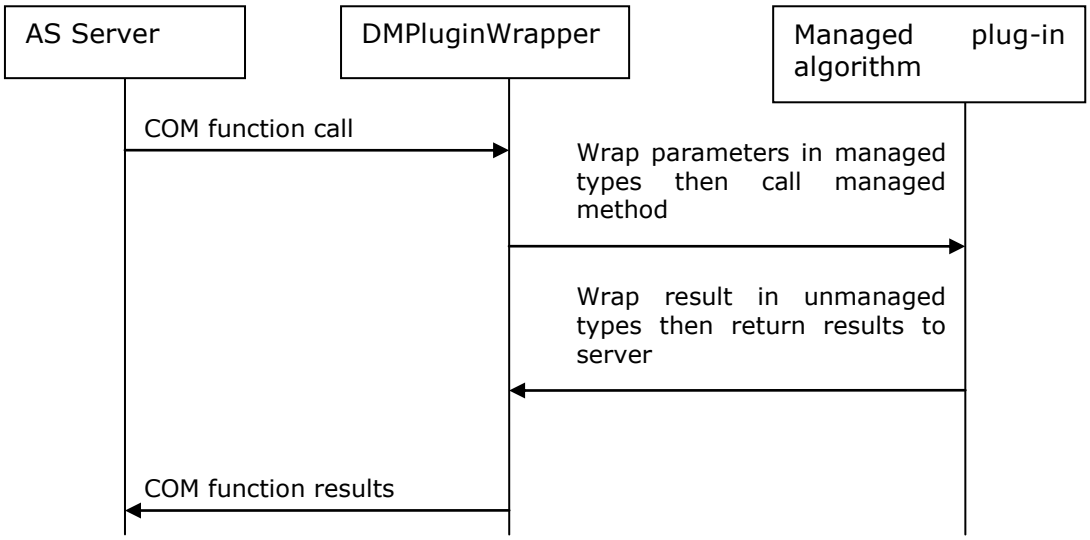
Thực hiện

- Cài đặt DMMgdPlugInAPI.msi sau khi tải về tại liên kết trên.

File này chứa tất cả các hướng dẫn, gói mã nguồn, kể cả dữ liệu để kiểm thử cho việc cấu hình và triển khai thuật toán tích hợp.

- Mở project DMPluginWrapper và lần lượt build trong Debug và Release để có file .DLL.

Để tích hợp được thuật toán vào Analysis Services (AS) cần có một bộ phận trung gian làm nhiệm vụ giao tiếp giữa AS và thuật toán. Bộ phận này chính là DMPluginWrapper và được xem là "Primary Interop Assembly", cơ chế hoạt động được biểu diễn như hình 3.9.



Hình 2.19 Tương tác giữa AS Server và thuật toán tích hợp

Kiến trúc của thuật toán tích hợp được gọi là COM (Compatible Language), các thành phần cần thiết của kiến trúc này cũng được cài đặt trong DMPluginWrapper để tương

tác với thuật toán tích hợp. AS Server khi được khởi động sẽ sử dụng COM để cập nhật thông tin về thuật toán tích hợp thông qua DMPluginWrapper. COM là một hộp đen được Microsoft cung cấp nằm trong DMPluginwrapper.DLL mà ta có được bằng cách build project DMPluginwrapper.

- Đảm bảo trong thư mục "C:\Program Files\Microsoft SQL Server\90\SDK\Include" có chứa file "oledbdm.h". Nếu SQL Server được cài trong thư mục khác thì cần sửa lại project và sửa đường dẫn "include" đến đúng thư mục có chứa file.
- Xác định đường dẫn của file "gacutil.exe" trong máy. Có thể tham khảo các đường dẫn sau:

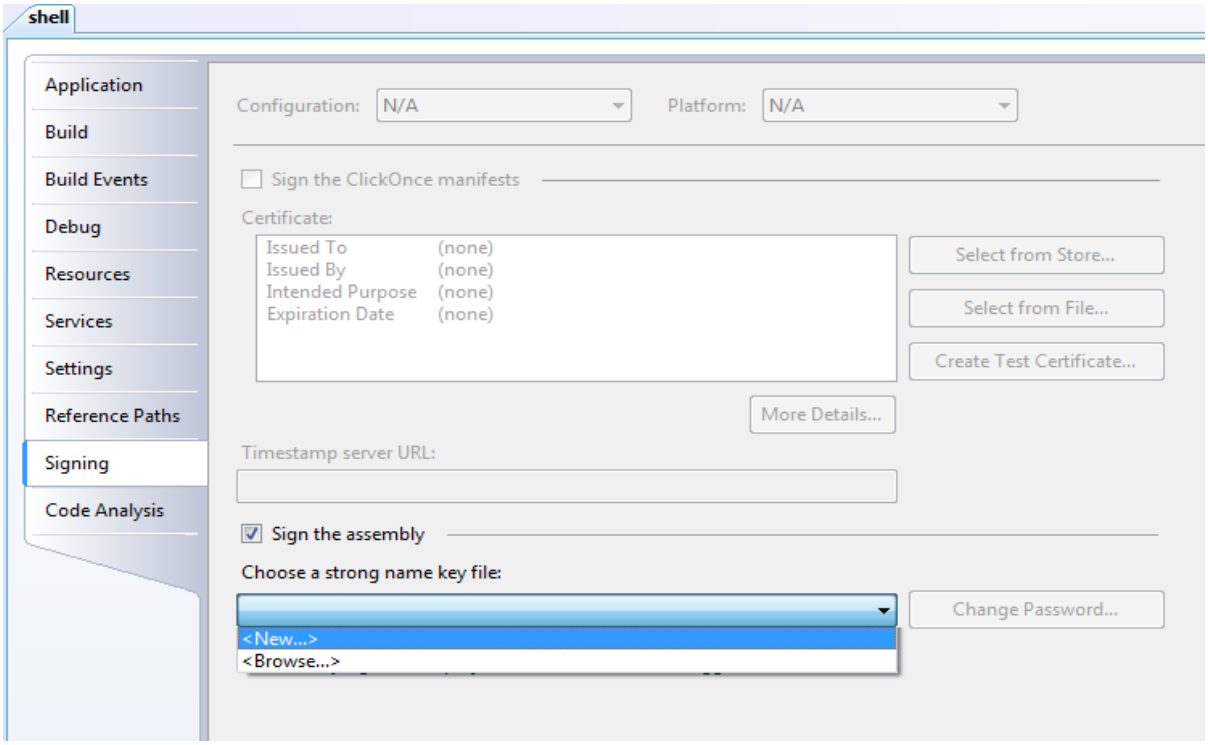
"C:\Program Files\Microsoft SDKs\Windows\v6.0A\bin\gacutil.exe" (hoặc)
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\gacutil.exe"

GAC Ultility (Global Assembly Cache) là nơi cập nhật thông tin của DMPluginWrapper (COM interop assembly) trên máy.

- Thực hiện trên CMD lần lượt cho cả Debug và Release để tiến hành cập nhật Assembly bằng các câu lệnh sau:

```
cd\
cd <thu_muc\dmpluginwrapper\bin\Debug(hoặc Release)>
"C:\Program Files\Microsoft SDKs\Windows\v6.0A\bin\gacutil.exe" /u DMPluginWrapper
"C:\Program Files\Microsoft SDKs\Windows\v6.0A\bin\gacutil.exe" /if
DMPluginWrapper.DLL
```

- Mở project Shell. Project này chính là khung của thuật toán tích hợp.
- Tạo Key mới cho project Shell bằng cách click phải chọn "Properties/Signing".



Hình 2.20 Tạo Key cho khung Shell

- Tại "Choose a strong name Key file" chọn "New" và nhập tên vào (có thể nhập mật khẩu). Ví dụ Key có tên là Shell, sau khi nhấn "OK" sẽ tạo ra file Shell.snk.
- Thêm DMPluginWrapper vào project bằng cách vào "File\Add Existing Project". Chọn thư mục đã build DMPluginWrapper và chọn file "DMPluginWrapper.vcproj".
- Thêm Reference của project bằng cách click phải vào "Project\Add reference". Click tab Projects và chọn DMPluginWrapper.
- Xác định đường dẫn của file "RegAsm.exe" trong máy. Có thể tham khảo đường dẫn sau:

"C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\RegAsm.exe"

Đây là file đăng ký Assembly với AS Server để AS Server sau khi khởi động sẽ cập nhật Assembly.

- Chọn Properties của project Shell, tab Build Events. Copy vào phần Post-Build (run on success build) các câu lệnh:

```
"C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\RegAsm.exe" $(TargetPath)
"C:\Program Files\Microsoft SDKs\Windows\v6.0A\bin\gacutil.exe" /u $(TargetName)
"C:\Program Files\Microsoft SDKs\Windows\v6.0A\bin\gacutil.exe" /if $(TargetPath)
```

"/u": xóa phiên bản hiện tại của Assembly.
"/if": cài đặt lập tức phiên bản Assembly sau khi thực hiện build thành công.

- Lần lượt build trong Debug và Release của project Shell để có file .DLL.

Lưu ý

- Nếu build Shell không thành công, xóa phần Post-Build và thực hiện build lại Shell. Sau đó thực hiện trên CMD lần lượt cả Debug và Release các câu lệnh sau:

```
cd\
cd <thu_muc\Shell\bin\Debug(hoặc Release)>
"C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\RegAsm.exe" Shell.DLL
"C:\Program Files\Microsoft SDKs\Windows\v6.0A\bin\gacutil.exe" /u Shell
"C:\Program Files\Microsoft SDKs\Windows\v6.0A\bin\gacutil.exe" /if Shell.DLL
```

- Nếu sau khi xóa Post-Build nhưng build Shell vẫn không thành công thì cần làm lại Shell được hướng dẫn trong tài liệu "A Tutorial for Contructing a Managed Plug-In Algorithm" (2006) tại phần tài liệu tham khảo hoặc sử dụng file Shell được nhóm cung cấp tại phụ lục C.
- Một số lỗi thường gặp:
  - Build Release của Shell thành công nhưng tiếp đó build Debug bị lỗi.
  - Lỗi "Can't locate input assembly" thường gặp ở Win XP khi build Shell.
  - Lỗi "Administrator permission" thường gặp ở Win Seven khi build project và chạy CMD, vì vậy cần chạy CMD và VS ở chế độ "Run as Administrator".

- Khi build thành công, bước tiếp theo là đăng ký thuật toán với Analysis Services. Có 2 cách:
  - Cách 1: sử dụng câu truy vấn XMLA để thêm thông tin thuật toán tích hợp vào file "msmdsrv.ini".
    + Mở SQL Server Management Studio.
    + Connect với Server type "Analysis Services" và chọn Server name (Instance) muốn đăng ký thuật toán tích hợp.
    + Tạo truy vấn XMLA bằng cách vào "File\New\Analysis Services XMLA Query".
    + Copy những câu XMLA sau đây vào và thực thi:

```
<!--
 Template for registering a plug-in algorithm
 Replace MyPluginAlgorithm with the ServiceName of your algorithm
 Replace 00000000-0000-0000-0000-000000000000 with the Guid of your Algorithm
 After deploying, you will need to restart the server to load the plug-in
-->
<Alter
    AllowCreate="true"
    ObjectExpansion="ObjectProperties"
    xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
 <Object />
 <ObjectDefinition>
  <Server
       xmlns:xsd="http://www.w3.org/2001/XMLSchema"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <Name>.</Name>
    <ServerProperties>
      <ServerProperty>
        <!-- Replace MyPluginAlgorithm in the next line
           with the ServiceName of your algorithm -->
<Name>DataMining\Algorithms\MyCompany_Managed_Plugin_Algorithm\Enabled</Name>
        <Value>true</Value>
      </ServerProperty>
      <ServerProperty>
        <!-- Replace MyPluginAlgorithm in the next line
             with the ServiceName of your algorithm -->
<Name>DataMining\Algorithms\MyCompany_Managed_Plugin_Algorithm\CLSID</Name>
```

```
28.         <!-- Specify your algorithm GUID in the next line -->
29.         <Value>44503EAB-570E-4b25-A9F4-043949A7D78E</Value>
30.       </ServerProperty>
31.     </ServerProperties>
32.   </Server>
33. </ObjectDefinition>
34. </Alter>
```

Chú ý tên thuật toán (dòng 27) và số GUID (dòng 29) phải khớp với thông tin trong Metadata.cs của Shell, khi đăng ký thuật toán khác nên tạo tên và số GUID mới. Có thể tạo GUID trong VS bằng cách vào "Tools\Create GUID".

Câu truy vấn XMLA này sẽ cập nhật thêm tên và số GUID của thuật toán tích hợp, sau đó kích hoạt thuật toán trong Instance tương ứng để Analysis Services có thể thấy khi khởi động.

- Cách 2: Sửa trực tiếp.
  - + Tìm file "msmdsrv.ini" trong máy. Có thể tham khảo đường dẫn:

"C:\Program Files\Microsoft SQL Server\MSAS10.MSSQLSERVER\OLAP\Config" Mỗi Instance đều có chứa Analysis Services và một file "msmdsrv.ini" tương ứng. Khi đăng ký file nào thì thuật toán sẽ xuất hiện trong Instance đó.

+ Mở file "msmdsrv.ini" tìm đến thẻ <Algorithms> và copy vào đoạn sau:

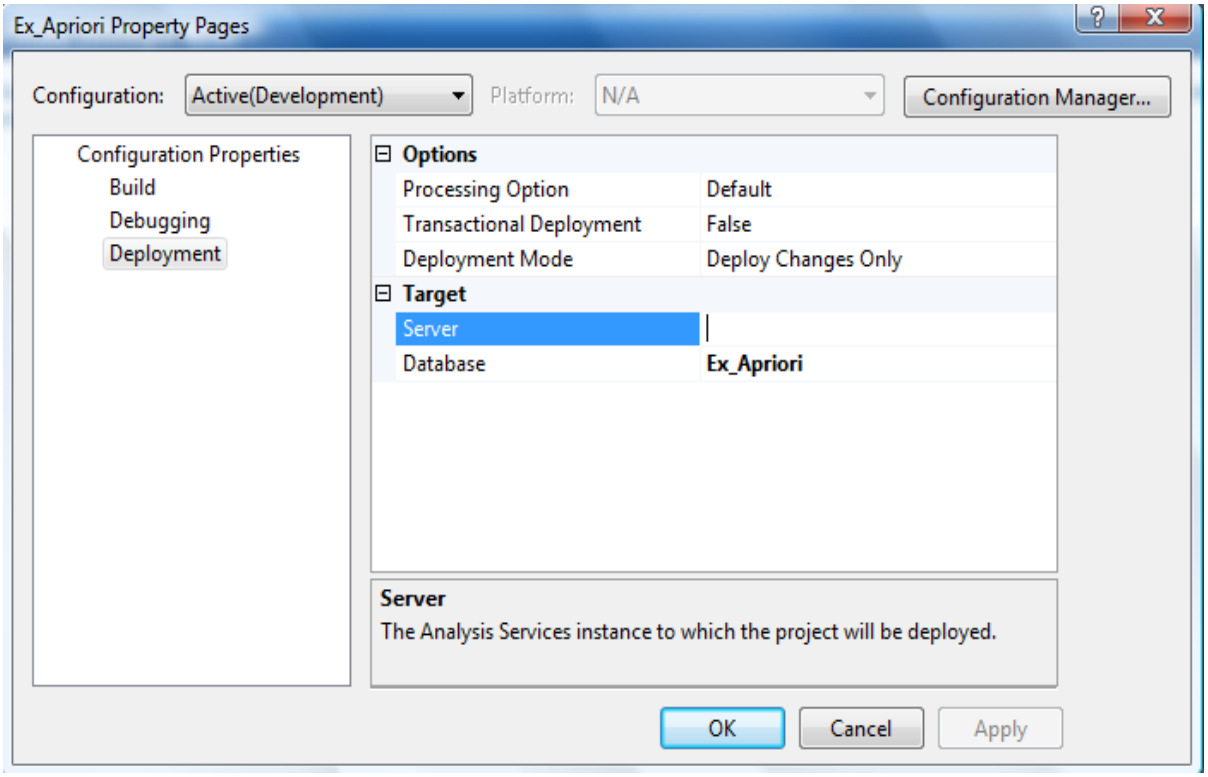
```
<MyCompany_Managed_Plugin_Algorithm>
        <Enabled>1</Enabled>
        <CLSID>44503EAB-570E-4b25-A9F4-043949A7D78E</CLSID>
  </MyCompany_Managed_Plugin_Algorithm>
```

+ Sau khi Stop và Start Services thì file "msmdsrv.ini" sẽ được Analysis Services cập nhật.
+ Nếu dùng Win Seven bị lỗi "Access denied" thì có thể copy file ra nơi khác, sửa và copy đè vào file cũ.
+ Khởi động lại Analysis Services trong Services.msc.

Kiểm thử

Mở BI Dev Studio, hoặc VS:

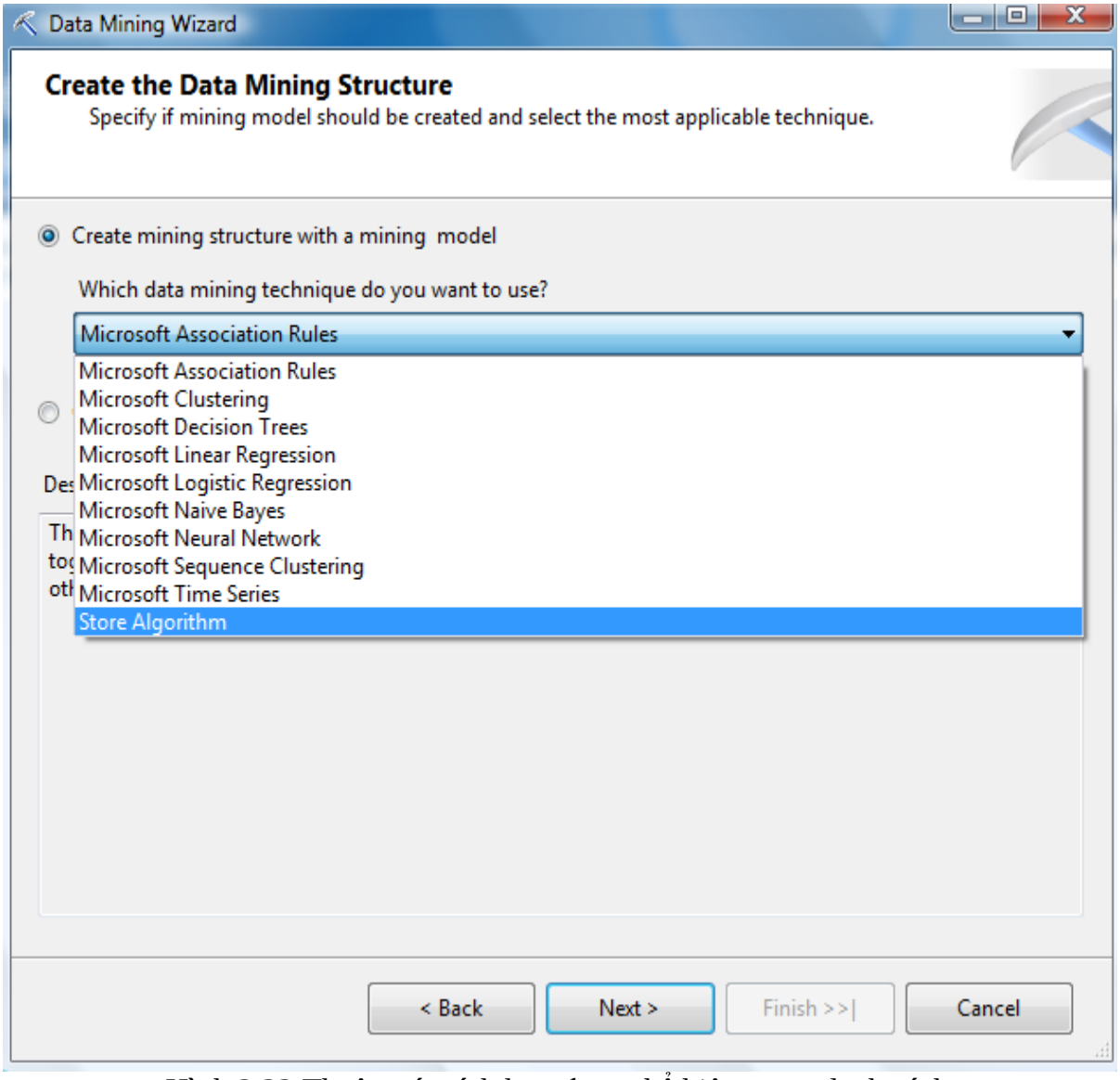
- Chọn "File\New\Project".
- Tại Project Types, click "Business Intelligence Projects".
- Tại Templates, click "Analysis Services Project". Đặt tên project sau đó click "OK".
- Sau đó click phải project chọn "Properties/Deployment" thay Server name bằng Server name của Instance đã đăng ký thuật toán.



Hình 2.21 Chọn Server Name

- Thực hiện kết nối tạo Data Source đến CSDL
  - Tại Solution Explorer, click phải vào Data Sources, và chọn "New Data Source".
  - Trang đầu, click Next.
  - Tại trang Select how to define the connection, click "New".
  - Trong hộp thoại Connection Manager, chọn loại CSDL cần dùng. Ví dụ như "Microsoft Jet 4.0 OLEDB Provider" cho dữ liệu Access và "SQL Server Native Client 10.0" cho dữ liệu SQL Server, sau đó click "OK".
  - Chọn dữ liệu và loại kết nối. Nếu CSDL không đặt mật khẩu thì chọn "Use the Service Account".
  - Click Next, sau đó click Finish.
- Tạo Data Source View
  - Tại Solution Explorer, click phải vào Data Source Views và chọn "New Data Source View".
  - Trang đầu, click "Next". Tại trang Select Data Source, click "Next" tiếp.
  - Tại trang Select Tables and Views, click nút ">" để chọn bảng, và click "Next". Trang tiếp theo, click "Finish".

- Tạo Data Structure và Data Model
  - Tại Solution Explorer, click phải vào Mining Structures, và chọn "New Mining Structure"**.**
  - Trang đầu click "Next". Tại trang Definition Method click "Next".
  - Tại trang "Select the Data Mining Technique" nếu đã đăng ký thành công thì trong danh sách sẽ hiện tên thuật toán.



Hình 2.22 Thuật toán tích hợp được thể hiện trong danh sách

- Click "Next".
- Trang kế tiếp, Chọn Data Source View và click "Next".
- Trang kế tiếp, chọn Case và/hoặc Nest.
- Trang tiếp theo, chọn Key, Input, Predict. Click "Next".
- Trang tiếp theo, click "Next".

- Đặt tên cho Mining Structure và Mining Model và click "Finish".
- Deploy project bằng cách nhấn F5 hay chọn sang Tab Viewer.
- Dùng Microsoft Generic Content Tree Viewer (default).
- Trong lúc khai thác, thuật toán có thể phát sinh lỗi không thể đăng nhập CSDL chứa dữ liệu đã chọn. Cần xem User thuật toán sử dụng là gì và mở MSSQL, click phải CSDL có dữ liệu đó, chọn "Properties\Files\Owner\Browse" và chọn User tương ứng. Ví dụ: Network Service.

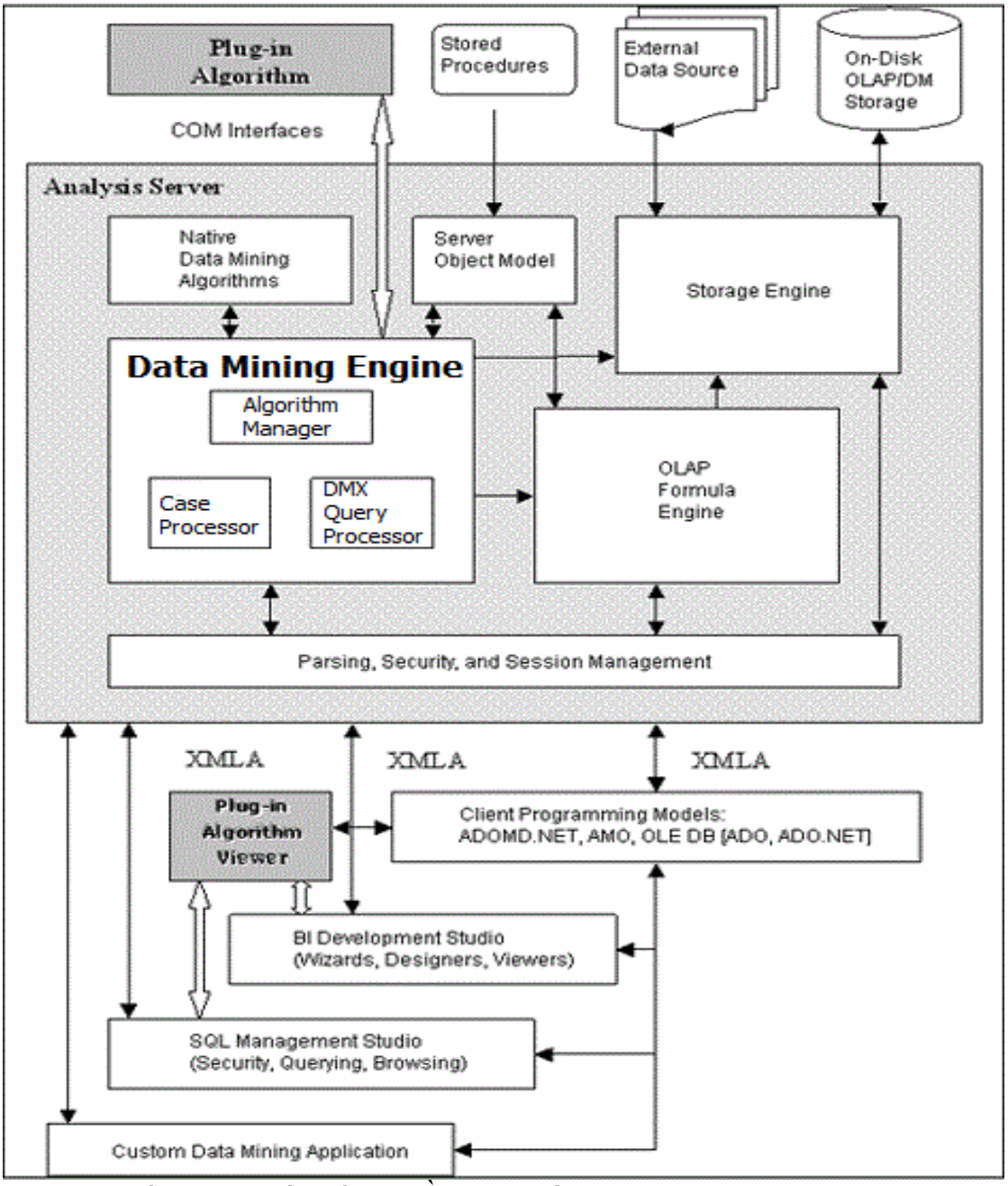
Nếu tên thuật toán không xuất hiện, có thể là do Analysis Services đang chạy khi build project Shell.

Lúc này ta cần:

- Save project và đóng chương trình.
- Stop Analysis Services và build lại project Shell (cả Post-Build hoặc CMD).
- Start Analysis Services, mở BI Dev Studio hoặc VS và mở lại project để kiểm thử.

2.8.3. Cơ chế hoạt động của thuật toán tích hợp [6]

Phần 2.8.2 đã phần nào mô tả cơ chế tương tác giữa Analysis Server và thuật toán tích hợp thông qua COM (DMPluginWrapper). Sơ đồ 2.23 thể hiện rộng hơn về mối tương quan giữa các thành phần trong tính năng khai thác dữ liệu của SQL Server Management Studio.

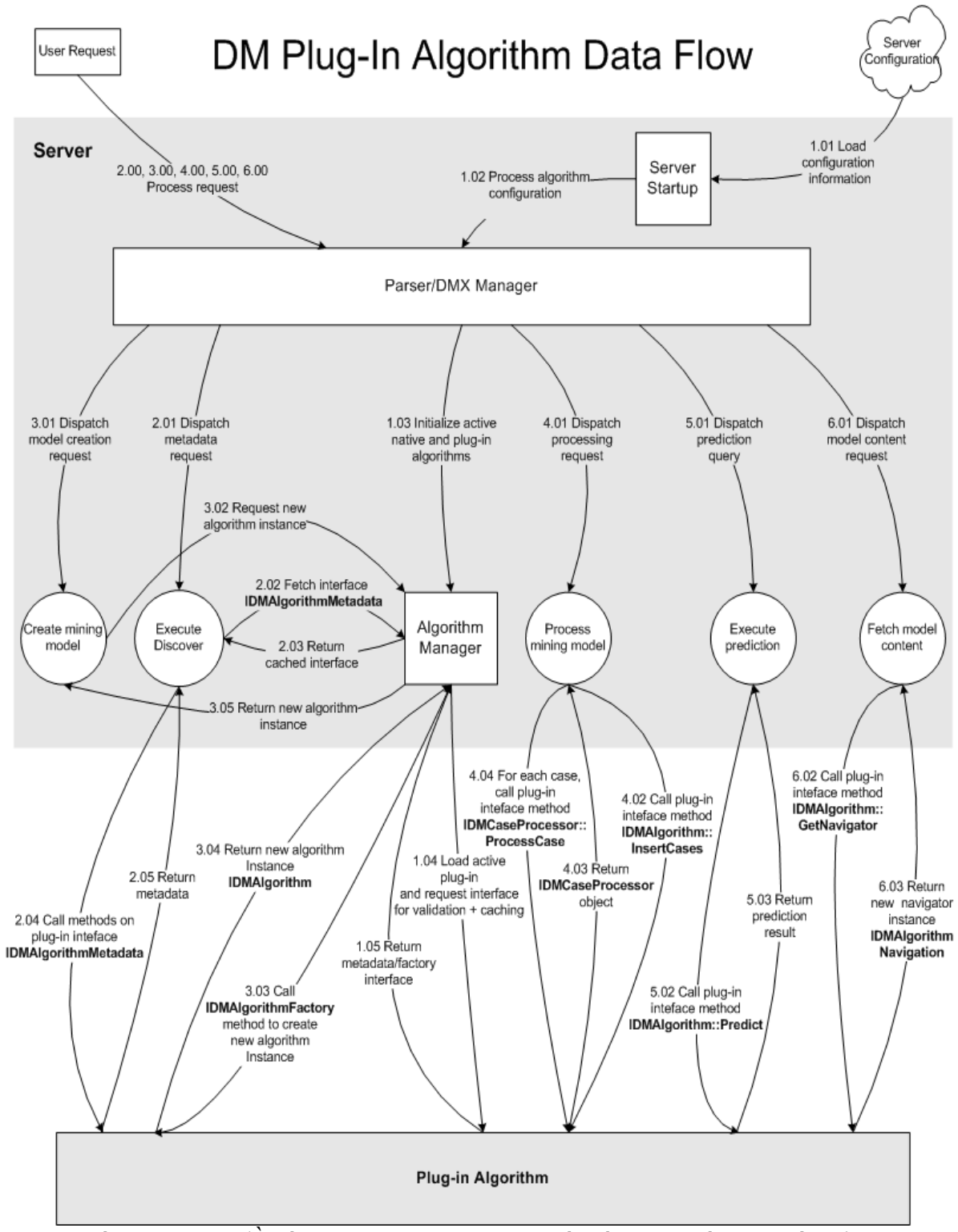


Hình 2.23 Các thành phần khai thác dữ liệu của SQL Server

Algogithm Manager: quản lý việc đăng ký và hủy đăng ký đang tồn tại của các thuật toán.

DMX (Data Mining Extensions) là một ngôn ngữ truy vấn dùng để khai thác dữ liệu trong Microsoft SQL Server Analysis Services. DMX có thể dùng để tạo cấu trúc của các mô hình khai thác dữ liệu mới, quản lý mô hình và dự đoán kết quả mà mô hình mang lại.

Ngoài việc tương tác với AS Server để được kích hoạt và sử dụng, toàn bộ cơ chế xử lý dữ liệu và thể hiện tri thức của thuật toán tích hợp còn thông qua một bộ phiên dịch được gọi là Parser. Hình 2.24 thể hiện chi tiết cơ chế xử lý dữ liệu của Parser.



Hình 2.24 Sơ đồ dòng dữ liệu giữa thuật toán tích hợp và bộ phiên dịch

Khung thuật toán tích hợp có ba lớp cơ bản là: Algorithm.cs, Algorithmnavigator.cs Metadata.cs.

- Metadata.cs: kế thừa AlgorithmMetadataBase. Đây là lớp chứa các thông tin khai báo, định nghĩa của thuật toán như: tên, số GUID, mô tả, các tham số cần dùng, kiểu Viewer sử dụng, các hàm, chức năng cần sử dụng của Microsoft hỗ trợ…
- Algorithmnavigator.cs: kế thừa AlgorithmNavigationBase. Lớp này xử lý dữ liệu đầu ra để thể hiện lên Microsoft Generic Content Tree Viewer.
- Algorithm.cs: đây là phần xử lý chính của thuật toán. Gồm hai lớp là MyCaseProcessor kế thừa ICaseProcessor và lớp Algorithm kế thừa AlgorithmBase. Phần lõi xử lý của thuật toán tích hợp được lập trình trong lớp Algorithm. Tùy theo độ phức tạp của thuật toán mà có thể tạo thêm các lớp mới bên trong hoặc ngoài Algorithm.cs để xử lý dữ liệu.

3. GIẢI QUYẾT VẤN ĐỀ

3.1. Triển khai thuật toán tích hợp Apriori

Bên cạnh ba lớp khung của thuật toán tích hợp Algorithm.cs, Algorithmnavigator.cs, Metadata.cs, nhóm tạo lớp AprioriAlgo.cs để cài đặt thuật toán Apriori gồm các thành phần chính là chuyển đổi thành dữ liệu dạng chuỗi qua hàm AddCase, AddItem, khai thác dữ liệu và cho ra tri thức được thực hiện bằng InsertCase, findRule. Đồng thời, các lớp dùng để lưu trữ cấu trúc dữ liệu là Rule, Node và Itemset. Chi tiết từng hàm được mô tả trong phụ lục C.

Bước chọn Input Table, Case chính là bảng ta cần khai thác, để xác định cần dựa vào mục đích, ví dụ: ta có bảng Customer chứa ID và các thông tin của khách hàng (tên, giới tính, tuổi...) và bảng Product chứa ID Transaction, ID Customer, thông tin sản phẩm được mua (tên, loại, giá...). Ta muốn tìm những sản phẩm nào được mua cùng với nhau, tuy nhiên nếu chọn bảng Product là Case thì không đúng, vì nhiều sản phẩm được mua bởi cùng một khách hàng, và với đa số khách hàng sẽ mua những mặt hàng nào cùng với nhau. Vì vậy, bảng Customer chính là bảng cần khai thác. Ta chỉ chọn một Case duy nhất.

Nest: chứa các dữ liệu chi tiết liên quan đến bảng Case. Số lượng bảng Nest tùy ý, thường thì ta có nhiều Nest đối với CSDL lớn. Với ví dụ trên, bảng Product chính là Nest.

Các trường hợp thường xảy ra khi xét các bảng có tham chiếu với nhau:

- Chỉ có một bảng: bảng này chính là Case, không có Nest.
- Có một bảng chính, một hoặc nhiều bảng tham chiếu (dạng Detail): bảng chính là Case, các bảng tham chiếu là các Nest, có thể không chọn Nest khi không cần khai thác.
- Có nhiều bảng chính vừa được tham chiếu, vừa tham chiếu bảng khác: ta chỉ được chọn một bảng duy nhất làm Case, Nest vẫn là các bảng tham chiếu như trên. Đây là trường hợp một bảng có thể vừa là Case vì là đối tượng để khai thác, vừa là Nest vì tham chiếu bảng khác.

Cần lưu ý các khái niệm sau:

- Key: mỗi bảng chỉ nên có một Key, Key chính là định danh của kết quả sau khi khai thác được.
- Case Key: thường thì Case Key là khoá chính của bảng.
- Nest Key: là tên của các dữ liệu trong Case của Nest. Vì vậy, nếu chọn là khoá ngoại thì không đúng. Cụ thể Nest Key là cột xác định tên của các thuộc tính của một dòng để người dùng hiểu được. Nếu chọn Key là một ID hay mã nào đó như khoá ngoại, chẳng hạn một khách hàng có Customer_ID = 5, kết quả khai thác ra được <tên_sản_phẩm> của 5, <số_lượng> của 5, <loại> của 5 gây tối nghĩa và đối với một số trường hợp là vô nghĩa, trong một lúc sẽ gây khó hiểu nếu người dùng không hiểu rõ về Nest Key là gì, không biết khi chọn Nest là Customer_ID thì kết quả '5' chính là Customer_ID. Nếu ta chọn Nest Key là

cột tên sản phẩm, ví dụ: máy tính, kết quả khai thác ra được sẽ là <Customer_ID> của máy tính (người mua), <số_lượng> của máy tính, <loại> của máy tính. Đây là kết quả chính xác mà ta muốn có được.

- Input: dữ liệu thuật toán sẽ dựa vào để khai thác.
- Predict: dữ liệu thuật toán sẽ khai thác được (đầu ra).

Áp dụng trong luật kết hợp thì Input là chỉ xuất hiện bên vế trái và Predict là chỉ xuất hiện bên vế phải, nếu chọn cả hai sẽ xuất hiện cả hai vế. Càng chọn nhiều dữ liệu vào thì thuật toán càng cho ra kết quả phức tạp, nếu dữ liệu có quá nhiều cột sẽ gây khó chọn vì vậy mà có chức năng Suggest Input (không đảm bảo chính xác 100%), tuy nhiên nếu chọn ít sẽ có thể không có luật sinh ra.

Dữ liệu liên tục là dữ liệu mà thuật toán có thể sử dụng các phép toán số học như "+", "-" trong quá trình khai thác. Thường thì dữ liệu này ở dạng số. Dữ liệu rời rạc là dữ liệu ở dạng phạm trù và không thể tính toán, thuật toán sẽ xem các dữ liệu như là một giá trị chính xác và không thay đổi. Dữ liệu rời rạc thường ở dạng chữ và kí tự như giới tính. Có thể nhận dạng loại dữ liệu trong tự nhiên bằng một số cách như là xác định giá trị của chúng. Ví dụ về tuổi, ta có thể nói một người 22 tuổi, tuy nhiên cũng có thể nói người đó 22 tuổi và 4 tháng, ví dụ khác là khi đo chiều dài thì có thể thay đổi theo đơn vị và dụng cụ đo, vì vậy tuổi và chiều dài được xác định là dữ liệu liên tục. Về dữ liệu rời rạc, ví dụ: câu hỏi trắc nghiệm có bốn lựa chọn, kết quả chỉ nằm trong phạm trù 1, 2, 3, hoặc 4; hay giá trị ghi nhận số lần thi lại của một học sinh, học sinh có thể thi lại 1 lần, 2 lần hoặc 3 lần, tất cả các giá trị này đều là số nhưng chúng là dữ liệu rời rạc và khoảng cách giữa hai giá trị kề nhau đều bằng nhau. Có thể chuyển dữ liệu liên tục thành dữ liệu rời rạc bằng một số phương pháp, ví dụ gom cụm độ tuổi, nhóm người có độ tuổi nhỏ hơn 18 là vị thành niên, lớn hơn hoặc bằng 18 là trưởng thành… lúc này giá trị số được chuyển thành giá trị chữ.

Nếu ta không chọn Case, Nest, Key, Input, Predict và kiểu dữ liệu liên tục hoặc rời rạc hợp lý thì kết quả thuật toán khai thác ra được không có ý nghĩa.

Dữ liệu của nhóm thử nghiệm có 122824 dòng, sử dụng dữ liệu chứng khoán trong khoản ba năm là 2008 đến 2010.

Do kết quả được Microsoft Generic Content Tree Viewer thể hiện quá nhiều nên nhóm không thể biểu hiện bằng hình ảnh. Bằng cách thực hiện câu lệnh Debug.Assert (tham số), thuật toán sẽ hiện thông báo về giá trị của các tham số đó.

Lưu ý là câu lệnh Debug.Assert chỉ có tác dụng trên Window XP.

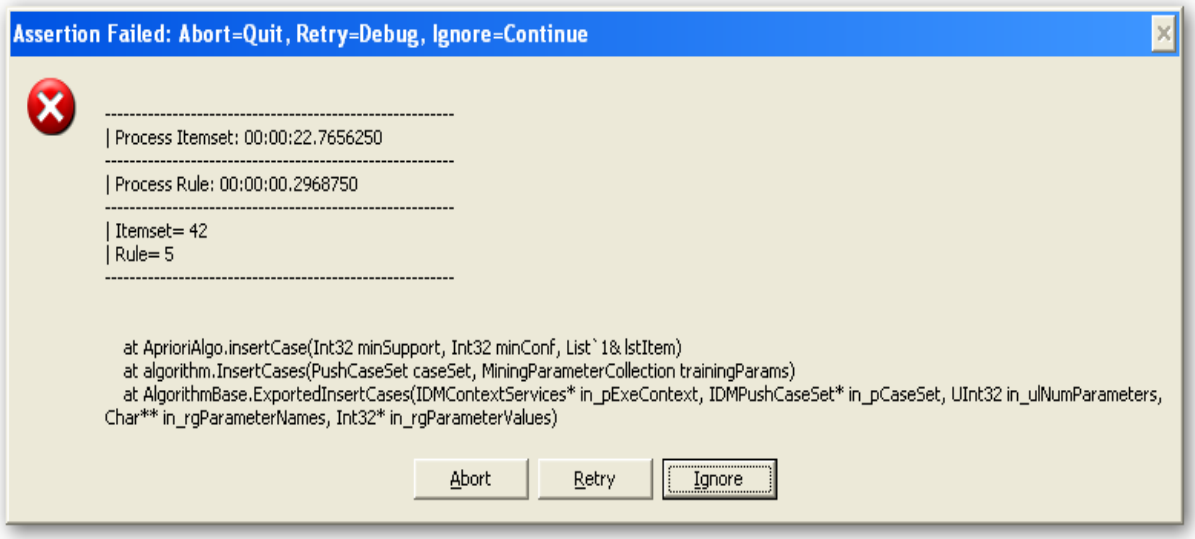
Hình 3.1 thể hiện giá trị thời gian khai thác bộ dữ liệu, kết quả khai thác là tổng số bộ dữ liệu phổ biến và luật của thuật toán Apriori.

Chú ý nếu sử dụng dữ liệu khác để kiểm thử, trường hợp thuật toán cho ra rất nhiều bộ dữ liệu và luật có thể làm Microsoft Generic Content Tree Viewer bị quá tải, thể hiện bằng câu thông báo: "XML was time out before it completed". Tuy nhiên ta không cần dùng Viewer để xem kết quả mà sẽ dùng các câu truy vấn DMX sau để xem kết quả trong SQL Server với Server Type là Analysis Services và Database là project đã tạo:

```
SELECT Node_Description, Node_Support, Node_Probability  FROM
[tên_Mining_Model].CONTENT
WHERE Node_Type = 7              --Xem bộ dữ liệu

SELECT Node_Description, Node_Probability FROM
[tên_Mining_Model].CONTENT
WHERE Node_Type = 8              --Xem luật
```

Có thể tạo Mining Structure, Mining Model bằng DMX mà không cần sử dụng BI Dev Studio. Cách này giúp việc chọn thuật toán khai thác và ngưỡng giá trị dễ dàng hơn.



Hình 3.1 Kết quả khai thác bộ dữ liệu của Apriori thể hiện bởi Debug.Assert

Kết quả thuật toán tìm được 5 luật từ 42 bộ dữ liệu phổ biến, thời gian chuyển dữ liệu thành đầu vào thuật toán là 1 tiếng 30 phút và thời gian khai thác bộ dữ liệu là 22 phút, vậy tổng thời gian cần thiết để hoàn thành là khoảng 1 tiếng 50 phút. Cấu hình máy kiểm thử là Window XP, CPU Core 2 Duo 2.2GHz, RAM 2GB.

3.2. Triển khai thuật toán tích hợp Bide

Giống như Apriori, nhóm tạo lớp BideAlgo.cs chứa các hàm chuyển dữ liệu dạng chuỗi, xử lý và tìm luật. Lớp lưu trữ cấu trúc CSDLC là SDB.cs. Để có thể sử dụng cả hai thuật toán tích hợp là Apriori và Bide trong một project, AprioriAlgo và BideAlgo sẽ kế thừa StoreAlgorithm và được gọi thông qua lớp trung gian AlgorithmFactory và Factory bằng câu lệnh Switch (Case). Lớp StoreAlgorithm chứa các thành phần chung của Apriori và Bide như đếm độ phổ biến của các bộ dữ liệu và tạo luật từ những bộ dữ liệu phổ biến. Kiến trúc này được nhóm tham khảo mẫu thiết kế Factory Method, tuy nhiên do giao diện sử dụng BI Dev Studio nên không thể chuyển qua lại giữa hai thuật toán bằng thao tác chọn của người dùng mà phải sửa trực tiếp dòng Switch (Case) hoặc biến khởi tạo thuật toán. Mô tả chi tiết các lớp và hàm được trình bày trong phụ lục C.

Sử dụng dữ liệu như cũ, tuy nhiên kết quả ra được khi chạy Bide giống như của Apriori. Điều này có thể do số lượng dữ liệu nhóm chọn không nhiều dẫn đến kết quả đạt được đều là các chuỗi đóng. Thời gian thực hiện khoảng 6 tiếng.

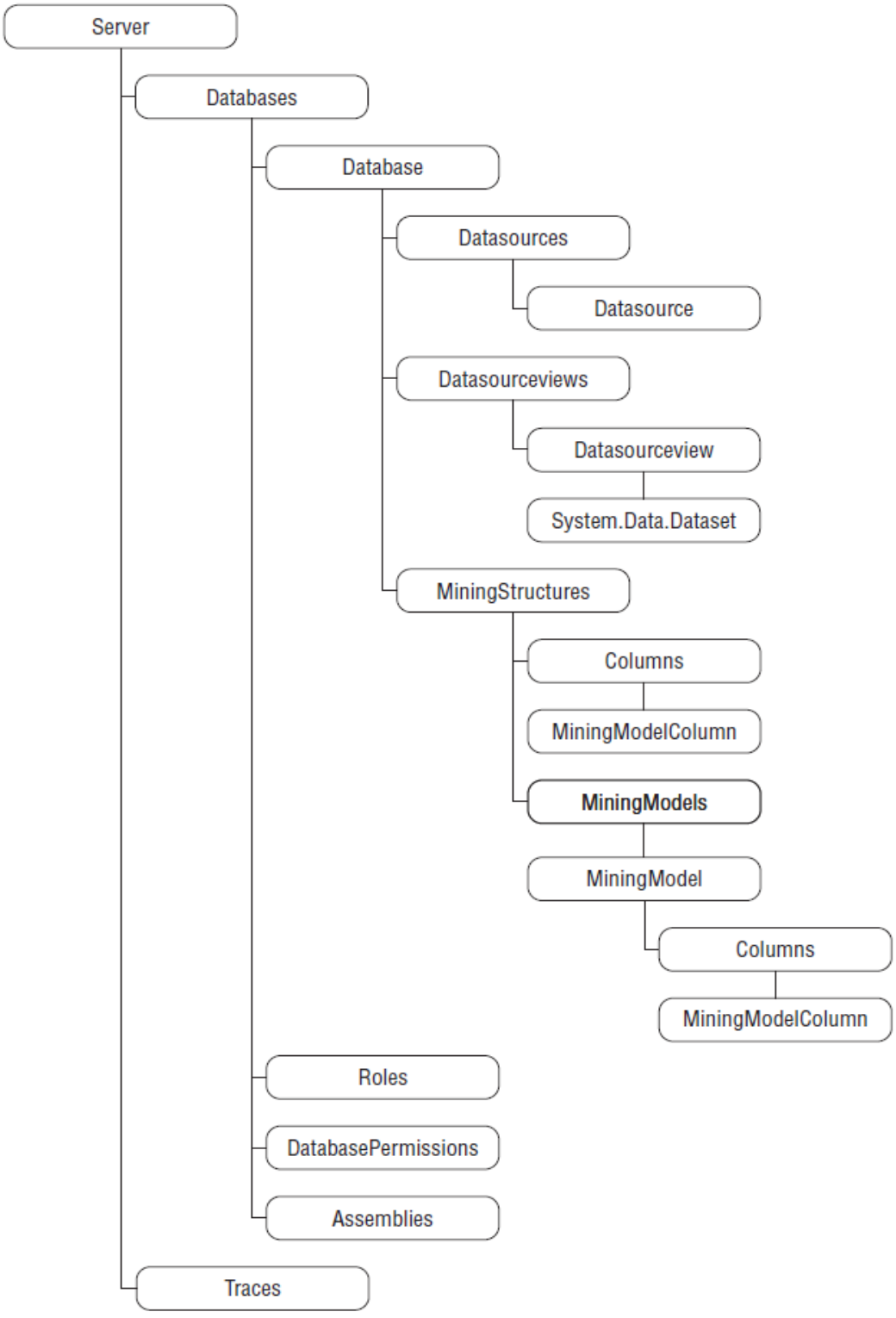
3.3. Ứng dụng sử dụng thuật toán tích hợp trong MSSQL Analysis Services

Để có thể tạo được ứng dụng sử dụng thuật toán tích hợp độc lập với giao diện BI Dev Studio, ta cần đẩy dữ liệu đầu vào vào thuật toán và lấy tri thức tìm được thông qua DMX (Data Mining Extentions). Điều này cũng được Microsoft cung cấp các gói APIs để hỗ trợ.

API	TYPE	REFERENCES
ADO	Native	Microsoft ActiveX Data Objects
ADOMD.NET	Managed	`Microsoft.AnalysisServices.AdomdClient`
Server ADOMD.NET	Managed	`Microsoft.AnalysisServices.AdomdServer`
AMO	Managed	`Microsoft.AnalysisServices` `Microsoft.DataWarehouse.Interfaces`

Hình 3.2 Các gói APIs hỗ trợ khai thác dữ liệu của Microsof cung cấp

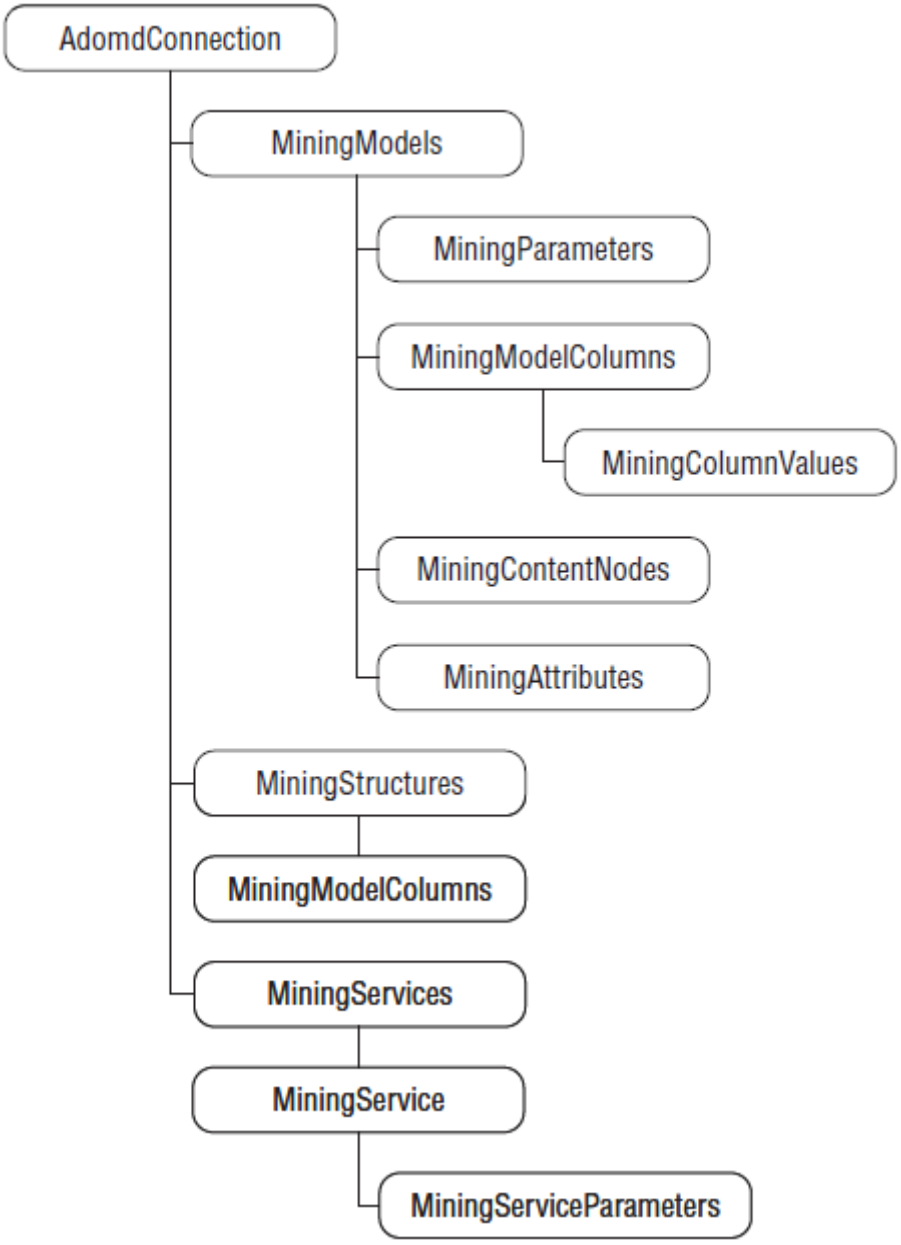
AMO (Analysis Management Objects) là API dùng để quản lý, thực hiện các thao tác như kết nối, tạo mới và xử lý Database, Data Source, Data Source View, Mining Structure, Mining Model.. trong Analysis Services. AMO có thể sử dụng trên các đối tượng trong hình 3.3.



Hình 3.3 Các đối tượng AMO có thể thao tác

ADOMD.NET (ActiveX Data Objects Multidimentional for .NET) là API được dùng để thực hiện các câu truy vấn dự đoán sau khi thuật toán đã khai thác dữ liệu. Loại truy vấn này bao gồm truy vấn đơn giản như xem kết quả khai thác được đến những truy vấn mức cao nhằm dự đoán dựa trên các kết quả đó.

ADOMD.NET có thể sử dụng trên các đối tượng trong hình 3.4.

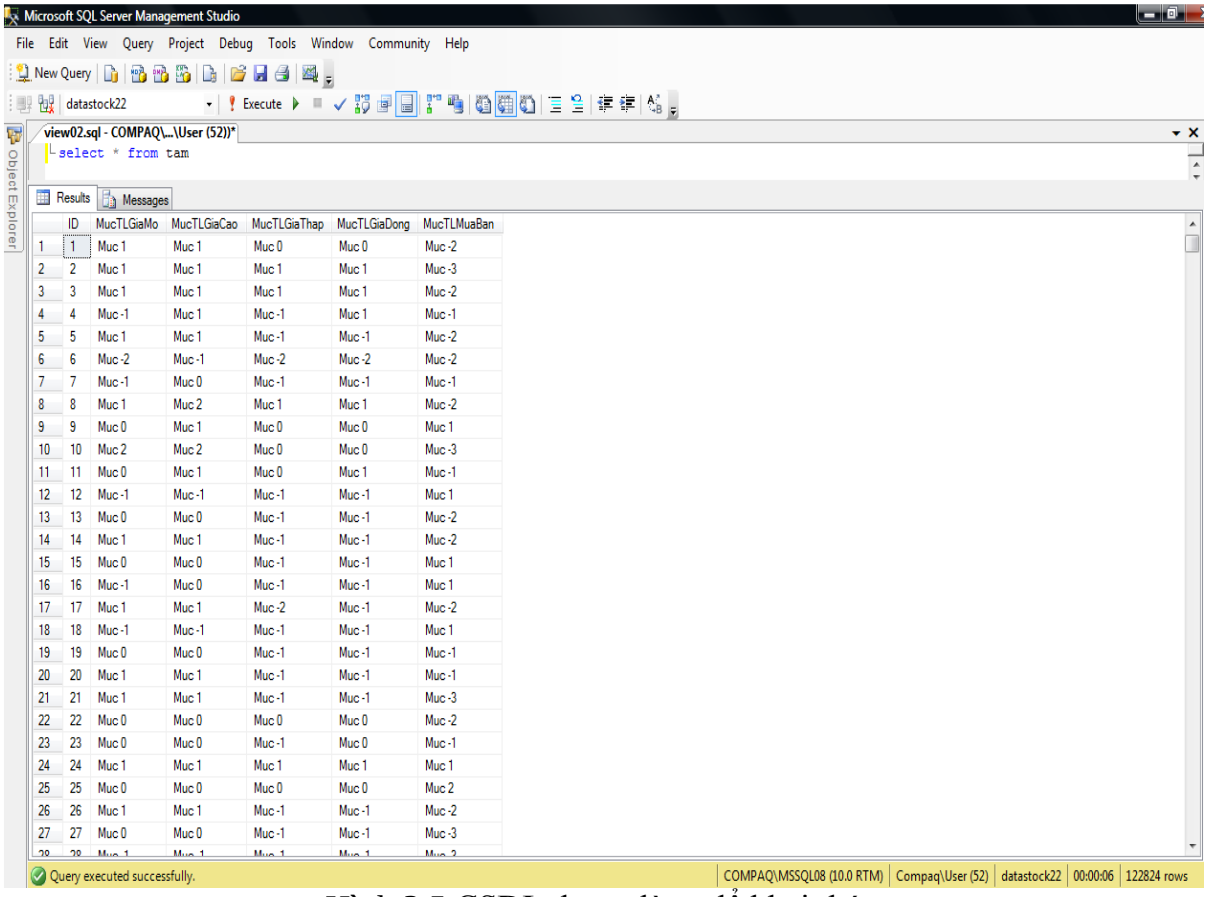


Hình 3.4 Các đối tượng ADOMD.NET có thể thao tác

Như vậy, ta sử dụng AMO để đẩy dữ liệu đầu vào và ADOMD.NET để xem kết quả khai thác được của thuật toán. Khóa luận của nhóm chỉ dừng ở mức luật kết hợp và chưa thực hiện chức năng dự đoán hay tư vấn đầu tư nên chưa sử dụng phần dự đoán của ADOMD.NET.

Nhóm tạo ba lớp chính là AnalysisService.cs để sử dụng AMO, lớp ASHelp.cs tạo kết nối, truy vấn kết quả bằng ADOMD.NET và lớp DBHelp để kết nối dữ liệu cần khai thác. Mô tả chi tiết về các lớp và hàm được trình bày trong phụ lục C.

Với dữ liệu vẫn như cũ, hình 3.5 thể hiện dữ liệu đầu vào của nhóm sử dụng để khai thác dữ liệu.



ID	MucTLGiaMo	MucTLGiaCao	MucTLGiaThap	MucTLGiaDong	MucTLMuaBan
1	Muc 1	Muc 1	Muc 0	Muc 0	Muc -2
2	Muc 1	Muc 1	Muc 1	Muc 1	Muc -3
3	Muc 1	Muc 1	Muc 1	Muc 1	Muc -2
4	Muc -1	Muc 1	Muc -1	Muc 1	Muc -1
5	Muc 1	Muc 1	Muc -1	Muc -1	Muc -2
6	Muc -2	Muc -1	Muc -2	Muc -2	Muc -2
7	Muc -1	Muc 0	Muc -1	Muc -1	Muc -1
8	Muc 1	Muc 2	Muc 1	Muc 1	Muc -2
9	Muc 0	Muc 1	Muc 0	Muc 0	Muc 1
10	Muc 2	Muc 2	Muc 0	Muc 0	Muc -3
11	Muc 0	Muc 1	Muc 0	Muc 1	Muc -1
12	Muc -1	Muc -1	Muc -1	Muc -1	Muc 1
13	Muc 0	Muc 0	Muc -1	Muc -1	Muc -2
14	Muc 1	Muc 1	Muc -1	Muc -1	Muc -2
15	Muc 0	Muc 0	Muc -1	Muc -1	Muc 1
16	Muc -1	Muc 0	Muc -1	Muc -1	Muc 1
17	Muc 1	Muc 1	Muc -2	Muc -1	Muc -2
18	Muc -1	Muc -1	Muc -1	Muc -1	Muc 1
19	Muc 0	Muc 0	Muc -1	Muc -1	Muc -1
20	Muc 1	Muc 1	Muc -1	Muc -1	Muc -1
21	Muc 1	Muc 1	Muc -1	Muc -1	Muc -3
22	Muc 0	Muc 0	Muc 0	Muc 0	Muc -2
23	Muc 0	Muc 0	Muc -1	Muc 0	Muc -1
24	Muc 1	Muc 1	Muc 1	Muc 1	Muc 1
25	Muc 0	Muc 0	Muc 0	Muc 0	Muc 2
26	Muc 1	Muc 1	Muc -1	Muc -1	Muc -2
27	Muc 0	Muc 0	Muc -1	Muc -1	Muc -3

Hình 3.5 CSDL được dùng để khai thác

Đây là dữ liệu đã qua bước tiền xử lý, thuật toán của nhóm sử dụng dữ liệu rời rạc, vì vậy tất cả giá trị của giá đều chuyển sang chữ. Toàn bộ quy trình khai thác dữ liệu đều được nhóm áp dụng vào ứng dụng này.

Bước 1

Quá trình làm sạch dữ liệu được thực hiện chung với tích hợp dữ liệu khi tải dữ liệu tự động về từ ba trang web của site Cafef.vn. Đây là chương trình tự động do nhóm tự viết, mô tả chi tiết được trình bày tại phụ lục A. Quá trình xử lý như sau:

- Trường hợp ngày nhằm vào thứ bảy và chủ nhật sẽ không có dữ liệu.
- Trường hợp ngày nhằm vào ngày lễ được nghỉ thì nội dung sẽ rỗng.

Bước 2

Bước chọn và chuyển hoá, dữ liệu được gom cụm để chuyển sang dạng rời rạc với quy tắc gồm hai phần: tỷ lệ các loại giá và tỷ lệ chênh lệch khối lượng mua và bán.

- Tỷ lệ các giá: tỷ lệ tăng giảm của giá mở, giá cao, giá thấp, giá đóng so với giá tham chiếu. Quy tắc thể hiện mức tăng là "+" và giảm là "-".

Bảng 3.1 Tỷ lệ các giá

Tỷ lệ	**Mức**
TL < -6%	-3
-6% <= TL < -3%	-2
-3% <= TL < 0%	-1
TL = 0%	0
0% < TL <= 3%	1
3% < TL <= 6%	2
TL > 6%	3

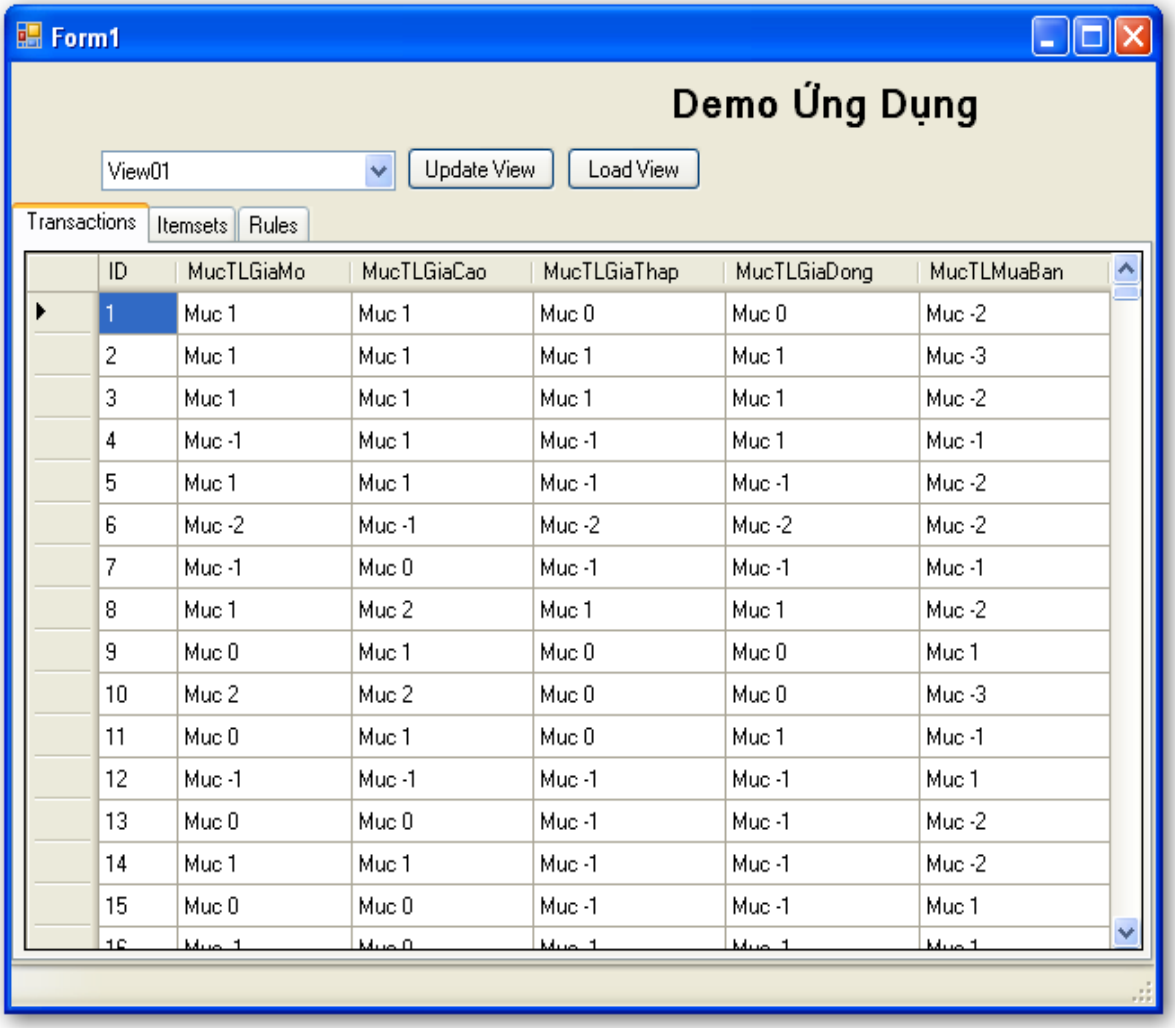
- Tỷ lệ chênh lệch mua và bán

Bảng 3.2 Tỷ lệ mua và bán

Tỷ lệ	**Mức**
TL < -60%	-3
-60% <= TL < -30%	-2
-30% <= TL < 0%	-1
TL = 0%	0
0% < TL <= 30%	1
30% < TL <= 60%	2
TL > 60%	3

Chỉ có một bảng nên Case cũng là bảng đó. Nhóm sử dụng Key là ID, Input là MucTLGiaMo, MucTLGiaCao, MucTLGiaThap, MucTLGiaĐong, và Predict_Only là MucTLMuaBan.

Kết quả ra được như hình 3.6.



ID	MucTLGiaMo	MucTLGiaCao	MucTLGiaThap	MucTLGiaDong	MucTLMuaBan
1	Muc 1	Muc 1	Muc 0	Muc 0	Muc -2
2	Muc 1	Muc 1	Muc 1	Muc 1	Muc -3
3	Muc 1	Muc 1	Muc 1	Muc 1	Muc -2
4	Muc -1	Muc 1	Muc -1	Muc 1	Muc -1
5	Muc 1	Muc 1	Muc -1	Muc -1	Muc -2
6	Muc -2	Muc -1	Muc -2	Muc -2	Muc -2
7	Muc -1	Muc 0	Muc -1	Muc -1	Muc -1
8	Muc 1	Muc 2	Muc 1	Muc 1	Muc -2
9	Muc 0	Muc 1	Muc 0	Muc 0	Muc 1
10	Muc 2	Muc 2	Muc 0	Muc 0	Muc -3
11	Muc 0	Muc 1	Muc 0	Muc 1	Muc -1
12	Muc -1	Muc -1	Muc -1	Muc -1	Muc 1
13	Muc 0	Muc 0	Muc -1	Muc -1	Muc -2
14	Muc 1	Muc 1	Muc -1	Muc -1	Muc -2
15	Muc 0	Muc 0	Muc -1	Muc -1	Muc 1

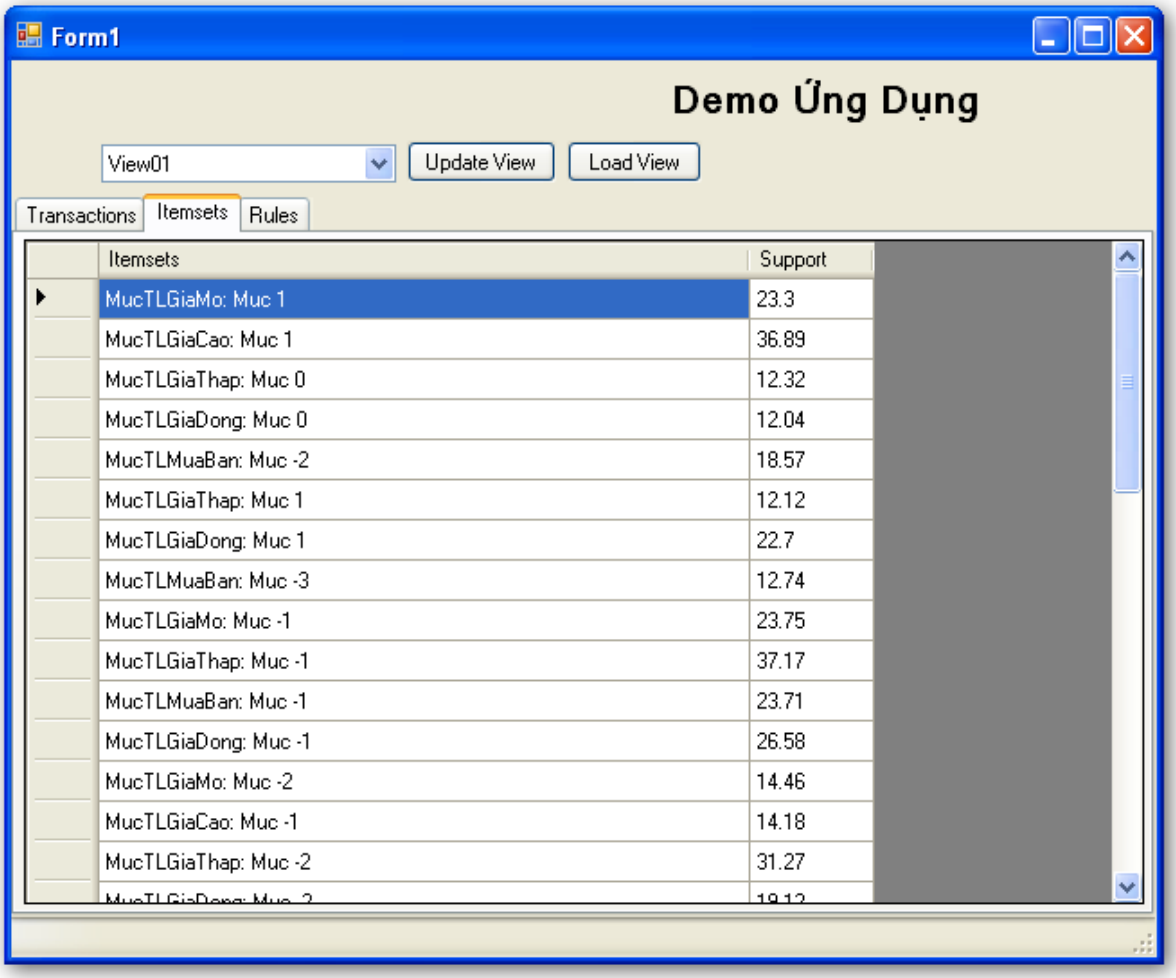
Hình 3.6 Giao diện Tab Transactions của ứng dụng

Transactions là dữ liệu rời rạc sau khi chuyển hóa từ CSDL.

Bước 3

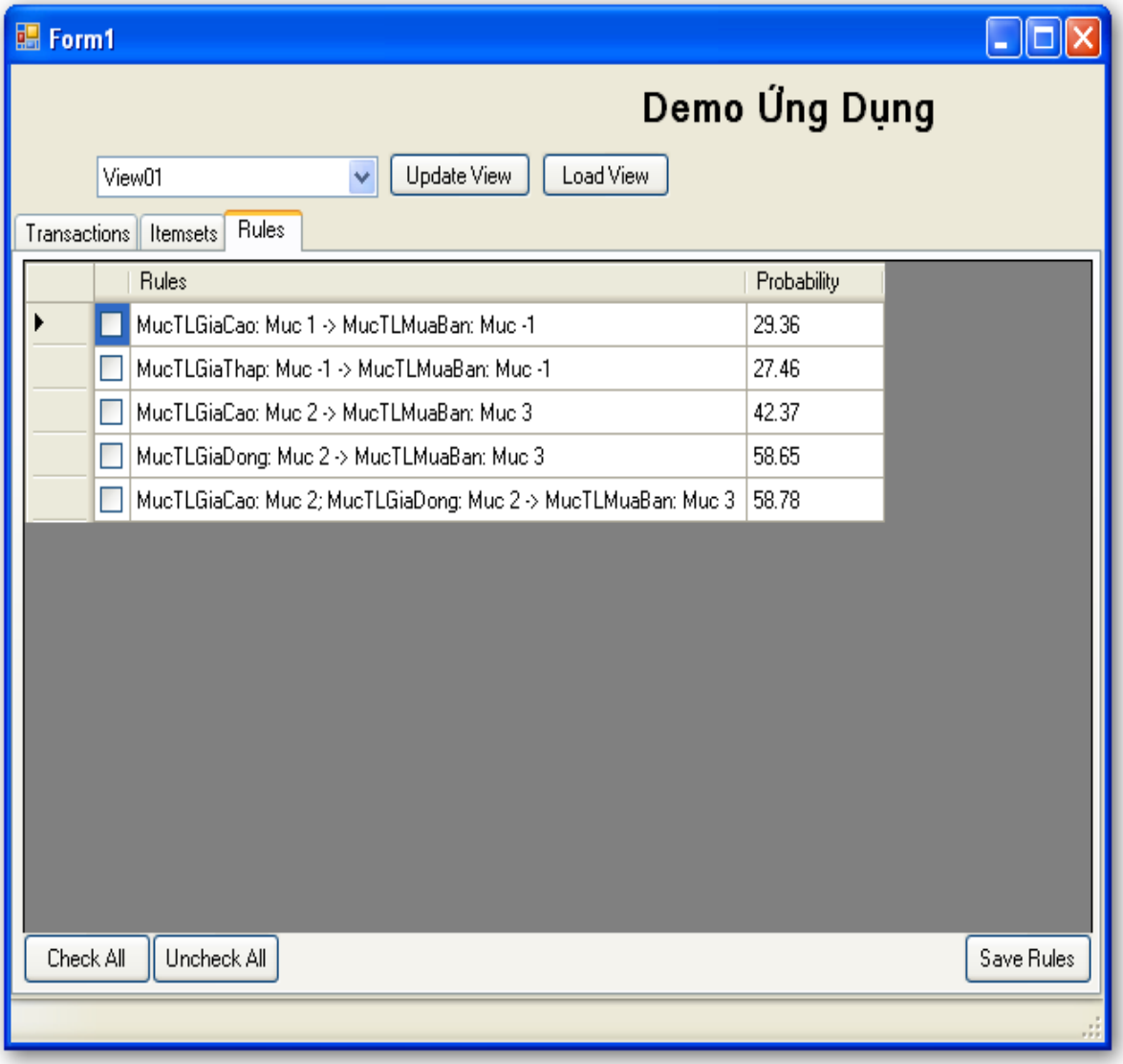
Thực hiện khai thác dữ liệu bằng cách bấm nút "Update View". Hai ngưỡng giá trị để đánh giá mẫu là độ phổ biến nhỏ nhất và độ tin cậy nhỏ nhất, cả hai đều bằng 10%. Kết quả khai thác được như hình 3.7 và hình 3.8.

Bước 4



Hình 3.7 Giao diện Tab Itemsets của ứng dụng

Các bộ dữ liệu được trình bày chung với thông tin về độ phổ biến có đơn vị là (%).



Hình 3.8 Giao diện Tab Rules của ứng dụng

Ở hình 3.8, ta có thể thực hiện các thao tác cơ bản là chọn hoặc bỏ chọn tất cả bằng nút "Check All" và "Uncheck All". Sau đó chọn "Save Rules" để lưu lại các luật được chọn. Các luật này được lưu trong bảng CSDL do nhóm ấn định trước.

4. KẾT QUẢ, ĐÁNH GIÁ KẾT QUẢ VÀ HƯỚNG MỞ RỘNG

Kết quả đạt được

Về nội dung

Nhóm đã đạt được hầu hết các mục tiêu đề ra. Cụ thể như sau:

1. Nhóm đã biết được dữ liệu chứng khoán là loại dữ liệu chuỗi thời gian.
2. Nhóm đã xây dựng ứng dụng tự động tải dữ liệu chứng khoán.
3. Nhóm đã khảo sát và nghiên cứu chi tiết về thuật toán Bide là một thuật toán phát triển từ sau thuật toán Clospan.
4. Nhóm có thể hiểu Data Mining Engine trong SQL.
5. Nhóm đã tìm hiểu về các thuật toán được sử dụng trong Analysis Services của Microsoft SQL Server 2008.
6. Nhóm đã cấu hình và cài đặt thành công thuật toán tích hợp Apriori và thuật toán tích hợp Bide vào Analysis Services.
7. Nhóm đã xây dựng ứng dụng độc lập sử dụng thuật toán tích hợp vào Analysis Services để khai thác dữ liệu chứng khoán.

Về cách làm việc

- Nhóm có thể viết báo cáo đúng chuẩn ISO.
- Biết cách sử dụng các phầm mềm giúp tăng hiệu quả làm việc nhóm (Teamviewer, Dropbox).

Ưu điểm

- Thuật toán tích hợp và ứng dụng độc lập được nhóm kiểm thử trên ba hệ điều hành là Window XP, Vista và Seven.
- Cấu trúc của hai thuật toán tích hợp được lập trình theo dạng mẫu thiết kế Factory Method.

Khuyết điểm

- Do thời gian hạn chế nên nhóm chưa hoàn thiện được ứng dụng độc lập, cụ thể là dữ liệu đầu vào phải cố định trong mã nguồn, không có giao diện tùy ý cho người dùng.
- Vì lĩnh vực nhóm tiếp cận còn mới nên nguồn tài liệu sử dụng là tiếng Anh, trong quá trình dịch các thuật ngữ có thể chưa được chính xác.

- Vì kỹ thuật lập trình còn hạn chế nên mã chưa được tối ưu, dẫn đến tốc độ của Bide không nhanh hơn được Apriori.

Hướng mở rộng

- Tiếp tục hoàn thiện giao diện ứng dụng độc lập để giúp người dùng tùy chọn dữ liệu cần khai thác.
- Phát triển thêm phần xử lý tri thức tìm được nhằm đẩy ứng dụng lên một bậc là hệ chuyên gia tư vấn đầu tư chứng khoán.
- Tiếp tục nghiên cứu áp dụng các thuật toán mới khai thác dữ liệu chứng khoán hiệu quả nhất.

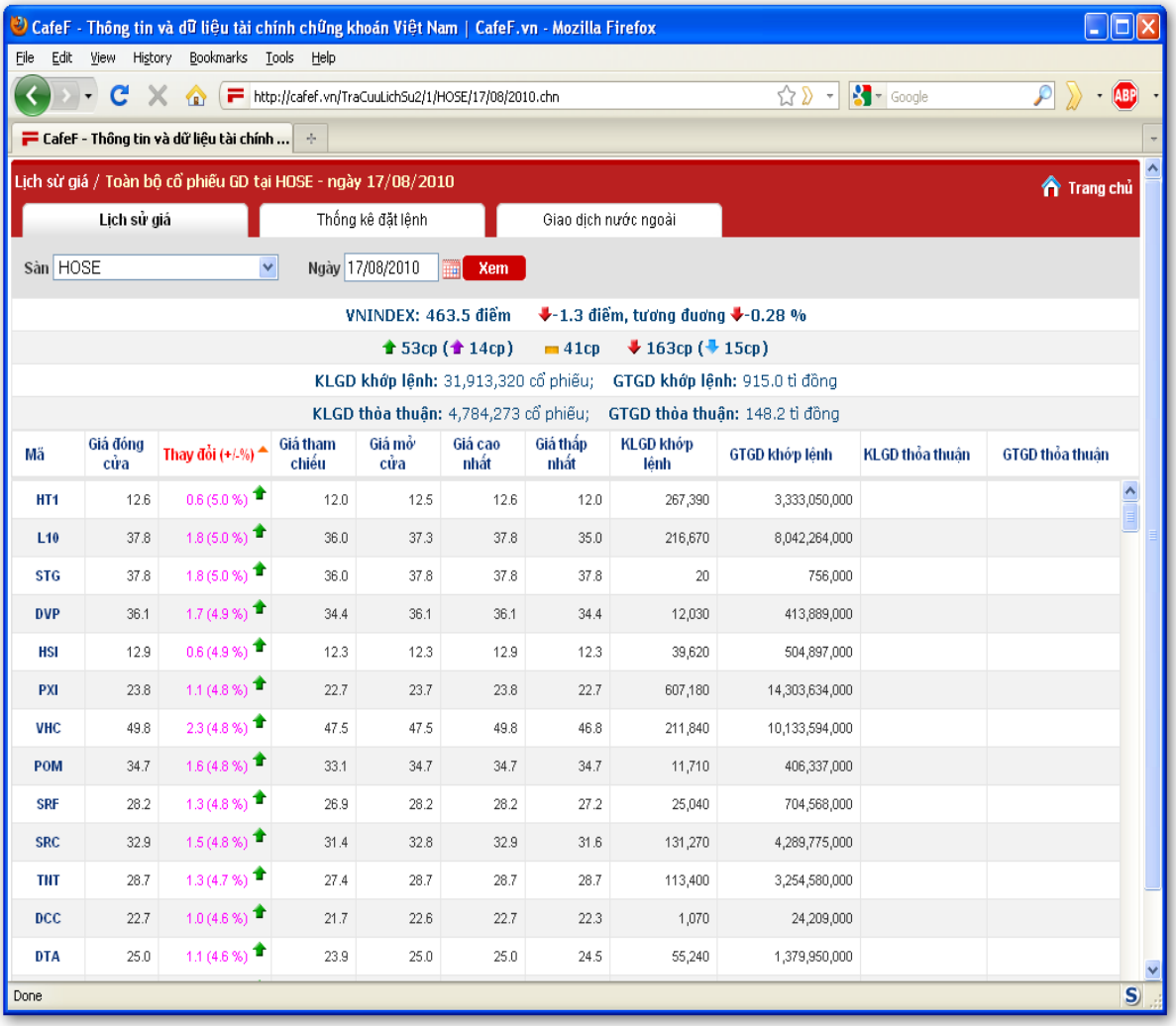
5. PHỤ LỤC

5.1. Phụ lục A: Ứng dụng tải dữ liệu tự động và CSDL

5.1.1. Giới thiệu trang web Cafef

Dữ liệu được sử dụng cho chương trình nằm trong ba trang web của Cafef.vn:

- Trang lịch giá giá: bao gồm thông tin về giao dịch khớp lệnh của các cổ phiếu.



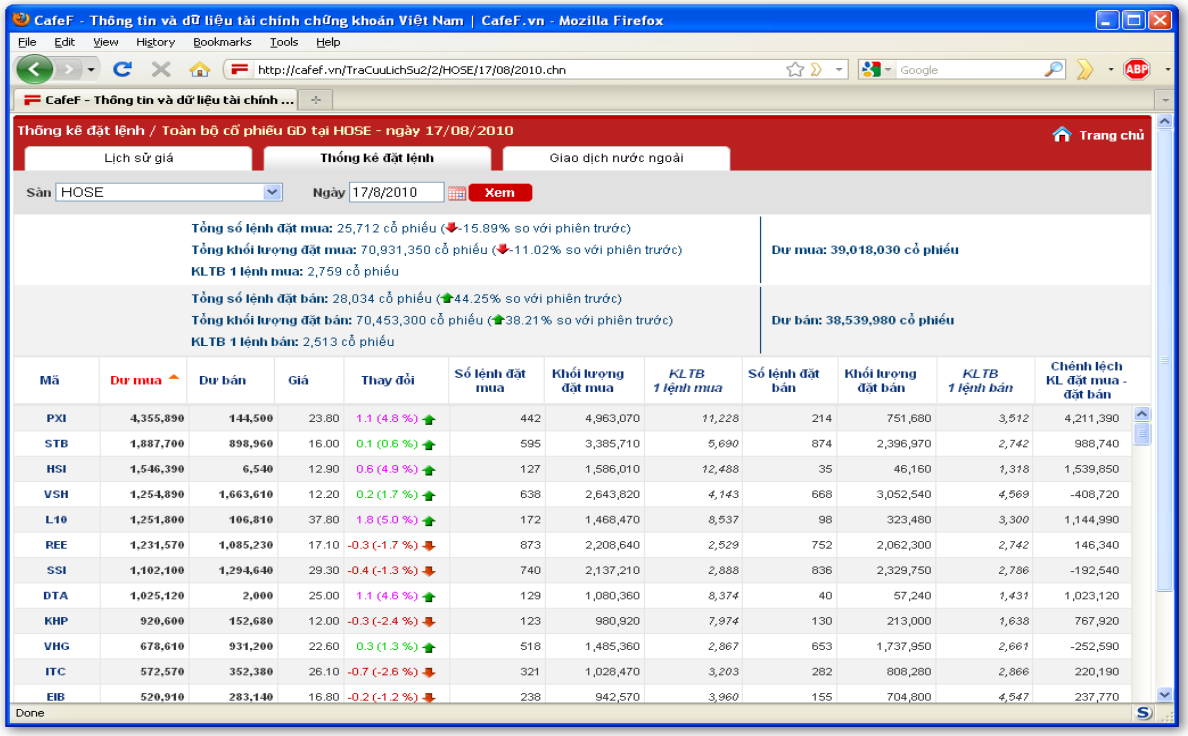
Hình 5.1 Trang web dữ liệu chứng khoán của Cafef trên sàn HOSE

Địa chỉ trang web có dạng:

http://cafef.vn/TraCuuLichSu2/**1/HOSE/DD/MM/YYYY**.chn.

Ví dụ: lịch sử giá của sàn HOSE trong ngày 17/08/2010 có địa chỉ trang web là: http://cafef.vn/TraCuuLichSu2/1/HOSE/17/08/2010.chn.

- Thống kê đặt lệnh**:** bao gồm thông kê chi tiết việc đặt lệnh mua bán của các mã.

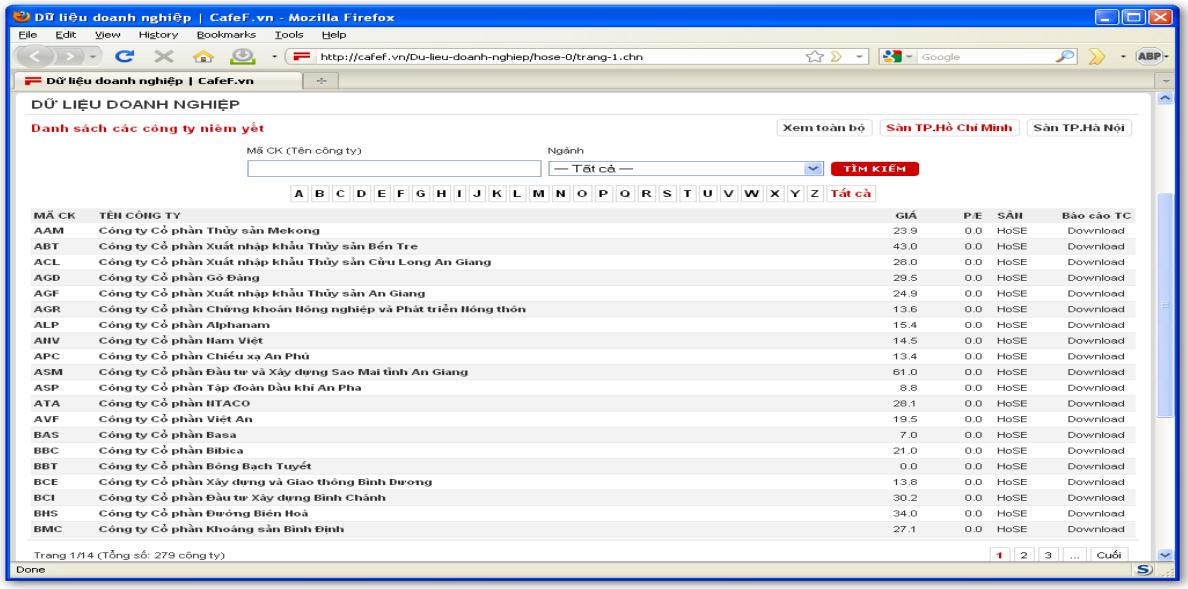


Hình 5.2 Trang web thống kê đặt lệnh của Cafef trên sàn HOSE

Địa chỉ trang web có dạng:
http://cafef.vn/TraCuuLichSu2/**2/HOSE/DD/MM/YYYY**.chn.

Ví dụ: Thống kê đặt lệnh của sàn HOSE trong ngày 17/08/2010 có địa chỉ trang web là: http://cafef.vn/TraCuuLichSu2/2/HOSE/17/08/2010.chn.

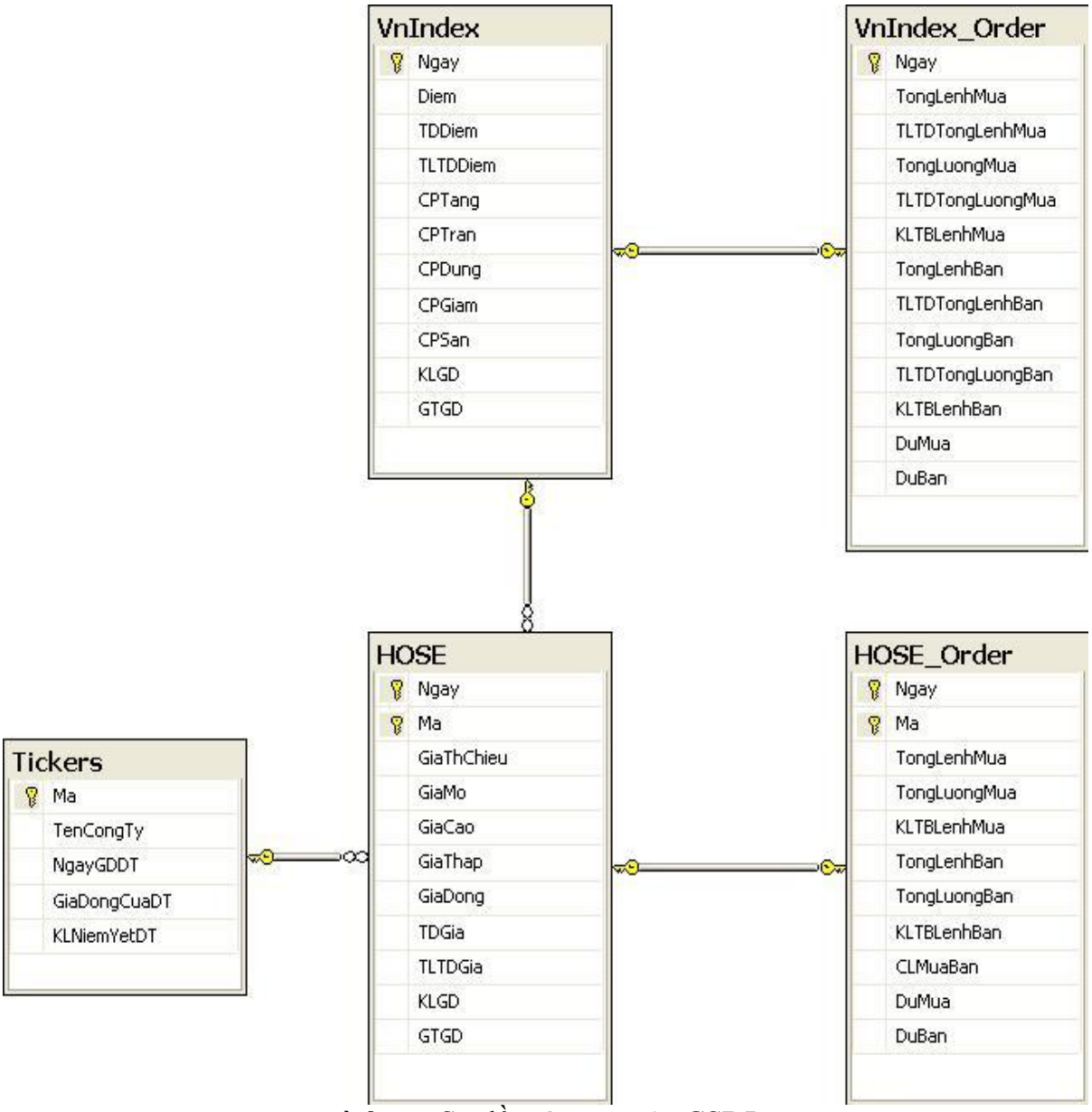
- Dữ liệu doanh nghiệp**:** danh sách các doanh nghiệp đã lên sàn.



Hình 5.3 Trang web các công ty lên sàn của Cafef

Địa chỉ trang web là: http://cafef.vn/Du-lieu-doanh-nghiep/hose-0/trang-1.chn.

5.1.2. Cơ sở dữ liệu



Hình 5.4 Sơ đồ toàn cục của CSDL

Gồm có 5 bảng dữ liệu:

Bảng 5.1 VnIndex: thống kê điểm của VnIndex

Tên dữ liệu	Loại dữ liệu	Ghi chú
Ngay	DateTime	Ngày
Diem	Real	Điểm
TDDiem	Real	Thay đổi điểm
TLTDDiem	Real	Tỷ lệ thay đổi điểm
CPTang	SmallInt	Tổng cổ phiếu tăng giá (giá đóng cửa so với giá tham chiếu)
CPTran	SmallInt	Tổng cổ phiếu tăng giá bằng với giá trần
CPDung	SmallInt	Tổng cổ phiếu đứng giá
CPGiam	SmallInt	Tổng cổ phiếu giảm giá
CPSan	SmallInt	Tổng cổ phiếu giảm giá bằng với giá sàn
KLGD	BigInt	Tổng khối lượng giao dịch
GTGD	BigInt	Tổng giá trị giao dịch

Bảng 5.2 HOSE: thống kê các loại giá, giao dịch của các cổ phiếu

Tên dữ liệu	Loại dữ liệu	Ghi chú
Ngay	DateTime	Ngày
Ma	Varchar(10)	Mã
GiaThChieu	Real	Giá tham chiếu
GiaMo	Real	Giá mở cửa
GiaCao	Real	Giá cao nhất
GiaThap	Real	Giá thấp nhất
GiaDong	Real	Giá đóng cửa
TDGia	Real	Thay đổi giá
TLTDGia	Real	Tỷ lệ thay đổi giá
KLGD	BigInt	Khối lượng giao dịch
GTGD	BigInt	Giá trị giao dịch

Bảng 5.3 VnIndex_Order: thống kê việc đặt lệnh của VnIndex

Tên dữ liệu	Loại dữ liệu	Ghi chú
Ngay	DateTime	Ngày
TongLenhMua	BigInt	Tổng số lệnh đặt mua
TLTDTongLenhMua	Real	Tỷ lệ thay đổi tổng số lệnh đặt mua so với ngày trước
TongLuongMua	BigInt	Tổng khối lượng đặt mua
TLTDTongLuongMua	Real	Tỷ lệ thay đổi của tổng khối lượng đặt mua
KLTBLenhMua	BigInt	Khối lượng trung bình một lệnh mua
TongLenhBan	BigInt	Tổng số lệnh đặt bán
TLTDTongLenhBan	Real	Tỷ lệ thay đổi của tổng số lệnh đặt bán
TongLuongBan	BigInt	Tổng khối lượng đặt bán
TLTDTongLuongBan	Real	Tỷ lệ thay đổi của tổng khối lượng đặt bán
KLTBLenhBan	BIgInt	Khối lượng trung bình một lệnh bán
DuMua	BigInt	Dư mua
DuBan	BigInt	Dư bán

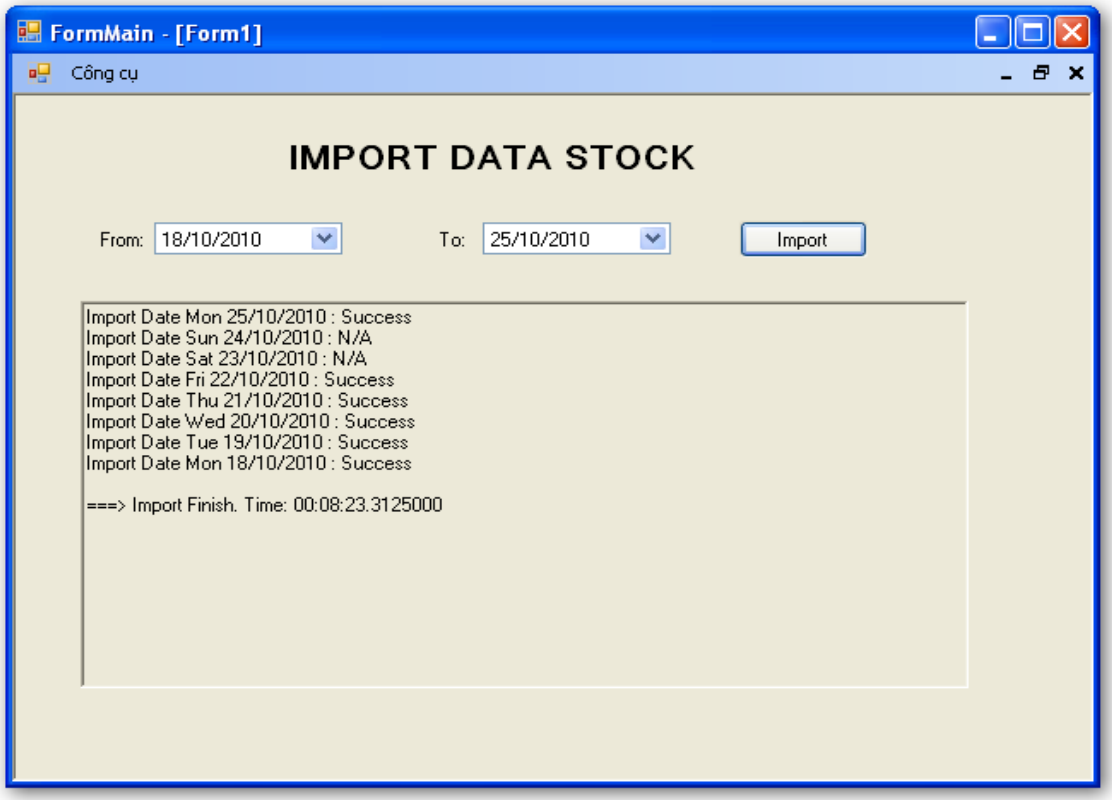
Bảng 5.4 Tickers: danh sách các doanh nghiệp

Tên dữ liệu	Loại dữ liệu	Ghi chú
Ma	varchar(10)	Mã cổ phiếu
TenCongTy	nvarchar(200)	Tên công ty
NgayGDDT	datetime	Ngày giao dịch đầu tiên
GiaDongCuaDT	real	Giá đóng cửa phiên giao dịch đầu tiên
KLNiemYetDT	bigint	Khối lượng cổ phiếu niêm yết lần đầu

Bảng 5.5 HOSE_Order: thống kê việc đặt lệnh của các cổ phiếu

Tên dữ liệu	Loại dữ liệu	Ghi chú
Ngay	DateTime	Ngày
Ma	Varchar(10)	Mã
TongLenhMua	BigInt	Tổng số lệnh đặt mua
TongLuongMua	BigInt	Tổng khối lượng đặt mua
KLTBLenhMua	BigInt	Khối lượng trung bình một lệnh mua
TongLenhBan	BigInt	Tổng số lệnh đặt bán
TongLuongBan	BigInt	Tổng khối lượng đặt bán
KLTBLenhBan	BIgInt	Khối lượng trung bình một lệnh bán
CLMuaBan	BigInt	Chênh lệch khối lượng mua bán
DuMua	BigInt	Dư mua
DuBan	BigInt	Dư bán

5.1.3. Chương trình Import dữ liệu



Hình 5.5 Giao diện chương trình Import

Người dùng chọn ngày bắt đầu và ngày kết thúc để chương trình tiến hành import dữ liệu.

Chương trình sẽ tải trang web của Cafef.vn và đọc nội dung của trang web, import dữ liệu vào cơ sở dữ liệu.

- Trang lịch sử giá: những Html Element ID trong trang lịch sử giá cần chú ý nằm trong bảng 5.6.

Bảng 5.6 Html Element ID cần lưu ý của trang lịch sử giá

Html Element ID	Content
div_cf_BoxContent	Trang chủ
	Lịch sử giá / Toàn bộ cổ phiếu GD tại HOSE - ngày 02/12/2009 Lịch sử giá, Thống kê đặt lệnh, Giao dịch nước ngoài
ctl18_lblIndexPoint	VNINDEX: 499.1 điểm -15.8 điểm, tương đương -3.07 %
ctl18_divToTalUp	59cp
ctl18_spanKichTran	(0cp)
ctl18_divTotalNochange	41cp
ctl18_divTotalDown	56cp
ctl18_spanKichSan	(0cp)
ctl18_div1	KLGD khớp lệnh: 5,339,020 cổ phiếu; GTGD khớp lệnh: 115.5 tỷ đồng
ctl18_div2	KLGD thỏa thuận: 292,030 cổ phiếu; GTGD thỏa thuận: 15.6 tỷ đồng
table2sort	CSG 8.2 0.5 (6.5 %) 7.7 8.0 8.2 7.9 26,300 213,300,000 6.5
	SVC 12.1 0.7 (6.1 %) 11.4 11.7 12.1 11.5 29,500 354,860,000 6.1
	BBT 4.2 0.2 (5.0 %) 4.0 4.0 4.2 4.2 2,720 11,424,000 5.0
	LCG 44.1 2.1 (5.0 %) 42.0 44.1 44.1 43.8 92,860 4,094,726,000 5.0
	TDH 30.1 1.4 (4.9 %) 28.7 30.1 30.1 30.1 74,490 2,242,149,000 4.9

Ta dùng WebBrowser để tải trang lịch sử giá về. Ta sẽ truyền Element ID vào để WebBrowser rút ra nội dung trong ID đó. Một Element ID tương ứng với một nội dung chứa trong Element đó. Để rút ra được giá trị mong muốn thì một dòng nội dung, với mỗi khoảng trắng ta tách ra thành mảng bằng: Split(' '). Ta lưu dữ liệu bằng vị trí của mảng đó.

- Trang thống kê đặt lệnh: những Html Element ID cần lưu ý trong bảng 5.7.

Bảng 5.7 Html Element ID cần lưu ý của trang thống kê đặt lệnh

Html Element ID	Content
div_cf_BoxContent	Trang chủ
	Thống kê đặt lệnh / Toàn bộ cổ phiếu GD tại HOSE - ngày 02/12/2009 Lịch sử giá, Thống kê đặt lệnh, Giao dịch nước ngoài
ctl18_div1	Tổng số lệnh đặt mua: 32,412 cổ phiếu (-10.41% so với phiên trước)
ctl18_div2	Tổng khối lượng đặt mua: 72,679,210 cổ phiếu (-2.98% so với phiên trước)
ctl18_div5	Khối lượng trung bình một lệnh mua: 2,242 cổ phiếu
ctl18_divDumua	Dư mua: 21,081,490 cổ phiếu
ctl18_div3	Tổng số lệnh đặt bán: 41,430 cổ phiếu (71.34% so với phiên trước)
ctl18_div4	Tổng khối lượng đặt bán: 107,314,370 cổ phiếu (81.87% so với phiên trước)
ctl18_div6	Khối lượng trung bình một lệnh bán: 2,590 cổ phiếu
ctl18_divDuban	Dư bán: 55,716,650 cổ phiếu
table2sort	EIB 3,524,460 8,245,460 24.30 -0.1 (-0.4 %) 3,731 13,314,950 3,568 4,106 18,035,950 4,392 -4,721,000
	DIG 1,096,670 874,070 99.00 -2.0 (-2.0 %) 665 2,132,150 3,206 591 1,909,550 3,231 222,600
	PGD 1,091,010 248,920 62.00 2.5 (4.2 %) 630 2,029,650 3,221 236 1,187,560 5,032 842,090
	VFMVF1 965,240 1,991,690 16.00 -0.4 (-2.4 %) 706 3,067,640 4,345 757 4,094,090 5,408 -1,026,450
	STB 893,820 6,214,800 25.30 -1.3 (-4.9 %) 1,892 4,967,800 2,625 2,618 10,288,780 3,930 -5,320,980

- Trang Dữ liệu Doanh Nghiệp

Bảng 5.8 HOSE_Order: thống kê việc đặt lệnh của các cổ phiếu

Html Element ID
CafeF_ThiTruongNiemYet_TongSoTrang
CafeF_ThiTruongNiemYet_Content

Ta sẽ dùng ID CafeF_ThiTruongNiemYet_TongSoTrang để biết được tổng số trang. Từ đó, ta sẽ dùng vòng lặp tải từng trang về. Sau đó, ta sẽ đọc bảng dữ liệu trong ID CafeF_ThiTruongNiemYet_Content với mã và tên công ty. Lưu dữ liệu đó vào CSDL. Mã nguồn của chương trình được ghi trong đĩa CD.

5.2. Phụ lục B: Chi tiết khảo sát các thuật toán Sequential Pattern Mining

Các site xác thực hội nghị

Wikipedia cung cấp danh sách các hội nghị về khoa học máy tính, kèm theo liên kết tham thảo đánh giá các hội nghị theo chuyên đề nhất định. http://en.wikipedia.org/wiki/List_of_computer_science_hội nghis#References

Với chuyên đề khai thác dữ liệu ta có thể tìm được tất cả hội nghị liên quan. Từ các "Links References", ta thấy Microsoft Academic đánh giá các hội nghị, tạp chí chuyên đề, tổ chức, bài báo và các tác giả lớn trong lĩnh vực khai thác dữ liệu (Srikant thứ ba, Jiawei Han thứ hai sau Agrawal) theo thứ tự từ trên xuống.
http://academic.research.microsoft.com/CSDirectory/Author_category_7.htm

Ngoài ra, ta có danh sách các hội nghị mà Jiawei Han tham dự cũng như nêu tại đây
http://www.cs.sfu.ca/~han/conf.html
http://www.cs.uiuc.edu/homes/hanj/pubs/index.htm

Bên cạnh Microsoft Academic, có nhiều đánh giá khác trong các trang web của các tác giả lớn trong phần "References" của Wikipedia.

Một trang web khác cho ta thấy được phần nào so sánh chất lượng giữa các hội nghị về CSDL và khai thác dữ liệu bằng cách thể hiện tỷ lệ bài báo được chấp nhận so với số bài báo được đề nghị.
http://wwwhome.cs.utwente.nl/~apers/rates.html

Các site xác thực bài báo

http://portal.acm.org/portal.cfm trang web này cho thấy thông tin nhanh của bài báo được tìm thấy mà chưa rõ nguồn gốc (tác giả, năm, hội nghị…). Thực chất đây là một thư viện hỗ trợ cho các tổ chức có nhu cầu nghiên cứu, nếu muốn tải về phải tốn phí.

Bên cạnh đó, các trang web phổ biến cũng được dùng để tìm bài báo là: Google, Citeseer, Docjax.

DBLP Bibliography - Home Page trang web này liệt kê tất cả kỉ yếu hội nghị, các hội thảo và tiểu luận đi kèm, và các tạp chí chuyên đề. Ta có thể tìm được tất cả bài báo trong một hội nghị xác định.

Các hội nghị đã tìm kiếm

ACM-SIGMOD International Conference on Management of Data (SIGMOD).
ACM SIGMOD-SIGACT-SIGART International Symposium on Principles of Database Systems (PODS).
International Conference on Very Large Data Bases (VLDB).
International Conference on Data Engineering (ICDE).
International Conference on Knowledge Discovery and Data Mining (KDD).
SIAM Int. Conf. on Data Mining (SIAMDM).
Int. Conf. on Extending Data Base Technology (EDBT).
Int. Conf. on Database Theory (ICDT).

IEEE Int. Conf. on Data Mining (ICDM).
Int. Conf. on Database Systems for Advanced Applications (DASFAA).
Int. Conf. on Information and Knowledge Management (CIKM).
Int. Conf. on Data Warehousing and Knowledge Discvery (DaWak).
Int. Database Engineering and Applications Symposium (IDEAS).

Các tạp chí khoa học đã tìm kiếm

Computer Science and Technology.
Intelligent Information Systems.
Systems and Software.
IEEE Transactions on Knowledge and Data Engineering (IKDE).
Computer and System Sciences (JCSS).
Scientific Research (EJSR).
Computers.

Độ chính xác của khảo sát

Đây là các hội nghị và tạp chí khoa học nổi tiếng, được đánh giá cao tại các trang web xếp hạng. Các hội nghị này cũng có nhiều chuyên đề về khai thác dữ liệu, cụ thể là SPAM. Bên cạnh đó, tất cả hội nghị trên là hầu hết các hội nghị mà tác giả Jiawei Han tham dự, đây là tác giả khá nổi tiếng sau hai tác giả kinh điển đặt ra vấn đề SPAM là Agrawal & Srikant (1995), như trang web Microsoft Academic cũng đã sắp hạng tác giả này trên cả Srikant. Chính vì vậy đối với các hội nghị chuyên ngành về khai thác dữ liệu nói chung như trên, tin rằng khi tìm sẽ khó có thể bỏ sót bài báo dẫn đến khảo sát không chính xác.

Bài khảo sát sẽ phân thành hai mục: mục tác giả Jiawei Han và mục các tác giả khác.

Bài khảo sát này cũng tham khảo các khảo sát của những tác giả khác thực hiện về SPAM trong những năm gần đây. Vì vậy, tính chính xác của bài là chấp nhận được, đặc biệt về mặt liệt kê các bài báo được trình bày trong các hội nghị và chuyên đề khoa học.

Tất cả tài liệu được ghi trong đĩa CD, thích hợp xem ở “List” và “Maximized Window”, với quy ước ghi như sau: Năm_Tên bài báo_Tên hội nghị.

JIAWEI HAN

TSP: Mining Top-K Closed Sequential Patterns
(ICDM 2003)

Mining Frequent Patterns without Candidate Generation A Frequent-Pattern Tree Approach
(journal Data Mining and Knowledge Discovery (DAMI) 2004)

BIDE: Efficient Mining of Frequent Closed Sequences
(ICDE 2004)

From sequential pattern mining to structured pattern mining: a pattern-growth approach
(Journal of Computer Science and Technology 2004)

IncSpan: Incremental Mining of Sequential Patterns in Large Database
(KDD 2004)

Parallel Mining Of Closed Sequential Patterns
(KDD 2005)

A Sampling-based Framework For Parallel Data Mining
(ACM SIGPLAN symposium on Principles and practice of parallel programming 2005)

SeqIndex: Indexing Sequences by Sequential Pattern Analysis
(SDM (SIAMDM) 2005)

Mining Compressed Frequent-Pattern Sets
(VLDB 2005)

Constraint-based sequential pattern mining: the pattern-growth methods
(Journal of Intelligent Information Systems (JIIS) 2007)

Efficient Discovery of Frequent Approximate Sequential Patterns
(ICDM 2007)

Mining Colossal Frequent Patterns by Core Pattern Fusion
(ICDE 2007)

Frequent Closed Sequence Mining without Candidate Maintenance
(journal IEEE Transactions on Knowledge and Data Engineering – IEEE TKDE 2007)
BIDE được trình bày lại

CISpan: Comprehensive Incremental Mining Algorithms of Closed Sequential Patterns for Multi-Versional Software Mining
(SDM (SIAMDM) 2008)

Efficient Mining of Closed Repetitive Gapped Subsequences from a Sequence Database
(ICDE 2009)

CÁC TÁC GIẢ KHÁC

Mining Sequential Patterns with Item Constraints
(DaWak 2004)

Generalized Sequential Pattern Mining with Item Intervals
(Journal of computers 2004)

Scalable sequential pattern mining for biological sequences
(CIKM 2004)

Sequential pattern mining with approximated constraints
(Int. Conf Applied Computing (IADIS) 2004)

SQUIRE: Sequential Pattern Mining with Quantities
(ICDE 2004)

An Efficient Algorithm for Mining Frequent Sequences by a New Strategy without Support Counting
(ICDE 2004)

Sequential Pattern Mining in Multiple Streams
(ICDM 2005)

Processing Sequential Patterns in Relational Databases
(Dawak 2005)

Striking Two Birds With One Stone Simultaneous Mining of Positive and Negative Spatial Patterns
(SDM (SIAMDM) 2005)

Summarizing Sequential Data with Closed Partial Orders
(SDM (SIAMDM) 2005)

Efficient Mining of Maximal Sequential Patterns Using Multiple Samples
(SDM (SIAMDM) 2005)

CBS A New Classification Method by Using Sequential Patterns
(SDM (SIAMDM) 2005)

Efficient Sequential Pattern Mining Algorithms
(4th WSEAS 2005)

LAPIN-SPAM: An Improved Algorithm for Mining Sequential Pattern
(ICDE 2005 workshop)

Effective Database Transformation and Efficient Support Computation for Mining Sequential Patterns
(DASFAA 2005)

Mining sequential patterns from data streams a centroid approach
(Journal of Intelligent Information Systems (JIIS) 2006)

Using sequential pattern mining for links recommendation in adaptive hypermedia educational systems
(Current Developments in Technology-Assisted Education 2006)

Privacy preserving sequential pattern mining in distributed databases
(CIKM 2006)

Efficient mining of max frequent patterns in a generalized environment
(CIKM 2006)

COBRA: Closed Sequential Pattern Mining Using Bi-phase Reduction Approach
(DaWak 2006)

HYPE: Mining Hierarchical Sequential Pattern Mining
(ACM international workshop on Data warehousing and OLAP 2006)

Effective Sequential Pattern Mining Algorithms for Dense Database
(National Data Engineering WorkShop (DEWS) 2006)
LAPIN được trình bày lại

PAID: Mining Sequential Patterns by Passed Item Deduction in Large Databases
(IDEAS 2006)

Closed Multidimensional Sequential Pattern Mining
(ITNG 2006)

SQUIRE: Sequential Pattern Mining with Quantities
(Journal of Systems and Software 2007)

IMCS: incremental mining of closed sequential patterns
(8th international conf. on web-age information management conf. on Advances in data and web management 2007)

Effective Sequential Pattern Mining Algorithms by Last Position Induction for Dense Databases
(DASFAA 2007)
LAPIN được trình bày lại

PRISM: A Prime-Encoding Approach for Frequent Sequence Mining
(ICDM 2007)

A General Model for Sequential Pattern Mining with a Progressive Database
(journal IEEE Transactions on Knowledge and Data Engineering – IEEE TKDE 2008)

Mining and Ranking Generators of Sequential Patterns
(SDM 2008)

Efficient Mining of Recurrent Rules from a Sequence Database
(DASFAA 2008)

Scalable complex pattern search in sequential data
(CIKM 2008)

Efficient Frequent Pattern Mining over Data Streams
(CIKM 2008)

Mining sequential patterns by Prefixspan algorithm with approximation
(WSEAS ACS 2008)

Prism An effective approach for frequent sequence mining via prime-block encoding
(Journal of Computer and System Sciences (JCSS) 2009 2010)

Mining Sequential Patterns Using Hybrid Evolutionary Algorithm
(World Academy of Science, Engineering and Technology 2009)

Mining Complex Spatio-Temporal Sequence Patterns
(SDM (SIAMDM)2009)

Mining Constraint-based Multidimensional Frequent Sequential Pattern in Web Logs
(European Journal of Scientific Research (EJSR) 2009 volume 36 issue 3)

Vertical Mining of Frequent Patterns from Uncertain Data
(JCIS 2009)

Frequent Pattern Mining with Uncertain Data
(ACM SIGKDD 2009)

Efficient algorithms for mining constrained frequent patterns from uncertain data
(ACM SIGKDD Workshop 2009)

Margin-Closed Sequential Pattern Mining
(KDD 2010-ACM SIGKDD 2010 workshop)

A Sequential Pattern Mining Algorithm using Rough Set Theory
(kk-grc2010-ex)

Lưu ý: nhiều bài báo được trình bày đầu tiên tại một hội nghị, sau đó lại được trình bày tiếp vào hội nghị hoặc chuyên đề khoa học khác vào năm mới hơn, nhưng trong khoảng thời gian giữa lần trình bày đầu tiên và lần trình bày trong hội nghị khác, một bài báo khác trình bày cải tiến từ thuật toán trong bài báo cũ, thành ra năm trình bày của thuật toán cải tiến lại nằm trước năm trình bày của thuật toán được cải tiến. Ví dụ: FP-Tree là gốc, về sau mới đến Prefixspan (2001), Clospan (2003), tuy nhiên lại trình bày năm 2004 tại DAMI dễ gây hiểu lầm.

Với bài khảo sát này, mặc dù không tìm kiếm toàn bộ hội nghị quốc tế có chuyên đề liên quan khai thác dữ liệu, cụ thể là SPAM, nhưng có thể khẳng định nắm bắt được phần cốt lõi là xu hướng, quá trình phát triển chính của SPAM tới thời điểm hiện tại.

5.3. Phụ lục C: Mô tả chức năng các lớp và hàm

Các phim hướng dẫn, mã nguồn Shell và ứng dụng được ghi trong đĩa CD.

5.3.1. Chức năng các lớp và hàm của khung Shell

Bảng 5.9 Lớp khung Metadata.cs

Metadata.cs	
static public MiningParameterCollection DeclareParameters()	Khởi tạo Parameter
public override string GetServiceName()	Xuất tên ServiceName
public override string GetDisplayName()	Xuất tên hiển thị
public override string GetServiceDescription()	Xuất mô tả Service
public override PlugInServiceType GetServiceType()	Loại thuật toán
public override string GetViewerType()	Kiểu hiện thị kết quả của thuật toán
public override MiningScaling GetScaling()	Phạm vi Training Cases của thuật toán
public override MiningTrainingComplexity GetTrainingComplexity()	Thời gian thực thi Traning Cases của thuật toán
public override MiningPredictionComplexity GetPredictionComplexity()	Thời gian thực thi Predict sau khi Traning Cases của thuật toán.
public override MiningExpectedQuality GetExpectedQuality()	Chất lượng của kết quả sau khi thực thi Predict
public override bool GetSupportsDrillThrough()	Cho phép xem thông tin nội bộ bên dưới của Case, theo dạng khoan cắt
public override bool GetCaseIdModeled()	Trả về True nếu mỗi Case ID là một biến.
public override MarginalRequirements GetMarginalRequirements()	Xuất ra thông tin của Attributes
public override MiningParameterCollection GetParametersCollection()	Xuất ra tập hợp các Parameter
public override object ParseParameterValue(int parameterIndex,string parameterValue)	Chuyển kiểu dữ liệu Parameter
public override MiningColumnContent[] GetSupInputContentTypes()	Loại dữ liệu đầu vào
public override MiningColumnContent[] GetSupPredictContentTypes()	Loại dữ liệu đầu ra
public override SupportedFunction[] GetSupportedStandardFunctions()	Các hàm hỗ trợ (bao gồm DMX) của Microsoft
public override void ValidateAttributeSet(AttributeSet attributeSet)	Kiểm tra tính hợp lệ của loại dữ liệu đầu vào trước khi đưa vào Training Case

Bảng 5.10 Lớp khung Algorithmnavigator.cs

Algorithmnavigator.cs	
public algorithmnavigator (algorithm currentAlgorithm, bool dmDimension)	Hàm khởi tạo của Navigator
protected override int GetCurrentNodeId()	Lấy Node ID hiện tại
protected override NodeType GetNodeType()	Loại của Node (itemset, ass rule)
protected override string GetNodeUniqueName()	UniqueName của Node (thường thì chính là Name)
protected override uint[] GetNodeAttributes()	Các Attributes của Node
protected override AttributeStatistics[] GetNodeDistribution()	Thông tin Distribution của Node

Bảng 5.11 Lớp khung Algorithm.cs

Algorithm.cs	
protected override void Initialize()	Khởi tạo thuật toán
protected override void InsertCases(PushCaseSet CaseSet, MiningParameterCollection trainingParams)	Hàm xử lý của thuật toán
protected override object GetTrainingParameterActualValue(int paramOrdinal)	Lấy dữ liệu Parameter
protected override void SaveContent(PersistenceWriter writer)	Lưu kết quả đã xử lý
protected override void LoadContent(PersistenceReader reader)	Lấy kết quả đã xử lý
protected override AlgorithmNavigationBase GetNavigator(bool forDMDimensionContent)	Hiển thị kết quả

Bảng 5.12 Lớp tự tạo Node.cs

Node.cs: cấu trúc dữ liệu để hiện thị	
public string Description	Mô tả dữ liệu
public string Caption	Mô tả dữ liệu
public double Support	Số lần xuất hiện
public double Probability	Xác suất của dữ liệu
public List<AttributeStatistics> Distribution	Gồm danh sách ứng viên của dữ liệu
public void Load(ref PersistenceReader reader)	Xuất kết quả
public void Save(ref PersistenceWriter writer)	Lưu kết quả

Bảng 5.13 Lớp tự tạo Itemset.cs

Itemset.cs: cấu trúc dữ liệu tạm thời để xử lý thuật toán	
public double Support	Độ phổ biến của mục dữ liệu
public int Count	Số các ứng viên trong mục dữ liệu
public List<AttributeStatistics> getAttributeStatistics()	Chuyển kiểu sang AttributeStatistics
public bool Containt(Itemset item)	Xem bộ dữ liệu này có chứa bộ dữ liệu kia không
public bool Equals(Itemset item)	Kiểm tra xem hai bộ dữ liệu có bằng nhau không

Bảng 5.14 Lớp tự tạo Rule.cs

Rule.cs: cấu trúc Rule để xử lý thuật toán	
public double Confident	Độ tin cậy, xác suất

Bảng 5.15 Lớp tự tạo SDB.cs

SDB.cs: cấu trúc lưu trữ cơ sở dữ liệu chiếu	
public Dictionary<int, int> dicSdb;	Danh sách CSDL chiếu với Key là vị trí dòng trong danh sách Transaction và Value là vị trí.
public int Count	Tổng số dòng trong CSDL chiếu

Bảng 5.16 Lớp tự tạo Factory.cs

Factory.cs: lớp trừu tượng Factory	
abstract public StoreAlgorithm CreateAlgorithm(String nameAlgo,algorithm algo)	Khởi tạo thuật toán

Bảng 5.17 Lớp tự tạo AlgorithmFactory.cs

AlgorithmFactory.cs: kế thừa từ lớp trừu tượng Factory để tạo ra các đối tượng thuật toán riêng biệt	
override public StoreAlgorithm CreateAlgorithm(String nameAlgo,algorithm algo)	Khởi tạo thuật toán

Bảng 5.18 Lớp tự tạo StoreAlgorithm.cs

StoreAlgorithm.cs: lớp trừu tượng của kho thuật toán	
abstract public void addCase(MiningCase mcase)	Thêm Case vào dữ liệu Transaction
abstract public void insertCase(int minSupport, int minConf, ref List<Node> lstItem)	Xử lý dữ liệu Transaction bằng thuật toán để tạo ra bộ dữ liệu và luật
abstract public void runAlgorithm(double minSupport)	Chạy thuật toán
public List<Itemset> calculateSupportDic(double minSupport, List<Itemset> generalList, List<Itemset> currentList)	Kiểm tra điều kiện phổ biến của dữ liệu
public void addAllFequentItems(List<Itemset> dicFrequentItems)	Thêm bộ dữ liệu vào danh sách các bộ dữ liệu
public void GenerateRules(double minConf, ref List<Rule> lstRulesReturn)	Tạo luật từ danh sách luật và độ tin cậy
public void GenerateCombination(double minConf, Itemset items, int position, Rule generalRule, ref List<Rule> lstRulesReturn)	Kết hợp luật để tạo luật mới
public void findRule(double minConf)	Chạy thuật toán tìm luật
public bool checkHasItems(List<Itemset> list, Itemset item)	Kiểm tra bộ dữ liệu có trong danh sách chưa
public double GetSupport(Itemset item, List<Itemset> generalList)	Đếm độ phổ biến của bộ dữ liệu trong danh sách

Bảng 5.19 Lớp tự tạo AprioriAlgo.cs

AprioriAlgo.cs: thuật toán Apriori kế thừa từ StoreAlgorithm	
private void addItem(ref Itemset items, uint key, StateValue value)	Thêm một mục dữ liệu vào bộ dữ liệu theo thứ tự của Key
private void getL1FrequentItems(List<Itemset> list, ref List<Itemset> lstFrequentItems)	Xuất ra bộ dữ liệu phổ biến có độ dài bằng 1
private List<Itemset> GenerateCandidates(List<Itemset> lstFrequentItems)	Tìm các ứng viên từ bộ dữ liệu phổ biến để kết hợp
private Itemset getCandidate(Itemset FirstItems, Itemset SecondItems)	So sánh hai bộ dữ liệu để tìm ứng viên
private Itemset getItemLastAttribute(Itemset FirstItems, Itemset SecondItems)	So sánh hai bộ dữ liệu để tìm ứng viên ở vị trí cuối cùng

Bảng 5.20 Lớp tự tạo BideAlgo.cs

BideAlgo.cs: thuật toán Bide kế thừa từ StoreAlgorithm	
private void getL1FrequentItems(List<Itemset> list, ref List<Itemset> lstFrequentItems)	Xuất ra bộ dữ liệu phổ biến có độ dài bằng 1
private bool BackScan(Itemset items, SDB sdb)	Kiểm tra BackScan
private List<Itemset> getLFI(SDB sdb, Itemset items, double minSupport)	Tìm những mục dữ liệu trong CSDL thỏa độ phổ biến
private List<Itemset> getFEI(List<Itemset> lstLFI, Itemset itemset)	Tìm những mục dữ liệu trong LFI có độ phổ biến ≥ độ phổ biến của tiền tố đang xét
private Itemset getItemsetFI(Itemset items1, Itemset items2)	Được tính từ đầu chuỗi đến lần xuất hiện đầu tiên của tiền tố
private int getLF(Itemset items1, Itemset items2)	Vị trí cuối cùng trong FI
private Itemset getFIi(Itemset itemsFI, Itemset items)	Lấy phần chuỗi từ đầu đến vị trí i của tiền tố
private Itemset getSemi(Itemset itemsFI, Itemset itemsChk, int chk)	Lấy phần chuỗi giữa
private bool checkSemi(List<Itemset> lstSemi)	Kiểm tra Semi
private bool getPositionFirst(out int position, Itemset items1, Itemset items2)	Tìm vị trí đầu tiên của tiền tố trong chuỗi
private SDB getPseudoProjected(SDB GeneralSDB, ref Itemset items)	Xuất cơ sở dữ liệu chiếu từ cơ sở dữ liệu tổng quát
private SDB getPseudoProjected(Itemset items)	Xuất cơ sở dữ liệu chiếu từ Transaction
private double getSupportInFreq(Itemset items)	Đếm độ phổ biến trong bộ dữ liệu phổ biến

5.3.2. Chức năng các lớp và hàm của ứng dụng sử dụng thuật toán tích hợp

Bảng 5.21 Hai lớp sử dụng AMO và ADOMD.NET

AnalysisService.cs	Tạo Data Source, Data Source View, Mining Structure, Mining Model trong Analysis Services. Lớp sẽ sử dụng thư viện Microsoft.AnalysisServices.
ASHelp.cs	Hỗ trợ thực thi câu truy vấn DMX. Lớp sẽ sử dụng thư viện Microsoft.AnalysisServices.AdomdClient để thực thi DMX. Dường như gần giống với thư viện System.Data.SqlClient nhưng AdomdClient chỉ dùng để SELECT dữ liệu.

Bảng 5.22 Các hàm của lớp AnalysisService.cs

Server svr	Dùng để kết nối Analysis Services
public void CreateDatabase()	Tạo Database
public bool InititalizeDatabase()	Khởi tạo biến Database
public void CreateDataAccessObjects()	Tạo Data Source và Data Source View
public void CreateMiningStructure()	Tạo Mining Structure
public void CreateModels()	Tạo Mining Model
public void ProcessDatabase()	Chạy thuật toán
public void modifyProperties(int minSupport,int minConf,string nameAlgo)	Sửa lại thông số độ phổ biến và độ tin cậy tên thuật toán sẽ chạy trong Mining Model.
public void getProperties(out int minSupport, out int minConf,out string nameAlgo)	Xuất thông số thuật toán trong Mining Model

TÀI LIỆU THAM KHẢO

[1] B.Crivat, *A Tutorial for Contructing a Managed Plug-In Algorithm*, Microsoft ® SQL Server™ 2005 Analysis Services (2006)

[2] J.Han, H.Cheng, D.Xin, X.Yan, *Frequent pattern mining: current status and future directions*, Springer Science & Business Media, LLC (2007)

[3] J.Han, M.Kamber, "Data Mining Concept and Techniques" 2nd edition, Dartmouth Publishing, Inc, 2006

[4] J.Han, J.Pei, X.Yan, *Sequential Pattern Mining by Pattern-Growth Principles and Extensions*, University of Illinois at Urbana-Champaign, U.S.A (2005)

[5] S.Laxman, P.S.Sastry, *A survey of temporal data mining, Department of Electrical Engineering*, Indian Institute of Science, Bangalore, India (2006)

[6] J.MacLennan, Z.Tang, B.Crivat, "Data Mining with Microsoft SQL Server 2008", Wiley Publishing, Inc, Indiana, 2009

[7] F.Masseglia, M.Teisseire, P.Poncelet, *Sequential Pattern Mining: A Survey on Issues and Approache*, INRIA Sophia Antipolis, LIRMM University of Montpellier II, Pascal Poncelet, EMA/LGI2P, France (2005)

[8] O.Rud, "Data Mining Cookbook: Modeling Data for Marketing, Risk, and Customer Relationship Management", Wiley Computer Publishing, 2001

[9] A.Tiwari, R.K.Gupta, D.P.Agrawal, *A survey of frequent pattern mining current status & challenge issues*, Information Technology Journal (2010)

[10] J.Wang, J.Han, *BIDE: Efficient Mining of Frequent Closed Sequences*, ICDE (2004)

[11] X.Yan, J.Han, R.Afshar, *Clospan: Mininig Closed sequential Patterns in Large Datasets*, SDM (2003)

[12] Q.Zhao, S.S.Bhowmick, *Sequential Pattern Mining: A Survey*, Nanyang Technological University, Singapore (2003)