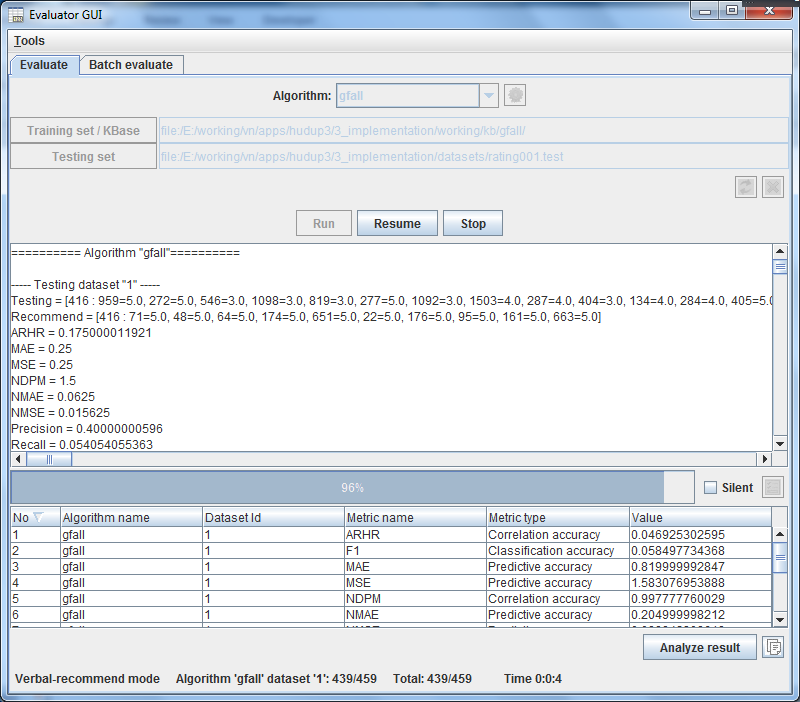
*Bản mô tả sản phẩm*

**Nền tảng hỗ trợ phát triển giải pháp khuyến nghị trong thương mại điện tử**

**Nhà phát triển sản phẩm: Công ty TNHH MTV Lập trình Hướng Dương**

**Phiên bản: 11.0**



**Bản mô tả**

Sáng chế là “*nền tảng hỗ trợ phát triển giải pháp khuyến nghị trong thương mại điện tử*”. Đây là một phần mềm máy tính, một kiến trúc khung, một “hệ điều hành” trong lĩnh vực thương mại điện tử nhằm giúp các nhà khoa học, các nhà phát triển phần mềm xây dựng các giải pháp khuyến nghị dựa trên nền tảng này. Thuật ngữ khuyến nghị (hay tư vấn) đề cập đến các giải pháp (hay thuật toán) công nghệ thông tin có mục đích giới thiệu các sản phẩm mà người mua có thể yêu thích trong một trang web thương mại điện tử nhằm tăng doanh số bán hàng.

Bạn cần phát triển một giải pháp giới thiệu các mặt hàng mà người dùng có thể yêu thích trong một trang web thương mại điện tử. Bạn là một nhà khoa học, sau nhiều năm nghiên cứu, bừng dậy một thuật toán khuyến nghị mới. Ý tưởng của bạn hay, giải pháp của bạn rất hữu ích, bạn hết sức hào hứng nhưng:

1. Bạn đối mặt với những tính toán phức tạp trong xử lý dữ liệu, những mô hình đa dạng trong lĩnh vực khuyến nghị. Thay vì bắt tay ngay vào việc hiện thực hóa ý tưởng, bạn tốn rất nhiều thời gian và công sức ít nhất từ 1 – 2 năm để giải quyết những phức tạp này.
2. Bạn không thể đánh giá được chất lượng giải pháp của bạn theo những tiêu chuẩn phổ biến.
3. Bạn không có môi trường mô phỏng ứng dụng thương mại điện tử để kiểm định tính khả thi của giải pháp bạn đưa ra.

Sáng chế “*nền tảng hỗ trợ phát triển giải pháp khuyến nghị trong thương mại điện tử*” giúp bạn giải quyết 3 vấn đề trên một cách hoàn mỹ và bạn sẽ:

1. Hiện thức hóa được giải pháp rất nhanh và dễ
2. Kiểm định được giải pháp theo các tiêu chuẩn chất lượng
3. Xác định được tính khả thi của giải pháp trong ứng dụng thực tế

Nền tảng còn có một tính năng ưu việt khác, đó là cung cấp hai thuật toán khuyến nghị tối ưu giúp bạn đối sánh các giải pháp với nhau. Sáng chế nhằm phục vụ bạn – là nhà khoa học hay nhà phát triển phần mềm nhằm giải quyết ba vấn đề cốt lõi trên. Sáng chế xây dựng ba nấc giải pháp:

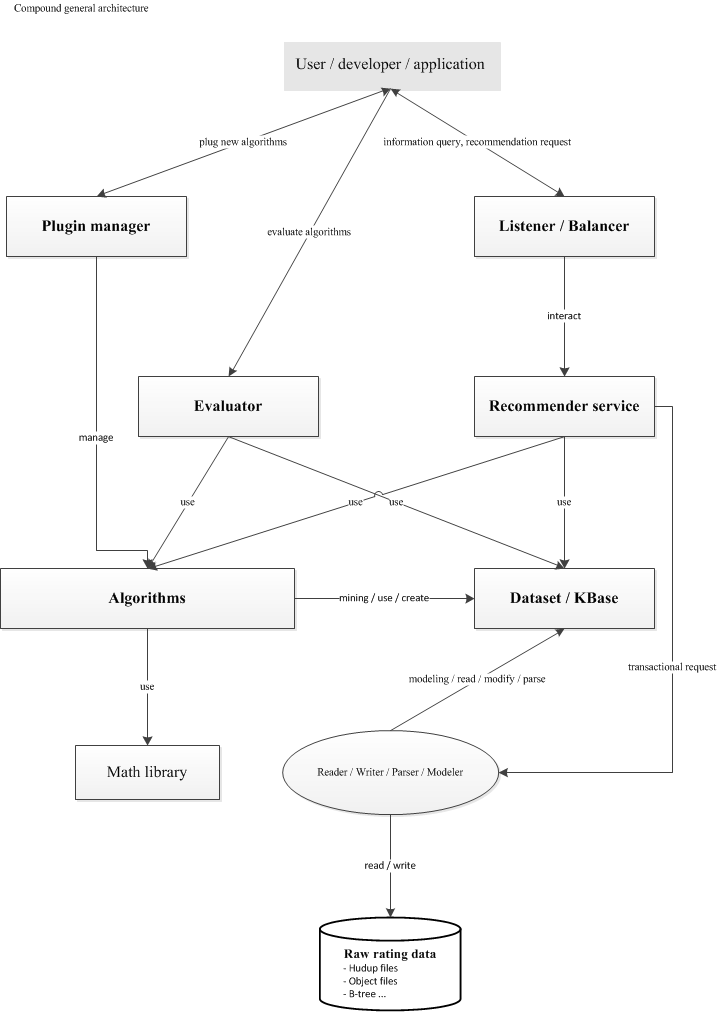
1. Nấc cơ sở: Xây dựng mô hình giải thuật và mô hình dữ liệu giúp bạn sử dụng để tạo ra một sản phẩm phần mềm mới với chi phí thấp nhất.
2. Nấc tiêu chuẩn đo lường: Xây dựng các tiêu chuẩn đánh giá và bộ đánh giá giải thuật giúp bạn xác định được chất lượng giải pháp khuyến nghị.
3. Nấc môi trường mô phỏng: Xây dựng trình chủ khuyến nghị hay môi trường mô phỏng khuyến nghị giúp bạn kiểm định tính khả thi của giải pháp khuyến nghị.

Hiện tại trên thế giới có một số sản phẩm mã nguồn mở hoặc có phiên bản mở có chức năng tương đương với sáng chế bao gồm:

1. *Carleton* (http://www.cs.carleton.edu/cs\_comps/0607/recommend/recommender): được phát triển bởi đại học Carleton, Minnesota, Hoa Kỳ. Phần mềm đơn giản chỉ cài đặt một số thuật toán khuyến nghị và đánh giá thuật toán trên độ đo RMSE. Phần mềm mang tính ví dụ chưa trở thành một nền tảng hay một hệ thống khuyến nghị. Tuy nhiên điểm độc đáo của Carleton là áp dụng trong học tập, giới thiệu những khóa học cho sinh viên dựa trên học bạ của họ. Cấu trúc lớp lập trình trong Carleton tương đối rõ ràng.
2. *Cofi* (http://www.nongnu.org/cofi/): hiện đã ngưng hỗ trợ. Điểm độc đáo của phần mềm này viết trên nền Java nên chạy đa nền đa hệ. Phần mềm đơn giản chỉ cài đặt và đánh giá một số thuật toán, chưa trở thành một nền tảng hay một hệ thống khuyến nghị.
3. *Colfi* (http://colfi.wz.cz/): được phát triển bởi giáo sư Lukáš Brožovský, đại học Charles, Prague. Phần mềm xây dựng một trình chủ khuyến nghị cho dịch vụ hẹn hò. Quy mô của phần mềm này lớn hơn Carleton và Cofi. Phần mềm cài đặt và đánh giá một số thuật toán lọc cộng tác và xây dựng một trình chủ khuyến nghị. Tuy nhiên chưa có tùy biến cho giải thuật và các độ đo, trình chủ còn khá sơ sài. Phần mềm mang nhiều tính nghiên cứu, chưa thể trở thành một nền tảng hay hệ thống khuyến nghị vững chắc. Tuy nhiên điểm độc đáo của phần mềm thể hiện được dịch vụ hẹn hò qua tương tác trang web – máy chủ.
4. *Crab* (http://muricoca.github.io/crab/): trình chủ khuyến nghị được viết bằng ngôn ngữ Python, phát triển bởi phòng thí nghiệp Muricoca http://www.muricoca.com/. Ưu điểm của Crab là xây dựng một trình chủ khuyến nghị cùng guồng máy khuyến nghị có cơ chế đánh giá giải thuật. Nhược điểm của Grab so với sáng chế này là không hỗ trợ nhà phát triển xây dựng một giải thuật mới qua 3 giai đoạn: cài đặt, đánh giá và mô phỏng. Hơn nữa, kiến trúc chưa linh động và chưa chuẩn, giải thuật bên trong trình chủ khuyến nghị chưa đặc sắc chỉ xoay quanh giải thuật SVD và láng giềng gần nhất.
5. *Duine* (http://www.duineframework.org/): được phát triển bởi viện Telematica, Novay đây thực sự là một nền tảng khuyến nghị vững chắc có thể cạnh tranh với sáng chế. Kiến trúc rất linh động và rất tốt. Mặc dù quy mô hơi kém so với sáng chế nhưng Duine có ưu điểm là tập trung vào hiệu năng của guồng máy khuyến nghị. Nhược điểm của Duine so với sáng chế là không hỗ trợ nhà phát triển xây dựng một giải thuật mới qua 3 giai đoạn: cài đặt, đánh giá và mô phỏng. Bộ đánh giá thuật toán của Duine chưa được chuẩn hóa, khả năng tùy biến chưa cao và khá sơ sài.
6. *easyrec* (http://easyrec.org/recommendation-engine) được phát triển bởi tổ chức IntelliJ IDEA (http://www.jetbrains.com/idea/) và Research Studios Austria, Forschungsgesellschaft mbH (http://www.researchstudio.at/home\_en.html). Điểm độc đáo của easyrec là tính tiện dụng, hỗ trợ tư vấn qua mạng, cho phép bạn nhúng guồng máy khuyến nghị vào trang web qua lời gọi hàm đến easyrec. Nhược điểm của easyrec so với sáng chế là không hỗ trợ nhà phát triển xây dựng thuật toán mới. Hơn nữa phân khúc khách hàng của easyrec hoàn toàn khác với chúng tôi.
7. *GraphLab* (http://graphlab.org) là bộ công cụ đa dụng hỗ trợ lọc cộng tác (một lĩnh vực trong khuyến nghị), gom cụm, thị giác máy tính, phân tích đồ thị… được tài trợ bởi Office of Naval Research, Army Research Office, DARPA và Intel. Bộ công cụ này rất lớn và đa dụng, đây là ưu điểm và cũng là nhược điểm. Một số nhà phát triển khi quen dùng bộ công cụ này trong các lĩnh vực khác sẽ có xu hướng tiếp tục sử dụng khi nghiên cứu sang khuyến nghị. Tuy nhiên, riêng đối với lĩnh vực khuyến nghị, bộ công cụ này hỗ trợ rất hạn chế, chỉ cài đặt một số thuật toán lọc cộng tác chưa thể trở thành một nền tảng hay trình chủ khuyến nghị.
8. *LensKit* (http://lenskit.grouplens.org): viết bằng Java chạy đa nền đa hệ, được phát triển bởi nhóm nghiên cứu GroupLens, đại học Minnesota (http://www.grouplens.org/). Điểm mạnh của LensKit là hỗ trợ việc xây dựng và đánh giá giải thuật khuyến nghị rất tốt, cơ chế đánh giá (evaluation) rất tinh xảo. Điểm yếu của LensKit là chưa hỗ trợ trình mô phỏng hay trình chủ khuyến nghị để các nhà phát triển thử nghiệm trong môi trường tương tác khách chủ. Cấu trúc thư viện lớp còn phân mảnh nhưng có điểm hay là tận dụng môi trường phát triển Maven. Nhìn chung, tuy thiếu trình chủ khuyến nghị nhưng LensKit là một nền tảng rất tốt.
9. *Mahout* (http://mahout.apache.org/) được phát triển bởi tổ chức Apache (http://www.apache.org), là bộ công cụ rất mạnh hỗ trợ khai thác dữ liệu và học máy trong đó có cài đặt một số thuật toán khuyến nghị như láng giềng gần nhất. Thử nghiệm giải thuật trên Mahout rất đơn giản, Mahout hướng đến người dùng cuối nhiều hơn nhà phát triển phần mềm. Điểm mạnh và điểm yếu của Mahout rất gần với GraphLab, mục tiêu chủ yếu của Mahout là bộ công cụ đa dụng nên không hỗ trợ hoàn toàn khuyến nghị. Nếu bạn phát triển một phần mềm khai khoáng dữ liệu hay học máy thì nên dùng Mahout, còn bạn muốn tập trung vào khuyến nghị thì nên dùng sáng chế của chúng tôi.
10. *MyMedia* (http:// mymediaproject.codeplex.com): là phần mềm giới thiệu các sản phẩm đa phương tiện như phim, ảnh. Điểm đặc sắc của MyMedia là cài đặt các thuật toán khai thác mạng xã hội, khuyến nghị, cá nhân hóa tập trung vào thông tin giải trí đa phương tiện. MyMedia là một nền tảng khuyến nghị thông tin đa phương tiện rất mạnh hướng đến người dùng cuối là các tổ chức, công ty hoạt động trong lĩnh vực giải trí, đa phương tiện. Điểm yếu của MyMedia là không cài đặt trình chủ khuyến nghị và cơ chế đánh giá giải thuật chưa hiệu quả. Nhìn chung MyMedia hướng đến người dùng cuối. MyMedia viết bằng ngôn ngữ lập trình hiện đại C#, được phát triển bởi tổ chức EU Framework 7 Programme Networked Media initiative cùng với các nhà cộng tác: EMIC, BT, the BBC, Technical University of Eindhoven, University of Hildesheim, Microgenesis and Novay.
11. *MyMediaLite* (http://mymedialite.net) là bộ thư viện nhỏ, gọn cài đặt và đánh giá một số thuật toán khuyến nghị. Điểm mạnh của MyMediaLite là nhỏ gọn nhưng cài đặt khá nhiều các giải thuật khuyến nghị và các độ đo, kiến trúc rất trong sáng. Điểm yếu: chưa cài đặt trình chủ khuyến nghị, các độ đo chưa có khả năng tùy biến, bộ đánh giá giải thuật khá sơ sài.
12. *Recommenderlab* (A Framework for Developing and Testing Recommendation Algorithms – Michael Hahsler 2011) được phát triển bởi Michael Hahsler tài trợ bởi NSF Industry/University Cooperative Research Center for Net-Centric Software & Systems. Recommenderlab là gói mở rộng ngôn ngữ thống kê R với mục tiêu là xây dựng một cơ sở hạ tầng khuyến nghị trên nền R. Điểm đặc sắc của Recommenderlab là tận dụng khả năng xử lý dữ liệu xuất sắc của R. Khả năng đánh giá, so sánh các thuật toán trong Recommenderlab cũng rất tốt. Điểm yếu là chưa phát triển trình chủ khuyến nghị vì Recommenderlab không thể chạy độc lập với R. Recommenderlab phù hợp với nghiên cứu và đánh giá giải thuật trong thời gian ngắn; việc phát triển, đánh giá và thử nghiệm một giải thuật cần một nền tảng mạnh mẽ hơn như MyMedia, Mahout, LensKit, Duine và sáng chế chúng tôi là lựa chọn tốt nhất.
13. *SVDFeature* (http://svdfeature.apexlab.org) viết bằng C++ được phát triển bởi các nhà phát triển Tianqi Chen , Weinan Zhang , Qiuxia Lu , Kailong Chen , Zhao Zheng , Yong Yu. SVD là một giải thuật lọc cộng tác xử lý ma trận rất hiệu quả trong lĩnh vực khuyến nghị. SVDFeature tập trung vào cài đặt giải thuật SVD một cách tốt nhất. SVDFeature không phải là một nền tảng khuyến nghị nhưng có điểm đặc sắc là xử lý dữ liệu ma trận lớn và tăng tóc giải thuật SVD.
14. *Vogoo* (http://sourceforge.net/projects/vogoo/). Sự độc đáo của Vogoo là hướng đến người dùng cuối, cài đặt giải thuật khuyến nghị ngay trên trang web viết bằng ngôn ngữ web PHP, điều này mang đến sự tiện dụng tối đa khi xây dựng trang web thương mại điện tử bằng PHP. Điểm yếu là Vogoo quá sơ sài, không phải là một nền tảng khuyến nghị mặc dù qua khảo sát tôi nhận thấy Vogoo độc đáo nhất, thư viện cực kỳ nhỏ gọn, nếu yêu cầu phát triển nhanh đặt lên hàng đầu, Vogoo là số một.

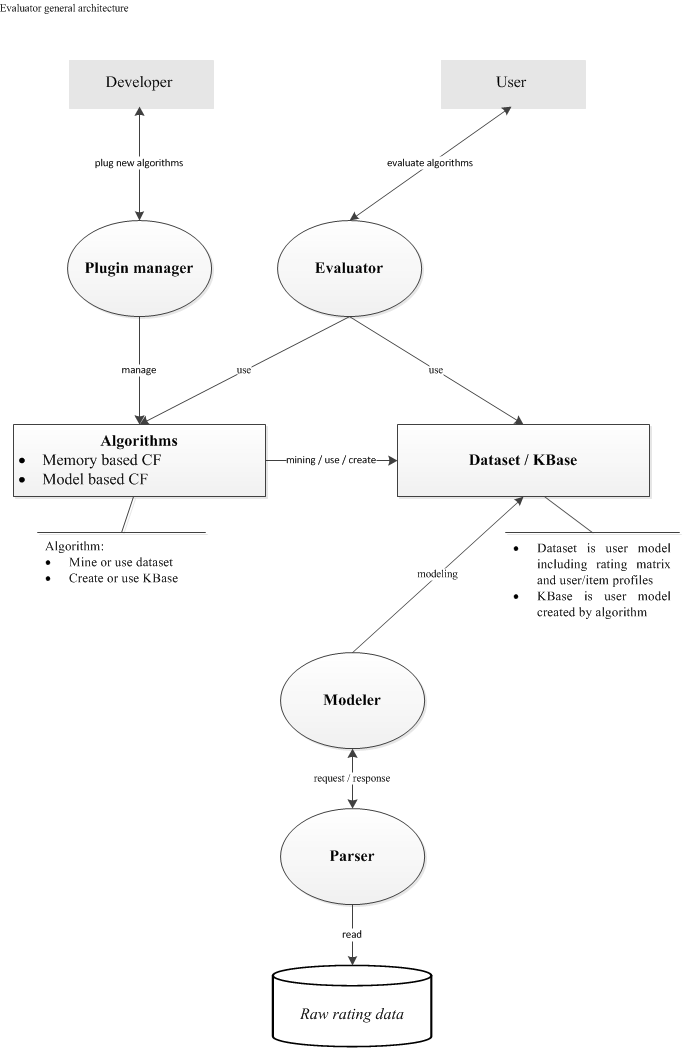
Qua khảo sát 14 phần mềm khuyến nghị tiêu biểu, sáng chế này độc đáo nhất và tối ưu nếu xét theo khía cạnh hỗ trợ xuyên suốt nhà phát triển xây dựng một giải pháp khuyến nghị mới từ khâu cài đặt, khâu kiểm định chất lượng đến khâu thử nghiệm. Hơn nữa kiến trúc sản phẩm tạo thành từ sáng chế rất linh động và tùy biến cao. Các tiêu chuẩn kiểm định (độ đo) và giải thuật đều được chuẩn hóa theo một khuôn mẫu nhưng người dùng vẫn có thể thêm những độ đo và giải thuật mới.

Sáng chế là một phần mềm máy tính được cấu thành từ 3 bộ: bộ quản lý giải thuật và độ đo (plugin manager), bộ đánh giá (evaluator) và dịch vụ khuyến nghị (recommender service) tương ứng với 3 nấc giải pháp mà sáng chế đề xuất. Sau đây là kiến trúc tổng quan của sáng chế:



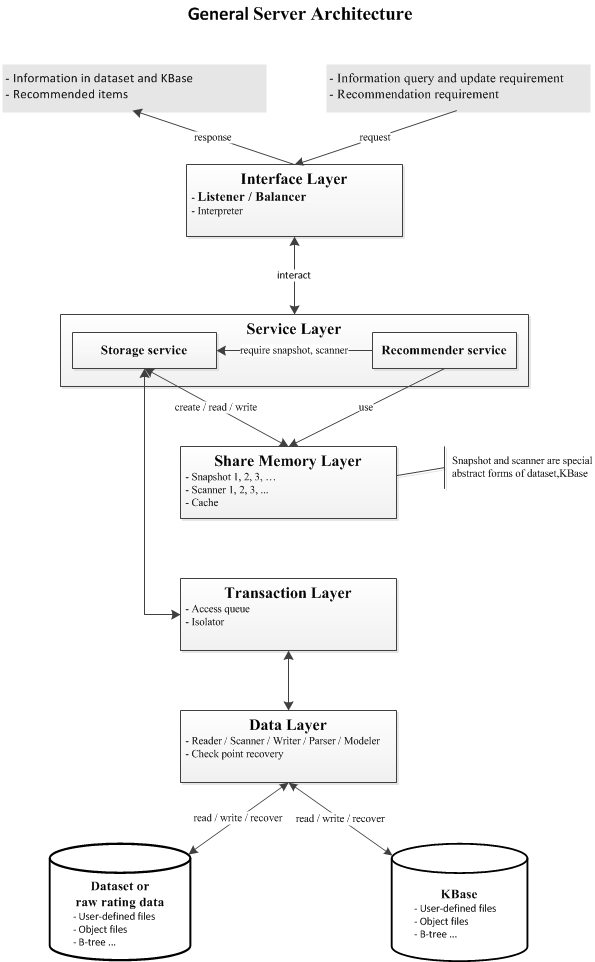
**Hình 1**. Kiến trúc tổng quan

Sau đây là kiến trúc bộ đánh giá (evaluator)



**Hình 2**. Kiến trúc bộ đánh giá (evaluator)

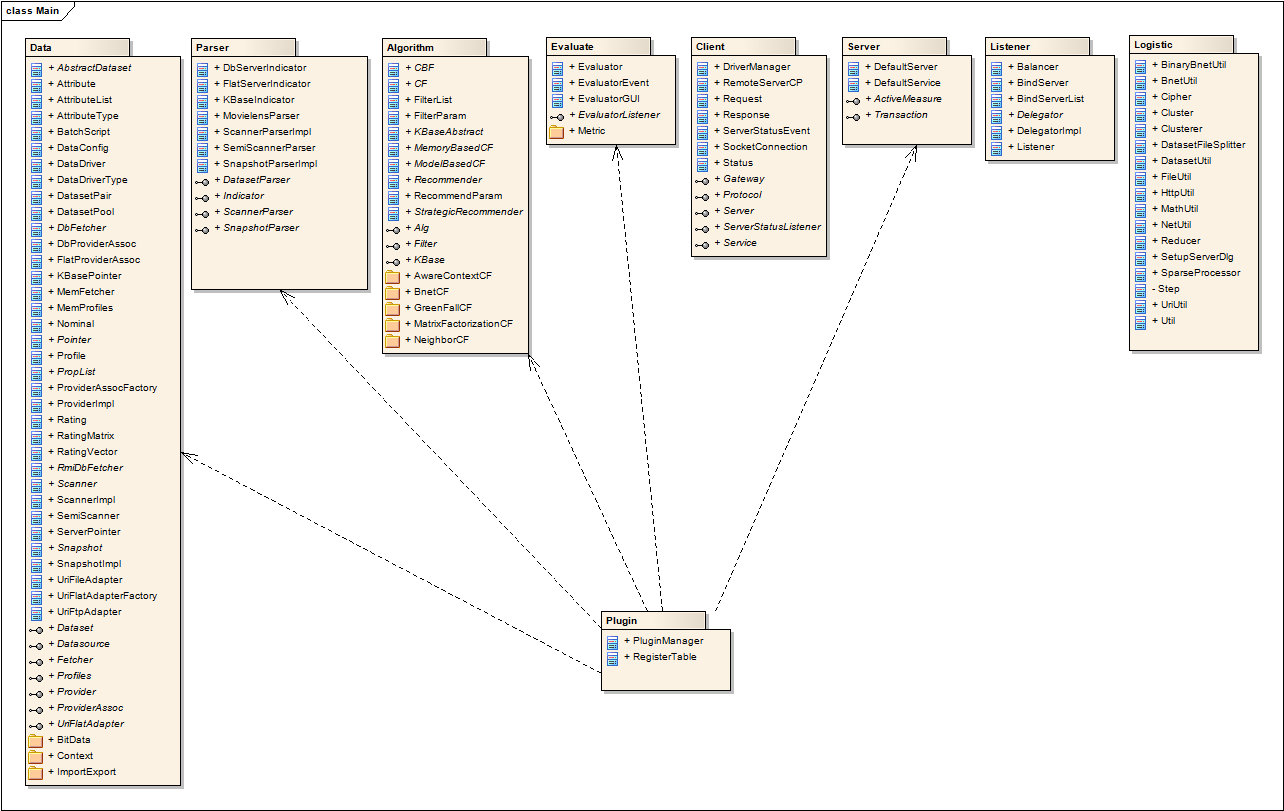
Sau đây là kiến trúc dịch vụ khuyến nghị



**Hình 3**. Kiến trúc dịch vụ khuyến nghị (recommender service or receommender server)

Kiến trúc tổng quan 3 bộ được phân rã thành 9 gói (package):

* *Data* (*dữ liệu*): chuẩn hóa và mô hình hóa dữ liệu ở mức trừu tượng
* *Parser* (*phân tích*): xử lý dữ liệu
* *Algorithm* (*giải thuật*): mô hình giải thuật khuyến nghị
* *Evaluate* (*đánh giá*): đánh giá giải thuật theo các độ đo đã được cài đặt
* *Client* (*khách*), *Server* (chủ) và *Listener* (*bộ nghe*): cung cấp dịch vụ khuyến nghị trong môi trường mạng máy tính khách – chủ
* *Logistic* (*tiện ích*): các tiện ích tính toán và toán học
* *Plugin* (*phích cắm*): quản lý các giải thuật và độ đo



**Hình 4**. Chín gói của sáng chế

Mỗi gói gồm nhiều lớp (class) phần mềm máy tính tạo thành các sơ đồ lớp (kiến trúc lớp) nội tại. Kiến trúc mỗi gói được mô tả trong phụ lục A. Riêng gói *Algorithm* còn cung cấp 2 thuật toán khuyến nghị: lọc cộng tác dựa trên mạng Bayesian (*BnetCF*) và lọc cộng tác dựa trên khai thác tập phổ biến cực đại (*GreenFallCF*). Chi tiết các thuật toán được mô tả trong 2 bài báo phụ lục B.

Sáng chế giúp xây dựng các giải thuật khuyến nghị một cách nhanh chóng và dễ dàng cùng kiểm định tính chính xác của giải thuật qua các độ đo và kiểm định tính khả thi của giải pháp trong môi trường mô phỏng. Giả sử bạn cần cài đặt thuật toán lọc cộng tác Green Fall dựa trên khai thác tập phổ biến cực đại, thay vì viết cả phần mềm lớn tốn rất nhiều thời gian bao gồm xử lý dữ liệu, thi công chi tiết thuật toán, cài đặt chi tiết các độ đo, kiểm định thuật toán, xây dựng môi trường mô phỏng thuật toán trong ứng dụng thời gian thực, bạn chỉ cần sử dụng phần mềm từ sáng chế và thực hiện 3 bước sau:

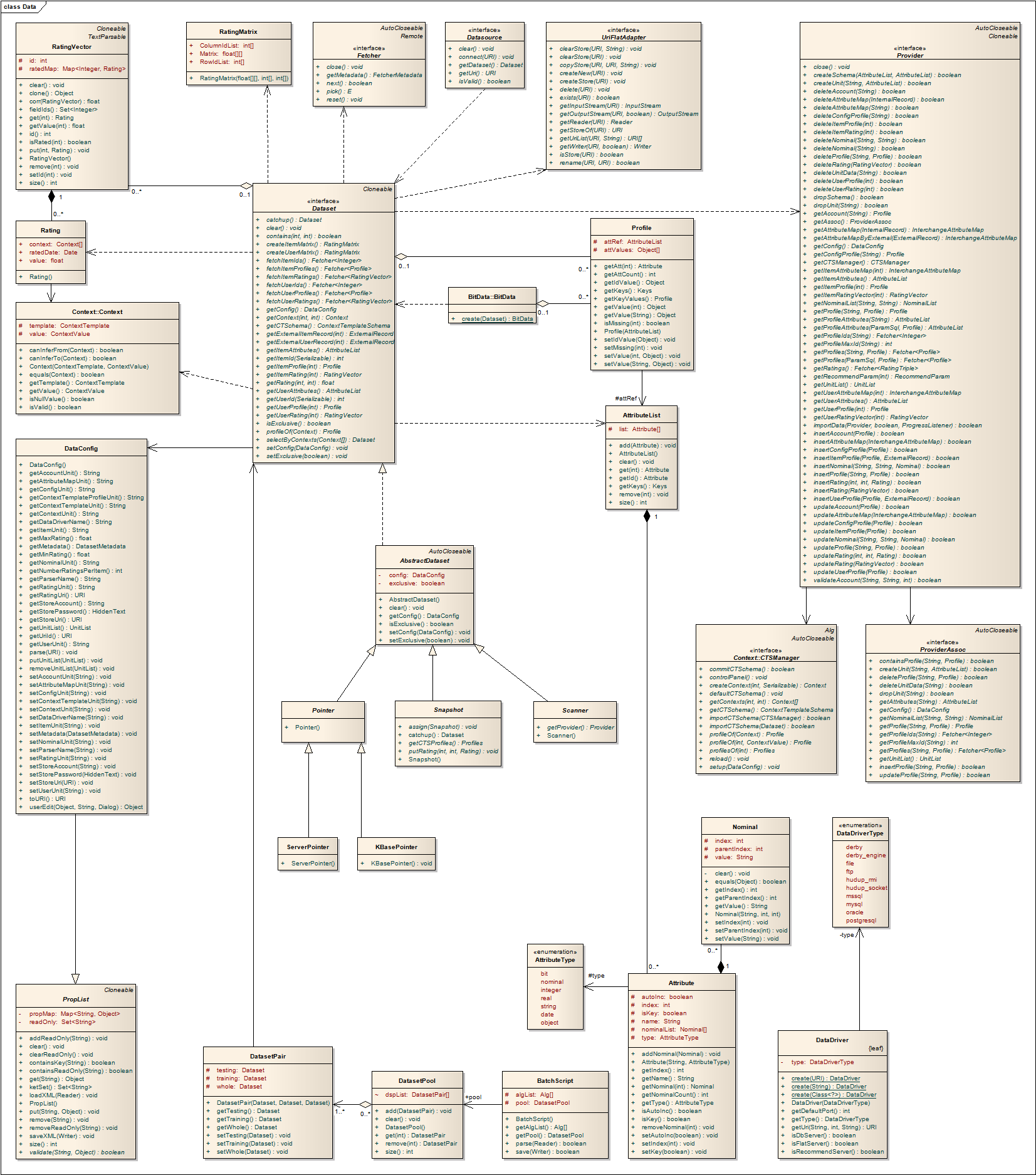
1. Kế thừa lớp (class) *MemoryBasedCF* hay *ModelBasedCF* trong gói *Algorithm* và cài đặt ý tưởng của bạn trong 2 phương thức (method) *estimate* và *recommend* trong 2 lớp trên.
2. Khởi động bộ đánh giá *Evaluator* (xem hình 1 và 2) để kiểm định và so sánh GreenFall với các thuật toán qua các độ đo đã được cài đặt sẵn. Bạn hoàn toàn có thể thêm vào độ đo mới
3. Cấu hình trình chủ khuyến nghị *Recommender service* (xem hình 1 và 3) để nhúng Green Fall vào trong trình chủ và khởi động trình chủ để kiểm định tính khả thi của Green Fall trong ứng dụng thời gian thực

Các thao tác thực hiện trong 3 bước rất đơn giản, chủ yếu là cấu hình qua giao diện phần mềm, ngoài bước 1 – bạn cần viết những dòng mã nguồn (code line) để thi công ý tưởng. Mô hình thuật toán đã được thiết kế và cài đặt chặt chẽ, tất cả những dòng mã nguồn bạn viết gói gọn trong 2 phương thức *estimate* và *recommend*. Ước lượng thời gian và công sức để cài đặt và kiểm định một thuật toán khuyến nghị khoảng 2 năm, nếu sử dụng sáng chế sẽ rút ngắn khoảng 1 tuần khi ý tưởng của bạn rõ ràng. Điều này có nghĩa chi phí phát triển thuật toán giảm xuống rất nhiều còn khoảng 1% so với tổng chi phí ban đầu.

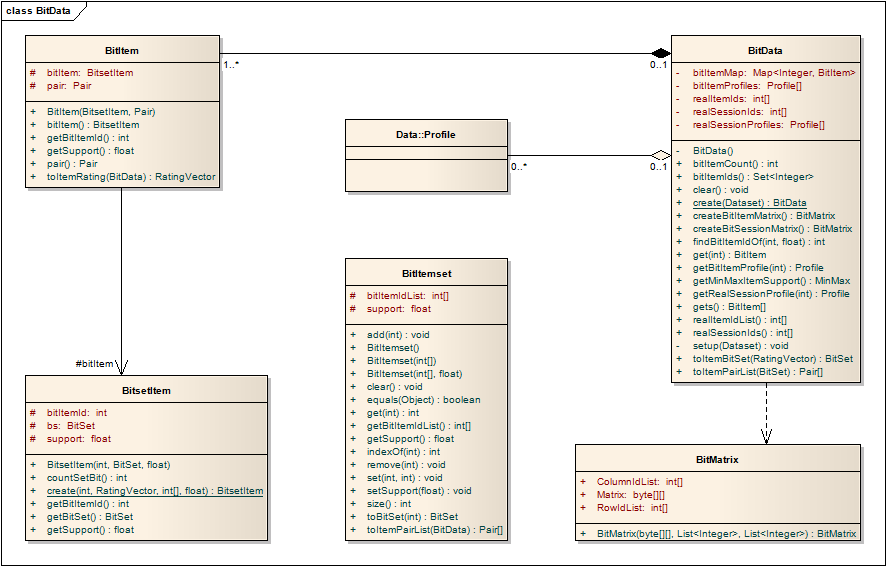
Bản mô tả sản phẩm gồm 2 phụ lục:

* Phụ lục A: Từ hình 5 đến hình 19 mô tả tất cả các lớp (class) phần mềm thi công kiến trúc này.
* Phụ lục B: Chi tiết 2 thuật toán: lọc cộng tác dựa trên khai thác tập phổ biến cực đại (*Green Fall*) và lọc cộng tác dựa trên suy diễn mạng Bayesian (*Bayesian*).

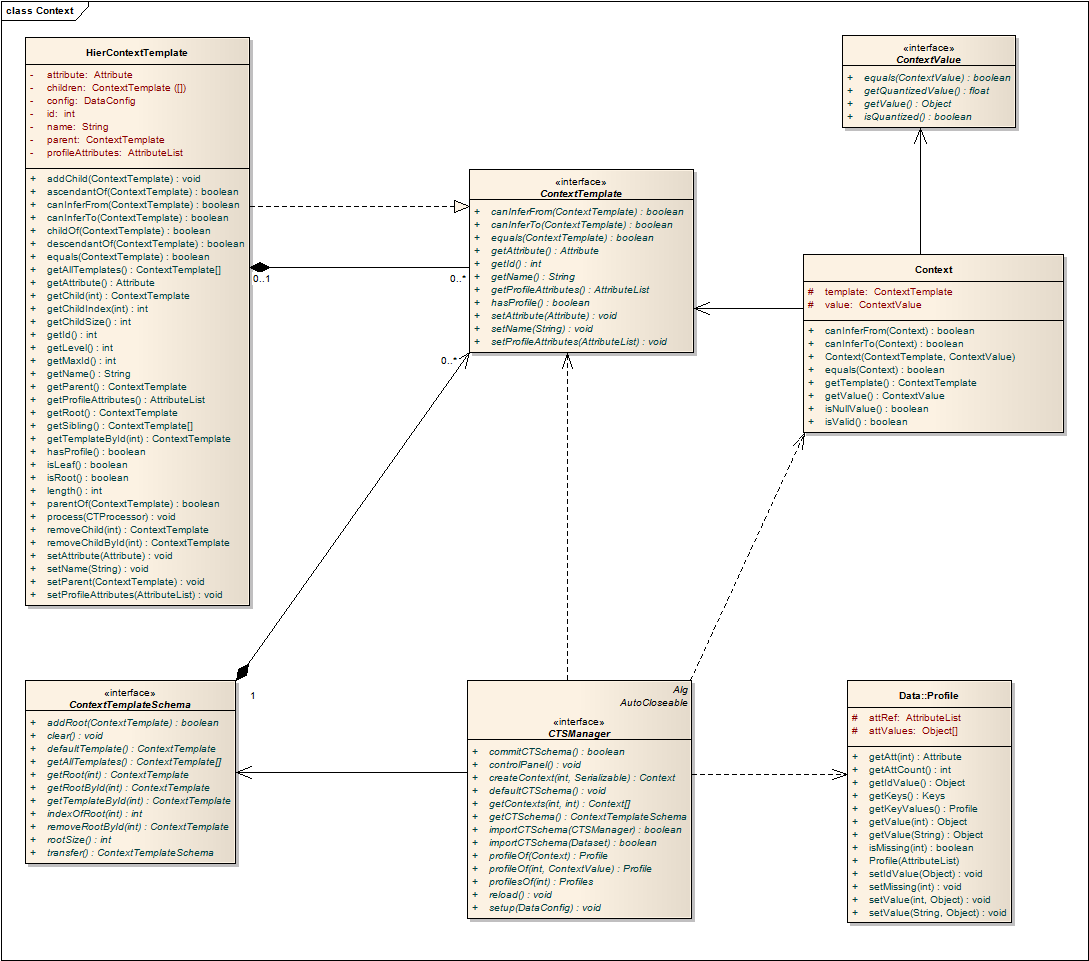
**Phụ lục A**



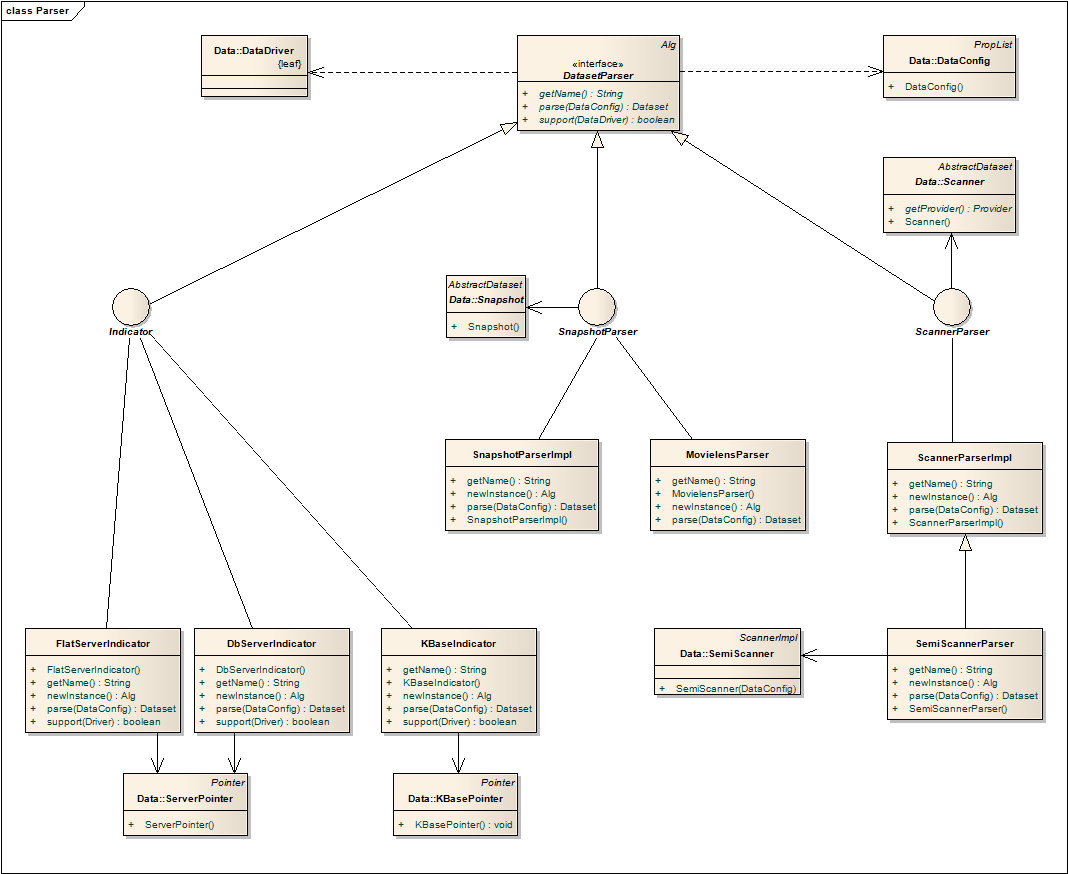
**Hình 5**. Sơ đồ lớp gói *Data* (*dữ liệu*)



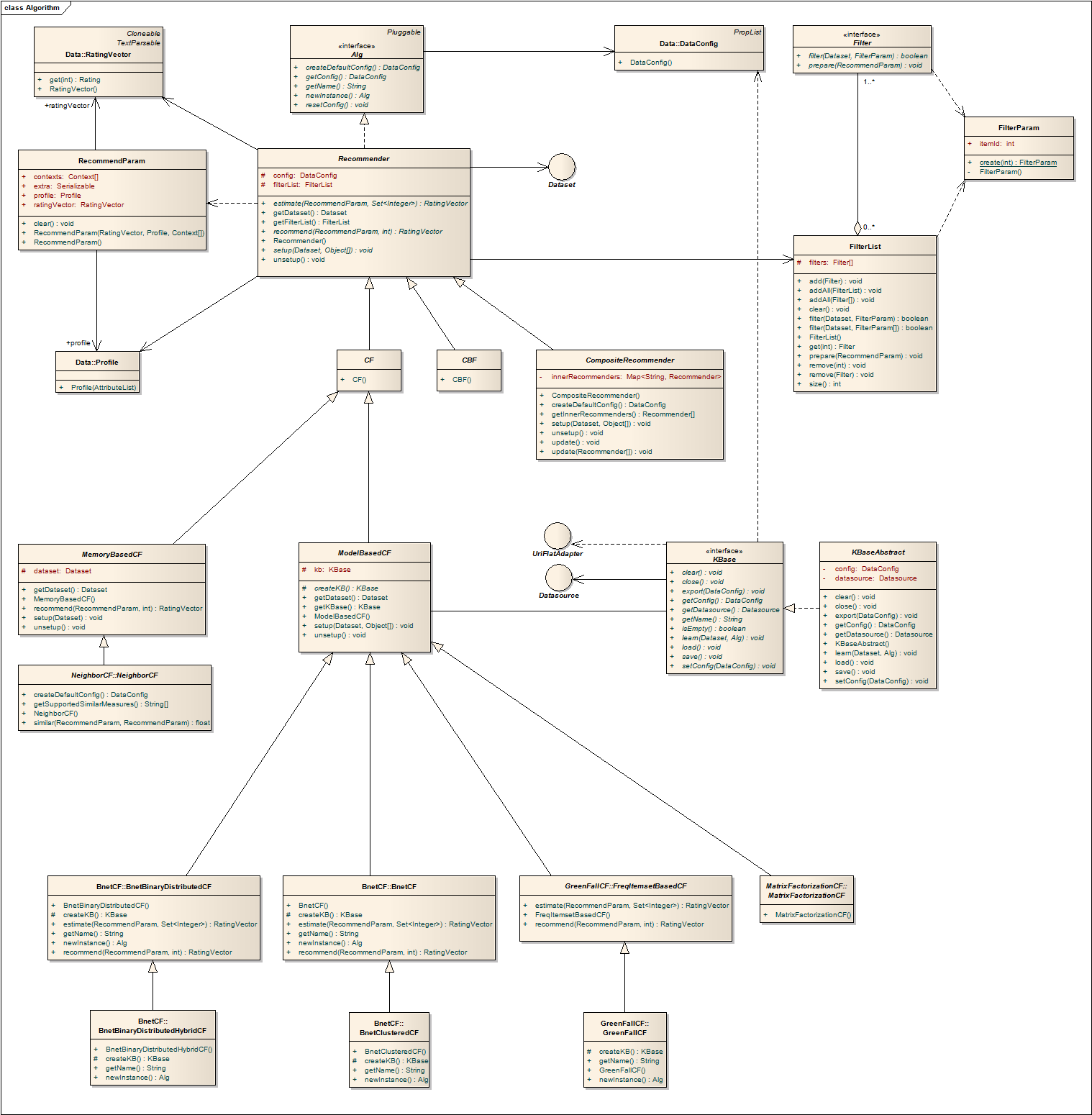
**Hình 6**. Sơ đồ lớp gói *Data.BitData* (*dữ liệu nhị phân*)



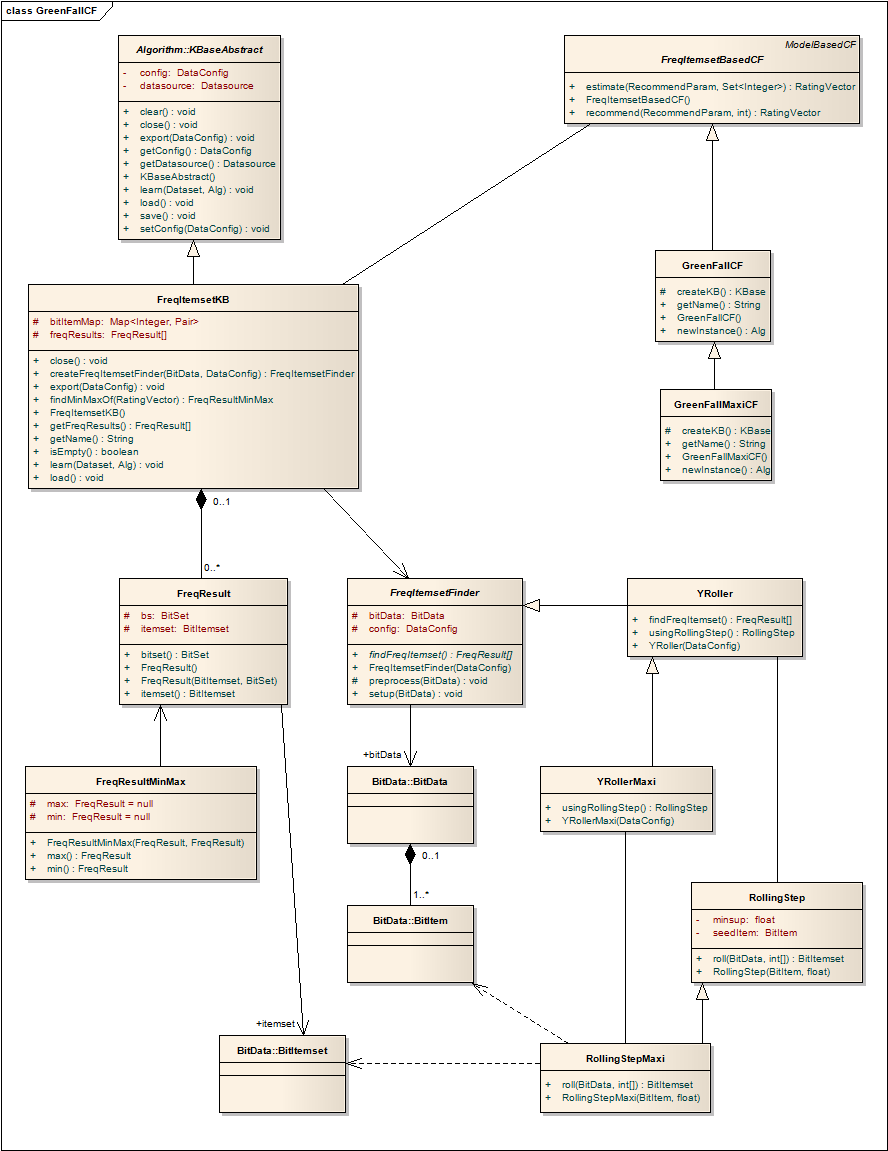
**Hình 7**. Sơ đồ lớp gói *Data.Context* (*dữ liệu ngữ cảnh*)



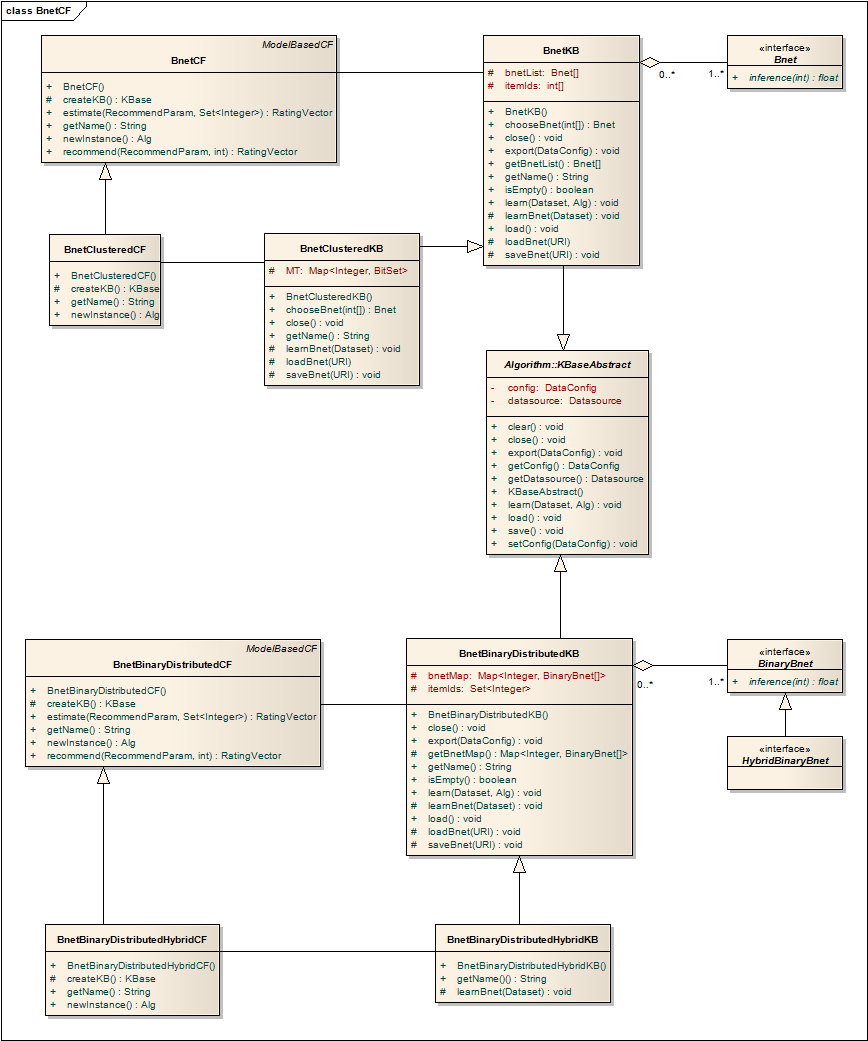
**Hình 8**. Sơ đồ lớp gói *Parser* (*phân tích*)



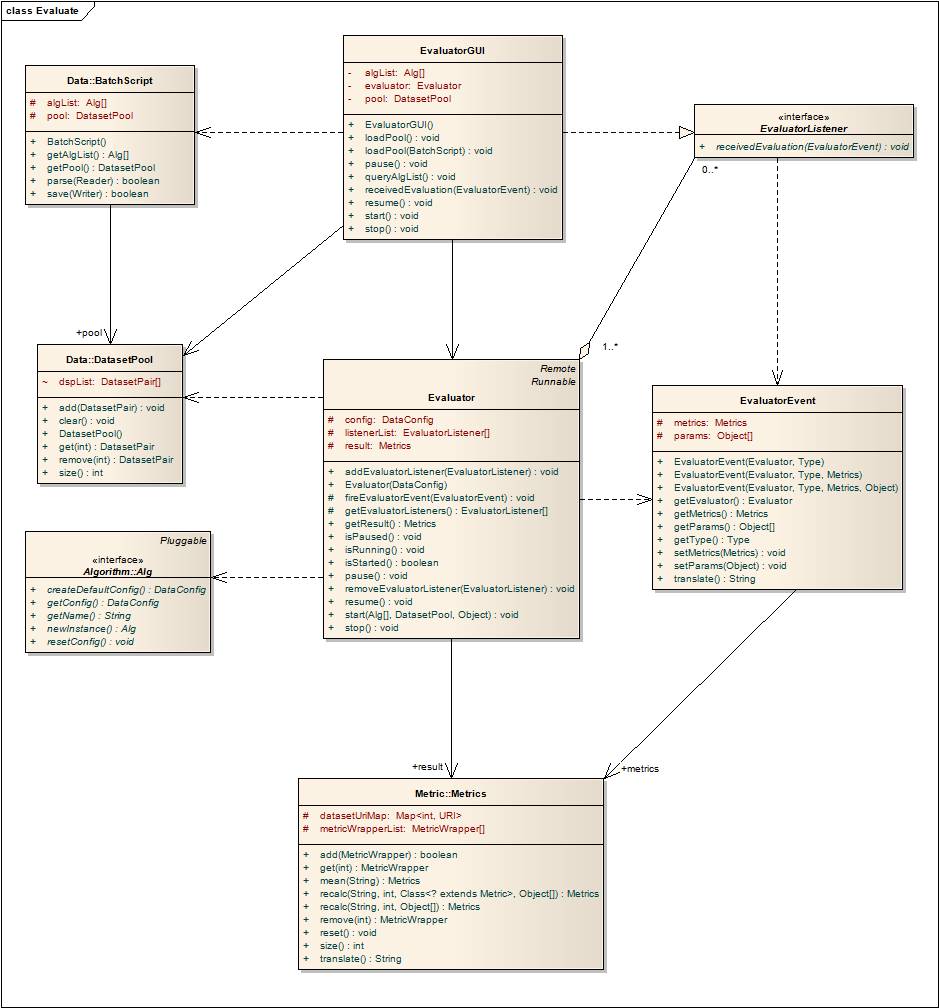
**Hình 9**. Sơ đồ lớp gói *Algorithm* (*giải thuật*)



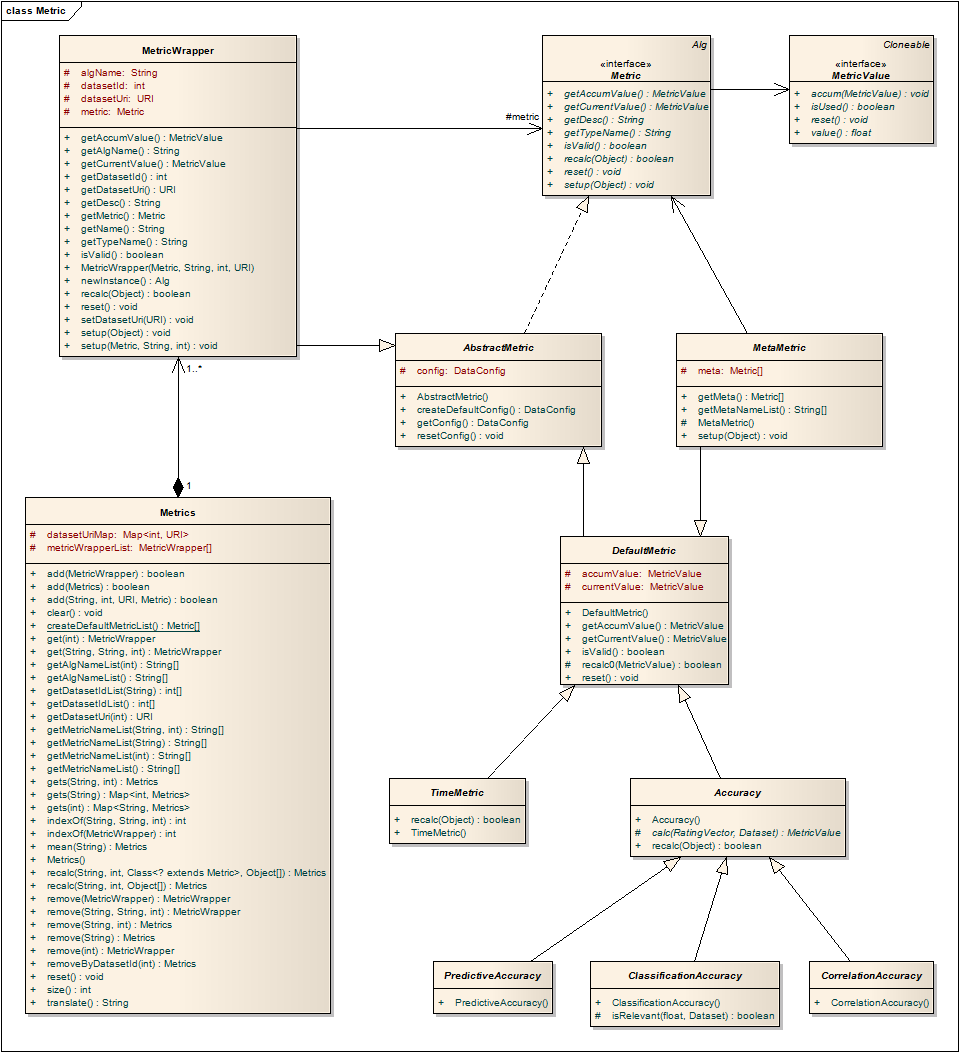
**Hình 10**. Sơ đồ lớp gói *Algorithm.GreenFallCF* (giải thuật *Green Fall*)



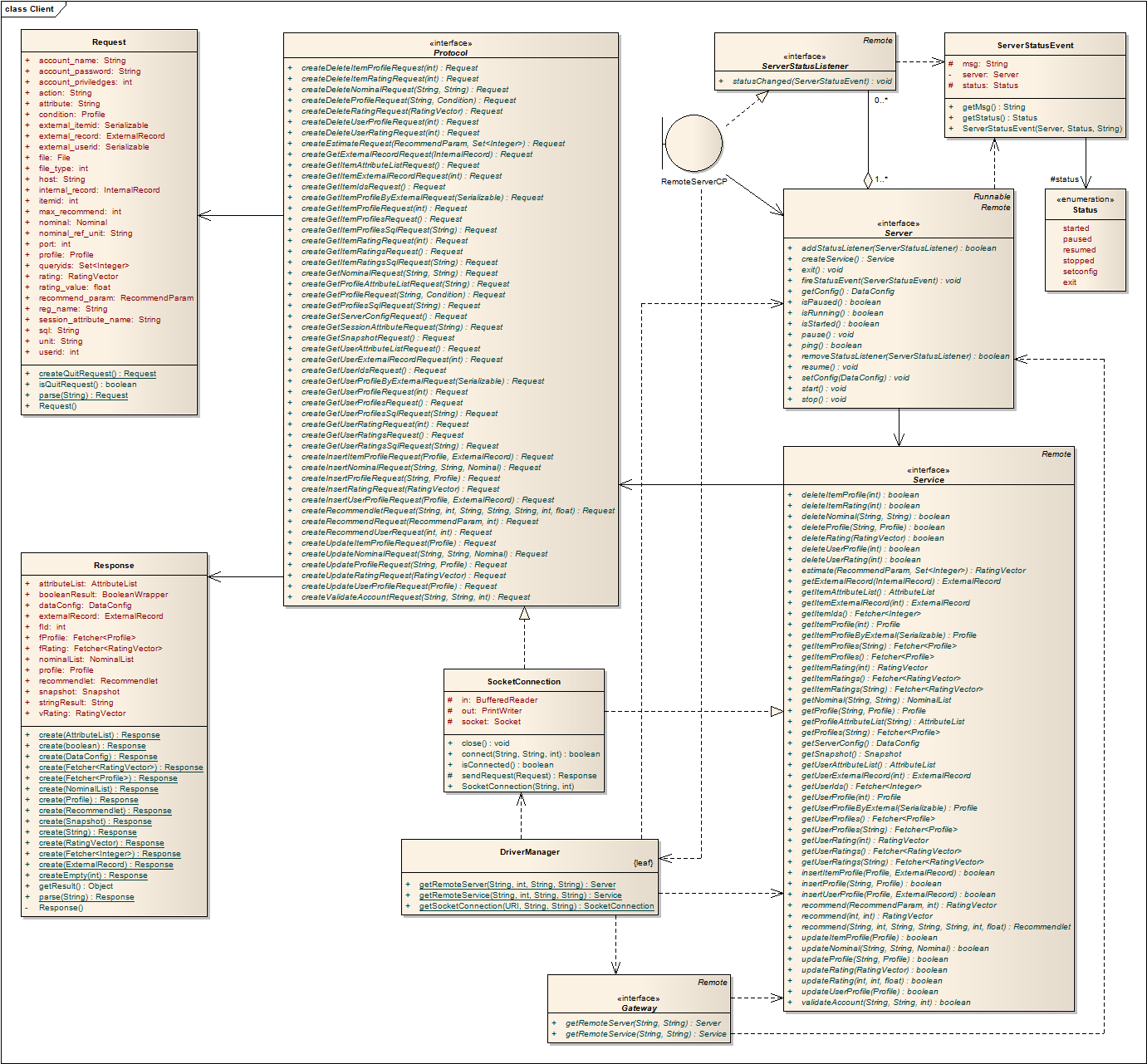
**Hình 11**. Sơ đồ lớp gói *Algorithm.BnetCF* (*giải thuật Bayesian*)



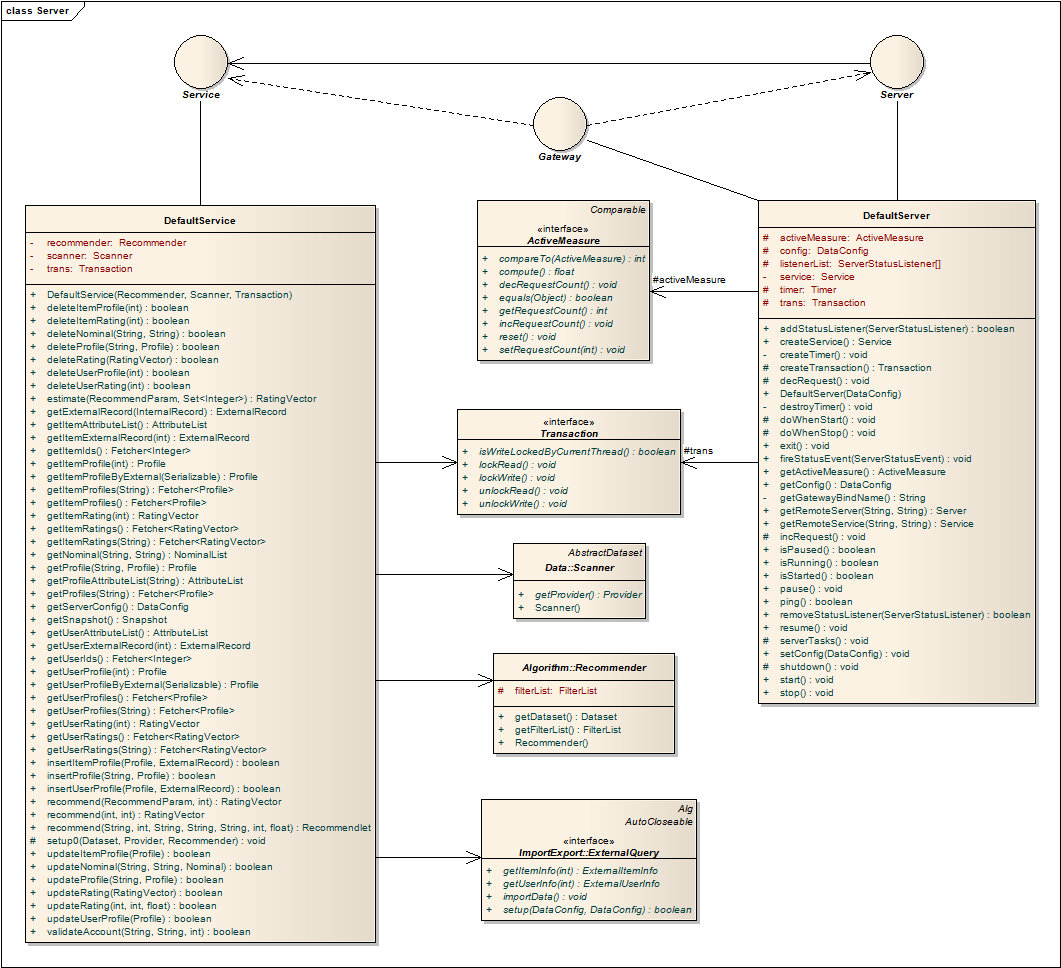
**Hình 12**. Sơ đồ lớp gói *Evaluate* (*đánh giá*)



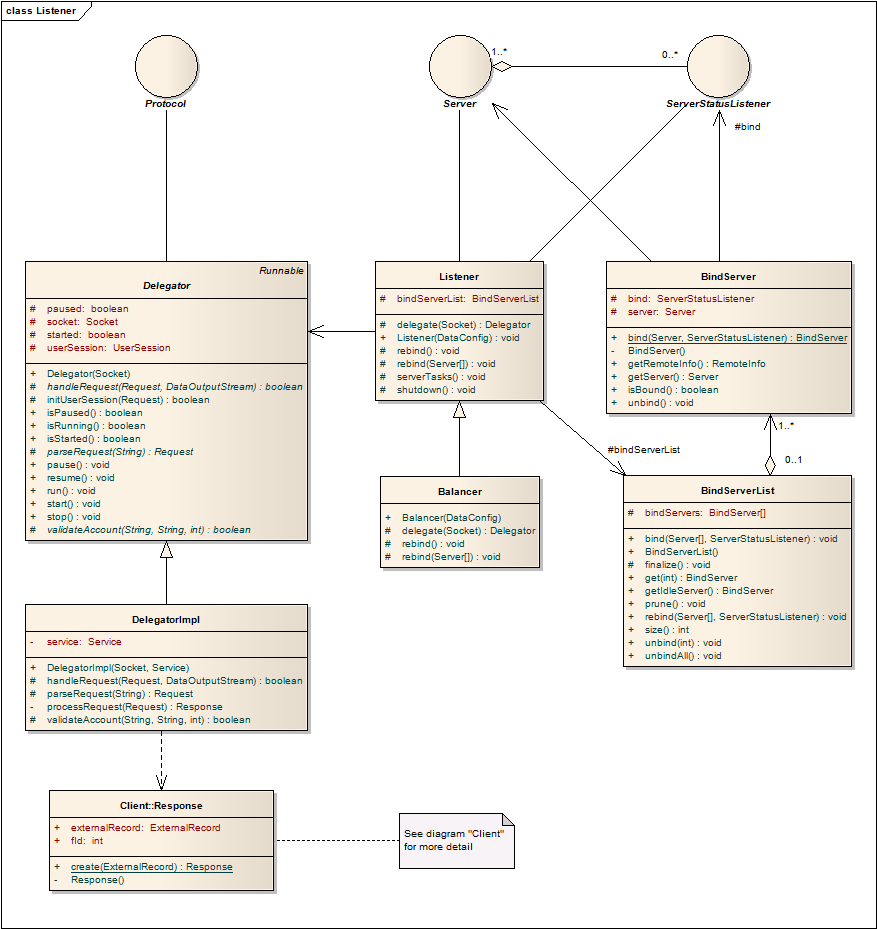
**Hình 13**. Sơ đồ lớp gói *Evaluate.Metric* (*độ đo*)



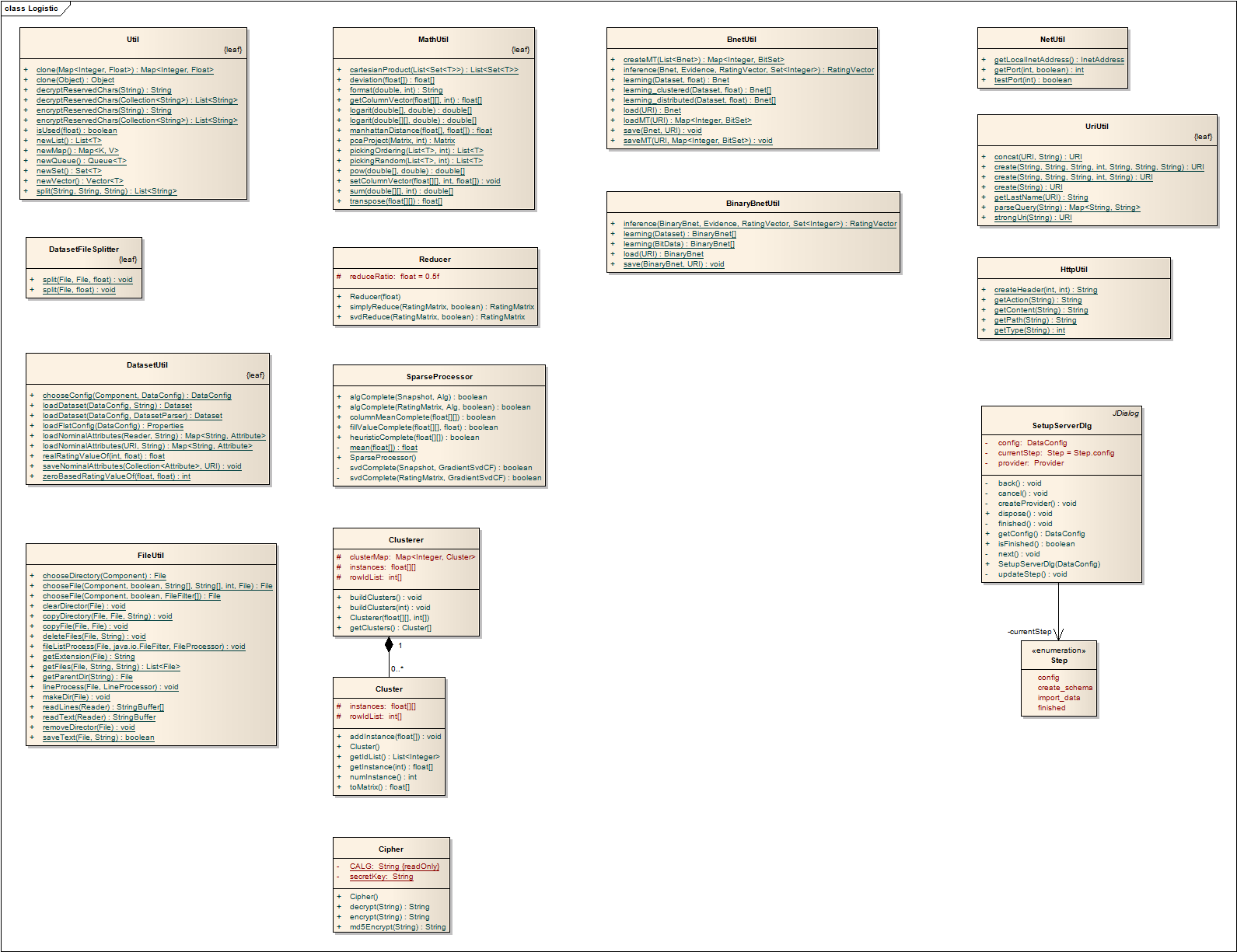
**Hình 14**. Sơ đồ lớp gói *Client* (*khách*)



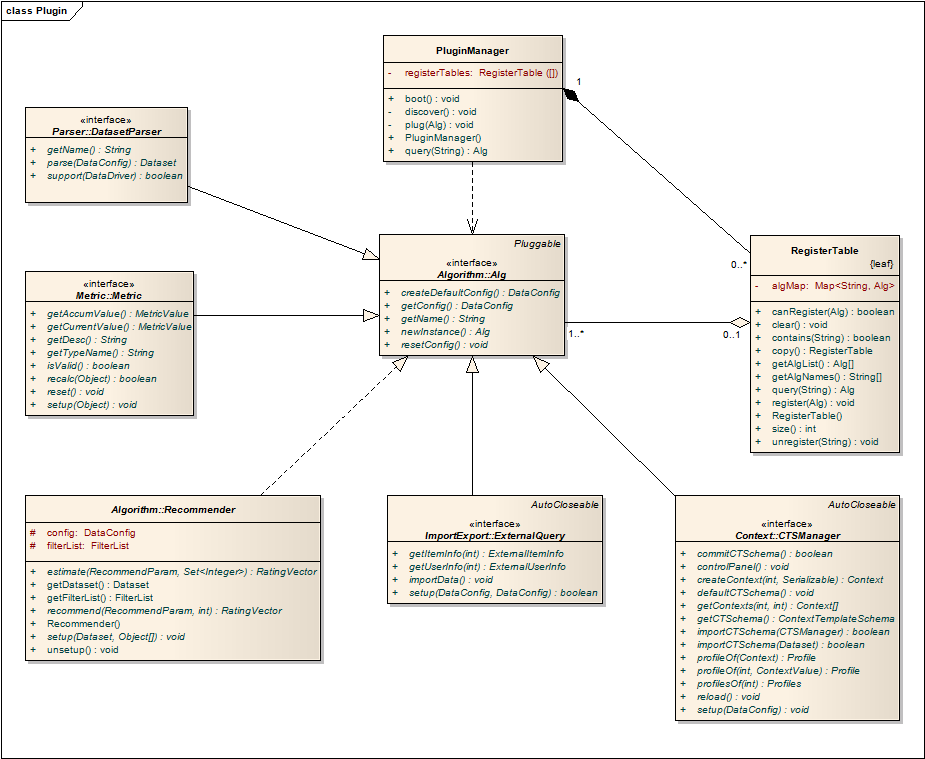
**Hình 15**. Sơ đồ lớp gói *Server* (*chủ*)



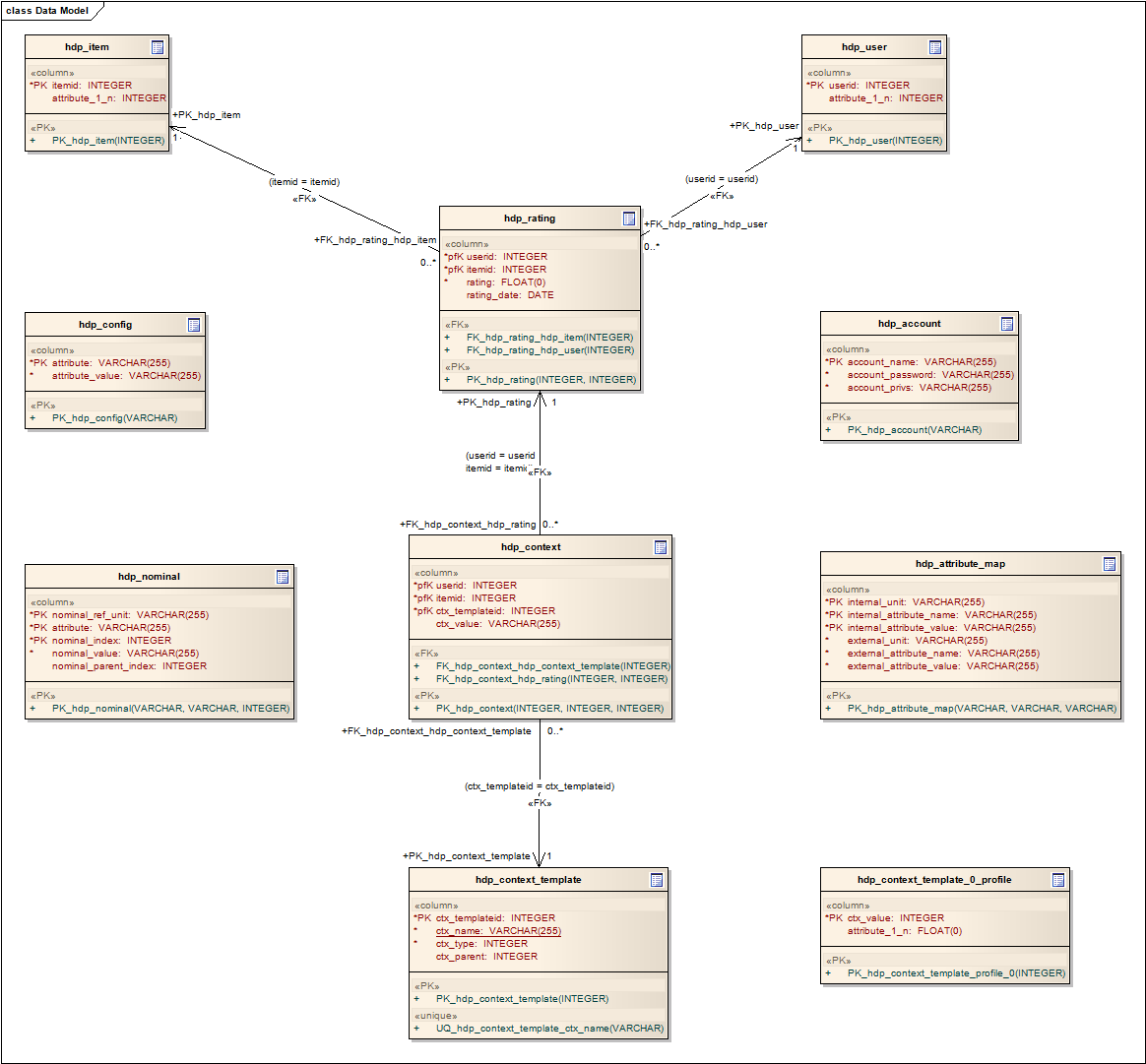
**Hình 16**. Sơ đồ lớp gói *Listener* (*bộ nghe*)



**Hình 17**. Sơ đồ lớp gói *Logistic* (*tiện ích*)



**Hình 18**. Sơ đồ lớp gói *Plugin* (*phích cắm*)



**Hình 19**. Sơ đồ cơ sở dữ liệu

**Phụ lục B**

**Một cách tiếp cận mới cho lọc cộng tác dựa trên khai thác tập phổ biến**

Nguyễn Phước Lộc

Công ty TNHH MTV Lập trình Hướng Dương, Tp. Hồ Chí Minh city, Việt Nam

**Tóm tắt**

Lọc cộng tác (collaborative filtering – CF) là một trong những giải thuật khuyến nghị phổ biến, mỗi mặt hàng giới thiệu cho một người dùng dựa vào kết quả khảo sát các đánh giá từ cộng đồng nhiều người dùng. Có hai cách tiếp cận cho CF: nạp ức (memory-based) và mô thức (model-based).

Chúng tôi đề xuất một giải thuật CF mô thức mới dựa trên khai thác tập phổ biến từ cơ sở dữ liệu. Theo đó, những mặt hàng trong tập phổ biến sẽ được khuyến nghị cho người dùng. Tác vụ khai khoáng tập phổ biến được thực hiện ngoại tuyến nên không ảnh hướng đến tác vụ khuyến nghị, vì vậy giải thuật đề xuất có khả năng đáp ứng tức thời yêu cầu khuyến nghị.

Do trọng tâm của giải thuật CF mô thức mới là khai thác tập phổ biến, chúng tôi cũng đề xuất một giải thuật khai khác tập phổ biến mới được gọi là Roller nhằm tăng hiệu quả của tiến trình khai thác tập phổ biến. Giải thuật Roller dựa trên giả định “*độ hỗ trợ* (*support*) *của một mặt hàng càng lớn thì khả năng xuất hiện mặt hàng này trong một tập phổ biến nào đó càng cao*”. Đây là giải thuật thông minh (heuristic), mô phỏng thao tác lăn sơn tường, theo đó, tập phổ biến được khởi gán và mở rộng qua động tác lăn sơn tường.

Ngoài ra, chúng tôi cũng đề xuất những kỹ thuật tăng cường như biểu diễn nhị phân (bit representation), so khớp nhị phân (bit matching) cũng như khai thác nhị phân (bit mining) nhằm gia tăng tốc độ thuật toán. Điểm cơ bản của các kỹ thuật này là giảm không gian lưu trữ và thời gian xử lý bằng toán tử nhị phân như *AND, NOT*.

**Từ khóa**: lọc cộng tác, khai thác tập phổ biến, so khớp nhị phân, khai thác nhị phân

**1. Giới thiệu**

Hệ thống khuyến nghị (recommendation system) là hệ thống giới thiệu những mặt hàng mà người dùng có khả năng yêu thích nhất trong một cơ sở dữ liệu lớn gồm rất nhiều mặt hàng. Mặt hàng có thể là sản phẩm, dịch vụ, sách, tạp chí và tất cả những gì mà người dùng với vai trò khách hàng có thể mua. Những mặt hàng được giới thiệu đó thể hiện sở thích (interests) hay mối quan tâm của người dùng.

Hệ thống khuyến nghị được xây dựng theo hai hướng: lọc nội dung (content-based filtering – CBF) và lọc cộng tác (collaborative filtering – CF) [1]:

* CBF giới thiệu một mặt hàng với người dùng nếu mặt hàng này tương tự về nội dung với những mặt hàng mà họ đã yêu thích với ngữ nghĩa rằng mặt hàng yêu thích là mặt hàng có đánh giá cao. Nội dung của một mặt hàng là những thuộc tính của nó và tập nội dung tất cả các mặt hàng tạo thành ma trận, gọi là ma trận nội dung (content matrix).
* Với tiếp cận khác, CF giới thiệu một mặt hàng với người dùng nếu những láng giềng của họ cũng thích mặt hàng đó với định nghĩa láng giềng là những người giống nhau về sở thích. Đánh giá của một người dùng lên một mặt hàng thể hiện sở thích của họ đối với mặt hàng đó. Tất cả đánh giá người dùng trên các mặt hàng tạo thành ma trận, gọi là ma trận đánh giá (rating matrix).

Cả hai giải thuật CBF và CF đều có ưu nhược điểm riêng. CBF tập trung vào nội dung mặt hàng và sở thích cá nhân người dùng nên mục tiêu của nó là khuyến nghị những mặt hàng khác nhau cho những người dùng khác nhau. Điểm mạnh của CBF là khả năng đưa ra những khuyến nghị đặc thù nhưng CBF không hướng đến cộng đồng như CF. Khi đó những mặt hàng mà người dùng có thể thích bị ẩn trong cộng đồng, không thể phát hiện được bằng CBF. Đây là nhược điểm của CBF, một nhược điểm khác, CBF có thể dự đoán sai sở thích người dùng nếu số lượng người dùng quá lớn.

Nếu nội dung mỗi mặt hàng quá lớn với nhiều thuộc tính, giải thuật CBF tiêu thụ nhiều tài nguyên máy tính và do đó thời gian xử lý sẽ chậm; ngược lại tốc độ không thành vấn đề đối với CF trong trường hợp này do CF không tập trung vào các thuộc tính mặt hàng. Thay vào đó, CF chỉ quan tâm đến đánh giá của người dùng trên mỗi mặt hàng và đây chính là điểm mạnh của CF. Tuy nhiên, mặc dù không đối mặt với vấn đề dữ liệu nội dung các mặt hàng quá lớn, nhược điểm của CF cũng nảy sinh từ đây, bởi vì CF sẽ đưa ra những khuyến nghị ngoài mong đợi trong trường hợp mà một số mặt hàng có thể phù hợp với người dùng nhưng họ chưa đánh giá chúng. Đây là vấn đề dữ liệu thưa, nghĩa là ma trận đánh giá bị thiếu nhiều giá trị hay nói cách khác có quá nhiều mặt hàng chưa được đánh giá. Hai kỹ thuật hữu ích được đề xuất để khắc phục nhược điểm này của CF:

* Kết hợp CBF và CF: kỹ thuật này gồm hai giai đoạn, giai đoạn đầu áp dụng CBF để hoàn chỉnh ma trận đánh giá, giai đoạn sau áp dụng CF để thực hiện khuyến nghị. Kỹ thuật này hữu ích khi cải thiện độ chính xác của khuyến nghị nhưng tốn nhiều thời gian cho giai đoạn đầu đóng vai trò như bước tiền xử lý. Kỹ thuật dùng cả ma trận đánh giá và ma trận nội dung
* Mô hình hóa ma trận đánh giá như một mô hình suy diễn, ước lượng giá trị bị thiếu và thực hiện khuyến nghị dựa trên mô hình suy diễn này. Giải thuật CF theo cách tiếp cận mô hình này được gọi là CF mô thức (model-based CF), còn một cách tiếp cận khác là CF nạp ức (memory-based CF). CF mô thức áp dụng các phương pháp thống kê và máy học để khai thác ma trận đánh giá.

Mặc dù cách tiếp cận mô thức không cho kết quả chính xác như cách tiếp cận kết hợp nhưng nó giải quyết vấn đề dữ liệu khổng lồ và ma trận thưa; hơn nữa nó có thể đáp ứng yêu cầu người dùng một cách tức thời qua cơ chế suy diễn nhanh nhạy. Vì vậy, bài báo này tập trung vào cách tiếp cận mô thức dựa trên khai khoáng tập phổ biến. Đây là hướng tiếp cận đầy tiềm năng vì tận dụng kỹ thuật khai khoáng [4], dẫn đến tốc độ nhanh và chất lượng khuyến nghị cao. Ý tưởng chính giải thuật được đề cập trong phần 2. Chi tiết giải thuật khai khoáng thông minh được mô tả chi tiết trong phần 3. Phần 4 là đánh giá và phần 5 là kết luận. Lưu ý trong bài báo các thuật ngữ như ma trận đánh giá (rating matrix), cơ sở dữ liệu đánh giá (rating database), tập huấn luyện (training set) có cùng nghĩa.

**2. Một giải thuật lọc cộng tác mới dựa trên khai thác tập phổ biến**

Giả sử với vector đánh giá *u* = {*item 1 = 3, item 2 = 5, item 3 = 2*}, nghĩa là người dùng *u* đánh giá các mặt hàng *item 1, item 2, item 3* lần lượt với các giá trị *3, 5, 2*. Giải thuật CF đề xuất, dựa trên khai thác tập phổ biến, gồm hai tiến trình sau:

* Tiến trình mô hình hóa: hoạt động ngoại tuyến với mục đích khai thác danh sách các tập phổ biến từ ma trận đánh giá.
* Tiến trình khuyến nghị: bất cứ khi nào người dùng *u* yêu cầu những mặt hàng yêu thích, một tập phổ biến *s* được rút từ *S* sao cho *s* chứa các mặt hàng *item 1, item 2, item 3*; ví dụ *s* = {*item 1, item 2, item 3, item 5, item 7*}. Các mặt hàng thêm vào là 5 và 7 sẽ được khuyến nghị cho người dùng. Không giống với tiến trình mô hình hóa, tiến trình khuyến nghị hoạt động trực tuyến đáp ứng tức thời yêu cầu khuyến nghị.

Tiến trình mô hình hóa tốn nhiều thời gian hơn tiến trình khuyến nghị nhưng do hoạt động ngoại tuyến nên không ảnh hưởng đến tiến trình khuyến nghị. Tuy nhiên vấn đề quan trọng phát sinh khi tập phổ biến *s* = {*item 1, item 2, item 3, item 5, item 7*} không thể hiện giá trị đánh giá của các mặt hành *item 1, item 2, item 3* vốn là 3, 5, 2 tương ứng trong vector đánh giá *u*. Điều này có nghĩa là vector đánh giá *u* và tập phổ biến *s* không khớp hoàn toàn. Một khó khăn khác xuất hiện khi không thể ước lượng các giá trị bị thiếu, ví dụ: không thể ước lượng giá trị đánh giá cho các mặt hàng *5* và *7*. Các vấn đề này được giải quyết bằng kỹ thuật *biến đổi nhị phân* (binary transformation). Lưu ý, thuật ngữ *bit* và *nhị phân* cùng ý nghĩa trong bài báo này.

Giả sử với ma trận đánh giá có dòng đại diện người dùng, cột đại diện mặt hàng, mỗi ô là giá trị đánh giá của một người dùng lên một mặt hàng. Các đánh giá dao động trong khoảng {*1 … 5*} với *5* biểu thị sự yêu thích nhiều nhất. Bảng biểu *1* thể hiện một ma trận đánh giá mẫu.

Mỗi mặt hàng được “tách” thành *5* mặt hàng con (sub-items) tương ứng với *5* giá trị đánh giá khả dĩ {*1 … 5*}. Mỗi mặt hàng con *item\_j\_k* có hai trạng thái nhị phân *1* và *0* cho biết người dùng có hay không đánh giá mặt hàng *j* với giá trị *k*. Ví dụ, trường hợp *item\_2\_5* có trạng thái *1* cho biết người dùng đã đánh giá mặt hàng *2* với giá trị *5*. Bây giờ ma trận đánh giá biến đổi thành ma trận đánh giá nhị phân (bit rating matrix), trong đó mỗi ô là một đánh giá nhị phân đối với một mặt hàng con (sub-item). Quy ước ô rỗng là ô nhận trạng thái *0*; nghĩa là không có người dùng đánh giá trên ô đó. Ma trận đánh giá nhị phân thể hiện trong bảng biểu 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Item 1 | Item 2 | Item 3 | Item 4 |
| User 1 | *3* | *5* | *2* | *1* |
| User 2 | *3* | *5* | *2* | *1* |
| User 3 | *1* | *5* | *4* |  |

**Bảng 1.** Ma trận đánh giá

|  |  |  |  |
| --- | --- | --- | --- |
|  | User 1 | User 2 | User 3 |
| Item\_1\_1 |  |  | *1* |
| Item\_1\_3 | *1* | *1* |  |
| Item\_2\_5 | *1* | *1* | *1* |
| Item\_3\_2 | *1* | *1* |  |
| Item\_3\_4 |  |  | *1* |
| Item\_4\_1 | *1* | *1* |  |

**Table 2.** Ma trận đánh giá nhị phân

Mỗi tập phổ biến *s* trích xuất từ ma trận đánh giá nhị phân sẽ ở dạng nhị phân, *s =* (*item\_j1\_k1, item\_j2\_k2, …, item\_jn\_kn*), trong đó mỗi phần tử *item\_j\_k* là một mặt hàng con nhị phân (*bit sub-item*). Sau đó mỗi vector đánh giá được biến đổi thành vector đánh giá nhị phân (*bit rating vector*) *u* = (*item\_j1\_k1, item\_j2\_k2,…, item\_jn\_kn*). Việc so khớp giữ tập phổ biến nhị phân và vector đánh giá nhị phân trở nên dễ dàng. Với ví dụ trên, vector đánh giá *u* = (*item 1 = 3, item 2 = 5, item 3 = 2*) được chuyển thành vector đánh giá nhị phân *u =* (*item\_1\_3, item\_2\_5, item\_3\_2*), giả sử có hai tập phổ biến nhị phân *s1 =* (*item\_1\_3, item\_2\_5, item\_3\_2, item\_4\_1*) and *s2 =* (*item\_1\_1, item\_2\_5, item\_3\_4*). Chúng ta nhận thấy *s1* khớp với *u* nhất, do đó mặt hàng *4*, mặt hàng khác biệt trong *s1* sẽ được khuyến nghị cho người dùng với đánh giá ước lượng là *1*.

Vậy, các vấn đề trên đã được giải quyết nhưng giải thuật chúng tôi cần được cải tiến. Nếu số lượng tập phổ biến lớn và mỗi tập phổ biến có nhiều mặt hàng, sẽ có sự bùng nổ tổ hợp làm giảm hiệu năng máy tính khi so khớp vector đánh giá và tập phổ biến. Do đó chúng tôi đề xuất một phương pháp cải tiến việc so khớp, được gọi là kỹ thuật so khớp nhị phân (*bit matching*).

**2.1. Biểu diễn nhị phân và so khớp nhị phân**

Giả sử vector đánh giá hay tập phổ biến có *4* mặt hàng và mỗi mặt hàng có *5* giá trị đánh giá, chúng ta dùng một tập bit (bit set) dài *4 \* 5* = *20* bit để biểu diễn vector đánh giá này. Tập bit được chia thành những cụm (cluster) hay nhóm (group), ví dụ, nếu mặt hàng có *5* giá trị đánh giá thì mỗi cụm sẽ có *5* bit. Như vậy mỗi cụm đại diện một mặt hàng con (sub-item) và vị trí của một bit trong một cụm cho biết giá trị đánh giá của mặt hàng con tương ứng. Nếu cụm có một bit được bật, mặt hàng con tương ứng được đánh giá với giá trị là vị trí của bit được bật đó trong cụm. Sau đây là một ví dụ của tập bit.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *0* | *0* | *1* | *0* | *0* | *0* | *0* | *0* | *0* | *1* | *0* | *1* | *0* | *0* | *0* | *0* | *0* | *0* | *0* | *0* |
| Cụm *1*  (*item 1 = 3*) | | | | | Cụm *2*  (*item 2 = 5*) | | | | | Cụm *3*  (*item 3 = 2*) | | | | | Cụm *4* | | | | |

**Bảng 3**. Biểu diễn bit

Ví dụ, vector đánh giá *u* = (*item 1 = 3, item 2 = 5, item 3 = 2*) được chuyển thành *u =* (*item\_1\_3, item\_2\_5, item\_3\_2*) biểu diễn dưới dạng nhị phân *u =* (*00100 00001 01000 00000*) gồm *4* cụm. Tập phổ biến *s1 =* (*item\_1\_3, item\_2\_5, item\_3\_2, item\_4\_1*) có dạng nhị phân là *s1 =* (*00100 00001 01000 10000*). Tập phổ biến *s2 =* (*item\_1\_1, item\_2\_5, item\_3\_4*)có dạng nhị phân là *s2 =* (*10000 00001 00010 00000*). Để so khớp *s1* (hay *s2*) với *u*, ta dùng toán tử nhị phân AND:

* Nếu *s1 AND u = u* thì *s1* khớp với *u*
* Ngược lại, nếu *s1 AND u u* thì *s1* không khớp *u*

Nếu *s1* khớp với *u* thì toán tử nhị phân *AND-NOT* được dùng để rút trích mặt hàng cần khuyến nghị với người dùng. Đặt *r\_item* là mặt hàng cần khuyến nghị, ta có:

*r\_item* = *s1 AND* (*NOT u*) = (*00000 00000 00000 10000*)

Tập bit của *r\_item* cho thấy bit đầu tiên của cụm thứ tư được bật, do đó mặt hàng *4* được khuyến nghị với giá trị ước lượng là *1*. Nhìn chung, giải thuật đề xuất gồm 3 bước:

* *Bước 1*: Biến đổi ma trận đánh giá thành ma trận đánh giá nhị phân.
* *Bước 2*: Rút trích những tập phổ biến từ ma trận đánh giá nhị phân bằng kỹ thuật khai khoáng.
* *Bước 3*:Vector người dùng và tập phổ biến biểu diễn dưới dạng tập bit. Toán tử so khớp bit được sử dùng để tìm ra tập phổ biến khớp với vector người dùng, đó là tập phổ biến thích hợp. Các mặt hàng cần khuyến nghị cho người dùng sẽ được rút trích từ tập phổ biến thích hợp này, hơn nữa, các giá trị bị thiếu cũng được ước lượng.

**2.2. Mã giả tựa C cho giải thuật đề xuất**

Đặt *D, B* và *S* lần lượt là ma trận đánh giá, ma trận đánh giá nhị phân và danh sách các tập phổ biến. Giả sử *matched\_itemset* và *r\_item* lần lượt là tập phổ biến thích hợp (đã được khớp) và mặt hàng khuyến nghị. Đặt *bitset*(…) là hàm biến đổi mặt hàng thành tập bit và *count*(…) là hàm đếm số bit *1* trong tập bit. Đặt *bit\_transform* là hàm biến đổi ma trận mặt hàng thành ma trận mặt hàng nhị phân và *mining\_frequent\_itemset* là hàm khai thác tập phổ biến từ ma trận đánh giá nhị phân (xem mục *3.1, 3.2*). Sau đây là mã giả tựa *C* cho giải thuật đề xuất:

*B = bit\_transform*(*D*)

*S = mining\_frequent\_itemset*(*B*)

*matched\_itemset = null*

*max\_count = -1*

*For each s S*

*bs = bitset*(*u*) *AND bitset*(*s*)

*If bs = bitset*(*u*) *&& count*(*bs*) *> max\_count then*

*matched\_itemset = s*

*max\_count = count*(*bs*)

*End If*

*End For*

*r\_item = bitset*(*matched\_itemset*) *AND* (*NOT bitset*(*u*))

Bước thứ hai – khai khoáng tập phổ biến từ ma trận đánh giá nhị phân, quan trọng nhất; do đó phần tiếp theo sẽ đề cập đến phương pháp khai thác tập phổ biến.

**3. Khai thác tập phổ biến**

Chúng tôi đề xuất phương pháp khai thác tập phổ biến dựa trên giả định “*độ hỗ trợ* (support) *của một mặt hàng càng lớn thì khả năng xuất hiện mặt hàng này trong một tập phổ biến nào đó càng cao*”. Nói cách khác, những mặt hàng có độ hỗ trợ lớn có xu hướng kết hợp với nhau để tạo thành tập phổ biến. Giải thuật chúng tôi, tên gọi *Roller*, là một giải thuật thông minh (heuristic). Ý tưởng chính giống thao tác lăn sơn tường. Hãy tưởng tượng, dữ liệu huấn luyện, cụ thể là ma trận đánh giá, được mô phỏng như một bức tường. Các mặt hàng được bố trí trên mặt tường theo thứ tự giảm dần độ hỗ trợ, nghĩa là, mặt hàng phổ biến thấp theo sau mặt hàng phổ biến cao. Sau đó, chúng ta có một cây lăn tường và lăn nó trên bức tường, quét qua các mặt hàng theo thứ tự giảm dần độ hỗ trợ. Nếu một mặt hàng thỏa ngưỡng hỗ trợ dưới (minimum support – *min\_sup*) thì nó được thêm vào tập phổ biến và thao tác lăn này tiếp tục cho đến khi không còn mặt hàng nào thỏa ngưỡng hỗ trợ dưới. Lần tiếp theo, tất cả mặt hàng trong tập phổ biến trước đó sẽ bị gỡ khỏi bức tường và thao tác lăn tường mới lại tiếp tục để tìm tập phổ biến khác. Giải thuật gồm bốn bước sau:

* *Bước 1*: tính độ hỗ trợ tất cả các mặt hàng và sắp xếp chúng trên bức tường theo một danh sách ứng với độ hỗ trợ giảm dần, tiêu chuẩn loại trừ là những mặt hàng không thỏa ngưỡng hỗ trợ dưới *min\_sup*. Nói cách khác, bức tường chỉ gồm các mặt hàng thỏa ngưỡng hỗ trợ dưới, được gọi là mặt hàng phổ biến (frequent items) và mặt hàng đầu tiên trong danh sách là mặt hàng có độ hỗ trợ lớn nhất.
* *Bước 2*: Tập phổ biến thứ *i* (i = ) được khởi tạo là mặt hàng đầu tiên trong danh sách và có độ hỗ trợ cũng bằng độ hỗ trợ của mặt hàng đầu tiên. Mặt hàng đầu tiên này trở thành mặt hàng hiện hành và sẽ được gỡ khỏi danh sách.
* *Bước 3*: Nếu không còn mặt hàng nào trong danh sách, giải thuật sẽ dừng. Ngược lại:
  + Nếu mặt hàng hiện hành là mặt hàng cuối cùng trong danh sách thì tất cả mặt hàng trong tập phổ biến thứ *i* sẽ bị gỡ khỏi danh sách và *i* được tăng lên *1* (*i = i + 1*). Trở lại bước 2.
  + Nếu mặt hàng hiện hành KHÔNG phải mặt hàng cuối cùng trong danh sách thì mặt hàng kế tiếp được xét và trở thành mặt hàng hiện hành.
* *Bước 4*:
  + Kiểm tra độ hỗ trợ của mặt hàng hiện hành, nếu thỏa ngưỡng hỗ trợ dưới thì độ hỗ trợ của tập phổ biến thứ *i*, ký hiệu *support*(*itemset i*) được cộng tích lũy với mặt hàng hiện hành, chính là tổng tất cả giao dịch chứa cả tập phổ biến thứ *i* và mặt hàng hiện hành. Nếu *support*(*itemset i*) lớn hơn hoặc bằng ngưỡng hỗ trợ dưới thì mặt hàng hiện hành được thêm vào tập phổ biến thứ *i*.
  + Trở lại bước 3.

Bước *3* và *4* tương tự thao tác sơn lăn tường với cây lăn là tập phổ biến thứ *i*. Mỗi lần lặp là một lần lăn, sau mỗi lần lăn, tập phổ biến sẽ dày lên vì đính càng nhiều mặt hàng.

Gọi *I =* {*i1, i2, …, im*} là danh sách các mặt hàng và *S* là danh sách các tập phổ biến. Đặt *C =* {*c1, c2, …, cn*} là danh sách các mặt hàng có độ hỗ trợ thỏa ngưỡng hỗ trợ dưới và sắp theo thứ tự giảm dần, *C I*. Đặt *si* là tập phổ biến thứ *i*. Đặt *c* là mặt hàng hiện hành. Đặt *filter\_minimum\_support*(…) là hàm lọc ra những mặt hàng có độ hỗ trợ lớn hơn hoặc bằng ngưỡng hỗ trợ dưới. Đặt *sort*(…), *first*(…), *next*(…), *last*(…) lần lượt là các hàm sắp xếp, lấy phần tử đầu tiên, lấy phần tử cuối cùng từ một danh sách các mặt hàng. Sau đây là mã giả tựa *C* cho giải thuật Roller được lập trình như hàm *mining\_frequent\_itemsets*.

*C = filter\_minimum\_support*(*I*)

*C = sort*(*C*)

*i = 1*

*While* (*C ≠ ∅*)

*c = first*(*C*)

*si =* {*c*}

*While* (*true*)

*If c = last*(*C*) *then*

*S = S si*

*C = C / S*

*break*

*Else*

*c = next*(*C*)

*b = bitset*(*si*) *AND bitset*(*c*)

*If count*(*b*) *≥ min\_sup then*

*si = si* {*c*}

*End If*

*End If*

*End While*

*i = i +1*

*End While*

Mặc dù giải thuật Roller có thể bỏ sót vài tập phổ biến nhưng nó chạy nhanh hơn giải thuật khai thác tập phổ biến truyền thống. Mục tiếp theo sẽ trình bày cách cải tiến giải thuật Roller bằng kỹ thuật khai khoáng nhị phân.

**3.1. Khai khoáng nhị phân**

Khi ma trận đánh giá biến đổi thành ma trận đánh giá nhị phân, mặt hàng và tập phổ biến trở thành cụm (mặt hàng con) và tập bit (xem mục 2). Độ hỗ trợ của mặt hàng và tập phổ biến là số lượng bit *1* trong tập bit. Bảng biểu sau mô tả ma trận đánh giá nhị phân.

|  |  |  |  |
| --- | --- | --- | --- |
|  | User 1 | User 2 | User 3 |
| Item\_1\_1 | *0* | *0* | *1* |
| Item\_1\_3 | *1* | *1* | *0* |
| Item\_2\_5 | *1* | *1* | *1* |
| Item\_3\_2 | *1* | *1* | *0* |
| Item\_3\_4 | *0* | *0* | *1* |
| Item\_4\_1 | *1* | *1* | *0* |

**Bảng 4**. Ma trận đánh giá nhị phân

|  |  |  |  |
| --- | --- | --- | --- |
|  | User 1 | User 2 | User 3 |
| Item\_2\_5 | *1* | *1* | *1* |
| Item\_1\_3 | *1* | *1* | *0* |
| Item\_3\_2 | *1* | *1* | *0* |
| Item\_4\_1 | *1* | *1* | *0* |

**Bảng 5**. Mặt hàng phổ biến thỏa ngưỡng hỗ trợ dưới (là 2)

Bước *1* sắp xếp các mặt hàng con theo độ hỗ trợ giảm dần và loại bỏ mặt hàng nào không thỏa ngưỡng hỗ trợ dưới. Giả sử với ngưỡng hỗ trợ là 2, ta có các mặt hàng con (cụm): *Item\_2\_5 =* (*111*)*, Item\_1\_3 =* (*110*)*, Item\_3\_2 =* (*110*)*, Item\_4\_1 =* (*110*).

Bước *2* khởi tạo tập phổ biến *s1* với mặt hàng con đầu tiên *Item\_2\_5* trong danh sách và có độ hỗ trợ là *3*.

*s1 =* (*1111*) *và support*(*s1*) *= count*(*111*) *= 3*

Lưu ý, *support*(…) biểu diễn độ hỗ trợ của tập phổ biến và *count*(…) đếm số bit *1* trong tập bit.

Bước *3* và bước *4* lần lượt duyệt các mặt hàng con *Item\_1\_3, Item\_3\_2, Item\_4\_1*.

* Duyệt *Item\_1\_3*: *s1 = s1 ⋃* {*Item\_1\_3*} *=* (*111*) *AND* (*110*) *=* (*110*) *→ support*(*s1*) *= 2*.
* Duyệt *Item\_3\_2*: *s1 = s1 ⋃* {*Item\_3\_2*} *=* (*110*) *AND* (*110*) *=* (*110*) *→ support*(*s1*) *= 2*.
* Duyệt *Item\_4\_1*: s1 = *s1 ⋃* {*Item\_4\_1*} *=* (*110*) *AND* (*110*) *=* (*110*) *→ support*(*s1*) *= 2*.

Tập phổ biến sau cùng *s1 =* (*100*) bao gồm các mặt hàng *Item\_2\_5, Item\_1\_3, Item\_3\_2, Item\_4\_1*. Tập bit của tập phổ biến được tích lũy bởi từng mặt hàng qua mỗi lần lặp. Điều này làm thuật toán chạy nhanh hơn và chi phí cho toán tử nhị phân cùng đếm số bit *1* là không đáng kể.

**3.2. Cải tiến giải thuật Roller**

Giải thuật Roller có thể bỏ sót một số tập phổ biến trong trường hợp những mặt hàng không có độ hỗ trợ cao và chúng ở cuối danh sách. Mặc dù chúng không có nhiều cơ hội gia nhập vào một tập phổ biến nhưng chúng thật sự có khả năng kết thành tập phổ biến hoặc đóng góp vào một tập phổ biến nào đó, hay nói cách khác, chúng không làm giảm đi nhiều độ hỗ trợ của một tập phổ biến nếu như thêm chúng vào tập phổ biến đó. Tuy nhiên rất khó phát hiện các mặt hàng dạng này cùng với tính hữu ích của chúng. Để khắc phục nhược điểm này, giải thuật Roller cần cải tiến để không bỏ sót những mặt hàng như thế.

Theo đó, trong bước 3, thay vì chọn mặt hàng kế tiếp là mặt hàng hiện hành, chúng ta sẽ chọn mặt hàng có độ hỗ trợ *giả cực đại* (pseudo-maximum) làm mặt hàng hiện hành. Khái niệm *giả cực đại* định nghĩa sau. Giải thuật Roller cải tiến như sau:

* *Bước 1*: tính độ hỗ trợ tất cả các mặt hàng và sắp xếp chúng trên bức tường theo một danh sách ứng với độ hỗ trợ giảm dần, tiêu chuẩn loại trừ là những mặt hàng không thỏa ngưỡng hỗ trợ dưới *min\_sup*. Nói cách khác, bức tường chỉ gồm các mặt hàng thỏa ngưỡng hỗ trợ dưới, được gọi là mặt hàng phổ biến (frequent items) và mặt hàng đầu tiên trong danh sách là mặt hàng có độ hỗ trợ lớn nhất.
* *Bước 2*: Tập phổ biến thứ *i* (i = ) được khởi tạo là mặt hàng đầu tiên trong danh sách và có độ hỗ trợ cũng bằng độ hỗ trợ của mặt hàng đầu tiên. Mặt hàng đầu tiên này trở thành mặt hàng hiện hành và sẽ được gỡ khỏi danh sách.
* *Bước 3*: Nếu không còn mặt hàng nào trong danh sách, giải thuật sẽ dừng. Ngược lại:
  + Nếu mặt hàng hiện hành là mặt hàng cuối cùng trong danh sách thì tất cả mặt hàng trong tập phổ biến thứ *i* sẽ bị gỡ khỏi danh sách và *i* được tăng lên *1* (*i = i + 1*). Trở lại bước 2.
  + Nếu mặt hàng hiện hành KHÔNG phải mặt hàng cuối cùng trong danh sách thì mặt hàng hiện hành là mặt hàng khi kết hợp với tập phổ biến thứ *i* sẽ tạo thành một tập phổ biến mới có độ hỗ trợ lớn nhất. Mặt hàng này được gọi là mặt hàng có độ hỗ *trợ giả cực đại* (pseudo-maximum).
* Bước 4:
  + Kiểm tra độ hỗ trợ của mặt hàng hiện hành, nếu thỏa ngưỡng hỗ trợ dưới thì độ hỗ trợ của tập phổ biến thứ *i*, ký hiệu *support*(*itemset i*) được cộng tích lũy với mặt hàng hiện hành, chính là tổng tất cả giao dịch chứa cả tập phổ biến thứ i và mặt hàng hiện hành. Nếu *support*(*itemset i*) lớn hơn hoặc bằng ngưỡng hỗ trợ dưới thì mặt hàng hiện hành được thêm vào tập phổ biến thứ *i*.
  + Trở lại bước 3.

Sau đây là mã giả tựa C cho giải thuật Roller cải tiến được lập trình như hàm *mining\_frequent\_itemsets*.

*C = filter\_minimum\_support*(*I*)

*C = sort*(*C*)

*i = 1*

*While* (*true*)

*c = first*(*C*)

*si =* {*c*}

*C = C /* {*c*}

*S = S si*

*While* (*true*)

*c = null*

*pseudo\_maximum = -1*

*For each item d C*

*temp = bitset*(*S*) *AND bitset*(*d*)

*If count*(*temp*) *≥ min\_sup && count*(*temp*)*> pseudo\_maximum then*

*c = d*

*pseudo\_maximum = count*(*temp*)

*End If*

*End For*

*If c = null then*

*break*

*Else*

*si = si* {*c*}

*C = C /* {*c*}

*End If*

*End While*

*i = i + 1*

*End While*

Giải thuật Roller cải tiến chậm hơn một chút so với Roller thông thường do phải tìm những mặt hàng giả cực đại trong bước *3* nhưng phát hiện nhiều tập phổ biến hơn, vì thế đạt được độ chính xác cao hơn.

**4. Đánh giá**

Chúng tôi sử dụng cơ sở dữ liệu Movielen [1] gồm 100.000 đánh giá của 943 người dùng trên 1682 mặt hàng. Cơ sở dữ liệu được chia thành 5 nhóm (folder), mỗi nhóm chứa 80% tập huấn luyện (training set) và 20% tập kiểm thử (testing set). Tập huấn luyện độc lập tập kiểm thử trong cùng một bộ.

Thiết lập hệ thống phần cứng và phần mềm bao gồm: bộ xử lý Dual-Core CPU E5700 @ 3.00GHz, RAM 2GB, RAM còn trống 1GB, Microsoft Windows 7 Ultimate 2009 32-bit, Java 7 HotSpot (TM) Client VM. Giải thuật đề xuất được so sánh với bốn giải thuật khác: Green Fall là giải thuật khuyến nghị dựa trên khai thác tập phổ biến, láng giềng gần nhất dựa trên mặt hàng (item-based), láng giềng gần nhất dựa trên người dùng (user-based) và SVD [5].

Đánh giá sử dụng 7 độ đo: *MAE, MSE, RMSE, độ chuẩn xác, độ bao phủ, F1, ARHR* [11] and *thời gian xử lý*. Thời gian xử lý được tính bằng giây. MAE và MSE đo lường độ chính xác giữa giá trị dự đoán (ước lượng) và giá trị đánh giá thực tế. MAE và MSE càng nhỏ thì độ chính xác càng cao. Độ chuẩn xác, độ bao phủ, F1 đo lường chất lượng khuyến nghị, nghĩa là đo lường mức độ yêu thích của người dùng đối với những mặt hàng được khuyến nghị, nói cách khác, những mặt hàng mà hệ thống khuyến nghị cho người dùng phản ánh bao nhiêu phần trăm sở thích và mối quan tâm của người dùng. ARHR cũng là độ đo chất lượng khuyến nghị theo một khía cạnh khác – đo lường mức độ phù hợp giữa thứ tự các mặt hàng được khuyến nghị với thứ tự các mặt hàng mà người dùng đánh giá. Các độ đo chất lượng càng lớn, giải thuật càng tốt. Sau đây là kết quả đánh giá:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Our method** | **Item-based** | **User-based** | **SVD** |
| MAE | *0.8529* | *0.5222* | *0.9319* | *0.5363* |
| MSE | *1.6304* | *0.6675* | *2.1664* | *1.1734* |
| Độ chuẩn xác | *0.1058* | *0.0245* | *0.0014* | *0.0041* |
| Độ bao phủ | *0.0404* | *0.0092* | *0.0005* | *0.0015* |
| F1 | *0.0576* | *0.0131* | *0.0008* | *0.0021* |
| ARHR | *0.0341* | *0.0040* | *0.0003* | *0.0012* |
| Thời gian xử lý | *0.0050* | *8.5607* | *7.5558* | *0.0207* |

**Bảng 5.1**. Kết quả đánh giá

Giải thuật chúng tôi hiệu quả hơn nhiều so với các giải thuật khác qua các độ đo chất lượng như *độ chuẩn xác, độ bao phủ, F1, ARHR* mặc dù độ chính xác thấp hơn item-based, user-based, SVD qua các độ đo MAE, MSE. Điểm xuất sắc nhất là giải thuật chạy rất nhanh gấp 4 lần giải thuật chạy nhanh nhất SVD.

**5. Kết luận**

Giải thuật chúng tôi khác với những giải thuật CF mô thức khác qua việc phát hiện sở thích người dùng. Tập phổ biến khai thác từ cơ sở dữ liệu được xem như mẫu quan tâm hay sở thích người dùng. Chúng tôi sử dụng một kỹ thuật khai khoáng mới, gọi là Roller, để trích xuất tập phổ biến để thay thế phương pháp khai khoáng truyền thống vốn tiêu tốn nhiều thời gian và tài nguyên máy tính. Qua đánh giá, giải thuật Roller được chứng minh ưu việt với hiệu năng cao, tốc độ nhanh, độ chuẩn xác và độ bao phủ đều cao, tiêu tốn ít tài nguyên máy tính. Nhược điểm duy nhất là Roller có thể bỏ sót vài mẫu sở thích người dùng do giả định thông minh (heuristic). Tuy nhiên kỹ thuật khai khoáng nhị phân và khái niệm độ hỗ trợ *giả cực đại* đã khắc phục nhược điểm trên.

Chúng tôi dự định phát triển một giải thuật CF mô thức mới dựa trên suy diễn mạng Bayesian để phát hiện sở thích người dùng. Đây là phương pháp thống kê sẽ được so sánh với phương pháp khai khoáng trong nghiên cứu này để chúng ta có một góc nhìn mới về lĩnh vực khuyến nghị.

**Lời cảm ơn**

Cảm ơn bà Đỗ Thị Minh Phụng – Trường Đại học Công nghệ Thông tin, Đại học Quốc Gia Thành phố Hồ Chí Minh, Việt Nam và ông Vũ Ngọc Đồng đã giúp tôi hoàn thiện và cải tiến nghiên cứu này bằng những góp ý và lời khuyên bổ ích.

**Reference**

1. [Su, Khoshgoftaar 2009]. Xiaoyuan Su and Taghi M. Khoshgoftaar. A Survey of Collaborative Filtering Techniques. Hindawi Publishing Corporation, Advances in Artificial Intelligence, Volume 2009, Article ID 421425, 19 pages, doi:10.1155/2009/421425.
2. [Movielens dataset 2011]. Home page is http://www.movielens.org. Download dataset from http://www.grouplens.org/node/12.
3. [Herlocker, Konstan, Terveen, Riedl 2004]. Jonathan. L. Herlocker, Joseph. A. Konstan, Loren G. Terveen, and John. T. Riedl. Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems, vol. 22, no. 1, pp. 5–53, 2004.
4. [Han, Kamber 2006]. Jiawei Han and Michelline Kamber. Data Mining: Concepts and Techniques. Second Edition. © 2006 by Elsevier Inc.
5. [Ricci, Rokach, Shapira, Paul 2011]. Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. Recommender Systems Handbook. ISBN: 978-0-387-85819-7, e-ISBN: 978-0-387-85820-3, DOI: 10.1007/978-0-387-85820-3. Springer New York Dordrecht Heidelberg London, Springer Science+Business Media Copyright, LLC 2011.
6. [Ungar, Foster 1998]. L. H. Ungar and D. P. Foster. Clustering Methods for Collaborative Filtering. Proceedings of the Workshop on Recommendation Systems, AAAI Press, 1998.
7. [Breese, Heckerman, Kadie 1998]. John S. Breese, David Heckerman and Carl Kadie. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. Technical Report MSR-TR-1998. Microsoft Research, Microsoft Corporation, One Microsoft Way, Redmon, WA 98052.
8. [Hofmann 2004]. Thomas Hofmann. Latent Semantic Models for Collaborative Filtering. ACM Transactionson Information Systems, Vol.22, No.1, January 2004, Pages 89-115.
9. [Shani, Heckerman, Brafman 2005]. G. Shani, D. Heckerman, and R. I. Brafman. An MDP-based Recommender System. Journal of Machine Learning Research, vol. 6, pp. 1265-1295, 2005.

**Một cách tiếp cận mới cho lọc cộng tác dựa trên suy diễn mạng Bayesian**

Nguyễn Phước Lộc

Công ty TNHH MTV Lập trình Hướng Dương, Tp. Hồ Chí Minh city, Việt Nam

**Tóm tắt**

Lọc cộng tác (collaborative filtering – CF) là một trong những giải thuật khuyến nghị phổ biến, mỗi mặt hàng giới thiệu cho một người dùng dựa vào kết quả khảo sát các đánh giá từ cộng đồng nhiều người dùng. Có hai cách tiếp cận cho CF: nạp ức (memory-based) và mô thức (model-based).

Cách tiếp cận mô thức vượt trội hơn với khả năng đáp ứng thời gian thực do tận dụng cơ chế suy diễn cho tác vụ khuyến nghị. Tuy nhiên vấn đề dữ liệu thưa và dữ liệu không đầy đủ vẫn còn là nghiên cứu mở cần được cải tiến để cơ chế suy diễn ngày càng chính xác hơn với tốc độ cao hơn.

Chúng tôi đề xuất một giải thuật CF mô thức mới dựa trên mạng Bayesian khi mô hình suy diễn mạng Bayesian được xác nhận là ưu việt do cơ chế suy diễn dựa trên bằng chứng – rất thích hợp đối với dữ liệu đánh giá. Do chất lượng mạng Bayesian phụ thuộc vào sự đầy đủ của dữ liệu huấn luyện, chất lượng kém đi nếu dữ liệu chứa nhiều giá trị bị thiếu. Chúng tôi đề xuất hai phương thức để ước lượng giá trị bị thiếu là giải thuật trung bình và giải thuật cực đại hóa kỳ vọng (EM).

**1. Giới thiệu**

Hệ thống khuyến nghị (recommendation system) là hệ thống giới thiệu những mặt hàng mà người dùng có khả năng yêu thích nhất trong một cơ sở dữ liệu lớn gồm rất nhiều mặt hàng. Mặt hàng có thể là sản phẩm, dịch vụ, sách, tạp chí và tất cả những gì mà người dùng với vai trò khách hàng có thể mua. Những mặt hàng được giới thiệu đó thể hiện sở thích (interests) hay mối quan tâm của người dùng.

Hệ thống khuyến nghị được xây dựng theo hai hướng: lọc nội dung (content-based filtering – CBF) và lọc cộng tác (collaborative filtering – CF) [1]:

* CBF giới thiệu một mặt hàng với người dùng nếu mặt hàng này tương tự về nội dung với những mặt hàng mà họ đã yêu thích với ngữ nghĩa rằng mặt hàng yêu thích là mặt hàng có đánh giá cao. Nội dung của một mặt hàng là những thuộc tính của nó và tập nội dung tất cả các mặt hàng tạo thành ma trận, gọi là ma trận nội dung (content matrix).
* Với tiếp cận khác, CF giới thiệu một mặt hàng với người dùng nếu những láng giềng của họ cũng thích mặt hàng đó với định nghĩa láng giềng là những người giống nhau về sở thích. Đánh giá của một người dùng lên một mặt hàng thể hiện sở thích của họ đối với mặt hàng đó. Tất cả đánh giá người dùng trên các mặt hàng tạo thành ma trận, gọi là ma trận đánh giá (rating matrix).

Cả hai giải thuật CBF và CF đều có ưu nhược điểm riêng. CBF tập trung vào nội dung mặt hàng và sở thích cá nhân người dùng nên mục tiêu của nó là khuyến nghị những mặt hàng khác nhau cho những người dùng khác nhau. Điểm mạnh của CBF là khả năng đưa ra những khuyến nghị đặc thù nhưng CBF không hướng đến cộng đồng như CF. Khi đó những mặt hàng mà người dùng có thể thích bị ẩn trong cộng đồng, không thể phát hiện được bằng CBF. Đây là nhược điểm của CBF, một nhược điểm khác, CBF có thể dự đoán sai sở thích người dùng nếu số lượng người dùng quá lớn.

Nếu nội dung mỗi mặt hàng quá lớn với nhiều thuộc tính, giải thuật CBF tiêu thụ nhiều tài nguyên máy tính và do đó thời gian xử lý sẽ chậm; ngược lại tốc độ không thành vấn đề đối với CF trong trường hợp này do CF không tập trung vào các thuộc tính mặt hàng. Thay vào đó, CF chỉ quan tâm đến đánh giá của người dùng trên mỗi mặt hàng và đây chính là điểm mạnh của CF. Tuy nhiên, mặc dù không đối mặt với vấn đề dữ liệu nội dung các mặt hàng quá lớn, nhược điểm của CF cũng nảy sinh từ đây, bởi vì CF sẽ đưa ra những khuyến nghị ngoài mong đợi trong trường hợp mà một số mặt hàng có thể phù hợp với người dùng nhưng họ chưa đánh giá chúng. Đây là vấn đề dữ liệu thưa, nghĩa là ma trận đánh giá bị thiếu nhiều giá trị hay nói cách khác có quá nhiều mặt hàng chưa được đánh giá. Hai kỹ thuật hữu ích được đề xuất để khắc phục nhược điểm này của CF:

* Kết hợp CBF và CF: kỹ thuật này gồm hai giai đoạn, giai đoạn đầu áp dụng CBF để hoàn chỉnh ma trận đánh giá, giai đoạn sau áp dụng CF để thực hiện khuyến nghị. Kỹ thuật này hữu ích khi cải thiện độ chính xác của khuyến nghị nhưng tốn nhiều thời gian cho giai đoạn đầu đóng vai trò như bước tiền xử lý. Kỹ thuật dùng cả ma trận đánh giá và ma trận nội dung
* Mô hình hóa ma trận đánh giá như một mô hình suy diễn, ước lượng giá trị bị thiếu và thực hiện khuyến nghị dựa trên mô hình suy diễn này. Giải thuật CF theo cách tiếp cận mô hình này được gọi là CF mô thức (model-based CF), còn một cách tiếp cận khác là CF nạp ức (memory-based CF). CF mô thức áp dụng các phương pháp thống kê và máy học để khai thác ma trận đánh giá.

Mặc dù cách tiếp cận mô thức không cho kết quả chính xác như cách tiếp cận kết hợp nhưng nó giải quyết vấn đề dữ liệu khổng lồ và ma trận thưa; hơn nữa nó có thể đáp ứng yêu cầu người dùng một cách tức thời qua cơ chế suy diễn nhanh nhạy. Vì vậy bài báo này tập trung vào CF mô thức với suy diễn mạng Bayesian. Đây là cách tiếp cận đầy tiềm năng bởi vì nó mở ra một hướng mới trong lĩnh vực khuyến nghị. Phần hai trong bài báo mô tả ý tưởng chính của giải thuật CF mô thức dựa trên mạng Bayesian. Phần 3 là cải tiến cho giải thuật đề xuất. Phần 4 mô tả kết quả đánh giá. Phần 5 là kết luận.

Lưu ý trong bài báo các thuật ngữ như ma trận đánh giá (rating matrix), cơ sở dữ liệu đánh giá (rating database), tập huấn luyện (training set) có cùng nghĩa. Mỗi dòng và cột trong ma trận đánh giá tương ứng với người dùng và mặt hàng, mỗi ô là giá trị đánh giá của một người dùng lên một mặt hàng cụ thể. Vì thế mỗi dòng đại diện một vector người dùng hay vector đánh giá; các vector này được xem như hồ sơ người dùng. Người dùng đang yêu cầu những mặt hàng khuyến nghị được gọi là hoạt nhân. Vector tương ứng với hoạt nhân là hoạt vector.

|  |  |  |  |
| --- | --- | --- | --- |
|  | *item1* | *item2* | *item3* |
| *user1* | r11 = 1 | r12 = 2 | r13 = 1 |
| *user2* | r21 = 2 | r22 = 1 | r23 = 2 |
| *user3* | r31 = 4 | r32 = 1 | r33 = 5 |
| *user4* | r41 = 1 | r42= 2 | r43 = ? (bị thiếu) |

**Table 1.** Ma trận đánh giá (người dùng 4 là hoạt nhân)

Giả sử  và  tương ứng là vector người dùng *i* và hoạt vector *a*, với *rij* là giá trị đánh giá của người dùng *i* lên mặt hàng *j*. Dấu hỏi (?) biểu thị giá trị bị thiếu và mục tiêu của giải thuật CF là ước lượng các giá trị bị thiếu. Ma trận đánh giá trong ví dụ trên là ma trận người dùng (user-based matrix) với mỗi dòng là một vector người dùng; ngược lại ma trận mặt hàng (item-based matrix) chứa các vector mặt hàng.

**2. Một giải thuật CF mới dựa trên mạng Bayesian**

Ý tưởng chính của CF mô thức (model-based CF) là xác lập một mô hình suy diễn tối ưu có thể đáp ứng yêu cầu khuyến nghị theo thời gian thực. Bên cạnh đó, CF mô thức cũng giải quyết các vấn đề quan trọng như *ma trận thưa* và *cừu đen*. Chúng tôi đề nghị một giải thuật CF mô thức mới dựa trên suy diễn mạng Bayesian [3] nhằm đạt độ chính xác cao và giải quyết vấn đề ma trận thưa. Nhìn chung giải thuật đề xuất tập trung vào xây dựng mạng Bayesian từ ma trận đánh giá. Mỗi nút trong mạng đại diện một mặt hàng và mỗi cạnh biểu diễn mối quan hệ phụ thuộc giữa hai nút. Bất cứ khi nào yêu cầu khuyến nghị xảy ra, cơ chế suy diễn của mạng Bayesian sẽ quyết định mặt hàng nào cần tư vấn cho người dùng, dựa theo xác suất hậu nghiệm (posterior probabilities) của mặt hàng đó. Xác suất này càng lớn, khả năng người dùng mua mặt hàng càng cao và do đó, mặt hàng này có tần suất mua cao và nên khuyến nghị nó với người dùng. Do xây dựng mạng Bayesian từ ma trận đầy đủ (complete matrix) dễ dàng và hiệu quả hơn từ ma trận thưa, chúng tôi thay thế những giá trị bị thiếu (missing values) trong ma trận thưa bằng các giá trị ước lượng (estimated value) nhằm biến ma trận thưa thành ma trận đầy đủ. Kỹ thuật ước lượng giá trị bị thiếu, sẽ thảo luận sau. Giải thuật đề xuất gồm 4 bước:

1. Chuyển vị ma trận người dùng (user-based matrix) thành ma trận mặt hàng (item-based matrix).
2. Điền những giá trị bị thiếu trong ma trận mặt hàng.
3. Xây dựng mạng Bayesian từ ma trận mặt hàng.
4. Thực hiện thao tác khuyến nghị.

Bước *1*, *2* và *3* được thực hiện ngoại tuyến nên không ảnh hưởng đến khả năng đáp ứng thời gian thực của tiến trình khuyến nghị trong bước *4*.

**2.1. Chuyển vị ma trận người dùng thành ma trận mặt hàng**

|  |  |  |  |
| --- | --- | --- | --- |
|  | *item1* | *item2* | *item3* |
| *user1* | r11 = 1 | r12 = 3 | r13 = **?** |
| *user2* | r21 = 3 | r22 = **?** | r23 = 5 |
| *user3* | r31 = 4 | r32 = 2 | r33 = 1 |
| *user4* | r41 = **?** | r42 = **?** | r43 = 3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *user1* | *user2* | *user3* | *user4* |
| *item1* | r11 = 1 | r21 = 3 | r31 = 4 | r41 = **?** |
| *item2* | r12 = 3 | r22 = **?** | r32 = 2 | r42 = **?** |
| *item3* | r13 = **?** | r23 = 5 | r33 = 1 | r43 = 3 |

**Bảng 2.** Chuyển vị ma trận người dùng thành ma trận mặt hàng

Ma trận người dùng là dạng ban đầu của ma trận đánh giá. Mỗi dòng trong ma trận người dùng là tập đánh giá của một người dùng cụ thể lên nhiều mặt hàng. Ngược lại, đối với ma trận mặt hàng, mỗi dòng là tập đánh giá từ nhiều người dùng lên một mặt hàng cụ thể. Bước này thực hiện việc chuyển vị từ ma trận người dùng thành ma trận mặt hàng. Đây là bước tiền xử lý, tuy đơn giản nhưng rất quan trọng do mạng Bayesian được cấu thành từ các nút mặt hàng. Do số lượng người dùng không giới hạn và gia tăng rất nhiều so với mặt hàng trong ứng dụng thực tế, chúng ta sử dụng ma trận mặt hàng để giữ kích thước mạng Bayesian luôn nhỏ nhằm nhằm cải thiện tốc độ suy diễn đối với tác vụ khuyến nghị (xem bước 4). Bảng 2 là một ví dụ chuyển vị từ ma trận người dùng thành ma trận mặt hàng.

**2.2. Điền những giá trị bị thiếu trong ma trận mặt hàng**

Mạng Bayesian học từ ma trận đánh giá đầy đủ (complete rating matrix) có chất lượng tốt hơn học từ ma trận thưa. Cách đơn giản nhất để hoàn chỉnh dữ liệu không đầy đủ (incomplete data) là thay thế những giá trị bị thiếu bằng các giá trị trung bình (average values). Giá trị trung bình chính là ước lượng (estimate) của giá trị bị thiếu, hay còn gọi giá trị ước lượng. Tiến trình thay thế, còn được gọi tiến trình ước lượng, mang tính chất lặp (iterative) và chồng (overlap):

* Tiến trình thay thế được thực hiện qua nhiều lần lặp. Việc thay thế giá trị bị thiếu tại một lần lặp dựa trên những giá trị được ước lượng trong lần lặp trước đó.
* Giá trị ước lượng trước tiên được xác định như là trung bình (mean) vector người dùng (user vector). Nếu vector người dùng rỗng thì trung bình vector mặt hàng trở thành giá trị ước lượng.

Ví dụ, giả sử với ma trận mặt hàng trong bước *1*, *r41* và *r42* được thay thế như sau:

* *r41 = r43 / 1 = 3*
* *r42 =* (*r41 + r43*) */ 2 =* (*3 + 3*) */ 2 = 3*

Giá trị *r42* được tính theo giá trị thay thế của *r41*, đây là kỹ thuật chồng (overlapping). Bằng cách khác, giá trị *r42* có thể bằng với *r41*, cùng là trung bình (mean) của vector người dùng *user4*. Nếu vậy, giải thuật nhanh hơn do trung bình vector chỉ tính một lần. Tuy nhiên, kỹ thuật chồng hiệu quả và vững chắc hơn. Nếu vector *user4* rỗng thì trung bình của nó không xác định nhưng tiến trình thay thế không gián đoạn. Theo đó, *r42* được gán bằng trung bình (mean) của vector mặt hàng *item2*, ta có *r42 =* (*r12 + r32*) */ 2 = 2.5*. Những giá trị bị thiếu còn lại như *r22*, *r13* được ước lượng như sau:

* *r22 =* (*r21 + r23*) */ 2 =* (*3 + 5*) */ 2 = 4*
* *r13 =* (*r11 + r12*) */ 2 =* (*1 + 3*) */ 2 = 2*

Theo đó, ma trận mặt hàng được ước lượng một cách đầy đủ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *user1* | *user2* | *user3* | *user4* |
| *item1* | 1 | 3 | 4 | 3 |
| *item2* | 3 | 4 | 2 | 3 |
| *item3* | 2 | 5 | 1 | 3 |

**Bảng 3.** Ma trận mặt hàng đầy đủ

Kỹ thuật trung bình nhanh như không chính xác bởi vì các giá trị thay thế chưa phản ảnh đánh giá thật sự của người dùng trên một mặt hàng. Một cách tiếp cận khác để điền giá trị bị thiếu là áp dụng thuật toán cực đại hóa kỳ vọng (expectation maximization – EM). Thuật toán EM là tiến trình lặp nhằm ước lượng giá trị bị thiếu trong dữ liệu ẩn hay không đầy đủ. Mỗi lần lặp của giải thuật EM gồm 2 bước:

* Tại bước kỳ vọng (bước E): dữ liệu bị thiếu được ước lượng trên dữ liệu quan sát và giá trị ước lượng các tham số thống kê. Tham số thống kê thường là kỳ vọng và phương sai.
* Tại bước cực đại hóa (bước M): Tham số thống kê được ước lượng bằng cách cực đại hóa hàm khả năng với giả định rằng đã biết dữ liệu bị thiếu.

Giải thuật EM chính xác hơn giải thuật trung bình do thuật toán EM áp dụng mô hình thống kê vào dữ liệu đánh giá nhằm ước lượng giá trị bị thiếu với một tiến trình lặp mà theo đó, tham số thống kê sau mỗi lần lặp được cải thiện.

**2.3. Xây dựng mạng Bayesian từ ma trận mặt hàng**

Xây dựng mạng Bayesian từ ma trận mặt hàng đầy đủ được tạo ở bước 1 và 2. Mỗi nút trong mạng có năm giá trị {*1, 2, 3, 4, 5*} tương ứng với đánh giá người dùng: *5-*thích nhất và *1-*không thích nhất. Mỗi nút đi kèm một bảng xác suất điều kiện (conditional probability table – CPT) xác định những xác suất tiên nghiệm.

Mạng Bayesian được xây dựng bằng phương pháp máy học, có hai cách tiếp cận:

* Cách tiếp cận điểm tiêu chuẩn (scored-based): giả sử mỗi mạng Bayesian được gán một điểm theo tiêu chuẩn δ, mạng nào có điểm δ cao nhất là mạng tốt nhất. Tiêu chuẩn δ là xác suất hậu nghiệm trên toàn mạng Bayesian với một bộ dữ liệu huấn luyện là ma trận mặt hàng.
* Cách tiếp cận dựa trên các ràng buộc (constraint-based): giả sử có một tập các ràng buộc (constraint), mạng Bayesian tốt nhất sẽ thỏa mãn tất cả các ràng buộc. Các ràng buộc được định nghĩa như các luật liên quan đến điều kiện Markov.

Ý tưởng chính của các phương pháp máy học là tìm ra cấu trúc mạng Bayesian phù hợp nhất. Mỗi phương pháp có nhiều giải thuật không đề cập ở đây vì chúng vượt ra ngoài phạm vi bài báo. Nhìn chung học mạng Bayesian quan trọng nhất trong giải thuật chúng tôi bởi vì cấu trúc mạng Bayesian ảnh hưởng lớn nhất đến kết quả khuyến nghị. Bây giờ chúng ta có một ví dụ đơn giản cho việc học mạng Bayesian bằng cách áp dụng phương pháp điểm tiêu chuẩn vào ma trận mặt hàng đầy đủ (xem bảng 3). Để thuận tiện, chúng ta chuyển ma trận mặt hàng trong bảng 3 thành ma trận nhị phân với mỗi ô được gán *1* nếu giá trị đánh giá tương ứng lớn hơn hay bằng *3*.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *user1* | *user2* | *user3* | *user4* |
| *item1* | 0 | 1 | 1 | 1 |
| *item2* | 1 | 1 | 0 | 1 |
| *item3* | 0 | 1 | 0 | 1 |

**Bảng 4.** Ma trận mặt hàng nhị phân

Mạng Bayesian được học từ ma trận nhị phân trong bảng *4*. Đặt *I1*, *I2* và *I3* đại diện các mặt hàng *1*, *2* và *3* tương ứng. *I1*, *I2* và *I3* là các biến (nút) nhị phân nhận hai giá trị *0* và *1*. Phương pháp điểm tiêu chuẩn sẽ tìm ra mạng Bayesian tốt nhất trong không gian tìm kiếm theo tiêu chuẩn δ. Không gian tìm kiếm là tập các mạng Bayesian, ở đây gồm 3 mạng được ký hiệu *BN1*, *BN2* và *BN3*.

*BN1*

*BN3*

*BN2*

**Hình 1.** Không gian tìm kiếm mạng Bayesian: *BN1, BN2* và *BN3*

Gọi *D* là ma trận nhị phân trong bảng *4*, bảng xác suất điều kiện của mỗi nút được tính theo công thức sau:

Với *a* và *b* là các giá trị nhị phân, chúng ta có các kết quả sau:

* *P*(*I1 = 1*) *= 3/4, P*(*I1 = 0*) *= 1/4*
* *P*(*I2 = 1*) *= 3/4, P*(*I2 = 0*) *= 1/4*
* *P*(*I3 = 1*) *= 1/2, P*(*I3 = 0*) *= 1/2*
* *P*(*I2 = 1| I1 = 1*) *= 2/3, P*(*I2 = 0| I1 = 1*) *= 1/3, P*(*I2 = 1|I1 = 0*) *= 1, P*(*I2 = 0|I1 = 0*) *= 0*
* *P*(*I3 = 1| I1 = 1*) *= 2/3, P*(*I3 = 0| I1 = 1*) *= 1/3, P*(*I3 = 1|I1 = 0*) *= 0, P*(*I3 = 0|I1 = 0*) *= 1*
* *P*(*I1 = 1| I2 = 1*) *= 2/3, P*(*I1 = 0| I2 = 1*) *= 1/3, P*(*I1 = 1|I2 = 0*) *= 1, P*(*I1 = 0|I2 = 0*) *= 0*
* *P*(*I3 = 1| I2 = 1*) *= 2/3, P*(*I3 = 0| I2 = 1*) *= 1/3, P*(*I3 = 1|I2 = 0*) *= 0, P*(*I3 = 0|I2 = 0*) *= 1*
* *P*(*I1 = 1| I3 = 1*) *= 1, P*(*I1 = 0| I3 = 1*) *= 0, P*(*I1 = 1|I3 = 0*) *= 1/2, P*(*I1 = 0|I3 = 0*) *= 1/2*
* *P*(*I2 = 1| I3 = 1*) *= 1, P*(*I2 = 0| I3 = 1*) *= 0, P*(*I2 = 1|I3 = 0*) *= 1/2, P*(*I2 = 0|I3 = 0*) *= 1/2*

Gọi *g1, g2* và *g3* lần lượt là các phân bố xác suất hợp của các mạng *BN1, BN2* và *BN3*, ta có *g1*(*I1, I2, I3*) *= P*(*I1*)*P*(*I2|I1*)*P*(*I3|I1*) , *g2*(*I1, I2, I3*) *= P*(*I2*)*P*(*I1|I2*)*P*(*I3|I2*) và *g3*(*I1, I2, I3*) *= P*(*I3*)*P*(*I1|I3*)*P*(*I2|I3*). Gọi *δ1*, *δ2* and *δ3* lần lượt là các điểm tiêu chuẩn của *BN1, BN2* và *BN3*. Những điểm tiêu chuẩn này biểu diễn xác suất hậu nghiệm của *g1, g2* và *g3* trên bộ dữ liệu ma trận nhị phân *D*. Áp dụng định lý Bayes, xác suất hậu nghiệm của mạng Bayesian *BNi* được xác định như sau:

Giả sử tất cả mạng có cùng xác suất tiên nghiệm *gi*(*I1, I2, I3*) và cùng xác suất trên toàn dữ liệu huấn luyện *gi*(*D*), khi đó *gi*(*I1, I2, I3*) và *gi*(*D*) đều là các hằng số. Không mất tính tổng quát, điểm tiêu chuẩn *δi* được định nghĩa là hàm khả năng *g*(*D|I1, I2, I3*).

Nếu ta gọi mỗi cột (tương ứng với từng người dùng) của *D* là một *trường hợp* (case) trong toàn dữ liệu huấn luyện thì có *4* trường hợp: *c1 =* (*I1 = 0, I2 = 1, I3 = 0*), *c2 =* (*I1 = 1, I2 = 1, I3 = 1*), *c3 =* (*I1 = 1, I2 = 0, I3 = 0*) và *c4 =* (*I1 = 1, I2 = 1, I3 = 1*). Giả sử các trường hợp độc lập lẫn nhau, điểm tiêu chuẩn là tích các xác suất hậu nghiệm trên tất cả trường hợp.

Cụ thể, đối với mạng *BN1* và phân bố hợp *g1*(*I1, I2, I3*) *= P*(*I1*)*P*(*I2|I1*)*P*(*I3|I1*), điểm tiêu chuẩn của mạng *BN1* là:

Tương tự, điểm tiêu chuẩn của mạng *BN2* và *BN3* là:

Dễ dàng nhận thấy điểm tiêu chuẩn *δ3* = 0.0039 lớn nhất, nên mạng *BN3* được chọn là mạng Bayesian tốt nhất được học từ ma trận mặt hàng.

*P*(*I3=1*) = 1/2

*P*(*I3=0*) = 1/2

*I3=1 I3=0*

*I2 =1* 1 1/2

*I2 =0* 0 1/2

*I3=1 I3=0*

*I1 =1* 1 1/2

*I1 =0* 0 1/2

**Hình 2.** Mạng Bayesian tốt nhất cùng các bảng xác suất điều kiện được học từ ma trận mặt hàng

**2.4. Thực hiện thao tác khuyến nghị**

Thao tác khuyến nghị được thực hiện trên cơ chế suy diễn dựa trên bằng chứng của mạng Bayesian. Theo đó, xác suất hậu nghiệm (posterior probabilities, ký hiệu *PoP*) của từng nút trong mạng được tính trước tiên và sau đó, nút mặt hàng nào có *PoP* cao nhất sẽ được khuyến nghị cho người dùng. Toàn mạng Bayesian là mẫu mua sắm của người dùng và những mặt hàng mà họ đánh giá trở thành những nút bằng chứng trong mạng. Phương thức khuyến nghị này có 2 ưu điểm:

* Việc tận dụng mạng Bayesian như là những mẫu mua sắm sẽ giúp phát hiện những sở thích cùng mối quan tâm của người dùng và do đó sẽ cải thiện chất lượng khuyến nghị.
* Suy diễn dựa trên bằng chứng trong mạng Bayesian là một cơ chế suy diễn mạnh mẽ và vững chắc. Điều này sẽ giảm thiểu trung bình lỗi khi ước lượng giá trị bị thiếu.

Giả sử người dùng *U* đã đánh giá mặt hàng *1* với giá trị *3*, hệ thống khuyến nghị sẽ chịu trách nhiệm giới thiệu mặt hàng nào cho người dùng *U*. Giả sử mạng Bayesian tốt nhất (hình 2) được chọn là mạng đích phục vụ cho thao tác khuyến nghị, khi đó *I1* trở thành nút bằng chứng nhị phân có giá trị 1 do mặt hàng *1* được đánh giá là *3*. Sau đây là mạng Bayesian đích với nút bằng chứng được tô bóng:

*P*(*I3=1*) = 1/2

*P*(*I3=0*) = 1/2

*I3=1 I3=0*

*I2 =1* 1 1/2

*I2 =0* 0 1/2

*I3=1 I3=0*

*I1 =1* 1 1/2

*I1 =0* 0 1/2

**Hình 3.** Mạng Bayesian đích với nút bằng chứng I1 = 1

Gọi *PoP*(*I2*), *PoP*(*I3*) tương ứng là xác suất hậu nghiệm của các nút *I2*, *I3* và gọi *g*(*I1, I2, I3*) *= P*(*I3*)*P*(*I1|I3*)*P*(*I2|I3*) là phân bố xác suất hợp của mạng Bayesian đích, ta có:

Khai triển *PoP*(*I2*) và *PoP*(*I3*) ta có:

Dễ dàng tính phân bố xác suất hợp *g*(*I1, I2, I3*) *= P*(*I3*)*P*(*I1|I3*)*P*(*I2|I3*) khi giá trị các biến *I1*, *I2* và *I3* được xác định bằng cách áp dụng các bảng xác suất điều kiện của mạng Bayesian đích. Vì thế, các xác suất hậu nghiệm *PoP*(*I2*) và *PoP*(*I3*) có giá trị cụ thể như sau:

Do xác suất hậu nghiệm *PoP*(*I3*) lớn nhất nên mặt hàng 3 được khuyến nghị cho người dùng với mức ưu tiên cao nhất. Mạng Bayesian đích trong ví dụ này rất nhỏ chỉ với 3 nút nên độ phức tạp tính toán không đáng kể và không ảnh hưởng đến khả năng đáp ứng thời gian thực của hệ thống khuyến nghị nhưng khi kích thước mạng trở nên khổng lồ với rất nhiều nút thì độ phức tạp tính toán trở thành vấn đề quan trọng cần được giải quyết. Phần tiếp theo đề cập đến một kỹ thuật cải tiến nhằm khắc phục vấn đề độ phức tạp tính toán trên.

**3. Một cải tiến giải thuật đề xuất – mạng Bayesian phân cụm**

Mặt dầu số lượng mặt hàng có giới hạn và không tăng nhiều như số lượng người dùng nhưng nó vẫn còn lớn. Hiển nhiên mạng Bayesian hợp thành từ những nút mặt hàng tương quan với nhau nhưng có thể chứa những nút không tương quan do nó được học từ dữ liệu lớn như vậy. Những nút không tương quan này khiến khả năng suy diễn thiếu chính xác. Đây là vấn đề rời rạc hay bất tương quan giữa các nút trong mạng Bayesian, cần được giải quyết.

Giả sử một cửa hàng gồm nhiều mặt hàng chia thành các loại. Áo thun, áo sơ mi, quần dài, quần bò… cùng một loại tương liên nhau là loại quần áo. Các mặt hàng quần áo, một cách tự nhiên, tạo thành một nhóm nút con liên kết nhau trong mạng Bayesian. Nhưng những mặt hàng khác, không liên quan đến quần áo, trở thành những nút rời rạc không liên kết với bất cứ mặt hàng quần áo nào. Cơ chế suy diễn trong mạng Bayesian nếu có những nút rời rạc sẽ dẫn đến sự thiếu chính xác. Vấn đề này được giải quyết khi xây dựng nhiều mạng Bayesian con, mỗi mạng con ứng với một nhóm mặt hàng tương liên. Nói cách khác, toàn bộ mạng Bayesian lớn được phân rã thành nhiều mạng Bayesian cá thể (individual Bayesian network) sao cho mối quan hệ giữa các nút trong mạng cá thể chặt chẽ hơn. Ví dụ, chúng ta xây dựng ba mạng Bayesian ứng với ba nhóm mặt hàng trong siêu thị: quần áo, đồ nội thất và điện gia dụng.

Trong trường hợp dữ liệu huấn luyện như ma trận đánh giá không đặc tả rõ ràng các nhóm mặt hàng, kỹ thuật gom cụm được áp dụng để phân nhóm mặt hàng. Theo đó, có hai bước cải tiến việc xây dựng mạng Bayesian:

* Áp dụng các phương pháp gom cụm như *K-mean, K-centroid* để phân nhóm mặt hàng. Chúng ta có thể phân nhóm một cách thủ công, khi đó một mặt hàng có thể thuộc nhiều hơn một nhóm.
* Đối với mỗi nhóm mặt hàng:
  + Dữ liệu huấn luyện được thu gọn. Cụ thể là, đối với mỗi dòng của ma trận đánh giá, những cột không tương ứng với các mặt hàng trong nhóm này sẽ bị loại bỏ.
  + Mạng Bayesian được học từ dữ liệu huấn luyện thu gọn này, được gọi là mạng Bayesian cá thể (individual Bayesian network).

Lưu ý, mỗi nút có thể thuộc về nhiều hơn một mạng cá thể, đây là một nhược điểm nhưng thường xuất hiện trong ứng dụng thương mại, một mặt hàng có thể được xếp vào nhiều nhóm.

Mỗi lần có tác vụ khuyến nghị được yêu cầu như trong bước 4, tiến trình suy diễn sẽ được thực thi trên mạng Bayesian cá thể thay vì trên toàn mạng lớn như trước đây. Tốc độ được cải thiện khi số lượng nút trong mạng cá thể nhỏ hơn rất nhiều so với toàn mạng lớn. Nhưng một vấn đề khác đặt ra là làm thế nào chọn đúng mạng cá thể để thực hiện tác vụ khuyến nghị khi mà rất tốn thời gian và tài nguyên máy tính nếu chúng ta duyệt qua tất cả các mạng cá thể cùng với nút của chúng để tìm đúng mạng thích hợp chứa nhiều nhất các mặt hàng yêu thích của người dùng. Vì thế, cần có một cách tiếp cận khác. Đây là một nghiên cứu mở nhưng chúng tôi đề xuất một giải pháp được gọi là *bảng ánh xạ* (mapping table – MT).

Ý tưởng chính của kỹ thuật bảng ánh xạ là tạo ra một bảng ánh xạ (mapping table) tại thời điểm xây dựng mạng Bayesian (học mạng). Mỗi dòng của bảng ánh xạ là một cặp khóa-giá trị. Khóa là tên nút mặt hàng, giá trị là tập bit nhị phân xác định nút nào thuộc về mạng cá thể nào. Mỗi bit của tập bit này biễu diễn sự xuất hiện của một mạng cá thể hay nói cách khác mạng cá thể này có chứa hay không chứa nút được đặc tả bởi khóa. Giả sử có 3 mạng cá thể *BN0, BN1, BN2* và 6 nút *A, B, C, D, E, F, G, H*, bảng ánh xạ được mô tả như sau:

|  |  |
| --- | --- |
| A | 100 |
| B | 100 |
| C | 010 |
| D | 010 |
| E | 001 |
| F | 001 |
| G | 011 |
| H | 011 |

**Bảng 5.** Bảng ánh xạ

Bảng ánh xạ được diễn dịch như sau:

* Nút *A* và *B* thuộc mạng *BN0*.
* Nút *C* và *D* thuộc mạng *BN1*.
* Nút *E* và *F* thuộc mạng *BN2*.
* Nút *G* và *H* thuộc mạng *BN1* và *BN2* tương ứng.

Cho trước một người dùng cùng với các mặt hàng mà người đó đã đánh giá, đối với từng mạng cá thể, số lượng nút mặt hàng xuất hiện trong mạng này được đếm. Mạng nào có số nút lớn nhất là mạng thích hợp để thực hiện khuyến nghị bằng những tác vụ suy diễn. Ví dụ, giả sử các mặt hàng mà người dùng *a* đã đánh giá là *E, F, G* và *H*, ta có:

* Tổng số nút chứa trong *BN0* là *0*, *t0* = *0* do *BN0* không chứa bất kỳ nút nào người dùng *a* đánh giá.
* Tổng số nút chứa trong *BN1* là *2*, *t1* = *2* do *BN1* chứa *2* nút *G* và *H*.
* Tổng số nút chứa trong *BN2* là *4*, *t2* = *4* do *BN2* chứa *4* nút *E, F, G* và *H*.

Do *t4* lớn nhất nên *BN2* là mạng cá thể thích hợp để thực hiện suy diễn khuyến nghị.

**4. Đánh giá**

Chúng tôi sử dụng cơ sở dữ liệu Movielen [1] gồm 100.000 đánh giá của 943 người dùng trên 1682 mặt hàng. Cơ sở dữ liệu được chia thành 5 nhóm (folder), mỗi nhóm chứa 80% tập huấn luyện (training set) và 20% tập kiểm thử (testing set). Tập huấn luyện độc lập tập kiểm thử trong cùng một bộ.

Thiết lập hệ thống phần cứng và phần mềm bao gồm: bộ xử lý Dual-Core CPU E5700 @ 3.00GHz, RAM 2GB, RAM còn trống 1GB, Microsoft Windows 7 Ultimate 2009 32-bit, Java 7 HotSpot (TM) Client VM. Giải thuật đề xuất được so sánh với bốn giải thuật khác: Green Fall là giải thuật khuyến nghị dựa trên khai thác tập phổ biến, láng giềng gần nhất dựa trên mặt hàng (item-based), láng giềng gần nhất dựa trên người dùng (user-based) và SVD.

Đánh giá sử dụng 7 độ đo: *MAE, MSE, RMSE, độ chuẩn xác, độ bao phủ, F1, ARHR* [11] and *thời gian xử lý*. Thời gian xử lý được tính bằng giây. MAE và MSE đo lường độ chính xác giữa giá trị dự đoán (ước lượng) và giá trị đánh giá thực tế. MAE và MSE càng nhỏ thì độ chính xác càng cao. Độ chuẩn xác, độ bao phủ, F1 đo lường chất lượng khuyến nghị, nghĩa là đo lường mức độ yêu thích của người dùng đối với những mặt hàng được khuyến nghị, nói cách khác, những mặt hàng mà hệ thống khuyến nghị cho người dùng phản ánh bao nhiêu phần trăm sở thích và mối quan tâm của người dùng. ARHR cũng là độ đo chất lượng khuyến nghị theo một khía cạnh khác – đo lường mức độ phù hợp giữa thứ tự các mặt hàng được khuyến nghị với thứ tự các mặt hàng mà người dùng đánh giá. Các độ đo chất lượng càng lớn, giải thuật càng tốt. Sau đây là kết quả đánh giá:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Giải thuật đề xuất** | **Green Fall** | **Item-based** | **User-based** | **SVD** |
| MAE | *0.6127* | *0.8529* | *0.5222* | *0.9319* | *0.5363* |
| MSE | *0.9023* | *1.6304* | *0.6675* | *2.1664* | *1.1734* |
| Độ chính xác | *0.1334* | *0.1058* | *0.0245* | *0.0014* | *0.0041* |
| Độ bao phủ | *0.0513* | *0.0404* | *0.0092* | *0.0005* | *0.0015* |
| F1 | *0.0731* | *0.0576* | *0.0131* | *0.0008* | *0.0021* |
| ARHR | *0.0454* | *0.0341* | *0.0040* | *0.0003* | *0.0012* |
| Thời gian | *1.8169* | *0.0050* | *8.5607* | *7.5558* | *0.0207* |

**Bảng 6**. Kết quả đánh giá

Giải thuật chúng tôi hiệu quả hơn nhiều so với các giải thuật khác qua các độ đo chất lượng như *độ chuẩn xác, độ bao phủ, F1, ARHR*. Độ chính xác xấp xỉ item-based, user-based, SVD và tốt hơn Green Fall qua các độ đo MAE, MSE và RMSE. Giải thuật tốn nhiều thời gian hơn Green Fall và SVD nhưng ít thời gian hơn item-based và user-based.

**5. Kết luận**

Điểm cơ bản của giải thuật đề xuất là xây dựng mạng Bayesian từ ma trận đánh giá (rating matrix) và áp dụng suy diễn trong mạng Bayesian cho việc khuyến nghị mặt hàng. Mạng Bayesian là đồ thị không vòng gồm một tập nút và một tập cạnh. Nút đại diện mặt hàng và cạnh biểu diễn mối quan hệ phụ thuộc giữa hai nút. Bất cứ khi nào tác vụ khuyến nghị khởi hoạt, cơ chế suy diễn trong mạng Bayesian sẽ thực thi dựa vào các bằng chứng từ ma trận đánh giá, theo đó, mặt hàng nào có xác suất hậu nghiệm cao nhất sẽ được giới thiệu đến người dùng. Vì độ chính xác của tiến trình khuyến nghị phụ thuộc vào chất lượng của cấu trúc mạng Bayesian nên có hai cách để đạt được cấu trúc mạng tốt nhất:

* Chọn thuật toán học mạng thích hợp – không đề cập trong nghiên cứu này.
* Điền những giá trị bị thiếu bằng các giá trị ước lượng nhằm biến ma trận đánh giá không đầy đủ (ma trận thưa) thành ma trận đầy đủ. Mạng Bayesian học từ dữ liệu đầy đủ sẽ đạt chất lượng cao.

Giải thuật trung bình và EM được đề xuất để bổ sung và hoàn chỉnh ma trận thưa. Phương pháp trung bình nhanh và đơn giản hơn EM nhưng kém chính xác.

Giải thuật đề xuất được đánh giá và so sánh với các giải thuật nạp ức và mô thức khác như Green Fall, item-based, user-based và SVD. Kết quả cho thấy giải thuật đề xuất hiệu quả nhất khi đạt chất lượng khuyến nghị cao qua các độ đo như độ chuẩn xác, độ bao phủ, F1 và ARHR mặc dù Green Fall là giải thuật nhanh nhất đáp ứng yếu tố khuyến nghị thời gian thực. Tuy nhiên, thời gian mô hình hóa dữ liệu trong giải thuật đề xuất dài hơn Green Fall, nói cách khác, thời gian xây dựng mạng Bayesian lâu hơn thời gian khai thác tập phổ biến. Nhìn chung giải thuật khuyến nghị dựa trên mạng Bayesian tối ưu với tiêu chí chất lượng khuyến nghị và giải thuật khuyến nghị dựa trên khai thác tập phổ biến Green Fall tối ưu với tiêu chí tốc độ đáp ứng thời gian thực.

**Lời cảm ơn**

Cảm ơn bà Đỗ Thị Minh Phụng – Trường Đại học Công nghệ Thông tin, Đại học Quốc Gia Thành phố Hồ Chí Minh, Việt Nam và ông Vũ Ngọc Đồng đã giúp tôi hoàn thiện và cải tiến nghiên cứu này bằng những góp ý và lời khuyên bổ ích.

**Tham khảo**

1. [Su, Khoshgoftaar 2009]. Xiaoyuan Su and Taghi M. Khoshgoftaar. A Survey of Collaborative Filtering Techniques. Hindawi Publishing Corporation, Advances in Artificial Intelligence, Volume 2009, Article ID 421425, 19 pages, doi:10.1155/2009/421425.
2. [Movielens dataset 2011]. Home page is http://www.movielens.org. Download dataset from http://www.grouplens.org/node/12.
3. Richard E. Neapolitan. Learning Bayesian Networks. Northeastern Illinois University Chicago, Illinois 2003.
4. [Ungar, Foster 1998]. L. H. Ungar and D. P. Foster. Clustering Methods for Collaborative Filtering. Proceedings of the Workshop on Recommendation Systems, AAAI Press, 1998.
5. [Han, Kamber 2006]. Jiawei Han and Michelline Kamber. Data Mining: Concepts and Techniques. Second Edition. © 2006 by Elsevier Inc.
6. [Breese, Heckerman, Kadie 1998]. John S. Breese, David Heckerman and Carl Kadie. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. Technical Report MSR-TR-1998. Microsoft Research, Microsoft Corporation, One Microsoft Way, Redmon, WA 98052.
7. [Heckerman, Chickering, Meek, Rougnthwaite 2000]. David Heckerman, David Maxwell Chickering, Christopher Meek, Robert Rougnthwaite and Carl Kadie. Dependecy Networks for Inference, Collaborative Filtering and Data Visualization. Journal of Machine Learning Research 1 (2000) 49-75.
8. [Hofmann 2004]. Thomas Hofmann. Latent Semantic Models for Collaborative Filtering. ACM Transactionson Information Systems, Vol.22, No.1, January 2004, Pages 89-115.
9. [Shani, Heckerman, Brafman 2005]. G. Shani, D. Heckerman, and R. I. Brafman. An MDP-based Recommender System. Journal of Machine Learning Research, vol. 6, pp. 1265-1295, 2005.
10. [Rennie 2004]. Jason D. M. Rennie (jrennie@csail.mit.edu). A Short Tutorial on Using Expectation-Maximization with Mixture Models. MIT University, March 3, 2004
11. [Herlocker, Konstan, Terveen, Riedl 2004]. Jonathan. L. Herlocker, Joseph. A. Konstan, Loren G. Terveen, and John. T. Riedl. Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems, vol. 22, no. 1, pp. 5–53, 2004.