TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**KHAI THÁC TẬP DỮ LIỆU LỚN**

**BẢO VỆ HỆ THỐNG KHUYẾN NGHỊ BẰNG CÁCH XÁC ĐỊNH**

**CÁC XẾP HẠNG GIẢ**

*Người hướng dẫn*: **TS. Bùi Thanh Hùng**

*Người thực hiện*: **NGUYỄN DUY HÀN LÂM (MSSV: 196005004)**

Khoá  **: 2019**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2019**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**KHAI THÁC TẬP DỮ LIỆU LỚN**

**BẢO VỆ HỆ THỐNG KHUYẾN NGHỊ BẰNG CÁCH XÁC ĐỊNH**

**CÁC XẾP HẠNG GIẢ**

*Người hướng dẫn*: **TS. Bùi Thanh Hùng**

*Người thực hiện*: **NGUYỄN DUY HÀN LÂM (MSSV: 196005004)**

Khoá  **: 2019**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2019**

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của TS Bùi Thanh Hùng. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 28 tháng 9 năm 2016*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Nguyễn Duy Hàn Lâm*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

MỤC LỤC

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN ii](#_Toc23071040)

[MỤC LỤC 1](#_Toc23071041)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 4](#_Toc23071042)

[CHƯƠNG 1: HỆ THỐNG GỢI Ý VÀ TÍNH BẢO MẬT CỦA HỆ THỐNG 6](#_Toc23071043)

[1.1 Hiện tượng long-tail (khởi nguồn ra đời của hệ thống gợi ý) 7](#_Toc23071044)

[1.2 Giới thiệu sơ lược về hệ thống gợi ý (Recommender System) 7](#_Toc23071045)

[1.3 Content-based Recommendation System 8](#_Toc23071046)

[1.3.1 Xây dựng utility matrix 8](#_Toc23071047)

[1.3.2 Item profile 8](#_Toc23071048)

[1.3.3 Xây dựng hàm mất mát 8](#_Toc23071049)

[1.3.4 Lập trình trên Python 8](#_Toc23071050)

[1.4 Neighborhood-Based Collaborative Filtering (User-User Collabrative Filtering) 8](#_Toc23071051)

[1.4.1 Xây dựng Similarity function 8](#_Toc23071052)

[1.4.2 Rating prediction 8](#_Toc23071053)

[1.4.3 Khuyết điểm 8](#_Toc23071054)

[1.5 Neighborhood-Based Collaborative Filtering (Item-Item Collabrative Filtering) 8](#_Toc23071055)

[1.5.1 Khắc phục khuyết điểm của User-User Collaborative Filtering 8](#_Toc23071056)

[1.5.2 Lập trình Neighborhood-Based Collaborative Filtering trên Python 8](#_Toc23071057)

[1.6 Matrix Factorization Collaborative Filtering 8](#_Toc23071058)

[1.6.1 Ý tưởng thuật toán 9](#_Toc23071059)

[1.6.2 Xây dựng và tối ưu hàm mất mát 9](#_Toc23071060)

[1.6.3 Bias 9](#_Toc23071061)

[1.6.4 Non-negative Matrix Factorization 9](#_Toc23071062)

[1.6.5 Incremental Matrix Factorization 9](#_Toc23071063)

[1.6.6 Các vấn đề khác 9](#_Toc23071064)

[1.6.7 Lập trình trên Python 9](#_Toc23071065)

[1.7 Các lược đồ cho CSDL đa chiều 9](#_Toc23071066)

[1.8 Các mô hình lưu trữ hỗ trợ OLAP 10](#_Toc23071067)

[1.8.1 Mô hình Multidimentional OLAP (MOLAP) 10](#_Toc23071068)

[1.8.2 Mô hình Relational OLAP (ROLAP) 12](#_Toc23071069)

[1.8.3 Mô hình Hybird OLAP (HOLAP) 13](#_Toc23071070)

[CHƯƠNG 2: OLAP VÀ KHAI PHÁ DỮ LIỆU 15](#_Toc23071071)

[2.1 Tóm tắt các khái niệm 16](#_Toc23071072)

[2.1.1 OLTP (Online Transaction Processing) 16](#_Toc23071073)

[2.1.2 OLAP (Online Analytical Processing) 16](#_Toc23071074)

[2.1.3 Sự khác nhau giữa OLAP và OLTP 17](#_Toc23071075)

[2.1.4 Chuyển đổi dữ liệu từ OLTP tới OLAP 18](#_Toc23071076)

[2.2 Quy trình thực hiện OLAP. 19](#_Toc23071077)

[2.3 Tóm tắt mô hình hóa OLAP. 20](#_Toc23071078)

[2.4 Tóm tắt các loại thao tác trong OLAP. 20](#_Toc23071079)

[2.4.1 Slice 20](#_Toc23071080)

[2.4.2 Dice 21](#_Toc23071081)

[2.4.3 Drill Down hay Drill Up 21](#_Toc23071082)

[2.4.4 Roll up 21](#_Toc23071083)

[2.4.5 Pivot 21](#_Toc23071084)

[CHƯƠNG 3: ỨNG DỤNG OLAP VÀO BÀI TOÁN CỤ THỂ 22](#_Toc23071085)

[3.1 Mô tả bài toán 22](#_Toc23071086)

[3.2 Mô hình thực thể ban đầu (CDM Model) 23](#_Toc23071087)

[3.3 Mô hình vật lý ban đầu của bài toán 24](#_Toc23071088)

[3.4 Triển khai OLAP cho bài toán 25](#_Toc23071089)

[3.5 Áp dụng OLAP trên hệ quản trị cơ sở dữ liệu 26](#_Toc23071090)

[3.6 Áp dụng OLAP trên chương trình 32](#_Toc23071091)

[TÀI LIỆU THAM KHẢO 39](#_Toc23071092)

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

[Hình 1.1: Vị trí của OLAP 6](#_Toc482868018)

[Hình 1.2: Lược đồ hình sao 10](#_Toc482868019)

[Hình 1.3: Lược đồ bông tuyết 10](#_Toc482868020)

[Hình 1.4: Mô hình dữ liệu MOLAP 11](#_Toc482868021)

[Hình 1.5: Mô hình dữ liệu ROLAP 13](#_Toc482868022)

[Hình 2.1: Mô tả thao tác Slice[2] 21](#_Toc482868023)

[Hình 2.2: Mô tả thao tác Dice[2] 21](#_Toc482868024)

[Hình 2.3: Mô tả thao tác Drill Down/Up[2] 22](#_Toc482868025)

[Hình 2.4: Mô tả thao tác Pivot[2] 22](#_Toc482868026)

[Hình 3.1: Mô hình thực thể của bài toán 24](#_Toc482868027)

[Hình 3.2: Mô hình vật lý của bài toán 24](#_Toc482868028)

[Hình 3.3: Mô hình CDM sau khi thêm bảng sự thật 25](#_Toc482868029)

[Hình 3.4: Mô hình vật lý sau khi thêm bảng sự thật 26](#_Toc482868030)

[Hình 3.5: Dữ liệu minh họa cho bảng NHA\_CUNG\_CAP 26](#_Toc482868031)

[Hình 3.6: Dữ liệu minh họa cho bảng LOAI\_HANG 27](#_Toc482868032)

[Hình 3.7: Dữ liệu minh họa cho bảng NGUYEN\_VAT\_LIEU 27](#_Toc482868033)

[Hình 3.8: Dữ liệu minh họa cho bảng KHACH\_HANG 27](#_Toc482868034)

[Hình 3.9: Dữ liệu minh họa cho bảng sự thật SALE 28](#_Toc482868035)

[Hình 3.10: Câu lệnh Join các bảng của mô hình star schema 28](#_Toc482868036)

[Hình 3.11: Một phần kết quả của truy vấn join 28](#_Toc482868037)

[Hình 3.12: Áp dụng câu lệnh join cho truy vấn dùng OLAP 28](#_Toc482868038)

[Hình 3.13: Kết quả áp dụng kỹ thuật OLAP 29](#_Toc482868039)

[Hình 3.14: Câu lệnh group by đơn giản trên nền OLAP 29](#_Toc482868040)

[Hình 3.15: Kết quả sau khi dùng câu lệnh group by 29](#_Toc482868041)

[Hình 3.16: Ví dụ về Drill Down 29](#_Toc482868042)

[Hình 3.17: Kết quả của ví dụ về Drill Down 30](#_Toc482868043)

[Hình 3.18: Câu lệnh Slice 30](#_Toc482868044)

[Hình 3.19: Kết quả của câu lệnh Slice 30](#_Toc482868045)

[Hình 3.20: Câu lệnh thao tác Dice 31](#_Toc482868046)

[Hình 3.21: Kết quả câu lệnh Dice 31](#_Toc482868047)

[Hình 3.22: Câu lệnh Roll up 31](#_Toc482868048)

[Hình 3.23: Kết quả câu lệnh Roll Up 32](#_Toc482868049)

[Hình 3.24: Giao diện của chức năng 1 33](#_Toc482868050)

[Hình 3.25: Phương thức ở lớp Database phục vụ cho chức năng 1 33](#_Toc482868051)

[Hình 3.26: Phương thức ở lớp OLAP\_DAO phục vụ cho chức năng 1 33](#_Toc482868052)

[Hình 3.27: Phương thức ở lớp OLAP\_BUS phục vụ cho chức năng 1 34](#_Toc482868053)

[Hình 3.28: Phương thức phục vụ hiển thị kết quả của chức năng 1 34](#_Toc482868054)

[Hình 3.29: Kết quả của chức năng 1 34](#_Toc482868055)

[Hình 3.30: Phương thức ở lớp Database phục vụ cho chức năng 2 35](#_Toc482868056)

[Hình 3.31: Phương thức ở lớp OLAP\_DAO phục vụ cho chức năng 2 35](#_Toc482868057)

[Hình 3.32: Phương thức ở lớp OLAP\_BUS phục vụ cho chức năng 2 35](#_Toc482868058)

[Hình 3.33: Phương thức phục vụ hiển thị kết quả của chức năng 2 35](#_Toc482868059)

[Hình 3.34: Giao diện chung và kết quả chức năng 2 36](#_Toc482868060)

[Hình 3.35: Phương thức ở lớp Database phục vụ cho chức năng 3 36](#_Toc482868061)

[Hình 3.36: Phương thức ở lớp OLAP\_DAO phục vụ cho chức năng 3 36](#_Toc482868062)

[Hình 3.37: Phương thức ở lớp OLAP\_BUS phục vụ cho chức năng 3 37](#_Toc482868063)

[Hình 3.38: Giao diện chung và kết quả chức năng 3 37](#_Toc482868064)

[Hình 3.39: Phương thức ở lớp Database phục vụ cho chức năng 4 37](#_Toc482868065)

[Hình 3.40: Phương thức ở lớp OLAP\_DAO phục vụ cho chức năng 4 37](#_Toc482868066)

[Hình 3.41: Phương thức ở lớp OLAP\_BUS phục vụ cho chức năng 4 38](#_Toc482868067)

[Hình 3.42: Giao diện chung và kết quả chức năng 4 38](#_Toc482868068)

**DANH MỤC BẢNG BIỂU**

[Bảng 1: So sánh các mô hình OLAP 14](#_Toc482869075)

[Bảng 2: So sánh OLTP và OLAP 17](#_Toc482869076)

CHƯƠNG 1: HỆ THỐNG GỢI Ý

Ngày nay, có rất nhiều dữ liệu, thông tin có sẵn trên Internet nhưng rất khó để lọc ra những thông tin cần thiết từ nguồn dữ liệu, thông tin dồi dào này. Một trong những giải pháp để giải quyết vấn đề này, đó chính là sử dụng các thuật toán để đưa ra gợi ý giúp ta tìm được thông tin mà ta muốn, ta gọi các hệ thống đưa ra gợi ý này là “Recommender System”. Có 2 loại recommender system là: Content-based Recommendation System (Hệ thống đưa ra các gợi ý dựa trên các dữ liệu, thông tin, vật thể,…) và Collaborative filtering Recommendation System (Hệ thống đưa ra các gợi ý dựa trên hành vi, ý muốn của người dùng). Trong đó, Collaborative filtering Recommendation System được sử dụng phổ biến vì có tính khả thi cao do không cần phải phân loại nguồn đầu vào cho hệ thống (dữ liệu, thông tin, vật thể, đối tượng,…) vì đôi khi các nguồn rất khó hoặc không thể phân loại được. Collaborative filtering Recommendation System rất hiệu quả và hữu ích trong việc đưa ra các gợi ý cho người dùng, tuy niên hệ thống này có một khuyết điểm lớn, nằm ở chính nguồn đầu vào của hệ thống. Các nguồn đầu vào của hệ thống đôi khi sẽ bị sai lệch, không chính xác ảnh hưởng đến đầu ra là các gợi ý mà hệ thống đã đưa ra. Đôi khi các nguồn đầu vào này sai lệch không phải do khách quan, ngẫu nhiên mà có một số kẻ cố tình tạo dựng các nguồn dữ liệu giả nhằm tấn công vào hệ thống.

Những kẻ tấn công ẩn danh vào như những người dùng bình thường, họ tham gia vào quá trình xây dựng nguồn dữ liệu đầu vào cho các hệ thống gợi ý, tạo ra các hành vi sai, không đúng hay không hợp lý, từ đó khiến quá trình huấn luyện của hệ thống để đưa ra các gợi ý trở nên bị sai lệch. Từ đó làm giảm tính khách quan và độ chính xác của hệ thống. Điều này ảnh hưởng nghiêm trọng đến độ tin cậy của hệ thống. Tài liệu này nhằm mục đích thảo luận về các thể loại tấn công khác nhau có thể ảnh hưởng đến hệ thống recommender và các giải pháp được sử dụng để phát hiện các cuộc tấn công này.

* 1. Hiện tượng long-tail (khởi nguồn ra đời của hệ thống gợi ý)

Trong các cửa hàng hay siêu thị ta thường đa phần thấy có 2 tình trạng xảy ra. Một là có nhiều sản phẩm được rất rất nhiều người dùng mua và lựa chọn. Hai là có rất nhiều mặt hàng sản phẩm còn trên kệ, tuy nhiên chúng đôi khi vẫn có những khách hàng mua chúng. Ngoài ra ta còn thấy được đa phần số lượng các mặt hàng hay nhãn hàng được mua nhiều thì rất ít, còn các mặt hàng hay nhãn hàng được mua ít thì lại chiếm đa số trên kệ hàng.

Từ đó, ta có thể mô phỏng được hình dáng của đồ thị phân phối xác suất sẽ có 2 phần. Phần đầu là các đối tượng có xác suất xảy ra còn. Phần còn lại là một tập hợp dài các đối tượng có xác suất xảy ra thấp hơn hoặc thấp hơn rất nhiều. Ta đặt biệt danh cho tập hợp dài này là “long-tail”.

Tuy nhiên, trong kinh doanh, các chủ buôn thường không muốn điều này. Họ thường luôn muốn bán hết các mặt hàng có lượng mua ít càng nhanh càng tốt để có thể dành tài nguyên (kho, kệ hàng,…) cho các mặt hàng khác quan trọng hơn.

Để có thể đẩy nhanh được các mặt hàng có lượng kinh doanh thấp này, ta cần phải cho càng nhiều người dùng biết càng tốt vì có cơ hội họ sẽ mua chúng. Một trong những giải pháp đó chính là sử dụng hệ thống gợi ý. Hệ thống gợi ý sẽ dựa trên các đặc điểm của món hàng hay của người dùng mà từ đó có thể đưa ra các gợi ý phù hợp cho người dùng. Từ đó có thể nâng cao được cơ hội người mua sẽ chọn và mua các món hàng đó.

* 1. Giới thiệu sơ lược về hệ thống gợi ý (Recommender System)

Hệ thống gợi ý là hệ thống dựa trên tập dữ liệu đầu vào, qua quá trình huấn luyện, sẽ cho được ở đầu ra một kết quả. Kết quả này là một lời gợi ý dành cho các đối tượng trong tập đối tượng. Ví dụ ta có tập dữ liệu đầu vào là các bộ phim và tập đối tượng là những người xem phim, khi đưa vào hệ thống, sau khi huấn luyện, hệ thống sẽ đưa ra được các gợi ý là những bộ phim mà những người xem trong tập đối tượng nên xem.

Có 2 loại hệ thống gợi ý: Content-based và Collaborative filtering.

Content-based recommendation system: đánh giá các thuộc tính của item rồi từ đó phân loại các item. Dựa vào từng user mà sẽ đưa ra các gợi ý item phù hợp với ý muốn của người dùng. Ví dụ: có rất nhiều bộ phim với nhiều thể loại khác nhau như hình sự, kinh dị, trinh thám, hoạt hình, hài hước,… tùy theo sở thích của người xem ta sẽ gợi ý các bộ phim khác có thể loại tương tự với thể loại của các bộ phim mà người dùng đã xem trước đó.

Collaborative filtering recommendation system: đưa ra gợi ý cho người dùng dựa vào các khuôn mẫu hay hành vi mà người đó hành động. Ví dụ: người A thích xem phim kinh dị, bên cạnh đó người cũng thích xem phim hành động. Người B thích phim kinh dị, như vậy qua quá trình huấn luyện, hệ thống có thể gợi ý cho người B nên xem phim hành động.

* 1. Content-based Recommendation System

Ở mục này ta sẽ tìm hiểu tổng quát cách xây dựng 1 Content-based recommendation system.

* + 1. Xây dựng utility matrix

Các bài toán liên quan đến recommendation system đều có 2 phần là user (người dùng) và item (các đối tượng mà bài toán sẽ xử lý). Content-based recommendation system sẽ đưa ra gợi ý cho các user dựa trên mức độ quan tâm của từng user đối với từng item. Rồi từ đó sẽ đưa ra cái nhìn chung và cuối cùng là đưa ra gợi ý item khác cho các user.

Utility matrix là ma trận ở đó các hàng của nó sẽ là các item. Còn các cột sẽ là từng user. Giá trị của mỗi ô trong ma trận sẽ biểu thị mức độ quan tâm của user với item tương ứng. Có 2 loại giá trị biểu thị cho mức độ quan tâm là giá trị trống (nếu user không biểu thị sự quan tâm đến item) và 2 là mức độ quan tâm cụ thể.

Ta sẽ dựa trên các item được user đánh giá cụ thể, đưa vô mô hình để huấn luyện và đầu ra hệ thống sẽ gợi ý cho các user các item chưa được đánh giá mà user có khả năng “thích” chúng.

Càng nhiều item được đánh giá thì độ chính xác của mô hình huấn luyến càng cao.

Sau đây là một ví dụ điển hình cho utility matrix:

**Bảng 1: Ví dụ về ultility matrix**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F |
| Batman | 5 | 5 | 0 | 0 | 1 | ? |
| Spyderman | 5 | ? | ? | 0 | ? | ? |
| Superman | ? | 4 | 1 | ? | ? | 1 |
| Tom & Jerry | 1 | 1 | 4 | 4 | 4 | ? |
| Doraemon | 1 | 0 | 5 | ? | ? | ? |

Theo như ví dụ trên, ta có 6 user lần lượt là A, B, C, D, E, F được đánh số từ 0 (không thích) cho đến 5 sao (rất thích) thể hiện mức độ quan tâm/đánh giá của user ứng với từng bộ phim. Ngoài ra các ô “?” biểu thị rằng user chưa đánh giá bộ phim. Nhiệm vụ của recommendation system chính là dự đoán giá trị tại các ô “?” rồi đưa ra gợi ý cho người dùng.

Batman, Spyder, Superman là các bộ phim hành động, Tom & Jerry, Doraemon là các bộ phim hoạt hình. Ta có thể thấy user A rất thích phim hành động vì Batman và Spyder đều được đánh giá 5 sao. User B cũng thích thể loại phim hành động vì Batman và Superman đều được đánh giá 5 sao và 4 sao. Tuy nhiên user A chưa đánh giá phim Superman, tuy nhiên Superman là phim hành động. Vì vậy hệ thống cần phân loại các bộ phim và sao cho có thể gợi ý phim Superman cho user A.

* + 1. Item profile

Như đã nói ở các phần trước, content-based recommendation system sẽ dựa trên nội dung, thuộc tính của các item để từ đó đưa ra gợi ý, vì vậy ta cần phải xây dựng 1 bộ hồ sơ (profile) dành cho mỗi item.

Ở ví dụ mục 1.3.1, một bộ phim thường có một số thuộc tính mà ta có thể liệt kê như: đạo diễn, diễn viên, thể loại, v.v. Ở đây, tài liệu sẽ xét theo thể loại của bộ phim. Ta chỉ xét 2 thể loại là hành động và hoạt động. Mỗi bộ phim sẽ được biểu diễn thành các vector 2 chiều, chiều thứ nhất biểu diễn mức độ hành động của bộ phim, và chiều thứ 2 biểu diễn mức độ hoạt hình của bộ phim.

Ta sẽ biểu diễn lại ví dụ ở mục 1.3.1 thành:

**Bảng 2: Ví dụ về item profile**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | Item feature vector |
| Batman | 5 | 5 | 0 | 0 | 1 | ? | x1 = [0.99, 0.02] |
| Spyderman | 5 | ? | ? | 0 | ? | ? | x2 = [0.91, 0.11] |
| Superman | ? | 4 | 1 | ? | ? | 1 | x3 = [0.95, 0.05] |
| Tom & Jerry | 1 | 1 | 4 | 4 | 4 | ? | x4 = [0.01, 0.99] |
| Doraemon | 1 | 0 | 5 | ? | ? | ? | x5 = [0.03, 0.98] |
| User model |  |  |  |  |  |  |  |

Bài toán của ta chính là đi tìm mô hình sao cho có thể fix được các có trong mô hình. Sau đó ta sẽ dự đoán các giá trị ở các ô “?”.

* + 1. Xây dựng hàm mất mát

Gọi số user là N, số item là M, utility matrix được mô tả là một ma trận Y. Thành phần ở hàng thứ m, cột thứ n của Y biểu diễn cho mức độ quan tâm của user n lên sản phẩm m mà hệ thống thu thập được. Các giá trị “?” ở bảng 2 ở trên khi biểu diễn ở ma trận Y sẽ là các ký thự trống. Ngoài ra ta sẽ biểu diễn một ma trận R, ma trận R sẽ cho ta biết một user đã đánh giá một item nào đó hay chưa, nếu sản phẩm thứ i được user j đánh giá thì ô đó sẽ có giá trị là 1, còn nếu chưa sẽ là 0.

Ta giả sử rằng, mức độ đánh giá item của các item sẽ là 1 một hàm y = f(x) ở đó x là các feature vector chúng ta sẽ cần tìm trọng số w và phần bù đại số (bias) b. Cụ thể ứng với mỗi user n, và item m ta sẽ có được độ đánh giá theo:

Sau khi ta đã huấn luyện xong, ta có được w và b phù hợp, ta sẽ thế vào mô hình để đi tìm các giá trị y ở các ô “?”. Để tính toán xem mức độ sai số ta sẽ sử dụng hàm mất mát cho từng user n như sau:

Tổng quát ta sẽ tính sai số dựa trên trung bình của các item m mà user thứ i đã đánh giá:

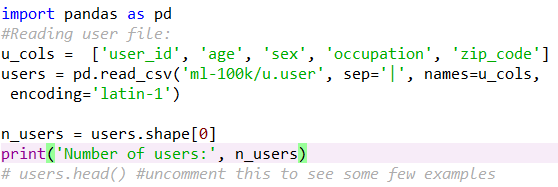
Trong đó:

Ta đặt là vector chứa các giá trị 1 tương ứng với các item mà user m đã đánh giá. Từ đó ta có thể viết lại biểu thức thành:

* + 1. Lập trình trên Python

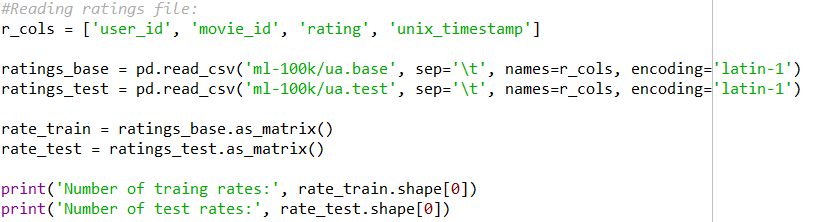
Bộ dữ liệu sử dụng: MovieLens 100k, năm 1998, bởi GroupLens.

Đọc file dữ liệu user:



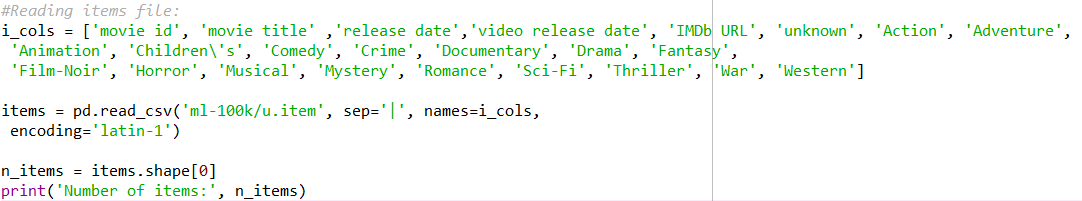
Hình 1.1 Code đọc file dữ liệu user bằng Python

Tiếp theo ta sẽ đọc file rating chứa dữ liệu item cũng như các đánh giá về item của các user, sau đó ta phân tập dữ liệu thành 2 phần là tập huấn luyện (train) để huấn luyện và xây dựng mô hình và tập kiểm thử (test) để kiểm tra tính đúng đắn của mô hình mà ta tạo được:



Hình 1.2 Đọc file rating và phân tập dữ liệu thành tập train, test

Tiếp theo ta sẽ xây dựng item profile, chứa item và các thuộc tính của item, ở đây ta sẽ chọn thuộc tính của các item chính là các thể loại phim của các item (cột 6 đến cột 24) và các thông tin cơ bản của item (cột 1 đến cột 5).



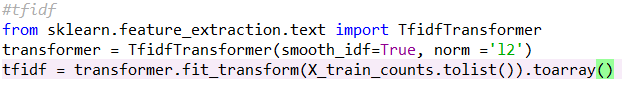
Hình 1.3 Xây dựng item profile

Ta lọc ra, bỏ ra các thông tin cơ bản (cột 1 đến cột 5) và chỉ lấy 19 cột thể loại phim (cột 6 đến cột 24) để đem đi xử lý.



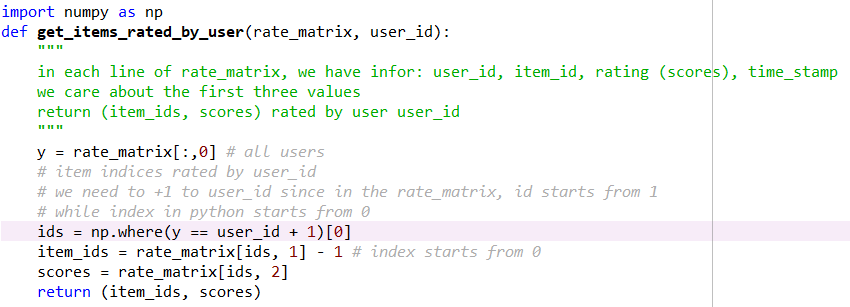
Hình 1.4 Lấy thông tin thể loại phim của từng item

Ta sử TF-IDF để xây dựng bộ feature vector cho từng item.



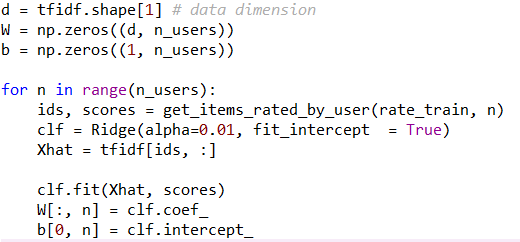
Hình 1.5 Xây dụng bộ feature vector cho item

Ta sẽ xây dựng hàm lấy danh sách các item mà user đánh giá. Ta sẽ lấy theo tất cả user (ma trận R).



Hình 1.6 Hàm lấy danh sách các item mà user đã đánh giá

Tiếp theo, ta sẽ tính Ridge Regression cho mỗi user:



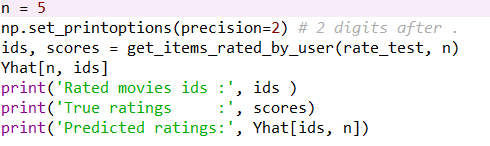
Hình 1.7 Tính Ridge Regression cho từng user

Cuối cùng ta sẽ hoàn thiện bộ feature vector bằng cách dự đoán các giá trị bằng công thức:



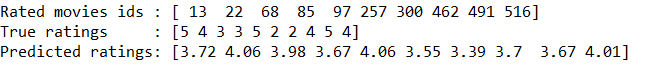
Hình 1.8 Dự đoán các giá trị

Thực hiện chạy chương trình trên user có id = 5



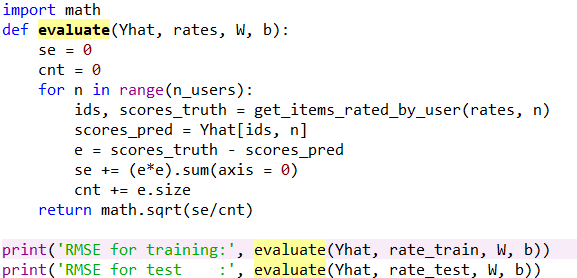
Hình 1.9 Chạy chương trình hệ thống trên user có id = 5

Sau khi chạy chương trình, ta sẽ có kết quả tương ứng:



Hình 1.10 Kết quả sau khi chạy thử chương trình

Cuối cùng, ta sẽ tiến hành đánh giá sai số bằng RMSE:



Hình 1.11 Thực hiện đánh giá sai số RMSE

Kết quả của sai số RMSE:



Hình 1.12 Kết quả sai số RMSE

Dựa trên các giá trị dự đoán đánh giá cao mà ta sẽ đưa ra được gợi ý các bộ phim mà user có id = 5 nên xem.

* 1. Neighborhood-Based Collaborative Filtering (User-User Collabrative Filtering)

Ở mục 1.3 ta đã tìm hiểu xong về Content-based Recommendation System. Cốt lõi của content-based chính là xây dựng một bộ các feature vector để từ đó ta tiến hành đi xây dựng mô hình, dự đoán và đưa ra đánh giá sai số. Tuy nhiên cách làm này có 2 khuyết điểm cơ bản. Thứ nhất, khi xây dựng mô hình, ta không thể tham khảo giá trị đánh giá của user khác. Thứ hai, không phải lúc nào ta cũng có được một bản mô tả mối liên hệ giữa item và user.

Để tránh được tình trạng không thể phân loại được item, hay không thể xây dựng được một bộ item profile, feature vector tốt. Ta có thể sử dụng một cách làm khác để có thể đưa ra 1 gợi ý cho user. Ta sẽ tham khảo các user khác, dựa vào chuỗi hành vi của user đang xét, ta sẽ tiến hành đi tìm xem có chuỗi hành vi của user nào khác có điểm chung (tương đồng) với user đang xét. Sau khi tìm được, ta sẽ gợi ý cho user đang xét những item dựa trên user khác đã tìm được.

Ví dụ, A thích xem phim Batman và Spyderman (đều đánh giá 5 sao), B thích xem phim Batman và Superman (đánh giá lần lượt là 5 sao, 4 sao). Từ đó ta suy ra được A và B đều thích xem phim hành động và đều cùng xem phim Batman (điểm chung). Nếu A là user đang được xét để đưa ra gợi ý, dựa vào hệ thống ta tìm được user B có nhiều điểm chung so với user A. Và hệ thống thấy rằng user B ngoài điểm chung thì user B có xem thêm phim Superman. Vì vậy có khả năng rằng user A có thể thích phim Superman.

Để xây dựng được 1 hệ thống như vậy, ta cần phải giải quyết 2 vấn đề:

* Cách xác định sự giống nhau giữa 2 user
* Khi đã xác định được 2 user nào giống nhau, từ đó làm cách nào để có thể dự đoán được mức độ đánh giá của user đang xét với các item.

Ở mục này, ta sẽ đi tìm xem cách để có thể tìm ra được sự giống nhau giữa 2 user. Ta gọi quá trình này là “User-user Collaborative Filtering”.

* + 1. Xây dựng Similarity function

Như đã đề cập, ta thấy rằng cốt lõi của Collaborative Filtering chính là đi tìm sự giống nhau giữa 2 user. Vì vậy việc đầu tiên mà ta cần làm đó chính là đi xây dựng hàm đánh giá mức độ giống nhau giữa 2 user.

Xét ví dụ bảng 1 ở trên ta có 6 user (A, B, C, D, E, F) và có 5 bộ phim (Batman, Spyderman, Superman, Tom & Jerry, Doraemon). Trong mỗi ô là giá trị đánh giá của các user dành cho các bộ phim mà họ đã xem. Như đã nói ở mục 1.3 điều ta cần làm chính là làm sao có thể điền các giá trị còn thiếu vào các ô có giá trị “?”. Tuy nhiên ta sẽ không đi xây dựng item profile có chứa item feature vector nữa mà ta sẽ đi tìm sự tương đồng giữa 2 user.

**Bảng 3: Nhắc lại về Utility Matrix**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F |
| Batman | 5 | 5 | 0 | 0 | 1 | ? |
| Spyderman | 5 | ? | ? | 0 | ? | ? |
| Superman | ? | 4 | 1 | ? | ? | 1 |
| Tom & Jerry | 1 | 1 | 4 | 4 | 4 | ? |
| Doraemon | 1 | 0 | 5 | ? | ? | ? |

Điều đầu tiên ta làm đó là đặt sự tương đồng giữa 2 user ui, uj là sim (ui, uj). Nhìn bảng 3 ta thấy rằng user A và user B đều thích các bộ phim hành động Batman, Spyderman, Superman (đều được đánh giá 4 sao và 5 sao) và không thích các bộ phim hoạt hình như Tom & Jerry hay Doraemon cho lắm (vì chúng chỉ được đánh giá từ 0 cho đến 1 sao). Và điều này lại xảy ra ngược lại đối với các user D, E, F. Vì vậy ta cần phải xây dựng 1 hàm tìm điểm chung (similarity function) tốt để có thể đảm bảo rằng:

Hay nói một cách tổng quát:

Để xác định mức độ đánh giá của user A với bộ phim Superman, ta cần phải dựa vào user B. Ta may mắn vì user B không những tương đồng với user A mà còn đã đánh giá phim Superman. Vì vậy ta không cần tìm user khác.

Để đo được độ tương quan (similarity) giữa 2 user, ta sẽ đi xây dựng feature vector cho từng user (khác với content-based là ta đi xây dựng feature vector cho từng item). Feature vector sẽ được xây dựng dựa trên user, do đó sẽ rất khó khăn vì không phải item đều có giá trị đánh giá của user. Có 1 cách khắc phục tình trạng này đó là ta sẽ điền các giá trị tạm thời vào chỗ còn thiếu. Tuy nhiên, giá trị tạm thời này như thế nào là hợp lý để điền vào chỗ trống, là một vấn đề mà ta phải cân nhắc.

Nếu ta điền vào chỗ trống là 0, giá trị này là không tốt vì trong một số trường hợp 0 là giá trị nhỏ nhất, khiến kết quả không được khách quan, cũng như bị lệch, sai số nhiều trong một vài trường hợp. Một giải pháp nữa đó là ta sẽ điền giá trị trung bình được tính dựa trên giá trị đánh giá cao nhất và giá trị đánh giá thấp nhất. Tuy nhiên, ta làm như vậy một vài trường user sẽ không đúng vì đôi khi có một vài user sẽ rất dễ tính hay khó tính sẽ đánh giá nghiên về một bên của giá trị trung bình, nghĩa là các user này sẽ đánh giá quá thấp hoặc đánh giá quá cao so với mức ngưỡng trung bình. Vì vậy, trường hợp tối ưu đó là ta sẽ tính trung bình cộng các giá trị đánh giá của từng user sau đó ta sẽ lấy giá trị này điền tạm thời vào các ô còn thiếu của từng user.

Để hiểu rõ hơn quá trình thực hiện, ta sẽ xét ví dụ utility matrix sau:

**Bảng 4: Một ví dụ khác về utility matrix**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | U0 | U1 | U2 | U3 | U4 | U5 | U6 |
| I0 | 5 | 5 | 2 | 0 | 1 | ? | ? |
| I1 | 4 | ? | ? | 0 | ? | 2 | ? |
| I2 | ? | 4 | 1 | ? | ? | 1 | 1 |
| I3 | 2 | 2 | 3 | 4 | 4 | ? | 4 |
| I4 | 2 | 0 | 4 | ? | ? | ? | 5 |
| Mean rating | 3.25 | 2.75 | 2.5 | 1.33 | 2.5 | 1.5 | 3.33 |

Sau đó, ta sẽ đi chuẩn hóa dữ liệu bằng cách trừ các ô có giá trị cho giá mean rating. Còn các ô “?” ta sẽ điền 0.

**Bảng 5: Một ultility matrix đã được chuẩn hóa**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | U0 | U1 | U2 | U3 | U4 | U5 | U6 |
| I0 | 1.75 | 2.25 | -0.5 | -1.33 | -1.5 | 0 | 0 |
| I1 | 0.75 | 0 | 0 | -1.33 | 0 | 0.5 | 0 |
| I2 | 0 | 1.25 | -1.5 | 0 | 0 | -0.5 | -2.33 |
| I3 | -1.25 | -0.75 | 0.5 | 2.67 | 1.5 | 0 | 0.67 |
| I4 | -1.25 | -2.75 | 1.5 | 0 | 0 | 0 | 1.67 |

Từ đây ta có thể thấy rõ hơn rằng giá trị càng âm thì càng cho thấy user không đánh cao item. Ngược lại nếu giá trị càng dương thì càng cho thấy user đánh giá cao item. Còn các giá trị 0 biểu thị user chưa đánh giá item đó. Như vậy là đáp ứng được tối ưu các điều kiện cần của bài toán đặt ra.

Tiếp theo ta sẽ tính toán độ tương đồng giữa 2 user bằng hàm cosine hay cos (u1, u2).

Từ bảng 5 ta đi tính độ tương đồng cho các user, ta được kết quả sau:

**Bảng 6: Similarity giữa các user**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | U0 | U1 | U2 | U3 | U4 | U5 | U6 |
| U0 | 1 | 0.83 | -0.58 | -0.79 | -0.82 | 0.2 | -0.38 |
| U1 | 0.83 | 1 | -0.87 | -0.40 | -0.55 | -0.23 | -0.71 |
| U2 | -0.58 | -0.87 | 1 | 0.27 | 0.32 | 0.47 | 0.96 |
| U3 | -0.79 | -0.40 | 0.27 | 1 | 0.87 | -0.29 | 0.18 |
| U4 | -0.82 | -0.55 | 0.32 | 0.87 | 1 | 0 | 0.16 |
| U5 | 0.2 | -0.23 | 0.47 | -0.29 | 0 | 1 | 0.56 |
| U6 | -0.38 | -0.71 | 0.96 | 0.18 | 0.16 | 0.56 | 1 |

Sau khi tính xong độ tương đồng giữa các user, ta có thể thấy user u0 có nhiều nét tương đồng so với user u1 vì cos của u0 với u1 cao hơn so với cos của u0 với các user còn lại.

* + 1. Rating prediction

Ta đã hoàn thành thao tác tìm cặp user có mối quan hệ tương đồng với nhau. Ở mục này ta sẽ đi hoàn thiện nốt bài toán bằng việc đi dự đoán các giá trị còn thiếu.

Để dự đoán giá trị cho một ô còn thiếu ta làm các bước sau:

* Chọn k tương ứng với số dữ liệu cần tham khảo để dự đoán giá trị đang xét
* Lấy ra tập các user đã đánh giá item đang xét.
* Tính điểm tương đồng giữa user có item đang xét với các user này, chọn ra k user có điểm tương đồng cao nhất với user đang xét, ta gọi tập này .
* Tính tổng các tích được tính theo giá trị đánh giá của user tương đồng nhân với lại độ tương đồng giữa user đang xét với user đã đánh giá.
* Sau đó ta lấy tổng trên chia cho tổng k độ tương đồng (lấy dương độ tương đồng).

Nói ngắn gọn hơn ta dự đoán các giá trị bằng công thức:

Như vậy là ta đã có dự đoán và điền được hoàn thiện các chỗ còn thiếu:

**Bảng 7: Thay các giá trị chưa đánh giá (0) bằng các giá trị dự đoán**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | U0 | U1 | U2 | U3 | U4 | U5 | U6 |
| I0 | 1.75 | 2.25 | -0.5 | -1.33 | -1.5 | **0.18** | **-0.63** |
| I1 | 0.75 | **0.48** | **-0.17** | -1.33 | **-1.33** | 0.5 | **0.05** |
| I2 | **0.91** | 1.25 | -1.5 | **-1.84** | **-1.78** | -0.5 | -2.33 |
| I3 | -1.25 | -0.75 | 0.5 | 2.67 | 1.5 | **0.59** | 0.67 |
| I4 | -1.25 | -2.75 | 1.5 | **1.57** | **1.56** | **1.59** | 1.67 |

Như vậy từ đây ta sẽ cộng lại các giá trị đánh giá cho giá trị mean hồi nãy mà ta đã trừ đi, ta sẽ khôi phục lại được các giá trị đánh giá gốc. Từ đây ta có thể đưa ra các gợi ý cho user dựa vào các điểm đánh giá.

* + 1. Khuyết điểm

Trong thực tế thường có rất nhiều user việc đi tìm similarity giữa các user sẽ rất nhiều, việc chứa các kết quả similarity này đôi khi gặp khó khăn vì tài nguyên bộ nhớ có hạn khiến không thể lưu trữ hết các tính toán similarity.

Ngoài ra, utility matrix thường rất thưa (rất ít đánh giá). Dẫn tới bảng chuẩn hóa bị thưa và cần rất nhiều tính toán để có thể điền hết utility matrix.

* 1. Neighborhood-Based Collaborative Filtering (Item-Item Collabrative Filtering)
     1. Khắc phục khuyết điểm của User-User Collaborative Filtering

Để khắc phục được một số khuyết điểm nhỏ của User-User Collaborative Filtering mà ta đã nói ở mục 1.4.3 ở trên, ta sẽ đi tìm similarity giữa item với item, thay vì là user với user.

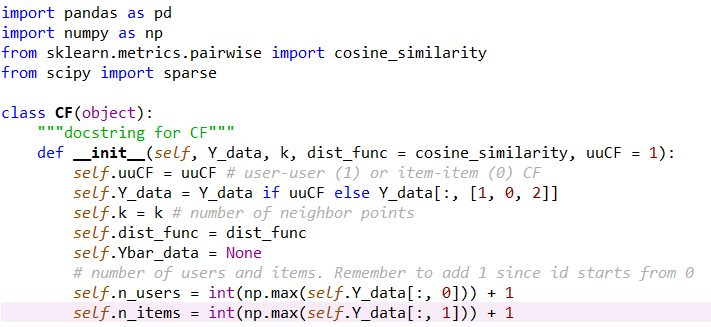
Do:

* Số lượng item thường ít hơn số lượng user, trong một vài trường hợp sẽ thuận lợi về mặt tính toán.
* Trong một vài trường hợp item sẽ được đánh giá nhiều dẫn tới việc đi tìm trung bình cộng sẽ dễ dàng hơn cũng như tính toán similarity sẽ chính xác hơn.

Về mặt tính toán, Item-Item Collaborative Filtering được tính bằng cách chuyển vị utility matrix, và xem các item như đang rate các user. Sau khi tính ra kết quả cuối cùng, ta sẽ chuyển vị lại một lần nữa để thu được kết quả

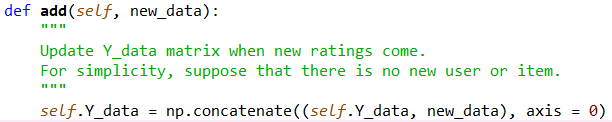
* + 1. Lập trình Neighborhood-Based Collaborative Filtering trên Python

Đầu tiên, ta khởi tạo class dành cho nghiệp vụ đưa ra gợi ý. Ta gọi class này là CF. Hàm khởi tạo gồm Y\_data là dữ liệu rating đầu vào, k là số lượng user sẽ được tham khảo, dist\_function là hàm tính similarity, uuCF là biến cờ (nếu là 1 ta sử dụng user-user collaborative filtering, nếu là 0 ta sử dụng item-item collaborative filtering).



Hình 1.13 Khai báo và khởi tạo lớp CF

Ta viết hàm để thêm đối tượng vào utility matrix.



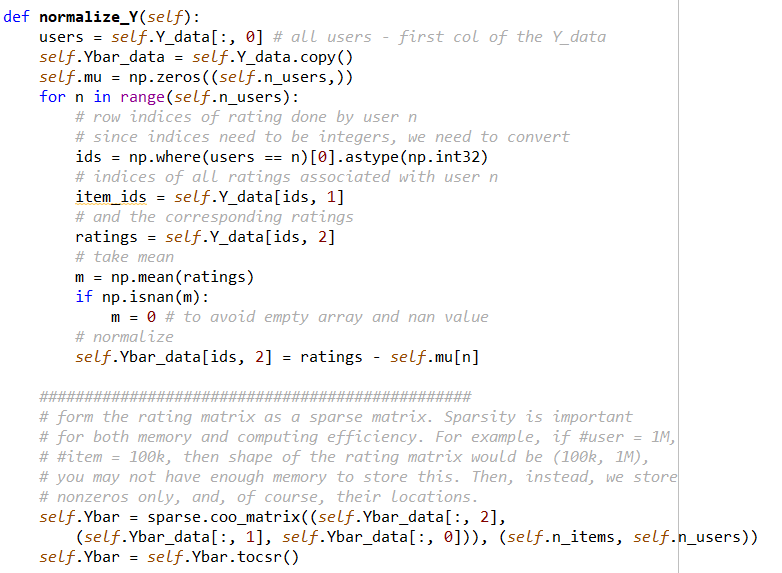
Hình 1.14 Hàm thêm đối tượng vào utility matrix

Tiếp tục ta viết hàm tính toán similarity giữa các đối tượng.



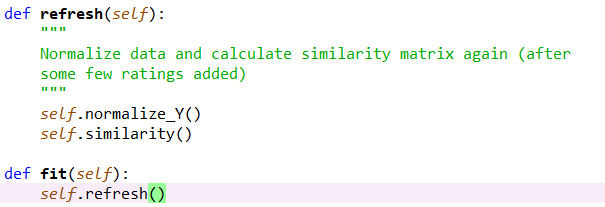
Hình 1.15 Hàm tính similarity

Tiếp theo ta sẽ xây dựng hàm để chuẩn hóa dữ liệu.



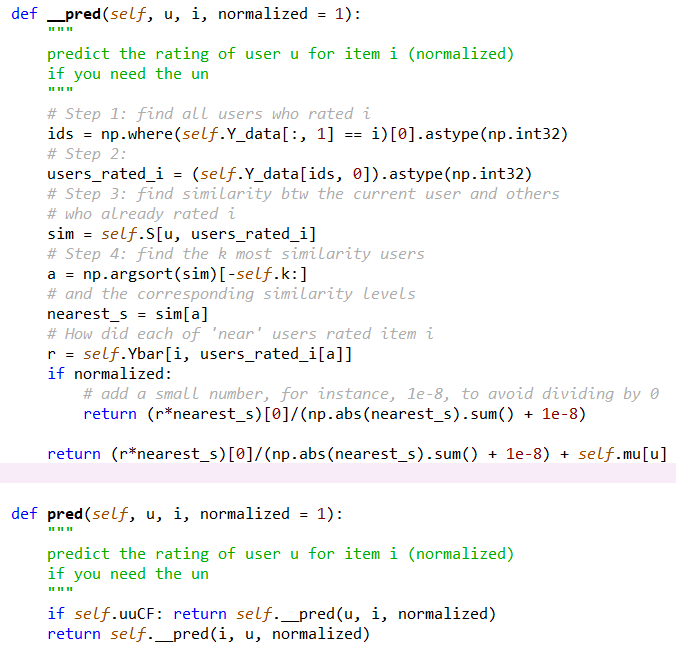
Hình 1.16 Hàm chuẩn hóa dữ liệu

Khi một dữ liệu mới thêm vào utility matrix, ta phải chuẩn hóa lại utility matrix và tính lại similarity giữa các đối tượng.



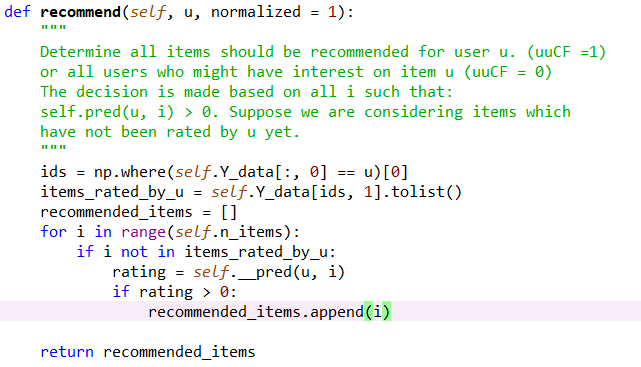
Hình 1.17 Thực hiện chuẩn hóa và tính similarity khi có dữ liệu mới

Cuối cùng ta sẽ đi xây dựng hàm dự đoán các giá trị đánh giá.



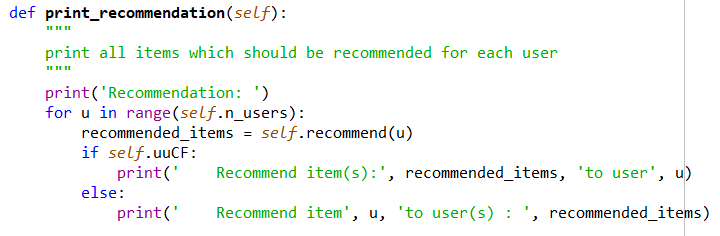
Hình 1.18 Hàm dự đoán

Sau khi xây dựng xong mô hình, ta đi thực hiện đưa ra gợi ý.



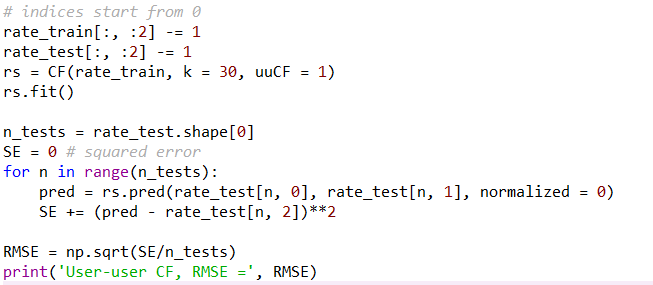
Hình 1.19 Hàm đưa ra gợi ý

Cuối cùng, ta sẽ ra kết quả.



Hình 1.20 In kết quả gợi ý ra màn hình

Ta sẽ chạy thử chương trình.

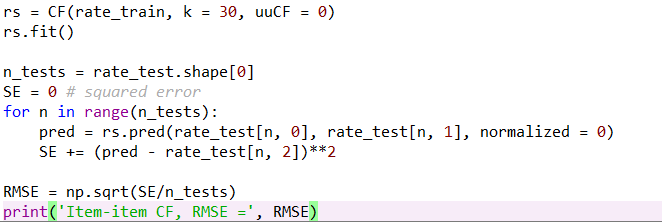


Hình 1.21 Chạy và test RMSE mô hình trên tập dữ liệu



Hình 1.22 Kết quả RMSE sau khi chạy User-User Collaborative Filtering

Tiếp tục ta chạy thử trên Item-Item Collaborative.



Hình 1.23 Chạy và test RMSE Item-Item Collaborative Filtering



Hình 1.24 Kết quả RMSE sau khi chạy Item-Item Collaborative Filtering

Về cơ bản ta đã giải quyết được khái niệm thế nào là 1 recommendation system cũng như các loại, cách thức hoạt động của hệ thống. các mục, tiêu đề sau ta sẽ bàn về các vấn đề bảo mật của hệ thống.

* 1. Giới thiêu sơ lược về vấn đề về bảo mật

Như đã giới thiệu ở các phần trước, hệ thống đưa ra gợi ý chưa có cơ chế kiểm soát chặt chẽ nên rất dễ bị tấn cống ở nguồn dữ liệu đầu vào. Trong đó các cuộc tấn công, các nguồn dữ liệu đầu vào này thường chứa các đối tượng user hoặc item nhiễu có thể làm ảnh hưởng đến độ chính xác đầu ra của hệ thống. Hầu hết các cuộc tấn công này thường xuất phát từ các thế lực cạnh tranh, thù địch với người sử dụng hệ thống. Dựa trên các mô hình tấn công, các cuộc tấn công có thể được chia thành các thể loại khác nhau như push attack, nuke attack, standard attack hay obfuscated attack.

* 1. Tổng kết chương 1

Như vậy là ta đã tìm hiểu xong về hệ thống gợi ý cũng như tài liệu cũng đã giới thiệu sơ về lỗ hỏng bảo mật cơ bản của 1 hệ thống gợi ý.

Ta có thể ôn lại khái quát chương 1 qua bảng sau:

**Bảng 8: Khái quát nội dung chương 1**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Loại hệ thống | Đặc điểm hệ thống | Các kỹ thuật thường sử dụng | Ưu điểm | Khuyết điểm | Các ví dụ thực tế từ các doanh nghiệp |
| 1 | Content-based Filtering | Đưa ra gợi ý dựa trên các item đã đánh giá của người dùng. | Bayesian Classifier, Cluster analysis, Decision tree, Artificial Neural Network. | Đưa ra gợi ý chỉ dựa trên một mình user đang xét. | Sẽ không thể thực hiện được nếu không phân loại được item. | Pandora, Radio, Rotten, Tomatoes, Jinni. |
| 2 | Collaborative Filtering | Dựa trên các khuôn mẫu về hành vi của người dùng, hay item để đưa ra gợi ý cho người dùng hay item khác. | Bayesian network, Clustering, Artificial Neural Network, Linear Regression, Probabilistic model, Graph theory, Matrix Factorization. | Có thể thực hiện trên bất kỳ item hay user nào. | Có độ thưa lớn, Phụ thuộc (đưa ra các gợi ý) vào các item được đánh giá cao hay được đánh giá nhiều. | GroupLens, Amazon, LinkedIn. |
| 3 | Hybrid Filtering | Kết hợp cả hai Content-based và Collaborative Filtering. | Sử dụng kết hợp các phương pháp của Content-based và Collaborative Filtering trong 1 bài toán. | Có khả năng giải quyết được các khuyết điểm còn tồn đọng của cả hai. |  | Netflix, NewsDude. |

CHƯƠNG 2: PHÁT BIỂU BÀI TOÁN

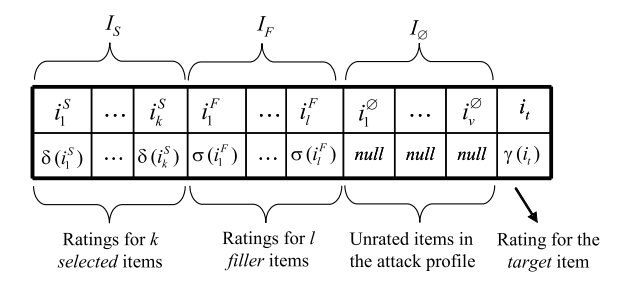
Với sự tiến bộ của các hệ thống gợi ý, ngày nay ta có rất nhiều kỹ thuật để có thể làm tăng sự chú ý hay làm giảm sự chú ý đến 1 item. Sau khi tìm hiểu về hệ thống gợi ý ở chương 1, ta thấy cốt lõi của hệ thống gợi ý chính là dựa vào tập dữ liệu đầu vào, từ tập dữ liệu mà ta đem đi tiến hành tiến hành phân tích, dự đoán và đưa ra gợi ý. Vì vậy tập dữ liệu rất quan trọng, ảnh hưởng rất lớn đến kết quả đầu ra của hệ thống. Do đó, tập dữ liệu đầu vào sẽ là một trong những mục tiêu để kẻ tấn công nhắm vào để gây hại đến hệ thống. Những kiểu tấn công này được gọi là các các cuộc tấn công xâm nhập hồ sơ (profile injection) hay được gọi là các cuộc tấn công cò mồi (shilling attack). Các cuộc tấn công này sẽ chèn tập dữ liệu các user giả, hay các dữ liệu đánh giá giả nhằm khiến quá trình tính toán bị sai lệch từ đó khiến chương trình đưa ra các dự đoán sai khiến hệ thống đưa ra các gợi ý không chính xác.

* 1. Bộ hồ sơ giả

Một bộ các hồ sơ giả bao gồm một tập hợp các hồ sơ giả được thêm vào hệ thống bởi kẻ tấn công. Dạng chung của các profile này được hiển thị trong Hình 2.1. Mỗi bộ profile gồm có 4 thành phần: mục tiêu sẽ bị tấn công (target (t)), một bộ các item được chọn (selected) với các đặc điểm cụ thể được xác định bởi kẻ tấn công (s), một bộ các item phụ thường được chọn ngẫu nhiên (filler (f)) và một bộ các item chưa được đánh giá (∅).

Các mô hình tấn công khác nhau có thể được xác định bằng tỉ lệ cũng như số lượng item của 4 thành phần này (số lượng các item được chọn (s), tỷ lệ của các item phụ (f) và cách xếp hạng cụ thể được gán cho từng item này sao cho đạt được mục đích của kẻ tấn công). Ta sẽ tìm hiểu kĩ hơn về các thành phần này ở mục sau.

Tập hợp các item (user) được chọn, sẽ được tiến hành dựa trên mục đích của kẻ tấn công. Trong một số cuộc tấn công, thành phần các selected item này thường sẽ là tập rỗng. Thay vào đó, tập hợp các item phụ (filler item) sẽ được chọn để tấn công (trong các trường hợp ta có thể coi các filler item trở nên ngẫu nhiên trở thành các selected item). Vì bộ các selected item thường có kích thước nhỏ, nên kích thước của mỗi hồ sơ (tổng số xếp hạng) được xác định chủ yếu bằng chính kích thước của bộ các item phụ.



Hình 2.1 Cấu trúc của một user giả [2]

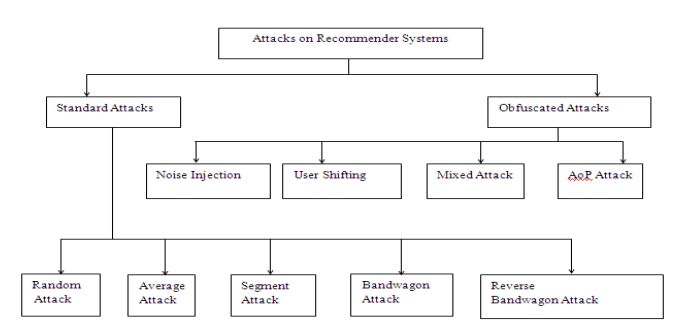
Ta có thể thấy rằng user thật thường có tính khách quan, gồm 2 phần: phần thứ nhất là các item đã được đánh giá (thường có rất ít item được đánh giá) và phần thứ hai là các item chưa được đánh giá (thường còn rất nhiều item chưa được đánh giá), vì không có bất kỳ user thật nào có thể đánh giá phần lớn hầu hết hệ thống được.

Theo hình 2.1, ta thấy được các user giả thường sẽ đối lập với các user thật. Các user này cũng được chia thành 2 phần chính: phần thứ nhất là các item chưa được đánh giá lại rất ít (đối lập với user thật là các chưa được đánh giá lại nhiều), trong khi các item được đánh giá lại rất nhiều.

Ngoài ra, ta còn thấy được trong phần các item được đánh giá của user gồm có nhiều thành phần con, các thành phần con này có những mục đích tấn công khác nhau. Các thành phần con này bao gồm: các item mục tiêu (target item it), các item được chọn (selected item is) và các item phụ (filler item if).

* 1. Các loại hình tấn công

Ta có thể khái quát được các loại hình tấn công thông qua hình vẽ sau:



Hình 2.2 Các loại hình tấn công hệ thống gợi ý [1]

* + 1. Chiều không gian tấn công

Các cuộc tấn công bằng các hồ sơ giả có thể được phân loại dựa trên trình độ (độ phức tạp sử dụng), ý định cụ thể và quy mô của kẻ tấn công.

Từ quan điểm của kẻ tấn công, một cuộc tấn công tốt để chống lại một hệ thống là một cuộc tấn công mà nó sẽ vừa có khả năng gây ra tác động lớn nhất cho hệ thống mà vừa sao cho nỗ lực của kẻ tấn công sẽ sử dụng là ít nhất. Có nhiều cách khác nhau để đánh giá sự nỗ lực của kẻ tấn công đã sử dụng, nhưng trong bài viết này, chúng ta sẽ chủ yếu tập trung nhấn mạnh về các vấn đề khác quan trọng hơn như: Cách thức chuẩn bị của kẻ tấn công cho một cuộc tấn công cụ thể như thế nào? Kiến thức dành riêng cho các hệ thống gợi ý cụ thể (chúng ta đã biết ở chương 1). Cách phòng thủ đơn giản của 1 hệ thống gợi ý nên sử dụng. Ở đây, chúng ta tạm phân ra 2 loại tấn công cơ bản là:

Tấn công trình độ cao (độ phức tạp cao): Một cuộc tấn công có trình độ cao là một cuộc tấn công đòi hỏi kẻ tấn công phải nắm vững và hiểu rõ chi tiết về hệ thống mục tiêu. Các yếu tố về hệ thống đánh giá như: giá trị trung bình của các đánh giá, hay giá trị chuẩn hóa của các đánh giá là một trong những giá trị mà kẻ tấn công sẽ phải chuẩn bị.

Tấn công trình độ ​​thấp (độ phức tạp thấp): Một cuộc tấn công có trình độ thấp là một cuộc tấn công mà ở đó kẻ tấn công chỉ nắm, hoặc hiểu sơ sài về hệ thống hoặc kẻ tấn công không biết gì về hệ thống.

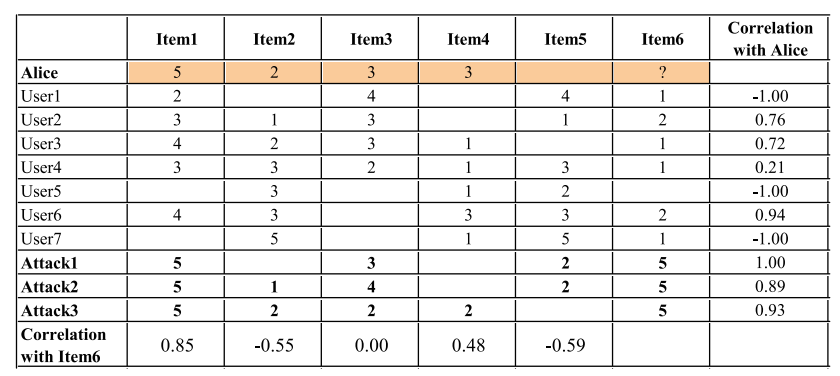
Yếu tố thứ hai ảnh hưởng đến một cuộc tấn công đó chính là ý định của kẻ tấn công. Dựa trên ý định của kẻ tấn công ta có 2 loại kiểu tấn công, đó là “push” và “nuke”. Kẻ tấn công có thể chèn các hồ sơ giả để nâng điểm đánh giá (push) hay để làm giảm đánh giá của một sản phẩm hay item nào đó. Mục tiêu cuối cùng vẫn là làm cho hệ thống gợi ý trở nên làm việc kém hiệu quả. Chúng ta sẽ tạm thời quên đi các mục tiêu khác nằm ngoài hệ thống (ví dụ dùng các hình thức này cho mục đích cạnh tranh thương mại, buôn bán, v.v). Chúng ta, ở đây chỉ tập trung quan tâm chủ yếu đến yếu tố chính xác của hệ thống.

Kích thước của một cuộc tấn công có thể được đo bằng nhiều cách. Chúng ta sẽ xem xét đến 2 thông số chính. Đó là số lượng hồ sơ giả được thêm bởi kẻ tấn công và số lượng xếp hạng được cung cấp trong mỗi hồ sơ. Chúng ta giả định rằng kẻ tấn công có khả năng tự động hóa quá trình tiêm nhiễm các hồ sơ giả vào hệ thống. Do đó, số lượng hồ sơ là một biến số quan trọng bởi vì 1 hồ sơ có thể được tạo một cách dễ dàng bằng các chương trình đăng ký trực và bằng phương tiện này, chủ sở hữu trang web vô tình phải chịu chi phí tổn thất từ các hồ sơ mới vì đa phần kẻ tấn công sử dụng các profile giả để tấn công hơn là sử dụng các item giả. Tuy nhiên các bộ hồ sơ giả này rất dễ bị phát hiện vì nó không có tính khách quan so với các bộ hồ thật. Ví dụ một bộ hồ sơ giả có thể sẽ có các dấu hiệu như có chứa toàn bộ các giá trị đánh giá dành cho tất cả các item hoặc cho các giá trị đánh giá bất thường. Vì vậy còn một cách tấn công khác đó chính là sử dụng các item giả. Cách làm này tuy không hiệu quả cũng như không dễ dàng nhưng lại rất khó bị phát hiện.

* + 1. Giới thiệu sơ lược về các loại hình tấn công

Để thực hiện 1 cuộc tấn công chống lại 1 Collaborative filtering recommendation system ta cần 2 yếu tố. Một là 1 tập hợp các hồ sơ giả dùng để tấn công, trong đó mỗi hồ sơ sẽ chứa các dữ liệu xếp hạng thiên vị (không chính xác). Và yếu tố thứ hai chính là mục tiêu, mục tiêu ở đây chính là đối tượng mà kẻ tấn công muốn hệ thống đề xuất gợi ý nhằm làm tăng sự chú ý tới đối tượng này (ta gọi cuộc tấn công này là push attack), hoặc nhằm làm giảm sự chú ý tới đối tượng này (ta gọi cuộc tấn công này là nuke attack).

Ta xét push attack thông qua ví dụ sau:

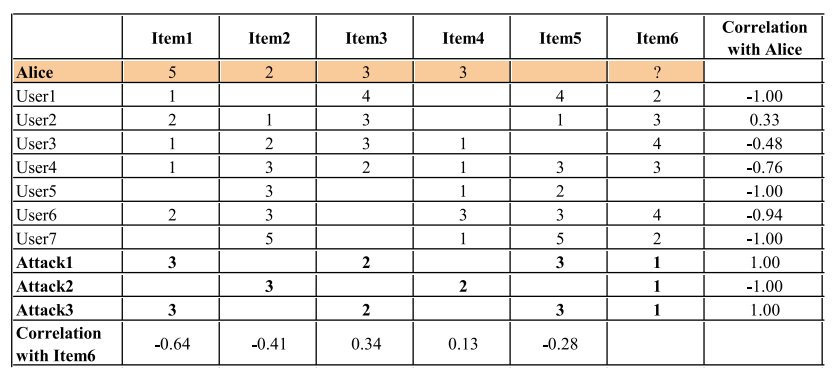


Hình 2.3 Ví dụ về push attack [2]

Hình 2.3 ở trên minh họa cho push attack. Ở ví dụ trên các Item sẽ được đánh giá theo thang điểm từ 1 đến 5 (1: Cực kỳ không thích đến 5: Cực kỳ thích). Hệ thống đang xét và đưa ra gợi ý cho user Alice. Các ứng cử viên khác đang được xét là các User được đánh số từ 1 đến 7. Các user có tên “Attack” từ 1 cho đến 3 là các profile giả do thủ phạm đưa vào hệ thống.

Bình thường nếu không có các hồ sơ (profile) giả, hệ thống sẽ đưa ra gợi ý cho Alice dựa trên User6, và đưa ra gợi Item6 dựa trên Item2. Với 3 profile giả được thủ phạm đưa vào, thì user Alice sẽ được ra gợi ý dựa trên user Attack1 thay vì là User6. Và với 3 lần đánh giá 5 (cực kỳ thích) trên cả 3 profile giả, Item6 sẽ vô tình trở nên được yêu thích và sẽ được gợi ý nhiều bởi hệ thống (Item6 giờ sẽ được đưa ra gợi ý dựa trên Item1 mà Item1 có vẻ được yêu thích tại nhiều user (Alice, User3, User6)), thay vì trước đó chỉ dựa vào Item2 (vốn chỉ có User7 yêu thích).

Tương tự, ta xét nuke attack thông qua ví dụ sau:



Hình 2.4 Ví dụ về nuke attack [2]

Bình thường nếu không có các profile giả, hệ thống sẽ đưa ra gợi ý cho Alice dựa trên User2 và Item6 sẽ được dự đoán 3 điểm trên Alice. Tuy nhiên do có các profile giả nên thay vì là User2 sẽ là profile Attack1. Mà ở profile Attack1, Item6 bị đánh giá 1 nên Item6 sẽ từ mức độ yêu thích trung bình giảm xuống mức độ yêu thích trở nên cực kỳ tệ.

* + 1. Các định nghĩa chung

**Định nghĩa 1:** Ta có mô hình chung sẽ là:

Trong đó:

Với:

* : Target item (mục tiêu của kẻ tấn công)
* I: Tập Item
* U: Tập User
* : Một phần của tập I. Gồm 2 phần là: IS tập các selected item đã được định sẵn các giá trị đánh giá, IF là tập các item được dùng để điều chỉnh các trị số sao cho đạt được mục đích tấn công.
* : là các Ánh xạ (mapping) các giá trị trong tập IS, IF, IT tương ứng đến tập R (tập các rating).

Tập hợp các tham số được sử dụng trong hàm lựa chọn χ sẽ được định nghĩa riêng cho từng mô hình tấn công cụ thể. Tập hợp các item được chọn, IS, có thể được xác định theo một số yếu tố. Các yếu tố này có thể bao gồm các giá trị phân phối xếp hạng giữa các item hoặc user, giá trị xác suất của một item cụ thể được tính dựa trên các đánh giá cao, các đánh giá thường xuyên hoặc các đánh giá liên quan đến một phân khúc người dùng cụ thể. Ta sẽ tìm hiểu kĩ hơn về vấn đề này ở các mục sau.

**Định nghĩa 2:**

Một profile tấn công trong 1 mô hình tấn công là 1 tập của các cặp (item, rating).

Trong đó:

* là mô hình tấn công

Nói chung, 1 cuộc tấn công bằng các profile giả trong một collaborative system, là 1 cuộc tấn công bằng 1 tập các profile giả cùng loại (tức là, dựa trên cùng một mô hình tấn công) được thêm vào cơ sở dữ liệu. Mục tiêu của một cuộc tấn công như vậy là để tăng (trong trường hợp push attack) hoặc giảm (trong trường hợp nuke attack) giá trị xếp hạng dự đoán trên một (nhiều) item mục tiêu cho một người dùng nhất định (hoặc một nhóm người dùng).

**Định nghĩa 3:**

Một cuộc tấn công profile-injection có kích thước n chống lại một recommendation system S là 1 tập tất cả các profile giả dùng để tấn công trong mô hình tấn công M, được thêm vào database user profile UP.

* + 1. Random attack

Trong random attack, các hồ sơ giả được tạo bằng cách sinh các giá trị đánh giá ngẫu nhiên dựa trên các phân phối tổng thể về các giá trị đánh giá của người dùng (đối tượng bị kẻ tấn công nhắm đến) trong cơ sở dữ liệu (không tính các target item của đối tượng). Đây là phương pháp tấn công rất đơn giản để thực hiện tuy nhiên tính hiệu quả của nó là không cao (is = 0, if = ngẫu nhiên, it = tối đa).

Mô hình của random attack như sau:

Trong đó:

* .
* , ta đặt là tỉ lệ các item filler được random trong .
* , với là các giá trị trung bình và độ lệch chuẩn của các giá trị đánh giá, tức là các giá trị sẽ được lấy ngẫu nhiên xung quanh giá trị trung bình của phân phối chuẩn các giá trị đánh giá trên toàn database
* .
  + 1. Average attack

Trong average attack, các giá trị đánh giá của các item phụ (filler item) của các hồ sơ giả được tính bằng cách tính trung bình giá trị đánh giá của các item đang xét trên tất cả các user. Đây là một phương pháp tấn công hiệu quả tuy nhiên đòi hỏi kẻ tấn công phải có kiến thức cũng như biết trước về hệ thống (is = 0, if = Average, it = tối đa).

Mô hình của average attack như sau:

Trong đó:

* .
* , ta đặt là tỉ lệ các item filler được định nghĩa trong .
* , với là các giá trị trung bình và độ lệch chuẩn của các giá trị đánh giá, tức là các giá trị sẽ được lấy ngẫu nhiên xung quanh giá trị trung bình của phân phối chuẩn các giá trị đánh giá của item i.
* .
  + 1. Segment attack

Tấn công phân đoạn về cơ bản nhắm vào một nhóm người dùng cụ thể có thể đã đánh giá các item mục tiêu. Nói cách khác, segment attack sẽ làm cho một nhóm các item trở nên phổ biến một cách bất bình thường (is = tối đa, if = tối thiểu, it = tối đa).

* + 1. Bandwagon attack (Popular attack)

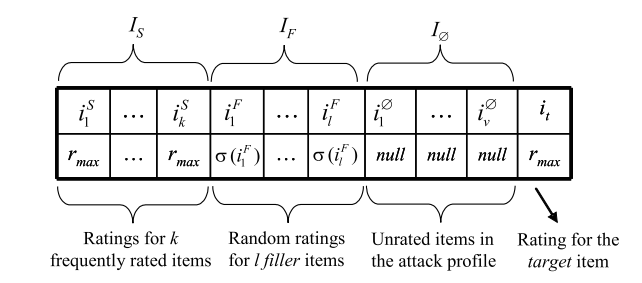
Mục tiêu của cuộc tấn công bandwagon là tạo mối liên kết (tạo sự ảnh hưởng) từ các item bị tấn công đến một số lượng nhỏ các item khác được đánh giá thường xuyên hoặc có tính liên quan đến item bị tấn công.

Mô hình của average attack như sau:

Trong đó:

* là tập các item được chọn để tấn công. Các item này thường có số lượng được đánh giá lớn trong . Tập IS được chọn sao cho khả năng xảy ra cao đối với mỗi:
* với là giá trị sẽ được sử dụng trong R.
* là tập các item được chọn ngẫu nhiên từ tập , ta đặt là tỉ lệ các item filler được định nghĩa trong .
* , với là các giá trị trung bình và độ lệch chuẩn của các giá trị đánh giá, tức là các giá trị sẽ được lấy ngẫu nhiên xung quanh giá trị trung bình của phân phối chuẩn trên toàn database.
* .

Ta có minh họa cho Bandwagon attack như sau:



Hình 2.5 Cấu trúc 1 bộ profile dùng trong Bandwagon attack [2]

Ta thấy qua hình 2.5 một số điểm như sau: IS là tập chứa các item có tần suất được đánh giá nhiều trong hệ thống, IF là tập chứa item filler dùng để điều chỉnh hệ số sao cho cuộc tấn công đạt được mục đích (thường lấy random), tập các item rỗng không liên quan và tập IT các đối tượng mục tiêu.

* + 1. Reverse-Bandwagon attack

Reverse Bandwagon là một biến thể của Bandwagon Attack. Thay vì chọn các item có tần suất được đánh giá nhiều thì reverse-bandwagon attack sẽ chọn các item có tần suất đánh giá ít để thực hiện tấn công.

* + 1. User shifting

Trong các loại tấn công này, về cơ bản chúng tôi tăng hoặc giảm tất cả các xếp hạng cho một tập hợp con các mục trên mỗi hồ sơ tấn công bằng một số lượng không đổi để giảm sự giống nhau giữa các hồ sơ tấn công.

* + 1. Mixed attack

Trong Tấn công hỗn hợp, tấn công là như nhau mục tiêu nhưng cuộc tấn công đó được tạo ra từ cuộc tấn công khác nhau mô-đun.

* + 1. Noise injection

Loại tấn công này được thực hiện bởi thêm một số tiếng ồn để xếp hạng theo một tiêu chuẩn bình thường phân phối nhân với một hằng số, được sử dụng để chi phối lượng tiếng ồn được thêm vào. Điều này thêm tiếng ồn có thể được sử dụng để ảnh hưởng đến đầu ra được tạo ra.

* + 1. Average over Popular attack (AoP)

Tấn công AoP [15] là được thiết kế để che giấu cuộc tấn công trung bình bằng cách chọn phụ các mục có xác suất bằng nhau từ top x% phổ biến nhất các mục hơn là từ toàn bộ cơ sở dữ liệu.

Ngoài các danh mục nêu trên để phân loại các cuộc tấn công, các cuộc tấn công cũng có thể được phân loại thành: đẩy [16] và nuke [16] các cuộc tấn công trong đó, trong các cuộc tấn công đẩy, xếp hạng cao hơn là được trao cho các mục tiêu, để quảng bá sản phẩm khi ở trong các cuộc tấn công nuke, xếp hạng thấp hơn được trao cho các mục tiêu, để hạ cấp một sản phẩm. Bảng 2 đưa ra một cái nhìn tổng quan về sự khác biệt thuộc tính của các mô hình tấn công nhất định.

TÀI LIỆU THAM KHẢO

**Tiếng Anh:**

1. Saakshi Kapoor, Dr.Vishal Gupta and Rohit Kumar (2017), A review of attacks and its detection attributes on collaborative recommender systems.
2. Bamshad Mobasher, Robin Burke, Runa Bhaumik, Chad Williams (2007), towards trustworthy recommender systems: an analysis of attack models and algorithm robustness.