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](#_Toc26822119)

[MỤC LỤC 1](#_Toc26822120)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 4](#_Toc26822121)

[CHƯƠNG 1: HỆ THỐNG GỢI Ý 6](#_Toc26822122)

[1.1 Hiện tượng long-tail (khởi nguồn ra đời của hệ thống gợi ý) 7](#_Toc26822123)

[1.2 Giới thiệu sơ lược về hệ thống gợi ý (Recommender System) 7](#_Toc26822124)

[1.3 Content-based Recommendation System 8](#_Toc26822125)

[1.3.1 Xây dựng utility matrix 8](#_Toc26822126)

[1.3.2 Item profile 9](#_Toc26822127)

[1.3.3 Xây dựng hàm mất mát 10](#_Toc26822128)

[1.3.4 Lập trình trên Python 11](#_Toc26822129)

[1.4 Neighborhood-Based Collaborative Filtering (User-User Collabrative Filtering) 14](#_Toc26822130)

[1.4.1 Xây dựng Similarity function 15](#_Toc26822131)

[1.4.2 Rating prediction 18](#_Toc26822132)

[1.4.3 Khuyết điểm 19](#_Toc26822133)

[1.5 Neighborhood-Based Collaborative Filtering (Item-Item Collabrative Filtering) 20](#_Toc26822134)

[1.5.1 Khắc phục khuyết điểm của User-User Collaborative Filtering 20](#_Toc26822135)

[1.5.2 Lập trình Neighborhood-Based Collaborative Filtering trên Python 20](#_Toc26822136)

[1.6 Giới thiêu sơ lược về vấn đề về bảo mật 26](#_Toc26822137)

[1.7 Tổng kết chương 1 26](#_Toc26822138)

[CHƯƠNG 2: PHÁT BIỂU BÀI TOÁN 28](#_Toc26822139)

[2.1 Bộ hồ sơ giả 28](#_Toc26822140)

[2.2 Các loại hình tấn công 29](#_Toc26822141)

[2.2.1 Chiều không gian tấn công 30](#_Toc26822142)

[2.2.2 Giới thiệu sơ lược về các loại hình tấn công 31](#_Toc26822143)

[2.2.3 Các định nghĩa chung 33](#_Toc26822144)

[2.2.4 Random attack 35](#_Toc26822145)

[2.2.5 Average attack 35](#_Toc26822146)

[2.2.6 Segment attack 36](#_Toc26822147)

[2.2.7 Bandwagon attack (Popular attack) 38](#_Toc26822148)

[2.2.8 Reverse-Bandwagon attack 39](#_Toc26822149)

[2.2.9 Love/Hate attack 39](#_Toc26822150)

[2.2.10 User shifting 40](#_Toc26822151)

[2.2.11 Mixed attack 40](#_Toc26822152)

[2.2.12 Noise injection 40](#_Toc26822153)

[2.2.13 Average over Popular attack (AoP) 40](#_Toc26822154)

[2.3 Tổng kết chương 40](#_Toc26822155)

[CHƯƠNG 3: GIẢI QUYẾT BÀI TOÁN 42](#_Toc26822156)

[3.1 Các kỹ thuật nhận dạng 42](#_Toc26822157)

[3.2 Các giải thuật phân loại profile 42](#_Toc26822158)

[3.2.1 Rating Deviation from Mean Agreement (RDMA) 42](#_Toc26822159)

[3.2.2 Weighted Deviation from Mean Agreement (WDMA) 43](#_Toc26822160)

[3.2.3 Degree of Similarity with Top Neighbors (DegSim) 43](#_Toc26822161)

[3.2.4 Length variance 43](#_Toc26822162)

[3.2.5 Mean variance 43](#_Toc26822163)

[3.2.6 Filler Mean Target Difference (FMTD) 44](#_Toc26822164)

[3.2.7 Target Model Focus (TMF) 44](#_Toc26822165)

[3.3 Tổng kết chương 45](#_Toc26822166)

[CHƯƠNG 4: THỰC NGHIỆM 46](#_Toc26822167)

[4.1 Dữ liệu đầu vào 46](#_Toc26822168)

[4.2 Phương thức sử dụng 46](#_Toc26822169)

[4.3 Kết quả và đánh giá 47](#_Toc26822170)

[4.4 Tổng kết 50](#_Toc26822171)

[TÀI LIỆU THAM KHẢO 51](#_Toc26822172)

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

[**Bảng 1: Ví dụ về ultility matrix** 9](#_Toc26822173)

[**Bảng 2: Ví dụ về item profile** 10](#_Toc26822174)

[Hình 1.1 Code đọc file dữ liệu user bằng Python 11](#_Toc26822175)

[Hình 1.2 Đọc file rating và phân tập dữ liệu thành tập train, test 12](#_Toc26822176)

[Hình 1.3 Xây dựng item profile 12](#_Toc26822177)

[Hình 1.4 Lấy thông tin thể loại phim của từng item 12](#_Toc26822178)

[Hình 1.5 Xây dụng bộ feature vector cho item 12](#_Toc26822179)

[Hình 1.6 Hàm lấy danh sách các item mà user đã đánh giá 13](#_Toc26822180)

[Hình 1.7 Tính Ridge Regression cho từng user 13](#_Toc26822181)

[Hình 1.8 Dự đoán các giá trị 13](#_Toc26822182)

[Hình 1.9 Chạy chương trình hệ thống trên user có id = 5 13](#_Toc26822183)

[Hình 1.10 Kết quả sau khi chạy thử chương trình 14](#_Toc26822184)

[Hình 1.11 Thực hiện đánh giá sai số RMSE 14](#_Toc26822185)

[Hình 1.12 Kết quả sai số RMSE 14](#_Toc26822186)

[**Bảng 3: Nhắc lại về Utility Matrix** 16](#_Toc26822187)

[**Bảng 4: Một ví dụ khác về utility matrix** 17](#_Toc26822188)

[**Bảng 5: Một ultility matrix đã được chuẩn hóa** 17](#_Toc26822189)

[**Bảng 6: Similarity giữa các user** 18](#_Toc26822190)

[**Bảng 7: Thay các giá trị chưa đánh giá (0) bằng các giá trị dự đoán** 19](#_Toc26822191)

[Hình 1.13 Khai báo và khởi tạo lớp CF 21](#_Toc26822192)

[Hình 1.14 Hàm thêm đối tượng vào utility matrix 21](#_Toc26822193)

[Hình 1.15 Hàm tính similarity 21](#_Toc26822194)

[Hình 1.16 Hàm chuẩn hóa dữ liệu 22](#_Toc26822195)

[Hình 1.17 Thực hiện chuẩn hóa và tính similarity khi có dữ liệu mới 22](#_Toc26822196)

[Hình 1.18 Hàm dự đoán 23](#_Toc26822197)

[Hình 1.19 Hàm đưa ra gợi ý 24](#_Toc26822198)

[Hình 1.20 In kết quả gợi ý ra màn hình 24](#_Toc26822199)

[Hình 1.21 Chạy và test RMSE mô hình trên tập dữ liệu 25](#_Toc26822200)

[Hình 1.22 Kết quả RMSE sau khi chạy User-User Collaborative Filtering 25](#_Toc26822201)

[Hình 1.23 Chạy và test RMSE Item-Item Collaborative Filtering 25](#_Toc26822202)

[Hình 1.24 Kết quả RMSE sau khi chạy Item-Item Collaborative Filtering 25](#_Toc26822203)

[**Bảng 8: Khái quát nội dung chương 1** 26](#_Toc26822204)

[Hình 2.2 Các loại hình tấn công hệ thống gợi ý [1] 30](#_Toc26822205)

[Hình 2.3 Ví dụ về push attack [2] 32](#_Toc26822206)

[Hình 2.4 Ví dụ về nuke attack [2] 33](#_Toc26822207)

[Hình 2.5 Cấu trúc 1 profile của Segment attack 37](#_Toc26822208)

[Hình 2.5 Cấu trúc 1 bộ profile dùng trong Bandwagon attack [2] 39](#_Toc26822209)

[Hình 2.6 Bảng tóm tắt các thể loại tấn công [2] 41](#_Toc26822210)

[Hình 4.1 Đọc file “ua\_attack.base” có chứa rating giả 47](#_Toc26822211)

[Hình 4.2 Hàm tính giá trị RDMA cho mỗi user 47](#_Toc26822212)

[Hình 4.3 Quét và tính RDMA cho từng user ở file gốc “ua.base” 47](#_Toc26822213)

[Hình 4.4 Quét và tính RDMA cho từng user ở file tấn công “ua\_attack.base” 48](#_Toc26822214)

[Hình 4.5 Thực hiện đánh giá RDMA 48](#_Toc26822215)

[Hình 4.6 Kết quả của chương trình sử dụng RDMA 49](#_Toc26822216)

[Hình 4.7 Kết quả của chương trình sử dụng WDMA 49](#_Toc26822217)

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:

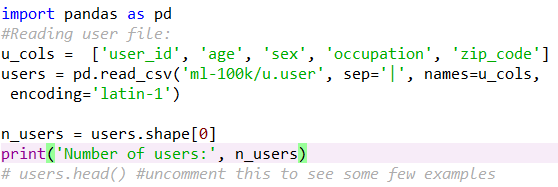
Trong đó:

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

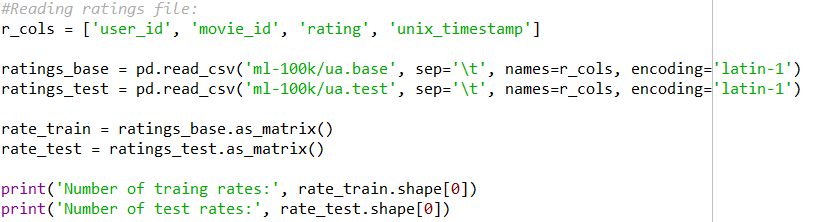
Bộ dữ liệu sử dụng: MovieLens 100k, năm 1998, bởi GroupLens.

Đọc file dữ liệu user:



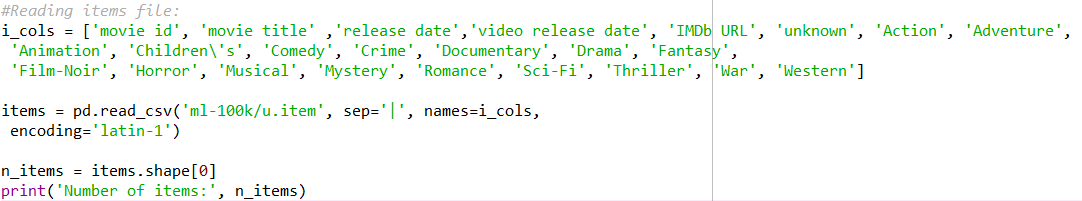
Hình 1.1 Code đọc file dữ liệu user bằng Python

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).



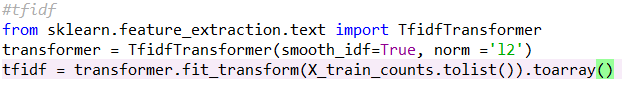
Hình 1.3 Xây dựng item profile

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.



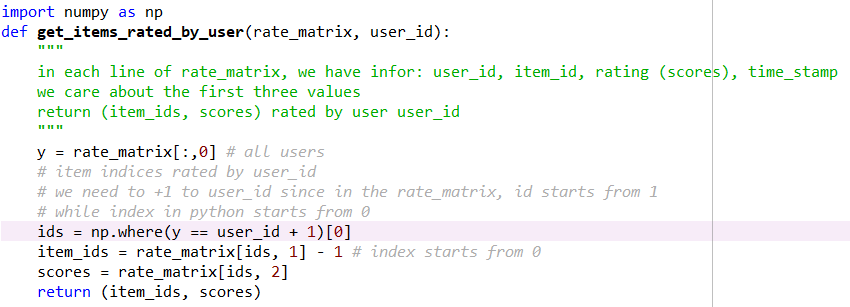
Hình 1.4 Lấy thông tin thể loại phim của từng item

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



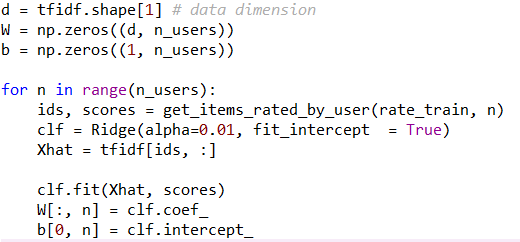
Hình 1.5 Xây dụng bộ feature vector cho item

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:



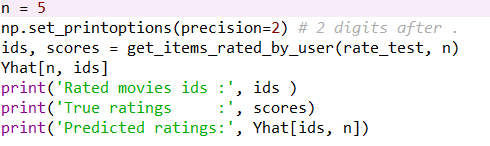
Hình 1.7 Tính Ridge Regression cho từng user

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



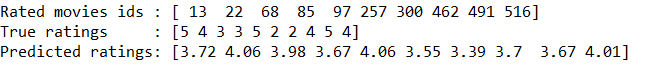
Hình 1.8 Dự đoán các giá trị

Thực hiện chạy chương trình trên user có id = 5



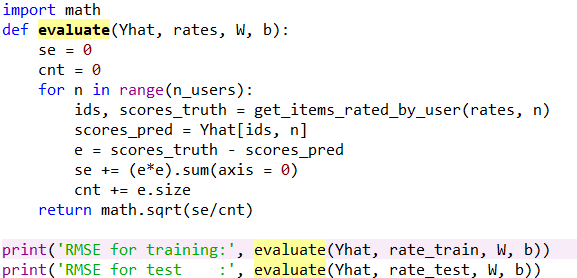
Hình 1.9 Chạy chương trình hệ thống trên user có id = 5

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.

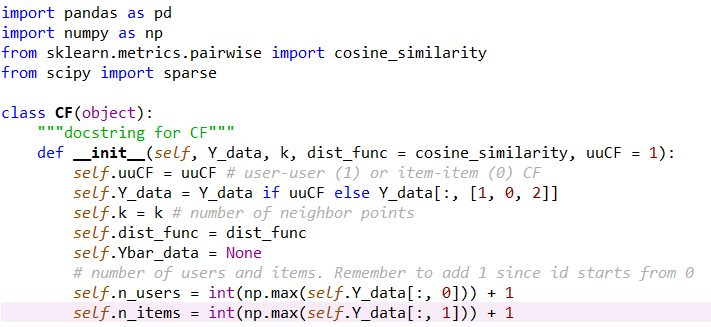
Do:

* 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

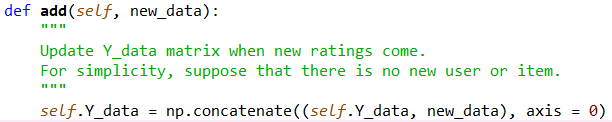
* + 1. Lập trình Neighborhood-Based Collaborative Filtering trên Python

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.



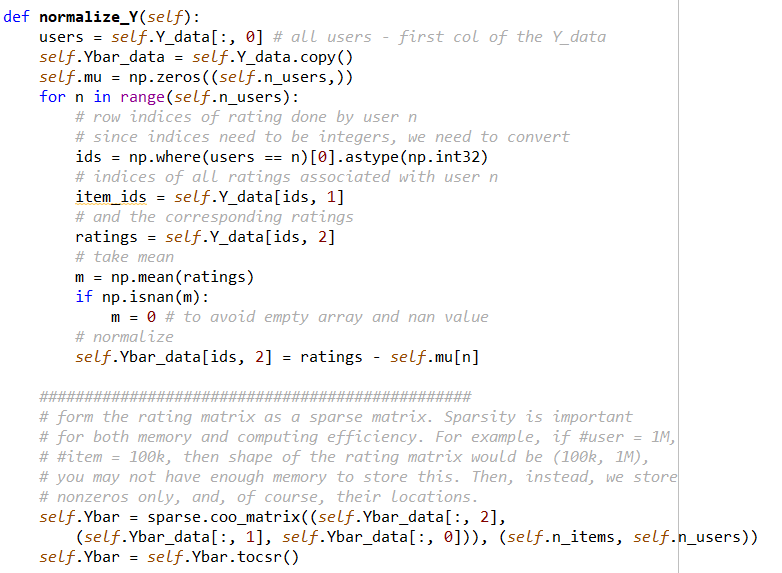
Hình 1.14 Hàm thêm đối tượng vào utility matrix

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



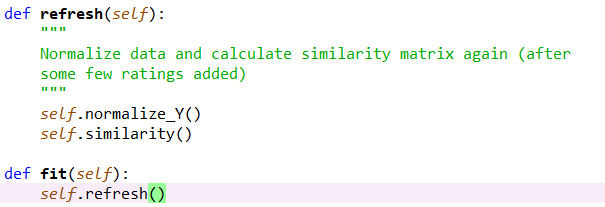
Hình 1.15 Hàm tính similarity

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



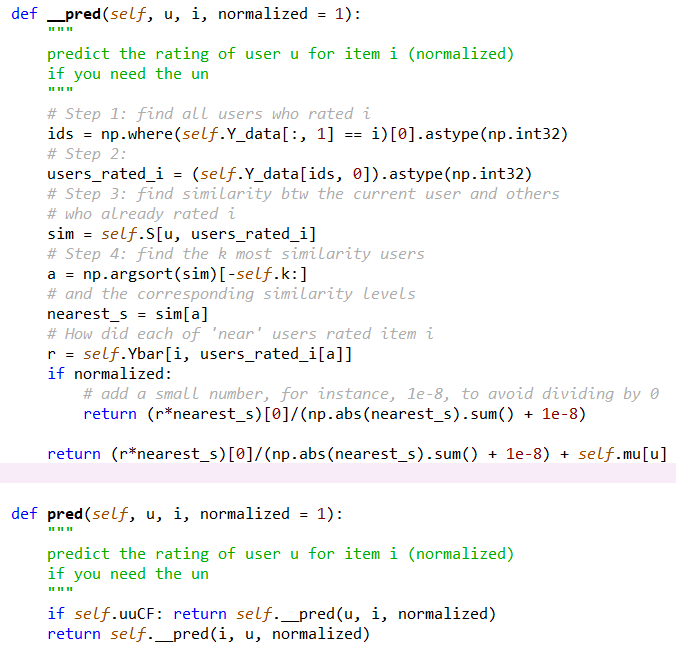
Hình 1.16 Hàm chuẩn hóa dữ liệu

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



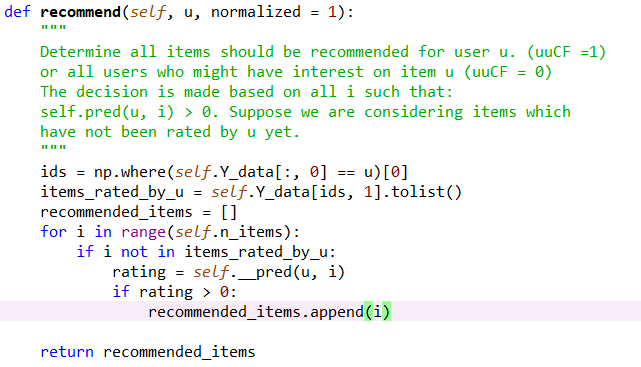
Hình 1.17 Thực hiện chuẩn hóa và tính similarity khi có dữ liệu mới

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



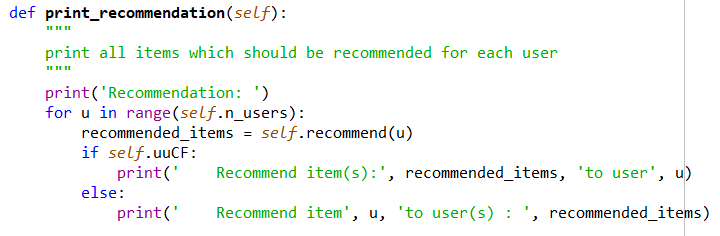
Hình 1.18 Hàm dự đoán

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



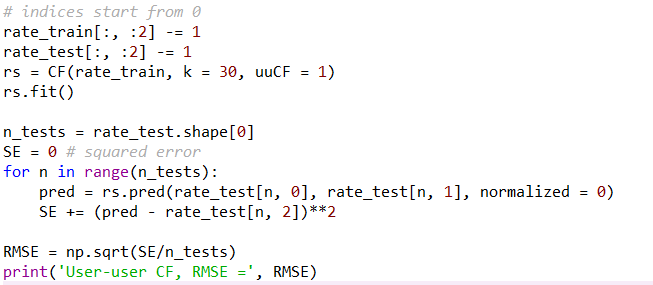
Hình 1.19 Hàm đưa ra gợi ý

Cuối cùng, ta sẽ ra kết quả.



Hình 1.20 In kết quả gợi ý ra màn hình

Ta sẽ chạy thử chương trình.

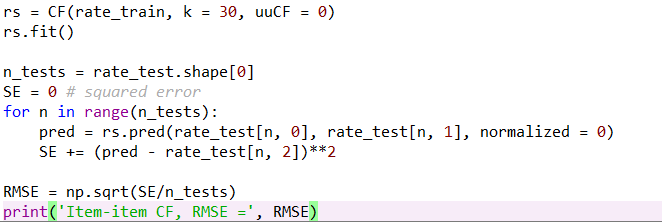


Hình 1.21 Chạy và test RMSE mô hình trên tập dữ liệu



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

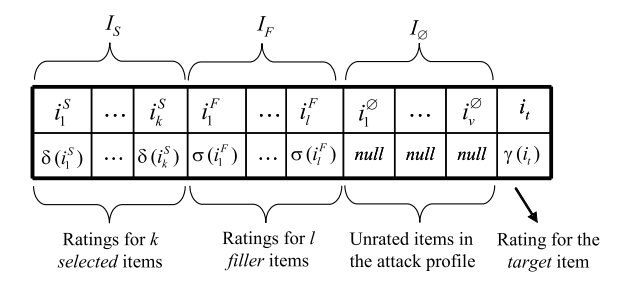
Với sự tiến bộ của các hệ thống gợi ý, ngày nay ta có rất nhiều kỹ thuật để có thể làm tăng sự chú ý hay làm giảm sự chú ý đến 1 item. Sau khi tìm hiểu về hệ thống gợi ý ở chương 1, ta thấy cốt lõi của hệ thống gợi ý chính là dựa vào tập dữ liệu đầu vào, từ tập dữ liệu mà ta đem đi tiến hành tiến hành 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]

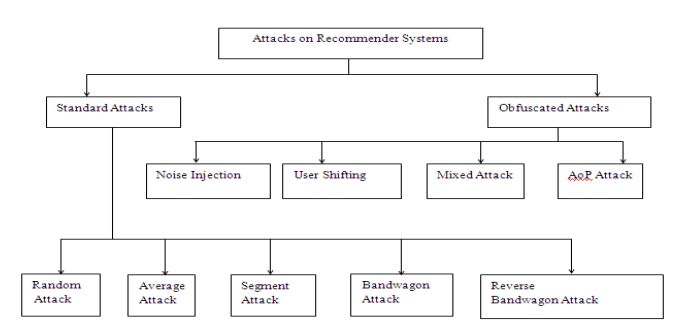
Ta có thể thấy rằng user thật thường có tính khách quan, gồm 2 phần: phần thứ nhất là các item đã được đánh giá (thường có rất ít item được đánh giá) và phần thứ hai là các item chưa được đánh giá (thường còn rất nhiều item chưa được đánh giá), vì không có bất kỳ user thật nào có thể đánh giá phần lớn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

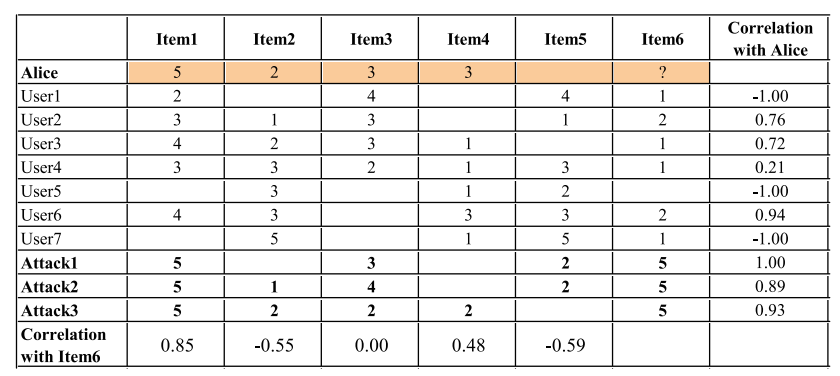
Yếu tố thứ hai ảnh hưởng đến một cuộc tấn công đó chính là ý định của kẻ tấn công. Dựa trên ý định của kẻ tấn công ta có 2 loại kiểu tấn công, đó là “push” và “nuke”. Kẻ tấn công có thể chèn các hồ sơ giả để nâng điểm đánh giá (push) hay để làm giảm đánh giá của một sản phẩm hay item nào đó. Mục tiêu 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:

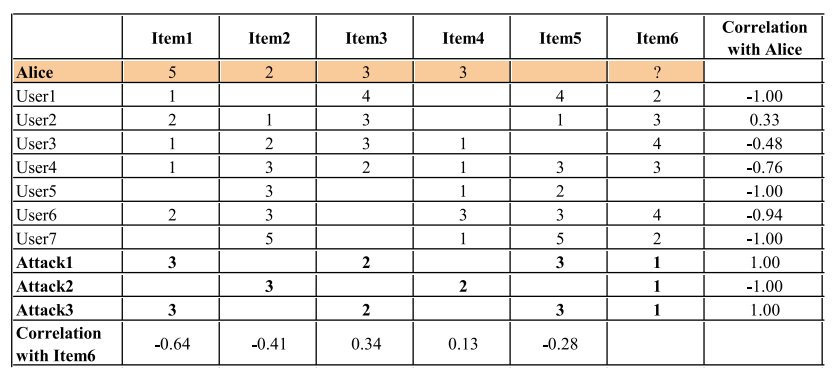


Hình 2.3 Ví dụ về push attack [2]

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).

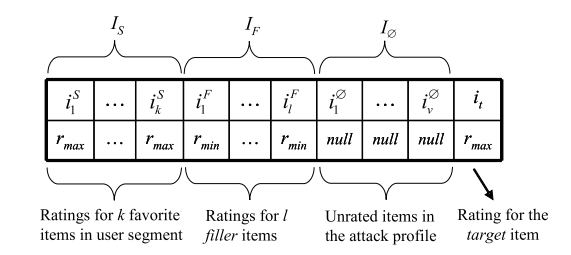
Chúng ta giả định rằng kẻ tấn công có một số item cụ thể mà thủ phạm muốn hệ thống đề xuất vì mục đích cá nhân. Kẻ tấn công, giả sử, sẽ nhận được giá trị pi “hoa hồng” hay lợi nhuận cho mỗi item i được mua. Gọi sự kiện mà hệ thống giới thiệu sản phẩm i cho người dùng là và sự kiện người dùng mua item i là Bu,i. Xác suất người dùng sẽ mua i nếu được khuyến nghị, chúng ta có thể biểu diễn dưới dạng xác suất có điều kiện: . Với tất cả người dùng U truy cập hệ thống trong một khoảng thời gian, lợi nhuận dự kiến sẽ là:

Kẻ tấn công hy vọng sẽ tăng lợi nhuận của hắn bằng cách tăng , xác suất hệ thống sẽ giới thiệu item cho một user u nhất định.

Tuy nhiên, không phải ai cũng sẽ quan tâm (đánh giá) item i của đối tượng. Nói cách khác, sẽ chỉ có một số phân khúc người dùng S khác biệt so với phần còn lại số người dùng N = U - S, muốn mua item i đó mà thôi, tức là:

Mục đích của thủ phạm cuối cùng vẫn là làm cách nào có thể làm tăng giá trị profit. Trong trường hợp xấu nhất , kẻ tấn công nên tập trung vào nhóm người dùng (S) thích item i hơn là nhóm người dùng không thích (N) không thích item i.

Do đó Segement attack sẽ có 2 hướng để lựa chọn. Một là cố gắng tấn công, ép hệ thống gợi ý đến nhóm người dùng N này, tuy nhiên cách này không hiệu quả vì có nhiều khả năng không những khiến nhóm người dùng N thích item i mà còn khiến nhóm người dùng này phê bình xấu tới item i, khiến mục đích của cuộc tấn công phản tác dụng. Ngoài ra, điều này còn sẽ khiến người dùng nghi ngờ về tính đúng của hệ thống, từ đó cuộc tấn công sẽ bị lộ. Còn hướng thứ hai chính là, kẻ tấn công sẽ tập trung tấn công mạnh vào nhóm người S, khiến số lượng item i đã được quan tâm mạnh, sẽ được quan tâm mạnh hơn nữa.



Hình 2.5 Cấu trúc 1 profile của Segment attack

Mô hình của segment attack như sau:

Trong đó:

* là tập các item được chọn để tấn công, được định nghĩa sẵn trong dùng để định nghĩa nhóm người dùng mà kẻ tấn công sẽ nhắm đến.
* 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 giá trị sẽ được sử dụng trong R.
* .

Sau đó, nhóm mục tiêu người dùng (segment) trong mô hình segment attack có thể được định nghĩa thành trong cơ sở dữ liệu sao cho:

Trong đó là xếp hạng của item i trong profile và là ngưỡng xếp hạng tối thiểu được chỉ định trước.

* + 1. Bandwagon attack (Popular attack)

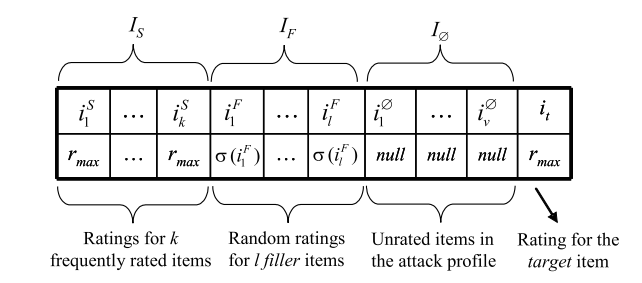
Mục tiêu của cuộc tấn công bandwagon là tạo mối liên kết (tạo sự ảnh hưởng) từ các item bị tấn công đến một số lượng nhỏ các item khác được đánh giá thường xuyên hoặc có tính liên quan đến item bị tấn công.

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. Love/Hate attack

Các item được chọn mà kẻ tấn công nhắm vào sẽ được gán giá trị nhỏ nhất. Ngược lại các giá trị còn lại mà kẻ tấn công không nhắm vào sẽ được gán giá trị lớn nhất.

Trong đó:

* .
* là tập các item được chọn ngẫu nhiên từ tập , ta đặt là tỉ lệ các item filler được định nghĩa trong .
* với là giá trị sẽ được sử dụng trong R.
* .
  + 1. User shifting

Trong các loại tấn công này, về cơ bản thủ phạm sẽ tăng hoặc giảm tất cả các xếp hạng cho một tập hợp con các item trên mỗi hồ sơ tấn công bằng một lượng không đổi để giảm sự giống nhau giữa các hồ sơ tấn công để tránh bị phát hiện.

* + 1. Mixed attack

Trong mixed attack, tấn công sẽ được thực hiện bằng cách phối hợp nhiều hình thức tấn công lại với nhau.

* + 1. Noise injection

Loại hình tấn công này được thực hiện bằng cách thêm một số giá trị đánh giá nhiễu sao cho kết quả của phân phối chuẩn là bội số của một hằng số , được sử dụng để chi phối lượng các giá trị nhiễu được thêm vào. Các giá trị nhiễu đôi khi sẽ ảnh hưởng đến kết quả đầu ra của hệ thống.

* + 1. Average over Popular attack (AoP)

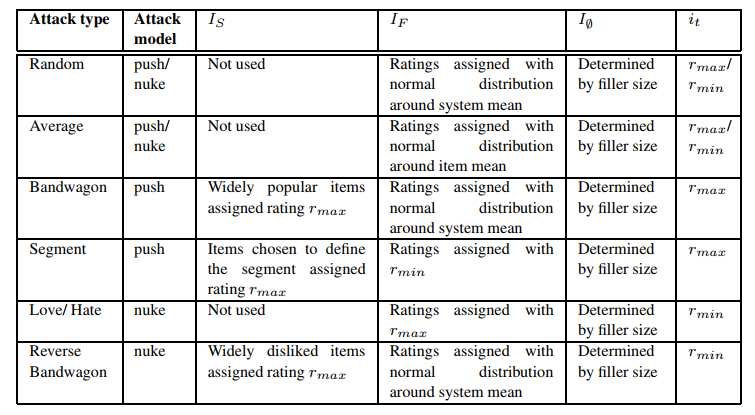
Tấn công AoP là cuộc tấn công được thiết kế để che giấu average attack bằng cách chọn các filler item có xác suất bằng nhau từ top x% phổ biến nhất từ toàn bộ cơ sở dữ liệu.

* 1. Tổng kết chương

Thông thường sẽ có 3 yếu tố chính ảnh hưởng đến 1 cuộc tấn công. Thứ nhất đó là kiến thức cũng như sự chuẩn bị của kẻ tấn công như thế nào. Kẻ tấn công càng có sự hiểu biết rộng, sâu về hệ thống thì cuộc tấn công sẽ càng có độ phức tạp cao và chi tiết. Yếu tố thứ 2 ảnh hưởng đến cuộc tấn công chính là mục đích của kẻ tấn công: kẻ tấn công muốn làm giảm hay làm tăng mức độ quan tâm của các item mục tiêu. Và yếu tố thứ 3 ảnh hưởng đến 1 cuộc tấn công đó chính là quy mô của 1 cuộc tấn công: lớn hay nhỏ, ảnh hưởng nhiều hay ảnh hưởng ít.

Vấn đề chính trong chương 2 chính là bàn luận cách làm thế nào thủ phạm có thể tấn công vào 1 hệ thống. Kẻ tấn công sẽ tấn công vào hệ thống bằng cách sử dụng các profile giả. Mỗi profile giả sẽ gồm có 4 phần: các item được kẻ tấn công lựa chọn (selected item) để thực hiện tấn công, các item được kẻ tấn công sử dụng để điều chỉnh trọng số (item filler – các item phụ), các item trung lập (không bị ảnh hưởng cũng như không được kẻ tấn công sử dụng), và các item mục tiêu (target item). Tùy vào hình thức tấn công mà số lượng item cũng như giá trị đánh giá của các item trong từng thành phần sẽ khác nhau.

Ta có thể tóm tắt cách cấu hình các thành phần theo từng loại tấn công bằng bảng sau:



Hình 2.6 Bảng tóm tắt các thể loại tấn công [2]

CHƯƠNG 3: GIẢI QUYẾT BÀI TOÁN

Chương này tài liệu xin trình bày một số cách để phòng chống cũng như bảo vệ hệ thống gợi ý khỏi các cuộc tấn công.

3.1 Các kỹ thuật nhận dạng

Gần đây đã có nhiều nghiên cứu nhằm phát hiện và ngăn chặn ảnh hưởng của các cuộc tấn công bằng cách chèn các profile giả vào hệ thống. Một số phương pháp tính toán cũng như giải thuật được sử dụng để nhận dạng các khuôn mẫu của các profile giả này. Ngoài ra còn có một thuật toán tính toán mức độ lan rộng của thước đo tương tự để từ đó có thể phát hiện và gom nhóm chung các tấn công cùng loại.

Đa phần các công cụ để nhận dạng các profile giả sẽ tính toán mức độ sai lệch giữa giá trị trung bình của các rating trước và sau khi bị tấn công. Sau khi dùng các công cụ này ta tính toán được ra một giá trị đặc biệt cho mỗi user hoặc item. Rồi từ đó ta sẽ đem đi phân loại ra đâu là profile thật, đâu là profile giả. Các profile giả thường có giá trị đặc biệt này cao hơn các profile còn lại. Tài liệu sẽ nói rõ hơn phần này ở mục các mục sau.

3.2 Các giải thuật phân loại profile

3.2.1 Rating Deviation from Mean Agreement (RDMA)

RDMA là công cụ dùng để nhận dạng các hồ sơ giả bằng cách tính độ lệch chuẩn trung bình của các item được user đánh giá sau đó sẽ được đem chia số lượng user đã đánh giá cho item đang xét.

Công thức RDMA được tính như sau:

Trong đó là số lượng item user u đã đánh giá, là giá trị đánh giá của user u dành cho item i, là số lượng giá trị đánh giá của item i xét trên toàn bộ user và là giá trị đánh giá trung bình của item i xét trên toàn bộ user.

3.2.2 Weighted Deviation from Mean Agreement (WDMA)

WDMA có chức năng tương tự như RDMA, điểm khác đó là RDMA sử dụng trong phép chia, thì WDMA sử dụng để làm tăng thêm độ chính xác của công thức.

Công thức của WDMA như sau:

Các kí hiệu trong công thức tương tự với định nghĩa được viết ở RDMA.

3.2.3 Degree of Similarity with Top Neighbors (DegSim)

DegSim nắm bắt được sự tương đồng trung bình của một nhóm hồ sơ k-NearestNeighbor. Vì các hồ sở giả thường có tỉ lệ cao nằm cạnh nhau, vì vậy DegSim có khả năng gom nhóm lại các hồ sơ này.

3.2.4 Length variance

Length variance được giới thiệu để nắm bắt vùng phủ (số lượng đánh giá) các giá trị đánh giá của các profile khác biệt như thế nào so với vùng phủ trung bình của các item khác trong cơ sở dữ liệu. Vì đa phần, một user thật không có khả năng đánh giá hết một lượng lớn các item, chỉ có những profile giả được tạo và chạy máy tự động mới có khả năng đó. Công thức của length variance được tính như sau:

Trong đó là số lượng đánh giá trung bình của các item trên toàn user.

3.2.5 Mean variance

Trong mỗi profile sẽ có các khu vực có độ tương đồng cao với các khuôn mẫu của các profile giả. Vì thế chúng ta có thể chia 1 profile ra thành 3 phân vùng: phân vùng chứa các item có khả năng cao sẽ trở thành selected item, phân vùng chứa các item có khả năng cao sẽ trở thành filler item, phân vùng chứa các item có khả năng cao sẽ không rơi vào tầm ngắm của thủ phạm.

Thông thường các item được đánh giá cao, rất cao (hoặc thấp, rất thấp) sẽ thường có khả năng rất lớn sẽ trở thành các mục tiêu của kẻ tấn công, vì vậy ta sẽ khoanh vùng các item này trước. Sau đó, ta sẽ tiếp tục khoanh vùng các item có khả năng lớn sẽ trở thành các filler item của thủ phạm.

Chúng ta sẽ cứ khoanh vùng cho đến khi nào giá trị MeanVar đạt giá trị cực đại cho push attack vào đạt giá trị cực tiểu khi ta muốn tránh nuke attack. Công thức MeanVar được tính như sau:

Trong đó là profile của user u, là item có năng cao trở thành mục tiêu, là giá trị đánh giá của user u dành cho item i, là giá trị đánh giá trung bình của item i xét trên toàn bộ user. là số lượng item được đánh giá trong profile . Sau đó chúng ta sẽ chọn item target t từ tập cho đến khi nào MeanVar(t,u) đạt cực tiểu.

3.2.6 Filler Mean Target Difference (FMTD)

FMTD sẽ giúp ta nhận dạng được các trường hợp bandwagon attack hoặc segment attack. Trong các dạng tấn công này, là các item có chứa các giá trị đánh giá cực đại hoặc cực tiểu trong profile của user u. Các item còn lại của user u sẽ được gán thành . Công thức tính FMTD như sau:

3.2.7 Target Model Focus (TMF)

Thông thường trong 1 cuộc tấn công, kẻ tấn công sẽ thêm vào hệ thống rất nhiều profile giả. Do đó, nếu một hệ thống bị tấn công, có khả năng sẽ có nhiều hồ sơ tấn công nhắm vào cùng một item. Để nắm bắt được điều này, ta sẽ sử dụng TMF. Thuộc tính này tận dụng phân vùng được xác định bởi các thuộc tính dành riêng cho mô hình để phát hiện nồng độ của các mục tiêu. Sử dụng các phân vùng này, thuộc tính TMF tính toán mức độ phân vùng của một cấu hình đã cho tập trung vào các item chung cho các phân vùng tấn công khác. Do đó, thuộc tính TMF cố gắng đo lường khả năng 1 profile có thể trở thành mục tiêu. Để tính toán TMF, đặt qi,m là tổng số lần mỗi item i được được đánh giá trong các profile thuộc nhóm Pu,T được sử dụng trong phân vùng m cho các thuộc tính dành riêng cho mô hình. Đặt Tu là liên kết của tất cả các item được xác định cho người dùng u trong bất kỳ tập hợp đích Pu,T nào được sử dụng bởi các thuộc tính dành riêng cho mô hình. TMF được tính như sau:

3.3 Tổng kết chương

Trong chương 3 ta đã tìm hiểu một số cách để có thể nhận biết được các profile giả. Có 2 cách chung. Một là sử dụng các công thức, tính toán, rồi từ đó dựa vào sự cách biệt giá trị giữa profile thật và profile giả để phân biệt (RDMA, WDMA, Length Variance. Hai là sử dụng các công thức, tính toán, rồi từ đó khoanh các đối tượng nhạy cảm, sau đó điều chỉnh và cuối cùng tính lại công thức, nếu giá trị của công thức tối ưu thì hệ thống về cơ bản có thể an toàn (Mean Variance, FMTD, TMF).

CHƯƠNG 4: THỰC NGHIỆM

Chương này ta sẽ áp dụng các lý thuyết ở các phần trên vào bài toán ví dụ.

4.1 Dữ liệu đầu vào

Dữ liệu gốc mà tài liệu sử dụng có nguồn gốc từ trang web <https://grouplens.org/datasets/movielens/>.

Để mô phỏng cho cuộc tấn công ta sẽ tạo ra 1 file có chứa các profile giả. Ta đặt tên file này là “ua\_attack.base” có chứa rating giả dựa trên file gốc là “ua.base”.

Cấu trúc file “ua\_attack.base” như sau:

* Các user có id từ 1 cho đến 893, ta giữ nguyên các đánh giá gốc như trong file “ua.base”.
* Các user có id từ 893 cho 943 (50 user), ta sẽ chỉnh các đánh giá lại thành các rating giả. Mỗi user sẽ có 50 đánh giá. Trong đó có đánh giá item id 19 sẽ là 1. Còn 49 item id còn lại sẽ được chọn ngẫu nhiên để đánh giá. Các giá trị đánh giá của 49 item còn lại sẽ là 1. (Dựa trên mô hình Love/Hate Attack).

4.2 Phương thức sử dụng

Ta sẽ sử dụng tính giá trị RDMA cho từng user.

Sau đó ta tìm giá trị RDMA trung bình. Từ đó ta sẽ so từng RDMA, nếu giá trị RDMA nào cao hơn so với RDMA trung bình, ta sẽ coi đó là profile giả.

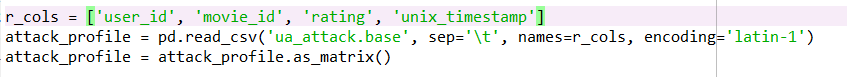
Nếu profile nào là thật mà có giá trị RDMA nhỏ hơn RDMA trung bình, ta sẽ đếm giá trị true positvie lên 1. Ngược lại ta sẽ tính false positive lên 1.

Nếu profile nào là giả mà có trị RDMA cao hơn RDMA trung bình, ta sẽ đếm giá trị true positive lên 1. Ngược lại ta sẽ tính false negative lên 1.

Cuối cùng ta sẽ dùng 2 giá trị precision và recall để đánh giá công thức.

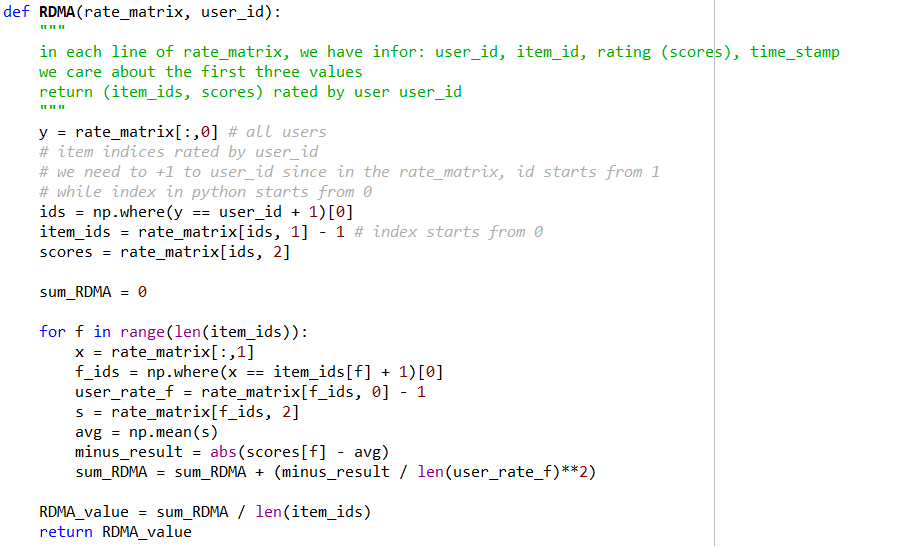
4.3 Kết quả và đánh giá

Đầu tiên ta cho đọc file “ua\_attack.base” có chứa các rating giả.



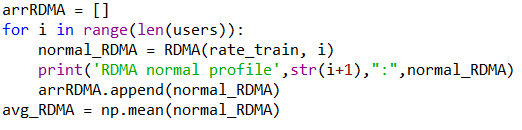
Hình 4.1 Đọc file “ua\_attack.base” có chứa rating giả

Sau đó, ta viết hàm để tính RDMA cho từng user.



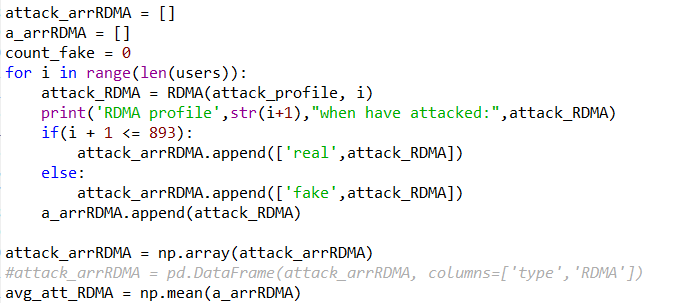
Hình 4.2 Hàm tính giá trị RDMA cho mỗi user

Sau đó ta sẽ chạy và xem RDMA của từng user với dữ liệu gốc trước.



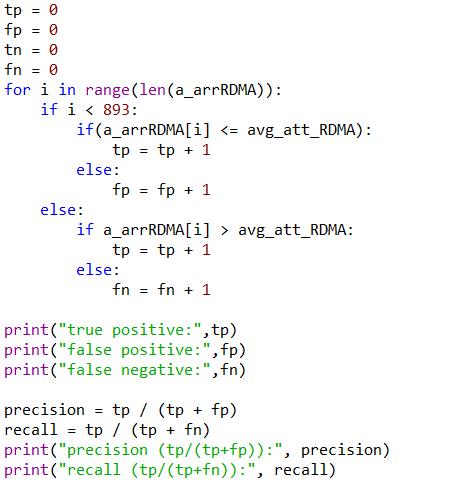
Hình 4.3 Quét và tính RDMA cho từng user ở file gốc “ua.base”

Tiếp đến ta sẽ chạy và xem RDMA của từng user với dữ liệu tấn công mà ta tạo ở file “ua\_attack.base”.



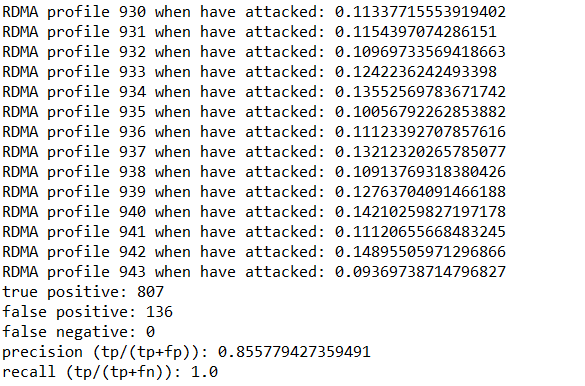
Hình 4.4 Quét và tính RDMA cho từng user ở file tấn công “ua\_attack.base”

Ta đi đếm các giá true\_positive, false\_positive, false\_negative và xuất kết quả precision và recall ra màn hình. Cuối cùng ta tính precision và recall.



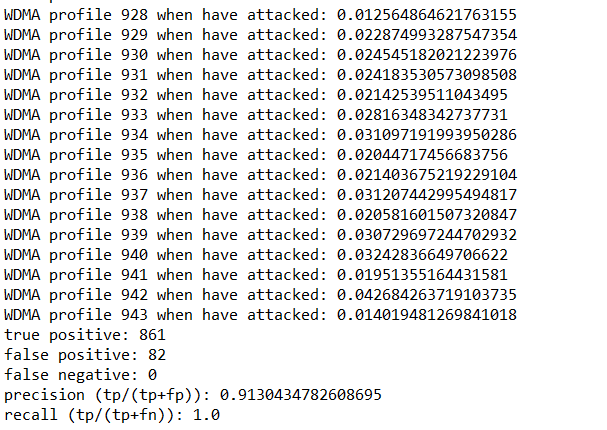
Hình 4.5 Thực hiện đánh giá RDMA

Sau khi chạy chương trình ta được kết quả:



Hình 4.6 Kết quả của chương trình sử dụng RDMA

Tương tự ta làm cho WDMA. Sau khi chạy chương trình ta được:



Hình 4.7 Kết quả của chương trình sử dụng WDMA

Sau khi chạy chương trình ta có rút ra một vài nhận xét sau:

* Cả RDMA và WDMA đều có tỉ lệ chính xác cao. (precision của RDMA là 0.855779427359491, precision của WDMA là 0.9130434782608695).
* WDMA cho precision = 0.9130434782608695 cao hơn so với precsion = 0.855779427359491. Tức là WDMA sẽ cho kết quả chính xác hơn vì như ở chương 3 ta đã nói do WDMA sẽ dùng l2 thay vì là l.
* Tỉ lệ bỏ sót của cả 2 đều rất thấp vì recall của cả hai đều cao, recall của cả hai đều bằng 1.

4.4 Tổng kết

Như vậy ta đã tìm hiểu xong cơ bản về 1 recommendation system.

Từ chương 1 ta đã tìm hiểu được có 2 loại recommendation chính là content-based và collaborative filtering. Cách thức vận hành và cách thức đánh giá cho từng recommendation system.

Chương 2 ta đã tìm hiểu một recommendation system sẽ bị tấn công bằng các profile giả do thủ phạm đưa vào hệ thống. Ngoài ra ta còn biết được cấu trúc của 1 profile giả sẽ gồm có 4 phần: phần thứ nhất gồm các item được lựa chọn để tấn công (selected item), phần thứ 2 gồm các item được sử dụng để phối hợp và điều chỉnh tấn công (điều kiện trọng số ở đầu ra hệ thống) (filler item), phần thứ 3 gồm các item không được đánh giá (null item), và phần thứ 4 chính là các item mục tiêu (target item) mà thủ phạm nhắm tới. Đồng thời trong chương 2 ta cũng đã tìm hiểu và hiểu rõ một số loại hình tấn công phổ biến, cơ bản lên recommendation system.

Chương 3 ta đã tìm hiểu được một số kỹ thuật tính toán RDMA, WDMA, v.v. để có thể nhận biết được các profile giả. Hoặc ta sử dụng các kỹ thuật tính toán này để dần dần đem đi cải thiện hoặc tìm phương án phòng hờ cho hệ thống (Mean variance, FTMD, TMF,v.v.).

Chương 4, ta đã áp dụng, cũng như đánh giá thực hành được các lý thuyết mà ta đã đề cập ở chương 2, chương 3.

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.