**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**



**ĐỒ ÁN MÔN HỌC**

**ĐỀ TÀI: PHÂN TÍCH DỮ LIỆU THƯƠNG MẠI & HỆ GỢI Ý**

Giảng viên hướng dẫn : Thân Quang Khoát

Bộ môn : Nhập môn KHDL

Mã lớp học : 128735

|  |  |
| --- | --- |
| Lê Thành Long | 20194099 |
| Ninh Văn Nghĩa | 20190060 |
| Lê Đại Thắng | 20194166 |

**HÀ NỘI, 12/2021**

**LỜI CẢM ƠN**

*Quá trình thực hiện đề tài, nhóm đã được trao đổi với thầy cùng anh trợ giảng, từ đó nhận được nhiều sự giúp đỡ và góp ý thiết thực. Xin gửi lời cảm ơn đến thầy và anh! Năm mới chúc mọi người sức khoẻ và thành công !*

**TÀI LIỆU THAM KHẢO & THƯ VIỆN SỬ DỤNG**

**Tài liệu tham khảo**

*He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. (2017). Neural Collaborative Filtering. Proceedings of the 26th International Conference on World Wide Web. [1]*

*Tran, Q., Tran, L., Hai, L.C., Linh, N.V., & Than, K. (2021). From Implicit to Explicit feedback: A deep neural network for modeling sequential behaviours and long-short term preferences of online users. ArXiv, abs/2107.12325. [2]*

*Vu, T., Nguyen, D.Q., Nguyen, D.Q., Dras, M., & Johnson, M. (2018). VnCoreNLP: A Vietnamese Natural Language Processing Toolkit.*ArXiv, abs/1801.01331*. [3]*

[*https://github.com/yihong-chen/neural-collaborative-filtering*](https://github.com/yihong-chen/neural-collaborative-filtering) *[4]*

**Thư viện sử dụng**

*Pytorch, Keras, Tensorflow, Pandas, Numpy, VnCoreNLP, NLTK, Matplotlib*

**MỤC LỤC**

[1. Mô tả bài toán 6](#_Toc93261921)

[2. Các bước thực hiện 6](#_Toc93261922)

[3. Thu thập dữ liệu 6](#_Toc93261923)

[4. Lưu trữ dữ liệu 7](#_Toc93261924)

[5. Làm sạch và tiền xử lý dữ liệu 7](#_Toc93261925)

[6. Phân tích và trực quan hóa dữ liệu 8](#_Toc93261926)

[7. Dự đoán lượt mua sản phẩm 13](#_Toc93261927)

[7.1 Tổng quan 13](#_Toc93261928)

[7.2 Random Forest Regression 15](#_Toc93261929)

[7.2.1 Dự đoán sử dụng toàn bộ features (one hot + numeric) 15](#_Toc93261930)

[7.2.2 Khi loại bỏ avg\_rating và n\_reviews 16](#_Toc93261931)

[7.3 Linear Regression 16](#_Toc93261932)

[7.3.1 Chỉ sử dụng các numerical features: 16](#_Toc93261933)

[7.3.2 Linear Regression sử dụng Mean Encoding cho các categorical features: 18](#_Toc93261934)

[7.4 Neural Network 18](#_Toc93261935)

[7.5 Kết Luận 19](#_Toc93261936)

[8. Bài toán dự đoán rating 19](#_Toc93261937)

[8.1 Phương pháp phân rã ma trận (matrix factorization) 19](#_Toc93261938)

8.2 Phương pháp content-based…………………………………………………………………… 24

[8.3 Ứng dụng 1 số neural network đơn giản cho bài toán rating prediction…….26](#_Toc93261939)

[8.3.1 GMF - Generalized Matrix Factorization 26](#_Toc93261940)

[8.3.2 MLP - Multilayer perceptron 29](#_Toc93261941)

[8.3.3 Áp dụng cả GMF và MLP 32](#_Toc93261942)

[9. Bài toán hệ gợi ý 33](#_Toc93261943)

[9.1 Yêu cầu bài toán 31](#_Toc93261944)

[9.2 Phương pháp sử dụng 31](#_Toc93261945)

[9.2.1 GMF 32](#_Toc93261946)

[9.2.2 MLP 32](#_Toc93261947)

[9.2.3 NMF 33](#_Toc93261948)

[9.3 Chi tiết cài đặt 33](#_Toc93261949)

[9.3.1 Huấn luyện 33](#_Toc93261950)

[9.3.2 Xử lý tập dữ liệu 34](#_Toc93261951)

[9.3.3 Đánh giá 35](#_Toc93261952)

[9.3.4 Thông số cài đặt 35](#_Toc93261953)

[9.4 Thử nghiệm 36](#_Toc93261954)

[9.4.1 Eta 36](#_Toc93261955)

[9.4.2 Latent Size 36](#_Toc93261956)

[9.4.3 Quá trình chạy với 50 epochs 37](#_Toc93261957)

[9.4.4 Negative Sampling 38](#_Toc93261958)

[9.4.5 Top-K 39](#_Toc93261959)

[9.4.6 Đánh giá sau cùng 39](#_Toc93261960)

[9.5 Một vài Case Study 40](#_Toc93261961)

[9.5.1 Case Study 1 40](#_Toc93261962)

[9.5.2 Case Study 2 41](#_Toc93261963)

[9.5.3 Case Study 3 42](#_Toc93261964)

[10. Kết luận 43](#_Toc93261965)

# 1. Mô tả bài toán

Tại Việt Nam, các sàn thương mại điện tử đang phát triển rất mạnh trong vài năm trở lại đây. Sự bùng nổ này tạo ra một lượng lớn dữ liệu vô cùng phong phú và có thể được khai thác, sử dụng cho rất nhiều bài toán hiện nay. Một trong những bài toán mà được người người sử dụng các sàn thương mại điện tử này rất quan tâm đó là bài toán dự đoán sản phẩm được mua, dự đoán rating, hệ gợi ý, dự đoán cảm xúc,… Do đó trong dự án này, nhóm thực hiện thu thập dữ liệu trên sàn thương mại điện tử tiki.vn và thực hiện phân tích, xây dựng hệ gợi ý, hệ dự đoán rating.

# 2. **Các bước thực hiện**

* Crawl dữ liệu
* Lưu trữ dữ liệu
* Làm sạch, tiền xử lý và tích hợp dữ liệu
* Phân tích và trực quan hóa dữ liệu
* Xây dựng mô hình bài toán dự đoán lượt mua sản phẩm
* Xây dựng hệ dự đoán rating
* Xây dựng hệ gợi ý

# 3. Thu thập dữ liệu

Nguồn dữ liệu đã thu thập được lấy từ trang thương mại điện tử nổi tiếng tiki.vn . Mặc dù trên mạng cũng có rất nhiều bộ dữ liệu về thương mại điện tử tuy nhiên bộ datasets của nhóm rất dồi dào về các thông tin chi tiết sản phẩm và thông tin rating và đặc biệt các sản phẩm trên bộ dữ liệu của nhóm toàn là các sản phẩm mới mới đồng nghĩa là các review ở các bộ sản phẩm này cũng là gần đây.

Thư viện dùng để thu thập dữ liệu là BeautifulSoup kết hợp với API của tiki.

Dữ liệu thu thập được gồm 4 bộ chính item/detail\_list.csv, item/overall\_list.csv và rating/rating\_list.csv . Với ý nghĩa của các bộ dữ liệu lần lượt là thông tin chi tiết của sản phẩm, các thông số tổng quan về sản phẩm và thông tin bình luận đối với sản phẩm. Tập dữ liệu này được đính kèm trong phần source codes của nhóm.

# 4. Lưu trữ dữ liệu

Dữ liệu được lưu trữ trên cụm hadoop được cấu hình với 1 primery namenode + 1 secondery namenode + 2 datanode. Lý do nhóm chọn cách lưu trữ này bởi hệ sinh thái hadoop có nhiều ưu điểm trong lưu trữ và xử lý dữ liệu lớn như tính sẵn sàng, tính chịu lỗi tốt, kết hợp được nhiều tài nguyên tính toán,…

*Một vài hình ảnh minh hoạ cấu hình cụm và lưu trữ dữ liệu*

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application, table, Excel

Description automatically generated

*Graphical user interface, application, table, Excel

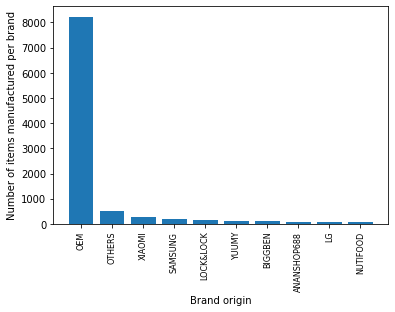
Description automatically generated*

# 5. Làm sạch và tiền xử lý dữ liệu

***Các bước làm sạch và tiền xử lý dữ liệu:***

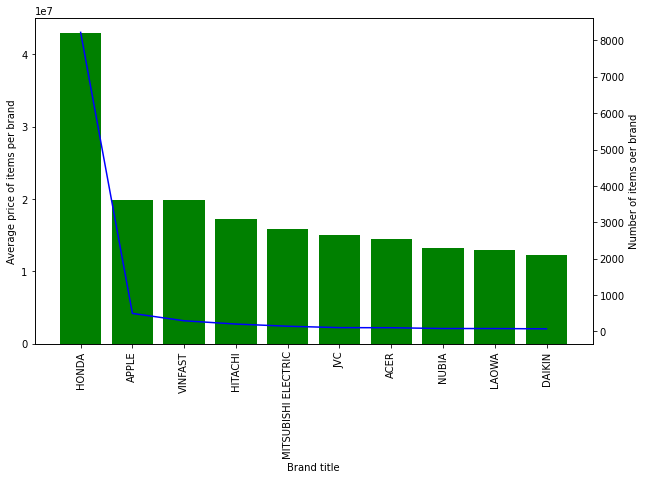
* Loại bỏ các hàng trùng nhau
* Loại bỏ các hàng trong detail\_list.csv có trường price <= 0 (kiểm tra dữ liệu hợp lệ)
* Thực hiện tách các tag, hay subcategories với từng item
* Chuyển đổi dữ liệu dạng Unix Timestamp sang Datetime để phân tích
* Lowercase với một số trường là string
* Chuyển đổi userID thành index và xoá tên người dùng gốc 🡪 Bảo mật thông tin người dùng
* One-hot encoding 🡪 Tạo ra các cột tương ứng với mỗi cột là giá trị 1 category, giá trị của hàng i tại cột ấy sẽ là 1 nếu sản phẩm i thuộc vào category đó và là 0 nếu ngược lại
* Một số bước tiền xử lý khác, tuỳ vào bài toán, sẽ được trình bày sau
* *Loại bỏ tất cả người dùng có lịch sử tương tác < 10, thay vào đó ta sẽ suggest cho những người này các sản phẩm nổi bật*
* *Rescale lại rating của người dùng đối với 1 số mô hình trong project. Nhóm gộp 1, 2 thành rating là 2.*

# 6. Phân tích và trực quan hóa dữ liệu



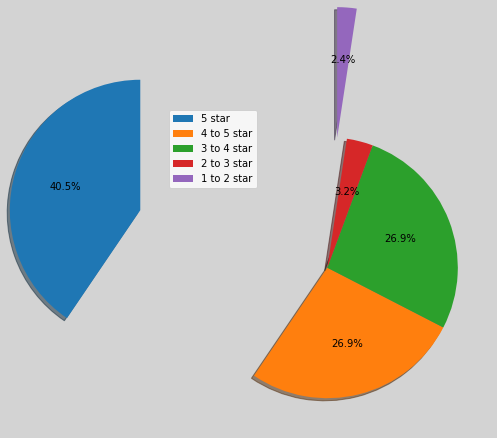
Biểu đồ số hàng hoá theo hãng sản xuất

Hầu hết các sản phẩm trong dữ liệu thu thập được đều là từ OEM (Original Equipment Manufacturing) – nhà sản xuất gốc. Còn các hãng khác ít hơn nhiều.



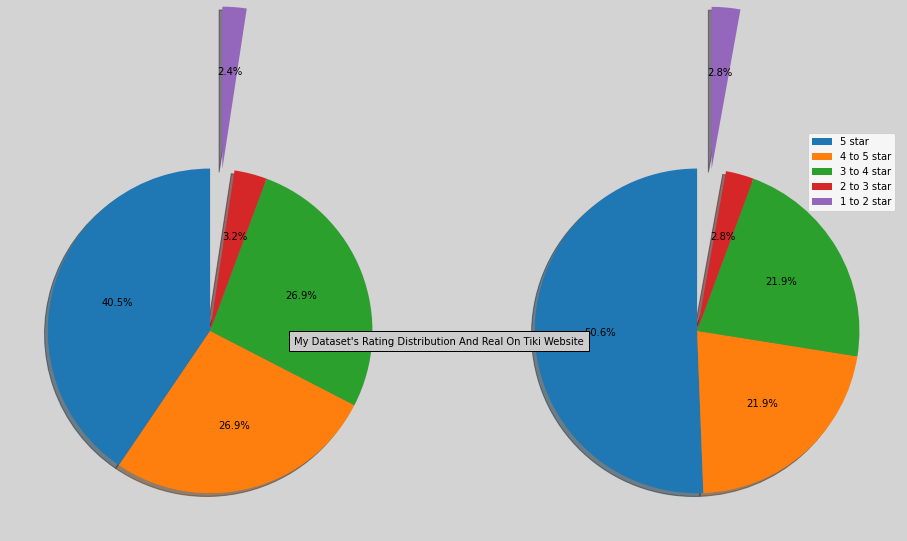
Biểu đồ số hàng hoá và giá bán trung bình theo hãng

Giá trung bình và số sản phẩm của Honda tương ứng cũng nhiều nhất, kế đến là các hãng khác như APPLE hay VINFAST, nhưng không chênh lệch nhiều.



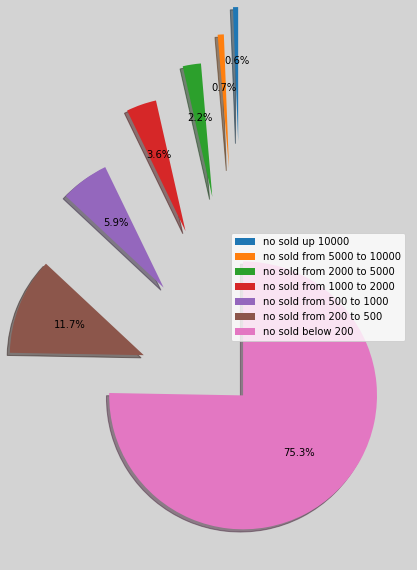
Biểu đồ phân bố rating trung bình

Rating trung bình phân bố chủ yếu vào phần 4-5 sao. Các rating từ 3 sao trở xuống chiếm trọng số thấp.



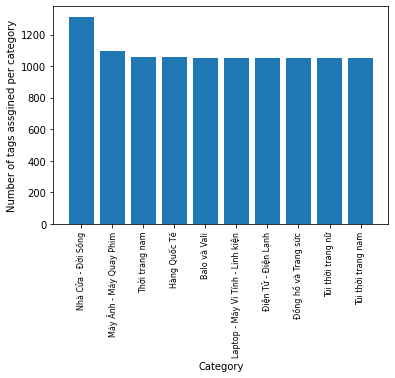
Biểu đồ so sánh tỉ lệ rating giữa dữ liệu thu thập và số liệu trên Tiki

Dựa trên biểu đồ, dễ thấy rằng tỉ lệ rating thu thập trong dữ liệu và rating thực thống kê trên Tiki khá tương đồng với nhau, biểu thị tính nhất quán giữa dữ liệu thu thập và thông số trên Tiki.



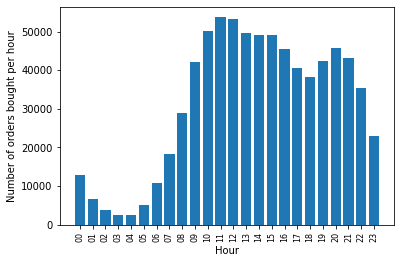
Biểu đồ thống kê số lượng sản phẩm bán ra

Hầu hết sản phẩm (75%) có số lượng bán ra dưới 200. Một số rất ít (0.6%) có số lượng bán ra trên 10000. Còn lại hầu hết từ 200-10000.



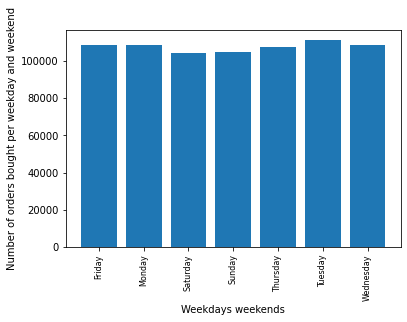
Biểu đồ thống kê lượng sản phẩm theo category

Số lượng sản phẩm ở các category khá cân bằng nhau.



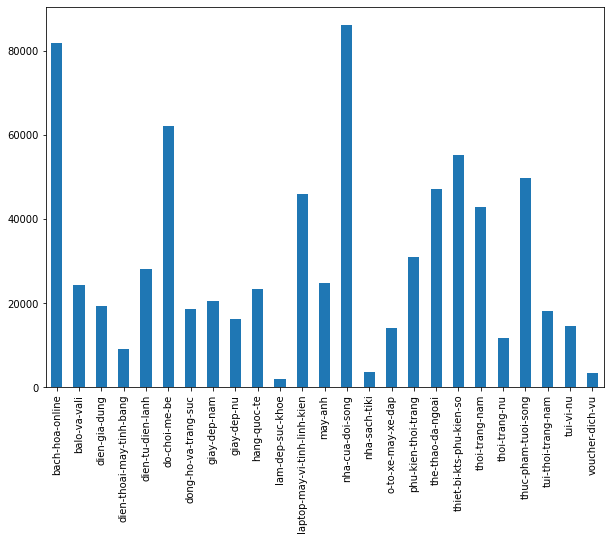
Biểu đồ thống kê thời gian review theo giờ

Thời gian review tập trung chủ yếu vào khung giờ từ 8-23 giờ. Các khung giờ rạng sáng chiếm thiểu số.



Biểu đồ thống kê thời gian review theo ngày trong tuần

Thời gian review theo ngày trong tuần phân bố khá đều, không chênh lệch vào ngày (thứ) nào.



Biểu đồ tổng quan thống kê số lượng sản phẩm theo category

Số lượng sản phẩm nhiều nhất ở nha-cua-doi-song, bach-hoa-online. Một số category có khá ít sản phẩm như lam-dep-suc-khoe hay nha-sach-tiki.

Chart, histogram

Description automatically generated

Biểu đồ thống kê tương tác (rating) của người dùng

Phần lớn người dùng tương tác với 1-15 items, con số giảm dần khi số items tăng. Hầu hết người dùng chỉ tương tác với 1-2 item.

Chart, histogram

Description automatically generated

Biểu đồ thống kê tương tác (rating) đối với item

Mỗi item có tương tác với 1-100 người dùng. Ngoại trừ 24% items có tương tác với 100 người dùng, các item còn lại ít được tương tác và xu hướng giảm dần.

Chart

Description automatically generated

Biểu đồ boxplot thống kê phân phối rating ở mỗi category

Biểu đồ boxplot khẳng định lại việc hầu hết rating ở các category đều tập trung ở mức 4-5.

# 7. Dự đoán lượt mua sản phẩm

## 7.1 Tổng quan

Metrics đánh giá mà nhóm sử dụng: ***Mean Absolute Error (MAE)*** hoặc ***Soft Interval Accuracy (SIA)***

Các features sử dụng cho việc dự đoán:

| **Feature name** | **Type** |
| --- | --- |
| category | categorical |
| price | numerical |
| avg\_rating | numerical |
| n\_rate\_5 | numerical |
| n\_rate\_4 | numerical |
| n\_rate\_3 | numerical |
| n\_rate\_2 | numerical |
| n\_rate\_1 | numerical |
| rate\_with\_img | numerical |
| discount | numerical |
| discount\_rate | numerical |
| n\_reviews | numerical |

Ta sẽ phân chia dữ liệu thành các tập train - validation - test set theo tỉ lệ: 60% - 20% - 20% và so sánh hiệu năng giữa các phương pháp học máy khác nhau.

**Các độ đo dùng để đánh giá**

MAE – Mean Absolote Error

A picture containing text, clock

Description automatically generated

MSE - Mean Squared Error

A picture containing text, clock, watch, gauge

Description automatically generated

RMSE - Root Mean Squared Error

A picture containing text, clock, watch

Description automatically generated

SIA - Soft Interval Accuracy

Một dự đoán được gói là đúng nói chênh lệch giữa dự đoán và thực tế <= siêu tham số epsilon: 

Và SIA sẽ bằng số mẫu đúng trên toàn bộ mẫu

## 7.2 Random Forest Regression

### 7.2.1 Dự đoán sử dụng toàn bộ features (one hot + numeric)

**Kết quả:**

| Dataset | MAE |
| --- | --- |
| train | 162.19768870757935 |
| val | 377.55629044530457 |
| test | 366.0717442925722 |

***Ghi chú*:** do kết quả dự đoán có thể lên đến đơn vị nghìn (lượt bán), để dễ giải thích, ta sẽ sử dụng độ đo MAE (thay vì MSE như thông thường). Do đó ta có thể đánh giá MAE trên tập test ~366 có nghĩa là mô hình dự đoán trung bình sai số khoảng 366 cho một sản phẩm trong tập test.

Bây giờ ta sẽ xét SIA metric.

* **Với epsilon = 1000 (cho phép sai số dự đoán lên đến 1000 lượt bán):**

| Dataset | SIA-1000 |
| --- | --- |
| train | 0.9723659134222706 |
| val | 0.9281437125748503 |
| test | 0.9372822299651568 |

* **epsilon = 500 (cho phép sai số dự đoán lên đến 500 lượt bán):**

| Dataset | SIA-500 |
| --- | --- |
| train | 0.9447998910971958 |
| val | 0.874251497005988 |
| test | 0.8819686411149826 |

* **epsilon = 200 (cho phép sai số dự đoán lên đến 200 lượt bán):**

| Dataset | SIA-200 |
| --- | --- |
| train | 0.8846991560032671 |
| val | 0.783886771910724 |
| test | 0.7811411149825784 |

***Nhận xét*:** ta thấy mô hình đạt được độ chính xác khá tốt. Hiệu năng trên tập val và test xấp xỉ nhau.

### 7.2.2 Khi loại bỏ avg\_rating và n\_reviews

| Dataset | MAE | SIA-1000 | SIA-500 | SIA-200 |
| --- | --- | --- | --- | --- |
| train | 490.7119 | 0.93908 | 0.88647 | 0.77151 |
| val | 745.2814 | 0.88051 | 0.80240 | 0.63745 |
| test | 695.6272 | 0.88219 | 0.80096 | 0.63371 |

***Nhận xét*:** Khi ta loại bỏ các thuộc tính như *số lượng bán ra* của các sản phẩm hay là *điểm số trung bình* người dùng cho sản phẩm đó thì kết quả của các độ đo tệ 1 cách trông thấy. Hay có nghĩa là khi giao diện tiki hiển thị các thông số này thì nó cũng phần nào *ảnh hưởng đến quyết định mua hàng* của người dùng trên sản phẩm đó

## 7.3 Linear Regression

### 7.3.1 Chỉ sử dụng các numerical features:

price, avg\_rating, n\_reviews, discount, discount\_rate, (Bỏ đi category)

**Kết quả:**

| Dataset | MAE | SIA-1000 | SIA-500 | SIA-200 |
| --- | --- | --- | --- | --- |
| train | 694.1867 | 0.89600 | 0.78934 | 0.50681 |
| val | 615.2560 | 0.89222 | 0.79205 | 5059880 |
| test | 558.5591 | 0.90026 | 0.79486 | 0.50675 |

***Nhận xét*:** Ta thấy mô hình Linear Regression có SIA thấp hơn so với khi sử dụng Random Forest khá nhiều. Hơn nữa, ta thấy mô hình có MAE trên tập val và tập test cao bất thường. Sau khi tìm hiểu nguyên nhân thì ta nhận thấy có một vài hệ số (coefficient) của một số feature có giá trị rất cao . Hơn nữa, ta nhận thấy các hệ số này thuộc về feature n\_reviews và số lượng rating theo từng điểm. Ta sẽ sử dụng L2 Regularization (Ridge) để giới hạn giá trị của các hệ số.

Kết quả khi ta sử dụng phươnng pháp Ridge Regression kết hợp cùng với việc tinh chỉnh hyperparameter α tốt nhất sử dụng phương pháp GridSearch trên tập val, ta có

**Kết quả như sau:**

**Best cross-validation MAE: 505.34610839045035 with alpha = 100**

| Dataset | MAE | SIA-1000 | SIA-500 | SIA-200 |
| --- | --- | --- | --- | --- |
| train | 681.8764 | 0.87952 | 0.78512 | 0.61469 |
| val | 602.0026 | 0.87616 | 0.77844 | 0.59771 |
| test | 541.0958 | 0.88328 | 0.78724 | 0.61476 |

***Nhận xét:***Khi sử dụng hồi quy ridge thì ta thấy kết quả tốt hơn so với ki sử dụng hồi quy tuyến tính các weight cũng nhỏ hơn so với mô hình hồi quy tuyến tính do sự ảnh hưởng của siêu tham số α

Ta thấy hiệu năng của mô hình *thấp hơn* phương pháp Random Forest khá nhiều (kể cả trên tập train). Điều này có thể là do mô hình vẫn chưa đủ phức tạp để nắm bắt được hết các thông tin từ input

### 7.3.2 Linear Regression sử dụng Mean Encoding cho các categorical features:

**Kết quả:**

| Dataset | MAE | SIA-1000 | SIA-500 | SIA-200 |
| --- | --- | --- | --- | --- |
| train | 817.7339 | 0.84025 | 0.74544 | 0.61203 |
| val | 736.7126 | 0.84186 | 0.73789 | 0.60452 |
| test | 674.4124 | 0.84495 | 0.74107 | 0.61324 |

***Nhận xét:***Ta thấy kết quả trên tập train cũng xấp xỉ kết quả trên tập val và test (ngược lại với mô hình Random Forest có độ lỗi trên tập train thấp hơn rất nhiều so với val và test). Có thể là do mô hình Linear Regression chưa phải là phù hợp nhất với tập dữ liệu của ta (*lưu ý là Linear Regression có thể đưa ra kết quả dự đoán mang giá trị âm*),đặc biệt là đối với các *categorical feature* ? (Ta sẽ kiểm chứng điều này trong phần Neural Network)

## 7.4 Neural Network

Tương tự như Linear Regression , ta sẽ tiền xử lý input bằng việc chuẩn hóa các features theo phương pháp Z-score normalization.

Ta sẽ thực nghiệm với 1 mạng Neural Network nhỏ như sau:

Input Vector -> FC(32 units) + ReLU -> FC(64 units) + ReLU -> Output (1 unit) + ReLU

Nguyên nhân ta sử dụng hàm ReLU activation lên output (thay vì Linear) là do ta muốn các giá trị dự đoán đều không âm.

Với weights của mỗi layer, ta sẽ sử dụng L2-Regularization với α =0.1 để hạn chế việc overfitting.

Hàm mất mát ta sẽ sử dụng là hàm MAE

**Sử dụng toàn bộ features (Onehot Encoding + numeric):**

| Dataset | MAE | SIA-1000 | SIA-500 | SIA-200 |
| --- | --- | --- | --- | --- |
| train | 424.3238 | 0.94671 | 0.90634 | 0.82453 |
| val | 335.2073 | 0.94502 | 0.90011 | 0.81818 |
| test | 302.4854 | 0.94774 | 0.90571 | 0.81446 |

## 7.5 Kết Luận

Hầu hết các sản phẩm đều có số lượt bán thấp (91.4% dữ liệu có số lượt bán dưới 1000). Do đó mà việc chúng ta làm giới hạn sai số dự đoán (epsilon) thấp sẽ giúp cho ta đánh giá một cách khách quan và phù hợp với bài toán và nhóm chọn metric SIA-200

Đối với mô hình Neural Network, khi ta sử dụng Onehot Encoding thì xem kết quả có vẻ hợp lý hơn so với khi sử dụng Mean Encoding cho các categorical features.

Theo các kết quả trên ta thấy 2 mô hình cho kết quả tốt nhất là Random Forest và Neural Network và đã sử dụng toàn bộ các features. *Tuy nhiên* mô hình Neural Network có thời gian *train lâu* hơn nhiều so với Random Forest.

Khi ta loại bỏ 2 thuộc tính *numerical avg\_rating và n\_reviews* thì ta thấy hiệu quả khi dự đoán được sold giảm xuống đáng kể hầu hết trên các mô hình và SIA-200 rất thấp. Điều đó chứng tỏ khi mua hàng trên tiki 2 giá trị avg\_rating(Điểm rating trung bình) và n\_reviews(Số lượng người reviews) đã ảnh hưởng 1 cách khách quan tới họ khi quyết định có mua sản phẩm hay không.

***Vì vậy để có thể tăng lượng mua hàng thì phải cải thiện sao cho điểm avg\_rating càng lớn càng tốt***

# 8. Bài toán dự đoán rating

## 8.1 Phương pháp phân rã ma trận (matrix factorization)

**Phân tích ma trận thành nhân tử**

**Xấp xỉ bằng 2 ma trận tiềm ẩn**

***Ý tưởng*** :Với ma trận user-item và giá trị của mỗi ô là điểm rating mà user đánh giá cho item, trong đó có các ô trống vì có những sản phẩm mà người dùng chưa mua. Ta sẽ xấp xỉ ma trận kết quả bằng 2 ma trận ẩn gọi là ma trận người dùng ẩn và ma trận sản phẩm ẩn

Gọi ma trận R là ma trận user-item chúng ta thành lập được khi đó ta sẽ xấp xỉ ma trận này bằng tích 2 ma trận ẩn P và Q.

Ma trận R có kích thước m x n với m người dùng và n sản phẩm.  
Gọi 2 ma trận ẩn là ma trận P có kích thước m x k và ma trận Q có kích thước n x k (lấy k << m, n)

Khi đó:

^R=PQT

Cụ thể đối với điểm rating người dùng u đánh giá sản phẩm i ta có:  
^Rui=puqTi

**Ý nghĩa của 2 ma trận tiềm ẩn**

Ma trận tiềm ẩn P của người dùng trong đó hàng pu của ma trận sẽ chứa các giá trị là mức độ yêu thích của người dùng u với các đặc trưng của 1 sản phẩm ví dụ như miêu tả, giá, hình ảnh, …

Ma trận tiềm ẩn Q của các sản phẩm trong đó hàng qi sẽ chứa các giá trị đo lường mức độ sở hữu các đặc trưng như là tiêu đề, miêu tả hay là hình ảnh sản phẩm

***Chú ý***: Ở các công thức trên có đề cập thì ta có thêm bias bu.

Vì sao lại phải thêm bias cho từng user?

Tùy thuộc vào tính cách đối với các sản phẩm của mỗi người mà họ có thể quyết định số điểm mà họ đánh giá cho sản phẩm đó.

***Lý do:*** Có người dễ tính, có người khó tính, dễ tính 1 cuốn sách hay có thể cho 5 sao nhưng mà khó tính có thể chỉ cho 4 sao hoặc thậm chí là 3 sao chẳng hạn.

Vậy để giải quyết vấn đề này nhóm đã lấy bias của mỗi user là trung bình cộng số điểm mà người đó rating cho những sản phẩm mà người ấy mua.

**Hàm mất mát**

Hàm lỗi nhóm sử dụng là trung bình cộng của bình phương độ chênh lệch kết quả dự doán và kết quả thực tế và dùng thêm l2 regularization để tránh overfitting

Graphical user interface, text, application, Word

Description automatically generated

**Đánh giá mô hình**

Kết quả:

* ***Khi lamda=0***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 1.0439 | 1.9835 | 0.62074 | 0.35644 | 0.27500 |
| test | 1.1729 | 2.4282 | 0.58087 | 0.32911 | 0.26582 |

* ***Khi lamda=0.1***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.50272 | 0.53481 | 0.84673 | 0.62301 | 0.42093 |
| test | 0.55698 | 0.65482 | 0.81935 | 0.58907 | 0.39576 |

* ***Khi lamda=0.5***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.50272 | 0.53481 | 0.85769 | 0.55698 | 0.43130 |
| test | 0.55698 | 0.65482 | 0.83458 | 0.60736 | 0.40702 |

* ***Khi lamda=1***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.50272 | 0.53481 | 0.85770 | 0.63992 | 0.43130 |
| test | 0.55698 | 0.65482 | 0.83458 | 0.60736 | 0.40702 |

***Nhận xét***: Khi ***lamda=0***, ta thấy mô hình đạt độ chính xác khá thấp cả tập train và tập test, đặc biệt tập tain có độ chính xác còn cao hơn tập test. Việc loại bỏ hiệu chỉnh l2 khiến cho mô hình học khá tệ

Khi cho ***lamda != 0*** ta thấy mô hình đạt được độ chính xác khá tốt và tập test và tập train không chênh lệch nhiều Hiệu năng trên tập val và test lệch nhau khá ít ở các độ đo MAE và MSE.

Khi ***lamda=0.1, 0.5, 1*** kết quả gần như xấp xỉ nhau nên nhóm chọn ***lamda=0.5*** làm kết quả cuối cùng

Đối với SIA metric thì ta thấy 83% mẫu test mình dự đoán trong vùng chấp nhận được với sai số là 1 và tiếp tục 2 độ đo SIA 0.5 và 0.25 ở mức trung bình lần lượt là 61% và 41% trên tập test. Mô hình cho kết quả tốt hơn khi thêm hiểu chỉnh l2. Nhóm dự đoán có thể do trong quá trình review sản phẩm thì rất là khó để chúng ta phân biêt được khi nào nên rating 2, khi nào nên rating 3, nên nhóm quyết định sẽ lấy độ đo SIA 1 làm độ đo chính cho các mô hình phía sau và bên cạnh đó có 2 độ đo giúp ta giám sát nữa là MAE tính trung bình trị tuyệt đối độ chênh lệch và MSE trung bình phương độ chênh lệch.

***Với lamda=0.5:***

* learning\_rate=0.1:

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.50273 | 0.53481 | 0.84595 | 0.62260 | 0.42102 |
| test | 0.55699 | 0.65482 | 0.82091 | 0.59001 | 0.39600 |

* learning\_rate=0.5:

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.50272 | 0.53481 | 0.85769 | 0.63992 | 0.43129 |
| test | 0.55698 | 0.65482 | 0.83458 | 0.60736 | 0.40702 |

* learning\_rate=0.75:

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.50272 | 0.53481 | 0.85769 | 0.63992 | 0.43130 |
| test | 0.55698 | 0.65482 | 0.83458 | 0.60736 | 0.40702 |

* learning\_rate=1:

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.50272 | 0.53481 | 0.85769 | 0.63992 | 0.43130 |
| test | 0.55698 | 0.65482 | 0.83458 | 0.60736 | 0.40702 |

***Nhận xét:*** Khi thay đổi các giá trị learning rate thì ta thấy các với các giá trị 0.5 trở lên thì các độ đo khá xấp xỉ nhau nên ta chọn ***learning\_rate=0.75*** làm giá trị cuối cùng

8.3 Phương pháp Content-based

#### Xây dựng feature vector bằng TF-IDF

Trong bộ dataset item/detail\_list.csv, thì chúng ta có hơn 1000 cột chi tiết là 1000 cột lĩnh vực, 1 sản phẩm có thể thuộc nhiều lĩnh vực

***VD:*** Lĩnh vực của 1 sản phẩm là ***bách hóa online/sữa và các sản phẩm từ sữa/sữa nước/sữa chua uống & ăn*** khi đó các cột bách hóa online, sữa và các sản phẩm từ sữa và sữa nước, sữa chua uống & ăn của sản phẩm có giá trị là 1 và các cột lĩnh vực còn lại có giá trị 0.

Ta sẽ xây dựng ma trận feature vector bằng TF-IDF trong đó từng hàng của ma trận chính là feature vector của từng sản phẩm

#### Xây dựng mô hình cho mỗi user

Sau khi có được ma trận feature vector, ta sẽ xây dựng mô hình cho riêng từng user, ở đây nhóm chúng tôi dùng mô hình hồi quy đơn giản, củ thể ở đây bọn em dùng các mô hình như hỗ quy tuyến tính (Linear Regression) và thêm mô hình hồi quy Ridge (Ridge Regression)

Graphical user interface, text, chat or text message

Description automatically generated

***Đầu vào:***

* Đầu vào X của user u là feature extractor của các sản phẩm mà u đánh giá
* Đầu vào y của user u là các điểm đánh giá cho các sản phẩm mà u đã mua hoặc xem

#### Đánh giá mô hình

Table

Description automatically generated

Mô hình Ridge Regression:

Table

Description automatically generated

Table

Description automatically generated

Table

Description automatically generated

***Nhận xét***:

Xét trên độ đo chính mà nhóm chúng tôi sử dụng thì trên cả 2 mô hình Ridge và Linear Regression là SIA 1, ta thấy mô hình Linear Regession có 81.6% mẫu trên tập test được dự đoán thuộc vùng chấp nhận được còn mô hình Ridge với λ=5,100,100,500,1000 thì cao hơn 1 chút, khoảng 82.5% mẫu trên tập test được dự đoán thuộc vùng chấp nhận được.

Tuy nhiên thì khi sử dụng mô hình linear regression thì ta thấy mô hình bị ***overfit*** quá nặng vì vậy ta sử dụng hồi quy ridge, mô hình có vẻ hiểu quả hơn hồi quy tuyến tính, sở dĩ là vì có siêu tham số alpha giúp cho hồi quy ridge chống hiện tượng overfitting.  
Ta chọn λ=100λ=100. Dễ thấy mô hình này tệ hơn mô hình Matrix factorization ở trên

## 8.3 Ứng dụng 1 số neural network đơn giản cho bài toán rating prediction

**Áp dụng MLP và GMF cho phương pháp phân rã ma trận matrix factorization**

**Ý tưởng**

Như đã biết thì phương pháp phân rã ma trận đã đề cập ở trên thì chúng ta đi tìm 2 ma trận ẩn của người dùng và sản phẩm sao cho tích của chúng làm sao để sát với ma trận đánh giá nhất có thể là càng tốt.

Tính đơn giản của phép nhân tích vô hướng của 2 ma trận, rất khó để học được các thuộc tính ẩn của cả user và sản phẩm.

Vì vậy ở mục này nhóm xin đề cập tới 1 mạng phân rã ma trận nơ ron có tên là NeuMF. Sở dĩ nhóm đề cập tới mạng nơ ron nhân tạo ở đây bởi vì tính linh hoạt và tính phi tuyến của nó.

Ở trong mục này nhóm sẽ đề cập tới 2 mạng con có tên là phân rã ma trận tổng quát (***Generalized Matrix Factorization) hay GMF*** và 1 mạng con nữa là ***MLP***

nhóm sẽ sử dụng từng mạng một và kết hợp chúng lại với nhau và đánh giá kết quả của các mạng này ở mục sau.

### 8.3.1 GMF - Generalized Matrix Factorization

**Mô hình**

Mạng con GMF tương tự như cấu trúc ở phương pháp Matrix Factorization nhóm đã đề cập ở trước với 2 ma trận tiềm ẩn user embedding của user và item embedding của item với size lần lượt là ***(num\_users + 1) \* latent\_dim và (num\_items + 1) \* latent\_dim***

***Trong đó:*** num\_users và num\_items là số lượng user và item, latent\_dim là ***số đặc trưng ẩn*** của sản phẩm và user mà ta muốn model của ta học.

***Vì sao lại cộng 1 trong num\_users và num\_items?***

Như đã biết thì hệ thống không có số lượt user và item cố định vì vậy khi đầu vào là 1 user hoặc 1 item mới mà không có trong hệ thống trước đó thì user hoặc item đó sẽ mang giá trị là ***<OOV>(Tương tự như 1 từ không tồn tại trong từ điển)***

***Chú ý:*** trong quá trình cài đặt thuật toán nhóm có thể thêm ***Embedding regularizer l2*** để có thể tránh overfitting cho cả user và item

Sau khi tạo 2 ma trận tiềm ẩn cho user và item thì ta làm dãn 2 ma trận này và thực hiện tính tích 2 ma trận này (element-wise) và cuối cùng cho qua 1 lớp ***Dense size 1*** và đó chính là điểm rating được predict.

Cấu trúc mạng và tính toán tham số:

Chart

Description automatically generated with medium confidence

**Đánh giá mô hình**

* Matrix Factorization đơn giản

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.50271 | 0.53480 | 0.85769 | 0.63992 | 0.43129 |
| test | 0.55698 | 0.65481 | 0.83458 | 0.60735 | 0.40701 |

* Matrix Factorization khi áp dụng thêm GMF *Kernels:*
* ***Glorot-normal***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.47750 | 1.01939 | 0.81558 | 0.77183 | 0.71556 |
| test | 1.74336 | 4.13424 | 0.27226 | 0.12539 | 0.04804 |

* ***Glorot-uniform***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.46440 | 0.94952 | 0.81783 | 0.77583 | 0.72455 |
| test | 1.64591 | 3.66682 | 0.26914 | 0.14101 | 0.06289 |

* ***He-normal***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.46788 | 0.97932 | 0.82076 | 0.77300 | 0.72133 |
| test | 1.69154 | 3.88677 | 0.26914 | 0.13828 | 0.05703 |

* ***He-uniform***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.47528 | 1.01240 | 0.81656 | 0.77329 | 0.72279 |
| test | 1.72836 | 4.03914 | 0.26054 | 0.13007 | 0.04804 |

***Nhận xét:*** Hầu hết trên toàn bộ độ đo trên tập train, test, GMF tệ hơn mô hình matrix factorization. Mô hình đang bị overfit quá nặng và nhìn qua mô hình là vẫn chưa có thêm nhiều ***tính phi tuyến của mạng neural network*** như ta đã nói từ trước.

Ta sẽ cải thiện với mô hình dưới.

### 8.3.2 MLP - Multilayer perceptron

**Mô hình**

Mô hình MLP là 1 mạng neural network đa tầng, cũng như GMF nhóm cũng tạo ra 2 ma trận tiềm ẩn cho user và item.

Sau khi qua 2 ma trận tiềm ẩn thì flatten 2 ma trận này và concatenate chúng lại với nhau.

Xây dụng 1 neural network có ***num\_layer(là 1 list các số nguyên mà mỗi số ứng với số node ẩn của từng tầng trừ tâng đàu tiên ra vì đó là tống số chiều đầu ra của 2 lớp embedding ẩn )*** với hàm kích hoạt là ***relu***

Và lớp cuối cùng chắc chắn là lớp ***Dense size 1*** có giá trị đầu ra là rating được dự đoán

Cấu trúc mạng và tính toán tham số

Chart, box and whisker chart

Description automatically generated

**Đánh giá mô hình**

* Matrix Factorization đơn giản

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.50271 | 0.53480 | 0.85769 | 0.63992 | 0.43129 |
| test | 0.55698 | 0.65481 | 0.83458 | 0.60735 | 0.40701 |

* Ta sẽ đánh giá mô hình khi sử dụng 1 trong 4 kernel dưới đây
  + ***Glorot-normal***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.43510 | 0.36746 | 0.89460 | 0.68206 | 0.43406 |
| test | 0.60169 | 0.63101 | 0.81679 | 0.53593 | 0.29843 |

* + ***Glorot-uniform***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.42707 | 0.35787 | 0.89929 | 0.69495 | 0.44012 |
| test | 0.58926 | 0.617563 | 0.81640 | 0.53789 | 0.30312 |

* + ***He-normal***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.43920 | 0.37190 | 0.89529 | 0.68196 | 0.43152 |
| test | 0.59376 | 0.62358 | 0.81640 | 0.53789 | 0.30312 |

* + ***He-uniform***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.43018 | 0.36219 | 0.89744 | 0.69134 | 0.44373 |
| test | 0.58838 | 0.61101 | 0.82226 | 0.53867 | 0.30976 |

***Nhận xét:*** Khi sử dụng thêm mạng neural đa tầng ta thấy ngay hiệu quả , tất cả độ đo đều tốt hơn so với mô hình ***Matrix Factorization đơn giản đã đề cập trước đó***.

Khi sử dụng mạng neural ta đã thấy thêm tính phi tuyến nó hiệu quả như thế nào và trong bài toán của nhóm tôi, ***đối với mô hình MLP*** thì ***kernel He-uniform*** có vẻ ***hiểu quả hơn các kernel còn lại***

### 8.3.3 Áp dụng cả GMF và MLP

**Mô hình**

Khi ta áp dụng của GMF và MLP mô hình sẽ có cấu trúc như sau:

Diagram

Description automatically generated

Do áp dụng 2 mô hình lại với nhau nên ở đầu ra của mỗi mạng sẽ có 1 ma trận , khi đó ta tiếp tục concatenate 2 matrix này và cho đi qua lớp ***Dense size 1 cuối cùng*** để đưa ra kết quả rating. Cấu trúc mạng và tính toán tham số:

Diagram

Description automatically generated

**Đánh giá mô hình**

* Matrix Factorization đơn giản

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.50271 | 0.53480 | 0.85769 | 0.63992 | 0.43129 |
| test | 0.55698 | 0.65481 | 0.83458 | 0.60735 | 0.40701 |

* Matrix Factorization khi áp dụng cả GMF và MLP
  + ***Glorot-normal***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.370366 | 0.37036 | 0.89626 | 0.69398 | 0.43944 |
| test | 0.58680 | 0.61751 | 0.82148 | 0.55195 | 0.31445 |

* + ***Glorot-uniform***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.42703 | 0.36557 | 0.89695 | 0.69906 | 0.44774 |
| test | 0.58149 | 0.61673 | 0.82070 | 0.55312 | 0.32460 |

* + ***He-normal***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.43519 | 0.37082 | 0.89402 | 0.68714 | 0.44041 |
| test | 0.59795 | 0.63042 | 0.81406 | 0.53789 | 0.30273 |

* + ***He-normal***

| **Dataset** | **MAE** | **MSE** | **SIA 1** | **SIA 0.5** | **SIA 0.25** |
| --- | --- | --- | --- | --- | --- |
| train | 0.42194 | 0.35407 | 0.90076 | 0.69935 | 0.45106 |
| test | 0.59106 | 0.62197 | 0.82265 | 0.53984 | 0.30898 |

***Nhận xét:*** Khi sử dụng cả GMF và NLP ta thấy mặc dù nó hiệu quả hơn so với mô hình ***Matrix Factorization*** thông thường nhưng mà nó gần như xấp xỉ các độ đo so với mô hình khi chi dùng MLP. Điều này chứng minh để mô hình học tốt các đặc trưng ẩn của người dùng và sản phẩm thì mô hình nên cải thiện các tham số ở NLP vì có thêm mạng GMF thì kết quả vẫn không thay đổi nhiều.

Đặc biệt thì ***kernel Glorot-normal*** có vẻ tốt hơn trong trường hợp này so với những kernel khác

# 9. Bài toán hệ gợi ý

## 9.1 Yêu cầu bài toán

Với nhu cầu tăng lượng doanh thu cũng như khai thác tối đa các mặt hàng hiện có sao cho phù hợp với sở thích người dùng, bài toán hệ gợi ý đang được các doanh nghiệp chú trọng phát triển.

Trong đồ án môn học này, nhóm xét đến bài toán gợi ý top-k (Top-K Recommendation), với mô tả bài toán như sau: Với 1 user u và lịch sử tương tác {a, b, c,…}, cần đưa ra top k items mà user u có khả năng quan tâm nhất.

## 9.2 Phương pháp sử dụng

Nhóm triển khai các mô hình gợi ý dựa trên mạng nơ-ron nhân tạo được đề xuất bởi He et al., 2017[1], bao gồm:

* Generalized Matrix Factorization (GMF)
* Multi-Layer Perceptron (MLP)
* Neural Matrix Factorization (NMF)

Ý tưởng chính của các mô hình này đều dựa trên việc học ra biểu diễn ẩn của user và item thông qua các kiến trúc mạng nơ-ron, từ đó mô hình hoá được sở thích của user, cũng như tương tác giữa user và item.

### 9.2.1 GMF

Diagram

Description automatically generated

Kiến trúc của GMF khá đơn giản, như đề cập bởi He et al., 2017[1],đây thực chất là dạng tổng quát hoá của phương pháp Matrix Factorization. Mỗi user và item sẽ được khởi tạo ngẫu nhiên, sau đó, thực hiện element-wise product cho mỗi cặp user-item đầu vào, trước khi cho qua một Dense Layer để tính được trọng số quan tâm (preference score) giữa user và item.

### 9.2.2 MLP

Diagram

Description automatically generated

Thay vì thực hiện element-wise product, kiến trúc MLP thực hiện gộp (concatenate) biểu diễn ẩn của user và item, từ đó cho qua các Dense Layer (hay MLP Layer), trước khi tính ra trọng số quan tâm (preference score) tương tự như trên.

### 9.2.3 NMF

Diagram

Description automatically generated

Từ minh hoạ trong hình cũng dễ thấy được, NMF thực chất là kết hợp của hai kiến trúc GMF và MLP, trong đó output vector (trước khi cho qua Dense Layer cuối để tính trọng số quan tâm) của hai kiến trúc được gộp lại (concatenate), sau đó mới cho qua Dense Layer để tính trọng số quan tâm.

## 9.3 Chi tiết cài đặt

### 9.3.1 Huấn luyện

Phỏng theo Quyen et al., 2021[2], nhóm thực hiện huấn luyện với phương pháp negative sampling, trong đó tỉ lệ negative/positive = 4

Hàm lỗi sử dụng ở đây là Cross Entropy, mục sau sẽ trình bày chi tiết hơn tại sao không chọn RMSE.

Text

Description automatically generated

### 9.3.2 Xử lý tập dữ liệu

Với tập dữ liệu rating ban đầu thu được từ trang thương mại Tiki, nhóm thực hiện loại bỏ các user với lịch sử tương tác ít hơn 10. Thống kê chung về tập dữ liệu được trình bày trong bảng sau:

|  |  |
| --- | --- |
| Dataset | Tiki-Ecommerce |
| # Explicit | 56480 |
| # Users | 3405 |
| # Items | 11312 |
| # Sparsity | 99.85% |

Tương tự, biểu đồ về phân bố rating được thể hiện ở hình dưới:

Chart, pie chart

Description automatically generated

Dựa trên phân bố rating, có thể thấy phần đa các rating thuộc vào đoạn 4-5 (chiếm gần 80%) dẫn đến việc mất cân bằng dữ liệu nghiêm trọng khi sử dụng dữ liệu tường minh (explicit). Do đó, nhóm lựa chọn chỉ sử dụng dữ liệu tiềm ẩn (implicit), trong đó một cặp user-item sẽ được gán nhãn có tương tác (1-positive) nếu tồn tại rating và không tương tác (0-negative) nếu không tồn tại rating tương ứng.

### 9.3.3 Đánh giá

Nhóm thực hiện đánh giá theo chiến lược *leave-one-out*. Item cuối trong chuỗi tương tác được dùng để đánh giá, các item còn lại dùng để huấn luyện. Sau cùng, mỗi ground-truth item (item cuối) được trộn với 99 negative items. Bài toán trở về việc xếp hạng 100 items và đánh giá theo thứ tự xếp hạng của ground-truth item.

Trong đây, nhóm sử dụng 2 độ đo phổ biến là Hit Ratio (HR@K) và Normalized Discounted Cumulative Count (nDCG).

Hit Ratio là độ đo phản ánh dựa trên việc item cuối có nằm trong top K item gợi ý ra bởi hệ thống gợi ý hay không.

Text

Description automatically generated with medium confidence

Mặt khác, nDCG đánh trọng số cao hơn cho các hệ thống gợi ý trong đó item cuối (ground-truth item) được xếp hạng cao hơn các item khác, khi nằm trong top K item được gợi ý ra.

Text

Description automatically generated with low confidence

### 9.3.4 Thông số cài đặt

Nhóm thử nghiệm với các thông số cài đặt sau:

* Giải thuật học: Adam (=0.9, =0.999)
* Eta = {10, 1, 1e-1, 1e-2, 1e-3, 1e-4, 1e-5, 1e-6}
* Latent Size = {4, 8, 16, 32, 64}
* Batch Size = 512
* Negative Sampling Ratio = 4

Kết quả tuning tham số chỉ ra các bộ tham số phù hợp với từng mô hình (Eta, Latent Size):

* GMF: Eta = 1e-1, Latent Size = 8
* MLP: Eta = 1e-2, Latent Size = 32
* NMF: Eta = 1e-2, Latent Size = 16

Phần tiếp theo trình bày chi tiết các thử nghiệm nhóm thực hiện.

## 9.4 Thử nghiệm

### 9.4.1 Eta

Chart, line chart

Description automatically generated

Nhóm thực hiện tuning tham số Eta (tốc độ học, hay learning rate), sử dụng item kề cuối làm đánh giá (validation). Từ trên đồ thị dễ thấy với cả ba mô hình, giá trị Eta hoặc quá lớn (10, 1) hoặc quá nhỏ (1e-5, 1e-6) khiến quá trình học không ổn định, hoặc quá chậm, dẫn đến kết quả chỉ ra không tốt. Với GMF, giá trị Eta tốt nhất là 1e-1, trong khi với MLP và NMF, giá trị này là 1e-2.

### 9.4.2 Latent Size

Chart, line chart

Description automatically generated

Thực hiện tuning tương tự, ta thu được biểu đồ như trên với Latent Size (kích thước vectơ embedding của user và item trong mỗi mô hình).

Từ trên biểu đồ, có vẻ Latent Size ở mức 16 và 32 cho kết quả ổn nhất cho cả 3 mô hình. Tuy nhiên kết quả chi tiết trên tập test (trình bày ngay sau đây) cho thấy cần thực hiện tuning sát sao hơn.

### 9.4.3 Quá trình chạy với 50 epochs

Chart, scatter chart

Description automatically generated

Với Latent Size (K) = 8, cả 3 mô hình đều cho thấy quá trình học ổn định. Về Hit Ratio, MLP cho kết quả nhỉnh hơn NMF, và thấp nhất là GMF. Về nDCG, trái lại, NMF cho kết quả tốt nhất, trong khi MLP thấp hơn, còn GMF ngang ngửa MLP. Như vậy, với K=8, NMF cho thấy khả năng mô hình hoá cặn kẽ hơn, tức các item liên quan khi được tìm thấy, có xu hướng được xếp hạng cao hơn, biểu thị sự mô hình hoá chi tiết hơn về sở thích người dùng, tuy không bao quát bằng MLP (các item liên quan được tìm thấy nhiều hơn, nhưng xu hướng xếp hạng không cao). GMF cũng cho thấy biểu hiện tương tự với K=8.

Graphical user interface, chart, scatter chart

Description automatically generated

Với K=16, MLP tiếp tục đạt kết quả tốt nhất về khía cạnh Hit Ratio, tuy nhiên vẫn thấp hơn chút so với NMF về nDCG. GMF cho kết quả hơi thấp hơn với K=8, đồng thời chênh lệch đáng kể với NMF và MLP.

Graphical user interface, chart, scatter chart

Description automatically generated

Với K=32, quá trình học của GMF trở nên bất ổn định, điều này biểu thị rõ ràng trên đồ thị, khi mà kết quả dao động liên tục ở cả hai độ đo. Như vậy, GMF không hoạt động tốt với K=32, và nhìn chung, kết quả cũng không cao với K=16, khi so với K=8. Trái lại, MLP vẫn cho kết quả tốt, thậm chí nhỉnh hơn một chút, so với K=16. NMF có kết quả thấp hơn, thậm chí chêch lệnh lớn với MLP theo Hit Ratio và chỉ ngang bằng với MLP theo nDCG. Ở giai đoạn giữa thậm chí NMF bị MLP vượt lên trước với nDCG.

Qua đây, có thể thấy được, các giá trị Latent Size tốt nhất với GMF, MLP, NMF lần lượt là 8, 32, 16.

### 9.4.4 Negative Sampling

Graphical user interface, chart, application

Description automatically generated

Tuy nhóm không thực hiện tuning với Negative Sampling Ratio, một thí nghiệm nhỏ khác chứng minh hiệu quả của việc dùng Negative Sampling so với chỉ huấn luyện thông thường.

Cả 3 mô hình được huấn luyện và kiểm thử trên tập test sau 10 epochs, ký hiệu 0- đối với huấn luyện thường và 4- với việc dùng Negative Sampling (ratio = 4). Đồ thị biểu hiễn rõ ràng khi dùng Negative Sampling, kết quả của các mô hình được cải thiện đáng kể.

### 9.4.5 Top-K

Chart, scatter chart

Description automatically generated

Ở phần chạy thử với 50 epochs, ta thấy được NMF có xu hướng cho giá trị nDCG tốt hơn MLP, tuy hơi thấp hơn về Hit Ratio. Ở đây thực hiện một thử nghiệm khác, với số epochs ít hơn để tiết kiệm thời gian, khi cho K (Top-K) thay đổi từ 1-10, ta thu được đồ thị như hình. Dễ thấy với K thấp, thì Hit Ratio của NMF cao hơn hẳn MLP, tương tự GMF cũng cao hơn (K=1, 2) nhưng khi K tăng thì MLP lại vượt lên, tuy ở đây vẫn giữ ngang mức với NMF chứ chưa vượt hẳn như thí nghiệm trên. Song, chỉ từ Hit Ratio, ta lý giải được tại sao NMF cho giá trị nDCG cao hơn hẳn MLP ở đồ thị con bên phải, từ đó có một góc nhìn thú vị.

### 9.4.6 Đánh giá sau cùng

Chart, bar chart

Description automatically generated

Về kết quả cuối trên tập test, MLP cho kết quả cao nhất với HR@10, trong khi NMF-Pretrain (khởi tạo trọng số từ weight của MLP và GMF) cho kết quả tốt nhất với nDCG@10, tuy không cải thiện nhiều với NMF. GMF đạt kết quả thấp nhất, có thể do kiến trúc đơn giản nhất trong các mô hình trên, do đó chưa mô hình hoá được tốt biểu diễn của người dùng. NMF tuy thấp hơn MLP về HR@10, nhưng cho kết quả cao hơn về nDCG@10.

## 9.5 Một vài Case Study

Ở trong mục này, nhóm xét đến một vài Case Study trích ra từ kết quả phán đoán của các mô hình tương ứng.

### 9.5.1 Case Study 1

Text

Description automatically generated

Nhìn chung, người dùng này có sở thích với đồ gia dụng và bách hoá, chăm sóc sức khoẻ và làm đẹp. Một số sản phẩm kể đến như máy xay thịt, tông đơ, đồ ăn (khô gà, mực sợi), vật phẩm dưỡng da (tinh chất dưỡng ẩm, sữa rửa mặt, kem dưỡng ẩm). Đây là một người dùng có sở thích khá đa dạng.

Text

Description automatically generated

Với GMF, về khía cạnh thể loại, đã gợi ý ra một vài sản phẩm khá hợp lý, như sữa đậu nành (người dùng có mua sữa chua), bắp giò heo, cơm cháy (đồ ăn), dầu gội (liên hệ với tông đơ), đèn bàn học (liên hệ với do-choi-me-be). Tuy nhiên, một số sản phẩm còn chưa hợp lý, như gợi ý TP-Link, hay Cát Vệ Sinh cho Mèo (dù cùng thể loại bách hoá nhưng không gần với sở thích người dùng).

Text

Description automatically generated

Bên cạnh các gợi ý tốt (chanh, đèn bàn học, kem chống nắng, tất nữ, bông trang điểm), thì MLP cũng có những gợi ý chưa hợp lý như cảm biến áp suất oto, hay áo bóng đá (người dùng có sở thích với fitness, chứ chưa hẳn là với bóng đá)

Text

Description automatically generated

Được cho là kết hợp của cả 2, NMF cũng có những kết quả hợp lý (nồi chiên không dầu, sữa đậu nành, sữa bột, yến mạch, đồ chơi lắp ráp, kem chống nắng) và chưa hợp lý (USB, Ví nam da bò, đồng hồ nam).

### 9.5.2 Case Study 2

Text

Description automatically generated

Case Study 2 thuộc về một người dùng có sở thích nội trợ, toàn bộ đều là về thực phẩm.

Graphical user interface, text

Description automatically generated

Kết quả gợi ý của GMF khá tệ, ngoại trừ trà oolong có một ít liên quan, còn lại đều là các vật phẩm không liên hệ với sở thích người dùng.

Text

Description automatically generated

Kết quả của MLP có một chút trùng với GMF, nhưng đã tốt hơn với các gợi ý hợp lý như tỏi, chả giò.

Text

Description automatically generated

NMF cũng có hai kết quả liên quan (Bánh bột lọc, cam sành), nhưng còn lại thì chưa hợp lý.

### 9.5.3 Case Study 3

Text

Description automatically generated

Case Study cuối thuộc về một người dùng nữ, có sở thích về thời trang, làm đẹp, và nội trợ chung (quan tâm về máy giặt, nồi chảo).

Text

Description automatically generated

GMF gợi ý ra một vài kết quả hợp lý (Kem dưỡng giảm mụn, giày vải nữ, mặt dây chuyền), nhưng phần đa vẫn là các gợi ý không liên quan.

Text

Description automatically generated

Với MLP, hầu hết vẫn là các kết quả gợi ý không liên quan, ngoại trừ áo thun.

Text

Description automatically generated

Với NMF, kết quả khá khẩm hơn chút khi gợi ý ra được dụng cụ cho nội trợ, việc nhà như cây lau nhà, dép đi trong nhà, nước rửa tay hay kệ chén bát, bên cạnh áo thun. Tuy nhiên, các kết quả khác như dây mạng, quần lót nam, vốn không liên hệ đến lịch sử người dùng, biểu thị chúng ta cần xây dựng hệ gợi ý phù hợp hơn và cải tiến nhiều hơn, trước khi cài đặt trong hệ thống thực.

# 10. Kết luận

Trong dự án này nhóm đã thu thập dữ liệu từ trang thương mại điện tử tiki.vn từ đó thực hiện các bước xử lý, phân tích dự liệu, áp dụng vào các bà toán thực tế như là: Dự đoán việc mua sản phẩm, Dự đoán rating, Xây dựng hệ gợi ý. Qua đây nhóm đã thu được nhiều kinh nghiệm quý báu, học được nhiều kiến thức mới. Trong tương lai nhóm sẽ thu thập thêm dữ liệu từ các trang thương mại điện tử khác và giải quyết thêm nhiều bài toán khác như là dự đoán cảm xúc người dùng, cải thiện hệ gợi ý tốt hơn, phù hợp với yêu cầu người dùng hơn.