**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**BÁO CÁO**

**Hệ gợi ý**

**H&M Personalized Fashion Recommendations**

**Phùng Đức Phong 20204677**

**Nguyễn Thùy Dương 20204536**

**Ngô Văn Huy 20204657**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | TS. Ngô Văn Linh  Chữ ký của GVHD |
| **Khoa:** | Khoa học máy tính |
| **Trường:** | Công nghệ Thông tin và Truyền thông |

**Hà Nội, 1-2024**

**LỜI CẢM ƠN**

Chúng em xin được gửi lời cảm ơn đến TS. Ngô Văn Linh đã giảng dạy và hỗ trợ chúng em trong quá trình học và thực hiện bài tập lớn học phần Hệ gọi ý, giúp chúng em hoàn thành tốt và đầy đủ nhiệm vụ của học phần này.

MỤC LỤC

[CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI 7](#_Toc155493861)

[1.1. Đặt vấn đề 7](#_Toc155493862)

[1.2. Bố cục báo cáo 7](#_Toc155493863)

[CHƯƠNG 2. DỮ LIỆU 8](#_Toc155493864)

[2.1. H&M Personalized Fashion Recommendation 8](#_Toc155493865)

[**2.1.1. DỮ LIỆU** 8](#_Toc155493866)

[2.3. EDA 9](#_Toc155493867)

[2.4. Đánh giá 12](#_Toc155493868)

[**2.4.1. Các chỉ số phân loại** 12](#_Toc155493869)

[**2.4.2. Mean Average Precision at 12 (mAP@12)** 13](#_Toc155493870)

[CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG 13](#_Toc155493871)

[3.1. Decision Tree 13](#_Toc155493872)

[3.2. Bert 14](#_Toc155493873)

[3.3. Nearest Neighbor 15](#_Toc155493874)

[3.4. Gradient Boosting 15](#_Toc155493875)

[3.5. Light GBM in Learning for Ranking 16](#_Toc155493876)

[3.6. SVD(Singular Vector Decomposition) Ranking 17](#_Toc155493877)

[3.7. CATBoost 17](#_Toc155493878)

[CHƯƠNG 4. THIẾT KẾ VÀ KIẾN TRÚC MÔ HÌNH 19](#_Toc155493879)

[4.1. Tổng quan kiến trúc 19](#_Toc155493880)

[4.2. Triển khai các thành phần 19](#_Toc155493881)

[**4.2.1. Ariticle Encode** 19](#_Toc155493882)

[**4.2.2. Customer Encode** 19](#_Toc155493883)

[**4.2.3. ClickPredict** 19](#_Toc155493884)

[CHƯƠNG 5. KẾT QUẢ THỰC NGHIỆM 20](#_Toc155493885)

[5.1. Đối với SVD 20](#_Toc155493886)

[5.2. Đối với CATBoost 20](#_Toc155493887)

[5.3. Đối với LightGBM 20](#_Toc155493888)

[5.4. Đối với Ensemble 20](#_Toc155493889)

[CHƯƠNG 6. KẾT LUẬN 21](#_Toc155493890)

[6.1. Kết luận 21](#_Toc155493891)

[6.2. Hướng phát triển trong tương lai 21](#_Toc155493892)

[TÀI LIỆU THAM KHẢO 22](#_Toc155493893)

# **CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI**

## 1.1. Đặt vấn đề

Tập đoàn H&M (H&M group) là một tập đoàn gồm các thương hiệu và doanh nghiệp với 53 thị trường trực tuyến và khoảng 4850 cửa hàng. Các cửa hàng trực tuyến cung cấp cho người mua hàng nhiều sự lựa chọn. Tuy nhiên, với việc có quá nhiều lựa chọn, khách hàng sẽ không thể tìm kiếm nhanh chóng mặt hàng mà họ cần mua, điều này dễ khiến khách hàng nhanh chóng bỏ cuộc và cuối cùng là không mua gì cả. Chính vị vậy, để nâng cao trải nghiệm mua hàng của người dùng, việc tích hợp thêm hệ gợi ý là điều cần thiết.

## 1.2. Bố cục báo cáo

Phần còn lại của báo cáo được tổ chức như sau:

# **CHƯƠNG 2. DỮ LIỆU**

## 2.1. H&M Personalized Fashion Recommendation

Trong những năm gần đây, chứng khoán là một kênh đầu tư sinh lời giành được rất nhiều sự quan tâm của các nhà đầu tư, từ những người có thâm niên trong việc đầu tư đến những người chưa tham gia đầu tư bao giờ. Để có thể xây dựng tốt một nền tảng dữ liệu chứng khoán, việc nắm được những kiến thức về thị trường chứng khoán là điều hết sức cần thiết.

Tập dữ liệu của bài toán được lấy từ cuộc thi trên Kaggle với tên gọi H&M Personalized Fashion Recommendation, có đường dẫn như dưới đây:

(<https://www.kaggle.com/competitions/h-and-m-personalized-fashion-recommendations>)

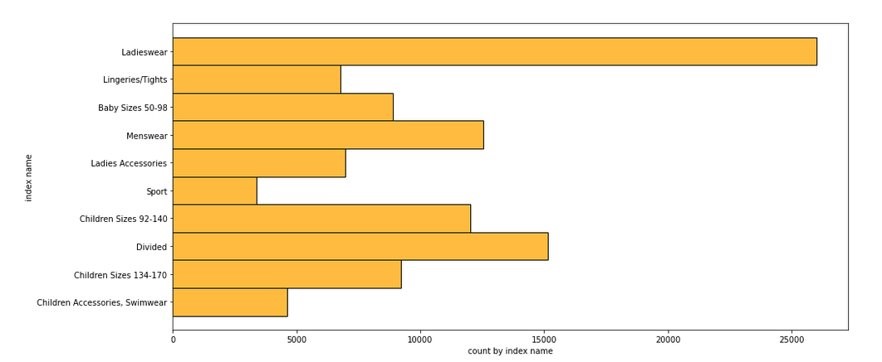
### **2.1.1. DỮ LIỆU**

Trong cuộc thi này, H&M cung cấp các tệp tin về thông tin của các sản phẩm, thông tin của khách hàng và cuối cùng là thông tin giao dịch. Cụ thể:

* images (106k sản phẩm): là thư mục chứa hình ảnh của các sản phẩm
* articles.csv (106k sản phẩm): tệp chứa các thông tin liên quan đến sản phẩm, cụ thể bao gồm 25 trường như dưới đây:
  + article\_id: định danh của sản phẩm
  + product\_code: mã sản phẩm
  + prod\_name: tên sản phẩm
  + product\_type\_no: mã thể loại của sản phẩm
  + product\_type\_name: tên thể loại của sản phẩm
  + product\_group\_name: tên nhóm sản phẩm
  + graphical\_appearance\_no: mã số hình dáng đồ họa của sản phẩm
  + graphical\_appearance\_name: tên hình sáng đồ họa của sản phẩm
  + colour\_group\_code: mã nhóm màu
  + colour\_group\_name: tên nhóm màu
  + perceived\_colour\_value\_id: định danh giá trị màu biểu kiến của sản phẩm
  + perceived\_colour\_value\_name: tên màu biểu kiến của sản phẩm
  + department\_no: mã phòng ban
  + department\_name: tên phòng ban
  + index\_code: mã index
  + index\_name: tên index
  + index\_group\_no: mã nhóm index
  + index\_group\_name: tên nhóm index
  + section\_no: mã danh mục
  + section\_name: tên danh mục
  + garment\_group\_no: định danh cho một nhóm các sản phẩm dựa trên một số đặc trưng về chức năng.
  + garment\_group\_name: tên cho một nhóm các sản phẩm dựa trên một số đặc trưng về chức năng.
  + detail\_desc: mô tả chi tiết về sản phẩm
  + perceived\_colour\_master\_id: định danh màu biểu kiến chủ đạo
  + perceived\_colour\_master\_name: tên màu biểu kiến chủ đạo
* customers.csv (1.3m khách hàng): tệp chứa các thông tin của khách hàng, cụ thể:
  + customer\_id: định danh cho khách hàng.
  + FN: khách hàng có nhận thư về tin tức thời trang hay không.
  + Active: tình trạng hoạt động của khách hàng.
  + club\_member\_status: tình trạng đăng ký làm thành viên câu lạc bộ của khách hàng.
  + fashion\_news\_frequency: tần suất cập nhật tin tức thời trang của khách hàng.
  + age: tuổi của khách hàng.
  + postal\_code: postal code của khách hàng (đã được mã hóa)
* transactions\_train.csv (31.8m giao dịch): tệp chứa các thông tin giao dịch của khách hàng, cụ thể:
  + t\_dat: thời gian giao dịch.
  + customer\_id: định danh khách hàng.
  + article\_id: định danh sản phẩm.
  + price: giá thanh toán.
  + sales\_channel\_id: kênh bán (online hay offline)

## 2.3. EDA

Khai phá dữ liệu là một bước quan trọng để tiếp cận bài toán và hiểu dữ liệu, cũng như tìm ra những ý tưởng từ dữ liệu. Dưới đây làm một số phân tích từ dữ liệu của nhóm. Đầu tiên là một số phân tích từ tập articles.csv



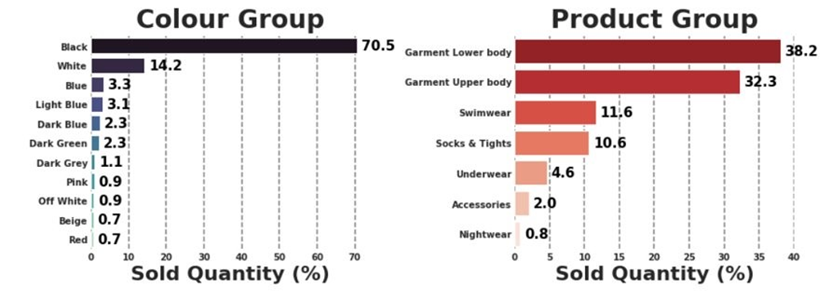
Có thể thấy, tập dữ liệu này cho thấy một sự đa dạng đáng chú ý trong các danh mục sản phẩm, với sự tập trung chủ yếu vào quần áo nữ. Điều là hợp lí với thực tế vì trang phục và quần áo thường đa số phục vụ nhu cầu làm đẹp của phụ nữ.

Ngoài ra, ở top 3 sản phẩm ưa chuộng của từng lứa tuổi. Nhìn chung, Ladieswear, Lingeries/Tights, Divided là 3 loại sản phẩm được mua nhiều nhất ở mọi lứa tuổi.

A diagram of clothing with different colors and numbers

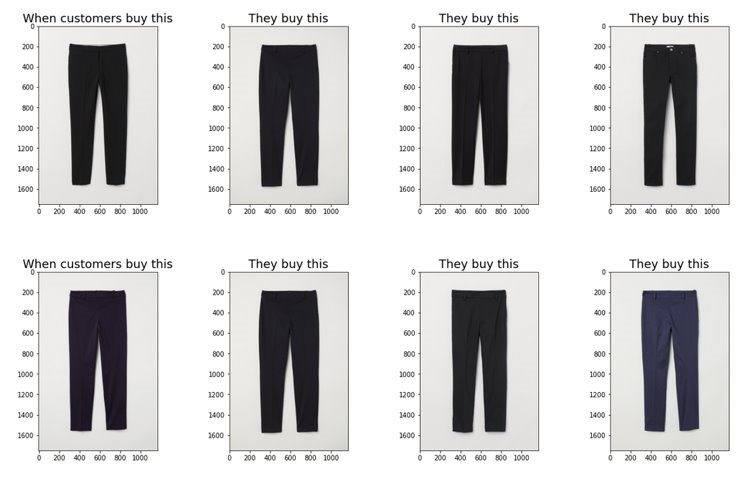
Description automatically generated

Bên cạnh đó, Thống kê về mua bán theo màu sắc và nhóm sản phẩm:



Từ biểu đồ trên, ta có thể thấy được nhu cầu mua của khách hàng rõ rệt. Hơn 70% khách hàng thích mua sản phẩm màu đen và cùng với đó gần 40% sức mua là quần, váy.

Tiếp theo là một vài thống kê về giao dịch của khách hàng dựa trên tập dữ liệu từ file transaction.csv.



Hình ảnh trên cho thấy sản phẩm được mua cùng nhau. Có thể thấy, khách hàng thường thích mua cùng một loại sản phẩm như quần , áo, váy , các loại sản phẩm này có cùng một style và thường có màu sắc khác nhau, hoặc chỉ một loại màu sắc.

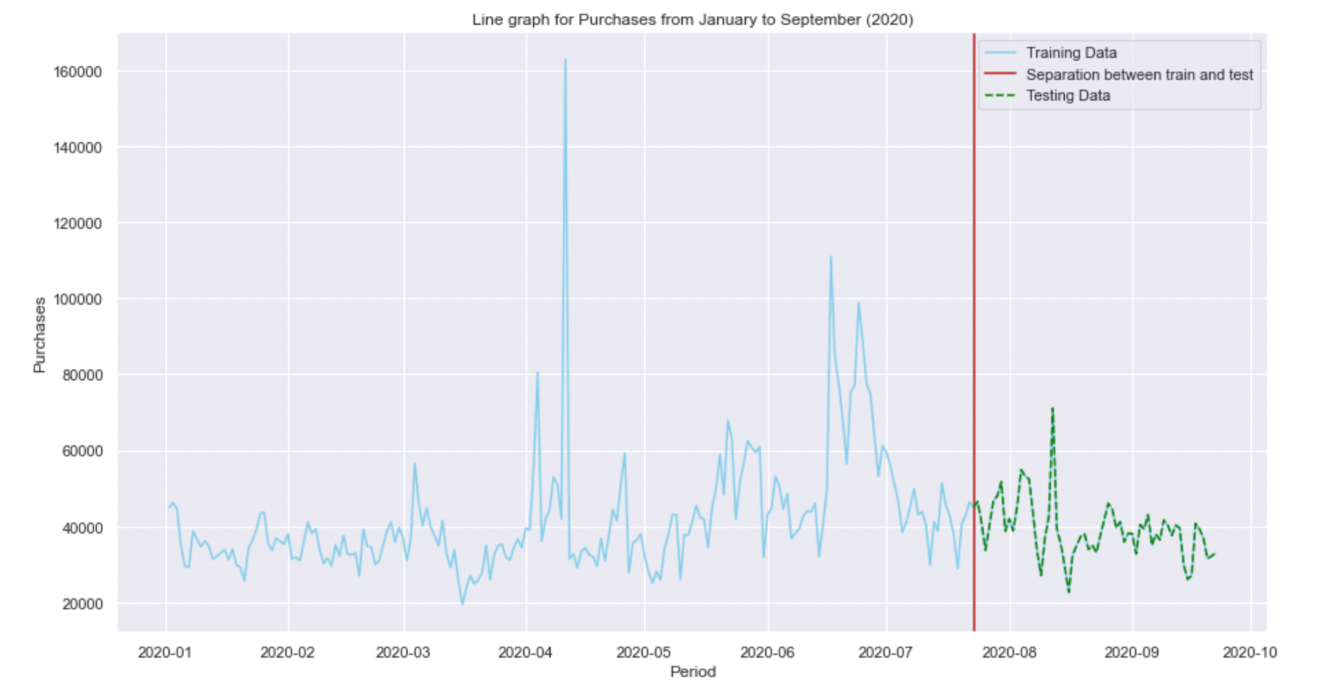
Tệp giao dịch chứa danh sách các giao dịch mua do khách hàng thực hiện trong khoảng thời gian 2 năm, với các chi tiết như giá của mặt hàng, id duy nhất của khách hàng, id duy nhất của mặt hàng và ngày mua.

A graph showing the different colored lines

Description automatically generated with medium confidence

Để hiểu xu hướng chung của các giao dịch mua này, biểu đồ đường trên đã được tạo, biểu thị số lượng giao dịch mua được thực hiện trong các tháng và khoảng tin cậy tương ứng của chúng. Có thể thấy, dữ liệu được truy xuất bắt đầu từ tháng 9 năm 2018 và tiếp tục cho đến tháng 12 năm 2020. Dường như có một mô hình tổng thể về việc tăng lượng mua hàng trong nửa đầu năm, đạt đỉnh điểm vào tháng 6, sau đó giảm mạnh hơn trong suốt nửa đầu năm. một nửa thứ hai. Giữa mỗi năm, xu hướng hàng tháng dường như gần như giống hệt nhau, như được mô tả trong biểu đồ năm 2019 và 2020.

Do lượng dữ liệu khổng lồ và những hạn chế về mặt tính toán, nhóm em đã quyết định lấy mẫu một phần của tập dữ liệu cho mục đích của mình. Nhóm em đã trích xuất tất cả các giao dịch được thực hiện trong năm 2020, sau đó chia thành các tập dữ liệu huấn luyện, kiểm tra và tối ưu.



Bộ dữ liệu kiểm tra và tối ưu được lấy mẫu ngẫu nhiên từ dữ liệu của 2 tháng qua, do đó, mô hình sẽ cố gắng dự đoán những gì khách hàng có khả năng mua dựa trên lịch sử mua hàng trước đó của họ trong 7 tháng trước đó. Để hiểu xu hướng trong dữ liệu được lấy mẫu này và sự phân chia giữa dữ liệu huấn luyện và dữ liệu kiểm tra, biểu đồ đường bên dưới đã được vẽ. Từ hình ảnh trực quan này, có thể suy ra rằng các giao dịch mua dường như nằm trong một phạm vi gần đúng, sai lệch duy nhất so với điều này xảy ra vào tháng 6, phù hợp với các phân tích của chúng tôi về tập dữ liệu hoàn chỉnh.

## 2.4. Đánh giá

### **2.4.1. Các chỉ số phân loại**

Dưới đây hình ảnh confusion matrix đối với phân lớp nhị phân:

Table

Description automatically generated

Trong đó,

* TP (True Positive): dương tính đúng (số nhãn + được dự đoán là +)
* TN (True Negative): âm tính đúng (số nhãn - được dự đoán là -)
* FP (False Positive): dương tính giả (số nhãn + được dự đoán là -)
* FN (False Negative): âm tính giả (số nhãn – được dự đoán là +)

Các thang đo đánh giá tính hiệu quả của bài toán phân lớp được tính toán dựa trên confusion matrix được tính toán như sau:

Các thang đo đánh giá tính hiệu quả của bài toán phân lớp được tính toán dựa trên confusion matrix được tính toán như sau:

* Độ chính xác (accuracy): xác định mức độ chính xác của kết quả phân lớp cho cả phân lớp âm và dương tính. Độ chính xác được tính theo công thức:
* Độ nhạy (Recall): xác định mức độ chính xác của kết quả phân lớp cho cả phân lớp âm và dương tính. Độ chính xác được tính theo công thức:
* Độ chính xác phép đo (precision): xác định tỷ số dương tính được xác định đúng trên tổng số dương tính. Độ chính xác phép đo tính theo công thức
* Điểm F1 (F1 score): là trung bình điều hòa của giá trị độ nhạy và độ chính xác phép  
  đo. Điểm F1 được tính theo công thức:

### **2.4.2. Mean Average Precision at 12 (mAP@12)**

Trong bài toán lần này, MAP@12 sẽ được sử dụng với công thức như dưới đây:

, trong đó:

* : Số người dùng.
* : Số mặt hàng mà người dùng mua thực tế.
* : Số mặt hàng được dự đoán người dùng sẽ mua.
* : Chỉ số Precision tại điểm cắt thứ k
* : Hàm chỉ số với giá trị là 1 nếu mặt hàng ở vị trí thứ k được dự đoán đúng, ngược lại là 0.

# **CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG**

## 3.1. Decision Tree

Decision Tree là một trong những thuật toán máy học phổ biến nhất hiện nay, được dùng trong cả bài toán phân lớp và hồi quy. Cây quyết định là cây mà mỗi nút biểu diễn một đặc trưng (tính chất), mỗi nhánh (branch) biểu diễn một quy luật (rule) và mỗi lá biểu biễn một kết quả (giá trị cụ thể hay một nhánh tiếp tục).

Diagram

Description automatically generated

Ưu điểm của thuật toán:

1. Cây quyết định bắt trước mức độ suy nghĩ của con người nên nó đơn giản để hiểu và chuẩn bị dữ liệu.
2. Cây quyết định giúp bạn thấy được logic từ dữ liệu

Có một vài thuật toán để tạo một cây quyết định, trong đó:

1. CART (Classification and Regression Trees) → dùng Gini Index(Classification) để kiểm tra.

Ý tưởng của thuật toán:

* Thuật toán sẽ xây dựng cây bắt đầu từ nút gốc
* Tại mỗi nút, chọn một thuộc tính với giá trị phân loại của nó (đánh giá bằng Gini) và chia các ví dụ học thành các nửa (tùy thuộc vào lựa chọn, nếu chia thành 2 nửa và giá trị phân loại là 10 thì có thể chia 1 nửa và nửa còn lại 10), cụ thể như sau:
* Với mỗi thuộc tính, tìm giá trị phân loại trong thuộc tính đó. Đối với thuộc tính số (cũng là cách cài đặt trong thư viện sklearn), ta lấy các giá trị kiểm tra, là các giá trị trung bình giữa 2 giá trị liên tiếp trong thuộc tính đó. Nếu giá trị kiểm tra nào cho Gini Impurity thấp hơn sẽ được sử dụng để làm giá trị phân loại, sau đây là cách tính Gini Impurity. Trước hết, với mỗi giá trị kiểm tra, ta chia được dữ liệu thành các nửa, mỗi nửa ta sẽ tính giá trị Gini trên nửa đó như sau:

, trong đó , là số lượng các nhãn i của nửa này và n là tổng số lượng các ví dụ trong nửa này. Khi đó Gini Impurity của giá trị kiểm tra này sẽ là:

, trong đó N là tổng số lượng các ví dụ cho đến hiện tại, là số lượng ví dụ trong nửa thứ k, là giá trị Gini trong nửa thứ k.

* Chọn giá trị phân loại có Gini Impurity thấp nhất, sau đó tiến hành tạo các nút con theo giá trị phân loại này. Lặp lại quá trình tìm giá trị phân loại đối với mỗi nút con này cho đến khi vẽ xong toàn bộ cây (hoặc gặp điều kiện dừng).
* Chú ý: đối với CART, mỗi thuộc tính có thể lặp lại nhiều lần (nhưng giá trị phân loại mỗi lần là khác nhau). Trong cài đặt của sklearn, do phải tránh trường hợp có quá nhiều thuộc tính, CART sẽ lựa chọn thuộc tính trước sau đó mới tìm giá trị phân loại của thuộc tính này (có thể lựa chọn ngẫu nhiên).

1. ID3 (Iterative Dichotomiser 3) → dùng Entropy function và Information gain để kiểm tra.

## 3.2. Bert

[BERT](https://arxiv.org/pdf/1810.04805.pdf) là viết tắt của cụm từ Bidirectional Encoder Representation from Transformer có nghĩa là mô hình biểu diễn từ theo 2 chiều ứng dụng kỹ thuật Transformer. BERT được thiết kế để huấn luyện trước các biểu diễn từ (pre-train word embedding). Điểm đặc biệt ở BERT đó là nó có thể điều hòa cân bằng bối cảnh theo cả 2 chiều trái và phải.

Cơ chế attention của Transformer sẽ truyền toàn bộ các từ trong câu văn đồng thời vào mô hình một lúc mà không cần quan tâm đến chiều của câu. Do đó Transformer được xem như là huấn luyện hai chiều (bidirectional) mặc dù trên thực tế chính xác hơn chúng ta có thể nói rằng đó là huấn luyện không chiều (non-directional). Đặc điểm này cho phép mô hình học được bối cảnh của từ dựa trên toàn bộ các từ xung quanh nó bao gồm cả từ bên trái và từ bên phải.

Các kiến trúc model BERT:

Hiện tại có nhiều phiên bản khác nhau của model BERT. Các phiên bản đều dựa trên việc thay đổi kiến trúc của Transformer tập trung ở 3 tham số: L: số lượng các block sub-layers trong transformer, H: kích thước của embedding véc tơ (hay còn gọi là hidden size), A: Số lượng head trong multi-head layer, mỗi một head sẽ thực hiện một self-attention. Tên gọi của 2 kiến trúc bao gồm:

* BERTBASE(L=12, H=768, A=12): Tổng tham số 110 triệu.
* BERTLARGE(L=24, H=1024, A=16): Tổng tham số 340 triệu.

Như vậy ở kiến trúc BERT Large chúng ta tăng gấp đôi số layer, tăng kích thước hidden size của embedding véc tơ gấp 1.33 lần và tăng số lượng head trong multi-head layer gấp 1.33 lần.

Các tác vụ sử dụng model BERT có thể kể đến như:

1. Fine-tuning model BERT: Một điểm đặc biệt ở BERT mà các model embedding trước đây chưa từng có đó là kết quả huấn luyện có thể fine-tuning được. Chúng ta sẽ thêm vào kiến trúc model một output layer để tùy biến theo tác vụ huấn luyện.
2. Masked ML (MLM):  tác vụ cho phép chúng ta fine-tuning lại các biểu diễn từ trên các bộ dữ liệu unsupervised-text bất kỳ. Chúng ta có thể áp dụng Masked ML cho những ngôn ngữ khác nhau để tạo ra biểu diễn embedding cho chúng.
3. Next Sentence Prediction (NSP)

## 3.3. Nearest Neighbor

K-nearest neighbor là một trong những thuật toán supervised-learning đơn giản nhất (mà hiệu quả trong một vài trường hợp) trong Machine Learning. Khi training, thuật toán này *không học* một điều gì từ dữ liệu training (đây cũng là lý do thuật toán này được xếp vào loại [lazy learning](https://en.wikipedia.org/wiki/Lazy_learning)), mọi tính toán được thực hiện khi nó cần dự đoán kết quả của dữ liệu mới. K-nearest neighbor có thể áp dụng được vào cả hai loại của bài toán Supervised learning là [Classification](https://machinelearningcoban.com/2016/12/27/categories/#classification-phan-loai) và [Regression](https://machinelearningcoban.com/2016/12/27/categories/#regression-hoi-quy). KNN còn được gọi là một thuật toán [Instance-based hay Memory-based learning](https://en.wikipedia.org/wiki/Instance-based_learning).

Với KNN, trong bài toán Classification, label của một điểm dữ liệu mới (hay kết quả của câu hỏi trong bài thi) được suy ra trực tiếp từ K điểm dữ liệu gần nhất trong training set. Label của một test data có thể được quyết định bằng major voting (bầu chọn theo số phiếu) giữa các điểm gần nhất, hoặc nó có thể được suy ra bằng cách đánh trọng số khác nhau cho mỗi trong các điểm gần nhất đó rồi suy ra label. Chi tiết sẽ được nêu trong phần tiếp theo.

Ưu điểm của KNN

1. Độ phức tạp tính toán của quá trình training là bằng 0.
2. Việc dự đoán kết quả của dữ liệu mới rất đơn giản.
3. Không cần giả sử gì về phân phối của các class.

Nhược điểm của KNN

1. KNN rất nhạy cảm với nhiễu khi K nhỏ.
2. Như đã nói, KNN là một thuật toán mà mọi tính toán đều nằm ở khâu test. Trong đó việc tính khoảng cách tới *từng* điểm dữ liệu trong training set sẽ tốn rất nhiều thời gian, đặc biệt là với các cơ sở dữ liệu có số chiều lớn và có nhiều điểm dữ liệu. Với K càng lớn thì độ phức tạp cũng sẽ tăng lên. Ngoài ra, việc lưu toàn bộ dữ liệu trong bộ nhớ cũng ảnh hưởng tới hiệu năng của KNN.

## 3.4. Gradient Boosting

Boosting là một phương pháp học máy thuộc họ các phương pháp kết hợp (ensemble), là kĩ thuật sử dụng nhiều mô hình yếu phối hợp với nhau để tạo thành một mô hình dự báo mạnh hơn. Một mô hình điển hình của phương pháp kết hợp là Rừng ngẫu nhiễn, với việc sử dụng nhiều cây quyết định yếu được học trên các tập dữ liệu khác nhau (và hi vọng là chúng độc lập), sau đó kết hợp dự đoán từ các cây này để cho ra kết quả cuối cùng. Khác với Rừng ngẫu nhiên, Boosting mặc dù cũng xây dựng một lượng lớn các mô hình (thường là cùng loại), nhưng các mô hình sau sẽ học cách sửa những lỗi sai của mô hình trước (ví dụ những mẫu dữ liệu mà mô hình trước dự đoán sai), tạo thành một chuỗi các mô hình mà mô hình sau sẽ tốt hơn mô hình trước bởi trọng số được cập nhật qua mỗi mô hình (cụ thể ở đây là trọng số của những dữ liệu dự đoán đúng sẽ không đổi, còn trọng số của những dữ liệu dự đoán sai sẽ được tăng thêm). Cuối cùng, chúng ta sẽ lấy kết quả của mô hình cuối cùng trong chuỗi mô hình này làm kết quả trả về. Với Gradient Boosting, Trước khi vào thuật toán, ta định nghĩa 1 số ký hiệu sau:

* tập dữ liệu huấn luyện.
* nhãn của tập dữ liệu huấn luyện.
* là số lượng vòng lặp.
* : là số lượng mẫu huấn luyện.
* giá trị dữ đoán của cây quyết định thứ cho tập dữ liệu .
* một lớp các hàm, dưới đây ta giả sử lớp hàm này được xây dựng từ các cây quyết định.
* giá trị mất mát giữa nhãn và giá trị dự đoán (ví dụ như hàm binary cross-entropy).
* residual-response từ cây quyết định thứ gửi cho cây quyết định thứ .

Về ý tưởng, Gradient Boosting tại mỗi vòng lặp sẽ tạo một cây quyết định mới để học sự sai khác giữa giá trị dự đoán và giá trị thực, từ đó mô hình cuối cùng sẽ dùng nó để điều chỉnh lại dự đoán của mình. Dưới đây là chi tiết thuật toán của Jerome Friedman vào năm 1999:

* Khởi tạo f ̂(X)=f ̂\_0 (X)=γ,γϵR (Có thể chọn γ=argmin┬γϵR⁡[L(y,γ)] )
* Với mỗi vòng lặp t=1..M, lặp lại các bước sau:
* Tính giá trị residual-response:
  + r\_t=(r\_0t,r\_1t,…,r\_nt ), r\_it=-⁡[∂L(y,f(x))/∂f(x) ](f ̂(x\_i )),i=1..N
* Huấn luyện một cây quyết định mới với tập dữ liệu X và nhãn là r\_t. Giả sử cây quyết định sau khi học là h\_t (X).
* Tìm hệ số ρ\_t=argmin┬ρϵR⁡[L(y,f ̂(X)+ρ∙h\_t (X))]
* Lưu lại f ̂\_t (X)=ρ\_t∙h\_t (X)
* Cập nhật f ̂(X)←f ̂(X)+f ̂\_t (X)
* Sử dụng f ̂(X) để đưa ra giá trị dự đoán.

## 3.5. Light GBM in Learning for Ranking

Light Gradient Boosting Machine (LGBM) là một thuật toán học máy thuộc họ các thuật toán Gradient Boosting Decision Tree (GBDTs), một mô hình tập hợp các cây quyết định. LGBM được Microsoft đề xuất lần đầu vào năm 2016, có khả năng khắc phục nhược điểm chi phí tính toán lớn của các mô hình GBDTs, với khả năng tính toán có thể nhanh hơn đến 20 lần, mặc dù vẫn giữ được độ chính xác tương đương.

Ý tưởng của thuật toán: Các mô hình GBDTs có lượng chi phí tính toán lớn chủ yếu do trong quá trình xây dựng cây quyết định. Các cây quyết định được xây dựng bằng cách duyệt qua toàn độ đặc trưng trong tập dữ liệu, với mỗi đặc trưng lại duyệt qua toàn bộ mẫu để có thể ước lượng được điểm phân chia (hay giá trị phân loại). Như vậy với mỗi lần hình thành cây, độ phức tạp là vào khoảng O(NxM), với N là số đặc trưng và M là số mẫu huấn luyện. Để giải quyết vấn đề này, LGBM sử dụng 2 kỹ thuật:

 GOSS (Gradient-based One Side Sampling): Các mẫu huấn luyện có gradient thấp trong mỗi lần lặp sẽ có lỗi huấn luyện nhỏ, điều này có thể xem xét rằng những mẫu như này không cần thiết phải đưa vào huấn luyện nữa. Tuy nhiên, các mẫu này cũng không thể loại bỏ hoàn toàn ra khỏi tập huấn luyện được, do với số lượng của chúng nếu bị loại bỏ sẽ dẫn đến việc phân phối dữ liệu có thể khác biệt, ảnh hưởng đến độ chính xác của mô hình. Vì vậy, GOSS sẽ chỉ loại bỏ ngẫu nhiên một phần các mẫu có gradient thấp, và giữ lại các mẫu có gradient cao.

 EFB (Exclusive Feature Bundling): Thông thường, mặc dù số lượng các đặc trưng là rất lớn, tuy nhiên các đặc trưng thường lại loại trừ lẫn nhau (ví dụ các đặc trưng hiếm khi đồng thời khác 0, tiêu biểu là biểu diễn one-hot). Do đó, chúng ta có thể loại trừ những đặc trưng này đi mà vẫn có thể giữ độ chính xác tương đương. Tuy nhiên, đây là một bài toán không có lời giải trong thời gian đa thức. Để đạt được một lời giải hiệu quả, EFB là thuật toán tham lam có độ phức tạp cỡ hằng số, dựa trên bài toán tô màu đồ thị.

Learning to Rank hay học xếp hạng là ứng dụng của học máy trong việc xây dựng các mô hình xếp hạng cho hệ thống truy xuất thông tin. Ví dụ, dữ liệu huấn luyện có thể bao gồm các danh sách các mục với một số thứ tự từng phần được chỉ định giữa các mục trong mỗi danh sách. Thứ tự này thường được tạo ra bằng cách đưa ra điểm số cho mỗi mục. Mục tiêu của việc xây dựng mô hình xếp hạng là xếp hạng các danh sách mới, chưa được nhìn thấy theo cách tương tự như xếp hạng các mục trong dữ liệu huấn luyện.

## 3.6. SVD(Singular Vector Decomposition) Ranking

SVD là một kỹ thuật lọc cộng tác rất phổ biến cho các bài toán gợi ý. Nó đã được sử dụng rất nhiều trong quá khứ khi có sự bùng nổ của RecSys (Netflix Prize).

SVD (Singular Value Decomposition) cũng có thể được áp dụng trong bài toán xếp hạng và học xếp hạng, nhưng cách tiếp cận sẽ khác biệt so với Light Gradient Boosting Machine (LGBM). Dưới đây là một giả định cơ bản về cách SVD có thể được sử dụng trong bài toán xếp hạng:

* + Biểu diễn Ma trận Đánh giá: Giả sử bạn có một ma trận đánh giá *R* với các hàng đại diện cho người dùng, cột đại diện cho sản phẩm và các giá trị đánh giá ở mỗi ô.
  + Phân rã SVD: Áp dụng SVD cho ma trận đánh giá R để có được phân rã UΣVT, trong đó U, Σ, và V tương ứng với các ma trận trong phương trình SVD.
  + Chọn giá trị đơn: Chọn một số lượng giá trị đơn quan trọng nhất để giảm chiều của ma trận. Các giá trị đơn này thường đại diện cho các đặc trưng quan trọng trong dữ liệu.
  + Xây dựng ma trận xếp hạng dự đoán: Tính toán ma trận xếp hạng dự đoán bằng cách sử dụng chỉ một số lượng giá trị đơn đã chọn. Ma trận xếp hạng này có thể được tính toán bằng cách nhân các ma trận U, Σ, và VT đã được giảm chiều
  + Dự đoán và Đề Xuất: Sử dụng ma trận xếp hạng dự đoán để dự đoán xếp hạng cho sản phẩm chưa được đánh giá cho mỗi người dùng. Đề xuất các sản phẩm có xếp hạng cao nhất cho người dùng dựa trên dự đoán.

Tính toán SVD có thể được thực hiện bằng nhiều thư viện máy học như NumPy, SciPy hoặc thậm chí các thư viện chuyên biệt cho bài toán đề xuất như Surprise.

Lưu ý rằng trong thực tế, các biến thể khác nhau của mô hình xếp hạng SVD cũng có thể được sử dụng, và các kỹ thuật như Stochastic Gradient Descent (SGD) cũng có thể được kết hợp để cải thiện hiệu suất.

## 3.7. CATBoost

CatBoost là một thuật toán học máy thuộc họ các thuật toán Gradient Boosting Decision Tree (GBDTs), một mô hình tập hợp các cây quyết định. CatBoost nổi bật với khả năng xử lý tốt với dữ liệu có đặc điểm dạng category (như biến categorical), giảm thiểu sự phải mã hóa trước dữ liệu. Nó được phát triển bởi Yandex và đã trở thành một trong những lựa chọn phổ biến trong cộng đồng máy học.

Ý tưởng của thuật toán:

Trong quá trình xây dựng cây quyết định, CatBoost giảm chi phí tính toán bằng cách sử dụng GOSS (Gradient-based One Side Sampling). Các mẫu huấn luyện có gradient thấp được coi là có đóng góp ít vào lỗi huấn luyện, và một phần của chúng được loại bỏ ngẫu nhiên.

Đối với EFB (Exclusive Feature Bundling), CatBoost cũng có cơ chế tự động hỗ trợ. Các đặc trưng có thể tự động được nhóm lại mà không cần phải thực hiện quá trình mã hóa one-hot trước đó.

Học xếp hạng (Learning to Rank) là một ứng dụng quan trọng của học máy, trong đó mục tiêu là xây dựng các mô hình xếp hạng cho hệ thống truy xuất thông tin. Dữ liệu huấn luyện thường bao gồm danh sách các mục với thứ tự giữa chúng, được tạo ra thông qua việc đánh điểm cho mỗi mục. Mô hình học xếp hạng sau đó được sử dụng để xếp hạng các danh sách mới, chưa được quan sát trước đó.

# **CHƯƠNG 4. THIẾT KẾ VÀ KIẾN TRÚC MÔ HÌNH**

## 4.1. Tổng quan kiến trúc

A diagram of a company

Description automatically generated

Đối với hệ thống gợi ý, chúng ta mong muốn khi cung cấp thông tin của các đề xuất tiềm năng (những đặc trưng của customer, article, transaction) thì hệ thống sẽ trả về một điểm số về mức độ “tốt” của đề xuất đó. Nếu ta coi mọi article là một đề xuất tiềm năng cho mỗi customer thì có quá nhiều khả năng (với bộ dữ liệu có khoảng 100 nghìn articles và 1.4 triệu customers sẽ tạo ra 140 tỷ ứng cử viên - quá nhiều và tốn kém để xem xét)

Cấu trúc tổng quan của hệ thống đề xuất gồm 2 giai đoạn:

1. RETRIEVAL — tạo ra một danh sách ngắn khoảng 30-40 các đề xuất tiềm năng (các ứng cử viên) cho mỗi khách hàng.
2. RANKING — với danh sách ứng cử viên có được, sau khi feed chúng qua mô hình ta xếp hạng chúng dựa trên điểm số tương thích và lấy ra danh sách có thứ tự của 12 ứng cử viên tốt nhất.

Phương pháp tiếp cận hai giai đoạn cho phép đưa ra các đề xuất từ một tập dữ liệu lớn (hàng triệu đề xuất) trong khi vẫn chắc chắn rằng số lượng nhỏ xuất hiện trên thiết bị được cá nhân hóa và hấp dẫn đối với người dùng. Hơn nữa, thiết kế này cho phép kết hợp các ứng cử viên được tạo ra bởi các nguồn khác.

## 4.2. Triển khai các thành phần

### **4.2.1. Candidate Retrieval (Recall)**

Hệ thống sử dụng một vài chiến lược phù hợp sau để đưa ra danh sách các ứng cử viên tiềm năng cho từng khách hàng:

* 1. Sử dụng BERT để embedding text features thành vector & KNN để đưa ra những items tương đồng  
     Bên cạnh các trường dữ liệu được thống kê của sản phẩm, thì trường “detail\_describe” là đoạn văn bản ngắn mô tả chi tiết sản phẩm và có thể chứa những thông tin hữu ích như chất liệu vải, tên nhà thiết kế, hoa văn, ... phản ánh rõ nét hơn về sản phẩm. Vì vậy nên chúng em cố gắng tận dụng trường thông tin này, ghép với các trường text còn lại để đưa ra mô hình BERT để học được vector đặc trưng. Đồng thời kết hợp với thuật toán KNN đủ nhanh để đưa ra gợi ý những sản phẩm tương tự tiềm năng cho khách hàng.
  2. Repurchase

Từ bộ dữ liệu, phát hiện ra rằng khách hàng có xu hướng mua lại các article mà họ vừa mua gần đây (thực tế, đã được chỉ ra rằng nhiều trong số những "mua lại" này có thể là các trao đổi sang các kích thước khác). Vì vậy, chiến lược này chỉ đơn giản đề xuất 12 articles cuối cùng mà một khách hàng đã mua gần đây. Nếu không có đủ 12 articles trong lịch sử gần đây của một khách hàng thì điền vào phần còn lại bằng các articles phổ biến gần đây.

* 1. Trending items weekly

Với đặc tính theo xu hướng của thời trang, thì những sản phẩm gần đây chắc chắn sẽ có nhiều ảnh hưởng và được khách hàng ưa thích hơn, đồng thời những sản phẩm đó sẽ có khả năng còn hàng nhiều hơn, gia tăng khả năng bán hàng.

### **4.2.2. Ranking**

Ngay từ đầu, chúng em quyết định phát triển các mô hình xếp hạng một cách riêng biệt, nhằm mục đích có những mô hình đa dạng để sau này có thể kết hợp các dự đoán. Mỗi phương pháp có những cách chuẩn bị dữ liệu training khác nhau.

Dữ liệu huấn luyện:

Trong ngữ cảnh của học máy xếp hạng, dữ liệu huấn luyện phải chứa cả các mẫu tích cực (positive samples) và tiêu cực (negative samples). Các mẫu tích cực là các giao dịch lịch sử thực sự, chúng em đánh dấu chúng là 1 (đã mua). Các mẫu tiêu cực là mẫu đánh dấu là 0 (không mua). Tại sao chúng ta cần các mẫu tiêu cực? Bởi vì chúng tôi sẽ sử dụng học máy xếp hạng không chỉ để sắp xếp sản phẩm mà khách hàng đã mua mà còn các sản phẩm mà khách hàng không mua. Sau đây là cách chúng em tạo ra các mẫu tiêu cực:

Các kỹ thuật tạo mẫu tiêu cực:

1. Sử dụng kết quả từ Retrieval vì nó chứa cả các mẫu tích cực và tiêu cực.
2. Chọn ngẫu nhiên N sản phẩm có sẵn làm mẫu tiêu cực.

Kỹ thuật tạo đặc trưng:

Trong ngữ cảnh của học máy xếp hạng, có 3 loại đặc trưng:

1. Đặc trưng Người dùng (User feature): Xuất phát từ dữ liệu của khách hàng. Ví dụ: tuổi, số giao dịch trong 90 ngày gần đây, giá trung bình trong 90 ngày gần đây, v.v.
2. Đặc trưng Sản phẩm (Item feature): Xuất phát từ dữ liệu của sản phẩm. Ví dụ: màu sắc sản phẩm, loại sản phẩm, giá sản phẩm, số lần mua trong 7 ngày gần đây, v.v.
3. Đặc trưng Người dùng-Sản phẩm (User-Item feature): Xuất phát từ sự tương tác giữa khách hàng và sản phẩm. Ví dụ: sự khác biệt giữa giá trung bình của khách hàng và giá sản phẩm, số lần khách hàng mua sản phẩm cùng màu sắc/loại sản phẩm/v.v. Quan trọng để có các đặc trưng Người dùng-Sản phẩm tốt để có được một mô hình hoạt động hiệu quả.

Mô hình học máy được sử dụng trong chiến lược xếp hạng là LGBMRanker từ thư viện LightGBM của Microsoft. Việc huấn luyện khá lâu nên các tham số được kiểm thử thủ công với 2 siêu tham số n\_estimators và max\_leaves lần lượt với các giá trị (400, 15), (300, 7), (200, 5).

### **4.2.3. Embedding**

The final submission là sự kết hợp của 5 mô hình khác nhau: 3 phiên bản của mô hình LightGBM, SVD Ranking và rule base gợi ý theo độ tuổi của khách hàng nhằm tăng sự đa dạng và chính xác của kết quả.

# **CHƯƠNG 5. KẾT QUẢ THỰC NGHIỆM**

Chương này sẽ trình bày chi tiết việc hiểu dữ liệu thông qua các công cụ trực quan hóa. Các kết quả thực nghiệm của phương pháp mô hình hóa chủ đề và kết quả triển khai hệ thống cũng được trình bày và phân tích ở chương này.

A close-up of a computer screen

Description automatically generated

Chiến lược chia tập dữ liệu giai đoạn huấn luyện

## 5.1. Đối với SVD



Đối với SVD, kết quả sẽ là 1432

## 5.3. Đối với LightGBM

## 

## 5.4. Đối với Ensemble



# **CHƯƠNG 6. KẾT LUẬN**

## 6.1. Kết luận

Số lượng article của H&M là rất lớn, cộng thêm thời gian huấn huyện ít, dẫn đến mô hình khó dự đoán trên tập kiểm thử. Ngoài ra, do tập dữ liệu rất lớn nên trong quá trình huấn luyện, việc quản lý tài nguyên hệ thống trở nên quan trọng. Do các phiên làm việc bị ngắt trong quá trình huấn luyện nên việc huấn luyện trở nên rất lâu. Do đó có nhiều hướng mà nhóm em chưa thử, giả dụ như CatBoost,…

## 6.2. Hướng phát triển trong tương lai

Dạng dữ liệu và bài toán khá mới dẫn đến việc tìm ra cách xử lý ban đầu trở nên khó khăn. Đặc biệt với kích thước dữ liệu lớn như vậy, việc xử lý dữ liệu sẽ mất nhiều thời gian, dẫn đến việc kiểm thử cách làm hạn chế. Tuy nhiên, việc đã hiểu cách thức và triển khai được sẽ là bước đệm để phát triển tiếp mô hình.

Các trường dữ liệu khác đang không được sử dụng để thêm thông tin vào giao dịch, các trường này đóng góp khá nhiều vào xu hướng mua hàng của khách hàng (ví dụ độ tuổi, mức giá). Khai thác được những trường này sẽ có ích cho mô hình dự đoán.

Sử dụng toàn bộ danh sách sản phẩm của H&M làm candidates cho mô hình có vẻ không hiệu quả, lí do là số lượng quá lớn, vừa gây tốn thời gian dự đoán, đồng thời mô hình sẽ tự phải yêu cầu có lượng thời gian huấn luyện đủ nhiều đủ bao gồm hết các sản phẩm này. Lí do là khi huấn luyện, các mẫu âm được lấy ngẫu nhiên nên nếu không huấn luyện đủ lâu, mô hình sẽ có thể không học được biểu diễn của các mẫu khác (chưa được lấy), đồng thời mô hình cũng cần phải học được cả biểu diễn của đầy đủ khách hàng.

Ngoài ra, hiện tượng cold start (các khách hàng là người mới, ít mua hàng) khiến cho việc mã hóa các khách hàng này là tương đối khó khi chỉ dựa vào lịch sử giao dịch, nhóm sẽ cần tìm hiểu thêm về các cách khắc phục (các rule-based về candidates gợi ý là điều khá tốt cho việc này, ví dụ như popular articles).

# **TÀI LIỆU THAM KHẢO**

[1] Bài giảng môn học Hệ gợi ý: Ngô Văn Linh, Đại học Bách Khoa Hà Nội, 2023.

[2] Neural News Recommendation with Attentive Multi-View Learning: Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, Xing Xie, 12 Jul 2019.

[3] Kaggle H&M Personalized Fashion Recommendations notebooks, 2022.