

**BỘ CÔNG THƯƠNG**  
**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

=====

**ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**  
**NGÀNH KHOA HỌC MÁY TÍNH**

**TÍCH HỢP HỆ THỐNG ĐỀ XUẤT SÁCH MLOPS CHO WEBSITE**  
**BÁN SÁCH BOOKIES**

**GVHD** : ThS. Lê Thị Thủy  
**Sinh viên** : Nguyễn Văn Việt  
**Mã số sinh viên** : 2021608149

## LỜI CẢM ƠN

Trong quá trình học tập và rèn luyện tại Đại Học Công Nghiệp Hà Nội và trong thời gian thực hiện thực tập tốt nghiệp đề tài “Tích hợp hệ thống đề xuất sách MLOps cho Bookies”, em xin gửi lời cảm ơn về sự giúp đỡ của các thầy cô, giảng viên, cán bộ trong Khoa Công nghệ thông tin đã giúp em có được kiến thức và hoàn thiện đề tài.

Đặc biệt, em xin gửi lời cảm ơn tới cô Lê Thị Thủy, cô đã trực tiếp hướng dẫn và chỉ bảo, trang bị kiến thức cho em trong suốt quá trình tìm hiểu và hoàn thành đề tài.

Cuối cùng, với sự cố gắng của bản thân nhưng với vốn kiến thức kỹ năng và kinh nghiệm xây dựng trang web còn yếu, đề tài của em không thể tránh khỏi những thiếu sót trong quá trình hoàn thiện. Vì vậy em rất mong nhận được nhận được sự đánh giá và nhận xét của thầy cô để em có thể khắc phục những thiếu sót của đề tài, giúp em nâng cao kiến thức của bản thân và hơn nữa là phục vụ cho công việc sau này.

*Em xin chân thành cảm ơn*

## MỤC LỤC

LỜI CẢM ƠN .....	i
DANH MỤC CÁC THUẬT NGỮ, KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT ...	iv
DANH MỤC HÌNH ẢNH .....	v
DANH MỤC BẢNG BIỂU .....	vi
MỞ ĐẦU .....	1
CHƯƠNG 1 KHẢO SÁT HỆ THỐNG .....	2
1.1 Tổng quan về đề tài.....	2
1.1.1 Giới thiệu chung.....	2
1.1.2 Tính cấp thiết của đề tài .....	2
1.1.3 Mục tiêu nghiên cứu.....	3
1.1.4 Đối tượng và phạm vi nghiên cứu.....	3
1.1.5 Phương pháp nghiên cứu.....	4
1.1.6 Tính thực tiễn của đề tài .....	5
1.2 Giới thiệu chung về hệ thống đề xuất và tổng quan về MLOps.....	5
1.2.1 Hệ thống đề xuất .....	5
1.2.2 Tổng quan về MLOps .....	7
1.2.3 Sơ đồ hệ thống đề xuất sách.....	8
CHƯƠNG 2 CƠ SỞ LÝ THUYẾT .....	12
2.1 Tổng quan về thuật toán đề xuất.....	12
2.1.1 Phân loại lọc cộng tác.....	12
2.1.2 Các kỹ thuật đánh giá hiệu quả hệ thống đề xuất. ....	13
2.2 Mô hình Factorization Machine (FM) .....	17
2.2.1 Giới thiệu về mô hình Factorization Machine .....	17

2.2.2 Cách hoạt động.....	17
2.3 Mô hình mạng neuron sâu(Deep Neuron Network – DNN) .....	19
2.3.1 Giới thiệu về mô hình học sâu. ....	19
2.3.2 Cách hoạt động của mô hình.....	20
2.4 Mô hình DeepFM .....	21
2.4.1 Giới thiệu về mô hình.....	21
2.4.2 Kiến trúc mô hình.....	21
2.4.3 Luồng dữ liệu trong mô hình DeepFM. ....	24
2.5 Ứng dụng của MLOPs trong hệ thống đề xuất sách.....	27
2.5.1 Vòng đời của một hệ thống MLOPs. ....	27
2.5.2 Lợi ích của MLOPs. ....	30
CHƯƠNG 3 XÂY DỰNG ỨNG DỤNG. ....	32
3.1 Thiết kế và triển khai module .....	32
3.1.1 Xử lý dữ liệu.....	32
3.1.2 Huấn luyện mô hình DeepFM.....	36
3.2 Kết quả thực nghiệm và đánh giá .....	41
KẾT LUẬN .....	42
TÀI LIỆU THAM KHẢO.....	43

## DANH MỤC CÁC THUẬT NGỮ, KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT

Viết tắt	Ý nghĩa
MLOPS	Machine Learning Operations
API	Application Programming Interface
FM	Factorization Machine
DNN	Deep Neuron Network
TPR	Machine Learning Operations
FPR	Application Programming Interface
ROC	Factorization Machine
AUC	Deep Neuron Network
DevOps	Machine Learning Operations
ML	Application Programming Interface
ReLU	Factorization Machine
JSON	Deep Neuron Network
CI/CD	Continuous Integration/Continuous Deployment

## DANH MỤC HÌNH ẢNH

Hình 1.1 Các thuật toán đề xuất .....	6
Hình 2.1 Cấu trúc model Factorization Machine .....	18
Hình 2.2 Hình minh họa Neuron Network .....	19
Hình 2.3 Hình minh họa cấu trúc của DeepFM .....	22
Hình 2.4 Hình minh họa luồng hoạt động của DeepFM .....	24
Hình 2.5 Quy trình MLOps trong hệ thống đề xuất sách .....	27
Hình 3.1 Code đọc dữ liệu .....	32
Hình 3.2 Loại bỏ các rating nhỏ hơn 0 .....	33
Hình 3.3 Đưa dữ liệu bảng địa chỉ về đúng định dạng .....	33
Hình 3.4 Sử dụng labelEncoding cho dữ liệu rời rạc .....	34
Hình 3.5 Chuẩn bị dữ liệu train và test .....	35
Hình 3.6 Thư mục chứa dữ liệu được làm sạch .....	35
Hình 3.7 Cấu hình mô hình DeepFM .....	36
Hình 3.8 Cấu hình quá trình huấn luyện của mô hình DeepFM .....	37
Hình 3.9 Các phiên bản model được lưu bằng MLflow .....	38
Hình 3.10 Lấy model trong MLflow .....	39
Hình 3.11 Dữ liệu nhận từ Client .....	39
Hình 3.12 Dữ liệu gửi tới Client .....	40
Hình 3.13 Lấy model trong MLflow .....	40
Hình 3.14 Chỉ số đánh giá model DeepFM cho hệ thống đề xuất sách .....	41

**DANH MỤC BẢNG BIỂU**

Bảng 1.1 Cấu trúc hệ thống MLOps .....	8
--	---

## MỞ ĐẦU

Trong thời đại công nghệ số, nhu cầu cá nhân hóa trải nghiệm người dùng ngày càng trở nên quan trọng, đặc biệt trong lĩnh vực thương mại điện tử và nội dung số. Các hệ thống đề xuất (Recommendation Systems) đã trở thành một trong những công nghệ cốt lõi giúp nâng cao mức độ hài lòng của người dùng và tăng hiệu quả kinh doanh. Website Bookies, một nền tảng bán sách trực tuyến, đang hướng đến việc ứng dụng trí tuệ nhân tạo để cải thiện khả năng gợi ý sách phù hợp cho từng người dùng, từ đó gia tăng thời gian tương tác, tỷ lệ mua hàng và sự trung thành với nền tảng.

Tuy nhiên, việc xây dựng một mô hình đề xuất hiệu quả không chỉ dừng lại ở khâu huấn luyện và triển khai mô hình. Hệ thống cần được giám sát, cập nhật liên tục và tích hợp chặt chẽ với quy trình vận hành thực tế. Do đó, việc áp dụng MLOps (Machine Learning Operations) – một tập hợp các quy trình và công cụ nhằm tự động hóa toàn bộ vòng đời của mô hình học máy – là một hướng tiếp cận hiện đại, giúp đảm bảo hiệu suất, tính ổn định và khả năng mở rộng của hệ thống đề xuất.

Đề tài “Xây dựng hệ thống MLOps đề xuất sách cho website Bookies” nhằm mục tiêu thiết kế và phát triển một hệ thống đề xuất tích hợp MLOps, bao gồm các bước từ thu thập dữ liệu, huấn luyện mô hình, triển khai, giám sát đến cập nhật liên tục. Qua đó, đề tài không chỉ mang lại giá trị thực tiễn cho Bookies mà còn góp phần nghiên cứu và ứng dụng các xu hướng công nghệ tiên tiến trong lĩnh vực trí tuệ nhân tạo và kỹ thuật phần mềm.



## CHƯƠNG 1 KHẢO SÁT HỆ THỐNG

### 1.1 Tổng quan về đề tài.

#### 1.1.1 Giới thiệu chung.

Trong bối cảnh công nghệ phát triển vượt bậc, việc xử lý và khai thác thông tin trên các nền tảng thương mại điện tử trở nên phức tạp hơn bao giờ hết. Người dùng thường đối mặt với tình trạng quá tải thông tin khi phải lựa chọn giữa hàng ngàn sản phẩm và dịch vụ khác nhau. Để giải quyết vấn đề này, các hệ thống gợi ý (Recommendation Systems) đã được ứng dụng rộng rãi, đóng vai trò như một công cụ hỗ trợ mạnh mẽ giúp cá nhân hóa trải nghiệm người dùng. Bằng cách phân tích dữ liệu hành vi, sở thích và lịch sử mua sắm, các hệ thống này không chỉ giúp người dùng tìm kiếm sản phẩm phù hợp một cách nhanh chóng mà còn góp phần tối ưu hóa doanh thu và chiến lược kinh doanh của các doanh nghiệp. Đây chính là một bước tiến quan trọng trong việc nâng cao hiệu quả hoạt động thương mại điện tử và đáp ứng nhu cầu ngày càng đa dạng của thị trường.

#### 1.1.2 Tính cấp thiết của đề tài

Đề tài “Tích hợp hệ thống đề xuất cho website Bookies” hướng đến việc nghiên cứu, xây dựng và tích hợp một hệ thống gợi ý sách thông minh vào nền tảng thương mại điện tử Bookies – một website chuyên cung cấp sách cho nhiều đối tượng người dùng.

Thông qua việc áp dụng các thuật toán học máy, kết hợp với quy trình triển khai MLOps nhằm đảm bảo khả năng mở rộng, cập nhật và giám sát mô hình hiệu quả, hệ thống đề xuất sẽ giúp người dùng:

- Khám phá những cuốn sách phù hợp với sở thích cá nhân.
- Tiết kiệm thời gian tìm kiếm.
- Và nâng cao sự hài lòng trong quá trình trải nghiệm.

Bên cạnh đó, hệ thống đề xuất sách giúp website Bookies:

- Tăng tỷ lệ tương tác và chuyển đổi.

- Thúc đẩy doanh thu.
- Giữ chân người dùng trung thành.
- Tạo lợi thế cạnh tranh so với các nền tảng cùng lĩnh vực.

Đề tài không chỉ tập trung vào thuật toán đề xuất mà còn chú trọng vào việc triển khai hệ thống trong môi trường thực tế, bao gồm: kiến trúc hệ thống, pipeline xử lý dữ liệu, tích hợp frontend/backend, và đo lường hiệu quả hệ thống.

### 1.1.3 Mục tiêu nghiên cứu.

Đề tài hướng đến việc phát triển một hệ thống đề xuất thông minh, từ đó mang lại lợi ích cho cả người dùng lẫn nền tảng thương mại điện tử Bookies. Các mục tiêu cụ thể của nghiên cứu bao gồm:

- Cá nhân hóa trải nghiệm người dùng, giúp khách hàng tiếp cận dễ dàng với những nội dung sách yêu thích.
- Tăng doanh số bán hàng khi gợi ý những cuốn sách phù hợp với khách hàng
- Tối ưu hóa quy trình vận hành của hệ thống bằng cách tích hợp hệ thống MLOPS tự động
- Giúp cửa hàng Bookies quản lý danh mục sản phẩm tốt hơn, phân loại sách dựa trên nhu cầu, và dự đoán được các xu hướng sách sắp tới, từ đó giúp tối ưu hóa việc nhập hàng và phân phối.

### 1.1.4 Đối tượng và phạm vi nghiên cứu.

Đối tượng nghiên cứu của đề tài là hệ thống đề xuất sách dựa trên các kỹ thuật học máy và khai thác dữ liệu người dùng. Cụ thể:

- Mô hình gợi ý gợi ý: DeepFM.
- Quy trình triển khai hệ thống học máy trong môi trường thực tế, bao gồm cả MLOps.
- Tập dữ liệu người dùng và sản phẩm từ website Bookies: bao gồm thông tin sách, hành vi người dùng (đánh giá của người dùng)

- Kiến trúc hệ thống tích hợp giữa mô hình đề xuất và nền tảng website Bookies.

Đề tài tập trung vào các nội dung sau:

- Xây dựng hệ thống gợi ý sách dựa trên dữ liệu có sẵn của website Bookies.
- Tích hợp hệ thống đề xuất với website Bookies thông qua API, đảm bảo hoạt động ổn định và phản hồi nhanh.
- Triển khai và quản lý mô hình học máy sử dụng quy trình MLOps, bao gồm: huấn luyện, kiểm thử, triển khai, giám sát và cập nhật mô hình.
- Đánh giá hiệu quả mô hình đề xuất thông qua các chỉ số như: Precision, Recall, F1-score...

#### 1.1.5 Phương pháp nghiên cứu.

Để đạt được các mục tiêu nghiên cứu đã đề ra, đề tài sử dụng các phương pháp nghiên cứu sau:

- Phương pháp thu thập và phân tích dữ liệu: sử dụng bộ dữ liệu tương tác của người dùng trên Kaggle.
- Phương pháp xây dựng mô hình đề xuất: nghiên cứu các kỹ thuật xây dựng hệ thống gợi ý như lọc cộng tác, lọc theo nội dung và lọc kết hợp.
- Phương pháp triển khai và tích hợp hệ thống: sử dụng kiến trúc microservices để triển khai mô hình gợi ý độc lập với hệ thống chính của website Bookies. Xây dựng RESTful API để kết nối mô hình với frontend. Đồng thời, áp dụng quy trình MLOps để tự động hóa pipeline gồm các bước: tiền xử lý, huấn luyện, triển khai, giám sát và cập nhật mô hình.
- Phương pháp đánh giá mô hình: Sử dụng các chỉ số phổ biến để đánh giá độ chính xác của mô hình như Precision, Recall, F1-score, AUC, Logloss, Accuracy

### 1.1.6 Tính thực tiễn của đề tài

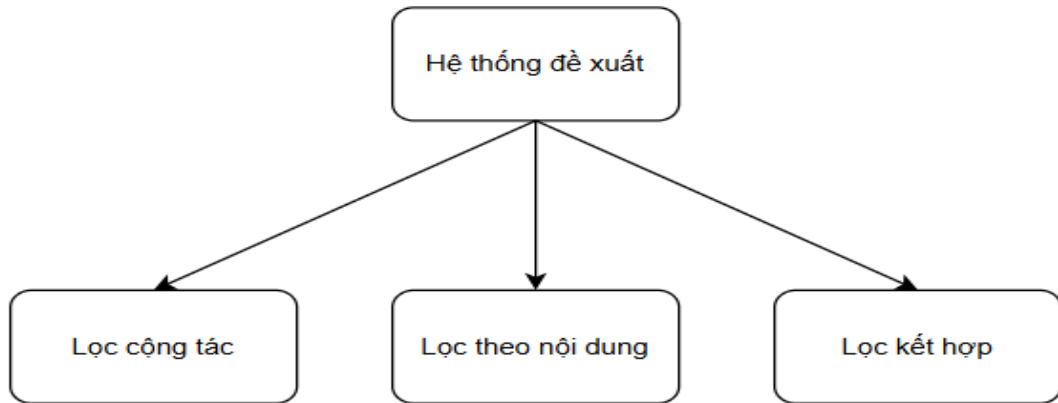
Dự án “Tích hợp hệ thống đề xuất cho website Bookies” mang lại nhiều lợi ích thực tiễn quan trọng trong bối cảnh thương mại điện tử và bán sách trực tuyến hiện nay. Một số lợi ích chính có thể kể đến như sau:

- Hệ thống đề xuất giúp người dùng giải quyết vấn đề quá tải thông tin khi phải đối mặt với một loạt các sản phẩm trên các nền tảng thương mại điện tử. Bằng cách cung cấp các gợi ý sách phù hợp với sở thích cá nhân, hệ thống không chỉ tiết kiệm thời gian mà còn nâng cao trải nghiệm người dùng, đặc biệt là đối với những người yêu sách có gu cá nhân rõ rệt.
- Từ góc độ doanh nghiệp, việc áp dụng hệ thống đề xuất chính xác có thể thúc đẩy doanh số bán hàng thông qua việc khuyến khích khách hàng mua thêm sách. Hơn nữa, hệ thống này còn cung cấp dữ liệu phân tích hành vi người dùng, hỗ trợ các bộ phận kinh doanh trong việc xây dựng chiến lược marketing hiệu quả, quản lý tồn kho và nhập hàng phù hợp với xu hướng tiêu dùng.
- Hệ thống có khả năng mở rộng và áp dụng cho nhiều lĩnh vực khác như thời trang, mỹ phẩm, thiết bị điện tử nhờ vào kiến trúc modular và ứng dụng MLOps. Điều này giúp doanh nghiệp dễ dàng nâng cấp và triển khai hệ thống mà không gây gián đoạn hoạt động kinh doanh.

## 1.2 Giới thiệu chung về hệ thống đề xuất và tổng quan về MLOps

### 1.2.1 Hệ thống đề xuất

Hệ thống đề xuất (Recommendation System) là một công cụ quan trọng trong việc cá nhân hóa trải nghiệm người dùng trên các nền tảng thương mại điện tử và nội dung số. Với sự hỗ trợ của trí tuệ nhân tạo và học máy, các hệ thống này có thể đưa ra các đề xuất dựa trên hành vi và sở thích của người dùng.



Hình 1.1 Các thuật toán đề xuất

- Các phương pháp phổ biến:
  - Hệ thống đề xuất dựa trên nội dung: phương pháp đề xuất nội dung dựa trên sự tương tự giữa các mục nội dung. Thuật toán này sử dụng thông tin về nội dung của các mục để đề xuất các mục tương tự. Các phương pháp lọc nội dung bao gồm sử dụng thông tin từ người dùng (user-based content filtering) và sử dụng thông tin từ sản phẩm (item-based content filtering).
  - Hệ thống đề xuất dựa trên lọc cộng tác: là phương pháp sử dụng thông tin từ người dùng khác để đề xuất nội dung tương tự. Thuật toán này dựa trên nguyên lý rằng những người dùng có sở thích tương tự sẽ thích những nội dung tương tự. Các phương pháp lọc cộng tác bao gồm lọc cộng tác dựa trên người dùng (user-based collaborative filtering) và lọc cộng tác dựa trên sản phẩm (item-based collaborative filtering).
  - Hệ thống đề xuất kết hợp: phương pháp kết hợp cả hai phương pháp trên để đề xuất nội dung. Thuật toán này sử dụng cả thông tin từ người dùng và thông tin về nội dung của các mục để đề xuất nội dung phù hợp nhất. Các phương pháp lọc kết hợp bao gồm việc sử dụng kỹ thuật học máy để kết hợp cả hai phương pháp.

- Vai trò của hệ thống đề xuất:
  - Tăng cường trải nghiệm người dùng: Gợi ý những cuốn sách phù hợp giúp người dùng tiết kiệm thời gian tìm kiếm và khám phá được những đầu sách giá trị.
  - Tăng tỷ lệ tương tác và doanh thu: Những đề xuất chính xác thúc đẩy người dùng mua hoặc đọc nhiều sách hơn.
  - Cá nhân hóa nội dung: Hệ thống hiểu người dùng hơn qua dữ liệu lịch sử và hành vi, từ đó tạo ra trải nghiệm cá nhân hóa.

### 1.2.2 Tổng quan về MLOps

MLOps (Machine Learning Operations) là một lĩnh vực quan trọng trong việc ứng dụng học máy vào thực tiễn, tập trung vào việc tối ưu hóa quy trình triển khai, giám sát và bảo trì các mô hình học máy. MLOps không chỉ bao gồm các công cụ kỹ thuật mà còn là sự kết hợp giữa các phương pháp quản lý, tự động hóa và tích hợp liên tục nhằm đảm bảo các mô hình hoạt động hiệu quả, đáng tin cậy và dễ dàng mở rộng. Điều này đặc biệt quan trọng trong bối cảnh các tổ chức ngày càng phụ thuộc vào dữ liệu và trí tuệ nhân tạo để đưa ra quyết định chiến lược. MLOps giúp giảm thiểu rủi ro, cải thiện hiệu suất và đảm bảo rằng các mô hình học máy có thể thích ứng với những thay đổi trong môi trường thực tế một cách nhanh chóng và linh hoạt.

- Vai trò của MLOps:
  - Tự động hóa quy trình huấn luyện và triển khai mô hình: Giảm sự phụ thuộc vào thao tác thủ công.
  - Đảm bảo tái sử dụng và khả năng mở rộng: Cho phép dễ dàng nâng cấp mô hình hoặc thay đổi dữ liệu đầu vào.
  - Giám sát hiệu suất mô hình theo thời gian: Theo dõi độ chính xác của mô hình gợi ý sau khi được tích hợp trên website Bookies.
- Các thành phần chính của quy trình MLOps:
  - Data Pipeline: Thu thập, xử lý, lưu trữ dữ liệu người dùng và dữ liệu sách.

- Training Pipeline: Huấn luyện các mô hình đề xuất với các thuật toán như lọc cộng tác, lọc theo nội dung và lọc kết hợp.
- Serving Pipeline: Triển khai mô hình dưới dạng API để tích hợp với website Bookies.
- Monitoring and Retraining: Theo dõi hiệu suất, đánh giá độ chính xác và tái huấn luyện mô hình khi cần thiết để duy trì độ chính xác.

### 1.2.3 Sơ đồ hệ thống đề xuất sách.

Hệ thống MLOps đề xuất sách gồm có 4 thành phần chính bao gồm: Data pipeline (Luồng dữ liệu), Training Pipeline (Luồng huấn luyện mô hình), Serving Pipeline (Luồng triển khai), Model Registry (Đăng ký mô hình), Observability (Giám sát hệ thống).

Bảng 1.1 Cấu trúc hệ thống MLOps

STT	Thành phần	Mục đích	Quy trình
1	Data Pipeline	Quá trình này bao gồm việc thu thập, làm sạch và xử lý dữ liệu để tạo ra các đặc trưng mang tính đại diện, giúp mô hình hiểu rõ hơn về xu hướng và hành vi của người dùng. Các kỹ thuật phổ biến như mã hóa dữ liệu, chuẩn hóa, và trích xuất đặc trưng được áp dụng để đảm bảo tính chính xác và hiệu quả của mô hình. Sau khi hoàn thiện, các đặc trưng này cần được lưu trữ một cách có tổ chức và an toàn, sử dụng các cơ sở dữ liệu hoặc hệ thống lưu trữ phù hợp để dễ dàng truy cập và quản lý.	1. Thu thập dữ liệu 2. Xử lý các giá trị null, giá trị không hợp lệ 3. Lưu lại dữ liệu làm sạch

		Điều này không chỉ hỗ trợ trong việc cải thiện hiệu suất của mô hình mà còn tạo nền tảng cho việc mở rộng và tối ưu hóa trong tương lai.	
2	Trainning Pipeline	<p>Quá trình này bao gồm việc chuẩn bị dữ liệu, lựa chọn mô hình phù hợp, tối ưu hóa các tham số và đánh giá hiệu suất của mô hình. Đầu tiên, dữ liệu cần được làm sạch và tiền xử lý để đảm bảo chất lượng và tính nhất quán. Sau đó, việc lựa chọn mô hình phụ thuộc vào bài toán cụ thể, có thể là phân loại, hồi quy hoặc các bài toán phức tạp hơn như xử lý ngôn ngữ tự nhiên hay nhận diện hình ảnh. Trong giai đoạn huấn luyện, mô hình sẽ học cách tối ưu hóa các tham số dựa trên tập dữ liệu huấn luyện nhằm giảm thiểu sai số. Sau khi hoàn thành, mô hình cần được đánh giá trên tập dữ liệu kiểm tra để đảm bảo khả năng tổng quát hóa và hiệu quả khi áp dụng vào thực tế. Việc thực hiện đúng quy trình huấn luyện</p>	<p>1. Kéo dữ liệu đã đã được làm sạch để sử dụng trong huấn luyện.</p> <p>2. Huấn luyện mô hình học máy</p> <p>3. Lưu mô hình và số liệu liên quan vào bằng Mlflow</p>



		không chỉ giúp cải thiện hiệu suất mà còn đảm bảo tính đáng tin cậy của hệ thống học máy.	
3	Monitoring	<p>Module này cung cấp một giải pháp toàn diện để theo dõi, ghi nhận và quản lý quá trình huấn luyện mô hình học máy một cách hệ thống và minh bạch. Thông qua tích hợp với MLflow, người dùng có thể dễ dàng ghi lại chi tiết từng lần huấn luyện, bao gồm các siêu tham số, độ chính xác, độ lỗi và các thông tin quan trọng khác. Đồng thời, module này hỗ trợ lưu trữ và quản lý mô hình một cách hiệu quả, đảm bảo rằng các mô hình sau khi huấn luyện được bảo toàn đầy đủ để phục vụ việc tải lại, triển khai hoặc so sánh trong tương lai. Khả năng so sánh hiệu suất giữa các phiên bản mô hình giúp tối ưu hóa quy trình lựa chọn mô hình tốt nhất. Đặc biệt, việc hỗ trợ tái hiện thí nghiệm thông qua việc log chi tiết từng lần chạy đảm bảo khả năng tái lập kết quả một cách</p>	<p>1. Khởi tạo experiment trong MLflow để tổ chức và theo dõi các lần huấn luyện mô hình</p> <p>2. Thực hiện huấn luyện mô hình và tính toán các chỉ số đánh giá như loss và accuracy.</p> <p>3. Ghi lại các chỉ số đánh giá (loss, accuracy, v.v.) vào hệ thống MLflow để theo dõi hiệu suất mô hình.</p> <p>4. Lưu trữ mô hình đã huấn luyện và các tệp liên quan dưới dạng artifacts.</p>

		đáng tin cậy. Ngoài ra, giao diện trực quan MLflow UI cho phép người dùng dễ dàng giám sát tiến trình và trực quan hóa kết quả huấn luyện, mang lại sự tiện lợi và hiệu quả trong quản lý dự án học máy.	
4	Serving Pipeline	Đảm bảo khả năng triển khai mô hình học máy vào môi trường thực tế một cách hiệu quả, tự động và ổn định, nhằm cung cấp các gợi ý sách phù hợp cho người dùng cuối. Quy trình này giúp kết nối liền mạch giữa mô hình đã huấn luyện và hệ thống website Bookies thông qua các bước như tải mô hình, xử lý dữ liệu đầu vào và trả về kết quả đề xuất qua API.	<ol style="list-style-type: none"> <li>1. Tải mô hình từ MLflow để thực hiện dự đoán</li> <li>2. Gửi đi API chứa dữ liệu đề xuất.</li> </ol>

## CHƯƠNG 2 CƠ SỞ LÝ THUYẾT

### 2.1 Tổng quan về thuật toán đề xuất.

Badrul Sarwar, nhà nghiên cứu tại GroupLens Research, đã nhấn mạnh rằng: “Hệ thống đề xuất là một công nghệ được thiết kế để cá nhân hóa các trải nghiệm trực tuyến bằng cách tự động dự đoán những gì người dùng có thể quan tâm, dựa trên hành vi của họ hoặc của những người dùng tương tự.”. Yehuda Koren, nhà khoa học tại Google và từng làm việc tại Netflix, bổ sung rằng: “Hệ thống đề xuất không chỉ là công cụ để gợi ý, mà còn là cách giúp người dùng khám phá nội dung hoặc sản phẩm mới mà họ có thể không nhận ra là mình cần”. Paul Resnick, giáo sư tại Đại học Michigan, đã chỉ ra rằng: “Hệ thống đề xuất là một giải pháp cho vấn đề quá tải thông tin, bằng cách chọn lọc và gợi ý những nội dung phù hợp với sở thích và nhu cầu của người dùng”. Joseph A. Konstan, đồng sáng lập GroupLens Research, nhấn mạnh rằng: “Hệ thống đề xuất là một công cụ để tối ưu hóa sự tương tác giữa người dùng và nội dung, bằng cách cung cấp các gợi ý dựa trên dữ liệu lịch sử và mô hình hành vi”. Cuối cùng, Andreas Hotho, một chuyên gia về khai phá dữ liệu, cho rằng: “Hệ thống đề xuất là một phần quan trọng của các hệ thống thông tin, đóng vai trò quyết định trong việc hỗ trợ quyết định của người dùng và cá nhân hóa trải nghiệm”.

Tóm lại, hệ thống đề xuất (Recommendation System) là một công nghệ thông minh được thiết kế để cá nhân hóa trải nghiệm người dùng thông qua việc tự động gợi ý các sản phẩm, dịch vụ, hoặc nội dung dựa trên sở thích và hành vi của họ. Mục tiêu chính của hệ thống này là dự đoán những gì người dùng có thể quan tâm, giúp họ khám phá những tùy chọn mới mà có thể họ chưa nhận ra.

#### 2.1.1 Phân loại lọc cộng tác

Lọc cộng tác là phương pháp sử dụng thông tin từ người dùng khác để đề xuất nội dung tương tự. Thuật toán này dựa trên nguyên lý rằng những người dùng có sở thích tương tự sẽ thích những nội dung tương tự. Các phương pháp lọc cộng tác bao gồm như sau: lọc cộng tác dựa trên người dùng (user-based

collaborative filtering) và lọc cộng tác dựa trên sản phẩm (item-based collaborative filtering).

Lọc nội dung là phương pháp đề xuất nội dung dựa trên sự tương tự giữa các mục nội dung. Thuật toán này sử dụng thông tin về nội dung của các mục để đề xuất các mục tương tự. Các phương pháp lọc nội dung bao gồm sử dụng thông tin từ người dùng (user-based content filtering) và sử dụng thông tin từ sản phẩm (item-based content filtering).

Lọc kết hợp là phương pháp kết hợp cả hai phương pháp trên để đề xuất nội dung. Thuật toán này sử dụng cả thông tin từ người dùng và thông tin về nội dung của các mục để đề xuất nội dung phù hợp nhất. Các phương pháp lọc kết hợp bao gồm việc sử dụng kỹ thuật học máy để kết hợp cả hai phương pháp.

### 2.1.2 Các kỹ thuật đánh giá hiệu quả hệ thống đề xuất.

#### 2.1.2.1 Accuracy (Độ chính xác)

Độ chính xác là tỷ lệ phần trăm các mẫu dữ liệu được phân loại chính xác bởi mô hình. Accuracy được tính bằng công thức như sau:

$$Accuracy = (TP + TN) / (TP + TN + FP + FN)$$

Trong đó:

- TP (True Positive): Dự đoán đúng là dương (1)
- TN (True Negative): Dự đoán đúng là âm (0)
- FP (False Positive): Dự đoán sai là dương
- FN (False Negative): Dự đoán sai là âm

Độ chính xác là thước đo trực quan và dễ hiểu nhất về hiệu suất phân loại của mô hình. Tuy nhiên, nó có thể bị ảnh hưởng bởi tỷ lệ phần trăm các lớp trong tập dữ liệu. Ví dụ, nếu tập dữ liệu có 90% là mẫu thuộc lớp A và 10% là mẫu thuộc lớp B, mô hình chỉ cần dự đoán tất cả các mẫu là A cũng đạt độ chính xác 90%, nhưng hiệu quả phân loại thực tế không tốt.

#### 2.1.2.2 Log Loss (Binary Cross-Entropy)

Log Loss, hay còn gọi là Logistic Loss hoặc Binary Cross-Entropy, là một hàm mất mát quan trọng được sử dụng phổ biến trong các bài toán phân

loại, đặc biệt là phân loại nhị phân. Hàm này đo lường mức độ sai lệch giữa xác suất dự đoán của mô hình và nhãn thực tế, từ đó giúp tối ưu hóa mô hình trong quá trình huấn luyện.

Giá trị Log Loss càng nhỏ thì mô hình càng có khả năng dự đoán chính xác hơn, đồng nghĩa với việc xác suất dự đoán gần với nhãn thực tế hơn. Công thức tính Log Loss được xây dựng dựa trên lý thuyết xác suất và entropy, đảm bảo rằng các dự đoán sai lệch sẽ bị phạt nặng hơn, đặc biệt là khi mô hình dự đoán với độ tự tin cao nhưng lại sai. Điều này khiến Log Loss trở thành một công cụ hiệu quả để đánh giá và cải thiện hiệu suất của các mô hình học máy trong việc xử lý dữ liệu phân loại. Công thức như sau:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

Trong đó:

- $Y_i \in \{0,1\}$ : Nhãn thực tế
- $y^i \in (0,1)$ : Xác suất dự đoán
- $N$ : Tổng số mẫu

### 2.1.2.3 AUC-ROC

AUC - ROC là một công cụ quan trọng trong việc đánh giá hiệu suất của các mô hình phân loại, đặc biệt khi mô hình được sử dụng để phân biệt giữa hai lớp. AUC (Area Under the Curve) đo lường diện tích dưới đường cong ROC (Receiver Operating Characteristic), biểu thị khả năng của mô hình trong việc phân biệt đúng giữa các lớp.

Một mô hình có giá trị AUC gần 1 cho thấy khả năng phân loại tốt, trong khi giá trị AUC gần 0.5 cho thấy hiệu suất phân loại không tốt hơn so với dự đoán ngẫu nhiên. Phương pháp này không chỉ cung cấp cái nhìn tổng quan về hiệu quả của mô hình mà còn giúp so sánh và lựa chọn mô hình tối ưu trong các bài toán phân loại.

Đường cong ROC (Receiver Operating Characteristic) là công cụ quan trọng trong phân tích và đánh giá hiệu suất của các mô hình phân loại, đặc biệt trong lĩnh vực học máy và khai phá dữ liệu. Biểu đồ này minh họa mối quan hệ giữa TPR (True Positive Rate - tỷ lệ dương tính đúng) và FPR (False Positive Rate - tỷ lệ dương tính sai) khi thay đổi ngưỡng phân loại, cho phép người dùng đánh giá khả năng phân biệt giữa các lớp của mô hình. Một mô hình lý tưởng sẽ có đường cong ROC đi sát góc trên bên trái, ứng với TPR cao và FPR thấp.

Một điểm đáng chú ý là AUC không bị ảnh hưởng bởi sự mất cân bằng dữ liệu, điều này giúp nó trở thành một chỉ số tin cậy trong nhiều trường hợp khi các lớp không đồng đều về số lượng. Việc sử dụng AUC - ROC hỗ trợ các nhà nghiên cứu và kỹ sư dữ liệu đánh giá chính xác khả năng phân loại của mô hình, từ đó đưa ra các cải thiện phù hợp.

#### 2.1.2.4 Precision (Độ chính xác của lớp dương)

Độ chính xác (Precision) là một chỉ số quan trọng trong lĩnh vực học máy và phân loại, được sử dụng để đánh giá mức độ chính xác của mô hình khi dự đoán một lớp cụ thể. Precision được định nghĩa là tỷ lệ giữa số lượng mẫu được dự đoán đúng thuộc một lớp so với tổng số mẫu được dự đoán thuộc lớp đó. Precision được tính bằng công thức như sau:

$$Precision = TP / (TP + FP)$$

Trong đó:

- TP (True Positive): Số lượng mẫu thuộc lớp A được dự đoán đúng là A.
- FP (False Positive): Số lượng mẫu thuộc lớp B được dự đoán sai là A.

Độ chính xác cho biết tỷ lệ tin cậy của các dự đoán dương tính (dự đoán là A) của mô hình. Giá trị Precision cao nghĩa là mô hình ít dự đoán sai âm tính (FP). Tuy nhiên, Precision cao không đồng nghĩa với độ chính xác cao, vì nó phụ thuộc vào tỷ lệ FP.

- Ưu điểm:
  - Cho biết tỷ lệ tin cậy của các dự đoán dương tính.
  - Phù hợp cho các bài toán cần ít sai sót dương tính.

- Nhược điểm:
  - Phụ thuộc vào tỷ lệ FP.
  - Không cung cấp thông tin về hiệu suất dự đoán cho các mẫu âm tính.

#### 2.1.2.5 Recall (Độ bao phủ)

Độ nhạy, hay còn được gọi là Recall, là một chỉ số quan trọng trong các bài toán phân loại, đặc biệt trong lĩnh vực học máy và khai phá dữ liệu. Chỉ số này thể hiện khả năng của mô hình trong việc nhận diện đúng các mẫu thuộc một lớp cụ thể. Nói cách khác, độ nhạy đo lường tỷ lệ giữa số lượng mẫu dương thực sự được dự đoán đúng và tổng số mẫu dương thực sự. Recall được tính bằng công thức như sau:

$$Recall = TP / (TP + FN)$$

Trong đó:

- TP (True Positive): Số lượng mẫu thuộc lớp A được dự đoán đúng là A.
- FP (False Positive): Số lượng mẫu thuộc lớp B được dự đoán sai là A.

Độ nhạy cho biết tỷ lệ bao quát của các dự đoán dương tính của mô hình. Giá trị Recall cao nghĩa là mô hình ít dự đoán sai dương tính (FN). Recall cao thường đi kèm với Precision thấp và ngược lại, do sự đánh đổi giữa việc giảm thiểu FP và FN.

#### 2.1.2.6 F1-Score

F1-Score là một chỉ số quan trọng trong lĩnh vực học máy và xử lý dữ liệu, được sử dụng để đánh giá hiệu suất của các mô hình phân loại. Đây là trung bình điều hòa giữa Precision (độ chính xác) và Recall (khả năng thu hồi), hai thước đo thường được sử dụng để phân tích sự cân bằng giữa các kết quả dự đoán đúng và sai. Công thức tính F1-Score được biểu diễn như sau:

$$F1 = 2 * (Precision * Recall) / (Precision + Recall)$$

Chỉ số này đặc biệt hữu ích trong các bài toán mà dữ liệu không cân bằng, tức là khi số lượng mẫu thuộc các lớp khác nhau có sự chênh lệch lớn. Trong những trường hợp này, việc chỉ dựa vào Precision hoặc Recall riêng lẻ có thể dẫn đến đánh giá sai lệch về hiệu suất của mô hình. F1-Score giúp cung cấp

một cái nhìn toàn diện hơn, đảm bảo rằng mô hình không chỉ tập trung vào việc tăng độ chính xác mà còn cải thiện khả năng nhận diện đầy đủ các mẫu quan trọng.

- Ưu điểm: Đánh giá tổng thể hiệu suất phân loại cho một lớp Cân bằng giữa Precision và Recall.
- Nhược điểm: Không cung cấp thông tin chi tiết về hiệu suất cho từng loại sai sót.

## 2.2 Mô hình Factorization Machine (FM)

### 2.2.1 Giới thiệu về mô hình Factorization Machine

Factorization Machine – FM được đề xuất bởi Rendle (2010), là một thuật toán học có giám sát, có thể sử dụng cho các nhiệm vụ phân loại, hồi quy và xếp hạng. Đây là một phương pháp phổ biến và có ảnh hưởng lớn trong việc dự đoán và đưa ra gợi ý, đặc biệt phù hợp cho mục đích đề xuất sản phẩm sách của đề tài. Điểm mạnh của FM so với hồi quy tuyến tính và phân tích ma trận là:

- Khả năng mô hình hóa tương tác giữa các biến ở bậc cao, thường được thiết lập ở bậc hai
- Thuật toán tối ưu nhanh của FM giúp giảm thời gian tính toán đa thức xuống độ phức tạp tuyến tính, rất hiệu quả đối với dữ liệu đầu vào thưa và có chiều cao.
- FM khắc phục dữ liệu thưa (sparse): Bằng cách học các vector tiềm ẩn riêng biệt cho từng đặc trưng

Nhờ những ưu điểm này, FM được áp dụng rộng rãi trong các hệ thống đề xuất hiện đại.

### 2.2.2 Cách hoạt động.

Xét một cách hình thức, giả sử  $X$  là vector đặc trưng đại diện cho một mẫu dữ liệu, và  $y$  là nhãn tương ứng, có thể là một giá trị thực hoặc là nhãn phân loại. Trong ngữ cảnh hệ thống đề xuất sách,  $X$  bao gồm các đặc trưng như người dùng, sách, tác giả, và các thông tin bổ sung (như thể loại, số lượng đánh



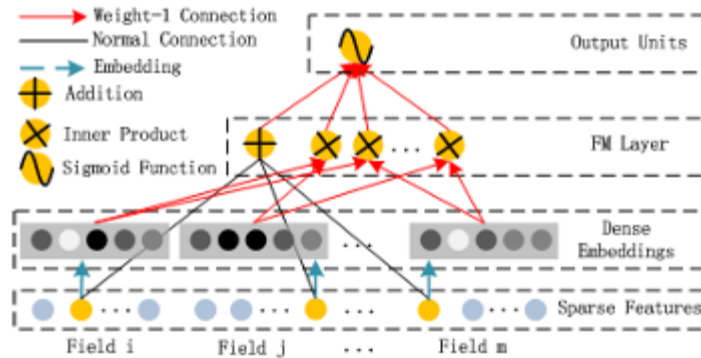
giá, đánh giá trung bình), còn  $y$  là điểm đánh giá do người dùng gán cho cuốn sách.

Mô hình Factorization Machine (FM) bậc hai được định nghĩa như sau:

$$\hat{y}(x) = \mathbf{w}_0 + \sum_{i=1}^d \mathbf{w}_i x_i + \sum_{i=1}^d \sum_{j=i+1}^d \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j$$

Trong đó:

- $\mathbf{w}_0$  là hệ số thiên lệch toàn cục (global bias);
- $\mathbf{w}_i$  là trọng số của biến đặc trưng thứ  $i$ ;
- $\mathbf{v}_i \in \mathbb{R}^k$  là vector nhúng (embedding) của đặc trưng  $i$ ;
- $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$  tích vô hướng giữa hai vector nhúng;
- $k$  là số chiều của không gian tiềm ẩn.



Hình 2.1 Cấu trúc model Factorization Machine

Trong hệ thống đề xuất sách, điều này cho phép mô hình tự động học được mối quan hệ giữa người dùng và sách, hoặc giữa tác giả và thể loại, mà không cần phải thiết kế thủ công các đặc trưng tương tác. Trong trường hợp đặc trưng  $x_i$  biểu diễn sách và  $x_j$  biểu diễn người dùng, thì thuật ngữ tương tác chính là tích vô hướng giữa vector nhúng người dùng và vector nhúng sách - tương tự như trong mô hình phân rã ma trận (matrix factorization).

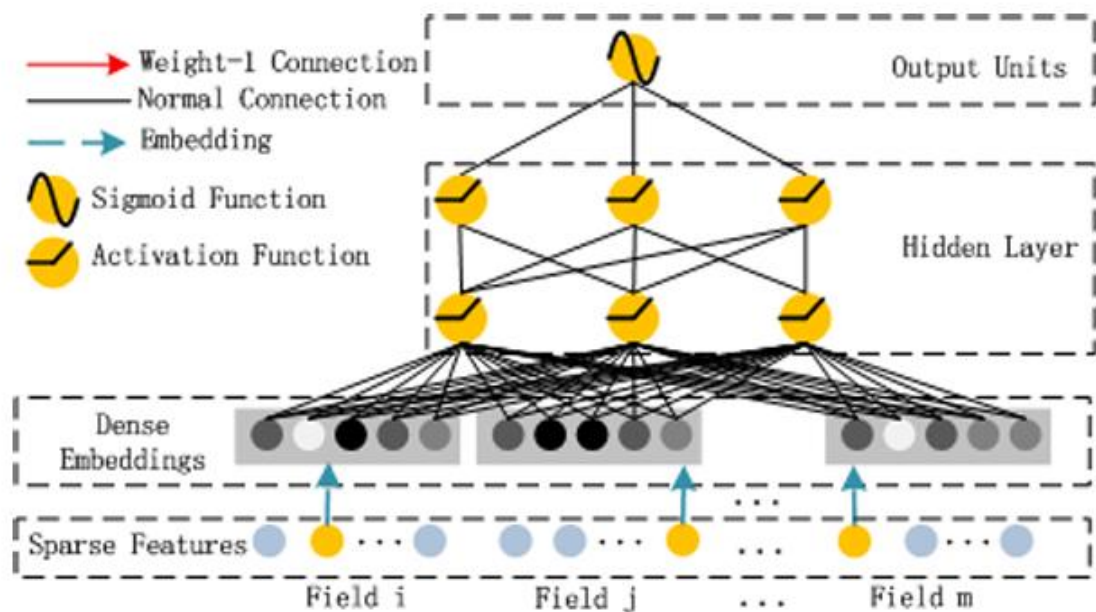
Dễ dàng nhận thấy rằng hai thành phần đầu tiên trong biểu thức tương ứng với mô hình hồi quy tuyến tính, trong khi thành phần thứ ba mở rộng mô hình phân rã ma trận bằng cách mô hình hóa tự động các tương tác phi tuyến giữa mọi cặp đặc trưng.

## 2.3 Mô hình mạng neuron sâu(Deep Neuron Network – DNN)

### 2.3.1 Giới thiệu về mô hình học sâu.

Mạng neuron học sâu (Deep Neuron Network – DNN) là một phương pháp tiên tiến trong trí tuệ nhân tạo, được thiết kế dựa trên mạng nơ-ron nhân tạo truyền thống nhưng mở rộng với nhiều lớp ẩn, cho phép xử lý các mối quan hệ phức tạp và phi tuyến trong dữ liệu.

Nhờ khả năng tự động trích xuất đặc trưng từ dữ liệu đầu vào, DNN đã trở thành công cụ mạnh mẽ trong các lĩnh vực như thị giác máy tính, xử lý ngôn ngữ tự nhiên và hệ thống gợi ý. Đặc biệt, trong hệ thống gợi ý, DNN giúp học biểu diễn phi tuyến giữa người dùng và sản phẩm, đồng thời tích hợp các yếu tố ngữ cảnh như nội dung, địa điểm và hành vi. Điều này không chỉ tăng cường độ chính xác của dự đoán mà còn giảm thiểu sự phụ thuộc vào các kỹ thuật xử lý thủ công và mang lại hiệu quả cao.



Hình 2.2 Hình minh họa Neuron Network

Mô hình DNN (Deep Neuron Network) gồm có 3 thành phần chính:

- Lớp đầu vào (Input Layer): Tiếp nhận dữ liệu đầu vào (vector đặc trưng của người dùng và sản phẩm (sau khi mã hóa hoặc embedding))

- Lớp ẩn (Hidden Layers): Diễn ra các phép biến đổi phi tuyến tính của dữ liệu. Mỗi lớp bao gồm nhiều neuron hay node, mỗi node tính toán giá trị đầu ra thông qua hàm kích hoạt (Sigmoid).
- Lớp đầu ra (Output Layer): Sẽ đưa ra kết quả cuối cùng

### 2.3.2 Cách hoạt động của mô hình.

Quá trình hoạt động của 1 mô hình Deep Neuron Network (DNN) gồm 3 bước chính như sau:

Bước 1: Thực hiện lan truyền tiến (Forward Propagation) được thực hiện nhằm tính toán đầu ra dự đoán của mô hình dựa trên dữ liệu đầu vào. Quá trình này diễn ra tuần tự qua từng lớp của mạng nơ-ron, từ lớp đầu vào đến lớp đầu ra. Tại mỗi node (nút thần kinh) trong một lớp, dữ liệu đầu vào sẽ được xử lý bằng cách thực hiện phép nhân giữa mỗi đầu vào với trọng số (weight) tương ứng. Sau đó, các tích này được cộng dồn lại, kèm theo một giá trị bias (độ lệch), nhằm tăng tính linh hoạt cho mô hình. Kết quả của phép tính này tạo ra một giá trị tổng có trọng số. Công thức như sau:

$$z = \mathbf{w}^T \mathbf{x} + b, \quad a = \sigma(z)$$

Trong đó:

- $\mathbf{x}$  là vector đầu vào,
- $\mathbf{w}$  là vector trọng số,
- $b$  là hệ số bias,
- $\sigma$  là hàm kích hoạt (activation function),
- $a$  là đầu ra của nơ-ron.

Bước 2: Sau khi quá trình lan truyền tiến (forward propagation) được thực hiện và mô hình tạo ra đầu ra dự đoán, bước tiếp theo là tính toán hàm mất mát nhằm đo lường mức độ sai lệch giữa giá trị dự đoán của mô hình và giá trị thực tế trong tập dữ liệu huấn luyện. Hàm mất mát (loss function) đóng vai trò rất quan trọng trong quá trình huấn luyện vì nó cung cấp một chỉ số định lượng giúp đánh giá hiệu quả của mô hình. Giá trị mất mát càng nhỏ thì mô hình càng

chính xác trong việc đưa ra dự đoán. Ngược lại, nếu giá trị này lớn, mô hình cần tiếp tục được điều chỉnh (lan truyền ngược) để cải thiện hiệu suất. Trong đề tài này, Binary Cross-Entropy được lựa chọn làm hàm mất mát

Bước 3 Sau khi tính toán hàm mất mát nhằm đo lường sai lệch giữa dự đoán và giá trị thực tế, bước tiếp theo trong quá trình huấn luyện mạng nơ-ron là thực hiện lan truyền ngược (backpropagation) và tối ưu hóa (optimization). Mục tiêu của bước này là điều chỉnh các tham số (trọng số và bias) của mô hình sao cho giá trị của hàm mất mát được giảm xuống tối thiểu.

## 2.4 Mô hình DeepFM

### 2.4.1 Giới thiệu về mô hình.

DeepFM (Deep Factorization Machine) là một mô hình lai (hybrid model) kết hợp giữa Factorization Machines (FM) và Deep Neural Networks (DNN), được thiết kế để học các tương tác đặc trưng (feature interactions) ở cả bậc thấp và bậc cao.

Mô hình này được sử dụng trong đề tài nhằm giải quyết các bài toán dự đoán đánh giá của người tiêu dùng với các sản phẩm, với khả năng tự động trích xuất đặc trưng phức tạp mà không cần kỹ thuật chọn đặc trưng thủ công.

### 2.4.2 Kiến trúc mô hình.

Giả sử tập dữ liệu huấn luyện bao gồm  $n$  mẫu dữ liệu có dạng  $(x, y)$ , trong đó  $X$  là một bản ghi dữ liệu gồm  $m$  trường thông tin, thường biểu diễn mối quan hệ giữa người dùng và sách, còn  $y \in \mathbb{R}$  là giá trị đánh giá (rating) mà người dùng gán cho cuốn sách tương ứng. Các trường trong bản ghi  $X$  có thể bao gồm các thuộc tính định danh như mã người dùng, mã sách, tên tác giả và các trường số như số lượt đánh giá trung bình của sách, hoặc số lượt đánh giá mà người dùng đã thực hiện.

Các trường định danh (ví dụ: người dùng, sách, tác giả) được mã hóa dưới dạng vector one-hot, trong khi các trường số có thể được giữ nguyên hoặc

được rời rạc hóa và mã hóa tương tự. Sau quá trình tiền xử lý, mỗi mẫu dữ liệu được biểu diễn dưới dạng cặp  $(x,y)$ , trong đó:

$$X = [X_{\text{field}1}, X_{\text{field}2}, \dots, X_{\text{field}m},]$$

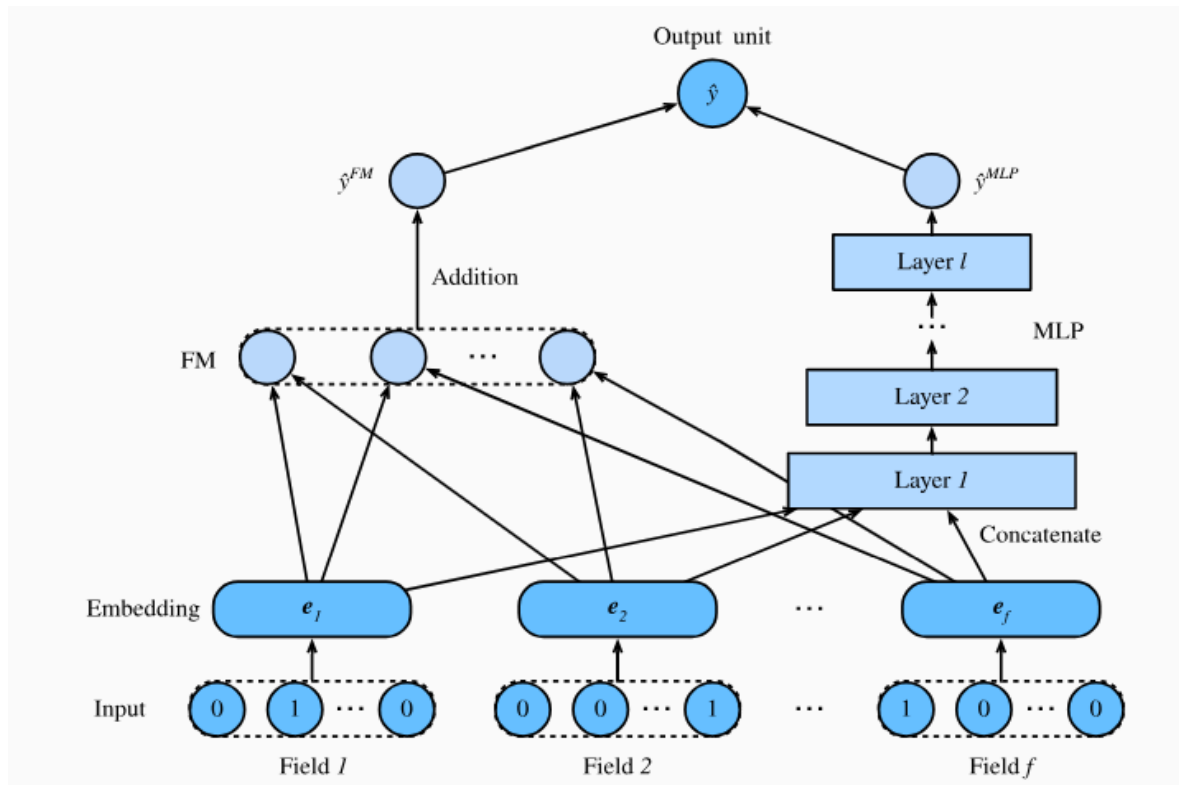
là một vector  $m$  chiều, với mỗi  $x_{\text{field}j}$  là biểu diễn vector của trường thứ  $j$  trong bản ghi dữ liệu.

Thông thường, vector đặc trưng  $x$  có số chiều lớn và rất thưa (sparse), đặc biệt trong trường hợp có số lượng người dùng và sách lớn. Nhiệm vụ của mô hình dự đoán đánh giá là xây dựng một hàm hồi quy:

$$Y = \text{RatingModel}(x)$$

nhằm ước lượng giá trị đánh giá mà người dùng có thể gán cho một cuốn sách nhất định trong một ngữ cảnh cụ thể, từ đó hỗ trợ hệ thống đề xuất cá nhân hóa hiệu quả hơn.

Trong đề tài này, dữ liệu đầu vào bao gồm các trường như người dùng (user), cuốn sách (book), tác giả (author), và có thể bao gồm các đặc trưng liên tục hoặc phân loại bổ sung khác.



Hình 2.3 Hình minh họa cấu trúc của DeepFM

Như minh họa trên trong mô hình DeepFM (Deep Neuron Network) bao gồm hai thành phần chính như sau:

- Thành phần FM: dùng để học tương tác đặc trưng bậc hai (ví dụ: tương tác giữa người dùng và sách, hoặc giữa sách và tác giả).
- Thành phần Deep (mạng nơ-ron sâu): dùng để học tương tác đặc trưng bậc cao hơn, ví dụ như các mối quan hệ phức tạp giữa người dùng, thể loại sách, và phong cách của tác giả.

Cả hai thành phần đều dùng chung đầu vào  $X$ , bao gồm các biểu diễn vector nhúng (embedding vectors) của từng trường. Đối với mỗi đặc trưng  $i$ , mô hình sử dụng:

- Một trọng số vô hướng  $w_i$  để biểu diễn tầm quan trọng bậc một của đặc trưng đó (ví dụ, mức ảnh hưởng trực tiếp của người dùng đến rating),
- Một vector tiềm ẩn  $V_i$  để biểu diễn ảnh hưởng tương tác của đặc trưng  $i$  với các đặc trưng khác (ví dụ: tương tác giữa người dùng và tác giả, hoặc giữa sách và người dùng).

Vector  $V_i$  là vector được ánh xạ từ các vector đặc trưng khi được đưa qua khối embedding được sử dụng đồng thời trong:

- Thành phần FM: để mô hình hóa tương tác đặc trưng bậc hai;
- Thành phần Deep: để học các tương tác đặc trưng bậc cao hơn.

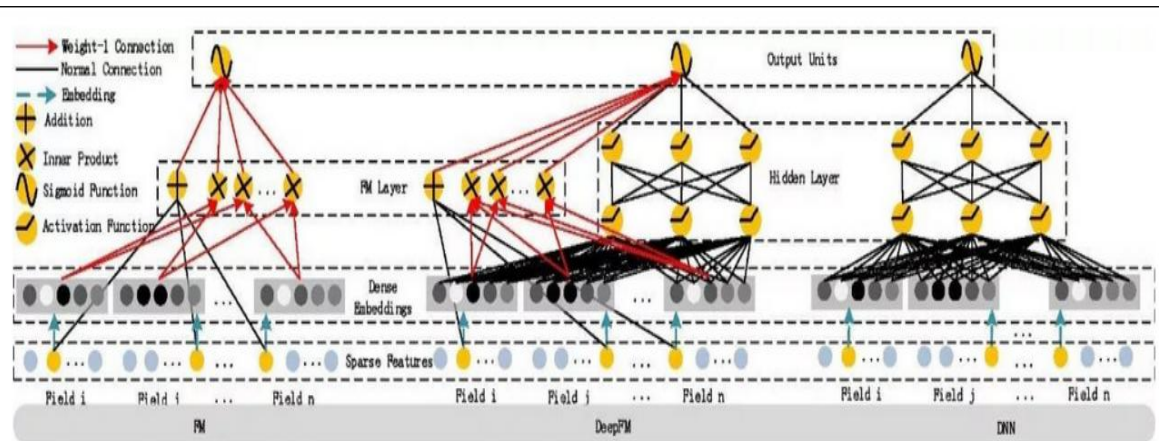
Tất cả các tham số, bao gồm  $w_i$ ,  $V_i$ , và các tham số mạng như trọng số lớp  $W$ , bias  $b$  đều được huấn luyện đồng thời trong quá trình tối ưu mô hình, nhằm dự đoán giá trị rating đầu ra như sau:

$$\hat{y} = \sigma(y_{FM} + y_{DNN})$$

Trong đó:

- $\hat{y} \in [0,1]$  là giá trị rating được chuẩn hóa hoặc xác suất suy ra từ rating,
- $y_{FM}$  là đầu ra của thành phần Factorization Machine,
- $y_{DNN}$  là đầu ra của thành phần mạng nơ-ron sâu (DNN).

### 2.4.3 Luồng dữ liệu trong mô hình DeepFM.



Hình 2.4 Hình minh họa luồng hoạt động của DeepFM

Mô hình DeepFM xử lý dữ liệu đầu vào thông qua một luồng dữ liệu cụ thể nhằm khai thác các tương tác đặc trưng một cách hiệu quả. Luồng dữ liệu này được chia thành các bước chính như sau:

Bước 1: Tiền xử lý dữ liệu đầu vào (Input Features) là bước rất quan trọng trong quy trình xây dựng mô hình học máy hoặc hệ thống đề xuất. Ở giai đoạn này, dữ liệu đầu vào bao gồm nhiều đặc trưng dạng rời rạc (sparse features), cụ thể như `user_id` (mã định danh người dùng), `book_id` (mã định danh sách), `isbn` (mã chuẩn sách quốc tế), `authors` (tác giả), `title` (tên sách), và `location` (vị trí địa lý). Vì các đặc trưng này mang tính chất phân loại và không liên tục, chúng cần được chuyển đổi thành một dạng số học phù hợp để mô hình có thể xử lý hiệu quả. Một trong những phương pháp phổ biến được áp dụng là ánh xạ các đặc trưng rời rạc này sang dạng one-hot encoding — tức là chuyển mỗi giá trị thành một vector nhị phân với vị trí duy nhất được đánh dấu là 1 và các vị trí còn lại là 0. Sau khi chuyển đổi thành dạng one-hot, các vector này tiếp tục được nhúng (embedding) để tạo ra các biểu diễn đặc trưng dạng vector liên tục có chiều thấp hơn, giúp giảm kích thước dữ liệu và tăng khả năng học các mối quan hệ phức tạp giữa các đặc trưng. Quá trình này đảm bảo rằng dữ liệu đầu vào được chuẩn hóa, dễ tiếp nhận bởi các mô hình học sâu và nâng cao hiệu suất tổng thể của hệ thống.



Bước 2: Embedding Layer (Lớp nhúng) đóng vai trò then chốt trong quá trình biến đổi các đặc trưng rời rạc thành các biểu diễn số học có ý nghĩa, giúp mô hình học máy có thể xử lý hiệu quả các thông tin phức tạp từ dữ liệu đầu vào. Ở bước này, tất cả các đặc trưng rời rạc đã được chuyển đổi sơ bộ ở bước trước sẽ được đưa vào lớp embedding. Mỗi giá trị riêng lẻ của các đặc trưng này sẽ được ánh xạ thành một vector nhúng (embedding vector) có kích thước cố định, ví dụ như một vector có chiều dài 8 hoặc 16 tùy thuộc vào thiết kế mô hình. Những vector này không chỉ giúp giảm chiều dữ liệu từ dạng one-hot encoding có kích thước rất lớn, mà còn cho phép mô hình học được các mối quan hệ tiềm ẩn và các mẫu phức tạp giữa các đặc trưng, mà nếu dùng biểu diễn gốc thì khó nhận diện được. Đặc biệt, các vector nhúng này được chia sẻ và sử dụng đồng thời cho cả hai thành phần quan trọng trong kiến trúc DeepFM: phần FM (Factorization Machines) và phần Deep (mạng nơ-ron sâu). Việc dùng chung các vector nhúng này không chỉ giúp tiết kiệm bộ nhớ và tăng tốc độ tính toán mà còn đảm bảo tính nhất quán và đồng bộ trong việc học biểu diễn đặc trưng cho toàn bộ mô hình, từ đó nâng cao độ chính xác và hiệu quả của hệ thống đề xuất hoặc mô hình dự đoán.

Bước 3: Nhánh FM – Học tương tác bậc hai là một thành phần rất quan trọng trong mô hình DeepFM, chuyên xử lý và khai thác các mối quan hệ tương tác giữa các đặc trưng ở mức độ đơn giản hơn so với nhánh Deep, nhưng lại rất hiệu quả trong việc nắm bắt các tương tác bậc hai. Cụ thể, các vector nhúng được tạo ra từ lớp embedding sẽ được sử dụng để tính toán các tương tác đặc trưng bậc hai dựa trên công thức của Factorization Machines (FM). FM là một phương pháp mạnh mẽ nhằm mô hình hóa các tương tác giữa các cặp đặc trưng mà không cần phải khai triển đầy đủ ma trận tương tác lớn, giúp giảm đáng kể độ phức tạp tính toán và tránh hiện tượng quá khớp (overfitting). Quá trình này dựa trên việc nhân các vector nhúng của từng đặc trưng với nhau theo từng cặp, sau đó tổng hợp lại thành một giá trị duy nhất. Kết quả đầu ra của nhánh FM là một giá trị đơn, thể hiện sự kết hợp giữa thành phần tuyến tính và phi tuyến bậc



hai của các đặc trưng đầu vào. Giá trị này cung cấp cho mô hình một cái nhìn rõ ràng về các mối quan hệ tương tác cơ bản giữa các đặc trưng, đồng thời hỗ trợ nhánh Deep trong việc học các đặc trưng trừu tượng hơn, từ đó làm tăng độ chính xác và khả năng tổng quát hóa của mô hình trong các bài toán dự đoán phức tạp.

Bước 4: Nhánh Deep – Học tương tác bậc cao đóng vai trò quan trọng trong việc khai thác các mối quan hệ phức tạp và không tuyến tính giữa các đặc trưng đầu vào mà các phương pháp truyền thống khó có thể phát hiện. Ở giai đoạn này, tất cả các vector nhúng được tạo ra từ lớp embedding sẽ được nối lại với nhau thành một vector duy nhất, tổng hợp toàn bộ thông tin từ các đặc trưng rời rạc dưới dạng các biểu diễn số học liên tục. Vector tổng hợp này sau đó được đưa vào một chuỗi các lớp ẩn của mạng neuron truyền thẳng đa tầng (Deep Neural Network – DNN), nơi mà mỗi lớp ẩn thực hiện quá trình lan truyền tiến (forward propagation). Trong quá trình này, dữ liệu sẽ lần lượt được xử lý qua các neuron với các trọng số học được, và đặc biệt là thông qua các hàm kích hoạt phi tuyến, trong trường hợp này là hàm sigmoid. Hàm sigmoid giúp mô hình có khả năng học các tương tác phi tuyến phức tạp giữa các đặc trưng, đồng thời giới hạn giá trị đầu ra trong khoảng từ 0 đến 1, giúp ổn định quá trình huấn luyện và tối ưu hóa mô hình. Nhờ vào cấu trúc đa lớp và các hàm kích hoạt phi tuyến, nhánh Deep này có thể phát hiện và học các đặc trưng trừu tượng, từ đó cải thiện đáng kể hiệu suất dự đoán của mô hình, đặc biệt là trong các bài toán có tính tương tác bậc cao và dữ liệu đa dạng.

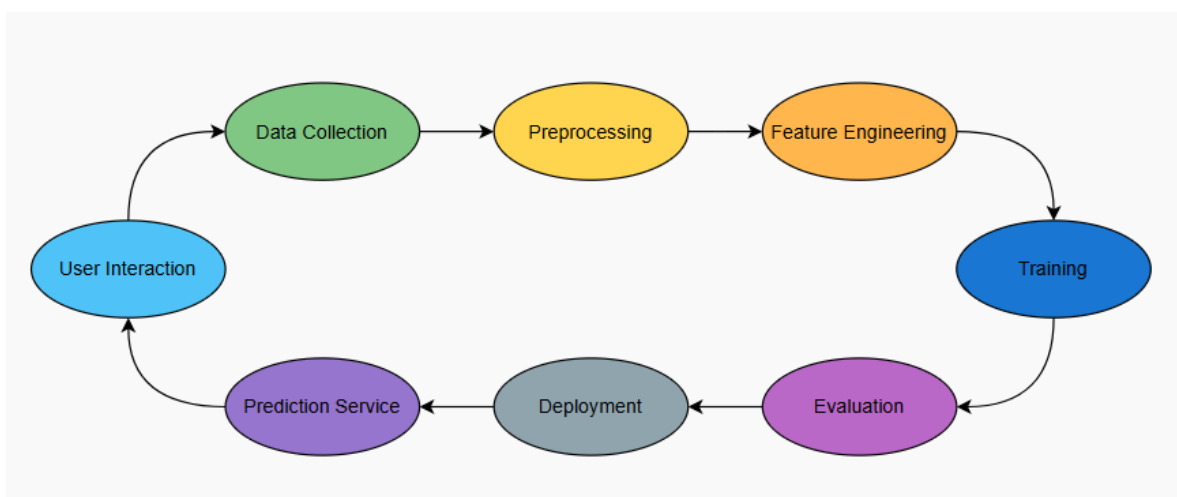
Bước 5: Kết hợp kết quả hai nhánh (Fusion) là bước quan trọng để tổng hợp thông tin từ cả hai thành phần chính trong mô hình DeepFM, nhằm tận dụng tối đa sức mạnh của từng nhánh và nâng cao hiệu quả dự đoán. Ở giai đoạn này, đầu ra của nhánh FM, vốn chịu trách nhiệm học các tương tác bậc hai đơn giản và hiệu quả giữa các đặc trưng, sẽ được kết hợp cùng với đầu ra của nhánh Deep, chuyên sâu vào việc học các tương tác phi tuyến và phức tạp hơn ở mức độ cao. Có hai cách phổ biến để kết hợp hai đầu ra này: một là cộng

trực tiếp các giá trị đầu ra lại với nhau, hai là ghép nối (concatenate) hai đầu ra thành một vector lớn hơn và sau đó đưa qua một lớp fully connected (lớp kết nối đầy đủ) cuối cùng để mô hình có thể học cách cân chỉnh trọng số giữa các đầu ra từ hai nhánh sao cho phù hợp nhất với bài toán. Kết quả đầu ra tổng hợp này sau đó sẽ được đưa vào lớp đầu ra (output layer) của mô hình, nơi áp dụng hàm kích hoạt sigmoid. Hàm sigmoid đảm bảo rằng giá trị đầu ra nằm trong khoảng từ 0 đến 1, phù hợp cho các bài toán phân loại nhị phân hoặc dự đoán xác suất. Nhờ việc kết hợp hiệu quả này, mô hình DeepFM có thể vừa khai thác được các tương tác đặc trưng đơn giản mà cũng không bỏ sót các mối quan hệ phức tạp, từ đó mang lại độ chính xác cao và khả năng tổng quát hóa tốt hơn trong nhiều ứng dụng thực tế.

## 2.5 Ứng dụng của MLOPs trong hệ thống đề xuất sách.

MLOPs (Machine Learning Operations) là một tập hợp các nguyên tắc và thực hành nhằm tự động hóa và chuẩn hóa quy trình triển khai và vận hành các mô hình học máy trong môi trường sản xuất. MLOps kết hợp giữa Khoa học dữ liệu (Data Science) và Kỹ thuật phần mềm (DevOps) để đảm bảo rằng mô hình ML có thể hoạt động ổn định, dễ quản lý, và có thể mở rộng.

### 2.5.1 Vòng đời của một hệ thống MLOPs.



Hình 2.5 Quy trình MLOps trong hệ thống đề xuất sách

Vòng đời của MLOps là một quy trình liên tục và tuần hoàn, bắt đầu từ việc người dùng tương tác với hệ thống như tìm kiếm, đánh giá hoặc xem các

đề xuất. Những hành vi này được ghi nhận và trở thành nguồn dữ liệu mới cho hệ thống. Sau đó, dữ liệu được thu thập từ nhiều nguồn khác nhau, bao gồm lịch sử hành vi, đánh giá của người dùng và thông tin chi tiết về sản phẩm (ví dụ: sách). Tiếp theo, dữ liệu sẽ trải qua giai đoạn tiền xử lý, nơi các bước như làm sạch, xử lý giá trị thiếu, chuẩn hóa và mã hóa được thực hiện để đảm bảo dữ liệu ở trạng thái tối ưu cho mô hình học máy. Quá trình tạo đặc trưng (feature engineering) tiếp tục trích xuất hoặc xây dựng các đặc trưng hữu ích nhằm tăng cường hiệu suất dự đoán của mô hình.

Sau khi dữ liệu đã được chuẩn bị, mô hình học máy sẽ được huấn luyện để học hỏi từ các mẫu dữ liệu và xây dựng năng lực dự đoán. Kết quả của mô hình sau đó được đánh giá qua các chỉ số như độ chính xác hoặc độ lỗi để xác định mức độ hiệu quả và tính sẵn sàng triển khai. Khi đạt yêu cầu chất lượng, mô hình được triển khai vào môi trường sản xuất và tích hợp vào các hệ thống thực tế như website hoặc API. Tại đây, mô hình bắt đầu cung cấp dịch vụ dự đoán, chẳng hạn như gợi ý sách phù hợp cho từng người dùng.

Kết quả dự đoán quay lại với người dùng, từ đó tạo ra dữ liệu mới thông qua các tương tác tiếp theo. Vòng đời này tiếp tục lặp lại, không ngừng cải tiến chất lượng hệ thống thông qua việc học hỏi từ dữ liệu mới.

#### 2.5.1.1 Thu thập và xử lý dữ liệu (*Data Collection & Processing*)

Để xây dựng một mô hình học máy hiệu quả, việc thu thập và xử lý dữ liệu là bước không thể thiếu và đòi hỏi sự chú ý đặc biệt. Trước tiên, dữ liệu về sách, đánh giá của người dùng được thu thập từ Kaggle với định dạng csv. Sau khi thu thập, dữ liệu cần được làm sạch để đảm bảo chất lượng, bao gồm xử lý các giá trị thiếu, loại bỏ giá trị dữ liệu nhiễu và định dạng lại giá trị của các trường dữ liệu. Sau đó, thực hiện lưu dữ liệu vừa được làm sạch vào 1 file csv. Tiếp theo, quá trình trích xuất đặc trưng (feature engineering) là bước quan trọng giúp chuyển đổi dữ liệu từ dạng rời rạc sang vector thông qua phương pháp label encoding. Cuối cùng, dữ liệu cần được phân chia thành các tập huấn luyện, kiểm tra và xác thực (train/test) theo tỉ lệ 80:20 để đảm bảo mô hình

được đánh giá một cách toàn diện và khách quan. Tất cả các bước này đều cần được thực hiện một cách cẩn thận và có hệ thống để đảm bảo rằng mô hình có thể học hỏi và dự đoán chính xác từ dữ liệu đã được chuẩn bị kỹ lưỡng.

#### 2.5.1.2 Huấn luyện mô hình (*Model Training*)

Quá trình này bao gồm một số bước quan trọng. Đầu tiên, thiết kế model phục vụ DeepFM (Deep Factorization Machine) và thiết kế kiến trúc mô hình bao gồm xác định số lượng lớp, loại hàm kích hoạt. Tiếp theo, quá trình huấn luyện mô hình được thực hiện bằng cách đưa dữ liệu vào, tính toán hàm mất mát và cập nhật trọng số thông qua các thuật toán tối ưu Adam. Tất cả các bước này cần được thực hiện một cách cẩn thận và có hệ thống để đảm bảo mô hình đạt được hiệu quả tốt nhất.

#### 2.5.1.3 Đánh giá và kiểm thử mô hình (*Evaluation & Testing*)

Đối với các hệ thống đề xuất, các chỉ số như Recall, Loss và Accuracy được áp dụng để đo lường mức độ phù hợp và chất lượng gợi ý. Việc lựa chọn đúng tiêu chí đánh giá không chỉ giúp tối ưu hóa hiệu suất mà còn đảm bảo mô hình đáp ứng tốt yêu cầu thực tế.

#### 2.5.1.4 Triển khai mô hình (*Model Deployment*)

Triển khai dưới dạng API, nơi mô hình FastAPI, giúp dễ dàng tích hợp với các hệ thống khác. Ngoài ra, đối với các tác vụ xử lý dữ liệu lớn định kỳ, dạng batch là lựa chọn tối ưu, ví dụ như việc chạy các tác vụ hàng đêm để tạo gợi ý sản phẩm hoặc phân tích dữ liệu. Để hỗ trợ quá trình triển khai, việc đóng gói mô hình bằng Docker giúp đảm bảo tính nhất quán khi chạy trên các môi trường khác nhau. Đồng thời, quản lý phiên bản mô hình thông qua model registry giúp theo dõi và kiểm soát các thay đổi.

#### 2.5.1.5 Tái huấn luyện (*Retraining*)

Tái huấn luyện và cập nhật mô hình là một quy trình quan trọng nhằm đảm bảo mô hình luôn phù hợp với dữ liệu mới và đáp ứng tốt các yêu cầu thực tế. Quy trình này có thể được thực hiện theo lịch định kỳ (ví dụ: hàng tuần, dựa

trên cấu hình trong `Mlflow_tracking.yaml`). Kết quả của mô hình mới sẽ được đánh giá kỹ lưỡng và nếu hiệu suất được cải thiện so với phiên bản hiện tại, mô hình mới sẽ được triển khai vào môi trường sản xuất. Quy trình này không chỉ giúp duy trì hiệu quả của hệ thống mà còn đảm bảo khả năng thích nghi với những thay đổi trong môi trường dữ liệu.

### 2.5.2 Lợi ích của MLOPs.

MLOps mang lại nhiều lợi ích quan trọng, không chỉ giúp quá trình phát triển mô hình học máy trở nên hiệu quả hơn, mà còn đảm bảo rằng mô hình có thể được vận hành ổn định trong môi trường thực tế.

#### 2.5.2.1 Tự động hóa quy trình phát triển và triển khai mô hình

Tự động hóa quy trình phát triển và triển khai mô hình giúp giảm thiểu thao tác thủ công trong các giai đoạn như xử lý dữ liệu, huấn luyện, đánh giá và triển khai. Điều này không chỉ tối ưu hóa thời gian mà còn giảm thiểu rủi ro sai sót do con người gây ra. Nhờ tích hợp các quy trình CI/CD tự động, tốc độ đưa mô hình vào môi trường sản xuất được cải thiện đáng kể, đảm bảo tính liên tục và hiệu quả trong vận hành. Bên cạnh đó, việc thiết lập lịch tái huấn luyện định kỳ giúp mô hình luôn được cập nhật với dữ liệu mới, duy trì hiệu suất và độ chính xác cao trong các ứng dụng thực tế.

#### 2.5.2.2 Đảm bảo khả năng tái sử dụng và truy vết

Đảm bảo mọi bước trong quá trình phát triển và triển khai mô hình đều được ghi nhận và có thể lặp lại. Từ việc thu thập và xử lý dữ liệu, mã nguồn sử dụng, các tham số huấn luyện cho đến kết quả cuối cùng của mô hình, tất cả đều được lưu trữ một cách có hệ thống.

#### 2.5.2.3 Quản lý hiệu quả mô hình và dữ liệu

Quản lý hiệu quả mô hình và dữ liệu đóng vai trò quan trọng trong việc đảm bảo tính nhất quán và tối ưu hóa quy trình làm việc. Việc theo dõi phiên bản của mô hình, dữ liệu và các đặc trưng không chỉ giúp ghi nhận lịch sử thay đổi mà còn cung cấp khả năng quay lại trạng thái trước đó khi cần thiết, từ đó

giảm thiểu rủi ro và sai sót. Đồng thời, việc lưu trữ và tái sử dụng các mô hình đã được chứng minh hiệu quả cùng với các tập đặc trưng tối ưu giúp tiết kiệm đáng kể chi phí và thời gian phát triển.

#### *2.5.2.4 Giám sát và cảnh báo hiệu suất mô hình*

Hệ thống giám sát tự động giúp đảm bảo hiệu suất hoạt động của các mô hình trong thực tế. Bằng cách liên tục theo dõi và phân tích dữ liệu, hệ thống có khả năng phát hiện sớm các dấu hiệu bất thường hoặc suy giảm hiệu quả. Điều này không chỉ giúp tránh được các rủi ro tiềm ẩn mà còn tạo điều kiện để điều chỉnh và tối ưu hóa mô hình kịp thời.

#### *2.5.2.5 Khả năng mở rộng*

Mở rộng hiệu quả các mô hình học máy để xử lý các tập dữ liệu lớn hơn và đáp ứng khối lượng công việc tăng cao. Bằng cách tối ưu hoá quy trình và sử dụng tài nguyên một cách hợp lý, các mô hình học máy có thể được triển khai trên quy mô lớn mà vẫn đảm bảo hiệu suất và độ chính xác.

## CHƯƠNG 3 XÂY DỰNG ỨNG DỤNG.

### 3.1 Thiết kế và triển khai module

#### 3.1.1 Xử lý dữ liệu.

##### 3.1.1.1 Nguồn dữ liệu

Dữ liệu được sử dụng trong bài viết bao gồm ba bảng chính: bảng Books, Ratings và Users. Bảng Books chứa thông tin chi tiết về các đầu sách, bao gồm tiêu đề, mã sách và tên tác giả, với tổng cộng 271.360 đầu sách. Bảng Ratings ghi nhận các đánh giá từ người dùng, bao gồm tiêu đề sách, mã người dùng và điểm đánh giá, với tổng số 1.148.780 bản ghi. Cuối cùng, bảng Users lưu trữ thông tin về người dùng như ID người dùng và địa chỉ, với tổng cộng 278.858 bản ghi. Các dữ liệu này cung cấp một cơ sở thông tin phong phú để phân tích và khai thác thông tin liên quan đến sở thích đọc sách và hành vi đánh giá của người dùng.

##### 3.1.1.2 Làm sạch dữ liệu.

Để chuẩn bị dữ liệu cho quá trình phân tích, bước đầu tiên là đọc và nhập dữ liệu từ các tệp Books.csv, Users.csv và Ratings.csv. Các tệp này chứa thông tin quan trọng như ISBN, tiêu đề sách (Book-Title), tác giả sách (Book-Author), mã người dùng (User-ID), vị trí (Location), độ tuổi (Age), năm (Year) và đánh giá sách (Book-Rating). Việc tổ chức và kiểm tra dữ liệu từ các tệp này là nền tảng để đảm bảo tính chính xác và đầy đủ của thông tin trước khi tiến hành các bước phân tích tiếp theo. Quá trình này yêu cầu sự cẩn thận trong việc xác định định dạng, xử lý các giá trị thiếu và loại bỏ những dữ liệu không hợp lệ nhằm tối ưu hóa chất lượng đầu vào cho các mô hình phân tích.

```
import numpy as np
import pandas as pd

# Read data
df_book = pd.read_csv('Books.csv', usecols=['ISBN', 'Book-Title', 'Book-Author'])
df_rating = pd.read_csv('Ratings.csv', usecols=['User-ID', 'ISBN', 'Book-Rating'])
df_user = pd.read_csv('Users.csv', usecols=['User-ID', 'Location'])
```

Hình 3.1 Code đọc dữ liệu

Để thuận tiện hơn trong quá trình xử lý dữ liệu, cần chuẩn hóa tên cột bằng cách đổi "Book-Title" thành "Title", "Book-Author" thành "Author", "Book-Rating" thành "Rating" và "User-ID" thành "User". Tiếp theo, tiến hành lọc bỏ các dữ liệu không hợp lệ nhằm đảm bảo chất lượng phân tích. Cụ thể, loại bỏ các đánh giá có giá trị "Rating" bằng 0, vì đây thường được hiểu là khách hàng không thực hiện đánh giá. Đồng thời, cần loại bỏ các bản ghi của người dùng có trường "Location" chứa giá trị 'n/a, n/a, n/a', do đây là thông tin không đầy đủ hoặc không chính xác. Các bước này giúp cải thiện tính chính xác và đáng tin cậy của dữ liệu trước khi tiến hành các phân tích tiếp theo.

```
df_user = df_user[df_user['Location'] != 'n/a, n/a, n/a']
df_rating = df_rating[df_rating['Book-Rating'] > 0]
df_rating['Book-Rating'] = df_rating['Book-Rating'].apply(lambda x: 1 if x > 5 else 0)
```

Hình 3.2 Loại bỏ các rating nhỏ hơn 0

Để đảm bảo chất lượng dữ liệu, cần thực hiện xử lý và chuẩn hóa thông tin Location của người dùng một cách cẩn thận. Việc này giúp dữ liệu đồng nhất và dễ dàng phân tích hơn. Đối với các trường hợp dữ liệu bị thiếu, cần loại bỏ các dòng chứa giá trị NA trong tất cả các bảng dữ liệu liên quan. Quy trình này không chỉ giúp cải thiện tính chính xác của phân tích mà còn đảm bảo rằng các kết quả đầu ra không bị ảnh hưởng bởi dữ liệu không đầy đủ hoặc không hợp lệ.

```
def process_location(location):
    # Ensure the input is a string before processing
    location_str = str(location)
    if location_str == 'n/a, n/a, n/a':
        return None
    parts = [part.strip() for part in location_str.split(',')]
    valid_parts = [part for part in parts if part != 'n/a']
    if not valid_parts:
        return None
    elif len(valid_parts) == 1:
        return valid_parts[0]
    else:
        return valid_parts[-1]

# Convert the 'Location' column to string type before applying the function
df_user['Location'] = df_user['Location'].astype(str)

df_user['Location'] = df_user['Location'].apply(process_location)
```

Hình 3.3 Đưa dữ liệu bảng địa chỉ về đúng định dạng



Tiếp theo, để đảm bảo tính chính xác và chất lượng của dữ liệu, cần thực hiện bước xử lý các trường hợp trùng lặp trong tập dữ liệu đánh giá. Cụ thể, chúng ta sẽ loại bỏ những đánh giá lặp lại giữa người dùng (User) và tiêu đề sách (Title), nhằm đảm bảo rằng mỗi người dùng chỉ được phép đánh giá một cuốn sách duy nhất một lần. Quá trình này không chỉ giúp giảm thiểu sự dư thừa trong dữ liệu mà còn tăng độ tin cậy của các phân tích tiếp theo, đồng thời tối ưu hóa hiệu quả của các mô hình dự đoán hoặc gợi ý. Việc kiểm tra và loại bỏ dữ liệu trùng lặp cần được thực hiện một cách cẩn thận và có hệ thống để tránh làm mất đi những thông tin quan trọng hoặc gây sai lệch trong dữ liệu.

#### 3.1.1.3 *Embedding các trường dữ liệu cho huấn luyện*

```
def transform(self, data: pd.DataFrame, features: list[str]) -> pd.DataFrame:
    data = data.copy()
    for feat in features:
        if feat in self.encoders:
            data[feat] = self.encoders[feat].transform(data[feat].astype(str))
    return data
```

Hình 3.4 Sử dụng labelEncoding cho dữ liệu rời rạc

Cuối cùng, việc mã hóa các cột dữ liệu dạng categorical như User, ISBN, Author và Location là một bước quan trọng. Sử dụng LabelEncoder giúp chuyển đổi các giá trị dạng chuỗi trong các cột này thành các giá trị số, tạo điều kiện thuận lợi cho việc xử lý và phân tích dữ liệu. Quá trình này không chỉ giúp làm sạch dữ liệu mà còn đảm bảo rằng dữ liệu được tối ưu hóa, giảm thiểu rủi ro sai lệch trong các thuật toán phân tích. Đây là một phần không thể thiếu trong quy trình tiền xử lý dữ liệu, góp phần nâng cao hiệu quả và độ chính xác của các mô hình phân tích sau này.

#### 3.1.1.4 *Chuẩn bị dữ liệu cho mô hình*

Để xây dựng mô hình DeepFM, trước tiên cần tạo nhãn (label) dựa trên ngưỡng rating, với giá trị mặc định là 5. Các giá trị rating lớn hơn hoặc bằng 5 sẽ được gán nhãn là 1 (positive), trong khi các giá trị nhỏ hơn 5 sẽ được gán nhãn là 0 (negative).

Sau khi xử lý nhãn, dữ liệu cần được chia thành hai tập riêng biệt: tập train và tập test, với tỷ lệ phân chia thường sử dụng là 80:20 nhằm đảm bảo đủ dữ liệu để huấn luyện và kiểm tra mô hình. Tiếp theo, cần chuẩn bị các feature columns để làm đầu vào cho mô hình DeepFM. Quá trình này bao gồm việc xác định các đặc trưng dạng sparse (thưa) và dense (dày), sau đó mã hóa các đặc trưng này bằng các phương pháp phù hợp như label encoding hoặc embedding. Điều này giúp mô hình có khả năng xử lý cả thông tin tuyến tính và phi tuyến tính từ dữ liệu, đảm bảo hiệu quả trong việc dự đoán và phân loại.

```
sparse_features = ['User', 'ISBN', 'Title', 'Author', 'Location']
data = encode_data(data, sparse_features)

data['label'] = (data['Rating'] >= rating_threshold).astype(int)

sparse_feature_columns = create_feature_columns(data, sparse_features, embed_dim)
dense_feature_columns = [] # nếu có, bạn có thể thêm

feature_columns = [dense_feature_columns, sparse_feature_columns]

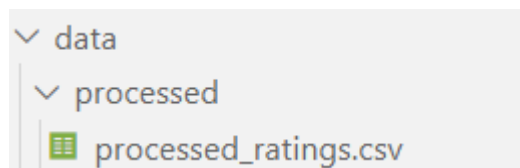
X_train, X_test, y_train, y_test = split_dataset(data, sparse_features, label_column='label',
                                                test_size=test_size)

return feature_columns, (X_train, y_train), (X_test, y_test)
```

Hình 3.5 Chuẩn bị dữ liệu train và test

#### 3.1.1.5 Lưu dữ liệu đã được xử lý.

Sau khi hoàn tất quá trình làm sạch và xử lý dữ liệu thô, toàn bộ dữ liệu đã được chuẩn hóa sẽ được lưu trữ trong thư mục "processed" dưới định dạng CSV. Việc sử dụng định dạng này không chỉ đảm bảo tính tổ chức và dễ dàng truy xuất mà còn hỗ trợ tối ưu hóa quy trình làm việc, đặc biệt trong các dự án yêu cầu sự đồng nhất và khả năng tích hợp cao. Thư mục "processed" đóng vai trò như một nguồn dữ liệu trung tâm, giúp giảm thiểu rủi ro sai sót và tăng cường hiệu quả trong các bước phân tích và báo cáo tiếp theo.



Hình 3.6 Thư mục chứa dữ liệu được làm sạch

### 3.1.2 Huấn luyện mô hình DeepFM

#### 3.1.2.1 Thiết kế cấu trúc model của DeepFM.

Lớp Embedding có nhiệm vụ chuyển đổi các đặc trưng phân loại (categorical features) thành các vector liên tục (dense vectors) có thể học được. Điều này giúp giảm chiều dữ liệu và học được mối quan hệ ngữ nghĩa giữa các giá trị rời rạc. Trong mô hình đề xuất, các đặc trưng được embedding bao gồm: User ID, ISBN (mã sách), Title (tên sách), Author (tác giả) và Location (vị trí người dùng). Mỗi đặc trưng sau khi được ánh xạ sẽ có một vector embedding tương ứng, được cập nhật trong quá trình huấn luyện mô hình.

```
model = DeepFM(
    feature_columns=(dense_feature_columns, sparse_feature_columns),
    k=6,
    w_reg=1e-3,
    v_reg=1e-3,
    hidden_units=[32, 16],
    output_dim=1,
    activation='relu'
)

# Biên dịch mô hình
model.compile(
    loss='binary_crossentropy',
    optimizer='adam',
    metrics=['accuracy']
)
```

Hình 3.7 Cấu hình mô hình DeepFM

Lớp FM chịu trách nhiệm học các tương tác bậc hai giữa các đặc trưng trong dữ liệu. Lớp này bao gồm hai phần: phần tuyến tính (linear part) giúp mô hình học trọng số cho từng đặc trưng riêng lẻ; và phần tương tác (interaction part) giúp mô hình học được mối quan hệ giữa từng cặp đặc trưng với nhau thông qua việc nhân tích vô hướng giữa các vector embedding. Để tránh hiện tượng quá khớp (overfitting), mô hình sử dụng hệ số regularization là  $w\_reg = 1e-3$  cho phần tuyến tính và  $v\_reg = 1e-3$  cho phần tương tác.

Lớp Deep là một mạng nơ-ron sâu (deep neural network) dùng để học các tương tác phi tuyến và phức tạp hơn giữa các đặc trưng. Cấu trúc của mạng gồm hai lớp ẩn với số lượng neuron lần lượt là 32 và 16, sử dụng hàm kích hoạt ReLU nhằm tăng khả năng biểu diễn phi tuyến của mô hình. Cuối cùng, lớp đầu ra (output layer) gồm một neuron duy nhất và sử dụng hàm kích hoạt sigmoid để đưa ra xác suất dự đoán đầu ra, đặc biệt phù hợp với các bài toán phân loại nhị phân (binary classification) như dự đoán đánh giá người dùng.

```
# Huấn Luyện
model.fit(
    X_train,
    y_train,
    batch_size=32,
    epochs=5,
    validation_split=0.2
)
```

Hình 3.8 Cấu hình quá trình huấn luyện của mô hình DeepFM

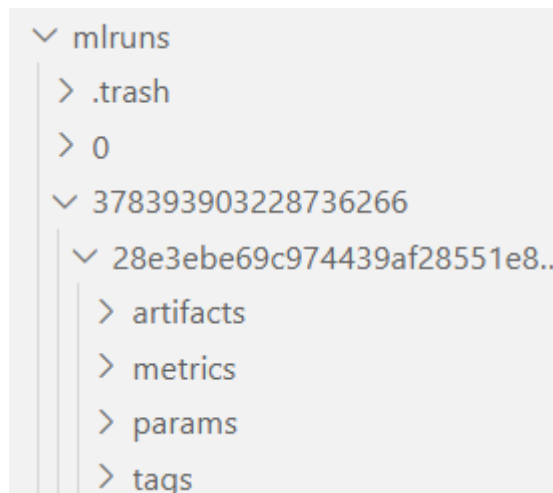
Trong quá trình huấn luyện mô hình, mỗi epoch được thiết kế để mô hình đi qua toàn bộ tập dữ liệu huấn luyện một lần, và tổng cộng có 5 epoch được thực hiện. Mỗi epoch được chia thành nhiều batch, mỗi batch chứa 32 mẫu dữ liệu. Trong từng batch, quy trình huấn luyện bao gồm bước forward pass để tính toán dự đoán từ mô hình, sau đó là bước tính toán loss và các metrics để đánh giá hiệu suất. Tiếp theo, backward pass được thực hiện nhằm cập nhật trọng số của mô hình dựa trên gradient, với sự hỗ trợ của thuật toán tối ưu Adam, giúp tự động điều chỉnh learning rate để tăng hiệu quả huấn luyện.

Sau khi hoàn thành mỗi epoch, mô hình sẽ được đánh giá trên tập validation để kiểm tra hiệu suất trên dữ liệu chưa được huấn luyện. Các chỉ số quan trọng như độ chính xác Accuracy được theo dõi nhằm đánh giá khả năng phân loại và hiệu quả tổng thể của mô hình trong suốt quá trình huấn luyện.

### 3.1.2.2 Quản lý model và theo dõi phiên bản.

Mỗi lần huấn luyện mô hình được ghi lại dưới dạng một experiment riêng biệt trong MLflow. Thông tin được ghi nhận bao gồm:

- Hyperparam:  $k=10$ ,  $w\_reg = 1e-3$ ,  $v\_reg = 1e-3$ ,  $hidden\_units = [32, 16]$
- Metrics đánh giá: accuracy, AUC và loss trên tập train/validation
- Thông số huấn luyện: số epoch = 5, batch\_size = 32
- Thời gian huấn luyện và tên mô hình được lưu theo timestamp



Hình 3.9 Các phiên bản model được lưu bằng MLflow

Quản lý phiên bản mô hình đóng vai trò quan trọng trong việc theo dõi và cải thiện hiệu suất của các mô hình học sâu, đặc biệt là DeepFM. Sau mỗi lần huấn luyện, mô hình cần được lưu trữ dưới dạng các phiên bản riêng biệt trong thư mục mlruns để đảm bảo khả năng truy xuất và so sánh giữa các phiên bản. Điều này không chỉ giúp đánh giá hiệu quả của các thay đổi mà còn hỗ trợ việc kiểm soát chất lượng và tái sử dụng mô hình trong tương lai. Việc duy trì một quy trình quản lý phiên bản khoa học và có hệ thống sẽ góp phần tối ưu hóa quá trình phát triển và đảm bảo tính nhất quán trong sản phẩm cuối cùng.

### 3.1.2.3 Triển khai mô hình.

Sau khi hoàn tất quá trình huấn luyện mô hình gợi ý sách DeepFM và lưu trữ phiên bản mới nhất dưới định dạng .pkl, chúng tôi đã triển khai hệ thống thông qua một API được xây dựng bằng FastAPI. API này cho phép tích hợp dễ dàng vào các nền tảng khác, đảm bảo khả năng xử lý yêu cầu nhanh chóng

và hiệu quả. Sự kết hợp giữa mô hình DeepFM và kiến trúc API hiện đại giúp tối ưu hóa trải nghiệm người dùng, đồng thời cung cấp các gợi ý sách chính xác và phù hợp hơn dựa trên sở thích và hành vi của họ.

```
# Load model
MODEL_DIR = "models"
latest_model = max([os.path.join(MODEL_DIR, f) for f in os.listdir(MODEL_DIR) if f.endswith('.pkl')],
                    key=os.path.getctime)
model = joblib.load(latest_model)
```

Hình 3.10 Lấy model trong MLflow

Ứng dụng FastAPI được triển khai nhằm hỗ trợ việc dự đoán từ mô hình huấn luyện, với khả năng nhận đầu vào là các đặc trưng của sách đã qua xử lý và trả về kết quả dự đoán một cách nhanh chóng và chính xác. Khi API khởi động, hệ thống sẽ tự động kiểm tra và tải mô hình mới nhất từ thư mục "models", đảm bảo sử dụng phiên bản tối ưu nhất. Bên cạnh đó, việc sử dụng Pydantic để định nghĩa dữ liệu đầu vào giúp kiểm soát chặt chẽ kiểu dữ liệu và định dạng JSON, giảm thiểu rủi ro sai sót. Endpoint chính /predict được thiết kế để nhận các thông tin đầu vào như userId, n\_books và top\_k, đảm bảo tính linh hoạt và đáp ứng tốt các yêu cầu dự đoán từ phía người dùng.

```
{
  "userId": 276726,
  "n_books": 50,
  "top_k": 5
}
```

Hình 3.11 Dữ liệu nhận từ Client

Sau khi nhận được userId, hệ thống sẽ tiến hành ánh xạ vào bộ dữ liệu để truy xuất thông tin chi tiết của người dùng như tuổi (age) và vị trí địa lý (location) tương ứng. Tiếp theo, thông tin người dùng bao gồm userId, location và age sẽ được kết hợp với số lượng sách (n\_books) trong cơ sở dữ liệu liên quan. Đồng thời, toàn bộ các đặc trưng sẽ được mã hóa thành các giá trị rời rạc dựa trên file mã hóa (encode file) đã được lưu trữ trước đó. Dữ liệu sau khi xử lý sẽ được đưa vào mô hình để phân tích và dự đoán. Kết quả dự đoán trả về dưới dạng JSON, giúp việc tích hợp vào các hệ thống giao diện người dùng

(frontend) hoặc các hệ thống đề xuất phía client trở nên nhanh chóng và hiệu quả, đảm bảo tính linh hoạt và khả năng mở rộng cho ứng dụng.

```
[
  {
    "title": "TUMBLING",
    "author": "Diane McKinney-whetstone",
    "publisher": "Touchstone",
    "year": 1997,
    "prediction": 0.980593204498291,
    "image_url": "http://images.amazon.com/images/P/0684837242.01.LZZZZZZZ.jpg"
  },
  {
    "title": "The Magical Worlds of Harry Potter: A Treasury of Myths, Legends, and Fascinating Facts",
    "author": "David Colbert",
    "publisher": "Berkley Publishing Group",
    "year": 2002,
    "prediction": 0.9570724368095398,
    "image_url": "http://images.amazon.com/images/P/0425187012.01.LZZZZZZZ.jpg"
  },
  {
    "title": "Nachrichten aus Mittelerde.",
    "author": "John Ronald Reuel Tolkien",
    "publisher": "Klett-Cotta",
    "year": 2001,
    "prediction": 0.9205407500267029,
    "image_url": "http://images.amazon.com/images/P/3608932461.01.LZZZZZZZ.jpg"
  }
]
```

Hình 3.12 Dữ liệu gửi tới Client

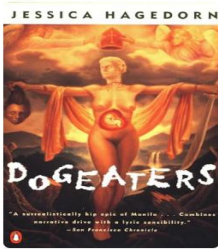
Khi người dùng truy cập vào hệ thống thông qua phía client, một yêu cầu sẽ được gửi đến endpoint /predict cùng với dữ liệu chứa ID của người dùng. Hệ thống sẽ tự động xử lý và phân tích dữ liệu này để dự đoán mức độ yêu thích của người dùng đối với toàn bộ danh sách sách có trong cơ sở dữ liệu. Từ kết quả phân tích, hệ thống sẽ lựa chọn và đề xuất danh sách 10 cuốn sách phù hợp nhất với sở thích cá nhân của người dùng. Quy trình này không chỉ giúp cá nhân hóa trải nghiệm mà còn đảm bảo đáp ứng tối ưu nhu cầu và sở thích đọc sách của từng người dùng, mang lại giá trị cao hơn trong việc tương tác với hệ thống.

### Recommended for you

Based on your interest



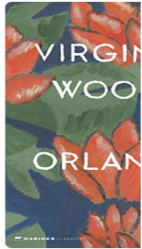
**Goddesses in Everywoman**  
A New Psychology of Women  
Jean Shinoda Bolen  
Feminine Archetypes  
★ 5 🛒 100  
100.000 đ 150.000 đ



**Dogeaters**  
(Contemporary Americ...  
Jessica Hagedorn  
Cultural Reflections  
★ 5 🛒 100  
100.000 đ 150.000 đ



**The Scarlet Letter**  
(Penguin Popular...  
Nathaniel Hawthorne  
A Tale of Sin  
★ 5 🛒 100  
100.000 đ 150.000 đ

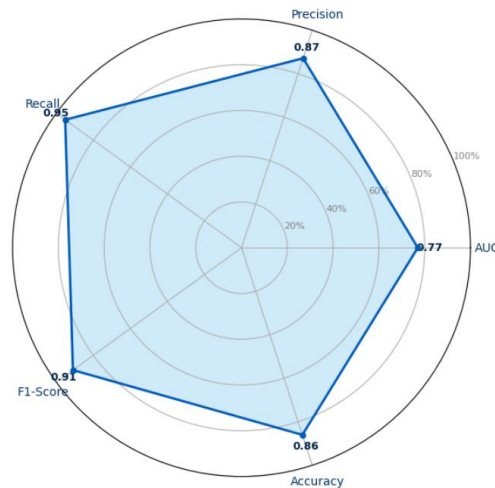


**Orlando: A Biog**  
Virginia Woolf  
A Life Unbound  
★ 5 🛒 100  
100.000 đ

Hình 3.13 Lấy model trong MLflow



### 3.2 Kết quả thực nghiệm và đánh giá



Hình 3.14 Chỉ số đánh giá model DeepFM cho hệ thống đề xuất sách

Mô hình DeepFM đạt hiệu suất tổng thể ở mức khá, đặc biệt nổi bật ở chỉ số Recall đạt 0.952, cho thấy khả năng bao phủ tương đối các đầu sách ưa thích trong hệ thống đề xuất. F1-score đạt 0.91, phản ánh sự cân bằng tốt giữa Precision và Recall, trong khi Precision đạt 0.871, thể hiện mô hình có khả năng dự đoán chính xác phần lớn các đề xuất phù hợp. Tuy nhiên, chỉ số AUC ở mức 0.770 cho thấy khả năng phân biệt giữa các đề xuất đúng và sai vẫn còn chưa được tốt. Tỷ lệ Accuracy đạt 0.860, chứng minh hiệu suất dự đoán tổng thể của mô hình là đáng tin cậy. Với hiệu suất Recall vượt trội, mô hình này đặc biệt phù hợp cho các ứng dụng yêu cầu không bỏ sót các mục tiêu quan trọng, tuy nhiên model cần tối ưu thêm Precision và AUC để giảm thiểu các đề xuất sai lệch và nâng cao hiệu quả phân loại.



## KẾT LUẬN

Hệ thống đã tích hợp các kỹ thuật hiện đại như embedding, Factorization Machine và mạng nơ-ron sâu để khai thác các mối quan hệ tiềm ẩn giữa người dùng và sản phẩm, từ đó cải thiện độ chính xác trong gợi ý. Bên cạnh đó, việc đưa MLflow vào pipeline huấn luyện và triển khai giúp theo dõi hiệu suất, quản lý phiên bản mô hình và hỗ trợ triển khai thực tế một cách linh hoạt và có hệ thống.

Tuy nhiên, hệ thống vẫn còn tồn tại một số điểm hạn chế, đặc biệt là sự phụ thuộc lớn vào chất lượng dữ liệu đầu vào — bao gồm thông tin người dùng và dữ liệu mô tả sách. Trong các tình huống cold-start, như người dùng hoặc sách mới chưa có lịch sử tương tác, độ chính xác của hệ thống gợi ý chưa đạt tối ưu.

Trong tương lai, để khắc phục các điểm yếu trên và nâng cao hiệu suất hệ thống, cần tích hợp thêm dữ liệu nội dung như phân tích văn bản mô tả sách, đánh giá của người dùng, cũng như tri thức chuyên ngành. Định hướng phát triển tiếp theo là xây dựng mô hình đề xuất lai (hybrid recommendation), kết hợp các phương pháp collaborative filtering, content-based filtering và tri thức ngữ nghĩa. Điều này sẽ giúp hệ thống phục vụ hiệu quả hơn cho nhiều nhóm người dùng, đồng thời đóng góp vào việc nâng cao doanh thu, cải thiện trải nghiệm khách hàng và tăng lợi thế cạnh tranh cho Bookies trong hành trình chuyển đổi số toàn diện.

## TÀI LIỆU THAM KHẢO

- [1] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, “DeepFM: A Factorization-Machine based Neural Network for CTR Prediction,” in Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI), 2017, pp. 1725–1731.
- [2] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, “DeepFM”, Dive into Deep Learning. April, 5, 2025.
- [3] Aggarwal, C. C. (2016). Recommender Systems: The Textbook. Springer.
- [4] Ricci, F., Rokach, L., & Shapira, B. (2015). Recommender Systems Handbook. Springer.
- [5] Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer
- [6] Kim Falk. (2019). Practical Recommendation Systems. Manning.