**TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP**

**KHOA ĐIỆN TỬ**

**Bộ môn: Công nghệ Thông tin.**

**BÀI TẬP KẾT THÚC MÔN HỌC**

MÔN HỌC

**KHOA HỌC DỮ LIỆU**

**Sinh viên: . . . . . . . . .ĐÀO NGUYỄN PHÚ QUÝ . . . . . . . . . . . . .**

**Lớp: . . . . . . . . . . . . . K57KMT. . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**Giáo viên GIẢNG DẠY: . . . . . . . . TS.NGUYỄN VĂN HUY. . . . . .**

**Link GitHub: ..https://github.com/daonguyenphuquy/btl-thay-huy….**

**.**

**THÁI NGUYÊN - 2025**

|  |  |
| --- | --- |
| **TRƯỜNG ĐHKTCN** | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM** |
| **KHOA ĐIỆN TỬ** | ***Độc lập - Tự do - Hạnh phúc*** |

**BÀI TẬP KẾT THÚC MÔN HỌC**

**MÔN HỌC: KHOA HỌC DỮ LIỆU**

BỘ MÔN : CÔNG NGHỆ THÔNG TIN

*Sinh viên: ĐÀO NGUYỄN PHÚ QUÝ*

*Lớp*: ……K57KMT…. *Ngành:………* KĨ THUẬT MÁY TÍNH…………………………

*Giáo viên hướng dẫn: NGUYỄN VĂN HUY*

*Ngày giao đề*  *Ngày hoàn thành*

*Tên đề tài :  Khuyến nghị phim cá nhân hóa*

*Yêu cầu :*

|  |
| --- |
| **GIÁO VIÊN HƯỚNG DẪN** |
| *(Ký và ghi rõ họ tên)* |

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*Thái Nguyên, ngày….tháng…..năm 20....*

**GIÁO VIÊN HƯỚNG DẪN**

*(Ký ghi rõ họ tên)*

**MỤC LỤC**

[**LỜI NÓI ĐẦU** 5](#_Toc199446599)

[**LỜI CẢM ƠN** 6](#_Toc199446600)

[**Chương 1: GIỚI THIỆU ĐẦU BÀI** 7](#_Toc199446601)

[**1.1. Tổng quan về đề tài** 7](#_Toc199446602)

[**1.2. Mục tiêu của đề tài** 7](#_Toc199446603)

[**1.3. Thách thức và giải pháp** 7](#_Toc199446604)

[**1.4. Kiến thức ứng dụng** 8](#_Toc199446605)

[**Chương 2: CƠ SỞ LÝ THUYẾT** 9](#_Toc199446606)

[**2.1. Hệ thống khuyến nghị (Recommendation Systems)** 9](#_Toc199446607)

[**2.2. Collaborative Filtering và SVD** 10](#_Toc199446608)

[**2.3. Đánh giá hệ thống khuyến nghị** 11](#_Toc199446609)

[**2.4. Flask và phát triển ứng dụng web** 11](#_Toc199446610)

[**2.5. Trực quan hóa dữ liệu với Chart.js** 12](#_Toc199446611)

[**Chương 3: THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH** 14](#_Toc199446612)

[**3.1. Sơ đồ khối hệ thống** 14](#_Toc199446613)

[**3.2. Sơ đồ khối các thuật toán chính** 15](#_Toc199446614)

[**3.3. Cấu trúc dữ liệu** 19](#_Toc199446615)

[**3.4. Chương trình** 20](#_Toc199446616)

[**Chương 4: THỰC NGHIỆM VÀ KẾT LUẬN** 24](#_Toc199446617)

[**4.1. Thực nghiệm** 24](#_Toc199446618)

[**4.2. Kết luận** 25](#_Toc199446619)

[**TÀI LIỆU THAM KHẢO** 28](#_Toc199446620)

# **LỜI NÓI ĐẦU**

Báo cáo này trình bày kết quả nghiên cứu và phát triển hệ thống khuyến nghị phim cá nhân hóa - một ứng dụng của khoa học dữ liệu trong lĩnh vực giải trí số. Trong thời đại bùng nổ thông tin và nội dung, các hệ thống khuyến nghị đóng vai trò quan trọng trong việc giúp người dùng tìm kiếm nội dung phù hợp với sở thích cá nhân.

Với mục tiêu xây dựng một ứng dụng khuyến nghị phim hiệu quả dựa trên đánh giá của người dùng, bài tập này đã áp dụng các kỹ thuật Collaborative Filtering kết hợp với Singular Value Decomposition (SVD) - một phương pháp phân tích ma trận được sử dụng rộng rãi trong các hệ thống khuyến nghị hiện đại.

Báo cáo sẽ trình bày chi tiết về cơ sở lý thuyết, quy trình thiết kế, xây dựng, thực nghiệm và đánh giá hệ thống khuyến nghị phim dựa trên bộ dữ liệu MovieLens.

# **LỜI CẢM ƠN**

Em xin gửi lời cảm ơn chân thành và sâu sắc đến TS. Nguyễn Văn Huy đã hướng dẫn, hỗ trợ và tạo điều kiện để em hoàn thành bài tập này. Những kiến thức, kinh nghiệm và phương pháp mà thầy đã truyền đạt trong suốt khóa học "Khoa học dữ liệu" là nền tảng quan trọng giúp em thực hiện thành công đề tài.

Em cũng xin cảm ơn tập thể các bạn trong lớp đã cùng trao đổi, chia sẻ kinh nghiệm và hỗ trợ trong quá trình học tập và nghiên cứu.

Đặc biệt, em xin gửi lời cảm ơn đến nhóm nghiên cứu MovieLens thuộc GroupLens Research đã cung cấp bộ dữ liệu phim quý giá, tạo điều kiện cho việc phát triển và thử nghiệm hệ thống khuyến nghị.

# **Chương 1: GIỚI THIỆU ĐẦU BÀI**

## **1.1. Tổng quan về đề tài**

Đề tài "Khuyến nghị phim cá nhân hóa" yêu cầu xây dựng một ứng dụng web/GUI có khả năng gợi ý phim dựa trên đánh giá của người dùng. Sử dụng bộ dữ liệu MovieLens làm nguồn dữ liệu đầu vào, hệ thống cần đưa ra danh sách phim được khuyến nghị phù hợp với sở thích của từng người dùng cụ thể.

Các tính năng chính cần triển khai bao gồm:

* Thuật toán Collaborative Filtering để phân tích mối tương quan giữa người dùng và phim
* Hiển thị danh sách khuyến nghị phim cho người dùng
* Trực quan hóa dữ liệu thông qua các biểu đồ đánh giá phim

## **1.2. Mục tiêu của đề tài**

- Xây dựng hệ thống khuyến nghị phim có khả năng dự đoán sở thích của người dùng dựa trên lịch sử đánh giá

- Thiết kế giao diện người dùng trực quan, thân thiện, dễ sử dụng

- Trực quan hóa dữ liệu phim thông qua các biểu đồ để người dùng hiểu hơn về xu hướng và đặc điểm của dữ liệu

- Áp dụng kiến thức về khoa học dữ liệu, học máy và phát triển ứng dụng web vào một bài toán thực tế

## **1.3. Thách thức và giải pháp**

**Thách thức:**

* Xử lý dữ liệu lớn và thưa (sparse data): Dữ liệu đánh giá phim thường rất thưa, mỗi người dùng chỉ đánh giá một tỷ lệ nhỏ trong tổng số phim.
* Tìm ra các mối tương quan ẩn giữa người dùng và phim
* Đảm bảo hiệu suất của hệ thống khi xử lý và tính toán trên lượng dữ liệu lớn
* Thiết kế giao diện trực quan và đáp ứng nhu cầu người dùng

**Giải pháp:**

* Áp dụng kỹ thuật Singular Value Decomposition (SVD) để giảm chiều dữ liệu và tìm ra các đặc trưng tiềm ẩn
* Sử dụng thuật toán Collaborative Filtering để dự đoán đánh giá và đề xuất phim
* Xây dựng hệ thống cache để lưu trữ kết quả tính toán, giảm thời gian xử lý
* Thiết kế giao diện web sử dụng Flask, HTML, CSS và JavaScript kết hợp với thư viện trực quan hóa Chart.js

## **1.4. Kiến thức ứng dụng**

Để thực hiện đề tài này, cần vận dụng các kiến thức sau:

* Phân tích và xử lý dữ liệu với Pandas và NumPy
* Thuật toán Collaborative Filtering và kỹ thuật SVD trong học máy
* Phát triển ứng dụng web với Flask
* Trực quan hóa dữ liệu với Chart.js
* Lập trình hướng đối tượng trong Python
* Thiết kế giao diện người dùng với HTML, CSS và JavaScript

# **Chương 2: CƠ SỞ LÝ THUYẾT**

## **2.1. Hệ thống khuyến nghị (Recommendation Systems)**

Hệ thống khuyến nghị là một công nghệ lọc thông tin nhằm dự đoán mức độ yêu thích của người dùng đối với các mục (items) và đề xuất những mục phù hợp nhất với sở thích của họ. Các hệ thống này đã trở thành một phần không thể thiếu trong nhiều ứng dụng thương mại điện tử, dịch vụ phát trực tuyến, và các nền tảng nội dung số khác.

**2.1.1. Phân loại hệ thống khuyến nghị**

1. **Content-based Filtering**: Đề xuất các mục tương tự với mục mà người dùng đã thích trước đó dựa trên đặc điểm của mục.
2. **Collaborative Filtering**: Đề xuất các mục dựa trên sự tương đồng giữa người dùng và/hoặc các mục.

* **User-based Collaborative Filtering**: Tìm kiếm người dùng tương tự và đề xuất mục mà họ đã thích.
* **Item-based Collaborative Filtering**: Tìm kiếm mục tương tự với mục mà người dùng đã thích.
* **Model-based Collaborative Filtering**: Sử dụng các mô hình học máy để tìm các mẫu ẩn trong dữ liệu.

1. **Hybrid Approaches**: Kết hợp các phương pháp khác nhau để tăng độ chính xác của khuyến nghị.

**2.1.2. Vấn đề Cold Start**

Vấn đề "Cold Start" (khởi đầu lạnh) xảy ra khi hệ thống không có đủ thông tin về người dùng mới hoặc mục mới để đưa ra khuyến nghị chính xác. Có hai loại chính:

* **User Cold Start**: Khi người dùng mới chưa có lịch sử tương tác
* **Item Cold Start**: Khi mục mới chưa được đánh giá bởi đủ người dùng

Giải pháp thường gặp bao gồm sử dụng thông tin nhân khẩu học, đề xuất các mục phổ biến, hoặc kết hợp với Content-based Filtering.

## 

## **2.2. Collaborative Filtering và SVD**

**2.2.1. Collaborative Filtering**

Collaborative Filtering (CF) là phương pháp khuyến nghị dựa trên giả định rằng người dùng có xu hướng thích những mục tương tự với những gì họ đã thích trong quá khứ, và những mục được thích bởi người dùng có sở thích tương tự.

Trong bài tập này, chúng ta tập trung vào Model-based Collaborative Filtering, cụ thể là sử dụng kỹ thuật SVD để giảm chiều dữ liệu và tìm ra các đặc trưng tiềm ẩn.

**2.2.2. Singular Value Decomposition (SVD)**

SVD là một phương pháp đại số tuyến tính để phân tách ma trận thành tích của ba ma trận con:

**A = U × Σ × V^T**

Trong đó:

* A là ma trận ban đầu (ma trận đánh giá người dùng-phim)
* U là ma trận các vector riêng của A×A^T (ma trận người dùng-đặc trưng)
* Σ là ma trận đường chéo chứa các giá trị riêng của ma trận A (mức độ quan trọng của các đặc trưng)
* V^T là ma trận chuyển vị của ma trận các vector riêng của A^T×A (ma trận đặc trưng-phim)

Trong hệ thống khuyến nghị, SVD giúp:

* Giảm chiều dữ liệu, xử lý được dữ liệu thưa
* Tìm ra các đặc trưng tiềm ẩn (latent features) mà người dùng quan tâm
* Dự đoán đánh giá của người dùng đối với phim chưa xem

**2.2.3. Ứng dụng SVD trong hệ thống khuyến nghị**

Các bước chính khi áp dụng SVD cho hệ thống khuyến nghị:

1. Xây dựng ma trận đánh giá user-item, với các giá trị thiếu được điền bằng 0 hoặc giá trị trung bình
2. Chuẩn hóa ma trận bằng cách trừ đi đánh giá trung bình của mỗi người dùng
3. Áp dụng SVD để phân tách ma trận thành U, Σ, và V^T
4. Chọn số lượng đặc trưng tiềm ẩn k (giảm chiều dữ liệu)
5. Tái tạo ma trận đánh giá gần đúng bằng cách nhân các ma trận đã giảm chiều
6. Dự đoán đánh giá cho các cặp user-item chưa có đánh giá

## **2.3. Đánh giá hệ thống khuyến nghị**

Các phương pháp đánh giá phổ biến:

1. **Root Mean Squared Error (RMSE)**: Đo lường sự khác biệt giữa đánh giá dự đoán và đánh giá thực tế.

**RMSE = sqrt(1/n \* Σ(predicted\_rating - actual\_rating)²)**

1. **Precision**: Tỷ lệ các mục được khuyến nghị mà người dùng thực sự thích.

**Precision = relevant\_items\_recommended / total\_items\_recommended**

1. **Recall**: Tỷ lệ các mục mà người dùng thích được khuyến nghị.

**Recall = relevant\_items\_recommended / total\_relevant\_items**

1. **F1 Score**: Trung bình điều hòa của Precision và Recall.

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

1. **Mean Average Precision (MAP)**: Đo lường chất lượng xếp hạng của các khuyến nghị.

## **2.4. Flask và phát triển ứng dụng web**

Flask là một micro web framework viết bằng Python, cung cấp các công cụ, thư viện và mẫu cần thiết để xây dựng ứng dụng web. Flask được thiết kế để đơn giản và dễ mở rộng, phù hợp cho các ứng dụng vừa và nhỏ.

**2.4.1. Kiến trúc MVC trong Flask**

Flask không áp đặt kiến trúc Model-View-Controller (MVC) một cách nghiêm ngặt, nhưng có thể được tổ chức theo mô hình này:

**- Model**: Xử lý dữ liệu và logic nghiệp vụ (trong trường hợp này là DataProcessor và MovieRecommender)

**- View**: Giao diện người dùng (các file HTML, CSS, JS)

**- Controller**: Điều khiển luồng ứng dụng (các route trong app.py)

**2.4.2. Jinja2 Template Engine**

Flask sử dụng Jinja2 làm công cụ tạo mẫu (template engine), cho phép nhúng mã Python trong các file HTML:

- Sử dụng {{ ... }} để hiển thị giá trị biến

- Sử dụng {% ... %} cho các cấu trúc điều khiển như vòng lặp và điều kiện

- Sử dụng {# ... #} cho các comment

**2.4.3. Flask Routing**

Routing trong Flask cho phép ánh xạ URL tới các hàm Python:

*@app.route('/user/<int:user\_id>')*

*def user\_profile(user\_id):*

*# Xử lý và hiển thị profile của người dùng*

## **2.5. Trực quan hóa dữ liệu với Chart.js**

Chart.js là một thư viện JavaScript mã nguồn mở để tạo các biểu đồ tương tác trên web. Nó hỗ trợ 8 loại biểu đồ chính: line, bar, radar, doughnut, pie, polar area, bubble, và scatter.

**2.5.1. Ưu điểm của Chart.js**

- Đáp ứng (responsive): Biểu đồ tự động điều chỉnh kích thước theo container

- Nhẹ: Kích thước thư viện nhỏ (~11KB khi nén)

- Tương tác: Hỗ trợ hoạt ảnh và tương tác người dùng

- Tùy biến: Nhiều tùy chọn để điều chỉnh giao diện và hành vi

**2.5.2. Tạo biểu đồ cơ bản với Chart.js**

*new Chart(document.getElementById('myChart'), {*

*type: 'bar',*

*data: {*

*labels: ['Red', 'Blue', 'Yellow'],*

*datasets: [{*

*label: 'My Dataset',*

*data: [12, 19, 3]*

*}]*

*},*

*options: {*

*// Các tùy chọn cấu hình*

*}*

*});*

# **Chương 3: THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH**

## **3.1. Sơ đồ khối hệ thống**

**3.1.1. Các module chính trong hệ thống**

Hệ thống khuyến nghị phim được thiết kế với 3 module chính:

1. **Module xử lý dữ liệu (DataProcessor)**:

- Tải và tiền xử lý dữ liệu từ bộ dữ liệu MovieLens

- Làm sạch và biến đổi dữ liệu (trích xuất năm, tách thể loại)

- Tính toán các thống kê của phim (số lượng đánh giá, điểm trung bình)

- Lấy URL poster phim từ TMDb API

- Lưu cache dữ liệu đã xử lý để tăng hiệu suất

1. **Module khuyến nghị (MovieRecommender)**:

- Xây dựng ma trận đánh giá người dùng-phim

- Áp dụng SVD để tìm các đặc trưng tiềm ẩn

- Dự đoán đánh giá cho các cặp người dùng-phim chưa có đánh giá

- Đề xuất phim dựa trên đánh giá dự đoán

- Tìm các phim tương tự dựa trên ma trận đặc trưng

1. **Module ứng dụng web (Flask Application)**:

- Cung cấp các API endpoint cho giao diện người dùng

- Xử lý yêu cầu và tương tác của người dùng

- Hiển thị kết quả khuyến nghị và trực quan hóa dữ liệu

- Kết nối các thành phần của hệ thống lại với nhau

**3.1.2. Biểu đồ phân cấp chức năng**

1. **Quản lý dữ liệu**

- Tải dữ liệu từ file CSV

- Tiền xử lý và làm sạch dữ liệu

- Tính toán thống kê phim

- Lấy thông tin bổ sung từ TMDb API

1. **Khuyến nghị phim**

- Xây dựng mô hình SVD

- Đề xuất phim cho người dùng cụ thể

- Tìm phim tương tự dựa trên nội dung

1. **Trực quan hóa dữ liệu**

- Hiển thị thống kê tổng quan

- Tạo biểu đồ phân phối đánh giá

- Tạo biểu đồ phân phối thể loại

- Tạo biểu đồ phân phối phim theo năm

- Tạo biểu đồ tương quan đánh giá và số lượng người đánh giá

1. **Giao diện người dùng**

- Hiển thị trang chủ với phim phổ biến

- Hiển thị trang khuyến nghị cho người dùng cụ thể

- Hiển thị trang phân tích dữ liệu

## **3.2. Sơ đồ khối các thuật toán chính**

**3.2.1. Thuật toán xử lý dữ liệu**

*ĐẦU VÀO: Các file CSV từ bộ dữ liệu MovieLens*

*|*

*v*

*KIỂM TRA CACHE*

*|*

*├── NẾU có cache -> Tải dữ liệu từ cache*

*|*

*└── NẾU không có cache:*

*|*

*v*

*TẢI DỮ LIỆU từ các file CSV*

*|*

*v*

*TRÍCH XUẤT NĂM từ tiêu đề phim*

*|*

*v*

*TÁCH THỂ LOẠI phim thành danh sách*

*|*

*v*

*THÊM TMDB ID từ file liên kết*

*|*

*v*

*LẤY URL POSTER từ TMDb API*

*|*

*v*

*TÍNH TOÁN THỐNG KÊ PHIM (số lượng đánh giá, điểm trung bình)*

*|*

*v*

*XỬ LÝ GIÁ TRỊ THIẾU*

*|*

*v*

*LƯU CACHE dữ liệu đã xử lý*

*|*

*v*

*ĐẦU RA: Dữ liệu phim, đánh giá, liên kết đã xử lý*

**3.2.2. Thuật toán khuyến nghị dựa trên SVD**

*ĐẦU VÀO: Dữ liệu đánh giá người dùng-phim*

*|*

*v*

*TẠO MA TRẬN ĐÁNH GIÁ (hàng = người dùng, cột = phim)*

*|*

*v*

*CHUẨN HÓA MA TRẬN (trừ đi đánh giá trung bình của mỗi người dùng)*

*|*

*v*

*ÁP DỤNG SVD để phân rã ma trận*

*|*

*v*

*TẠO MA TRẬN ĐÁNH GIÁ DỰ ĐOÁN = U × Σ × V^T + đánh giá trung bình*

*|*

*v*

*ĐẦU RA: Ma trận đánh giá dự đoán*

**3.2.3. Thuật toán đề xuất phim cho người dùng**

*ĐẦU VÀO: ID người dùng, Số lượng khuyến nghị*

*|*

*v*

*KIỂM TRA người dùng có trong dữ liệu huấn luyện không*

*|*

*├── NẾU không có -> Trả về phim phổ biến nhất*

*|*

*└── NẾU có:*

*|*

*v*

*LẤY ĐÁNH GIÁ DỰ ĐOÁN của người dùng*

*|*

*v*

*XÁC ĐỊNH CÁC PHIM đã được người dùng đánh giá*

*|*

*v*

*LỌC RA CÁC PHIM chưa được đánh giá*

*|*

*v*

*SẮP XẾP PHIM theo điểm dự đoán giảm dần*

*|*

*v*

*LẤY N PHIM có điểm dự đoán cao nhất*

*|*

*v*

*ĐẦU RA: Danh sách N phim được khuyến nghị*

**3.2.4. Thuật toán tìm phim tương tự**

*ĐẦU VÀO: ID phim, Số lượng phim tương tự cần tìm*

*|*

*v*

*KIỂM TRA phim có trong dữ liệu huấn luyện không*

*|*

*├── NẾU không có -> Trả về danh sách rỗng*

*|*

*└── NẾU có:*

*|*

*v*

*LẤY VECTOR ĐẶC TRƯNG của phim từ ma trận V^T*

*|*

*v*

*TÍNH ĐỘ TƯƠNG ĐỒNG COSINE giữa phim này và tất cả phim khác*

*|*

*v*

*SẮP XẾP PHIM theo độ tương đồng giảm dần*

*|*

*v*

*LẤY N PHIM có độ tương đồng cao nhất (bỏ qua phim đầu vào)*

*|*

*v*

*ĐẦU RA: Danh sách N phim tương tự*

## **3.3. Cấu trúc dữ liệu**

Hệ thống sử dụng bộ dữ liệu MovieLens Small, bao gồm các file CSV sau:

**3.3.1. movies.csv**

**- movieId** (int): ID duy nhất của phim

**- title** (string): Tiêu đề phim, thường bao gồm năm phát hành trong ngoặc đơn

**- genres** (string): Danh sách thể loại phim, phân cách bởi ký tự '|'

**3.3.2. ratings.csv**

**- userId** (int): ID duy nhất của người dùng

**- movieId** (int): ID duy nhất của phim

**- rating** (float): Đánh giá của người dùng cho phim, từ 0.5 đến 5.0 với bước 0.5

**- timestamp** (int): Thời gian đánh giá được ghi lại, định dạng Unix timestamp

**3.3.3. links.csv**

**- movieId** (int): ID duy nhất của phim

**- imdbId** (int): ID của phim trên IMDb

**- tmdbId** (int): ID của phim trên The Movie Database (TMDb)

**3.3.4. tags.csv**

**- userId** (int): ID duy nhất của người dùng

**- movieId** (int): ID duy nhất của phim

**- tag** (string): Tag được gán cho phim bởi người dùng

**- timestamp** (int): Thời gian tag được ghi lại, định dạng Unix timestamp

**3.3.5. Dữ liệu sau khi xử lý**

Sau khi xử lý, dữ liệu phim được bổ sung thêm các trường:

**- year** (int): Năm phát hành được trích xuất từ tiêu đề

**- genres\_list** (list): Danh sách thể loại dưới dạng list Python

**- poster\_url** (string): URL đến poster của phim từ TMDb

**- count** (int): Số lượng đánh giá cho phim

**- avg\_rating** (float): Điểm đánh giá trung bình của phim

## **3.4. Chương trình**

**3.4.1. DataProcessor (data\_processor.py)**

Lớp DataProcessor chịu trách nhiệm xử lý và quản lý dữ liệu:

*class DataProcessor:*

*def \_\_init\_\_(self, data\_path, tmdb\_api\_key='...'):*

*# Khởi tạo với đường dẫn đến thư mục dữ liệu MovieLens*

*def load\_or\_process\_data(self):*

*# Kiểm tra cache, nếu có thì tải, nếu không thì xử lý và lưu cache*

*def load\_data(self):*

*# Tải dữ liệu từ các file CSV*

*def get\_poster\_url(self, tmdb\_id):*

*# Lấy URL ảnh poster từ TMDb dựa trên tmdbId*

*def process\_data(self):*

*# Xử lý dữ liệu: làm sạch, biến đổi và tạo các features mới*

*def save\_cache(self):*

*# Lưu dữ liệu đã xử lý vào file cache*

*def get\_top\_rated\_movies(self, n=10, min\_ratings=10):*

*# Lấy n phim có đánh giá cao nhất (với ít nhất min\_ratings đánh giá)*

*def get\_all\_users(self):*

*# Lấy danh sách tất cả người dùng từ dữ liệu đánh giá*

*def get\_user\_ratings(self, user\_id):*

*# Lấy tất cả phim mà một người dùng đã đánh giá, kèm thông tin phim*

*def get\_movie\_info(self, movie\_id):*

*# Lấy thông tin chi tiết của một phim dựa trên ID*

*def get\_num\_users(self):*

*# Số lượng người dùng trong dataset*

*def get\_num\_movies(self):*

*# Số lượng phim trong dataset*

*def get\_num\_ratings(self):*

*# Tổng số đánh giá trong dataset*

*def get\_avg\_rating(self):*

*# Đánh giá trung bình của tất cả phim*

*def get\_rating\_distribution(self):*

*# Phân phối các đánh giá (bao nhiêu đánh giá cho mỗi điểm từ 0.5-5.0)*

*def get\_genre\_distribution(self):*

*# Phân phối các thể loại phim*

*def get\_movie\_data\_for\_charts(self):*

*# Chuẩn bị dữ liệu phim cho trực quan hóa*

**3.4.2. MovieRecommender (recommender.py)**

Lớp MovieRecommender triển khai thuật toán khuyến nghị phim:

*class MovieRecommender:*

*def \_\_init\_\_(self, data\_processor, num\_factors=50):*

*# Khởi tạo với bộ xử lý dữ liệu và số lượng nhân tố tiềm ẩn*

*def build\_model(self):*

*# Xây dựng mô hình khuyến nghị sử dụng SVD*

*def recommend\_for\_user(self, user\_id, num\_recommendations=10):*

*# Đưa ra khuyến nghị cho người dùng cụ thể*

*def get\_similar\_movies(self, movie\_id, num\_similar=10):*

*# Tìm các phim tương tự dựa trên ma trận đặc trưng từ SVD*

**3.4.3. Ứng dụng Flask (app.py)**

File app.py định nghĩa các route và điều khiển luồng ứng dụng:

*@app.route('/')*

*def index():*

*# Trang chủ của ứng dụng*

*@app.route('/user/<int:user\_id>')*

*def user\_profile(user\_id):*

*# Hiển thị trang profile của người dùng với các phim đã đánh giá*

*@app.route('/recommend', methods=['POST'])*

*def recommend():*

*# API endpoint để nhận khuyến nghị phim cho người dùng được chọn*

*@app.route('/analytics')*

*def analytics():*

*# Trang phân tích và trực quan hóa dữ liệu phim*

*@app.route('/api/movie\_data')*

*def movie\_data():*

*# API endpoint để lấy dữ liệu phim cho biểu đồ JavaScript*

**3.4.4. Giao diện người dùng (Templates)**

Hệ thống sử dụng ba template chính:

1. **index.html**: Trang chủ với form chọn người dùng và danh sách phim phổ biến
2. **recommendations.html**: Hiển thị khuyến nghị phim cho người dùng và lịch sử đánh giá
3. **analytics.html**: Hiển thị các biểu đồ và thống kê về dữ liệu phim

# **Chương 4: THỰC NGHIỆM VÀ KẾT LUẬN**

## **4.1. Thực nghiệm**

**4.1.1. Trang chủ**

Trang chủ hiển thị danh sách phim phổ biến nhất và form để người dùng chọn ID để nhận khuyến nghị cá nhân hóa.

Các tính năng chính:

- Danh sách phim phổ biến nhất với poster, tiêu đề, thể loại và đánh giá trung bình

- Form chọn ID người dùng và số lượng khuyến nghị

- Menu điều hướng đến các trang khác

**4.1.2. Trang khuyến nghị**

Trang khuyến nghị hiển thị danh sách phim được đề xuất cho người dùng cụ thể và lịch sử đánh giá của họ.

Các tính năng chính:

- Danh sách phim được khuyến nghị, sắp xếp theo độ phù hợp

- Thông tin chi tiết về mỗi phim bao gồm poster, tiêu đề, thể loại

- Bảng lịch sử đánh giá của người dùng, hiển thị các phim đã đánh giá, điểm đánh giá và thời gian

**4.1.3. Trang phân tích dữ liệu**

Trang phân tích dữ liệu hiển thị các biểu đồ và thống kê về bộ dữ liệu phim.

Các tính năng chính:

- Thống kê tổng quan về số lượng người dùng, phim, đánh giá và đánh giá trung bình

- Biểu đồ phân phối đánh giá

- Biểu đồ phân phối thể loại phim phổ biến

- Biểu đồ phân phối phim theo năm phát hành

- Biểu đồ tán xạ thể hiện tương quan giữa số lượng đánh giá và điểm trung bình

**4.1.4. Đánh giá chất lượng khuyến nghị**

Để đánh giá chất lượng của hệ thống khuyến nghị, chúng tôi đã tiến hành một số thử nghiệm:

1. **Kiểm tra các khuyến nghị cho người dùng khác nhau**:

- Người dùng có nhiều đánh giá nhận được khuyến nghị khá đa dạng và cá nhân hóa

- Người dùng có ít đánh giá vẫn nhận được khuyến nghị hợp lý dựa trên sở thích chung

1. **So sánh với phim phổ biến**:

- Khuyến nghị cá nhân hóa mang lại danh sách phim khác biệt đáng kể so với danh sách phim phổ biến

- Các khuyến nghị thường phản ánh thể loại phim mà người dùng đã đánh giá cao trước đó

1. **Kiểm tra thời gian phản hồi**:

- Việc sử dụng cache giúp tăng đáng kể tốc độ xử lý dữ liệu

- Tính toán khuyến nghị cho người dùng mới mất khoảng 2-3 giây

- Truy xuất khuyến nghị đã tính toán trước chỉ mất < 1 giây

## **4.2. Kết luận**

**4.2.1. Tổng kết kết quả đạt được**

Hệ thống khuyến nghị phim đã được xây dựng thành công với các tính năng:

- Xử lý và tiền xử lý dữ liệu từ bộ dữ liệu MovieLens

- Áp dụng thuật toán Collaborative Filtering kết hợp với SVD để dự đoán đánh giá phim

- Đề xuất phim cá nhân hóa cho người dùng cụ thể

- Tìm kiếm phim tương tự dựa trên đặc trưng tiềm ẩn

- Trực quan hóa dữ liệu phim thông qua các biểu đồ

- Giao diện web thân thiện với người dùng, dễ sử dụng

Hệ thống hoạt động tốt trên bộ dữ liệu MovieLens Small với khoảng 100,000 đánh giá của 600 người dùng cho 9,000 bộ phim. Việc sử dụng kỹ thuật SVD giúp giảm chiều dữ liệu và cải thiện chất lượng khuyến nghị, đặc biệt trong trường hợp dữ liệu thưa.

**4.2.2. Bài học kinh nghiệm**

Qua quá trình thực hiện đề tài, một số bài học kinh nghiệm đã được rút ra:

- Tiền xử lý dữ liệu là bước quan trọng, ảnh hưởng lớn đến chất lượng khuyến nghị

- Kỹ thuật giảm chiều như SVD rất hiệu quả trong việc xử lý dữ liệu thưa

- Cache dữ liệu và kết quả tính toán giúp cải thiện đáng kể hiệu suất hệ thống

- Trực quan hóa dữ liệu giúp hiểu rõ hơn về đặc điểm của dữ liệu và kết quả khuyến nghị

- Flask là framework hiệu quả để xây dựng ứng dụng web Python với độ phức tạp vừa phải

**4.2.3. Hướng phát triển tương lai**

Một số hướng phát triển để cải thiện hệ thống trong tương lai:

- Áp dụng các thuật toán học máy tiên tiến hơn như Deep Learning để nâng cao chất lượng khuyến nghị

- Kết hợp Content-based Filtering với Collaborative Filtering để xây dựng hệ thống lai (Hybrid)

- Cải thiện giao diện người dùng, thêm các tính năng tìm kiếm và lọc phim

- Mở rộng hệ thống để xử lý bộ dữ liệu lớn hơn (MovieLens 20M hoặc 25M)

- Thêm tính năng đánh giá phim trực tiếp để người dùng có thể cập nhật sở thích

- Cải thiện việc xử lý Cold Start Problem bằng cách kết hợp thông tin nhân khẩu học và phân tích nội dung phim

- Thêm khả năng giải thích khuyến nghị để người dùng hiểu tại sao một phim cụ thể được đề xuất

A screenshot of a movie

Description automatically generated

A screenshot of a computer

Description automatically generated

# **TÀI LIỆU THAM KHẢO**

1. Ricci, F., Rokach, L., & Shapira, B. (2022). Recommender Systems Handbook (3rd ed.). Springer.
2. Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8), 30-37.
3. Hug, N. (2020). Surprise: A Python library for recommender systems. Journal of Open Source Software, 5(52), 2174.
4. Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python (2nd ed.). O'Reilly Media.
5. Harper, F. M., & Konstan, J. A. (2015). The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems, 5(4), Article 19.
6. Downey, A. B. (2016). Think DSP: Digital Signal Processing in Python. O'Reilly Media.
7. McKinney, W. (2017). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython (2nd ed.). O'Reilly Media.
8. VanderPlas, J. (2016). Python Data Science Handbook: Essential Tools for Working with Data. O'Reilly Media.