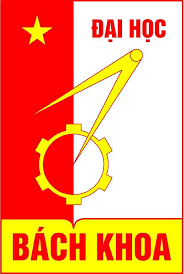
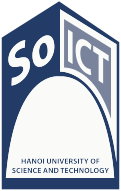
**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**NHẬP MÔN HỌC MÁY VÀ KHAI PHÁ DỮ LIỆU**

**Giảng viên hướng dẫn: PGS. Thân Quang Khoát**

**Đề tài: User’s Preference in Movies**

**Nhóm sinh viên thực hiện:**

**Họ và tên MSSV**

**Nguyễn Thành Long 20183585**

**Nguyễn Thành Long 20183948**

**Lê Hoàng Thái 20183983**

**Trần Hữu Hiếu 20180078**

**Hà Nội, 06/2021**

Mục lục

[**Phần mở đầu** 5](#_Toc74346978)

[**I.** **Giới thiệu bài toán** 6](#_Toc74346979)

[**1.** **Giới thiệu** 6](#_Toc74346980)

[**2.** **Các kĩ thuật chính trong RS** 6](#_Toc74346981)

[**3.** **Content-based filtering** 6](#_Toc74346982)

[**II.** **Mô tả và tiền xử lý dữ liệu** 8](#_Toc74346983)

[**1.** **Mô tả dữ liệu** 8](#_Toc74346984)

[**2.** **Tiền xử lí dữ liệu** 9](#_Toc74346985)

[**III.** **Phương pháp đánh giá** 10](#_Toc74346986)

[**IV.** **Xây dựng model** 11](#_Toc74346987)

[1. **KNN** 11](#_Toc74346988)

[**2.** **Matrix Factorization** 12](#_Toc74346993)

[**3.** **Neural Network** 14](#_Toc74346994)

[**4.** **Random Forest** 15](#_Toc74346995)

[**V.** **Kết quả thực nghiệm** 18](#_Toc74346996)

[**VI.** **Những khó khăn và hướng phát triển trong tương lai** 19](#_Toc74346997)

[**1.** **Những khó khăn trong quá trình làm việc** 19](#_Toc74346998)

[**2.** **Phát triển** 19](#_Toc74346999)

[**Kết luận** 19](#_Toc74347000)

[**Phân chia nhiệm vụ** 20](#_Toc74347001)

# **Phần mở đầu**

**Đặt vấn đề**

Ngày nay, giải trí là nhu cầu yếu của mỗi con người, một trong những nhu cầu phổ cập nhất là xem phim, và khi chọn phim, chúng ta có rất nhiều tiêu chí để lựa chọn đó là rate cao, thể loại mình thích, ... Với lượng thông tin ngày càng tăng trên internet và số lượng người dùng tăng lên đáng kể, điều quan trọng đối với các công ty là tìm kiếm, liên kết và cung cấp cho khách hàng những thông tin liên quan theo sở thích và thị hiếu của họ. Người dùng sử dụng các ứng dụng, trang web thường gặp các đôi chút khó khăn về tìm kiếm bộ phim phù hợp với nhu cầu, sở thích của họ do lượng phim lớn và thời gian có hạn. Và đó là lý do chúng em thiết kế hệ thống này để góp phần tối ưu hóa sự tiện lợi cho khách hàng.

1. **Giới thiệu bài toán**
2. **Giới thiệu**

Hệ thống gợi ý (Recommender Systems - RS) là một dạng của hệ thống lọc thông tin (information filtering), nó được sử dụng để dự đoán sở thích (preferences) hay xếp hạng (rating) mà người dùng có thể dành cho một mụcthông tin (item) nào đó mà họ chưa xem xét tới trong quá khứ (item có thể là bài hát, bộ phim, đoạn video clip, sách, bài báo,..).

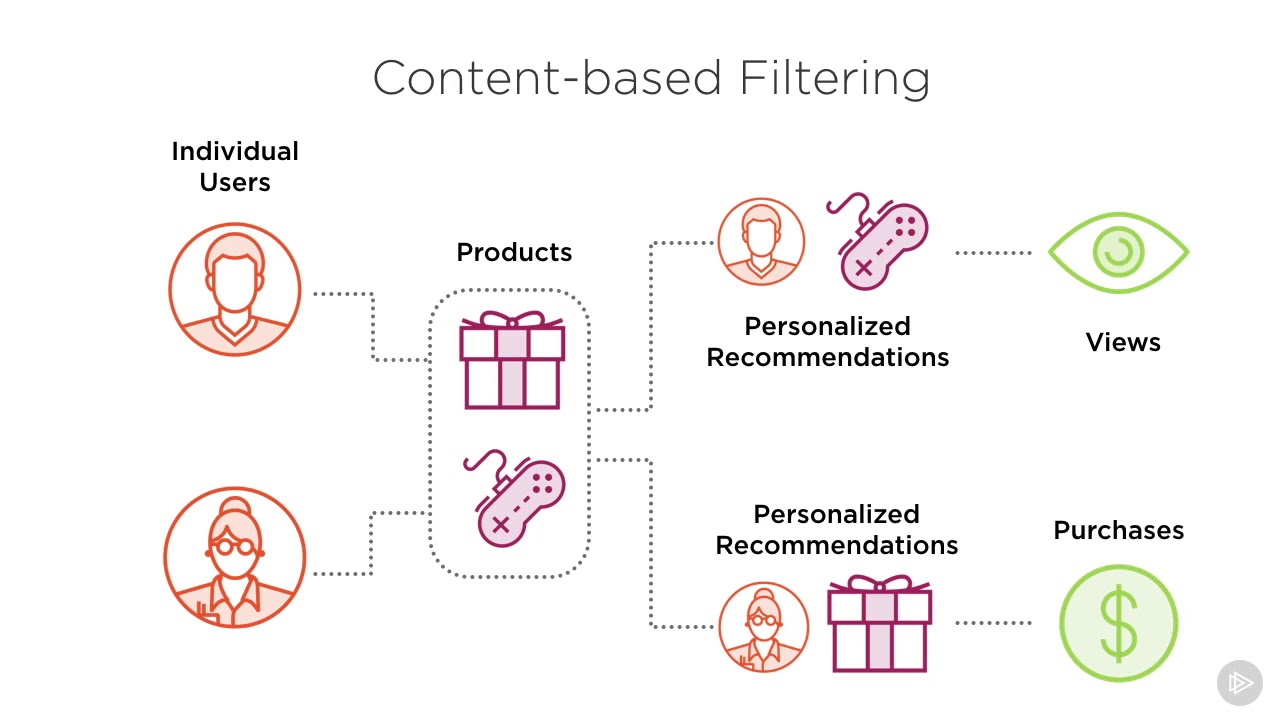
Và cụ thể với đề tài của chúng em, nhằm tối ưu hóa sử dụng của khách hàng (user), chúng ta quan tâm đến việc những khách hàng nào đã ‘yêu thích’ những bộ phim hay thể loại nào bằng cách dựa vào dữ liệu quá khứ của họ (dữ liệu này có thể là xếp hạng mà người dùng đã bình chọn, thể loại mà họ thường xem hay số lần click chuột,..) từ đó hệ thống sẽ dự đoán được người dùng có thể quan tâm đến bộ phim nào và đưa ra những gợi ý phù hợp cho họ.

1. **Các kĩ thuật chính trong RS**

Hiện tại, trong RS có rất nhiều giải thuật được đề xuất, trong số đó gồm giải thuật lọc theo nội dung (content-based filtering) giải thuật lọc cộng tác (collaborative filtering). ***Nhóm đề xuất sử dụng content-based filtering.***

1. **Content-based filtering**

**The content-based filtering method** là phương pháp dựa vào dữ liệu về các bộ phim mà người xem đã thích trong quá khứ để tính độ tương tự với bộ phim trong hệ thống. Từ đó, gợi ý những bộ phim tương tự với phim mà người xem đã thích, đã xem, đã mua trong quá khứ.



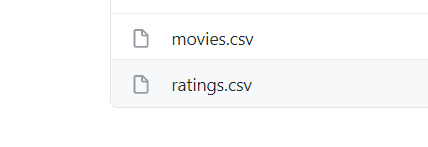
**Ưu điểm:**

* Gợi ý được những bộ phim phù hợp với sở thích của từng người xem riêng biệt.
* Gợi ý không phụ thuộc vào dữ liệu của các người xem khác.
* Gợi ý được những bộ phim tương tự với những bộ phim mà người xem đã thích trong quá khứ.

**Hạn chế:**

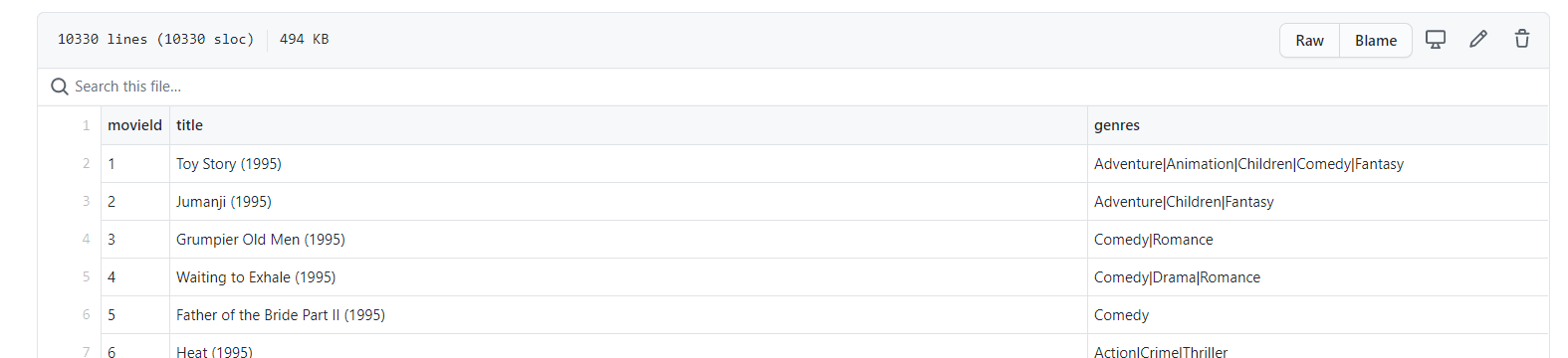
* Hồ sơ về bộ phim nếu không đúng có thể dẫn đến gợi ý sai.
* Gợi ý phụ thuộc hoàn toàn vào lịch sử của người xem. Vì vậy, không thể gợi ý nếu người xem không có lịch sử xem/thích các bộ phim trên hệ thống. Với người dùng mới, hệ thống không thể cung cấp gợi ý phù hợp.
* Không gợi ý được các bộ phim mới, chỉ có thể gợi ý các bộ phim tương tự như lịch sử đã xem/thích và không gợi ý được các sở thích mới của người xem.
* Người xem sẽ tiết kiệm được thời gian tìm kiếm phim, tìm được những bộ phim thực sự thích và cần thiết.

1. **Mô tả và tiền xử lý dữ liệu**
2. **Mô tả dữ liệu**

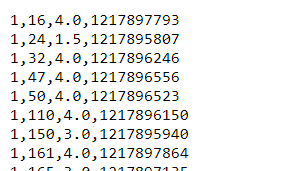


Dữ liệu gồm 2 file: movies.csv và ratings.csv, được lấy từ trang MovieLens.

File movies.csv có 10.000 bản ghi, gồm các trường: ID phim, tên phim (title), năm sản xuất, thể loại (genres). Chúng ta có thể nhìn thấy một số bộ phim quen thuộc như Toy Story, 12 Monkeys, Apollo 13, …



File ratings.csv gồm 100.000 bản ghi, gồm các trường (theo thứ tự từ trái sáng phải) là: ID của người đánh giá, ID của phim được đánh giá, số điểm (là 1 số trong khoảng 0,5 đến 5,0 và được làm tròn).



1. **Tiền xử lí dữ liệu**

Trong bước tiền xử lí dữ liệu, ở một số model (cụ thể là model Matrix Factorization) mà nhóm tác giả sử dụng sau này, cần có tập dữ liệu training mà số liệu của nó phải thực sự chân thực và tương đối ổn định. Tuy nhiên, trên thực tế, một số người dùng có thói quen xếp hạng bộ phim mà mình vừa xem với số sao rất cao hoặc rất thấp, một phần vì mức đánh giá tiêu chỉ của họ cao/thấp hơn hẳn so với mặt bằng chung, một phần có thể là do thói quen. Việc cho số sao không sát với mặt bằng chung của xã hội như vậy sẽ dẫn đến sai lệch không đáng có khi triển khai các model sau này. Bởi vậy, trong quá trình tiền xử lí dữ liệu, nhóm tác giả đã thực hiện chuẩn hoá (normalizations) toàn bộ tập dữ liệu, nhằm đưa các giá trị trong cùng một cột của các file .csv thành một giá trị tỉ lệ chung. Việc chuẩn hoá này cũng được thực hiện sao cho không làm biến dạng phạm vi giá trị.

1. **Phương pháp đánh giá**

Để thuận tiện cho việc so sánh độ chính xác của các mô hình, nhóm em dùng chung một phương pháp và hàm đánh giá.

Phương pháp đánh giá đã được sử dụng là phương pháp Hold-out. Ưu điểm của phương pháp này là tốc độ nhanh và đơn giản. Nhưng bù lại thì nó lại có nhược điểm là tùy theo cách chọn ngẫu nhiên data vào tập train và tập test mà tập train có thể không biểu diễn được hình dạng của bộ dữ liệu, từ đó dễ dẫn đến overfitting. Để giảm thiểu vấn đề này, dữ liệu của từng user được chia ngẫu nhiên thành 2 bộ dữ liệu train và test rồi mới gộp lại. Trong tổng số 105339 bản ghi, có 84013 bản ghi thuộc bộ dữ liệu train và 21326 bản ghi thuộc bộ test.

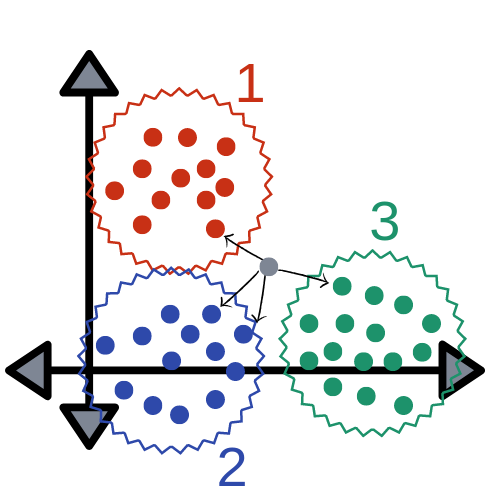
Để đánh giá dự đoán được mô hình đưa ra có giống với dự đoán của người dùng không, nhóm em sử dụng hàm Root-Mean-Square Deviation (RMSE) làm tiêu chuẩn.

Hàm này thường được sử dụng để đánh giá kết quả của các mô hình dự đoán trên một mô hình xác định nào đó (vì RMSE phụ thuộc vào kích thước bộ dữ liệu). RMSE là độ lệch chuẩn của các sai số trong phép dự đoán hiệu quả của mô hình. RMSE càng lớn thì chứng tỏ sai lệch càng nhiều. Mục tiêu của các mô hình là tối thiểu hóa giá trị RMSE.

1. **Xây dựng model**
2. **KNN**

**Khái niệm: K-nearest neighbor** là một trong những thuật toán **supervised-learning** đơn giản nhất (mà hiệu quả trong một vài trường hợp) trong Machine Learning. Khi training, thuật toán này không học một điều gì từ dữ liệu training, mọi tính toán được thực hiện khi nó cần dự đoán kết quả của dữ liệu mới. K-nearest neighbor có thể áp dụng được vào cả hai loại của bài toán **Supervised learning** là [**Classification**](https://machinelearningcoban.com/2016/12/27/categories/#classification-phan-loai)và [**Regression**](https://machinelearningcoban.com/2016/12/27/categories/#regression-hoi-quy).

**Thuật toán: KNN** sử dụng thước đo khoảng cách nhỏ nhất để tìm những người hàng xóm gần nhất của nó. Một phương pháp phổ biến là phương pháp khoảng cách Euclide (Căn bậc hai của tổng bình phương khoảng cách giữa hai điểm).



Trong bài toán này, nhóm của chúng em áp dụng thuật toán KNN các vector N trường (N là số lượng bộ phim). Hàm khoảng cách được sử dụng là khoảng cách Euclid. Đối với mỗi điểm dữ liệu x(i, j), KNN sẽ tìm k người dùng có khoảng cách nhỏ nhất tới người dùng i đang xét. Điểm đánh giá dự đoán của user đó đến bộ phim j bằng trung bình điểm trung bình đánh giá các user khác với phim j.

Nhóm đã thử chạy thuật toán với nhiều giá trị k khác nhau để tìm giá trị tốt nhất.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| k | 3 | 5 | 10 | 20 | 50 |
| RMSE | 3.361 | 3.358 | 3.359 | 3.372 | 3.396 |

Thuật toán KNN cho ra kết quả tương đối kém do bộ dữ liệu tương đối thưa, đa phần ma trận đánh giá đều ra kết quả bằng 0. Nếu để k quá nhỏ thì có thể dẫn đến overfitting nhưng khi để k quá lớn thì các hang xóm sẽ không còn giống với người dùng và dẫn đến underfitting.

1. **Matrix Factorization**

Ý tưởng của phương pháp này dựa vào việc mỗi bộ phim có thể mang một số thuộc tính ẩn. Mỗi thuộc tính ẩn này sẽ được gắn với một giá trị để đo các độ liên quan giữa bộ phim và thuộc tính đó. Tương tự ta cũng có mức độ yêu thích của một người dùng với từng thuộc tính ẩn. Kết hợp 2 yếu tố này ta thấy mức độ yêu thích của một người dùng với từng thuộc tính càng giống độ liên quan của một bộ phim tới các thuộc tính đó thì khả năng người dùng thích bộ phim là càng cao.

Điều này tương tự như cách làm của phương pháp content-based nhưng chúng ta không tập trung vào việc tìm các thuộc tính đó mà cố gắng dựa vào liên hệ giữa các người dùng với nhau để dự đoán giá trị độ yêu thích của người dùng và độ liên quan của bộ phim tới các thuộc tính ẩn.

Ta có thể biểu diễn các thuộc tính này dưới dạng ma trận, ví dụ:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| W | 1.5 | 3 | 2.2 | 2 |
| 0.25 | 0.15 | 0.4 | 1 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| X | |  | | Y | | |  | |
| 1.2 | 2.5 |  | 2.43 | 3.98 | 3.64 | 4.9 | |
| 0.35 | 1 | 0.78 | 1.2 | 1.17 | 1.7 | |
| 0.4 | 2 | 1.1 | 1.5 | 1.68 | 2.8 | |
| 1.1 | 1 | 1.9 | 3.45 | 2.82 | 3.2 | |
| 0.5 | 3 | 1.5 | 1.95 | 2.3 | 4 | |

Trong hình trên ta có ma trận W là ma trận liên quan của 4 bộ phim với 2 thuộc tính. X là ma trận độ yêu thích của 5 người dùng và 2 thuộc tính. Ma trận Y là ma trận độ yêu thích của 5 người dùng với 4 bộ phim

Ta có:

Hay

Ban đầu ta sẽ khởi tạo ngẫu nhiên giá trị của hai ma trận X và W. Sau đó, ta sẽ huấn luyện hai ma trận này để tích của chúng . Ở đây ta chỉ so sánh Y và khi giá trị y(i, j) khác 0, sau đó ta sẽ điền những phần bằng 0 trong Y bằng giá trị dự đoán của .

Phương pháp để huấn luyện mô hình này là phương pháp Gradient Decent. Hàm loss để so sánh kết quả dự đoán với giá trị thực là:

Hàm g(X,W) là một hàm regularization để tránh overfitting.

Trong đó là chuẩn Forbineous của X, có giá trị bằng căn bậc hai của tổng bình phương tất cả các phần tử trong ma trận. Đây là chuẩn thường được sử dùng làm hàm regularization cho các bài toán regression trên ma trận.

Sau khi thử nghiệm với số vòng lặp, learning rate khác nhau cho Gradient Decent và số thuộc tính ẩn k khác nhau, kết quả tốt nhất nhóm thu được là

với , hội tụ sau khoảng 50 vòng lặp.

Với learning rate lớn hơn thì quá trình hội tụ sẽ chậm hơn còn với learning rate cao hơn thì có khả năng nhảy qua mất điểm cực trị. Với giá trị số thuộc tính ẩn cao hơn thì bài toán trở nên phức tạp hơn, dễ dẫn đến overfitting, kết quả thu được cũng kém hơn so với **k = 5**.

1. **Neural Network**

Mô hình này sử dụng cùng chung một ý tưởng về các thuộc tính phụ giống như Matrix Factorization nhưng thay vì sử dụng Gradient Decent thì ta sử dụng một mạng neuron để huấn luyện X và W.

Trong mô hình này thì nhóm sử dụng công cụ Keras của Google để xây dựng mạng neuron. Tầng đầu tiên của mô hình là một tầng nhúng 2 ma trận X và W. Sau đó qua một tầng ẩn nữa và thu được ở tầng output là kết quả là giá trị đánh giá dự đoán của .

Trên các tầng này, nhóm còn dùng phương pháp Dropout để giảm overfitting. Phương pháp này được thực hiện bằng cách mỗi lần dữ liệu đến với một tầng thì chỉ có một phần ngẫu nhiên của tầng đó hoạt động, dữ liệu không được đưa đến những nút còn lại. Việc này khiến duyệt qua mạng đơn giản hơn và tránh được hiện tượng overfitting. Kết quả tốt nhất mà nhóm thu được với phương pháp này là .

1. **Random Forest**

**Khái niệm: Random Forest** là một thuật toán học có giám sát, sử dụng các cây (tree) để làm nền tảng. Đây là một tập hợp của các **Decision Tree**, mà mỗi cây được chọn theo một thuật toán dựa vào ngẫu nhiên.

+ **Decision Tree** là tên đại diện cho một nhóm thuật toán phát triển dựa trên *Cây quyết định*. Ở đó, mỗi *Node* của cây sẽ là các thuộc tính, và các nhánh là giá trị lựa chọn của thuộc tính đó. Bằng cách đi theo các giá trị thuộc tính trên cây, *Cây quyết định* sẽ cho ta biết giá trị dự đoán. Nhóm thuật toán Decision Tree có một điểm mạnh đó là có thể sử dụng cho cả bài toán Phân loại (Classification) và Hồi quy (Regression).

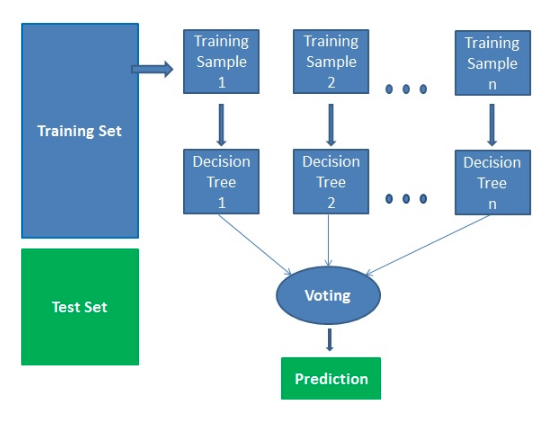
RF hoạt động theo bốn bước:

***B1:*** Chọn các mẫu ngẫu nhiên từ tập dữ liệu đã cho.

***B2:*** Thiết lập cây quyết định cho từng mẫu và nhận kết quả dự đoán từ mỗi quyết định cây.

***B3:*** Hãy bỏ phiếu cho mỗi kết quả dự đoán.

***B4:*** Chọn kết quả được dự đoán nhiều nhất là dự đoán cuối cùng.



Ưu điểm:

* Là một phương pháp chính xác và mạnh mẽ vì số cây quyết định tham gia vào quá trình này. Nó không bị vấn đề overfitting.
* Thuật toán có thể được sử dụng trong cả hai vấn đề phân loại và hồi quy.

Nhược điểm:

* Có số lượng lớn cây có thể làm cho thuật toán chậm và kém hiệu quả đối với các dự đoán thời gian thực.
* Mô hình khó hiểu hơn so với cây quyết định do ta có thể dễ dàng đưa ra quyết định bằng cách đi theo đường dẫn trong cây.

Ngoài cách lọc theo cộng tác Collaborative Filtering thì nhóm chúng em còn kết hợp song song với việc lọc theo nội dung Content-based filtering để tối ưu bài toán Recommend System.

Đầu tiên chúng em sẽ tiền xử lý dữ liệu hai File Ratings.csv và File Movies.csv. Với File Ratings.csv gồm bốn cột: UserId, MovieId, Rating, timestamp thì sẽ đọc vào DataFrame Ratingdata và xoá bỏ cột không cần thiết là timestamp.

Còn File thứ 2 gồm 3 cột: MovieId, title, genres thì chúng em sẽ xử lý như sau. Đầu tiên tương tự như trên, đọc file vào dataFrame moviedata. Từ trường title gồm 2 thông tin: movie name + year nên chúng em sẽ tách movieName ra thành 1 trường riêng. Tiếp theo từ cột Genres ta sẽ tạo ra 1 list các genre, với mỗi genre sẽ tương ứng với một trường riêng mới. Nếu dataFrame này có genres chứa 1 genra A bất thì trường A bằng 1 tại vị trí tương ứng với movie đó, ngược lại thì bằng 0. Cuối cùng in ra list các genres và moviedata trước và sau khi tiền xử lý.

Tiếp theo để training lọc theo nội dung chúng em đã chọn model RandomForest dạng Regression. Từ 2 dataFrame đã tiền xử lý, chúng em lọc ra trường UserId của Ratingdata và 19 trường genre của Moviedata làm Input Bộ dữ liệu thì được chia một cách ngẫu nhiên thành hai phần 80% Train- 20% Test.

Thuật toán Random Forest sẽ tạo ra K tree bằng cách sau. Đầu tiên sẽ dùng BootStrap Sampling để tạo ra Traning Set tương ứng với Cây thứ i. Với mỗi node cây sẽ chọn một tập con các Attributes và split theo tập con đó. Hàm dùng để đo độ Split là MSE. Cây sẽ được expand tối đa và không cần cắt tỉa. Giá trị output sẽ được xác định là giá trị trung bình dự đoán của tất cả các cây.

Cuối cùng chúng em dùng hàm đánh giá là Root Mean Squared Error để so sánh kết quả dự đoán với giá trị thực

Nhóm đã thử với số lượng cây khác nhau để tìm ra giá trị tốt nhất.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| K | 10 | 50 | 100 | 300 |
| RMSE | 0,976 | 0,98 | 0,974 | 0,977 |

Thuật toán RadomForest cho ra kết quả khá tốt khi tăng số cây lên.

1. **Kết quả thực nghiệm**

Các thư viện mà nhóm tác giả đã sử dụng: Pandas, Tensorflow, Sklearn.

Kết quả thu được với mỗi mô hình là:

|  |  |
| --- | --- |
| Model | RMSE |
| Collaborative Filtering | |
| KNN | 3,358 |
| MF | 0.974 |
| ANN | 0.921 |
| Content-based Filtering | |
| RF | 0.971 |

Ta thấy KNN thu được kết quả kém xa so với các phương pháp còn lại. Cả ba phương pháp còn lại đều có kết quả khá tương đồng, nhưng neural network là phương pháp cho kết quả tốt nhất.

Với kết quả RMSE sấp sỉ 1 và số lượng rating test vào khoảng 20000 thì sai số trung bình là rất nhỏ.

1. **Những khó khăn và hướng phát triển trong tương lai**
2. **Những khó khăn trong quá trình làm việc**

Trong quá trình tìm hiểu và phát triển hệ thống, nhóm còn gặp đôi chút khó khăn trong việc tìm kiếm nguồn dữ liệu, tìm kiếm giải pháp tối ưu cho các thuật toán được sử dụng trong đề tài. Đó là một rào cản lớn trong việc hoàn thành sản phẩm một cách tối ưu và hiệu quả nhất.

1. **Phát triển**

Tuy gặp một số khó khăn trong quá trình làm việc, thuật toán và hệ thống vẫn đang còn nhiều hạn chế. Trong tương lai chúng em sẽ tìm hiểu và đưa ra những giải pháp và hướng đi tốt hơn đối với sản phẩm, cải tiến và đưa ra thêm nhiều tính năng mới với mong muốn góp phần tạo ra sản phẩm tốt và hữu dụng đối với khách hàng.

# **Kết luận**

Thông qua việc thực hiện đề tài “**User’s Preference in Movie**”, nhóm đã tích lũy, làm rõ được rất nhiều kiến thức thực tế và các thuật toán được sử dụng trong đề tài. Với tinh thần học hỏi, làm việc nghiêm túc cùng với sự giúp đỡ lẫn nhau giữa các thành viên trong nhóm, chúng em đã hoàn thành đề tài với sản phẩm có độ tin cậy khá cao đối với người dùng. Tuy nhiên sản phẩm vẫn còn một số hạn chế nhất định mà chúng em vẫn chưa khắc phục được. Trong tương lai nhóm cũng sẽ tìm cách cải tiến, đưa ra tính năng mới để tạo ra một sản phẩm thông minh, phù hợp và thuận tiện với khách hàng.

# **Phân chia nhiệm vụ**

**Nguyễn Thành Long (20183585): viết phần model Matrix Factorization và Neural Network**

**Nguyễn Thành Long (20183948): tìm, đọc, xử lí data, viết báo cáo**

**Lê Hoàng Thái (20183983): viết phần model KNN**

**Trần Hữu Hiếu (20180078): viết phần model Random Forest**