**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**---🙠**🕮**🙢---**



**BÁO CÁO ĐỒ ÁN MÔN HỌC**

**Nhập môn Học máy và Khai phá dữ liệu**

**Đề tài: Movie Recommedation System**

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

Mã lớp : *131404*

Nhóm sinh viên thực hiện*: Cao Như Đạt – 20194013*

*Nguyễn Huy Hoàng – 20194058*

*Lê Huy Hoàng – 20190053*

MỤC LỤC

[Lời nói đầu 2](#_Toc13035)

[1. Giới thiệu bài toán: Movie Recommendation System 3](#_Toc7311)

[2. Tiến hành trên bộ dữ liệu TMDB 5](#_Toc18187)

[2.1 Tổng quan về bộ dữ liệu TMDB 5](#_Toc1293)

[2.2 Khai báo thư viện sử dụng 6](#_Toc30617)

[2.3 Tiền xử lý dữ liệu 6](#_Toc11116)

[2.4 Thuật toán 9](#_Toc31226)

[2.5 Một số đánh giá và kết luận 11](#_Toc27372)

[3. Tiến hành trên bộ dữ liệu MovieLens 12](#_Toc29235)

[4. Kết luận 12](#_Toc15924)

# Lời nói đầu

Hiện nay, hầu khắp trên các ứng dụng hay phương tiện truyền thông chúng ta đều nhận thấy sự xuất hiện của Recommendation System (Hệ thống gợi ý). Khi ta xem một video, YouTube tự động gợi ý các video có nội dung liên quan hoặc có thể tự chuyển tiếp khi ta xem xong. Tương tự với Neflix hay Spotify, những bộ phim hay bản nhạc mà họ gợi ý có ý nghĩa rất lớn trong sự phát triển mạnh mẽ của họ trong thời gian gần đây.

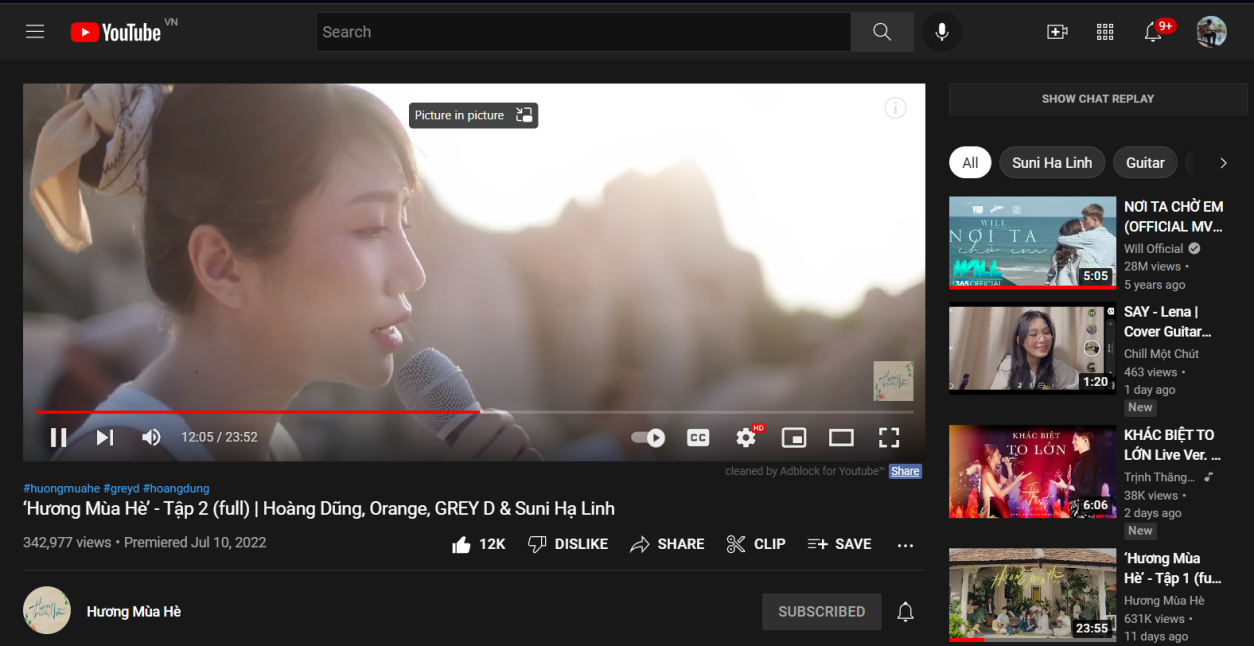
Nhận thấy đây là một đề tài thú vị và có nhiều ý nghĩa trong thực tiễn, nhóm chúng em quyết định lựa chọn đề tài Movie Recommendation System (Hệ thống gợi ý phim) làm đề tài báo cáo của môn học Nhập môn Học máy và Khai phá dữ liệu. Quá trình thực hiện đề tài không thể tránh khỏi thiếu sót, chúng em rất mong nhận được những sự góp ý của thầy!

# Giới thiệu bài toán: Movie Recommendation System

Một *hệ thống gợi ý* (hay cũng có thể gọi là *hệ thống khuyên dùng*, [tiếng anh](https://vi.wikipedia.org/wiki/Ti%E1%BA%BFng_Anh" \o "Tiếng Anh): *recommendation system*) là một lớp con của *[hệ thống lọc thông tin](https://vi.wikipedia.org/w/index.php?title=H%E1%BB%87_th%E1%BB%91ng_l%E1%BB%8Dc_th%C3%B4ng_tin&action=edit&redlink=1" \o "Hệ thống lọc thông tin (trang không tồn tại))*, tìm kiếm dự đoán "đánh giá" hoặc "ưa thích" của người dùng với một sản phẩm hoặc đối tượng nào đó.

Các hệ thống gợi ý được tận dụng trong nhiều lĩnh vực, chẳng hạn như trình tạo danh sách phát cho video và âm nhạc của một số ứng dụng như Netflix, YouTube và Spotify. Sau khi người xem một vài video hoặc nghe một số bài hát, hệ thống gợi ý sẽ nhận diện sở thích người dùng và mức độ tương tác với video/bài hát để tạo ra một danh sách gợi ý theo chủ đề và mức độ liên quan mà người dùng có thể ưa thích. Hệ thống gợi ý còn áp dụng cho các dịch vụ gợi ý sản phẩm ở Amazon, các gợi ý nội dung cho các nền tảng xã hội như Facebook hay Twitter.

Youtube tự động đề xuất các video khác liên quan khi người dùng sử dụng:



Hệ thống gợi ý (Recommendation system) dần trở thành một thành phần không thể thiếu của các sản phẩm điện tử có nhiều người dùng. Các sản phẩm cá nhân hóa điện tử ngày càng phổ biến với mục đích mang sản phẩm phù hợp tới người dùng hoặc giúp người dùng có các trải nghiệm tốt hơn. Nếu quảng cáo sản phẩm tới đúng người dùng, khả năng các món hàng được mua nhiều hơn. Nếu gợi ý một video mà người dùng nhiều khả năng thích hoặc gợi ý kết bạn đến đúng đối tượng, họ sẽ ở lại trên nền tảng của bạn lâu hơn. Khi họ ở trên nền tảng của bạn lâu hơn, họ sẽ nhìn thấy nhiều quảng cáo hơn và lợi nhuận từ quảng cáo sẽ lại càng nhiều hơn.

Quảng cáo điện tử ngoài việc giúp các doanh nghiệp bán được nhiều hàng còn giúp họ tiết kiệm được chi phí kho bãi. Họ sẽ không cần các cửa hàng ở vị trí thuận lợi để thu hút khách hàng hay phải trưng ra mọi mặt hàng ở vị trí đắc địa nhất trong cửa hàng. Mọi thứ có thể được cá nhân hóa sao cho mỗi người dùng nhìn thấy những sản phẩm khác nhau phù hợp với nhu cầu và sở thích của họ.

Quảng cáo trên Internet đã chiếm thị phần ngày càng cao so với quảng cáo truyền hình nhờ sự đa dạng và cá nhân hóa một cách tối đa. Một người dùng 20-30 tuổi thường xuyên nghe nhạc rap ít có khả năng thích nhạc Bolero. Một người dùng tìm kiếm các thông tin về xe hơi nhiều khả năng sắp mua xe và quan tâm tới những dịch vụ sửa và rửa xe. Một người dùng thường xuyên xe các video về làm vườn nhiều khả năng sẽ quan tâm tới việc mua bán hạt giống. Từ những thông tin thu thập được từ hành vi người dùng, hệ thống có thể gợi ý ra những lựa chọn phù hợp để đạt được hiệu quả cao nhất.

Các hệ thống gợi ý thường được chia thành ba nhóm chính:

**Lọc Nhân khẩu học (**Demographic Filtering):**** Đưa ra các đề xuất tổng quát cho mọi người dùng, dựa trên mức độ phổ biến và / hoặc thể loại phim. Hệ thống đề xuất những bộ phim tương tự cho những người dùng có các đặc điểm nhân khẩu học tương tự. Vì mỗi người dùng khác nhau nên cách làm này được coi là quá đơn giản. Ý tưởng cơ bản đằng sau hệ thống này là những bộ phim nổi tiếng hơn và được giới phê bình đánh giá cao hơn sẽ có xác suất được khán giả bình thường thích cao hơn.

****Lọc dựa trên nội dung (Content-based Filtering)****: nhóm thuật toán này gợi ý cho người dùng những sản phẩm tương tự như những sản phẩm mà người dùng đã có phản hồi tích cực. Hệ thống này cần xây dựng đặc trưng cho các sản phẩm sao cho những sản phẩm tương tự nhau có khoảng cách tới nhau nhỏ. Việc này khá tương tự như việc xây dựng các embedding cho các sản phẩm. Việc dự đoán cho mỗi người dùng hoàn toàn chỉ dựa trên lịch sử thông tin của người dùng đó.

****Lọc cộng tác (Collaborative Filtering)****: nhóm thuật toán này không chỉ dựa trên thông tin về sản phẩm tương tự mà còn dựa trên hành vi của những người dùng tương tự. Ví dụ: người dùng A, B, C đều thích các bài hát của Noo Phước Thịnh. Ngoài ra, hệ thống biết rằng B, C cũng thích các bài hát của Bích Phương nhưng chưa có thông tin về việc liệu user A có thích Bích Phương hay không. Dựa trên thông tin của những người dùng tương tự là B và C, hệ thống có thể dự đoán rằng A cũng thích Bích Phương và gợi ý các bài hát của ca sĩ này tới A.

Trong phạm vi nội dung của bài báo cáo, nhóm chúng em chỉ tập trung vào hệ thống gợi ý sử dụng phương pháp Lọc dựa trên nội dung (Content-based Filtering).

# Tiến hành trên bộ dữ liệu TMDB

## Tổng quan về bộ dữ liệu TMDB

Bộ dữ liệu TMDB được tạo từ The Movie Database API và được công khai trên Kagle: [TMDB 5000 Movie Dataset | Kaggle](https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata). Bộ dữ liệu mang các thông tin chi tiết của gần 5000 bộ phim trong hai file csv chứa các trường thông tin sau:

**tmdb\_5000\_movies.csv:**

*- budget:* ngân sách của phim

*- genres:* thể loại phim

*- homepage:* trang chủ của phim

*- id:* mã phim

*- keywords:* từ khóa

*- original\_language:* ngôn ngữ ban đầu sử dụng trong phim

*- original\_title:* tiêu đề phim ban đầu

*- overview:* tổng quan về phim

*- popularity:* độ phổ biến của phim

*- production\_companies:* hãng phim

*- production\_countries:* quốc gia

*- release\_date:* ngày phát hành

*- revenue:* thu nhập

*- runtime:* thười lượng phim

*- spoken\_languages:* ngôn ngữ nói trong phim

*- status:* trạng thái hiện tại của phim

*- tagline:* khẩu hiệu cho phim

*- title:* tiêu đề

*- vote\_average:* điểm bình chọn trung bình

*- vote\_count:* số lượng người bình chọn

**tmdb\_5000\_credits.csv:**

*- movie\_id:* mã phim (tương ứng với id của file trên)

*- title:* tiêu đề phim

*- cast:* dàn diễn viên (bao gồm id, tên trong phim, tên thật)

*- crew:* đoàn làm phim (bao gồm id, tên, bộ phận, công việc)

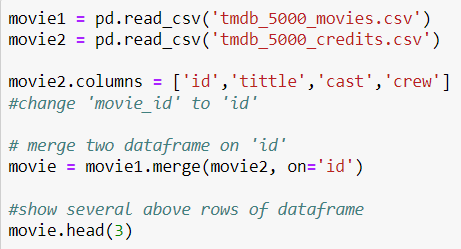
## Khai báo thư viện sử dụng

Một số thư viện được sử dụng như: *numpy, pandas, difflib, sklearn*

## 2.3 Tiền xử lý dữ liệu

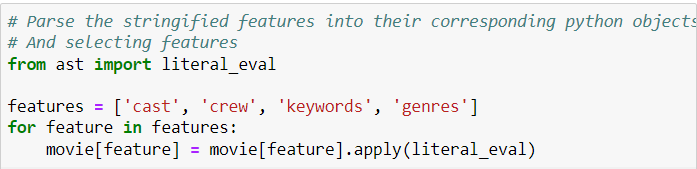
Có thể thấy bộ dữ liệu trên bị phân tách thành 2 file riêng biệt, mỗi file có quá nhiều trường dữ liệu, hơn nữa dữ liệu có trong mỗi trường chưa thể sử dụng ngay được do chúng tồn tại ở nhiều dạng khác nhau. Vì vậy, trước khi áp dụng các thuật toán, ta phải tiền xử lý dữ liệu:

- Ghép 2 file csv: cả hai file dữ liệu mang thông tin của các bộ phim đều có chung trường “id”, vì thế ta sẽ tiến hành ghép hai file này lại dựa trên trường dữ liệu chung này.

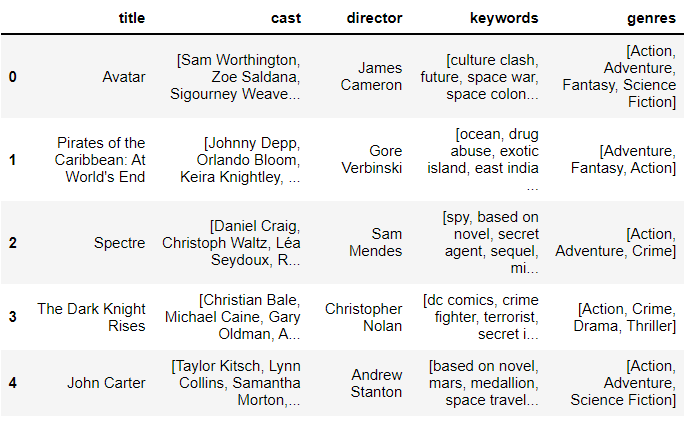


Sau đó, ta thu được một data frame với kích thước 4803x23 với 4803 là số bộ phim và 23 trường dữ liệu.

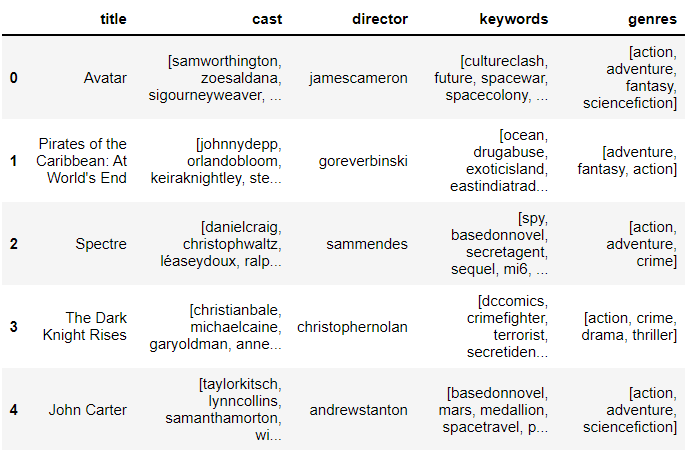
- Do các ô dữ liệu chỉ đang ở dạng string nên ta cần đưa chúng về dạng object tương ứng trong python ( ví dụ xâu “{“id”:1}” sẽ được đưa về dạng dictionary với key = id và value = 1. Tuy nhiên ở đây, ta không sử dụng hết tất cả các trường dữ liệu cho bài toán recommend mà chỉ chọn một số trường có ý nghĩa đối với việc gợi ý như *cast, crew, keyword, genres.* Vì vậy, ta cũng chỉ thực hiện công việc trên đối với các trường dữ liệu này!



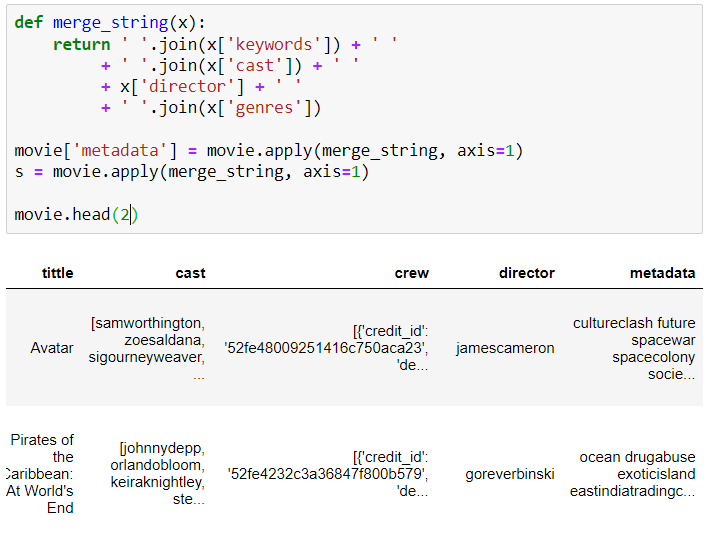
- Tiếp theo, từ trường *crew*, thực hiện trích xuất để lấy ra tên giám đốc sản xuất (Director) vì đây là công việc quan trọng nhất trong đoàn làm phim. Đối với trường *cast* ta trích xuất để lấy ra chỉ tên diễn viên (vì trường này còn chứa các thông tin khác như id, tên nhân vật trong phim). Sau bước này, những thông tin cần thiết sẽ có dạng như sau:



- Tuy nhiên ta cần thêm một bước nữa là chuyển các từ về dạng lowercase và bỏ dấu trắng đối với tên riêng của các diễn viên. Sau đó, dữ liệu sẽ có dạng như sau:



- Bước cuối cùng, gộp tất cả những thông tin của 4 trường *cast, director, keywords, genres* vào thành một xâu duy nhất (trường *metadata*). Điều này sẽ có ích cho việc ta sử dụng các thuật toán ở sau.



## 2.4 Thuật toán

Ta thực hiện mã hóa các xâu thu được ở bước tiền xử lý dữ liệu thành các vector. Hai thuật toán được sử dụng là CountVectorizer và TfidfVectorizer.

a, CountVectorizer

CountVectorizer là một công cụ được cung cấp bởi thư viện scikit-learn bằng Python. Nó được sử dụng để chuyển một văn bản nhất định thành một vector trên cơ sở tần suất (số lượng) của mỗi từ xuất hiện trong toàn bộ văn bản. Điều này rất hữu ích khi có nhiều văn bản như vậy và muốn chuyển đổi từng từ trong mỗi văn bản thành vector để sử dụng trong phân tích văn bản sâu hơn.

Ví dụ ta có một mảng A gồm 2 xâu dữ liệu “this is the first document” và “this document is the second document”. Sau khi sử dụng CountVectorizer ta thu được hai vector tần suất như sau:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | this | is | the | first | document | second |
| A[0] | 1 | 1 | 1 | 1 | 1 | 0 |
| A[1] | 1 | 1 | 1 | 0 | 2 | 1 |

b, TfidfVectorizer

TF-IDF (**term frequency-inverse document frequency) là một phương pháp thống kê để tính toán sự quan trọng của các từ trong một văn bản. TF-IDF đặc biệt hữu dụng trong các thuật toán Machine-learning sử dụng trong Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP).**

**- TF (term frequency): số lần một cụm từ xuất hiện trong một văn bản.**

**- IDF (inverse document frequency): độ đo mức độ phổ biến của một từ trên toàn bộ các văn bản. Lưu ý rằng nó là chung cho tất cả các văn bản, nên nếu một từ xuất hiện ở hầu hết các văn bản thì độ đo này sẽ tiệm cận về 1 (theo công thức dưới).**



**Trong đó, t : từ đang xét**

**n : số lượng văn bản**

**df(t) : số lượng văn bản chứa từ t.**

**Khi đó ta tính được các vector ứng với từng văn bản bao gồm giá trị tại từ t là tf(t) \* idf(t). Kết quả thu được cuối cùng là các vector trực chuẩn.**

**Ví dụ:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **cars** | **cheaper** | **diesel** | **petrol** |
| **d1** | **2** | **1** | **1** | **1** |
| **d2** | **0** | **1** | **1** | **1** |

**Với d1:**

tf-idf(“cars”) = tf(“cars”) x idf (“cars”) = 2 x 1.405465083 => 2.810930165

tf-idf(“cheaper”) = tf(“cheaper”) x idf (“cheaper”) = 1 x 1 => 1

tf-idf(“diesel”) = tf(“diesel”) x idf (“diesel”) = 1×1 => 1

tf-idf(“petrol”) = tf(“petrol”) x idf (“petrol”) = 1×1 => 1

Với d2:

tf-idf(“cars”) = tf(“cars”) x idf (“cars”) = 0 x 1.405465083 => 0

tf-idf(“cheaper”) = tf(“cheaper”) x idf (“cheaper”) = 1 x 1 => 1

tf-idf(“diesel”) = tf(“diesel”) x idf (“diesel”) = 1×1 => 1

tf-idf(“petrol”) = tf(“petrol”) x idf (“petrol”) = 1×1 => 1

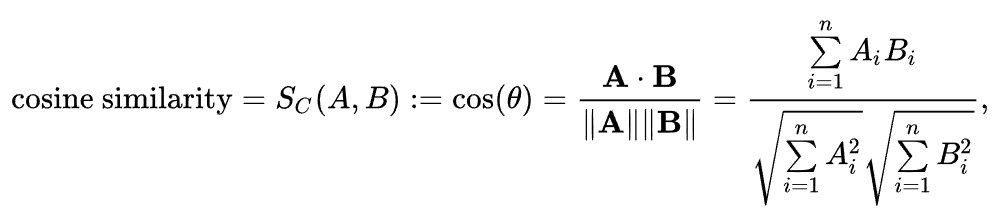
Sau khi trực chuẩn hóa ta thu được các vector như sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | cars | cheaper | diesel | petrol |
| d1 | 0.8513 | 0.3029 | 0.3029 | 0.3029 |
| d2 | 0 | 0.5774 | 0.5774 | 0.5774 |

**c, Cosine similarity**

**Sau khi mã hóa các xâu thành các vector theo hai cách trên, ta cần sử dụng một độ đo để tính toán sự liên quan giữa các vector với nhau. Ví dụ, hai vector có độ đo lớn thể hiện chúng có liên quan đến nhau nhiều hơn, và đây có thể là sự lựa chọn cho việc gợi ý phim.**

**Độ đo được sử dụng là Cosine similarity, được định nghĩa bằng cosine của góc tạo bởi 2 vector, do đó nó bằng tích vô hướng của 2 vector chia cho tích độ dài 2 vector.**

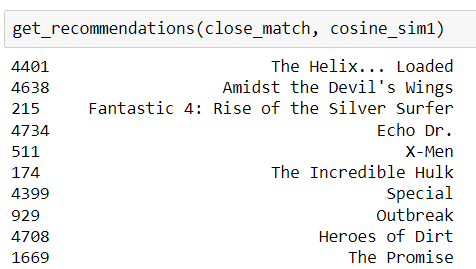


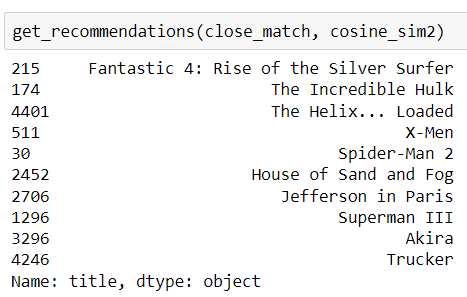
**Do các vector được mã hóa khi sử dụng CountVectorizer hay TfidfVectorizer đều có các thành phần lớn hơn 0 nên cosine similarity cho kết quả chạy từ 0 đến 1. Kết quả càng gần 1 càng cho thấy 2 vector gần nhau, và 2 phim tương ứng với 2 vector này sẽ có ích cho việc gợi ý.**

**Cuối cùng để gợi ý cho một phim X, ta chỉ cần chọn các bộ phim có giá trị cosine similarity với phim X lớn nhất!**

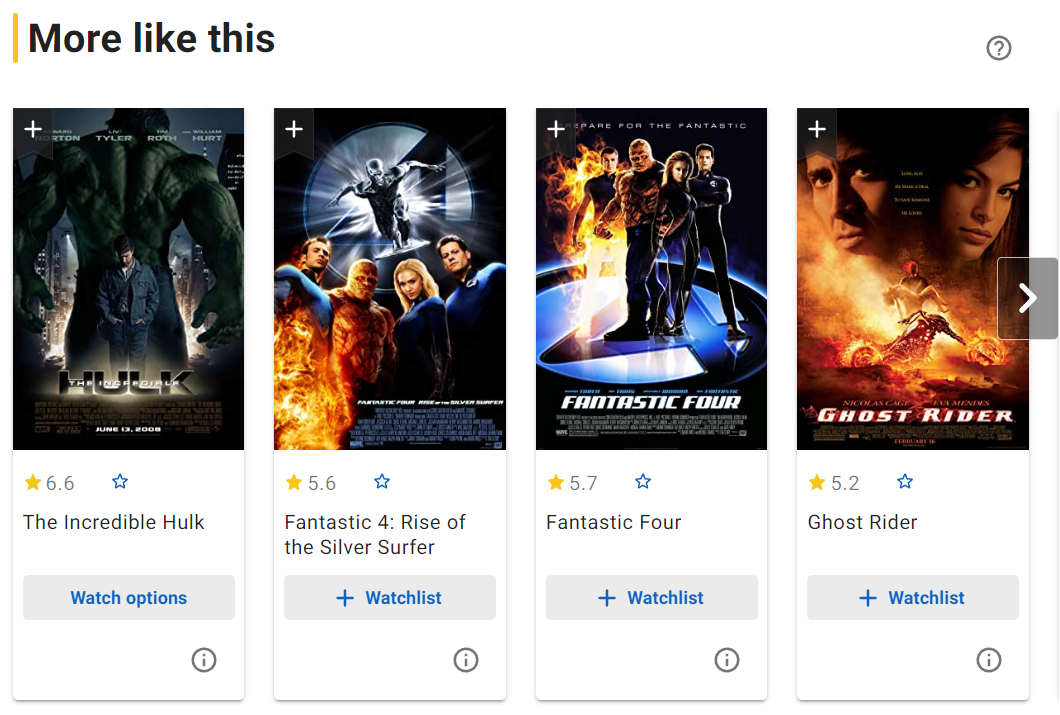
## 2.5 Một số đánh giá và kết luận

Một số gợi ý thu được như sau: (lưu ý *close\_match* là tên phim cần gợi ý - Hulk, được lấy từ input của người dùng, *cosine\_sim1* và *cosine\_sim2* lần lượt ứng với ma trận cosine similarity của CountVectorizer và TfidfVectorizer.





Thực nghiệm trên trang web [IMDb: Ratings, Reviews, and Where to Watch the Best Movies & TV Shows](https://www.imdb.com/?ref_=nv_home) với bộ phim Hulk, web tự động đề xuất các bộ phim khá giống với việc sử dụng các thuật toán ở trên.



Thực nghiệm đối với một số bộ phim khác cũng cho kết quả các bộ phim được gợi ý trùng khá nhiều với việc sử dụng các thuật toán trên. Tuy nhiên, để đánh giá chính xác độ tốt của thuật toán, ta cần phải có dữ liệu của người dùng về các thao tác người dùng chọn xem các bộ phim được gợi ý. Loại dữ liệu này tương ứng với *[TMDB 5000 Movie Dataset](https://www.kaggle.com/code/ibtesama/getting-started-with-a-movie-recommendation-system/data)* nhóm chúng em vẫn chưa thu thập được, vì vậy chưa thể có đánh giá một cách chính xác nhất cho thuật toán mà chỉ dựa vào sự tương quan với gợi ý của website nói trên.

# Tiến hành trên bộ dữ liệu MovieLens

# Kết luận

# Tài liệu tham khảo

[TMDB 5000 Movie Dataset | Kaggle](https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata)

[(483) Project 18. Movie Recommendation System using Machine Learning with Python - YouTube](https://www.youtube.com/watch?v=7rEagFH9tQg&ab_channel=Siddhardhan)

[Getting Started with a Movie Recommendation System | Kaggle](https://www.kaggle.com/code/ibtesama/getting-started-with-a-movie-recommendation-system/data)

[Machine Learning cơ bản (machinelearningcoban.com)](https://machinelearningcoban.com/2017/05/17/contentbasedrecommendersys/)

[Cosine similarity - Wikipedia](https://en.wikipedia.org/wiki/Cosine_similarity)

[Understanding TF-ID: A Simple Introduction (monkeylearn.com)](https://monkeylearn.com/blog/what-is-tf-idf/" \l ":~:text=TF-IDF (term frequency-,across a set of documents.)

[How sklearn’s Tfidfvectorizer Calculates tf-idf Values - Analytics Vidhya](https://www.analyticsvidhya.com/blog/2021/11/how-sklearns-tfidfvectorizer-calculates-tf-idf-values/)

[Using CountVectorizer to Extracting Features from Text - GeeksforGeeks](https://www.geeksforgeeks.org/using-countvectorizer-to-extracting-features-from-text/)