



# **BÁO CÁO PROJECT**

**GR1**

# HỆ THỐNG GỢI Ý PHIM

---



# CONTENT

- 01** GIỚI THIỆU
- 02** CÔNG NGHỆ
- 03** DỮ LIỆU
- 04** MÔ HÌNH
- 05** TRIỂN KHAI
- 06** TỔNG KẾT

# GIỚI THIỆU

## ĐẶT VĂN ĐỀ:

Trong thế giới phim ảnh đa dạng ngày nay, việc chọn bộ phim phù hợp có thể khó khăn. Hệ thống gợi ý phim hiệu quả giúp cá nhân hóa trải nghiệm xem phim, mang lại niềm vui tối đa cho người dùng.



# GIỚI THIỆU

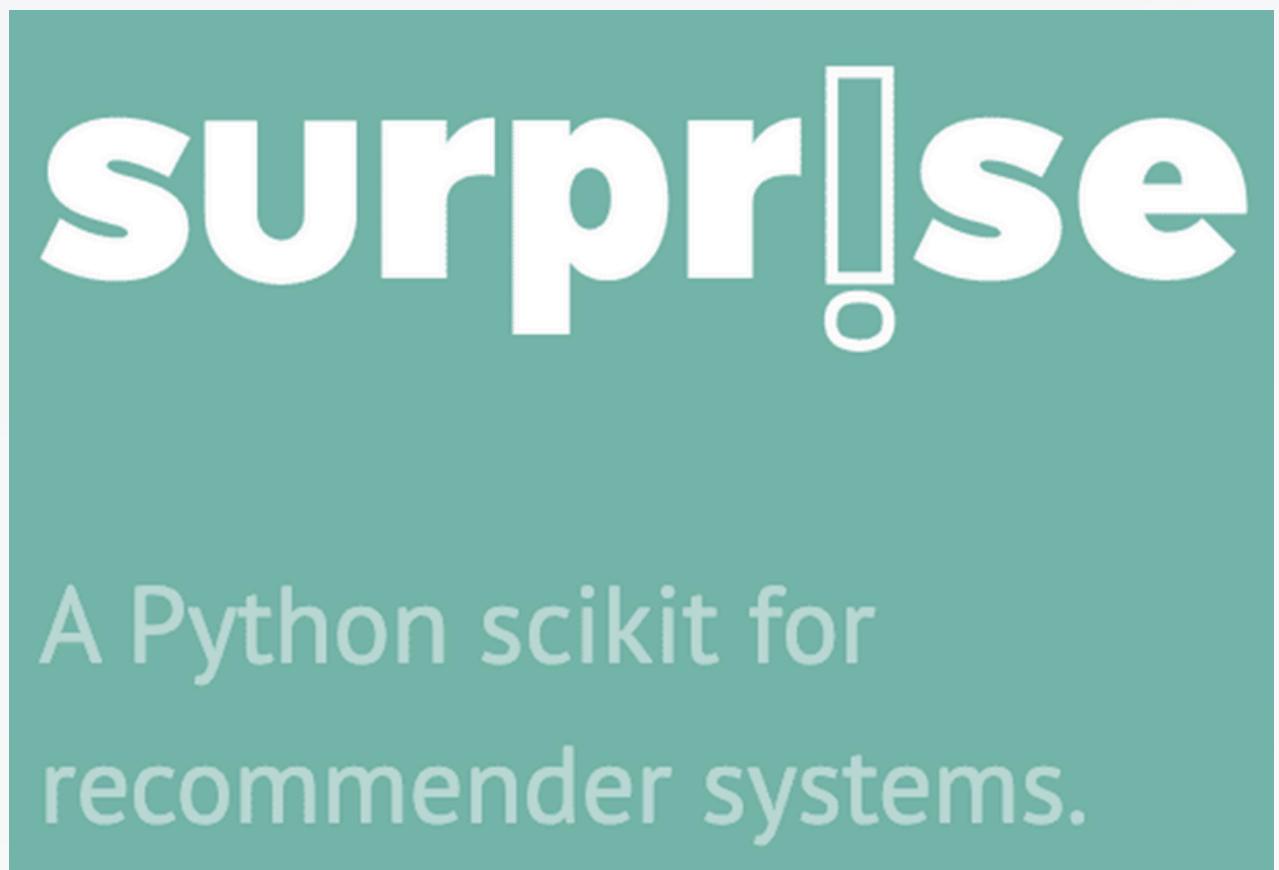
## MỤC TIÊU CỦA DỰ ÁN:

- Tiết kiệm thời gian cho người dùng để tìm ra một bộ phim phù hợp.
- Cá nhân hóa trải nghiệm: đề xuất các bộ phim phù hợp với sở thích và lịch sử xem phim của người dùng.
- Tăng cường khám phá: Khuyến nghị những bộ phim mới và đa dạng để mở rộng tầm nhìn về nội dung phim.



# CÔNG NGHỆ

- Ngôn ngữ lập trình: Python
- Thư viện hỗ trợ thuật toán: Surprise
- Thu thập và xử lý dữ liệu: BeautifulSoup, Pandas



# DỮ LIỆU

- Nguồn thu thập: IMDB
- Dữ liệu được xử lý để thuận tiện sử dụng

# DỮ LIỆU

IMDb Menu All Search IMDb IMDbPro Watchlist Sign In EN

 **The Godfather** (1972)  
User Reviews

[+ Review this title](#)

5,622 Reviews  Hide Spoilers Filter by Rating: Show All Sort by: Featured

**★ 10/10**

**An offer so good, I couldn't refuse**  
andrewburgereviews 1 April 2019

It is now past 1 PM and I just finished watching Francis Ford Coppola's "The Godfather". I should probably go to bed. It's late and tomorrow I have to wake up a bit early. But not early enough to postpone writing these lines. Now that I have seen it three times, the opportunity of sharing my thoughts and refreshed insights are too much of a good offer to sit on. So, bear with me.

This film works so well because it takes place in an underworld in which we are so embedded that we do not even observe it. Coppola puts us straight in the smack-dab center of what is, admittedly, a society made by criminals for criminals. It is also the reason why it's so welcoming. We are surrounded by its inhabitants--cold-blooded

**The Godfather**

Opinion  
Awards  
FAQ  
User Reviews  
User Ratings  
External Reviews  
Metacritic Reviews

[Explore More](#)

**User Lists** [Create a list »](#)

Related lists from IMDb users

 **Watched**  
a list of 46 titles created 17 Oct 2022

 **2024'te izlediklerim**  
a list of 29 titles created 5 months ago

# DỮ LIỆU

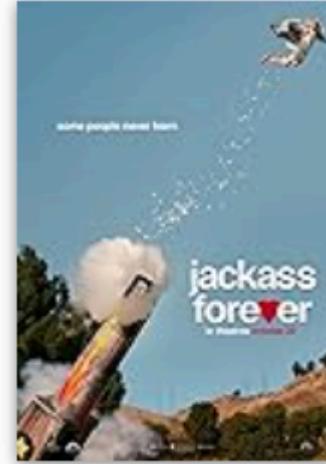
IMDb ≡ Menu All Search IMDb 🔍 IMDbPro Watchlist Sign In EN EN

 **andrewburgereviews**  
IMDb member since March 2018

 Lifetime Total **75+**  IMDb Member **6 years**

### Ratings

Most Recently Rated



Doctor Strange i... ★ 5    Jackass Forever ★ 9    The Power of th... ★ 9    The Batman ★ 8    Don't Look Up ★ 4

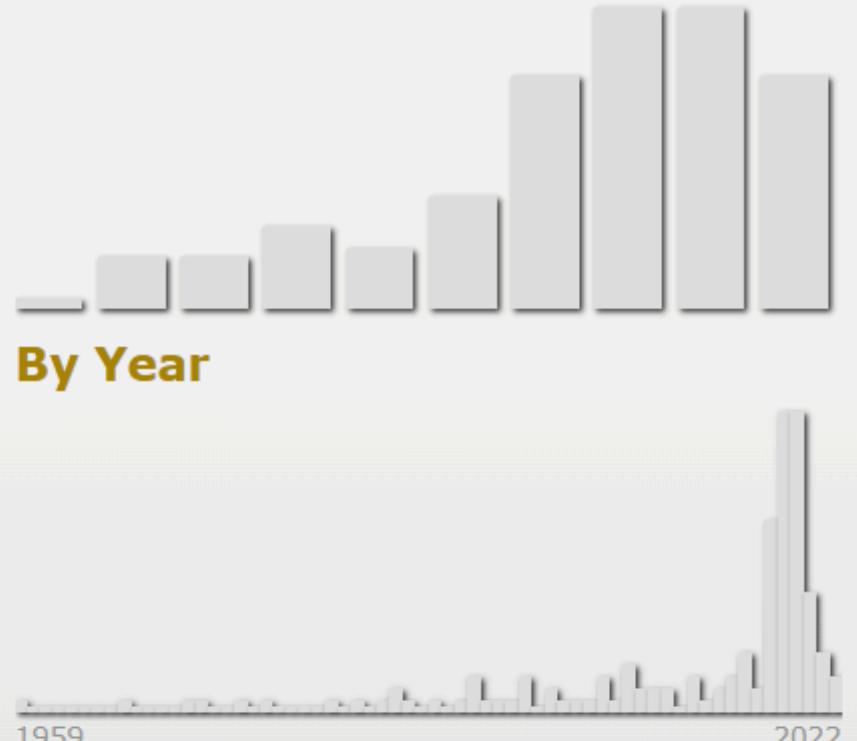
[See all 142 ratings »](#)

### Quick Links

[Ratings](#) [Lists](#)  
[Watchlist](#) [Checkins](#)  
[Reviews](#) [Poll Responses](#)  
[About this Page](#)

### Ratings Analysis

### Rating Distribution



By Year

1959 2022

[See more ▾](#)

# DỮ LIỆU

```
9
10<!DOCTYPE html>
11<html
12  xmlns:og="http://ogp.me/ns#"
13  xmlns:fb="http://www.facebook.com/2008/fbml">
14<head>
15
16<script type='text/javascript'>var ue_t0=ue_t0||new Date();</script>
17<script type='text/javascript'>
18window.ue_ihb = (window.ue_ihb || window.ueinit || 0) + 1;
19if (window.ue_ihb === 1) {
20
21var ue_csm = window,
22    ue_hob = +new Date();
23(function(d){var e=d.ue||[],f=Date.now||function(){return+new Date();};e.d=function(b){return f()-(b?0:d.ue_t0)};e.stub=function(b,a){if(!b[a]){var c=[];b[a]=function(){c.push([].slice.call(arguments),e.d(),d.ue_id]);}b[a].replay=function(b){for(var a;a=c.shift();)b(a[0],a[1],a[2]);}b[a].isStub=1}};e.exec=function(b,a){return function(){try{return b.apply(this,arguments)}catch(c){ue.LogError(c,{attribution:a||"undefined",logLevel:"WARN"})}}}(ue_csm);
24
25
26    var ue_err_chan = 'jserr';
27(function(d,e){function h(f,b){if(!(a.ec>a.mxe)&&f){a.ter.push(f);b=b||[];var c=f.logLevel||b.logLevel;c&&c!==k&&c!==m&&c!==n&&c!==p||a.ec++&&c!==k||a.ecf++&&b.pageURL+=""+(e.location?e.location.href:"");b.logLevel=c;b.attribution=f.attribution||b.attribution;a.erl.push({ex:f.info:b})}function l(a,b,c,e,g){d.LogError({m:a,f:b,l:c,c:""+e,err:g,fromOnError:1,args:arguments},g?{attribution:g.attribution,logLevel:g.logLevel}:void 0);return!1}var k="FATAL",m="ERROR",n="WARN",p="DOWNGRADED",a={ec:0,ecf:0,pec:0,ts:0,erl:[],ter:[],buffer:[],mxe:50,startTimer:function(){a.ts++;setInterval(function(){d.ue&&a.pec<a.ec&&d.uex("at");a.pec=a.ec},1E4)}},l.skipTrace=1;h.skipTrace=1;h.isStub=1;d.LogError=h;d.ue_err=a;e.onerror=1})(ue_csm>window);
28
29
30
31var ue_id = '384MKRMKCNX7MRTH85PE',
32    ue_url,
33    ue_navtiming = 1,
34    ue_mid = 'A1EVAM02EL8SFB',
35    ue_sid = '138-1958584-1820303',
36    ue_sn = 'www.imdb.com',
37    ue_furl = 'fls-na.amazon.com',
38    ue_surl = 'https://unagi-na.amazon.com/1/events/com.amazon.csm.nexusclient.prod',
39    ue_int = 0,
40    ue_fcsn = 1,
```

# DỮ LIỆU

- overall data:

Movie ID	Title	Year	Genre	User Rating	Director	Stars	User ID
tt9419884	Doctor Strange in the Multiverse of Madness	2022	Action, Adventure	5	Sam Raimi	Benedict Cumberbatch, Elizabeth Olsen, Chiwetel Ejiofor	ur86182727
tt1146622	Jackass Presents: Bad Grandpa On Vacation	2022	Documentary	9	Jeff Tremaine	Johnny Knoxville, Jeff Tremaine, Jennifer Schwalbach Smith	ur86182727
tt10293406	The Power of the Dog	2021	Drama, Western	9	Jane Campion	Benedict Cumberbatch, Frances McDormand, Kodi Smit-McPhee	ur86182727
tt1877830	The Batman	2022	Action, Crime	8	Matt Reeves	Robert Pattinson, Zoe Kravitz, Jacob Lofland	ur86182727
tt11286314	Don't Look Up	2021	Comedy, Drama	4	Adam McKay	Leonardo DiCaprio, Jennifer Lawrence, Meryl Streep	ur86182727

# DỮ LIỆU

movies.csv

movield	title	genres
1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy
2	Jumanji (1995)	Adventure Children Fantasy
3	Grumpier Old Men	Comedy Romance
4	Waiting to Explode	Comedy Drama Romance
5	Father of the Bride	Comedy
6	Heat (1995)	Action Crime Thriller
7	Sabrina (1995)	Comedy Romance
8	Tom and Huck	Adventure Children
9	Sudden Death	Action
10	GoldenEye (1995)	Action Adventure Thriller
11	American Pie	Comedy Drama Romance
12	Dracula: Dead and Loving It	Comedy Horror
13	Balto (1995)	Adventure Animation Children
14	Nixon (1995)	Drama
15	Cutthroat Island	Action Adventure Romance

# DỮ LIỆU

rating.csv

userId	movieId	rating	timestamp
1	31	2.5	1.26E+09
1	1029	3	1.26E+09
1	1061	3	1.26E+09
1	1129	2	1.26E+09
1	1172	4	1.26E+09
1	1263	2	1.26E+09
1	1287	2	1.26E+09
1	1293	2	1.26E+09
1	1339	3.5	1.26E+09
1	1343	2	1.26E+09
1	1371	2.5	1.26E+09
1	1405	1	1.26E+09
1	1953	4	1.26E+09
1	2105	4	1.26E+09
1	2150	3	1.26E+09
1	2193	2	1.26E+09
1	2294	2	1.26E+09
1	2455	2.5	1.26E+09
1	2968	1	1.26E+09

# CÁC THÔNG SỐ ĐÁNH GIÁ MÔ HÌNH GỢI Ý

- MAE, RMSE: chính xác
- Hit Rate, cHR, rHR: thực tế
- Coverage: độ phổ cập
- Diversity: độ đa dạng
- Novelty: độ mới mẻ
- Churn: độ thay đổi ngẫu nhiên
- Responsiveness: tốc độ thích ứng với thay đổi

# CÁC THÔNG SỐ ĐÁNH GIÁ MÔ HÌNH GỢI Ý

MAE, RMSE

$$\frac{\sum_{i=1}^n |y_i - x_i|}{n}$$

$$\sqrt{\frac{\sum_{i=1}^n (y_i - x_i)^2}{n}}$$

# CÁC THÔNG SỐ ĐÁNH GIÁ MÔ HÌNH GỢI Ý

Hit Rate: Tạo danh sách gợi ý Top N cho tất cả các user, nếu như gợi ý được bộ phim mà người dùng đó đã đánh giá/thích thì sẽ được tính là 1 hit. Hit Rate = hits/users

Leave-One-Out Cross-Validation (LOOCV): Chia tập luyện và tập kiểm tra, trong tập luyện bỏ đi ngẫu nhiên 1 phim từ lịch sử của mỗi người dùng.

```
LOOCV = LeaveOneOut(n_splits=1, random_state=1)
for train, test in LOOCV.split(data):
    self.LOOCVTrain = train
    self.LOOCVTest = test

    self.LOOCVAntiTestSet = self.LOOCVTrain.build_anti_testset()
```

# CÁC THÔNG SỐ ĐÁNH GIÁ MÔ HÌNH GỢI Ý

AHRH (Average Reciprocal Hit Rate): biến thể của Hit Rate, tính đến vị trí của mục được trúng trong danh sách gợi ý. Nó đánh trọng số cao hơn cho các mục xuất hiện ở các vị trí đầu. ARHR càng cao càng tốt.

$$\frac{\sum_{i=1}^n \frac{1}{rank_i}}{Users}$$

# CÁC THÔNG SỐ ĐÁNH GIÁ MÔ HÌNH GỌI Ý

cHR (Cumulative Hit Rate): tổng hợp của các Hit Rate qua các danh sách gợi ý khác nhau. Nó đánh giá khả năng của hệ thống gợi ý trên toàn bộ tập hợp các danh sách gợi ý.

rHR (Rating Hit Rate): đo lường tỷ lệ các gợi ý có xếp hạng đúng với mong đợi của người dùng. Nó đánh giá độ chính xác của các xếp hạng trong hệ thống gợi ý.

# CÁC THÔNG SỐ ĐÁNH GIÁ MÔ HÌNH GỢI Ý

**Coverage:** tỷ lệ các mục được hệ thống gợi ý đến tổng số mục trong tập dữ liệu. Nó đánh giá phạm vi bao phủ của hệ thống gợi ý.

**Diversity:** đánh giá mức độ khác nhau giữa các mục trong danh sách gợi ý. Nó quan trọng để tránh gợi ý các mục quá giống nhau.

**Novelty:** đánh giá mức độ mới mẻ của các mục trong danh sách gợi ý, tức là hệ thống gợi ý các mục mà người dùng chưa từng thấy hoặc ít tương tác.

# CÁC THÔNG SỐ ĐÁNH GIÁ MÔ HÌNH GỢI Ý

Algorithm	RMSE	MAE	HR	cHR	ARHR	Coverage	Diversity	Novelty
SVD	0.9034	0.6978	0.0298	0.0298	0.0112	0.9553	0.0445	491.5768
Random	1.4385	1.1478	0.0089	0.0089	0.0015	1.0000	0.0719	557.8365

SVD: Thuật toán có sẵn trong thư viện Surprise

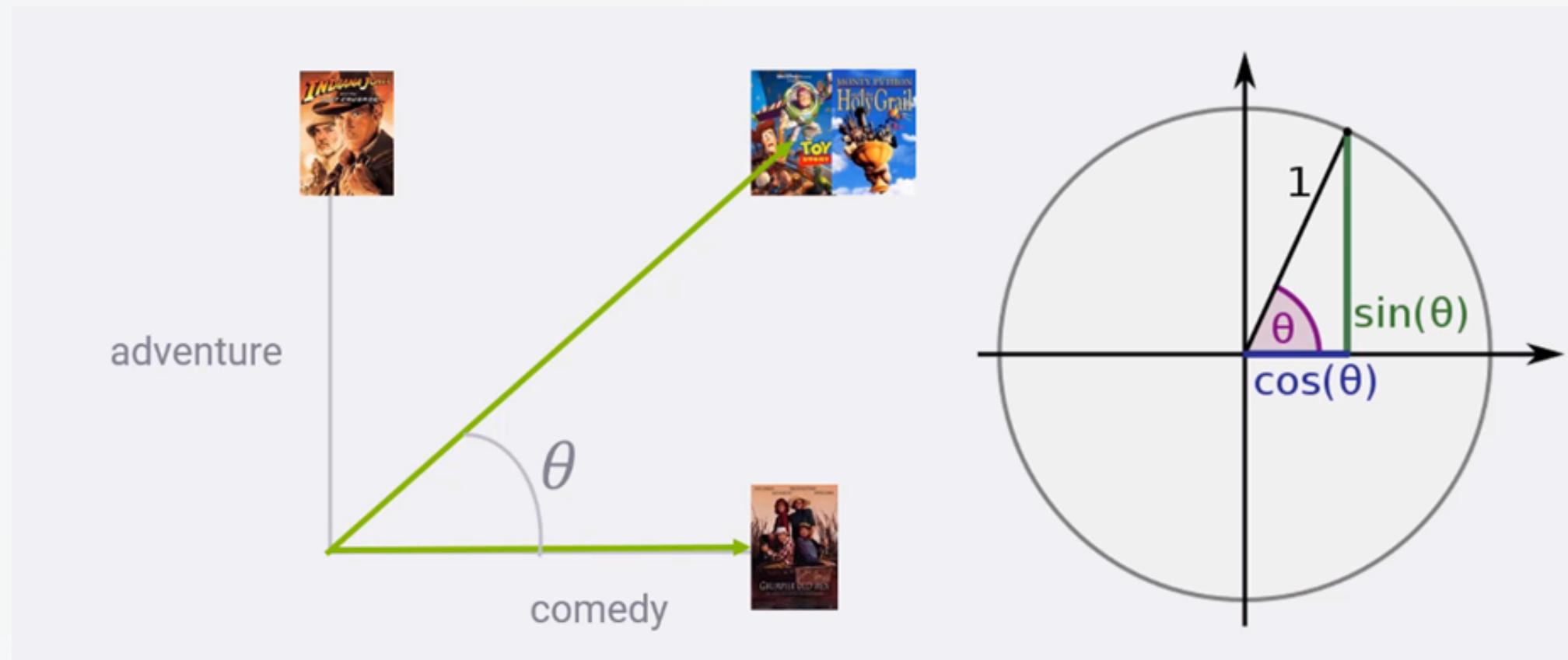
Random: Gợi ý ngẫu nhiên các bộ phim

# PHƯƠNG PHÁP GỌI Ý CONTENT-BASED FILTERING

**Biểu diễn nội dung của các bộ phim:** Mỗi phim có 18 thể loại (genres) có thể thuộc vào, mỗi thể loại sẽ là 1 chiều của vector đại diện cho 1 bộ phim

Biểu diễn người dùng: Mỗi người dùng được biểu diễn bằng một hồ sơ (profile) của các bộ phim họ đã tương tác trước đó.

Tính toán độ tương tự theo thể loại: Độ tương tự theo thể loại giữa các mặt hàng được tính bằng công thức cosine similarity (tính góc giữa 2 vector)



$$CosSim(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}}$$

## Công thức tính độ tương tự theo thể loại:

$$CosSim(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}}$$

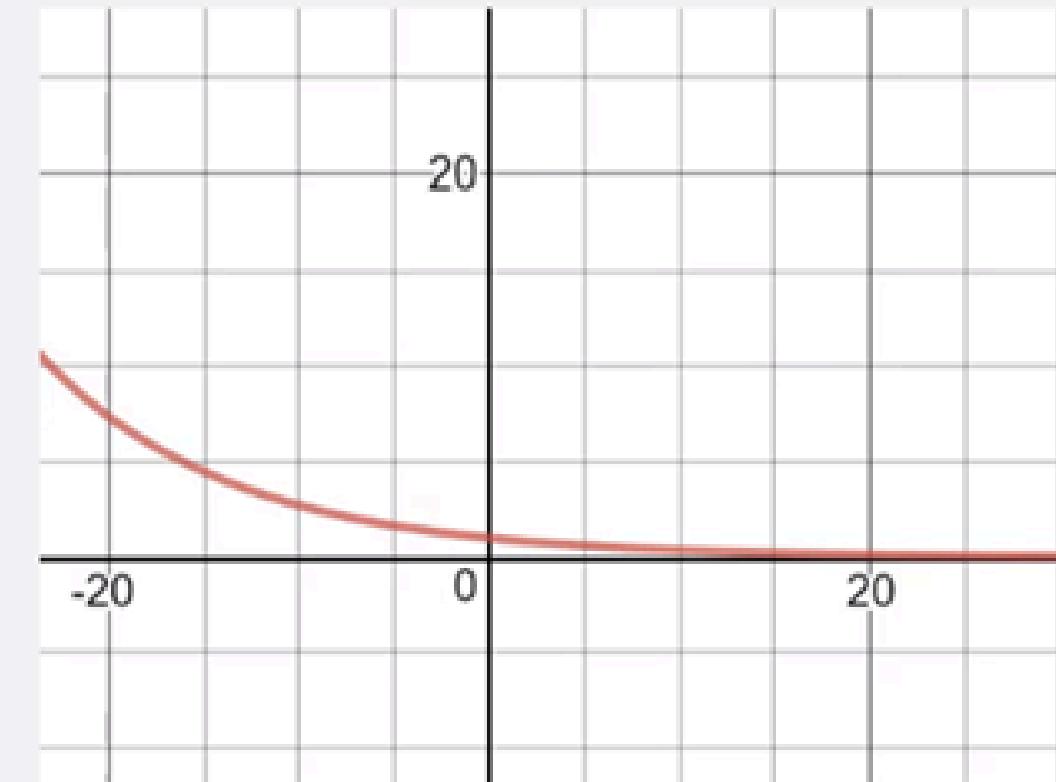


```
def computeGenreSimilarity(self, movie1, movie2, genres):
    genres1 = genres[movie1]
    genres2 = genres[movie2]
    sumxx, sumxy, sumyy = 0, 0, 0
    for i in range(len(genres1)):
        x = genres1[i]
        y = genres2[i]
        sumxx += x * x
        sumyy += y * y
        sumxy += x * y

    return sumxy/math.sqrt(sumxx*sumyy)
```

# Tính toán trọng số cho năm phát hành:

```
def computeYearSimilarity(self, movie1, movie2, years):  
    diff = abs(years[movie1] - years[movie2])  
    sim = math.exp(-diff / 10.0)  
    return sim
```



# Điểm tương đồng được tính theo công thức

```
self.similarities[thisRating, otherRating] = genreSimilarity * yearSimilarity
```

Chọn k hàng xóm gần nhất với một bộ phim đang được xét và đánh giá dự đoán sẽ là trung bình đánh giá của k bộ phim đó.

```
def estimate(self, u, i):

    if not (self.trainset.knows_user(u) and self.trainset.knows_item(i)):
        raise PredictionImpossible('User and/or item is unknown.')

    # Build up similarity scores between this item and everything the user rated
    neighbors = []
    for rating in self.trainset.ur[u]:
        genreSimilarity = self.similarities[i, rating[0]]
        neighbors.append( (genreSimilarity, rating[1]) )

    # Extract the top-K most-similar ratings
    k_neighbors = heapq.nlargest(self.k, neighbors, key=lambda t: t[0])

    # Compute average sim score of K neighbors weighted by user ratings
    simTotal = weightedSum = 0
    for (simScore, rating) in k_neighbors:
        if (simScore > 0):
            simTotal += simScore
            weightedSum += simScore * rating

    if (simTotal == 0):
        raise PredictionImpossible('No neighbors')

    predictedRating = weightedSum / simTotal

    return predictedRating
```

# DEMO

We recommend:

Presidio, The (1988) 3.841314676872932

Femme Nikita, La (Nikita) (1990) 3.839613347087336

Wyatt Earp (1994) 3.8125061475551796

Shooter, The (1997) 3.8125061475551796

Bad Girls (1994) 3.8125061475551796

The Hateful Eight (2015) 3.812506147555179

True Grit (2010) 3.812506147555179

Open Range (2003) 3.812506147555179

Big Easy, The (1987) 3.7835412549266985

Point Break (1991) 3.764158410102279

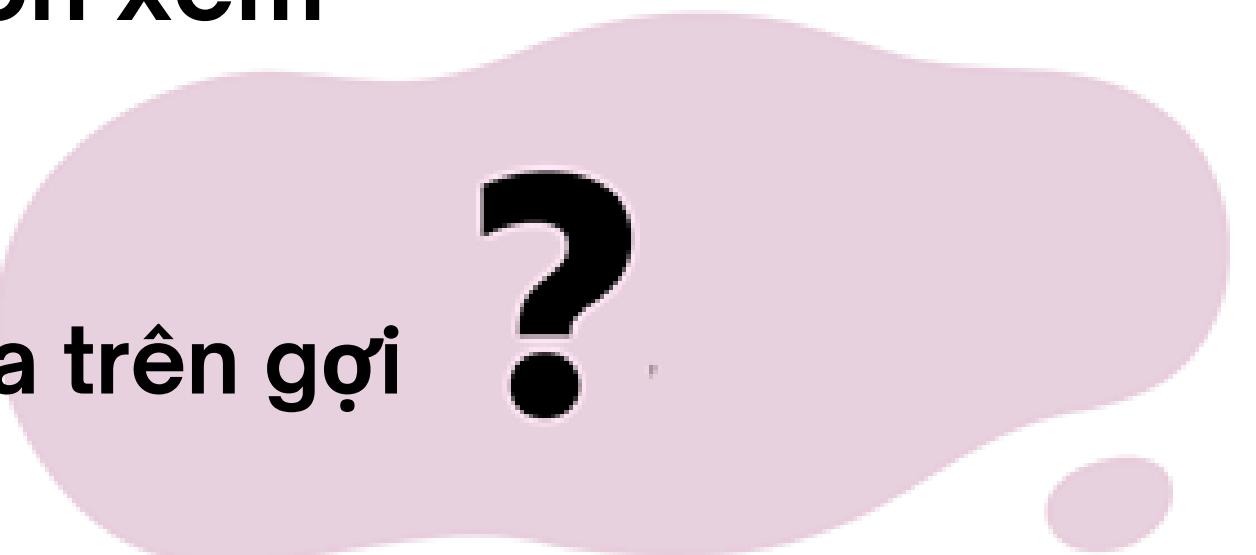
# Đánh giá phương pháp theo chỉ số đánh giá

Algorithm	RMSE	MAE	HR	CHR	ARHR	Coverage	Diversity	Novelty
ContentKNN	0.9375	0.7263	0.0030	0.0030	0.0017	0.9285	0.5700	4567.1964
Random	1.4385	1.1478	0.0089	0.0089	0.0015	1.0000	0.0719	557.8365

# COLLABORATIVE FILTERING – BÍ MẬT ĐẰNG SAU NHỮNG GỢI Ý "THẦN KỲ"

Các bạn đã bao giờ tự hỏi tại sao Netflix luôn biết bộ phim tiếp theo bạn muốn xem là gì?

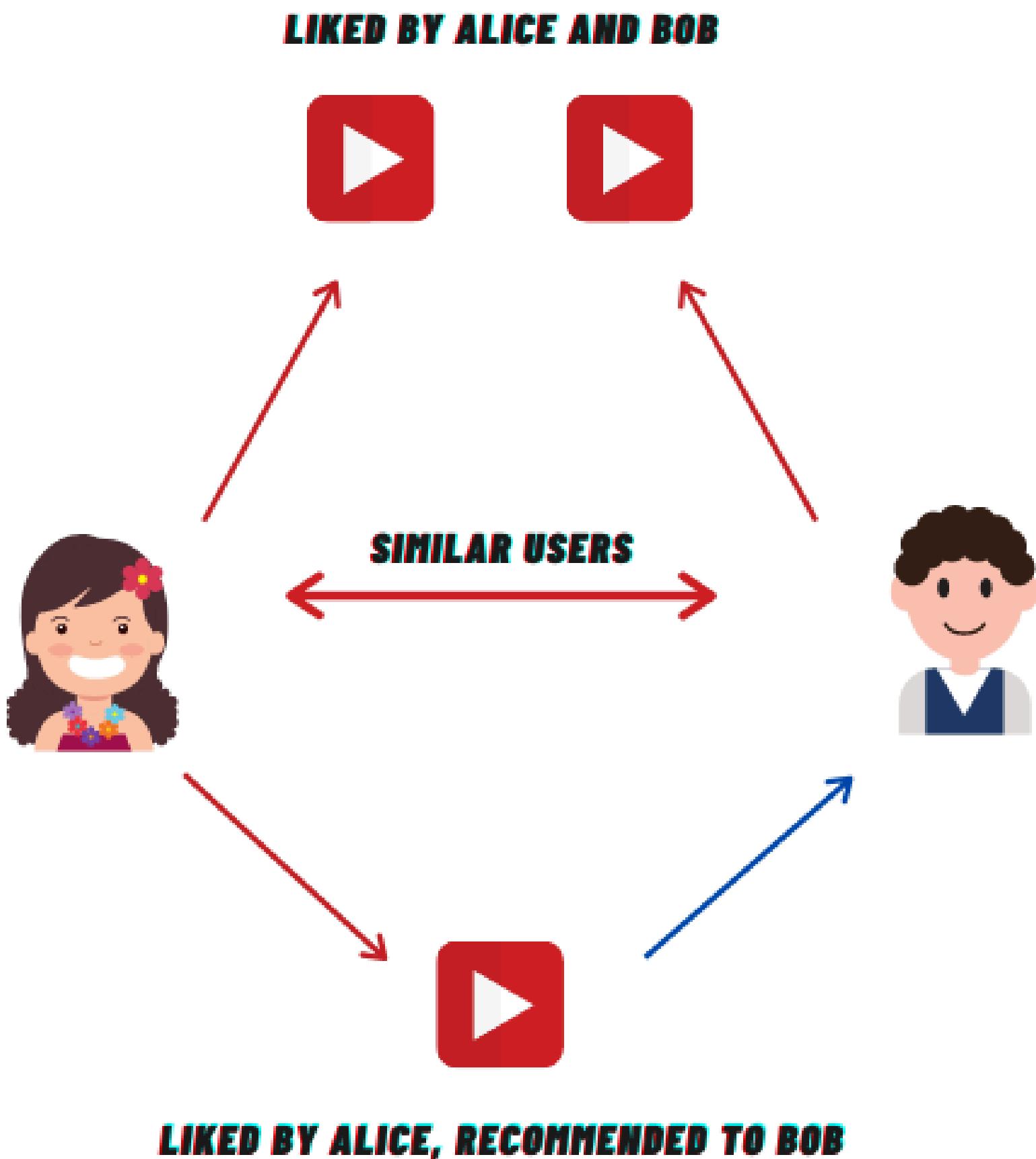
? 75% người dùng Netflix chọn phim dựa trên gợi ý của hệ thống. (\*)



➤ **Collaborative Filtering (Lọc Cộng Tác).**

(\*) Source: <https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers>

# COLLABORATIVE FILTERING



## Input:

- Ma trận đánh giá (rating matrix)
  - Dựa vào độ tương đồng giữa các người dùng
  - Sử dụng độ tương đồng Cosine

## Output:

- Top N bộ phim được đề xuất

amazon



# Ma trận dữ liệu:

	Indiana Jones	Star Wars	Empire Strikes Back	Incredibles	Casablanca
Bob	4	5			
Ted					1
Ann		5	5	5	

=> Chuyển các user thành các vector với chiều là những bộ phim, trọng số là đánh giá (VD: Bob (4, 5, 0, 0, 0))

=> Tương đồng qua công thức cosin:

# Khoảng cách Cosine: Đo góc giữa hai vectơ đánh giá của hai người dùng.

- $x$  là vectơ đại diện cho người đang xét
- $y$  là vectơ đại diện cho người tương đồng

$$CosSim(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}}$$

```
sim_options = { 'name': 'cosine',
                 'user_based': True
               }

model = KNNBasic(sim_options=sim_options)
model.fit(trainSet)
simsMatrix = model.compute_similarities()
```

# Ma trận tương đồng:

```
Done computing similarity matrix.  
[[1.          0.          0.          ... 1.          0.          1.          ]  
 [0.          1.          0.95561425 ... 0.776114   0.89144204 0.97993672]  
 [0.          0.95561425 1.          ... 0.99786069 0.94592126 0.98448284]  
 ...  
 [1.          0.776114   0.99786069 ... 1.          1.          0.9952275 ]  
 [0.          0.89144204 0.94592126 ... 1.          1.          0.96183401]  
 [1.          0.97993672 0.98448284 ... 0.9952275  0.96183401 1.          ]]
```

# Tìm người dùng tương đồng với người dùng đang được xét:

- **Lấy ID nội bộ của người dùng cần gợi ý (testUserInnerID).**
- **Lọc ra các người dùng tương tự có độ tương đồng trên 0.95 với người dùng cần gợi ý, lưu vào danh sách kNeighbors.**

```
# Get top N similar users to our test subject
# (Alternate approach would be to select users up to some
testUserInnerID = trainSet.to_inner_uid(testSubject)
similarityRow = simsMatrix[testUserInnerID]

similarUsers = []
for innerID, score in enumerate(similarityRow):
    if (innerID != testUserInnerID):
        similarUsers.append( (innerID, score) )

#kNeighbors = heapq.nlargest(k, similarUsers, key=lambda t
kNeighbors = []
for rating in similarUsers:
    if rating[1] > 0.95:
        kNeighbors.append(rating)
```

```
Done computing similarity matrix
[(1, 0.9748348733519079), (2, 0.9893261506452822),
 (3, 0.9897493266497985), (4, 0.9765338130686765),
 (6, 0.9967558386655043), (7, 0.9744948321), (13, 0.9938837346),
 (8, 0.9897493266497985), (9, 0.96108376),
 (10, 0.9897493266497985), (11, 0.9967558386655043),
 (12, 0.9744948321), (14, 0.9938837346),
 (15, 0.9897493266497985), (16, 0.96108376),
 (17, 0.9776747465435052), (20, 0.9967558386655043),
 (21, 0.9746178883551764), (22, 0.9897493266497985)]
```

```
# Get the stuff they rated, and add up ratings for each item, weighted by
candidates = defaultdict(float)
for similarUser in kNeighbors:
    innerID = similarUser[0]
    userSimilarityScore = similarUser[1]
    theirRatings = trainSet.ur[innerID]
    for rating in theirRatings:
        candidates[rating[0]] += (rating[1] / 5.0) * userSimilarityScore
```

Xây dựng từ điển **candidates** là các bộ phim mà k người dùng đã đánh giá:  
**candidates**: Một từ điển lưu trữ tổng trọng số đánh giá của mỗi bộ phim từ  
các người dùng tương tự.

- Với mỗi người dùng tương tự, duyệt qua các bộ phim họ đã đánh giá:
- Lấy ID bộ phim và điểm đánh giá.
- Chia điểm đánh giá cho 5 (chuẩn hóa về khoảng 0-1).
- Nhân điểm đánh giá đã chuẩn hóa với độ tương đồng giữa người dùng  
mục tiêu và người dùng tương tự.
- Cộng giá trị này vào tổng trọng số đánh giá của bộ phim trong từ điển  
**candidates**.

- Sắp xếp từ điển candidates theo thứ tự giảm dần của tổng trọng số đánh giá.
- Dừng khi đã gợi ý 10 bộ phim.



```
In [6]: runfile('C:/Users/Admin/Desktop/HUST/GR1/CollaborativeFiltering.ipynb', wdir='C:/Users/Admin/Desktop/HUST/GR1/CollaborativeFiltering')
Reloaded modules: MovieLens
Computing the cosine similarity matrix...
Done computing similarity matrix.
Computing the cosine similarity matrix...
Done computing similarity matrix.
Shawshank Redemption, The (1994) 229.96836194898867
Silence of the Lambs, The (1991) 205.43102781808292
Schindler's List (1993) 176.08290718596766
Jurassic Park (1993) 170.4933268092533
Fargo (1996) 155.93550957059838
Back to the Future (1985) 154.04871891085054
Usual Suspects, The (1995) 147.8810875119617
Godfather, The (1972) 145.92718848855256
Fugitive, The (1993) 139.1874367528664
Seven (a.k.a. Se7en) (1995) 136.39157573589637
Independence Day (a.k.a. ID4) (1996) 132.79149035042283
```

# Đánh giá Hit Rate

```
In [10]: runfile('C:/Users/Admin/Desktop/HUST/GR1/  
CollaborativeFiltering/EvaluateUserCF.py', wdir='C:/Users/Admin/  
Desktop/HUST/GR1/CollaborativeFiltering')  
Reloaded modules: MovieLens, RecommenderMetrics, EvaluationData  
Loading movie ratings...  
  
Computing movie popularity ranks so we can measure novelty later...  
Estimating biases using als...  
Computing the cosine similarity matrix...  
Done computing similarity matrix.  
Computing the cosine similarity matrix...  
Done computing similarity matrix.  
Computing the cosine similarity matrix...  
Done computing similarity matrix.  
HR 0.05514157973174367
```

# PHƯƠNG PHÁP GỢI Ý ITEM-BASED COLLABORATIVE

Item-Based Collaborative Filtering gợi ý các item cho người dùng dựa trên các item tương tự. Quá trình này bao gồm các bước chính:

- Xây dựng ma trận tương đồng
- Chọn sản phẩm yêu thích của người dùng
- Tìm các sản phẩm tương tự
- Loại bỏ các sản phẩm đã xem
- Đưa ra gợi ý

# Ma trận dữ liệu

	Bob	Ted	Ann
Indiana Jones	4		
Star Wars	5		5
Empire Strikes Back			5
Incredibles			5
Casablanca		1	

Chuyển các bộ phim thành vector với chiều là những người dùng, trọng số là đánh giá (VD: Indiana Jones (4, 0, 0))  
=> Tương đồng qua công thức cosin

# Ma trận tương đồng

Sử dụng thang điểm 1

Độ tương đồng tính dựa  
trên cosin bằng KNN

	$i_0$	$i_1$	$i_2$	$i_3$	$i_4$
$i_0$	1	0.55	0.38	0.39	0.38
$i_1$	0.55	1	0.92	0.45	0.42
$i_2$	0.38	0.92	1	0.37	0.24
$i_3$	0.39	0.45	0.37	1	0.33
$i_4$	0.38	0.42	0.24	0.33	1

# Công thức tính độ tương đồng

$$CosSim(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}}$$

```
sim_options = {'name': 'cosine',
               'user_based': False
               }

model = KNNBasic(sim_options=sim_options)
model.fit(trainSet)
simsMatrix = model.compute_similarities()
```

# Chọn sản phẩm yêu thích của người dùng

```
testUserInnerID = trainSet.to_inner_uid(testSubject)
testUserRatings = trainSet.ur[testUserInnerID]
kNeighbors = [rating for rating in testUserRatings if rating[1] > 4.0]
```

Chọn những sản phẩm mà người dùng đã đánh giá trên 4.0 điểm

## Tìm các sản phẩm tương tự

Tìm các item tương tự với những item mà người dùng đã đánh giá bằng cách tính tổng: score \* (rating / 5.0)

```
# Get similar items to stuff we liked (weighted by rating)
candidates = defaultdict(float)
for itemID, rating in kNeighbors:
    similarityRow = simsMatrix[itemID]
    for innerID, score in enumerate(similarityRow):
        candidates[innerID] += score * (rating / 5.0)
```

```
In [5]: runfile('C:/Users/Admin/Desktop/HUST/GR1/CollaborativeFiltering/SimpleItemCF.py',
wdir='C:/Users/Admin/Desktop/HUST/GR1/CollaborativeFiltering')
Reloaded modules: MovieLens, ContentKNNAlgorithm, EvaluationData, RecommenderMetrics,
EvaluatedAlgorithm, Evaluator
Computing the cosine similarity matrix...
Done computing similarity matrix.
Computing the cosine similarity matrix...
Done computing similarity matrix.
Machete (2010) 9.233960269854148
47 Ronin (2013) 9.230262169999088
Hairspray (2007) 9.213647604612948
Johnny English (2003) 9.197174512772522
13 Assassins (JÃ»san-nin no shikaku) (2010) 9.188221279130666
It's Kind of a Funny Story (2010) 9.187494372165327
For a Few Dollars More (Per qualche dollaro in piÃ¹) (1965) 9.176606337889563
Megamind (2010) 9.172178852622746
3:10 to Yuma (2007) 9.168695550350016
Shall We Dance? (2004) 9.164158350575471
Killing Them Softly (2012) 9.160745847603803
```

```
In [17]: runfile('C:/Users/Admin/Desktop/HUST/GR1/  
CollaborativeFiltering/EvaluateItemCF.py', wdir='C:/  
Users/Admin/Desktop/HUST/GR1/CollaborativeFiltering')  
Reloaded modules: MovieLens, ContentKNNAlgorithm,  
EvaluationData, RecommenderMetrics,  
EvaluatedAlgorithm, Evaluator  
Loading movie ratings...  
  
Computing movie popularity ranks so we can measure  
novelty later...  
Estimating biases using als...  
Computing the cosine similarity matrix...  
Done computing similarity matrix.  
Computing the cosine similarity matrix...  
Done computing similarity matrix.  
Computing the cosine similarity matrix...  
Done computing similarity matrix.  
HR 0.005952380952380952
```

# TỔNG KẾT

- Content-based: Đơn giản nhưng không có gì mới dùng
- Collaborative Filtering: User-based và Item-based có các chỉ số đánh giá và kết quả gợi ý khá tương đồng nhưng theo cảm nhận và lý thuyết. Item-based nhỉnh hơn vì độ ổn định của một bộ phim không bị ảnh hưởng quá mạnh bởi sự thay đổi của sở thích các người dùng

# TỔNG KẾT

- Đánh giá qua chỉ số Hit Rate
- Đánh giá qua quan sát và đọc hiểu kết quả cũng như lịch sử hoạt động
- Ví dụ của người dùng thực tế

# ví dụ thực tế

Người dùng ID 672: Trần Bình Minh

672	5349	5	1.075E+09	Spider-man 2002	
672	52722	5	1.075E+09	Spider-man 2003	
672	59315	5	1.075E+09	Iron man 2008	
672	77561	5	1.075E+09	Iron man 2	
672	88140	5	1.075E+09	Captain America: The First Avenger (2011)	
672	110102	5	1.075E+09	Captain America: The Winter Soldier (2014)	
672	122920	5	1.075E+09	Captain America: Civil War (2016)	
672	103228	5	1.075E+09	Pacific Rim (2013)	
672	134853	5	1.075E+09	Inside Out (2015)	
672	45517	5	1.075E+09	Cars (2006)	
672	51077	5	1.075E+09	Ghost Rider (2007)	

# VÍ DỤ THỰC TẾ

## User-based Collaborative Filtering

**Recommend những phim khá nổi, the Matrix, Lords of the Rings khá cuốn! Những bộ phim khác có khả năng cũng thích. 7 điểm.**

Matrix, The (1999) 114.92364337544109  
Lord of the Rings: The Fellowship of the Ring, The (2001) 108.42628174762113  
Forrest Gump (1994) 104.65707212659163  
Lord of the Rings: The Two Towers, The (2002) 103.93816041866705  
Star Wars: Episode IV - A New Hope (1977) 100.07030444649837  
Lord of the Rings: The Return of the King, The (2003) 99.62460408904153  
Shawshank Redemption, The (1994) 99.47172081294912  
Pulp Fiction (1994) 90.98149864382363  
Shrek (2001) 90.65900025887497  
Fight Club (1999) 88.51876924745737  
Star Wars: Episode V - The Empire Strikes Back (1980) 88.1373124594967

# VÍ DỤ THỰC TẾ

## Item-based Collaborative Filtering

Có những phim rất hay như *Cowboy Bebop*, còn lại là những phim mới nghe qua hoặc không biết nhưng đọc mô tả cũng khá thú vị. 6.5 điểm.

- Sorcerer's Apprentice, The (2010)  
10.995672314834092
- Beethoven (1992) 10.975425219380615
- Batman: The Dark Knight Returns, Part 2 (2013) 10.972388874011262
- Iron Will (1994) 10.93954905787692
- Day of the Doctor, The (2013)  
10.938215942991896
- Cowboy Bebop (1998) 10.933194358303698
- No Reservations (2007) 10.920581663174
- Wrong Guy, The (1997)  
10.919145030018058
- Polar Express, The (2004)  
10.876743020770821
- Hairspray (2007) 10.873586631960006
- Passion of the Christ, The (2004)  
10.87154467323717

# ví dụ thực tế

## Content-based Filtering

Không nhận ra phim nào, thấy linh tinh không  
liên quan.

- Sorcerer's Apprentice, The (2010)  
10.995672314834092
- Beethoven (1992) 10.975425219380615
- Batman: The Dark Knight Returns, Part 2  
(2013) 10.972388874011262
- Iron Will (1994) 10.93954905787692
- Day of the Doctor, The (2013)  
10.938215942991896
- Cowboy Bebop (1998) 10.933194358303698
- No Reservations (2007) 10.920581663174
- Wrong Guy, The (1997)  
10.919145030018058
- Polar Express, The (2004)  
10.876743020770821
- Hairspray (2007) 10.873586631960006
- Passion of the Christ, The (2004)  
10.87154467323717

**THANK'S FOR  
YOUR TIME**

