LECTURE 7 RECOMMENDER SYSTEM

LEK HSIANG HUI

OUTLINE

Introduction

Similarity Measures

Content-Based Recommender Systems

Introduction to Collaborative Filtering

User-based Collaborative Filtering (UBCF)

INTRODUCTION

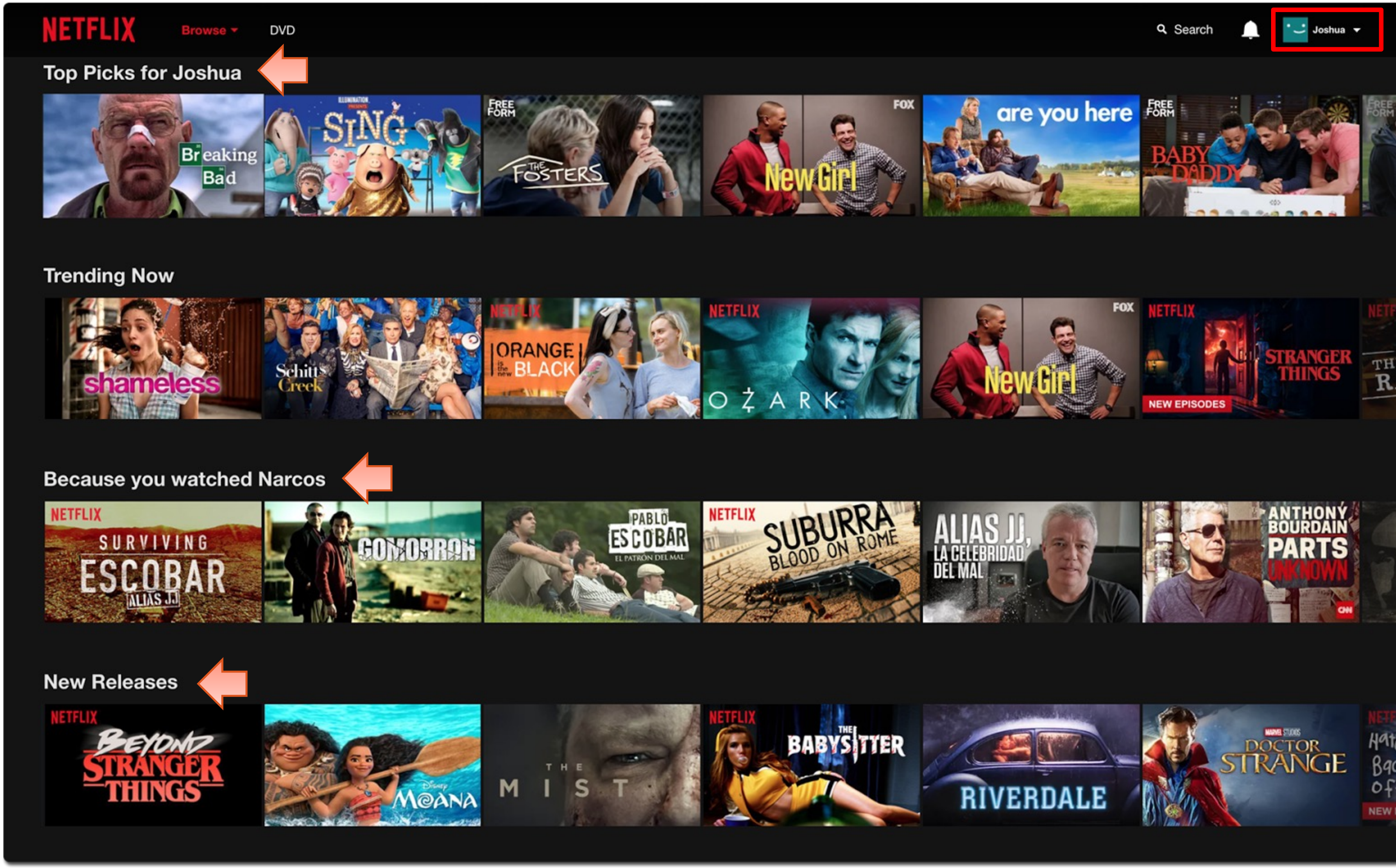
Introduction → Similarity Measures → Content-based Rec Systems → Introduction to CF → User-based CF

EXAMPLE OF RECOMMENDER SYSTEMS

amazon

Nuby Ice Gel Teether Keys

NETFLIX PERSONALIZATION



MANY OTHER RECOMMENDER SYSTEMS ONLINE

RECOMMENDER SYSTEMS

Recommender systems aim to:

- provide information that is relevant and useful
- make systems smarter and provide better user experience
- help businesses encourage more purchases

TYPES OF RECOMMENDATIONS

Editorial and hand curated

- Product of the Week
- Staff's favorites
- etc

Simple Aggregates

- Most popular, Top rated

Tailored to individual users

- Personalized recommendations

Will focus on this approach

THE RECOMMENDATION PROBLEM

U **= set of Users**

S **= set of Items**

Utility function *: U* x *S* → *R*

- ***R*** = set of ratings
- E.g. **1-5** stars, real number in **[0,1]**

UTILITY MATRIX

Objective:
Make use of existing data to predict the utility value of each item ***s*** ($\in S$) to each user ***u*** ($\in U$)

Then recommend the top ***k*** items to ***u***

Items

	X-Men	Antman	Frozen	Cinderella	Annabelle
Alice			5	5	2
Bob	4	5		1	
Charlie	3	2			5
…	…	…	…	…	…

Users

PREDICTION

2 common types of predictions:

Rating prediction

- Predict the rating score that a user is likely to give to an item (that is not seen)
- Recommendation is the unseen items with highest ratings

Item prediction

- Predict a ranked list of items that a user is likely to buy or use

KEY CHALLENGES

1. How to gather the ratings?

2. How to derive the unknown ratings?

	X-Men	Antman	Frozen	Cinderella	Annabelle
Alice			5	5	2
Bob	4	5		1	
Charlie	3	2			5
…	…	…	…	…	…

1. GATHER RATINGS

Explicit

- Ask users to rate items
- Doesn’t work well in practice – people can’t be bothered ☹

Implicit

- Learn ratings from user actions
    - E.g. purchase implies high rating
- What about low ratings?

2. DERIVE UNKNOWN RATINGS

Key Problem: Utility matrix is sparse

- Most of the entries are empty
- **Cold start** problem
  - New items have no ratings
  - New users have no history

SIMILARITY MEASURES

- Introduction
- Similarity Measures
- Content-based Rec Systems
- Introduction to CF
- User-based CF

SIMILARITY MEASURES

To find movies similar to a user's interest, there are a few similarity measures that can be adopted:

- Euclidean Distance
- Cosine Similarity
- Correlation
- Jaccard Similarity

User similarity:

- **u** = target user
- **v** = another user
- Each user is represented by their ratings of movies
- Want to find sim(**u**, **v**)
- Then recommend movies watched by similar users

Rating of **u** for movie i

EUCLIDEAN DISTANCE

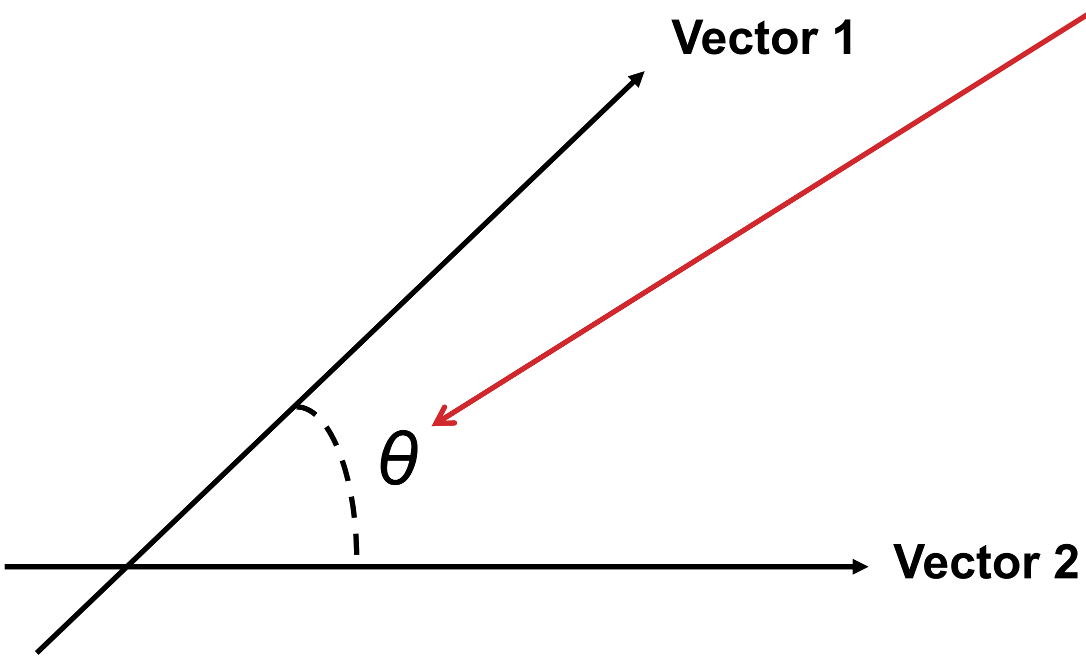
Euclidean distance is the square root of square differences in the components

$$d(\mathbf{u},\mathbf{v}) = \sqrt{(r_{\mathbf{u},1} - r_{\mathbf{v},1})^2 + \ldots + (r_{\mathbf{u},i} - r_{\mathbf{v},i})^2 + \ldots + (r_{\mathbf{u},n} - r_{\mathbf{v},n})^2}$$

	X-Men	Antman	Frozen	Cinderella	Annabelle
Alice			5	5	2
Bob	4	5		1	
…	…	…	…	…	…

COSINE SIMILARITY

Cosine similarity is a measure of similarity between 2 non-zero vectors



Smaller the angle means that they are more similar

Why is **cosine** function is used?

Think about the cosine graph

COSINE SIMILARITY

Cosine similarity is a measure of similarity between 2 non-zero vectors

$$cos(\theta) = cos(\mathbf{u}, \mathbf{v}) = \frac{\vec{r}_u . \vec{r}_v}{\|\vec{r}_u\| . \|\vec{r}_v\|} = \frac{\sum_i r_{\mathbf{u},i} r_{\mathbf{v},i}}{\sqrt{\sum_i^n {r_{\mathbf{u},i}}^2} \sqrt{\sum_i^n {r_{\mathbf{v},i}}^2}}$$

Consider every item. If a user has not rated the item, the rating is 0

Only consider common items where both **u** and **v** have rating

CORRELATION

The Pearson's Correlation Coefficient is another common similarity measure

$$cor(\mathbf{u},\mathbf{v}) = \frac{\sum_{i\in C}(r_{\mathbf{u},i} - \bar{r}_{\mathbf{u}})(r_{\mathbf{v},i} - \bar{r}_{\mathbf{v}})}{\sqrt{\sum_{i\in C}(r_{\mathbf{u},i} - \bar{r}_{\mathbf{u}})^2}\sqrt{\sum_{i\in C}(r_{\mathbf{v},i} - \bar{r}_{\mathbf{v}})^2}}$$

Note: regarding the **mean** value, there seems to be differing opinions whether it is average over <u>all items rated by the user u</u> or just average <u>over items common items</u>

We will stick with the former (i.e. all items rated by user u)

CORRELATION

$$cor(\mathbf{u},\mathbf{v}) = \frac{\sum_{i\in C}(r_{\mathbf{u},i}-\bar{r}_{\mathbf{u}})(r_{\mathbf{v},i}-\bar{r}_{\mathbf{v}})}{\sqrt{\sum_{i\in C}(r_{\mathbf{u},i}-\bar{r}_{\mathbf{u}})^2}\sqrt{\sum_{i\in C}(r_{\mathbf{v},i}-\bar{r}_{\mathbf{v}})^2}}$$

	X-Men	Antman	Frozen	Cinderella	Annabelle
Alice			5	5	2
Bob	4	5		1	
…	…	…	…	…	…

$$\bar{r}_{\text{Alice}} = (5+5+2)/3 = 4$$
$$\bar{r}_{\text{Bob}} = (4+5+1)/3 = 3.333$$
$$cor(\textbf{Alice},\textbf{Bob}) = \frac{(5-\bar{r}_{\textbf{Alice}})(1-\bar{r}_{\textbf{Bob}})}{\sqrt{(5-\bar{r}_{\textbf{Alice}})^2}\sqrt{(1-\bar{r}_{\textbf{Bob}})^2}} = \frac{(1)(-2.333)}{\sqrt{(1)^2}\sqrt{(-2.333)^2}} = -1$$

JACCARD SIMILARITY

Jaccard similarity is a method of finding portion of intersection

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

E.g. ***A*** = [watching, tv, and, reading, book]

B = [reading, LOTR]

$J(\boldsymbol{A},\boldsymbol{B}) = 1 / 6$

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

JACCARD SIMILARITY

Unlike previous measures, it does not consider the actual rating in the formula

How to calculate Jaccard similarity for users?

- Possible strategies:
- Convert utility matrix into Boolean flags (1 if rated, 0 if not rated)
- Or
- Treat ratings (3,4,5) as 1 and (1,2,blank) as 0

HANDS-ON: SIMILARITY MEASURES

Download and access:
Similarity Measures.ipynb

jupyter Similarity Measures Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

Similarity Measure

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
utility_matrix = pd.DataFrame([[np.nan, np.nan, 5, 5, 2],
                                [4, 5, np.nan, 1, np.nan]],
                    columns = ["x-men", "antman", "frozen", "cinderalla", "annabelle"],
                    index = ["alice", "bob"])
utility_matrix
```

Out[2]:

	x-men	antman	frozen	cinderalla	annabelle
alice	NaN	NaN	5.0	5	2.0
bob	4.0	5.0	NaN	1	NaN

Euclidean Distance

In [3]:
```
alice_rating = utility_matrix.loc["alice"]
alice_rating
```

Out[3]:
```
x-men         NaN
antman        NaN
frozen        5.0
cinderalla    5.0
annabelle     2.0
Name: alice, dtype: float64
```

CONTENT-BASED RECOMMENDER SYSTEMS

Introduction → Similarity Measures → **Content-based Rec Systems** → Introduction to CF → User-based CF

CONTENT-BASED RECOMMENDER SYSTEM

Idea: User recommended items similar to their preferences

- How to determine preferences?
- Could be determined based on the ratings given to different movies
- Could also build up user profile by explicitly asking user preferences

CONTENT-BASED RECOMMENDER SYSTEM

Idea: User recommended items similar to their preferences

- What does it mean by similar to their preferences?
- E.g. Movie recommendations:
    - Same actor(s), director, genre, etc

CONTENT-BASED RECOMMENDER SYSTEM

Strategy:

For each item → Item Profiles → Combine → User Profile → Compare against other items → Recommend top n items most similar to user profile

ITEM PROFILES

For each item, create an item profile

- What is an item profile?
- Perform **feature engineering** and come up with a list of features for each item
- E.g. Features for a movie (item):
  - Actors, Year, Movie Length, Language, Genre, etc

ITEM PROFILES

Encode the genre

	title	year	unknown	Action	Adventure	Animation	Children's
1	Toy Story (1995)	1995	0	0	0	1	1
2	GoldenEye (1995)	1995	0	1	1	0	0
3	Four Rooms (1995)	1995	0	0	0	0	0
4	Get Shorty (1995)	1995	0	1	0	0	0
5	Copycat (1995)	1995	0	0	0	0	0
6	Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)	1995	0	0	0	0	0
7	Twelve Monkeys (1995)	1995	0	0	0	0	0

	title	...	Tom Hanks	Elizabeth Taylor	John Travolta	Sigourney Weaver	Gong Li	Bruce Willis	Pierce Brosnan	Antonio Banderas
1	Toy Story (1995)	...	1	0	0	0	0	0	0	0
2	GoldenEye (1995)	...	0	0	0	0	0	0	1	0
3	Four Rooms (1995)	...	0	0	0	0	0	0	0	0
4	Get Shorty (1995)	...	0	0	1	0	0	0	0	0
5	Copycat (1995)	...	0	0	0	1	0	0	0	0
6	Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)	...	0	0	0	0	1	0	0	0
7	Twelve Monkeys (1995)	...	0	0	0	0	0	1	0	0

Encode the actors

USER PROFILE

Pick out the items that the user has rated, and combine them into a user profile

- How can we combine the item profiles?
- Possible strategies:
  - Weighted average of the rated item profiles
  - Weighted by the difference the user rating from the average rating of that item

Weighted average of the rated item profiles

USER PROFILE

Suppose user has watched these 3 movies:

	title	year	unknown	Action	Adventure	Animation	Children's
1	Toy Story (1995)	1995	0	0	0	1	1
3	Four Rooms (1995)	1995	0	0	0	0	0
6	Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)	1995	0	0	0	0	0

and given rating 5, 3, 2

We multiply 5 with 1st row, 3 with 2nd row, 2 with 3rd row and sum up these 3 rows

and divide by 3 (i.e. average)

to obtain a single row with *p* columns
(where ***p*** = number of predictors/features)

MAKING RECOMMENDATIONS

Using the user profile *x*, we can then compare its similarity with all the other item profiles *i*

- Using any of the similarity measures we have discussed
- E.g. $cor(\boldsymbol{x},\boldsymbol{i})$
- Then rank the similarity and choose the top ***k*** items with the highest similarity value

CONTENT-BASED APPROACH

Pros:

- No need data of other users (just item information and user preference)
- Able to recommend to users with unique tastes
- Able to recommend new and unpopular items
- Able to provide explanations of recommendations

CONTENT-BASED APPROACH

Cons:

- Feature engineering is not so straightforward
    - What features to use?
    - Did we miss out any features which are relevant?
- How to make recommendations if it’s a new user
- Overspecialization
    - Never recommends items outside user’s preferences
    - People might have multiple interests
    - Unable to exploit judgement of other users

INTRODUCTION TO COLLABORATIVE FILTERING

Introduction → Similarity Measures → Content-based Rec Systems → **Introduction to CF** → User-based CF

RECALL: UTILITY MATRIX

So far we have not make use of information of the different users and their ratings to aid in the recommendation

Items

	X-Men	Antman	Frozen	Cinderella	Annabelle
Alice			5	5	2
Bob	4	5		1	
Charlie	3	2			5
…	…	…	…	…	…

Users

COLLABORATIVE FILTERING

Collaborative Filtering (CF) make use of the ratings of other users to make the recommendations

The unique thing about CF compared to other approaches is that we do not need content information about the items

- Why is this a benefit?
- The recommender system can work for any items
- We do not need to handcraft different features for different domains

COLLABORATIVE FILTERING

2 main kinds of Collaborative Filtering approaches:

- User-based Collaborative Filtering
  - Making recommendation based on similarity between users
- Item-based Collaborative Filtering
  - Making recommendation based on similarity between items

USER-BASED COLLABORATIVE FILTERING

Introduction → Similarity Measures → Content-based Rec Systems → Introduction to CF → **User-based CF**

USER-BASED COLLABORATIVE FILTERING

Strategy:

User x

Compare preferences

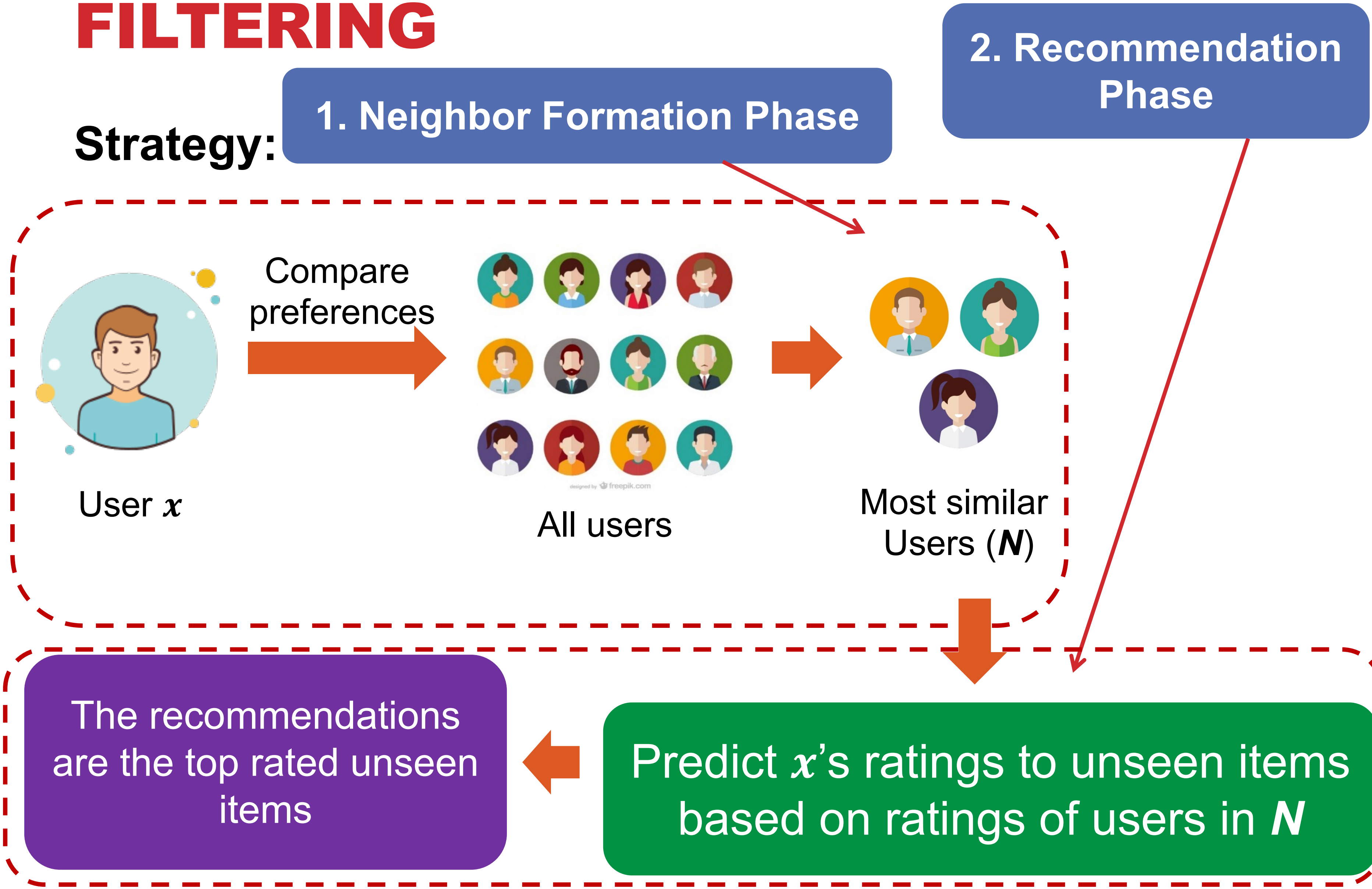
All users

Most similar Users (***N***)

Predict x's ratings to unseen items based on ratings of users in ***N***

The recommendations are the top rated unseen items

USER-BASED COLLABORATIVE FILTERING



NEIGHBORHOOD FORMATION PHASE

	X-Men	Antman	Frozen	Cinderella	Annabelle
Alice		?	5	5	2
Charlie	1		1	2	4
Dave		2	5		1
…	…	…	…	…	…

Suppose we want to predict what **Alice** is likely to give as rating for **Antman**

Find the set of users who have also watched **Antman** and determine the set of most similar users denoted as ***N***

NEIGHBORHOOD FORMATION PHASE

How to calculate the similarity between users?

- Have discussed the common similarity measures:
  - Euclidean Distance
  - Cosine Similarity
  - Correlation
  - Jaccard Similarity

SIMILARITY BETWEEN USERS

	X-Men	Antman	Frozen	Cinderella	Annabelle
Alice			5	5	2
Charlie	1		1	2	4
Dave		2	5		1
…	…	…	…	…	…

Given the above utility matrix, intuitively we want:

- sim(**Alice**, **Charlie**) < sim(**Alice**, **Dave**)
- Using Jaccard Similarity:
  - J(**Alice**, **Charlie**) = 3/4, J(**Alice**, **Dave**) = 2/4
  - J(**Alice**, **Charlie**) $\not<$ J(**Alice**, **Dave**)

SIMILARITY BETWEEN USERS

	X-Men	Antman	Frozen	Cinderella	Annabelle
Alice			5	5	2
Charlie	1		1	2	4
Dave		2	5		1
…	…	…	…	…	…

Given the above utility matrix, intuitively we want:

- sim(**Alice**, **Charlie**) < sim(**Alice**, **Dave**)
- Using Cosine Similarity:
  - cos(**Alice**, **Charlie**) = 0.667, cos(**Alice**, **Dave**) = 0.671
  - cos(**Alice**, **Charlie**) < cos(**Alice**, **Dave**)
  - But very close, so not so ideal

SIMILARITY BETWEEN USERS

	X-Men	Antman	Frozen	Cinderella	Annabelle
Alice			5	5	2
Charlie	1		1	2	4
Dave		2	5		1
…	…	…	…	…	…

Given the above utility matrix, intuitively we want:

- sim(**Alice**, **Charlie**) < sim(**Alice**, **Dave**)
- Using Pearson Correlation Coefficient:
  - cor(**Alice**, **Charlie**) = -0.912, cor(**Alice**, **Dave**) = 0.883
  - cor(**Alice**, **Charlie**) < cor(**Alice**, **Dave**)
  - Much better!

NEIGHBORHOOD FORMATION PHASE

Once we have all the similarity value between Alice and other users, we need to determine *N* (the set of most similar users)

- How to determine ***N***?
- 2 common approaches:
  - Rank the similarity values and choose ***k*** users with the highest similarity value
  - Choose all users with similarity value higher than a threshold

This is effectively doing the K-Nearest Neighbor (kNN) algorithm

- kNN is typically for classification, but now it can be used as part of the process to predict the rating of an unseen movie

RECOMMENDATION PHASE: RATING PREDICTION

Next step is to combine ratings of *N* to make a rating prediction

- How to combine the rating?
- Let $r_{x,i}$ be the rating prediction of movie $\boldsymbol{i}$ for user $\boldsymbol{x}$
  - $\hat{r}_{x,i} = \frac{1}{k} \sum_{y \in N} r_{y,i}$
  - or
  - $\hat{r}_{x,i} = \frac{\sum_{y \in N} sim(x,y) \cdot r_{y,i}}{\sum_{y \in N} sim(x,y)}$

Average rating for i based on *N*

Weightage average rating

RECOMMENDATION PHASE: RATING PREDICTION

The previous 2 approaches does not take into account x's average rating

Could also generate the rating prediction based on the average rating of x ($\bar{r}_x$):

- $\hat{r}_{x,i} = \bar{r}_x + \frac{\sum_{y \in N} sim(x,y) \cdot (r_{y,i} - \bar{r}_x)}{\sum_{y \in N} |sim(x,y)|}$

RECOMMENDATION PHASE: RATING PREDICTION

Example:

Current user : ***x*** , unseen movie : ***i***

N = *3* users : ***a***, ***b***, ***c***

Ratings of users for movie ***i*** **:** $r_{\boldsymbol{a},i}$= 4, $r_{\boldsymbol{b},i}$= 3, $r_{\boldsymbol{c},i}$= 5

sim(***x***, ***a***) = 0.9, sim(***x***, ***b***) = 0.8, sim(***x***, ***c***) = 0.7

Average ratings ***x*** gave for any movies: $\overline{r_x}$ = 2

Approach 1

$$\hat{r}_{x,i} = \frac{1}{k} \sum_{y \in N} r_{y,i} = \frac{1}{3}(4+3+5) = 4$$

Approach 2

$$\hat{r}_{x,i} = \frac{\sum_{y \in N} sim(x,y) \cdot r_{y,i}}{\sum_{y \in N} sim(x,y)} = \frac{0.9*4+0.8*3+0.7*5}{0.9+0.8+0.7} = 3.96$$

Notice that if we do not consider a user's average rating, the prediction can differ by quite a bit

Approach 3

$$\hat{r}_{x,i} = \bar{r}_x + \frac{\sum_{y \in N} sim(x,y) \cdot (r_{y,i} - \bar{r}_x)}{\sum_{y \in N} |sim(x,y)|} = 2 + \frac{0.9*(4-2)+0.8*(3-2)+0.7*(5-2)}{3} = 3.56$$

RECOMMENDATION PHASE: MAKING RECOMMENDATIONS

After obtaining the x's rating predictions of all the unseen items, the next step is to make recommendations

Note that most of the time we are more interested in the recommendation results rather than the rating prediction

- How do we make recommendations?
- Rank movies by highest ratings and choose top ***m*** movies with the highest rating
- Or choose movies above a certain rating threshold

COLLABORATIVE FILTERING

2 main kinds of Collaborative Filtering approaches:

- User-based Collaborative Filtering
  - Making recommendation based on similarity between users
- Item-based Collaborative Filtering
  - Making recommendation based on similarity between items

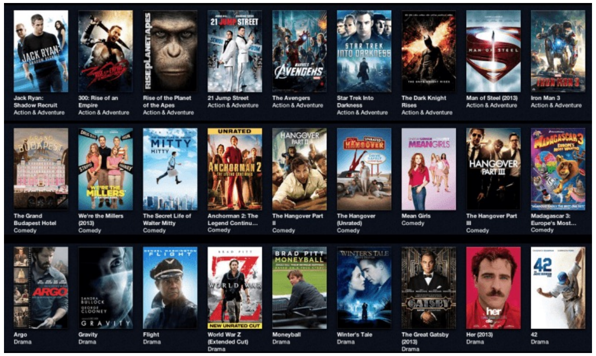
ITEM-BASED COLLABORATIVE FILTERING

Strategy:

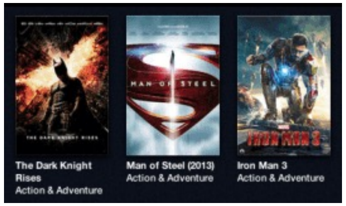
User $\boldsymbol{x}$

Unseen movie $\boldsymbol{i}$

Compare similarity



Movies rated by $\boldsymbol{x}$



Most similar movies ($\boldsymbol{N}(\boldsymbol{i};\boldsymbol{x})$)

Predict $\boldsymbol{x}$'s rating to $\boldsymbol{i}$ based on ratings of movies in $\boldsymbol{N}(\boldsymbol{i};\boldsymbol{x})$

The recommendations are the top rated unseen items

ITEM-BASED COLLABORATIVE FILTERING

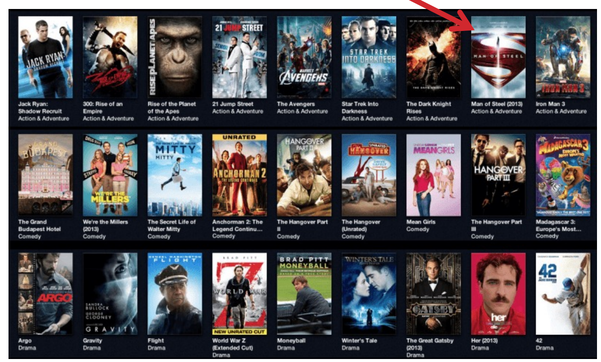
Strategy:

1. Neighbor Formation Phase

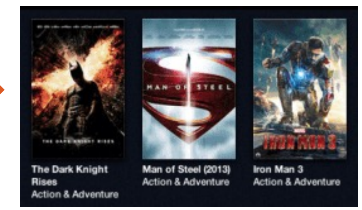
2. Recommendation Phase

User x
Unseen movie i

Compare similarity



Movies rated by x



Most similar movies ($N(i; x)$)

Predict x's rating to i based on ratings of movies in $N(i; x)$

The recommendations are the top rated unseen items

SUMMARY

Types of Recommendations

Making Recommendations using Aggregates

Similarity Measures

Content-based Recommender Systems

Collaborative Filtering Recommender Systems