EECS 445

Introduction to Machine Learning

Collaborative Filtering

Prof. Kutty

spectral clustering

review

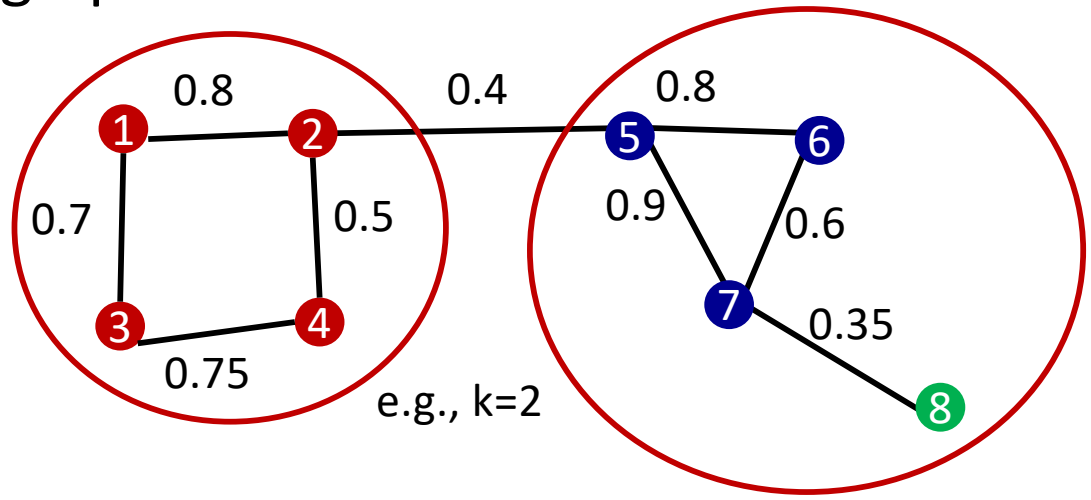
Spectral Clustering for k partitions

Input: $S_n = \{\bar{x}^{(i)}\}_{i=1}^{n}$, valid similarity metric, number of clusters k

Convert to graph

e.g.)

1	0
2	0
3	1
4	1
5	0
6	1
7	0
8	0



$G = (V, E)$

$A_1 = \{4, 5, 7\}$

solve graph partitioning problem

$$\min_{A_1,\ldots,A_k} RatioCut(A_1, \ldots, A_k)$$

$A_i \subseteq V$

$i \neq j \quad A_i \cap A_j = \phi$

$\bigcup_{i=1}^{k} A_i = V$

$i, j \in \{1, \ldots, k\}$

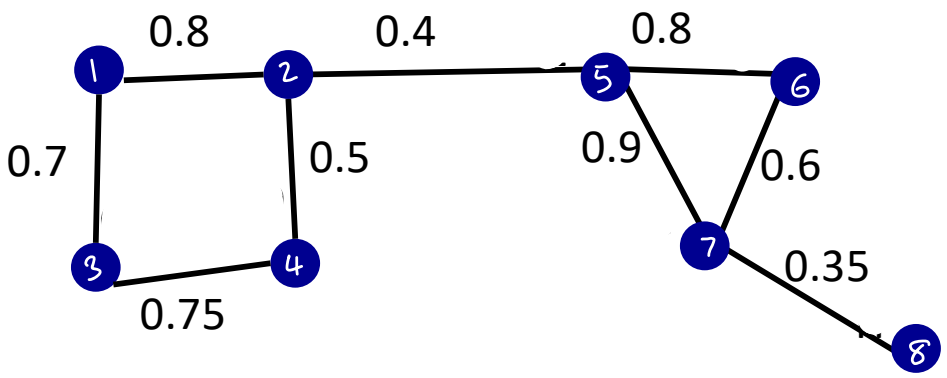
Output: clusters assignments

How to find min RatioCut

- NP hard so solve approximation instead
- Can show that $\bar{f}^T L \bar{f} \propto RatioCut(A, \bar{A})$
  - graph Laplacian $L = D - W$
  - k^{th} eigenvalue approximates cost of the ratiocut for k clusters
  - i^{th} eigenvector approximates an indicator vector corresponding to the i^{th} cut

- Note: for more details see Lecture 18, 19 and Discussions 5, 10

Spectral Clustering Example #2 revisited



Should be able to construct W, D, L

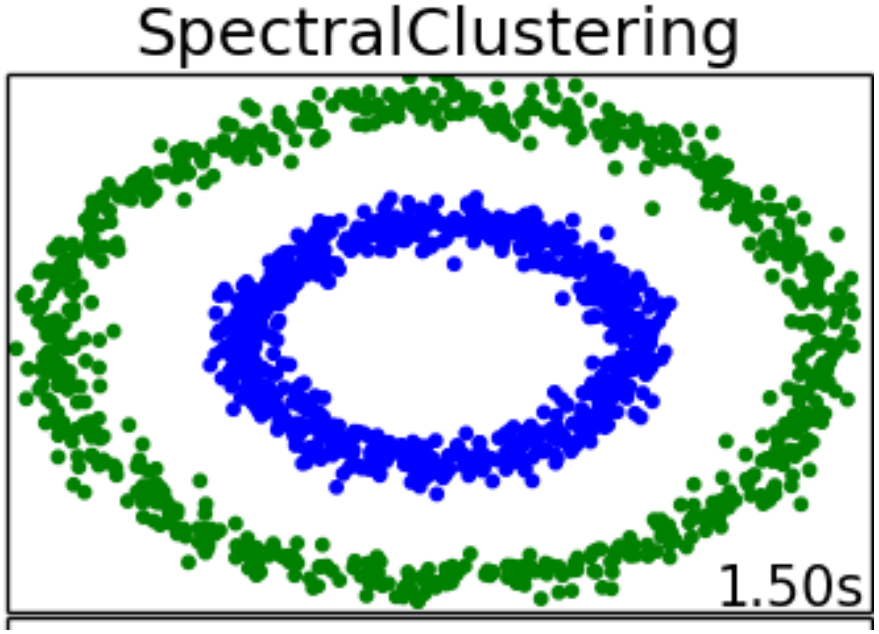
Eigenvectors

1	-0.3536	-0.3467	0.0704	0.6000	-0.2702	0.0594	0.4652	-0.3113
2	-0.3536	-0.2418	-0.0252	0.4060	0.5839	0.0496	-0.3458	0.4337
3	-0.3536	-0.3997	0.1334	-0.2603	-0.6097	-0.1178	-0.4406	0.2233
4	-0.3536	-0.3773	0.1106	-0.6251	0.4141	0.0655	0.3339	-0.2059
5	-0.3536	0.2188	-0.3571	0.0256	0.0968	-0.2652	-0.4338	-0.6566
6	-0.3536	0.2926	-0.4542	-0.1096	-0.1732	0.7125	0.0785	0.1654
7	-0.3536	0.3251	-0.2362	-0.0592	-0.0571	-0.6200	0.4008	0.4021
8	-0.3536	0.5291	0.7582	0.0227	0.0154	0.1160	-0.0583	-0.0509

Eigenvalues

0.0000	0.1349	0.4590	1.2624	1.6492	2.2200	2.7575	3.1170

Test your understanding:
Why does Spectral Clustering work here?



K=2

https://forms.gle/ffiBvNbPjHF8ghi77

Thank you for your feedback!

Stay the same:

- Theory + application
- Pacing for in-class notes
- In class problems/exercises and Lecture quizzes

Changes:

- Want more high level overview/recap
- Mid-lecture breaks
- More applications/examples of use cases
- More office hours (?)

collaborative filtering

“The most dominant approach for computing recommendations is collaborative filtering…“
Evaluating Recommender Systems: Survey and Framework
Eva Zangerle and Christine Bauer
ACM Computing Surveys 2022

Recommender systems

PANDORA

yelp
Real people. Real reviews.

LinkedIn

NETFLIX

YouTube

amazon

News

All the news you want.
All in one place.

News collects all the stories you want to read — so you no longer need to move from app to app to stay informed. And with a completely reimagined For You tab, it's easier than ever to find the stories that matter most to you. News also features breaking news notifications and subscriptions to some of your favorite publications.

npr one

NETFLIX

What We Watched: A Netflix Engagement Report

Hours viewed from January to June 2023

Title	Available Globally?	Release Date	Hours Viewed
[redacted]	Yes	2023-03-23	812,100,000
[redacted]	Yes	2023-01-05	665,100,000
[redacted]	Yes	2022-12-30	622,800,000
[redacted]	Yes	2022-11-23	507,700,000
[redacted]	Yes	2023-05-04	503,000,000
[redacted]	Yes	2023-02-09	440,600,000
[redacted]	No	2022-12-30	429,600,000
[redacted]	Yes	2023-02-23	402,500,000
[redacted]	Yes	2021-02-24	302,100,000
[redacted]	Yes	2023-05-25	266,200,000
[redacted]	Yes	2022-11-04	262,600,000
[redacted]	Yes	2023-01-01	252,500,000
[redacted]	Yes	2022-12-02	251,500,000
[redacted]	Yes	2023-05-12	249,900,000
[redacted]	Yes	2023-01-24	235,000,000
[redacted]	Yes	2023-01-14	234,800,000
[redacted]	Yes	2023-03-24	229,700,000
[redacted]	Yes	2023-04-06	221,100,000
[redacted]	Yes	2023-04-20	214,100,000
[redacted]	Yes	2023-03-10	209,700,000
[redacted]	No	2023-05-31	206,500,000
[redacted]	Yes	2023-01-12	205,500,000
[redacted]	Yes	2023-06-16	201,800,000
[redacted]	Yes	2023-05-18	200,700,000
[redacted]	Yes	2023-04-15	194,700,000
[redacted]	Yes	2023-03-16	192,900,000
[redacted]	Yes	2020-04-15	184,000,000

https://about.netflix.com/en/news/what-we-watched-a-netflix-engagement-report

Recommender Systems

NETFLIX

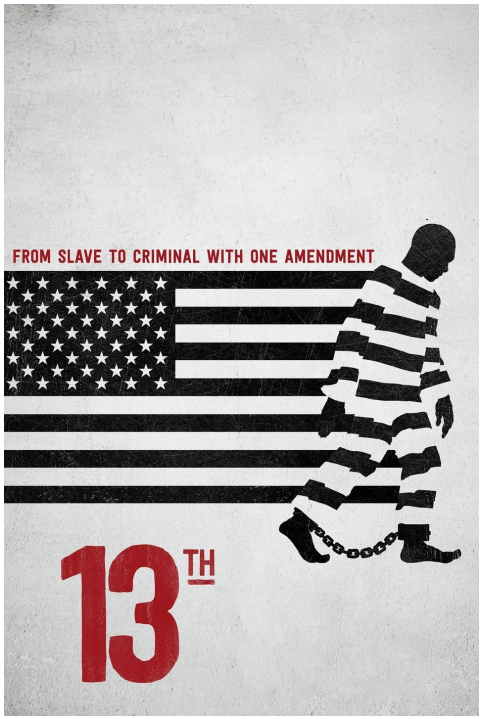
NETFLIX

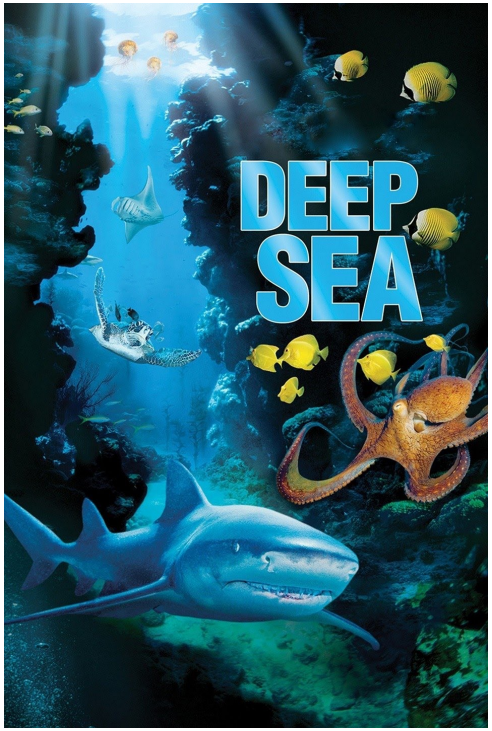
- Launched on January 16, 2007
- ~260.28 million users in more than 190 countries
- data on ~18k titles (representing 99% of all content viewed); record of nearly 100 billion hours viewed
- **Goal**: Predict ratings for titles a user hasn't seen

Collaborative Filtering – recommender problems that can be reduced to **matrix completion** (no additional info about the users or about the movies)

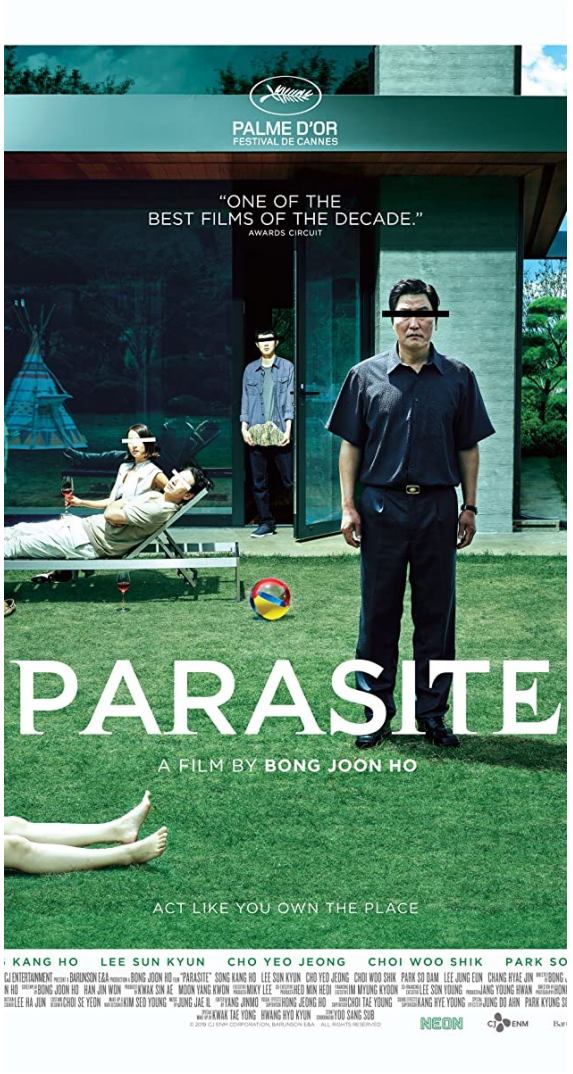
https://about.netflix.com/en/news/what-we-watched-a-netflix-engagement-report

if you liked





you might like



Steps:

- generate predictions
- pick movies to present to user

Matrix Completion

m movies

n users

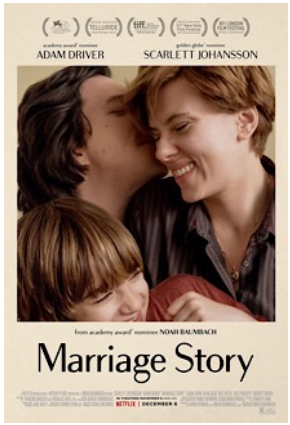
	5				4
		2	3		
	4				
				4	
1					
	2		3		
	5	1			3

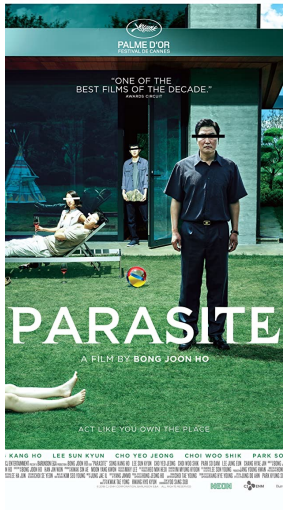
call this the utility (or user-item) matrix Y

How to solve for the missing ratings?

1) Matrix factorization
2) Nearest neighbor prediction

movies (from our own planet)

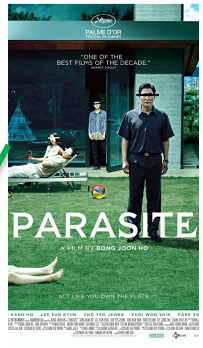


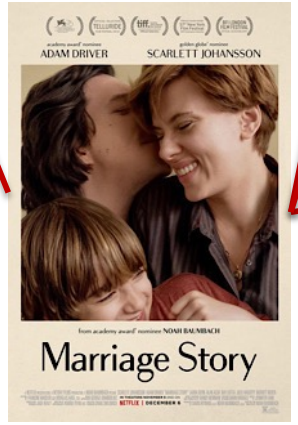


users (in a galaxy far far away…)

KNN prediction

target user





target item

Approach 2: Nearest Neighbor Prediction

Key idea:

Suppose user ***a*** has not rated movie ***i***

To predict the rating

- compute *similarity* between user ***a*** and all other users in the system
- find the *k* 'nearest neighbors' of user ***a*** who have rated movie ***i***
- compute a prediction based on these users' ratings of ***i***

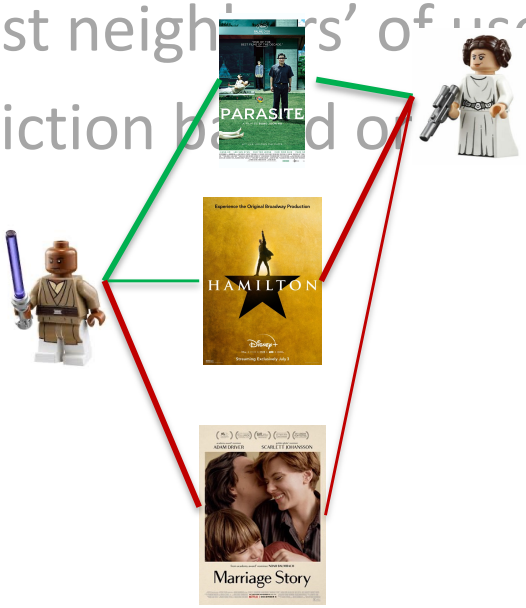
Approach 2: Nearest Neighbor Prediction

Key idea:

Suppose user ***a***  has not rated movie ***i*** 

To predict the rating

- compute *similarity* between user ***a*** and all other users in the system
- find the *k* 'nearest neighbors' of user ***a*** who have rated movie ***i***
- compute a prediction based on these users' ratings of ***i***



Correlation as a measure of similarity

Users: $a,\ b$

$R(a,b)$: set of movies rated by both users a and b

$$\tilde{Y}_{a:b} = \frac{1}{|R(a,b)|} \sum_{j \in R(a,b)} Y_{a_j}$$

average rating of user a for movies rated by both users

$\tilde{Y}_{b:a}$

$$sim(a,b) = \frac{\sum_{j \in R(a,b)} (Y_{a_j} - \tilde{Y}_{a:b})(Y_{b_j} - \tilde{Y}_{b:a})}{\sqrt{\sum_{j \in R(a,b)} (Y_{a_j} - \tilde{Y}_{a:b})^2 \sum_{j \in R(a,b)} (Y_{b_j} - \tilde{Y}_{b:a})^2}}$$

how much the ratings vary individually i.e., std dev of the ratings of a and b

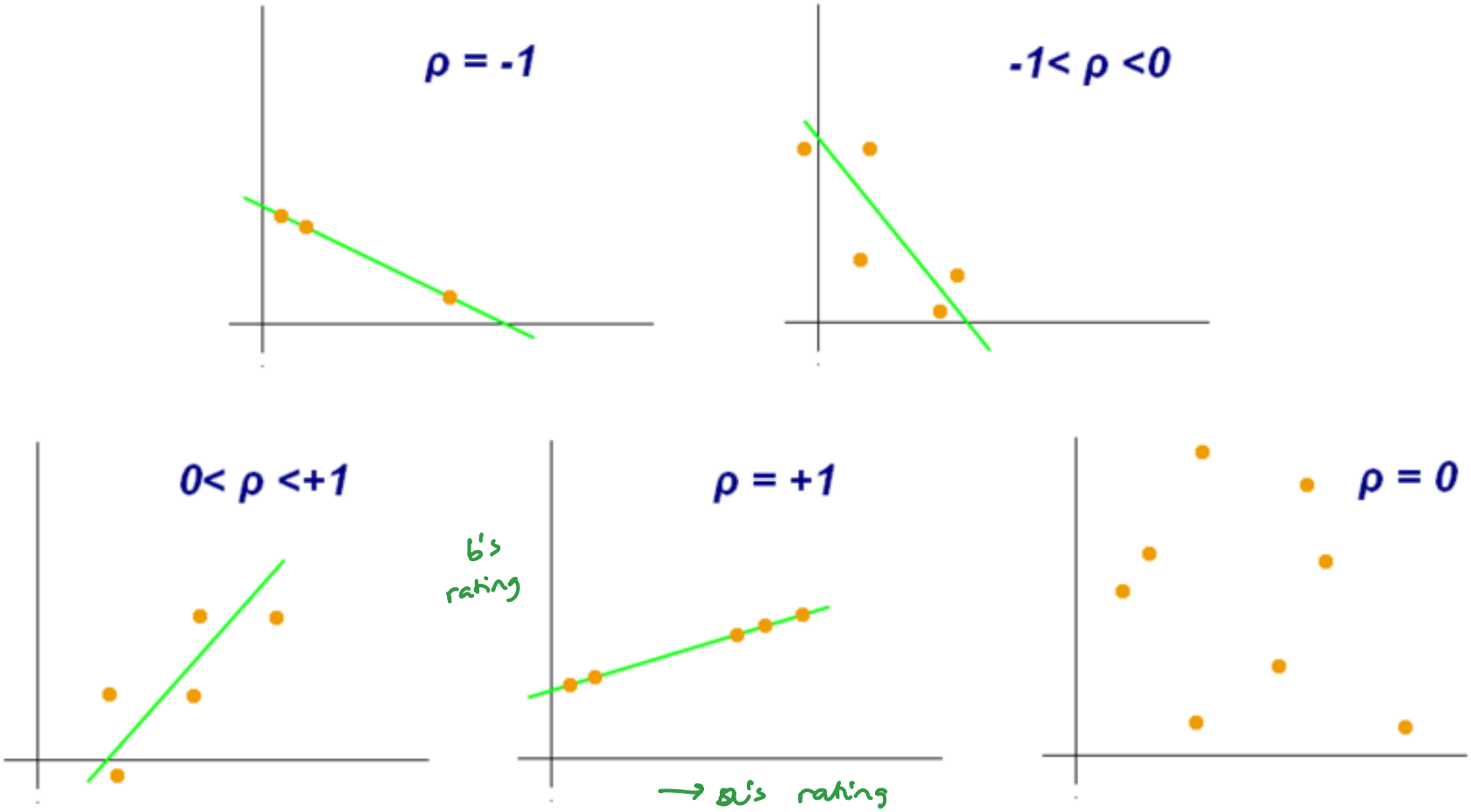
Sample Pearson's correlation coefficient

Correlation as a measure of similarity

$$sim(a,b) = \frac{\sum_{j \in R(a,b)} (Y_{a_j} - \tilde{Y}_{a:b})(Y_{b_j} - \tilde{Y}_{b:a})}{\sqrt{\sum_{j \in R(a,b)} (Y_{a_j} - \tilde{Y}_{a:b})^2 \sum_{j \in R(a,b)} (Y_{b_j} - \tilde{Y}_{b:a})^2}}$$

- measure of linear relationship between two variables
- ranges from [-1, 1]
  - -1, +1 → perfect linear relationship
  - 0 → no linear correlation
  - absolute value is a measure of degree of correlation

Correlation (intuition)



https://en.wikipedia.org/wiki/Pearson_correlation_coefficient

Example: similarity

a		5	?			4
			2	3		
		4				
					4	
	1					
		2		3		
b		5	1			3

$$sim(a,b) = \frac{\sum_{j \in R(a,b)} (Y_{a_j} - \tilde{Y}_{a:b})(Y_{b_j} - \tilde{Y}_{b:a})}{\sqrt{\sum_{j \in R(a,b)} (Y_{a_j} - \tilde{Y}_{a:b})^2 \sum_{j \in R(a,b)} (Y_{b_j} - \tilde{Y}_{b:a})^2}}$$

Example: similarity

a		5	?			4
b		5	1			3

$$sim(a,b) = \frac{\sum_{j \in R(a,b)} (Y_{a_j} - \tilde{Y}_{a:b})(Y_{b_j} - \tilde{Y}_{b:a})}{\sqrt{\sum_{j \in R(a,b)} (Y_{a_j} - \tilde{Y}_{a:b})^2 \sum_{j \in R(a,b)} (Y_{b_j} - \tilde{Y}_{b:a})^2}}$$

https://forms.gle/ffiBvNbPjHF8ghi77

$$|R(a,b)| = |\{2,6\}| = 2$$

$$\bar{Y}_{a:b} = \frac{5+4}{2} = 4.5$$

$$\bar{Y}_{b:a} = 4$$

$$sim(a,b) = \frac{(Y_{a2} - \bar{Y}_{a:b})(Y_{b2} - \bar{Y}_{b:a}) + (Y_{a6} - \bar{Y}_{a:b})(Y_{b6} - \bar{Y}_{b:a})}{\sqrt{((Y_{a2} - \bar{Y}_{a:b})^2 + (Y_{b2} - \bar{Y}_{b:a})^2)((Y_{a6} - \bar{Y}_{a:b})^2 + (Y_{b6} - \bar{Y}_{b:a})^2)}}$$

$$Y_{a2} - \bar{Y}_{a:b} = 5 - 4.5 = 0.5$$

$$Y_{b2} - \bar{Y}_{b:a} = 5 - 4 = 1$$

$$Y_{a6} - \bar{Y}_{a:b} = 4 - 4.5 = -0.5$$

$$Y_{b6} - \bar{Y}_{b:a} = 3 - 4 = -1$$

$$\therefore \quad sim(a,b) = \frac{(0.5)(1) + (-0.5)(-1)}{\sqrt{[0.5^2 + (-0.5)^2][(1)^2 + (-1)^2]}} = \frac{0.5 + 0.5}{\sqrt{(0.5)(2)}} = 1$$

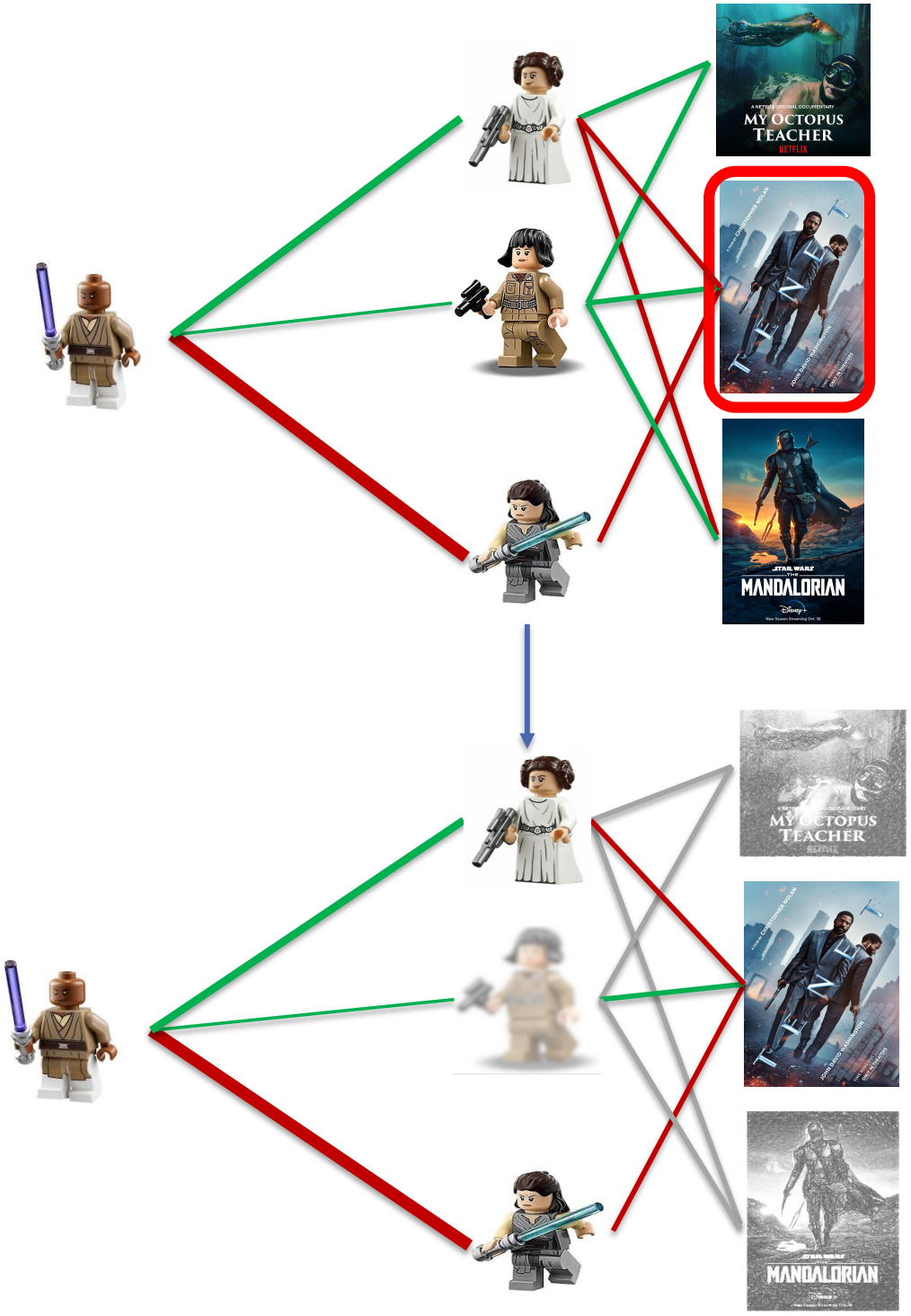
Approach 2: Nearest Neighbor Prediction

Key idea:

Suppose user ***a*** has not rated movie ***i***

To predict the rating

- compute *similarity* between user ***a*** and all other users in the system
- find the *k* 'nearest neighbors' of user ***a*** who have rated movie ***i***
- compute a prediction based on these users' ratings of ***i***



non increasing

ordered in decreasing order of *absolute* value

sim(a, user 1)
sim(a, user 2)
⋮
sim(a, user *k*)

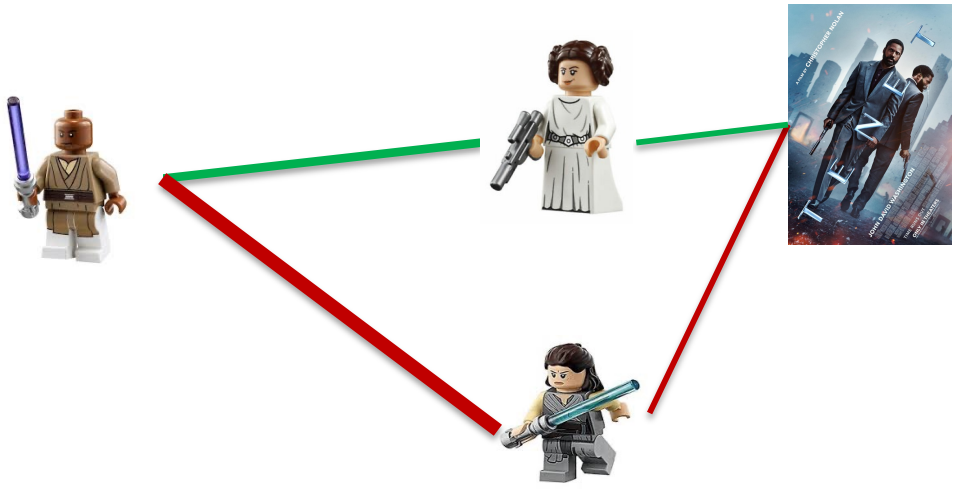
Approach 2: Nearest Neighbor Prediction

Key idea:

Suppose user ***a*** has not rated movie ***i***

To predict the rating

- compute *similarity* between user ***a*** and all other users in the system
- find the *k* 'nearest neighbors' of user ***a*** who have rated movie ***i***
- compute a prediction based on these users' ratings of ***i***



Prediction

$kNN(a, i)$ k “nearest neighbors”
i.e., the k most similar users to a,
who have rated movie i

$$\hat{Y}_{a_i} = \bar{Y}_a + \frac{1}{\sum_{b \in kNN(a,i)} |sim(a,b)|} \sum_{b \in kNN(a,i)} sim(a,b)(Y_{b_i} - \bar{Y}_b)$$

How to choose *k*?

- Some heuristics:
  - all users are considered as neighbors
  - Why could limiting neighborhood size result in more accurate predictions?
    - neighbors with low correlation tend to introduce noise
  - Variable *k*
    - varying neighborhoods are selected for each item based on a similarity threshold
  - offline analysis: values in the 20–50 range are a reasonable starting point in many domains.

How to solve for the missing ratings?

1) Matrix factorization
2) Nearest neighbor prediction