3. Recommender systems

Problem: What items to recommend to which user? Products, music, movies, dishes, learning material,...

Image source `https://sudonull.com/post/12374-Anatomy-of-recommendation-systems-Part-one`

Data and utility matrix

Data: User profiles, product descriptions, browsing and buying behaviour, explicit ratings.

Often possible to derive a **utility matrix $\mathbf{A}$**, where $\mathbf{A}[i, j]$ = utility of item j for user i

- $n \times d$ matrix, n=number users, d=number of items

Two types:

1. Only positive preferences (“likes”, browsing, buying)
2. Positive and negative preferences (“likes” and “dislikes”, ratings)

- extremely large and sparse matrices!

Example utility matrices (movie preferences)

	GLADIATOR	GODFATHER	BEN-HUR	GOODFELLAS	SCARFACE	SPARTACUS
U_1	1			5		2
U_2		5			4	
U_3	5	3		1		
U_4			3			4
U_5				3	5	
U_6	5		4			

(a) Ratings-based utility

	GLADIATOR	GODFATHER	BEN-HUR	GOODFELLAS	SCARFACE	SPARTACUS
U_1	1			1		1
U_2		1			1	
U_3	1	1		1		
U_4			1			1
U_5				1	1	
U_6	1		1			

(b) Positive-preference utility

Empty cell=unspecified; in data, e.g., –, na, 0 (if non-positive ratings).

Image source Aggarwal Ch 18

Main approaches: Content-based and preference-based (collaborative filtering)

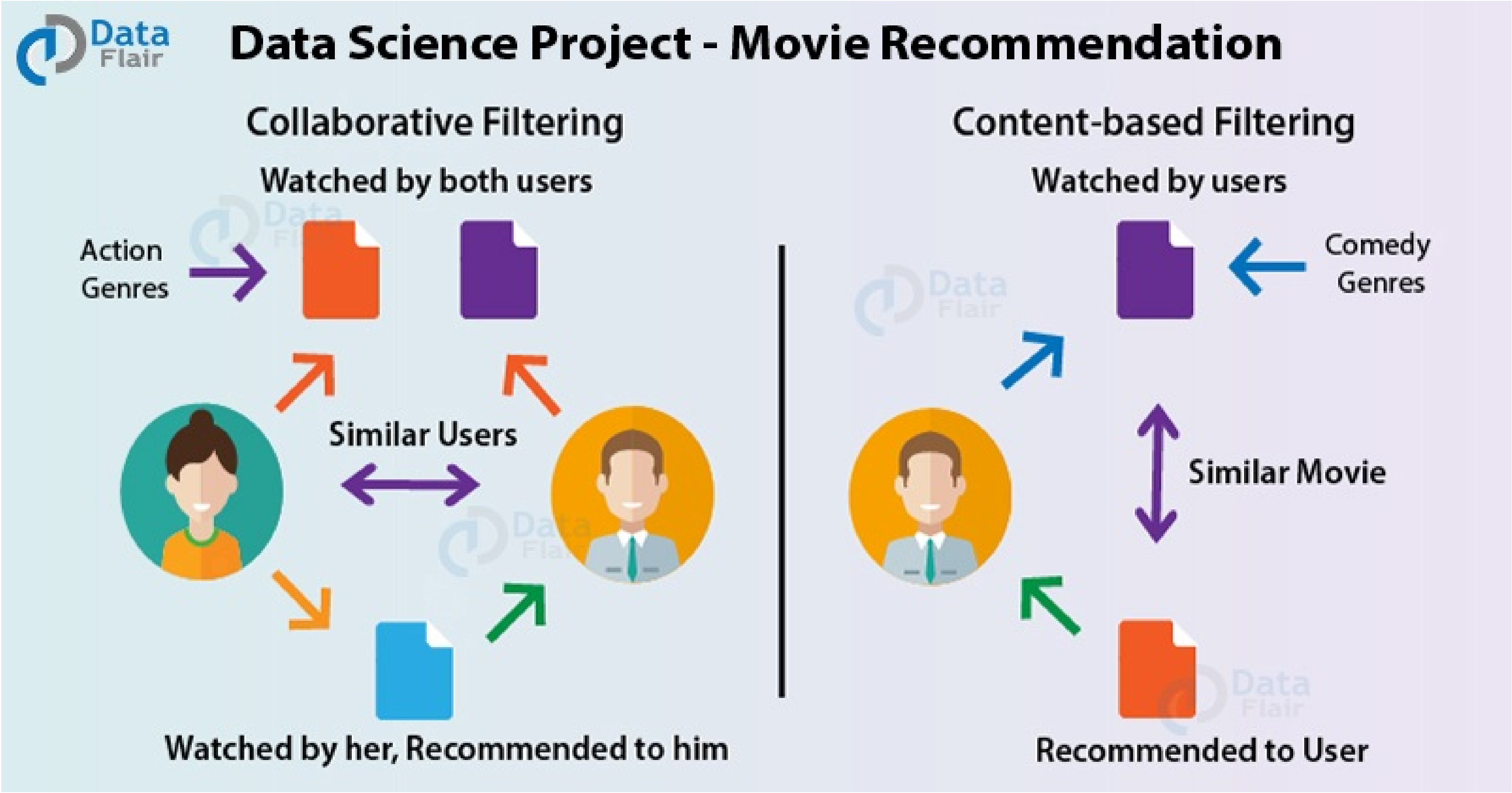


Image source
`https://data-flair.training/blogs/data-science-r-movie-recommendation/`

Content-based recommendations

Given

1. **item profile** = text descriptions, keywords
2. **user profile** = documents describing user's interests (e.g., descriptions of previously bought/liked items, explicitely specified or derived interests)

Search items whose profiles match (are similar) to the user's profile

a) **If no utility matrix**

- search K most similar items to the user profile
- e.g., tf-idf presentation + cos similarity

Content-based recommendations

b) **If utility matrix exists**, utilize the user's previous preferences!

= prediction task where vector $\mathbf{A}[i]$ = target values for user i

- If positive preference matrix, learn a classifier
- If numerical ratings, learn a regression model
- – training sets extremely small
- – over-specialization: recommendations tend to favour items described by the same keywords
  - e.g., recommend movies with the same actors as before

Collaborative filtering

Assumption: The user probably likes what other similar users have liked.

Approaches for recommendation

i) Neighbourhood-based

ii) Graph-based

iii) Clustering-based

iv) Latent factor -based

Neighbourhood-based methods: 1. user-based

Utilize **user–user similarity**

e.g., **Pearson correlation coefficient** r for similarity between two users' rating vectors $\mathbf{x} = (x_1, \ldots, x_d)$ and $\mathbf{y} = (y_1, \ldots, y_d)$:

$$r(\mathbf{x}, \mathbf{y}) = \frac{\sum_{j \in J}(x_j - \mu_x)(y_j - \mu_y)}{\sqrt{\sum_{j \in J}(x_j - \mu_x)^2 \sum_{j \in J}(y_j - \mu_y)^2}}$$

$J = \{j \mid x_j \neq na, y_j \neq na\}$ (items rated by both)
μ_x is average rating, two alternatives:

i) $\mu_x = \frac{1}{|J|} \sum_{j \in J} x_j$ (only common items) or

ii) $\mu_x = \frac{1}{|J_x|} \sum_{j \in J_x} x_j$, where $J_x = \{j \mid x_j \neq na\}$ (all rated items; more common approach)

Predict missing ratings in rating vector $\mathbf{x}$

1. search K nearest neighbours $NN_{\mathbf{x}}$ using similarity r
2. remove neighbours from $NN_{\mathbf{x}}$ if $r \leq \theta$ (negative or weak correlations)
3. normalize ratings: $y'_j = y_j - \mu_{\mathbf{y}}$ (since in different scales)
4. calculate predicted rating for all items j with missing entries in $\mathbf{x}$:

$$\tilde{x}_j = \frac{\sum_{\mathbf{y} \in NN_{\mathbf{x}}} w_{\mathbf{y}} \cdot y'_j}{\sum_{\mathbf{y} \in NN_{\mathbf{x}}} w_{\mathbf{y}}} + \mu_x$$

- $w_{\mathbf{y}} = 1$ or weigh by similarity $w_{\mathbf{y}} = r(\mathbf{x}, \mathbf{y})$
- i.e., weighted average rating by similar users + return to $\mathbf{x}$'s original scale

Example: Predict missing ratings ($K = 2$, $r \geq 0.5$)

	m_1	m_2	m_3	m_4	m_5	m_6
u_1	–	1	2	2	3	–
u_2	3	1	1	2	4	3
u_3	4	2	3	3	–	5
u_4	2	5	4	–	1	2

User means:

$\mu_1 = 2.000$
$\mu_2 = 2.333$
$\mu_3 = 3.400$
$\mu_4 = 2.800$

	u_1	u_2	u_3	u_4
u_1	1.000	0.836	0.927	-0.917
u_2	0.836	1.000	0.822	-0.974
u_3	0.927	0.822	1.000	-0.862
u_4	-0.917	-0.974	-0.862	1.000

m_1=Gladiator, m_2=Godfather, m_3=Ben-Hur, m_4=Goodfellas, m_5=Scarface, m_6=Spartacus

Example: Predict missing ratings ($K = 2$, $r \geq 0.5$)

	u_1	u_2	u_3	u_4
u_1	1.000	0.836	0.927	-0.917
u_2	0.836	1.000	0.822	-0.974
u_3	0.927	0.822	1.000	-0.862
u_4	-0.917	-0.974	-0.862	1.000

$\mu_1 = 2.000$
$\mu_2 = 2.333$
$\mu_3 = 3.400$
$\mu_4 = 2.800$

for u_1 nearest u_3 and u_2, predicted for u_1, m_1:

$$\frac{0.836 \cdot (3 - 2.333) + 0.927 \cdot (4 - 3.400)}{0.836 + 0.927} + 2.000 = 2.63 > \mu_1 \rightarrow \text{recommend}$$

for u_1, m_6 predicted 3.16
for u_3 nearest u_1 and u_2, for m_5 predicted 4.71
for u_4 not enough neighbours! (all $r < 0$)

Neighbourhood-based methods: 2. item-based

Utilize **item–item similarity**

$\mathbf{v}$ = jth item's rating vector, $\mathbf{x}$ = ith user's rating vector

1. search K nearest neighbours $NN_{\mathbf{v}}$ of $\mathbf{v}$
2. select a subset $NN_{\mathbf{v},\mathbf{x}} \subseteq NN_{\mathbf{v}}$ of those items's ratings that user i has rated: $NN_{\mathbf{v},\mathbf{x}} = \{\mathbf{u}_r \mid \mathbf{u}_r \in NN_{\mathbf{v}}, x_r \neq na\}$
3. Predicted rating is

$$\tilde{x}_j = \frac{\sum_{\mathbf{u}_r \in NN_{\mathbf{v},\mathbf{x}}} w_{\mathbf{v},\mathbf{u}_r} \cdot x_r}{\sum_{\mathbf{u}_r \in NN_{\mathbf{v},\mathbf{x}}} w_{\mathbf{v},\mathbf{u}_r}}$$

- i.e., weighted average rating on similar items by user i

Neighbourhood-based methods: 2. item-based

What similarity measure to use? Should we normalize ratings?

- Pearson correlation (+ mean centering can be used also here)
- **adjusted cosine similarity** = cosine similarity after mean centering each user's ratings
- **Problem**: cosine similarity with 0 vectors (movies with average ratings from everybody) is not defined $\leftrightarrow$ cos works better when all values positive

See Aggarwal 18.5.2.2

Graph-based methods

Idea: Create a bipartite **user-item graph** and utilize random walk approaches.

- graph $\mathbf{G} = (\mathbf{U} \cup \mathbf{V}, \mathbf{E})$
- $\mathbf{U}$ = nodes for users
- $\mathbf{V}$ = nodes for items
- $\mathbf{E}$ = edges such that $(u_i, v_j) \in \mathbf{E}$, $u_i \in \mathbf{U}$, $v_j \in \mathbf{V}$, if the ith user has rated the jth item
- if rating matrix (positive and negative preferences), the edges may have weights:
  - normalize rating $\mathbf{A}[i, j]$ by subtracting mean of ratings on row $\mathbf{A}_i$ $\rightarrow$ signed network

Bipartite user-item graph

	GLADIATOR	GODFATHER	BEN-HUR	GOODFELLAS	SCARFACE	SPARTACUS
U_1	1			1		1
U_2		1			1	
U_3	1	1		1		
U_4			1			1
U_5				1	1	
U_6	1		1			

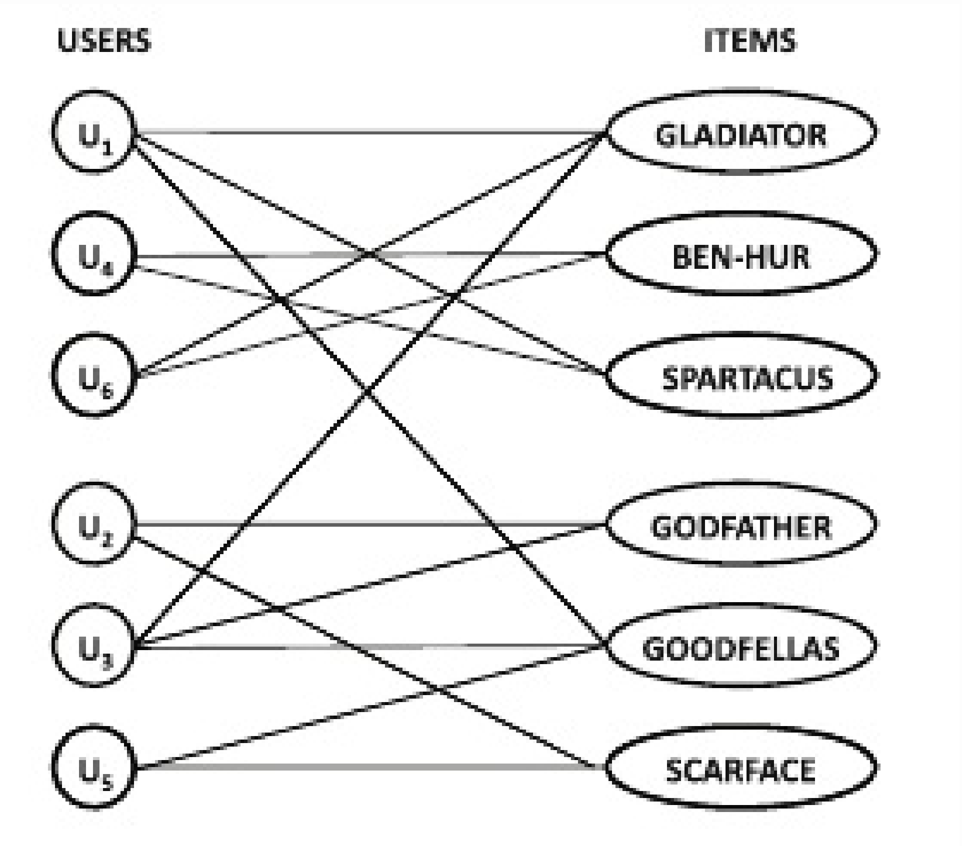


Image source Aggarwal Ch 18

Making recommendations

Let $\mathbf{G}$ be unweighted (presents only positive preferences).

Two approaches:

1. Use G only to determine nearest neighbours:

- determine K most similar users to the ith user using personalized PageRank or SimRank
- or K most similar items to the jth item
- make recommendations as before (user-based or item-based)

Making recommendations

2. Use PageRank values to decide recommendations:

i) given user i, search **item nodes** with largest PageRank values, when teleportation to user node u_i
$\rightarrow$ recommend these items to the ith user

ii) given item j, search **user nodes** with largest PageRank values, when teleportation to item node v_j
$\rightarrow$ recommend the jth item to these users

- teleportation probability α affects results
  - small α favours popular items
  - larger α makes recommendations more specific to the given user

Clustering-based methods

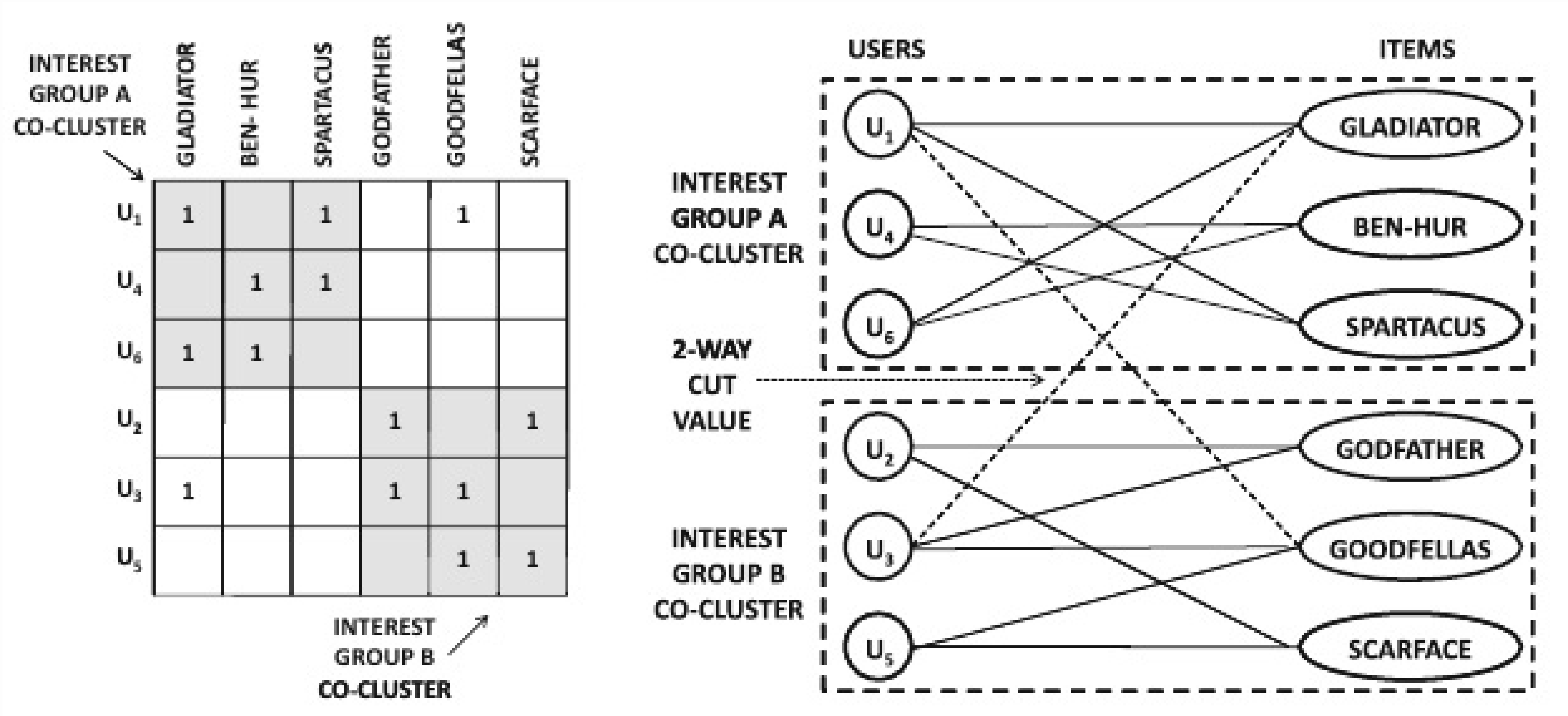
Idea: Determine peer groups (similar users or similar items) beforehand by clustering.

$\leftrightarrow$ neighbourhood-based methods determine them separately for all users

What clustering methods to use?

- problem: data sets very sparse (many missing values)
- adapt K-means:
  - calculate distances $d(\mathbf{x}, \mathbf{c}_i)$ and centroids $\mathbf{c}_i$ only over those dimensions where ratings available
- co-clustering approaches

Co-clustering of movie preference data



(a) Co-cluster

(b) User-item graph

Image source Aggarwal Ch 18

Latent factor -based methods

Idea: summarize correlations by latent factors → smaller dimensional representation of utility matrix $\mathbf{A} \approx \mathbf{F}_U \mathbf{F}_I^T$

- present n users by n k-dimensional latent factors, $\mathbf{F}_{U1}, \ldots, \mathbf{F}_{Un}$
- present d items by d k-dimensional latent factors, $\mathbf{F}_{I1}, \ldots, \mathbf{F}_{In}$
- k = new reduced dimensionality of latent representation
- estimate rating $\mathbf{A}[i, j] \approx \mathbf{F}_{Ui} \cdot \mathbf{F}_{Ij}$
- use (modified) SVD or other matrix factorization to get latent factors

Further reading Aggarwal 18.5.5 and 6.8

Summary

- Content-based recomendations: evaluate similarity between text descriptions and utilize only the user's own ratings
- Collaborative filtering: utilize all users' ratings
  - many approaches: neighbourhood-based, graph-based, clustering-based, latent factor-based
  - often 2 steps: determine a peer group and then calculate predicted ratings
  - sometimes 1 step: choose recommended items directly by PageRank or use matrix factorization

Further reading: Desrosiers and Karypis: A comprehensive survey of neighborhood-based recommendation methods. In Recommender Systems Handbook, 2011.