Stanford·CS124 | From Languages to Information (2021)

CS124(2021)· 课程资料包 @ShowMeAI

视频	课件	笔记	代码
中英双语字幕	一键打包下载	官方笔记翻译	作业项目解析

视频 ·B 站 [扫码或点击链接]
https://www.bilibili.com/video/BV1YA411w7ym

课件 & 代码 · 博客 [扫码或点击链接]
http://blog.showmeai.tech/cs124/

文本处理 编辑距离 神经嵌入 倒排索引 信息检索 序列标注 PageRank 对话系统 推荐系统 社交网络分析 协同过滤

Awesome AI Courses Notes Cheatsheets是 ShowMeAI 资料库的分支系列，覆盖最具知名度的 TOP20+ 门 AI 课程，旨在为读者和学习者提供一整套高品质中文学习笔记和速查表。

点击课程名称，跳转至课程**资料包**页面，**一键下载**课程全部资料！

机器学习	深度学习	自然语言处理	计算机视觉
Stanford · CS229	Stanford · CS230	Stanford · CS224n	Stanford · CS231n
# Awesome AI Courses Notes Cheatsheets· 持续更新中			
知识图谱	图机器学习	深度强化学习	自动驾驶
Stanford · CS520	Stanford · CS224W	UCBerkeley · CS285	MIT · 6.S094

微信公众号

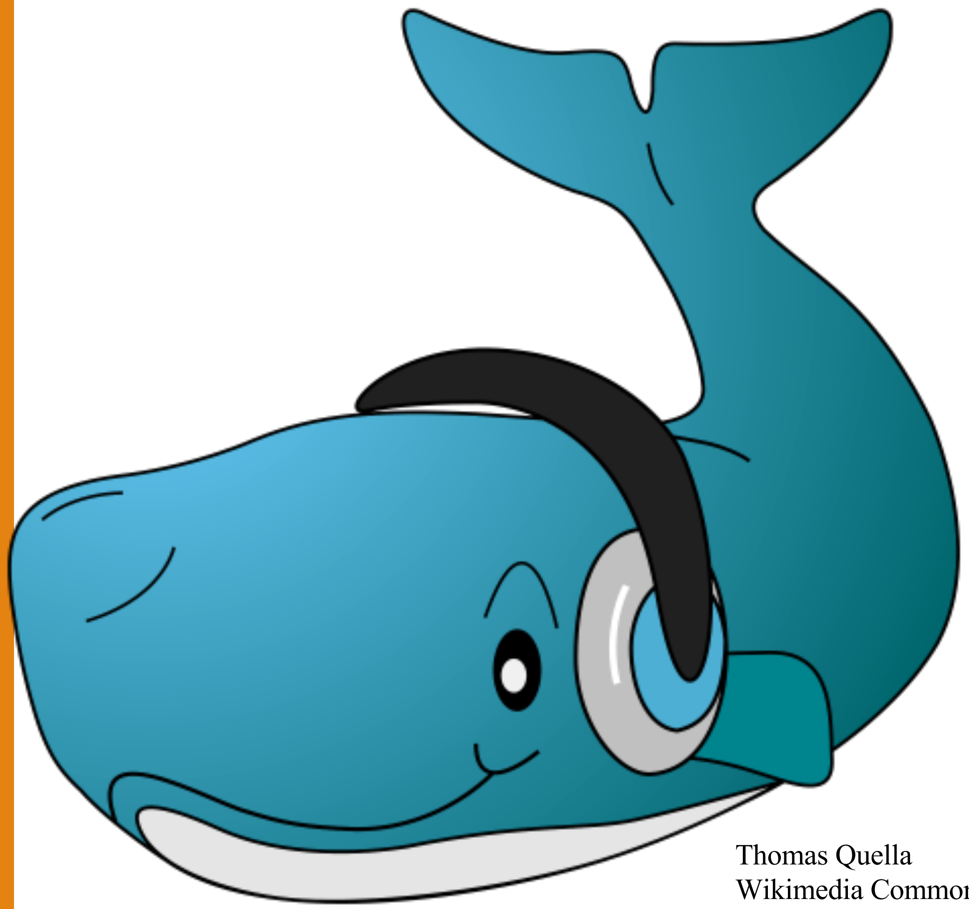
资料下载方式 2：扫码点击底部菜单栏

称为 **AI 内容创作者？** 回复 [添砖加瓦]

Recommender Systems and Collaborative Filtering

Introduction to Recommender Systems

Recommender systems: The task

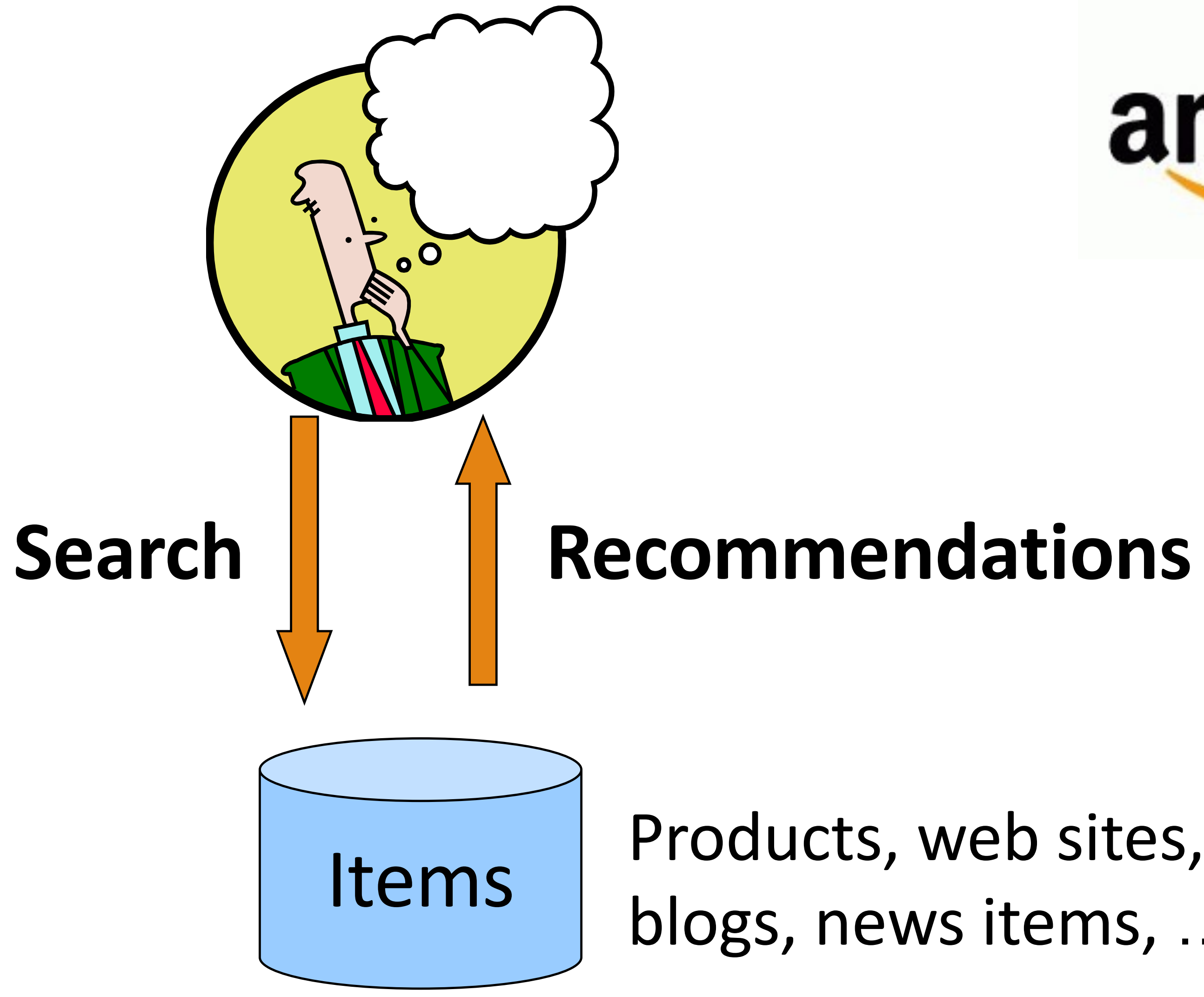


Thomas Quella
Wikimedia Commons

Plays an Ella Fitzgerald song
What should we recommend next?

Customer W

Recommendations



Products, web sites, blogs, news items, …

Types of Recommendations

Editorial and hand curated
- List of favorites
- Lists of “essential” items

Simple aggregates
- Top 10, Most Popular, Recent Uploads

Tailored to individual users
- Amazon, Netflix, Apple Music…

Today's class

Knowing how personalized recommendations work

Relevant for building practical news or product recommenders.

Relevant for understanding how misinformation spreads

QAnon Supporters And Anti-Vaxxers Are Spreading a Hoax That Bill Gates Created the Coronavirus

It has no basis in reality, but that hasn't slowed its spread across Facebook and Twitter

The Guardian

Las Vegas survivors furious as YouTube promotes clips calling shooting a hoax

'Fiction is outperforming reality': how YouTube's algorithm distorts truth

THE WALL STREET JOURNAL

How YouTube Drives People to the Internet's Darkest Corners

Google's video site often recommends divisive or misleading material, despite recent changes designed to fix the problems

Formal Model

X = set of **Users**

S = set of **Items**

Utility function *u*: *X* × *S* → *R*

- ***R*** = set of ratings
- ***R*** is a totally ordered set
- e.g., **1-5** stars, real number in **[0,1]**

Utility Matrix

		Harry Potter			Twilight	Star Wars		
		HP1	HP2	HP3	TW	SW1	SW2	SW3
Anita	A	4			5	1		
Beyonce	B	5	5	4				
Calvin	C				2	4	5	
David	D		3					3

Key Problems

1. **Gathering “known” ratings for matrix**
   - How to collect the data in the utility matrix

2. **Extrapolate unknown ratings from known ones**
   - Mainly interested in high unknown ratings
   - We are not interested in knowing what you don’t like but what you like

3. **Evaluating extrapolation methods**
   - How to measure performance of recommendation methods

(1) Gathering Ratings

Explicit

- Ask people to rate items
- Doesn’t work well in practice – people can't be bothered
- Crowdsourcing: Pay people to label items

Implicit

- Learn ratings from user actions
  - E.g., purchase (or watch video, or read article) implies high rating

(2) Extrapolating Utilities

Key problem: Utility matrix ***U*** is **sparse**

- Most people have not rated most items

- **The "Cold Start" Problem:**
  - New items have no ratings
  - New users have no history

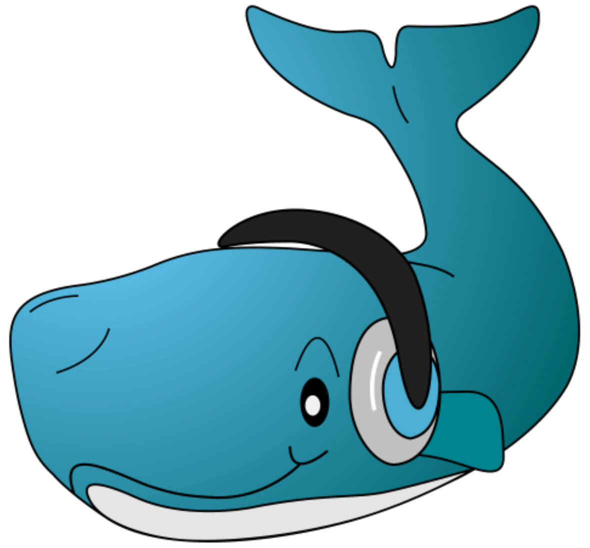
(2) Extrapolating Utilities

Three approaches to recommender systems:

1. Content-based
2. Collaborative Filtering
3. Latent factor (Neural embedding) based

$\left.\begin{matrix} 1. \\ 2. \end{matrix}\right\}$ **This lecture!**

Content-based vs. Collaborative Filtering



Thomas Quella
Wikimedia Commons

Database

- Ella Fitzgerald: Jazz, Mid-20th century, vocal legend, famous duets, …
- Louis Armstrong: Jazz, Mid-20th century, vocal legend, famous duets, …

Content-based

Suggest Louis Armstrong

Collaborative filtering

Customer W

- Plays Ella Fitzgerald
- What should we recommend next?

Customer D

- Plays Ella Fitzgerald
- Plays Louis Armstrong

Photographer: Paul Stafford for www.travelmag.com
https://www.flickr.com/photos/113306963@N05/33886542421

Recommender Systems and Collaborative Filtering

Introduction to Recommender Systems

Recommender Systems and Collaborative Filtering

Content-based Recommender Systems

Content-based Recommendations

Main idea: Recommend items to customer ***x*** similar to previous items rated highly by ***x***

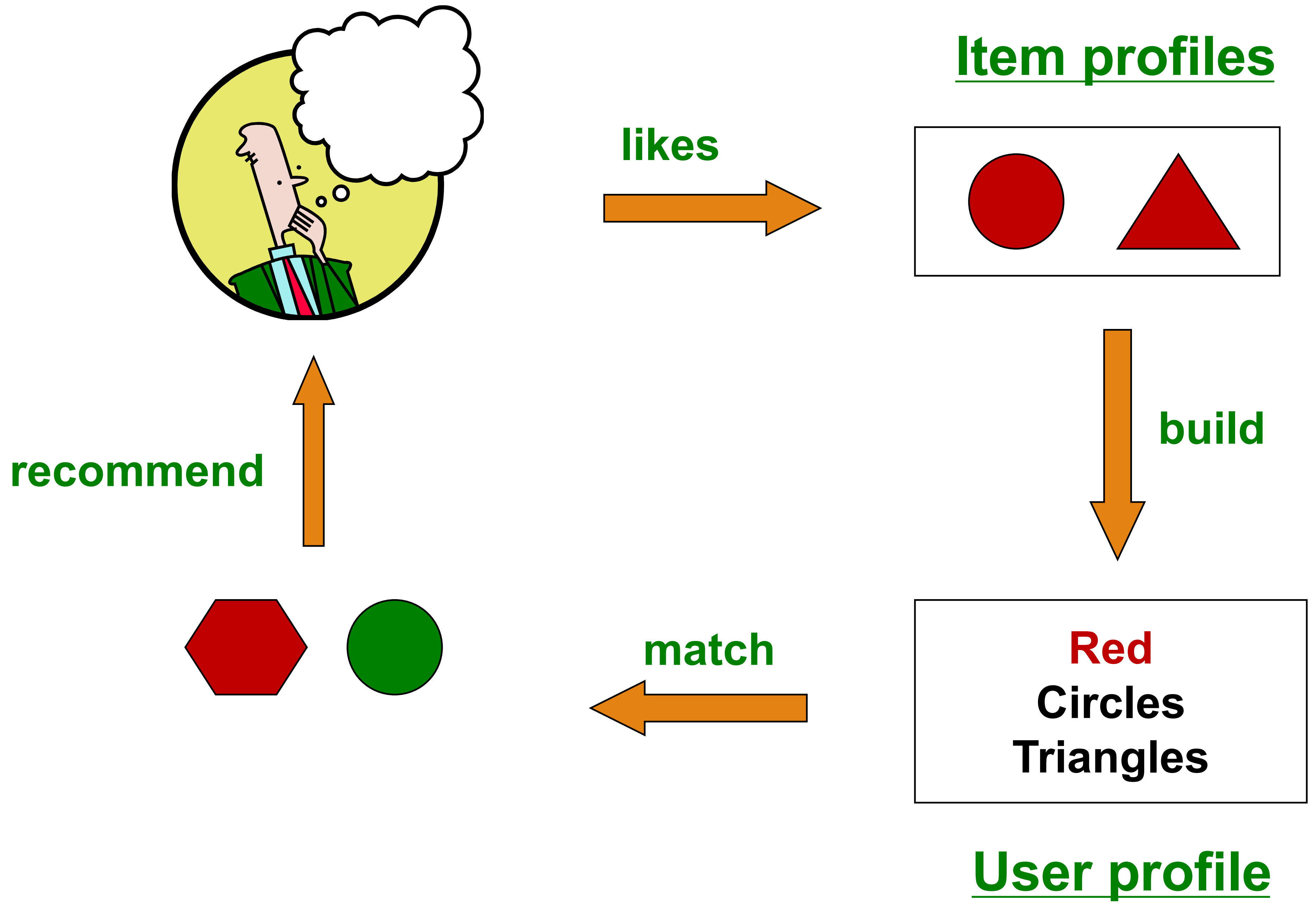
Movie recommendations

- Recommend movies with same actor(s), director, genre, …

Websites, blogs, news

- Recommend other sites with similar types or words

Plan of Action



Item Profiles

For each item, create an **item profile**

Profile is a set (vector) of features

- **Movies:** genre, director, actors, year…
- **Text:** Set of "important" words in document

How to pick important features?

- **TF-IDF** (Term frequency * Inverse Doc Frequency)
- For example use all words whose tf-idf > threshold, normalized for document length

Content-based Item Profiles

	Melissa McCarthy	Actor A	Actor B	…	Johnny Depp	Comic Genre	Spy Genre	Pirate Genre
Movie X	0	1	1	0	1	1	0	1
Movie Y	1	1	0	1	0	1	1	0

But what if we want to have real or ordinal features too?

Content-based Item Profiles

	Melissa McCarthy	Actor A	Actor B	…	Johnny Depp	Comic Genre	Spy Genre	Pirate Genre	Avg Rating
Movie X	0	1	1	0	1	1	0	1	3
Movie Y	1	1	0	1	0	1	1	0	4

For example "average rating"

Maybe we want a scaling factor α between binary and numeric features

Content-based Item Profiles

	Melissa McCarthy	Actor A	Actor B	…	Johnny Depp	Comic Genre	Spy Genre	Pirate Genre	Avg Rating
Movie X	0	1	1	0	1	1	0	1	3α
Movie Y	1	1	0	1	0	1	1	0	4α

Scaling factor α between binary and numeric features

$$\text{Cosine(Movie X, Movie Y)} = \frac{2+12\alpha^2}{\sqrt{5+9\alpha^2}\sqrt{5+16\alpha^2}}$$

$\alpha = 1$: 0.82 $\alpha = 2$: 0.94 $\alpha = 0.5$: 0.69

User Profiles

Want a vector with the same components/dimensions as items

- Could be 1s representing user purchases
- Or arbitrary numbers from a rating

User profile is aggregate of items:

- Weighted average of rated item profiles

Sample user profile

- Items are movies
- Utility matrix has 1 if user has seen movie
- 20% of the movies user U has seen have Melissa McCarthy
- U["Melissa McCarthy"] = 0.2

	Melissa McCarthy	Actor A	Actor B	…	
User U	0.2	.005	0	0	…

Prediction

- Users and items have the same dimensions!

	Melissa McCarthy	Actor A	Actor B	…	
Movie i	0	1	1	0	…
User x	0.2	.005	0	0	0

- So just recommend the items whose vectors are most similar to the user vector!

- Given user profile ***x*** and item profile ***i***,
- estimate $u(\boldsymbol{x}, \boldsymbol{i}) = \cos(\boldsymbol{x}, \boldsymbol{i}) = \frac{\boldsymbol{x} \cdot \boldsymbol{i}}{||\boldsymbol{x}|| \cdot ||\boldsymbol{i}||}$

Pros: Content-based Approach

+: No need for data on other users

- No user sparsity problems

+: Able to recommend to users with unique tastes

+: Able to recommend new & unpopular items

- No first-rater problem

+: Able to provide explanations

- Just list the content-features that caused an item to be recommended

Cons: Content-based Approach

- **Finding the appropriate features is hard**
  - E.g., images, movies, music
- **Recommendations for new users**
  - **How to build a user profile?**
- **Overspecialization**
  - Never recommends items outside user's content profile
  - People might have multiple interests
  - **Unable to exploit quality judgments of other users**

Recommender Systems and Collaborative Filtering

Content-based Recommender Systems

Recommender Systems and Collaborative Filtering

Collaborative Filtering: User-User

Collaborative filtering

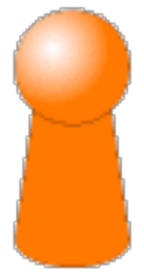
Instead of using content features of items to determine what to recommend

Find similar users and recommend items that they like!

Collaborative Filtering
Version 1: "User-User" Collaborative Filtering

Consider user ***x***

and unrated item ***i***



Find set ***N*** of other users whose ratings are "**similar**" to ***x***'s ratings

Estimate ***x***'s ratings for ***i*** based on ratings for ***i*** of users in ***N***

Collaborative filtering

Find similar users and recommend items that they like:

- Represent users by their rows in the **utility matrix**
- Two users are similar if their vectors are similar!

	Harry Potter			Twilight	Star Wars		
	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Finding Similar Users

Let $\boldsymbol{r_x}$ be the vector of user ***x***'s ratings

$\boldsymbol{r_x}$ = [*, _, _, *, ***]
$\boldsymbol{r_y}$ = [*, _, **, **, _]

$\boldsymbol{r_x} = \{1, 0, 0, 1, 3\}$
$\boldsymbol{r_y} = \{1, 0, 2, 2, 0\}$

Cosine similarity measure

- sim(***x***, ***y***) = cos($\boldsymbol{r_x}$, $\boldsymbol{r_y}$) = $\frac{r_x \cdot r_y}{||r_x||\ ||r_y||}$

Problem: This representation leads to unintuitive results

Problems with raw utility matrix cosine

	Harry Potter			**Twilight**	**Star Wars**		
	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Intuitively we want: sim(*A*, *B*) > sim(*A*, *C*)

$$\text{sim(A,B)} = \frac{4 \times 5}{\sqrt{4^2+5^2+1^2}\sqrt{5^2+5^2+4^2}} = 0.380$$

$$\text{sim(A,C)} = \frac{5 \times 2 + 1 \times 4}{\sqrt{4^2+5^2+1^2}\sqrt{2^2+4^2+5^2}} = 0.322$$

Yes, 0.380 > 0.322

But only barely works…

Problem with raw cosine

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- Problem with cosine:
  - C really loves SW
  - A hates SW
  - B just hasn't seen it
- Another problem: we'd like to normalize the raters
  - D rated everything the same; not very useful

Mean-Centered Utility Matrix: subtract the means of each row

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	$2/3$			$5/3$	$-7/3$		
B	$1/3$	$1/3$	$-2/3$				
C				$-5/3$	$1/3$	$4/3$	
D		0					0

- Now a 0 means no information
- And negative ratings means viewers with opposite ratings will have vectors in opposite directions!

Modified Utility Matrix: subtract the means of each row

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	$2/3$			$5/3$	$-7/3$		
B	$1/3$	$1/3$	$-2/3$				
C				$-5/3$	$1/3$	$4/3$	
D		0					0

$$\text{Cos(A,B)} = \frac{(2/3)\times(1/3)}{\sqrt{(2/3)^2+(5/3)^2+(-7/3)^2}\sqrt{(1/3)^2+(1/3)^2+(-2/3)^2}} = 0.092$$

$$\text{Cos(A,C)} = \frac{(5/3)\times(-5/3)+(-7/3)\times(1/3)}{\sqrt{(2/3)^2+(5/3)^2+(-7/3)^2}\sqrt{(-5/3)^2+(1/3)^2+(4/3)^2}} = -0.559$$

Now A and C are (correctly) way further apart than A,B

Terminological Note: subtracting the mean is **mean-centering**, not **normalizing**

(normalizing is dividing by a norm to turn something into a probability), but the textbook (and common usage) sometimes overloads the term “normalize”

Finding similar users with overlapping-item mean-centering

Let $\boldsymbol{r_x}$ be the vector of user ***x***'s ratings

$\boldsymbol{r_x}$ = {1, 0, 0, 1, 3}

$\boldsymbol{r_y}$ = {1, 0, 2, 2, 0}

$\boldsymbol{r_x}$ = [*, _, _, *, ***]

$\boldsymbol{r_y}$ = [*, _, **, **, _]

Mean-centering:

- For each user x, let $\overline{r_x}$ be mean of r_x (ignoring missing values)
- $\overline{r_x}$ = (1 + 1 + 3)/3 = 5/3 $\overline{r_y}$ = (1 + 2 + 2)/3 = 5/3
- Subtract this average from each of their ratings
  - (but do nothing to the "missing values"; they stay "null").
  - mean centered r_x = {-2/3, 0, 0, -2/3, 4/3}

One new idea: Keep only items they both rate (unlike 2 slides ago)

$\boldsymbol{r_x}$ = {-2/3, [redacted], [redacted] -2/3, [redacted]} $\boldsymbol{r_y}$ = {-2/3, [redacted], [redacted], 1/3, [redacted]}

$\boldsymbol{r_x}$ = {-2/3, -2/3} $\boldsymbol{r_y}$ = {-2/3, 1/3}

Now take cosine:

- Now compute cosine between user vectors
  - cos([-2/3, -2/3], [-2/3, 1/3])

Mean-centered overlapping-item cosine similarity

Let $\boldsymbol{r_x}$ be the vector of user $\boldsymbol{x}$'s ratings,and $\overline{r_x}$ be its mean (ignoring missing values)

Instead of basic cosine similarity measure

- sim($\boldsymbol{x}$, $\boldsymbol{y}$) = cos($\boldsymbol{r_x}$, $\boldsymbol{r_y}$) = $\frac{r_x \cdot r_y}{||r_x||\ ||r_y||}$

Mean-centered overlapping-item cosine similarity

(Variant of Pearson correlation)

- $\boldsymbol{S_{xy}}$ = items rated by both users $\boldsymbol{x}$ and $\boldsymbol{y}$

$$\boldsymbol{sim(x,y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})(r_{ys} - \overline{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})^2} \sqrt{\sum_{s \in S_{xy}} (r_{ys} - \overline{r_y})^2}}}$$

Rating Predictions

From similarity metric to recommendations for an unrated item *i*:

Let $\boldsymbol{r_x}$ be the vector of user ***x***'s ratings

Let ***N*** be the set of ***k*** users most similar to ***x*** who have rated item ***i***

Prediction for item *i* of user *x*:

- Rate i as the mean of what k-people-like-me rated i

$$r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$$

- Even better: Rate i as the mean weighted by their similarity to me …

$$r_{xi} = \frac{\sum_{y \in N} s_{xy}\, r_{yi}}{\sum_{y \in N} s_{xy}}$$

Shorthand:

$$\boldsymbol{s_{xy} = sim(x, y)}$$

- **Many other tricks possible…**

Recommender Systems and Collaborative Filtering

Collaborative Filtering: User-User

Recommender Systems and Collaborative Filtering

Collaborative Filtering: Item-Item

Collaborative Filtering Version 2: Item-Item Collaborative Filtering

So far: User-user collaborative filtering

Alternate view that often works better: Item-item

- For item ***i***, find other similar items
- Estimate rating for item ***i*** based on ratings for those similar items
- Can use same similarity metrics and prediction functions as in user-user model
- "Rate i as the mean of my ratings for other items, weighted by their similarity to i"

$$\boxed{r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \quad r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}}$$

N(i;x)…set of items rated by ***x*** *and* similar to ***i***
s_{ij}… similarity of items ***i*** and ***j***
r_{xj}…rating of user ***x*** on item ***j***

Item-Item CF (|N|=2)

users

movies	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3			5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

⬜ - unknown rating
🟨 - rating between 1 to 5

Item-Item CF (|N|=2)

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

🟥 - estimate rating of movie **1** by user **5**

Item-Item CF (|N|=2)

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12	sim(1,m)
1	1		3		?	5			5		4		1.00
2			5	4			4			2	1	3	..
3	2	4		1	2		3		4	3	5		?
4		2	4		5			4			2		..
5			4	3	4	2					2	5	..
6	1		3		3			2			4		

Neighbor selection:
Identify movies similar to movie **1**, **rated by user 5**

Here we use mean centered item-overlap cosine as similarity:
1) Subtract mean rating m_i from each movie i between rows
2) Compute (item-overlapping) cosine similarities

Item-Item CF (|N|=2)

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		?	5			5		4	

Subtract mean rating m_i from each movie i

$m_1 = (1+3+5+5+4)/5 = 18/5$

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	-13/5		-3/5		?	7/5			7/5		2/5	
3	-1	1		-2	-1		0		1	0	2	
6	-8/5		2/5		2/5			-3/5			7/5	

Showing computation only for #3 and #6

Neighbor selection:
Identify movies similar to movie **1**, **rated by user 5**

Here we use mean centered item-overlap cosine as similarity:
1) Subtract mean rating m_i from each movie i
2) Compute (item-overlapping) cosine similarities between rows

Item-Item CF (|N|=2)

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12	sim(1,m)
1	-13/5		-3/5		?	7/5			7/5		2/5		1.00
2			5	4			4			2	1	3	..
3	-1	1		-2	-1		0		1	0	2		?
4		2	4		5			4			2		..
5			4	3	4	2					2	5	..
6	-8/5		2/5		2/5			-3/5			7/5		?

Neighbor selection:
Identify movies similar to movie **1**, **rated by user 5**

Here we use mean centered item-overlap cosine as similarity:
1) Subtract mean rating ***m_i*** from each movie ***i***
2) Compute (item-overlapping) cosine similarities between rows

Compute Cosine Similarity:

For rows 1 and 3, they both have values for users 1, 9 and 11.

$$\text{sim}(1, 3) = \frac{(-13/5)(-1)+(7/5)(1)+(2/5)(2)}{\sqrt{(-13/5)^2+(7/5)^2+(2/5)^2} \cdot \sqrt{(-1)^2+(1)^2+(2)^2}} \approx 0.658$$

For rows 1 and 6, they both have values for users 1, 3 and 11.

$$\text{sim}(1, 6) = \frac{(-13/5)(-8/5)+(-3/5)(2/5)+(2/5)(7/5)}{\sqrt{(-13/5)^2+(-3/5)^2+(2/5)^2} \cdot \sqrt{(-8/5)^2+(2/5)^2+(7/5)^2}} \approx 0.768$$

Item-Item CF (|N|=2)

movies	users												
	1	2	3	4	5	6	7	8	9	10	11	12	sim(1,m)
1	1		3		?	5			5		4		1.000
2			5	4			4			2	1	3	..
<u>3</u>	2	4		1	2		3		4	3	5		<u>.658</u>
4		2	4		5			4			2		..
5			4	3	4	2					2	5	..
<u>6</u>	1		3		3			2			4		<u>.768</u>

Compute similarity weights:

$s_{1,3}$=.658, $s_{1,6}$=.768 (we compute $s_{1,2}$, $s_{1,4}$, $s_{1,5}$ too; let's assume those are smaller)

Item-Item CF (|N|=2)
Approximate rating with weighted mean

users

movies	1	2	3	4	5	6	7	8	9	10	11	12	sim(1,m)
1	1		3		**2.54**	5			5		4		**1.000**
2			5	4			4			2	1	3	..
3	2	4		1	2		3		4	3	5		.658
4		2	4		5			4			2		..
5			4	3	4	2					2	5	..
6	1		3		3			2			4		.768

Predict by taking weighted average:

$$r_{ix} = \frac{\sum_{j \in N(i;x)} s_{ij} \, r_{jx}}{\sum s_{ij}}$$

$r_{1,5}$ = (0.658*2 + 0.768*3) / (0.658+0.768) = 2.54

Item-Item vs. User-User

- **In practice, item-item often works better than user-user**
- **Why?** Items are simpler, users have multiple tastes
  - (People are more complex than objects)

Pros/Cons of Collaborative Filtering

+ Works for any kind of item

- No feature selection needed

- Cold Start:

- Need enough users in the system to find a match

- Sparsity:

- The user/ratings matrix is sparse
- Hard to find users that have rated the same items

- First rater:

- Cannot recommend an item that has not been previously rated

- Popularity bias:

- Cannot recommend items to someone with unique taste
- Tends to recommend popular items

- Ethical and social issues:

- Can lead to filter bubbles and radicalization spirals

Recommender Systems and Collaborative Filtering

Collaborative Filtering: Item-Item

Recommender Systems and Collaborative Filtering

Simplified item-item similarity computation for our tiny PA6 dataset

Simplified item-item for our tiny PA6 dataset

First, assume you've converted all the values to

+1 (like),

0 (no rating)

–1 (dislike)

users

movies	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3			5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

Simplified item-item for our tiny PA6 dataset

First, assume you've converted all the values to

+1 (like),

0 (no rating)

−1 (dislike)

users

movies

	1	2	3	4	5	6	7	8	9	10	11	12
1	-1		1			1			1		1	
2			1	1			1			-1	-1	1
3	-1	1		-1	-1		1		1	1	1	
4		-1	1		1			1			-1	
5			1	1	1	-1					-1	1
6	-1		1		1			-1			1	

Simplified item-item for our tiny PA6 dataset

Assume you've **binarized**, i.e. converted all the values to

- +1 (like), 0 (no rating) −1 (dislike)

For this binary case, some tricks that the TAs recommend:

- **Don't mean-center users, just keep the raw +1,0,-1**
- **Don't normalize (i.e. don't divide the product by the sum)**
- **i.e., instead of this:**

$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij}\, r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

- **Just do this:**

$$r_{xi} = \sum_{j \in N(i;x)} s_{ij}\, r_{xj}$$

s_{ij}… similarity of items ***i*** and ***j***
r_{xj}…rating of user ***x*** on item ***j***
N(i;x)…set of items rated by ***x***

- **Don't use mean-centered item-overlap cosine to compute s_{ij}**
  - **Just use cosine**

Simplified item-item for our tiny PA6 dataset

1. binarize, i.e. convert all values to
   - +1 (like), 0 (no rating) −1 (dislike)
2. The user x gives you (say) ratings for 2 movies m1 and m2
   - ***r_{xj}***…rating of user ***x*** on item ***j***
3. For each movie *i* in the dataset
   - $r_{xi} = \sum_{j \in (m1, m2)} s_{ij}\, r_{xj}$
   - Where ***s_{ij}***… cosine between vectors for movies ***i*** and ***j***
4. Recommend the movie *i* with max r_{xi}

Recommender Systems and Collaborative Filtering

Simplified item-item similarity computation for our tiny PA6 dataset

Recommender Systems and Collaborative Filtering

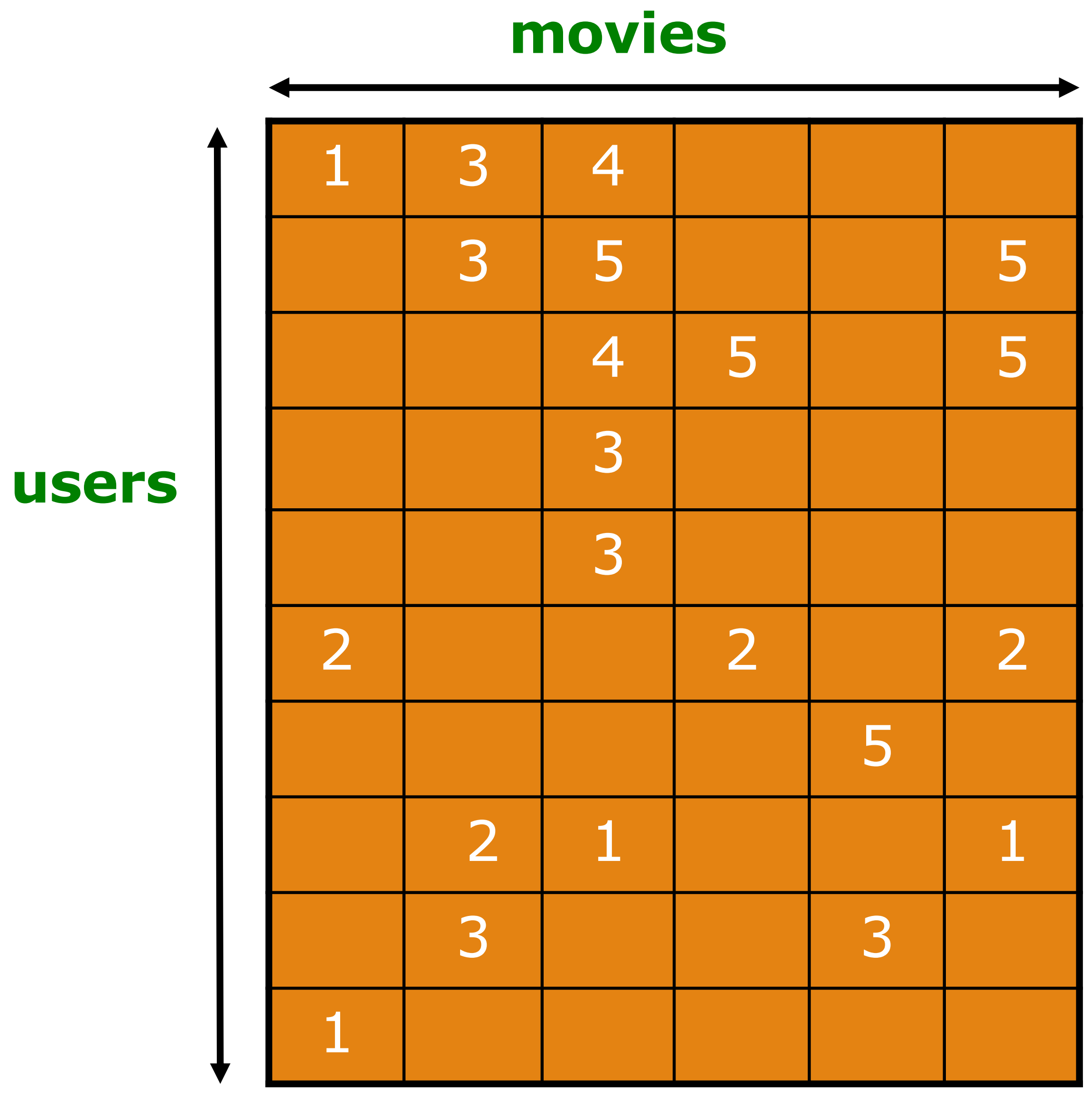
Evaluation and Implications

YouTube's Recommendation Algorithm

Covington, Adams, Sargin 2016. Deep Neural Networks for YouTube Recommendations

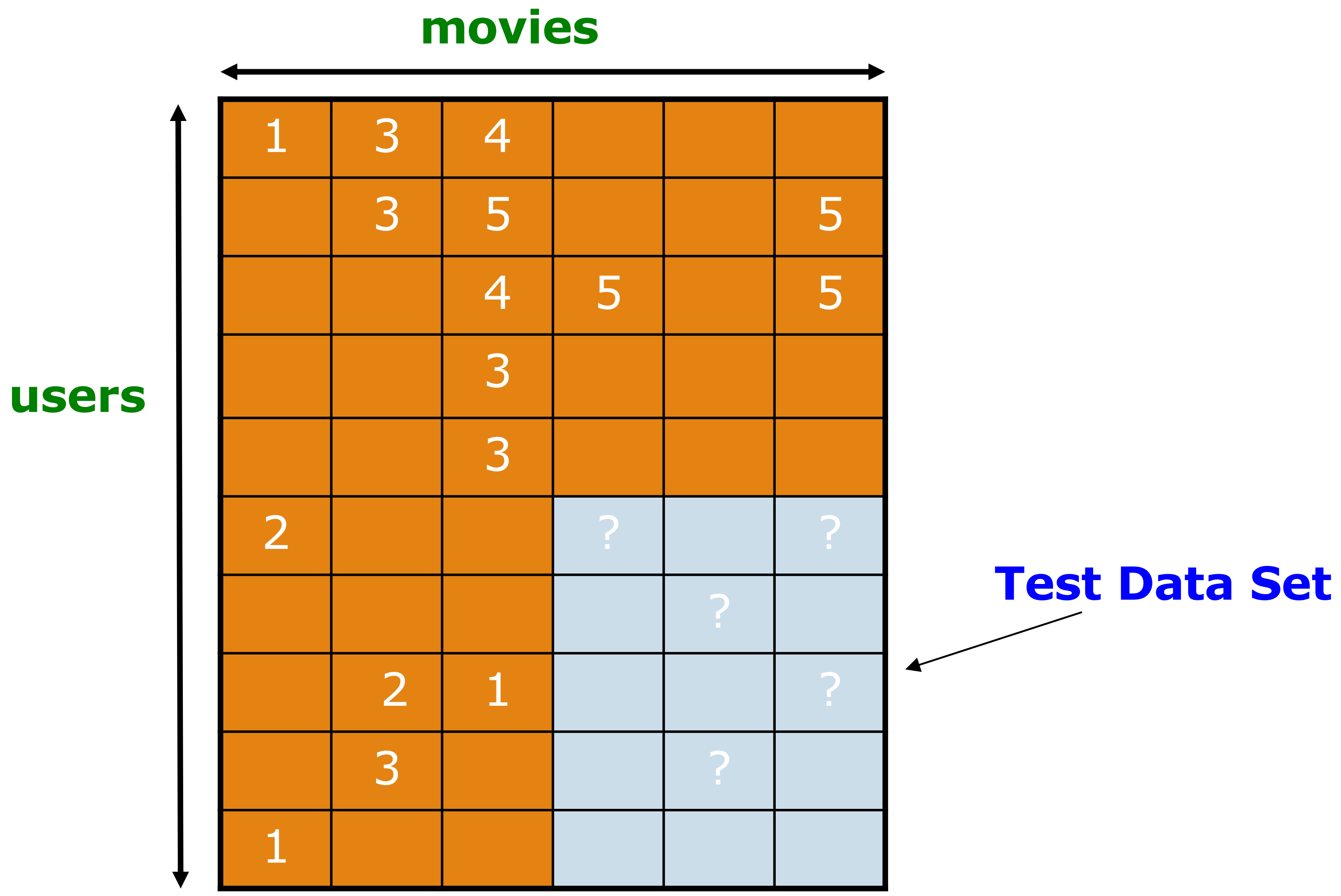
1. Represent each video and user as an embedding
2. Train a huge neural net classifier (softmax over millions of possible videos) to predict the next video the user will watch
3. Input features:
   - User's watch history (video ids)
   - User's recent queries (word embeddings)
   - Date, popularity, virality of video
4. Learn **embeddings** for **videos** and **users** in training

Evaluation


movies
users

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

Evaluation



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

Evaluating Predictions

Compare predictions with known ratings

- **Root-mean-square error** (RMSE)
  - $\sqrt{\frac{\sum_{xi}\left(r_{xi}-r_{xi}^{*}\right)^{2}}{N}}$
  - where $\boldsymbol{r_{xi}}$ is predicted, $\boldsymbol{r_{xi}^{*}}$ is the true rating of $\boldsymbol{x}$ on $\boldsymbol{i}$
- **Rank Correlation**:
  - Spearman's *correlation* between system's and user's complete rankings

But is predicting watching the right loss function?

What could go wrong? Ethical and societal implications in recommendation engines.

Milano, Silvia, Mariarosaria Taddeo, and Luciano Floridi. "Recommender systems and their ethical challenges." *AI & SOCIETY* 35, no. 4 (2020): 957-967.

- Spread of misinformation and propaganda
- Filter bubbles
- Inappropriate or unethical content
- Opacity
- Violating user privacy

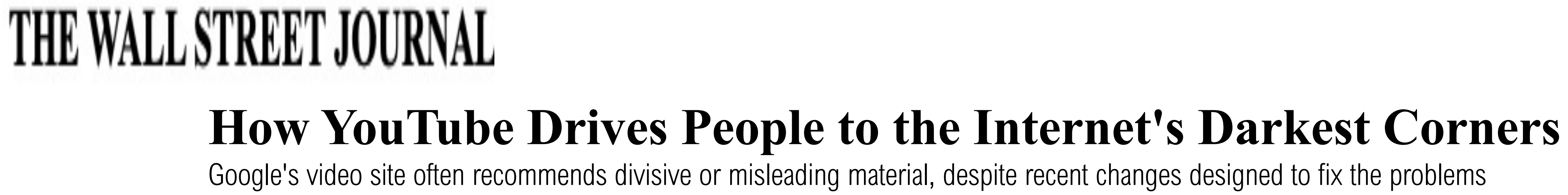
What could go wrong? Ethical and societal implications

Howard, Ganesh, Lioustiou. 2019. The IRA, Social Media, and Political Polarization in the United States, 2012-2018

Propaganda campaigns

- Russia Internet Research Agency (IRA)
  - attack on the United States 2013-2018
  - computational propaganda on YouTube, Facebook, Instagram, to misinform/polarize US voters.
  - Goal: induce African American, Mexican American voters to boycott elections

Ethical and societal implications: Filter bubbles



THE WALL STREET JOURNAL

How YouTube Drives People to the Internet's Darkest Corners

Google's video site often recommends divisive or misleading material, despite recent changes designed to fix the problems

"I realized really fast that YouTube's recommendation was putting people into filter bubbles," Chaslot said. "There was no way out. **If a person was into Flat Earth conspiracies, it was bad for watch-time to recommend anti-Flat Earth videos, so it won't even recommend them**."

"The question before us is the ethics of leading people down hateful rabbit holes full of misinformation and lies at scale just because it works to increase the time people spend on the site – and it does work"

- – Zeynep Tufekci

Open research questions

What would algorithms look like that could recommend but also include these social costs?

Recommender Systems and Collaborative Filtering

Evaluation and Implications

Stanford·CS124 | From Languages to Information (2021)

CS124(2021)· 课程资料包 @ShowMeAI

视频	课件	笔记	代码
中英双语字幕	一键打包下载	官方笔记翻译	作业项目解析

视频 ·B 站 [扫码或点击链接]
https://www.bilibili.com/video/BV1YA411w7ym

课件 & 代码 · 博客 [扫码或点击链接]
http://blog.showmeai.tech/cs124/

文本处理 信息检索 编辑距离 神经嵌入 序列标注 推荐系统 PageRank 社交网络分析 倒排索引 对话系统 协同过滤

Awesome AI Courses Notes Cheatsheets 是 ShowMeAI 资料库的分支系列，覆盖最具知名度的 **TOP20+** 门 AI 课程，旨在为读者和学习者提供一整套高品质中文学习笔记和速查表。

点击课程名称，跳转至课程**资料包**页面，**一键下载**课程全部资料！

机器学习	深度学习	自然语言处理	计算机视觉
Stanford · CS229	Stanford · CS230	Stanford · CS224n	Stanford · CS231n
# Awesome AI Courses Notes Cheatsheets· 持续更新中			
知识图谱	**图机器学习**	**深度强化学习**	**自动驾驶**
Stanford · CS520	Stanford · CS224W	UCBerkeley · CS285	MIT · 6.S094

微信公众号

资料下载方式 2：扫码点击底部菜单栏

称为 **AI 内容创作者？** 回复 [添砖加瓦]