Recommender Systems

Data Mining CSE2525

Nergis Tomen

09.01.2025

Overview

- **Recommender systems**
    - Collaborative filtering & the utility matrix
    - NMF for data imputation
    - Cross-validation
    - Performance metrics
    - Privacy
    - Alternatives to NMF

Recommender systems

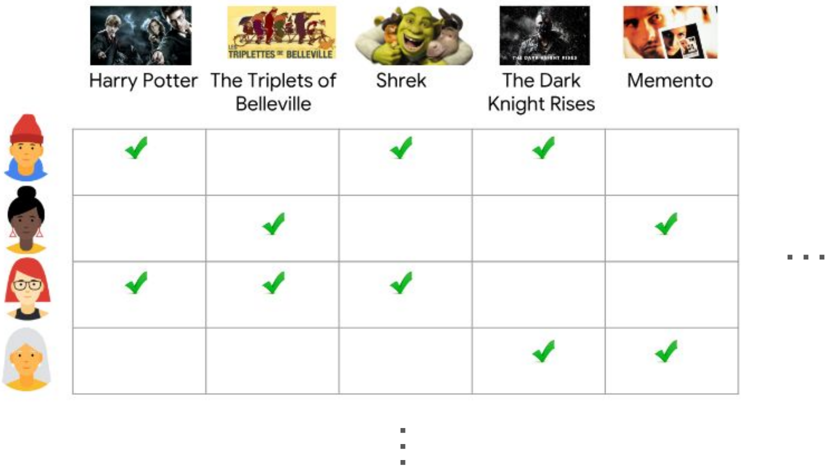
Consider a streaming service:

Simple picture → 4 subscribers, 5 movies… technically can recommend all movies to all subscribers.

Recommender systems

Consider a streaming service:

Real picture → 200 million subscribers, 10,000 movies.



Recommender systems: Data imputation

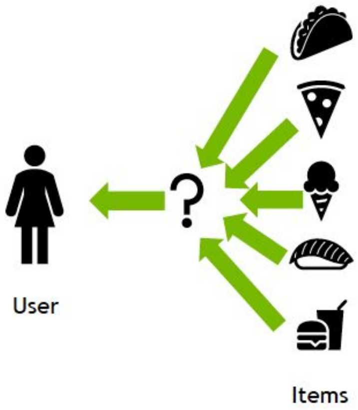
How to find the missing values? (a.k.a. the 'data imputation' problem.)

	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

Recommender systems

The aim of a recommender system is to **provide suggestions** for items that are most likely to be interesting to a user.



Recommender systems

The aim of a recommender system is to **provide suggestions** for items that are most likely to be interesting to a user.

Providers:

- have thousands/millions of items on offer

Users:

- have limited time

- have limited budget

- cannot watch, buy, eat everything

- may want to discover something new

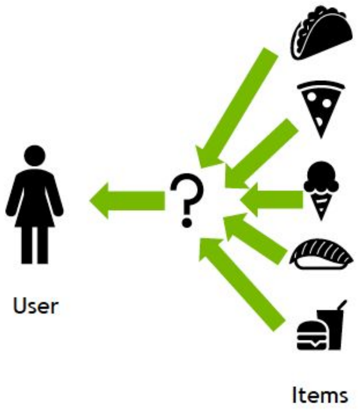
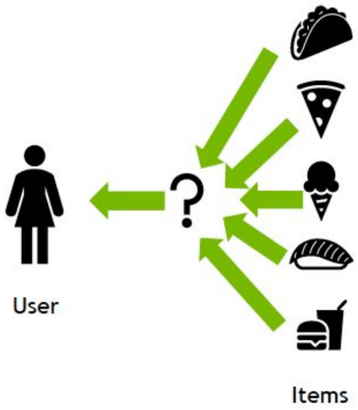


Image source: https://www.nvidia.com/en-us/glossary/data-science/recommendation-system/

Recommender systems

What are some examples of recommender systems you know?



Recommender systems

Used for recommending, for example:

- **Movies**, shows, **music** in streaming services

- **Products** to purchase in e-commerce

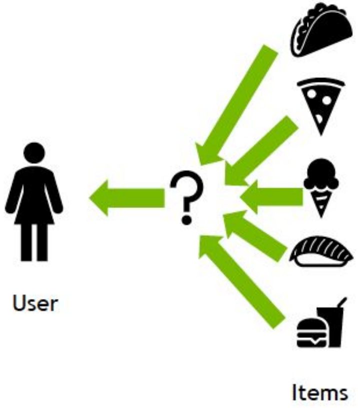


Image source: https://www.ncsc.gov.uk/guidance/social-media-how-to-use-it-safely, https://www.linkedin.com/pulse/ai-restaurant-recommendations-future-food-critics-josh-sapienza

Recommender systems

Used for recommending, for example:

- **Movies**, shows, **music** in streaming services

- **Products** to purchase in e-commerce

- Articles to read in online news sites

- **Content** in social media

- **Books** to buy or audiobooks to listen to

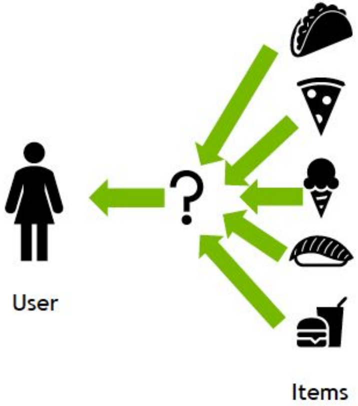


Image source: https://www.ncsc.gov.uk/guidance/social-media-how-to-use-it-safely, https://www.linkedin.com/pulse/ai-restaurant-recommendations-future-food-critics-josh-sapienza

Recommender systems

Used for recommending, for example:

- **Movies**, shows, **music** in streaming services

- **Products** to purchase in e-commerce

- Articles to read in online news sites

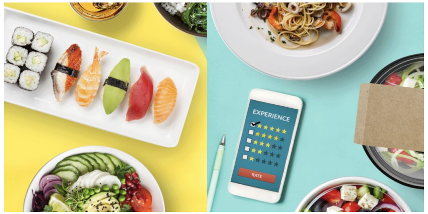
- **Content** in social media

- **Books** to buy or audiobooks to listen to

- Restaurants in a user's vicinity

- Potential **dates** in online dating

- etc…



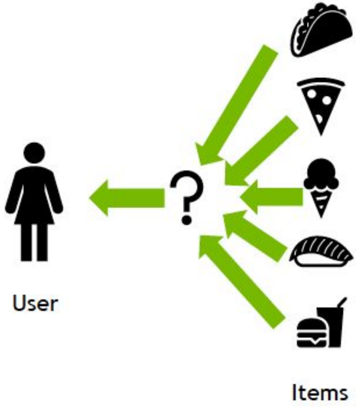


Image source: https://www.ncsc.gov.uk/guidance/social-media-how-to-use-it-safely, https://www.linkedin.com/pulse/ai-restaurant-recommendations-future-food-critics-josh-sapienza

Recommender systems

Used for recommending, for example:

- **Movies**, shows, **music** in streaming services

- **Products** to purchase in e-

- Articles to read in online ne

- **Content** in social media

- **Books** to buy or audiobooks

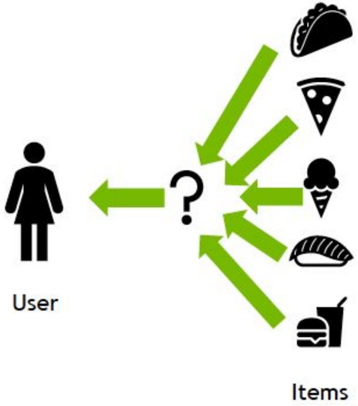
- Restaurants in a user's vicinity

- Potential **dates** in online dating

- etc…

2 popular approaches:

- Collaborative filtering
- Content-based filtering



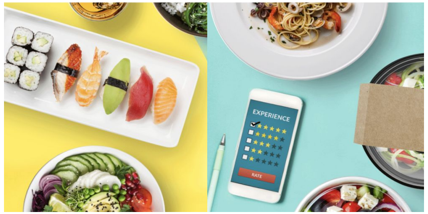


Image source: https://www.ncsc.gov.uk/guidance/social-media-how-to-use-it-safely, https://www.linkedin.com/pulse/ai-restaurant-recommendations-future-food-critics-josh-sapienza

Collaborative filtering

"Collaborative filtering, is the leveraging of **user preferences** in the form of ratings or buying behavior in a “collaborative” way, … to determine either **relevant users** for specific items, or **relevant items** for specific users in the recommendation process."

Data Mining the Textbook, Chapter 18.5

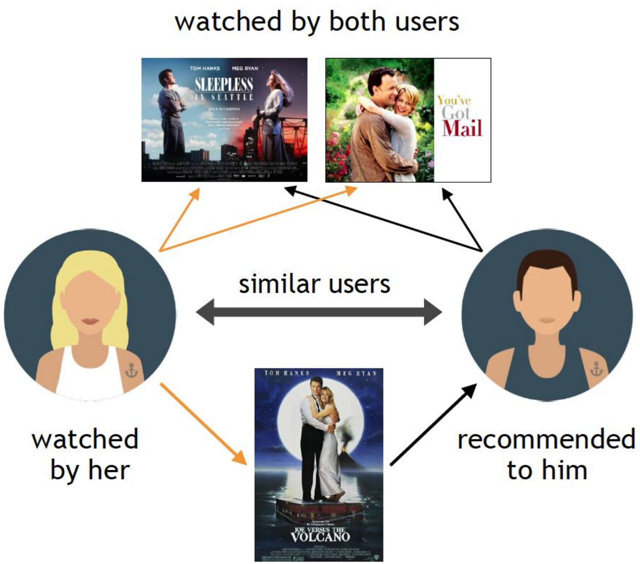


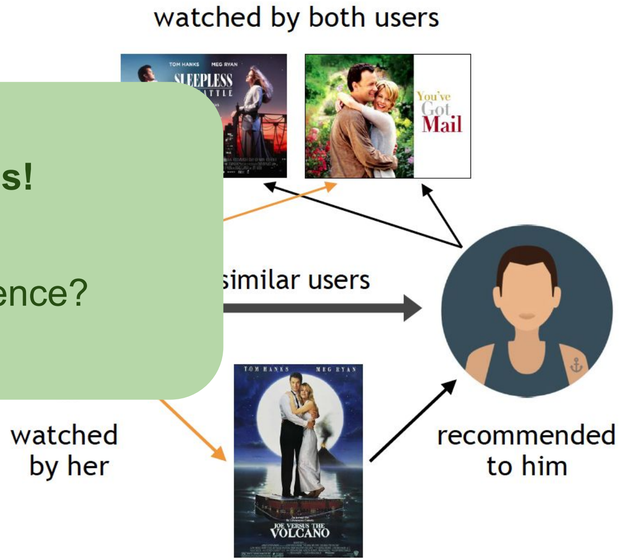
Image source: https://www.nvidia.com/en-us/glossary/data-science/recommendation-system/

Collaborative filtering

"Collaborative filtering, is the leveraging of **user preferences** in the f
buying behavior in a "collab
determine either **relevant**
items, or **relevant items** fo
the recommendatio

This is what NMF does!

Remember linear dependence?



Data Mining the Textbook, Chapter 18.5

Content-based filtering

"**Users** and **items** are both associated with **feature-based descriptions**. For example, … the text of the item description (meta-data). A user might also have explicitly specified their interests in their profile (knowledge-based)… or can be **inferred from** their buying or browsing **behavior**."

Data Mining the Textbook, Chapter 18.5

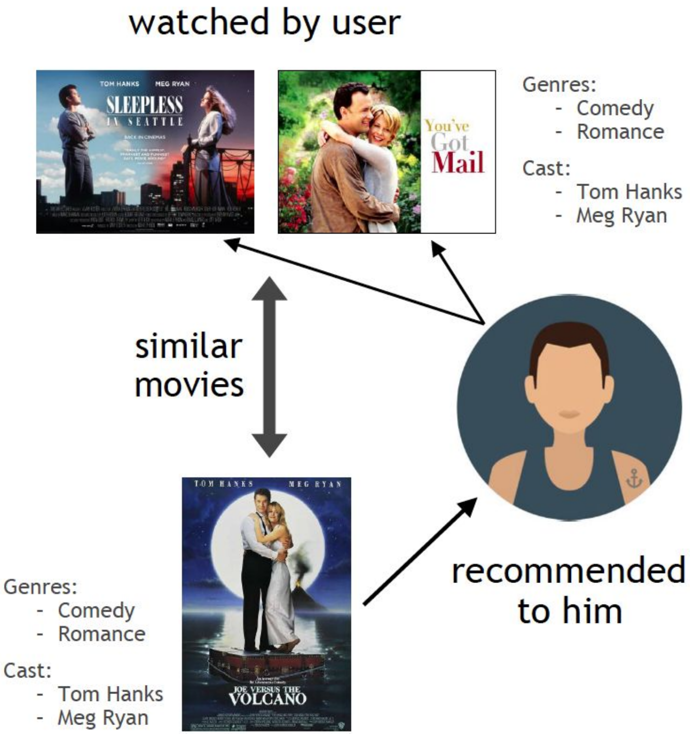
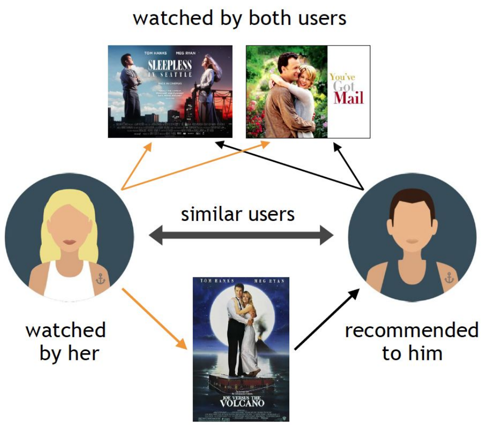


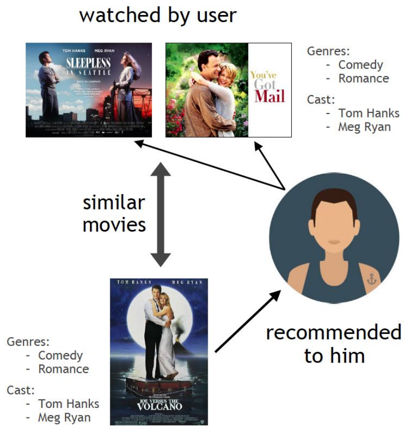
Image source: https://www.nvidia.com/en-us/glossary/data-science/recommendation-system/

Recommender systems

Collaborative Filtering



Content-based Filtering



Often the two approaches are combined.

Image source: https://www.nvidia.com/en-us/glossary/data-science/recommendation-system/

Questions?

Long-tail phenomenon of online recommender systems

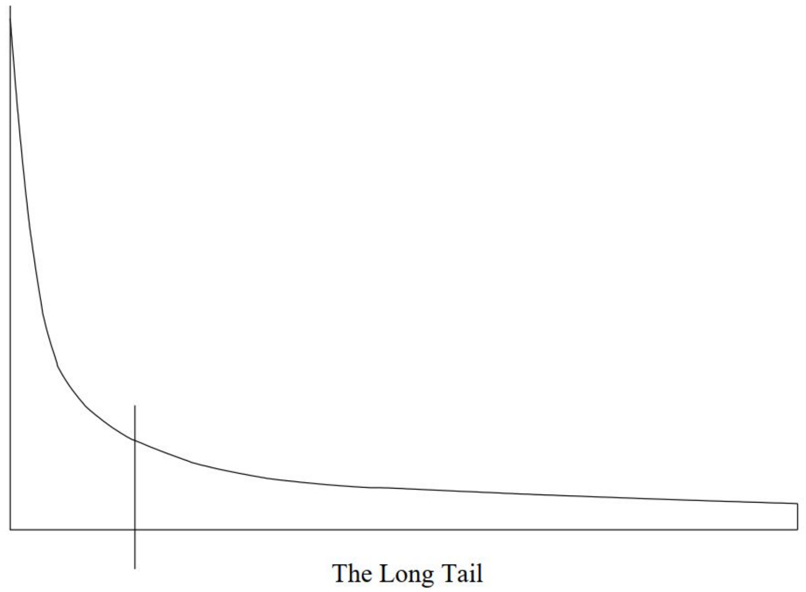


Figure 9.2: The long tail: physical institutions can only provide what is popular, while on-line institutions can make everything available

Long-tail phenomenon of online recommender systems

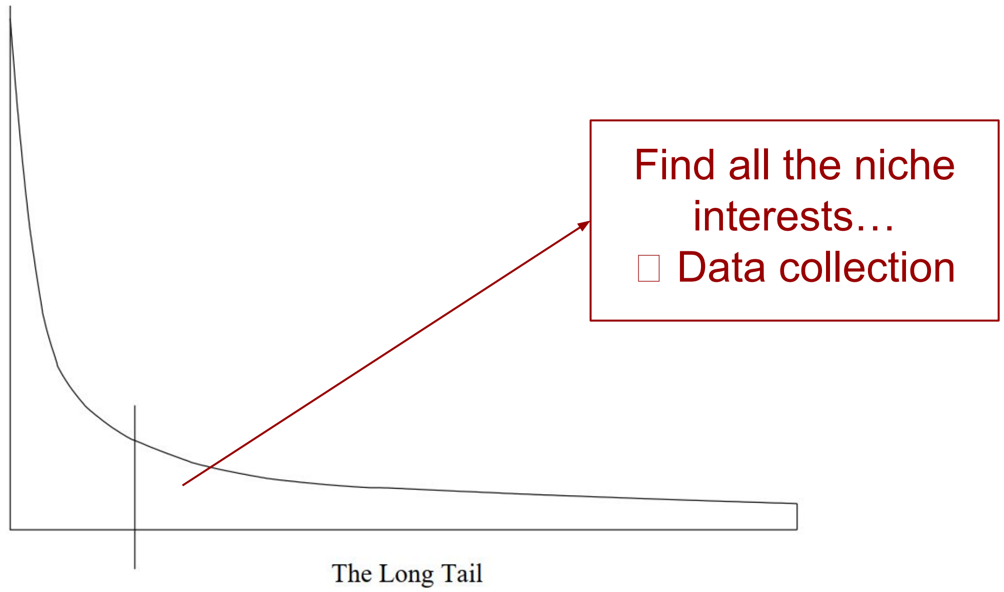


Figure 9.2: The long tail: physical institutions can only provide what is popular, while on-line institutions can make everything available

Utility matrix

Online services collect an increasingly **large amount of information** about user behaviour!

There are 2 main types of utility matrices.

Utility matrix

There are 2 main types of utility matrices.

Items

Users

	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

Image source: "Data Mining The Textbook" by Aggarwal, Chapter 18.5

Utility matrix

There are 2 main types of utility matrices.

Items

Users gave ratings (explicitly collected)

Users

	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

Users watched a movie (implicitly collected)

Image source: "Data Mining The Textbook" by Aggarwal, Chapter 18.5

Utility matrix

There are 2 main types of utility matrices.

Which one do you think is better?

Items

Users gave ratings (explicitly collected)

Users

	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

Users watched a movie (implicitly collected)

Image source: "Data Mining The Textbook" by Aggarwal, Chapter 18.5

Utility matrix

There are 2 main types of utility matrices.

	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

Utility matrix

There are 2 main types of utility matrices.

Might be harder to collect

	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

Might be harder to work with

Collaborative filtering

watched by user

Genres:
- Comedy
- Romance

Cast:
- Tom Hanks
- Meg Ryan

Advantages:

- **Doesn't rely on** hand-crafted or machine-extracted (e.g. from text) **'content'** (both prone to errors)

Collaborative filtering

Advantages:

- **Doesn't rely on** hand-crafted or machine-extracted (e.g. from text) **'content'** (both prone to errors)

- Can work with **implicitly** collected data (e.g. user watched a movie)

- Can work with **explicitly** collected data (e.g. user rated a movie)

Collaborative filtering

Advantages:

- **Doesn't rely on** hand-crafted or machine-extracted (e.g. from text) **'content'** (both prone to errors)

- Can work with **implicitly** collected data (e.g. user watched a movie)

- Can work with **explicitly** collected data (e.g. user rated a movie)

Disadvantages:

- Needs many non-unique (e.g. similar, linearly dependent) users which interact with many different items.

- **Cold start:** Hard to find recommendations for 'new' items or users with little past information or activity.

- **Scalability:** Big data needs big processing power.

- **Sparsity:** Thousands of items, each user interacts with only a few.

Questions?

Collaborative filtering using NMF

Question: How to find the missing values? (a.k.a. the 'data imputation' problem.)

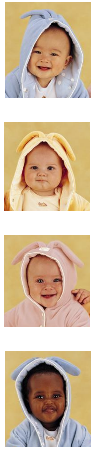
Use NMF!

	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

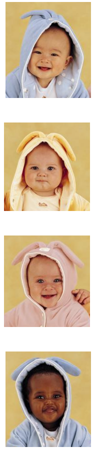
Utility matrix

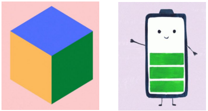
Missing values?



3	1	NaN	3	1
1	2	4	1	3
3	1	1	NaN	1
4	3	NaN	4	4

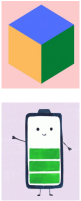
≈





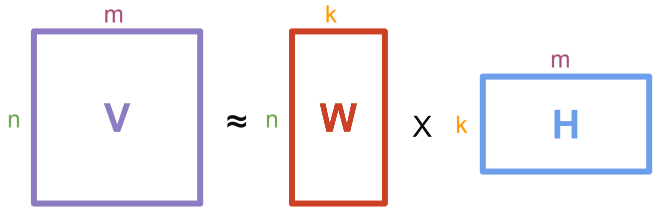
?	?
?	?
?	?
?	?

X



?	?	?	?	?
?	?	?	?	?

How is it computed?



Algorithm: Multiplicative update in steps (lab assignment).

1) Initialize **W** and **H** randomly (all W_{ij} and H_{ij} drawn from a uniform distribution in the interval (0,1)).

Compute the reconstruction error $E = ||\mathbf{V}\text{-}\mathbf{WH}||^2$. (treat missing values as "unknown"→no loss contribution!)

2) Individually update **W** and **H** to minimize $||\mathbf{V}\text{-}\mathbf{WH}||^2$ using (treat missing values like 0s→no contribution to matmul!)

$$W_{ij} \leftarrow W_{ij}\, (\mathbf{VH}^T)_{ij} \,/\, ((\mathbf{WHH}^T)_{ij} + \varepsilon)$$

$$H_{ij} \leftarrow H_{ij}\, (\mathbf{W}^T\mathbf{V})_{ij} \,/\, ((\mathbf{W}^T\mathbf{WH})_{ij} + \varepsilon)$$

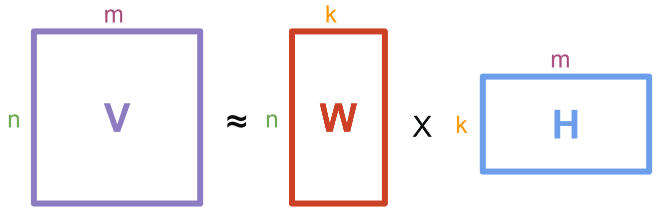
Make sure there is no division by 0 (can use an ε term).

3) Compute the new reconstruction error $E_{new} = ||\mathbf{V}\text{-}\mathbf{WH}||^2$ (treat missing values as "unknown"→no loss contribution!)

4) Stop updating and end optimization if $E - E_{new}$ < a predefined error tolerance.

5) While ($E - E_{new}$) isn't small enough, repeat steps 2-4 for a predefined number of maximum iterations.

How is it computed?



Changes to "standard" NMF:

1) Compute the reconstruction error $E = ||\mathbf{V\text{-}WH}||^2$. (treat missing values as "unknown"→no loss contribution!)

2) Individually update **W** and **H** (treat missing values as unknown/0s→no contribution to matrix multiplication!)

- **Recommended solution** (also for computing the loss):

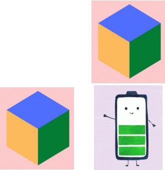
Use the numpy masked array module (`np.ma`):

```
>>> a = np.ma.array([[1, 2, 3], [4, 5, 6]], mask=[[1, 0, 0], [0, 0, 0]])
>>> b = np.ma.array([[1, 2], [3, 4], [5, 6]], mask=[[1, 0], [0, 0], [0, 0]])
>>> np.ma.dot(a, b)
        masked_array(
        data=[[21, 26],
              [45, 64]])
```

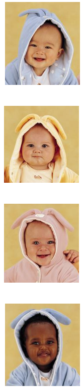
NMF example from last time

NMF using **multiplicative update** algorithm, random initialization in (0,1), no normalization:

Total reconstruction error: 5.33e-10



3	1	1	3	1
1	2	4	1	3



1	0	3	1	1	3	1
0	1	1	2	4	1	3
1	0	3	1	1	3	1
1	1	4	3	5	4	4

prediction

3.02	0.86	0.68	3.02	0.77
0.63	1.73	3.56	0.63	2.65

prediction

0.97	0.09
0.10	1.10
0.97	0.09
1.08	1.20

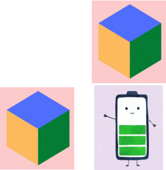
3.00000	1.00000	1.00001	3.00000	0.99999
1.00001	2.00000	3.99999	1.00001	3.00001
3.00000	1.00000	1.00001	3.00000	0.99999
4.00000	3.00000	5.00000	4.00000	4.00000

Reconstruction

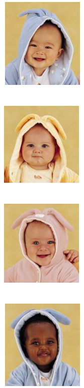
Example 1, 1 missing value

NMF using **multiplicative update** algorithm, random initialization in (0,1), no normalization:

Reconstruction error known data: 6.29e-07
Reconstruction error missing data: 1.67e-06



3	1	1	3	1
1	2	4	1	3



1	0
0	1
1	0
1	1

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

prediction

3.48	0.86	0.50	3.48	0.68
0.61	1.34	2.70	0.61	2.02

prediction

0.82	0.219
0.03	1.48
0.82	0.22
0.85	1.70

3.00004	0.99979	1.00035	3.00005	0.99960
0.99997	2.00012	3.99976	0.99997	3.00024
2.99993	1.00020	1.00129	2.99993	1.00028
4.00003	2.99992	5.00012	4.00003	3.99984

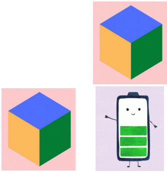
Reconstruction

Example 2, 3 missing values

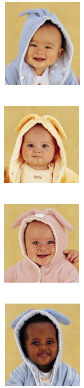
NMF using **multiplicative update** algorithm, random initialization in (0,1), no normalization:

Reconstruction error known data: 2.76e-08

Reconstruction error missing data: **12.15**



3	1	1	3	1
1	2	4	1	3



1	0
0	1
1	0
1	1

3	1	NaN	3	1
1	2	4	1	3
3	1	NaN	3	1
4	3	NaN	4	4

prediction

3.15	0.58	2.32	3.15	0.30
0.72	1.70	3.34	0.72	2.57

prediction

0.89	0.29
0.09	1.16
0.89	0.29
0.94	1.45

3.00000	0.99994	**3.01226**	3.00000	1.00006
1.00006	2.00005	4.00000	1.00005	2.99993
3.00000	0.99994	**3.01226**	3.00000	1.00006
3.99999	3.00001	**7.01226**	3.99998	4.00002

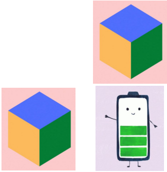
Reconstruction

Example 2, 3 missing values

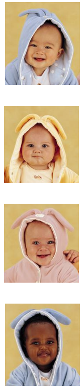
NMF using **multiplicative update** algorithm, random initialization in (0,1), no normalization:

Reconstruction error known data: 2.76e-08

Reconstruction error missing data: **12.15**



3	1	1	3	1
1	2	4	1	3



1	0
0	1
1	0
1	1

3	1	NaN	3	1
1	2	4	1	3
3	1	NaN	3	1
4	3	NaN	4	4

Sparsity problem…

prediction

3.15	0.58	2.32	3.15	0.30
0.72	1.70	3.34	0.72	2.57

prediction

0.89	0.29
0.09	1.16
0.89	0.29
0.94	1.45

3.00000	0.99994	3.01226	3.00000	1.00006
1.00006	2.00005	4.00000	1.00005	2.99993
3.00000	0.99994	3.01226	3.00000	1.00006
3.99999	3.00001	7.01226	3.99998	4.00002

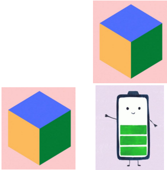
Reconstruction

Example 3, missing column

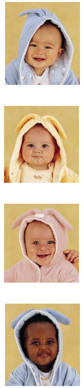
NMF using **multiplicative update** algorithm, random initialization in (0,1), no normalization:

Reconstruction error known data: 3.23e-08

Reconstruction error missing data: **43.00**



3	1	1	3	1
1	2	4	1	3



1	0	3	1	NaN	3	1
0	1	1	2	NaN	1	3
1	0	3	1	NaN	3	1
1	1	4	3	NaN	4	4

prediction

3.38	0.61	0.00	3.38	0.30
0.75	1.55	0.00	0.75	2.33

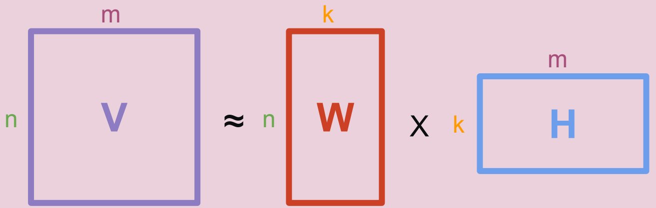
prediction

0.82	0.33
0.01	1.29
0.82	0.33
0.83	1.61

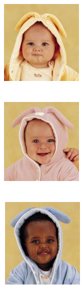
3.00000	0.99996	0.00	3.00000	1.00004
1.00009	2.00002	0.00	1.00008	2.99992
3.00000	0.99996	0.00	3.00000	1.00004
3.99997	3.00001	0.00	3.99997	4.00004

Reconstruction

Example 3, missing column



$H_{ij} \leftarrow H_{ij} \, (\mathbf{W}^T\mathbf{V})_{ij} \, / \, ((\mathbf{W}^T\mathbf{W}\mathbf{H})_{ij}+\varepsilon)$ → **Why?**



0	1	1	2	NaN	1	3
1	0	3	1	NaN	3	1
1	1	4	3	NaN	4	4

NMF using **multiplicative update** algorithm, random initialization in (0,1), no normalization:

Reconstruction error known data: 3.23e-08

Reconstruction error missing data: **43.00**

prediction

3.38	0.61	**0.00**	3.38	0.30
0.75	1.55	**0.00**	0.75	2.33

prediction

0.82	0.33
0.01	1.29
0.82	0.33
0.83	1.61

3.00000	0.99996	**0.00**	3.00000	1.00004
1.00009	2.00002	**0.00**	1.00008	2.99992
3.00000	0.99996	**0.00**	3.00000	1.00004
3.99997	3.00001	**0.00**	3.99997	4.00004

Reconstruction

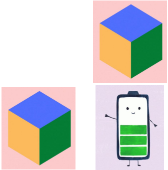
Example 3, missing column

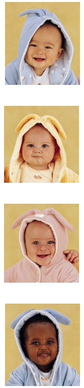
NMF using **multiplicative update** algorithm, random initialization in (0,1), no normalization:

Reconstruction error known data: 3.23e-08

ta: **43.00**

on





1	0
0	1
1	0
1	1

4	3	NaN	4	4

3.38	0.30
0.75	2.33

3.00000	1.00004
1.00008	2.99992
3.00000	1.00004

0.83	1.61	3.99997	3.00001	0.00	3.99997	4.00004

Reconstruction

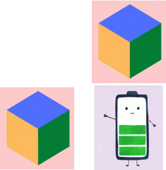
$$\underset{n\times m}{V} \approx \underset{n\times k}{W} \times \underset{k\times m}{H}$$

$W_{ij} \leftarrow W_{ij}\,(\mathbf{VH}^T)_{ij} / ((\mathbf{WHH}^T)_{ij}+\varepsilon)$ → entries update to zero if missing **row** in V

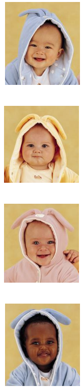
$H_{ij} \leftarrow H_{ij}\,(\mathbf{W}^T\mathbf{V})_{ij} / ((\mathbf{W}^T\mathbf{WH})_{ij}+\varepsilon)$ → entries update to zero if missing **column** in V

~cold start problem

Example 4, sparse data



3	1	1	3	1
1	2	4	1	3



1	0
0	1
1	0
1	1

NaN	1	1	NaN	1
NaN	NaN	NaN	NaN	3
3	1	NaN	3	1
4	NaN	NaN	4	4

NMF using **multiplicative update** algorithm, random initialization in (0,1), no normalization:

Reconstruction error known data: 2.08e-07

Reconstruction error missing data: **41.24**

prediction

2.88	0.58	0.95	2.88	0.58
0.87	1.52	1.36	0.87	1.51

prediction

0.24	**0.57**
1.26	1.50
0.95	**0.29**
0.67	2.38

1.18	1.00	1.00	1.18	1.00
4.92	3.00	3.23	4.92	3.00
3.00	1.00	1.30	3.00	1.00
4.00	4.00	3.88	4.00	4.00

Reconstruction

Questions?

Recommendations

NMF using **multiplicative update** algorithm, random initialization in (0,1), no normalization:

Reconstruction error known data: 2.08e-07
Reconstruction error missing data: **41.24**

So far, we only considered reconstruction…

Question?

How do we build a recommender system?

prediction

2.88	0.58	0.95	2.88	0.58
0.87	1.52	1.36	0.87	1.51

prediction

0.24	**0.57**
1.26	1.50
0.95	**0.29**
0.67	2.38

1.18	1.00	1.00	1.18	1.00
4.92	3.00	3.23	4.92	3.00
3.00	1.00	1.30	3.00	1.00
4.00	4.00	3.88	4.00	4.00

Reconstruction

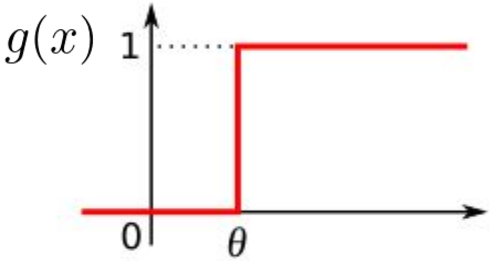
Recommendations

Threshold the elements of the reconstruction **V'=WH**:

$$g(V'_{ij}) \in \{0, 1\}$$

where

$$g(x) = \begin{cases} 0 & \text{if} \quad x < \theta \\ 1 & \text{if} \quad x \geq \theta \end{cases}$$



and 1 means recommend, 0 means don't recommend…

Recommendations

W

2.88	0.58	0.95	2.88	0.58
0.87	1.52	1.36	0.87	1.51

H

0.24	**0.57**
1.26	1.50
0.95	**0.29**
0.67	2.38

Reconstruction **V'**

1.18	1.00	1.00	1.18	1.00
4.92	3.00	3.23	4.92	3.00
3.00	1.00	1.30	3.00	1.00
4.00	4.00	3.88	4.00	4.00

g(**V'**), θ=2.5

0	**0**	**0**	**0**	**0**
1	**1**	**1**	**1**	**1**
1	**0**	**0**	**1**	**0**
1	**1**	**1**	**1**	**1**

Recommendation accuracy

Accuracy is the percentage of correct recommendations.

g(**V'**), θ=2.5

0	0	0	0	0
1	1	1	1	1
1	0	0	1	0
1	1	1	1	1

Binarized reconstruction

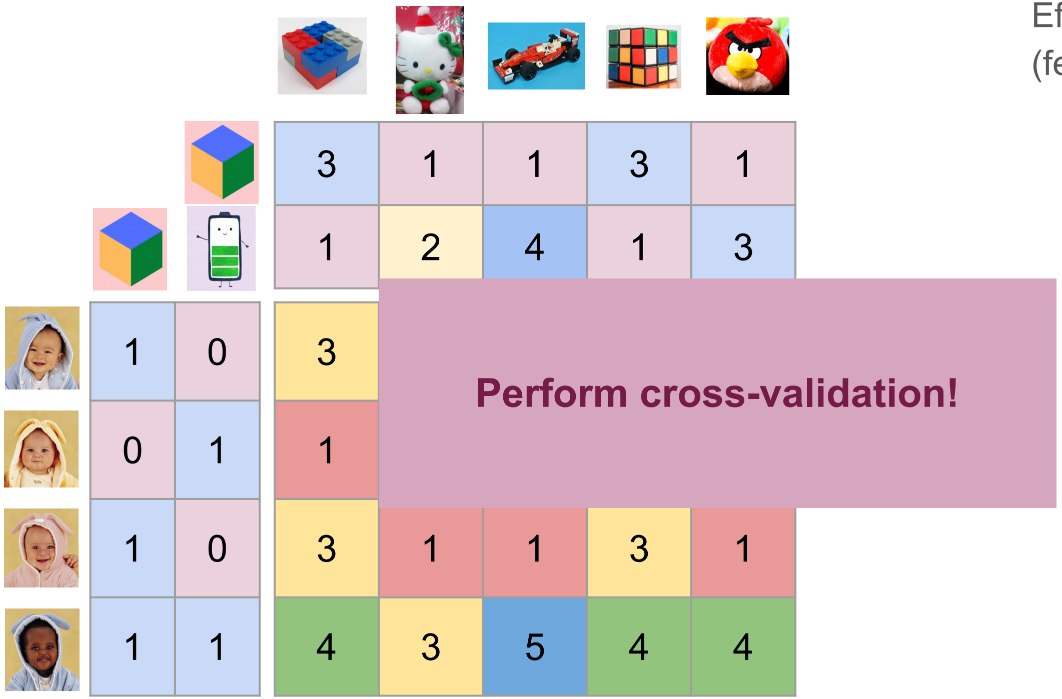
Ground truth

1	0	0	1	0
0	0	1	0	1
1	0	0	1	0
1	1	1	1	1

In this case 4/9 ≈ 44.4%.

Questions?

From last time…



NMF using **multiplicative update** algorithm, random initialization in (0,1), no normalization.

Effect of using different number of components (features "k"):

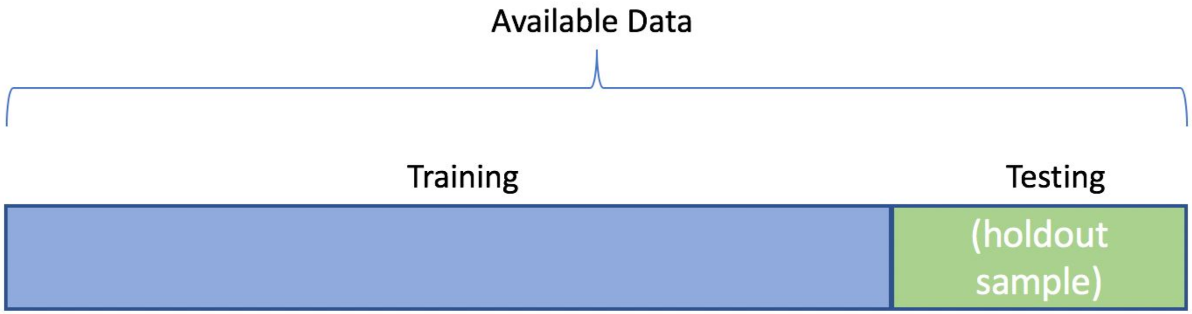
	Total reconstruction error
1 component	15.24
2 components	**5.33e-10**
3 components	2.76e-08
4 components	7.56e-05

Perform cross-validation!

Cross-validation

Assume we evaluate NMF performance using the recommendation accuracy.

We optimize **W** and **H** on a training dataset, evaluate on an independent test set.



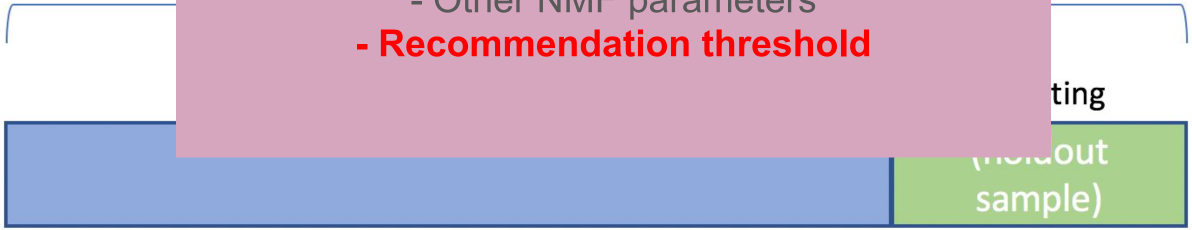
Cross-validation

Assume we evaluate NMF performance using the recommendation accuracy.

We optimize **W** and **H** on … nt test set.

Potential design choices:

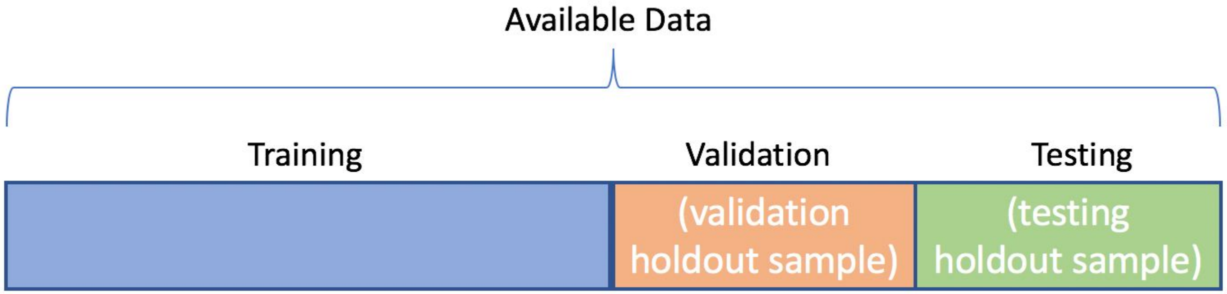
- Pre-processing
- **Number of features k**
- Other NMF parameters
- **Recommendation threshold**



Cross-validation

We don't want to overfit our model parameters/design choices to the test set.

Use an independent validation set!

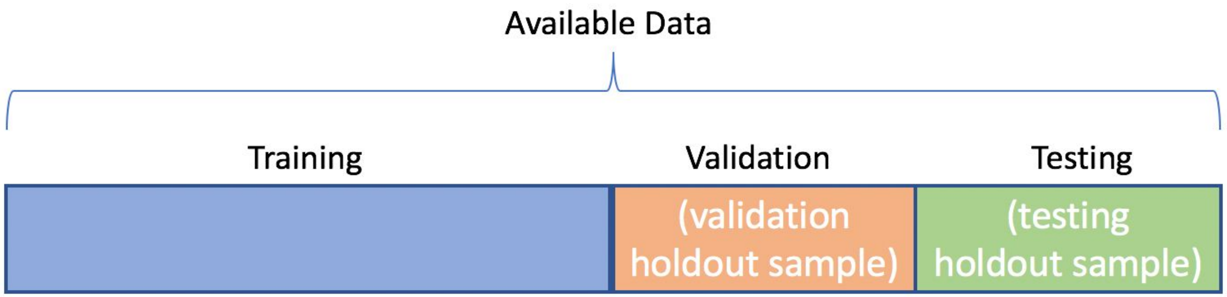


Cross-validation

We don't want to overfit … est set.

Use an independent val…

How to divide the data for cross-validation?



Cross-validation

Leave-one-out cross-validation (mask out one entry of **V** during training)

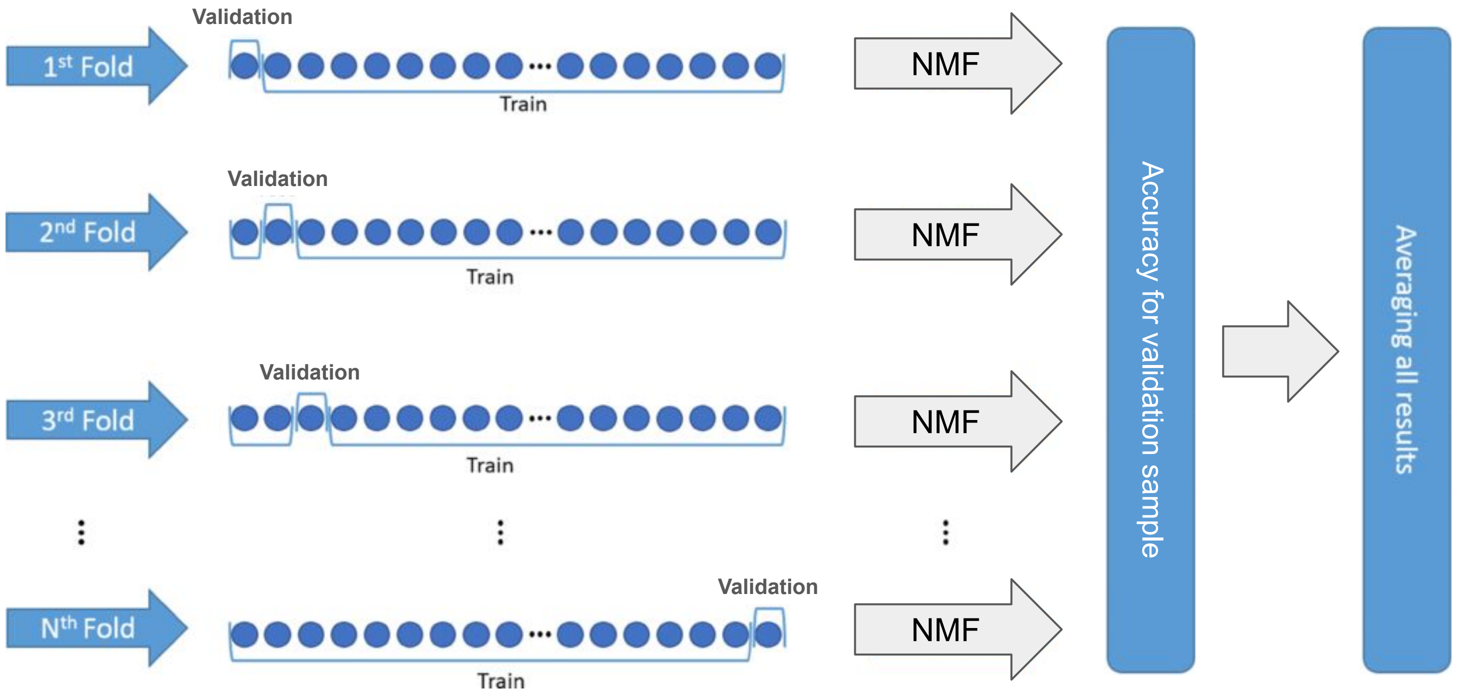


Image source: https://www.researchgate.net/figure/The-schematic-of-data-division-in-Leave-one-out-cross-validation-methodology-in-SVM_fig1_332425167

Cross-validation

Leave-one-out cross-validation (mask out one entry of **V** during training)

NaN	Validation	1	NaN	1
NaN	NaN	NaN	NaN	3
3	1	NaN	3	1
4	NaN	NaN	4	4

1st fold

Cross-validation

Leave-one-out cross-validation (mask out one entry of **V** during training)

NaN	1	Validation	NaN	1
NaN	NaN	NaN	NaN	3
3	1	NaN	3	1
4	NaN	NaN	4	4

2nd fold

Cross-validation

Leave-one-out cross-validation (mask out one entry of **V** during training)

NaN	1	1	NaN	Validation
NaN	NaN	NaN	NaN	3
3	1	NaN	3	1
4	NaN	NaN	4	4

3rd fold

Cross-validation

Leave-one-out cross-validation (mask out one entry of **V** during training)

NaN	1	1	NaN	1
NaN	NaN	NaN	NaN	Validation
3	1	NaN	3	1
4	NaN	NaN	4	4

4th fold

Cross-validation

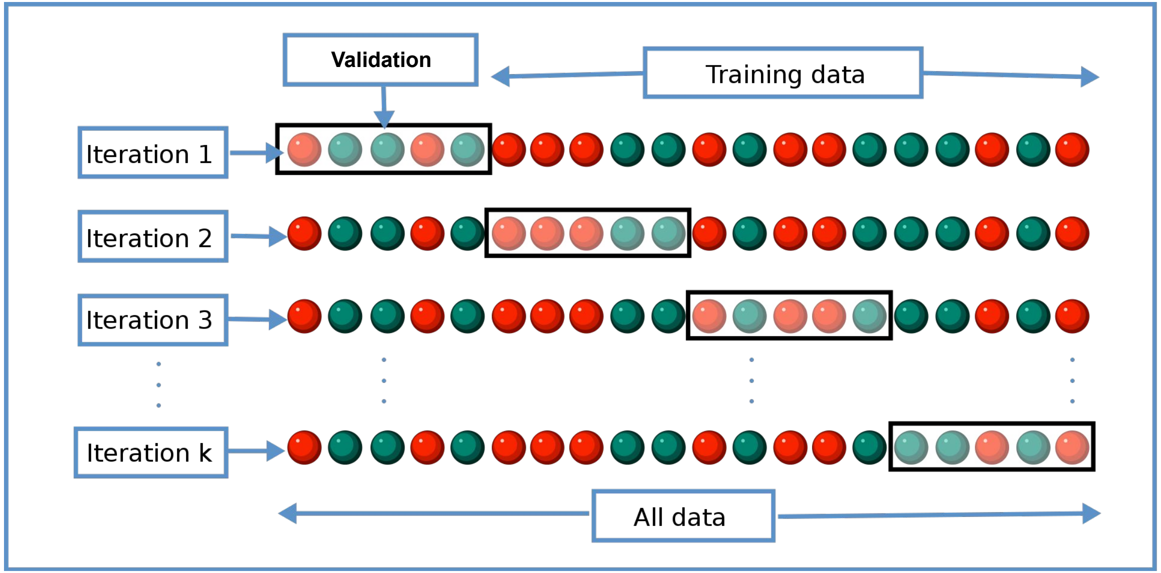
Leave-one-out cross-validation (mask out one entry of **V** during training)

NaN	1	1	NaN	1
NaN	NaN	NaN	NaN	3
Validation	1	NaN	3	1
4	NaN	NaN	4	4

5th fold, etc…

Cross-validation

k-fold cross-validation (mask out mutually exclusive subsets of entries in **V** for training)



k-fold cross-validation

For many tasks, our **data matrix** is an *n*-by-*d* matrix which has *n* samples, and *d* features.

***d* features**

***n* samples**

sepal length	sepal width	petal length	petal width
5.1	3.5	1.4	0.2
4.9	3	1.4	0.2
6.5	3.2	5.1	2
6.4	2.7	5.3	1.9
6.8	3	5.5	2.1
6.7	3.1	4.4	1.4
5.6	3	4.5	1.5
5.8	2.7	4.1	1

k-fold cross-validation

For many tasks, we can pick **whole ‘samples’ (rows)** as a holdout validation set (e.g. remember PCA).

***d* features**

***n* samples**

sepal length	sepal width	petal length	petal width
5.1	3.5	1.4	0.2
4.9	3	1.4	0.2
6.5	3.2	5.1	2
6.4	2.7	5.3	1.9
6.8	3	5.5	2.1
6.7	3.1	4.4	1.4
5.6	3	4.5	1.5
5.8	2.7	4.1	1

1st fold

k-fold cross-validation

For many tasks, we can pick **whole 'samples' (rows)** as a holdout validation set (e.g. remember PCA).

***d* features**

***n* samples**

sepal length	sepal width	petal length	petal width
5.1	3.5	1.4	0.2
4.9	3	1.4	0.2
6.5	3.2	5.1	2
6.4	2.7	5.3	1.9
6.8	3	5.5	2.1
6.7	3.1	4.4	1.4
5.6	3	4.5	1.5
5.8	2.7	4.1	1

2nd fold

k-fold cross-validation

For many tasks, we can pick **whole ‘samples’ (rows)** as a holdout validation set (e.g. remember PCA).

***d* features**

***n* samples**

sepal length	sepal width	petal length	petal width
5.1	3.5	1.4	0.2
4.9	3	1.4	0.2
6.5	3.2	5.1	2
6.4	2.7	5.3	1.9
6.8	3	5.5	2.1
6.7	3.1	4.4	1.4
5.6	3	4.5	1.5
5.8	2.7	4.1	1

3rd fold

k-fold cross-validation

For many tasks, we can pick **whole ‘samples’ (rows)** as a holdout validation set (e.g. remember PCA).

***d* features**

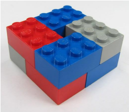
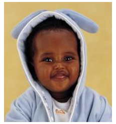
***n* samples**

sepal length	sepal width	petal length	petal width
5.1	3.5	1.4	0.2
4.9	3	1.4	0.2
6.5	3.2	5.1	2
6.4	2.7	5.3	1.9
6.8	3	5.5	2.1
6.7	3.1	4.4	1.4
5.6	3	4.5	1.5
5.8	2.7	4.1	1

4th fold

m toys

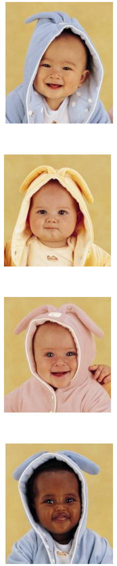
n babies

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

How do we pick the cross-validation folds for NMF?

m toys

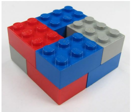
n babies



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

For multiplicative update algorithm, we cannot mask whole rows!

m toys

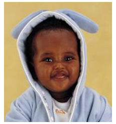




Remember: We cannot have whole row/column missing for the multiplicative update algorithm…

$W_{ij} \leftarrow W_{ij} \, (\mathbf{VH}^T)_{ij} / ((\mathbf{WHH}^T)_{ij}+\varepsilon)$ → entries update to zero if missing **row** in V

$H_{ij} \leftarrow H_{ij} \, (\mathbf{W}^T\mathbf{V})_{ij} / ((\mathbf{W}^T\mathbf{WH})_{ij}+\varepsilon)$ → entries update to zero if missing **column** in V

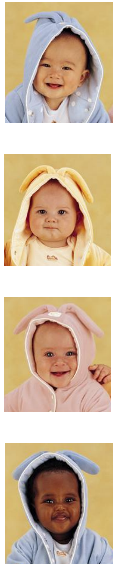


4	3	5	4	4

For multiplicative update algorithm, we cannot mask whole rows!

m toys

n babies

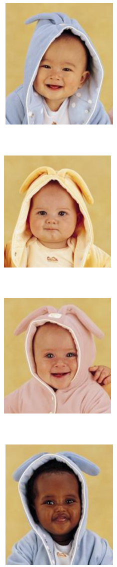


3	1	1	3	1
1	2	4	1	3
3	1	1	3	1
4	3	5	4	4

Conceptually, we cannot perform **collaborative filtering** if we don't have any information about a user/item.

m toys

n babies



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

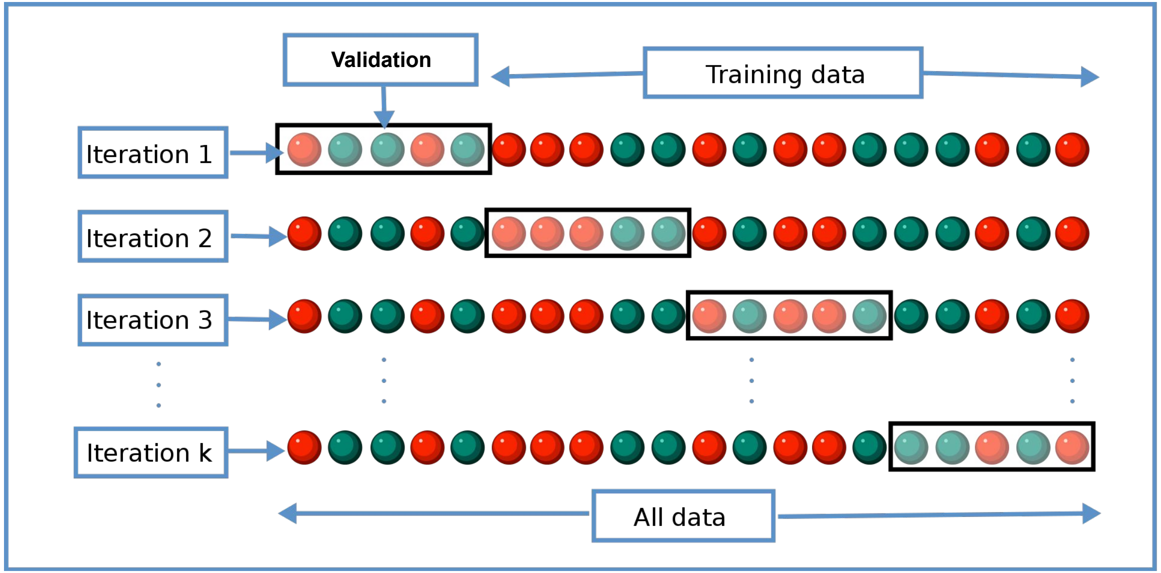
We can pick mutually exclusive cross-validation folds from different rows/columns.

Cross-validation

k-fold cross-validation (mask out mutually exclusive subsets of entries in **V** for training)

Be careful not to mask out **all entries** in a row/column!

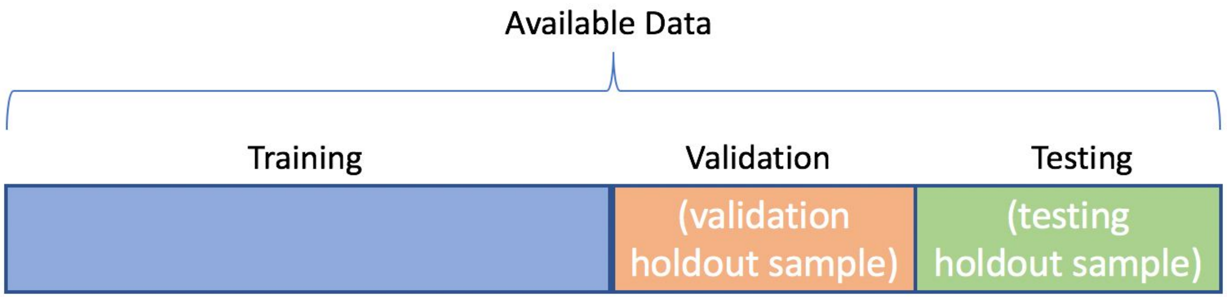
(Multiplicative update will set values to zero!)



Cross-validation - Recommender systems

Data is large → - **leave-one-out**: computationally expensive

Data is sparse → - **k-fold with small k**: training data might be too small, not informative



Cross-validation - Recommender systems

Data is large → - **leave-one-out**: computationally expensive

Data is sparse → - **k-fold with small k**: training data might be too small not informative

If it's computationally too expensive to use large number of folds, we can also use **one** independent validation set and **one** independent test set (e.g. lab assignment).

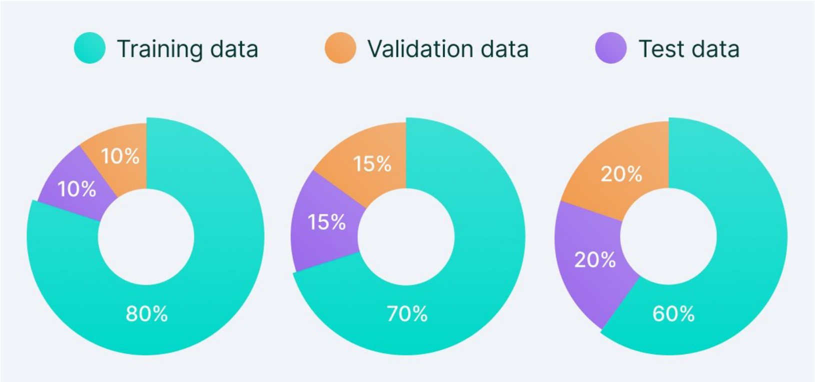


Image source: https://www.v7labs.com/blog/train-validation-test-set

Questions?

Performance metrics

- Evaluation metrics for **predictions**
    - Reconstruction error: Mean squared error ($1/N * ||\mathbf{V}-\mathbf{WH}||^2$),

      root mean squared error
- Evaluation metrics for **recommendations**
    - Recommendation accuracy: Percentage of correct recommendations.

How do we evaluate NMF performance?

Performance metrics

- Further evaluation metrics for **recommendations**

    - True positive/false positive rates

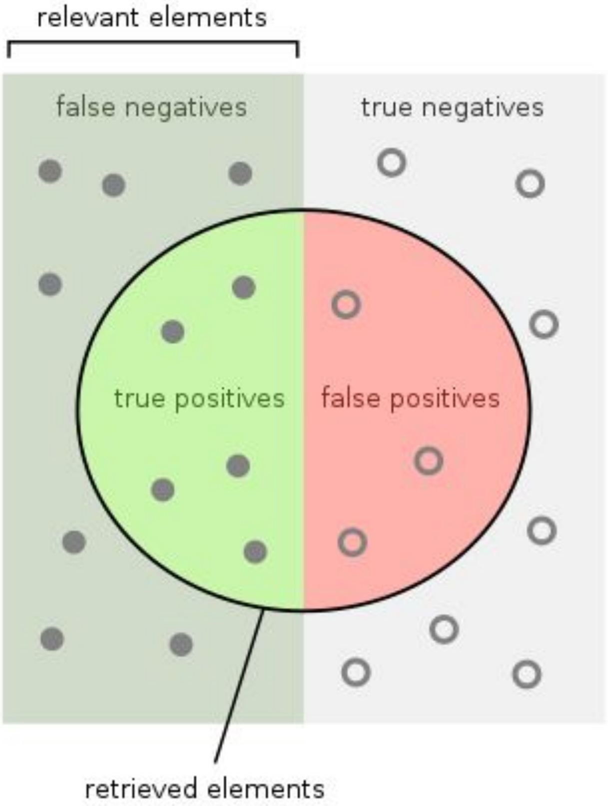


Image source: https://en.wikipedia.org/wiki/Precision_and_recall

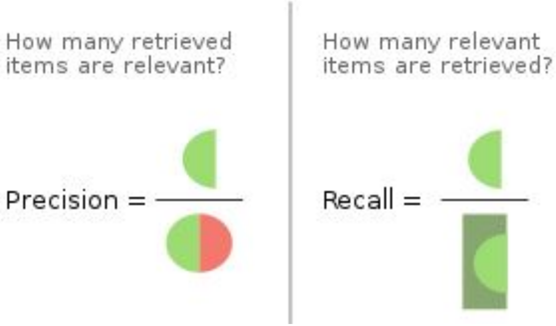
Performance metrics

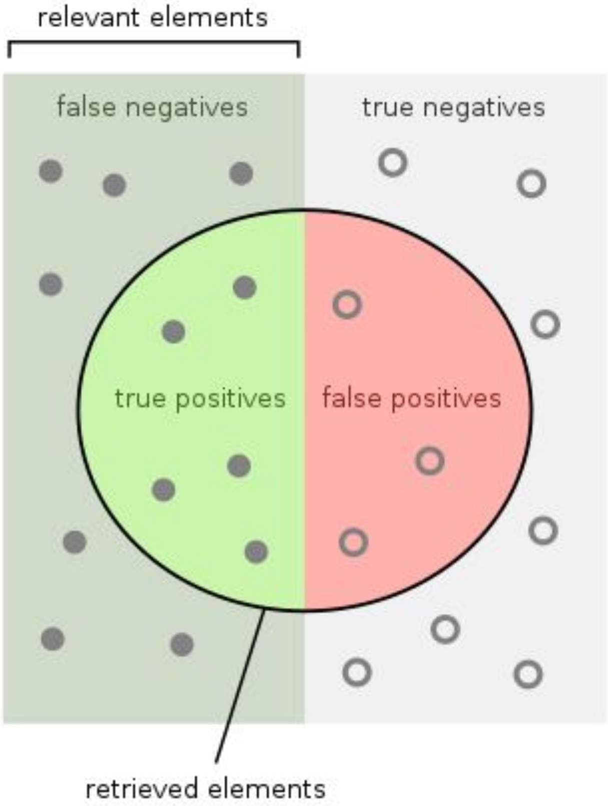
- Further evaluation metrics for **recommendations**

  - True positive/false positive rates

  - $$\text{Precision} = \frac{tp}{tp + fp}$$

    $$\text{Recall} = \frac{tp}{tp + fn}$$





Performance metrics

- Further evaluation metrics for **recommendations**

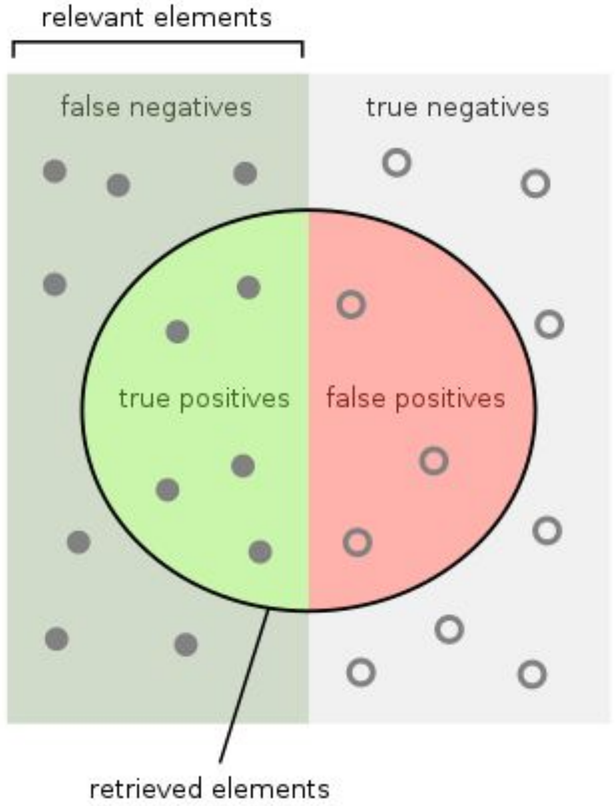
    - True positive/false positive rates

    - $$\text{Precision} = \frac{tp}{tp + fp}$$

      $$\text{Recall} = \frac{tp}{tp + fn}$$

    - F1-score:

$$F_1 = \frac{2}{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}} = 2\frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$



Performance metrics - Recommender systems

- **Online** evaluation: User’s online reactions, e.g. the clicks or views a recommendation gets, etc.
    - A/B testing

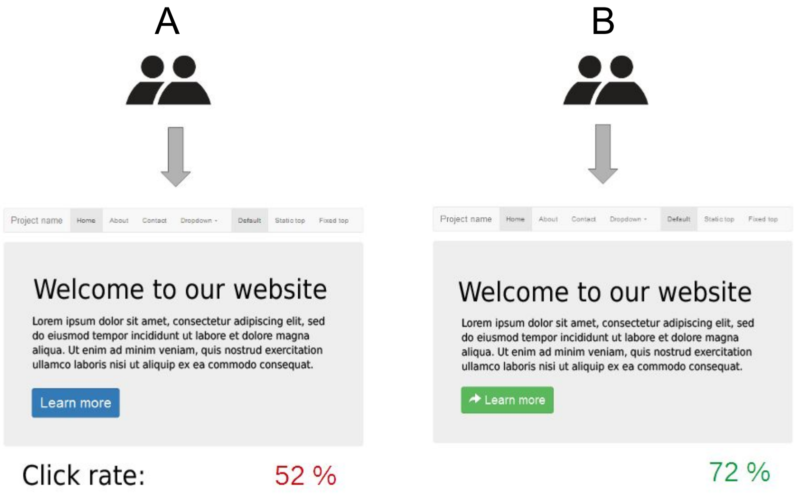


Image source: https://en.wikipedia.org/wiki/A/B_testing

Performance metrics - Recommender systems

- **Ranking** based recommendations:

So far we only performed naive thresholding to get recommendations

1.18	1.00	1.00	1.28	1.00
4.92	3.00	3.23	4.82	3.00
3.00	1.00	1.30	3.00	1.00
4.00	4.00	3.88	4.00	4.00

Reconstruction **V'**

$\xrightarrow{g(\mathbf{V'}),\ \theta=2.5}$

0	**0**	**0**	**0**	**0**
1	**1**	**1**	**1**	**1**
1	**0**	**0**	**1**	**0**
1	**1**	**1**	**1**	**1**

Performance metrics - Recommender systems

- **Ranking** based recommendations:

If we want to recommend *k* items to each user, we have to consider rankings!

1.18	1.00	1.00	1.28	1.00
4.92	3.00	3.23	4.82	3.00
3.00	1.00	1.30	3.00	1.00
4.00	4.00	3.88	4.00	4.00

Reconstruction **V'**

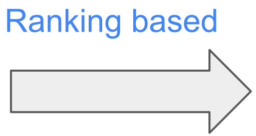
2	**-**	**-**	**1**	**-**
1	**4**	**3**	**2**	**-**
-	**-**	**1**	**-**	**-**
-	**1**	**2**	**-**	**-**

Performance metrics - Recommender systems

- **Ranking** based evaluation metrics include

  - Spearman’s rank correlation: Pearson’s correlation between predicted ranks and ground truths

  - Hit rate at K: Percentage of users for which at least one hit takes place for the top K recommendations.



2	-	-	1	-
1	4	3	2	-
-	-	1	-	-
-	1	2	-	-

Other considerations

- **Diversity** (exploration) – Users tend to be more satisfied if recommendations are diverse.
- **Recommender persistence** – It can be more effective to re-show recommendations than showing new items.

Other considerations

- **Diversity** (exploration) – Users tend to be more satisfied if recommendations are diverse.
- **Recommender persistence** – It can be more effective to re-show recommendations than showing new items.
- **Robustness** – What if two users are using the same account? How to deal with unreliable data?
- **Serendipity** – Serendipity is a measure of "how surprising the recommendations are". If you're an e-commerce dairy farm, milk is not a surprising recommendation, but biscuits might be.

Other considerations

- **Diversity** (exploration) – Users tend to be more satisfied if recommendations are diverse.
- **Recommender persistence** – It can be more effective to re-show recommendations than showing new items.
- **Robustness** – What if two users are using the same account? How to deal with unreliable data?
- **Serendipity** – Serendipity is a measure of "how surprising the recommendations are". If you're an e-commerce dairy farm, milk is not a surprising recommendation, but biscuits might be.
- **Privacy** – Recommender systems usually have to deal with privacy concerns because users might have to reveal sensitive and personally identifying information. Building user profiles using collaborative filtering can be problematic from a privacy point of view.

Questions?

Summary

- Recommender systems are **ubiquitous**

- There are **multiple** possible **approaches** (collaborative, content-based, hybrid)

- Different **types of utility matrices** (ranking-based, preference-based, dense, sparse, etc.)

Summary

- Recommender systems are **ubiquitous**

- There are **multiple** possible **approaches** (collaborative,
  content-based, hybrid)

- Different **types of utility matrices** (ranking-based, preference-based, dense, sparse, etc.)

- Important to think about **cross-validation** when picking methods and making design choices

Summary

- Recommender systems are **ubiquitous**

- There are **multiple** possible **approaches** (collaborative, content-based, hybrid)

- Different **types of utility matrices** (ranking-based, preference-based, dense, sparse, etc.)

- Important to think about **cross-validation** when picking methods and making design choices

Next time:

- **Privacy** is important to consider when dealing with (big) user data

- **Alternatives** to NMF provide different methods for CF

Image source: https://www.youtube.com/watch?v=drWug-hqLmI