Computational Communication Science 2
Week 3 - Lecture
» Recommender systems«

Anne Kroon

a.c.kroon@uva.nl, @annekroon

April 14, 2025

Digital Society Minor, University of Amsterdam

Today

Recap: Week 2

Recommender Systems

Knowledge-based Recommender Systems

Content-based Recommender Systems

Collaborative Recommender Systems

Everything clear from last weeks?

Recap: Week 2

What we covered last week

- Text Preprocessing: Tokenization, stopword removal, stemming/lemmatization
- Building text representations: Count Vectorization, TF-IDF
- Word embeddings: Basics and their applications

Recommender Systems


You
Tube

Recommender Systems in Communication Science

New research questions

1. Political communication and journalism. E.g., to craft personalized news diets. This may have, however, consequences for the diversity of news diets and for democracy (Locherbach & Trilling, 2018; Möller et al., 2018)
2. Organizational and corporate communication. E.g., applications in the field of hiring and recruitment.
3. Persuasive communication. E.g., in the health domain–recommendation algorithms for tailored health interventions (Kim et al., 2019)
4. Entertainment communication. E.g., movie recommenders

Recommender Systems

Two central problems within Recommender Systems

1. Predicting problem. Typical problem involves a lot of missing data (e.g., user only rated a small subset of movies/news articles/ etc.) How can we deal with missing data?
2. Ranking problem. Given n items, can we identify the top k items to recommend?

These problems are not isolated; rather, they are connected.

Recommender Systems

How do recommender systems learn?

1. Explicit user preferences. Ratings or responses
2. Implicit user preferences. E.g., clicks or viewing time
3. Content. E.g., based on text similarity techniques

Recommender Systems

Types of recommender systems (Locherbach & Trilling, 2018; Möller et al., 2018; Wieland et al., 2021)

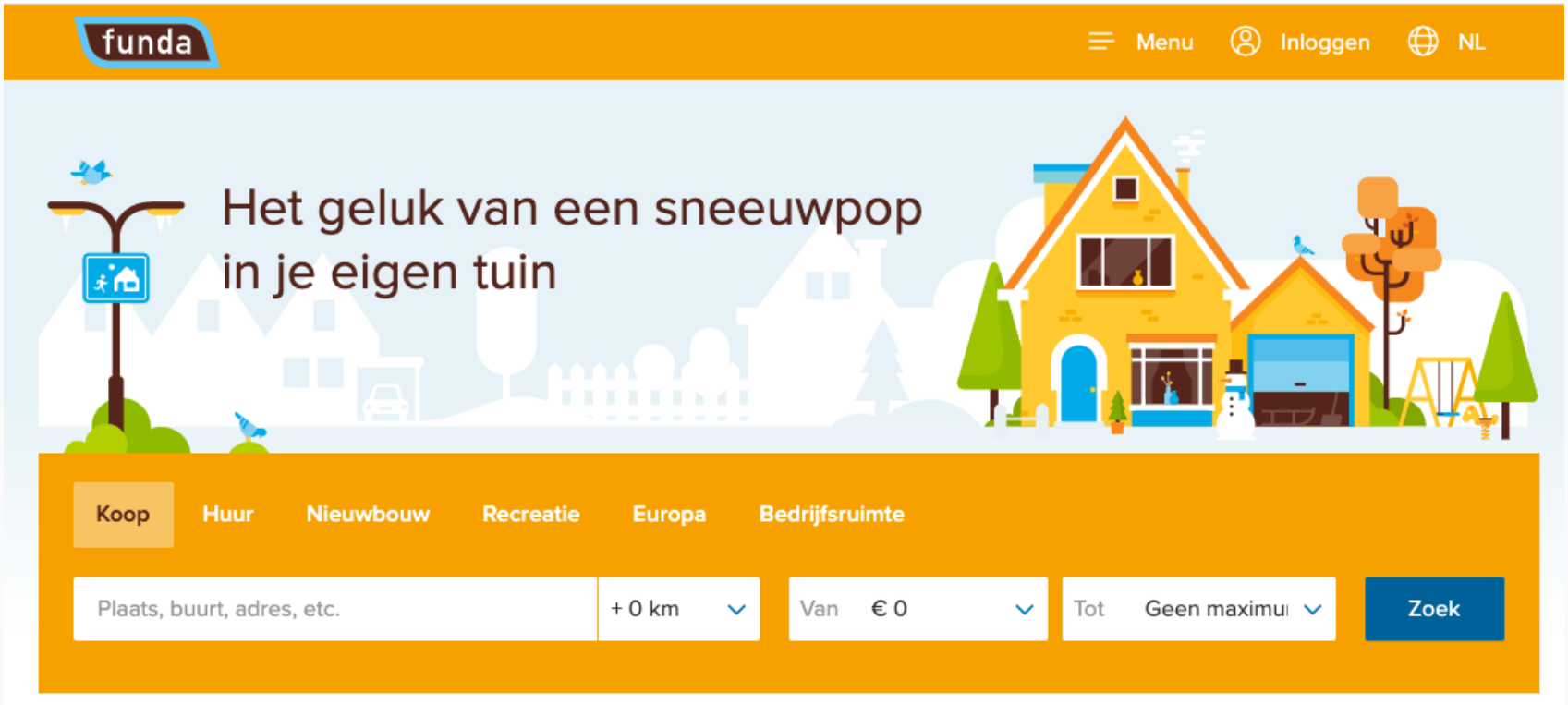
1. 'Basic' knowledge-base recommender systems
2. Content-based recommender systems
3. Collaborative recommenders

Knowledge-based RecSys

Knowledge-based recommender system

When to use?

- To overcome the **cold start problem**; when we do not have ratings of individual users.
- Simple model. It does not rely on user's explicit or implicit ratings, but on specific queries.
- Typical use case: Real-estate. Buying a house is, for most families, a rare/ single event.


funda
Menu
Inloggen
NL
Het geluk van een sneeuwpop
in je eigen tuin
Koop
Huur
Nieuwbouw
Recreatie
Europa
Bedrijfsruimte
Plaats, buurt, adres, etc.
+ 0 km
Van € 0
Tot Geen maximu
Zoek

Use case: ImDb database

	genres	title	tagline	release_date	vote_average	vote_count
0	[action, adventure, fantasy, science fiction]	Avatar	Enter the World of Pandora.	2009-12-10	7.2	11800
1	[adventure, fantasy, action]	Pirates of the Caribbean: At World's End	At the end of the world, the adventure begins.	2007-05-19	6.9	4500
2	[action, adventure, crime]	Spectre	A Plan No One Escapes	2015-10-26	6.3	4466
3	[action, crime, drama, thriller]	The Dark Knight Rises	The Legend Ends	2012-07-16	7.6	9106
4	[action, adventure, science fiction]	John Carter	Lost in our world, found in another.	2012-03-07	6.1	2124

What are relevant variables to use in a knowledge-based recommender system?

Knowledge-based recommender system

How can we work with user input without a front-end (such as the website of funda? $\rightarrow$ enter python's native input() function.

```
1 print("What is your favorite movie genre?")
2 genre = input()
```

```
1 what is your favorite movie genre?
2 [...]
```

Improving knowledge-based recommender system

When to use?

- It is important to think about ways to make the recommendation relevant for individuals
- Do you have more information in your db that make your top-listed recommendations as relevant as possible?

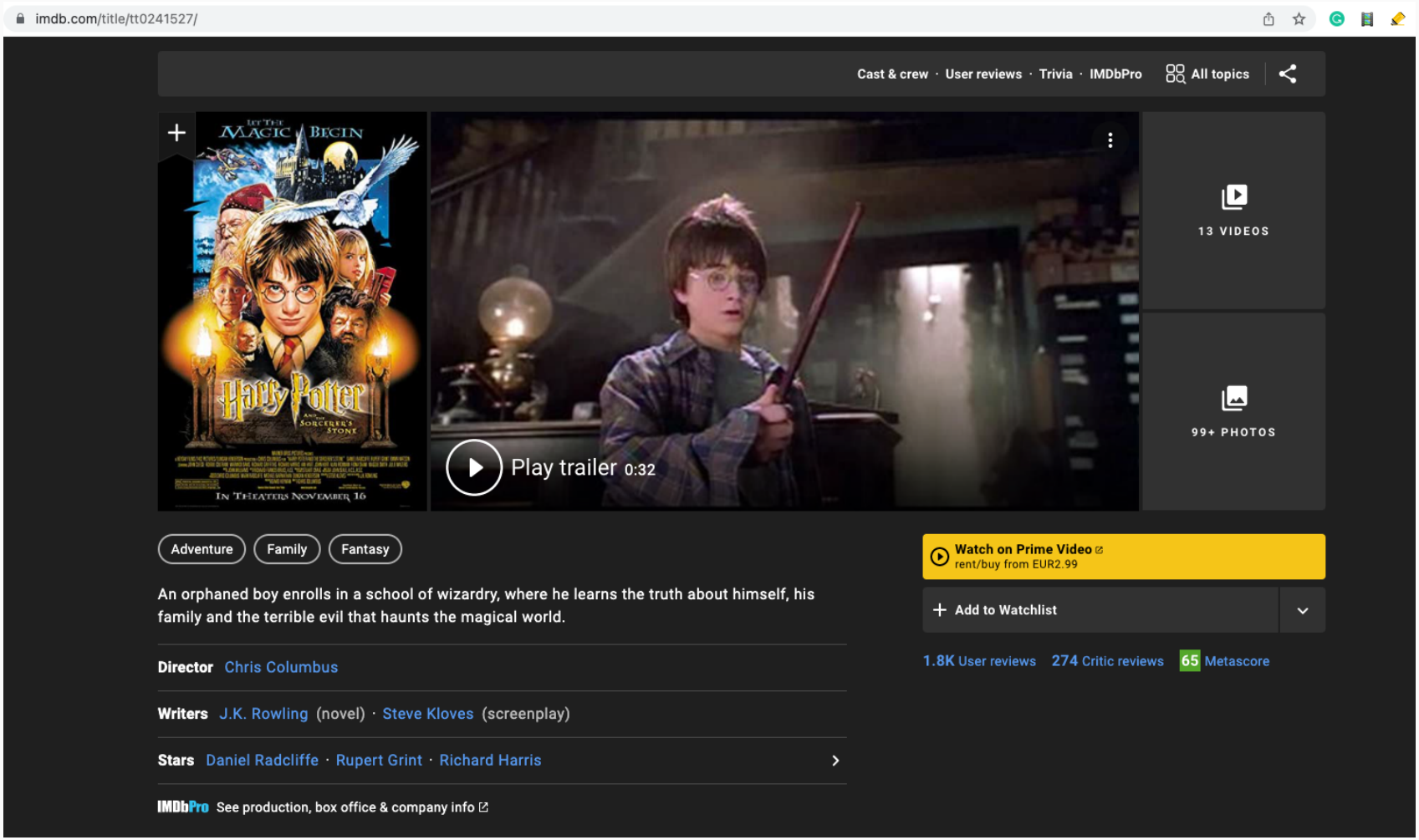
```
recommend_movies = movies.sort_values('vote_average',
↪   ascending=False)
```

Content-based RecSys

Content-based systems

- Recommends items based on user's profiles.
- Profiles are based on e.g., ratings, and represents user's tasts/ preferences.
    - For example, how often a user has clicked on, or liked, a movie.
- Recommendation is based on similarity beween items in the content.
    - Content is here: e.g., genre, tags, plot, authors, directors, location, etc.

Example of a content-based recsys



Example of a content-based recsys

More like this

★ 7.5 ☆

Harry Potter and the Chamber of Secrets

Watch options

★ 7.9 ☆

Harry Potter and the Prisoner of Azkaban

Watch options

★ 7.7 ☆

Harry Potter and the Goblet of Fire

Watch options

★ 7.5 ☆

Harry Potter and the Order of the Phoenix

Watch options

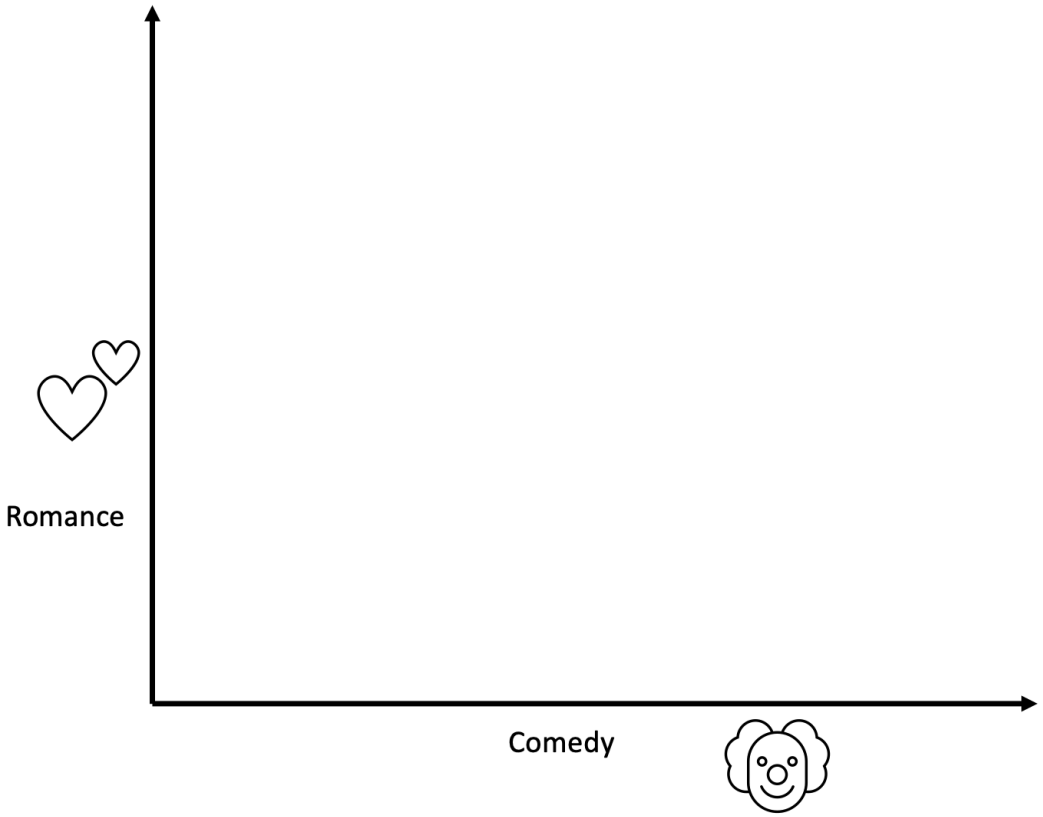
Use case: ImDb database

Let's have a look at our use case again.

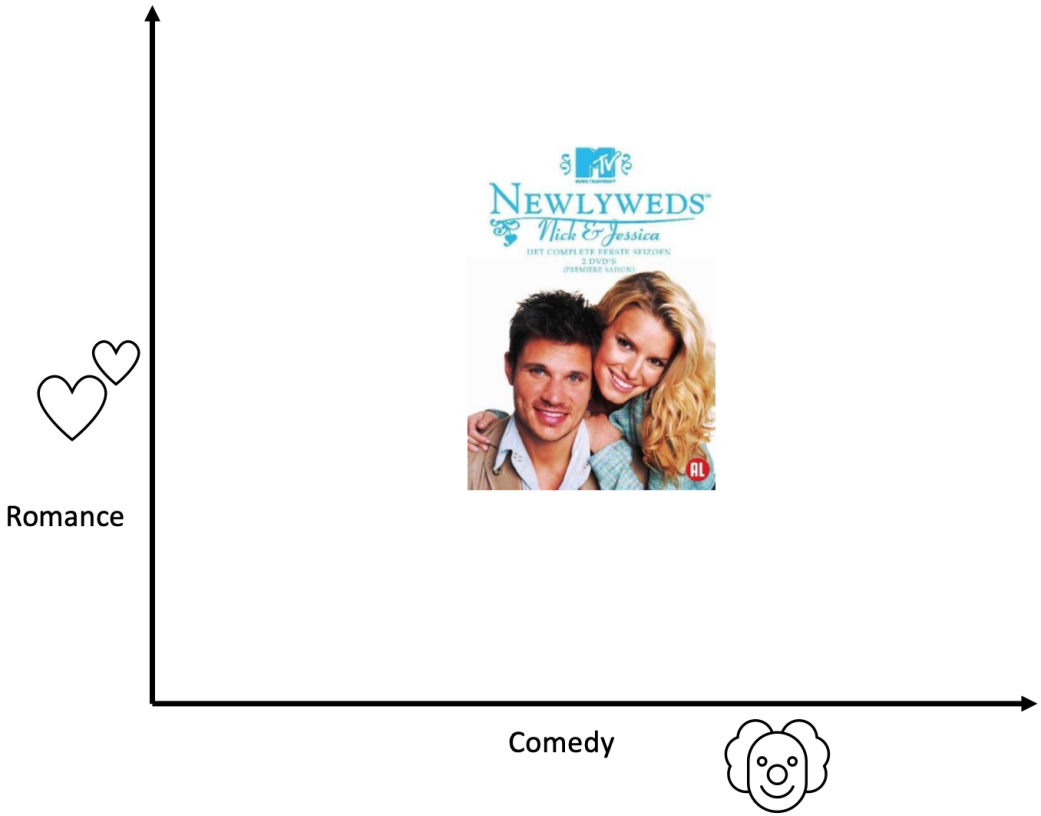
Are there attributes that you could use to estimate similarity in movies?

	genres	title	tagline	release_date	vote_average	vote_count
0	[action, adventure, fantasy, science fiction]	Avatar	Enter the World of Pandora.	2009-12-10	7.2	11800
1	[adventure, fantasy, action]	Pirates of the Caribbean: At World's End	At the end of the world, the adventure begins.	2007-05-19	6.9	4500
2	[action, adventure, crime]	Spectre	A Plan No One Escapes	2015-10-26	6.3	4466
3	[action, crime, drama, thriller]	The Dark Knight Rises	The Legend Ends	2012-07-16	7.6	9106
4	[action, adventure, science fiction]	John Carter	Lost in our world, found in another.	2012-03-07	6.1	2124

Similarity between movies

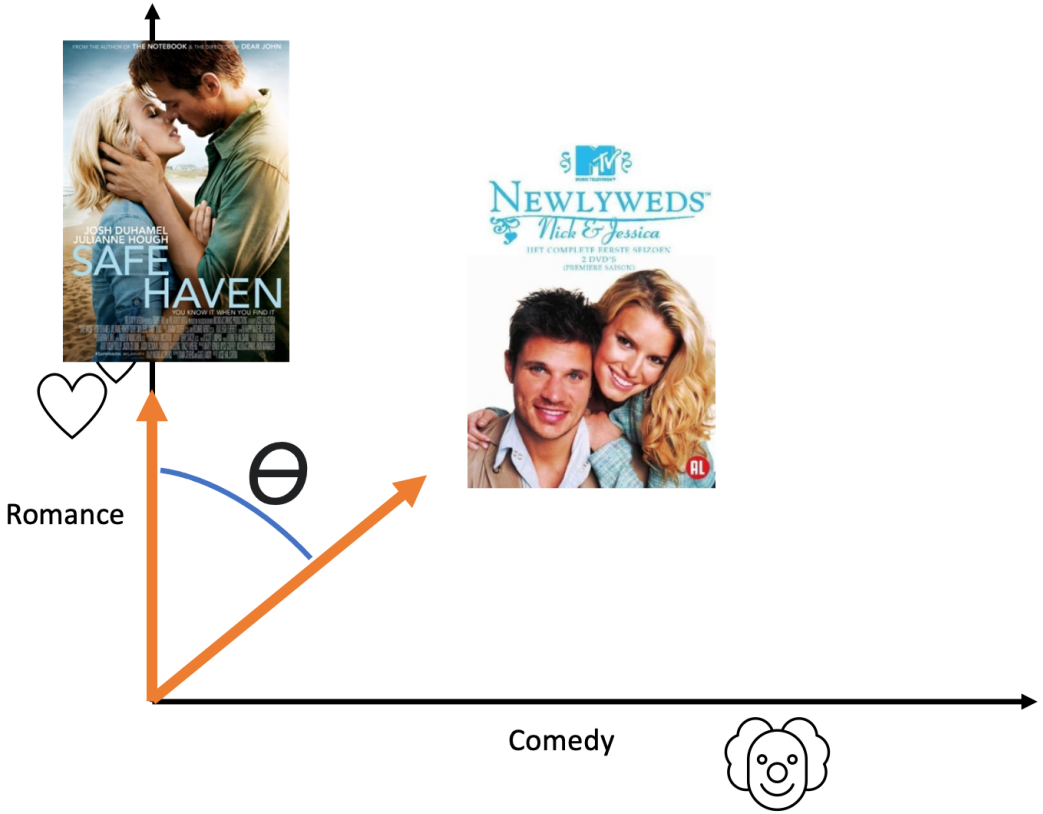


Similarity between movies

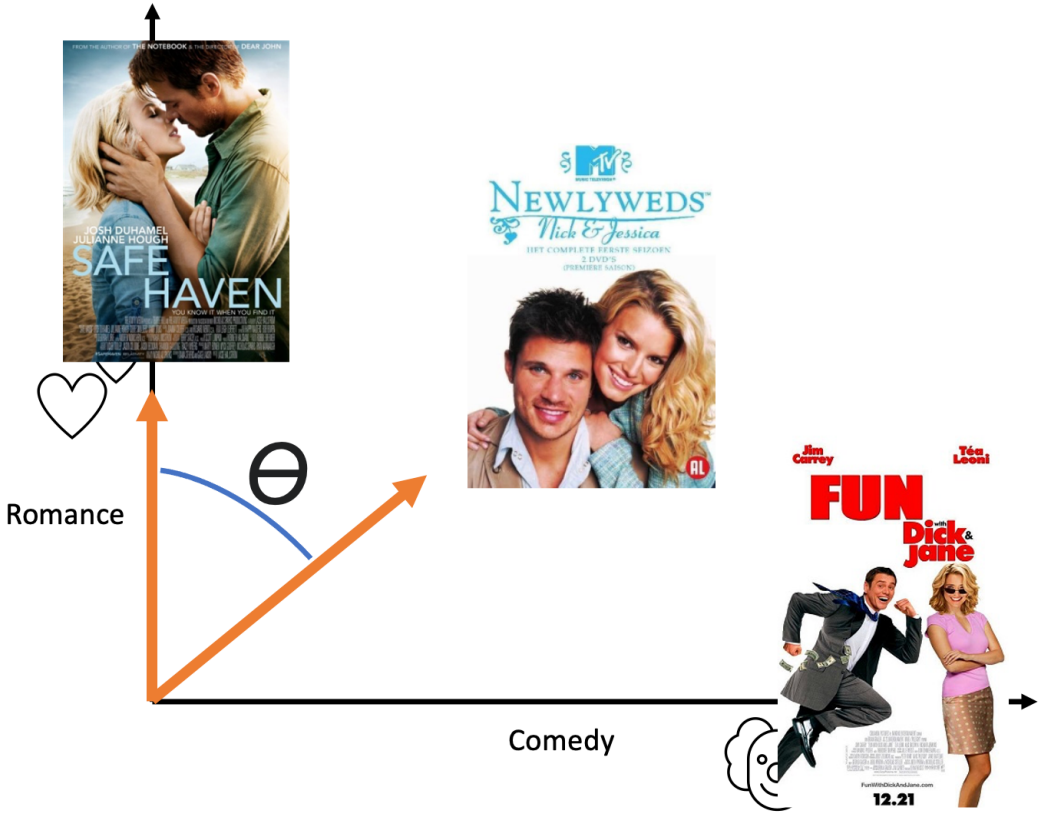


Similarity between movies

Similarity between movies



Similarity between movies



Definition and Applications

Cosine Similarity

- Measures similarity between two documents irrespective of size.

Applications

- Linguistic alignment in communication (Brinberg & Ram, 2021)
- Political speech and public opinion overlap (Hager & Hilbig, 2020)

Mathematical Representation

$$\text{similarity} = \cos(\theta) = \frac{\mathrm{A} \cdot \mathrm{B}}{\|\mathrm{A}\|\|\mathrm{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

- Measures cosine of the angle between vectors.
- 0 (orthogonal, dissimilar), 1 (identical, similar).

Implementation in Python

```
1  from sklearn.feature_extraction.text import CountVectorizer
2  from sklearn.metrics.pairwise import cosine_similarity
3  import pandas as pd
4
5  documents = ["When I eat breakfast, I usually drink some tea", "I like
   ↪  my tea with my breakfast", "She likes cereal and coffee"]
6  vec = CountVectorizer(stop_words='english')
7  count_matrix = vec.fit_transform(documents)
8  cos_sim = cosine_similarity(count_matrix)
9  print(cos_sim)
```

Considerations and Soft Cosine Similarity

Considerations

- Computationally fast, used in recommender systems.
- Requires exact word matches, limiting semantic understanding.

Soft Cosine Similarity

- Uses word embeddings to capture semantic meaning.
- Accounts for synonyms and contextual similarity (Sidorov et al., 2014).

Content-based RecSys

Building blocks of content-based RecSys

Feature selection and engineering

- Feature engineering is very important here. What qualities or attributes do we want to incorporate?
- Think about this critically. Instead of using genres or authors, you could think about crafting features yourself, such as topics.

how can we identify similar items?

1. Cosine similarity
2. Soft cosine similarity
3. LDA: topic scores

Benefits

- Content-based recommender systems can be very efficient...
- They are often part of more complex recommender systems that leverage (deep) supervised learning

Drawbacks

- Features that are not part of the user profile will be neglected; e.g., if the user does not like Super Hero movies, the recommender system wil never recommend this.
- does not use the power community data. Recommendations may be obvious (e.g., *Harry Potter* recommendation when you like *Lord of the Rings*)

Collaborative RecSys

Collaborative filtering

What are collaborative systems?

- Tries to overcome some of the limitations of content-based systems
- Leverages the power of the community, tries to give relevant, but also suprising recommendations.
- Very succesful models in industry settings

Types of collaborative systems

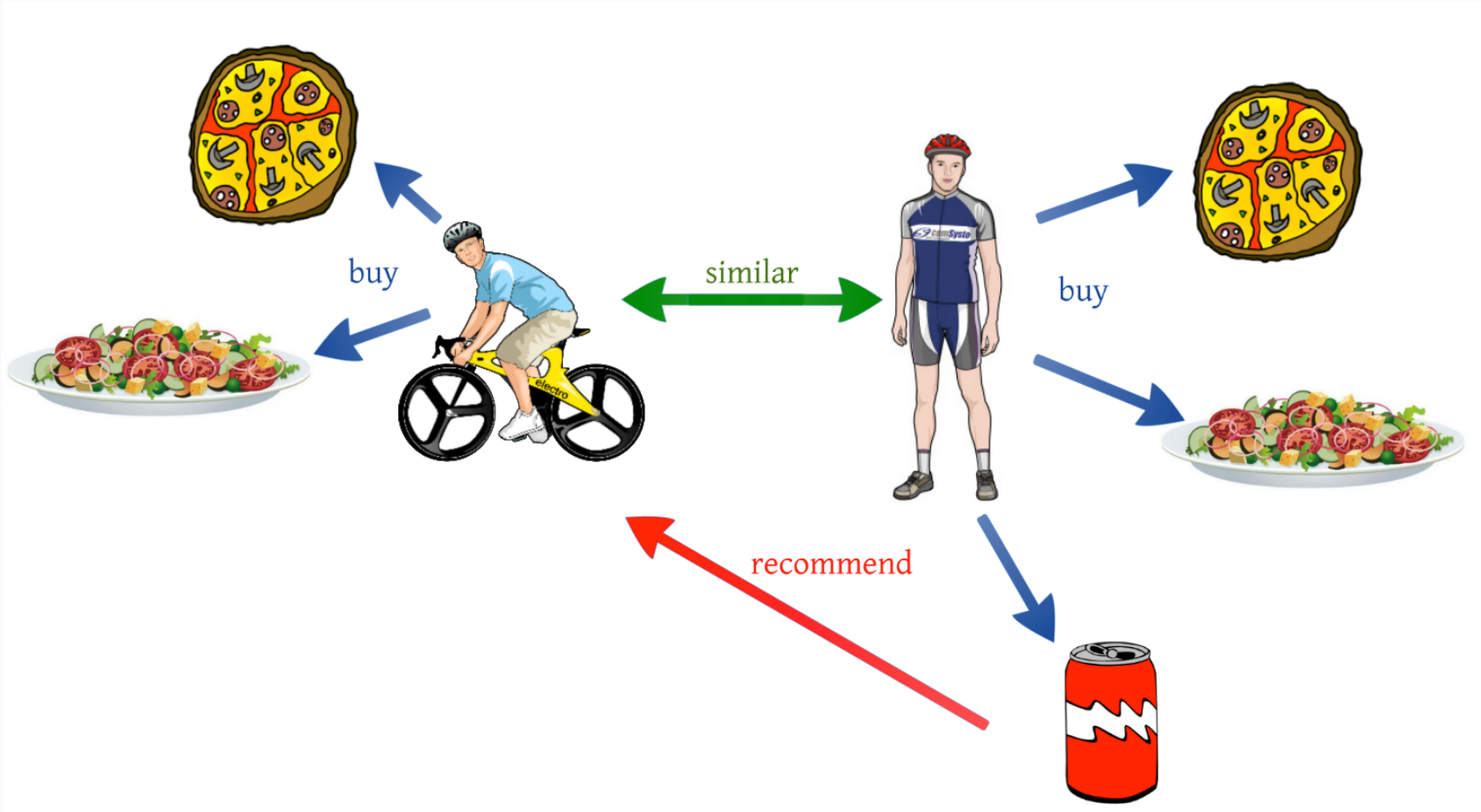
1. User-based collaborative filtering
2. Item-based collaborative filtering

User-based filtering

Similar users...

- If we find users that are similar in terms of the things they liked in the past...
- ...we can use this information to predict what they like in the future

User-based filtering



[1] Source: https://medium.com/@akhilesh3091999/recommender-system-bb032b16ab67

Item-based filtering

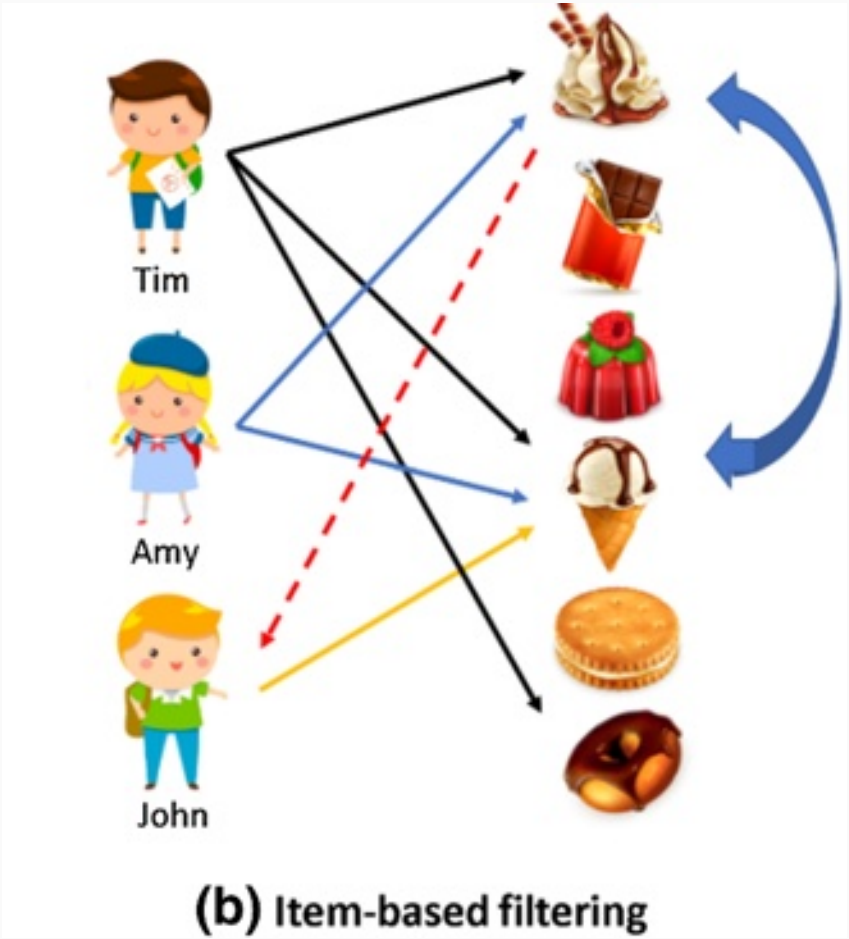
Similar items...

- If two items is rated similarly by a group of people
- ...these product might be similar

Example...

- If Alex, Sanne and Marthe like skincare product X and Y, and Denise buys skincare product Y, X will be recommended.

User-based filtering



(b) Item-based filtering

[2] Source: https://predictivehacks.com/item-based-collaborative-filtering-in-python/

Example: Amazon

Reflection on the role of recommender systems in society

Food for thought

- Consequences of recommender systems for democratic values in society

Build your own recommender system

Practice with the materials!

- To be able to do this correctly, it is essential that you understand the code of this week's lab session.
- Carefully walk through this week's assignment, and to whether questions arise.
- It's up to you to decide whether you want to build a simple knowledge-based or content-based recommender system. Base your selection on the available data columns.

Have fun!

References i

References

Brinberg, M., & Ram, N. (2021).**Do new romantic couples use more similar language over time? Evidence from intensive longitudinal text messages.** *Journal of Communication*, *71*(3), 454–477. https://doi.org/10.1093/joc/jqab012

Hager, A., & Hilbig, H. (2020).**Does public opinion affect political speech?** *American Journal of Political Science*, *64*(4), 921–937. https://doi.org/10.1111/ajps.12516

References ii

Kim, H. S., Yang, S., Kim, M., Hemenway, B., Ungar, L., & Cappella, J. N. (2019).**An experimental study of recommendation algorithms for tailored health communication.** *Computational Communication Research*, *1*(1), 103–129. https://doi.org/10.5117/ccr2019.1.005.sukk

Locherbach, F., & Trilling, D. (2018).**3bij3: A framework for testing effects of recommender systems on news exposure.** *Proceedings - IEEE 14th International Conference on eScience, e-Science 2018*, 350–351. https://doi.org/10.1109/eScience.2018.00093

References iii


Möller, J., Trilling, D., Helberger, N., & van Es, B. (2018).**Do not blame it on the algorithm: an empirical assessment of multiple recommender systems and their impact on content diversity.** *Information Communication and Society*, *21*(7), 959–977. https://doi.org/10.1080/1369118X.2018.1444076


Sidorov, G., Gelbukh, A., Gómez-Adorno, H., & Pinto, D. (2014).**Soft similarity and soft cosine measure: Similarity of features in vector space model.** *Computacion y Sistemas*, *18*(3), 491–504. https://doi.org/10.13053/CyS-18-3-2043

References iv

Wieland, M., Von Nordheim, G., & Kleinen-Von Königslöw, K. (2021). **One recommender fits all? An exploration of user satisfaction with text-based news recommender systems.** *Media and Communication*, *9*(4), 208–221. https://doi.org/10.17645/mac.v9i4.4241