

Recommender systems

Giovanni Colavizza

Text Mining
Amsterdam University College

February 5, 2021

Announcements

- **TBD.**

Overview

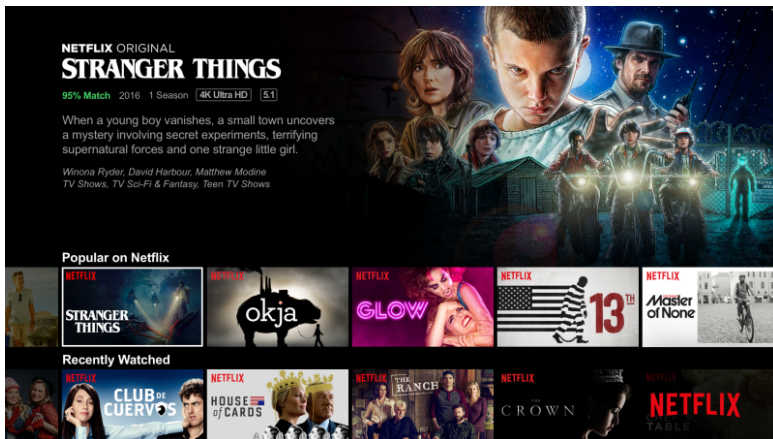
- 1 General concepts
- 2 k-Nearest Neighbours
- 3 More concepts
- 4 Evaluation

General concepts

Definition

- Recommender systems (RS) focus on **predicting the relationship between users and items**.
- The relationship can take many forms: advertisement, preference (e.g., “likes”), retrieval (e.g., of information on the Web).
- How are preference data acquired?
 - ▶ **Implicitly**, e.g., videos watched.
 - ▶ **Explicitly**, e.g., likes on videos.
- How are users and items characterized?
 - ▶ Demographic features;
 - ▶ Social information;
 - ▶ Internet of Things (e.g., geolocation);
 - ▶ Contents;
 - ▶ ...

Examples

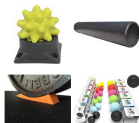


Examples

Recommended for you, Thomas



Literature & Fiction
82 ITEMS



Exercise & Fitness Equipment
8 ITEMS



Health, Fitness & Dieting Books
37 ITEMS



Tableware
12 ITEMS



Prime Video – Unlimited Streaming for
Prime Members
12 ITEMS



Coffee, Tea & Espresso
98 ITEMS



Biographies & Memoirs
17 ITEMS



Engineering Books
7 ITEMS

Examples

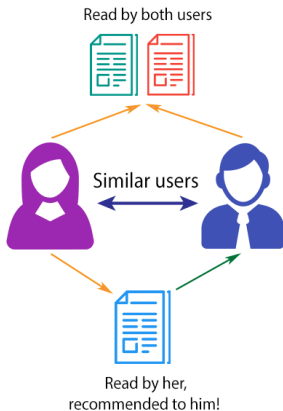


Classification of RS

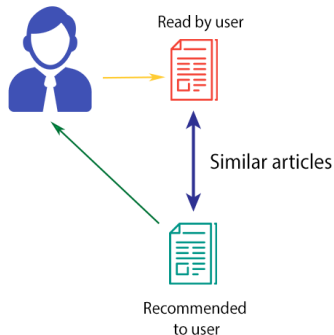
- **Approach: collaborative vs content-based filtering.**
- **Methods: memory vs model-based.**
- Algorithms and techniques.
- Kind and quality of recommendation.
- ...

Collaborative vs content-based filtering

COLLABORATIVE FILTERING



CONTENT-BASED FILTERING



Collaborative vs content-based filtering

- **Collaborative filtering (CF)**: for a given user, recommendations are based on the preferences from those users that have most in common with him/her.
- Intuition: people with similar taste will make similar choices.
- **Content-based filtering (CBF)**: recommend items similar to those that the user considered positively in the past.
- Intuition: recommendations are based on the relevant characteristics of the objects intended for recommendation (text, video, sound).

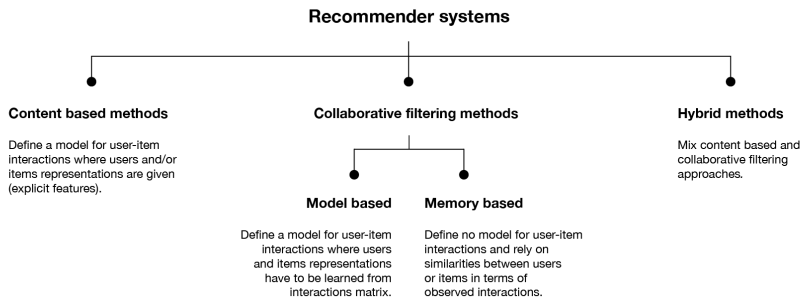
Collaborative vs content-based filtering

- Collaborative filtering (CF): popular methods include **k-Nearest Neighbours** and **matrix factorization**.
- Content-based filtering (CBF): given user profiles and the properties of items, standard algorithms are used to decide if an item should be recommended to a given user (**classification**), or how the given user would score a certain item (**regression**).
- Note: it is rare to find “pure” filtering approached: most systems use combinations, they are thus **hybrid**, and use all the state-of-the-art machinery available (neural networks).

Methods

- **Memory-based:** based on the direct use of data (i.e., they use only the matrix of user-items ratings). These methods are also called *non-parametric* in ML parlance. Example: k-NN.
- **Model-based:** based on a model which uses the data in turn. These methods are also called *parametric* in ML parlance. Example: matrix factorization.
- *Reminder: A parametric model is one that can be parametrized by a finite number of parameters. We encountered this before.*

Classification of RS



https:

[//towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada](https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada)

k-Nearest Neighbours

k-Nearest Neighbours

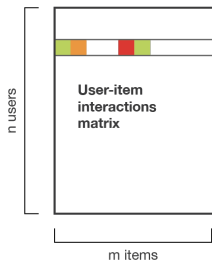
- The most popular *memory-based* algorithm for *collaborative filtering* recommendations;
- the method most popular at the beginning of RS;
- conceptually simple and easy to implement, and it generally produces good predictions;
- based on similarity measures;
- two main versions: **user-to-user** and **item-to-item**.
- Remember the word-context matrix? Let us build a **user-item matrix** now!

User-to-user

positive interactions

neutral interactions

negative interactions



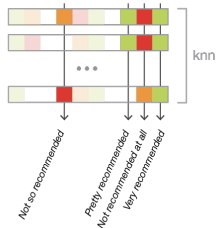
User we want to make a recommendation for is represented by its row in the matrix...



... and we search the K nearest neighbours of this user in the matrix

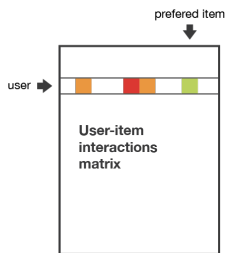


We can then recommend the most popular items among the K nearest neighbours

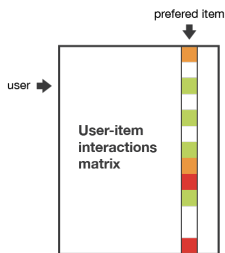


<https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada>

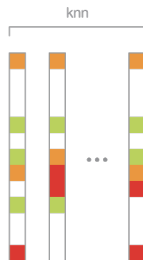
Item-to-item



We identify the preferred item of user we want to make recommendation for.



The preferred item is represented by its column in the matrix.



We can search and recommend the K nearest items to this "preferred item"

<https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada>

k-Nearest Neighbours with user-to-user

Given an active user a :

- ① use a similarity measure to determine the k most-similar users to a ;
- ② obtain the prediction on item i for user a by using one aggregation approaches on the item ratings in a 's neighborhood:
 - ▶ average
 - ▶ weighted sum
 - ▶ adjusted weighted aggregation (deviation-from-mean)
 - ▶ ...
- ③ choose the top- n items by selecting the n items with the highest scores calculated by applying the previous steps on the items that have not yet been rated by user a .

k-Nearest Neighbours with item-to-item

Given an active user a :

- 1 use a similarity measure to determine the k most-similar items to those already rated by a ;
- 2 choose the top- n items by selecting the n items with the highest scores on the items that have not yet been rated by user a .

Similarity measures

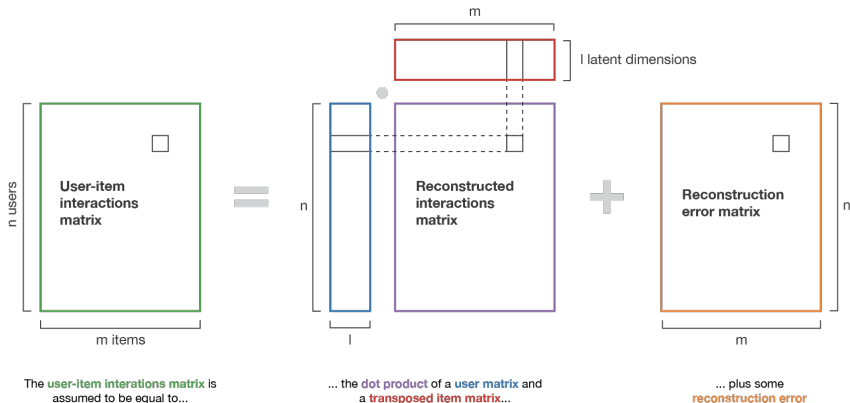
- ① Many options, recall class on word vectors and see <https://docs.scipy.org/doc/scipy/reference/spatial.distance.html>.
- ② *Euclidean*.
- ③ *Cosine*: considers the angle between two vectors, not the magnitude (norm).
- ④ *Pearson* correlation: strength of linear dependence.
- ⑤ It is usually easy to *turn a distance into a similarity* (take the inverse or subtract from the max value it can take).
- ⑥ More during the lab..

User-to-user vs item-to-item

- ① User-to-user does not **scale** well: $O(ndk)$ with n users, d items and k neighbours. Imagine how many users and items are on Amazon..
- ② Item-to-item k-NN usually reduces this scalability problem. $O(d^2)$ but in practice much less, *why?*
- ③ In practice, *approximate k-NN* methods as well as *caching* help a lot.
- ④ Sparsity is a huge issues for both approaches.
- ⑤ User-to-user k-NN is simple to implement and often more appropriate with smaller in-memory datasets that change frequently.
- ⑥ The two approaches can be combined.

More concepts

Matrix factorization



<https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada>

Matrix factorization

- ① We can solve the sparsity problem and use **dense vectors**, as we did with word vectors.
- ② We can then use k-NN on dense vectors instead (or as features for other models).
- ③ Matrix factorization, e.g., SVD, is done using standard optimization techniques: define a loss function (e.g., MSE of the original vs reconstructed matrix) and minimize it.

Cold-start problem

- ❶ Issue raising when we do not have enough data to make reliable recommendations:
 - ▶ **new community or user**: when starting up a new RS or for a new user. Solution 1: encourage users to make ratings (e.g., Netflix). Solution 2: use CF recommendations only when you have enough data. Solution 3: other data and hybrid.
 - ▶ **New item**: new items usually are not rated, so that they are not recommended. Not a big deal if you have items that can be discovered also by other means. Solution: to have a group of motivated users that rate new items or use hybrid.

More challenges

- ① **Rich get richer**: when highly popular items dominate your RS.
- ② Information confinement area (or “**filter bubbles**”): when serendipity does not happen and users get too much of the same.
- ③ Solutions: mix signals, emphasize recency.

Evaluation

Metrics

- ➊ **Prediction metrics** between predicted and actual: mean absolute/squared error, root mean squared error, normalized mean absolute error, ..
- ➋ **Set recommendation metrics**: evaluate the reduced set of recommendations returned by the RS using standard metrics (precision, recall, F-measure, accuracy).
- ➌ **Rank recommendation metrics**: take into consideration the ranking (ordering) of the results.
- ➍ **Novelty metrics**: degree of difference between the recommended items and those known by the user.
- ➎ **Diversity metrics**: degree of differentiation among the recommended items (to keep variety high).
- ➏ **Stability metrics**: how much the predictions produced by an RS change in a given period of time.

A/B Testing



References

- ① J. Bobadilla, F. Ortega, A. Hernando and A. Gutiérrez (2013): *Recommender systems survey*.
- ② T. Segaran (2007): *Programming Collective Intelligence*. (Chapters 2 and 3).
- ③ Recent neural network-based approaches:
 - ▶ <https://link.springer.com/article/10.1007/s10791-017-9321-y>
 - ▶ <https://arxiv.org/pdf/1707.07435.pdf>
 - ▶ <https://arxiv.org/pdf/1907.06902.pdf>