Artificial Intelligence Applied to the Web

Chapter 4 Part 3 - Recommendation Systems

Diego Cornejo, Felipe Hernández and Juan Velásquez

University of Chile
Departament of Industrial Engineering

Spring 2025

Outline

Recommendation Systems

Motivation

- Information overload
  - Many choices available
  - "The paradox of choice" (Jam experiment, choice overload)
- Recommender system
  - Provide aid
  - Set of items + user "context" $\rightarrow$ selection of items (predicted to be "good" for the user)
- What recommender systems do you know?
- What recommender systems would you like to have?

Recommendation Systems

The Paradox of Choice

Figure 1: Fuente: *Your Marketing Rules*

Recommendation Systems

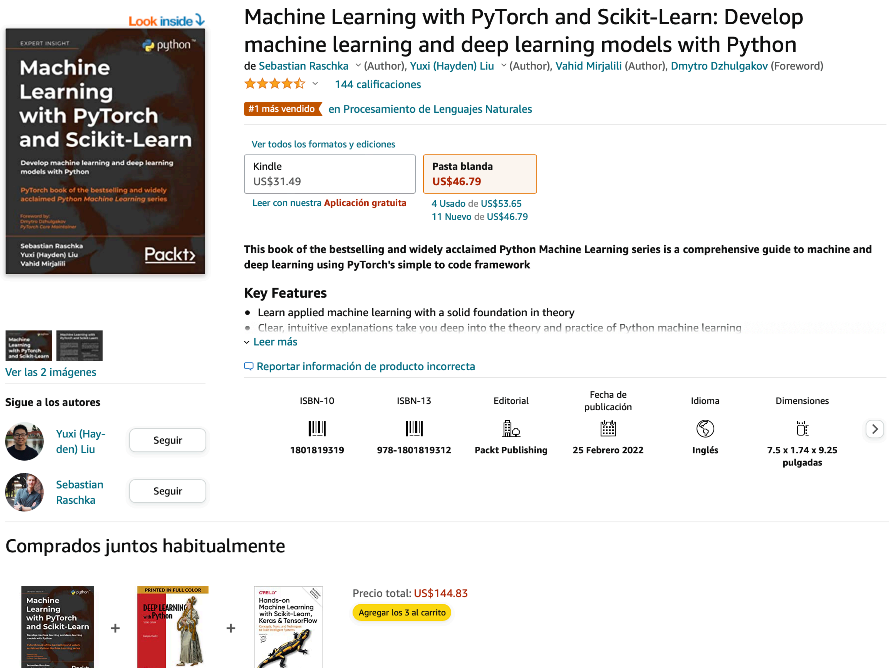
Examples of Applications

- Movies, online videos, music.
- Books
- Software (apps)
- Products in general
- People (dating, friends)
- Services (restaurants, accommodation, ...)
- Research articles
- Jokes

Recommendation Systems

Examples of Applications - Amazon

It uses item-to-item collaborative filtering recommendations on most pages of their website and e-mail campaigns.



Recommendation Systems

Examples of Applications - Netflix

- A data-driven company that leverages recommendation systems to boost customer satisfaction.
- The 75% of Netflix viewing is due to recommendations.
- The Netflix Prize (2009): The most accurate movie recommendation algorithm wins a prize worth $1,000,000.

Recommendation Systems

Examples of Applications - Spotify

- *"Discover Weekly"* which is a personalized list of 30 songs based on users unique music taste.
- *"AI Dj"* a personalized lineup of music recommendations with generative AI.
- Collaborative filtering: Filtering songs by comparing users' historical listening data with other users' listening history.
- Natural language processing: Scraping the Web for information about specific artists and songs.
- Audio file analysis: To analyze the audio file's characteristics (tempo, loudness, key and time signature) to prepare the recommendations.

Recommendation Systems

Where's the value in recommendations?

- Netflix: 2/3 of the movies watched
- Amazon: 35% sales
- Google news: recommendations → 38% more clickthrough

Recommender Systems

Definition

Definition

A recommendation system (or recommender system) is a class of machine learning that uses data to help predict, narrow down, and find what people are looking for among an exponentially growing number of options (NVIDIA).

Recommender Systems

Types

- Collaborative filtering: Considers the information of the user and other similar users.
- Content-based: Uses information about the object and the users past experience.
- Knowledge-based: Uses knowledge about how an object meets a need.
- Community-based: Uses information associated with the users “friends”.
- Hybrid approaches: A combination of the previously mentioned approaches.

Recommender Systems

Functions: Provider's point of view

- Sell more items.
- Sell more diverse items (long tail)
- Increase user satisfaction, fidelity.
- Better understand what users want.

Recommender Systems

Functions: User's point of view

- Looking for something:
  - Find some good items.
  - Find all good items (closer to information retrieval) recommend a sequence, a bundle.
- Just browsing.
- Side-effects (collaborative filtering systems):
  - Express self.
  - Help others.
  - Influence others.

Recommender Systems

The Usefulness of Recommendations

- Implementing recommendations is non-trivial.
- Is it worthwhile? It depends ...
  - Is there “large” number of items?
  - Do users know exactly what are they looking for?

Recommender Systems

RecSys and Information Retrieval

Information Retrieval

Is the activity of obtaining information resources relevant to an information need from a collection of information resources (Wikipedia).

Recommender System

The goal of a Recommender System is to generate meaningful recommendations to a collection of users for items or products that might interest them (Melville, Sindhwani).

Recommender Systems

RecSys and Information Retrieval

- RecSys and IR closely connected (many similar or analogical techniques)
- Different goals:
  - IR – “I know what I’m looking for”
  - RecSys – “I’m not sure what I’m looking for”

Recommender Systems

Serendipity

- Unsought finding: unexpected, but useful result.
- Do not recommend items the user already knows or would find anyway, try something more interesting
- Example – books:
  - I like books by Remarque, Potok, Skacel recommending
  - Another book by Remarque not very useful.
  - Recommending Munro = Serendipity.

Recommender Systems

Warning: Implementing Personalized Systems is Difficult

- (Sometimes) complex algorithms.
- (Always) difficult debugging, testing, evaluation.
  - Personalization $\rightarrow$ different behaviour for each user
  - Hard to distinguish bugs and surprising results

Recommender Systems

Collaborative Filtering (CF)

- The most prominent approach to generate recommendations
  - Used by large, commercial e-commerce sites
  - Well-understood, various algorithms and variations exist
  - Applicable in many domains (book, movies, DVDs, ..)
- Approach
  - Use the *wisdom of the crowd* to recommend items
- Basic assumption and idea
  - Users give ratings to catalog items (implicitly or explicitly)
  - Customers who had similar tastes in the past, will have similar tastes in the future

Collaborative Filtering

Pure CF Approaches

- Input
  - Only a matrix of given user–item ratings
- Output types
  - A (numerical) prediction indicating to what degree the current user will like or dislike a certain item
  - A top-N list of recommended items

Collaborative Filtering

User-based nearest-neighbor collaborative filtering (1)

- Main idea
  - Given a ratings database and a user u, identify other users that had similar preferences to those of u in the past. We refer to these users as *nearest neighbors.*
  - Then, for every p item that u has not seen, a prediction is computed based on the ratings for p made by the nearest neighbors.

Collaborative Filtering

User-based nearest-neighbor collaborative filtering (2)

- Example: A database of ratings of the current user, Alice, and some other users is given

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

- Determine whether Alice will like or dislike Item5, which Alice has not yet seen.

Collaborative Filtering

User-based nearest-neighbor collaborative filtering (3)

- Some first questions
  - How do we measure similarity?
  - How many neighbors should we consider?
  - How do we generate a prediction from the neighbors' ratings?

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

Collaborative Filtering

Measuring user similarity (1)

- A popular similarity measure in user-based CF: Pearson correlation
  - a, b: users
  - $r_{a,p}$: rating of a user a for a item p.
  - P: set of items, rated both by users a and b.
- Possible similarity values between -1 and 1.

$$\mathrm{sim}(a,b) = \frac{\sum_{p\in P}\left(r_{a,p}-\overline{r}_a\right)\left(r_{b,p}-\overline{r}_b\right)}{\sqrt{\sum_{p\in P}\left(r_{a,p}-\overline{r}_a\right)^2}\sqrt{\sum_{p\in P}\left(r_{b,p}-\overline{r}_b\right)^2}}$$

Collaborative Filtering

Measuring user similarity (2)

- A popular similarity measure in user-based CF: Pearson correlation
  - a, b: users
  - $r_{a,p}$: rating of a user a for a item p.
  - P: set of items, rated both by users a and b.
- Possible similarity values between -1 and 1.

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

sim = 0,85

sim = 0,00

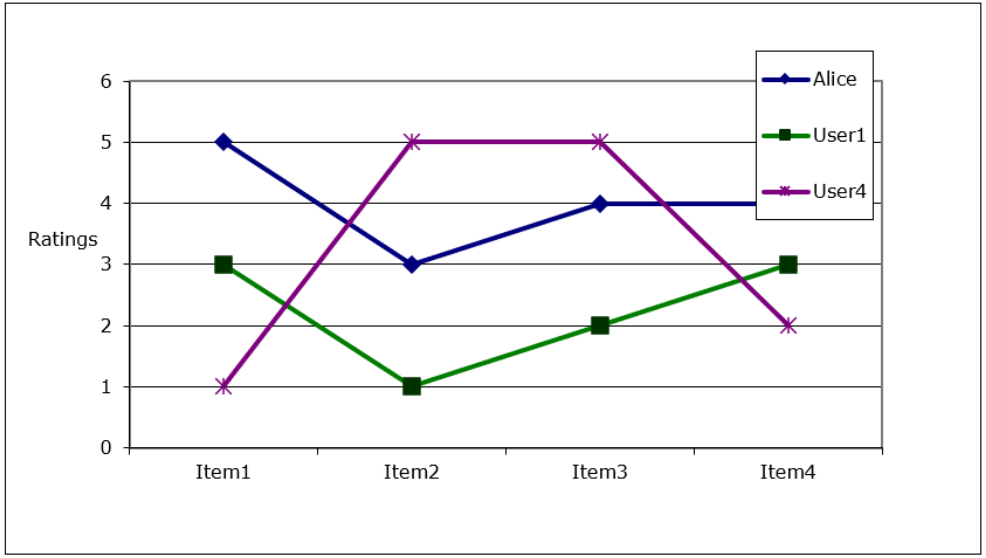
sim = 0,70

sim = -0,79

Collaborative Filtering

Pearson correlation

- Takes differences in rating behavior into account



- Works well in usual domains, compared with alternative measures, such as cosine similarity

Collaborative Filtering

Making predictions

A common prediction function:

$$\text{pred}(a,p) = \overline{r}_a + \frac{\sum_{b \in N} \left[\text{sim}(a,b) * \left(r_{b,p} - \overline{r}_b\right)\right]}{\sum_{b \in N} \text{sim}(a,b)}$$

- Calculate, whether the neighbors' ratings for the unseen item i are higher or lower than their average
- Combine the rating differences – use the similarity with a as a weight
- Add/subtract the neighbors' bias from the active user's average and use this as a prediction

Collaborative Filtering

Improving the metrics / prediction function

- Not all neighbor ratings might be equally "valuable"
  - Agreement on commonly liked items is not so informative as agreement on controversial items
  - **Possible solution:** Give more weight to items that have a higher variance
- Case amplification
  - Intuition: Give more weight to "very similar" neighbors, i.e., where the similarity value is close to 1.
- Neighborhood selection
  - Use similarity threshold or fixed number of neighbors

Collaborative Filtering

Memory-based and model-based approaches

- User-based CF is said to be "memory-based"
  - the rating matrix is directly used to find neighbors / make predictions
  - does not scale for most real-world scenarios
  - large e-commerce sites have tens of millions of customers and millions of items

Collaborative Filtering

Memory-based and model-based approaches

- Model-based approaches
  - based on an offline pre-processing or “model-learning” phase
  - at run-time, only the learned model is used to make predictions
  - models are updated / re-trained periodically
  - large variety of techniques used
  - model-building and updating can be computationally expensive
  - item-based CF is an example for model-based approaches

Collaborative Filtering

Item-based collaborative filtering

- Basic idea
  - Use the similarity between items (and not users) to make predictions
- Example:
  - Look for items that are similar to Item5
  - Take Alice's ratings for these items to predict the rating for Item5

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

Collaborative Filtering

Item-based collaborative filtering

- Produces better results in item-to-item filtering
- Ratings are seen as vector in n-dimensional space
- Similarity is calculated based on the angle between the vectors

$$\mathrm{sim}\left(\vec{a}, \vec{b}\right) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * \left|\vec{b}\right|}$$

Collaborative Filtering

Item-based collaborative filtering

- Adjusted cosine similarity
  - Take average user ratings into account, transform the original ratings
  - U: set of users who have rated both items a and b

$$\operatorname{sim}(a,b) = \frac{\sum_{u\in U}\left(r_{u,a}-\bar{r}_u\right)\left(r_{u,b}-\bar{r}_u\right)}{\sqrt{\sum_{u\in U}\left(r_{u,a}-\bar{r}_u\right)^2}\sqrt{\sum_{u\in U}\left(r_{u,b}-\bar{r}_u\right)^2}}$$

Collaborative Filtering

Making predictions

- A common prediction function:

$$\text{pred}(u,p) = \frac{\sum_{i \in \text{ ratedItem}(u)} \text{sim}(i,p) * r_{u,i}}{\sum_{i \in \text{ ratedItem}(u)} \text{sim}(i,p)}$$

- Neighborhood size is typically also limited to a specific size
- Not all neighbors are taken into account for the prediction
- An analysis of the MovieLens dataset indicates that “in most real-world situations, a neighborhood of 20 to 50 neighbors seems reasonable” (Herlocker et al. 2002)

Collaborative Filtering

Pre-processing for item-based filtering

- Item-based filtering does not solve the scalability problem itself
- Pre-processing approach by Amazon.com (in 2003)
  - Calculate all pair-wise item similarities in advance
  - The neighborhood to be used at run-time is typically rather small, because only items are taken into account which the user has rated
  - Item similarities are supposed to be more stable than user similarities

Collaborative Filtering

Pre-processing for item-based filtering

- Memory requirements
  - Up to N^2 pair-wise similarities to be memorized (N = number of items) in theory
  - In practice, this is significantly lower (items with no co-ratings)
  - Further reductions possible
    - Minimum threshold for co-ratings
    - Limit the neighborhood size (might affect recommendation accuracy)

Collaborative Filtering

More on ratings – Explicit ratings

- Probably the most precise type of rating.
- Optimal scale is an area of research, more granular scales can capture preferences better and improve the user experience.
- Main problem is data sparsity. Users are often reluctant to provide ratings, leading to sparse matrices and poor recommendations.

Collaborative Filtering

More on ratings – Implicit ratings

- User actions automatically observed and logged by the system (e.g., purchases, clicks, time on page).
- They are abundant and easy to collect, as they require no extra effort from the user.
- The main problem: ambiguity. You can't be certain if an action truly reflects a user's preference (e.g., was a book bought for themselves or as a gift?).
- Implicit ratings are more plentiful but noisier than explicit ratings.

Collaborative Filtering

Data sparsity problems

- Cold start problem
  - How to recommend new items? What to recommend to new users?
- Straightforward approaches
  - Ask/force users to rate a set of items
  - Use another method (e.g., content-based, demographic or simply non-personalized) in the initial phase
  - Default voting: assign default values to items that only one of the two users to be compared has rated (Breese et al. 1998)

Collaborative Filtering

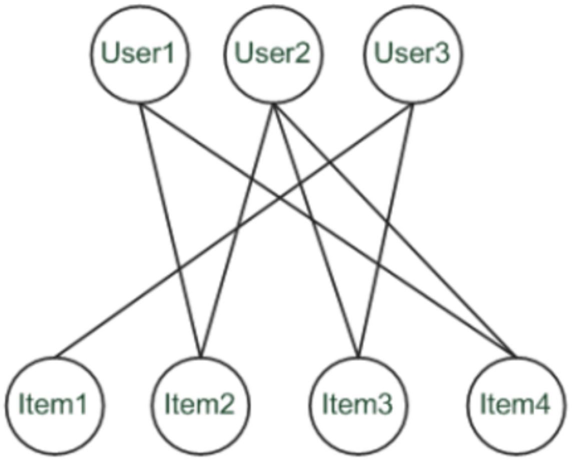
Data sparsity problems

- Alternatives
  - Use better algorithms (beyond nearest-neighbor approaches)
  - Example:
    - In nearest-neighbor approaches, the set of sufficiently similar neighbors might be too small to make good predictions
    - Assume “transitivity” of neighborhoods

Collaborative Filtering

Graph-based methods (1)

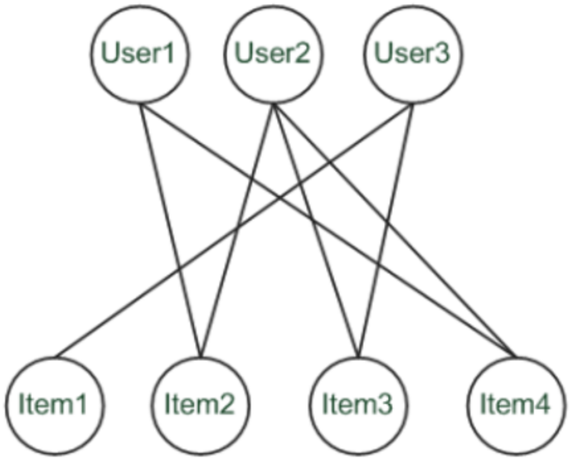
- “Spreading activation” (Huang et al. 2004)
  - Exploit the supposed “transitivity” of customer tastes and thereby augment the matrix with additional information
  - Assume that we are looking for a recommendation for User1
  - When using a standard CF approach, User2 will be considered a peer for User1 because they both bought Item2 and Item4
  - Thus Item3 will be recommended to User1 because the nearest neighbor, User2, also bought or liked it



Collaborative Filtering

Graph-based methods (2)

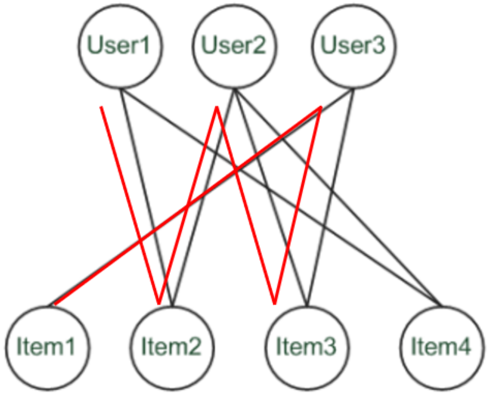
- "Spreading activation" (Huang et al. 2004)
  - In a standard user-based or item-based CF approach, paths of length 3 will be considered – that is, Item3 is relevant for User1 because there exists a three-step path (User1–Item2–User2–Item3) between them
  - Because the number of such paths of length 3 is small in sparse rating databases, the idea is to also consider longer paths (indirect associations) to compute recommendations
  - Using path length 5, for instance



Collaborative Filtering

Graph-based methods (3)

- “Spreading activation” (Huang et al. 2004)
  - Idea: Use paths of lengths > 3 to recommend items
  - Length 3: Recommend Item3 to User1
  - Length 5: Item1 also recommendable



Collaborative Filtering

More model-based approaches

- Plethora of different techniques proposed in the last years, e.g.,
  - Matrix factorization techniques, statistics
    - Singular Value Decomposition (SVD), Principal Component Analysis (PCA)
  - Association rule mining
    - Compare: shopping basket analysis
  - Probabilistic models
    - Clustering models, Bayesian networks, probabilistic Latent Semantic Analysis (pLSA)
  - Various other machine learning approaches
- Costs of pre-processing
  - Usually not discussed
  - Incremental updates possible?

Collaborative Filtering

2000: Application of Dimensionality Reduction in Recommender System, B. Sarwar et al., WebKDD Workshop

- Basic idea: Trade more complex offline model building for faster online prediction generation
- Singular Value Decomposition for dimensionality reduction of rating matrices
  - Captures important factors/aspects and their weights in the data
  - factors can be genre, actors but also non-understandable ones
  - Assumption that k dimensions capture the signals and filter out noise (K = 20 to 100)
- Constant time to make recommendations
- Approach also popular in IR (Latent Semantic Indexing), data compression,...

Collaborative Filtering

Matrix factorization

- Informally, the SVD theorem (Golub and Kahan 1965) states that a given matrix M can be decomposed into a product of three matrices as follows

$$M = U \times \Sigma \times V^T$$

- where U and V are called *left* and *right* singular vectors and the values of the diagonal of Σ are called the singular values
- We can approximate the full matrix by observing only the most important features – those with the largest singular values
- In the example, we calculate U, V and Σ (with the help of some linear algebra software) but retain only the two most important features by taking only the first two columns of U and V^T

Collaborative Filtering

Example for SVD-based recommendation

SVD: $M_k = U_k \times \Sigma_k \times V_k^T$

U_k	Dim1	Dim2
Alice	0.47	-0.30
Bob	-0.44	0.23
Mary	0.70	-0.06
Sue	0.31	0.93

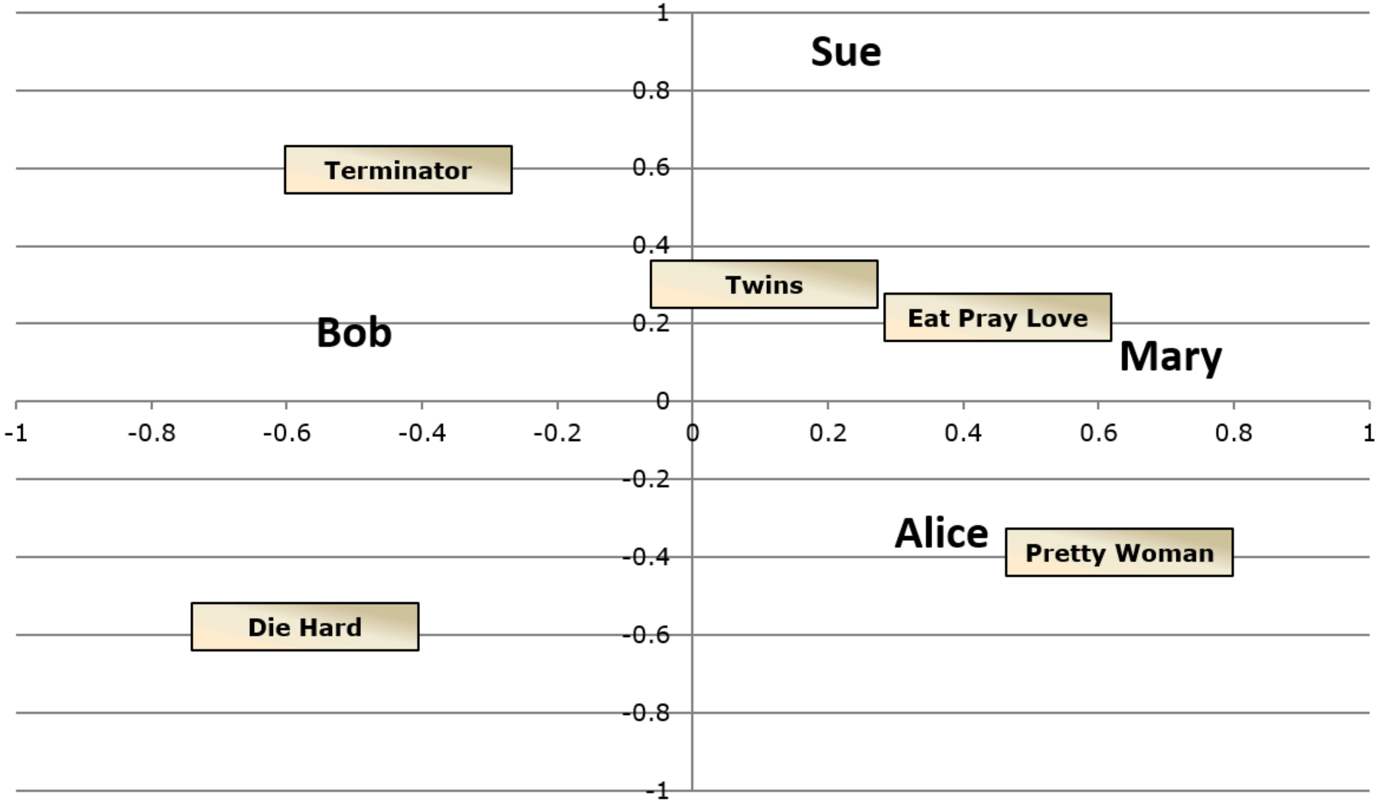
V_k^T	Terminator	Die Hard	Twins	Eat Pray Love	Pretty Woman
Dim1	-0.44	-0.57	0.06	0.38	0.57
Dim2	0.58	-0.66	0.26	0.18	-0.36

Σ_k	Dim1	Dim2
Dim1	5.63	0
Dim2	0	3.23

Prediction: $\hat{r}_{ui} = \overline{r}_u + U_{k(\text{Alice})} \times \Sigma_k \times V_k^{T(\text{EPL})}$

Collaborative Filtering

The projection of U and V^T in the 2 dimensional space (U_2, V_2^T)



Collaborative Filtering

Discussion about dimensionality reduction (Sarwar et al. 2000a)

- Matrix factorization
  - Generate low-rank approximation of matrix
  - Detection of latent factors
  - Projecting items and users in the same n-dimensional space
- Prediction quality can decrease because...
  - the original ratings are not taken into account

Collaborative Filtering

Discussion about dimensionality reduction (Sarwar et al. 2000a)

- Prediction quality can increase as a consequence of...
  - filtering out some “noise” in the data and
  - detecting nontrivial correlations in the data
- Depends on the right choice of the amount of data reduction
  - number of singular values in the SVD approach
  - Parameters can be determined and fine-tuned only based on experiments in a certain domain
  - Koren et al. 2009 talk about 20 to 100 factors that are derived from the rating patterns

Collaborative Filtering

Association rule mining

- Commonly used for shopping behavior analysis
  - aims at detection of rules such as
  - "If a customer purchases beer then he also buys diapers in 70% of the cases"
- Association rule mining algorithms
  - can detect rules of the form $X \to Y$ (e.g. beers $\to$ diapers) from a set of sales transactions $D = \{t_1, t_2, ..., t_n\}$
  - measure of quality: support, confidence
    - used e.g. as a threshold to cut off unimportant rules
- let $\mathrm{freq}(X) = \{x \mid x \subseteq t_i, t_i \in D\}$
- support $= \frac{\mathrm{freq}(X \cup Y)}{|D|}$, confidence $= \frac{\mathrm{freq}(X \cup Y)}{\mathrm{freq}(X)}$

Collaborative Filtering

Recommendation based on Association Rule Mining

- Simplest approach
  - Transform 5-point ratings into binary ratings (1 = above user average)
- Mine rules such as
  - Item1 $\rightarrow$ Item5
    - Support (2/4), confidence (2/2) (without Alice)

	Item1	Item2	Item3	Item4	Item5
Alice	1	0	0	0	?
User1	1	0	1	0	1
User2	1	0	1	0	1
User3	0	0	0	1	1
User4	0	1	1	0	0

Collaborative Filtering

Recommendation based on Association Rule Mining

- Make recommendations for Alice (basic method)
  - Determine "relevant" rules based on Alice's transactions (the above rule will be relevant as Alice bought Item1)
  - Determine items not already bought by Alice
  - Sort the items based on the rules' confidence values
- Different variations possible
  - dislike statements, user associations ...

	Item1	Item2	Item3	Item4	Item5
Alice	1	0	0	0	?
User1	1	0	1	0	1
User2	1	0	1	0	1
User3	0	0	0	1	1
User4	0	1	1	0	0

Collaborative Filtering

Probabilistic methods

- Basic idea (simplistic version for illustration):
  - Given the user/item rating matrix
  - Determine the probability that user Alice will like an item i
  - Base the recommendation on such these probabilities

Collaborative Filtering

Probabilistic methods

- Calculation of rating probabilities based on Bayes Theorem
  - How probable is rating value “1” for Item5 given Alice’s previous ratings?
  - Corresponds to conditional probability $P(\text{Item5} = 1 \mid X)$, where
    - X = Alice’s previous ratings = (Item1 =1, Item2=3, Item3= ...)
  - Can be estimated based on Bayes’ Theorem

$$P(Y|X) = \frac{P(X|Y) \times P(Y)}{P(X)}$$

,

$$P(Y|X) = \frac{\prod_{i=1}^{d} P(X_i|Y) \times P(Y)}{P(X)}$$

Collaborative Filtering

Probabilistic methods (2)

	Item1	Item2	Item3	Item4	Item5
Alice	1	3	3	2	?
User1	2	4	2	2	4
User2	1	3	3	5	1
User3	4	5	2	3	3
User4	1	1	5	2	1

X = (Item1 =1, Item2=3, Item3= …)

$$P(X|Item5=1)$$
$$= P(Item1=1|Item5=1) \times P(Item2=3|Item5=1)$$
$$\times P(Item3=3|Item5=1) \times P(Item4=2|Item5=1) = \frac{2}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2}$$
$$\approx 0.125$$
$$P(X|Item5=2)$$
$$= P(Item1=1|Item5=2) \times P(Item2=3|Item5=2)$$
$$\times P(Item3=3|Item5=2) \times P(Item4=2|Item5=2) = \frac{0}{0} \times \cdots \times \cdots \times \cdots$$
$$= 0$$

Collaborative Filtering

Probabilistic methods (3)

- More to consider
  - Zeros (smoothing required)
  - like/dislike simplification possible

Collaborative Filtering

Practical probabilistic approaches

- Use a cluster-based approach (Breese et al. 1998)
  - assume users fall into a small number of subgroups (clusters)
  - Make predictions based on estimates
    - probability of Alice falling into cluster c
    - probability of Alice liking item i given a certain cluster and her previous ratings
    - $P(C = c, v_1, ..., v_n) = P(C = c) \prod_{i=1}^{n} P(v_i | C = c)$
  - Based on model-based clustering (mixture model)
    - Number of classes and model parameters have to be learned from data in advance (EM algorithm)

Collaborative Filtering

Practical probabilistic approaches

- Others:
  - Bayesian Networks, Probabilistic Latent, ...
- Empirical analysis shows:
  - Probabilistic methods lead to relatively good results (movie domain)
  - No consistent winner; small memory-footprint of network model

Collaborative Filtering

RF-Rec predictors (Gedikli et al. 2011)

- Idea: Take rating frequencies into account for computing a prediction
- Basic scheme: $\hat{r}_{u,i} = \text{argmax}_{v \in R} f_{\text{user}(u,v)} * f_{\text{item}(i,v)}$
  - R: Set of all rating values, e.g., $R = \{1, 2, 3, 4, 5\}$ on a 5-point rating scale
  - $f_{\text{user}(u,v)}$ and f_{item} basically describe how often a rating v was assigned by user u and to item i respectively.

Collaborative Filtering

RF-Rec predictors (Gedikli et al. 2011)

- Example:

	Item1	Item2	Item3	Item4	Item5
Alice	1	1	?	5	4
User1	2		5	5	5
User2			1	1	
User3		5	2		2
User4	3		1	1	
User5	1	2	2		4

- $p(\text{Alice}, \text{Item3}) = 1$

Collaborative Filtering

Collaborative Filtering Issues

- Pros:
  - Well-understood, works well in some domains, no knowledge engineering required
- Cons:
  - Requires user community, sparsity problems, no integration of other knowledge sources, no explanation of results

Artificial Intelligence Applied to the Web

Chapter 4 Part 3 - Recommendation Systems

Diego Cornejo, Felipe Hernández and Juan Velásquez

University of Chile
Departament of Industrial Engineering

Spring 2025