Recommendation Systems
Hybrid approaches

Thanks for source slides and material to: J. Leskovec, A. Rajaraman, J. Ullman: Mining of Massive Datasets

Three Approaches to Recommendation Systems

- **1) Content-based**
  - Use characteristics of an item
  - Recommend items that have similar content to items user liked in the past
  - Or items that match pre-defined attributes of the user
- **2) Collaborative filtering**
  - Build a model from a user's past behavior (items previously purchased or rated) and similar decisions made by other users
  - Use the model to predict items that the user may like
  - Collaborative: suggestions made to a user utilize information across the entire user base
- **3) Hybrid approaches**

HYBRID RECOMMENDER SYSTEMS

Hybrid Recommender Systems

- **Implement two or more different recommenders and combine predictions**
  - Perhaps using a linear model

- **Example: Add content-based methods to collaborative filtering**
  - Item profiles for **new item problem**
  - Demographics to deal with **new user problem**

- **Additional reading: "Hybrid Web Recommender Systems" by Robin Burke**
  - Burke, R. (2007). Hybrid web recommender systems. In *The adaptive web* (pp. 377-408). Springer Berlin Heidelberg.
  - Some slides adapted from PPT by Jae-wook Ahn.

Hybrid Recommender Systems

- **Mix of recommender systems**
- **Recommender system classification based on its knowledge source**
  - **Collaborative Filtering (CF)**
    - User ratings only
  - **Content-based (CN)**
    - Product features, user ratings
    - Classifications of users' likes/dislikes
  - **Demographic**
    - User ratings, user demographics
  - **Knowledge-based (KB)**
    - Domain knowledge, product features, user's need/query
    - Inferences about a user's needs and preferences.

Netflix Example

- Before green-lighting *House of Cards*, Netflix knew:
  - A lot of users watched the David Fincher directed movie *The Social Network* from beginning to end
  - The British version of "*House of Cards*" has been well watched
  - Those who watched the British version "*House of Cards*" also watched specific actor films and/or films directed by David Fincher.
- Netflix Used Big Data To Identify The Movies That Are Too Scary To Finish
  - A typical Netflix customer will lose interest in 60 to 90 seconds when choosing something to watch
  - 80% of the content we watch on Netflix is influenced by the company's recommendation system.

Read: https://blog.kissmetrics.com/how-netflix-uses-analytics/

Ref: @bernardmarr

Strategies for Hybrid Recommendation

- **Combination of multiple recommendation techniques together for producing output**

- Different techniques of ***different types***
  - Most common implementations
  - Most promise to resolve cold-start problem

- Different techniques of the ***same type***
  - Example: NewsDude – naïve Bayes + k-nearest neighbor.

Seven Types of Hybrid Recommender Systems (Taxonomy by Burke, 2002)

1. **Weighted:** The score of different recommendation components are combined numerically
2. **Switching:** The system chooses among recommendation components and applies the selected one.
3. **Mixed:** Recommendations from different recommenders are presented together.
4. **Feature Combination:** Features derived from different knowledge sources are combined together and given to a single recommendation algorithm.
5. **Feature Augmentation:** One recommendation technique is used to compute a feature or set of features, which is then part of the input to the next technique.
6. **Cascade:** Recommenders are given strict priority, with the lower priority ones breaking ties in the scoring of the higher ones.
7. **Meta-level:** One recommendation technique is applied and produces some sort of model, which is then the input used by the next technique.

1. WEIGHTED HYBRID

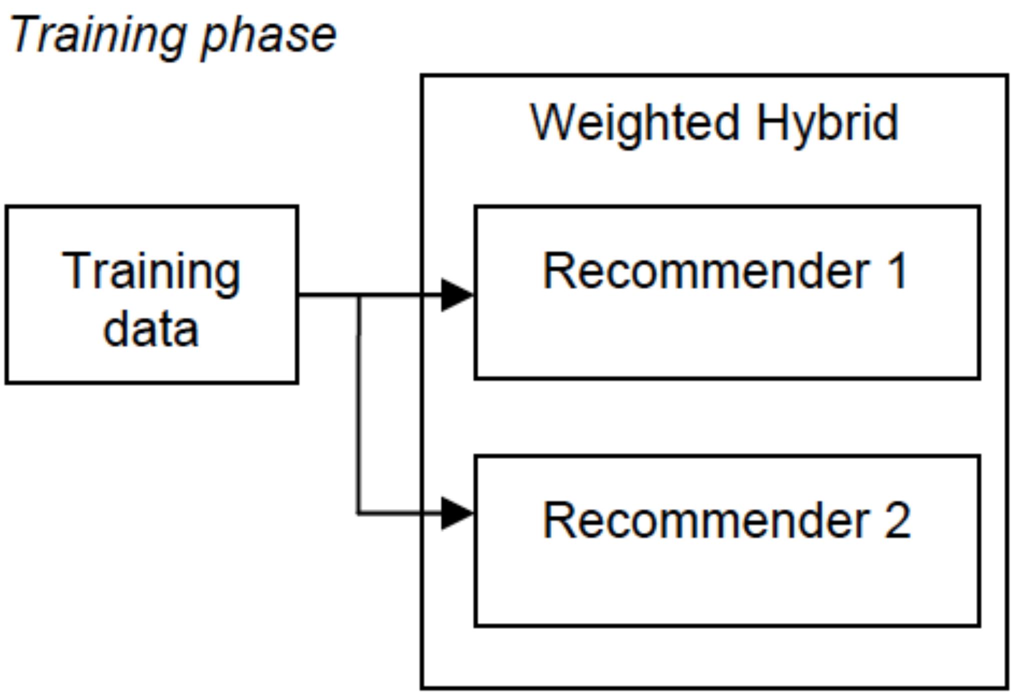
1. Weighted Hybrid

- Each component of the hybrid scores a given item and the ***scores are combined*** using a ***linear formula***
- Combines evidence from both recommenders in a **static manner** (weighting doesn't change)
- Appropriate when component recommenders have consistent relative accuracy across the product space

- The movie recommender system in [32] has two components: one, using **collaborative techniques**, identifies **similarities between rating profiles and makes predictions** based on this information.
- The second component uses **simple semantic knowledge** about the **features of movies**, compressed dimensionally via latent semantic analysis, and recommends movies that are **semantically similar to those the user likes**.
- The output of the two components is combined using a linear weighting scheme.

1. Weighted Hybrid

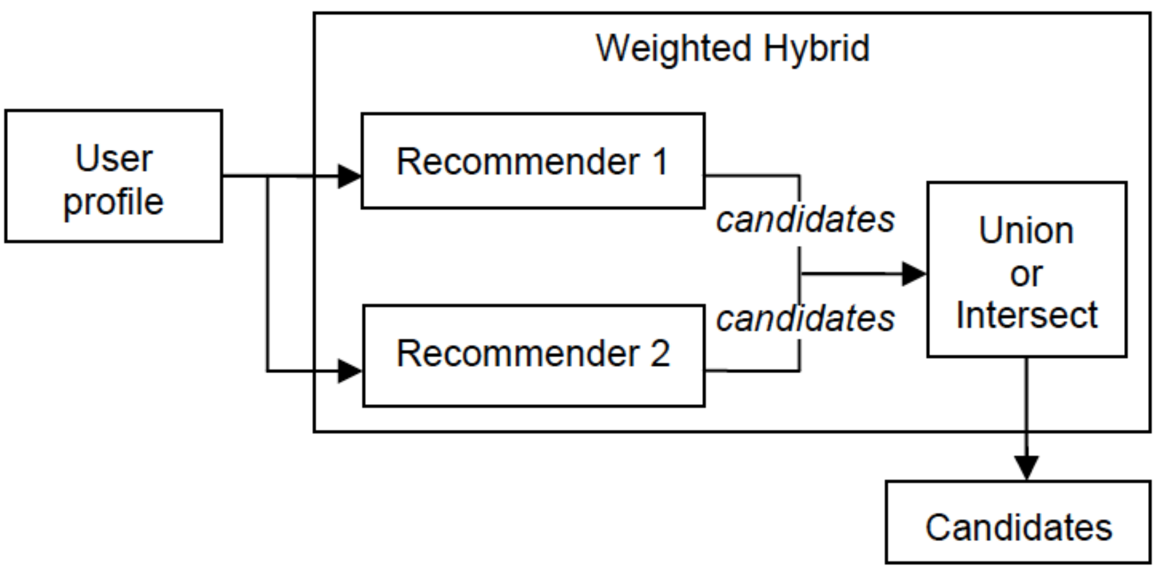
- **Training phase: each individual recommender processes the training data**
- Note: all hybrid algorithms include training phase



1. Weighted Hybrid

- **Candidate generation: When a prediction is being generated for a test user, the recommenders jointly propose candidates**
  - **Some** recommendation techniques (e.g., content-based algorithms) make **predictions on any item**
  - **Others are limited** (e.g., collaborative filtering **can't make predictions** if there are **no peer users** who have rated the item)
  - **Candidate generation necessary to identify items that will be considered**

Candidate generation



1. Weighted Hybrid

- **The sets of candidates must then be rated jointly**

Two approaches:

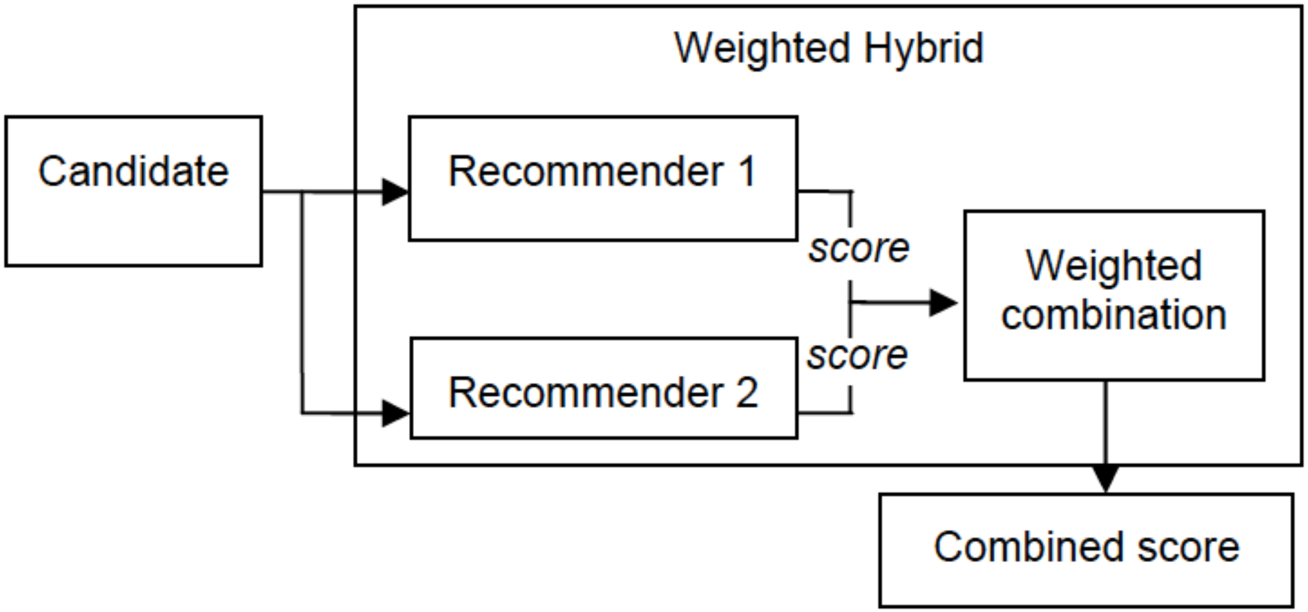
- **Intersection of candidate sets**: possible only a small number of candidates shared between candidate sets
- **Union of candidate sets:** must decide how to handle cases in which it is not possible for a recommender to rate a given candidate
  - May give such a candidate a neutral (neither liked nor disliked) score.

1. Weighted Hybrid

Scoring:

- Each candidate is then rated by the two recommenders
- A linear combination of the two scores computed, which becomes the item's predicted rating

Scoring



1. Weighted Hybrid

Weighted hybrid recommender:

- Burke classification discusses static, linear combination of weights
- More generally: Weights can be linear combination, use **weighted majority voting** or **weighted average voting**
- Example: P-Tango system
  - **Initially: CF and content-based recommenders have equal weight**
  - **Gradually adjusts weighting as predictions are confirmed or found incorrect.**

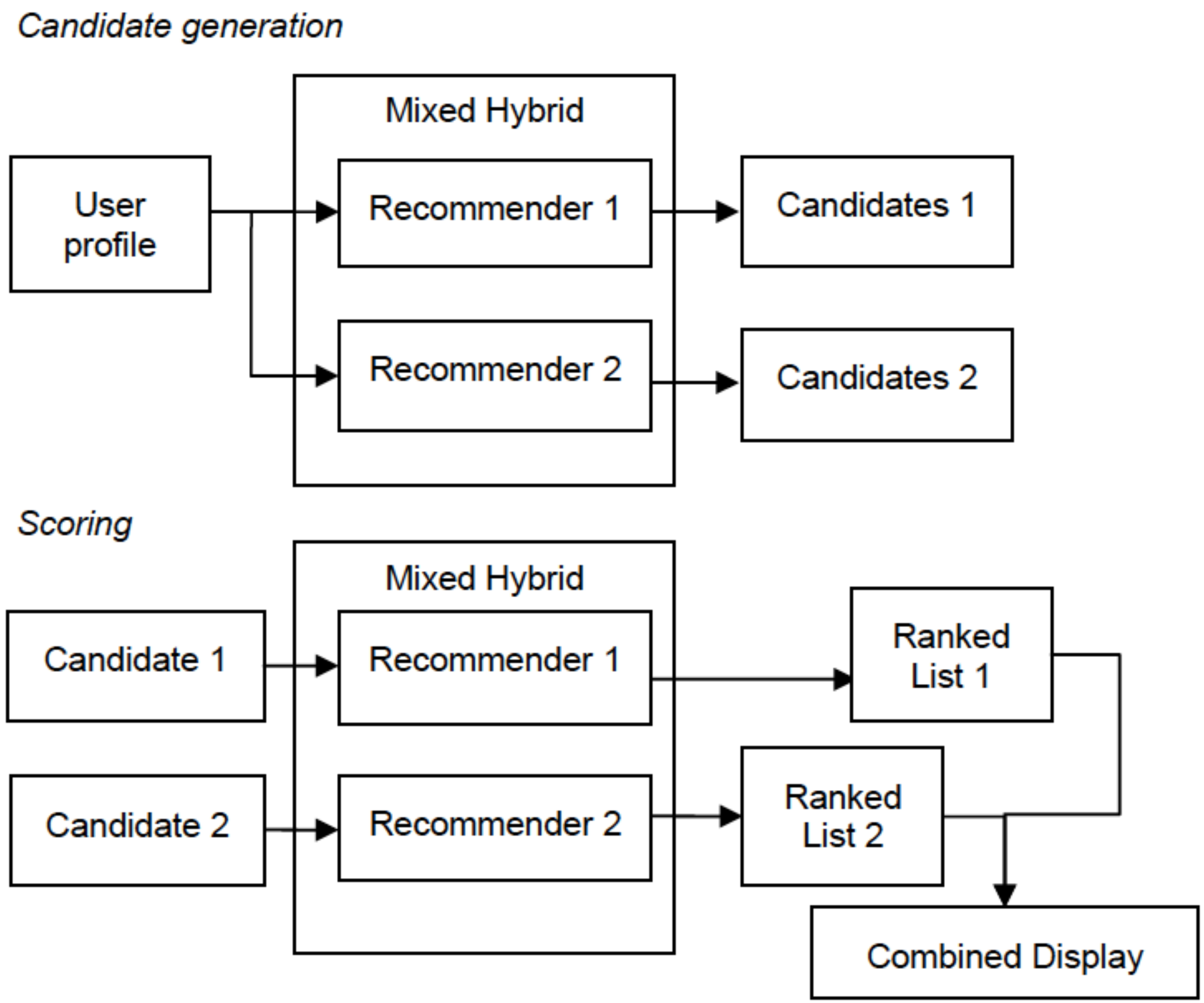
2. MIXED HYBRID

2. Mixed Hybrid

- **A mixed hybrid presents recommendations of its different components side-by-side in a combined list**
- There is **no attempt to combine evidence between recommenders**

- PTV recommends television shows [48]. It has both content-based and collaborative components, but because of the sparsity of the ratings and the content space, it is difficult to get both recommenders to produce a rating for any given show.
- Instead the components **each produce their own set of recommendations** that are combined before being shown to the user.

2. Mixed Hybrid



3. SWITCHING HYBRID

3. Switching Hybrid

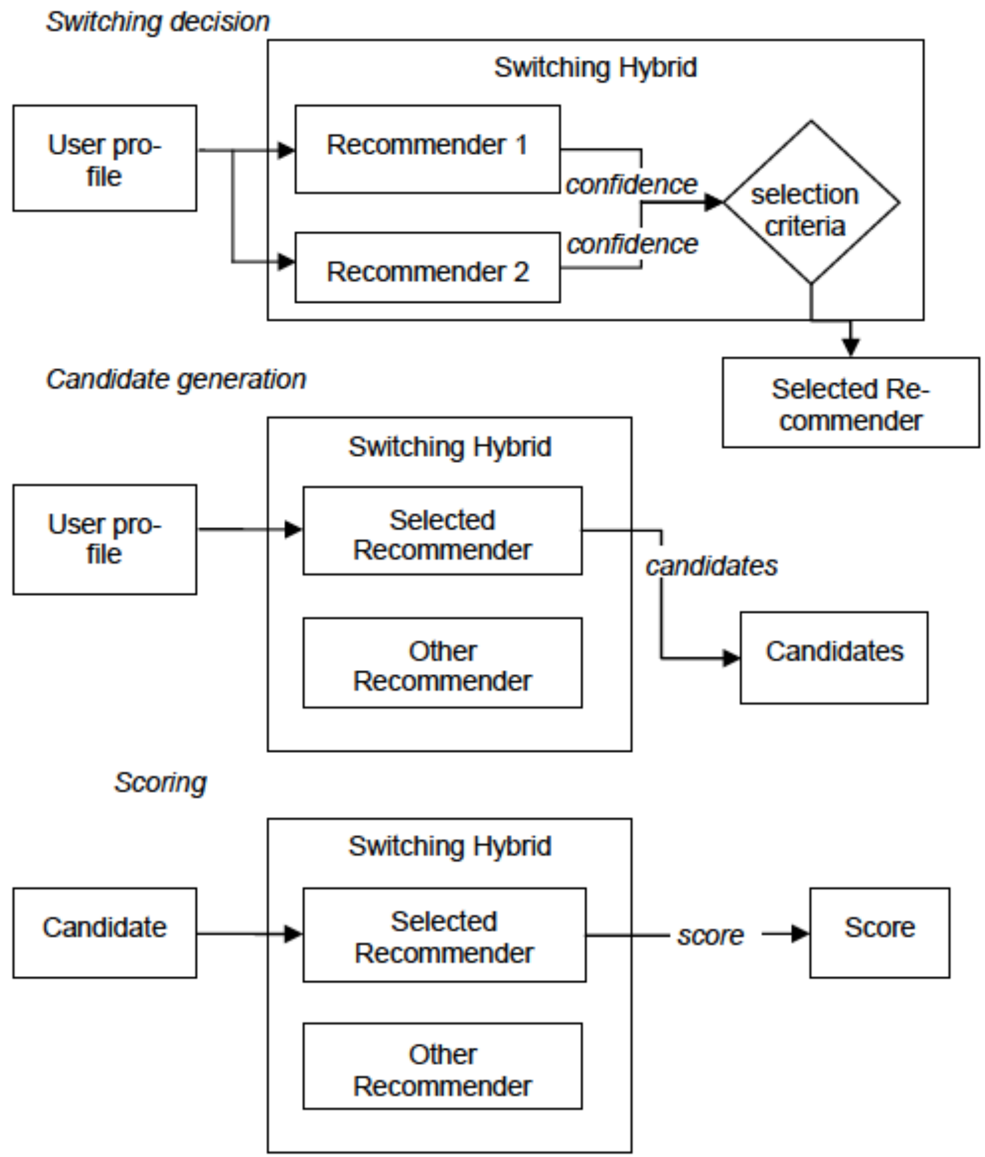
- **Selects a single recommender from among its constituents based on the recommendation situation**
  - For a different profile, a different recommender might be chosen
- Takes into account that **components may not have consistent performance for all types of users**
- **Assumes that some reliable criterion is available on which to base the switching decision**
  - E.g., **Confidence values inherent in the recommendation components.**

3. Switching Hybrid

- NewsDude [4] recommends news stories.
- It has three recommendation components:
  - a content-based nearest-neighbor recommender,
  - a collaborative recommender and
  - a second content-based algorithm using a naive Bayes classifier.
  - The recommenders are ordered.

  - The nearest neighbor technique is used first.
  - If it cannot produce a recommendation with high confidence, then the collaborative recommender is tried, and so on, with the naive Bayes recommender at the end of line.

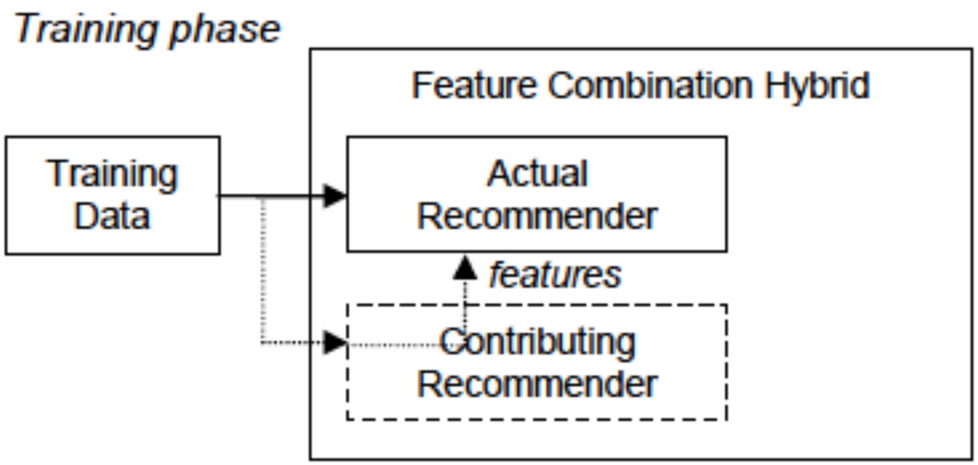
3. Switching Hybrid



4. FEATURE COMBINATION

4. Feature Combination

- **Inject features of one source** (such as collaborative recommendation) **into an algorithm designed to process data with a different source** (such a content-based recommendation)
- **A virtual "contributing recommender"**
- Features would ordinarily be processed by one recommender are instead used as part of the input to the actual recommender
- **Hybrid** is the **knowledge sources** involved
- Borrows the recommendation logic from another technique rather than employing a separate component that implements it



4. Feature Combination

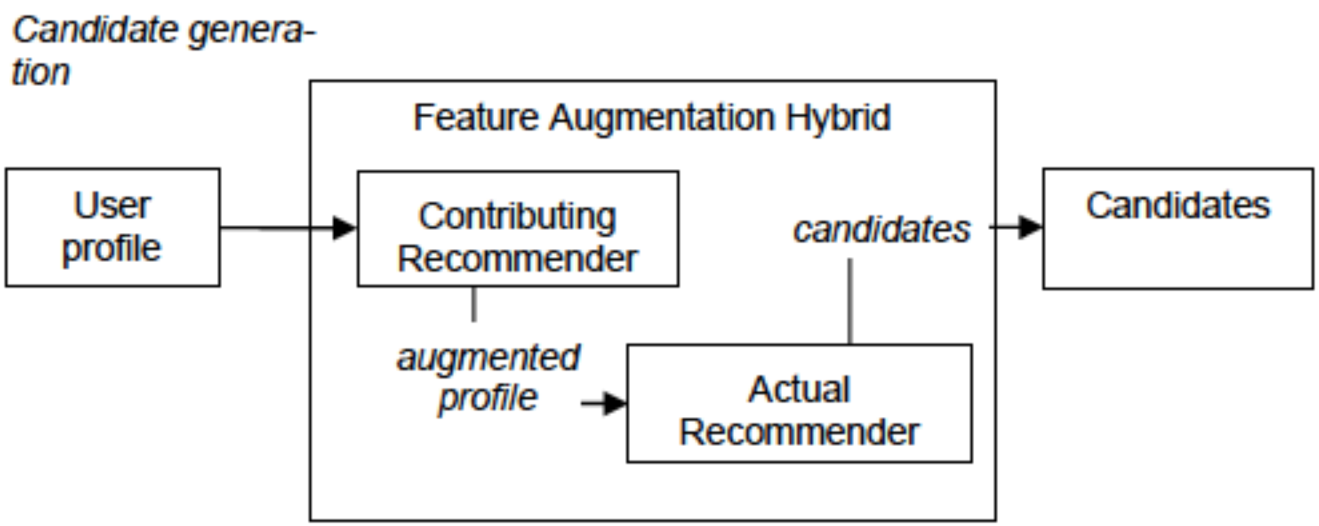
- Basu, Hirsh and Cohen [3] used the inductive rule learner Ripper [16] to learn content-based rules about user's likes and dislikes. They were able to improve the system's performance by adding collaborative features, thereby treating a fact like "User1 and User2 liked Movie X" in the same way that the algorithm treated features like "Actor1 and Actor2 starred in Movie X".

- **Content-based recommender** works in the typical way
- **Builds a learned model for each user (a user profile)**
- But **user rating data is combined with the product features**
- Content-based recommender **draws from a knowledge source associated with collaborative filtering recommandation.**

5. FEATURE AUGMENTATION

5. Feature Augmentation

- Feature augmentation hybrid **generates a new feature for each item by using the recommendation logic of the contributing domain**
  - E.g. use association rule mining over the collaborative data to derive new content features for content-based recommandation
- At each step, the **contributing recommender intercepts the data headed for the actual recommender and augments it** with its own contribution
  - not raw features as in feature combination, but the result of some computation



5. Feature Augmentation

- Melville, Mooney and Nagarajan [30] coin the term "content-boosted collaborative filtering." This algorithm learns a content-based model over the training data and then uses this model to generate ratings for unrated items. This makes for a set of profiles that is denser and more useful to the collaborative stage of recommendation that does the actual recommending.

- Employed when there is:
  - **a well-developed strong primary recommendation component**
  - **desire to add additional knowledge sources**
- Augmentation can usually be done off-line.

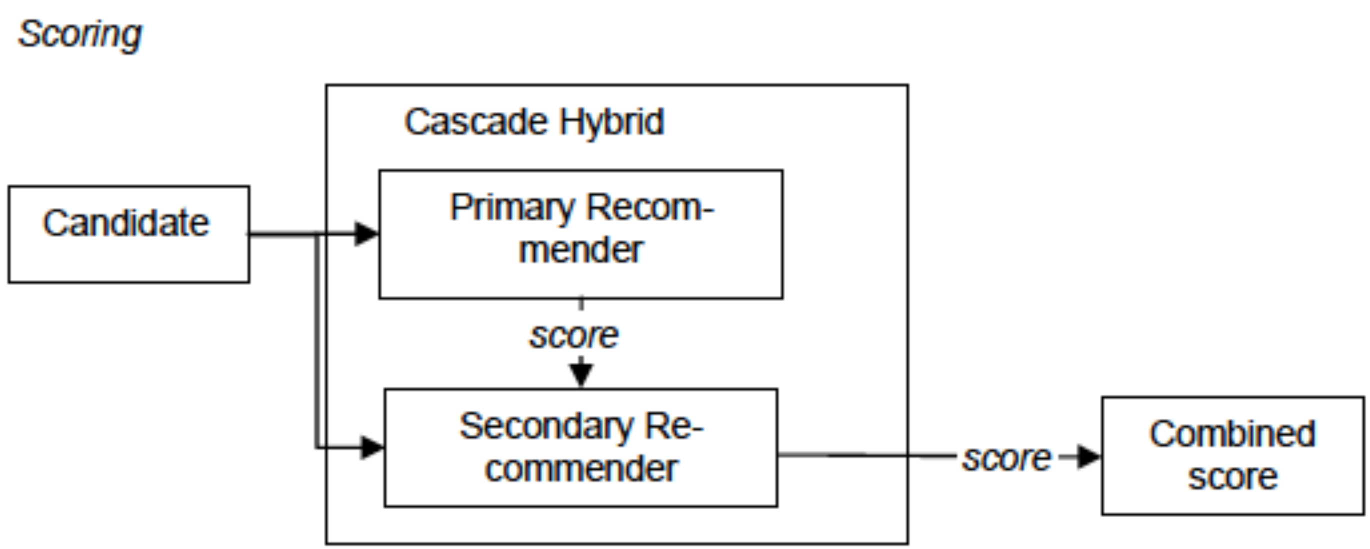
Content-Boosted CF Example

- Uses naïve Bayes as the content classifier
- Fills in missing values of rating matrix with predictions of the content predictor
  - Form a **pseudo rating matrix**: observed ratings unchanged, **missing ratings replaced by predictions of content predictor**
- Then **make predictions over the pseudo ratings matrix using weighted Pearson correlation based CF**
  - Give higher weight for item that more users rated
  - Gives higher weight for active user.

6. CASCADE HYBRID

6. Cascade Hybrid

- Create a **strictly hierarchical hybrid**
- A **weak recommender cannot overturn decisions made by a stronger one**, but can merely **refine them**
- Order-dependence
- A cascade recommender uses a secondary recommender only to break ties in the scoring of the primary one



6. Cascade Hybrid

- The knowledge-based Entree restaurant recommender [10] was found to return too many equally-scored items, which could not be ranked relative to each other. Rather than additional labor-intensive knowledge engineering (to produce finer discriminations), the hybrid EntreeC was created by adding a collaborative re-ranking of only those items with equal scores.

- This taxonomy is very strict
- **Real-world systems might have other refinements that are not exclusive (e.g., only breaking ties).**

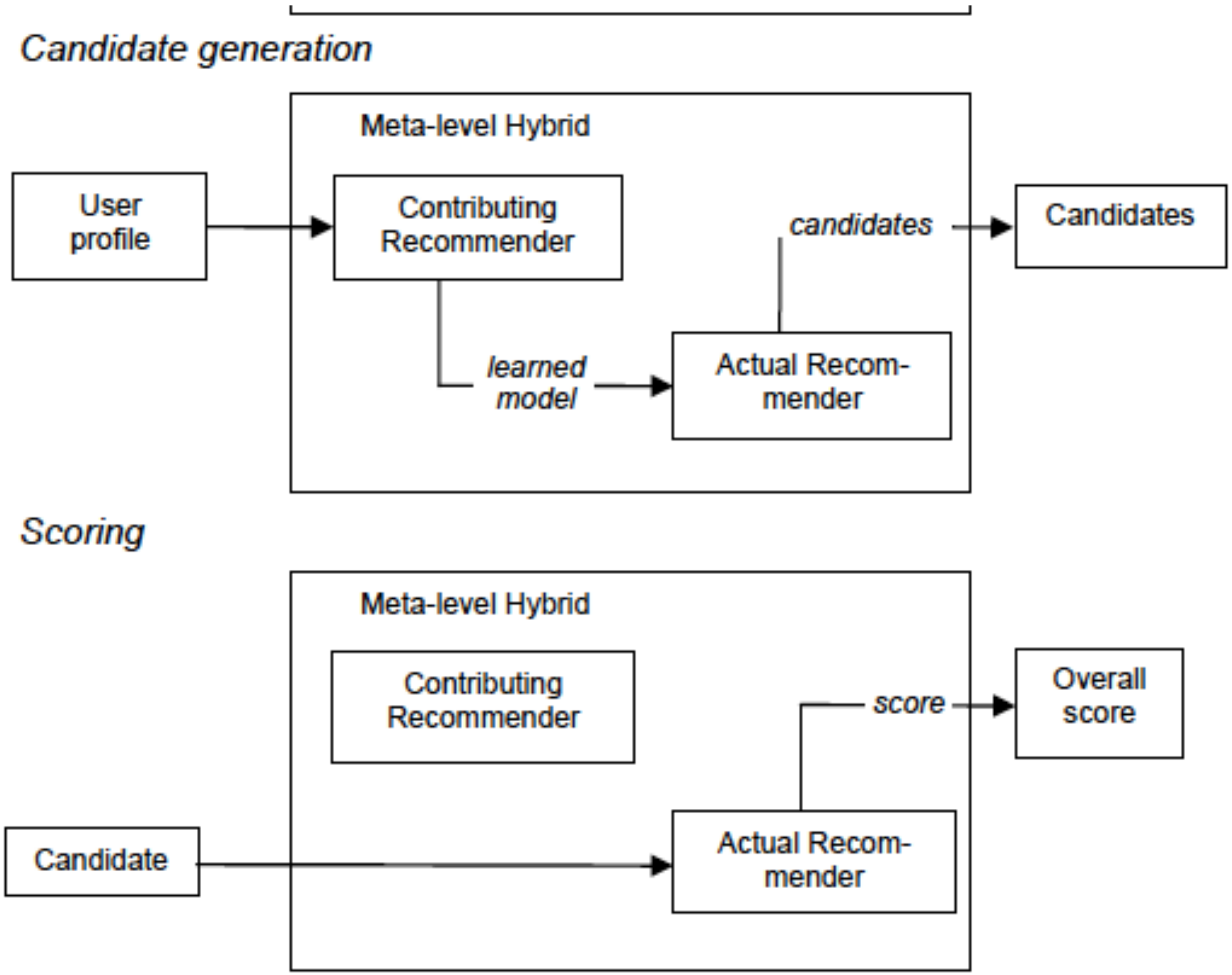
7. META-LEVEL HYBRID

7. Meta-Level Hybrid

- **Uses a model learned by one recommender as input for another**
- Similar to the feature augmentation hybrid in that the **contributing recommender** is providing **input to the actual recommender**
- Difference: in meta-level hybrid, **contributing recommender completely replaces the original knowledge source with a learned model that the actual recommender uses**
  - Actual recommender does not work with any raw profile data

- Pazzani [36] used the term "collaboration through content" to refer to his restaurant recommender that used the naive Bayes technique to build models of user preferences in a content-based way. With each user so represented, a collaborative step was then be performed in which the vectors were compared and peer users identified.

7. Meta-level Hybrid



Hybrid Recommendation Systems

Personality diagnosis

- **Combines memory-based and model-based CF**
- Active user generated by choosing one of the other users uniformly at random, adding Gaussian noise to their ratings
  - * *Gaussian noise equal to Normal distribution noise*
- **Given active user's known ratings, can calculate probability active user is same "personality type" as other users**
  - Predict probability they will like new items
- Can be regarded as a **clustering method** with one user per cluster
- Makes better predictions than:
  - Pearson correlation-based and vector similarity-based CF algorithms
  - Bayesian clustering and Bayesian networks.

Summary: Seven Types of Hybrid Recommender Systems (Taxonomy by Burke, 2002)

1. **Weighted:** The score of different recommendation components are combined numerically
2. **Switching:** The system chooses among recommendation components and applies the selected one.
3. **Mixed:** Recommendations from different recommenders are presented together.
4. **Feature Combination:** Features derived from different knowledge sources are combined together and given to a single recommendation algorithm.
5. **Feature Augmentation:** One recommendation technique is used to compute a feature or set of features, which is then part of the input to the next technique.
6. **Cascade:** Recommenders are given strict priority, with the lower priority ones breaking ties in the scoring of the higher ones.
7. **Meta-level:** One recommendation technique is applied and produces some sort of model, which is then the input used by the next technique.

More on

LATENT FACTOR MODELS

Matrix Decomposition Techniques in Machine Learning and Information Retrieval

Thomas Hofmann

Associate Professor
Department of Computer Science
Brown University

th@cs.brown.edu
www.cs.brown.edu/~th

BROWN

recommind™
software that understands

Latent Factor Models

AIM3 – Scalable Data Analysis and Data Mining

11 – Latent factor models for Collaborative Filtering

Sebastian Schelter, Christoph Boden, Volker Markl

Fachgebiet Datenbanksysteme und Informationsmanagement
Technische Universität Berlin

http://www.dima.tu-berlin.de/

Latent Structure

- Given a matrix that "encodes" data ...
- Potential problems
  - too large
  - too complicated
  - missing entries
  - noisy entries
  - lack of structure
  - ...

$$\mathbf{A} = \begin{pmatrix} a_{11} & \dots & a_{1j} & \dots & a_{1m} \\ \dots & \dots & \dots & \dots & \dots \\ a_{i1} & \dots & a_{ij} & \dots & a_{im} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & \dots & a_{nj} & \dots & a_{nm} \end{pmatrix}$$

- Is there a **simpler** way to **explain** entries?
- There might be a **latent structure** underlying the data.
- How can we "find" or "reveal" this structure?

Latent Factor Models

Idea

- ratings are deeply influenced by a set of **factors** that are very **specific to the domain** (e.g. amount of action in movies, complexity of characters)
- these factors are in general **not obvious**, we might be able to think of some of them but it's hard to estimate their impact on the ratings
- the goal is to infer those so called **latent factors** from the rating data by using mathematical techniques

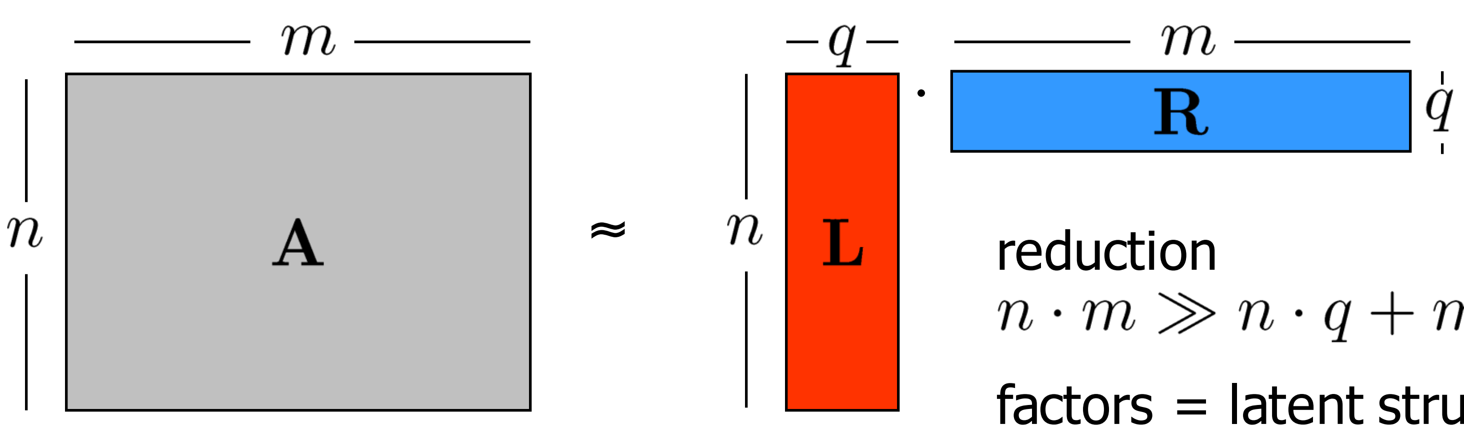
Matrix Decomposition

- Common approach: approximately **factorize** matrix

$$\mathbf{A} \approx \hat{\mathbf{A}} = \mathbf{L} \cdot \mathbf{R}$$

approximation   left factor   right factor

- Factors are typically constrained to be "**thin**"



reduction

$$n \cdot m \gg n \cdot q + m \cdot q$$

factors = latent structure (?)

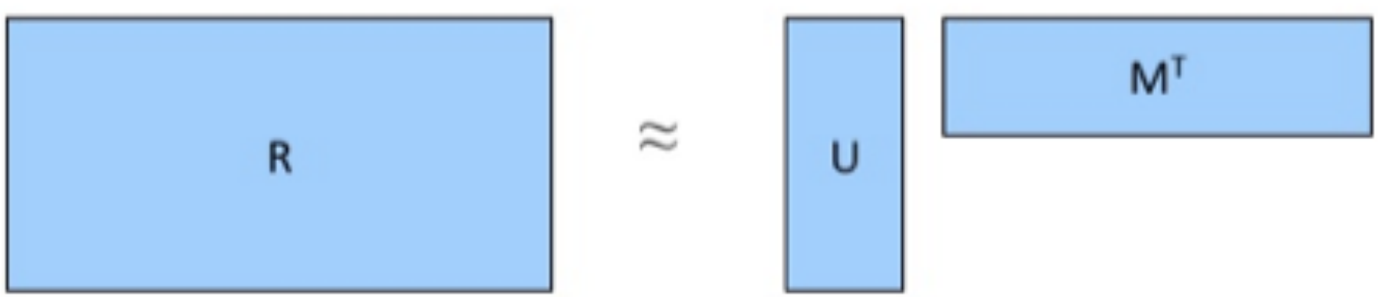
Latent Factor Models

- **Approach**
  - users and items are characterized by **latent factors**, each user and item is mapped onto a **latent feature space** $u_i, m_j \in R^{n_f}$
  - each rating is approximated by the dot product of the **user feature vector** and the **item feature vector** $r_{ij} \approx m_j^T u_i$
  - **prediction of unknown ratings** also uses this dot product
  - **squared error** as a measure of loss $(r_{ij} - m_j^T u_i)^2$

Latent Factor Models

- **Approach**
  - **decomposition of the rating matrix** into the product of a user feature and an item feature matrix
  - row in U: vector of a user's affinity to the features
  - row in M: vector of an item's relation to the features

  - closely related to **Singular Value Decomposition** which produces an optimal low-rank optimization of a matrix



pLSA – Latent Variable Model

- Structural modeling assumption (mixture model)

$$\hat{p}_{\mathrm{LSA}}(w|d) = \sum_{z} p(w|z;\theta)p(z|d;\pi)$$

Document language model

Latent concepts or topics

Concept expression probabilities

Document-specific mixture proportions

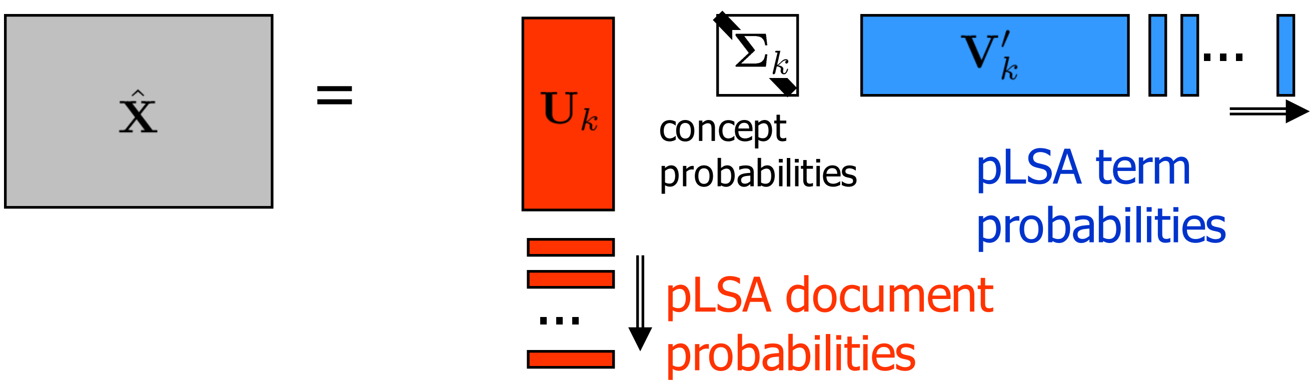
Model fitting

- **Document language model*=A statistical **language model** is a probability distribution over sequences of words.
- [Hofmann, Proceedings ACM SIGIR, 1999]

pLSA: Matrix Decomposition

- Mixture model can be written as a **matrix factorization**
- Equivalent symmetric (joint) model

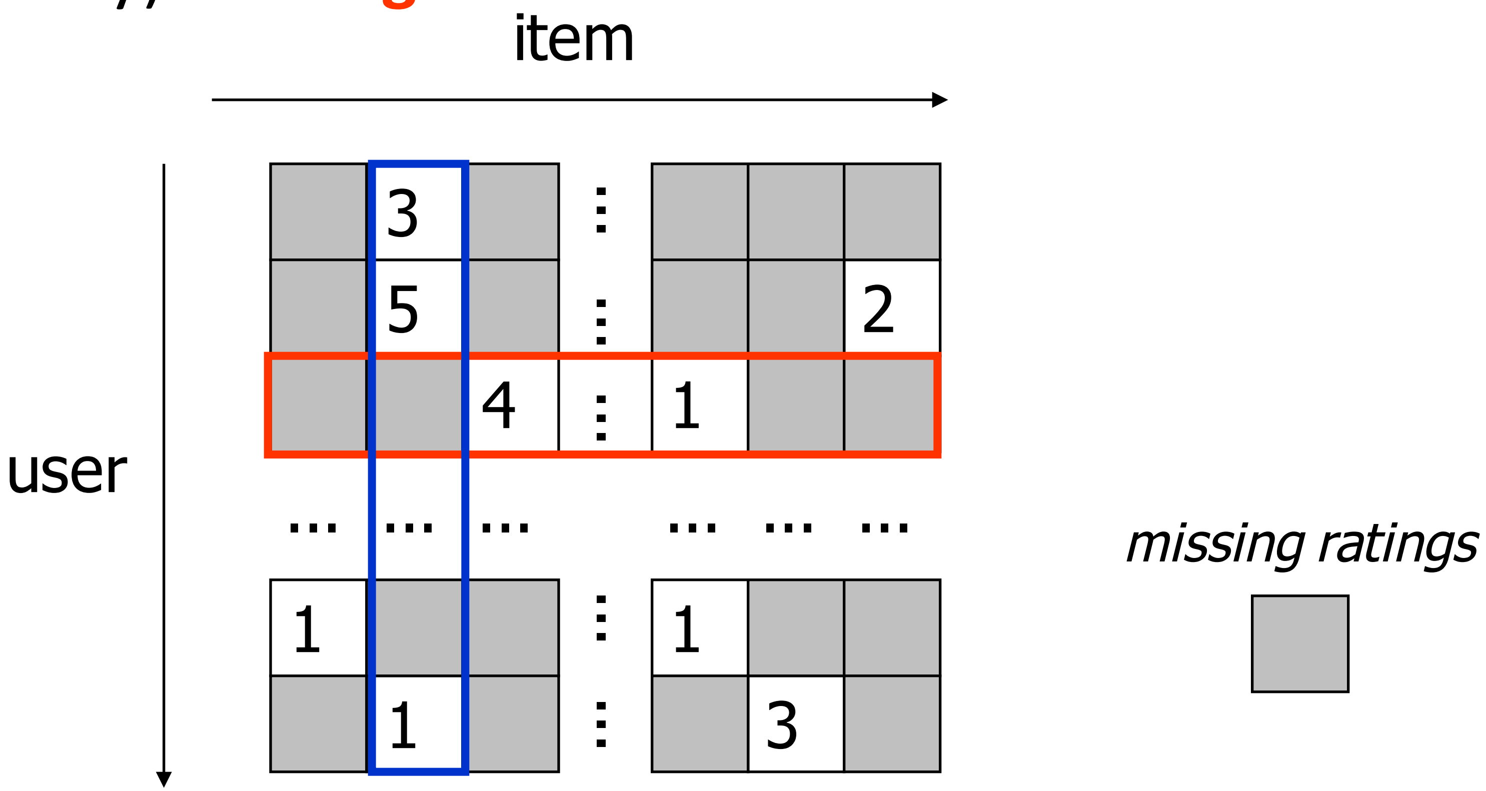
$$\hat{p}_{\mathrm{LSA}}(d,w) = \sum_z p(d|z)\,p(z)\,p(w|z)$$



- Contrast to LSA/SVD: **non-negativity** and **normalization** (intimate relation to non-negative matrix factorization).

Rating Matrix

- Rating matrix is typically a large matrix with many (mostly) **missing values**



pLSA-like Decomposition

- Generalization of pLSA (additional rating variable)

$$p_{\mathrm{LSA}}(r, y|u) = \sum_z \underbrace{p(r|y, z; \rho)}_{\text{extension to predict ratings}} \underbrace{p(y|z; \theta) p(z|u; \pi)}_{\text{standard pLSA model to explain sparseness pattern}}$$

Explicit decomposition of user preferences (each user can have **multiple interests**)

- Probabilistic model can be used to **optimize** specific **objectives**
- Data **compression** and **privacy** preservation

- Details
  - multinomial or Gaussian sampling model for rating variable
  - EM algorithm for (approximate) model fitting

T. Hofmann, *Latent Semantic Models for Collaborative Filtering*, ACM Transactions on Information Systems, 2004, Vol 22(1), pp. 89-115.