

ACA ML 2018

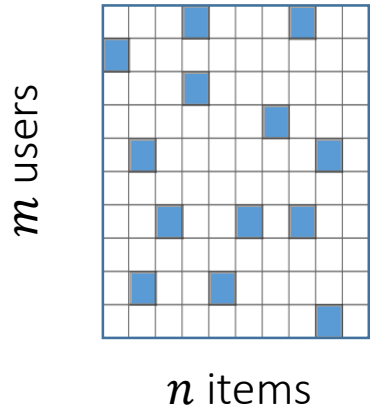
Workshop on Recommender Systems

Part 3

Recap of the 2st part

A general view on matrix factorization

utility matrix A



Incomplete data:

- known entries
- unknown entries

Task: find utility (or relevance) function f_U such that:

$$f_R: \text{Users} \times \text{Items} \rightarrow \text{Relevance score}$$

As optimization problem with some *loss function* \mathcal{L} :

$$\mathcal{L}(A, R) \rightarrow \min$$

Any factorization model consists of:

- Utility function to generate R
- Optimization objective defined by \mathcal{L}
- Optimization method (algorithm)

$$R = PQ^T$$

Simplistic view: latent features \leftrightarrow genres



ALS vs SGD vs SVD

ALS

More stable

Fewer hyper-parameters to tune

Higher complexity, however requires fewer iterations

Embarrassingly parallel

Higher communication cost in distributed environment

SGD

Sensitive to hyper-parameters

Requires special treatment of learning rate

Lower complexity, but slower convergence

Inherently sequential (parallelization is tricky for RecSys)

For binary feedback complexity changes: $nnz \rightarrow MN$

Unlike SVD:

More involved model selection (no rank truncation).

No global convergence guarantees!

Asynchronous SGD is non-deterministic.

Allow for custom optimization objectives. $\mathcal{L}(A, R) \rightarrow \mathcal{L}(f(A, R))$

For explicit feedback:

Algorithm	Overall complexity	Update complexity	Sensitivity	Optimality
SVD*	$O(nnz_A \cdot r + (M + N)r^2)$	$O(nnz_a \cdot r)$	Stable	Global
ALS	$O(nnz_A \cdot r^2 + (M + N)r^3)$	$O(nnz_a \cdot r + r^3)$	Stable	Local
CD	$O(nnz_A \cdot r)$	$O(nnz_a \cdot r)$	Stable	Local
SGD	$O(nnz_A \cdot r)$	$O(nnz_a \cdot r)$	Sensitive	Local

* For both standard and randomized implementations [71].

Task

- You have a binary utility matrix (with “true” zeros) resulted from some implicit feedback information.
 - What will be the complexity of SGD?
 - What will be the SGD-based solution if you omit zero values?
 - Is it reasonable to use bias terms?

What are your solutions to 2nd
home assignment?

Part 3

Today's lecture:

- Advanced matrix factorization techniques
 - Mixed implicit and explicit feedback models (NSVD, SVD++)
 - Bilinear models: SVDFeature and Factorization Machines. Hybrid models.
- What can be optimized
- Tensor factorization

Explicit to implicit

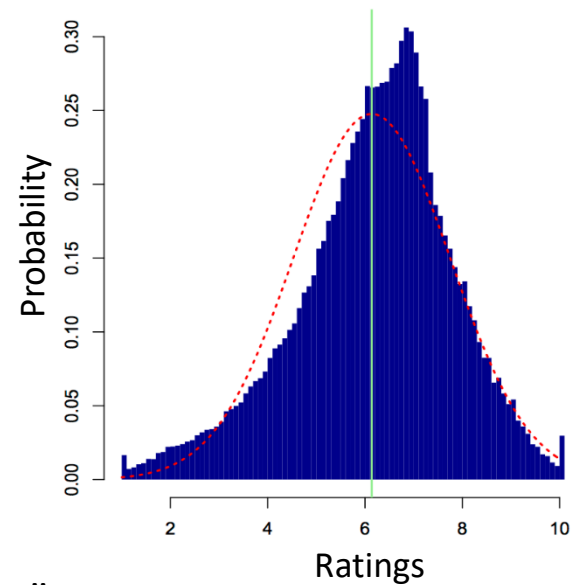
?	5	?
5	?	3
2	3	4

threshold

0	1	0
1	0	0
0	0	1

“binarize”

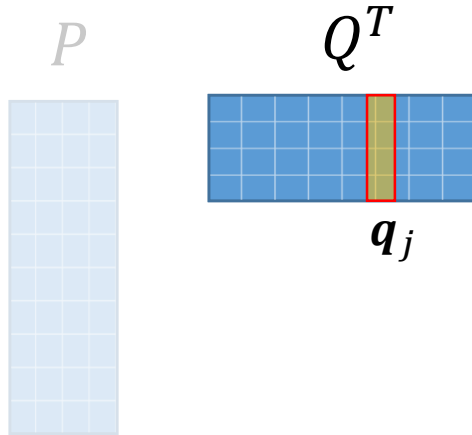
0	1	0
1	0	1
1	1	1



NSVD

Key idea: user is represented as a combination of items

A. Paterek, “Improving regularized singular value decomposition for collaborative filtering”, 2007.



$$r_{ij} = \left(\sum_{k \in S^{(i)}} q_k^T \right) q_j$$

omitting bias terms

$S^{(i)} = \{j: w_{ij} \neq 0\}$
indices of items, rated by user i ;
forms a “neighborhood” of items

- helpful in the case of extreme sparsity
- reduced storage requirements
- prone to overfitting

What is the corresponding matrix form?

Hint: you are given the matrix of (implicit) interactions S .

$$R = SQQ^T$$

Does it look familiar to you?

$$\text{SVD: } R = A_0 V V^T$$

SVD++

Key idea: user is described by implicit and explicit interactions

Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model", 2008.

$$\mathbf{p}_i \leftarrow \mathbf{p}_i + \frac{1}{|S^{(i)}|} \sum_{k \in S^{(i)}} \bar{\mathbf{q}}_k$$

Explicit part Implicit part

$$r_{ij} = \left(\mathbf{p}_i + \frac{1}{|S^{(i)}|} \sum_{k \in S^{(i)}} \bar{\mathbf{q}}_k \right)^T \mathbf{q}_j$$

independent latent spaces!

omitting bias terms

$$R = (P + S_n \bar{Q}) Q^T$$
$$S_n = D^{-1} S$$

binary

$$D = \text{diag}\{\|\mathbf{s}_1\|^2, \dots, \|\mathbf{s}_M\|^2\}$$

For several types of implicit feedback $P + S_n^{(1)} \bar{Q}^{(1)} + S_n^{(2)} \bar{Q}^{(2)} + \dots$ Increases the number of parameters!

More general representation

$$R = (P + S_n \bar{Q})Q^T = \overbrace{[I \ S_n] \begin{bmatrix} P \\ \bar{Q} \end{bmatrix}}^X \bar{P}$$

$$R = (X\bar{P})Q^T$$

X - is a “design” matrix

Matrix X encodes implicit information.

It can be extended to incorporate any side information.

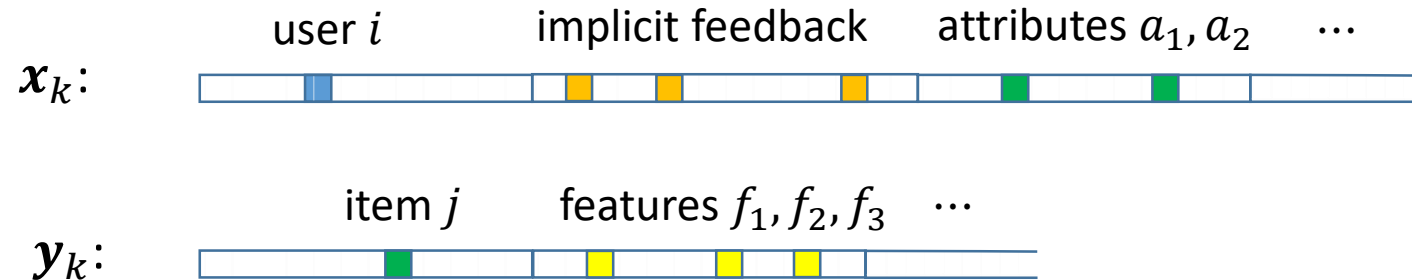


SVDFeature

T. Chen, et al. "Feature-based matrix factorization", 2011

$$R = (XP)(YQ)^T$$

$$X = [X_1 \ X_2 \ \dots \ X_m] \quad Y = [Y_1 \ Y_2 \ \dots \ Y_n]$$



Including bias terms:

$$r_{ij} = b_0 + \mathbf{t}^T \mathbf{x}_i + \mathbf{f}^T \mathbf{y}_j + \mathbf{x}_i^T P Q \mathbf{y}_j$$

Optimized with ALS, SGD.

$$b_0 = \sum_{g \in G} \gamma_g \mu_g \text{ is precomputed!}$$

$$\text{Model parameters: } \Theta = \{\mathbf{t}, \mathbf{f}, P, Q\}$$

Factorization Machines

Idea: polynomial expansion

S. Rendle, “*Factorization machines*”, 2010.

$$f(\mathbf{x}) = b_0 + \mathbf{b}^T \mathbf{x} + \mathbf{x}^T \mathbf{H} \mathbf{x} + \dots$$

Feature vector \mathbf{x}																	Target y					
$\mathbf{x}^{(1)}$	1	0	0	...	1	0	0	0	...	0.3	0.3	0.3	0	...	13	0	0	0	0	...	5	$y^{(1)}$
$\mathbf{x}^{(2)}$	1	0	0	...	0	1	0	0	...	0.3	0.3	0.3	0	...	14	1	0	0	0	...	3	$y^{(2)}$
$\mathbf{x}^{(3)}$	1	0	0	...	0	0	1	0	...	0.3	0.3	0.3	0	...	16	0	1	0	0	...	1	$y^{(2)}$
$\mathbf{x}^{(4)}$	0	1	0	...	0	0	1	0	...	0	0	0.5	0.5	...	5	0	0	0	0	...	4	$y^{(3)}$
$\mathbf{x}^{(5)}$	0	1	0	...	0	0	0	1	...	0	0	0.5	0.5	...	8	0	0	1	0	...	5	$y^{(4)}$
$\mathbf{x}^{(6)}$	0	0	1	...	1	0	0	0	...	0.5	0	0.5	0	...	9	0	0	0	0	...	1	$y^{(5)}$
$\mathbf{x}^{(7)}$	0	0	1	...	0	0	1	0	...	0.5	0	0.5	0	...	12	1	0	0	0	...	5	$y^{(6)}$
	A	B	C	...	TI	NH	SW	ST	...	TI	NH	SW	ST	...	Time	TI	NH	SW	ST	...		
	User				Movie					Other Movies rated						Last Movie rated						

Factorization Machines

$$r_{ij} = b_0 + \mathbf{t}^T \mathbf{x}_i + \mathbf{f}^T \mathbf{y}_j + \mathbf{x}_i^T \mathbf{PQ} \mathbf{y}_j$$

$$\mathbf{b} = \begin{bmatrix} \mathbf{t} \\ \mathbf{f} \end{bmatrix} \quad \mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \quad \text{for some fixed } i \text{ and } j$$

$$r(\mathbf{z}) = b_0 + \mathbf{b}^T \mathbf{z} + \mathbf{z}^T \mathbf{H} \mathbf{z} + \dots$$

characterizes intra- and inter-relations
between any encoded entities

Data is sparse \rightarrow impose low-rank structure on H

H is symmetric positive semi-definite

$$H = VV^T$$

V embeds all users, items and their side information

2nd order FM:

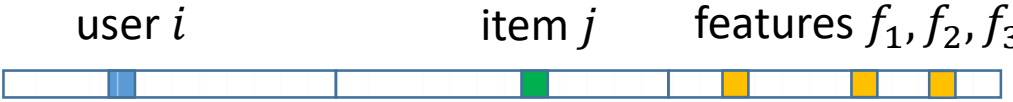
$$r(\mathbf{z}) = b_0 + \sum_k b_k z_k + \sum_k \sum_{\underline{k'=k+1}} \langle \mathbf{v}_k, \mathbf{v}_{k'} \rangle z_k z_{k'}$$

$\langle \cdot, \cdot \rangle$ is a scalar product

Model parameters: $\Theta = \{b_0, \mathbf{z}, V\}$



Example of Factorization Machines computation

$$\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{f} \end{bmatrix} \quad V = \begin{bmatrix} V_x \\ V_y \\ V_f \end{bmatrix} \quad V \in \mathbb{R}^{n \times r} \quad \mathbf{z} \in \mathbb{R}^n:$$


The diagram shows a horizontal bar representing the vector \mathbf{z} . It is divided into three main sections: 'user i' (blue), 'item j' (green), and 'features f_1, f_2, f_3 ' (yellow). Each section contains a small colored square corresponding to its label.

$$\mathbf{z}^T V V^T \mathbf{z} = \left([\mathbf{x}^T \quad \mathbf{y}^T \quad \mathbf{f}^T] \begin{bmatrix} V_x \\ V_y \\ V_f \end{bmatrix} \right) \left(\begin{bmatrix} V_x^T & V_y^T & V_f^T \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{f} \end{bmatrix} \right) = (\mathbf{x}^T V_x + \mathbf{y}^T V_y + \mathbf{f}^T V_f)(\mathbf{x}^T V_x + \mathbf{y}^T V_y + \mathbf{f}^T V_f)^T =$$

“self-interaction” terms actual 2nd order FM model $r(\mathbf{z})$ (w/o biases)

$$= \underbrace{\mathbf{x}^T V_x V_x^T \mathbf{x} + \mathbf{y}^T V_y V_y^T \mathbf{y} + \mathbf{f}^T V_f V_f^T \mathbf{f}}_{\text{“self-interaction” terms}} + 2 \underbrace{\left(\underbrace{\mathbf{x}^T V_x V_y^T \mathbf{y}}_{\text{user-item}} + \underbrace{\mathbf{x}^T V_x V_f^T \mathbf{f}}_{\text{user-feature}} + \underbrace{\mathbf{y}^T V_y V_f^T \mathbf{f}}_{\text{item-feature}} \right)}_{\text{interactions}}$$


$$\mathbf{v}_x = V_x^T \mathbf{x}, \quad \mathbf{v}_y = \dots$$

$$r(\mathbf{z}) = \frac{1}{2} \left[(\mathbf{v}_x + \mathbf{v}_y + \mathbf{v}_f)^T (\mathbf{v}_x + \mathbf{v}_y + \mathbf{v}_f) - (\|\mathbf{v}_x\|^2 + \|\mathbf{v}_y\|^2 + \|\mathbf{v}_f\|^2) \right] =$$

$$= \frac{1}{2} \sum_{l=1}^r \left(\left(\sum_{k=1}^n \mathbf{v}_{kl} z_k \right)^2 - \sum_{k=1}^n (\mathbf{v}_{kl} z_k)^2 \right)$$

reduces the number of operations

Connection of FM to other models

$$r(\mathbf{z}) = b_0 + \sum_k b_k z_k + \sum_k \sum_{k'=k+1} \delta_{kk'} \langle \mathbf{v}_k, \mathbf{v}_{k'} \rangle z_k z_{k'}$$



binary indicator variable, denotes “allowed” interactions

Example:

- restrict interactions to (user, item) pairs only
- forbid intra-relations between entities of the same type
- remove side information

What is the resulting model?

Matrix form of FM

$$\mathbf{z} = \begin{bmatrix} x \\ y \\ f \end{bmatrix} \quad V = \begin{bmatrix} V_x \\ V_y \\ V_f \end{bmatrix} \quad V \in \mathbb{R}^{n \times r} \quad \mathbf{z} \in \mathbb{R}^n:$$


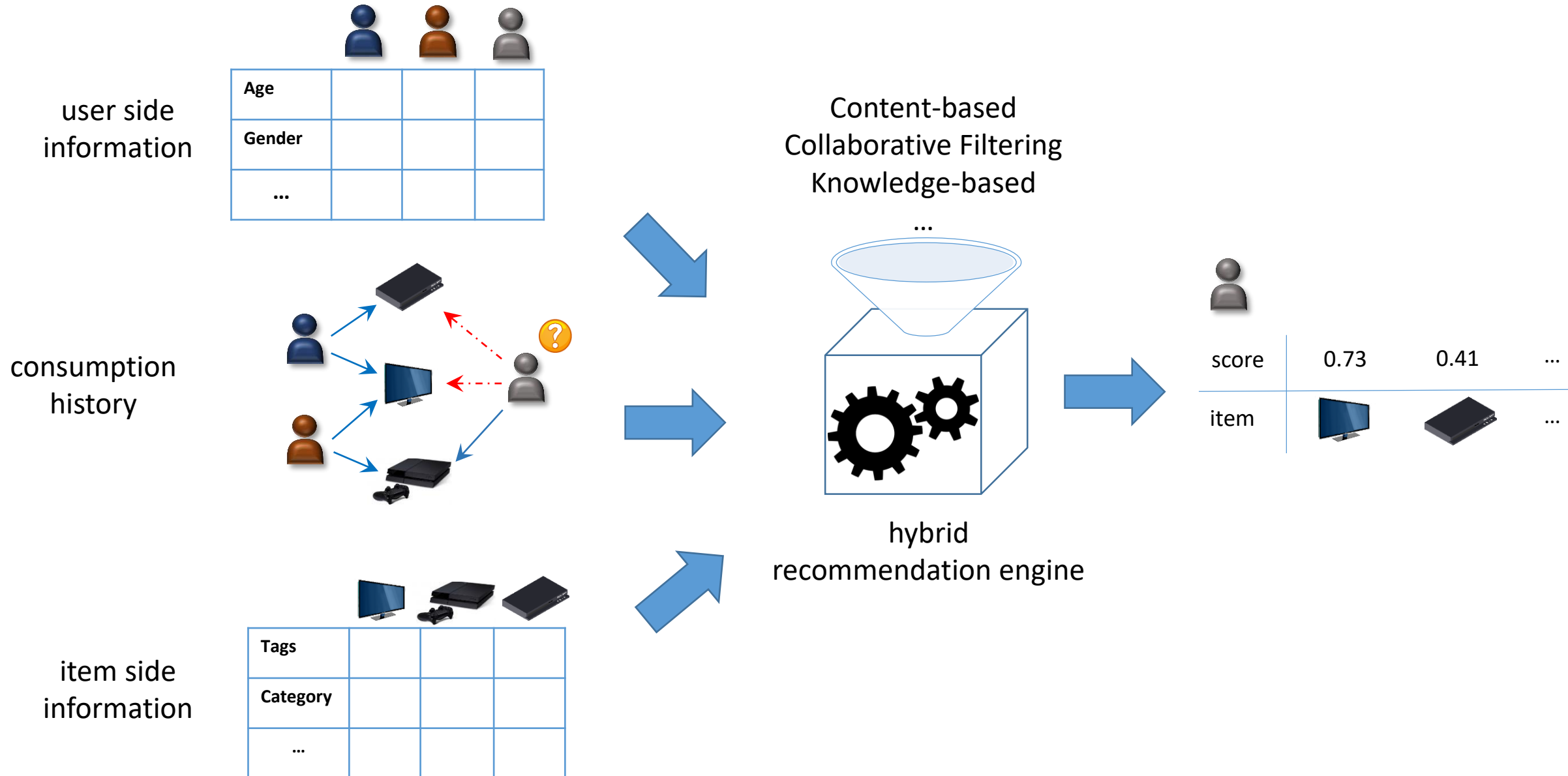
$$r(\mathbf{z}) = \frac{1}{2} \mathbf{z}^T \left(\begin{bmatrix} V_x \\ V_y \\ V_f \end{bmatrix} \begin{bmatrix} V_x^T & V_y^T & V_f^T \end{bmatrix} - \begin{bmatrix} V_x & 0 \\ 0 & V_y & V_f \end{bmatrix} \begin{bmatrix} V_x & 0 \\ 0 & V_y & V_f \end{bmatrix}^T \right) \mathbf{z} =$$

$$= \frac{1}{2} \mathbf{z}^T \begin{bmatrix} 0 & V_x V_y^T & V_x V_f^T \\ V_y V_x^T & 0 & V_y V_f^T \\ V_f V_x^T & V_f V_y^T & 0 \end{bmatrix} \mathbf{z} = \mathbf{z}^T H \mathbf{z}$$

$V_x \rightarrow P, V_y \rightarrow Q$ – conventional matrix factorization

$$H = \frac{1}{2} \begin{bmatrix} 0 & V_x V_y^T & V_x V_f^T \\ V_y V_x^T & 0 & V_y V_f^T \\ V_f V_x^T & V_f V_y^T & 0 \end{bmatrix}$$

Hybrid approach



Taxonomy

- **Ensemble design**

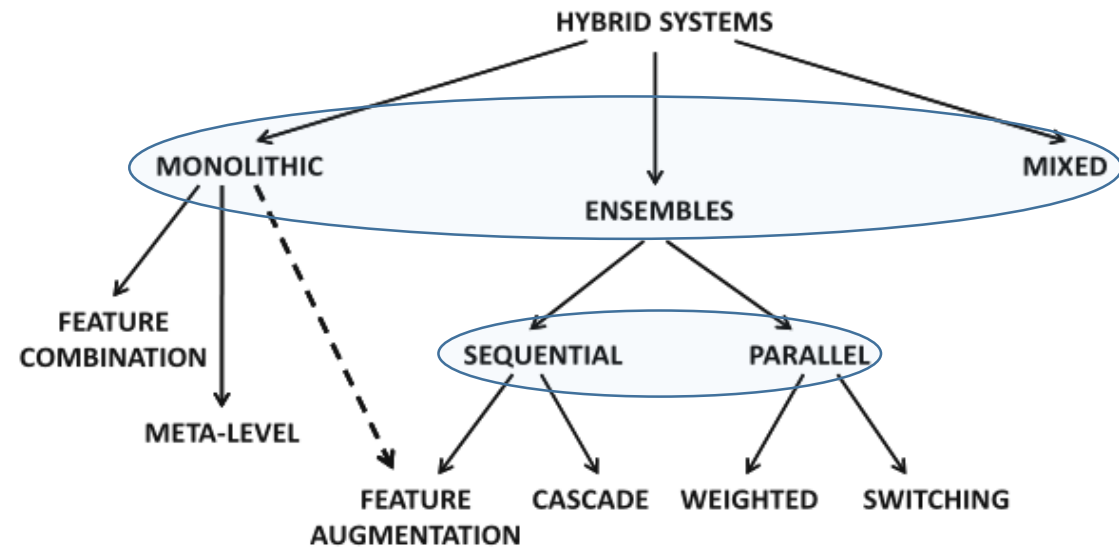
combines several “black-box” algorithms to produce a single more robust output

- **Monolithic design**

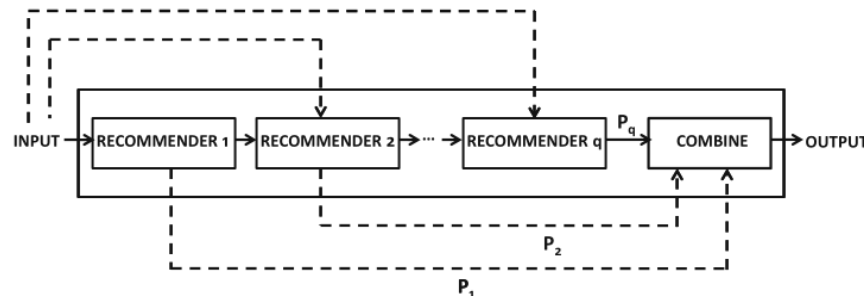
obtained by integration of heterogeneous data sources and customization of models

- **Mixed systems**

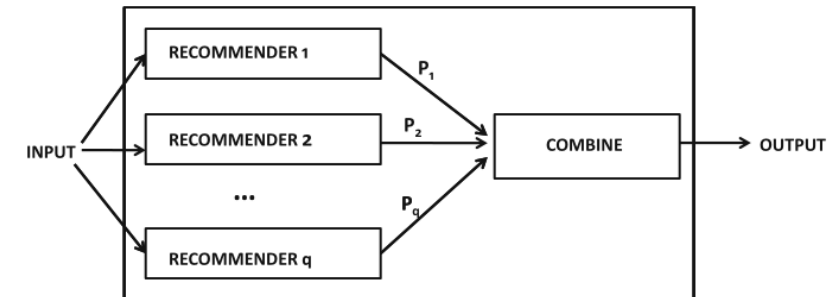
present outputs of several recommender models simultaneously



Sequential



Parallel

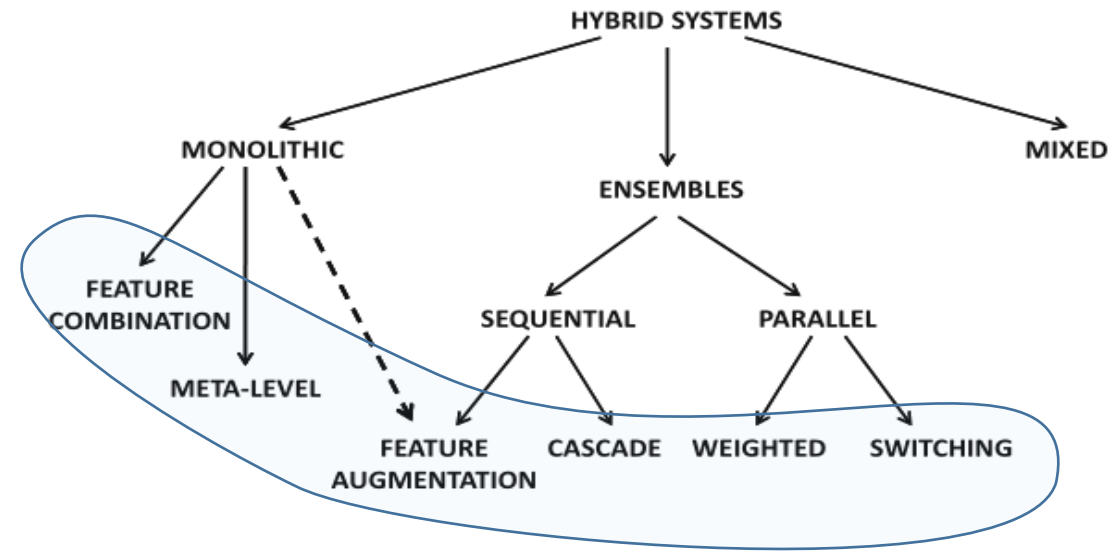


Taxonomy

Ensembles

- **Weighted**
combines outputs of several models with varying degree of importance
- **Switching**
situational choice of the most suitable recommendation model
- **Cascade**
every consequent model tries to improve upon the results of the previous models
- **Feature Augmentation**
one model generates new features or scores and feeds them into another model (stacking)

can be used “off-the-shelf”



Monolithic

- **Feature Combination**
several data sources are fed into a single model
- **Meta-level**
several models are glued together

no “off-the-shelf” solution

Task

Give an example of a feature combination approach

Meta-level system example

Rating matrix

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}	I_{12}
U_1	5	3	5	4	-	1	-	3	-	5	-	-
U_2	3	-	-	-	4	5	1	-	5	-	-	1
U_3	1	-	5	4	5	-	5	-	-	3	5	-

(a)

Test user

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}	I_{12}
U_4	5	-	1	-	-	4	-	-	3	-	-	5

(b)

Item features

	f_1	f_2	f_3	f_4
I_1	1	1	0	0
I_2	1	0	0	0
I_3	1	0	1	1
I_4	1	0	0	1
I_5	0	1	1	0
I_6	0	1	0	0
I_7	0	0	1	1
I_8	0	0	0	1
I_9	0	1	1	0
I_{10}	0	0	0	1
I_{11}	0	0	1	1
I_{12}	0	1	0	0

(c)

User profiles

	f_1	f_2	f_3	f_4
U_1	4	1	1	4
U_2	1	4	2	0
U_3	2	1	4	5

	f_1	f_2	f_3	f_4
U_4	1	4	1	0

Model: SVD over profiles
Prediction: folding-in + neighborhood

Some weighted hybrids

- Randomness Injection

Many matrix factorization methods are inherently randomized due to initialization

Single data , several models →
single model, several data samples

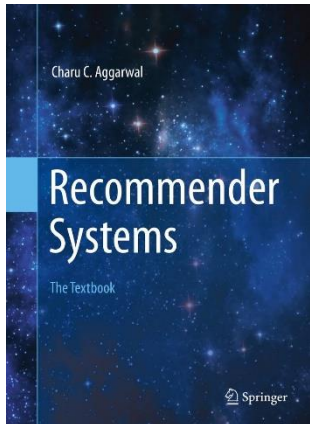
- Bagging

- Row-wise bootstrapping
- Entry-wise bagging

- Subsampling

- Row-wise subsampling
- Entry-wise subsampling

Not storage-efficient



Read more:

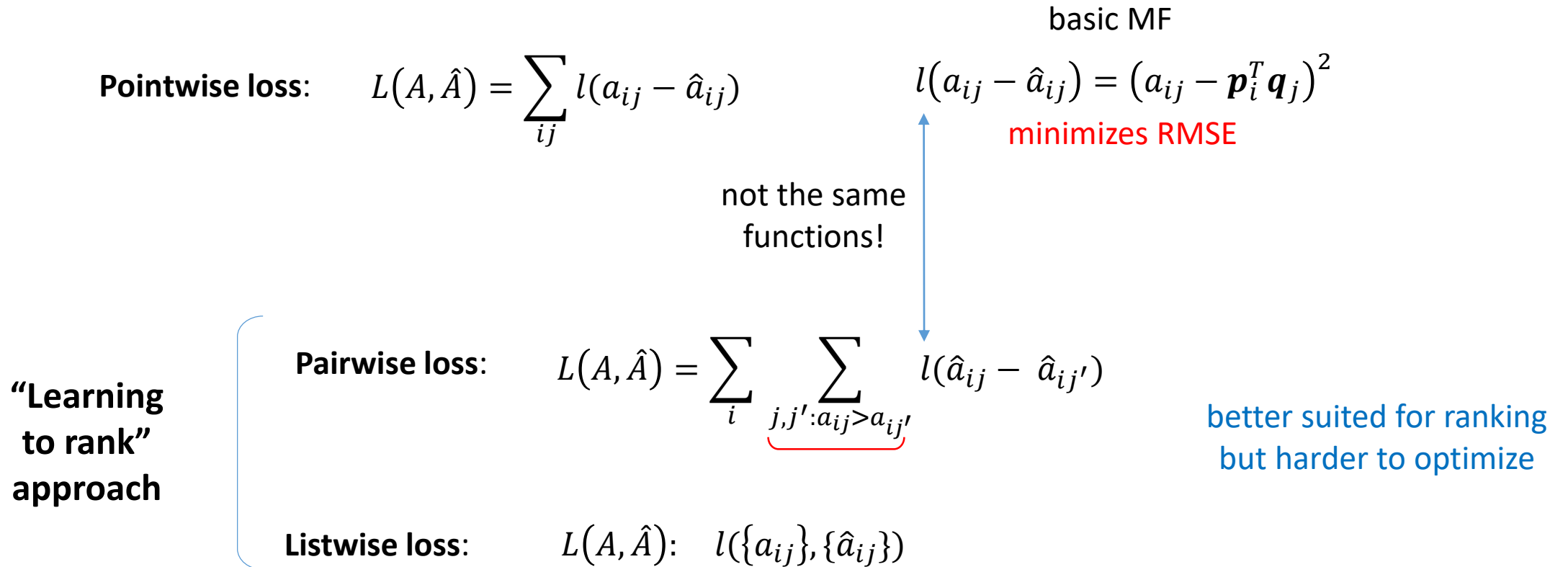
Charu C. Aggarwal. "Recommender Systems. The Textbook", 2016.

R. Burke. Hybrid recommender systems: Survey and experiments. User Modeling and User-adapted Interaction, 12(4), pp. 331–370, 2002.

Other factorization methods

- NMF
- MMF
- PMF
- CAMF
- SLIM
- FISM
- CliMF
- OrdRec
- CobaFi
- ...

Remark on optimization objectives



Bayesian personalized ranking (BPR)

- Key idea – 2 step:** 1. get predicted ratings (from any model)
2. try to improve ranking with a posteriori modification (max likelihood)

$$p(\Theta | >_u) \propto p(>_u | \Theta) p(\Theta)$$

$$p(i >_u j | \Theta) := \sigma(\hat{x}_{uij}(\Theta))$$

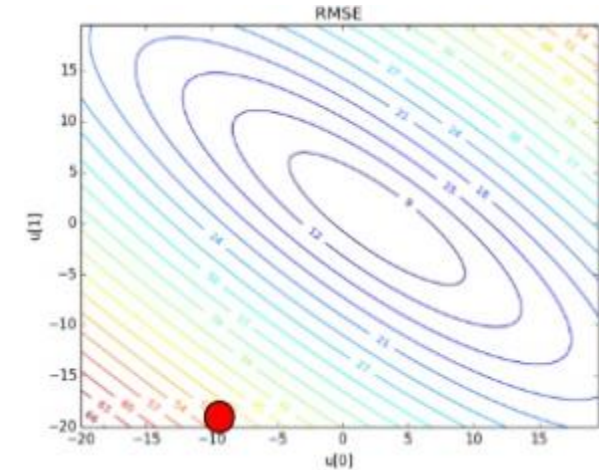
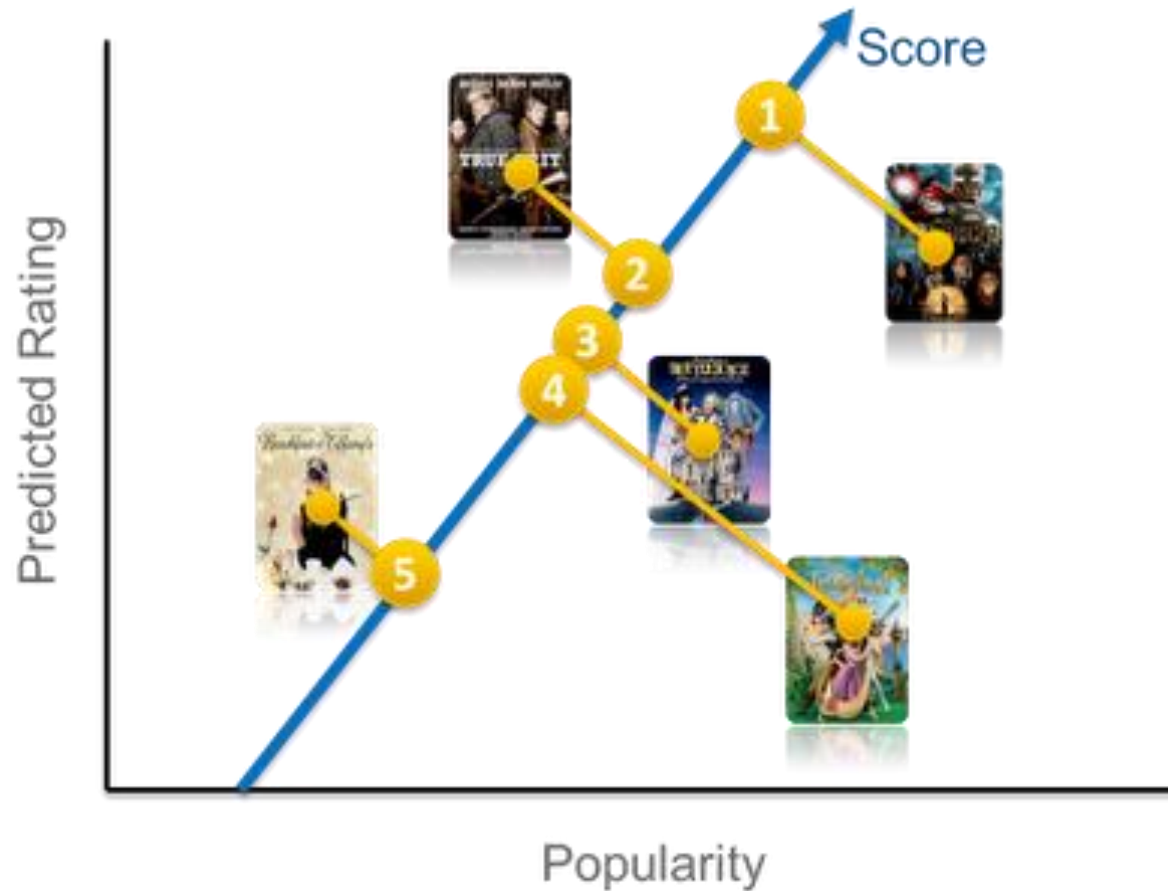
$$\hat{x}_{uij} := \hat{x}_{ui} - \hat{x}_{uj} \quad x_{ui}, x_{uj} \text{ are from some Collaborative Filtering model}$$

pair-wise objective:

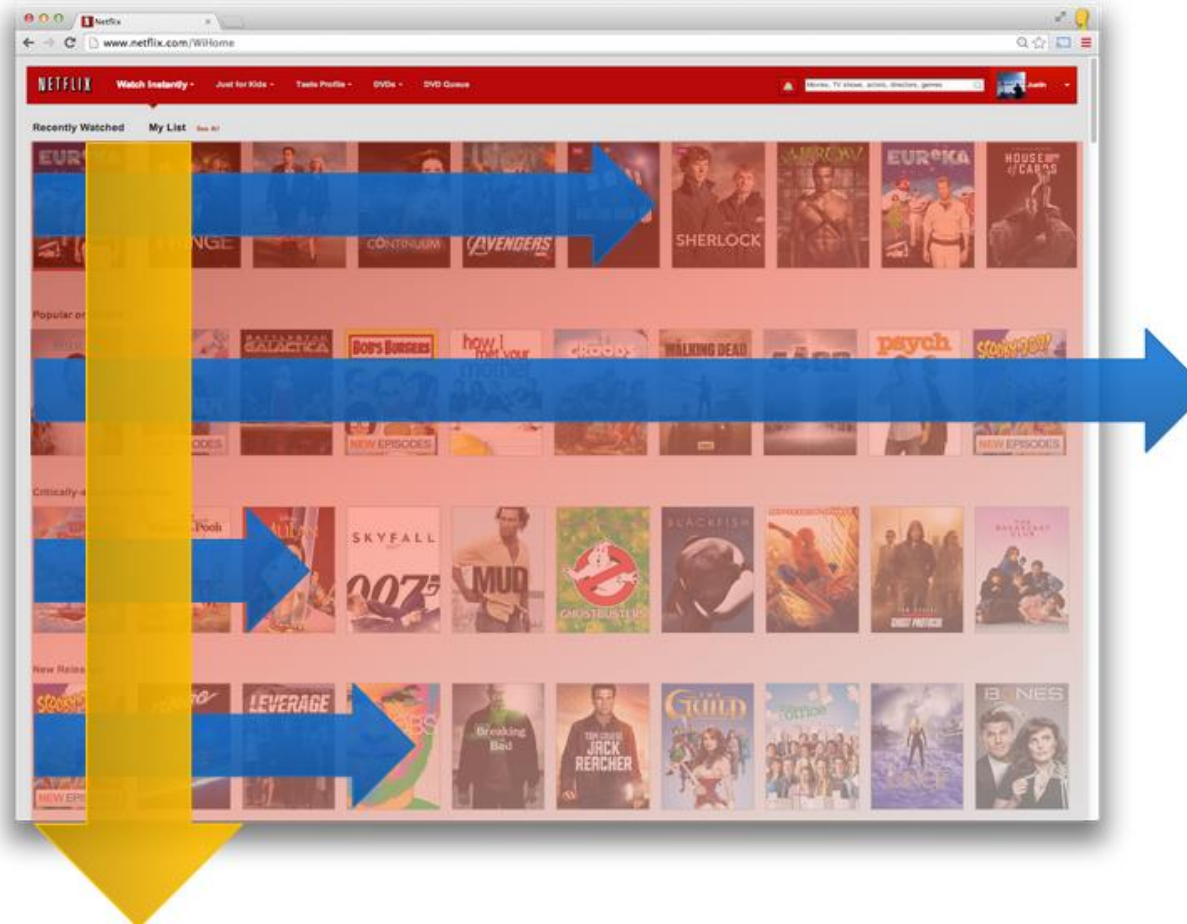
$$\sum_{(u,i,j) \in D_S} \ln \sigma(\hat{x}_{uij}) - \lambda_{\Theta} \|\Theta\|^2$$

Optimization with stochastic gradient descent.

Learning to rank – simple approach



User Experience



Reading: How important is the user interface for a recommender system?

<https://www.quora.com/How-important-is-the-user-interface-for-a-recommender-system/answer/Abhinav-Sharma?srid=cgo&share=e3430560>

Image source: <http://technocalifornia.blogspot.ru/2013/07/recommendations-as-personalized.html#!/2014/12/ten-lessons-learned-from-building-real.html>

Lessons learned from building real-life recommender systems

<http://www.slideshare.net/xamat/recsys-2016-tutorial-lessons-learned-from-building-reallife-recommender-systems>

Tensor factorization

Guess standard algorithm behavior

What is likely to be recommended in this case?



Explanation

predicted scores (*folding-in*):

$$\mathbf{r} \approx \mathbf{V}\mathbf{V}^T\mathbf{p}$$

top- n recommendations task:

$$\text{toprec}(\mathbf{p}, n) := \arg \max^n \mathbf{r}$$

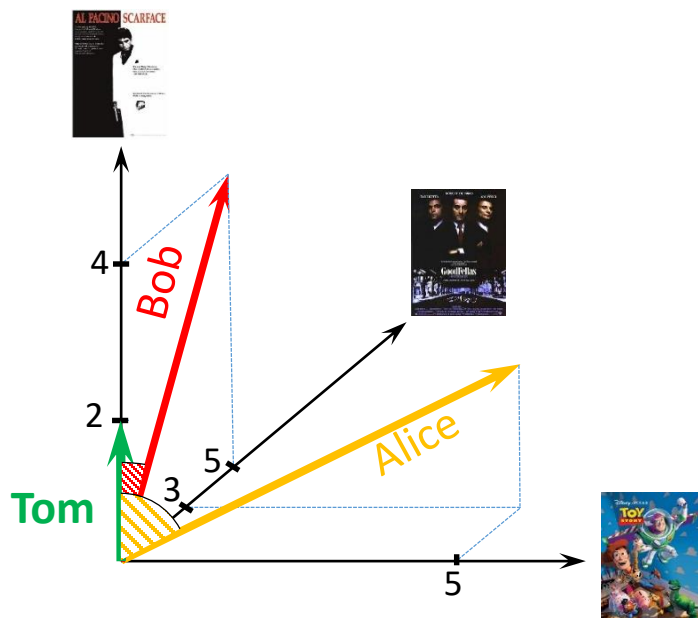
$$\arg \max \mathbf{V}\mathbf{V}^T (0, \dots, 0, \textcolor{red}{2}, 0, \dots, 0)^T \equiv \arg \max \mathbf{V}\mathbf{V}^T (0, \dots, 0, \textcolor{green}{5}, 0, \dots, 0)^T$$




☆☆☆☆☆

☆☆☆☆☆

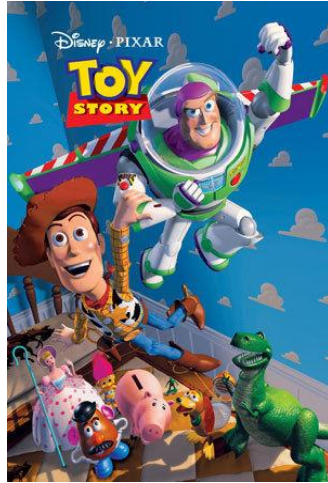
Rating value doesn't change ranking of the items!

Same for naïve similarity-based model:



			
Alice	2	5	3
Bob	4		5
Carol	2	5	
Tom	2	???	???
Prediction			
		2.6	3.1

Explicit feedback



2.5x better?



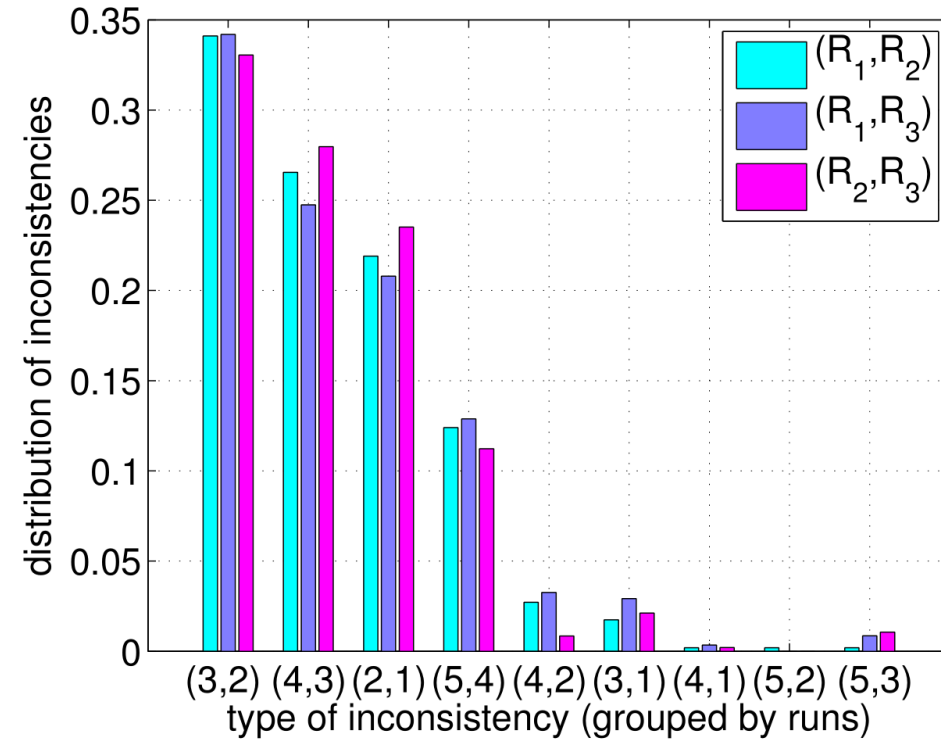
From neoclassical economics:
utility is an **ordinal concept**.

Explicit feedback

Traditional recommender models treat ratings as **cardinal numbers**.



Ratings scale consist of **unequal intervals*** :

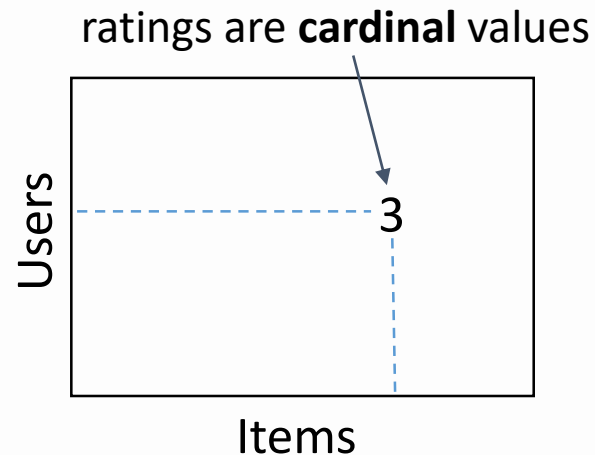


*X. Amatriain, J. M. Pujol, and N. Oliver "I like it... I like it not: Evaluating User Ratings Noise in Recommender Systems", UMAP '09 Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization

Restating the problem

Standard model

$$User \times Item \rightarrow Rating$$

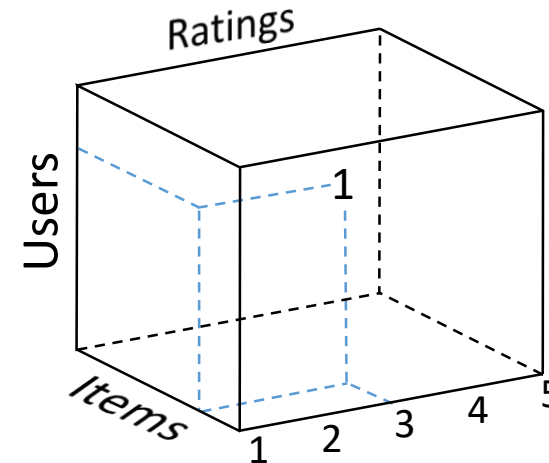


Technique: **Matrix factorization**

Collaborative Full Feedback model

CoFFee (proposed approach)

$$User \times Item \times Rating \rightarrow Relevance\ Score$$



Technique: **Tensor Factorization**

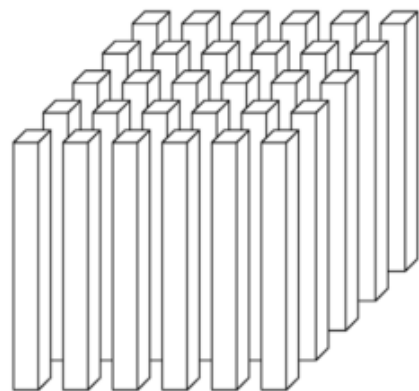
based on Tucker Decomposition*

$$\mathcal{A} \approx \mathcal{G} \times_1 U \times_2 V \times_3 W$$

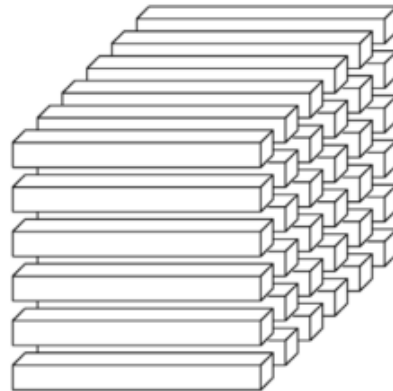
* T. G. Kolda and B. W. Bader, "Tensor Decompositions and Applications", 2009

General tensor concepts

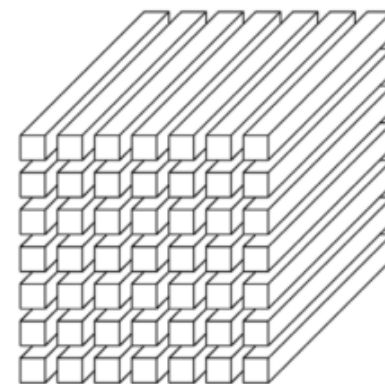
Fibers:



(a) Mode-1 (column) fibers: $\mathbf{x}_{:jk}$

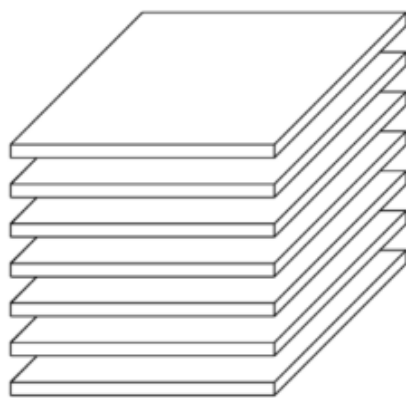


(b) Mode-2 (row) fibers: $\mathbf{x}_{i:k}$

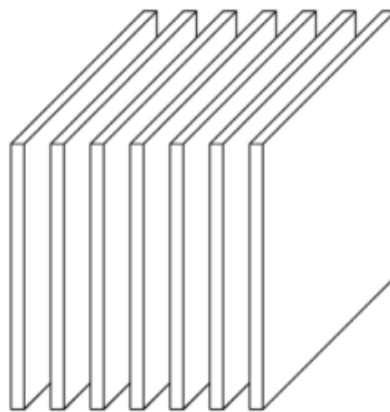


(c) Mode-3 (tube) fibers: $\mathbf{x}_{ij:}$

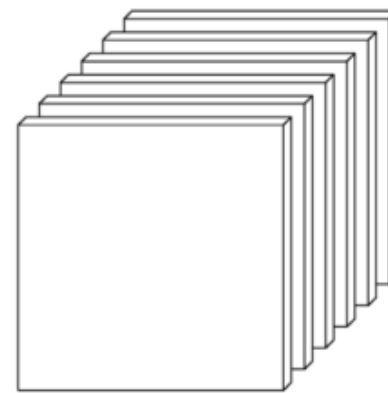
Slices:



(a) Horizontal slices: $\mathbf{X}_{i::}$

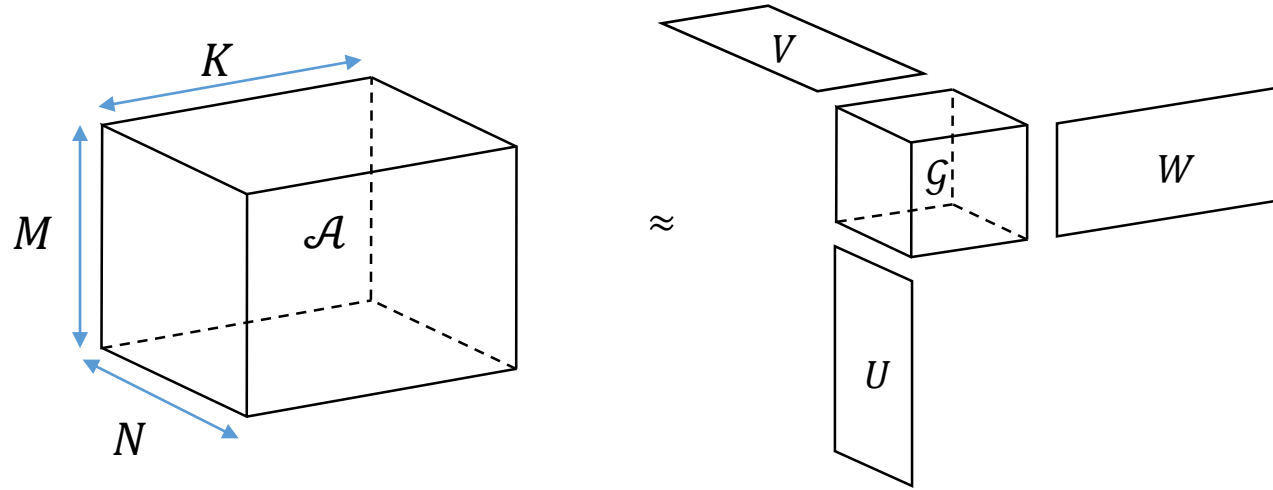


(b) Lateral slices: $\mathbf{X}_{:,j:}$

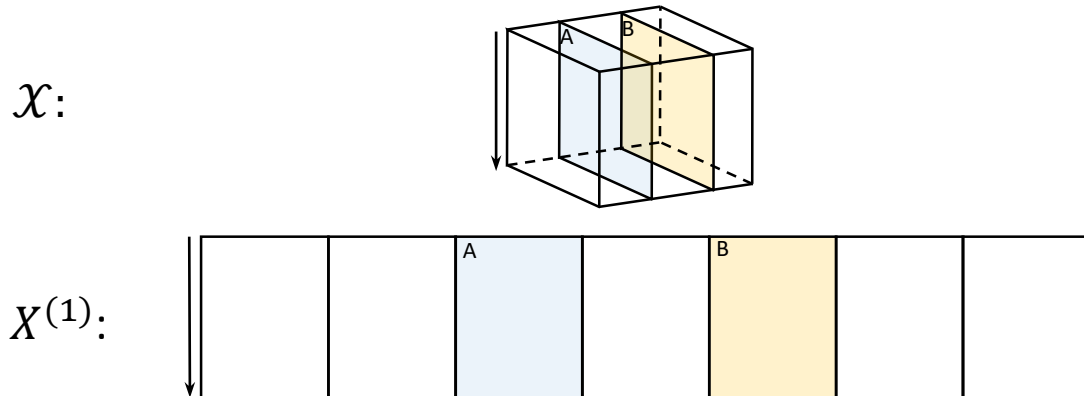


(c) Frontal slices: $\mathbf{X}_{:,k}$ (or \mathbf{X}_k)

Tucker decomposition



Tensor matricization (unfolding):



$$\mathcal{A} \approx \mathcal{G} \times_1 U \times_2 V \times_3 W$$

$$\mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times r_3}$$

$$U \in \mathbb{R}^{M \times r_1}, V \in \mathbb{R}^{N \times r_2}, W \in \mathbb{R}^{K \times r_3}$$

$$a_{ijk} \approx \sum_{\alpha, \beta, \gamma}^{r_1, r_2, r_3} g_{\alpha\beta\gamma} u_{i\alpha} v_{j\beta} w_{k\gamma}$$

General ALS procedure (HOOI)

Initialize V, W randomly; require orthonormal columns

Repeat until convergence:

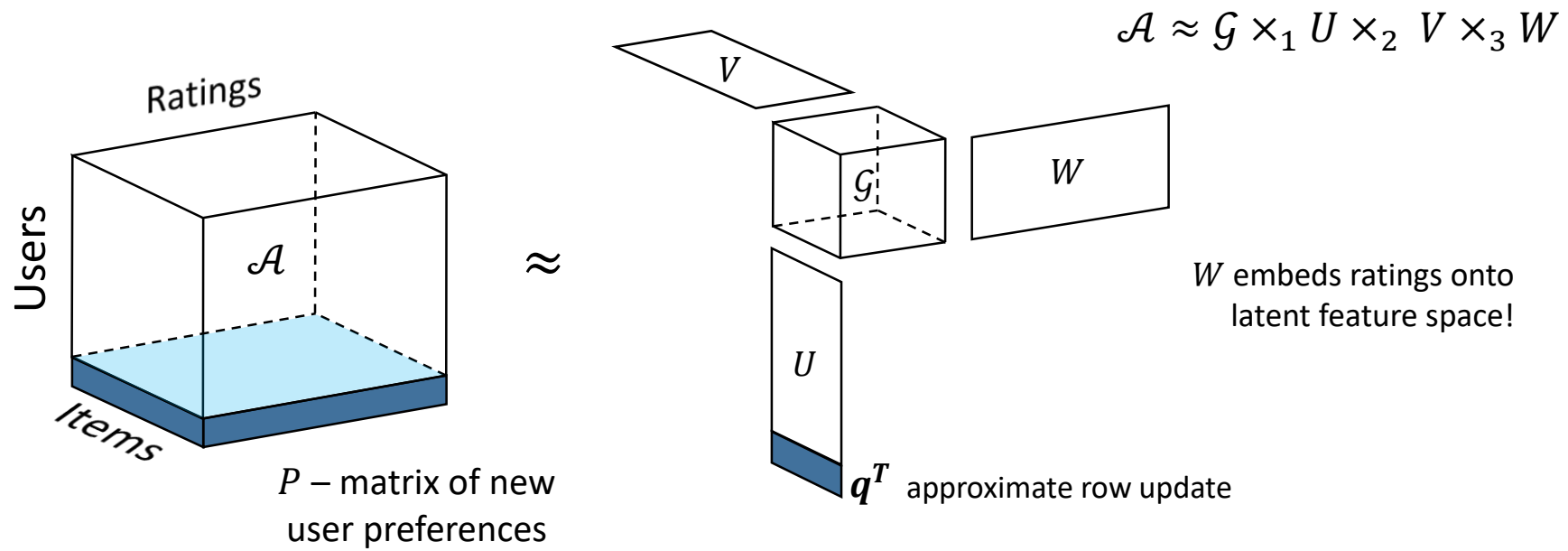
$$U \leftarrow r_1 \text{ left principal vectors of } (\mathcal{A} \times_2 V \times_3 W)^{(1)}$$

$$V \leftarrow r_2 \text{ left principal vectors of } (\mathcal{A} \times_1 U \times_3 W)^{(2)}$$

$$W \leftarrow r_3 \text{ left principal vectors of } (\mathcal{A} \times_1 U \times_2 V)^{(3)}$$

$$\mathcal{G} \leftarrow \mathcal{A} \times_1 U \times_2 V \times_3 W$$

Recommendations in real-time



“Shades of ratings”

Higher order *folding-in*:

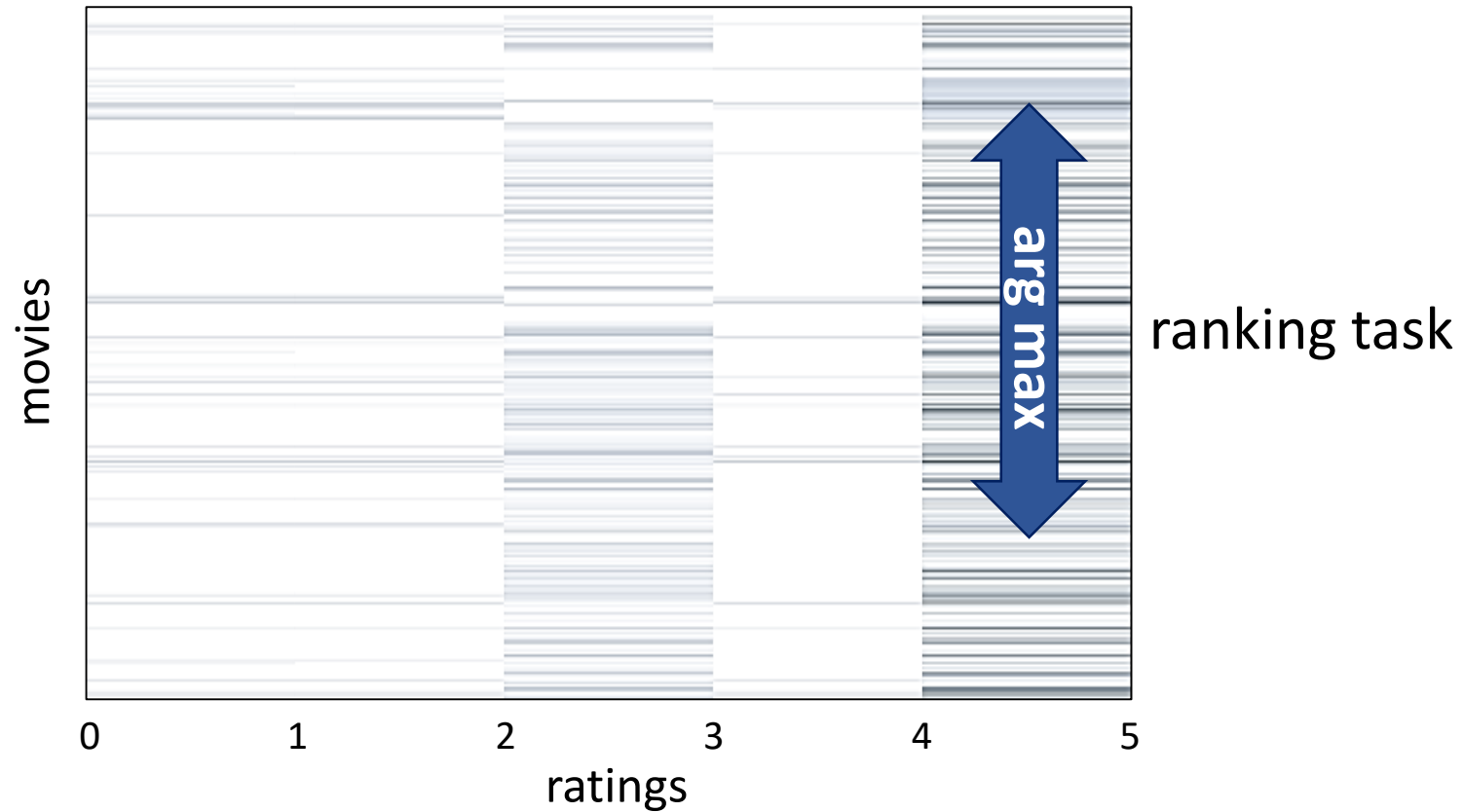
$$R = VV^T PWW^T \quad \text{items relevance matrix}$$

Compare to SVD: $r = VV^T p$

“Shades” of ratings

$$R = VV^T PWW^T$$

More dense colors correspond to higher relevance score.

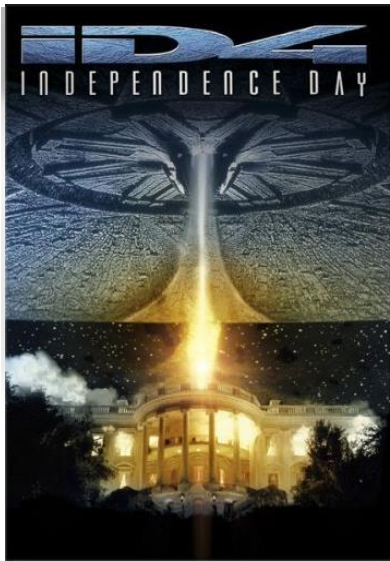
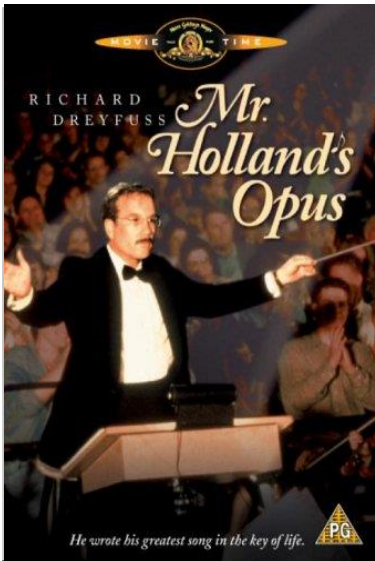
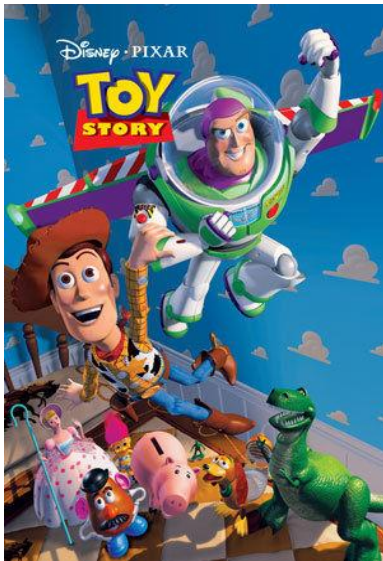
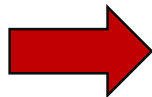


Granular view of user preferences,
concerning **all possible ratings**.



Model is **equally sensitive**
to any kind of feedback.

Warm-start with CoFFee



	Scarface ★☆☆☆☆	LOTR: The Two Towers ★☆☆☆☆	Star Wars: Episode VII - The Force Awakens ★★★★★
CoFFee	Toy Story Mr. Holland's Opus Independence Day	Net, The Cliffhanger Batman Forever	Dark Knight, The Batman Begins Star Wars: Episode IV - A New Hope
SVD	Reservoir Dogs Goodfellas Godfather: Part II, The	LOTR: The Fellowship of the Ring Shrek LOTR: The Return of the King	Dark Knight, The Inception Iron Man

nDCG calculation

$$DCG = \sum_i \frac{2^{rel_i} - 1}{\log_2(i + 1)} \quad rel_i - \text{true rating of a recommended item at position } i$$

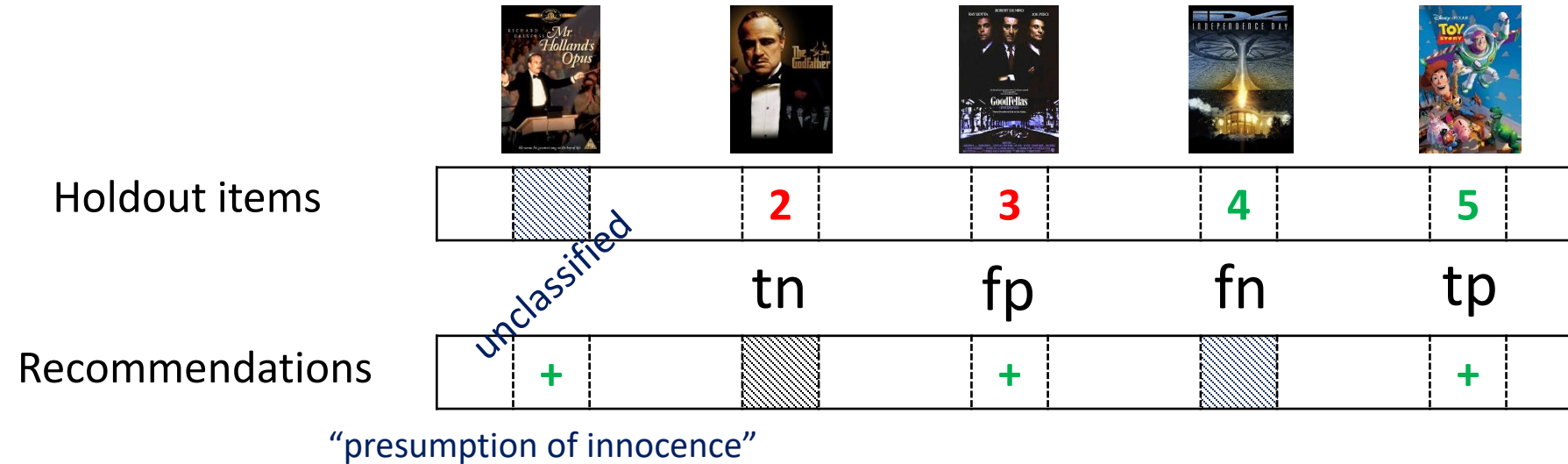
$$nDCG = \frac{DCG}{iDCG}$$

$iDCG = DCG$ with ideal ranking of items

Doesn't take into account the type of feedback: positive or negative.

Need to redefine metrics.

Using classical definition



Relevance based

$$Precision = \frac{tp}{tp + fp} \quad Recall = \frac{tp}{tp + fn}$$

Ranking based

$$DCG = \sum_p \frac{2^{r_p} - 1}{\log_2(p + 1)}$$

$p : \{r_p \geq \text{positivity threshold}\}$

r_p - value of positive feedback

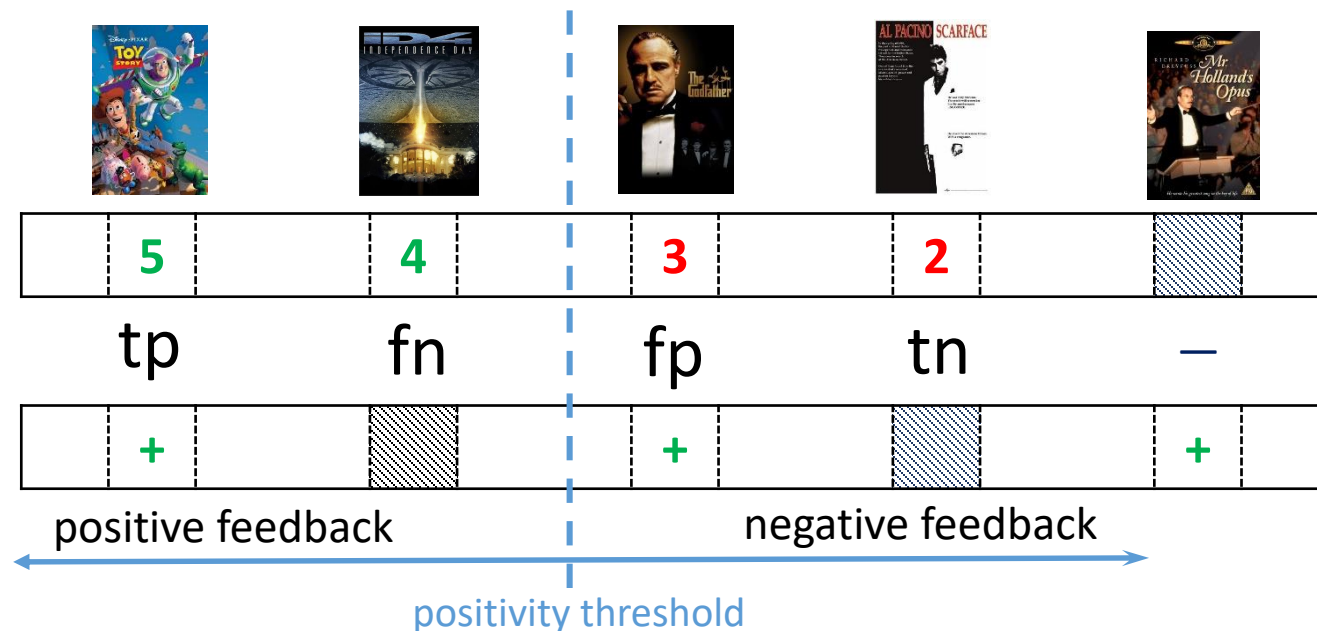
New ranking metric

Methodology:

1. split feedback into **positive** and **negative**
2. treat negative part differently

Known preferences
(rel_i)

Recommendations



Standard metric

$$DCG = \sum_{i \in I^+} \frac{2^{rel_i} - 1}{\log_2(i + 1)}$$

Discounted
Cumulative
Gain

$$I^+ = \{i: rel_i \geq r_p\} \quad r_p - \text{positivity threshold}$$

measures potential user's enjoyment

New metric

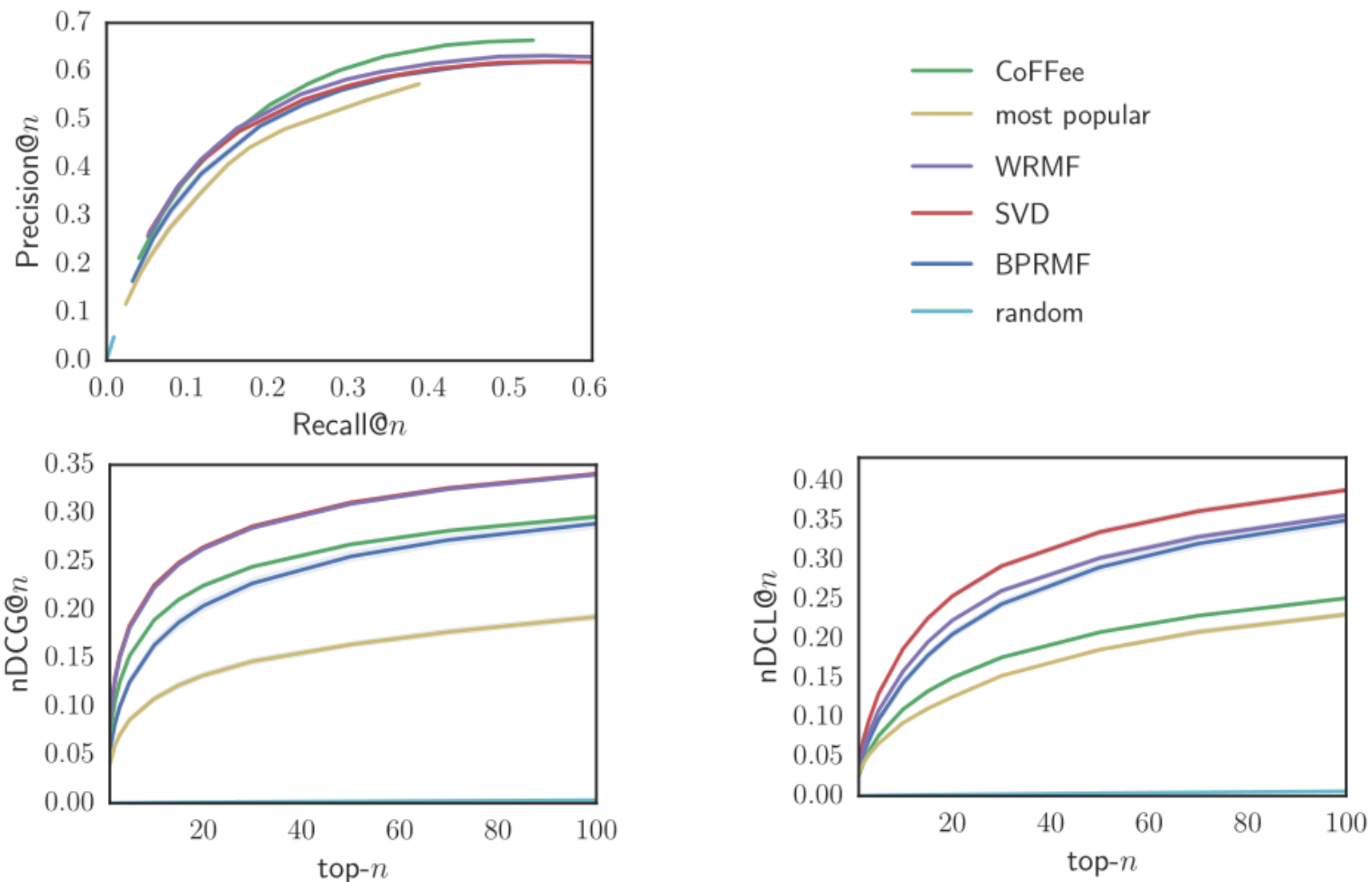
$$DCL = \sum_{i \in I^-} \frac{2^{r_p - rel_i} - 1}{-\log_2(i + 1)}$$

Discounted
Cumulative
Loss

$$I^- = \{i: rel_i < r_p\}$$

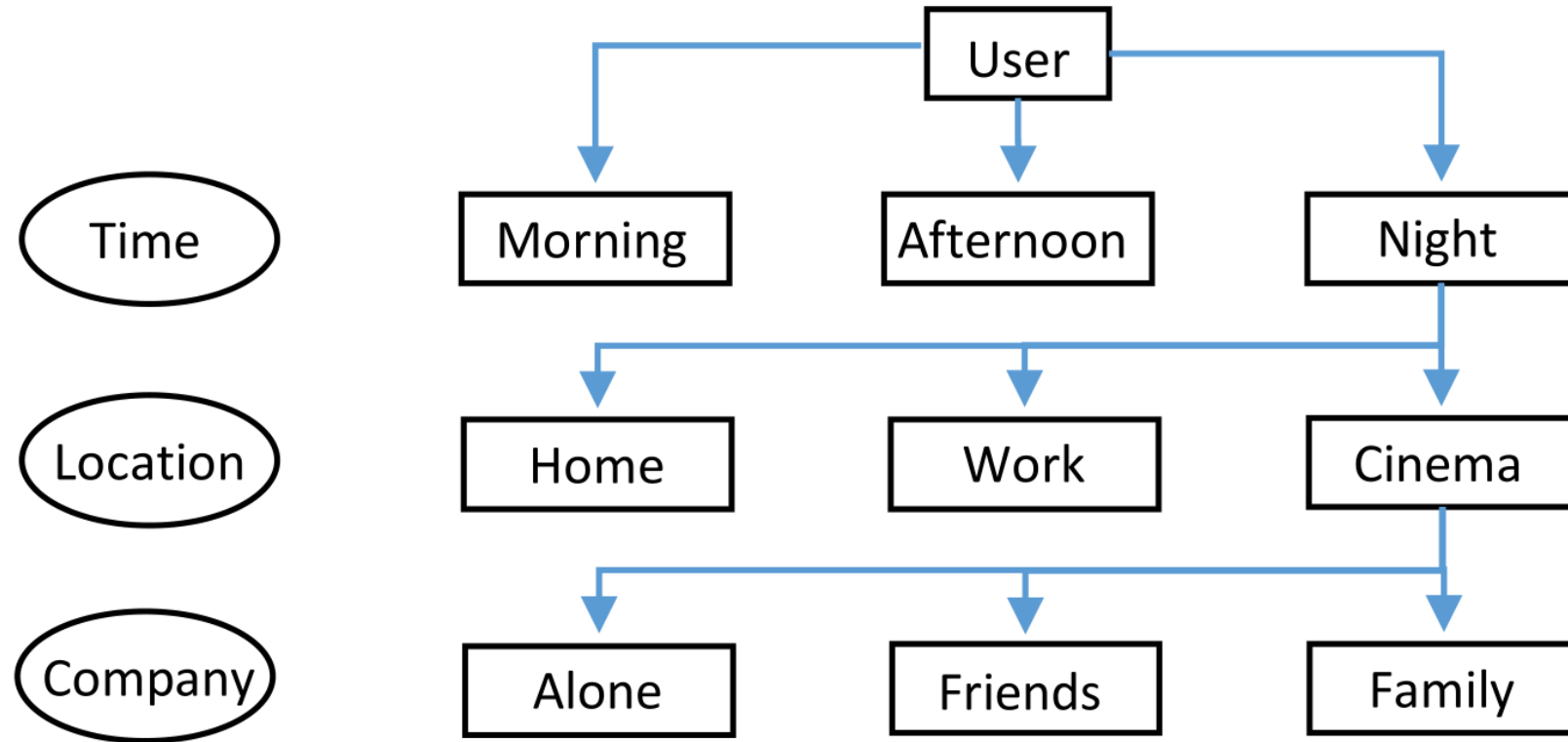
measures potential user's disappointment

Case study – which algorithm is better?



Read more: ***“Fifty shades of ratings: How to Benefit from Negative Feedback in Top- n Recommendations Tasks”***, Evgeny Frolov and Ivan Oseledets.
Proceedings of the 10th ACM Conference on Recommender Systems, 2016, pp. 91-98.

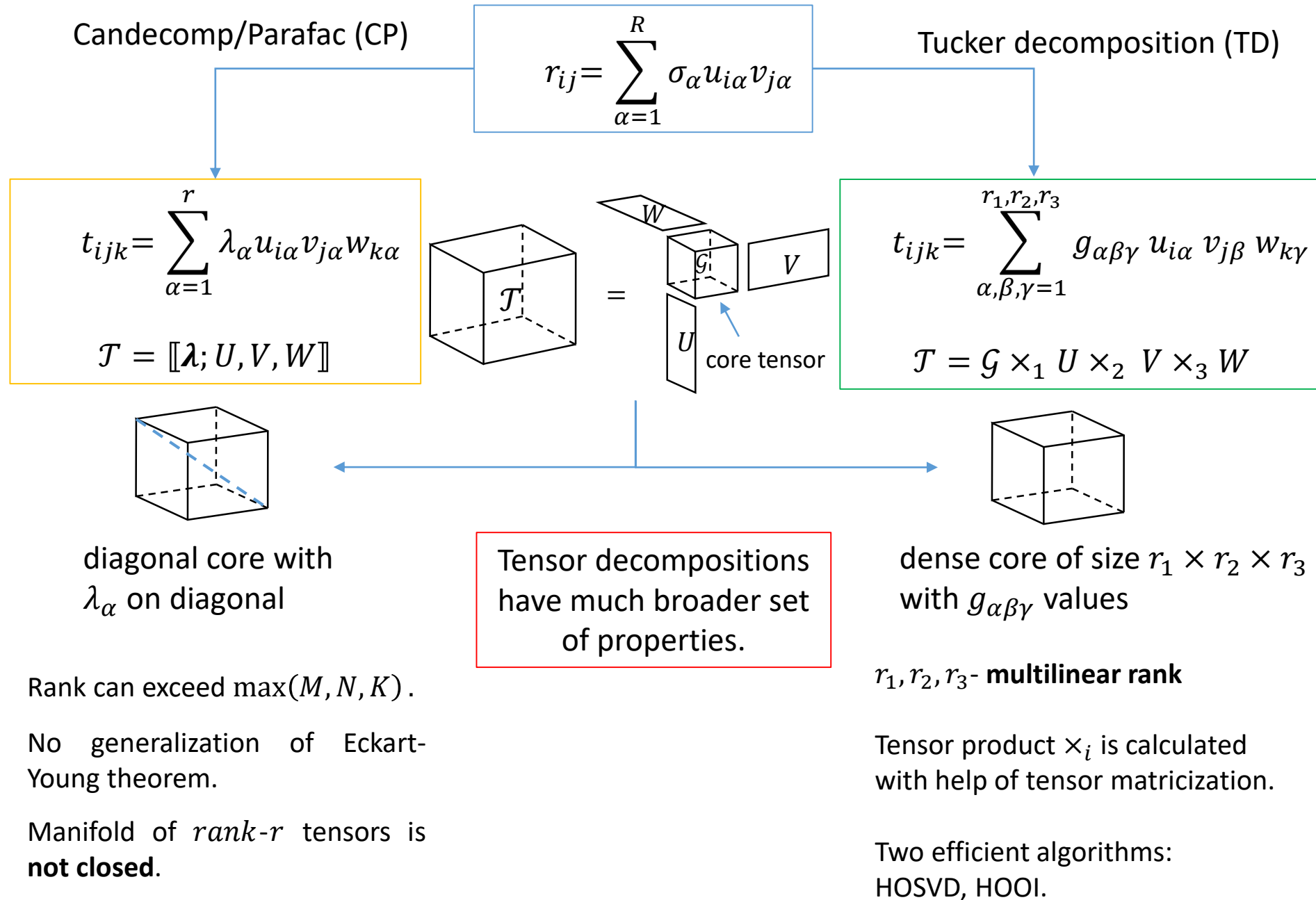
Tensors are also good for context-aware RecSys



Also: folksonomies, cross-domain RS, temporal models

Read more about tensors in recsys: “**Tensor methods and recommender systems**”, Evgeny Frolov and Ivan Oseledets.
WIREs Data Mining Knowledge Discovery 2017, vol. 7, issue 3.

Tensor decompositions – higher order generalization of SVD



Key properties of tensor decompositions

Properties	TD	CP
Storage requirements	$dnr + r^d$ curse of dimensionality	dnr
Uniqueness	No	Yes (under mild conditions)
Stability	stable	ill-posed*

*V. De Silva and L.-H. Lim. Tensor rank and the ill-posedness of the best low-rank approximation problem.

In recommender systems*:

Method	Year	Type	Algorithm
TagTR	2008	TD	HOSVD
Multiverse	2010	TD	SGD
PITF	2010	CP	SGD
TFMAP	2012	CP	SGD
GFF	2015	CP	ALS

*More examples in “Tensor Methods and Recommender Systems”, E.Frolov, I.Oseledets <http://arxiv.org/abs/1603.06038>

Your home task

- Download Movielens-10M dataset:
<http://files.grouplens.org/datasets/movielens/ml-10m.zip>
 - test data will be provided separately
 - it's not a part of the dataset
- Try your favorite models on it
- Try comparing several models/configs via cross-validation

This is the dataset that will be used during the team competition!