SYSTÈMES DE RECOMMANDATION

S2 (2023-2024)

Cours 9

~~Nicolas Baskiotis~~

~~nicolas.baskiotis@sorbonne-universite.fr~~

Slides de Vincent Guigue

Master 1 DAC

équipe MLIA, Institut des Systèmes Intelligents et Robotique (ISIR)
Sorbonne Université

Plan

Recommender Systems in several dates

SCIENCES SORBONNE UNIVERSITÉ

- 1998 Amazon item-to-item recommendation
- 004-Now Special sessions in recommender system in several important conferences & journals:

  AI Communications ; IEEE Intelligent Systems; International Journal of Electronic Commerce; International Journal of Computer Science and Applications; ACM Transactions on Computer-Human Interaction; ACM Transactions on Information Systems
- 2007 First ACM RecSys conference
- 2008 Netflix online services (& innovative HMI)
- 2008-09 Netflix RS prize
- 010-Now RS become essential : `YouTube`, `Netflix`, `Tripadvisor`, `Last.fm`, `IMDb`, etc...

Why developing a Recommender Sysytem? [Seller's/User's perspectives]

Seller

- Increase the number of items sold
- Sell more diverse items
- Increase the user satisfaction
  Increase user fidelity
  Virtuous loop:
  ⇒ improving the profiles & exploiting them
- Better understand what the user wants

User

- Find some good items *[precision issu*
  - quickly
  - and/or in a huge catalog
- Find all good items *[recall issue]*
  - Lawyers Information Retrieval task
- Being recommended a sequence / a bundle
- Just browsing
- Help others (forum profiles)

The value of recommendations

- Netflix: 2/3 of the movies watched are recommended
- Google News: recommendations generate 38% more clickthrough
- Amazon: 35% sales from recommendations
- Choicestream: 28% of the people would buy more music if they found what they liked.

The value of recommendations

SCIENCES SORBONNE UNIVERSITÉ

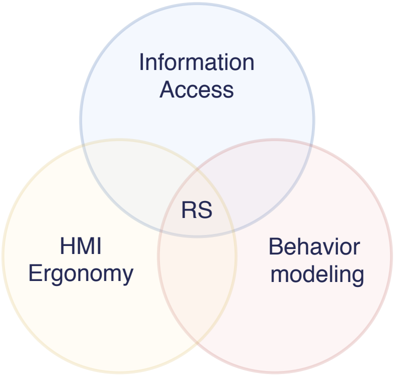
- Netflix: 2/3 of the movies watched are recommended
- Google News: recommendations generate 38% more clickthrough
- Amazon: 35% sales from recommendations
- Choicestream: 28% of the people would buy more music if they found what they liked.

Optimistic assumption:

small RMSE gain $\Rightarrow$ bigger qualitative gain

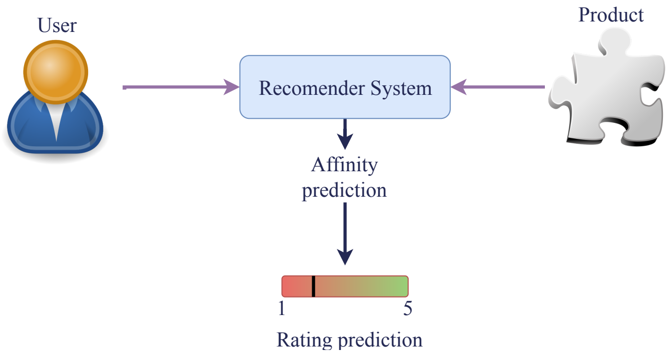
General position of Recommender Systems

SCIENCES SORBONNE UNIVERSITÉ



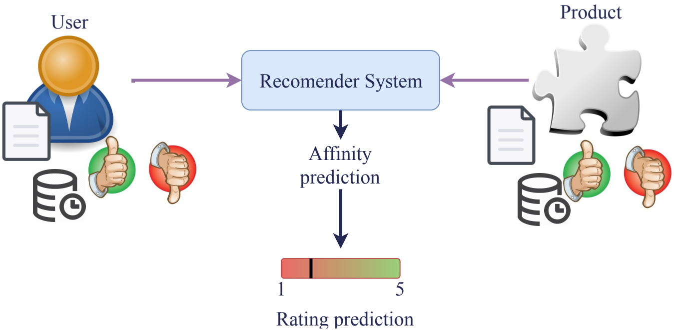
- Information Access:
  RS become inescapable when information sources grow exponentially
- Behavior modeling:
  - Business expertise
  - Crowd sourcing
- Ergonomy:
  Very important... But not the topic today

How to build a Recommender System? General map



How to build a Recommender System? General map

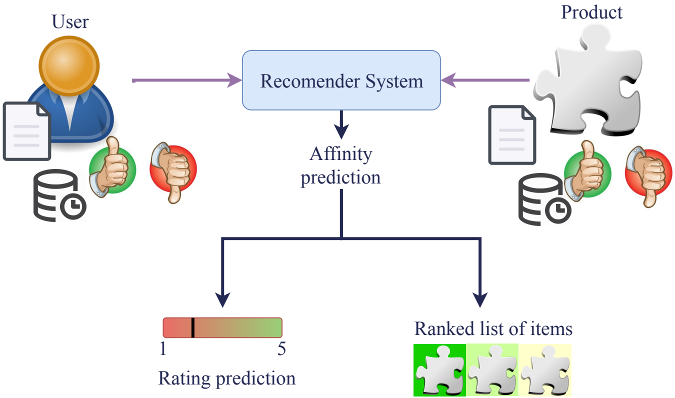
SCIENCES SORBONNE UNIVERSITÉ



First session: introduction to RecSys

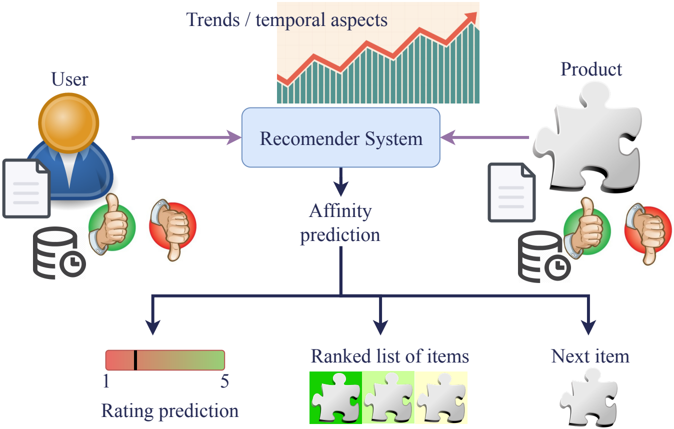
How to build a Recommender System? General map

SCIENCES SORBONNE UNIVERSITÉ



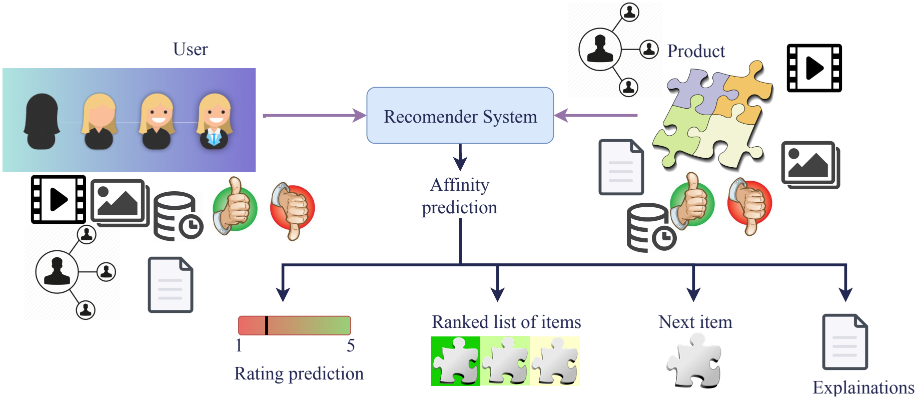
Second session: new systems, new evaluation metrics

How to build a Recommender System? General map



Third session: temporal aspects

How to build a Recommender System? General map



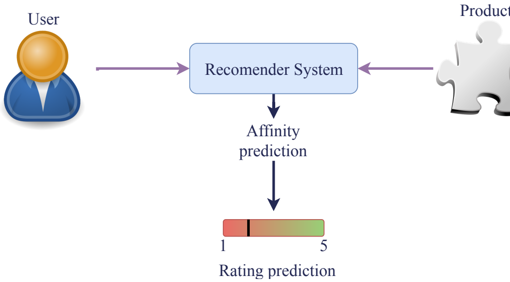
Forth session: deep learning for RecSys

The Recommender problem

Estimate a *utility function* that automatically predicts how a user **will like** an item.

Based on:

- Past behavior
- Relations to other users
- Item similarity
- Context
  - Time,
  - Sequence,
  - Item description,
  - User categorization:
    age, socio-professional category, ...



Different approaches

Affinity prediction:

- Content based
  - Tabular
  - Textual
- Collaborative filtering
  - Memory
  - Model

Choosing an approach:

- Depending on **available data**
  - Item descriptions
  - User/item Interactions
- Depending on **requirements**
  - Quick implementation
  - Efficient inference
  - Expected diversity : low/high

Plan

Content based approches (CB): Automation of editorial choices?

Mainly **item centered**

Understanding the product description to...

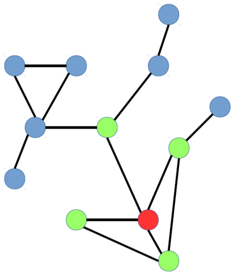
- Know which items are similar
  [global description]
- Focus on common points between various items
  [part of the description]

Paradigm of browsing:
you liked A, did you already consider B,C &D that are close?

[CB] Static implementation & scaling up

Learning step :

1. Feature engineering:
   Item description $\Rightarrow$ relevant vector
2. k-Nearest Neighbors graph
3. Product description update



Inference

- Presenting new informations within the product description

Issue: How to make the representation relevant?

[CB] Nature of the description & associated metrics

SCIENCES SORBONNE UNIVERSITÉ

- **Tabular description :**
  - Hierarchy of domains (e.g. product categories in online shops)
  - Descriptive features (often in a given domain)

    (e.g. camera $\Rightarrow$ definition, zoom, storage capacity, brand, ...)

$\Rightarrow$ mostly an engineering job + domain expert knowledge

(understanding & weighting the features)

[CB] Nature of the description & associated metrics

SCIENCES SORBONNE UNIVERSITÉ

- **Tabular description :**
  - Hierarchy of domains (e.g. product categories in online shops)
  - Descriptive features (often in a given domain)
    (e.g. camera ⇒ definition, zoom, storage capacity, brand, ...)

⇒ mostly an engineering job + domain expert knowledge
(understanding & weighting the features)

- **Textual description :** ⇒ Information Retrieval (IR)/NLP
  - Matching **raw texts**:
    preprocessing issues (stop words, basic language structure, ...)
  - **Keyword**-based Vector Space Model (TF-IDF, etc...)
  - **Topic** modeling: matching in the latent space
    - Internal or external topic modeling
  - **Ontology** / domain specific reference + mapping

⇒ Choosing a metric adapted to the representation
cosine for raw texts, KL for topic distribution, ...

[CB] User profiles

- Case 1 : explicit user profile
  - Textual description of the user...
- Case 2: no user profile
  - Query = stack of visited items
- Extracting items:
  - User = query, Item = document... An IR task : $p(u|i)$
  - Rocchio's relevance feedback
    1. Query $\Rightarrow$ set of responses
    2. first responses = query enrichments
    3. last (or other documents) $\approx$ negative query

Pros & Cons

SCIENCES SORBONNE UNIVERSITÉ

+ Explainable
+ Easy to implement
+ Scale up (& offline computations)
+ Can benefit from last advances to increase relevance

± (Often) not personalized...
but intrinsically robust to new users !

– Lack of an authority score (as in PageRank)
– Require an item description
– Not adapted to User Generated Contents (intrinsically)

⇒ Mostly an NLP engineering game to obtain baselines that will be combined to CF approaches...

Main usage

SCIENCES SORBONNE UNIVERSITÉ

Well adapted to

- Browsing reasonable sized catalog with meaningful explanatory variable
  - ⇒ Suggest blender with same capacities & price category
  - ⇒ Work of art in a museum collection : suggest pieces from an artist, from a period
- Textual dataset
  - Google scholar suggestions
  - Layer virtual assistant

Not adapted to

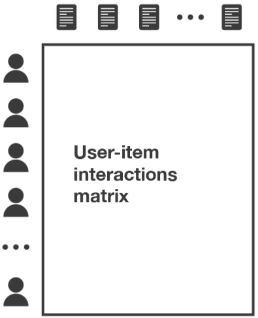
- Large catalog: lacks of authority score is penalizing.

Plan

General idea

Behavior modeling depending on **users' traces**:

- User Generated Contents (Explicit)
  - Ratings
  - Texts
  - Likes...
- Inferred informations
  - you liked what you purchase
  - you liked what you visit/rate
  - you don't like video you close less that 3 seconds after they started



Interaction data are valuable:

The best information filter is human...

Collaborative filtering = modeling humans from their traces

History: frequent item set

SCIENCES SORBONNE UNIVERSITÉ

Idea:

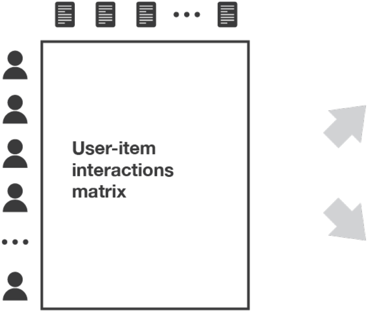
Extracting logical rules from (frequent) co-occurences

1. Frequent item set.
   e.g. receipt mining in a supermarket : $Milk, Beer, Diaper$
2. Extraction of the support. e.g. $(Milk, Diaper) \Rightarrow Beer$

+ Easy to understand
– Costly (combinatorial search)
– Not very robust to noise

A good explanation... but not an operational model

Organization of collaboratives approaches



No Model

- users and items are represented directly by their past interactions (large sparse vectors)
- recommendations are done following nearest neighbours information

---

Model

- new representations of users and items are build based on a model (small dense vectors)
- recommendations are done following the model information

Introduction to recommender systems, Baptiste Rocca
https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada

Neighborhood based approaches

SCIENCES SORBONNE UNIVERSITÉ

In collaborative filtering...

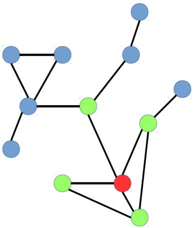
User domain: if you behave as user u, then you might be interested by u's choices

Item domain: item i is often associated to item i' in users' traces; if you visit i, you might be interested by i'

Same approach than Content Based...
Based on another behavior sensor!

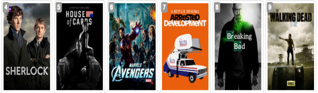
$\Rightarrow$ In the item domain = very **light inference**
... But such RS is **not personalized** !

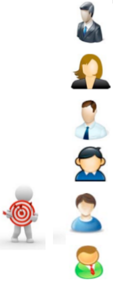
Item-to-item CF $\Rightarrow$ The early Amazon approach



Computing k-nearest neighbors in the user domain

Easy way to perform a personalized recommendation...



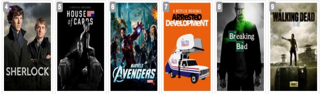


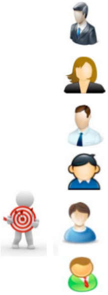
						sim(u,v)
2		2	4	5		NA
5		4			1	
		5		2		
	1		5		4	
		4			2	
4	5		1			NA

Credit: X. Amatrian

Computing k-nearest neighbors in the user domain

Easy way to perform a personalized recommendation…





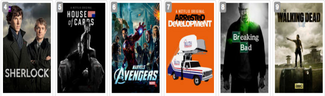
						sim(u,v)
2		2	4	5		NA
5		4			1	0.87
		5		2		
	1		5		4	
		4			2	
4	5		1			NA

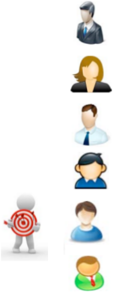
Credit: X. Amatrian

Computing k-nearest neighbors in the user domain

Easy way to perform a personalized recommendation...

$$\hat{r_{ui}} = \frac{\sum_{v \in \mathcal{U}} \alpha_{uv} r_{vi}}{\sum_{v \in \mathcal{U}} \alpha_{uv}}$$





						sim(u,v)
2		2	4	5		NA
5		4			1	0.87
		5		2		1
	1		5		4	-1
3.51*	3.81*	4	2.42*	2.48*	2	
4	5		1			NA

Credit: X. Amatrian

Computing k-nearest neighbors in the user domain

SCIENCES SORBONNE UNIVERSITÉ

Easy way to perform a personalized recommendation...

$$\hat{r_{ui}} = \frac{\sum_{v \in \mathcal{U}} \alpha_{uv} r_{vi}}{\sum_{v \in \mathcal{U}} \alpha_{uv}}$$

...But very **expensive** in the inference step !

- Bottleneck = Similarity computation + sort
- Complexity is $\mathcal{O}(n_u n_i + k n_u)$
  - Possible approximation (partitioning/hashing space) : LSH
- Possible implementation:
  - Isolate the neighborhood generation and predication steps.
  - "off-line component" / "model" – similarity computation, done earlier & stored in memory.
  - "on-line component" – prediction generation process.

Computing k-nearest neighbors in the user domain

Easy way to perform a personalized recommendation...
To be more efficient:

user normalized computations

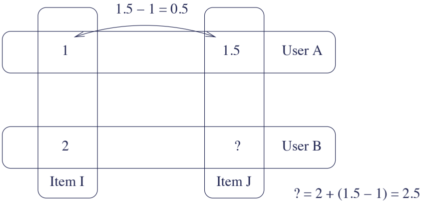
$$\hat{r_{ui}} = \mu_u + \frac{\sum_{v \in \mathcal{U}} \alpha_{uv}(r_{vi} - \mu_v)}{\sum_{v \in \mathcal{U}} \alpha_{uv}}$$

- Bottleneck = Similarity computation + sort
- Complexity is $\mathcal{O}(n_u n_i + k n_u)$
  - Possible approximation (partitioning/hashing space) : LSH
- Possible implementation:
  - Isolate the neighborhood generation and predication steps.
  - "off-line component" / "model" – similarity computation, done earlier & stored in memory.
  - "on-line component" – prediction generation process.

C.Desrosiers & G. Karypis, RecSys 2011
A comprehensive survey of neighborhood-based recommendation methods

Classical memory based alternative (1) : Slope one

SCIENCES SORBONNE UNIVERSITÉ

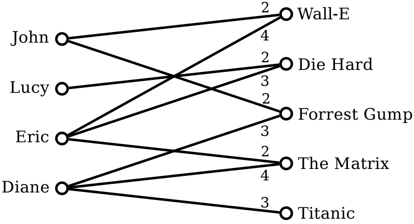


A simple and efficient approach to collaborative filtering:

- computing the average difference between items
- every user u with a rating on i & connected to u' can give a prediction on $r_{u'i}$

Classical memory based alternative (2) : Bi-partite graph approach

SCIENCES SORBONNE UNIVERSITÉ



- Shortest path between items
- Heaviest path = number of paths
- Random walk similarity
  - Markov model

\+ A new way to compute –offline– similarity between items
  - Longer dependancies

– Expensive online path computation (⇔ personalized reco)

– Usually gives the same results as matrix factorization

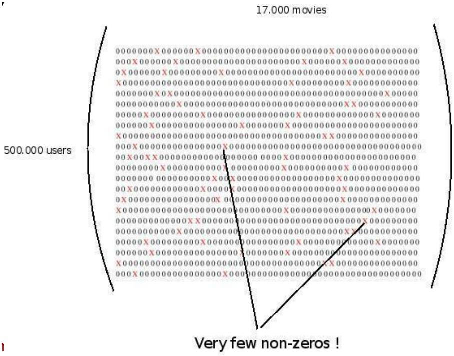
From memory to model: the missing value paradigm

Netflix Prize rating matrix

If you represent the Netflix Prize rating data in a User/Movie matrix you get...

- 500,000 x 17,000 = 8,500 M positions
- Out of which only 100M are not 0's!

Data Set	users	items	total	density
Jester	48483	100	3519449	0,725
MovieLens	6040	3952	1000209	0,041
EachMovie	74424	1649	2811718	0,022



Bayesian formulation : estimating missing values

Modeling:

$$p(m_1 = k | m_2, m_3, \dots)$$

Expectation-Maximization framework

- A lot of parameters to model the conditional distribution
  - Not adapted to large catalog + sparse observations
- Basic hypothesis: Missing Completely at Random (MCAR)

2		2	4	5	
5		4			1
		5		2	
	1		5		4
		4			2
4	5		1		

Matrix factorization

SCIENCES SORBONNE UNIVERSITÉ

Idea:

Compressing the representation of the matrix based on observed values is a strong way to reconstruct missing values.

- Singular Value Decomposition (SVD)
- Non Negative Matrix Factorization (NMF)
- ... & many variations

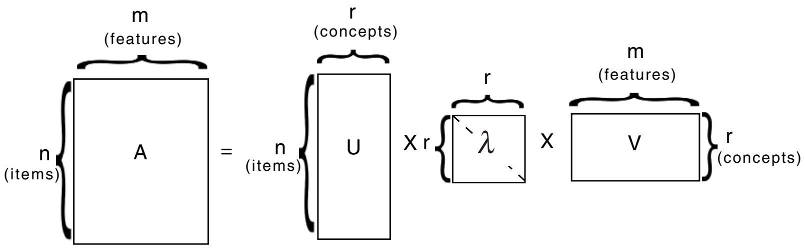
Link with *Minimum Description Length* paradigm:

What is the smallest modeling that can explain observed ratings?

Singular Value Decomposition

SCIENCES SORBONNE UNIVERSITÉ

Framework of matrix factorization over non square matrix = SVD



- Not adapted to missing values... ⇒ turn into 0.
- Weak reconstruction performance...

⇒ SVD for recommender systems... **Is not an SVD** !

SVD for recommender systems

SCIENCES SORBONNE UNIVERSITÉ

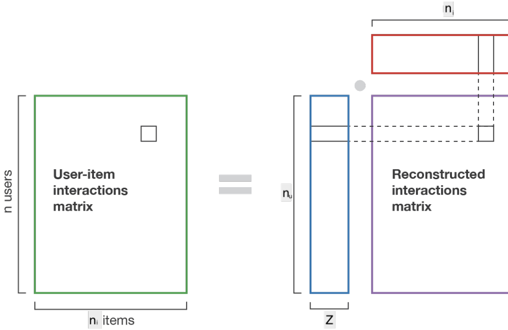
$$U = \{\mathbf{u}_1, \ldots, \mathbf{u}_{n_u}\}, I = \{\mathbf{i}_1, \ldots, \mathbf{i}_{n_i}\}$$

$$\mathbf{u} \in \mathbb{R}^z, \mathbf{i} \in \mathbb{R}^z$$

$$R = \{(u, i, r_{ui})\}$$

Estimator : $\hat{r}_{ui} = \mathbf{u}_u \cdot \mathbf{i}_i$

$$\mathcal{C} = \sum_{(u,i) \in R} (r_{ui} - \mathbf{u}_u \cdot \mathbf{i}_i)^2$$



Focus on missing values + Mean Square Error (MSE)

$$U^\star, I^\star = \arg\min_{U,I} \sum_{(u,i) \in R} (r_{ui} - \mathbf{u}_u \cdot \mathbf{i}_i)^2$$

SVD issues

SCIENCES SORBONNE UNIVERSITÉ

1. **Optimization**: (stochastic) gradient descent

$$\nabla_{\mathbf{u}}\mathcal{C} = -\sum_{i|(i,u)\in R} 2\mathbf{u}(r_{ui} - \mathbf{u}_u \cdot \mathbf{i}_i), \qquad \mathbf{u} \leftarrow \mathbf{u} - \varepsilon\nabla_{\mathbf{u}}\mathcal{C}$$

   - Fast convergence ...
   - ... but non convex formulation
   - First optimizers based on multiplicative updates... No longer used.

2. **Overfitting**:
   Even with $z = 20$: $\#param = 20 \times (n_u + n_i) \geq |R|$
   $\Rightarrow$ Regularization:

$$U^\star, I^\star = \arg\min_{U,I} \sum_{(u,i)\in R} (r_{ui} - \mathbf{u}_u \cdot \mathbf{i}_i)^2 + \lambda_u\|U\|_F^2 + \lambda\|I\|_F^2$$

   - Implementation : penalizing weights every λ iterations

Introducing the bias: baselines + model improvements

Let's go back to the basics... & the baselines

- General bias: $b = \bar{r}$
- User bias: $b_u = \frac{1}{|\{i|r_{ui} \neq \emptyset\}|} \sum_{i|r_{ui}\neq\emptyset} r_{ui}$

Hyp: one user always gives the same rate

- Item bias: $b_i = \frac{1}{|\{u|r_{ui} \neq \emptyset\}|} \sum_{u|r_{ui}\neq\emptyset} r_{ui}$

Strong Hyp: one item is always evaluated with the same rate

We obtain **three baselines**... And an advanced formulation:

$$\hat{r}_{ui} = b + b_u + b_i + \mathbf{u}_u \cdot \mathbf{i}_i$$

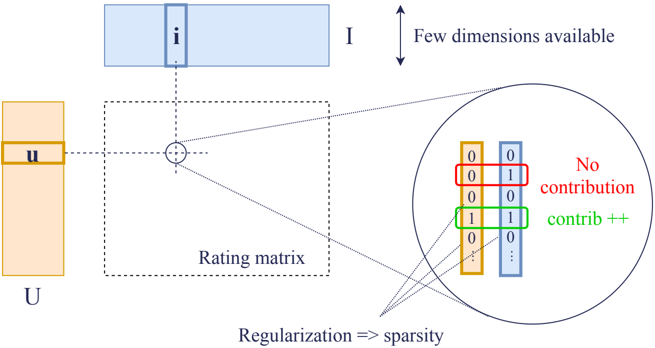
$\Rightarrow$ $\mathbf{u}_u, \mathbf{i}_i$ profiles encode the deviation wrt basic predictions

NMF: the promise of understandable aspects

SCIENCES SORBONNE UNIVERSITÉ

NMF: Non-negative Matrix Factorization = SVD + $\mathbf{u} \geq 0$ + $\mathbf{i} \geq 0$

$$U^\star, I^\star = \arg\min_{U,I} \sum_{(u,i)\in R} (r_{ui} - \mathbf{u}_u \cdot \mathbf{i}_i)^2 + \lambda_u \|U\|_F^2 + \lambda \|I\|_F^2$$



- Regularization brings rating to
- Issue = finding aspects shared by many users

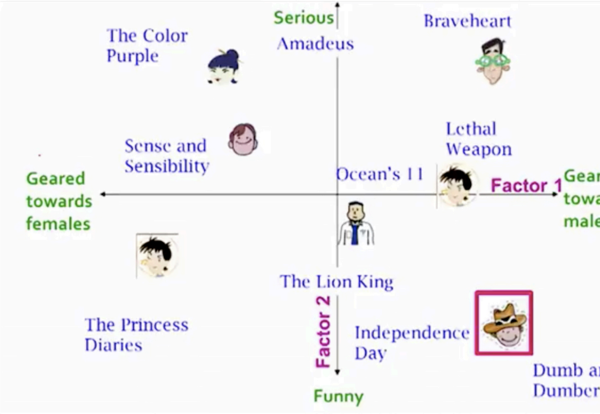
Expectations

What we expect:

- Efficient & reliable suggestions
- Explanations

What we have:

- Iterative (light) procedure + simple SGD
- Easy to enforce constraint:
  - Orthogonality
  - Specific initialization
  - Modeling of negative agreements
  - ...



Cold start

- Collaborative Filtering = exploiting traces...

  ⇒ What can I do at the beginning?

  - new users, new items

⇒ Hybrid systems (content based + collaborative filtering)

start = item description + content based recommendations

⇒ Forcing an initial feedback

  - e.g. Netflix

⇒ Using external source

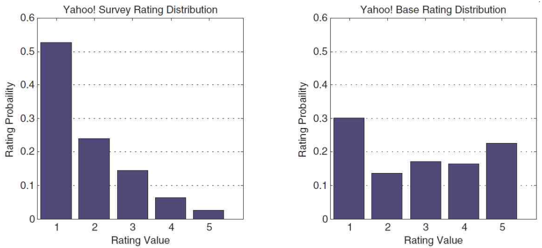
  - log in with Facebook, scanning user contacts, web history,...
  - building an item profile (editorial work)

Plan

Main issue : the weakness of MCAR hypothesis

Data are not Missing Completely At Random...

Graphs from [Marlin & Zemel '09]:



Survey: ask users to rate a random list of items: approximates **complete** data

Typical Data: users are free to choose which items to rate -> available data are **MNAR** : instead of giving low ratings, users tend to not give a rating at all.

Even different in product/movie domains:

- 60-80% of 4/5 ratings

Main issue : the weakness of MCAR hypothesis

Data are not Missing Completely At Random...

Predicting profile behavior on this kind of data:

H. Steck, KDD, 2010
Training and Testing of Recommender Systems on Data Missing Not at Random

Table 1: Simplistic Example for ratings missing not at random (MNAR): test data where users rated only what they liked or knew.

		users							
		horror fans				romance lovers			
	h	5		5	5				
m	o	5	5						
o	r		5		5				
v	.	5		5	5				
i	r					5	5		5
e	o						5	5	5
s	m					5		5	
	.						5	5	5

Credit: H. Steck

Main issue : the weakness of MCAR hypothesis

Data are not Missing Completely At Random...

Several outcomes:

- Changing the **error function**
  - Modelling missing values
  - Switching to a ranking criteria
- Changing **the task**
  - predicting rated item (not the rate)

How to evaluate RS performance?

Warning

We should not confuse **evaluation metrics** & **learning metrics**

$\Rightarrow$ MSE is a convenient learning metrics

(easily differentiable + convex ...)

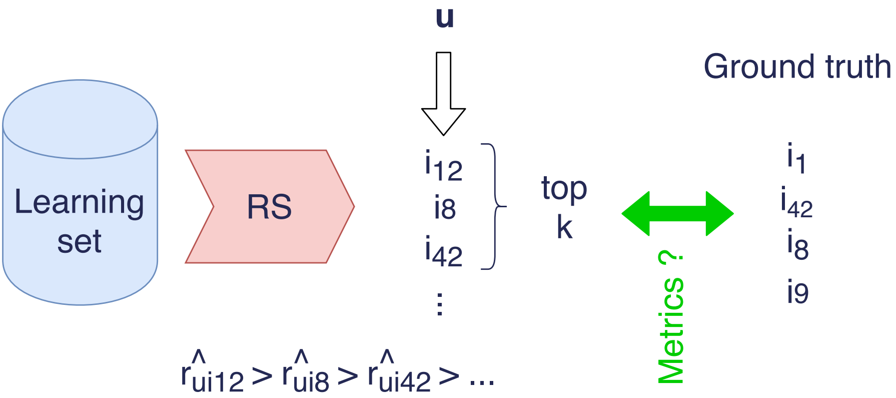
... but it is a poor evaluation metrics

... cf Netflix Challenge feedbacks

It do not tell us if we provide relevant suggestions

- What are the other available metrics?
  - Have a look towards the IR community
- Can we use those metrics during the learning step?

Precision / Recall



- **Precision :** Among our k prediction, how many are in the ground truth?
- **Recall :** Among our k prediction, what is the GT coverage ?

1/0 labeling, AUC metrics

SCIENCES SORBONNE UNIVERSITÉ

- Rendle popularize both 1/0 prediction & AUC metrics
- AUC = tradeoff between precision & recall
  - Percentage of correct binary ranking for ONE user
  - Aggregation over n_u users

$$AUC = \frac{1}{n_u} \sum_u \frac{1}{|E(u)|} \sum_{(i,j) \in E(u)} \delta(\mathbf{u} \cdot \mathbf{i} > \mathbf{u} \cdot \mathbf{j})$$

+ k not required

− **top** of the list = same impact as **bottom** of the list

Mean Average Precision (from the IR domain)

RS aim at proposing an ordered list of suggestion...

Which **head is far more important** than the rest.

For a user u with 4 liked items to discover:

$$query = \mathbf{u} \Rightarrow RS_1 \Rightarrow \begin{bmatrix} i_{12} \\ i_8 \\ i_{42} \\ i_1 \end{bmatrix} \quad \Leftrightarrow \quad \begin{bmatrix} i_1 \\ i_{42} \\ i_8 \\ i_9 \end{bmatrix} = GT$$

- Average precision (one query/user) :

$$\frac{1}{K}\sum_{k=1}^{K} precision@K = \frac{1}{4}(0 + \frac{1}{2} + \frac{2}{3} + \frac{3}{4}) = 0.478$$

- Mean Average Precision =

Averaging over the whole population

Mean Average Precision (from the IR domain)

RS aim at proposing an ordered list of suggestion...

Which **head is far more important** than the rest.

For a user u with 4 liked items to discover:

$$query = \mathbf{u} \Rightarrow RS_2 \Rightarrow \begin{bmatrix} i_1 \\ i_8 \\ i_{42} \\ i_{12} \end{bmatrix} \quad \Leftrightarrow \quad \begin{bmatrix} i_1 \\ i_{42} \\ i_8 \\ i_9 \end{bmatrix} = GT$$

- Average precision :

$$\frac{1}{4}\sum_{k=1}^{4} precision@K = \frac{1}{4}(1+1+1+\frac{3}{4}) = 0.9375$$

- Mean Average Precision =

Averaging over the whole population

Mean Reciprocal Rank

SCIENCES SORBONNE UNIVERSITÉ

At which rank is the first relevant item?

$$query = \mathbf{u} \Rightarrow RS \Rightarrow \begin{bmatrix} i_{12} \\ i_8 \\ i_{42} \\ i_1 \end{bmatrix} \quad \Leftrightarrow \quad \begin{bmatrix} i_1 \\ i_{42} \\ i_8 \\ i_9 \end{bmatrix} = GT$$

$$RR = \frac{1}{rank_i} = \frac{1}{2} \text{ on previous example}$$

Mean Reciprocal Rank = Averaging over the whole population

$\Rightarrow \approx$ *How many iterations to obtain a relevant item?*

nDCG : Normalized Discounted Cumulative Gain

We assume that we have a relevance score for each item...

$$query = \mathbf{u} \Rightarrow RS \Rightarrow \left[\begin{array}{c|cc} & i_{12} & ind = 1 \\ & i_8 & ind = 2 \\ & i_{42} & ind = 3 \\ \downarrow & i_1 & ind = 4 \end{array}\right] \Leftrightarrow \begin{bmatrix} 0 \\ 2 \\ 3 \\ 3 \end{bmatrix} = relevance$$

$$\mathrm{DCG_p} = \sum_{ind=1}^{p} \frac{relev_{ind}}{\log_2(ind+1)} = 0 + 1.26 + 1.5 + 1.29 = 4.05$$

$$\mathrm{nDCG} = \frac{\mathrm{DCG}}{\mathrm{IdealDCG}} = \frac{4.05}{3 + 1.89 + 1 + 0/0.86} = 0.69/0.6$$

Relative ideal (among suggestions) *vs* Absolute ideal (among all items)

ATOP

recall@k =

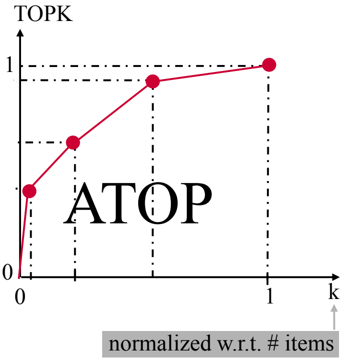
$$\frac{\#\text{relevant items in top k}}{\#\text{relevant items}}$$

Compute all recall@k...

until k match $R(u)$

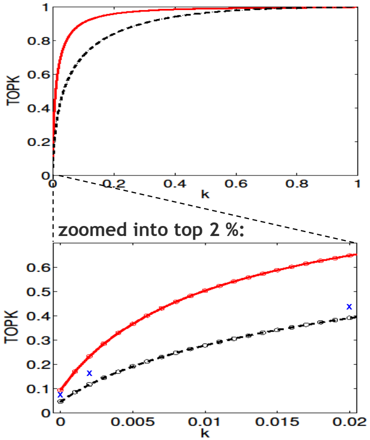
Compute the area under the curve

- focus on rated items
- numerical indicator + graphical details



H. Steck, KDD, 2010
Training and Testing of Recommender Systems on Data Missing Not at Random

ATOP



Comparison of Approaches:

— AllRank (RMSE = 1.106)

- - ignore missing ratings (RMSE = 0.921)

x integrated model [Koren ’08] (RMSE = 0.887)
(trained to minimize RMSE)

Large increase in Top-k Hit-Rate when accounting also for missing ratings when training on MNAR data.

39 % .... 50 % larger Top-k Hit-Rate: AllRank vs. integrated model

H. Steck, KDD, 2010
Training and Testing of Recommender Systems on Data Missing Not at Random

A/B testing & production launch

In a real situation

Designing an online Recommender System offers new performance indicators

- Online click, purchase, etc

A/B **testing:**

1. Defining some performance indicator with expert
2. Re-direct a small part of the customers to the new system B
   - make sure that the redirection is random (not biased)
3. Compare indicators from A and B

$\Rightarrow$ **Best evaluation**...

But only available **online** & with **access to the backoffice**

Serendipity : another important factor to evaluate...

... But very difficult to quantify

- Exploration / exploitation dilemma
- Clustering / categorization exploitation
  - propose items from different region
- Post processing / HMI issue

Idea to design a metric

1. Learn a strong baseline (SVD)
2. New system RS
3. Unexpectedness = RS\SVD
4. Serendipity = usefulness(Unexpectedness)

CF can offer serendipity

- increase neighborhood,
- increase implicit feedback weight
- ...

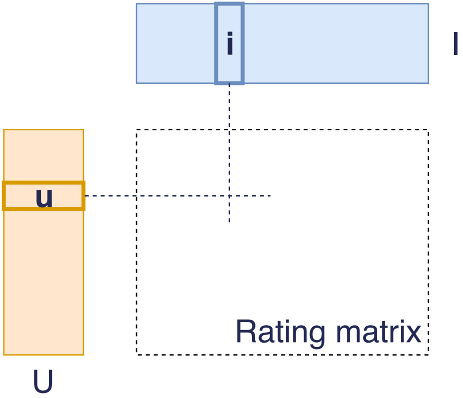
CB is not well adapted

- Clustering heuristics
- bad performance

M. Ge et al., RecSys, 2010
Beyond Accuracy: Evaluating Recommender Systems by Coverage and Serendipity

Plan

SVD is a NN architecture



$$U = \{\mathbf{u}_1, \ldots, \mathbf{u}_{n_u}\}$$

$$I = \{\mathbf{i}_1, \ldots, \mathbf{i}_{n_i}\}$$

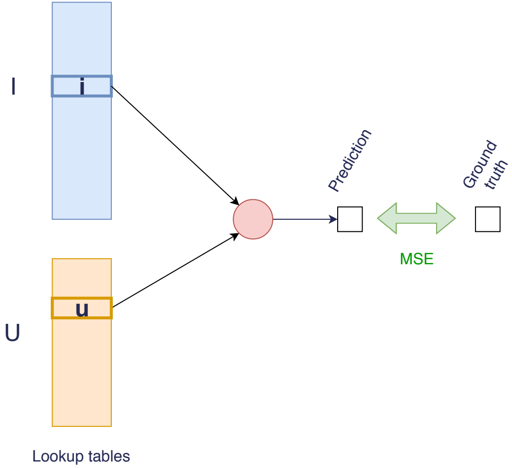
$$\mathbf{u} \in \mathbb{R}^z, \mathbf{i} \in \mathbb{R}^z$$

$$R = \{(u, i, r_{ui})\}$$

Estimator : $\hat{r}_{ui} = \mathbf{u}_u \cdot \mathbf{i}_i$

$$\mathcal{C} = \sum_{(u,i)\in R} (r_{ui} - \mathbf{u}_u \cdot \mathbf{i}_i)^2$$

SVD is a NN architecture



$$U = \{\mathbf{u}_1, \ldots, \mathbf{u}_{n_u}\}$$

$$I = \{\mathbf{i}_1, \ldots, \mathbf{i}_{n_i}\}$$

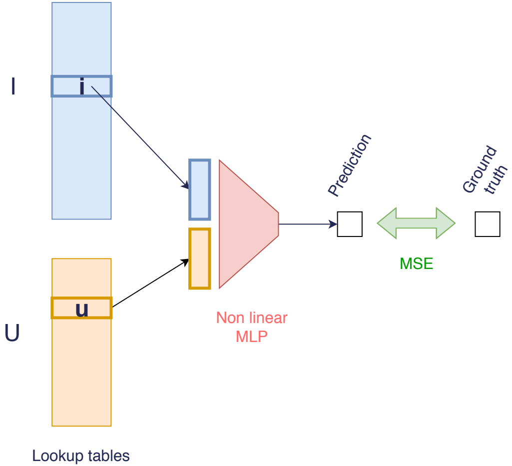
$$\mathbf{u} \in \mathbb{R}^z, \mathbf{i} \in \mathbb{R}^z$$

$$R = \{(u, i, r_{ui})\}$$

$$\text{Estimator : } \hat{r}_{ui} = \mathbf{u}_u \cdot \mathbf{i}_i$$

$$\mathcal{C} = \sum_{(u,i) \in R} (r_{ui} - \mathbf{u}_u \cdot \mathbf{i}_i)^2$$

MLP & RS: the simplest architecture

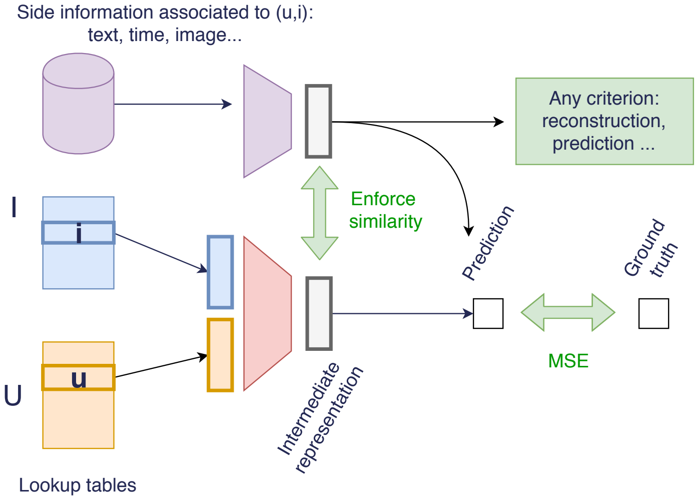


E.g. 2-layer perceptron

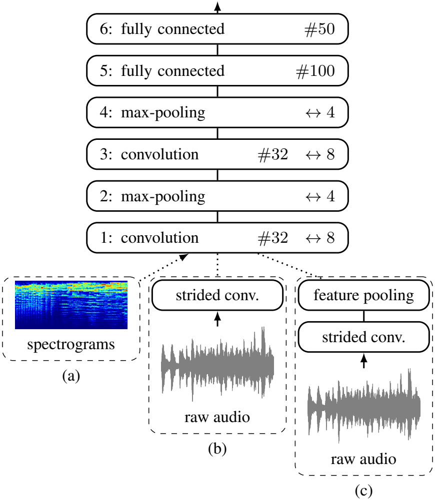
$$\mathbf{h} = f_1([\mathbf{u}_u \mathbf{i}_i] \cdot W_1)$$

$$\hat{r}_{ui} = f_2(\mathbf{h} \cdot W_2)$$

Deep = Easy handling of heterogeneous data

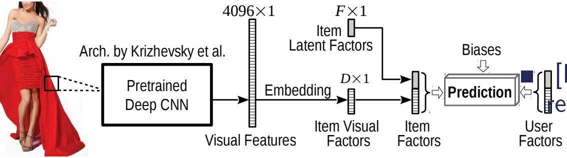


Deep = Easy handling of heterogeneous data

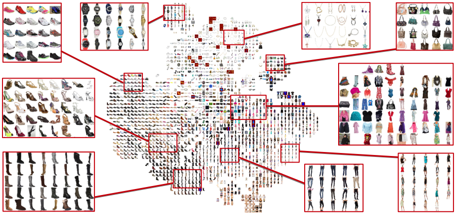


- [Dieleman, 2014] : audio recommendation
  - predict item profile from audio descriptors
  - ⇒ better understanding
- [He, 2015] : online product reco.
  - Image descriptors
- [Covington, 2016] : Youtube reco
- [Nedelec, 2017] : content2vec
  - Text + image descriptors

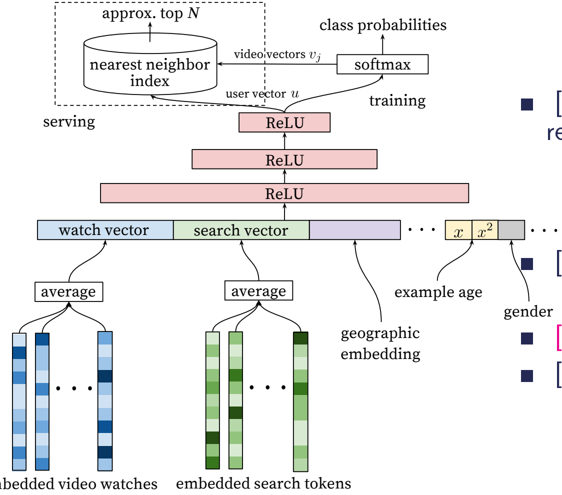
Deep = Easy handling of heterogeneous data



- [Dieleman, 2014] : audio recommendation
  - predict item profile from audio descriptors
  - ⇒ better understanding
- [He, 2015] : online product reco.
  - Image descriptors
- [Covington, 2016] : Youtube reco
- [Nedelec, 2017] : content2vec
  - Text + image descriptors



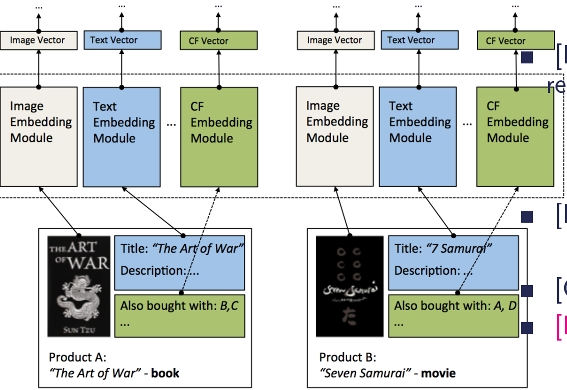
Deep = Easy handling of heterogeneous data



- [Dieleman, 2014] : audio recommendation
  - predict item profile from audio descriptors
  - ⇒ better understanding
- [He, 2015] : online product reco.
  - Image descriptors
- [Covington, 2016] : Youtube reco
- [Nedelec, 2017] : content2vec
  - Text + image descriptors

Deep = Easy handling of heterogeneous data

SCIENCES SORBONNE UNIVERSITÉ



- [Dieleman, 2014] : audio recommendation
  - predict item profile from audio descriptors
  - ⇒ better understanding
- [He, 2015] : online product reco.
  - Image descriptors
- [Covington, 2016] : Youtube reco
- [Nedelec, 2017] : content2vec
  - Text + image descriptors