Assignment 1

Collaborative Filtering

Benjamin Negrevergne, Alexandre Vérine

PSL University – Paris Dauphine

MILES
Machine Intelligence and Learning Systems

About this class

Educational goals

- Apply theoretical knowledge acquired during other classes & support intuition
- Provide an experience with (fairly) advanced research problems
- Practice reading scientific publications and giving oral presentations
- Go beyond the concept of traditional scholar evaluation (i.e. focus producing insights)

About this class

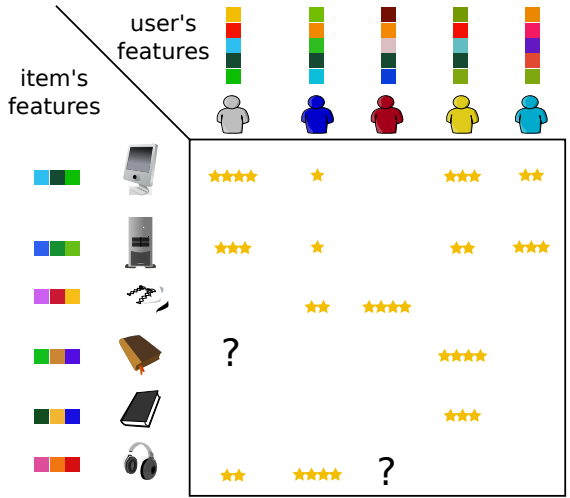
Educational goals

- Apply theoretical knowledge acquired during other classes & support intuition
- Provide an experience with (fairly) advanced research problems
- Practice reading scientific publications and giving oral presentations
- Go beyond the concept of traditional scholar evaluation (i.e. focus producing insights)

How?

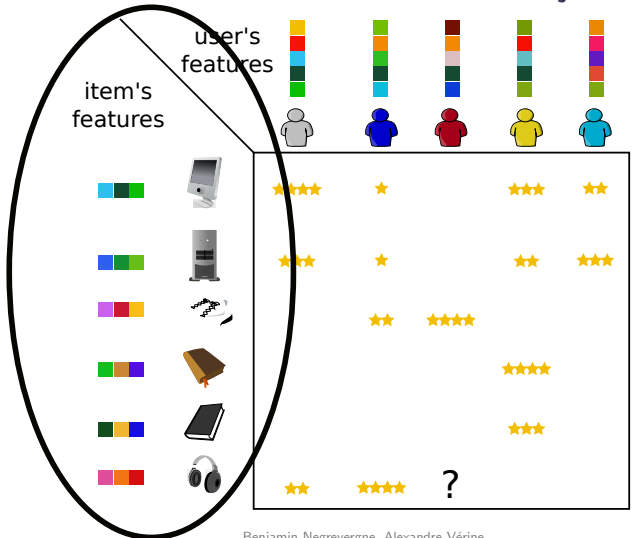
- 3 group assignments, focused on a series of selected topics
- Topics are meant to illustrate both one abstract concept and one practical application
- Several alternative approaches are presented / discussed
- Students implement one or several approach
- Share ideas & results during an oral presentations

Recommendation: general setting



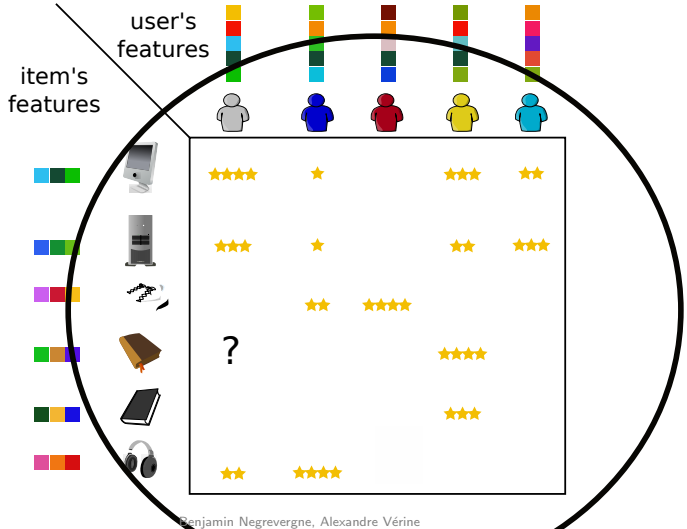
Content based filtering

Recommendations based on **intra-user or intra-object relationships**



Collaborative filtering

Recommendations based on **user-object relationship**



CF vs. CBF in recommender systems

Content-Based Filtering

- Able to deal with **cold start**
- Requires user/item features
- Disappointing performances

Collaborative filtering

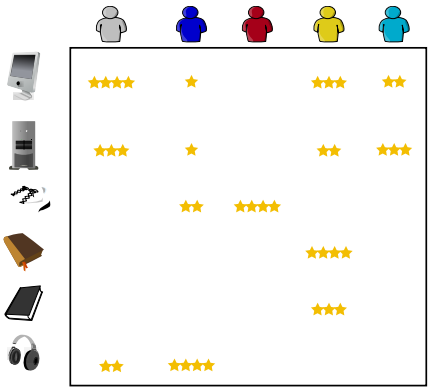
- Works with ratings only
- Good performance in practice
- Unable to deal with cold start

▶ The first assignment will focus on collaborative filtering

Outline

1. Neighborhood-based collaborative filtering
2. Model Based collaborative filtering
3. Evaluation
4. Expected work

Rating matrix



$$\rightarrow \qquad R = \begin{bmatrix} 4 & 1 & 0 & 3 & 2 \\ 3 & 1 & 0 & 2 & 3 \\ 0 & 2 & 4 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 2 & 4 & 0 & 0 & 0 \end{bmatrix}$$

Remark

- Value 0 is ambiguous: not rated or rated as zero
  particularly relevant with unitary ratings (e.g. online store)

User-based CF

Basic principle

1. compute similarity between **users** based on **preferred items**
2. predict unobserved ratings by combining grades of the **nearest users**

User-based CF

Basic principle

1. compute similarity between **users** based on **preferred items**
2. predict unobserved ratings by combining grades of the **nearest users**

Possible algorithm

How to predict unobserved rating $\hat{R}_{iu}$:

1. For all users v compute $Sim(u, v)$
2. Retain top-k nearest neighbors $v_1, \ldots, v_k$
3. set $\hat{R}_{iu} = \sum_{i=1}^{k} Sim(u, v_i) \cdot R_{iv}$

User-based CF

Basic principle

1. compute similarity between **users** based on **preferred items**
2. predict unobserved ratings by combining grades of the **nearest users**

Possible algorithm

How to predict unobserved rating $\hat{R}_{iu}$:

1. For all users v compute $Sim(u, v)$
2. Retain top-k nearest neighbors $v_1, \ldots, v_k$
3. set $\hat{R}_{iu} = \sum_{i=1}^{k} Sim(u, v_i) \cdot R_{iv}$

Example of similarity measure: *Jaccard similarity*

$$X_u = \{i : R_{iu} \geq 3\}$$

$$Sim(u, v) = Jaccard(X_u, X_v) = \frac{|X_u \cap X_v|}{|X_u \cup X_v|}$$

Pearson correlation coefficient

$$M_u = \{i : R_{iu} \text{ is observed } \}$$

- $M_u \cap M_v$ indexes of items rated by u and v

Pearson correlation coefficient

$$M_u = \{ i : R_{iu} \text{ is observed } \}$$

- $M_u \cap M_v$ indexes of items rated by u and v

$$Sim(u, v) = Pearson(u, v)$$

$$=$$

$$\frac{\sum_{k \in M_u \cap M_v} (r_{ku} - \mu_u) \cdot (r_{kv} - \mu_v)}{\sqrt{\sum_{k \in M_u \cap M_v} (r_{ku} - \mu_u)^2} \cdot \sqrt{\sum_{k \in I_u \cap I_v} (r_{kv} - \mu_v)^2}}$$

Where: $\mu_u = \frac{\sum_{k \in M_u} r_{ku}}{|M_u|}$

Computational considerations

Computational complexity for k recommendation

Performing k recommendation: $\mathcal{O}(n \cdot k)$

(assuming upper-bounded number of ratings per user)

▶ With $n = 58\ 000$ and $k = 28\ 000$,
an $\mathcal{O}(n \cdot k)$ algorithm running at 10 000 step / second takes $\approx$ 2 days to process.

Locality sensitive hashing functions

Properties:

- $h(u) = h(v) \Rightarrow sim(u, v) \geq \alpha$
- $h(u) \neq h(w) \Rightarrow sim(u, w) \leq \beta$

Recommendation with LSH

Algorithm

1. **offline**: For all users u, compute $h(u_i)$
2. **online**:
   For all users u s.t. $h(u) = h(v)$,
   Compute $Sim(v, u)$

c.f. **Mining massive datasets** (Free book)
`http://infolab.stanford.edu/~ullman/mmds/ch3.pdf`
Chapter 3: Finding similar items

Outline

1. Neighborhood-based collaborative filtering
2. Model Based collaborative filtering
3. Evaluation
4. Expected work

Matrix factorization in CF

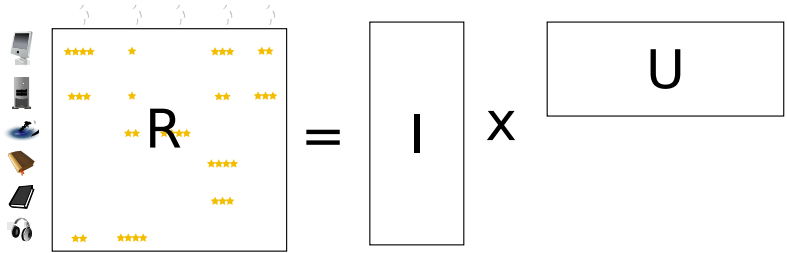
Idea: if ratings are correlated, then R can be approximated with a low rank matrix

Low rank matrix factorization

$$R \quad \approx \quad I \times U^\top$$

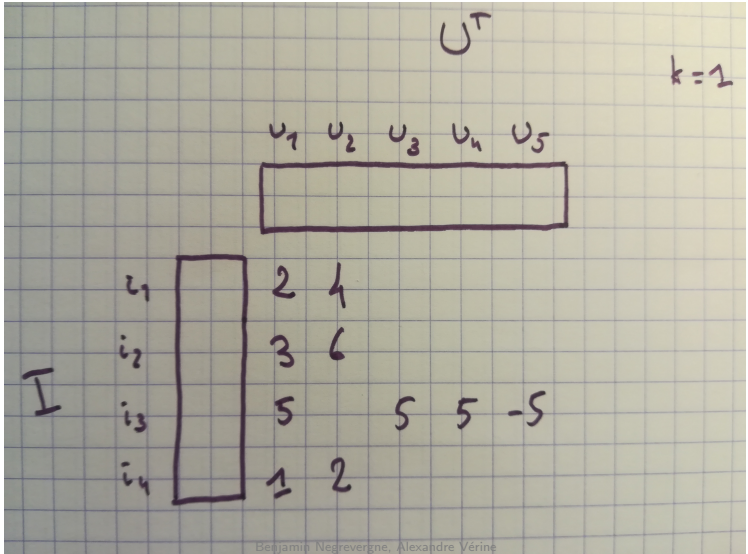
Where

- $R \in \mathbb{R}^{m \times n}$
- $I \in \mathbb{R}^{m \times k}$
- $U \in \mathbb{R}^{n \times k}$



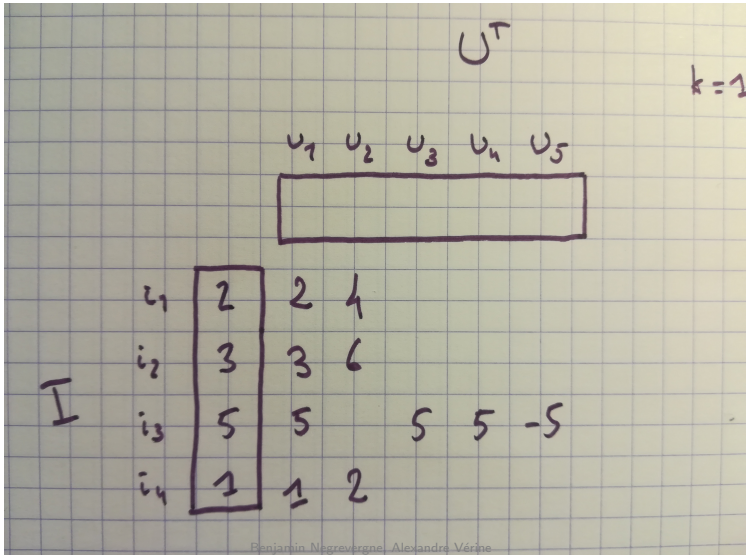
Matrix factorization demo

We search for $I \times U^\top \approx R$



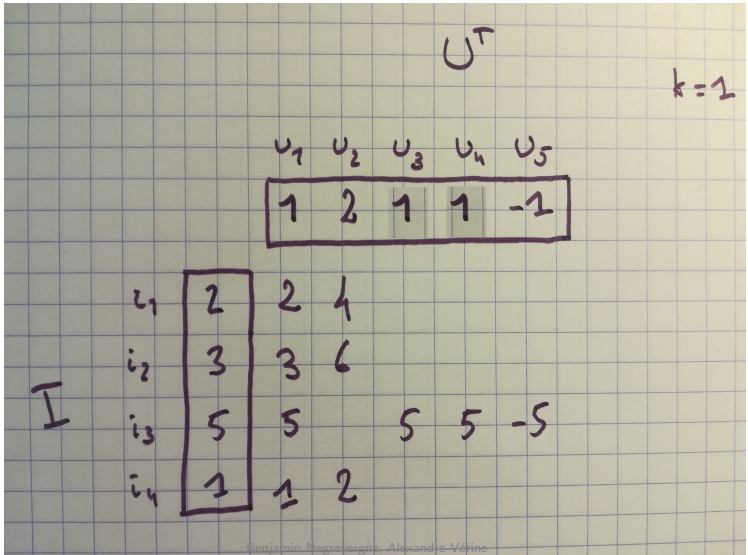
Matrix factorization demo

We search for $I \times U^\top \approx R$



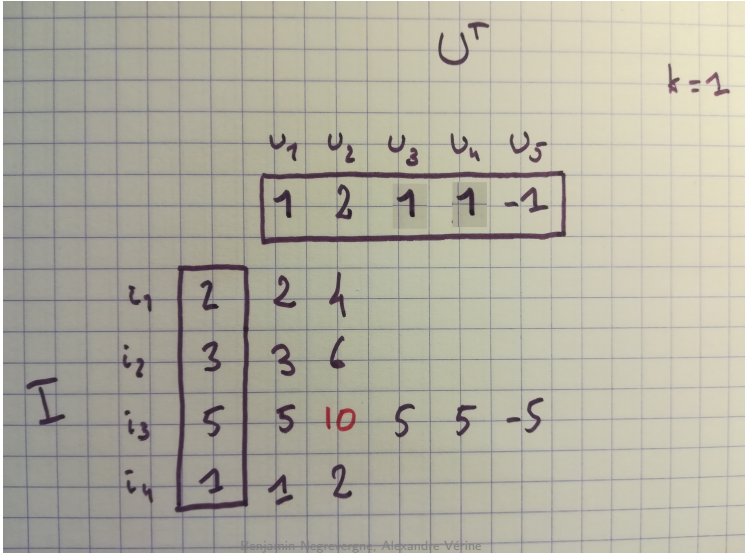
Matrix factorization demo

We search for $I \times U^\top \approx R$



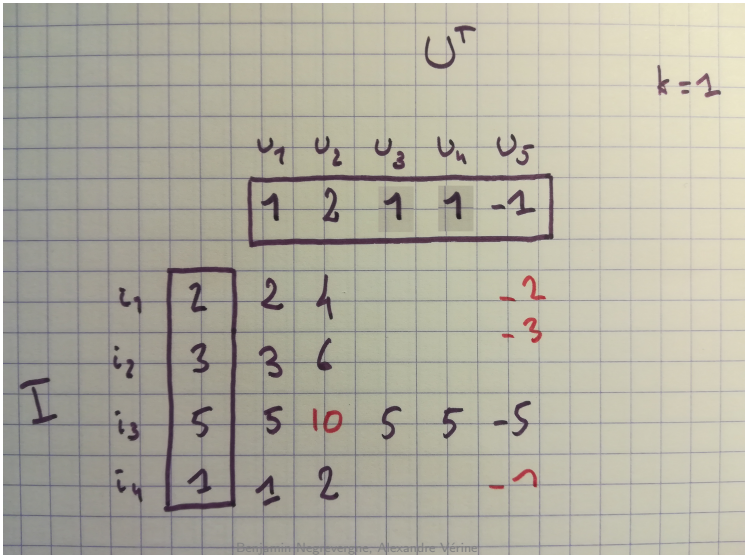
Matrix factorization demo

We search for $I \times U^\top \approx R$



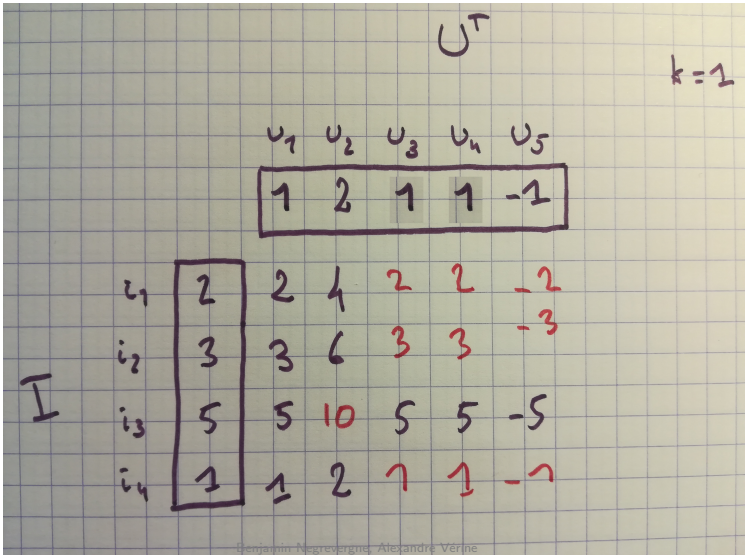
Matrix factorization demo

We search for $I \times U^\top \approx R$



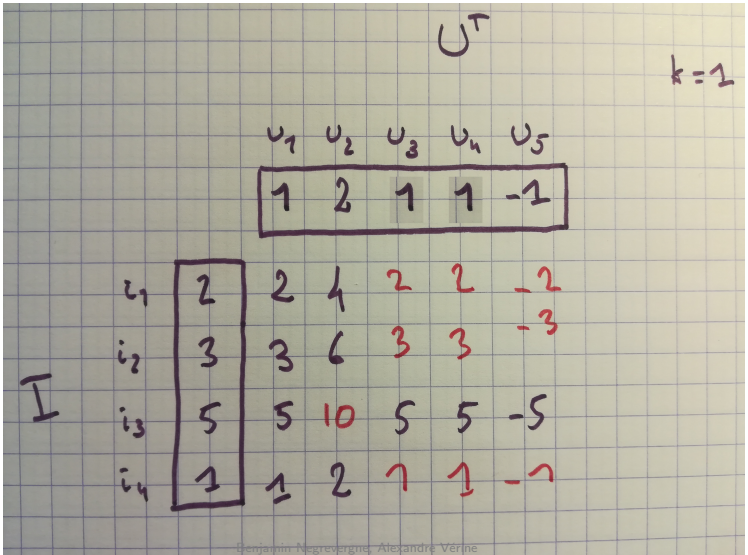
Matrix factorization demo

We search for $I \times U^\top \approx R$



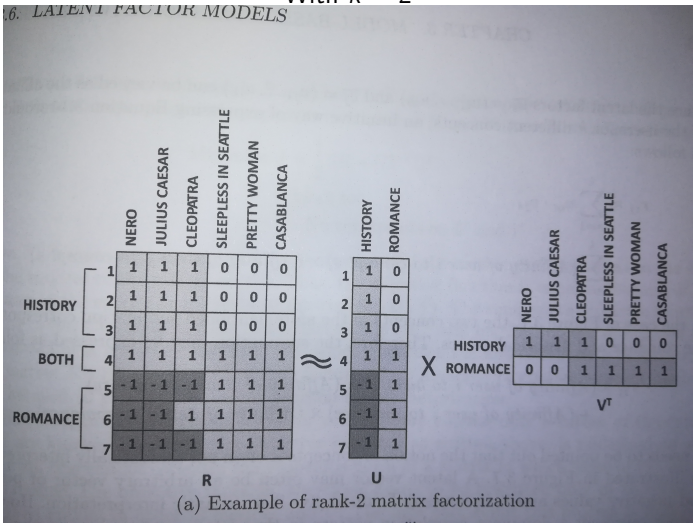
Matrix factorization demo

We search for $I \times U^\top \approx R$



Matrix factorization demo (k=2)

With $k = 2$



Machine Learning formulation

Regularized loss minimization on fixed-rank matrices
using the Frobenius matrix norm $\|\cdot\|_{\mathcal{F}}$

$$\min_{I,U} \quad \|R - IU^\top\|_{\mathcal{F}}^2$$

Where:

- $I \in \mathbb{R}^{m,k}$
- $U \in \mathbb{R}^{n,k}$
- $\|X\|_{\mathcal{F}}^2 = tr(X^\top X)$

Machine Learning formulation

Regularized loss minimization on fixed-rank matrices

using the Frobenius matrix norm $\|\cdot\|_{\mathcal{F}}$

$$\min_{I,U} \quad \|R - IU^\top\|_{\mathcal{F}}^2 \quad + \quad \lambda\|I\|_{\mathcal{F}}^2 \quad + \quad \mu\|U\|_{\mathcal{F}}^2$$

Where:

- $I \in \mathbb{R}^{m,k}$
- $U \in \mathbb{R}^{n,k}$
- $\|X\|_{\mathcal{F}}^2 = tr(X^\top X)$

Role of regularization

(same as always)

- avoid overfitting ("learning by heart")
- improve generalization ("prediction to unseen data")

Machine Learning formulation

Regularized loss minimization on fixed-rank matrices

using the Frobenius matrix norm $\|\cdot\|_{\mathcal{F}}$

$$\min_{I,U} \underbrace{\|R - IU^\top\|_{\mathcal{F}}^2 + \lambda\|I\|_{\mathcal{F}}^2 + \mu\|U\|_{\mathcal{F}}^2}_{C(I,U)}$$

Where:

- $I \in \mathbb{R}^{m,k}$
- $U \in \mathbb{R}^{n,k}$
- $\|X\|_{\mathcal{F}}^2 = tr(X^\top X)$

Role of regularization

(same as always)

- avoid overfitting ("learning by heart")
- improve generalization ("prediction to unseen data")

How to optimize

Cost function C

$$C(I, U) \quad = \quad \|R - IU^\top\|_{\mathcal{F}}^2 \quad + \quad \lambda\|I\|_{\mathcal{F}}^2 \quad + \quad \mu\|U\|_{\mathcal{F}}^2$$

Properties of C

- non-convex in U and I
- convex in U (with I fixed)
- convex in I (with U fixed)

Optimization strategy

- aim for any local minimum
- alternated minimization over U and I (while the other is fixed)

Derivatives

Cost function C

$$C(I,U) \quad = \quad \|R - IU^\top\|_\mathcal{F}^2 \quad + \quad \lambda\|I\|_\mathcal{F}^2 \quad + \quad \mu\|U\|_\mathcal{F}^2$$

$$C(I,U) = tr(R^\top R) - 2tr(R^\top IU^\top) + tr(UI^\top IU^\top) + \lambda tr(U^\top U) + \mu tr(I^\top I)$$

Partial derivatives

- $\frac{\partial C}{\partial U}(I,U) \quad = \quad -2R^\top I + 2UI^\top I + 2\mu U$
- $\frac{\partial C}{\partial I}(I,U) \quad = \quad -2RU + 2IU^\top U + 2\lambda I$

c.f. **The matrix cookbook**
`https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf`
Section 2.5 derivatives of Traces.

Algorithm

First approach: Gradient descent

at every step t,

- $I_{t+1} = I_t - \eta_t \frac{\partial C}{\partial I}(I_t, U_t)$
- $U_{t+1} = U_t - \xi_t \frac{\partial C}{\partial U}(I_t, U_t)$

Algorithm

First approach: Gradient descent

at every step t,

- $I_{t+1} = I_t - \eta_t \frac{\partial C}{\partial I}(I_t, U_t)$
- $U_{t+1} = U_t - \xi_t \frac{\partial C}{\partial U}(I_t, U_t)$

Second approach: Alternated Least-Square (ALS)

Setting the partial derivatives to zero, we have :

$$\frac{\partial C}{\partial I}(I, U) = -2RU + 2IU^\top U + 2\lambda I = 0$$
$$\frac{\partial C}{\partial U}(I, U) = -2R^\top I + 2UI^\top I + 2\mu U = 0$$

Algorithm

First approach: Gradient descent

at every step t,

- $I_{t+1} = I_t - \eta_t \frac{\partial C}{\partial I}(I_t, U_t)$
- $U_{t+1} = U_t - \xi_t \frac{\partial C}{\partial U}(I_t, U_t)$

Second approach: Alternated Least-Square (ALS)

Setting the partial derivatives to zero, we have :

$$\frac{\partial C}{\partial I}(I, U) = -2RU + 2IU^\top U + 2\lambda I = 0$$
$$\frac{\partial C}{\partial U}(I, U) = -2R^\top I + 2UI^\top I + 2\mu U = 0$$

Algorithm

First approach: Gradient descent

at every step t,

- $I_{t+1} = I_t - \eta_t \frac{\partial C}{\partial I}(I_t, U_t)$
- $U_{t+1} = U_t - \xi_t \frac{\partial C}{\partial U}(I_t, U_t)$

Second approach: Alternated Least-Square (ALS)

Setting the partial derivatives to zero, we have :

$$I = RU(U^\top U + \lambda \mathbb{I})^{-1}$$
$$U = R^\top I(I^\top I + \mu \mathbb{I})^{-1}$$

Algorithm

First approach: Gradient descent

at every step t,

- $I_{t+1} = I_t - \eta_t \frac{\partial C}{\partial I}(I_t, U_t)$
- $U_{t+1} = U_t - \xi_t \frac{\partial C}{\partial U}(I_t, U_t)$

Second approach: Alternated Least-Square (ALS)

Setting the partial derivatives to zero, we have :

$$I_{t+1} = RU_t(U_t^\top U_t + \lambda \mathbb{I})^{-1}$$
$$U_{t+1} = R^\top I_t(I_t^\top I_t + \mu \mathbb{I})^{-1}$$

Missing values

$$S = \{i, j : r_{ij} \text{ is observed}\}$$

$$\frac{\partial C}{\partial i_{iq}}(I, U) = 2 \cdot \sum_{j:(i,j)\in S} \left(r_{ij} - \sum_{s=1}^{k} i_{is} \cdot u_{js} \right) (-u_{jq}) + 2\lambda i_{iq}$$

$$\frac{\partial C}{\partial u_{jq}}(I, U) = 2 \cdot \sum_{i:(i,j)\in S} \left(r_{ij} - \sum_{s=1}^{k} i_{is} \cdot u_{js} \right) (-i_{iq}) + 2\mu u_{jq}$$

Alternative methods from linear algebra

- PCA and variants (including non-linear PCA) Generalized Principal Component Analysis by René Vidal Yi Ma and S.Shankar Sastry
  See section *PCA with Robustness to Missing Entries.*

- Lapacian Embedding - Dimension reduction with a local only approach
  ▶ `https://proceedings.neurips.cc/paper/2001/file/f106b7f99d2cb30c3db1c3cc0fde9ccb-Paper.pdf`

- Deep Matrix Factorization `https://www.ijcai.org/Proceedings/2017/0447.pdf`

Optimal transport

Monge-Kantorovich problem: coupling between two probability distributions. Let suppose you have two space X and Y and two probability measures μ and ν respectively on X and Y. Let c be a positive functions defined on $X \times Y$.
The Wasserstein metric is defined as:

$$W_c(\mu, \nu) = \min_{\pi \in \Pi(\mu,\nu)} \mathbb{E}_{(x,y)\sim\pi}(c(x, y))$$

where $\Pi(\mu, \nu)$ is the space of probability distributions on $X \times Y$ with marginals μ and ν.
You want to find the best matching π between the two distributions μ and ν.

Cost learning

You have observed matchings (user-items), you aim to learn what cost c fit the best the observed matching.
Then you use this cost c to determine which item to recommend. To do so, suppose you observed a matching $\hat{\pi}$ for a distribution of users $\hat{\mu}$ and one of items $\hat{\nu}$. Then you may solve the following inverse problem:

$$\min_{c \text{ s.t. } \pi \in \arg\min_{\pi \in \Pi(\hat{\mu},\hat{\nu})} \mathbb{E}_{(x,y)\sim\pi}(c(x,y))} KL(\hat{\pi}||\pi)$$

This project is more difficult, but a bit of investment inside will help you a lot in the future since Optimal transports is widely used in Machine Learning nowadays.
`https://arxiv.org/pdf/1802.03644.pdf`

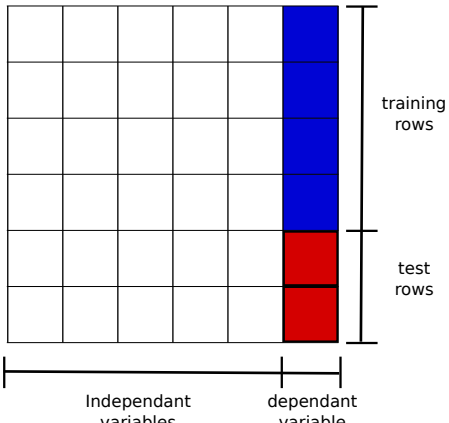
Outline

1. Neighborhood-based collaborative filtering
2. Model Based collaborative filtering
3. Evaluation
4. Expected work

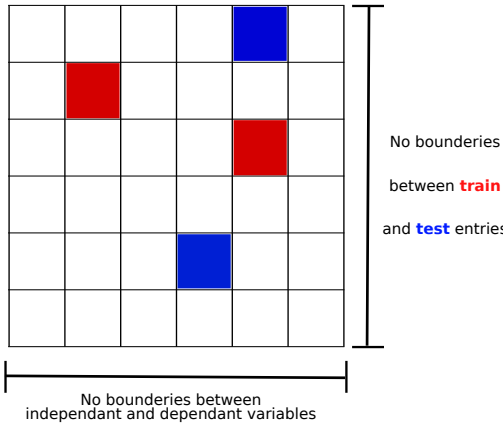
A Copy of the Movie Lens Dataset

- **ratings_train.npy**: 1600 Movies and 600 users. $\sim$ 30k ratings
- **ratings_test.npy**: 1600 Movies and 600 users. $\sim$ 30k ratings, to test your own method.
- **ratings_eval.npy**: Our dataset for the platform.
- **namesngenre.npy**: Movie names and genres to conduct an analysis on the dataset.

CF vs. the classification setting



Standard classification



Collaborative filtering

Evaluation metric

To evaluate the quality of your predictions, we will use the *Rooted Mean Squared Error* (RMSE):

$$RMSE(R, \hat{R}, T) = \sqrt{\frac{\sum_{(i,u) \in T} (R_{iu} - \hat{R}_{iu})^2}{|T|}}$$

Where:

- R in the original rating matrix (sparse)
- $\hat{R}$ is the estimated rating matrix (dense)
- $T = \{(u, i) \mid R_{iu} \text{ is in the testing set } \}$

The Evaluation Platform

You will be evaluated on a platform based on :

- RMSE
- Accuracy for exact ratings
- Time

Planning

Please check the course website at:
`https://www.lamsade.dauphine.fr/~bnegrevergne/ens/ProjetDataScience/`

Outline

1. Neighborhood-based collaborative filtering
2. Model Based collaborative filtering
3. Evaluation
4. Expected work

Expected work

- Implement at least MF with ALS or Gradient Descent
- Implement one/several alternative method of your choice
  - LSH, PCA, Optimal transport, etc.
- Conduct comparative experiments
- Write a small report
- Present your work

Source code/results

Code, slides and reports must be uploaded on github.

Classroom

How to join the classroom and get a github repository

1. Create a github account (or use an existing one)
2. Join the git classroom for assignment 1 (see course website)
3. Click on your name in the list
   Warning: Do not click on someone else's name!
4. Create a group, or join an existing group
   Warning: Do not create a team without coordinating with your partners!

Typical errors to avoid.

- Don't copy paste the lecture in your slides and report.
- Don't use screenshot of equation
- Don't use screenshot of your code.
- Go check what the PALM method is and don't mention it in your slides or your report.