CMSC5741 Big Data Tech. & Apps.

# Lecture 7: Recommender Systems / Matrix Factorization



Prof. Michael R. Lyu

Computer Science & Engineering Dept.

The Chinese University of Hong Kong

# The Netflix Problem

- Netflix database
  - About half a million users
  - About 18,000 movies
- People assign ratings to movies
- A sparse matrix



# The Netflix Problem

- Netflix database
  - Over 480,000 users
  - About 18,000 movies
  - Over 100,000,000 ratings
- People assign ratings to movies
- A sparse matrix
  - Only 1.16% of the full matrix is observed

$$\begin{bmatrix} x & & x & & x \\ & x & & & x \\ x & & x & & \\ x & & & & x \\ & & x & x & x \end{bmatrix}$$

# The Netflix Problem

- Netflix database
  - About half a million users
  - About 18,000 movies
- People assign ratings to movies
- A sparse matrix

$$\begin{bmatrix} x & & x & & x \\ & x & & & x \\ x & & x & & \\ x & & & & x \\ & & x & x & x \end{bmatrix}$$



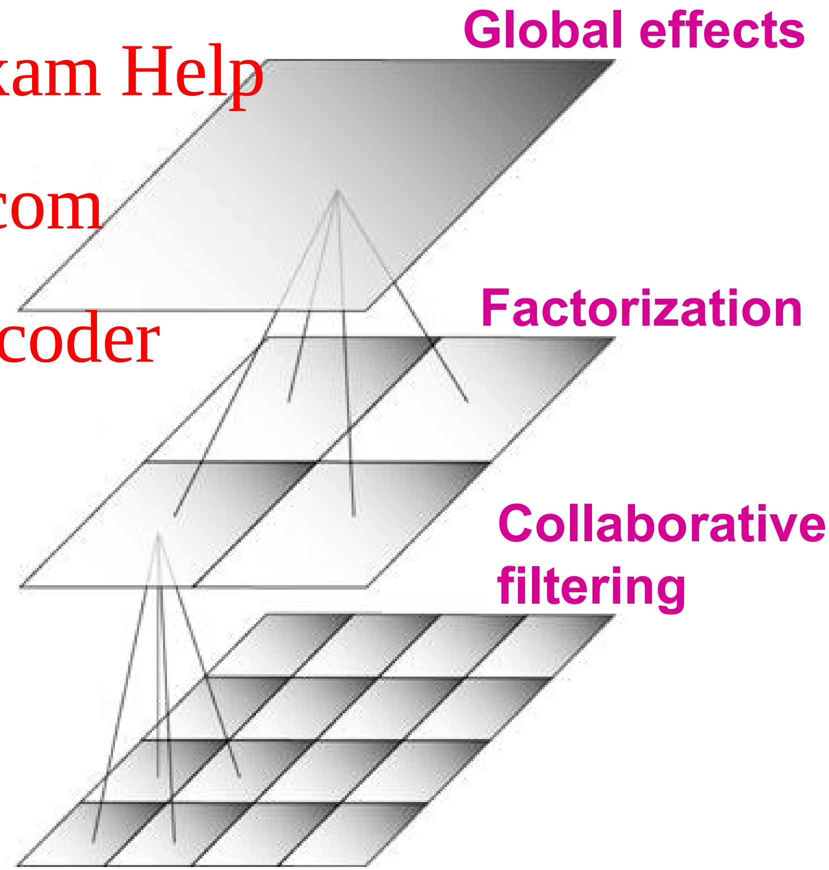
**Challenge:**
Complete the “Netflix Matrix”

Many such problems: collaborative filtering, partially filled out surveys …

# BellKor Recommender System

- **The winner of the Netflix Challenge!**
- **Multi-scale modeling of the data:**
  Combine top level, “regional” modeling of the data, with a refined, local view:
  - **Global:**
    - Overall deviations of users/movies
  - **Factorization:**
    - Addressing “regional” effects
  - **Collaborative filtering:**
    - Extract local patterns





# Modeling Local & Global Effects

- **Global:**
  - Mean movie rating: **3.7 stars**
  - *The Sixth Sense* is **0.5** stars above avg.
  - Joe rates **0.2** stars below avg.

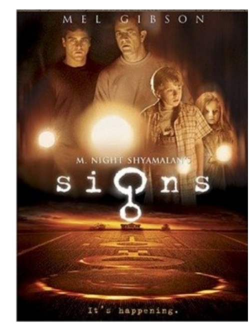
    ⇒ **Baseline estimation:**

    ***Joe* will rate *The Sixth Sense* 4 stars**




- **Local neighborhood (CF/NN):**
  - *Joe* didn't like related movie *Signs*
  - ⇒ **Final estimate:**

    ***Joe* will rate *The Sixth Sense* 3.8 stars**



# Outline

- Introduction
- LU Decomposition
- Singular Value Decomposition
- Probabilistic Matrix Factorization
- Non-negative Matrix Factorization
- Recent Development of Matrix Factorization methods in Collaborative Filtering

# Outline

- Introduction
- LU Decomposition
- Singular Value Decomposition
- Probabilistic Matrix Factorization
- Non-negative Matrix Factorization
- Recent Development of Matrix Factorization methods in Collaborative Filtering

# High Dimensional Data

| High dim. data | Graph data | Infinite data | Machine learning | Apps |
|---|---|---|---|---|
| Locality Sensitive Hashing | | | | Recommender Systems |
| Clustering | Community Detection | Web Advertising | Decision Trees | Association Rules |
| Dimensionality Reduction | Spam Detection | Queries on Streams | Perceptron, kNN | Duplicate Document Detection |

# Matrix Completion

- Matrix $X \in R^{N \times M}$
- Observe subset of entries
- Can we guess the missing entries?



$$\begin{bmatrix} x & ? & x & ? & x \\ ? & x & ? & ? & x \\ x & ? & x & ? & ? \\ x & ? & ? & ? & x \\ ? & ? & x & x & x \end{bmatrix}$$

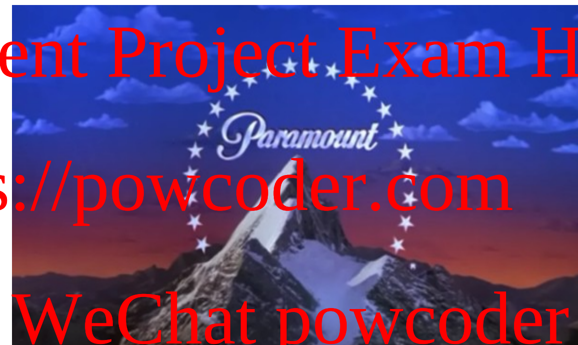
Everyone would agree this looks impossible.

# Massive High-dimensional Data

**Engineering/scientific applications:**
Unknown matrix often has (approx.) low rank.

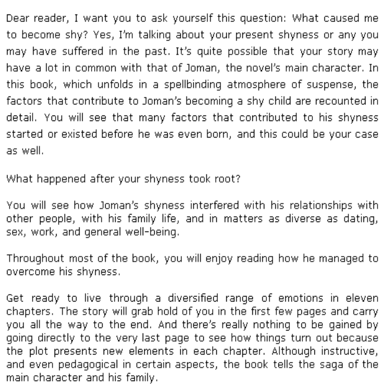


Images




Videos

**High-dimensionality but often low-dimensional structure**



Dear reader, I want you to ask yourself this question: What caused me to become shy? Yes, I'm talking about your present shyness or any you may have suffered in the past. It's quite possible that your story may have a lot in common with that of Joman, the novel's main character. In this book, which unfolds in a spellbinding atmosphere of suspense, the factors that contribute to Joman's becoming a shy child are recounted in detail. You will see that many factors that contributed to his shyness started or existed before he was even born, and this could be your case as well.

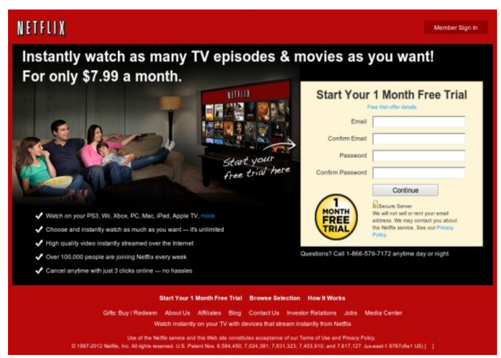
What happened after your shyness took root?

You will see how Joman's shyness interfered with his relationships with other people, with his family life, and in matters as diverse as dating, sex, work, and general well-being.

Throughout most of the book, you will enjoy reading how he managed to overcome his shyness.

Get ready to live through a diversified range of emotions in eleven chapters. The story will grab hold of you in the first few pages and carry you all the way to the end. And there's really nothing to be gained by going directly to the very last page to see how things turn out because the plot presents new elements in each chapter. Although instructive, and even pedagogical in certain aspects, the book tells the saga of the main character and his family.

Text




Web data

# Matrix Recovery Algorithm

Observation:
Try to recover a lowest complexity (rank) matrix that agrees with the observation



Recovery by minimum complexity (assuming no noise)





$$\begin{aligned} &\text{minimize} \quad \text{rank}(\hat{X}) \\ &\text{subject to} \quad \hat{X}_{ij} = X_{ij} \quad (i,j) \in \mathcal{Q}_{obs} \end{aligned}$$

- NP hard: not feasible for N > 10!
- Resort to other approaches
  - Select a low rank K, and approximate X by a rank K matrix X'

# Low Rank Factorization

- Assume X can be recovered by a rank K matrix X’
- Then X’ can be factorized into the product of

$U \in R^{K \times N}, V \in R^{K \times M}$



$$X' = U^T V$$



- Let $\mathcal{E}$ be the loss function

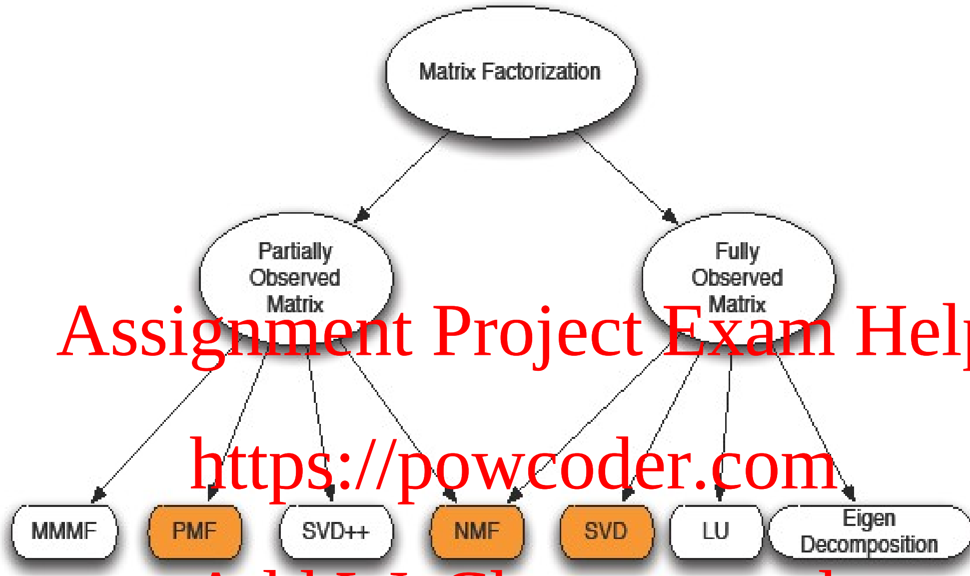


**Recovery by rank K matrix**

$$\text{minimize} \quad \sum_{i,j \in \mathcal{Q}_{obs}} \mathcal{E}(\hat{X}_{ij} - X_{ij})$$

$$\text{subject to} \quad \hat{X} = U^T V$$

# Overview of Matrix Factorization Methods






- Some methods are traditional mathematical way of factorizing a matrix.
  - SVD, LU, Eigen Decomposition
- Some methods are used to factorize partially observed matrix.
  - PMF, SVD++, MMMF
- Some methods have multiple applications.
  - NMF in image processing
  - NMF in collaborative filtering

# Outline

- Introduction
- LU Decomposition
- Singular Value Decomposition
- Probabilistic Matrix Factorization
- Non-negative Matrix Factorization
- Recent Development of Matrix Factorization methods in Collaborative Filtering

# LU Decomposition

**LU Decomposition**

The LU Decomposition factors a matrix as the product of a lower triangular matrix (L) and an upper triangular matrix (U).



$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}.$$



$$A = LU$$

Lower triangular matrix:
Every entry above the main diagonal are zero.

Upper triangular matrix:
Every entry below the main diagonal are zero.

# LU Decomposition

- LU Decomposition is useful when
  - Solving a system of linear equations
  - Inverting a matrix
  - Computing the determinant of a matrix
- LU Decomposition can be computed using a method similar to Gaussian Elimination

# LU Decomposition

- Computing LU Decomposition of a matrix A
  - Using Gaussian elimination to compute U
  - Apply inverse operation on the corresponding entry to I to get L



A ... U

$$\underbrace{\begin{bmatrix} 1 & 2 & 3 \\ 2 & -4 & 6 \\ 3 & -9 & -3 \end{bmatrix}}_{A} \xrightarrow[\text{R3 – 3R1}]{\text{R2 – 2R1}} \begin{bmatrix} 1 & 2 & 3 \\ 0 & -8 & 0 \\ 0 & -15 & -12 \end{bmatrix} \xrightarrow{\text{R2 * (-1/8)}} \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & -15 & -12 \end{bmatrix} \xrightarrow{\text{R3 + 15R2}} \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & -12 \end{bmatrix} \xrightarrow{\text{R3 * (-1/12)}} \underbrace{\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{U}$$

# LU Decomposition

- Computing LU Decomposition of a matrix A
  - Using Gaussian elimination to compute U
  - Apply inverse operation on the corresponding entry to I to get L.
    - Any row operations that involves adding a multiple of one row to another, for example, Ri + kRj, put the value –k in the ith-row, jth-column of the identity matrix.
    - Any row operations that involves getting a leading one on the main diagonal, for example, kRi, put the value 1/k in the position of the identity matrix where the leading one occurs.

# LU Decomposition

- Computing LU Decomposition of a matrix A
  - Using Gaussian elimination to compute U
  - Apply inverse operation on the corresponding entry to I to get L



$$\overset{I}{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}} \xrightarrow[\text{R3 + 3R1}]{\text{R2 + 2R1}} \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix} \xrightarrow{\text{R2 * (-8)}} \begin{bmatrix} 1 & 0 & 0 \\ 2 & -8 & 0 \\ 3 & 0 & 1 \end{bmatrix} \xrightarrow{\text{R3 - 15R2}} \begin{bmatrix} 1 & 0 & 0 \\ 2 & -8 & 0 \\ 3 & -15 & 1 \end{bmatrix} \xrightarrow{\text{R3 * (-12)}} \overset{L}{\begin{bmatrix} 1 & 0 & 0 \\ 2 & -8 & 0 \\ 3 & -15 & -12 \end{bmatrix}}$$

# LU Decomposition

- Computing LU Decomposition of a matrix A
  - Using Gaussian elimination to compute U
  - Apply inverse operation on the corresponding entry to I to get L



$$\underset{A}{\begin{bmatrix} 1 & 2 & 3 \\ 2 & -4 & 6 \\ 3 & -9 & 3 \end{bmatrix}} = \underset{L}{\begin{bmatrix} 1 & 0 & 0 \\ 2 & -8 & 0 \\ 3 & -15 & -12 \end{bmatrix}} \underset{U}{\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}$$

# In-class Practice 1

- Go to practice

# Outline

- Introduction
- LU Decomposition
- Singular Value Decomposition
- Probabilistic Matrix Factorization
- Non-negative Matrix Factorization
- Recent Development of Matrix Factorization methods in Collaborative Filtering

# Singular Value Decomposition

**Singular Value Decomposition**

The Singular Value Decomposition (SVD) of an NxM matrix A is a factorization of the form:



$$A = U\Sigma V^*$$



- $V^*$ is the conjugate transpose of $V$

- $U \in \mathbb{R}^{N\times N}$ is orthonormal matrix, i.e., $UU^* = I$
- $\Sigma \in \mathbb{R}^{N\times M}$ is rectangular diagonal matrix with positive entries
- $V^* \in \mathbb{R}^{M\times M}$ is orthonormal matrix, i.e., $VV^* = I$

# SVD v.s. Eigen Decomposition

Singular Value Decomposition

The Singular Value Decomposition (SVD) of an NxM matrix A is a factorization of the form:



$$A = U\Sigma V^*$$



- Diagonal entries of $\Sigma$ are called singular values of A.



- Columns of U and V are called left singular vectors and right singular vectors of A, respectively
- The singular values $\Sigma_{ii}s$ are arranged in descending order in $\Sigma$

# SVD v.s. Eigen Decomposition

Singular Value Decomposition

The Singular Value Decomposition (SVD) of an NxM matrix A is a factorization of the form:



$$A = U\Sigma V^*$$



- The left singular vectors of A are eigenvectors of AA*, because



$$AA^* = (U\Sigma V^*)(U\Sigma V^*)^* = U\Sigma\Sigma^T U^*$$

- The left singular vectors of A are eigenvectors of A*A, because

$$A^*A = (U\Sigma V^*)^*(U\Sigma V^*) = V\Sigma^T\Sigma V$$

- The singular values of A are the square roots of eigenvalues of both AA* and A*A.

# SVD Example

- We give an example of a simple SVD decomposition



$$\begin{bmatrix} 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 4 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & \sqrt{5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \\ 0 & 0 & 0 & 1 & 0 \\ -\sqrt{0.8} & 0 & 0 & 0 & \sqrt{0.2} \end{bmatrix}$$

$$A \qquad\qquad U \qquad\qquad \Sigma \qquad\qquad V^*$$

# SVD as Low Rank Approximation

## Low Rank Approximation

$$\operatorname{argmin}_{\tilde{A}} \quad ||A - \tilde{A}||_{Fro}$$

$$\text{s.t.} \quad \text{Rank}(\tilde{A}) = r$$



SVD gives the optimal solution.



## Solution (Eckart-Young Theorem)



Let $A = U\Sigma V^*$ be the SVD for A, and $\tilde{\Sigma}$ is the same as $\Sigma$ by keeping the largest *r* singular values. Then,

$$\tilde{A} = U\tilde{\Sigma}V^*$$

Is the solution to the above problem.

# SVD as Low Rank Approximation

**Solution (Eckart-Young Theorem)**

Let $A = U\Sigma V^*$ be the SVD for A, and $\tilde{\Sigma}$ is the same as $\Sigma$ by keeping the largest $r$ singular values. Then,

$$\tilde{A} = U\tilde{\Sigma}V^*$$

Is the solution to the above problem.

- It works when A is fully observed.
- What if A is only partially observed?

# Low Rank Approximation for Partially Observed Matrix

**Low Rank Approximation for Partially Observed Matrix**

$$\operatorname*{argmin}_{\tilde{A}} \sum_{i=1}^{N} \sum_{j=1}^{M} I_{ij} (A_{ij} - \tilde{A}_{ij})^2$$

$$\text{s.t.} \quad \text{Rank}(\tilde{A}) = r$$



- $I_{ij}$ is the indicator that equals 1 if $A_{ij}$ is observed and 0 otherwise
- We consider only the observed entries.
- A natural probabilistic extension of the above formulation is Probabilistic Matrix Factorization

# Outline

- Introduction
- LU Decomposition
- Singular Value Decomposition
- Probabilistic Matrix Factorization
- Non-negative Matrix Factorization
- Recent Development of Matrix Factorization methods in Collaborative Filtering

# Probabilistic Matrix Factorization



- A popular collaborative filtering (CF) method





- Follow the low rank matrix factorization framework

# Collaborative Filtering

Collaborative Filtering
The goal of collaborative filtering (CF) is to infer user preferences for items given a large but incomplete collection of preferences for many users.



- For example:
  - Suppose you infer from the data that most of the users who like "Star Wars" also like "Lord of the Rings" and dislike "Dune".
  - Then if a user watched and liked "Star Wars" you would recommend him/her "Lord of the Rings" but not "Dune".



- Preferences can be explicit or implicit:
  - Explicit preferences
    - Ratings assigned to items
    - Facebook "Like", Google "Plus“
  - Implicit preferences
    - Catalog browse history
    - Items rented or bought by users

# Content Based Filtering vs. Collaborative Filtering



**Content Based Filtering**

- Analyze the content of the item
- Match the item features with users preferences
- Item features are hard to extract
  - Music, Movies
- Can recommend new items

**Collaborative Filtering**

- User preferences are inferred from ratings
- Item features are inferred from ratings
- Cannot recommend new items
- Very effective with sufficient data

# CF as Matrix Completion

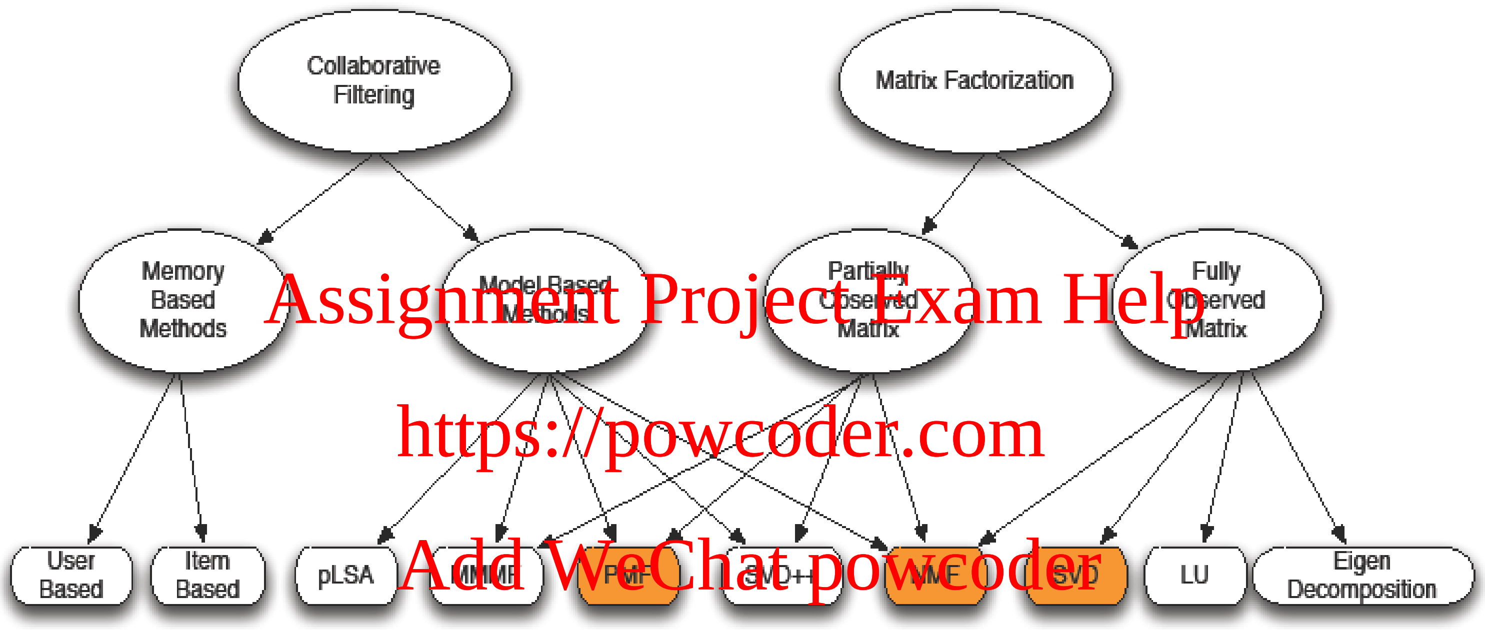
- CF can be viewed as a matrix completion problem

$$
\begin{array}{cc}
\text{Users} & \begin{bmatrix} x & & x & & x \\ & x & & & x \\ x & & x & & \\ x & & & & x \\ & & x & x & x \end{bmatrix} \\
 & \text{Items}
\end{array}
$$



- Task: given a user/item matrix with only a small subset of entries present, fill in (some of) the missing entries.
- PMF approach: low rank matrix factorization.

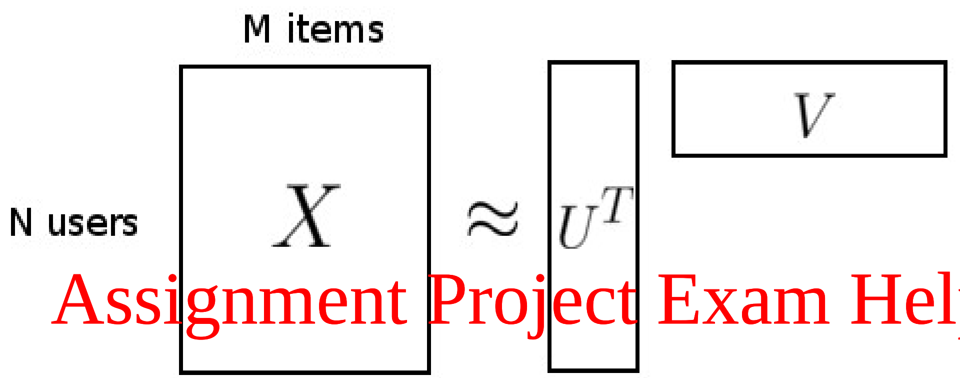
# Collaborative Filtering and Matrix Factorization






- Collaborative filtering can be formulated as a matrix factorization problem.
- Many matrix factorization methods can be used to solve collaborative filtering problem.
- The above is only a partial list.

# Notations






- Suppose we have M items, N users and integer rating values from 1 to D.
- Let *ij*th entry of X, $X_{ij}$, be the rating of user i for item j .
- $U \in R^{K \times N}$ is latent user feature matrix, $U_i$ denote the latent feature vector for user i .
- $V \in R^{K \times M}$ is latent item feature matrix, $V_j$ denote the latent feature vector for item j .

# Matrix Factorization: the Non-probabilistic View

- To predict the rating given by user i to item j,

$$\hat{R}_{ij} = U_i^T V_j = \sum_k U_{ik} V_{kj}$$



- Intuition
  - The item feature vector can be viewed as the input.
  - The user feature vector can be viewed as the weight vector.
  - The predicted rating is the output.
  - Unlike in linear regression, where inputs are given and weights are learned, we learn both the weights and the input by minimizing squared error.
  - The model is symmetric in items and users.

# Probabilistic Matrix Factorization

- PMF is a simple probabilistic linear model with Gaussian observation noise.
- Given the feature vectors for the user and the item, the distribution of the corresponding rating is:



$$P(R_{ij}|U_i, V_j, \sigma^2) = \mathcal{N}(R_{ij}|U_i^T V_j, \sigma^2)$$

$$P(R|U, V, \sigma^2) = \prod_{i=1}^{N} \prod_{j=1}^{M} \left[\mathcal{N}(R_{ij}|U_i^T V_j, \sigma^2)\right]^{I_{ij}}$$

- The user and item feature vectors adopt zero-mean spherical Gaussian priors:

$$P(U|\sigma_U^2) = \prod_{i=1}^{N} \mathcal{N}(U_i|0, \sigma_U^2 I) \qquad P(V|\sigma_V^2) = \prod_{j=1}^{M} \mathcal{N}(V_j|0, \sigma_V^2 I)$$

# Probabilistic Matrix Factorization

- Maximum A Posterior (MAP): Maximize the log-posterior over user and item features with fixed hyper-parameters.
- MAP is equivalent to minimizing the following objective function:



PMF objective function

$$\mathcal{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\|V_j\|_{Fro}^2$$

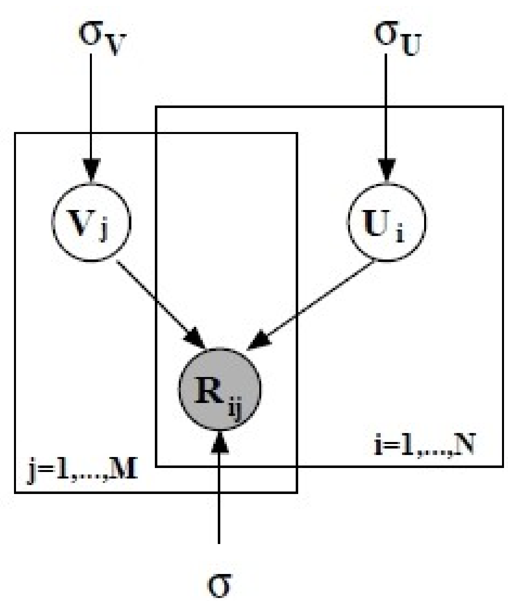
# Probabilistic Matrix Factorization

## PMF objective function

$$\mathcal{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\|V_j\|_{Fro}^2$$



- $\lambda_U = \sigma^2/\sigma_U^2, \lambda_V = \sigma^2/\sigma_V^2$ and $I_{ij}$ is indicator of whether user i rated item j
- First term is the sum-of-squared-error.
- Second and third term are quadratic regularization term to avoid over-fitting problem.



# In-class Practice 2

- Go to practice

# Probabilistic Matrix Factorization

PMF objective function

$$\mathcal{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N} \|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M} \|V_j\|_{Fro}^2$$



- If all ratings were observed, the objective reduces to the SVD objective in the limit of prior variances going to infinity.
- PMF can be viewed as a probabilistic extension of SVD.

# Probabilistic Matrix Factorization

PMF objective function

$$\mathcal{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N} \|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M} \|V_j\|_{Fro}^2$$





A trick to improve stability (the range of rating values)



- Map ratings to [0,1] by $(R_{ij} - 1)/(D - 1)$
- Pass $U_i^T V_j$ through logistic function

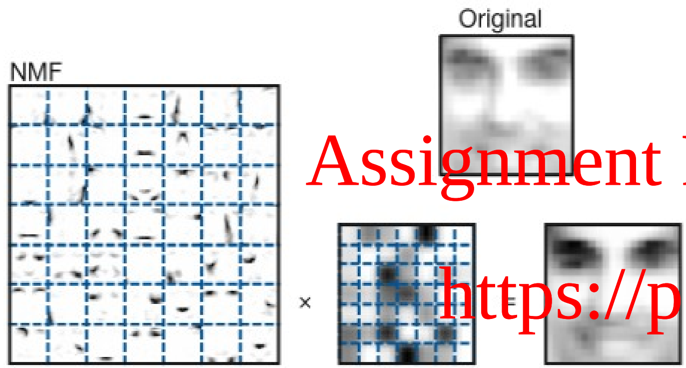
$$g(x) = \frac{1}{1 + \exp(-x)}$$

# Outline

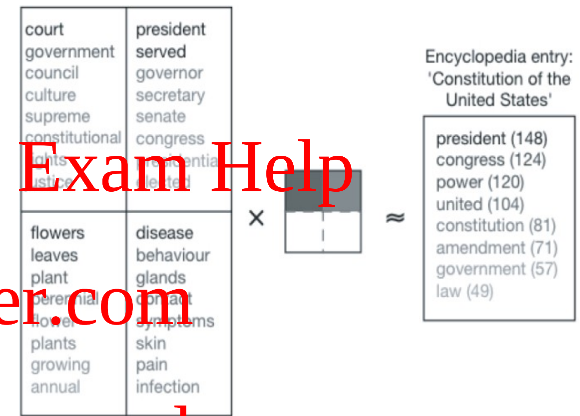
- Introduction
- LU Decomposition
- Singular Value Decomposition
- Probabilistic Matrix Factorization
- Non-negative Matrix Factorization
- Recent Development of Matrix Factorization methods in Collaborative Filtering

# Non-negative Matrix Factorization

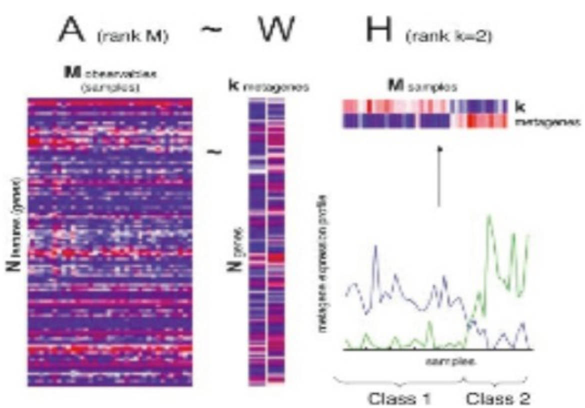
NMF is a popular method that is widely used in:



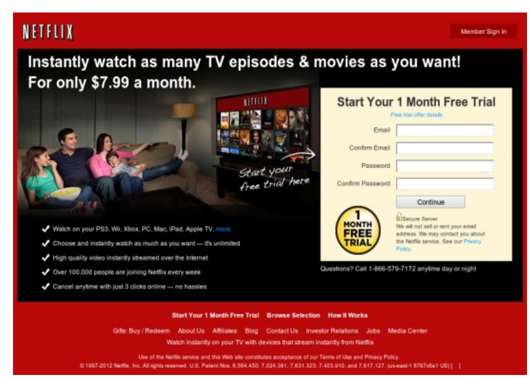

Images Mining




Text Mining




Metagenes Study




Collaborative Filtering

# Non-negative Matrix Factorization

- NMF fits in the low rank matrix factorization framework with additional non-negativity constraints.
- NMF can only factorize a Non-negative matrix $A \in \mathbb{R}^{N \times M}$ into basis matrix $W \in \mathbb{R}^{N \times K}$ and weight matrix $H \in \mathbb{R}^{K \times M}$



$$A \approx WH$$

$$\text{s.t.} \quad W, H \geq 0$$

# Interpretation with NMF

- Columns of W are the underlying basis vectors, i.e., each of the M columns of A can be built from K columns of W.
- Columns of H give the weights associated with each basis vector.

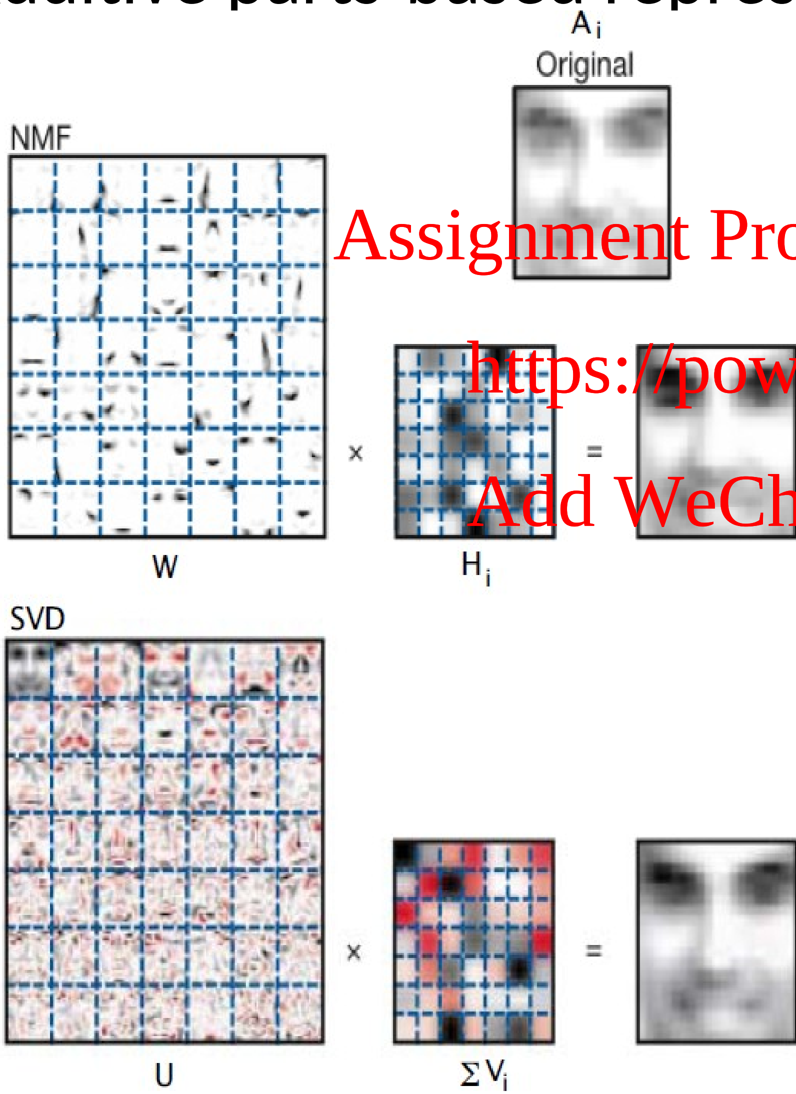


$$Ae_1 = WH_{*1} = [W_1]\,H_{11} + [W_2]\,H_{21} + \cdots + [W_K]\,H_{K1}$$

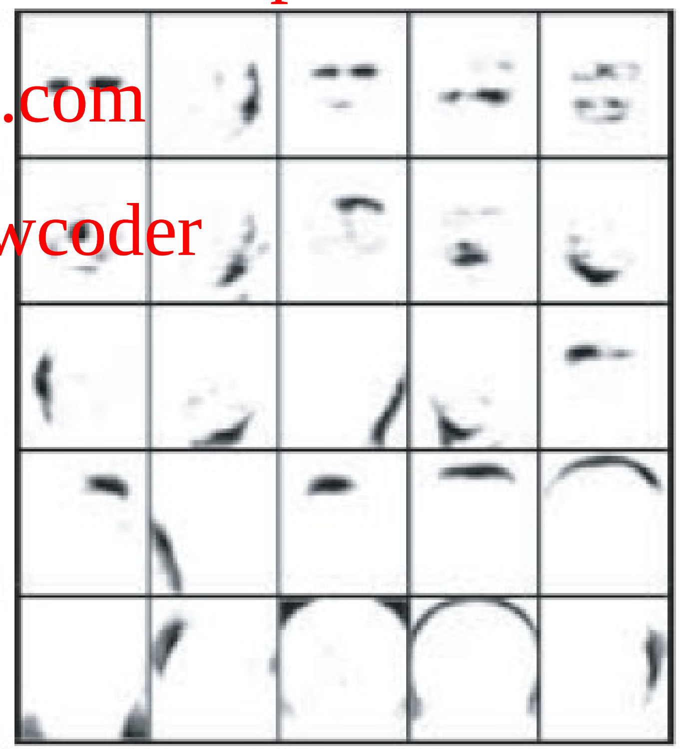
- W,H >= 0 commands additive parts-based representation.

# NMF in Image Mining

Additive parts-based representation





# NMF in Image Mining

- In image processing, we often assume Poisson Noise

NMF Poisson Noise

$$\begin{aligned} \min \quad & \sum_{i,j} \left( A_{ij} \log \frac{A_{ij}}{[WH]_{ij}} - A_{ij} + [WH]_{ij} \right) \\ \text{s.t.} \quad & W, H \geq 0 \end{aligned}$$



- Objective function can be changed to other form, the non-negative constraint is more important than the form of the objective function.

NMF Gaussian Noise

$$\begin{aligned} \min \quad & \|A - WH\|_{Fro}^2 \\ \text{s.t.} \quad & W, H \geq 0 \end{aligned}$$

# Inference of NMF

**NMF Gaussian Noise**

$$\min \quad \|A - WH\|_{Fro}^2$$
$$\text{s.t.} \quad W, H \geq 0$$



- Convex in W or H, but not both.
- Global min generally not achievable.
- Many number of unknowns: N×K for W and M×K for H (or $H^T$)

# Inference of NMF

## NMF Gaussian Noise

$$\min \quad \|A - WH\|_{Fro}^2$$

$$\text{s.t.} \quad W, H \geq 0$$



- Alternating gradient descent can get a local minima



$$F = \|A - WH\|_{Fro}^2$$



**Algorithm 1** Alternating gradient descent

$W \leftarrow \text{abs}(\text{randn}(N, K))$

$H \leftarrow \text{abs}(\text{randn}(M, K))$

**for** $i = 1 : MaxIteration$ **do**

$\quad H \leftarrow H - \eta \frac{\partial F}{\partial H}$, $H \leftarrow H. * (H \geq 0)$

$\quad W \leftarrow W - \eta \frac{\partial F}{\partial W}$, $W \leftarrow W. * (W \geq 0)$

**end for**

# Properties of NMF

- Basis vectors $W_i$ are not orthogonal
- $W_k, H_k \geq 0$ Have immediate interpretation
  - Example: large $w_{ij}$'s implies basis vector $W_i$ is mostly about terms j
  - Example: $h_{i1}$ denotes how much sample i is pointing in the "direction" of topic vector $W_1$



$$Ae_1 = WH_{*1} = [W_1]\,H_{11} + [W_2]\,H_{21} + \cdots + [W_K]\,H_{K1}$$

- NMF is algorithm-dependent: W, H not unique

# Outline

- Introduction
- LU Decomposition
- Singular Value Decomposition
- Probabilistic Matrix Factorization
- Non-negative Matrix Factorization
- Recent Development of Matrix Factorization methods in Collaborative Filtering

# Recent Development of MF methods in Collaborative Filtering

- The basic form of matrix factorization has been extended to improve prediction accuracy
  - SVD++ [Yehuda Koren 2008]
  - RLFM [Agarwal 2009]
  - Etc.

# SVD++

- SVD++ is a matrix factorization model which makes use of implicit feedback.



- In general, implicit feedback can refer to any kinds of users' history information that can help indicate users' preferences.

$$\begin{aligned}\hat{r}_{ui} = \quad & \mu + b_u + b_i \\ & + q_i^T \left( p_u + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right) \\ & + |R^k(i;u)|^{-\frac{1}{2}} \sum_{j \in R^k(i;u)} (r_{uj} - b_{uj}) w_{ij} + |N^k(i;u)|^{-\frac{1}{2}} \sum_{j \in N^k(i;u)} c_{ij}\end{aligned}$$

# 1st Tier

- The first term is the basis rate; it takes in account a global mean and the bias of both user and item.



$$\begin{aligned}\hat{r}_{ui} = \quad & \mu + b_u + b_i \\ & + q_i^T \left( p_u + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right) \\ & + |R^k(i;u)|^{-\frac{1}{2}} \sum_{j \in R^k(i;u)} (r_{uj} - b_{uj}) w_{ij} + |N^k(i;u)|^{-\frac{1}{2}} \sum_{j \in N^k(i;u)} c_{ij}\end{aligned}$$

# $2^{nd}$ Tier

- The second term is similar to the original SVD but takes in account the implicit feedback present in the set of rated items N(u)



$$
\begin{aligned}
\hat{r}_{ui} = \; & \mu + b_u + b_i \\
& + q_i^T \left( p_u + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right) \\
& + |R^k(i;u)|^{-\frac{1}{2}} \sum_{j \in R^k(i;u)} (r_{uj} - b_{uj}) w_{ij} + |N^k(i;u)|^{-\frac{1}{2}} \sum_{j \in N^k(i;u)} c_{ij}
\end{aligned}
$$

# 3rd Tier

- The third and fourth terms are the neighborhood terms. The former is the weighted bias of the basis rate and the actual rate, and the latter is the local effect of the implicit feedback



$$\begin{aligned}\hat{r}_{ui} = \quad & \mu + b_u + b_i \\ & + q_i^T \left( p_u + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right) \\ & + |R^k(i;u)|^{-\frac{1}{2}} \sum_{j \in R^k(i;u)} (r_{uj} - b_{uj}) w_{ij} + |N^k(i;u)|^{-\frac{1}{2}} \sum_{j \in N^k(i;u)} c_{ij}\end{aligned}$$

# RLFM

- Regression-based Latent Factor Model makes use of the side information that is available in many recommender systems
  - User demographic information
  - Properties of items (e.g. director, leading actor of a movie, genre of a movie)

# One-slide Takeaway

- Matrix Factorization is the key to recommender systems
- LU-decomposition
  - Decompose a matrix into a lower triangular matrix and an upper triangular matrix



- SVD decomposition



  - Decompose a matrix into , where  are orthonormal matrices and  is a diagonal matrix, whose values are called singular values



- Probabilistic Matrix Factorization
  - Factorize a partially observed matrix into the product of two low-rank matrices, usually used in recommender systems
- Non-negative Matrix Factorization
  - Factorize a matrix into the produce of two non-negative matrices, can be used to learn the "parts"

# References


[Salakhudinov Ruslan 2007] Probabilistic Matrix Factorization.

[Yehuda Koren 2008] Factorization Meets the Neighborhood: a Multifaceted Collaborative Filtering Model. KDD 2008

[Agarwal 2009] Regression-based Latent Factor Models. KDD 2009

[Chen 2011] User Reputation in a Common Rating Environment. KDD 2011

Low-rank modeling, Emmanuel Candes, http://videolectures.net/site/normal_dl/tag=623106/mlss2011_candes_lowrank_01.pdf

Matrix factorization methods for collaborative filtering, Andriy Mnih and Ruslan Salakhutdinov

The nonnegative matrix factorization, a tutorial, Barbara Ball, Atina Brooks and Amy Langville

Netflix algorithm: Prize Tribute Recommendation Algorithm in Python, Danny Tarlow

http://www.math.ust.hk/~macheng/math111/LU_Decomposition.pdf, Shiu-Yuen CHENG

# In-class Practice 1

## LU decomposition

Perform LU decomposition of the following matrix A:



$$\begin{pmatrix} 1 & -2 & 3 \\ 2 & -5 & 12 \\ 0 & 2 & -10 \end{pmatrix}$$

# In-class Practice 2

## PMF objective function

$$\mathcal{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N} \|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M} \|V_j\|_{Fro}^2$$



Write out the partial derivative of the above objective function with respect to $U_i$ and $V_j$.

We will explain how to solve the equation using the partial derivatives.