Recommendation system and matrix completion

DSA5103 Lecture 8

Yangjing Zhang

09-Mar-2023

NUS

Today's content

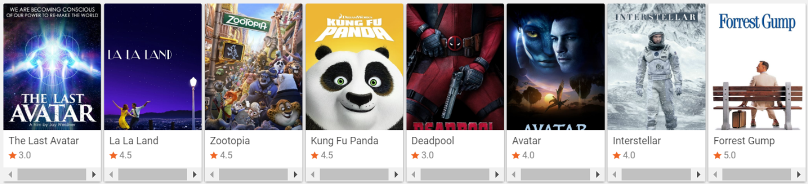
1. Recommendation systems
2. Collaborative filtering
3. Latent factor model
4. Matrix completion

Recommendation systems

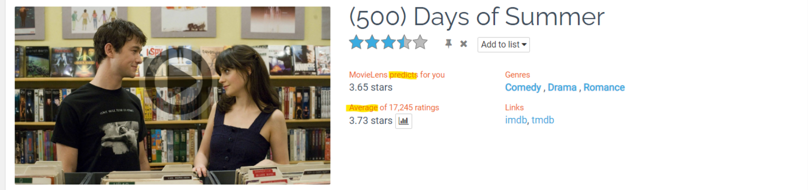
Movielens

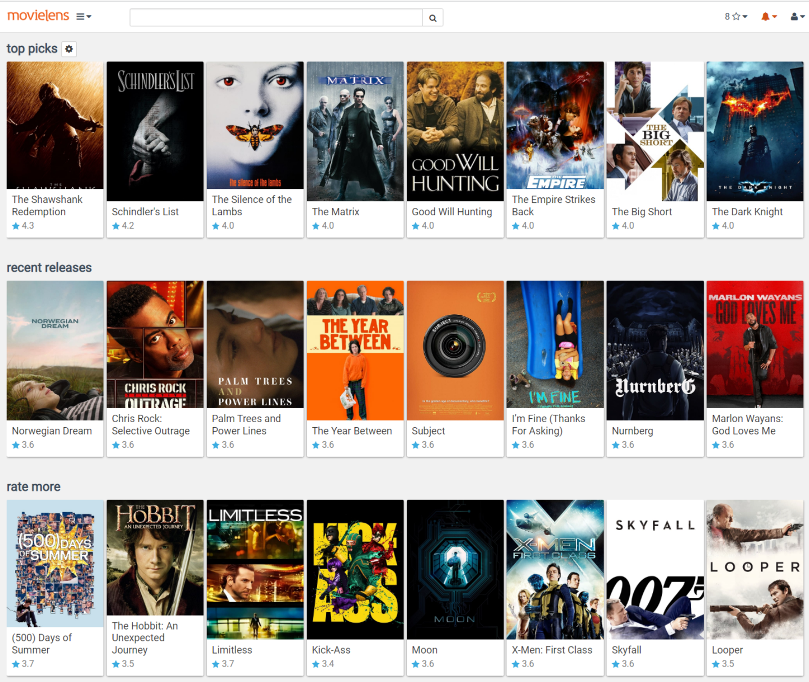
- Movielens (`https://movielens.org/`): non-commericial, personalized recommendations
- Rate movies to build your profile, then Movielens recommends other movies for you to watch

The movies I have rated:



For a particular movie, Movielens give a predicted rating:




movielens
8
top picks
The Shawshank Redemption
4.3
Schindler's List
4.2
The Silence of the Lambs
4.0
The Matrix
4.0
Good Will Hunting
4.0
The Empire Strikes Back
4.0
The Big Short
4.0
The Dark Knight
4.0
recent releases
Norwegian Dream
3.6
Chris Rock: Selective Outrage
3.6
Palm Trees and Power Lines
3.6
The Year Between
3.6
Subject
3.6
I'm Fine (Thanks For Asking)
3.6
Nurnberg
3.6
Marlon Wayans: God Loves Me
3.6
rate more
(500) Days of Summer
3.7
The Hobbit: An Unexpected Journey
3.5
Limitless
3.7
Kick-Ass
3.4
Moon
3.6
X-Men: First Class
3.6
Skyfall
3.6
Looper
3.5

Netflix Prize

Netflix provided a **training data** set of

- 100,480,507 ratings that 480,189 users gave to 17,770 movies

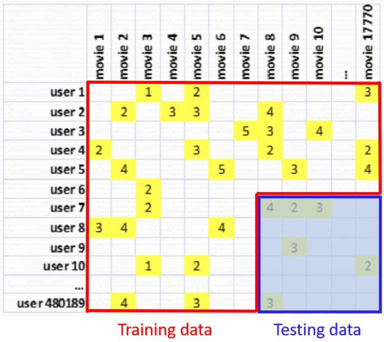
Testing data set contains

- 2,817,131 ratings only the judges know

Evaluation criterion

- root mean squared error RMSE $= \sqrt{\sum_{xi} (\underset{\text{predicted}}{r_{xi}} - \underset{\text{true}}{r^*_{xi}})^2}$

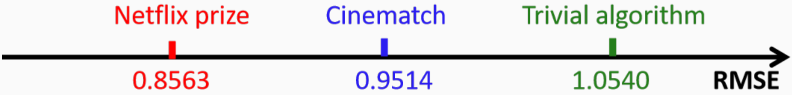
	movie 1	movie 2	movie 3	movie 4	movie 5	movie 6	movie 7	movie 8	movie 9	movie 10	...	movie 17770
user 1			1		2							3
user 2		2		3	3			4				
user 3							5	3		4		
user 4	2				3			2				2
user 5		4				5			3			4
user 6			2									
user 7			2					4	2	3		
user 8	3	4				4						
user 9									3			
user 10			1		2							2
...												
user 480189		4			3			3				



$$RMSE = \sqrt{\sum_{xi \in Test} (r_{xi} - r^*_{xi})^2}$$

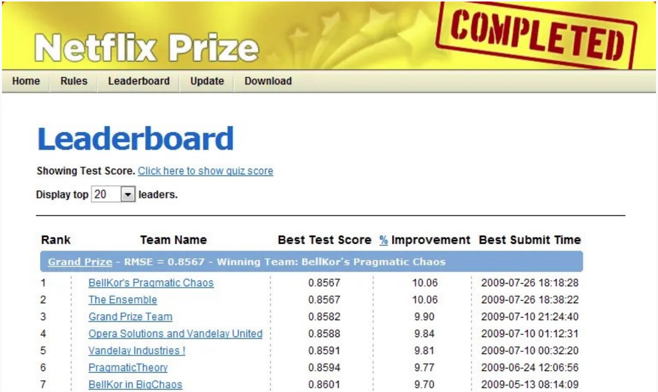
true rating of user x on item i

predicted rating



Netflix Prize

- Trivial algorithm uses average grades from the training data to predict, achieving RMSE $= 1.0540$
- Netflix's own algorithm, called *Cinematch*. It achieves RMSE $= 0.9514$, roughly a 10% improvement over the trivial algorithm
- to win the grand prize of 1,000,000, a participating team had to improve *Cinematch* by another 10%, RMSE ≤ 0.8563



Leaderboard

Showing Test Score. Click here to show quiz score

Display top 20 leaders.

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09

Utility matrix

- A recommendation system has two entities — users and items. $m =$ number of users, $n =$ number of items
- An $m \times n$ utility matrix consists of the ratings for user-item pairs; $r_{xi} =$ rating of user x of item i
- The utility matrix is usually very sparse, as most entries are unknown

Example. The following example of a utility matrix shows four users' ratings of six movies on a 1-5 scale. Blanks are where the users has not rated the movie.

	Avatar1	Avatar2	Zootopia	HarryPotter1	HarryPotter2	HarryPotter3
user1	4	1		4		
user2				5	5	4
user3	2	4	5			
user4			3		3	

In practice, the matrix would be even sparser. A user may rate only a tiny fraction of all available movies.

How to build a utility matrix

1. We can ask users to rate items.
   - Movie ratings are generally obtained this way (e.g., Movielens website), and some on-line stores (e.g., Shopee) try to obtain ratings from their purchasers
   - Generally users are unwilling to provide responses
   - The information may be biased by the fact that it comes from people willing to provide ratings
2. We can make inferences from users' behavior.
   - This sort of rating system is usually binary
   - The rating is 1 if a user buys a product/watches a movie/searches an item
   - The rating is 0 where the user has not purchased or viewed the item

Goal of a recommendation system

- The general goal of a recommendation system is to predict the blanks in the utility matrix
- But in many applications, it is only necessary to find some items in each row that are likely to be high
  - ▷ Not interested in knowing what the user does not like
  - ▷ Want to discover items that the user may like
- Applications:
  - ▷ Youtube recommends videos for users
  - ▷ Shopee recommends products for buyers
- Two approaches will be introduced today
  1. Collaborative filtering (CF): user-user CF, item-item CF
  2. Latent factor model

Collaborative filtering

User-user CF

Collaborative filtering would be the earliest and most popular method for recommendation systems. User-user CF includes two steps:

Step 1. For user x, identify the set N of other users similar to x; users are similar if their vectors of ratings are close according to some distance measures.

Example. User1 and user2 rated two movies in common, but their preferences of the two are opposite. We would expect that their distance is large under a reasonable distance measure.

	Avatar1	Avatar2	Zootopia	HarryPotter1	HarryPotter2	HarryPotter3
user1	**4**	**1**		**4**		
user2				5	5	4
user3	**2**	**4**	**5**			
user4			3		3	

User-user CF

Collaborative filtering would be the earliest and most popular method for recommendation systems. User-user CF includes two steps:

Step 1. For user x, identify the set $\mathcal{N}$ of other users similar to x; users are similar if their vectors of ratings are close according to some distance measures.

Step 2. Recommend what similar users like; estimate the ratings of a user x by looking at the users in $\mathcal{N}$.

User-user CF

Collaborative filtering would be the earliest and most popular method for recommendation systems. User-user CF includes two steps:

Step 1. For user x, identify the set $\mathcal{N}$ of other users similar to x; users are similar if their vectors of ratings are close according to some distance measures.

Step 2. Recommend what similar users like; estimate the ratings of a user x by looking at the users in $\mathcal{N}$.

Next, we give three choices of distance measures for step 1:

- Jaccard similarity
- Cosine similarity
- Normalized cosine similarity

Measure 1: Jaccard similarity

The Jaccard distance measure is suitable when the utility matrix is binary. For two sets

$$S_x = \text{items rated by user } x, \quad S_y = \text{items rated by user } y$$

the Jaccard similarity is

$$\mathrm{Sim}(x, y) = J(S_x, S_y) = \frac{|S_x \cap S_y|}{|S_x \cup S_y|} = \frac{\text{size of intersection}}{\text{size of union}} \in [0, 1]$$

- If $S_x \cap S_y = \emptyset$, then $J(S_x, S_y)$=0
- If $S_x = S_y$, then $J(S_x, S_y)$=1

Example

	product1	product2	product3	product4	product5
user1	1	1		1	
user2				1	1
user3	1	1		1	1
user4			1		1

Example

	product1	product2	product3	product4	product5
user1	1	1		1	
user2				1	1
user3	1	1		1	1
user4			1		1

User1 bought $S_1 = \{$product1,product2,product4$\}$

User2 bought $S_2 = \{$product4,product5$\}$

Their Jaccard similarity is $\frac{|\{\text{product4}\}|}{|\{\text{product1,product2,product4,product5}\}|} = \frac{1}{4}$ they are not that similar.

In contrast, user1 and user3 have a Jaccard similarity of $3/4$; they are very similar.

The Jaccard similarity of user1 and user4 is 0.

Measure 2: Cosine similarity

We treat blanks in the utility matrix as 0 and compute the cosine of two users' vectors of ratings r_x, r_y:

$$\mathrm{Sim}(x, y) = \mathrm{Cos}(r_x, r_y) = \frac{r_x^T r_y}{\|r_x\| \|r_y\|} \in [0, 1]$$

	Avatar1	Avatar2	Zootopia	HarryPotter1	HarryPotter2	HarryPotter3
user1	**4**	**1**		**4**		
user2				5	5	4
user3	**2**	**4**	**5**			
user4			3		3	

Measure 2: Cosine similarity

We treat blanks in the utility matrix as 0 and compute the cosine of two users' vectors of ratings r_x, r_y:

$$\mathrm{Sim}(x,y) = \mathrm{Cos}(r_x, r_y) = \frac{r_x^T r_y}{\|r_x\| \|r_y\|} \in [0,1]$$

	Avatar1	Avatar2	Zootopia	HarryPotter1	HarryPotter2	HarryPotter3
user1	**4**	**1**		**4**		
user2				5	5	4
user3	**2**	**4**	**5**			
user4			3		3	

Example. The cosine similarity between user1 and user3 is

$$\frac{4 \times 2 + 1 \times 4}{\sqrt{4^2+1^2+4^2}\sqrt{2^2+4^2+5^2}} = 0.3114,$$

as $r_1 = (4,1,0,4,0,0)^T$, $r_3 = (2,4,5,0,0,0)^T$.

Measure 3: Normalized cosine similarity

- $\bar{r}_x$ = average rating of user x
- We subtract each rating of user x by the average: $r_{xi} - \bar{r}_x$
- We treat blanks in the utility matrix as 0 and compute the cosine of two users' normalized vectors of ratings

$$\mathrm{Sim}(x, y) = \mathrm{Cos}(r_x - \bar{r}_x, r_y - \bar{r}_y) \in [-1, 1]$$

Example. Compute the normalized cosine similarity between user1 and user3.

	Avatar1	Avatar2	Zootopia	HarryPotter1	HarryPotter2	HarryPotter3
user1	4	1		4		
user2				5	5	4
user3	2	4	5			
user4			3		3	

Measure 3: Normalized cosine similarity

Example. We first subtract the row mean

$$\bar{r}_1 = (4+1+4)/3 = 3,\ \bar{r}_2 = \frac{14}{3},\ \bar{r}_3 = \frac{11}{3},\ \bar{r}_4 = 3$$

	Avatar1	Avatar2	Zootopia	HarryPotter1	HarryPotter2	HarryPotter3
user1	4 - 3	1 - 3		4 - 3		
user2				5 - $\frac{14}{3}$	5 - $\frac{14}{3}$	4 - $\frac{14}{3}$
user3	2 - $\frac{11}{3}$	4 - $\frac{11}{3}$	5 - $\frac{11}{3}$			
user4			3 - 3		3 - 3	

Measure 3: Normalized cosine similarity

Example. We first subtract the row mean

$$\bar{r}_1 = (4+1+4)/3 = 3,\ \bar{r}_2 = \frac{14}{3},\ \bar{r}_3 = \frac{11}{3},\ \bar{r}_4 = 3$$

	Avatar1	Avatar2	Zootopia	HarryPotter1	HarryPotter2	HarryPotter3
user1	4 - 3	1 - 3		4 - 3		
user2				5 - $\frac{14}{3}$	5 - $\frac{14}{3}$	4 - $\frac{14}{3}$
user3	2 - $\frac{11}{3}$	4 - $\frac{11}{3}$	5 - $\frac{11}{3}$			
user4			3 - 3		3 - 3	

- The normalized cosine similarity between user1 and user3 is

$$\frac{(4-3)\times(2-\frac{11}{3})+(1-3)\times(4-\frac{11}{3})}{\sqrt{(4-3)^2+(1-3)^2+(4-3)^2}\sqrt{(2-\frac{11}{3})^2+(4-\frac{11}{3})^2+(5-\frac{11}{3})^2}}$$

$$= -\,0.4410$$

User-user CF

Step 1. Several methods can be used to determine the neighbourhood $\mathcal{N}$:

1. Select users with a similarity score (Jaccard/cosine/normalized cosine similarity) above a certain threshold, for example, normalized cosine similarity ≥ 0
2. Select the top-N users ranked by similarity score
3. Do classification for users first and then select users within the same cluster
4. Other options...

Step 2. Predicted rating of user x of item $i =$ "averaged" ratings of users $\in \mathcal{N}$ of item i

1. $r_{xi} = \frac{1}{|\mathcal{N}|} \sum_{y \in \mathcal{N}} r_{yi}$ Naive average
2. $r_{xi} = \frac{\sum_{y \in \mathcal{N}} \text{Sim}(x, y) r_{yi}}{\sum_{y \in \mathcal{N}} \text{Sim}(x, y)}$ Weighted average
3. Other options...

Example

Example. Consider a matrix that shows four users rating 1-5 on five items. Predict Alice's ratings for item1 and item5 via user-user collaborative filtering. In particular, we select top-2 users ranked by cosine similarity, and use weighted average for prediction.

	item1	item2	item3	item4	item5
Alice		4	1	4	
Bob	2	1	5		
Carol		3	4	3	5
David	1	3	1	5	4

Example

	item1	item2	item3	item4	item5
Alice	?	4	1	4	?
Bob	2	1	5		
Carol		3	4	3	5
David	1	3	1	5	4

Step 1. Compute the cosine similarity between Alice and other users

$$\mathrm{Sim}(\mathrm{Alice}, \mathrm{Bob}) = \frac{4 \times 1 + 1 \times 5}{\sqrt{4^2 + 1^2 + 4^2}\sqrt{2^2 + 1^2 + 5^2}} = 0.2860,$$

$$\mathrm{Sim}(\mathrm{Alice}, \mathrm{Carol}) = 0.6346, \quad \mathrm{Sim}(\mathrm{Alice}, \mathrm{David}) = 0.7966$$

and then we identify two similar users $\mathcal{N} = \{\mathrm{Carol}, \mathrm{David}\}$

Step 2. Predict

$$r_{\mathrm{Alice,item1}} = 1$$

$$r_{\mathrm{Alice,item5}} = \frac{0.6346 \times 5 + 0.7966 \times 4}{0.6346 + 0.7966} = 4.4434$$

Example

Example. Predict Alice's ratings for item1 and item5 via user-user collaborative filtering. This time we select users with positive normalized cosine similarity, and use weighted average for prediction.

	item1	item2	item3	item4	item5
Alice		4	1	4	
Bob	2	1	5		
Carol		3	4	3	5
David	1	3	1	5	4

Normalize rows: $\bar{r}_A = 3, \bar{r}_B = 8/3, \bar{r}_C = 15/4, \bar{r}_D = 14/5$

	item1	item2	item3	item4	item5
Alice		1	-2	1	
Bob	-2/3	-5/3	7/3		
Carol		-3/4	1/4	-3/4	5/4
David	-9/5	1/5	-9/5	11/5	6/5

Example

	item1	item2	item3	item4	item5
Alice		1	-2	1	
Bob	-2/3	-5/3	7/3		
Carol		-3/4	1/4	-3/4	5/4
David	-9/5	1/5	-9/5	11/5	6/5

Step 1. Compute the normalized cosine similarity between Alice and other users

$$\mathrm{Sim}(\mathrm{Alice}, \mathrm{Bob}) < 0 (= -0.8783),$$
$$\mathrm{Sim}(\mathrm{Alice}, \mathrm{Carol}) < 0 (= -0.4924),$$
$$\mathrm{Sim}(\mathrm{Alice}, \mathrm{David}) = 0.6847$$

and $\mathcal{N} = \{\mathrm{David}\}$ since its similarity to Alice is positive.

Example

	item1	item2	item3	item4	item5
Alice		4	1	4	
Bob	2	1	5		
Carol		3	4	3	5
David	1	3	1	5	4

Step 1. Compute the normalized cosine similarity between Alice and other users

$$\mathrm{Sim}(\mathrm{Alice}, \mathrm{Bob}) < 0 (= -0.8783),$$
$$\mathrm{Sim}(\mathrm{Alice}, \mathrm{Carol}) < 0 (= -0.4924),$$
$$\mathrm{Sim}(\mathrm{Alice}, \mathrm{David}) = 0.6847$$

and $\mathcal{N} = \{\mathrm{David}\}$ since its similarity to Alice is positive.

Step 2. Predict

$$r_{\mathrm{Alice,item1}} = 1, \quad r_{\mathrm{Alice,item5}} = 4$$

Item-item CF

Instead of user-user CF, we can alternatively use item-item CF

Step 1. For item i, find other similar items

Step 2. Estimate rating for item i based on ratings for similar items

$$r_{xi} = \frac{\sum_{j \in \mathcal{N}(i,x)} \mathrm{Sim}(i,j) r_{xj}}{\sum_{j \in \mathcal{N}(i,x)} \mathrm{Sim}(i,j)}$$

where $\mathrm{Sim}(i,j) =$ similarity between items i and j, $\mathcal{N}(i,x) =$ the set of items rated by user x similar to i

$$r_{xi} = \frac{\sum_{y \in \mathcal{N}} \mathrm{Sim}(x,y) r_{yi}}{\sum_{y \in \mathcal{N}} \mathrm{Sim}(x,y)} \quad \text{v.s.} \quad r_{xi} = \frac{\sum_{j \in \mathcal{N}(i,x)} \mathrm{Sim}(i,j) r_{xj}}{\sum_{j \in \mathcal{N}(i,x)} \mathrm{Sim}(i,j)}$$

Example

Example. Predict Alice and Carol's ratings for item1 via item-item collaborative filtering. We select items with positive normalized cosine similarity, and use weighted average for prediction.

	item1	item2	item3	item4	item5
Alice		4	1	4	
Bob	2	1	5		
Carol		3	4	3	5
David	1	3	1	5	4

Normalize columns: $\bar{r}_1 = 3/2, \bar{r}_2 = 11/4, \bar{r}_3 = 11/4, \bar{r}_4 = 4, \bar{r}_5 = 9/2$

	item1	item2	item3	item4	item5
Alice		5/4	-7/4	0	
Bob	1/2	-7/4	9/4		
Carol		1/4	5/4	-1	1/2
David	-1/2	1/4	-7/4	1	-1/2

Example

	item1	item2	item3	item4	item5
Alice		5/4	-7/4	0	
Bob	1/2	-7/4	9/4		
Carol		1/4	5/4	-1	1/2
David	-1/2	1/4	-7/4	1	-1/2

Step 1. Compute the normalized cosine similarity between item1 and other items

$$\mathrm{Sim}(1,2) < 0, \quad \mathrm{Sim}(1,4) < 0,$$
$$\mathrm{Sim}(1,3) = 0.7921, \quad \mathrm{Sim}(1,5) = 0.5$$

and $\mathcal{N}(\mathrm{item1}, \mathrm{Alice}) = \{\mathrm{item3}\}$, $\mathcal{N}(\mathrm{item1}, \mathrm{Carol}) = \{\mathrm{item3}, \mathrm{item5}\}$

Example

	item1	item2	item3	item4	item5
Alice		4	1	4	
Bob	2	1	5		
Carol		3	4	3	5
David	1	3	1	5	4

Step 1. Compute the normalized cosine similarity between item1 and other items

$$\mathrm{Sim}(1,2) < 0,\ \mathrm{Sim}(1,4) < 0,$$
$$\mathrm{Sim}(1,3) = 0.7921,\ \mathrm{Sim}(1,5) = 0.5$$

and $\mathcal{N}(\mathrm{item1}, \mathrm{Alice}) = \{\mathrm{item3}\}$, $\mathcal{N}(\mathrm{item1}, \mathrm{Carol}) = \{\mathrm{item3}, \mathrm{item5}\}$

Step 2. Predict

$$r_{\mathrm{Alice,item1}} = 1$$
$$r_{\mathrm{Carol,item1}} = \frac{0.7921 \times 4 + 0.5 \times 5}{0.7921 + 0.5} = 4.3870$$

Latent factor model

Latent factor model

- Given a utility matrix $R \in \mathbb{R}^{m \times n}$. Define the index set

$$\Omega = \{(i,j) \mid \text{ rating of user } i \text{ of item } j \text{ is known}\}$$

- Latent factor model assumes that R has a low rank approximation

$$R \approx WH, \quad W\mathbb{R}^{m \times k}, \quad H \in \mathbb{R}^{k \times n}$$

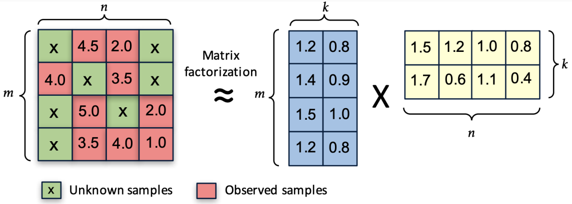


Figure 1: Image from internet

- Predict unknown rating $R_{ij} = W_i.H_{.j}$. For example, rating of user 1 of item 1 is predicted to be $1.2 \times 1.5 + 0.8 \times 1.7 = 3.16$; rating of user 1 of item 4 is predicted to be $1.2 \times 0.8 + 0.8 \times 0.4 = 1.34$

Latent factor model

- Latent factor models attempt to explain the ratings by characterizing both items and users on a number of factors k. For movie recommendation, we can associate these factors with idealized concepts like
  - drama / action / kid / others...
- The error should be small on known ratings, i.e., we minimize

$$\sum_{(i,j)\in\Omega} (R - WH)^2_{ij} = \sum_{(i,j)\in\Omega} \left(R_{ij} - W_{i\cdot} H_{\cdot j}\right)^2$$

  We do not impose any constraints on W and H
- Add regularization:

$$\min_{W,H} \quad \frac{1}{2} \sum_{(i,j)\in\Omega} \left(R_{ij} - W_{i\cdot} H_{\cdot j}\right)^2 + \frac{\lambda}{2} \left(\|W\|_F^2 + \|H\|_F^2\right)$$

- Optimization: gradient descent, alternating minimization [2] (BCD with W being a block, H being a block)

Matrix completion

Rank

For $X \in \mathbb{R}^{m \times n}$, the rank of X is

- the dimension of column space of X
- the dimension of row space of X
- the smallest k such that X can be factorized as

$$X = WH,\, W \in \mathbb{R}^{m \times k},\, H \in \mathbb{R}^{k \times n}$$

- the number of non-zero singular values

A utility matrix (without blanks) is usually assumed to be low-rank

- similar users may have similar ratings, i.e., the rows are likely to be similar (dependent)
- similar items may obtain similar ratings, i.e., the columns are likely to be similar (dependent)
- ratings are inherently determined by only a few factors

Singular value decomposition

The singular value decomposition (SVD) of an $m \times n$ real matrix X is a factorization of the form

$$X = U \Sigma V^T$$

- U is an $m \times m$ real orthogonal matrix ($U^T U = I$), the columns of U (denoted as u_i) are the left singular vectors of X
- V is an $n \times n$ real orthogonal matrix ($V^T V = I$), the columns of V (denoted as v_i) are the right singular vectors of X
- Σ is an $m \times n$ rectangular diagonal matrix
  - ▷ The diagonal entries $\sigma_i = \Sigma_{ii}$ are known as the singular values of X. The number of non-zero singular values is equal to the rank of X (let the rank be k)
  - ▷ Convention: $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k \geq 0$. $\sigma_i = \sigma_i(X)$ denotes the i-th largest singular value of X

The singular value decomposition can be written as $X = \sum_{i=1}^{k} \sigma_i u_i v_i^T$

Singular value decomposition

Note that the singular values are nonnegative. And by convention, we sort the singular values in descending order.

Example. The SVD of $X = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ is $X = I_4 X I_3$.

SVD of $X = \begin{bmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$: $X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} I_3$.

SVD of $X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$: $X = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

Nuclear norm

- The nuclear norm of $X \in \mathbb{R}^{m \times n}$ is defined as the sum of its singular values

$$\|X\|_* = \sum_{i=1}^{\min(m,n)} \sigma_i(X)$$

It is indeed a norm as we can show

- $\|X\|_* \geq 0$ (as singular values are nonnegative)
- $\|X\|_* = 0 \iff X = 0$
- $\|\lambda X\|_* = |\lambda| \|X\|_*$, $\lambda \in \mathbb{R}$
- $\|X + Y\|_* \leq \|X\|_* + \|Y\|_*$

The proof of the triangle inequality is nontrivial. The proof is based on

$$\|X\|_* = \max_{\sigma_1(Z) \leq 1} \langle Z, X \rangle$$

$\|X + Y\|_* = \max_{\sigma_1(Z) \leq 1} \langle Z, X + Y \rangle \leq \max_{\sigma_1(Z) \leq 1} \langle Z, X \rangle + \max_{\sigma_1(Z) \leq 1} \langle Z, Y \rangle =$
$\|X\|_* + \|Y\|_*$

Rank and nuclear norm of a matrix

- Rank function $\mathrm{rank} : \mathbb{R}^{m\times n} \to \mathbb{R}$ is non-convex, since

$$\text{for } A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix},\ B = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix},\ 0 < \lambda < 1$$

$$\mathrm{rank}(\lambda A + (1-\lambda)B) > \lambda \mathrm{rank}(A) + (1-\lambda)\mathrm{rank}(B)$$

- Nuclear norm $\|\cdot\|_* : \mathbb{R}^{m\times n} \to \mathbb{R}$ is convex since it is a norm

Example. Given

$$X = \begin{bmatrix} 3.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad Y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

We have $\mathrm{rank}(X) = 3$, $\|X\|_* = 3.5 + 3 + 2 = 8.5$, and $\mathrm{rank}(Y) = 2$, $\|Y\|_* = 1 + |-1| = 2$.

Convex envelope

Given a (possibly nonconvex) function $f : C \to \mathbb{R}$, the convex envelope of f is defined as the largest convex function g such that

$$g(x) \leq f(x), \quad \forall\, x \in C$$

Namely, among all convex functions below f, g is the best approximation to f.

Convex envelope

Given a (possibly nonconvex) function $f : C \to \mathbb{R}$, the convex envelope of f is defined as the largest convex function g such that

$$g(x) \leq f(x), \quad \forall\, x \in C$$

Namely, among all convex functions below f, g is the best approximation to f.

Theorem [1] The convex envelope of rank function on the set $\{X \in \mathbb{R}^{m\times n} \mid \sigma_1(X) \leq 1\}$ is the nuclear norm.

The theorem implies that the convex envelope of $\operatorname{rank}(X)$ on the set $\{X \in \mathbb{R}^{m\times n} \mid \sigma_1(X) \leq K\}$ is $\|X\|_*/K$.

Low-rank matrix completion

- Given a partially observed matrix $M \in \mathbb{R}^{m \times n}$. Ω is the index set of observed entries

$$M = \begin{bmatrix} 1 & ? & ? & ? \\ ? & 2 & 3 & ? \\ 3 & ? & ? & 1 \end{bmatrix}, \quad \Omega = \{(1,1),(2,2),(2,3),(3,1),(3,4)\}$$

- Assume the true matrix is low-rank, low-rank matrix completion seeks to find the lowest rank matrix X that matches the known entries

$$\begin{aligned} \min_X \quad & \operatorname{rank}(X) \\ \text{s.t.} \quad & X_{ij} = M_{ij}, \quad \forall\,(i,j) \in \Omega \end{aligned}$$

This problem is NP-hard.

- Nuclear norm is the best convex approximation to rank function

$$\begin{aligned} \min_X \quad & \operatorname{rank}(X) \\ \text{s.t.} \quad & X_{ij} = M_{ij} \end{aligned} \quad \overset{\text{convex relaxation}}{\Longrightarrow} \quad \begin{aligned} \min_X \quad & \|X\|_* \\ \text{s.t.} \quad & X_{ij} = M_{ij} \end{aligned}$$

Theoretical guarantee of recovery

$$\begin{aligned} \min_X \quad & \|X\|_* \\ \text{s.t.} \quad & X_{ij} = M_{ij},\, (i,j) \in \Omega \end{aligned}$$

Theorem (informal) [3] Suppose the true matrix has rank k. Suppose the location of observed entries are uniformly at random. Suppose we observe $\geq mnk$ entries. Then the above nuclear norm minimization problem recovers the true matrix with high probability.

For example, if a row is not sampled, then the recovery for that row is almost impossible

$$M = \begin{bmatrix} 1 & ? & 2 & ? \\ ? & ? & ? & ? \\ 3 & 1 & ? & 1 \end{bmatrix}$$

Algorithm 1: SDP

Due to the fact that

$$\|X\|_* = \min_{W_1, W_2} \quad \frac{1}{2}(\mathrm{Tr}(W_1) + \mathrm{Tr}(W_2))$$
$$\text{s.t.} \quad \begin{bmatrix} W_1 & X \\ X^T & W_2 \end{bmatrix} \succeq 0 \text{ (positive semidefinite)}$$

the nuclear norm minimization problem

$$\begin{aligned} \min_X \quad & \|X\|_* \\ \text{s.t.} \quad & X_{ij} = M_{ij},\, (i,j) \in \Omega \end{aligned}$$

can be cast as a semidefinite program (SDP) and solved by a generic SDP solver

$$\begin{aligned} \min_{W_1, W_2, X} \quad & \frac{1}{2}(\mathrm{Tr}(W_1) + \mathrm{Tr}(W_2)) \\ \text{s.t.} \quad & X_{ij} = M_{ij},\, (i,j) \in \Omega \\ & \begin{bmatrix} W_1 & X \\ X^T & W_2 \end{bmatrix} \succeq 0 \end{aligned}$$

Algorithm 2: PG

$$\begin{aligned} \min_{X} \quad & \|X\|_* \\ \text{s.t.} \quad & X_{ij} = M_{ij} \end{aligned} \quad \overset{\text{penalized}}{\Longrightarrow} \quad \min_{X} \quad \|X\|_* + \frac{1}{2\mu} \sum_{(i,j)\in\Omega} (X_{ij} - M_{ij})^2$$

We design PG for solving

$$\min_{X} \quad \underbrace{\frac{1}{2} \sum_{(i,j)\in\Omega} (X_{ij} - M_{ij})^2}_{f(X)\,\text{smooth}} + \underbrace{\mu\|X\|_*}_{g(X)\,\text{nonsmooth}}$$

PG needs two ingredients

- Gradient $\nabla f(X) \in \mathbb{R}^{m\times n}$

$$\left[\nabla f(X)\right]_{ij} = \frac{\partial f(X)}{\partial X_{ij}} = \begin{cases} X_{ij} - M_{ij}, & \text{if } (i,j) \in \Omega \\ 0, & \text{if } (i,j) \notin \Omega \end{cases}$$

- Proximal mapping $P_g(X)$

Proximal mapping of nuclear norm

Theorem The proximal mapping

$$P_{\mu\|\cdot\|_*}(Y) = \arg\min_X \left\{ \mu\|X\|_* + \frac{1}{2}\|X - Y\|_F^2 \right\}$$

is obtained by soft-thresholding the singular values of Y

$$Y = U\mathrm{Diag}(\sigma)V^T$$

$$\gamma_i = S_\mu(\sigma_i) = \begin{cases} \sigma_i - \mu, & \text{if } \sigma_i > \mu \\ 0, & \text{if } 0 \le \sigma_i \le \mu \end{cases}$$

$$P_{\mu\|\cdot\|_*}(Y) = U\mathrm{Diag}(\gamma)V^T$$

Proof The proof is based on

1. The nuclear and Frobenius norms are orthogonally invariant:
   $\|U\Sigma V^T\|_* = \|\Sigma\|_*$, $\|U\Sigma V^T\|_F = \|\Sigma\|_F$ for orthogonal U and V
2. Therefore, we can work with diagonal matrices X and Y

Proximal mapping of nuclear norm

Proof For $X = \mathrm{Diag}(\gamma)$, $Y = \mathrm{Diag}(\sigma)$, the problem

$$\min_X \left\{ \mu \|X\|_* + \frac{1}{2}\|X - Y\|_F^2 \right\}$$

becomes

$$\min_\gamma \quad \sum_i \underbrace{\left(\mu|\gamma_i| + \frac{1}{2}(\gamma_i - \sigma_i)^2 \right)}_{\text{Prox. map. of } |\cdot|}$$

$\Rightarrow \gamma_i = |\gamma_i| = S_\mu(\sigma_i)$

Alternative proof is based on the optimality condition $0 \in \mu\partial(\|\cdot\|_*)(X) + X - Y$, which requires the subdifferential of the nuclear norm.

Each PG iteration need an SVD, which can be time-consuming when the dimension is large.

M. Fazel.
Matrix rank minimization with applications.
PhD thesis, PhD thesis, Stanford University, 2002.

P. Jain, P. Netrapalli, and S. Sanghavi.
Low-rank matrix completion using alternating minimization.
In Proceedings of the forty-fifth annual ACM symposium on Theory of computing, pages 665–674, 2013.

B. Recht, M. Fazel, and P. A. Parrilo.
Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization.
SIAM review, 52(3):471–501, 2010.