Machine Learning: Recommender System

Prof. Mingkui Tan

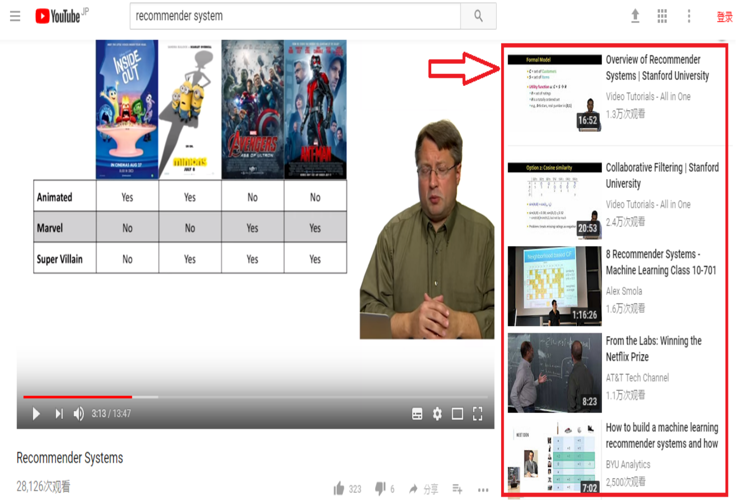
South China University of Technology
Southern Artificial Intelligence Laboratory(SAIL)

December 9, 2017

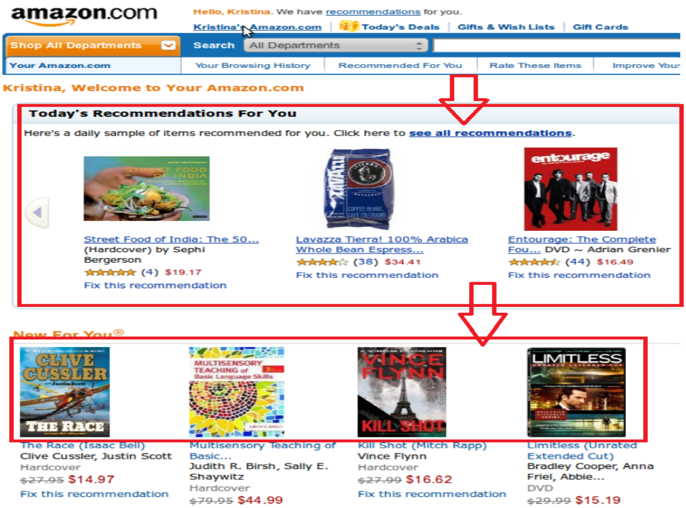
Content

1. Introduction
2. Memory-based Collaborative Filtering
3. Model-based Collaborative Filtering

Example: Recommender System



Example: Recommender System



Recommender System

Recommender System applies statistical and knowledge discovery techniques to the problem of making product recommendations.

Advantages of recommender systems:

- Improve conversion rate: Help customers find a product she/he wants to buy.
- Cross-selling: Suggest additional products.
- Improve customer loyalty: Create a value-added relationship.

The Value of Recommendations

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

Collaborative Filtering

Make **automatic predictions (filtering)** about the interests of a user by collecting preferences or taste information **from many other users (collaboration)**.

Type of CF Algorithms

- **Memory-based CF**: utilize the entire user-item database to generate a prediction.
  - **User-based CF**: find similar users to predict ratings.
  - **Item-based CF**: use similar items to predict ratings.
- **Model-based CF**: build a model from the rating data (**Matrix factorization**, etc.) and use this model to predict missing ratings.

User-based CF

Recommendations based on the **relationship between items**.

The basic steps

1. Identify **set of items** rated by the target user u.
2. Identify which **other users (neighbours)** were rated by the items set.
3. Compute **similarity** between each neighbour and select k **most similar neighbours**.
4. Predict ratings for the target user.

User-based CF

	Item1	Item2	Item3	Item4	Item5	Item6
User1	2			4	5	
User2	5		4			1
User3			5		2	
User4		1		5		4
User5			4			2
User6	4	5		1		

1. For the target user5, identify the relevant items set $I = \{\text{item3, item6}\}$.

User-based CF

	Item1	Item2	Item3	Item4	Item5	Item6
User1	2			4	5	
User2	5		4			1
User3			5		2	
User4		1		5		4
User5			4			2
User6	4	5		1		

2. Identify users set $\{$user2, user3, user4$\}$ were rated by the items in set $I = \{$item3, item6$\}$.

User-based CF

3. Compute **similarity** between each neighbour and select k **most similar neighbours**.
   - **Pearson correlation coefficient**
   - Cosine similarity
   - Others

Pearson correlation coefficient between user U and V:

$$sim(U,V) = \frac{\sum_{i \in I}(r_{ui} - \hat{r}_u)(r_{vi} - \hat{r}_v)}{\sqrt{\sum_{i \in I}(r_{ui} - \hat{r}_u)^2 \sum_{i \in I}(r_{vi} - \hat{r}_v)^2}}$$

- I denotes relevant items set.
- r_{ui} and r_{vi} denote the ratings of user U and V for item i.
- $\hat{r}_u$ and $\hat{r}_v$ denote the average ratings of user U and V.

User-based CF

	Item1	Item2	Item3	Item4	Item5	Item6	
User1	2			4	5		
User2	5		4			1	sim = 0.87
User3			5		2		sim = 1
User4		1		5		4	sim = -1
User5			4			2	
User6	4	5		1			

- Compute **similarity** between each neighbour based on **Pearson correlation coefficient**.
- For example, compute similarity between user2 and user5:

  $\hat{r}_{user2}$ (average rating) is (5+4+1)/3 = 3.34

  $\hat{r}_{user5}$ (average rating) is (4+2)/2 = 3

$$sim(user2, user5) = \frac{(4-3.34)\cdot(1-3.34)+(4-3)\cdot(2-3)}{\sqrt{((4-3.34)^2+(1-3.34)^2)\cdot((4-3)^2+(2-3)^2)}}$$

$$= 0.87$$

User-based CF

4. Make predictions based on the k nearest neighbors for user u.

Predict rating $r_{u,i}$ by

$$r_{u,i} = \hat{r}_u + \frac{\sum_{v \in K} sim(v,u)(r_{v,i} - \hat{r}_v)}{\sum_{v \in K} |sim(v,u)|}$$

- K denotes the set of k nearest neighbors.
- $|sim(v,u)|$ denotes the absolute value of similarity.
- $\hat{r}_u$ denotes the average rating of user u.
- $\hat{r}_v$ denotes the average rating of user v.
- $sim(v,u)$ denotes the similarity between user v and u.

User-based CF

	Item1	Item2	Item3	Item4	Item5	Item6	
User1	2			4	5		
User2	5		4			1	sim = 0.87
User3			5		2		sim = 1
User4		1		5		4	sim = -1
User5	3.51	3.81	4	2.42	2.48	2	
User6	4	5		1			

- Make predictions based on the k nearest neighbors.
- For example, make prediction of user5 for item1:

$$r_{user5,item1} = (4+2)/2 + \frac{0.87\cdot(5-(5+4+1)/3)}{|0.87|}$$
$$= 3.51$$

Item-based CF

Recommendations based on the **relationship between items**.

The basic steps

1. Identify set of users who rated the target item i.
2. Identify which other items (neighbours) were rated by the users set.
3. Compute similarity between each neighbour and select k most similar neighbours.
4. Predict ratings for the target item.

Item-based CF

	Item1	Item2	Item3	Item4	Item5	Item6
User1	2			4	5	
User2	5		4			1
User3			5		2	
User4		1		5		4
User5			4			2
User6	4	5		1		

1. For the target item5, identify the relevant set $U = \{$user2, user4, user5$\}$.

Item-based CF

	Item1	Item2	Item3	Item4	Item5	Item6
User1	2			4	5	
User2	5		4			1
User3			5		2	
User4		1		5		4
User5			4			2
User6	4	5		1		

2. Identify items {item1, item2, item3, item4} were rated by the users in set U.

Item-based CF

3. Compute **similarity** between each item (neighbour) and select k **most similar neighbours**.
    - **Pearson correlation coefficient**
    - Cosine similarity
    - Others

Pearson correlation coefficient between item I and J:

$$sim(I,J) = \frac{\sum_{u\in U}(r_{ui}-\hat{r}_i)(r_{uj}-\hat{r}_j)}{\sqrt{\sum_{u\in U}(r_{ui}-\hat{r}_i)^2 \sum_{u\in U}(r_{vj}-\hat{r}_j)^2}}$$

- U denotes relevant users set.
- r_{ui} and r_{uj} denote the ratings of user u for item I and item J.
- $\hat{r}_i$ and $\hat{r}_j$ denote the average ratings of item I and J.

Item-based CF

	Item1	Item2	Item3	Item4	Item5	Item6
User1	2			4	5	
User2	5		4			1
User3			5		2	
User4		1		5		4
User5			4			2
User6	4	5		1		
	sim = -1	sim = -1	sim = 0.86	sim = 1		

- Compute **similarity** between each neighbour based on **Pearson correlation coefficient**.
- For example, compute similarity between item1 and item6:

  $\hat{r}_{item3}$ (average rating) is $(4+5+4)/3 = 4.34$

  $\hat{r}_{item6}$ (average rating) is $(1+4+2)/3 = 2.34$

$$sim(item1, item6) = \frac{(4-4.34)\cdot(1-2.34)+(4-4.34)\cdot(2-2.34)}{\sqrt{((4-4.34)^2+(4-4.34)^2)\cdot((1-2.34)^2+(2-2.34)^2)}}$$
$$= 0.86$$

Item-based CF

4. Make predictions based on the k nearest neighbors for item i.

Predict rating $r_{u,i}$ by

$$r_{u,i} = \hat{r}_i + \frac{\sum_{j \in K} sim(j,i)(r_{u,j} - \hat{r}_j)}{\sum_{j \in K} |sim(j,i)|}$$

- K denotes the set of k nearest neighbors.
- $|sim(j,i)|$ denotes the absolute value of similarity.
- $\hat{r}_i$ denotes the average rating of item i.
- $\hat{r}_j$ denotes the average rating of item j.
- $sim(j,i)$ denotes the similarity between item j and i.

Item-based CF

	Item1	Item2	Item3	Item4	Item5	Item6
User1	2			4	5	2.94
User2	5		4			1
User3			5		2	2.48
User4		1		5		4
User5			4			2
User6	4	5		1		1.12
	sim = -1	sim = -1	sim = 0.86	sim = 1		

- Make predictions based on the k nearest neighbors.
- For example, make prediction of item6 for user1:

$$\begin{aligned} r_{item6,user1} &= (1+4+2)/3 + \frac{-1\cdot(2-(2+5+4)/3)+1\cdot(4-(4+5+1)/3)}{|-1|+|1|} \\ &= 2.94 \end{aligned}$$

Memory-based CF

Differences between Item-based CF and User-based CF

User-based similarity is more dynamic.

- Precompute user neighbourhood may lead to poor predictions.

Item-based similarity is static.

- Precompute item neighbourhood may provide accurate results.

Memory-based CF

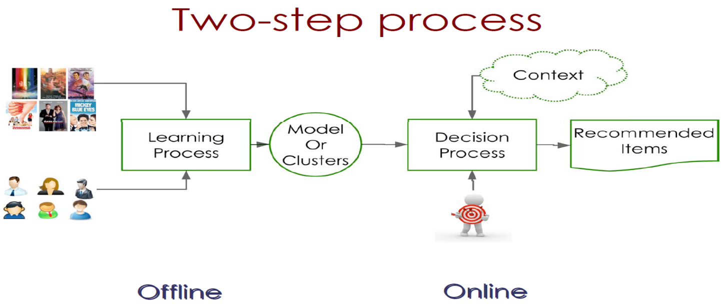
Strengths

- Require minimal knowledge engineering efforts.
- Users and products are symbols without any internal structure or characteristics.
- Produce good-enough results in most cases.

Weaknesses

- Require a large number of explicit and reliable ratings.
- Highly time consuming to compute similarity with millions of users & items.

Challenges of Memory-based CF



- Two-step process: similarity computation in the offline setting andd prediction process in the online setting.
- Accuracy problem: difficult to make accurate predictions based on nearest neighbors.
- Sparsity problem: the number of observed samples is less than 1%.

Model-Based CF Algorithms

Models are learned from the **underlying data** rather than heuristics

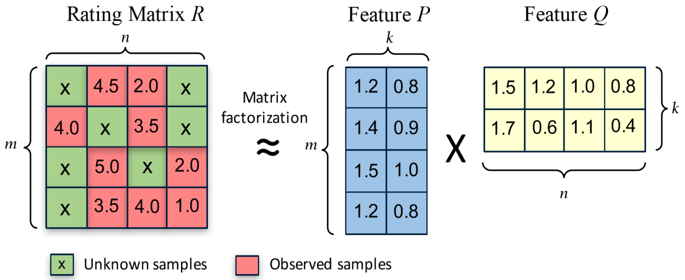
Models of user ratings (or purchases):

- **Matrix Factorization**
- Clustering (classification)
- Association Rules
- Other models

Matrix Factorization is the **most widely used** algorithm.

Matrix Factorization

- Give a rating matrix $\mathbf{R} \in \mathbb{R}^{m \times n}$, with **sparse ratings** from m users to n items.
- Assume rating matrix $\mathbf{R}$ can be factorized into the multiplication of two **low-rank feature matrices** $\mathbf{P} \in \mathbb{R}^{m \times k}$ and $\mathbf{Q} \in \mathbb{R}^{k \times n}$.



An example of matrix factorization (m=4, n=4, k=2).

Determine an Objective Function

- Determine an **objective function**.
  - **Squared error loss**:
    $$\mathcal{L}(r_{u,i}, \hat{r}_{u,i}) = (r_{u,i} - \hat{r}_{u,i})^2$$
  - Binary hinge loss:
    $$\mathcal{L}(r_{u,i}, \hat{r}_{u,i}) = \max(0, 1 - r_{u,i}\hat{r}_{u,i})$$
- **Notes:** r_{ui} denotes the **actual rating** of user u for item i and $\hat{r}_{ui}$ denotes the **prediction**.

Alternating Least Square (ALS) for MF

ALS is to minimize the following objective function:

$$\mathcal{L} = \sum_{u,i} (r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i)^2 + \lambda(\sum_{u} n_{\mathbf{p}_u} ||\mathbf{p}_u||^2 + \sum_{i} n_{\mathbf{q}_i} ||\mathbf{q}_i||^2)$$

- $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_m]^\top \in \mathbb{R}^{m \times k}$.
- $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_n] \quad \in \mathbb{R}^{k \times n}$.
- $r_{u,i}$ denotes the **actual rating** of user u for item i.
- λ is **regularization parameter** to avoid overtting.
- $n_{\mathbf{p}_u}$ and $n_{\mathbf{q}_i}$ denote **the number of total ratings** on user u and item i, respectively.

General Steps of ALS

Algorithm 1 General Steps of ALS

1: **Require** rating matrix $\mathbf{R}$, feature matrices $\mathbf{P}$, $\mathbf{Q}$ and regularization parameter λ.
2: **Optimize** $\mathbf{P}$ while fixing $\mathbf{Q}$.
3: **Optimize** $\mathbf{Q}$ while fixing $\mathbf{P}$.
4: **Repeat** the above processes until **convergence**.

Optimize $\mathbf{P}$ while fixing $\mathbf{Q}$

- Objective function:

$$\mathcal{L} = \sum_{u,i} (r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i)^2 + \lambda(\sum_u n_{\mathbf{p}_u} ||\mathbf{p}_u||^2 + \sum_i n_{\mathbf{q}_i} ||\mathbf{q}_i||^2)$$

- Optimize $\mathbf{P}$ while **fixing** $\mathbf{Q}$:

  The first order optimality:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{p}_u} = 0$$

$$\Rightarrow \sum_{r_{u,i} \neq 0} (\mathbf{q}_i \mathbf{q}_i^\top + \lambda n_{\mathbf{p}_u} I) \cdot \mathbf{p}_u = \mathbf{Q}^\top \cdot \mathbf{R}_{u*}^\top$$

$$\Rightarrow \mathbf{p}_u = (\mathbf{q}_i \mathbf{q}_i^\top + \lambda n_{\mathbf{p}_u} I)^{-1} \cdot \mathbf{Q}^\top \cdot \mathbf{R}_{u*}^\top$$

- $\mathbf{R}_{u*}$ denotes the u-th row of rating matrix $\mathbf{R}$.
- Update **all** $\mathbf{p}_u$ with the above formula.

Optimize $\mathbf{Q}$ while fixing $\mathbf{P}$

- Objective function:

$$\mathcal{L} = \sum_{u,i} (r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i)^2 + \lambda(\sum_u n_{\mathbf{p}_u} ||\mathbf{p}_u||^2 + \sum_i n_{\mathbf{q}_i} ||\mathbf{q}_i||^2)$$

- Optimize $\mathbf{Q}$ while **fixing** $\mathbf{P}$:

  The first order optimality:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{q}_i} = 0$$

$$\Rightarrow \sum_{r_{u,i} \neq 0} (\mathbf{p}_u \mathbf{p}_u^\top + \lambda n_{\mathbf{q}_i} I) \cdot \mathbf{q}_i = \mathbf{P}^\top \cdot \mathbf{R}_{*i}^\top$$

$$\Rightarrow \mathbf{q}_i = (\mathbf{p}_u \mathbf{p}_u^\top + \lambda n_{\mathbf{q}_i} I)^{-1} \cdot \mathbf{P}^\top \cdot \mathbf{R}_{*i}$$

- $\mathbf{R}_{*i}$ denotes the i-th column of rating matrix $\mathbf{R}$.
- Update **all** $\mathbf{q}_i$ with the above formula.

ALS for MF

Algorithm 2 ALS Algorithm

1: **Require** rating matrix $\mathbf{R}$, feature matrices $\mathbf{P}$, $\mathbf{Q}$ and regularization parameter λ.
2: **Optimize** $\mathbf{P}$ while fixing $\mathbf{Q}$:
$$\mathbf{p}_u = (\mathbf{q}_i\mathbf{q}_i^\top + \lambda n_{\mathbf{p}_u} I)^{-1} \cdot \mathbf{Q}^\top \cdot \mathbf{R}_{u*}^\top.$$
3: **Optimize** $\mathbf{Q}$ while fixing $\mathbf{P}$:
$$\mathbf{q}_i = (\mathbf{p}_u\mathbf{p}_u^\top + \lambda n_{\mathbf{q}_i} I)^{-1} \cdot \mathbf{P}^\top \cdot \mathbf{R}_{*i}.$$
4: **Repeat** the above processes until **convergence**.

Why SGD ?

Time complexity per iteration of ALS:

$$O(|\Omega|k^2 + (m+n)k^3)$$

- $|\Omega|$ denotes the number of observed samples.
- k denotes the rank.
- m and n denote the number of users and items.
- ALS is not scalable to large-scale datasets.

Time complexity per iteration of SGD:

$$O(|\Omega|k)$$

- SGD is scalable to large-scale datasets.

Stochastic Gradient Descent (SGD) for MF

SGD is to minimize the following objective function:

$$\mathcal{L} = \sum_{u,i \in \Omega} (r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i)^2 + \lambda_p ||\mathbf{p}_u||^2 + \lambda_q ||\mathbf{q}_i||^2$$

- $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_m]^\top \in \mathbb{R}^{m \times k}$.
- $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_n] \quad \in \mathbb{R}^{k \times n}$.
- $r_{u,i}$ denotes the **actual rating** of user u for item i.
- Ω denotes the set of **observed samples** from rating matrix $\mathbf{R}$.
- λ_p and λ_q are **regularization parameters** to avoid overtting.

General Steps of SGD

Algorithm 3 General Steps of SGD

1: **Require** feature matrices $\mathbf{P}$, $\mathbf{Q}$, observed set Ω, regularization parameters λ_p, λ_q and learning rate α.
2: **Randomly** select an observed sample $r_{u,i}$ from observed set Ω.
3: Calculate the **gradient** w.r.t to the objective function.
4: **Update** the feature matrices $\mathbf{P}$ and $\mathbf{Q}$ with learning rate α and gradient.
5: **Repeat** the above processes until **convergence**.

Calculate the gradient

- Objective function:

$$\mathcal{L} = (r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i)^2 + \lambda_p ||\mathbf{p}_u||^2 + \lambda_q ||\mathbf{q}_i||^2$$

- **Randomly** select an observed sample $r_{u,i}$.
- Calculate the **prediction error**:

$$E_{u,i} = r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i$$

- Calculate the **gradient**:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{p}_u} = E_{u,i}(-\mathbf{q}_i) + \lambda_p \mathbf{p}_u$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{q}_i} = E_{u,i}(-\mathbf{p}_u) + \lambda_q \mathbf{q}_i$$

Update the feature matrices

- Objective function:
$$\mathcal{L} = (r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i)^2 + \lambda_p ||\mathbf{p}_u||^2 + \lambda_q ||\mathbf{q}_i||^2$$
- **Randomly** select an observed sample $r_{u,i}$.
- Calculate the **prediction error**:
$$E_{u,i} = r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i$$
- Calculate the **gradient**:
$$\frac{\partial \mathcal{L}}{\partial \mathbf{p}_u} = E_{u,i}(-\mathbf{q}_i) + \lambda_p \mathbf{p}_u$$
$$\frac{\partial \mathcal{L}}{\partial \mathbf{q}_i} = E_{u,i}(-\mathbf{p}_u) + \lambda_q \mathbf{q}_i$$
- Update the feature matrices $\mathbf{P}$ and $\mathbf{Q}$ with **learning rate** α:
$$\mathbf{p}_u = \mathbf{p}_u + \alpha(E_{u,i}\mathbf{q}_i - \lambda_p \mathbf{p}_u)$$
$$\mathbf{q}_i = \mathbf{q}_i + \alpha(E_{u,i}\mathbf{p}_u - \lambda_q \mathbf{q}_i)$$

SGD for MF

Algorithm 4 SGD Algorithm

1: **Require** feature matrices $\mathbf{P}$, $\mathbf{Q}$, observed set Ω, regularization parameters λ_p, λ_q and learning rate α.

2: **Randomly** select an observed sample $r_{u,i}$ from observed set Ω.

3: Calculate the **gradient** w.r.t to the objective function:

$$E_{u,i} = r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{p}_u} = E_{u,i}(-\mathbf{q}_i) + \lambda_p \mathbf{p}_u$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{q}_i} = E_{u,i}(-\mathbf{p}_u) + \lambda_q \mathbf{q}_i$$

4: **Update** the feature matrices $\mathbf{P}$ and $\mathbf{Q}$ with learning rate α and gradient:

$$\mathbf{p}_u = \mathbf{p}_u + \alpha(E_{u,i}\mathbf{q}_i - \lambda_p \mathbf{p}_u)$$

$$\mathbf{q}_i = \mathbf{q}_i + \alpha(E_{u,i}\mathbf{p}_u - \lambda_q \mathbf{q}_i)$$

5: **Repeat** the above processes until **convergence**.

Matrix Factorization

Comparison between SGD and ALS

- ALS is easier to **parallelize** than SGD.
- ALS **converges faster** than the SGD.
- SGD has less **storage complexity** than ALS. (ALS needs to store the rating matrix $\mathbf{R}$)
- SGD has less **computational complexity** than ALS. (ALS needs to compute the matrix-vector multiplication)

Accuracy Measures

Mean Absolute Error (MAE) computes the deviation between predicted ratings and actual ratings.

$$\text{MAE} = \frac{1}{|\Omega|} \sum_{u,i \in \Omega} |\hat{r}_{u,i} - r_{u,i}|$$

- Ω denotes the **observed set**.
- $|\Omega|$ denotes the **number of observed set**.
- $r_{u,i}$ denotes the **actual rating** and $\hat{r}_{u,i}$ denotes the **prediction**.

Root Mean Square Error (RMSE) is similar to MAE, but places more emphasis on larger deviation.

$$\text{RMSE} = \sqrt{\sum_{u,i \in \Omega} (\hat{r}_{u,i} - r_{u,i})^2 / |\Omega|}$$

Thank you!

Questions?