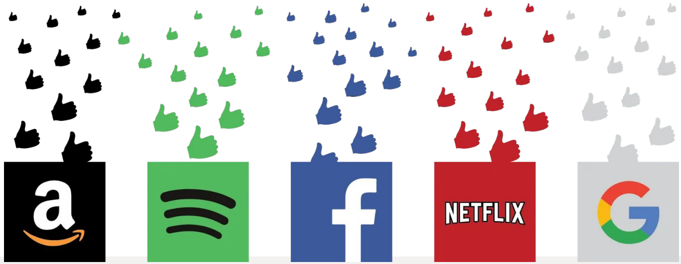
Collaborative Filtering



CS 189/289A Project T Final

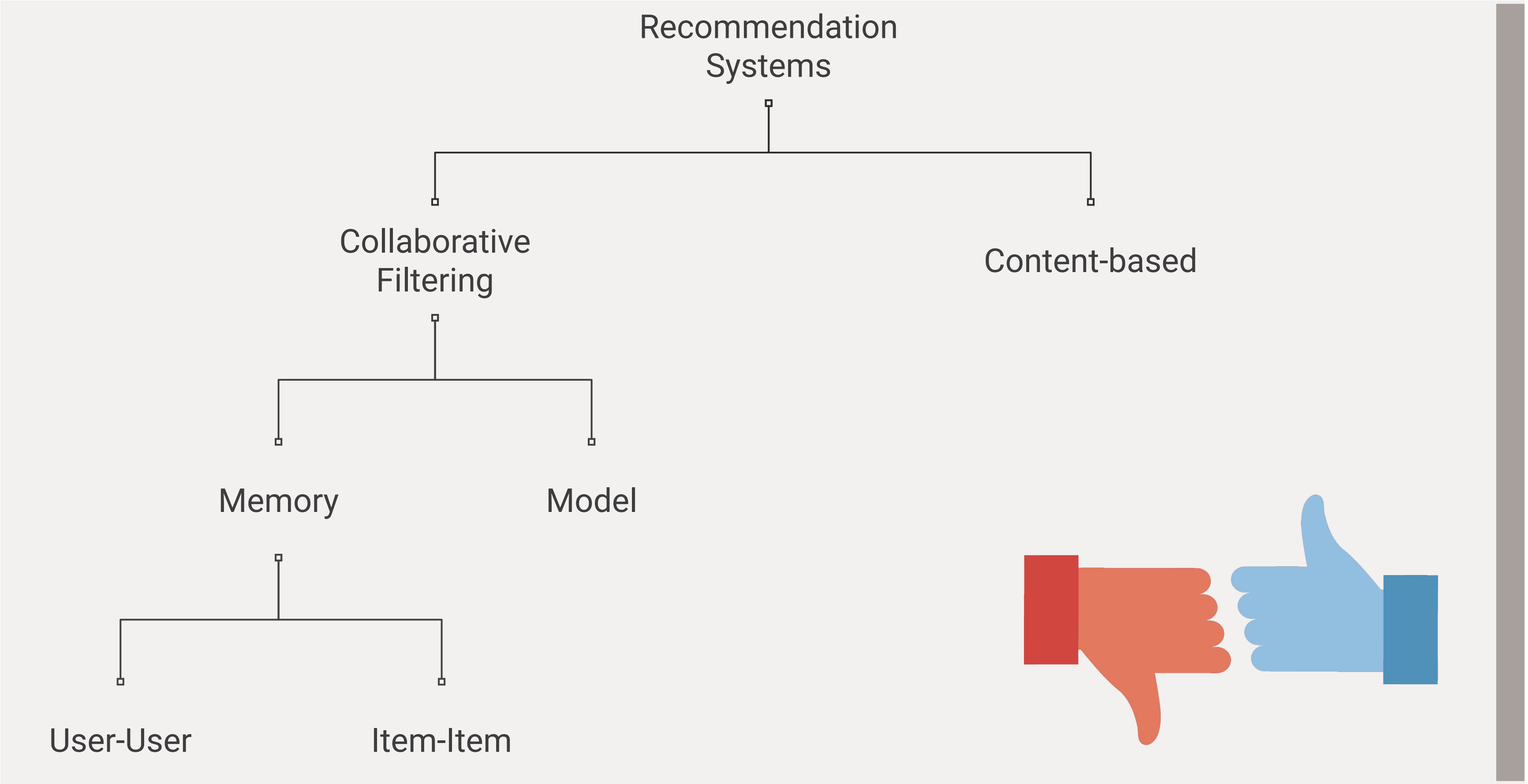
Team MA
Maxwell Chen
Abinav Routhu

Intro

Recommendation Systems & Their Classification

Why Recommendation Systems?

- According to McKinsey, 35% of Amazon.com's revenue is generated by its recommendation engine
- Netflix estimates the recommendation system saves the company around $1 Billion annually
- Recommendations are responsible for 70% of the time people spend watching videos on YouTube


Recommendation Systems
Collaborative Filtering
Content-based
Memory
Model
User-User
Item-Item

Paradigm

Content-Based

- Require featurization
- Conceptually simple (out of box classical models)
- Phrased as regression or classification
- Difficult to exploit user-user **and** item-item relations

Collaborative Filtering

- Implicit featurization
- Novel concepts (specific to Rec. Sys.)
- Phrased as clustering or optimization problem
- Information-efficient

*Commercial deployments are overwhelmingly *hybrid* systems.

Memory Approach

The data as it is

User-Interaction Matrix

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
User A	4	5	3	2	~~0~~	3
User B	~~0~~	~~0~~	~~0~~	5	~~0~~	~~0~~
User C	2	~~0~~	1	2	1	2
User D	~~0~~	5	~~0~~	2	~~0~~	~~0~~
User E	~~0~~	5	1	2	~~0~~	1

What would User D rate Item 3? User E rate Item 6?

- User A rates Item 2 a 5
- User B has only rated Item 4
- User C has not rated Item 2
- User C gives low ratings
- Item 4 is very popular
- Item 2 is very highly rated

- User-Interaction Matrix can be **massive** but always very **sparse** (mostly null entries)
- Sparsity not structured -- no expectation which items have been rated
- In most use cases, the # of users >> # of items
- Can decipher patterns based on similarities between users!
- Not necessary to abstract a model, only work with data, only use what's *in memory*

$$\left\{\begin{matrix} 3 & 1 & 3 \\ 1 & \cancel{0} & 3 \\ \cancel{0} & \cancel{0} & \cancel{0} \\ 4 & 5 & 3 \\ \cancel{0} & \cancel{0} & 4 \\ 2 & \cancel{0} & 1 \\ \cancel{0} & 5 & \cancel{0} \\ \cancel{0} & 5 & 1 \\ 3 & \cancel{0} & \cancel{0} \\ \cancel{0} & \cancel{0} & \cancel{0} \\ \cancel{0} & \cancel{0} & \cancel{0} \end{matrix}\right.$$

Memory Approach

User-User

1. Identify similar users to User X
2. Find highly rated items from set of similar users
3. Recommend top items not yet rated by User X

User-User

1. **Identify similar users to User X**

How?

Use K-NN algorithm on row vectors!

User-User

1. **Identify similar users to User X**

2. **Find highly rated items from set of similar users**

Look at column sums of submatrix.

User-User

1. **Identify similar users to User X**

2. **Find highly rated items from set of similar users**

3. **Recommend top items not yet rated by User X**

   Check against original U-I Matrix.

User-User (In action)

User D wants recommendations.

$$\begin{pmatrix} 3 & 5 & 4 & 2 & \cancel{0} & 3 \\ \cancel{0} & \cancel{0} & \cancel{0} & 5 & \cancel{0} & \cancel{0} \\ 2 & \cancel{0} & 1 & 2 & 1 & 2 \\ \cancel{0} & 5 & \cancel{0} & 2 & \cancel{0} & \cancel{0} \\ \cancel{0} & 5 & 1 & 2 & \cancel{0} & 1 \end{pmatrix}$$

User-User (In action)

User D is most similar to User A and User E.

$$\begin{pmatrix} 3 & 5 & 4 & 2 & \cancel{0} & 3 \\ \cancel{0} & \cancel{0} & \cancel{0} & 5 & \cancel{0} & \cancel{0} \\ 2 & \cancel{0} & 1 & 2 & 1 & 2 \\ \cancel{0} & 5 & \cancel{0} & 2 & \cancel{0} & \cancel{0} \\ \cancel{0} & 5 & 1 & 2 & \cancel{0} & 1 \end{pmatrix}$$

User-User (In action)

Calculate the column sums of the submatrix.

$$\begin{pmatrix} 3 & 5 & 4 & 2 & \cancel{0} & 3 \\ \cancel{0} & 5 & 1 & 2 & \cancel{0} & 1 \end{pmatrix}$$

$$\mathbf{3} \quad \mathbf{10} \quad \mathbf{5} \quad \mathbf{4} \quad \mathbf{0} \quad \mathbf{4}$$

User-User (In action)

User D has already rated Item 2 and 4.

User D is recommended Item 3 and Item 6.

$$
\begin{pmatrix}
3 & 5 & 4 & 2 & \cancel{0} & 3 \\
\cancel{0} & \cancel{0} & \cancel{0} & 5 & \cancel{0} & \cancel{0} \\
2 & \cancel{0} & 1 & 2 & 1 & 2 \\
\cancel{0} & 5 & \cancel{0} & 2 & \cancel{0} & \cancel{0} \\
\cancel{0} & 5 & 1 & 2 & \cancel{0} & 1
\end{pmatrix}
$$

$$
3 \quad 10 \quad \mathbf{5} \quad 4 \quad 0 \quad \mathbf{4}
$$

User-User

Advantages

- Simple and intuitive
- Well understood
- Very personalized
- Adaptive with different similarity measures

Disadvantages

- Scales poorly because of k-nn runtime
- Unstable (very few values determine result)
- Most similarity measures have bad edge cases

Item-Item

1. Find User X’s top rated items
2. Find other similar items for each item
3. Recommend most frequently found items in search

Item-Item

1. **Find User X's top rated items**

   How?

   Sort by rating.

Item-Item

1. **Find User X's top rated items**

2. **Find other similar items for each item**

   Run k-nearest neighbors on the columns on the UI Interaction Matrix.

Item-Item

1. Find User X's top rated items
2. Find other similar items for each item
3. Recommend most frequently found items in search

Search for repeats in list.

Model Approach

Latent Space Assumption & Matrix Factorization

SVD Algorithm

A=UT

$$
\begin{pmatrix}
4 & 5 & 3 & 2 & \cancel{0} & 3 \\
\cancel{0} & \cancel{0} & \cancel{0} & 5 & \cancel{0} & \cancel{0} \\
2 & \cancel{0} & 1 & 2 & 1 & 2 \\
\cancel{0} & 5 & \cancel{0} & 2 & \cancel{0} & \cancel{0} \\
\cancel{0} & 5 & 1 & 2 & \cancel{0} & 1
\end{pmatrix}
$$

The End