NYU | Center for Data Science

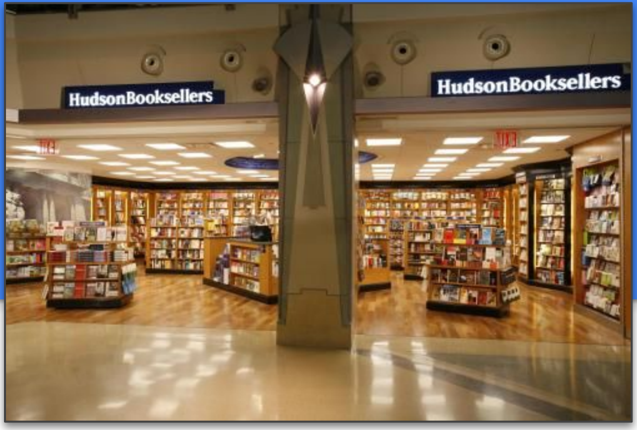
Week 10.1: Recommender systems

DS-GA 1004: Big Data

This week

- Recommender systems
- Collaborative filtering algorithms
- Ranking and evaluation

Why recommendation?



- **Physical objects occupy space**
    - Brick-and-mortar shops must satisfy physical constraints
    - Curators must prioritize for some notion of utility
    - Serving the most customers, maximizing sales/profit, etc.
- This is not true for **digital items**!
    - The web, e-books, news articles, movies, music, … take up no physical space
    - Without curation, this quickly becomes overwhelming

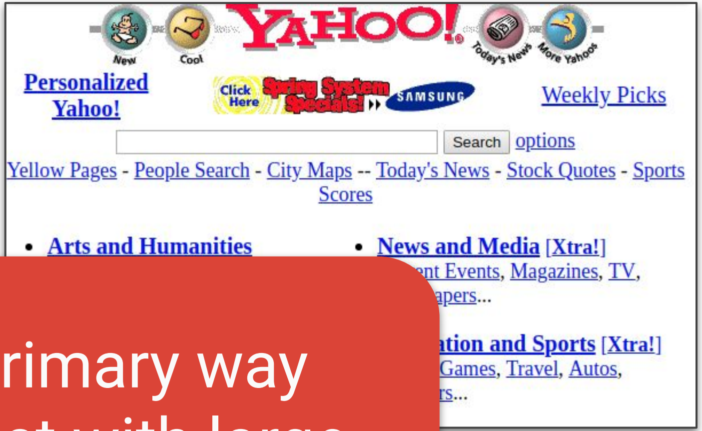
Search ⇒ Recommendation

- Early solutions relied on **indexing** and **search**
  - Users must describe what they want
  - Same query by different users gets same results
- **Recommendation** = **search** + **personalization**
  - A user's past history informs the ranking of results
  - **Other users' histories are also informative!**

Yahoo.com in April, 1997 [archive.org]

Search ⇒ Recommendation

- Early solutions relied on **indexing** and **search**
  - Users must describe what they want
  - Same query by different users gets same results
- **Recommendation = search + personalization**



April, 1997 [archive.org]

Recommender systems are now the primary way that most people (unknowingly) interact with large collections!

Personalization

- Traditional search / information retrieval:
    - Model the **relevance** of an **item** in response to a **query**
- Personalized search / recommendation
    - Same as above, but model can access a description of the **user**
- To model **relevance** per user, we'll need to collect data
    - This is known as **feedback**
- This approach works best in the big data regime

Implicit vs Explicit feedback

Explicit feedback

- Examples:
  - $$$ (Purchases)
  - ★★★☆☆
  - 👍👎
  - +Subscribe / Conversion
- **Strong signal** (+/-)
- **Relatively rare**: users will not always provide explicit feedback

Implicit vs Explicit feedback

Explicit feedback

- Examples:
  - $$$ (Purchases)
  - ★★★☆☆
  - 👍👎
  - +Subscribe / Conversion
- **Strong signal** (+/-)
- **Relatively rare**: users will not always provide explicit feedback

Implicit feedback

- Examples:
  - Click-through
  - Downloads
  - Play counts
  - Abandonment / skipping
- **Weak signal** (+ only, usually)
- **Much more abundant** than explicit feedback

How is feedback collected?

- **Explicit feedback** makes sense when

  Time/effort to rate item is much less than time to consume the item!

  - **Examples**: movies, books, restaurants, …
- **Implicit feedback** is the more common approach
  - **Examples**: Songs, news articles, short videos, search engine results, …

Collaborative filtering

- ***Utility matrix* (R)**: feedback for sparsely observed interactions

Items

Users

		1			1		
				0	0		1
	1	1			1		
		1		0			

- **Task**: predict the missing entries $\rightarrow R_{ui} \approx f(\mathrm{User} = u, \mathrm{Item} = i)$

- **Evaluation**: depends on the feedback mechanism`

Baseline #1: Popularity

- **This should always be the first thing you try.**
  - 1. Compute the **average utility** for each item

$$P[i] \leftarrow (\textstyle\sum_u R[u, i]) \;/\; |R[:, i]|$$

  - 2. f = items ranked by descending $P[i]$
- This produces the **same ranking for all users**. **Personalization** should improve, but by how much?
- AKA: the **short head**; **greatest hits**; **Starbucks mix**, …

Improving the popularity baseline

- Which would you rather have?
  - Item 1: **1000 ratings**, avg = ★★★★☆
  - Item 2: **1 rating**, avg = ★★★★★
- Few **interactions** $|R[:, i]|$ ⇒ **unstable estimates** $P[i]$
- **Quick fix 1**: discard items with too few ratings
- **Quick fix 2**: use a **prior**:

$$P[i] \leftarrow \left(\sum_u R[u, i]\right) / \left(|R[:, i]| + \beta\right)$$

  - Equivalent to having $\beta>0$ **extra observations** with $R=0$
  - AKA "**damping**" or "**pseudo-counts**"

Global, item, and user bias

- Model each interaction as a combination of **global**, **item**, and **user** terms:

$$R[u, i] \approx \mu + b[i] + b[u]$$

Global, item, and user bias

- Model each interaction as a combination of **global**, **item**, and **user** terms:

$$R[u, i] \approx \mu + b[i] + b[u]$$

- The average rating over all interactions:

$$\mu = (\textstyle\sum_{u,i} R[u, i])/(|R| + \beta_g)$$

Global, item, and user bias

- Model each interaction as a combination of **global**, **item**, and **user** terms:

$$R[u, i] \approx \mu + b[i] + b[u]$$

- The average rating over all interactions:

$$\mu = (\Sigma_{u,i} R[u, i])/(|R| + \beta_g)$$

- Average (users) difference from μ for item i:

$$b[i] = (\Sigma_u R[u, i] - \mu)/(|R[:, i]| + \beta_i)$$

Global, item, and user bias

- Model each interaction as a combination of **global**, **item**, and **user** terms:

$$R[u, i] \approx \mu + b[i] + b[u]$$

- The average rating over all interactions:

$$\mu = (\textstyle\sum_{u,i} R[u, i])/(|R| + \beta_g)$$

- Average (users) difference from μ for item i:

$$b[i] = (\textstyle\sum_{u} R[u, i] - \mu)/(|R[:, i]| + \beta_i)$$

- Average (items) difference from $\mu+b[i]$ for user u:

$$b[u] = (\textstyle\sum_{i} R[u, i] - \mu - b[i])/(|R[u]| + \beta_u)$$

Predicting from the bias model

- $R[u, i] \approx \mu + b[i] + b[u]$ lets us estimate all interactions
- Predictions for user u: sort items i by descending $\mu + b[i] + b[u]$
- μ and $b[u]$ are constant in this prediction, since u is fixed.
- Equivalently, we can sort by the item bias $b[i]$
- This isn't any more powerful than the original popularity model
  - but it is more interpretable!

Summary

part 1

- Recommender systems are everywhere —
  you should know how they work.
- The utility matrix encodes interactions between users and items.

  Not all entries are observed!
- Always start with a non-personalized bias model.

  This will let you measure the impact of personalization.