Recommender Systems

Hüsnü Sensoy, **Global Maksimum AI Team**

Recommender Systems

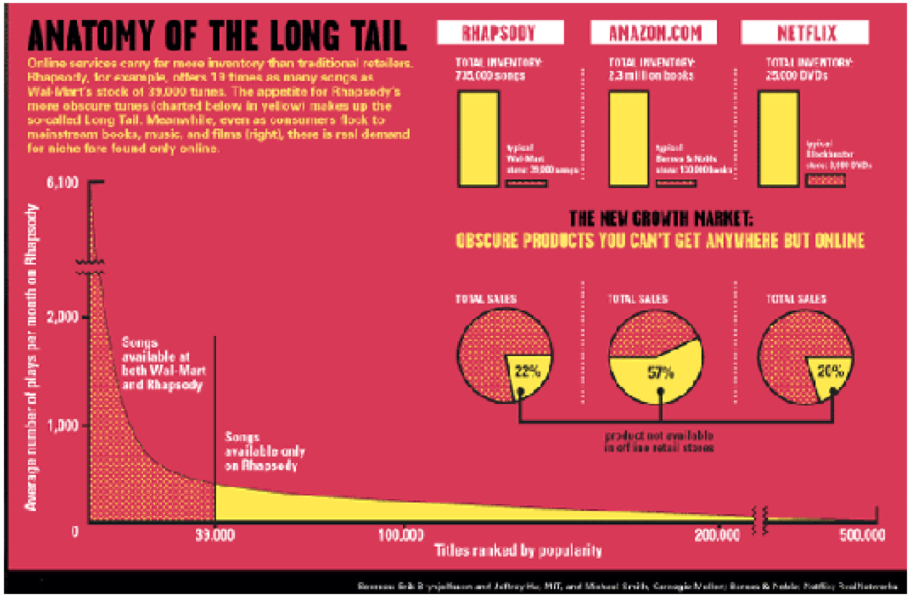
- Overview Wednesday
- Content-based Systems Wednesday
- Collaborative Filtering Friday & Saturday
- Evaluating Recommender Systems Friday & Saturday

This section is based on [Jeff Ullmanns MMDS Course]

Overview

Long Tail Distribution

How **Into the Air** made **Touching the Void** a bestseller



Read the Article

More choices necessitates better filters

- Books, movies, music, news articles
- Fashion
- Bids
- People (friend recommendations of social media)

Type of Recommendations

- Editorial and hand curated
  - List of favorites
  - List of "essential" items
- Simple Aggregates
  - Top 10, Most Popular, Recent Uploads
- **Tailored to individual user**
  - Netflix, Trendyol, Sahibinden.com...

Formal Model

- C = set of **Customers**
- S = set of **Items**
- Utility Function $u : C \times S \to R$
  - R = set of Ratings
  - R is a totally ordered set
  - e.g., **0-5** stars, real numbers in **[0,1]**

Totally Ordered Set

A Short Update on Classification vs Regression

- How does it sound to compare different y values of a **regression** problem ?
  - **Rental Price:** [1000₺, 10000₺]
  - **Dam Capacity:** [0,100]
- How does it sound to compare different y values of a **classification** problem ?
  - **Sentiment:** Positive, Negative
  - **Image classification:** Cat, Dog

Utility Matrix

	Avatar	*BladeRunner*	*Matrix*	*Pirates*
Alice	1		.2	
Bob		0.5		0.3
Carol	.2		$1^{Carol \heartsuit Matrix}$	
David				0.4

Extrapolating Utility Matrix

Utility Matrix

	Avatar	*BladeRunner*	*Matrix*	*Pirates*
Alice	1		.2	
Bob		0.5		0.3
Carol	.2		1	
David				0.4

Problems to Address

1. Building Utility Matrix
    - How to collect data ?
2. Extrapolate unknown ratings usign the known ones
    - Note that **utility matrix** is highly sparse.
3. Evaluating extrapolation methods
    - How to measure the **performance** of a recommender system ?

Building Utility Matrix

- **Explicit**: Ask people to rate items
  - 👍 Usually gives a better estimator for a given user
  - 👎 Doesn't scale very well (think of utility matrix size) 🤔
- **Implicit**: Learn ratings from user actions
  - 👍 Much scalable
  - 👎 Defining rules might be challenging
  - 👎 How about low ratings ?
- **Explicit + Implicit**: Combine two

👈

Extrapolating Utility Matrix

- **Key Problem**: Matrix **U** is sparse
  - r value is missing for most (u, s) pair. 🤔
  - Cold Start Problem
    - A **row** of **U** is completely empty 🤔
    - A **column** of **U** is completely empty 🤔
- **Three main approaches**

1. Content-based
2. Collaborative
3. Latent factor based 👈

Content-based Systems

Content-based Recommendations

Main Idea: Recommend items to customer c similar to previous items rated higly by c

- **Movie**
  - Same actor(s), director, genre...
- **Website, blog, news or any document**
  - Documents with **similar** content
- **People**
  - Recommend people with many common friends, common hobbies, etc.

Item Profiles

- For each item, create an **item profile**
- Convenient way to think of the item profile as a vector
  - One entry per feature
  - Vector might be boolean/real-valued

Item Profiles

How to incorporate Text Features

- Important words in text features
- Defining importance of a word as the combination (multiplication) of two
  - Presence of a word in a document (**tf**)
  - A word is trivial if it occurs in many documents (**idf**)

Term Frequency (tf)

f_{ij} = frequency of $word_i$ in $document_j$

🤔We obtain **term frequency** by normalizing with **frequency of the most frequent word in that sentences.**

$$tf_{ij} = \frac{f_{ij}}{max_k f_{kj}}$$

Refer Term Frequency for more term frequency calculation heuristics.

Inverse Document Frequency (idf)

n_i = Number of documents in which $word_i$ is mentioned.

N = Total number of documents.

$$idf_i = \frac{N}{n_i}$$

Refer Inverse Document Frequency for more idf calculation heuristics.

Learn tf-idf by example

- For a simple case let's assume we have 2 text items to be profiled
  - s^1: this^1 is^2 a^3 sample^4
  - s^2: this^1 is^2 another^5 sample^4
- tf
  - $tf_{11} = \frac{1}{1}, tf_{21} = \frac{1}{1}, tf_{31} = \frac{1}{1}, tf_{41} = \frac{1}{1}, tf_{51} = \frac{0}{1}$
  - $tf_{12} = \frac{1}{1}, tf_{22} = \frac{1}{1}, tf_{32} = \frac{0}{1}, tf_{42} = \frac{1}{1}, tf_{52} = \frac{1}{1}$
- $idf_1 = log\frac{2}{2}, idf_2 = log\frac{2}{2}, idf_3 = log\frac{2}{1}, idf_4 = log\frac{2}{2}, idf_5 = log\frac{2}{1}$
- $p(s_1) = \langle 0, 0, 1, 0, 0 \rangle$
- $p(s_2) = \langle 0, 0, 0, 0, 1 \rangle$

Boolean Utility Matrix

- Assume customer c has watched 5 movies (remember **implicit** building of utility matrix.)
  - 2 movies featuring actor A
  - 3 movies featuring actor B
- Profile of user is $[0.4 \ 0.6]$

Star Rating

Making Recommendations

- We have **user profiles**
- We have **item profiles**
- We can generate prediction by using vector similarity between two
  - $U(w_1, w_2) = \frac{w_1 w_2}{|w_1||w_2|}$

Pros: Content-based Approach

- No need for data on other users
- Able to recommend to users with unique tastes
- Able to recommend new & unpopular items
- Explanations for recommended items

Cons: Content-based Approach

- Finding the appropriate featurse is hard 🤔
  - Images 🤔
  - Movies 🤔
  - Music 🤔
- Overspecialization
  - Never recommends items outside user's content profile
  - Unable to exploit quality judgements of other users
- Cold-start problem for new users