Introduction to Machine Learning

MODULE 5 Recommender Systems

Modules for this course

1. Overview: What is Machine learning
2. Categories of machine learning
3. Notation
4. Machine Learning application approach
5. **Recommender Systems**
6. Building a Recommender Engine

Module 5

Recommender Systems

Objectives

Objectives

- What is a Recommender System
- What is the difference between content based and collaborative filtering Recommender systems
- Which limitations recommender systems frequently encounter
- How collaborative filtering can identify similar users and items

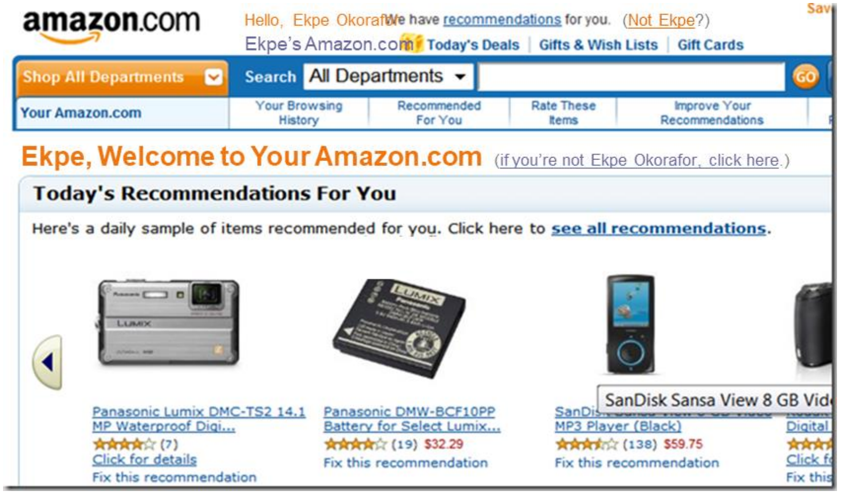
Outline

- What is a recommender system?
- Types of collaborative filtering
- Limitations of recommender systems
- Fundamental concepts
- Essential points
- Conclusion
- Hands-On Exercise: Implementing a Basic Recommender

Outline

- What is a recommender system?
- Types of collaborative filtering
- Limitations of recommender systems
- Fundamental concepts
- Essential points
- Conclusion
- Hands-On Exercise: Implementing a Basic Recommender

What is a Recommender System?



amazon.com

Sav

Hello, Ekpe OkorafWe have recommendations for you. (Not Ekpe?)

Ekpe's Amazon.com | Today's Deals | Gifts & Wish Lists | Gift Cards

Shop All Departments | Search | All Departments | GO

Your Amazon.com | Your Browsing History | Recommended For You | Rate These Items | Improve Your Recommendations

Ekpe, Welcome to Your Amazon.com (if you're not Ekpe Okorafor, click here.)

Today's Recommendations For You

Here's a daily sample of items recommended for you. Click here to see all recommendations.

Panasonic Lumix DMC-TS2 14.1 MP Waterproof Digi...
★★★★☆ (7)
Click for details
Fix this recommendation

Panasonic DMW-BCF10PP Battery for Select Lumix...
★★★★☆ (19) $32.29
Fix this recommendation

SanDisk Sansa View 8 GB Vid
MP3 Player (Black)
★★★½☆ (138) $59.75
Fix this recommendation

Digital
★★★★
Click f
Fix this

NETFLIX

Types of Recommendations

- **Editorial and hand curated**
  - List of favorites
  - Lists of “essential” items

- **Simple aggregates**
  - Top 10, Most Popular, Recent Uploads

- **Tailored to individual users**
  - Amazon, Netflix, …

Today

Formal Model

- ***X*** = set of **Users**
- ***S*** = set of **Items**

- **Utility function *u*: *X* × *S* → *R***
  - ***R*** = set of ratings
  - ***R*** is a totally ordered set
  - e.g., **0-5** stars, real number in **[0,1]**

Utility Matrix

	Avatar	LOTR	Matrix	Pirates
Alice	1		0.2	
Bob		0.5		0.3
Carol	0.2		1	
David				0.4

Key Problems

- **(1) Gathering “known” ratings for matrix**
    - How to collect the data in the utility matrix
- **(2) Extrapolate unknown ratings from the known ones**
    - Mainly interested in high unknown ratings
        - We are not interested in knowing what you don’t like but what you like
- **(3) Evaluating extrapolation methods**
    - How to measure success/performance of recommendation methods

(1) Gathering Ratings

- **Explicit**
  - Ask people to rate items
  - Doesn’t work well in practice – people can’t be bothered
  - Crowdsourcing: Pay people to label items

- **Implicit**
  - Learn ratings from user actions
    - E.g., purchase implies high rating
  - What about low ratings?

(2) Extrapolating Utilities

- **Key problem:** Utility matrix ***U*** is **sparse**
  - Most people have not rated most items
  - **Cold start:**
    - New items have no ratings
    - New users have no history

- **Three approaches to recommender systems:**
  - **1)** Content-based
  - **2)** Collaborative
  - **3)** Latent factor based

Today!

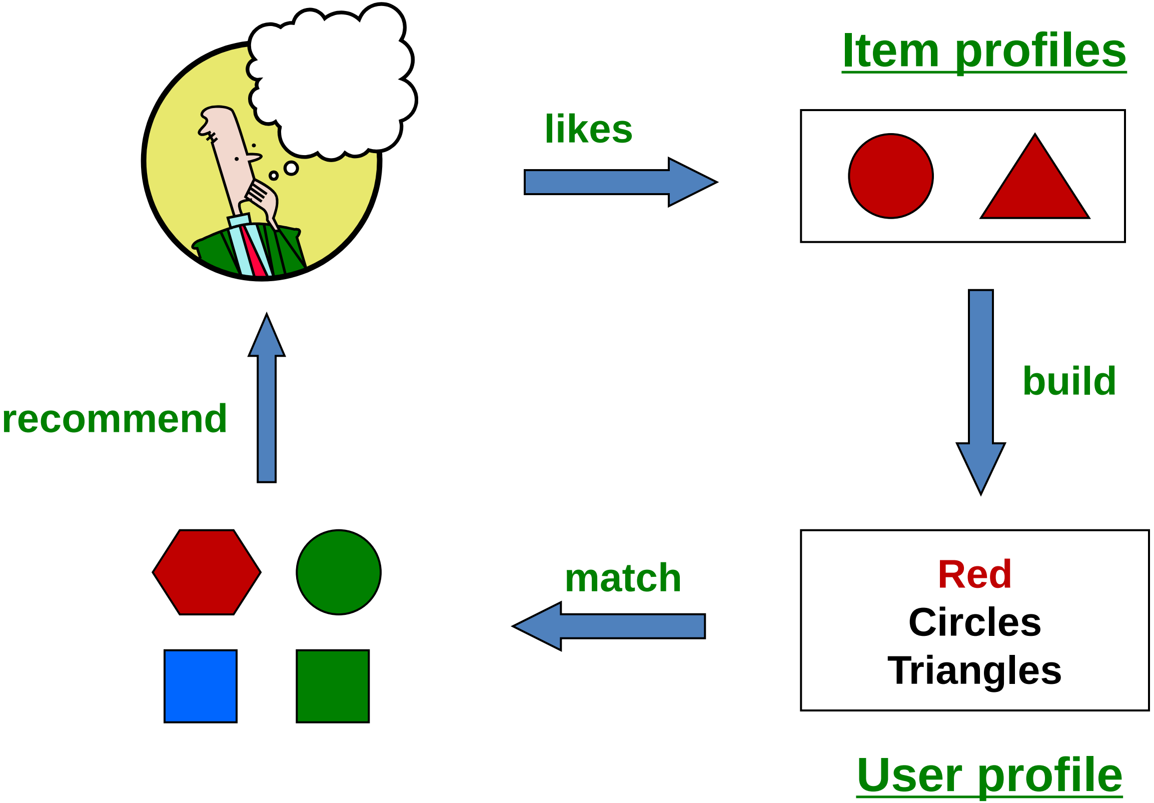
Content-Based Recommendations

- **Main idea:** Recommend items to customer ***x*** like previous items rated highly by ***x***

Example:

- **Movie recommendations**
  - Recommend movies with same actor(s), director, genre, …
- **Websites, blogs, news**
  - Recommend other sites with "similar" content

Plan of Action



Item profiles

likes

build

Red
Circles
Triangles

User profile

match

recommend

Items Profile

- For each item, create an **item profile**

- **Profile is a set (vector) of features**
  - **Movies:** author, title, actor, director,…
  - **Text:** Set of "important" words in document

- **How to pick important features?**
  - Usual heuristic from text mining is **TF-IDF** (Term frequency * Inverse Doc Frequency)
    - **Term … Feature**
    - **Document … Item**

Sidenote: TF-IDF

f_{ij} = frequency of term (feature) ***i*** in doc (item) ***j***

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

Note: we normalize TF to discount for "longer" documents

n_i = number of docs that mention term ***i***

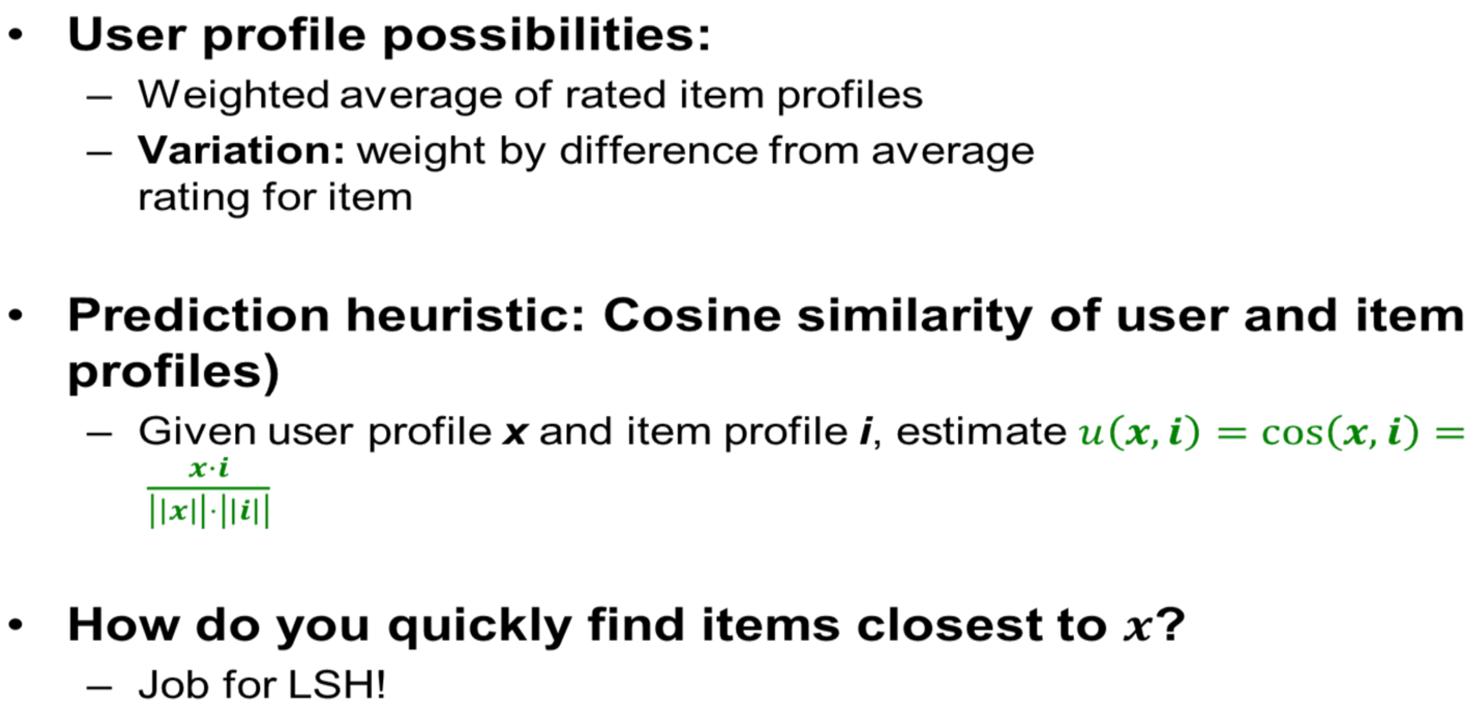
N = total number of docs

$$IDF_i = \log \frac{N}{n_i}$$

TF-IDF score: ***$w_{ij} = TF_{ij} \times IDF_i$***

Doc profile = set of words with highest **TF-IDF** scores, together with their scores

User Profiles and Prediction



- **User profile possibilities:**
  - Weighted average of rated item profiles
  - **Variation:** weight by difference from average rating for item
- **Prediction heuristic: Cosine similarity of user and item profiles)**
  - Given user profile ***x*** and item profile ***i***, estimate $u(\boldsymbol{x}, \boldsymbol{i}) = \cos(\boldsymbol{x}, \boldsymbol{i}) = \frac{\boldsymbol{x} \cdot \boldsymbol{i}}{||\boldsymbol{x}|| \cdot ||\boldsymbol{i}||}$
- **How do you quickly find items closest to x?**
  - Job for LSH!

Pros: Content-based Approach

- **+: No need for data on other users**
  - No cold-start or sparsity problems
- **+: Able to recommend to users with unique tastes**
- **+: Able to recommend new & unpopular items**
  - No first-rater problem
- **+: Able to provide explanations**
  - Can provide explanations of recommended items by listing content-features that caused an item to be recommended

Cons: Content-based Approach

- **–: Finding the appropriate features is hard**
  - E.g., images, movies, music
- **–: Recommendations for new users**
  - **How to build a user profile?**
- **–: Overspecialization**
  - Never recommends items outside user's content profile
  - People might have multiple interests
  - **Unable to exploit quality judgments of other users**

Types of Collaborative Filtering

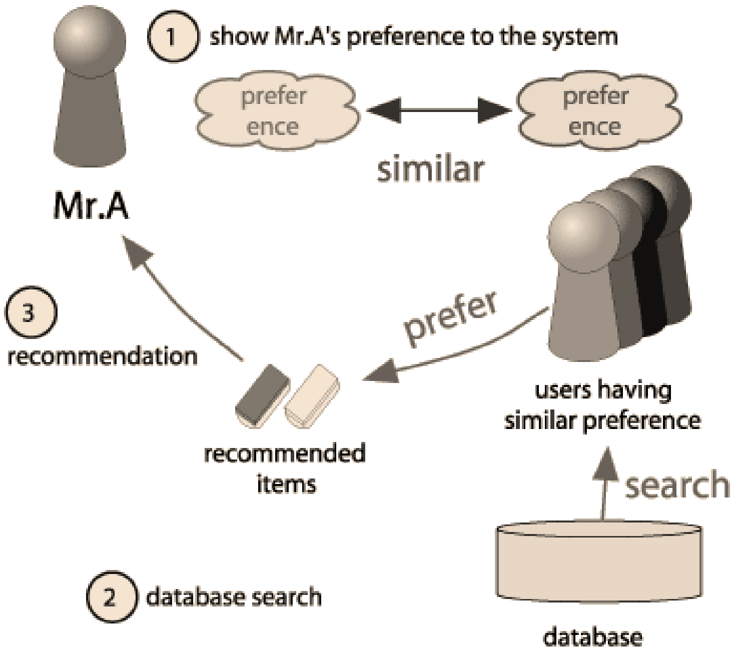
- **Collaborative filtering can be subdivided into two main types**
- **User-based: “What do users similar to you like?”**
  - For a given user, find other people who have similar tastes
  - Then, recommend items based on past behavior of those users
- **Item-based: “What is similar to other items you like?”**
  - Given items that a user likes, determine which items are similar
  - Make recommendations to the user based on those items

User-Based Collaborative Filtering

- Consider user ***x***

- Find set ***N*** of other users whose ratings are "**similar**" to ***x***'s ratings

- Estimate ***x***'s ratings based on ratings of users in ***N***



Finding “Similar” Users

- Let $\boldsymbol{r_x}$ be the vector of user $\boldsymbol{x}$’s ratings
- **Jaccard similarity measure**
  - **Problem:** Ignores the value of the rating
- **Cosine similarity measure**
  - sim($\boldsymbol{x}$, $\boldsymbol{y}$) = cos($\boldsymbol{r_x}$, $\boldsymbol{r_y}$) = $\frac{r_x \cdot r_y}{||r_x|| \cdot ||r_y||}$
  - **Problem:** Treats some missing ratings as “negative”
- **Pearson correlation coefficient**
  - $\boldsymbol{S_{xy}}$ = items rated by both users $\boldsymbol{x}$ and $\boldsymbol{y}$

$$sim(x,y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})(r_{ys} - \overline{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})^2} \sqrt{\sum_{s \in S_{xy}} (r_{ys} - \overline{r_y})^2}}$$

Rating Predictions

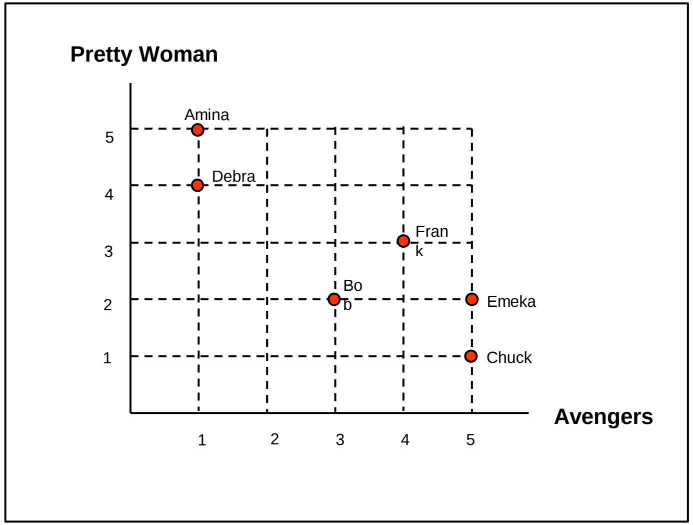
From similarity metric to recommendations:

- Let $\boldsymbol{r_x}$ be the vector of user ***x***'s ratings
- Let ***N*** be the set of ***k*** users most similar to ***x*** who have rated item ***i***
- **Prediction for item *i* of *user x*:**
  - $r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$
  - Or even better: $r_{xi} = \frac{\sum_{y \in N} s_{xy} \cdot r_{yi}}{\sum_{y \in N} s_{xy}}$

- **Many other tricks possible…**

User-Based Collaborative Filtering

- **User-based collaborative filtering is social**
  - It takes a “people first” approach, based on common interests
- **In this example, Amina and Debra have similar tastes**
  - Each is likely to enjoy a movie that the other rated highly



Item-Based Collaborative Filtering

- **So far: User-based collaborative filtering**
- **Another view: Item-based**
  - For item ***i***, find other similar items
  - Estimate rating for item ***i*** based on ratings for similar items
  - Can use same similarity metrics and prediction functions as in user-user model

$$\boxed{r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}}$$

s_{ij}… similarity of items ***i*** and ***j***
r_{xj}…rating of user ***x*** on item ***j***
N(i;x)… set items rated by ***x*** similar to ***i***

Item-Based Collaborative Filtering

users

movies

	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3			5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

□ - unknown rating 🟨 - rating between 1 to 5

Item-Based Collaborative Filtering

users

movies	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

🟥 - estimate rating of movie **1** by user **5**

Item-Based Collaborative Filtering

users

movies	1	2	3	4	5	6	7	8	9	10	11	12	**sim(1,m)**
1	1		3		?	5			5		4		**1.00**
2			5	4			4			2	1	3	**-0.18**
3	2	4		1	2		3		4	3	5		**0.41**
4		2	4		5			4			2		**-0.10**
5			4	3	4	2					2	5	**-0.31**
6	1		3		3			2			4		**0.59**

Neighbor selection:
Identify movies similar to movie **1**, **rated by user 5**

Here we use Pearson correlation as similarity:

1) Subtract mean rating m_i from each movie i

$m_1 = (1+3+5+5+4)/5 = \mathbf{3.6}$

row 1: *[-2.6, 0, -0.6, 0, 0, 1.4, 0, 0, 1.4, 0, 0.4, 0]*

2) Compute cosine similarities between rows

Item-Based Collaborative Filtering

users

movies	1	2	3	4	5	6	7	8	9	10	11	12	sim(1,m)
1	1		3		?	5			5		4		1.00
2			5	4			4			2	1	3	-0.18
3	2	4		1	2		3		4	3	5		<u>0.41</u>
4		2	4		5			4			2		-0.10
5			4	3	4	2					2	5	-0.31
6	1		3		3			2			4		<u>0.59</u>

Compute similarity weights:

$s_{1,3}$=0.41, $s_{1,6}$=0.59

Item-Based Collaborative Filtering

users

movies

	1	2	3	4	5	6	7	8	9	10	11	12	sim(1,m)
1	1		3		?	5			5		4		**1.00**
2			5	4			4			2	1	3	**-0.18**
3	2	4		1	2		3		4	3	5		**0.41**
4		2	4		5			4			2		**-0.10**
5			4	3	4	2					2	5	**-0.31**
6	1		3		3			2			4		**0.59**

Predict by taking weighted average:

$$r_{ix} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{jx}}{\sum s_{ij}}$$

$r_{1.5}$ = (0.41*2 + 0.59*3) / (0.41+0.59) = 2.6

Common Practice

- Define **similarity** ***s_{ij}*** of items ***i*** and ***j***
- Select ***k*** nearest neighbors ***N(i; x)***
  - Items most similar to ***i***, that were rated by ***x***
- Estimate rating ***r_{xi}*** as the weighted average:

$$r_{xi} = b_{xi} + \frac{\sum_{j \in N(i;x)} s_{ij} \cdot (r_{xj} - b_{xj})}{\sum_{j \in N(i;x)} s_{ij}}$$

baseline estimate for r_{xi}

$$\boldsymbol{b_{xi} = \mu + b_x + b_i}$$

- ***μ*** = overall mean movie rating
- ***b_x*** = rating deviation of user ***x***
  = *(avg. rating of user* ***x****)* **– *μ***
- ***b_i*** = rating deviation of movie ***i***

Item- vs User-based CF

	Avatar	LOTR	Matrix	Pirates
Alice	1		0.8	
Bob		0.5		0.3
Carol	0.9		1	0.8
David			1	0.4

- **In practice, it has been observed that item-based often works better than user-based**
- **Why?** Items are simpler, users have multiple tastes

Pros/Cons of Collaborative Filtering

- **+ Works for any kind of item**
  - No feature selection needed
- **- Cold Start:**
  - Need enough users in the system to find a match
- **- Sparsity:**
  - The user/ratings matrix is sparse
  - Hard to find users that have rated the same items
- **- First rater:**
  - Cannot recommend an item that has not been previously rated
  - New items, Esoteric items
- **- Popularity bias:**
  - Cannot recommend items to someone with unique taste
  - Tends to recommend popular items

Summary

- Recommendation systems use several different technologies. We can classify these systems into two broad groups.
  1. **Content-based** systems examine properties of the items recommended. For instance, if a Netflix user has watched many action movies, then recommend a movie classified in the database as having the “action” genre.
  2. **Collaborative filtering** systems recommend items based on similarity measures between users and/or items. The items recommended to a user are those preferred by similar users. This sort of recommendation system can use on similarity search and on clustering. However, these technologies by themselves are not sufficient, and there are some new algorithms that have proven effective for recommendation systems.