Recommender systems

Goal

- To know the motivation behind recommender systems (RS)
- To understand how RS works and know the principle behind RS methods/algorithms

NTNU

Motivation

Problem: How to **recommend items to users** to make user, content partner, websites happy!

Recommendations

Products, web sites, blogs, news items

From Scarcity to Abundance

- Shelf space scarce commodity for traditional retailers
  - Also: TV networks, movie theatres,…
- Web enables near-zero-cost dissemination of information about products
  - From scarcity to abundance
- More choice necessitates better filters
  - Recommendation engines
  - How Into Thin Air made Touching the Void a bestseller: http://www.wired.com/wired/archive/12.10/tail.html

The Long Tail Phenomenon

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, Apple Music, Spotify, Facebook Ads…

Formal Model

- ***X*** = Set of **Customers**
- ***S*** = Set of **Items**

- **Utility function u: *X*** x ***S* –> *R***
  - ***R*** = Set of **ratings**
  - ***R*** 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
    - 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**:
  - **Sparsity** of utility matrix U
    - Most people have not rated most items
  - **Cold start**:
    - New items - no ratings
    - New users - no history
- **Three approaches to recommender systems:**

  1) Content-based
  2) Collaborative
  3) Latent factor based

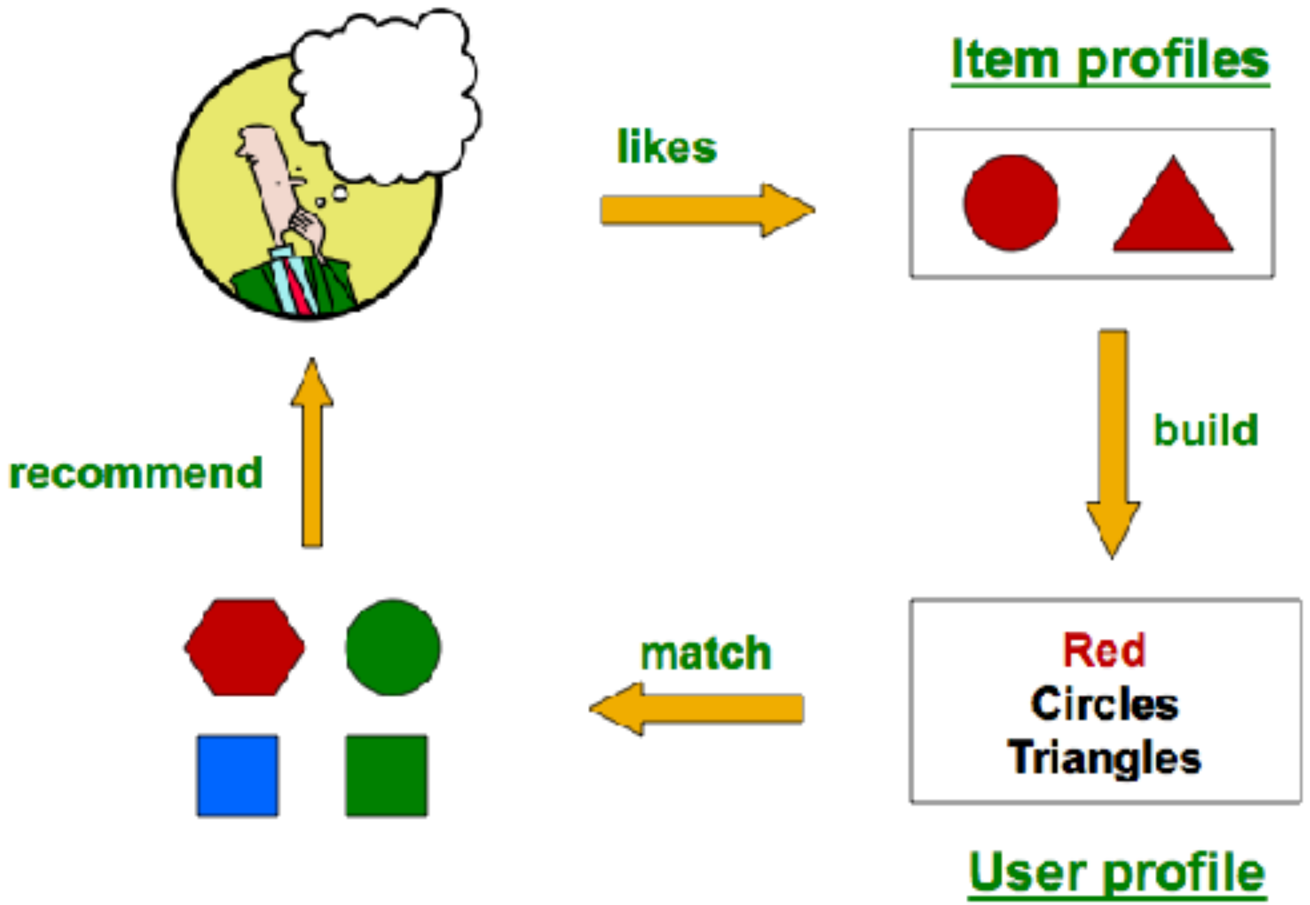
Content-based Recommendations

- **Main idea**:
  Recommend items to customer *X* similar to 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

- For each item, create an item profile
- Profile: 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

Item Features

- Description of items in terms of attributes
  - E.g.: Type, Director, Actors
- Description via keywords
- Possibility to look at content itself, like the text

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: $w_{ij} = TF_{ij} \times IDF_i$

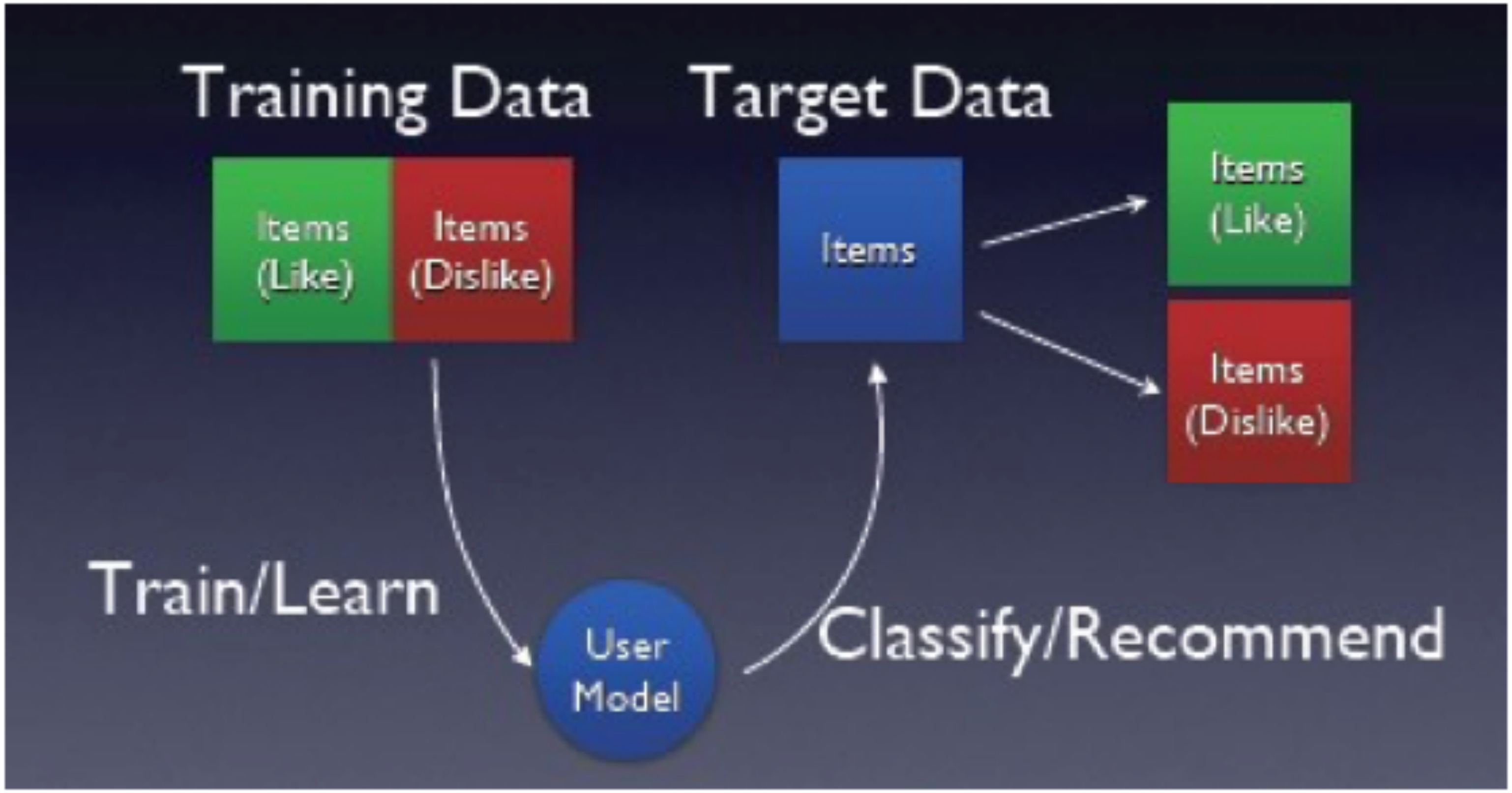
Doc profile = set of words with highest **TF-IDF** scores, together with their scores

User Profiles and Prediction

- User profile possibilities:
  - Weighted average of rated item profiles
  - Variation: weight by difference from average rating for item
  - ...
- Prediction heuristic
  - Given user profile *x* and item profile *i*, estimate

$$u(\boldsymbol{x}, \boldsymbol{i}) = \cos(\boldsymbol{x}, \boldsymbol{i}) = \frac{\boldsymbol{x} \cdot \boldsymbol{i}}{||\boldsymbol{x}|| \cdot ||\boldsymbol{i}||}$$

Learning a User Model



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-bases Approach

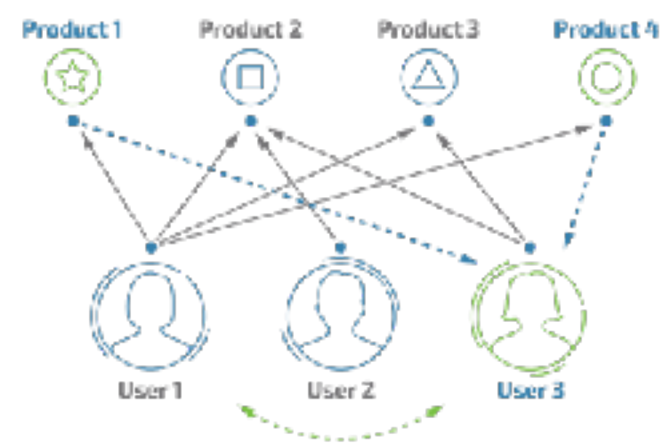
- **–: Finding the appropriate features is hard**
  - E.g., images, movies, music
- **–: Recommendations for new users**
  - How to build a user profile?
- **–: Overspecialization**
  - Never recommends items outside user's content profile
  - People might have multiple interests
  - **Unable to exploit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

Collaborative Filtering

How can we exploit quality judgments of other users to provide relevant recommendation?

Collaborative Filtering

- Maintain a database of many **users' ratings of a variety of items**.
- For a given user, **find other similar users whose ratings strongly correlate** with the current user.
- **Recommend items rated highly by these similar users**, but not rated by the current user.
- Almost all existing commercial recommenders use this approach (e.g. Amazon).



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*

Find “similar users”

r_x = [*, _, _, *, ***]
r_y = [*, _, **, **, _]

- Let r_x be the vector of user x's ratings
- **Jaccard similarity measure**
  - **Problem**: Ignores the value of the rating
- **Cosine similarity measure**

$$sim(x,y) = cos(r_x, r_y) = \frac{r_x \cdot r_y}{\|r_x\| \cdot \|r_y\|}$$

  - **Problem:** Treats missing ratings as “negative”
- **Pearson correlation coefficient**
  - S_{xy} = Set of items rated by both users x and y

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

r_x, r_y as sets:
r_x = {1, 4, 5}
r_y = {1, 3, 4}

r_x, r_y as points:
r_x = {1, 0, 0, 1, 3}
r_y = {1, 0, 2, 2, 0}

$\mathbf{r_x}$, $\mathbf{r_y}$ … avg. rating of **x**, **y**

Similarity Metric

$$sim(x,y) = \frac{\sum_i r_{xi} \cdot r_{yi}}{\sqrt{\sum_i r_{xi}^2} \cdot \sqrt{\sum_i r_{yi}^2}}$$

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- Intuitively we want: sim(A, B) > sim(A, C)
- **Jaccard similarity**: 1/5 < 2/4
- **Cosine similarity**: 0.380 > 0.322
  - Considers missing ratings as "negative"
  - Solution: subtract the (row) mean

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	$2/3$			$5/3$	$-7/3$		
B	$1/3$	$1/3$	$-2/3$				
C				$-5/3$	$1/3$	$4/3$	
D		0					0

sim A,B vs. A,C:
0.092 **>** -0.559

Notice cosine sim. is correlation when data is centered at 0

Rating Predictions

- **From similarity metric to recommendations:**
  - Let r_x: vector of user x's ratings
  - Let N : set of k users most similar to x who have rated item i
- Prediction for item s of user x:
  - $r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$
  - $r_{xi} = \frac{\sum_{y \in N} s_{xy} \cdot r_{yi}}{\sum_{y \in N} s_{xy}}$
  - Other options?
- Many other tricks possible…

Shorthand:
$s_{xy} = sim(x, y)$

J. Leskovec, A. Rajaraman, J. Ullman: Mining of Massive Datasets, http://www.mmds.org

Item-Item Collaborative Filtering

- So far: **User-user collaborative filtering**
- Another view: **Item-item**
  - 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

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

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

Item-Item CF (|N|=2)

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-Item CF (|N|=2)

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-Item CF (|N|=2)

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>

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-Item CF (|N|=2)

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

Compute similarity weights:

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

Item-Item CF (|N|=2)

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	

Predict by taking weighted average:

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

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

CF: Common Practice

Before:

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

- 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-Item vs. User-User

	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: item-item often works better than user-user
- 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

Pros/Cons of Collaborative Filtering (2)

- **- 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

Cold Start

- **New item problem**:
  - small number of users that rated an item, accurate prediction for this item cannot be generated.
- **New user problem**:
  - small number of items rated by a user, it is unlikely that there could be an overlap of items rated by this user and active users. User− to−user similarity cannot be reliably computed.
- **New community problem**:
  - Without sufficient ratings, it's hard to differentiate value by personalized CF recommendations.
- Clear **reward systems** necessary to convince users to vote or rate items.

Possible solution to cold start

- **As the solution for new user problem:**
  - Displaying non−personalized recommendation until the user has rated enough
  - Asking the user to describe their taste in aggregate
  - Asking the user for demographic information and using ratings of other users with similar demographics as recommendations
- **As the solution for new item problem:**
  - Recommending items through non−CF techniques content analysis or metadata
  - Randomly selecting items with few or no ratings and asking user to rate those items.
- **As the solution for new community problem:**
  - Provide ratings incentives to a small “bootstrap” subset of the community, before inviting the entire community.

Data Sparsity & Ratings scarcity

- The ratings matrix is sparse and only a small fraction of all possible user item entries is known.
- Many CF algorithms designed specifically for data sets with many more users than items (e.g., the MovieLens data set has 65,000 users and 5,000 movies).
- CF may be inappropriate in a domain where there are many more items than users.
- Implicit vs. explicit ratings

Evaluation of Collaborative Filtering

- To determine the quality of the predictions and recommendations
  - **Accuracy**
    - Rating accuracy : error between the predicted ratings and the true ratings. **Mean Absolute Error (MAE)** = average absolute difference between the predicted ratings and the actual rating given by a user
  - **Precision**
  - **Rank accuracy**: half−life utility.
  - **Novelty / Serendipity** (Karypis, 2001)
  - **Coverage** (Sarwar, et. al., 2000)
  - **Learning Rate** (Schein, et. al., 2001)
  - **Confidence** (Herlocker,2000)
  - **User Satisfaction** (Swearingen & Sinha, 2001; Dahlen, B. J., 1998)
  - Site Performance

Hybrid Methods

- **Implement two or more different recommenders and combine predictions**
  - Perhaps using a linear model
- **Add content-based methods to collaborative filtering**
  - Item profiles for new item problem
  - Demographics to deal with new user problem