Recommender Systems (I)

©Dr. Min Chi
mchi@ncsu.edu

AI Academy

Examples:

amazon.com

PANDORA

StumbleUpon

del.icio.us

NETFLIX

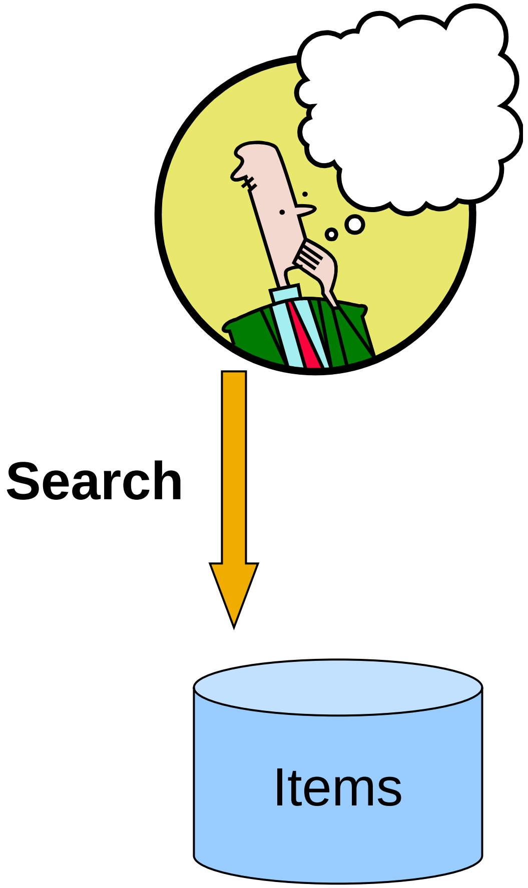
movielens
helping you find the *right* movies

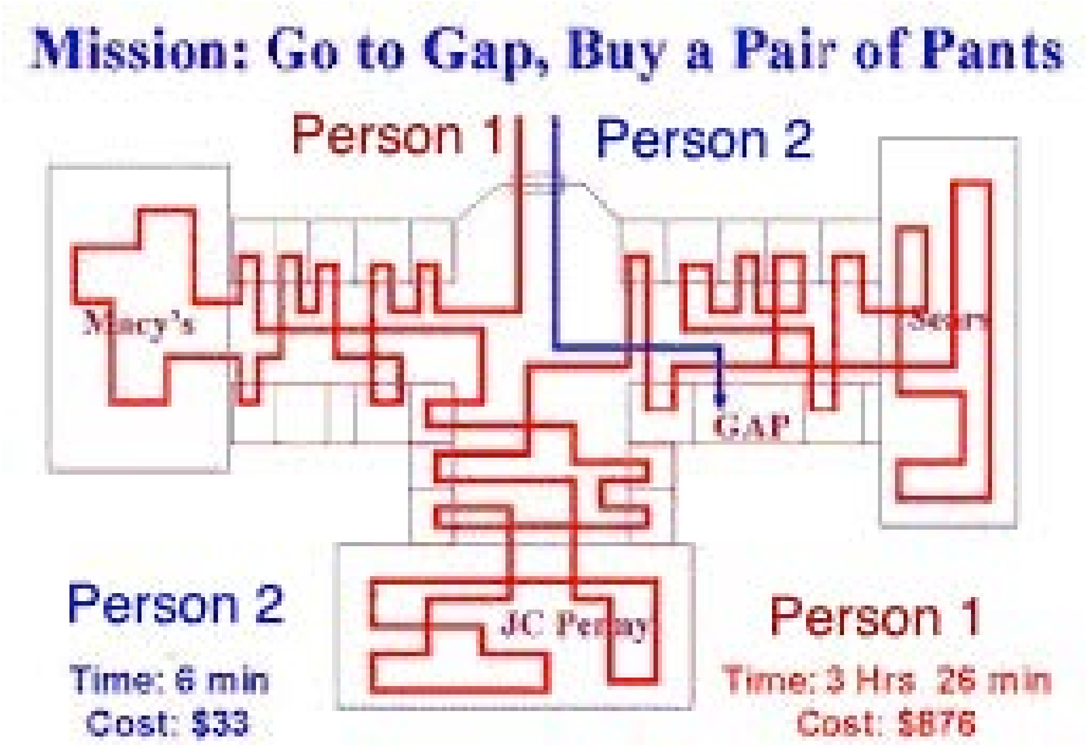
last.fm
the social music revolution

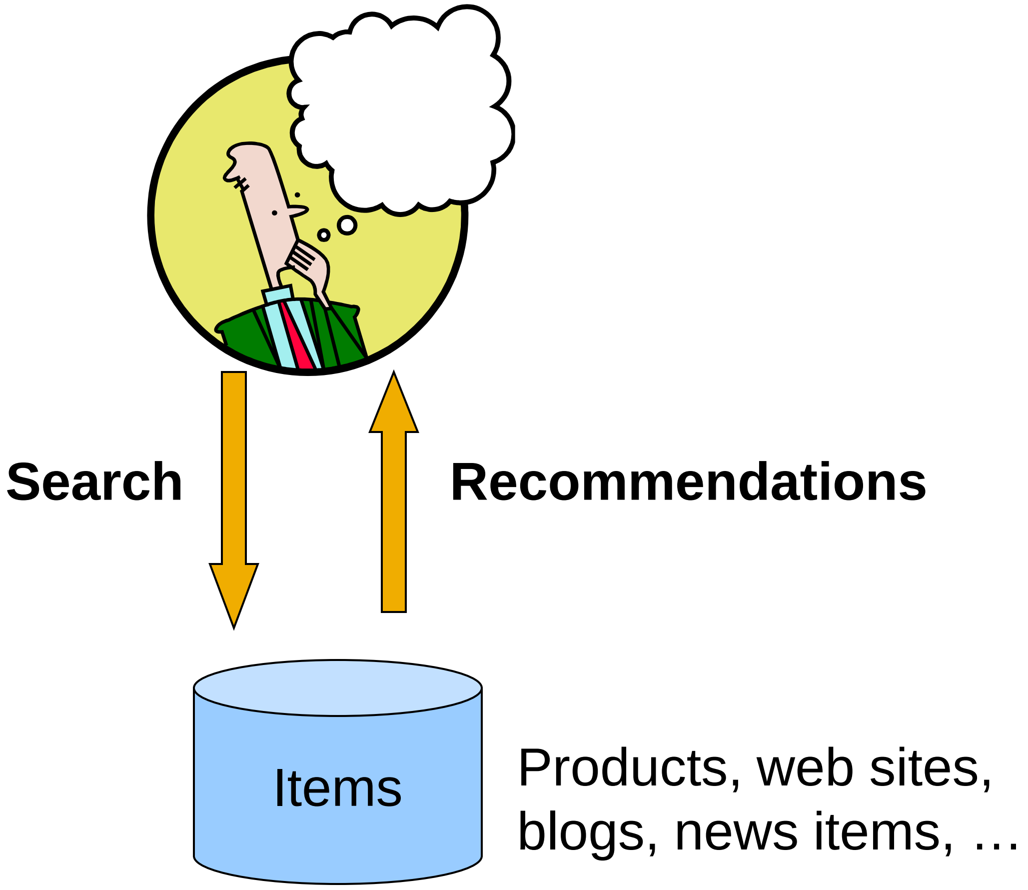
Google News

YouTube

XBOX LIVE


Search
Items


Mission: Go to Gap, Buy a Pair of Pants
Person 1
Person 2
GAP
Person 2
Time: 6 min
Cost: $33
Person 1
Time: 3 Hrs 26 min
Cost: $876

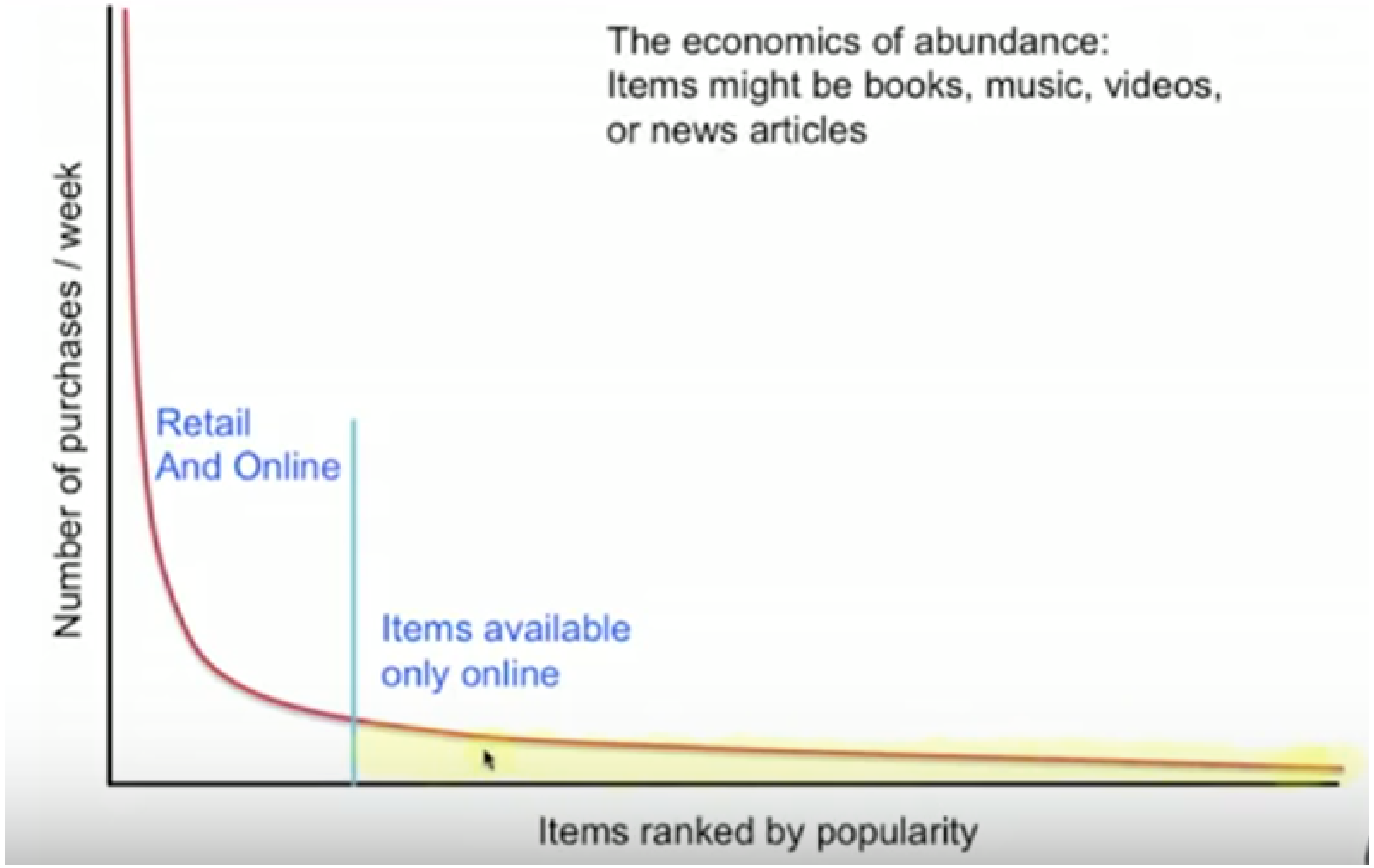


Examples:

amazon.com
PANDORA
StumbleUpon
del.icio.us
NETFLIX
movielens
helping you find the *right* movies
last.fm
the social music revolution
Google News
YouTube
XBOX LIVE

From Scarcity to Abundance

- **Shelf space is a scarce commodity for traditional retailers**
  - Also: TV networks, movie theaters,…
- **Web enables near-zero-cost dissemination of information about products**
  - From scarcity to abundance
  - Long Tail phenomenon.


The economics of abundance:
Items might be books, music, videos, or news articles
Number of purchases / week
Retail And Online
Items available only online
Items ranked by popularity

Example

- Books, movies, music, news articles
- People (friend recommendations on facebook, Linkedln, and Twitter)

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

Types of Recommendations

- **Editorial and hand curated**
  - List of favorites
  - Lists of “essential” items
- **Simple aggregates**
  - Top 10, Most Popular, Recent Uploads
- **Personalized to individual users**
  - Amazon, Netflix, …

Terms

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

- **Utility function** u: $X \times S \to R$
  - R = set of ratings
  - 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 Issues

- **Gathering "known" ratings for matrix**
  - How to collect the data in the utility matrix
- **Derive 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
- **Evaluating methods**
  - How to measure success/performance of recommendation methods

1. Gathering Ratings

- **Explicit**
  - Ask people to rate items
  - Doesn't scale: only a small fraction of users leave ratings and reviews.
  - Crowdsourcing: Pay people to label items
- **Implicit**
  - Learn ratings from user actions
    - E.g., purchase implies high rating
  - What about low ratings?

2. Deriving Unknown from Known

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 Filtering
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

- same actor(s), director, genre, …

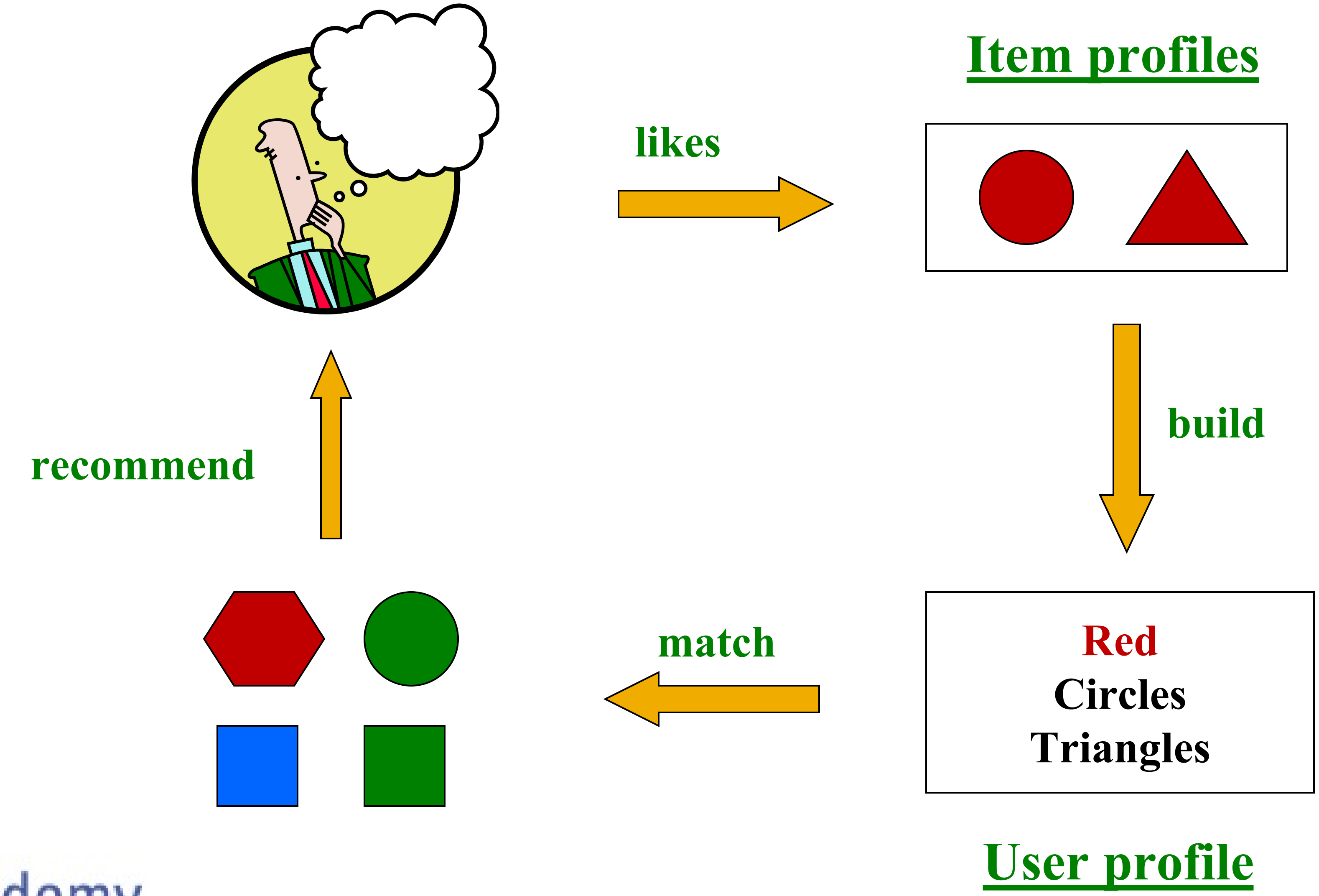
Websites, blogs, video, news

- Articles with “similar” content

People

- People with many common friends.

Plan of Action



Item Profile for Each Item

Profile is a set (vector) of features

- **Movies:** author, title, actor, director,…
- **Text:** Set of "important" words in document
- **People:** Set of friends.

Item profile is a vector

- boolean or real-vlaues

Example: Documents

Profile = set of "important" words in item

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

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

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 Profile

- User has rated items with profiles i_1, ... i_n
- (weighted) average of rated item profiles.
- **Variation:** weight by difference from average rating for item

…

-

(Weighted) Average of Rated Item Profiles

[0, 1]

Items are movies, feature are actor A and actor B

Item Profile: vector with 0 or 1 for each actor

Suppose user X has watched 5 movies:

2 movies featuring actor A
3 movies featuring actor B

[0.4, 0.6]

User profile = mean of item profiles

Feature A's weight = 2/5 = 0.4
Feature B's weight = 3/5 = 0.6

Weight by Difference from Average Rating for Item

Items are movies, feature are actorA and actor B

Item Profile: vector with 0 or 1 for each actor

Suppose user X has watched 5 movies (1-5):

0 2

2 movies featuring actor A: rated 3 and 5

3 movies featuring actor B: rated 1, 2 and 4

-2 -1 1

useful step: Normalize ratings by substracting user's mean rating (3) User profile = mean of item profiles

Feature A's normalized ratings: 0, +2

weight = (0+2)/2 = 1

$[1, -\frac{2}{3}]$

Feature B's normalized ratings: -2, -1, +1

weight = -2/3

User Profile and Prediction

$[1, -\frac{2}{3}]$

Prediction heuristic:

- Given user profile *x* and item profile *i*, estimate

$[1, 0]$

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

$$= \frac{1 \cdot 1 - \frac{2}{3} \cdot 0}{\sqrt{1 + \frac{4}{9}} \cdot 1}$$

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

- 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**

Stay Connected

Dr. Min Chi

Associate Professor
mchi@ncsu.edu
(919) 515-7825

AI Academy

AI Academy

go.ncsu.edu/aiacademy

NC STATE UNIVERSITY