BIG DATA ANALYTICS

Recommendation Systems

ESSEC BUSINESS SCHOOL

Predicting user tastes

- Predicting user responses to options: **A recommendation system**
- Examples:
    - News articles to on-line newspaper readers
    - Suggestions to customers of an on-line retailer

Technology

- **Content-based** systems: properties of the items
  - A Netflix user has watched many cowboy movies $\Longrightarrow$ recommend a "cowboy" movie
- **Collaborative filtering** systems: similarity measures between users and/or items
  - The items recommended to a user are those preferred by similar users

From scarcity to abundance

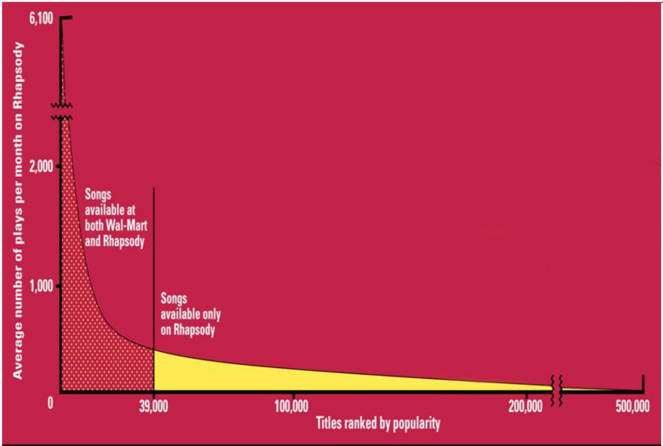
- Shelf space is a scarce commodity for traditional retailers
  - Also: TV networks, movie theaters, ...
- Web enables near – zero – cost dissemination of information about products

From scarcity to abundance

Recommendation in the physical world

- Impossible to tailor the store to each individual customer
- The choice is based on the aggregate numbers:
  - a bookstore will display only the books that are most popular
  - a newspaper will print only the articles the most people will be interested

The Long Tail



Source: Chris Anderson (2004)

Example

- "Touching the Void": not a big seller in its day
- Another book on the same topic: "Into Thin Air"
- Amazon: a few people bought both books
- Recommending "Touching the Void" to people who bought, or were considering, "Into Thin Air"
- "Touching the Void" became very popular!

The Netflix problem

- Netflix database
  - About half a million users
  - About 18,000 movies
- People rate movies
- Sparsely sampled entries

The Netflix problem

- Netflix database
  - About half a million users
  - About 18,000 movies
- People rate movies
- Sparsely sampled entries

$$
\begin{array}{cc}
 & \text{Movies} \\
\text{Users} & \begin{bmatrix}
\times & & & & \times & \\
 & & \times & \times & & \\
\times & & \times & & & \\
 & \times & & & & \times \\
\times & & & & & \\
 & & \times & \times & &
\end{bmatrix}
\end{array}
$$

Problem

Complete the "Netflix matrix"

The Utility Matrix

- Two classes of entities: users and items
    - Users have preferences for certain items
    - Preferences must be teased out of the data
- **A utility matrix**: for each user-item pair, a value of the degree of preference of that user for that item
    - Values from an ordered set (e.g., integers $\{1, \ldots, 5\}$ = the number of stars)
    - Matrix is sparse: that most entries are "unknown"

Utility (Preference) Matrix

	Harry Potter 1	Harry Potter 2	Harry Potter 3	Twilight	Star Wars 1	Star Wars 2	Star Wars 3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Rows: Users
Columns: Movies (in general Items)
Values: The rating of the user for the movie

How can we fill the empty entries of the matrix?

Filling Utility Matrix?

- It is not necessary to predict every blank entry
- For each row: discover some entries that are likely to be high
- **Find a large subset of items with the highest ratings**

Populating the Utility Matrix I

- Acquiring data from utility matrix is difficult
- **Ask users to rate items**
  - Movie ratings
  - On-line stores
  - YouTube
- Users are unwilling to provide responses
- The information is biased

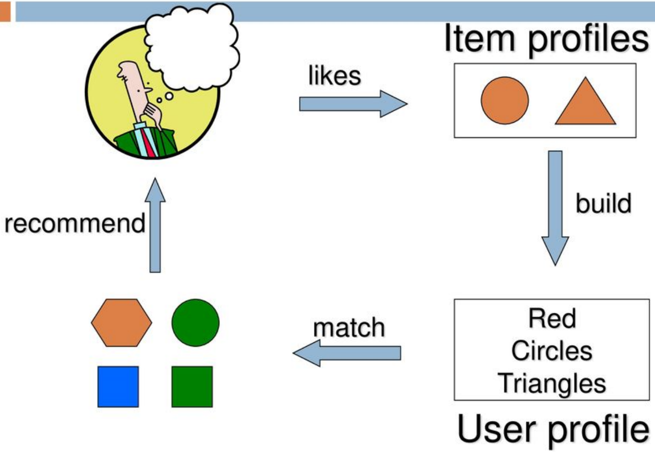
Populating the Utility Matrix II

- Make inferences from users' behavior:
  - If a user buys a product $\Longrightarrow$ user "likes" this item
  - Rating system with one value: 1
- Also infer interest from behavior other than purchasing:
  - e. g. if a customer views information about an item

Content-Based Recommendations

- Main idea: Recommend items to customer x similar to previous items rated highly by x
- Movie recommendations:
  - movies with same actor(s), director, genre, ...
- Websites, blogs, news
  - other sites with "similar" content

Plan of action



Movies Profiles

Construct for each item a profile with important characteristics

- For a movie:
    - The set of actors of the movie
    - The director
    - The year in which the movie was made
    - The genre or general type of movie
    - Other features of movies?

Item Profiles

- The information available from descriptions of movies
  - Movie reviews generally assign a genre to a movie
  - Internet Movie Database (IMDB)
- Obtaining features from available data:
  - Descriptions written by the manufacturer (e.g., the screen size and cabinet color for a TV)
  - Books $\sim$ movies
  - Music products: artist, composer, and genre.

Recommendation system for documents?

- Other classes of items?
- Many kinds of documents for which a recommendation system can be useful:
  - News articles
  - Web pages want to see?
  - Blogs

Discovering Features of Documents

- Characterizing the topic of a document
  - eliminate stop words
  - compute the TF.IDF score for each word in the document
  - the ones with the highest scores are the words that characterize the document
- The features of a document = the n words with the highest TF.IDF scores
  - same n for all documents
  - n a fixed percentage of the words in the document
  - all words whose TF.IDF scores are above a given threshold

Topic based similarity

- Documents: sets of words that express the subjects or main ideas of the document
- Measures of similarity between two documents:
  - the Jaccard distance between the sets of words
  - the cosine distance between the sets, treated as vectors.

Two Kinds of Document Similarity

- Previous lecture: a method for finding documents that were "similar", using shingling, minhashing, and LSH
- Lexical notion of similarity – documents are similar if they contain large, identical sequences of characters
- Recommendation systems: the occurrences of important words in both documents
- The methodology for finding similar documents remains almost the same

Discovering Features of Images

- How to get features for a database of images?
- Images: an array of pixels
- Simple properties of pixels, e.g. the average amount of red are useless

Tagging pictures

- Obtain information by inviting users to tag the items by entering words that describe the image
- A picture with a lot of red:
  - "Tiananmen Square" or
  - "Sunset at Malibu".

Obtaining Item Features From Tags

- One of the earliest attempts: the site del.icio.us
  - a method of search where users entered a set of tags as their search query
- We can use the tags for recommendation:
  - A user retrieves or bookmarks many pages with a certain set of tags
  - We can recommend other pages with the same tags

Tags: not always feasible

- Only works if users are willing to take the trouble to create the tags
- We need enough tags
  - occasional erroneous may bias the system

Creating User's Profile

- We need to create vectors that describe the user's preferences
- Utility matrix: connection between users and items
  - user purchases
  - ratings
- Aggregation of the profiles of items that user likes: :
  - e.g., the average of the components for which the utility matrix has 1's

Example

- Items are movies
- Boolean profiles with components corresponding to actors
- The utility matrix has a 1 if the user has seen the movie and is blank otherwise
- 20% of the movies that user U likes have Julia Roberts as one of the actors
- $\implies$ the user profile for U will have 0.2 in the component for Julia Roberts.

User Profiles from ratings

- The utility matrix: ratings 1 – 5
- Weight the vectors of the profiles of items by the utility value
- $+$ normalize the utilities by subtracting the average value for a user:
  - negative weights for items with a below-average rating
  - positive weights for items with above-average ratings.

Example

- Movies: the utility matrix has entries that are ratings in the 1 – 5 range
- Suppose user U gives an average rating of 3
- Three movies with Julia Roberts as an actor: ratings of 3, 4, and 5
- The user profile of U : the component for Julia Roberts $=$ average of 3 - 3, 4 - 3, and 5 - 3 $=$ 1.

Predictions

- Given profile vectors for both users x and items v
  - Estimate the degree to which a user would prefer an item: $\cos(x, v) = x \cdot v/|x||v|$
  - **Cosine distance** $= 1 - \cos(x, v)$
- Example:
  - In the user's profile components for actors = likelihood that the actor will appear in a movie the user likes
  - **the highest recommendations = lowest cosine distance between the user's and item's vectors**
  - movies with lots of actors that appear in many of the movies the user likes

Pros: content based approach

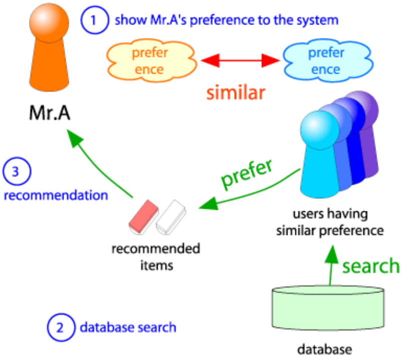
+ No need for data on other users
  - No cold--start or sparsity problems
+ Able to recommend to users with unique tastes
+ Able to recommend new & unpopular items
+ Able to provide explanations
  - listing content--features that caused an item to be recommended

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**

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



Collaborative Filtering

- **Focus on the similarity of the user ratings for two items**
- The item-profile vector for an item is replaced by its column in the utility matrix
- Users: rows in the utility matrix
  - Users are similar if their vectors are close according to some distance measure
- Recommendation for a user U:
  - look at the users that are most similar to U
  - recommend items that these users like
- Identifying similar users and recommending what similar users like: **collaborative filtering**

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

Item-Item CF (|N|=2)

users / movies

movies \ users	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		**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-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		**2.6**	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:

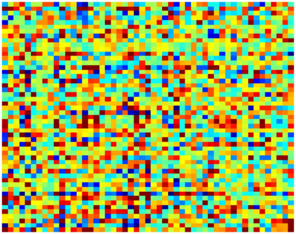
$r_{1.5} = (0.41*2 + 0.59*3) / (0.41+0.59) = 2.6$

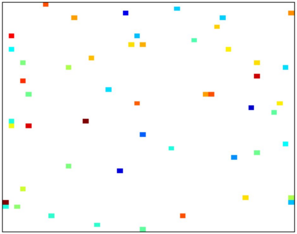
$$r_{ix} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{jx}}{\sum s_{ij}}$$

CF: common practice

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

Matrix Completion





Problem

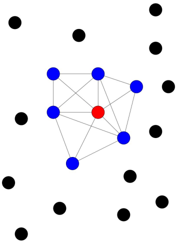
Infer missing entries

Global positioning from local distances

- Points $\{x_j\}_{1 \le j \le n} \in \mathbb{R}^d$
- Partial information about distances $M_{ij} = \|x_i - x_j\|$

Example (Singer, Biswas et al.)

- Low-powered wirelessly networked sensors
- Each sensor can construct a distance estimate from nearest neighbor



Global positioning from local distances

- Points $\{x_j\}_{1 \le j \le n} \in \mathbb{R}^d$
- Partial information about distances $M_{ij} = \|x_i - x_j\|$

Example (Singer, Biswas et al.)

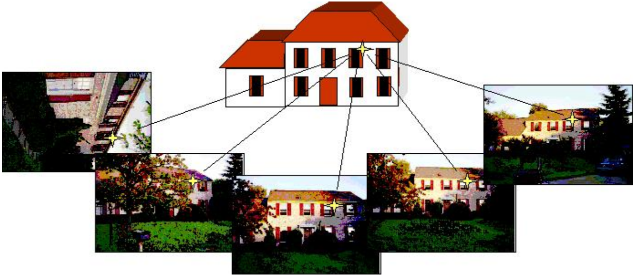
- Low-powered wirelessly networked sensors
- Each sensor can construct a distance estimate from nearest neighbor

$$\begin{array}{cc} & i \\ j & \begin{bmatrix} \times & & & & \times & \\ & & \times & \times & & \\ \times & & \times & & & \\ & \times & & & & \times \\ \times & & & & & \\ & \times & & \times & & \end{bmatrix} \end{array}$$

Problem

Locate the sensors

Structure-from-motion problem



Problem

Recover 3D shape from 2D images

Structure-from-motion problem

- P features over F frames
- (x_{fp}, y_{fp}) = position of feature p at frame f
- $2F \times P$ measurement matrix

$$\begin{bmatrix} x_{11} & \cdots & x_{1P} \\ & \cdots & \\ x_{F1} & \cdots & x_{FP} \\ y_{11} & \cdots & y_{1P} \\ & \cdots & \\ y_{F1} & \cdots & y_{FP} \end{bmatrix}$$

Structure-from-motion problem

- P features over F frames
- (x_{fp}, y_{fp}) = position of feature p at frame f
- W a $2F \times P$ measurement matrix
- Occlusions $\rightarrow W$ partially filled in

$$\begin{bmatrix} \times & ? & ? & ? & \times & ? \\ ? & ? & \times & \times & ? & ? \\ \times & ? & \times & ? & ? & ? \\ ? & \times & ? & ? & ? & \times \\ \times & ? & ? & ? & ? & ? \\ ? & \times & ? & \times & ? & ? \end{bmatrix}$$

Problem

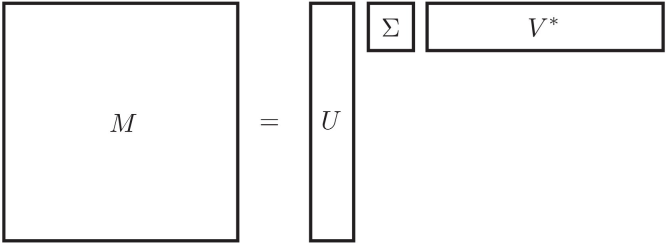
Recover the missing measurements

Low-dimensional structure

Engineering/scientific applications: unknown matrix has often (approx.) low rank

- Netflix matrix
- Sensor-net matrix: $\|x_i - x_j\|^2$, $\{x_i\} \in \mathbb{R}^d$
  - rank 2 if $d = 2$
  - rank 3 if $d = 3$
  - $\ldots$
- Many others (e.g. machine learning, computer vision ...)

Dimension reduction



$M \in \mathbb{R}^{m_1 \times m_2}$ of rank r depends upon $(m_1 + m_2 - r)r$ free parameters

- $r \ll \min(m_1, m_2) \Rightarrow (m_1 + m_2 - r)r \ll m_1 m_2$
- Completion impossible if $n < (m_1 + m_2 - r)r$

Trace - norm heuristics

$$\begin{array}{ll} \textbf{minimize} & \operatorname{rank}(A) \\ \textbf{subject to} & \mathcal{O}_n(A) = b \end{array}$$

$\mathcal{O}_n$ linear transformation supplying information about M
(e.g. $\mathcal{O}_n(M) = \{M_{ij}\}_{(i,j)\in E}$, E subset of entries)

(Usually) NP-hard

Trace - norm heuristics

$$\begin{array}{ll} \textbf{minimize} & \mathrm{rank}(A) \\ \textbf{subject to} & \mathcal{O}_n(A) = b \end{array}$$

$\mathcal{O}_n$ linear transformation supplying information about M (e.g. $\mathcal{O}_n(M) = \{M_{ij}\}_{(i,j)\in E}$, E subset of entries)

(Usually) NP-hard

Convex relaxation of the rank minimization program $\rightarrow$ trace norm heuristics: Fazel (2002), Hindi, Boyd and Fazel (2001)

$$\begin{array}{ll} \textbf{minimize} & \|A\|_* \\ \textbf{subject to} & \mathcal{O}_n(A) = b \end{array}$$

Trace norm ($\sigma_i(A)$ is ith largest singular value of A)

$$\|A\|_* = \Sigma\, \sigma_i(A).$$

Non-noisy case

- Candès/Recht (2008), Candès/Tao (2009)
- Gross (2009), Recht (2009)
- Different approach Keshavan et al (2009) (OPTSPACE)

Recht (2009)

Exact reconstruction with high probability if

$$n > C \log^2(m)\,(m_1 + m_2)\mathrm{rank}(M)$$

$m = \min\{m_1, m_2\}$.

Exact reconstruction: conditions

- **Sampling uniformly at random**
- **"Incoherence" condition**:

$A \in \mathbb{R}^{m_1 \times m_2} = U\,DV^T$, $\mathrm{rank}(A) = r$, $\nu = O(1)$ and $d = \max(m_1, m_2)$

$$\left\|U^T e_i\right\|^2 \leq \frac{\nu r}{d}, \quad \left\|V^T e_i\right\|^2 \leq \frac{\nu r}{d}$$

and

$$\left|U\,V^T\right|_{ij}^2 \leq \frac{\nu r}{d^2}$$

(intuition: column and row spaces cannot be aligned with basis vectors)

Pros/Cons Collaborative Filtering

+ **Works for ant kind of items**
  - 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
– **Popularity bias:**
  - Cannot recommend items to someone with unique taste
  - Tends to recommend popular items

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

References

- J. Leskovec, A. Rajaraman and J. D. Ullman *Mining of Massive Datasets* (2014), Chapter 9 and 11 (SVD)
- Candès, E.J. and Recht, B. (2009) Exact matrix completion via convex optimization. *Fondations of Computational Mathematics*, **9(6)**, 717-772.
- O. Klopp. Noisy low-rank matrix completion with general sampling distribution.
  Bernoulli, 20(1):282–303, 2014.