

Data Mining for Web-based Applications

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

1

“Association Rules”

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

2

The Market-Basket Model

- ◆ A large set of items,
 - ▶ e.g., things sold in a supermarket.
- ◆ A large set of baskets, each of which is a small set of the items,
 - ▶ e.g., the things one customer buys on one day.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

3

Support

- ◆ Simplest question: find sets of items that appear “frequently” in the baskets.
- ◆ Support for itemset I = the number of baskets containing all items in I .
- ◆ Given a support threshold s , sets of items that appear in $> s$ baskets are called frequent itemsets.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

4

Example

◆ Items = {milk, coke, pepsi, beer, juice}.

◆ Support = 3 baskets.

- ▶ B1 = {m, c, b} B2 = {m, p, j}
- ▶ B3 = {m, b} B4 = {c, j}
- ▶ B5 = {m, p, b} B6 = {m, c, b, j}
- ▶ B7 = {c, b, j} B8 = {b, c}

◆ Frequent itemsets:

- ▶ {m}, {c}, {b}, {j}, {m, b}, {c, b}, {j, c}.

Based on notes from "CS345, Autumn 2006: Data Mining"
<http://www.stanford.edu/class/cs345a>

5

Applications --- (1)

◆ Real market baskets: chain stores keep terabytes of information about what customers buy together.

- ▶ Tells how typical customers navigate stores, lets them position tempting items.
- ▶ Suggests tie-in "tricks," e.g., run sale on diapers and raise the price of beer.

◆ High support needed, or no \$\$'s .

Based on notes from "CS345, Autumn 2006: Data Mining"
<http://www.stanford.edu/class/cs345a>

6

Applications --- (2)

- ◆ “Baskets” = documents; “items” = words in those documents.
 - ▶ Lets us find words that appear together unusually frequently, i.e., linked concepts.
- ◆ “Baskets” = sentences, “items” = documents containing those sentences.
 - ▶ Items that appear together too often could represent plagiarism.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

7

Applications --- (3)

- ◆ “Baskets” = Web pages; “items” = linked pages.
 - ▶ Pairs of pages with many common references may be about the same topic.
- ◆ “Baskets” = Web pages p ; “items” = pages that link to p .
 - ▶ Pages with many of the same links may be mirrors or about the same topic.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

8

Important Point

- ◆ “Market Baskets” is an abstraction that models any many-many relationship between two concepts: “items” and “baskets.”
 - ▶ Items need not be “contained” in baskets.
- ◆ The only difference is that we count co-occurrences of items related to a basket, not vice-versa.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

9

Scale of Problem

- ◆ WalMart sells 100,000 items and can store billions of baskets.
- ◆ The Web has over 100,000,000 words and billions of pages.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

10

Association Rules

- ◆ If-then rules about the contents of baskets.
 - ▶ $\{i_1, i_2, \dots, i_k\} \rightarrow j \approx$ "if a basket contains all of i_1, \dots, i_k then it is likely to contain j ."
 - ▶ Confidence of this association rule \approx the probability of j given i_1, \dots, i_k .
 - ▶ Interest of this association rule \approx the absolute value of the amount by which the confidence differs from the probability of j .

Based on notes from "CS345, Autumn 2006: Data Mining"
<http://www.stanford.edu/class/cs345a>

11

Example

- | | |
|-----------------------|------------------------|
| ▶ $B_1 = \{m, c, b\}$ | $B_2 = \{m, p, j\}$ |
| ▶ $B_3 = \{m, b\}$ | $B_4 = \{c, j\}$ |
| ▶ $B_5 = \{m, p, b\}$ | $B_6 = \{m, c, b, j\}$ |
| ▶ $B_7 = \{c, b, j\}$ | $B_8 = \{b, c\}$ |

- ◆ An association rule:
 - ▶ $\{m, b\} \rightarrow c$.
 - ▶ Confidence = $2/4 = 50\%$.
 - ▶ Interest = $|2/4 - 5/8| = 1/8$ (not very interesting)

Based on notes from "CS345, Autumn 2006: Data Mining"
<http://www.stanford.edu/class/cs345a>

12

Relationships Among Measures

- ◆ Rules with high support and confidence may be useful even if they are not “interesting.”
 - ▶ We don’t care if buying bread causes people to buy milk, or whether simply a lot of people buy both bread and milk.
- ◆ But high interest suggests a cause that might be worth investigating.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

13

Finding Association Rules

- ◆ A typical question: “find all association rules with support $\geq s$ and confidence $\geq c$.”
 - ▶ Note: “support” of an association rule is the support of the set of items it mentions.
- ◆ Hard part: finding the high-support (frequent) itemsets.
 - ▶ Checking the confidence of association rules involving those sets is relatively easy.

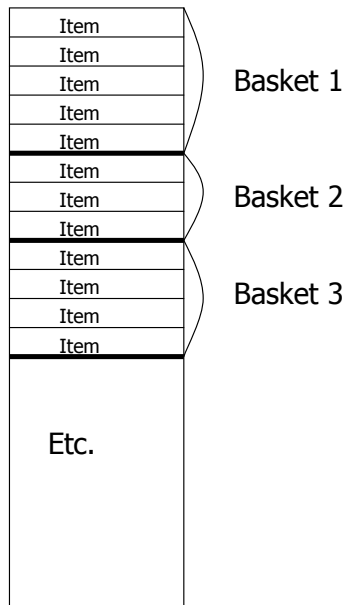
Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

14

Computation Model

- ◆ Typically, data is kept in a “flat file” rather than a database system.

- ▶ Stored on disk.
- ▶ Stored basket-by-basket.
- ▶ Expand baskets into pairs, triples, etc. as you read baskets.



- ◆ The true cost of mining disk-resident data is usually the number of disk I/O's.
- ◆ In practice, association-rule algorithms read the data in passes --- all baskets read in turn.
- ◆ Thus, we measure the cost by the number of passes an algorithm takes.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

15

Main-Memory Bottleneck

- ◆ For many frequent-itemset algorithms, main memory is the critical resource.
 - ▶ As we read baskets, we need to count something, e.g., occurrences of pairs.
 - ▶ The number of different things we can count is limited by main memory.
 - ▶ Swapping counts in/out is a disaster.
- ◆ The hardest problem often turns out to be finding the frequent pairs.
- ◆ We'll concentrate on how to do that, then discuss extensions to finding frequent triples, etc.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

16

Naïve Algorithm

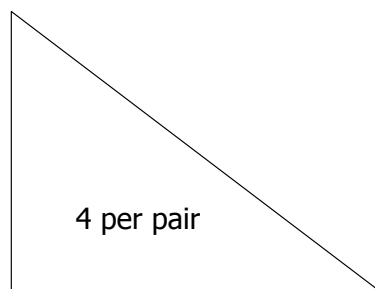
- ◆ Read file once, counting in main memory the occurrences of each pair.
 - ▶ Expand each basket of n items into its $n(n-1)/2$ pairs.
- ◆ Fails if $(\#items)^2$ exceeds main memory.
 - ▶ Remember: $\#items$ can be 100K (Wal-Mart) or 10B (Web pages).

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

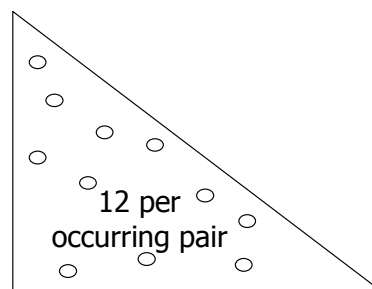
17

Details of Main-Memory Counting

- ◆ Two approaches:
 - ▶ Count all item pairs, using a triangular matrix.
 - ▶ Keep a table of triples $[i, j, c]$ = the count of the pair of items $\{i, j\}$ is c .
- ◆ (1) requires only (say) 4 bytes/pair.
- ◆ (2) requires 12 bytes, but only for those pairs with count > 0 .



Method (1)



Method (2)

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

18

Details of Approach #1

- ◆ Number items 1, 2, ...
- ◆ Keep pairs in the order $\{1,2\}, \{1,3\}, \dots, \{1,n\}, \{2,3\}, \{2,4\}, \dots, \{2,n\}, \{3,4\}, \dots, \{3,n\}, \dots, \{n-1,n\}$.
- ◆ Find pair $\{i, j\}$ at the position $(i-1)(n-i/2) + j - i$.
- ◆ Total number of pairs $n(n-1)/2$; total bytes about $2n^2$.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

19

Details of Approach #2

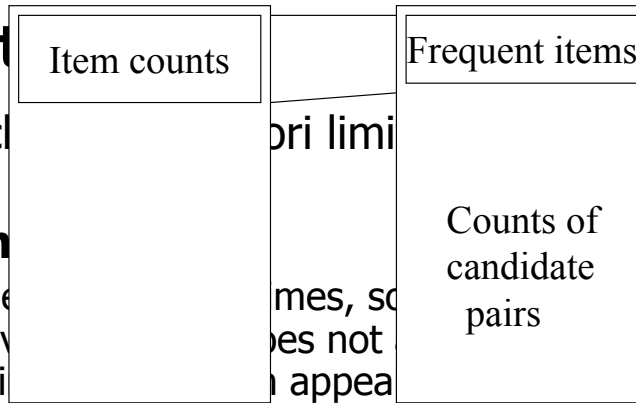
- ◆ You need a hash table, with i and j as the key, to locate (i, j, c) triples efficiently.
 - ▮ Typically, the cost of the hash structure can be neglected.
- ◆ Total bytes used is about $12p$, where p is the number of pairs that actually occur.
 - ▮ Beats triangular matrix if at most $1/3$ of possible pairs actually occur.

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

20

A-Priori Algorithm

- ◆ A two-pass approach for main memory.
- ◆ Key idea: **monotonicity**
 - ▶ if a set of items appears in a basket, then any subset of it also appears in that basket; Contrapositive: if a set of items does not appear in a basket, then no superset of it appears in that basket.
- ◆ Pass 1: Read baskets and count in main memory the occurrences of each item.
 - ▶ Requires only memory proportional to #items.
- ◆ Pass 2: Read baskets again and count in main memory only those pairs both of which were found in Pass 1 to be frequent.
 - ▶ Requires memory proportional to square of frequent items only.



Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

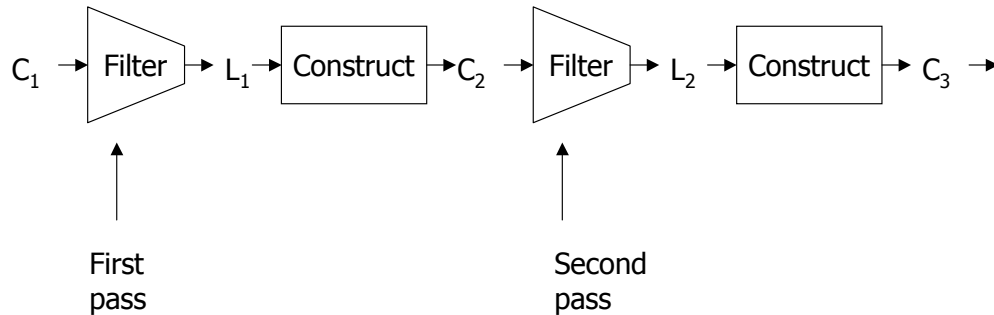
21

Detail for A-Priori

- ◆ You can use the triangular matrix method with n = number of frequent items.
 - ▶ Saves space compared with storing triples.
- ◆ Trick: number frequent items 1,2,... and keep a table relating new numbers to original item numbers.

Frequent Triples, Etc.

- ◆ For each k , we construct two sets of k -tuples:
 - ▶ C_k = candidate k -tuples = those that might be frequent sets (support $> s$) based on information from the pass for $k-1$.
 - ▶ L_k = the set of truly frequent k -tuples.



Based on notes from "CS345, Autumn 2006: Data Mining"
<http://www.stanford.edu/class/cs345a>

23

A-Priori for All Frequent Itemsets

- ◆ One pass for each k .
- ◆ Needs room in main memory to count each candidate k -tuple.
- ◆ For typical market-basket data and reasonable support (e.g., 1%), $k = 2$ requires the most memory.

Based on notes from "CS345, Autumn 2006: Data Mining"
<http://www.stanford.edu/class/cs345a>

24

Frequent Itemsets --- (2)

- ◆ C_1 = all items
- ◆ L_1 = those counted on first pass to be frequent.
- ◆ C_2 = pairs, both chosen from L_1 .
- ◆ In general, C_k = k -tuples, each $k-1$ of which is in L_{k-1} .
- ◆ L_k = members of C_k with support $\geq s$.

Based on notes from "CS345, Autumn 2006: Data Mining"
<http://www.stanford.edu/class/cs345a>

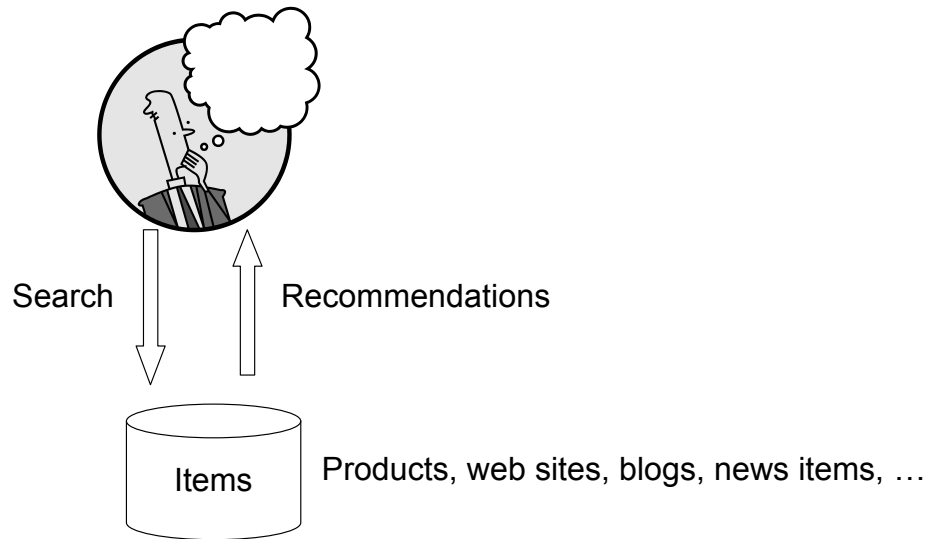
25

"Recommendation Systems"

Based on notes from "CS345, Autumn 2006: Data Mining"
<http://www.stanford.edu/class/cs345a>

26

Recommendations



Based on notes from "CS345, Autumn 2006: Data Mining"
<http://www.stanford.edu/class/cs345a>

27

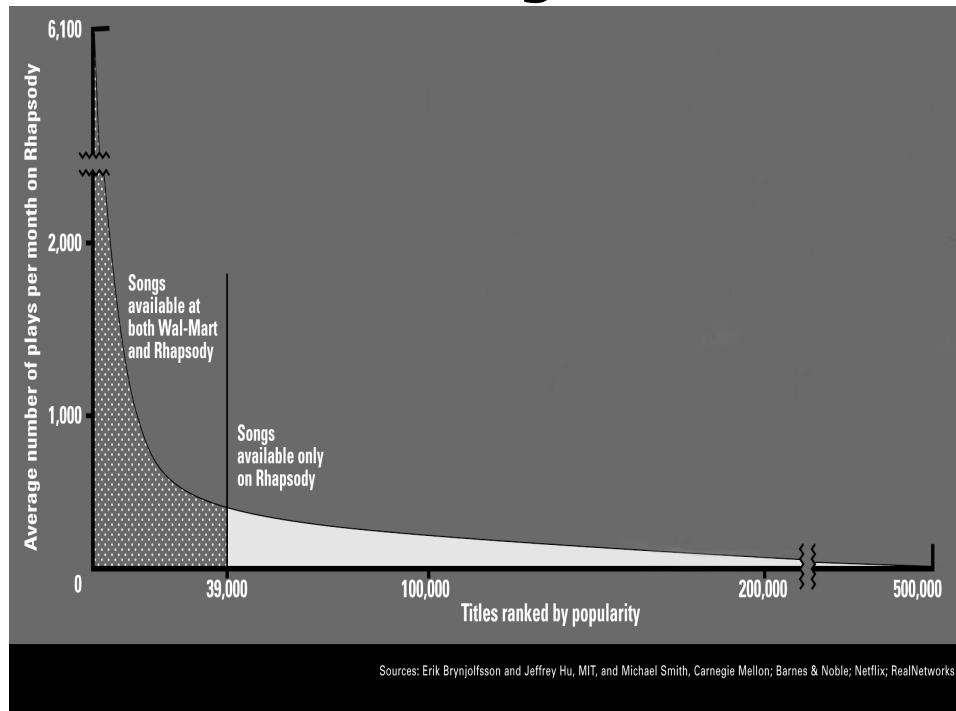
From scarcity to abundance

- ◆ Shelf space is a scarce commodity for traditional retailers
 - ▶ Also: TV networks, movie theaters,...
- ◆ The 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

Based on notes from "CS345, Autumn 2006: Data Mining"
<http://www.stanford.edu/class/cs345a>

28

The Long Tail



Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

29

Recommendation Types

- ◆ Editorial
 - ◆ Simple aggregates
 - ▶ Top 10, Most Popular, Recent Uploads
- ◆ Tailored to individual users
 - ▶ Amazon, Netflix, ...

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

30

Formal Model

- ◆ C = set of Customers
- ◆ S = set of Items
- ◆ Utility function $u: C \times S \rightarrow R$
 - ▶ R = set of ratings
 - ▶ R is a totally ordered set
 - ▶ e.g., 0-5 stars, real number in $[0,1]$

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

31

Utility Matrix

	King Kong	LOTR	Matrix	Nacho Libre
Alice	1		0.2	
Bob		0.5		0.3
Carol	0.2		1	
David				0.4

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

32

Key Problems

- ◆ Gathering “known” ratings for matrix
- ◆ Extrapolate unknown ratings from known ratings
 - ▶ Mainly interested in high unknown ratings
- ◆ Evaluating extrapolation methods

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

33

Gathering Ratings

- ◆ Explicit
 - ▶ Ask people to rate items
 - ▶ Doesn’t work well in practice – people can’t be bothered
- ◆ Implicit
 - ▶ Learn ratings from user actions
 - e.g., purchase implies high rating
 - ▶ What about low ratings?

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

34

Extrapolating Utilities

- ◆ Key problem: matrix U is sparse
 - ▶ most people have not rated most items
- ◆ Three approaches
 - ▶ Content-based
 - ▶ Collaborative
 - ▶ Hybrid

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

35

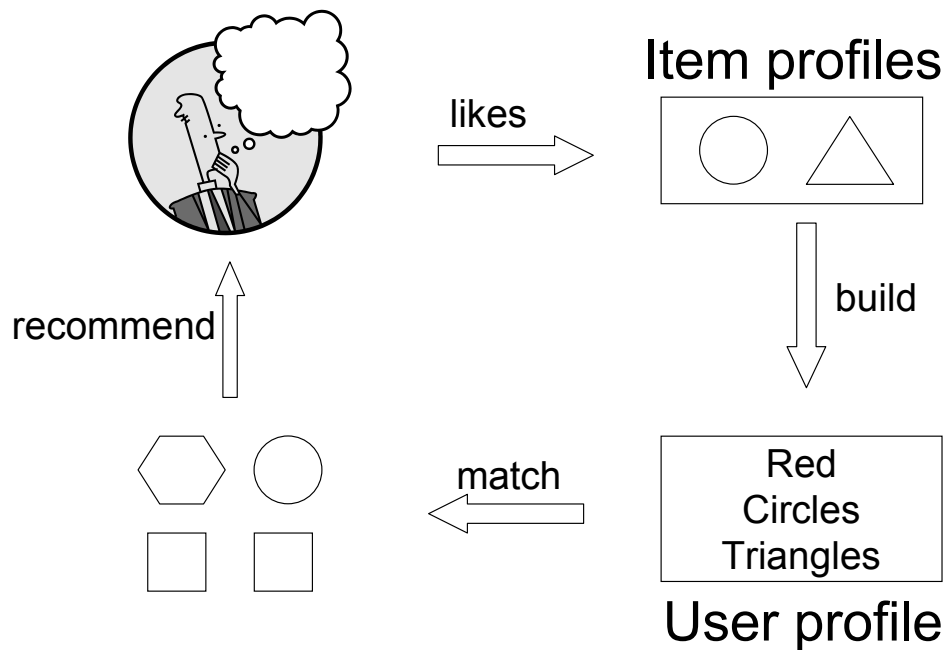
Content-based recommendations

- ◆ Main idea: recommend items to customer C similar to previous items rated highly by C
- ◆ Movie recommendations
 - ▶ recommend movies with same actor(s), director, genre, ...
- ◆ Websites, blogs, news
 - ▶ recommend other sites with “similar” content

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

36

Plan of action



Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

37

Item Profiles

- ◆ For each item, create an item profile
- ◆ Profile is a set of features
 - ▶ movies: author, title, actor, director,...
 - ▶ text: set of “important” words in document
- ◆ How to pick important words?
 - ▶ Usual heuristic is TF.IDF (Term Frequency times Inverse Doc Frequency)

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

38

TF.IDF

f_{ij} = frequency of term t_i in document d_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 $w_{ij} = TF_{ij} * IDF_i$

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

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

39

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 c and item profile s , estimate $u(c,s) = \cos(c,s) = c \cdot s / (|c| |s|)$
- ▶ Need efficient method to find items with high utility: later

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

40

Model-based approaches

- ◆ For each user, learn a classifier that classifies items into rating classes
 - ▶ liked by user and not liked by user
 - ▶ e.g., Bayesian, regression, SVM
- ◆ Apply classifier to each item to find recommendation candidates
- ◆ Problem: scalability

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

41

Limitations of Content-based recommendations

- ◆ Finding the appropriate features
 - ▶ e.g., images, movies, music
- ◆ Overspecialization
 - ▶ Never recommends items outside user’s content profile
 - ▶ People might have multiple interests
- ◆ Recommendations for new users
 - ▶ How to build a profile?

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

42

Collaborative Filtering

- ◆ Consider user c
- ◆ Find set D of other users whose ratings are “similar” to c ’s ratings
- ◆ Estimate user’s ratings based on ratings of users in D

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

43

Similar users

- ◆ Let r_x be the vector of user x ’s ratings
- ◆ Cosine similarity measure
 - ▶ $\text{sim}(x, y) = \cos(r_x, r_y)$
- ◆ Pearson correlation coefficient
 - ▶ S_{xy} = items rated by both users x and y

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

44

Rating predictions

- ◆ Let D be the set of k users most similar to c who have rated item s
- ◆ Possibilities for prediction function (item s):
 - ▶ $r_{cs} = 1/k \sum_{d \in D} r_{ds}$
 - ▶ $r_{cs} = (\sum_{d \in D} \text{sim}(c,d) r_{ds}) / (\sum_{d \in D} \text{sim}(c,d))$
 - ▶ Other options?
- ◆ Many tricks possible...

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

45

Complexity

- ◆ Expensive step is finding k most similar customers
 - ▶ $O(|U|)$
- ◆ Too expensive to do at runtime
 - ▶ Need to pre-compute
- ◆ Naïve precomputation takes time $O(N|U|)$
 - ▶ Simple trick gives some speedup
- ◆ Can use clustering, partitioning as alternatives, but quality degrades

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

46

Item-Item Collaborative Filtering

- ◆ So far: User-user collaborative filtering
- ◆ Another view
 - ▶ For item s , find other similar items
 - ▶ Estimate rating for item based on ratings for similar items
 - ▶ Can use same similarity metrics and prediction functions as in user-user model
- ◆ In practice, it has been observed that item-item often works better than user-user

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

47

Pros and cons of collaborative filtering

- ◆ Works for any kind of item
 - ▶ No feature selection needed
- ◆ New user problem
- ◆ New item problem
- ◆ Sparsity of rating matrix
 - ▶ Cluster-based smoothing?

Based on notes from “CS345, Autumn 2006: Data Mining”
<http://www.stanford.edu/class/cs345a>

48

Hybrid Methods

- ◆ Implement two separate recommenders and combine predictions
- ◆ Add content-based methods to collaborative filtering
 - ▶ item profiles for new item problem
 - ▶ demographics to deal with new user problem
 - ▶ Filterbots