2nd ACM International Conference on Recommender Systems (RecSys 2008)

Context-Aware Recommender Systems

October 23, 2008

Gedas Adomavicius, U. of Minnesota
Alex Tuzhilin, New York University

Quick Note on Terminology

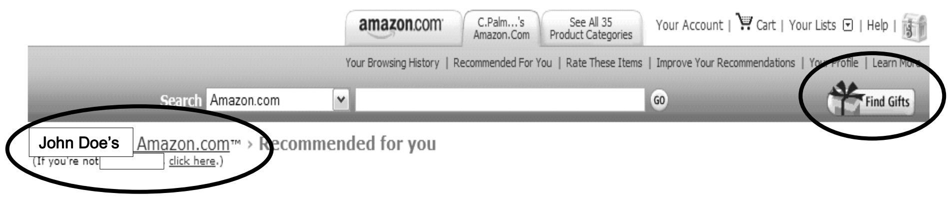
- Several terms have been used to describe recommender systems that can take advantage of contextual information
- Google (as of October 19, 2008)
  - Context-aware
    - "context-aware recommender systems" (113 results)
    - "context-aware recommendations" (151 results)
  - Contextual
    - "contextual recommender systems" (11 results)
    - "contextual recommendations" (775 results)
  - Context-dependent
    - "context-dependent recommender systems" (5 results)
    - "context-dependent recommendations" (17 results)
- In this tutorial, we use "context-aware" and "contextual", but the discussion is welcome regarding the most appropriate term

Motivation

Motivating Examples: Context-Dependent Recommendations

- Recommend a vacation
  - Winter vs. summer
- Recommend a purchase (e-retailer)
  - Gift vs. for yourself
- Recommend a movie
  - To a student who wants to see it on Saturday night with his girlfriend in a movie theater
- Recommendations depend on the *context*
  - It is sometimes important to know not only *what* to recommend to *whom*, but also under what *circumstances*

Rudimentary Contextual Recommendations: Amazon



- Amazon makes sure that
  - It is you
  - Has a "gift" button
- But there is *much more* to capturing and using contexts in recommendations…

Question: Does Context Matter?

- Matters enough for Amazon to add the "gift" button
- C.K. Prahalad, *Beyond CRM*, MWorld/AMA, 2004:
  - C.K. Prahalad predicts: "Customer Context" is the Next "Big Thing"
  - "The ability to reach out and touch customers anywhere at anytime means that companies must deliver not just competitive products but also unique, real-time customer experiences shaped by customer *context*"
- **Goal:** Demonstrate that certain contextual information does matter in *some* recommendation applications
  - E.g., recommending a vacation in the winter or a movie to see on Saturday night with a girlfriend in a movie theater

Outline of the Tutorial

- What is context?
- Incorporating context in recommender systems: a conceptual framework
  - Different paradigms for contextual recommender systems
- Additional capabilities for contextual recommender systems
- Future directions

What Is Context?

What Is Context? *(Palmisano et al. 2008)*

- Conditions or circumstances affecting some thing (Webster)
- 150(!) other definitions from various communities/disciplines
  - Presented in (*Bazire & Brezillon, CONTEXT'05*)
  - *DM/CRM:* those events which characterize the life of a customer and can determine a change in his/her preferences and status, and affect the customer's value for a company (*Berry & Linoff, 1997*)
  - *Context-aware systems:* the location of the user, the identity of people near the user, the objects around, and the changes in these elements (*Schilit & Theimer, 1994*)
  - *Marketing:* the same consumer may use different decision-making strategies and prefer different products or brands under different contexts (*Bettman et al. 1991, Lilien & Kotler 1992, Lussier & Olshavsky 1979, Klein & Yadav 1989, Bettman et al. 1991*)
- *Conclusion*: many different approaches and views!

Context in Context-Aware Systems

- Location of the user, identity of people and objects near the user (*Schilit & Theimer, 1994*)
- Date, season, temperature (*Brown, Bovey, & Chen 1997*)
- Physical and conceptual statuses of interest to a user (*Ryan, Pascoe, & Morse 1997*)
- Any information which can characterize and is relevant to the interaction between a user and an application (*Dey, Abowd & Salber 2001*)

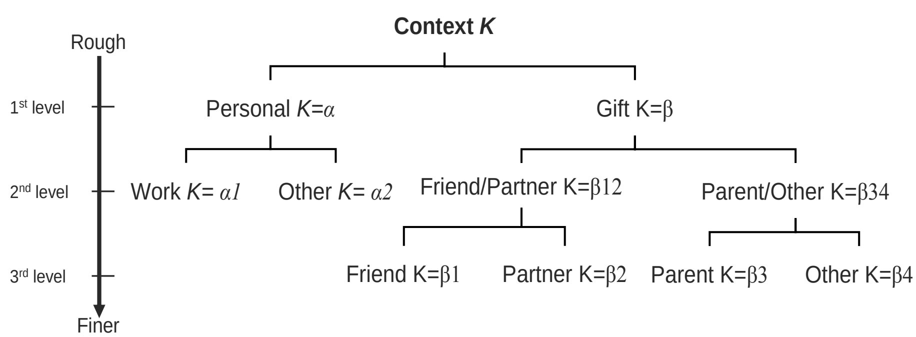
What Is Context in *Recommender Systems*?

- Additional information, besides information on *Users* and *Items*, that is *relevant* to recommendations
- Relevant in
  - Identifying pertinent subsets of data when computing recommendations
  - Building richer rating estimation models
  - Providing various types of constraints on recommendation outcomes
- Examples:
  - Exclude gift purchases when recommending products to you
  - Use only *winter-based* ratings when recommending a vacation in the winter

Defining Context via Contextual Variables

- Taxonomy of contextual information
- More vs. less granular context (levels of context)

The contextual information *K* of a purchase:



Example:
Temporal Contextual Variable

- Time-related context can be described using a temporal hierarchy with multiple temporal relationships of varying granularity, e.g.,
  - Time (e.g., 2008.10.19 11:59:59pm) → Date (2008.10.19) → Year (2008)
  - Time (2008.10.19 11:59:59pm) → Hour (11pm) → TimeOfDay (evening)
  - Date (2008.10.19) → Month (October) → Season (Fall)
  - Date (2008.10.19) → DayOfWeek (Sunday) → TimeOfWeek (Weekend)

Formalizing Contextual Information via Contextual Variables

- Formally, contextual information can be defined as a *vector* of contextual variables $c = (c_1,\ldots,c_n)$, where $c_i \in C_i$
  - $C = C_1 \times \ldots \times C_n$ denotes the space of possible values for a given context
    - Each component C_i is represented as a tree: it is defined as a hierarchical set of nodes (concepts)
    - If $c_i \in C_i$, then c_i represents one of the nodes in the hierarchy C_i
  - Example:
    - C = PurchaseContext × TemporalContext
    - c = (work, weekend), i.e., purchasing something for work on a weekend

Obtaining Context

- Explicitly specified by the user
  - E.g., “I want to watch a movie at home with my parents”
- Observed or deduced by the system
  - Time (from system clock)
  - Location (from GPS)
  - Deduced from user’s behavior (e.g., shopping for business or pleasure)
  - Etc.
- How to obtain the context is a separate problem that lies beyond the scope of this tutorial
  - Significant research literature on obtaining, inferring, and predicting context (e.g., for mobile computing)
  - We assume that the context is *given*

Incorporating Context in Recommender Systems: A Conceptual Framework

Traditional Recommendation Problem: Quick Overview

- Two types of entities: *Users* and *Items*
- Utility of item *i* for user *u* is represented by some rating *r* (where $r \in Rating$)
- Each user typically rates a *subset* of items
- Recommender system then tries to estimate the unknown ratings, i.e., to extrapolate rating function *R* based on the known ratings:
  - R: $Users \times Items \rightarrow Rating$
  - I.e., two-dimensional recommendation framework
- The recommendations to each user are made by offering his/her highest-rated items

Rating Estimation Problem

- Multitude of existing traditional 2D recommendation techniques
- They are often classified by:
  - Recommendation approach
    - Content-based, collaborative filtering, hybrid
  - Nature of the prediction technique
    - Heuristic-based, model-based

Traditional Recommender Systems: Content-Based Approaches

- Heuristic approaches
  - Item similarity methods (*Lang 1995; Pazzani & Billsus, 1997; Zhang et al. 2002*)
  - Instance-based learning (*Schwab et al. 2000*)
  - Case-based reasoning (*Smyth 2007*)
- Model-based approaches
  - Classification models (*Pazzani & Billsus 1997; Mooney & Roy 1998*)
  - One-class Naïve Bayes classifier (*Schwab et al. 2000*)
  - Latent-class generative models (*Zhang et al. 2002*)

Traditional Recommender Systems: Collaborative Filtering Approaches

- Heuristic approaches
  - Neighborhood methods
    - User-based algorithms (*Breese et al. 1998; Resnick et al. 1994; Sarwar et al. 1998*)
    - Item-based algorithms (*Deshpande & Karypis 2004; Linden et al. 2003; Sarwar et al. 2001*)
    - Similarity fusion (*Wang et al. 2006*)
    - Weighted-majority (*Delgado & Ishii 1999*)
    - Matrix reduction methods (SVD, PCA processing) (*Goldberg et al. 2001; Sarwar et al. 2000*)
  - Association rule mining (*Lin et al. 2002*)
  - Graph-based methods (*Aggarwal et al. 1999; Huang et al. 2004, 2007*)
- Model-based approaches
  - Matrix reduction methods (*Takacs et al. 2008; Toscher et al. 2008*)
  - Latent-class generative model (*Hofmann 2004; Kumar et al. 2001; Jin et al. 2006*)
  - User-profile generative model (*Pennock et al. 2000; Yu et al. 2004*)
  - User-based classifiers (*Billsus & Pazzani 1999; Pazzani & Billsus 1997*)
  - Item dependency (Bayesian) networks (*Breese et al. 1998; Heckerman et al. 2000*)

Traditional Recommender Systems: Hybrid Approaches

- Heuristic approaches
  - Combine recommendations
    - Weighted (*Claypool et al. 1999*)
    - Mixed (*Smyth & Cotter 2000*)
    - Switching (*Billsus & Pazzani 2000*)
    - Voting (*Pazzani 1999*)
  - Feature augmentation (*Melville et al. 2002; Soboroff & Nicholas 1999*)
  - Graph-based method (*Huang et al. 2004*)
- Model-based approaches
  - Classifier with multiple types of features (*Basu et al. 1998*)
  - Hierarchical Bayesian (*Ansari et al. 2000; Condliff et al. 1999*)
  - Latent-class generative models (*Popescul et al. 2001; Schein et al. 2002*)
  - Relational learning methods (probabilistic relational models) (*Getoor & Sahami 1999; Huang et al. 2004; Newton & Greiner 2004*)

Using Context for Recommendations

- Early work: task-focused recommendation (*Herlocker & Konstan 2001*)
  - Knowledge about user's task can lead to better recommendations
  - Operates within the traditional 2D *User* × *Item* space
    - Task specification: a list of sample items related to the task at hand
- Context-aware information access/retrieval (*Jones 2005*)
  - Assisting search- and querying-based user activities using the knowledge of context information – significant amount of work
  - Typically no modeling of (long-term) user preferences
    - E.g., "find all files created during a spring meeting on a sunny day outside an Italian restaurant in New York"
  - Key applications
    - Interactive and proactive retrieval of previously seen information
    - Support of mobile users (time- and location-based capabilities, domain knowledge base, rich user interface, etc.), e.g., travel applications
    - Human digital memories

Context in Recommender Systems

- Focus of this tutorial: *contextual recommender systems*
  - Modeling and predicting (long-term) user preferences (e.g., ratings)
- Data in traditional recommender systems
  - Rating information: <*user*, *item*, *rating*>
  - Also, descriptive information/attributes about items (e.g., movie genre) and users (e.g., demographics)
- Data in context-aware recommender systems
  - Rating information: <*user*, *item*, *rating*, *context*>
  - In addition to information about items and users, also may have descriptive information/attributes about context
    - E.g., context hierarchies (Saturday → Weekend)
- Fundamental questions:
  - How to model context with respect to user preferences?
  - Can traditional (non-contextual) recommender systems be used to generate context-aware recommendations?

Relevance of Contextual Information

- Not all contextual information is relevant for generating recommendations
- E.g., which contextual information is relevant when recommending a book?
  - For what purpose is the book bought? (Work, leisure, …)
  - When will the book be read? (Weekday, weekend, …)
  - Where will the book be read? (At home, at school, on a plane, …)
  - How is the stock market doing at the time of the purchase?
- Determining relevance of contextual information:
  - *Manually*, e.g., using domain knowledge of the recommender system's designer
  - *Automatically*, e.g., using feature selection procedures or statistical tests based on existing ratings data
- We assume that only the relevant contextual information is kept

Approaches to Integrating Context and User Preferences

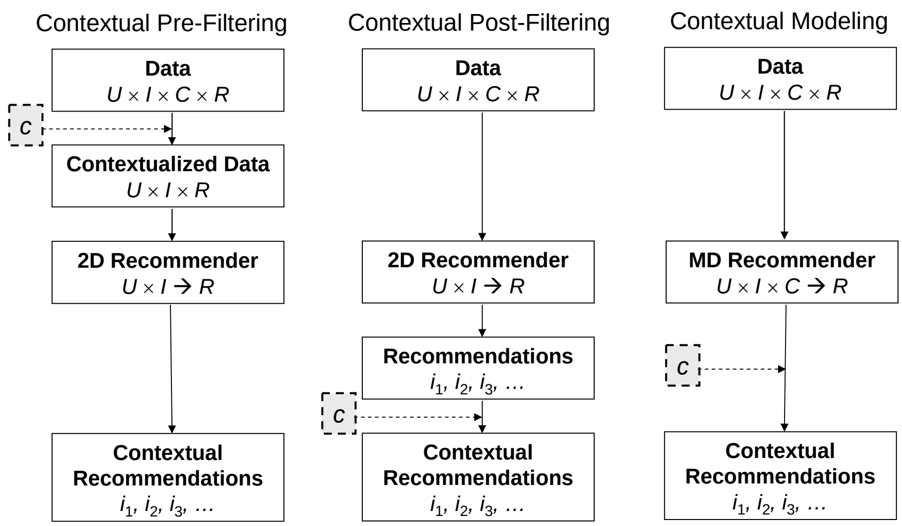
- *Loose coupling* of context and user preferences
  - Assumption: user preferences don't depend on the context; however, the item consumption may depend on the context
  - E.g., Rating(Me, "For Whom The Bell Tolls") = 9; however, I never read long and serious novels on a weekend
  - Allows to use traditional non-contextual recommenders
- *Tight coupling* of context and user preferences
  - Assumption: user preferences directly depend on the context
  - E.g., Rating(Me, "For Whom The Bell Tolls", Saturday) = 9
  - Requires more complex rating prediction techniques
- Which approach to use depends on the application

How to Use Context in the Recommendation Process

Context can be used in the following stages:

- Contextual pre-filtering
  - Loose coupling of context and user preferences
  - Contextual information *drives data selection* for that context
  - Ratings are predicted using a traditional recommender on the selected data
- Contextual post-filtering
  - Loose coupling of context and user preferences
  - Ratings predicted on the whole data using traditional recommender
  - The contextual information is used to *adjust* ("contextualize") the resulting set of recommendations
- Contextual modeling
  - Tight coupling of context and user preferences
  - Contextual information is used *directly* in the modeling technique as a part of rating estimation

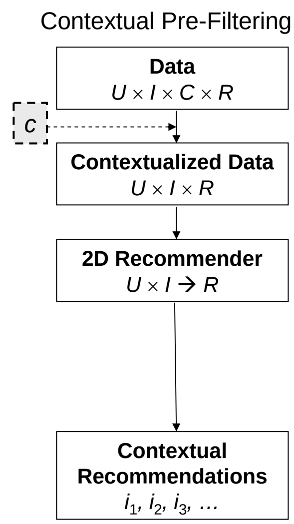
Paradigms for Incorporating Context in Recommender Systems



Contextual Pre-Filtering

Overview of Contextual Pre-Filtering

- *Pre-Filtering*: using contextual information to select the most relevant data for generating recommendations
- Context ***c*** serves as a *query* to select relevant ratings data Data(User, Item, Rating, Context), i.e.,
  - **SELECT** User, Item, Rating
    FROM Data
    WHERE Context = *c*
- *Example:* if a person wants to see a movie on Saturday, *only* the Saturday rating data is used to recommend movies



Exact vs. Generalized Pre-Filters

- *Exact pre-filtering*: Construction of the data filtering query based on the *exactly* specified context
- Exact context may be too narrow, e.g.,
  - Watching a movie with a girlfriend in a movie theater on Saturday
    - More formally, c = (Girlfriend, Theater, Saturday)
  - Certain aspects of the overly specific context may not be significant (e.g., Saturday vs. weekend)
  - Exact context may not have enough data for accurate rating prediction
- *Generalized pre-filtering*: Generalizing the data filtering query based on the specified context

Context Generalization

- Different possibilities for this generalization, based on the context taxonomy/granularity
- Example: generalizing ***c*** = (Girlfriend, Theater, Saturday)
- Assume the following contextual taxonomies (*is-a* or *belongs-to* relationships), derived from context hierarchies:
  - *Company*: Girlfriend → Friends → NotAlone → AnyCompany
  - *Place*: Theater → AnyPlace
  - *Time*: Saturday → Weekend → AnyTime
- The following are some examples of generalized context ***c***:
  - (Girlfriend, AnyPlace, Saturday)
  - (Friends, Theater, AnyTime)
  - (NotAlone, Theater, Weekend)

Generalized Pre-Filters

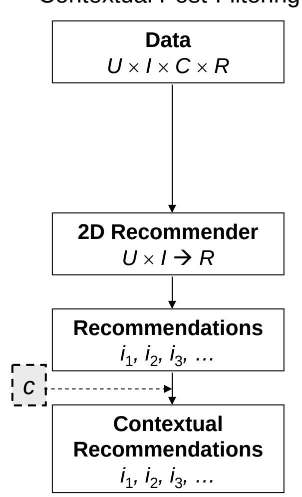
- *Definition:* $c' = (c'_1, \ldots, c'_k)$ is a *generalization of context* $c = (c_1, \ldots, c_k)$, iff $c_i \rightarrow c'_i$ for every i
- Contextualized ratings data is obtained via query
  - **SELECT** User, Item, Rating
    FROM Data
    WHERE Context = *c'*
- Choosing the Right Generalized Pre-Filter
  - *Manual approach:* domain knowledge
    - E.g., always generalize the days of week into “weekday” or “weekend”
  - *Automated approach:*
    - Evaluate the predictive performance of the recommender system on datasets from each generalized pre-filter
    - Choose the pre-filter with best performance
    - Important issue: computational complexity due to context granularity

Contextual Post-Filtering

Overview of Contextual Post-Filtering

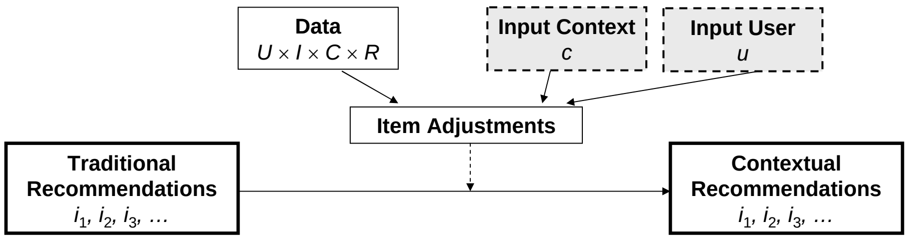
- *Post-filtering*: ignoring context in the recommendation phase, then adjusting the obtained recommendation using contextual information, e.g.,
  - Filtering out recommendations that are irrelevant (in a given context)
  - Adjusting the ranking of top-N recommendations (based on context)
- *Example:* if a person wants to see a movie on weekend, and on weekends she only watches comedies, filter out all non-comedies from the recommended movie list

Contextual Post-Filtering



Contextual Post-Filtering

- Basic idea
  - Analyze data for a given user in a given context to find specific item usage patterns or preferences
  - Use these patterns/preferences to adjust the item list, resulting in more “contextual” recommendations
- Interesting research issue
  - Incorporating context generalization techniques



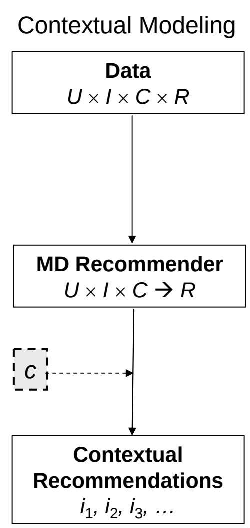
Contextual Post-Filtering Approaches

- Heuristic approaches
  - Find common item characteristics (attributes) for a given user in a given context; use these attributes to adjust the recommendations, e.g.,
    - Filter out recommended items that do not have a significant number of these characteristics, or
    - Reorder recommended items based on how many of these relevant characteristics they have
- Model-based approaches
  - Build predictive models that calculate the probability with which the user chooses a certain type of item in a given context (i.e., probability of relevance), e.g., probability of choosing different movie genres on different days of the week; then:
    - Filter out recommended items that have small probability of relevance, or
    - Reorder recommended items by weighting the predicted rating with the probability of relevance

Contextual Modeling

Overview of Contextual Modeling

- Contextual modeling: using contextual information *directly* in the modeling technique as a predictor of a user's rating for an item
  - Multidimensional recommender systems
- Example:
  - Building a predictive model (decision tree, regression, probabilistic model, etc.) that incorporates contextual information in addition to the user and item data
    - *Rating* = ***f*** (*User*, *Item*, *Context*)

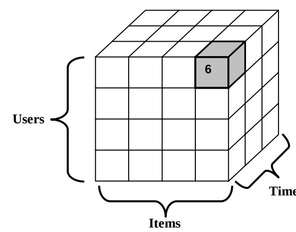


Multidimensional Recommender Systems

Traditional

		USERS			
		U_1	U_2	…	U_n
ITEMS	I_1		7	…	
	I_2	5		…	10
	⋮	⋮	⋮		⋮
	I_m	3		…	8

Multidimensional



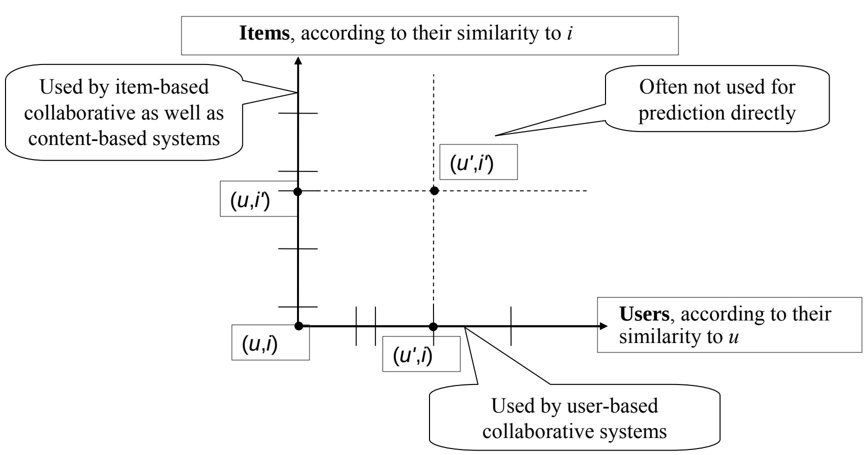
Context in the Multidimensional Recommendation (MD) Framework

- Incorporate contextual information as additional *dimensions* $D_1, \ldots, D_n$ in the *OLAP-based* recommendation space in addition to the *Users* and *Items* dimensions
  - $R: U \times I \times D_1 \times \ldots \times D_n \rightarrow Rating$
- Example: Dimensions for movie recommendation application
  - *User*
  - *Movie*
  - *Time* when the movie was seen (weekday, weekend)
  - *Company*: with whom (alone, boyfriend/girlfriend, family, etc.)
  - *Place* where the movie was seen (movie theater, at home)
  - *Time, Company, Place* are <u>*contextual*</u> dimensions.

Rating Estimation Problem

- As mentioned earlier, there have been many traditional 2D recommendation techniques developed
  - Heuristic-based approaches
  - Model-based approaches
- Can some of them be extended to incorporate contextual information?

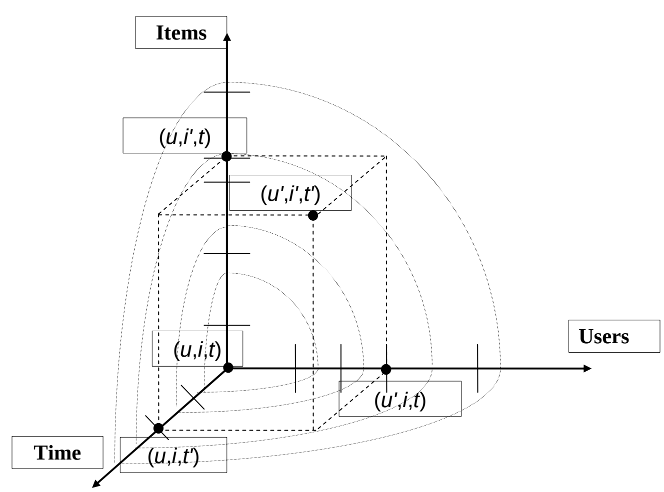
Heuristic-Based Approaches: Simplified Traditional View



Predicting Ratings Using Distance/Similarity Metrics

- Traditional (two-dimensional) user-based collaborative filtering:
  - $R(u,i) = k \, \Sigma_{u'} \, [sim(u,u') \times R(u',i)]$
- Extending to multidimensional settings
  - $\mathrm{R}(u,i,t) = k \, \Sigma_{(u,i} \, [W((u,i,t),(u',i',t')) \times R(u',i',t')]$
  - W – Weight
    - Typically inversely proportional to the distance, i.e.,
    - $W((u,i,t),(u',i',t')) = 1 \, / \, dist((u,i,t),(u',i',t'))$
  - What is an appropriate distance function?

Multidimensional Heuristic-Based Approaches



Multidimensional Distance Metrics

- Euclidean distance
  - *dist*((u,i,t),(u',i',t')) =
    SQRT[$w_u d_u(u,u')^2 + w_i d_i(i,i')^2 + w_t d_t(t,t')^2$]
- Manhattan distance
  - *dist*((u,i,t),(u',i',t')) =
    $w_u d_u(u,u') + w_i d_i(i,i') + w_t d_t(t,t')$
- Related work in data mining:
  - Clustering multidimensional data and assigning weights to different dimensions (attributes)

Model-Based Approaches: Hierarchical Bayesian Technique

- 2D recommendation approach (*Ansari et al. 2000*):
  - $r_{ui} = x'_{ui}\,\mu + z'_u\,\gamma_i + w'_i\,\lambda_u + e_{ui}$
    - r_{ui} – rating of user u for item (movie) i
    - x_{ui} – observed parameters for user u (e.g., age and gender), item i (e.g., movie genre, year, and expert ratings), and their interactions (e.g., interaction effects between gender and genre)
    - z_u – observed attributes of user u; w_i - observed attributes of item i
    - λ_u – unobserved effects of user u; γ_i – unobserved effects of movie i
    - e_{ui} – error term
  - Hierarchical regression-based Bayesian preference model that uses MCMC for estimation and prediction of model parameters
  - Allows integration of preferences, item and user characteristics
- Extending to multidimensional (context-dependent) space
  - Adding more terms to the regression model (*Adomavicius & Tuzhilin, 2005*)
  - Problem: more data is needed to estimate extra model parameters

Model-Based Approaches: Support Vector Machines (SVM)

- SVM-based predictive model for restaurant recommendation (*Oku et al. 2006*)
  - Classifies restaurants into “positive” and “negative”
  - Includes a variety of contextual data in the model as additional input dimensions, e.g.,
    - Time of visit (month, day of week, time of day)
    - Companion(s) (number, ages, relationship, status)
    - External factors (weather, temperature)
  - *Context-aware SVM* significantly outperforms context-less SVM in terms of recommendation accuracy and user’s satisfaction with recommendations

Combined Approaches

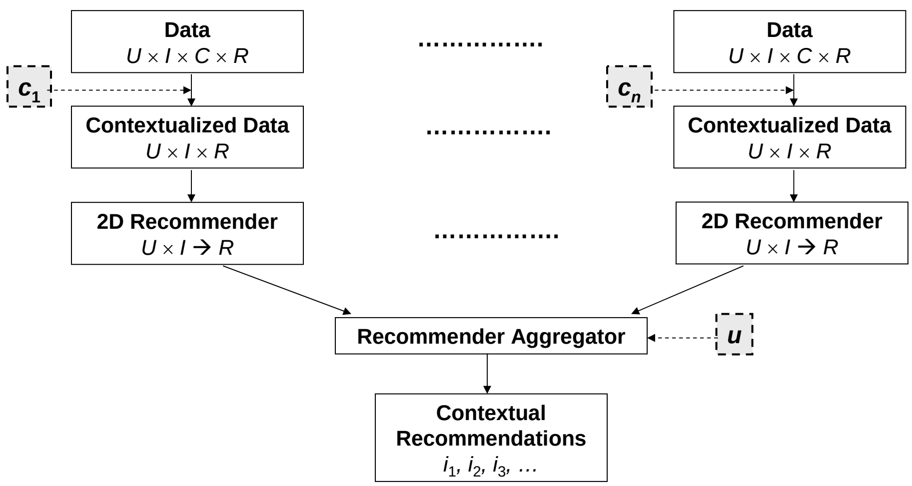
Combining Contextual Pre-Filtering, Post-Filtering, and Modeling Methods

- Complex contextual information can be split into several components
  - Each contextual component can be applied at the pre-filtering, post-filtering, and modeling stages
    - E.g., time information (weekday vs. weekend) can be used to pre-filter relevant data, but weather information (sunny vs. rainy) may be more appropriate to use as a post-filter
  - Recommendation results combined at the end using combined (e.g., ensemble) methods
- *Open research question:* how exactly to do it?
  - Will present an example below

Combining Multiple Pre-Filters

- Generate a number of different (exact or generalized) contexts
  - Motivation: there are multiple different (and potentially relevant) generalizations of a specific context
  - E.g., (Girlfriend, Theater, Saturday)
    - → (Friend, AnyPlace, Saturday), or
    - → (NotAlone, Theater, AnyTime), etc.
- Use pre-filters based on these contexts
- Combine recommendations resulting from each contextual pre-filter
  - Can be done in multiple ways, e.g., for a given context:
    - Choose the best-performing pre-filter
    - Use an "ensemble" of pre-filters

Combining Multiple Pre-Filters (cont.)



Example of Combining Contextual Pre-Filters: Reduction-Based Approach

- Proposed in (*Adomavicius et al. 2005*)
- Example: want to estimate $r_{JohnDoe,\ HarryPotter,\ Monday,\ Theater}$
  - Select all the ratings for the movies seen *on Mondays* in a *movie theater* and use *<u>any</u>* 2D estimation method on *user and movie dimension of <u>those</u>* ratings (i.e. build *<u>local models</u>*), i.e.,
    - Selection (4D → 4D, only relevant subset of ratings)
    - Projection (4D → 2D, only the main two dimensions)
    - Estimation (2D, using existing 2D estimation methods)
- Generalizable to an arbitrary multidimensional case
- *Problem:* possibly too few ratings for Monday movies in theaters

Reduction-Based Approach: Using Segments of Ratings

- Divide the rating space into *larger* segments based on generalized pre-filters
  - E.g., *Weekday* and *Weekend* (instead of *Monday* and *Saturday*)
  - Segments can overlap, for example:
    - Theater-Weekend (movies seen in a theater on weekend)
    - Theater-Friends (movies seen in a theater with friends)
- Build local models on these segments using the reduction-based approach
  - E.g., use 2D collaborative filtering (or any other 2D technique) only on the ratings from the segment
- Use local (segment-specific) models to estimate unknown ratings from these segments

Comparing 2D and Contextual Reduction-Based Approaches

- 2D approach – standard CF that ignores contextual information
- Which method is better (has higher predictive accuracy): 2D or contextual reduction-based?
- Neither 2D nor contextual reduction-based methods dominate each other in all the cases (*Adomavicius et al. 2005*)
  - Tradeoff between segment homogeneity vs. data sparsity
- Idea: combine the two methods

Combined Approach

- General idea:
  - Not every piece of contextual information matters
  - Given any 2D rating estimation method *A* (e.g., CF), find segments (pre-filters) where contextual reduction-based *A* method dominates 2D *A*; use reduction-based method on these segments and 2D *A* on others
- Algorithm for finding dominating segments (pre-filters):
  1. Given the set of known ratings *T*, determine all "large" contextual segments *SEGM*(*T*), e.g., segments having at least *N* ratings
  2. Keep only those segments in *SEGM*(*T*) for which reduction-based *A* significantly outperforms 2D *A*
  3. Also remove redundant (underperforming) sub-segments

Combined Approach (cont.)

- Rating Estimation:
  - To estimate rating *r*, determine the *best*-performing segment on which reduction-based *A* approach outperforms 2D *A*; use reduction-based *A* of this segment on *r*
  - If no such segment exists for *r*, then use the 2D *A* method

A Case Study of the Combined Approach: Movie Recommendations

- Built a Web site for the students to enter ratings of movies they saw (*Adomavicius et al. 2005*)
  - 117 students, 1755 ratings, time period 12 months
  - Dropped the students who rated less than 10 movies
    - *Result*: 202 movies, 62 students, 1457 ratings
- Dimensions:
  - *Student*
  - *Movie*
  - *Time* when the movie was seen (weekday, weekend)
  - *Company*: with whom (alone, boyfriend/girlfriend, family, etc.)
  - *Place* where the movie was seen (movie theater, at home)
- Ratings: scale from 1 to 13.
- Goal: compare 2D CF with reduction-based CF method

Decision-Support Measures

- "Good" ratings (>10)
  - only highly rated movies are recommended
- *Precision*: percentage of truly "good" movies among all the movies predicted as "good"
- *Recall*: percentage of movies predicted as "good" among all the actually good movies
- *F-measure*: 2 * Prec * Recall / (Prec + Recall)
  - We used F-measure in our experiments
- Split ratings set into 90-10% D_M and D_E sets for training and testing purposes

Determine Outperforming Segments (Step 1)

Identify large contextual segments/pre-filters (> 20% of ratings)

Name	Size	Description
Home	727	Movies watched at home
Friends	565	Movies watched with friends
NonRelease	551	Movies watched not during the 1st weekend of release
Weekend	538	Movies watched on weekends
Theater	526	Movies watched in the movie theater
Weekday	340	Movies watched on weekdays
GBFriend	319	Movies watched with girlfriend/boyfriend
Theater-Weekend	301	Movies watched in the movie theater on weekends
Theater-Friends	274	Movies watched in the movie theater with friends

Determine Outperforming Segments (Steps 2 and 3)

Step 2: Find outperforming segments (pre-filters)

Segment	Red.-based CF F-measure	2D CF F-measure
Theater/Weekend	0.641	0.528
Theater	0.608	0.479
Theater/Friends	0.607	0.504
Weekend	0.542	0.484

Step 3: Drop redundant underperforming segments

- *Theater/Friends* is a sub-segment of *Theater* with lower performance

Interpretation

- Reduction-based CF dominates the 2D CF on *some* but *not all* segments. Reason:
  - Local (segment-based) models provide more focused recommendations (segment homogeneity → higher accuracy), but have fewer ratings (higher sparsity → lower accuracy)
  - Especially critical for smaller segments

Overall Comparison (F-measure)

	Standard 2D CF	**Combined reduction-based CF**	**Difference in F-measure**
All predictions (1373 ratings)	0.463	0.526	0.063
Predictions for contextual segments (743 ratings)	0.450	0.545	0.095

Other Combination Approaches

- Simple:
  - Linear combinations of predictive models
- Complex:
  - Advanced machine learning techniques for model combination, e.g.,
    - Boosting (*Freund & Schapire 1999*)
    - Bagging (bootstrap aggregating) (*Breiman 1996*)
    - Stacking (*Wolpert 1992*)

Additional Capabilities for Contextual Recommender Systems

Additional Capabilities of Contextual Recommender Systems

- Additional recommendation opportunities
  - Recommend *contexts* to users and items
  - Recommend various entities (users, items, etc.) in certain contexts
- Novel types of user interaction capabilities in specifying and managing contexts
  - Simple specifications of contexts
  - Complex specifications of contexts
  - Broad literature on context-aware information access/retrieval *(Jones 2005)*
- *Conclusion:* need novel methods to support these interaction capabilities

Novel Context-Specific Recommendation Types

Some examples:

- Recommend various entities (users, items, etc.) in *certain contexts*
  - E.g., recommend the best courses for Jane in the *Spring semester*
  - Uses contextual information for recommendation filtering/retrieval
- Recommend *contexts* (as opposed to items) to users and items
  - E.g., recommend the best *times* for Joe to go on vacation to Hawaii

Conclusion: contextual information provides possibilities for new and richer types of recommendations

Use a *recommendation query language* to manage them

REQUEST: Query Language for Recommender Systems

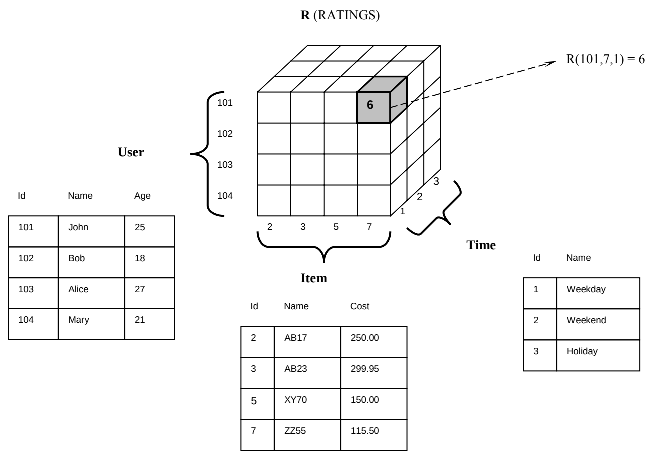
REQUEST: **<u>RE</u>**commendation **<u>QUE</u>**ry **<u>ST</u>**atements

- Allows users to customize recommendations
  - Based on multidimensional data modeling
  - Systematic mapping to a *recommendation algebra*
- Brief history:
  - Preliminary version: *Workshop on Electronic Commerce* (WELCOM), 2001 (*Adomavicius & Tuzhilin, 2001*)
  - Intermediate version: MISRC Working Paper 05-15, *University of Minnesota*, 2005 (*Adomavicius et al., 2005*)
  - Latest version: *Information Systems Research* journal, 2008 (conditionally accepted) (*Adomavicius et al., 2008*)

Overview of REQUEST: Main Features

- Supports not only queries based on user and item characteristics, but also
  - Contextual dimensions (i.e., context attributes)
  - Multiple numeric and Boolean ratings (rating vectors)
  - Different recommendation types
  - OLAP-style filtering and aggregation capabilities
  - Recommendation ranking

Multidimensional Recommendation Space for REQUEST (OLAP-based paradigm)



REQUEST Queries: Simple Examples

- "Classical" (non-contextual) query
  - *"Recommend top 3 movies to each user"*
    RECOMMEND MOVIE TO USER
    USING MovieRecommender
    BASED ON Rating
    SHOW TOP 3

Alice	Memento	10
	K-PAX	10
	Titanic	9
Bob	Star Wars	10
	Gladiator	9
	Notorious	8
Cindy	… … …	…

- Simple query with context information:
  - *"Recommend best 2 times for each user for all available Hawaii vacations during the spring"*
    RECOMMEND TIME TO USER, VACATION
    USING VacationRecommender
    RESTRICT Vacation.Destination = "Hawaii" AND Time.Season = "Spring"
    BASED ON Rating
    SHOW TOP 2

REQUEST Queries:
More Complex Examples

- [Rating aggregation] *"Recommend the top genre for each user, based on movies that are longer than 2 hours and would be watched on the weekend"*

  RECOMMEND MOVIE.Genre TO USER
  USING MovieRecommender
  RESTRICT MOVIE.Length > 120 AND TIME.TimeOfWeek = "Weekend"
  BASED ON Rating(AVG)

- [Complex recommendation type with multiple contextual dimensions] *"Recommend to Tom and his girlfriend top 3 movies and the best times to see them over the weekend"*

  RECOMMEND MOVIE, TIME TO USER, COMPANION
  USING MovieRecommender
  RESTRICT USER.Name = "Tom" AND TIME.TimeOfWeek ="Weekend"
  AND COMPANION.Type = "Girlfriend"
  BASED ON Rating
  SHOW TOP 3

REQUEST Syntax Overview

- High-level BNF of a REQUEST query:

  RECOMMEND *recommend_dim_list* **TO** *recipient_dim_list*
  USING *cube_name*
  [**RESTRICT** *dimension_restrictions*]
  [**PREFILTER** *preaggregation_measure_restrictions*]
  BASED ON *aggr_measure_list*
  [**POSTFILTER** *postaggregation_measure_restrictions*]
  [**SHOW** *measure_rank_restriction*]

Summary of REQUEST Capabilities

- Context-related capabilities
  - Flexibility in specifying and managing contexts
  - New types of recommendations involving contexts in novel ways
- Other capabilities (beyond the scope of this tutorial)
  - Rating aggregation
  - Specification of various restrictions on
    - Ratings
    - Dimensions (e.g., users, items, contextual)
  - Flexible recommendation ranking options
- In summary, REQUEST supports rich and flexible interaction capabilities for end-users

Future Directions

Future Research

- Establishing relevant contextual features
  - E.g., what context matters in a given application?
- Advanced techniques for learning context from data
  - E.g., use of latent variables (Hidden Markov Models, Bayesian Networks, etc.)
- Choosing the best approach for a given contextual recommendation setting
  - I.e., whether to use contextual pre-filtering, post-filtering, or modeling
- Contextual pre-filtering/reduction-based techniques
  - Finding relevant generalized pre-filters (currently: semi-automated expert-driven process)
  - Using different 2D recommendation approaches on different pre-filters (contextual segments)

Future Research (cont.)

- Contextual post-filtering
  - Techniques for finding specific item usage patterns or preferences in a given context
  - Converting item usage patterns into adjustments for recommended item list
  - Heuristic and model-based approaches for contextual post-filtering
- Contextual modeling techniques
  - Extending traditional 2D recommendation techniques to multidimensional recommendation space (which includes contextual dimensions)
  - Heuristic approaches
    - Multidimensional distance metrics
  - Model-based approaches
    - Scalability in estimating more model parameters; overcoming data sparsity in higher-dimensional predictive models
  - Using advanced machine learning techniques for building context-aware models for rating prediction

Future Research (cont.)

- Combined approaches
  - Optimal splitting of relevant contextual information into pre-filtering, post-filtering, and modeling components
  - Using advanced combination techniques (e.g., from machine learning)
- Developing richer interaction capabilities for context-aware recommender systems
  - Recommendation query languages
  - Intelligent user interfaces
- General issues for the recommender systems field:
  - Obtaining datasets on context-aware recommender systems for algorithmic development
  - Performing live experiments with human subjects to better evaluate the efficacy of the proposed contextual recommenders

Thank You!

Questions?