

ALPEN-ADRIA
UNIVERSITÄT
KLAGENFURT

The Quest for the Ultimate Recommender System

Lecture - Trends in e-Commerce
TU Wien 30.11.2009

Markus Zanker
Universität Klagenfurt

- 1 -

Agenda

- What are recommender systems for?
- How do they work?
- How can they be optimized?
- How to measure their success?
- What to expect?

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 2 -

Sales assistance on the Web

- Variety of "selling positions"
 - Logistics servicemen deliver and take orders
 - Medical detailers educate MDs about new products
 - Insurance sales assistants tailor products to buyer's needs
 - Talk focuses on B2C commerce and creative selling tasks
- "Sales assistance improves the quality of matches in horizontally differentiated markets" [Wernerfeldt, Marketing Science 13(1), 1994]
- "...listen and question in order to identify customer needs and come up with good product propositions" [Kotler, 1991]
- "Salespersons do better the better they understand customer decision making" [Weitz, Journal of Marketing Research 15, 1978]

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 3 -

Functions of Virtual Sales Assistants

- Several models of consumer behaviour, basically three early and interleaved stages: [Miles et al., IJHCS 52, 2000]

- **Search for products (design):** need is identified based on a set of more or less specific criteria for desired product, identification of problem space
- **Management of search criteria (intelligence):** search criteria are altered or refined due to additional information that comes up, opinions from peers, experiences made by search and assessments, exploration of problem space
- **Comparison of products/merchants (choice):** simultaneous and/or relative assessments based on a number of criteria, rankings based on applied decision strategy

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 4 -

Agenda

- What are recommender systems for?
- How do they work?
- How can they be optimized?
- How to measure their success?
- What to expect?

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 5 -

Recommender systems

- RS seen as a function [Adomavicius&Tuzhilin, TKDE 05]
- Given:
 - User model (e.g. ratings, preferences, demographics, situational context)
 - Item
- Find:
 - Relevance score
- Scores responsible for ranking
- In practical systems usually not all items will be scored, but task is to find most relevant ones (selection task)

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 6 -

Paradigms of recommender systems

Recommender systems reduce information overload by estimating relevance

The diagram shows a black cube labeled 'Recommendation component' with an arrow pointing to a table labeled 'Recommendation list'. The table has columns 'Item' and 'score' with rows: (i1, 0.9), (i2, 1), and (i3, 0.3).

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 7 -

Paradigms of recommender systems

Personalized recommendations

The diagram shows a black cube labeled 'Recommendation component' with an arrow pointing to a table labeled 'Recommendation list'. An arrow from a person icon labeled 'User profile & contextual parameters' points to the cube. The table has columns 'Item' and 'score' with rows: (i1, 0.9), (i2, 1), and (i3, 0.3).

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 8 -

Paradigms of recommender systems

Collaborative: "tell me what's popular among my peers"

The diagram shows a black cube labeled 'Recommendation component' with an arrow pointing to a table labeled 'Recommendation list'. Arrows from a person icon labeled 'User profile & contextual parameters' and a group of people icon labeled 'Community data' point to the cube. The table has columns 'Item' and 'score' with rows: (i1, 0.9), (i2, 1), and (i3, 0.3).

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 9 -

Paradigms of recommender systems

Content-based: "show me more of the same what I've liked"

The diagram shows a black cube labeled 'Recommendation component' with an arrow pointing to a table labeled 'Recommendation list'. An arrow from a person icon labeled 'User profile & contextual parameters' points to the cube. An arrow from a table labeled 'Product features' (with columns Title, Genre, Actors) points to the cube. The table has columns 'Item' and 'score' with rows: (i1, 0.9), (i2, 1), and (i3, 0.3).

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 10 -

Paradigms of recommender systems

Knowledge-based: "tell me what fits based on my needs"

The diagram shows a black cube labeled 'Recommendation component' with an arrow pointing to a table labeled 'Recommendation list'. Arrows from a person icon labeled 'User profile & contextual parameters', a table labeled 'Product features' (with columns Title, Genre, Actors), and a gear icon labeled 'Knowledge models' point to the cube. The table has columns 'Item' and 'score' with rows: (i1, 0.9), (i2, 1), and (i3, 0.3).

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 11 -

Paradigms of recommender systems

Hybrid: combinations of various inputs and/or composition of different mechanism

The diagram shows a black cube labeled 'Recommendation component' with an arrow pointing to a table labeled 'Recommendation list'. Arrows from a person icon labeled 'User profile & contextual parameters', a group of people icon labeled 'Community data', a table labeled 'Product features' (with columns Title, Genre, Actors), and a gear icon labeled 'Knowledge models' point to the cube. The table has columns 'Item' and 'score' with rows: (i1, 0.9), (i2, 1), and (i3, 0.3).

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at - 12 -

Recommender systems: input requirements

- Different types of input are naturally associated to different recommendation paradigms

	User profile & context factors	Community data	Product features	Knowledge models
Collaborative	Yes	Yes	No	No
Content-based	Yes	No	Yes	No
Knowledge-based	Yes	No	Yes	Yes

Recommender systems: basic techniques

	Pros	Cons
Collaborative	Nearly no ramp-up effort, serendipity of results, learns market segments	Requires some form of rating feedback, cold start for new users and new items
Content-based	Rating feedback easy to acquire, supports comparisons	Content-descriptions necessary, cold start for new users, no surprises
Knowledge-based	Deterministic rec's, assured quality, no cold-start, can resemble sales dialogue	Knowledge engineering effort to bootstrap, basically static, does not react to short-term trends

Collaborative filtering I

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

- Predict if Alice likes an item or not based on the opinions of her most similar peers
 - User2 seems to be closest (both like 1,3 and are indifferent about 2)
 - So CF will predict that Alice likes Item5
- Memory-based algorithms identify nearest neighbours (kNN)
 - Pearson or Cosine as similarity measure

Collaborative filtering II

- Model-based algorithms
 - Offline training phase
 - Identify hidden semantic associations from co-occurrence
 - Very time and space efficient!
- Highly complex and sophisticated models exist
 - User biases (avg. ratings, authority, demographics, mood, ...)
 - Item biases (controversial vs. unanimous opinion)
 - Context biases (time changing, campaigns, rating frequency in session)

Content-based filtering I

- The book universe:

Title	Genre	Author	Type	Price	Keywords
The Night of the Gun	Memoir	David Carr	Paperback	29.90	Press and journalism, drug addiction, personal memoirs, New York
The Lace Reader	Fiction, Mystery	Brunonia Barry	Hardcover	49.90	American contemporary fiction, detective, historical
Into the Fire	Romance, Suspense	Suzanne Brockmann	Hardcover	45.90	American fiction, Murder, Neo-nazism
...					

Content-based filtering II

- Recommend a book to Alice based on what she enjoyed reading:

Title	Genre	Author	Type	Price	Keywords
...	Fiction, Suspense	Brunonia Barry, Ken Follet, ..	Paperback	25.65	detective, murder, New York

- User profile is used as a query on the book universe
- Retrieve most similar ones first
 - TF-IDF measure
 - For semi-structured items additional measures needed
 - Weighting of attributes based on their relevance

Knowledge-based Recommendation I

- Conversational interaction strategy
 - Opposed to one-shot interaction
 - Elicitation of user requirements
 - Transfer of product knowledge (“educating users”)
- Explicit domain knowledge
 - Requirements elicitation from domain experts
 - System mimics the behaviour of experienced sales assistant
 - Best-practice sales interactions
 - Can guarantee “correct” recommendations (determinism)

Knowledge-based Recommendation II

- Different views on “knowledge”
 - Similarity functions
 - Determine matching degree between query and item (case-based RS)
 - Utility-based RS
 - E.g. MAUT – Multi-attribute utility theory
 - Interactive constraint/preference acquisition from users
 - Critiques or local preferences
 - Declarative knowledge-base (from domain expert)
 - E.g. Hard and soft constraints
- Hybridization
 - E.g. merging explicit knowledge with community data
 - Can ensure some policies based on e.g. availability, user context or profit margin

Constraint-based recommendation I

- Knowledge base
 - Usually mediates between user model and item properties
 - Variables
 - User model features (requirements), Item features (catalogue)
 - Set of constraints
 - Logical implications (IF user requires A THEN proposed item should possess property B)
 - Either hard or soft
 - Penalty value in case of constraint violation
- Derive a set of recommendable items
 - Fulfilling set of applicable constraints
 - Applicability of constraints depends on current user model
 - Explanations – transparent line of reasoning

Constraint-based recommendation II

- Several different recommendation tasks:
- Find a set of user requirements such that a subset of items fulfills all constraints
 - Ask user which requirements should be relaxed/modified such that some items exist that do not violate any constraint (e.g. Trip@dvce/Ricci)
 - Find a subset of items that satisfy the maximum set of weighted constraints
 - Similar to find a maximally succeeding subquery (XSS)
 - All proposed items have to fulfill the same set of constraints (e.g. Advisor Suite/Jannach)
 - Compute relaxations based on predetermined weights
 - Rank items according to weights of satisfied soft constraints
 - Rank items based on the ratio of fulfilled constraints
 - Does not require additional ranking scheme

Example

Knowledge Base:			Product catalogue:	
Weight	LHS	RHS	Powershot XY	
C1: 25	TRUE	Brand = Brand pref.	Brand	Canon
C2: 20	Motives = Landscape	Low. foc. Length = < 28	Lower focal length	35
C3: 15	TRUE	Price = < Max. cost	Upper focal length	140
			Price	420 EUR

Current user:		Lumix	
User model (requirements)		Brand	Panasonic
Motives	Landscape	Lower focal length	28
Brand preference	Canon	Upper focal length	112
Max. cost	350 EUR	Price	319 EUR

Ask user

- Do you want to relax your brand preference?
 - Accept *Panasonic* instead of *Canon* brand
- Or is photographing landscapes with a wide-angle lens and maximum cost less important?
 - Lower focal length > 28mm and Price > 350 EUR
- Computation of minimal revisions of requirements
 - Eventually guided by some predefined weights or past community behavior

Find XSS

- Two maximally succeeding subqueries
 - XSS = {{C1},{C2, C3}}
 - Selection can be based on constraints' weights
 - Relax C1 and recommend Lumix

Compute scores

- Applicable: LHS(c) is satisfied by user model, i.e. {C1,C2,C3}
- Satisfied: not applicable or RHS(c) is satisfied by catalogue item, i.e. {C1} for Powershot and {C2,C3} for Lumix

$$rec_{\gamma}(i, u, \Gamma) = \begin{cases} score(i, u, \Gamma) & : \forall c \in C_{hard} \text{ sat}(c) \\ 0 & : \text{else} \end{cases}$$

- Only items that satisfy all hard constraints receive positive score (fulfilled for both)

$$score(i, u, \Gamma) = \frac{1}{\sum_{c \in C} pen(c)} \times \sum_{\substack{c \in C \\ app(c) \wedge sat(c)}} pen(c)$$

- Ratio of penalty values of satisfied constraints
 - Ranked #1: Lumix 35/60
 - Ranked #2: Panasonic: 25/60
 - Cutoff recommendation list after n items

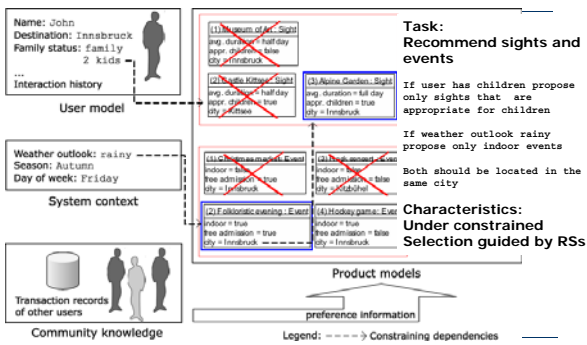
Constraint-based recommendation III

- More variants of constraint representation:
 - Interactive acquisition of solution preferences
 - E.g. user can ask „less price“ for Lumix
 - To explore cheaper variants of current proposal
 - Max. cost of 350 EUR for Canon brand initially specified, higher price sensitivity for Panasonic brand?
 - Aging/Outdating of „older“ preferences
 - Construction of decision model
 - Disjunctive RHS
 - IF location requ. = nearby THEN location = Ktn OR location = Stmk
 - E.g. The thermal spa should be located either in Carinthia or Styria

Constraint-based recommendation IV

- More variants of recommendation task
 - Find „diverse“ sets of items
 - Notion of similarity/dissimilarity
 - Idea that users navigate a product space
 - If recommendations are more diverse than users can navigate via critiques on recommended „entry points“ more efficiently (less steps of interaction)
 - Bundling of recommendations
 - Find item bundles that match together according to some knowledge
 - E.g. travel packages, skin care treatments or financial portfolios (see master thesis Aschinger)
 - RS for different item categories, CSP restricts configuring of bundles

Bundling items

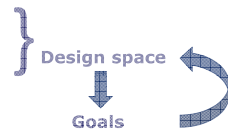


Constraint-based recommendation V

- Find optimal sequence of conversational moves
 - „Recommendation is less about optimal algorithms, but more about the interaction experience“
 - Asking for requirements, proposing items (that can be critiqued) or showing explanatory texts are all conversational moves
 - Interaction process towards preference elicitation and maybe also user persuasion
 - Exploit current state of the system as well as community information

Agenda

- What are recommender systems for?
- How do they work?
- How can they be optimized?
- How to measure their success?
- What to expect?

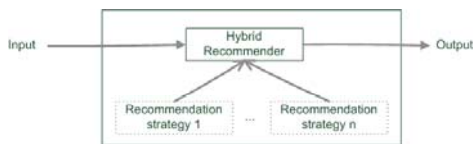


Hybrid recommender systems

- All three base techniques are naturally incorporated by a good sales assistance (at different stages of the sales act) **but** have their shortcomings
- Idea of crossing two (or more) species/implementations
 - *hybrida* [lat.]: denotes an object made by combining two different elements
 - Avoid some of the shortcomings
 - Reach desirable properties not (or only inconsistently) present in parent individuals
- Different hybridization designs
 - Parallel use of several systems
 - Monolithic exploiting different features
 - Pipelined invocation of different systems

Monolithic hybridization design

- Only a single recommendation component



- Hybridization is 'virtual' in the sense that
 - Features/knowledge sources of different paradigms are combined

Monolithic hybridization designs: Feature combination

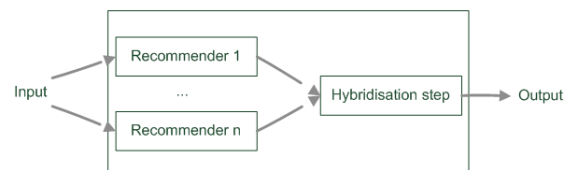
- Combination of several knowledge sources
 - E.g.: Ratings and user demographics or explicit requirements and needs used for similarity computation
- 'Hybrid' content features:
 - Social features: Movies liked by user
 - Content features: Comedies liked by user, dramas liked by user
 - Hybrid features: user likes many movies that are comedies, ...
- *"the common knowledge engineering effort that involves inventing good features to enable successful learning"*
[Basu et al., AAAI, 1998]

Monolithic hybridization designs: Feature augmentation

- Content-boosted collaborative filtering [Melville et al., AAAI, 2002]
 - Based on content features additional ratings are created
 - E.g. Alice likes Items 1 and 3 (unary ratings)
 - Item7 is similar to 1 and 3 by a degree of 0,75
 - Thus Alice likes Item7 by 0,75
 - Item matrices become less sparse
 - Significance weighting and adjustment factors
 - Peers with more co-rated items are more important
 - Higher confidence in content-based prediction, if higher number of own ratings
- Recommendation of research papers [Torres et al., JCDL, 2004]
 - Citations interpreted as collaborative recommendations

Parallelized hybridization design

- Output of several existing implementations combined
- Least invasive design
- Some weighting or voting scheme
 - Weights can be learned dynamically
 - Extreme case of dynamic weighting is switching



Parallelized hybridization design: Weighted

- Compute weighted sum:

$$rec_{weighted}(u, i) = \sum_{k=1}^n \beta_k \times rec_k(u, i)$$

Recommender 1			Recommender 2		
Item1	0.5	1	Item1	0.8	2
Item2	0		Item2	0.9	1
Item3	0.3	2	Item3	0.4	3
Item4	0.1	3	Item4	0	
Item5	0		Item5	0	

Recommender weights (0.5:0.5)		
Item1	0.65	1
Item2	0.45	2
Item3	0.35	3
Item4	0.05	4
Item5	0.00	

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at

- 37 -

Parallelized hybridization design: Weighted

- BUT, how to derive weights?
 - Estimate, e.g. by empirical bootstrapping
 - Dynamic adjustment of weights
- Empirical bootstrapping
 - Historic data is needed
 - Compute different weightings
 - Decide which one does best
- Dynamic adjustment of weights
 - Start with for instance uniform weight distribution
 - For each user adapt weights to minimize error of prediction

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at

- 38 -

Parallelized hybridization design: Weighted

- Let's assume Alice actually bought/clicked on items 1 and 4
 - Identify weighting that minimizes Mean Absolute Error (MAE)

$$MAE = \frac{\sum_{r_i \in R} \sum_{k=1}^n \beta_k \times |rec_k(u, i) - r_i|}{|R|}$$

Absolute errors and MAE						
Beta1	Beta2		rec1	rec2	error	MAE
0.1	0.9	Item1	0.5	0.8	0.23	0.61
		Item4	0.1	0.0	0.99	
0.3	0.7	Item1	0.5	0.8	0.29	0.63
		Item4	0.1	0.0	0.97	
0.5	0.5	Item1	0.5	0.8	0.35	0.65
		Item4	0.1	0.0	0.95	
0.7	0.3	Item1	0.5	0.8	0.41	0.67
		Item4	0.1	0.0	0.93	
0.9	0.1	Item1	0.5	0.8	0.47	0.69
		Item4	0.1	0.0	0.91	

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at

- 39 -

Parallelized hybridization design: Weighted

- BUT: didn't rec1 actually rank Items 1 and 4 higher?

Recommender 1			Recommender 2		
Item1	0.5	1	Item1	0.8	2
Item2	0		Item2	0.9	1
Item3	0.3	2	Item3	0.4	3
Item4	0.1	3	Item4	0	
Item5	0		Item5	0	

- Be careful when weighting!
 - Recommenders need to assign comparable scores over all users and items
 - Some score transformation could be necessary
 - Stable weights require several user ratings

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at

- 40 -

Parallelized hybridization design: Switching

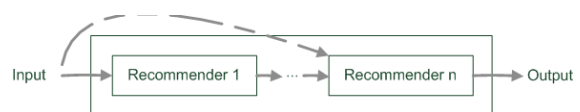
- Requires an oracle that decides on recommender
 - $\exists_1 k : 1 \dots n \text{ } rec_{switching}(u, i) = rec_k(u, i)$
- Special case of dynamic weights (all except one Beta is 0)
- Example:
 - Ordering on recommenders and switch based on some quality criteria
 - E.g. if too few ratings in the system use knowledge-based, else collaborative
 - More complex conditions based on contextual parameters, apply classification techniques

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at

- 41 -

Pipelined hybridization designs

- One recommender system pre-processes some input for the subsequent one
 - Cascade
 - Meta-level
- Refinement of recommendation lists (cascade)
- Learning of model (e.g. collaborative knowledge-based meta-level)



Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at

- 42 -

Pipelined hybridization designs: Cascade

- Successor's recommendations are restricted by predecessor

$$rec_{cascade}(u, i) = rec_n(u, i)$$

- Where for all $k > 1$

$$rec_k(u, i) = \begin{cases} rec_k(u, i) & : rec_{k-1}(u, i) \neq 0 \\ 0 & : \text{else} \end{cases}$$

- Subsequent recommender may not introduce additional items
- Thus produces very precise results

Pipelined hybridization designs: Cascade

Recommender 1			Recommender 2		
Item1	0.5	1	Item1	0.8	2
Item2	0		Item2	0.9	1
Item3	0.3	2	Item3	0.4	3
Item4	0.1	3	Item4	0	
Item5	0		Item5	0	

Recommender cascaded(rec1, rec2)		
Item1	0.80	1
Item2	0.00	
Item3	0.40	2
Item4	0.00	
Item5	0.00	

- Recommendation list is continually reduced
- First recommender excludes items
 - Remove absolute no-go items (e.g. knowledge-based)
- Second recommender assigns score
 - Ordering and refinement (e.g. collaborative)

Pipelined hybridization designs: Meta-level

- Successor exploits a model Delta built by predecessor

$$rec_{meta-level}(u, i) = rec_n(u, i, \Delta_{rec_{n-1}})$$

- Examples:

- Fab:
 - Online news domain
 - CB recommender builds user models based on weighted term vectors
 - CF identifies similar peers based on these user models but makes recommendations based on ratings
- Collaborative constraint-based meta-level RS
 - Collaborative filtering learns a constraint base
 - Knowledge-based RS computes recommendations

Agenda

- What are recommender systems for?
- How do they work?
- How can they be optimized?
- How to measure their success?
- What to expect?



Evaluating Recommender Systems

- A myriad of techniques has been proposed, **but**
 - Which one is best in a given application domain?
 - What are the success factors of different techniques?
 - Comparative analysis based on an optimality criterion?
- Research questions are:
 - Is a RS efficient with respect to a specific criteria like accuracy, user satisfaction, response time, serendipity, online conversion, ramp-up efforts,
 - Do customers like/buy recommended items?
 - Do customers buy items they otherwise would have not?
 - Are they satisfied with a recommendation after purchase?

Purpose and success criteria I

- Different perspectives/aspects
 - Depends on domain and purpose
 - No wholistic evaluation scenario exists
- Retrieval perspective
 - Reduce search costs
 - Provide 'correct' proposals
 - Users know in advance what they want
- Recommendation perspective
 - Serendipity
 - Users did not know about existence

Purpose and success criteria II

- Prediction perspective
 - Predict to what degree users like an item
 - Most popular evaluation scenario in research
- Interaction perspective
 - Give users a ‚good feeling‘
 - Educate users about the product domain
 - Persuade users as an intentional planned effect!?
- Finally, conversion perspective
 - Commercial situations
 - Increase ‚hit‘, ‚clickthru‘, ‚lookers to bookers‘ rates
 - Optimize sales margins and profit

Empirical research

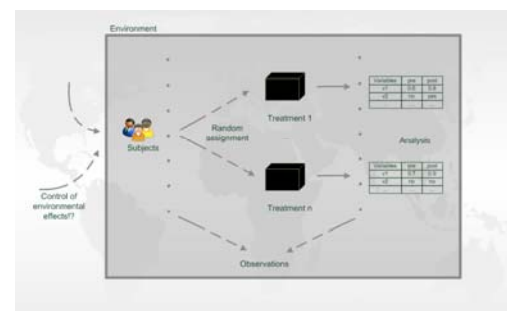
- Characterizing dimensions:
 - Who is the **subject** that is in the focus of research?
 - What **research methods** are applied?
 - In which **setting** does the research take place?

Subject	Online customers, students, historical online sessions, computers, ...
Research method	Experiments, quasi-experiments, non-experimental research
Setting	Lab, real-world scenarios

Research methods

- Experimental vs. non-experimental (observational) research methods
 - Experiment (test, trial):
 - „An experiment is a study in which at least one variable is manipulated and units are randomly assigned to different levels or categories of manipulated variable(s).“
 - Units: users, historic sessions, ...
 - Manipulated variable: type of RS, recommended items, ...
 - Categories of manipulated variable(s): content-based RS, collaborative RS

Experiment designs



Metrics: MAE and RMSE

- **Mean Absolute Error (MAE)** computes the deviation between predicted ratings and actual ratings

$$MAE = \frac{1}{n} \sum_{i=1}^n |p_i - r_i|$$

- **Root Mean Square Error (RMSE)** is similar to MAE, but places more emphasis on larger deviation

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (p_i - r_i)^2}$$

- RMSE is used, for example, in the evaluation of the Netflix Prize

Metrics: Precision and Recall I

- Precision and Recall, two standard measures from Information Retrieval, are two of the most basic methods for evaluating recommender systems

- E.g. Consider the movie predictions made by a simplified recommender that classes movies as good or bad
 - They can be split into four groups:

		Reality	
		Actually Good	Actually Bad
Prediction	Rated Good	True Positive (tp)	False Positive (fp)
	Rated Bad	False Negative (fn)	True Negative (tn)

Metrics: Precision and Recall II

- **Precision:** a measure of exactness, determines the fraction of relevant items retrieved out of all items retrieved
 - E.g. the proportion of recommended movies that are actually good

$$Precision = \frac{tp}{tp + fp} = \frac{|good\ movies\ recommended|}{|all\ recommendations|}$$

- **Recall:** a measure of completeness, determines the fraction of relevant items retrieved out of all relevant items
 - E.g. the proportion of all good movies recommended

$$Recall = \frac{tp}{tp + fn} = \frac{|good\ movies\ recommended|}{|all\ good\ movies|}$$

Metrics: Precision and Recall III

Offline experimentation	Online experimentation
Historic session	Live interaction
Ratings, transactions	Ratings, feedback
Unrated items unknown, but interpreted as dislikes	Dislikes of not recommended items unknown
False positives might be wrong	False negatives cannot be determined
Better for estimating Recall	Better for estimating Precision

Metrics: Rank Score

- **Rank Score** extends the recall metric to take the positions of correct items in a ranked list into account
 - Particularly important in recommender systems as lower ranked items may be overlooked by users
- Rank Score is defined as the ratio of the Rank Score of the correct items to best theoretical Rank Score achievable for the user, i.e.

$$rankscore = \frac{rankscore_p}{rankscore_{max}}$$

$$rankscore_p = \sum_{i \in h} 2^{-\frac{rank(i)-1}{\alpha}}$$

$$rankscore_{max} = \sum_{i=1}^{|T|} 2^{-\frac{i-1}{\alpha}}$$

Where:

- h is the set of correctly recommended items, i.e. hits
- $rank$ returns the position (rank) of an item
- T is the set of all items of interest
- α is the ranking half life

Offline experimentation

- Netflix competition
 - Web-based movie rental
 - Prize of \$1,000,000 for accuracy improvement of 10% compared to own Cinematch system.
- Historical dataset
 - ~480K users rated ~18K movies on a scale of 1 to 5
 - ~100M ratings
 - Last 9 ratings/user withheld
 - Probe set – for teams for evaluation
 - Quiz set – evaluates teams' submissions
 - Test set – used by Netflix to determine winner



After 3 years ...



Learnings

- Winning team combined more than 100 different predictors
 - Small group of controversial movies responsible for high share of error rate
 - E.g. singular value decomposition, a technique for deriving the underlying factors that cause viewers to like or dislike movies, can be used to find connections between movies
 - Interestingly, most teams used similar prediction techniques
- Very complex and specialized models
 - Switching of model parameters based on user/session features
 - Number of rated items
 - Number of rated items on particular day
- Content features added noise

Online experimentation

- Effectiveness of different algorithms for recommending cell phone games [Heglich and Jannach, RecSys'09]
- Involved 150,000 users on a commercial mobile internet portal
- A/B testing
- Results:
 - Personalized methods increased sales by 3.6% compared to impersonalized 'by release-date'
 - Differences between methods not significant



Non-experimental research

- Quasi-experiments
 - Lack random assignments of units to different treatments
- Non-experimental / observational research
 - Surveys / Questionnaires
 - Longitudinal research
 - Observations over long period of time
 - E.g. Customer life-time value, returning customers
 - Case studies
 - Focus group
 - Interviews
 - Think aloud protocols

Quasi-experimental

- SkiMatcher Resort Finder introduced by Ski-Europe.com to provide users with recommendations based on their preferences
- Conversational RS
 - question and answer dialog
 - matching of user preferences with knowledge base
- Delgado and Davidson evaluated the effectiveness of the recommender over a 4 month period in 2001
 - Classified as a quasi-experiment as users decide for themselves if they want to use the recommender or not



SkiMatcher Results

	July	August	September	October
Unique Visitors	10,714	15,560	18,317	24,416
• SkiMatcher Users	1,027	1,673	1,878	2,558
• Non-SkiMatcher Users	9,687	13,887	16,439	21,858
Requests for Proposals	272	506	445	641
• SkiMatcher Users	75	143	161	229
• Non-SkiMatcher Users	197	363	284	412
Conversion	2.54%	3.25%	2.43%	2.63%
• SkiMatcher Users	7.30%	8.55%	8.57%	8.95%
• Non-SkiMatcher Users	2.03%	2.61%	1.73%	1.88%
Increase in Conversion	359%	327%	496%	475%

[Delgado and Davidson, ENTER 2002]

Interpreting the Results

- The nature of this research design means that questions of causality **cannot** be answered, such as
 - Are users of the recommender systems more likely convert?
 - Does the recommender system itself cause users to convert?
- However, significant correlation between using the recommender system and making a request for a proposal
- Size of effect has been replicated in other domains!
 - Tourism
 - Electronic consumer products

What is popular?

- Evaluations on historical datasets measuring accuracy
- Most popular datasets
 - Movies (MovieLens, EachMovie, Netflix)
 - Web 2.0 platforms (tags, music, papers, ...)
- Most popular measures for accuracy
 - Precision/Recall
 - If item is either relevant or not
 - MAE (Mean Absolute Error), RMSE (Root Mean Squared Error)
 - If we want to predict ratings (on a Likert scale)

What is popular?

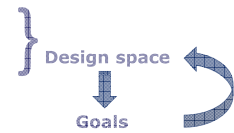
- Evaluation designs ACM TOIS 2004-2008
 - In total 12 articles on RS
 - 50% movie domain
 - 75% offline experimentation
 - 2 user experiments under lab conditions
 - 1 qualitative research
- Availability of data **heavily biases** what is done
 - Many „tag recommenders“ proposed recently
 - Tenor at RecSys'09 to foster live experiments
 - Public infrastructures to enable A/B tests

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at

- 67 -

Agenda

- What are recommender systems for?
- How do they work?
- How can they be optimized?
- How to measure their success?
- What to expect?



Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at

- 68 -

Own work

- **Hybridizing different base technologies**
 - KB and collaborative address users with different attitudes – like what they specify vs. like what others liked
- **Adaptive strategy selection**
 - Mining for behavioural cues that indicate appropriate recommendation paradigm
- **Learning of knowledge bases**
 - Learn functional relationships from past online transactions
- **Specific problem domains**
 - Generate *knowledgeable* explanations
 - Personalize navigation and interaction with maps

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at

- 69 -

Outlook hypotheses

„RS research will become much more diverse“

- Less focus on explicit ratings
 - But various forms of feedback mechanisms
- Less focus on algorithms
 - But also on interfaces and interaction processes
- Less focus on offline experimentation
 - But live experiments, real-world case studies, ...
- More focus on causal relationships
 - When, where and how to recommend?

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at

- 70 -

Thank you for your attention!

Questions?
 Questions?
 Questions?

Markus Zanker
 Intelligent Systems and Business Informatics
 Institute of Applied Informatics
 University Klagenfurt, Austria
 M: markus.zanker@uni-klu.ac.at
 P: +43 463 2700 3753
<http://www.configworks.com/mzanker>

- 71 -