Tutorial: Evaluation of Recommender Systems

ACM Symposium on Applied Computing (SAC 2012)
Riva del Garda, 26 March 2012

Dietmar Jannach
TU Dortmund

Markus Zanker
Alpen-Adria-Universität Klagenfurt
Recommender Systems: An Introduction
by Dietmar Jannach, Markus Zanker, Alexander Felfernig, Gerhard Friedrich

AVERAGE CUSTOMER RATING:

星级评分 (Be the first to review)

FORMAT:
Hardcover

NOOKbook (eBook) - not available
Tell the publisher you want this in NOOKbook format

NEW FROM BN.COM

$65.00 List Price
$52.00 Online Price
(You Save 20%)

Add to Cart

NEW & USED FROM OUR

New starting at $56.46 (You S
Used starting at $51.98 (You !

See All Prices

Table of Contents

Customers who bought this also bought
Agenda

- What are recommender systems for?
  - Introduction

- How do they work?
  - Collaborative Filtering
  - Content-based Filtering
  - Knowledge-Based Recommendations
  - Hybridization Strategies

- How to measure their success?
  - Evaluation techniques
Introduction
Problem domain

- **Recommendation systems (RS) help to match users with items**
  - Ease information overload
  - Sales assistance (guidance, advisory, persuasion,...)

  *RS are software agents that elicit the interests and preferences of individual consumers [...] and make recommendations accordingly. They have the potential to support and improve the quality of the decisions consumers make while searching for and selecting products online.*

  » [Xiao & Benbasat, MISQ, 2007]
Recommender systems

- **RS seen as a function** [AT05]
- **Given:**
  - User model (e.g. ratings, preferences, demographics, situational context)
  - Items (with or without description of item characteristics)
- **Find:**
  - Relevance score. Used for ranking.
- **At the end:**
  - Recommend items that are assumed to be relevant
Paradigms of recommender systems

Recommender systems reduce information overload by estimating relevance
Paradigms of recommender systems

Personalized recommendations

User profile & contextual parameters

Recommendation component

Recommendation list

<table>
<thead>
<tr>
<th>item</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>i1</td>
<td>0.9</td>
</tr>
<tr>
<td>i2</td>
<td>1</td>
</tr>
<tr>
<td>i3</td>
<td>0.3</td>
</tr>
</tbody>
</table>

© Dietmar Jannach, Markus Zanker and Gerhard Friedrich
Paradigms of recommender systems

Collaborative: "Tell me what's popular among my peers"
Paradigms of recommender systems

Content-based: "Show me more of the same what I've liked"
Paradigms of recommender systems

Knowledge-based: "Tell me what fits based on my needs"
Paradigms of recommender systems

Hybrid: combinations of various inputs and/or composition of different mechanism
Collaborative Filtering
Collaborative Filtering (CF)

- The most prominent approach to generate recommendations
  - used by large, commercial e-commerce sites
  - well-understood, various algorithms and variations exist
  - applicable in many domains (book, movies, DVDs, ..)

- Approach
  - use the "wisdom of the crowd" to recommend items

- Basic assumption and idea
  - Users give ratings to catalog items (implicitly or explicitly)
  - Customers who had similar tastes in the past, will have similar tastes in the future
1992: Using collaborative filtering to weave an information tapestry, D. Goldberg et al., Communications of the ACM

- Basic idea: "Eager readers read all docs immediately, casual readers wait for the eager readers to annotate"
- Experimental mail system at Xerox Parc that records reactions of users when reading a mail
- Users are provided with personalized mailing list filters instead of being forced to subscribe
  - Content-based filters (topics, from/to/subject...)  
  - Collaborative filters
- E.g. Mails to [all] which were replied by [John Doe] and which received positive ratings from [X] and [Y].

- Tapestry system does not aggregate ratings and requires knowing each other
- Basic idea: "People who agreed in their subjective evaluations in the past are likely to agree again in the future"
- Builds on newsgroup browsers with rating functionality
User-based nearest-neighbor collaborative filtering (1)

**The basic technique:**
- Given an "active user" (Alice) and an item I not yet seen by Alice
- The goal is to estimate Alice's rating for this item, e.g., by
  - find a set of users (peers) who liked the same items as Alice in the past and who have rated item I
  - use, e.g. the average of their ratings to predict, if Alice will like item I
  - do this for all items Alice has not seen and recommend the best-rated

<table>
<thead>
<tr>
<th></th>
<th>Item1</th>
<th>Item2</th>
<th>Item3</th>
<th>Item4</th>
<th>Item5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>?</td>
</tr>
<tr>
<td>User1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>User2</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>User3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>User4</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

© Dietmar Jannach, Markus Zanker and Gerhard Friedrich
**User-based nearest-neighbor collaborative filtering (2)**

- **Some first questions**
  - How do we measure similarity?
  - How many neighbors should we consider?
  - How do we generate a prediction from the neighbors' ratings?

<table>
<thead>
<tr>
<th></th>
<th>Item1</th>
<th>Item2</th>
<th>Item3</th>
<th>Item4</th>
<th>Item5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>?</td>
</tr>
<tr>
<td>User1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>User2</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>User3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>User4</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
Measuring user similarity

- **A popular similarity measure in user-based CF: Pearson correlation**

$$
sim(a, b) = \frac{\sum_{p \in P} (r_{a,p} - \bar{r}_a)(r_{b,p} - \bar{r}_b)}{\sqrt{\sum_{p \in P} (r_{a,p} - \bar{r}_a)^2} \sqrt{\sum_{p \in P} (r_{b,p} - \bar{r}_b)^2}}
$$

- $a, b$ : users
- $r_{a,p}$ : rating of user $a$ for item $p$
- $P$ : set of items, rated both by $a$ and $b$
- Possible similarity values between -1 and 1; $\bar{r}_a, \bar{r}_b$ = user's average ratings

<table>
<thead>
<tr>
<th></th>
<th>Item1</th>
<th>Item2</th>
<th>Item3</th>
<th>Item4</th>
<th>Item5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>?</td>
</tr>
<tr>
<td>User1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>User2</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>User3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>User4</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

sim = 0.85
sim = 0.70
sim = -0.79
Pearson correlation

- Takes differences in rating behavior into account

- Works well in usual domains, compared with alternative measures
  - such as cosine similarity
Making predictions

- A common prediction function:

\[ \text{pred}(a, p) = \bar{r}_a + \frac{\sum_{b \in N} \text{sim}(a, b) \times (r_{b,p} - \bar{r}_b)}{\sum_{b \in N} \text{sim}(a, b)} \]

- Calculate, whether the neighbors' ratings for the unseen item \( i \) are higher or lower than their average

- Combine the rating differences – use the similarity with as a weight

- Add/subtract the neighbors' bias from the active user's average and use this as a prediction
Improving the metrics / prediction function

- Not all neighbor ratings might be equally "valuable"
  - Agreement on commonly liked items is not so informative as agreement on controversial items
  - **Possible solution**: Give more weight to items that have a higher variance

- **Value of number of co-rated items**
  - Use "significance weighting", by e.g., linearly reducing the weight when the number of co-rated items is low

- **Case amplification**
  - Intuition: Give more weight to "very similar" neighbors, i.e., where the similarity value is close to 1.

- **Neighborhood selection**
  - Use similarity threshold or fixed number of neighbors
Basic idea:
- Use the similarity between items (and not users) to make predictions

Example:
- Look for items that are similar to Item5
- Take Alice's ratings for these items to predict the rating for Item5

<table>
<thead>
<tr>
<th></th>
<th>Item1</th>
<th>Item2</th>
<th>Item3</th>
<th>Item4</th>
<th>Item5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>?</td>
</tr>
<tr>
<td>User1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>User2</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>User3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>User4</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
More on ratings

- Pure CF-based systems only rely on the rating matrix
- Explicit ratings
  - Most commonly used (1 to 5, 1 to 7 Likert response scales)
  - Challenge
    - Users not always willing to rate many items; sparse rating matrices
    - How to stimulate users to rate more items?
- Implicit ratings
  - clicks, page views, time spent on some page, demo downloads ...
  - Can be used in addition to explicit ones; question of correctness of interpretation
Data sparsity problems

- **Cold start problem**
  - How to recommend new items? What to recommend to new users?

- **Straightforward approaches**
  - Ask/force users to rate a set of items
  - Use another method (e.g., content-based, demographic or simply non-personalized) in the initial phase

- **Alternatives**
  - Use better algorithms (beyond nearest-neighbor approaches)
  - Example:
    - In nearest-neighbor approaches, the set of sufficiently similar neighbors might be too small to make good predictions
    - Assume "transitivity" of neighborhoods
Memory-based and model-based approaches

- **User-based CF is said to be "memory-based"**
  - the rating matrix is directly used to find neighbors / make predictions
  - does not scale for most real-world scenarios
  - large e-commerce sites have tens of millions of customers and millions of items

- **Model-based approaches**
  - based on an offline pre-processing or "model-learning" phase
  - at run-time, only the learned model is used to make predictions
  - models are updated / re-trained periodically
  - large variety of techniques used
  - model-building and updating can be computationally expensive
Model-based approaches

- Plethora of different techniques proposed in the last years, e.g.,
  - Matrix factorization techniques, statistics
    - singular value decomposition, principal component analysis
  - Association rule mining
    - compare: shopping basket analysis
  - Probabilistic models
    - clustering models, Bayesian networks, probabilistic Latent Semantic Analysis
  - Various other machine learning approaches

- Costs of pre-processing
  - Usually not discussed
  - Incremental updates possible?
2000: *Application of Dimensionality Reduction in Recommender System*, B. Sarwar et al., WebKDD Workshop

- **Basic idea:** Trade more complex offline model building for faster online prediction generation
- **Singular Value Decomposition for dimensionality reduction of rating matrices**
  - Captures important factors/aspects and their weights in the data
  - Factors can be genre, actors but also non-understandable ones
  - Assumption that k dimensions capture the signals and filter out noise (K = 20 to 100)
- **Constant time to make recommendations**
- **Approach also popular in IR (Latent Semantic Indexing), data compression,…**
A picture says ...

Bob

Mary

Alice

Sue
2008: *Factorization meets the neighborhood: a multifaceted collaborative filtering model*, Y. Koren, ACM SIGKDD

- Merges neighborhood models with latent factor models
- Latent factor models
  - good to capture weak signals in the overall data
- Neighborhood models
  - good at detecting strong relationships between close items
- Combination in one prediction single function
  - Local search method such as stochastic gradient descent to determine parameters
  - Add penalty for high values to avoid over-fitting

\[
\hat{r}_{ui} = \mu + b_u + b_i + p_u^T q_i
\]

\[
\min_{p_u, q_i, b_u, b_i, (u,i) \in K} \sum (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda (\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2)
\]
Collaborative Filtering Issues

- **Pros:**
  - well-understood, works well in some domains, no knowledge engineering required

- **Cons:**
  - requires user community, sparsity problems, no integration of other knowledge sources, no explanation of results
Content-based recommendation
Content-based recommendation

- While CF – methods do not require any information about the items,
  - it might be reasonable to exploit such information; and
  - recommend fantasy novels to people who liked fantasy novels in the past

- What do we need:
  - some information about the available items such as the genre ("content")
  - some sort of user profile describing what the user likes (the preferences)

- The task:
  - learn user preferences
  - locate/recommend items that are "similar" to the user preferences
What is the "content"?

- The genre is actually not part of the content of a book
- Most CB-recommendation methods originate from Information Retrieval (IR) field:
  - goal is to find and rank interesting text documents (news articles, web pages)
  - the item descriptions are usually automatically extracted (important words)
- Fuzzy border between content-based and "knowledge-based" RS
- Here:
  - classical IR-based methods based on keywords
  - no expert recommendation knowledge involved
  - User profile (preferences) are rather learned than explicitly elicited
Content representation and item similarities

- Simple approach
  - Compute the similarity of an unseen item with the user profile based on the keyword overlap (e.g. using the Dice coefficient)

\[
sim(b_i, b_j) = \frac{2 \times |\text{keywords}(b_i) \cap \text{keywords}(b_j)|}{|\text{keywords}(b_i)| + |\text{keywords}(b_j)|}
\]

- Or combine multiple metrics in a weighted approach
Term-Frequency - Inverse Document Frequency (TF-IDF)

- Simple keyword representation has its problems
  - in particular when automatically extracted as
    - not every word has similar importance
    - longer documents have a higher chance to have an overlap with the user profile

- Standard measure: TF-IDF
  - Encodes text documents in multi-dimensional Euclidian space
    - weighted term vector
  - TF: Measures, how often a term appears (density in a document)
    - assuming that important terms appear more often
    - normalization has to be done in order to take document length into account
  - IDF: Aims to reduce the weight of terms that appear in all documents
TF-IDF

- **Compute the overall importance of keywords**
  - Given a keyword $i$ and a document $j$
    \[
    TF-IDF(i, j) = TF(i, j) \times IDF(i)
    \]

- **Term frequency (TF)**
  - Let $freq(i, j)$ number of occurrences of keyword $i$ in document $j$
  - Let $maxOthers(i, j)$ denote the highest number of occurrences of another keyword of $j$
    \[
    TF(i, j) = \frac{freq(i, j)}{maxOthers(i, j)}
    \]

- **Inverse Document Frequency (IDF)**
  - $N$: number of all recommendable documents
  - $n(i)$: number of documents in which keyword $i$ appears
    \[
    IDF(i) = \log \frac{N}{n(i)}
    \]
### Example TF-IDF representation

<table>
<thead>
<tr>
<th></th>
<th>Antony and Cleopatra</th>
<th>Julius Caesar</th>
<th>The Tempest</th>
<th>Hamlet</th>
<th>Othello</th>
<th>Macbeth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Antony</td>
<td>5.25</td>
<td>3.18</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.35</td>
</tr>
<tr>
<td>Brutus</td>
<td>1.21</td>
<td>6.1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Caesar</td>
<td>8.59</td>
<td>2.54</td>
<td>0</td>
<td>1.51</td>
<td>0.25</td>
<td>0</td>
</tr>
<tr>
<td>Calpurnia</td>
<td>0</td>
<td>1.54</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Cleopatra</td>
<td>2.85</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mercy</td>
<td>1.51</td>
<td>0</td>
<td>1.9</td>
<td>0.12</td>
<td>5.25</td>
<td>0.88</td>
</tr>
<tr>
<td>worser</td>
<td>1.37</td>
<td>0</td>
<td>0.11</td>
<td>4.15</td>
<td>0.25</td>
<td>1.95</td>
</tr>
</tbody>
</table>

Figure taken from http://informationretrieval.org
More on the vector space model

- **Vectors are usually long and sparse**

- **Improvements**
  - remove stop words ("a", "the", ..)
  - use stemming
  - size cut-offs (only use top n most representative words, e.g. around 100)
  - use additional knowledge, use more elaborate methods for feature selection
  - detection of phrases as terms (such as United Nations)

- **Limitations**
  - semantic meaning remains unknown
  - example: usage of a word in a negative context
    - "there is nothing on the menu that a vegetarian would like.."

- **Usual similarity metric to compare vectors: Cosine similarity (angle)**
Recommending items

- **Simple method: nearest neighbors**
  - Given a set of documents $D$ already rated by the user (like/dislike)
    - Find the $n$ nearest neighbors of a not-yet-seen item $i$ in $D$
    - Take these ratings to predict a rating/vote for $i$
    - (Variations: neighborhood size, lower/upper similarity thresholds..)
  - Good to model short-term interests / follow-up stories
  - Used in combination with method to model long-term preferences

- **Other methods**
  - Rocchio's feedback
  - Probabilistic methods
Probabilistic methods

- **Recommendation as classical text classification problem**
  - long history of using probabilistic methods

- **Simple approach:**
  - 2 classes: hot/cold
  - simple Boolean document representation
  - calculate probability that document is hot/cold based on Bayes theorem

<table>
<thead>
<tr>
<th>Doc-ID</th>
<th>recommender</th>
<th>intelligent</th>
<th>learning</th>
<th>school</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>?</td>
</tr>
</tbody>
</table>

\[ P(X|\text{Label}=1) = \frac{P(\text{recommender}=1|\text{Label}=1) \times P(\text{intelligent}=1|\text{Label}=1)}{P(\text{learning}=0|\text{Label}=1) \times P(\text{school}=0|\text{Label}=1)} \]

\[ = \frac{3/3 \times 2/3 \times 1/3 \times 2/3}{3/3 \times 2/3 \times 1/3 \times 2/3} \]

\[ \approx 0.149 \]
Improvements

- **Side note: Conditional independence of events does in fact not hold**
  - "New York", "Hong Kong"
  - Still, good accuracy can be achieved

- **Boolean representation simplistic**
  - positional independence assumed
  - keyword counts lost

- **More elaborate probabilistic methods**
  - e.g., estimate probability of term $v$ occurring in a document of class $C$ by relative frequency of $v$ in all documents of the class

- **Other linear classification algorithms (machine learning) can be used**
  - Support Vector Machines, ..

- **Use other information retrieval methods (used by search engines..)**
Limitations of content-based recommendation methods

- Keywords alone may not be sufficient to judge quality/relevance of a document or web page
  - up-to-dateness, usability, aesthetics, writing style
  - content may also be limited / too short
  - content may not be automatically extractable (multimedia)

- Ramp-up phase required
  - Some training data is still required
  - Web 2.0: Use other sources to learn the user preferences

- Overspecialization
  - Algorithms tend to propose "more of the same"
  - Or: too similar news items
Knowledge-Based Recommender Systems
Knowledge-Based Recommendation

- **Explicit domain knowledge**
  - Sales knowledge elicitation from domain experts
  - System mimics the behavior of experienced sales assistant
  - Best-practice sales interactions
  - Can guarantee “correct” recommendations (determinism) with respect to expert knowledge

- **Conversational interaction strategy**
  - Opposed to one-shot interaction
  - Elicitation of user requirements
  - Transfer of product knowledge ("educating users")
Knowledge-Based Recommendation

- 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
  - Logic-based knowledge descriptions (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 (Filtering)

Knowledge Base:

<table>
<thead>
<tr>
<th>LHS</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1 TRUE</td>
<td>Brand = Brand pref.</td>
</tr>
<tr>
<td>C2 Motives = Landscape</td>
<td>Low. foc. Length =&lt; 28</td>
</tr>
<tr>
<td>C3 TRUE</td>
<td>Price =&lt; Max. cost</td>
</tr>
</tbody>
</table>

Product catalogue:

<table>
<thead>
<tr>
<th>Powershot XY</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Brand</td>
<td>Canon</td>
</tr>
<tr>
<td>Lower focal length</td>
<td>35</td>
</tr>
<tr>
<td>Upper focal length</td>
<td>140</td>
</tr>
<tr>
<td>Price</td>
<td>420 EUR</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lumix</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Brand</td>
<td>Panasonic</td>
</tr>
<tr>
<td>Lower focal length</td>
<td>28</td>
</tr>
<tr>
<td>Upper focal length</td>
<td>112</td>
</tr>
<tr>
<td>Price</td>
<td>319 EUR</td>
</tr>
</tbody>
</table>

Current user:

<table>
<thead>
<tr>
<th>User model (SRS)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R1 Motives</td>
<td>Landscape</td>
</tr>
<tr>
<td>R2 Brand preference</td>
<td>Canon</td>
</tr>
<tr>
<td>R3 Max. cost</td>
<td>350 EUR</td>
</tr>
</tbody>
</table>
Constraint-based recommendation

- **A knowledge-based RS formulated as constraint satisfaction problem**

\[ CSP (X_I \cup X_U, D, SRS \cup KB \cup I) \]

- **Def.**
  - \(X_I, X_U\): Variables describing items and user model with domain \(D\) (e.g. lower focal length, purpose)
  - \(KB\): Knowledge base comprising constraints and domain restrictions (e.g. \(IF\) purpose=“on travel” \(THEN\) lower focal length < 28mm)
  - \(SRS\): Specific requirements of a user (e.g. purpose = “on travel”)
  - \(I\): Product catalog (e.g. \((id=1 \land lfl = 28mm) \lor (id=2 \land lfl= 35mm) \lor …\))

- **Solution: Assignment tuple \(\theta\) assigning values to all variables \(X_I\)**

\[-.6cm\]
\[
\begin{align*}
  &s.t. \ SRS \cup KB \cup I \cup \theta \text{ is satisfiable.}
\end{align*}
\]
Conversational strategies

- **Process consisting of multiple conversational moves**
  - Resembles natural sales interactions
  - Not all user requirements known beforehand
  - Customers are rarely satisfied with the initial recommendations

- **Different styles of preference elicitation:**
  - Free text query interface
  - Asking technical/generic properties
  - Images / inspiration
  - Proposing and Critiquing
Example: critiquing

- **Similarity-based navigation in item space**

- **Compound critiques**
  - More efficient navigation than with unit critiques
Limitations of knowledge-based recommendation methods

- **Cost of knowledge acquisition**
  - From domain experts
  - From users
  - From web resources

- **Accuracy of preference models**
  - Very fine granular preference models require many interaction cycles with the user or sufficient detailed data about the user
  - Preferences may depend on each other
  - Collaborative filtering models the preference of a user implicitly

- **Instability of preference models**
  - E.g. asymmetric dominance effects and decoy items
Hybridization Strategies
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
  - Avoid some of the shortcomings
  - Reach desirable properties not (or only inconsistently) present in parent individuals
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

- "Hybrid" user features:
  - Social features: Movies liked by user
  - Content features: Comedies liked by user, dramas liked by user
  - Hybrid features: users who like many movies that are comedies, ...
Parallelized hybridization design

- Output of several existing implementations combined
- Least invasive design
- Weighting or voting scheme applied
  - Weights can be learned dynamically
Parallelized hybridization design: Switching

- Special case of dynamic weights (all weights except one are 0)
- Requires an oracle that decides which recommender is used

- 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
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)
Pipelined hybridization designs: Cascade

- **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)

<table>
<thead>
<tr>
<th>Item</th>
<th>Rec 1</th>
<th>Rec 2</th>
<th>Rec cascaded</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item1</td>
<td>0.5</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>Item2</td>
<td>0</td>
<td>0.9</td>
<td>0</td>
</tr>
<tr>
<td>Item3</td>
<td>0.3</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>Item4</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>Item5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

© Dietmar Jannach, Markus Zanker and Gerhard Friedrich
Limitations and success of hybridization strategies

- Only few works that compare strategies from the meta-perspective
  - For instance, [Burke02]
  - Most datasets do not allow to compare different recommendation paradigms
    - i.e. ratings, requirements, item features, domain knowledge, critiques rarely available in a single dataset
  - Thus few conclusions that are supported by empirical findings
    - Monolithic: some preprocessing effort traded-in for more knowledge included
    - Parallel: requires careful matching of scores from different predictors
    - Pipelined: works well for two antithetic approaches

- Netflix competition – “stacking” recommender systems
  - Weighted design based on >100 predictors – recommendation functions
  - Adaptive switching of weights based on user model, parameters (e.g. number of ratings in one session)
Evaluating Recommender Systems
Agenda

- What is the current state-of-practice?

- „How to“ from different perspectives:
  - Empirical research principles
  - Information Retrieval
  - Machine Learning
  - HCI and Decision Support

- Outlook
What is popular?

- **Small quantitative survey in the literature (Jannach et al., 2010)**
  - Evaluation designs ACM TOIS 2004-2010
  - In total 15 articles on RS
  - Nearly 50% movie domain
  - 80% offline experimentation
  - 2 user experiments under lab conditions
  - 1 qualitative research

- **Wide-scale survey (Jannach et al., 2012)**
  - 330 publications from a predefined set of conferences and journals
  - 73 journal publications
  - 20% of total from IS community
# Publication outlets 1/2

<table>
<thead>
<tr>
<th>Conference</th>
<th>Field</th>
<th>#Pub.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACM Conf. on Human Factors in Comp. Syst. (CHI)</td>
<td>CS</td>
<td>13</td>
</tr>
<tr>
<td>ACM Conf. on Recommender Syst. (RecSys)</td>
<td>CS</td>
<td>86</td>
</tr>
<tr>
<td>Int. Conf. on Int. User Interfaces (IUI)</td>
<td>CS</td>
<td>17</td>
</tr>
<tr>
<td>Int. Conf. on Knowl. Disc. and DM (SIGKDD)</td>
<td>CS</td>
<td>22</td>
</tr>
<tr>
<td>Int. Conf. on Res. and Dev. in IR (SIGIR)</td>
<td>CS</td>
<td>33</td>
</tr>
<tr>
<td>Int. Conf. on World Wide Web (WWW)</td>
<td>CS</td>
<td>21</td>
</tr>
<tr>
<td>Int. Joint Conf. on AI (IJCAI)</td>
<td>CS</td>
<td>13</td>
</tr>
<tr>
<td>AAAI Conf. on AI (AAAI)</td>
<td>CS</td>
<td>10</td>
</tr>
<tr>
<td>Int. Conf. on Data Mining (ICDM)</td>
<td>CS</td>
<td>5</td>
</tr>
<tr>
<td>Americas Conf. on Information Systems (AMCIS)</td>
<td>IS</td>
<td>8</td>
</tr>
<tr>
<td>European Conf. on Information Systems (ECIS)</td>
<td>IS</td>
<td>6</td>
</tr>
<tr>
<td>Int. Conf. on Information Systems (ICIS)</td>
<td>IS</td>
<td>7</td>
</tr>
<tr>
<td>Med. Conf. on Information Systems (MCIS)</td>
<td>IS</td>
<td>5</td>
</tr>
<tr>
<td>Pac. Asia Conf. on Information Systems (PACIS)</td>
<td>IS</td>
<td>11</td>
</tr>
<tr>
<td>Journal</td>
<td>Field</td>
<td>#Pub.</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>ACM Trans. on Intell. Syst. and Techn. (TOIST)</td>
<td>CS</td>
<td>6</td>
</tr>
<tr>
<td>ACM Trans. on the Web (TWeb)</td>
<td>CS</td>
<td>5</td>
</tr>
<tr>
<td>AI Communications</td>
<td>CS</td>
<td>12</td>
</tr>
<tr>
<td>IEEE Intelligent Systems</td>
<td>CS</td>
<td>14</td>
</tr>
<tr>
<td>Int. Jrnl. of Human Computer Studies (IJHCS)</td>
<td>CS</td>
<td>5</td>
</tr>
<tr>
<td>World Wide Web (WWW)</td>
<td>CS</td>
<td>3</td>
</tr>
<tr>
<td>Inf. Syst. Res. (ISR)</td>
<td>IS</td>
<td>3</td>
</tr>
<tr>
<td>Int. Jrnl. of Electronic Comm. (IJEC)</td>
<td>IS</td>
<td>7</td>
</tr>
<tr>
<td>Jrnl. of Mgt. Information Systems (JMIS)</td>
<td>IS</td>
<td>7</td>
</tr>
<tr>
<td>Mgt. Information Systems Quarterly (MISQ)</td>
<td>IS</td>
<td>2</td>
</tr>
</tbody>
</table>
Research contributions

<table>
<thead>
<tr>
<th>Type of contribution</th>
<th>IS outlets</th>
<th>CS outlets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical artifacts (i.e. novel algorithms)</td>
<td>24 (36,9%)</td>
<td>189 (71,3%)</td>
</tr>
<tr>
<td>Empirical research</td>
<td>21 (32,3%)</td>
<td>18 (6,8%)</td>
</tr>
<tr>
<td>Both</td>
<td>9 (13,8%)</td>
<td>43 (16,2%)</td>
</tr>
<tr>
<td>Other</td>
<td>11 (16,9%)</td>
<td>15 (5,7%)</td>
</tr>
</tbody>
</table>

- **Topics:**
  - Social and semantic web (25% of CS papers, only 6% of IS papers)
  - Scalability, privacy (15% of CS papers)
  - Cold-start recommendations (10% of CS papers, 5% of IS papers)
  - UI design (CS 5,8%, IS 12,3%)
  - Transparency (CS 6,8%, IS 10,8%)
Recommended items

![Graph showing recommended items with CS and IS publications]

Markus Zanker, University Klagenfurt, markus.zanker@uni-klu.ac.at
Most popular CF techniques

- User-based neighborhood models
- Item-based neighborhood models
- Clustering methods
- Latent factor models
- Trust networks
- Graph-based and random walk...
- Other Machine Learning techniques

CS publications
IS publications
Most popular datasets

![Bar chart showing popularity of different datasets]

Proprietary data, MovieLens, Netflix, Yahoo!, Epinions, Delicious, Synthetic data sets, EachMovie, IMDB, Amazon.com, YouTube.com, Others

CS publications and IS publications are indicated by different colors.
Methodologies

- Offline experiments: 250
- User studies: 100
- Simulation on synthetic data: 50
- Formal proof: 10
- Case studies: 5
- Literature review: 5

CS publications

IS publications
<table>
<thead>
<tr>
<th>Measure</th>
<th>IS outlets</th>
<th>CS outlets</th>
</tr>
</thead>
<tbody>
<tr>
<td>IR measures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Precision and Recall</td>
<td>12</td>
<td>115</td>
</tr>
<tr>
<td>F1</td>
<td>2</td>
<td>20</td>
</tr>
<tr>
<td>Rank measures (e.g. NDCG)</td>
<td>9</td>
<td>27</td>
</tr>
<tr>
<td>ROC curve</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>Area under ROC</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>ML measures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean absolute Error</td>
<td>6</td>
<td>57</td>
</tr>
<tr>
<td>Root Mean Squared Error</td>
<td>0</td>
<td>49</td>
</tr>
<tr>
<td>Application quality</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computation time</td>
<td>2</td>
<td>28</td>
</tr>
<tr>
<td>Coverage metrics</td>
<td>2</td>
<td>28</td>
</tr>
<tr>
<td>Decision support quality</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perceived utility or user satisfaction</td>
<td>11</td>
<td>7</td>
</tr>
<tr>
<td>Online conversion</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>Diversity metrics</td>
<td>0</td>
<td>10</td>
</tr>
</tbody>
</table>
Agenda

- What is the current state-of-practice?

- „How to“ from different perspectives:
  - Empirical research principles
  - Information Retrieval
  - Machine Learning
  - HCI and Decision Support

- Outlook
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?

<table>
<thead>
<tr>
<th>Subject</th>
<th>Online customers, students, historical online sessions, computers, ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research method</td>
<td>Experiments, quasi-experiments, non-experimental research</td>
</tr>
<tr>
<td>Setting</td>
<td>Lab, real-world scenarios</td>
</tr>
</tbody>
</table>
Evaluation settings

- **Lab studies**
  - Expressly created for the purpose of the study
  - Extraneous variables can be controlled more easily by selecting study participants
  - But doubts may exist about participants motivated by money or prizes

- **Participants should behave as they would in a real-world environment**

- **Field studies**
  - Conducted in an preexisting real-world environment
  - Users are intrinsically motivated to use a system
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, groups of recommended items, explanation strategies ...
  - Categories of manipulated variable(s): content-based RS, collaborative RS
Experiment designs
Agenda

- What is the current state-of-practice?

- „How to“ from different perspectives:
  - Empirical research principles
  - Information Retrieval
  - Machine Learning
  - HCI and Decision Support

- Outlook
Evaluation in information retrieval (IR)

- **Historical Cranfield collection (late 1950s)**
  - 1,398 journal article abstracts
  - 225 queries
  - Exhaustive relevance judgements (over 300K)

- **Ground truth established by human domain experts**

<table>
<thead>
<tr>
<th>Prediction</th>
<th>Reality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rated Good</td>
<td>Actually Good</td>
</tr>
<tr>
<td>Rated Bad</td>
<td>Actually Bad</td>
</tr>
<tr>
<td></td>
<td>True Positive (tp)</td>
</tr>
<tr>
<td></td>
<td>False Negative (fn)</td>
</tr>
</tbody>
</table>

**All good items**

**All recommended items**
Metrics: Precision and Recall

- Recommendation is viewed as information retrieval task:
  - Retrieve (recommend) all items which are predicted to be “good”.

- 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
    
    \[
    \text{Precision} = \frac{tp}{tp + fp} = \frac{|\text{good movies recommended}|}{|\text{all recommendations}|}
    \]

- 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
    
    \[
    \text{Recall} = \frac{tp}{tp + fn} = \frac{|\text{good movies recommended}|}{|\text{all good movies}|}
    \]
Precision vs. Recall

- E.g. typically when a recommender system is tuned to increase precision, recall decreases as a result (or vice versa)
F₁ Metric

- The F₁ Metric attempts to combine Precision and Recall into a single value for comparison purposes.
  - May be used to gain a more balanced view of performance

\[
F₁ = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}
\]

- The F₁ Metric gives equal weight to precision and recall
  - Other F_β metrics weight recall with a factor of \( \beta \).
**Metrics: Rank position matters**

**For a user:**

- **Acturally good**
  - Item 237
  - Item 899

- **Recommended (predicted as good)**
  - Item 345
  - Item 237
  - Item 187

- **Rank metrics extend recall and precision to take the positions of correct items in a ranked list into account**
  - Relevant items are more useful when they appear earlier in the recommendation list
  - Particularly important in recommender systems as lower ranked items may be overlooked by users

© Dietmar Jannach, Markus Zanker and Gerhard Friedrich
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.

\[
\text{rankscore} = \frac{\text{rankscore}_p}{\text{rankscore}_{\text{max}}}
\]

\[
\text{rankscore}_p = \sum_{i \in h} 2^{\frac{\text{rank}(i) - 1}{\alpha}}
\]

\[
\text{rankscore}_{\text{max}} = \sum_{i=1}^{\vert T \vert} 2^{\frac{i-1}{\alpha}}
\]

Where:
- \( h \) is the set of correctly recommended items, i.e. hits
- \( \text{rank} \) returns the position (rank) of an item
- \( T \) is the set of all items of interest
- \( \alpha \) is the ranking half life, i.e. an exponential reduction factor
Metrics: Liftindex

- Assumes that ranked list is divided into 10 equal deciles $S_i$, where
  $$\sum_{i=1}^{10} S_i = |h|$$
  - Linear reduction factor

- Liftindex:
  $$\text{liftindex} = \begin{cases} 
\frac{1 \times S_1 + 0.9 \times S_2 + \ldots + 0.1 \times S_{10}}{\sum_{i=1}^{10} S_i} : & \text{if } |h| > 0 \\
0 : & \text{else}
\end{cases}$$

  > $h$ is the set of correct hits
Metrics: Normalized Discounted Cumulative Gain

- **Discounted cumulative gain (DCG)**
  - Logarithmic reduction factor
  
  \[
  DCG_{pos} = rel_1 + \sum_{i=2}^{pos} \frac{rel_i}{\log_2 i}
  \]
  
  Where:
  - \( pos \) denotes the position up to which relevance is accumulated
  - \( rel_i \) returns the relevance of recommendation at position \( i \)

- **Idealized discounted cumulative gain (IDCG)**
  - Assumption that items are ordered by decreasing relevance
  
  \[
  IDCG_{pos} = rel_1 + \sum_{i=2}^{\lceil k \rceil - 1} \frac{rel_i}{\log_2 i}
  \]

- **Normalized discounted cumulative gain (nDCG)**
  - Normalized to the interval \([0..1]\)
Example

- **Assumptions:**
  - $|\mathcal{T}| = 3$
  - Ranking half life (alpha) = 2

<table>
<thead>
<tr>
<th>Rank</th>
<th>Hit?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✔</td>
</tr>
<tr>
<td>2</td>
<td>✗</td>
</tr>
<tr>
<td>3</td>
<td>✗</td>
</tr>
<tr>
<td>4</td>
<td>✗</td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

\[ \text{rankscore} = \frac{\text{rankscore}_p}{\text{rankscore}_{\text{max}}} \approx 0.71 \]

\[ \text{rankscore}_p = \frac{1}{2^{-1}} + \frac{1}{2^{-1}} + \frac{1}{2^{-1}} = 1.56 \]

\[ \text{rankscore}_{\text{max}} = \frac{1}{2^{-1}} + \frac{1}{2^{-1}} + \frac{1}{2^{-1}} = 2.21 \]

\[ nDCG_5 \frac{DCG_5}{IDCG_5} \approx 0.81 \]

\[ DCG_5 = \frac{1}{\log_2 2} + \frac{1}{\log_2 3} + \frac{1}{\log_2 4} = 2.13 \]

\[ IDCG_5 = 1 + \frac{1}{\log_2 2} + \frac{1}{\log_2 3} = 2.63 \]

\[ \text{liftindex} = \frac{0.8 \times 1 + 0.6 \times 1 + 0.4 \times 1}{3} = 0.6 \]
Example cont.

- Reducing the ranking half life (alpha) = 1

<table>
<thead>
<tr>
<th>Rank</th>
<th>Hit?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>X</td>
</tr>
<tr>
<td>3</td>
<td>X</td>
</tr>
<tr>
<td>4</td>
<td>X</td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

\[
\text{rankscore} = \frac{\text{rankscore}_p}{\text{rankscore}_{\text{max}}} = 0.5
\]

\[
\text{rankscore}_p = \frac{1}{2^{1-1}} + \frac{1}{2^{1-1}} + \frac{1}{2^{1-1}} = 0.875
\]

\[
\text{rankscore}_{\text{max}} = \frac{1}{2^{1-1}} + \frac{1}{2^{1-1}} + \frac{1}{2^{1-1}} = 1.75
\]

Rankscore (exponential reduction) \(\ll\) Liftscore (linear reduction) \(\ll\) NDCG (log. reduction)
Average Precision

- Average Precision (AP) is a ranked precision metric that places emphasis on highly ranked correct predictions (hits)

- Essentially it is the average of precision values determined after each successful prediction, i.e.

\[
AP = \frac{1}{3} \left( \frac{1}{1} + \frac{2}{4} + \frac{3}{5} \right) = \frac{21}{30} = 0.7
\]

\[
AP = \frac{1}{3} \left( \frac{1}{2} + \frac{2}{3} + \frac{3}{4} \right) = \frac{23}{36} \approx 0.639
\]
Agenda

- What is the current state-of-practice?

- „How to“ from different perspectives:
  - Empirical research principles
  - Information Retrieval
  - Machine Learning
  - HCI and Decision Support

- Outlook
RS from a ML perspective

- Recommendation is concerned with learning from noisy observations \((x,y)\), where \(f(x) = \hat{y}\) has to be determined such that \(\sum (\hat{y} - y)^2\) is minimal.

- A huge variety of different learning strategies have been applied trying to estimate \(f(x)\)
  - Non parametric neighborhood models
  - MF models, SVMs, Neural Networks, Bayesian Networks,...
Error measures

- **Datasets with items rated by users**
  - MovieLens datasets 100K-10M ratings
  - Netflix 100M ratings

- **Historic user ratings constitute ground truth**

- **Metrics measure error rate**
  - 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
    \[
    RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (p_i - r_i)^2}
    \]
Data sparsity

- Natural datasets include historical interaction records of real users
  - Explicit user ratings
  - Datasets extracted from web server logs (implicit user feedback)

- Sparsity of a dataset is derived from ratio of empty and total entries in the user-item matrix:
  - Sparsity $= 1 - |R|/|I| \cdot |U|$
  - $R$ = ratings
  - $I$ = items
  - $U$ = users
### Example

| Nr. | UserID | MovieID | Rating ($r_i$) | Prediction ($p_i$) | $|p_i - r_i|$ | $(p_i - r_i)^2$ |
|-----|--------|---------|---------------|-------------------|-------------|----------------|
| 1   | 1      | 134     | 5             | 4.5               | 0.5         | 0.25           |
| 2   | 1      | 238     | 4             | 5                 | 1           | 1              |
| 3   | 1      | 312     | 5             | 5                 | 0           | 0              |
| 4   | 2      | 134     | 3             | 5                 | 2           | 4              |
| 5   | 2      | 767     | 5             | 4.5               | 0.5         | 0.25           |
| 6   | 3      | 68      | 4             | 4.1               | 0.1         | 0.01           |
| 7   | 3      | 212     | 4             | 3.9               | 0.1         | 0.01           |
| 8   | 3      | 238     | 3             | 3                 | 0           | 0              |
| 9   | 4      | 68      | 4             | 4.2               | 0.2         | 0.04           |
| 10  | 4      | 112     | 5             | 4.8               | 0.2         | 0.04           |

- **MAE = 0.46**
- **RMSE = 0.75**

Removing line nr. 4
- **MAE = 0.29**
- **RMSE = 0.42**

Removing lines 1,2,4,5
- **MAE = 0.1**
- **RMSE = 0.13**
Dilemma of establishing ground truth

- **IR measures are frequently applied, however:**

<table>
<thead>
<tr>
<th>Offline experimentation</th>
<th>Online experimentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ratings, transactions</td>
<td>Ratings, feedback</td>
</tr>
<tr>
<td>Historic session (not all recommended items are rated)</td>
<td>Live interaction (all recommended items are rated)</td>
</tr>
<tr>
<td>Ratings of unrated items unknown, but interpreted as “bad” (default assumption, user tend to rate only good items)</td>
<td>“Good/bad” ratings of not recommended items are unknown</td>
</tr>
<tr>
<td>If default assumption does not hold: True positives may be too small False negatives may be too small</td>
<td>False/true negatives cannot be determined</td>
</tr>
<tr>
<td>Precision may increase</td>
<td>Precision ok</td>
</tr>
<tr>
<td>Recall may vary</td>
<td>Recall questionable</td>
</tr>
</tbody>
</table>

Results from offline experimentation have limited predictive power for online user behavior.
Offline experimentation

- **Netflix competition**
  - Web-based movie rental
  - Prize of $1,000,000 for accuracy improvement (RMSE) 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 for leaderboard
    - Test set – used by Netflix to determine winner
Methodology

- Setting to ensure internal validity:
  - One randomly selected share of known ratings (training set) used as input to train the algorithm and build the model
  - Model allows the system to compute recommendations at runtime
  - Remaining share of withheld ratings (testing set) required as ground truth to evaluate the model’s quality
  - To ensure the reliability of measurements the random split, model building and evaluation steps are repeated several times

- N-fold cross validation is a stratified random selection procedure
  - N disjunct fractions of known ratings with equal size (1/N) are determined
  - N repetitions of the model building and evaluation steps, where each fraction is used exactly once as a testing set while the other fractions are used for training
  - Setting N to 5 or 10 is popular
Analysis of results

- **Are observed differences statistically meaningful or due to chance?**
  - Standard procedure for testing the statistical significance of two deviating metrics is the pairwise analysis of variance (ANOVA)
  - Null hypothesis $H_0$: observed differences have been due to chance
  - If outcome of test statistics rejects $H_0$, significance of findings can be reported

- **Practical importance of differences?**
  - Size of the effect and its practical impact
  - External validity or generalizability of the observed effects
Agenda

- What is the current state-of-practice?

- „How to“ from different perspectives:
  - Empirical research principles
  - Information Retrieval
  - Machine Learning
  - HCI and Decision Support

- Outlook
Online experimentation

- Effectiveness of different algorithms for recommending cell phone games
  [Jannach, Hegelich 09]

- Involved 150,000 users on a commercial mobile internet portal

- Comparison of recommender methods

- Random assignment of users to a specific method
Experimental Design

- A representative sample 155,000 customers were extracted from visitors to site during the evaluation period
  - These were split into 6 groups of approximately 22,300 customers
  - Care was taken to ensure that customer profiles contained enough information (ratings) for all variants to make a recommendation
  - Groups were chosen to represent similar customer segments

- A catalog of 1,000 games was offered

- A five-point ratings scale ranging from -2 to +2 was used to rate items
  - Due to the low number of explicit ratings, a click on the “details” link for a game was interpreted as an implicit “0” rating and a purchase as a “1” rating

- Hypotheses on personalized vs. non-personalized recommendation techniques and their potential to
  - Increase conversion rate (i.e. the share of users who become buyers)
  - Stimulate additional purchases (i.e. increase the average shopping basket size)
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 on answering research questions about how and why
    - E.g. answer questions like: *How recommendation technology contributed to Amazon.com’s becomes the world’s largest book retailer?*
  - 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

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

[Delgado and Davidson, ENTER 2002]
Interpreting the Results

- The nature of this research design means that questions of causality cannot be answered (lack of random assignments), such as
  - Are users of the recommender systems more likely convert?
  - Does the recommender system itself cause users to convert?
  Some hidden exogenous variable might influence the choice of using RS as well as conversion.

- 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 [Jannach et al., JITT 2009]
  - Electronic consumer products
Observational research

- Increased demand in niches/long tail products
  - Books ranked above 250,000 represent >29% of sales at Amazon, approx. 2.3 million books [Brynjolfsson et al., Mgt. Science, 2003]
  - Ex post from webshop data [Zanker et al., EC-Web, 2006]
Agenda

- What is the current state-of-practice?

- „How to“ from different perspectives:
  - Empirical research principles
  - Information Retrieval
  - Machine Learning
  - HCI and Decision Support

- Outlook
Reality check regarding $F_1$ and accuracy measures for RS

- **Real value lies in increasing conversions**
  - ...and satisfaction with bought items, low churn rate

- **Some reasons why it might be a fallacy to think $F_1$ on historical data is a good estimate for real conversion:**
  - Recommendation can be self-fulfilling prophecy
    - Users’ preferences are not invariant, but can be constructed [ALP03]
  - Position/Rank is what counts (e.g. serial position effects)
    - Actual choices are heavily biased by the item’s position [FFG+07]
  - Smaller recommendation sets increase users’ confidence in decision making
    - Effect of choice overload - large sets at the same time increase choice difficulty and reduce choice satisfaction [BKW+10]
  - Inclusion of weak (dominated) items increases users’ confidence
    - Replacing some recommended items by decoy items fosters choice towards the remaining options [TF09]
Bounded rationality

- Framing and reference dependence, e.g.
  - Presentation of the decision problem and its recommendations to the user (e.g. gains and losses)
  - Bias towards initial anchor point (in conversational RS)

- Cognitive consistency theory
  - Preferences are re-constructed in the course of decision making in order to avoid conflicts

- Serial position effects
  - Primacy and recency

- Decoy effects
  - Items below pareto frontier
  - Dominance relationships
Implications for RS

- Preferences cannot be assumed to be stable during RS interaction
  - Sequence, content and wording matter

- Preference reasoning
  - Independence of preferences may not be assumed, impacts the computation of the preference score

- Presentation/recommendation of items
  - Serial positions significantly influence the decision
  - Inclusion of decoy items
Which one will the majority select?

**Primacy and Decoy effect**

<table>
<thead>
<tr>
<th>Item</th>
<th>Item A</th>
<th>Item B</th>
<th>Decoy A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute 1</td>
<td>High</td>
<td>Low</td>
<td>Medium</td>
</tr>
<tr>
<td>Attribute 2</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
</tr>
</tbody>
</table>

**Primacy effect and opposite Decoy effect**

<table>
<thead>
<tr>
<th>Item</th>
<th>Item B</th>
<th>Item A</th>
<th>Decoy B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute 1</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Attribute 2</td>
<td>High</td>
<td>Low</td>
<td>Medium</td>
</tr>
</tbody>
</table>

**Primacy and Decoy effect**

<table>
<thead>
<tr>
<th>Item</th>
<th>Item B</th>
<th>Item A</th>
<th>Decoy B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute 2</td>
<td>High</td>
<td>Low</td>
<td>Medium</td>
</tr>
<tr>
<td>Attribute 1</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
</tr>
</tbody>
</table>
Discussion & summary

- General principles of empirical research and current state of practice in evaluating recommendation techniques were presented.
- Focus on how to perform empirical evaluations on historical datasets.
- Discussion about different methodologies and metrics for measuring the accuracy or coverage of recommendations.
- Overview of which research designs are commonly used in practice.
- From a technical point of view, measuring the accuracy of predictions is a well accepted evaluation goal.
  - but other aspects that may potentially impact the overall effectiveness of a recommendation system remain largely under developed.
Outlook

- Additional topics covered by the book “Recommender Systems - An Introduction”
  - Case study on the Mobile Internet
  - Attacks on CF Recommender Systems
  - Recommender Systems in the next generation Web (Social Web, Semantic Web)
  - Consumer decision making
  - Recommending in ubiquitous environments

- “RS research will become much more diverse”
  - Various forms of feedback mechanisms and preference representation
  - More focus on interfaces, interaction processes, explaining and trust-building
  - Plurality of evaluation methods complementing offline experiments

- More focus on causal relationships
  - When, where and how to recommend?
  - Consumer / sales psychology
  - Consumer decision making theories
Thank you for your attention!

Questions?
Questions?
Questions?

Dietmar Jannach
e-Services Research Group
Department of Computer Science
TU Dortmund, Germany
M: dietmar.jannach@tu-dortmund.de
P: +49 231 755 7272

Markus Zanker
Intelligent Systems and Business Informatics
Institute of Applied Informatics
University Klagenfurt, Austria
M: markus.zanker@aau.at
P: +43 463 2700 3705

http://www.recommenderbook.net

Recommender Systems – An Introduction by
Dietmar Jannach, Markus Zanker, Alexander Felfernig and Gerhard Friedrich
Cambridge University Press, 2010

http://recsys.acm.org

© Dietmar Jannach, Markus Zanker and Gerhard Friedrich
References


References


