CS630 Representing and Accessing Digital Information

Recommender Systems

Thorsten Joachims
Cornell University

Recommender Systems

- **Task definition**
- **Item-to-Item Similarity**
- **User-to-User Similarity**
- **Recommendation**
  - Content-based methods
  - Collaborative nearest neighbor methods
  - Collaborative model-based methods

Motivation

- **Matchmaking between users and items**
  - Filtering
  - Exploration
  - Marketing
  - etc.

Example: Amazon

Data

- **Explicit feedback**
  - Ratings
  - Reviews
  - Auctions
  - etc.
- **Implicit feedback**
  - Page visits
  - Purchase data
  - Browsing paths
  - etc.

Types of Recommendations

- **Item-to-Item associations**
  - More pages like this
  - “Users who bought this book also bought X”
- **User-to-User associations**
  - Which other user has similar interests?
- **User-to-Item associations**
  - Rating history describes user
  - Items are described by attributes
  - Items are described by ratings of other users

Recommender Systems

- **Task definition**
- **Item-to-Item Similarity**
- **User-to-User Similarity**
- **Recommendation**
  - Content-based methods
  - Collaborative nearest neighbor methods
  - Collaborative model-based methods

Item-to-Item Recommendation

- **Content-based approach**
  - Item is described by a set of attributes
    - Movies: e.g director, genre, year, actors
    - Documents: bag-of-word
  - Similarity metric defines relationship between items
    - e.g. cosine similarity
  - Examples
    - “related pages” in search engine
    - Google News

Item-to-Item Recommendation

- **Collaborative filtering**
  - Item is described by user interactions
    - Matrix V of n (number of users) rows and m (number of items) columns
    - Elements of matrix V is user feedback
  - Examples:
    - Rating given to item by each user
    - Users who viewed this item
  - Similarity metric between items
    - E.g. cosine

Recommender Systems

- **Task definition**
- **Item-to-Item Similarity**
- **User-to-User Similarity**
- **Recommendation**
  - Content-based methods
  - Collaborative nearest neighbor methods
  - Collaborative model-based methods

User-to-User Similarity

- **User is described by his/her ratings**
  - Matrix V of n (number of users) rows and m (number of items) columns. Elements of matrix V is user feedback.
- **Normalization**
  - Mean rating of user a $\bar{v}_a = \frac{1}{l_i}\sum_i v_{ai}$ l_i = # of ratings
- **Similarity measure between users**
  - Cosine $sim(a,b) = \sum_i \frac{v_{ai}}{\sqrt{\sum_k v_{ak}^2}} \frac{v_{bi}}{\sqrt{\sum_k v_{bk}^2}}$
  - Correlation $sim(a,b) = \sum_i \frac{(v_{ai}-\bar{v}_a)(v_{bi}-\bar{v}_b)}{\sqrt{\sum_k (v_{ak}-\bar{v}_a)^2 \sum_k (v_{ak}-\bar{v}_a)^2}}$
- **Problems**
  - data sparseness
  - Unknown vs. unseen

Recommender Systems

- **Task definition**
- **Item-to-Item Similarity**
- **User-to-User Similarity**
- **Recommendation**
  - Content-based methods
  - Collaborative model-based methods
  - Collaborative nearest neighbor methods

Content-Based Recommendation

- **Use the ratings as feedback**
  - Binary
  - Ordinal
- **Represent items using a set of features**
  - Movies: e.g director, genre, year, actors
  - Documents: bag-of-word
- **Learn function that predicts the rating for un-rated items**
  - Learn one function per user
  - Can use any machine learning method
- **Strengths and Weaknesses?**

Collaborative Nearest-Neighbor Methods

- **Idea: Recommend items that similar users like**
- **User is described by his/her ratings**
  - Matrix V of n (number of users) rows and m (number of items) colums. Elements of matrix V is user feedback.
- **Normalization**
  - Mean rating of user a $\bar{v}_a = \frac{1}{l_i}\sum_i v_{ai}$ l_i = # of ratings
- **Similarity measure between users**
  - Cosine $sim(a,b) = \sum_i \frac{v_{ai}}{\sqrt{\sum_k v_{ak}^2}} \frac{v_{bi}}{\sqrt{\sum_k v_{bk}^2}}$ (or Correlation)
- **Prediction via linear combination**

$$\hat{v}_{aj} = \bar{v}_a + \frac{1}{\sum_b |sim(a,b)|} \sum_i sim(a,b)(v_{bj} - \bar{v}_b)$$

Collaborative Model-Based Methods

- **Idea**
  - Learn a model offline
  - Use model to make predictions online
- **Approach: Model joint density of user ratings**
  - Cluster users
  - Approximate joint density with mixture model
- **Approach: Learn conditional model for each item**
  - Learn prediction rules
  - One rule for each item

Joint Density Modeling

- **Idea: Estimate distribution of ratings via mixture model**

$$P(v_1, ..., v_m) = \sum_{k=1}^{K} P(v_1, ..., v_m | u = k) P(u = k)$$

- **Assumptions:**
  - K disjoint user-interest classes
  - Each user is in exactly one interest class
  - Users within one class behave according to simple model, e.g. $P(v_1, ..., v_m | u = k) = \prod_{j=1}^{m} P(v_j | u = k)$
- **Prediction**
  - Classify user via mode $u = \arg\max_{k=1}^{K} P(\bar{v}_1, ..., \bar{v}_l | u = k) P(u = k)$
  - Bayesian classification
- **Extensions**
  - User can be in multiple classes (Hofmann & Puzicha, 1999)

Conditional Models

- **Idea: Learn a prediction rule for each item**

$$\bar{v}_{aj} = h(v_{a1}, ..., v_{am})$$

- **Learning Problem**
  - Classification: Predict rating class [Heckerman et al., 2000]
  - Regression: Predict rating score
  - Ordinal Regression: Predict ranking of items [Cohen et al., 1999]
- **Challenges:**
  - Handling missing ratings
  - Computational expense for learning m models
  - No ratings for new products

Cold-Start Problem

- **Problem: new users have too few ratings for effective recommendation**
- **Idea: Combine ratings with other user attributes**
  - Demographic attributes
  - Attributes from other domains
  - Questionnaires
- **Challenges:**
  - Designing combined models
  - Trading-off user attributes with rating attributes

Evaluation

- **Batch Evaluation**
  - Use historical data
  - Split into training and test part on a per-user basis
  - k ratings to describe user, remaining ratings for testing
  - Problems?
- **Online Evaluation**
  - Install recommender system in operational system
  - Controlled experiment with control group
    - Does the recommender system increase sales?
    - Does the recommender system make users return more often?
    - etc.