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

#### 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

#### 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

### 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 \( i \): \( \bar{r}_i = \frac{1}{\# \text{of ratings}} \sum_j r_{ij} \)
- **Similarity measure between users**
  - Cosine: \( s_{cos}(i, j) = \frac{\sum_{k=1}^{m} r_{ik}r_{jk}}{\sqrt{\sum_{k=1}^{m} r_{ik}^2 \sum_{k=1}^{m} r_{jk}^2}} \)
  - Correlation: \( s_{corr}(i, j) = \frac{\sum_{k=1}^{m} (r_{ik} - \bar{r}_i)(r_{jk} - \bar{r}_j)}{\sqrt{\sum_{k=1}^{m} (r_{ik} - \bar{r}_i)^2 \sum_{k=1}^{m} (r_{jk} - \bar{r}_j)^2}} \)
- **Problems**
  - Data sparseness
  - Unknown vs. unseen
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) columns. Elements of matrix V is user feedback.
- Normalization
  - Mean rating of user $u$ $\mu_u = \frac{1}{t_u} \sum_{i} r_{ui}$ $t_u$ = # of ratings
- Similarity measure between users $\cosine(u, v) = \frac{\sum_{i} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i} r_{ui}^2 \cdot \sum_{i} r_{vi}^2}}$ (or Correlation)
- Prediction via linear combination
  $$\hat{v}_{ui} = \mu_u + \frac{1}{\sum_{i} |\text{sim}(a, b)|} \sum_{i} \text{sim}(a, b)(r_{u_i} - \mu_u)$$

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
  $$P(x_1, ..., x_n) = \sum_{k=1}^{K} P(x_1, ..., x_n | h_k) \cdot P(h_k)$$
- 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. $P(x_1, ..., x_n | h_k) = \prod_{i=1}^{n} P(x_i | h_k)$
- Prediction
  - Classify user via mode $\hat{h} = \arg \max_{h} P(x_1, ..., x_n | h) \cdot P(h)$
    - Bayesian classification
- Extensions
  - User can be in multiple classes (Hoffmann & Puriza, 1999)

Conditional Models

- Idea: Learn a prediction rule for each item
  $$\hat{y}_a = f(x_1, ..., x_n)$$
- 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.