Overview of Recommendation Systems

CIS 601 Graduate Seminar
Dr. Sunnie S. Chung

Naveen Baskaran
Suhas Mallesh
The user interacts with a Web interface.

The Web server software communicates with the recommender system to choose products to suggest to the user.

The recommender system, in this case a collaborative filtering system, uses its database of ratings of products to form neighborhoods and make recommendations.

The Web server software displays the recommended products to the user.
Three goals

- Producing high quality recommendations
- Performing many recommendation per seconds for millions of customers and products
- Achieving high coverage in the face of sparsity
Collaborative filtering

- One of the successful recommendation technique
- Matching customer preferences to other customer preferences
- Produces high quality recommendations
- But performance degrades with customers and products
- Technology needed for addressing very large scale problems
Existing approaches

- Uses data of neighborhood of likeminded customers
- Pearson correlation or cosine similarity for neighborhood formation

**Prediction**
- How much customer C will like product P?
- Calculates using weighted co-rated items between C and neighbor J
- Uses Correlation based algorithm
  \[ c_{p_{pred}} = (\bar{c} + \sum_{J \in rates}(J_p - \bar{J})r_{CJ}) / \sum_J |r_{CJ}| \]

**Recommendation**
- Focus on products rated by the neighbors
- Provides the list of N products to the customer
Existing approaches - Limitations

- **Sparsity**
  - Rely on exact matches
  - Cause: Loss of system coverage and accuracy
  - Who have rated at least two items/products in common
  - Ex – Paul & Sue, Sue & Mike – Mike & Paul

- **Scalability**
  - Increasing customers and products

- **Synonymy**
  - Customer 1: Recycled letter pad – Rated High
  - Customer 2: Recycled memo pad – Rated High
  - Unable to compute correlation
  - Unable to discover latent association
  - Recycled office items
Evolved Recommendation Systems

- Item Similarity
- Bipartite Projection
- Spanning Tree

- They can be used to predict the rating for a product that a customer has never reviewed, based on the data of all other users and their ratings in the system.
- The above 3 systems will be applied on new users, old users and also on all users.
This recommendation system we discuss is inspired by Amazons item-based collaborative filtering.

In Amazons algorithm, they represent each item with a vector showing who bought/reviewed the item.

Similarity between these two products is defined by the cosine of the two vectors.

After calculating similarity between all product pairs, we will have an item-item matrix showing the similarity between the items.

Finally, the similarities can provide a good reference on some of the other products that a customer would buy.
Iterative algorithm is calculated in various ways, but a common method is to use the cosine measure we described earlier. This offline computation of the similar-items table is extremely time intensive, with $O(N^2M)$ as worst case.

Comparison between different collaborative methods:
- Traditional collaborative filtering is impractical on large data sets.
- Cluster models can perform much of the computation offline, but recommendation quality is relatively poor.
- Search-based models build keyword, category, and author indexes offline, but fail to provide recommendations with interesting, targeted titles. They also scale poorly for customers with numerous purchases and ratings.
The Amazon’s purchase network can be modeled as a bipartite graph. The nodes can be divided into two sets, one set for the customers and the other for the products. Edges exist only between two nodes in different sets. Each edge carries a rating, representing the rating of a product given by a customer.
We also discuss an algorithm that makes use of the graph structure. The algorithm explores a new way of defining similarities between products and customers.

In this algorithm, we still model the product-customer relationship as a bipartite graph. One set of the node is the product nodes and the other set is the customer nodes. An edge connecting a product to a customer indicates a review of that product. The edge also carries a rating associated with that review.

Figure 3: Sample Spanning Tree
Singular Value Decomposition

- Matrix factorization technique
  \[ R = U.S.V' \]
  
  \( U \) and \( V \) – two orthogonal matrices of size \( m \times r \) and \( n \times r \)
  
  \( S \) - Diagonal matrix size of \( r \times r \)

- SVD to capture latent relationships between customers and products that allow us to compute the predicted likeliness of a certain product by a customer.

- Second, SVD to produce a low-dimensional representation of the original customer-product space and then compute neighborhood in the reduced space. We then used that to generate a list of top-\( N \) product recommendations for customers.
Evaluation of Experiments

- **Prediction**
  - Coverage metrics
  - Statistical accuracy
    - Mean Absolute Error (MSE)
    - Root Mean Squared Error (RMSE)
    - Correlation between ratings and products
  - Decision support accuracy
    - Reversal rate
    - Weighted errors
    - ROC sensitivity

- **Recommendation**
  - Used Information retrieval – recall and precision
  
  \[ \text{Recall} = \frac{\text{Size of hit set}}{\text{Size of test set}} = \frac{|\text{test}\cap\text{top N}|}{|\text{test}|} \]
  
  \[ \text{Precision} = \frac{\text{Size of hit set}}{\text{Size of top N set}} = \frac{|\text{test}\cap\text{top N}|}{N} \]
  
  \[ \text{N, Recall, Precision} \rightarrow \text{Critical for quality judgement} \]
  
  \[ \text{F1 metric} = \frac{2 \times \text{Recall} \times \text{Precision}}{(\text{Recall} + \text{Precision})} \]
Experiment datasets

- **Dataset: Movie Lens, E-commerce**
  - **Movie lens**: Debuted 1997
  - 35,000 users, 3,000+ movies, 100,000 rating records
  - `<customer, product, rating>`

- **E-Commerce:**
  - Purchase information – 6,502 users, 23,554 catalog items
  - `<customer, product, purchase, amount>`
Outcome – Movie data set

Top-10 recommendation (Movie data set)

- ML Low-dim
- ML High-dim

F1 Metric

Dimension, k

High dimensional value at x = 0.8
Outcome – Commerce dataset

Top-10 recommendation
(Commerce data set)

- EC Low-dim
- EC High-dim

High dimensional value at $x = 0.6$
Results discussion

- The movie experiment reveals that the low dimensional results are better than the high dimensional counterpart at all values of $k$.

- E-commerce experiment the high dimensional result is always better, but as more and more dimensions are added low dimensional values improve.

- Study shows that Singular Value Decomposition (SVD) may be such a technology in some cases.

- Researchers tried several different approaches to using SVD for generating recommendations and predictions, and discovered one that can dramatically reduce the dimension of the ratings matrix from a collaborative filtering system.
We found that, in terms of effectiveness measured with mean squared error (MSE), for all users, Item Similarity has the best result, then followed by Spinning Tree, and Bipartite Projection is the worst.

For new users, Spinning Tree has the best result, then followed by Item Similarity, and Bipartite Projection cannot even generate result because of lack of data.

For old users, Bipartite Projection has the best result, then followed by Item Similarity, and Spinning Tree is the worst.

In terms of computational performance, Bipartite Projection is the fastest algorithm that gives result within fraction of seconds, while Item Similarity can be very computationally expensive.

Existing Amazon recommendation system uses a mixture of all these complex algorithms to get the results very less time since we have around 480 million products in USA and also around 110 million active customers.
Thank You