Recommendation System

Sayali Thorat and Gagan Solur Venkatesh

Abstract—“If I have 3 million customers on the Web, I should have 3 million stores on the Web.” –Jeff Bezos, CEO of Amazon.com™. To achieve this goal most of the E-commerce sites use recommendation algorithms to suggest products to online customers and to provide consumers with information to help them decide which products to purchase. The Collaborative Filtering is the most successful algorithm in the recommender systems' field. Till now different approaches and techniques have been proposed. In this project we developed recommendation system based on Collaborative filtering and tried to improve performance of Collaborative filtering by combining it with K-means algorithm to find similar products. We then compare the results based on precision-recall curve.

I. Introduction

Recommendation algorithms are used by E-commerce sites to suggest products to online customers and to provide consumers with information to help them decide which products to purchase. The products can be recommended based on the top overall sellers on a site, on the demographics of the consumer, or on an analysis of the past buying behavior of the consumer as a prediction for future buying behavior [1]. This helps to increase sales rate and overall profit margin.

Commonly used recommendation techniques are:

1. Content-Based Recommendations- based on profile attributes
2. Collaborative Filtering - based on historical interactions

Motivation for this project lies in the fact that in today's world buyers face increasing range of choices while sellers have to tailor their advertising effort to these choices [2]. In addition firms have to collect large volumes of transaction data which provide deep insight into customer to product interaction. It is therefore a challenge to develop recommendation systems to produce high quality recommendations for millions of users and items despite data sparsity.

II. Related work

In [4], Greg, Brent, and Jeremy discuss Item-to-Item collaborative filtering in their paper "Amazon.com recommendations". Here the authors focus on matching each users, purchased and rated items to other items and combine them to form a recommendation list. In [5] Badrul, George, Joseph and John compared different techniques for computing item to item similarities like cosine-based similarity and correlation-based similarity. In [6]; Marko and Yoav discussed benefits of hybrid recommendation system called “Fab” which incorporates advantages of both collaborative and content based recommendation systems. Authors have implemented hybrid content based collaborative system where they maintain user profile based on analyzed content and compare these profiles to find similar users which can then be used for collaborative recommendation. In [7]; Guy and Asela have discussed performance metrics for recommender systems in their paper "Evaluating Recommender Systems". In [8], Gilda and Mehregan have implemented collaborative filtering algorithm using k-means clustering.

III. Approach

High level implementation approach is explained in the Figure [1]. In this project we have developed recommendation mode based on targeted customer and community inputs. Inputs about the targeted customer are fed into the recommendation process to provide personalized recommendations. We can use targeted customer's details in form of rating of items brought, customer's purchase history, item's being viewed and search to provide more personalized experience. In this project are using purchase history to personalize recommendation.

Community input includes a broad range of data regarding how multiple individuals in a community perceive items. This input may be in the form of rating, product reviews, purchase history, popular items. This information helps to discover customer similarity and draw conclusion about recent trends or item similarity. We are using rating and popular item list for collaborative filtering and review test and populate items for content based filtering as community input.

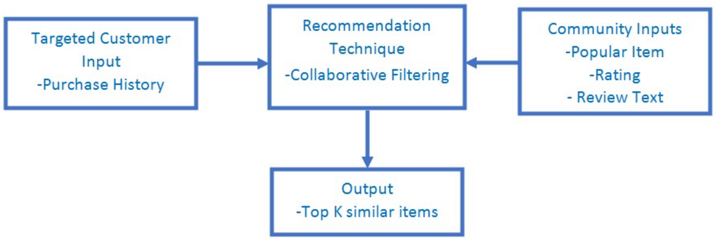
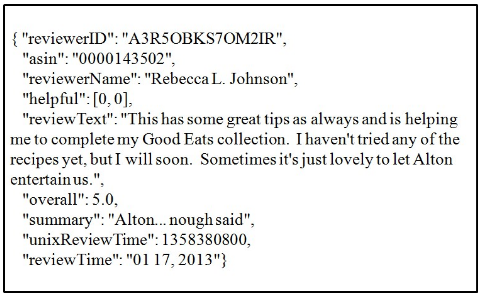


Figure 1: High level approach

We are implementing collaborative filtering algorithm and process input data based on purchase history of the customer. The output will be the most similar item in the form of suggestion.

A. Dataset

We are using Amazon's Movies and TVdata. Figure [2] gives brief idea about the dataset. Review data includes reviewer ID, product ID (asin), reviewer name, review text, rating (overall), summary, review time etc. Out of which we are using reviewer ID, product ID, rating and review text to build recommendation system.



```
{ "reviewerID": "A3R5OBKS7OM2IR",
  "asin": "0000143502",
  "reviewerName": "Rebecca L. Johnson",
  "helpful": [0, 0],
  "reviewText": "This has some great tips as always and is helping
me to complete my Good Eats collection.  I haven't tried any of the
recipes yet, but I will soon.  Sometimes it's just lovely to let Alton
entertain us.",
  "overall": 5.0,
  "summary": "Alton... nough said",
  "unixReviewTime": 1358380800,
  "reviewTime": "01 17, 2013"}
```

Figure 2: Sample review data

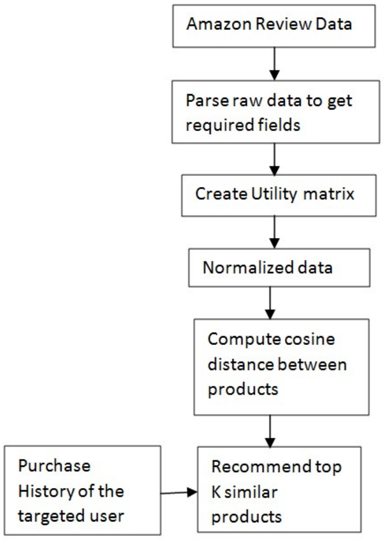

Amazon Review Data
Parse raw data to get required fields
Create Utility matrix
Normalized data
Compute cosine distance between products
Purchase History of the targeted user
Recommend top K similar products

Figure 3: Flowchart of collaborative filtering

IV. Collaborative Filtering

Collaborative filtering system recommends items based on the interests of a community of users without any analysis of item content. The items recommended to a user are those preferred by similar user.
Collaborative system commonly use user's rating for individual product. In this process use builds a personal profile by rating different product they are using.For any E-commerce application we are assuming that:

1. Number of users are more than the number of product.
2. Frequency of joining new user is more than the frequency of adding new product.

From these assumptions we can consider that the user to user matrix is more sparse than item to item matrix. Hence it will be more precise if we compare product to other product and find similarity between them. Hence in this project we have implemented Item to item collaborative filtering where we are comparing item with other items and recommend product to customer based on item similarity.

For collaborative filtering we are using asin(product ID), reviewerID and overall (rating) from review data. There many other fields like reviewerName, helful, reviewText, summary etc. that are not required to build collaborative filtering. So we tailored raw data set to get required fields. From this data we are considering products that customer purchase frequently. These details will be used to create utility matrix. Utility matrix is a customer to product matrix. Each entry in this matrix represents the degree of preference of that customer for that product. This is a sparse matrix as most of the customer doesn't rate most of the products. Once we have utility matrix we normalize each entry by subtracting average rating of that item from each rating. This will turn low rating into negative number and high rating into positive numbers. After filling empty values by 0, we are comparing each product with other by cosine distance. Based on the cosine distance we have created Item-to-Item similarity matrix which contains most similar 10 products for each popular product. Cosine distance can be calculated by formula:

$$\text{Similarity}(\vec{I_1}, \vec{I_2}) = \cos(\vec{I_1}, \vec{I_2}) = \frac{\vec{I_1}.\vec{I_2}}{\|\vec{I_1}\| * \|\vec{I_2}\|}$$

Once we have similarity matrix, we can recommend products to customer based on purchase history. If we have information about its purchase history then we can recommend similar items from item to item similarity matrix. Flowchart of collaborative filtering is shown in the Figure [3].

V. Results & Conclusion

In this project, we are interested to recommend the set of items to the user based on the top similar product which we get from our systems. In this case, to evaluate the systems, we are using precision recall curve to see the accuracy of the prediction of the systems [7].

We first start the evaluation with the testing data set from the data pre-processing step. The testing dataset consists about 20% of the total dataset. Then, we eliminated the users that only appear once in the dataset since we cannot calculate the precision-recall if the user only has 1 item in the purchase history. After we eliminated all users that only appear once, we hide 40% of the dataset and recommend set of items to each user based on our recommendation systems. We then analyze outcomes for the recommended and hide items.

	Predicted Positive	Predicted Negative
Actual Positive	True-Positive(TP)	False-Negative(FN)
Actual Negative	False-Positive(FP)	True-Negative(TN)

Table 1 [9]

Where,

True-Positive is the item that user bought and our systems recommended.
False-Positive is the item that user did not buy but our systems recommended.
True-Negative is the item that user did not buy and our systems did not recommend.

False-Negative is the item that user bought but we did not recommend.

Precision and recall can be calculated by using below formula:

Precision = $\frac{\#TP}{\#TP \quad \#FP}$ and

Recall = $\frac{\#TP}{\#TP+\#FN}$

We calculate the precision and recall based on the number of items that fall into each cell. In our project we are recommending maximum of 10 products to the user.

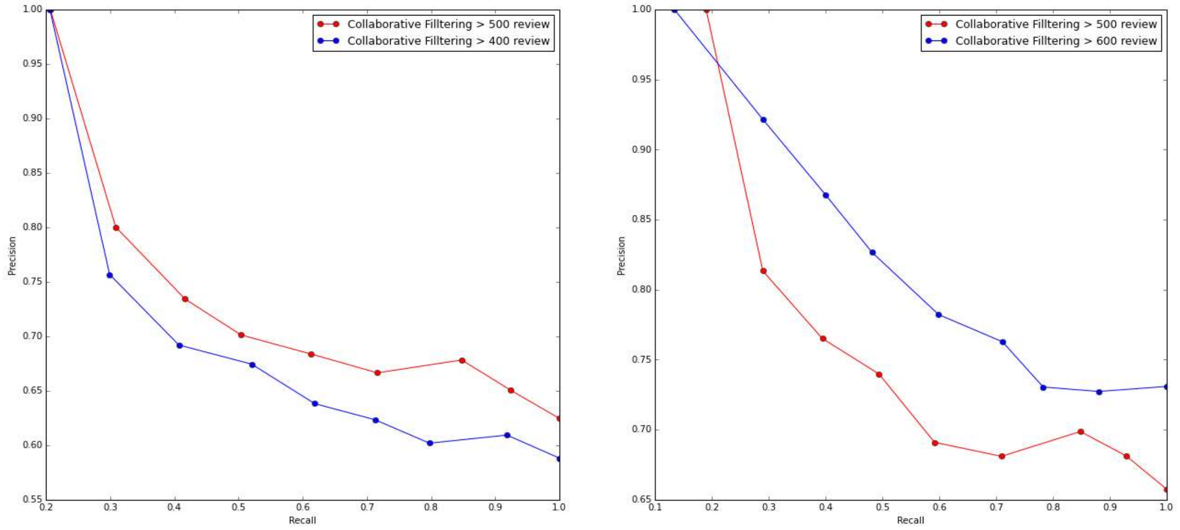


Figure 4: Result of collaborative filtering

From the figure (4), if we increase the numbers of recommended item the precision will decrease while the recall will increase.   We then use difference dataset by varying the number of reviews from more than 400 reviews to more than 600 reviews. We assume that if the items have more reviews, then it is more popular and if the item has lower review, it is less popular. If we consider more popular items into our system, the performance will be better.

Advantage of collaborative filtering is it works for any kind of data as we are considering only user ratings to build this method. We are computing similarity matrix offline hence it gives better performance. As we are considering only popular items, collaborative filtering face cold start problem. Also we cannot recommend product which is not rated previously. For precise recommendation we require enough user ratings to find match. As we have created similarity matrix for popular items, it tends to recommend popular items. Hence recommendation will not be accurate if user has unique taste.

VI. Future work

We are trying to improve performance of collaborative filtering by combining it with the k-means algorithm where we are finding similar users using k-means algorithm and recommending items from similarity matrix created by considering review history of the similar users.

The next step will be the implementation of Amezon's Item-to-Item recommendation algorithm mention in IEEE paper[4], which is more complicated algorithm as they are considering user's purchase history and review history to build item-to-item similarity matrix.

VII. References

[1] J.B. Schafer, J.A. Konstan, J. Riedl, "E-Commerce Recommendation Applications", Data Mining and Knowledge Discovery, pp. 115-153, 2001.

[2] Melville, P., Sindhwani, “Recommender system”, V. Encyclopedia of machine learning, pp. 1–9. (2010)

[3] Jeffrey D. Ullman Anand Rajaraman, Jure Leskovec. Mining of massive datasets, 2013. URL: http://infolab.stanford.edu/~ullman/mmds.html#latest.

[4] Greg Linden , Brent Smith , Jeremy York, Amazon.com Recommendations: Item-to-Item Collaborative Filtering, IEEE Internet Computing, v.7 n.1, p.76-80, January 2003 [doi>10.1109/MIC.2003.1167344]

[5] Badrul Sarwar , George Karypis , Joseph Konstan , John Riedl, Item-based collaborative filtering recommendation algorithms, Proceedings of the 10th international conference on World Wide Web, p.285-295, May 01-05, 2001, Hong Kong, Hong Kong [doi>10.1145/371920.372071]

[6] Marko Balabanović , Yoav Shoham, Fab: content-based, collaborative recommendation, Communications of the ACM, v.40 n.3, p.66-72, March 1997 [doi>10.1145/245108.245124]

[7] Gunawardana, A, & Shani, “Evaluating Recommender Systems”, Recommender Systems Handbook (9781489976369), G 2015, p. 26

[8] G. M. Dakhel, M. Mahdavi, "A new collaborative filtering algorithm using k-means clustering and neighbors' voting", Hybrid Intelligent Systems (HIS) 2011 11th International Conference on, pp. 179-184, 2011.

[9] Jesse Davis , Mark Goadrich, The relationship between Precision-Recall and ROC curves, Proceedings of the 23rd international conference on Machine learning, p.233-240, June 25-29, 2006, Pittsburgh, Pennsylvania