Building Personalized and Non Personalized Recommendation Systems

SNEHA KHATWANI
Student M.Tech, CSE Department,
Shri Ramdeobaba College of Engineering and Management, Nagpur, India
sneha_khatwani@rediffmail.com

DR. M.B. CHANDAK
HOD, CSE Department,
Shri Ramdeobaba College of Engineering and Management, Nagpur, India.
chandakmb@gmail.com

Abstract - The contents of e-Commerce such as music, movies, books and electronics goods are necessary for a modern life style. But, it becomes difficult to find content according to users likes and users preference. An approach which produces desirable results to solve such the problem is to develop "Recommender System." The Recommender System of an e-Commerce site selects and suggests the contents to meet user's preference automatically using data sets of previous users stored in database. There can be two types of recommendations viz. Personalized and Non- Personalized recommendations. Personalized recommendation takes into consideration users' previous history for rating and predicting items. On the other hand non-personalized recommendation systems recommend what is popular and relevant to all the users which can be a list of top-10 items for every new user. One of the most important techniques in the Recommender System is information filtering. The filtering techniques can be mainly classified into two categories viz. Collaborative Filtering and Content Based Filtering. Recommender system is a type of web intelligence technique that can make daily information filtering for users. This paper covers different techniques which can be used for creating personalized and non-personalized recommendations. This paper also explores the different packages of R i.e. Shiny which is used to create web applications and rmarkdown which is used to create dynamic documents.

Key Words: Collaborative Filtering, Content Based Filtering, Clustering.

1. Introduction

Recommendation engines are common among e-commerce (Amazon, Flipkart), social media (for e.g. LinkedIn, Facebook and Twitter) and content-based websites (Reddit). Amazon was one of the first sites to use a recommendation system. These engines use a variety of technologies and techniques that is used for filtering large amounts of data and provide a smaller, focused system of suggestions for the user. Netflix, for example, uses tagging of metadata on videos in conjunction with data about user behavior to come up with recommended movies and TV shows for specific users. LinkedIn uses the semi-structured data provided by members, including things like locations, job titles, skill sets and industries, to find recommendations for their "Jobs you might be interested in" section. Recommender systems are personalized or non-personalized information sources that provide recommendations: prediction or suggestions for items which could be of use to a user.

One prominent filtering techniques is Collaborative Filtering (CF). The word "collaborative" refers to the fact that users collaborate with each other in order to recommend items. It takes into account of user purchases and preferences. It is the process of finding information using the opinion of other users or using knowledge of other items. Fig 1 shows a simple collaborative filtering example. Predictions about user interests are made by collecting information from other users who have made similar kind of choices. It is a basic assumption while performing collaborative filtering that those individuals who have agreed in the past tend to agree again in the future. Another technique is known as content based filtering. This filtering can be accomplished by using description of items, and the history of different users at the same time is not required. Fig 2 shows a simple content based filtering example. For each user, the algorithms recommend items that are similar to its past purchases and similar to the content the user has liked in the past. Here are the steps to perform a recommendation:

1. Define descriptions of items. For example, in case of movies the description can be genre, actor, actress, director etc.
2. Define user profiles based on purchases.
3. Recommend to each user the items matching its profile

User profiles are based on their purchases, so the algorithms recommend items similar to past purchases. In many situations, we are able to build different collaborative and content-based filtering models. In machine learning, the approach of combining different models usually leads to better results. A simple example is collaborative filtering

combined with information about users and/or items. In the case of IBCF, the distance between items can take account of user preferences and item descriptions at the same time. Even in UBCF, the distance between users can take account of their preferences and personal data. In the case of recommendation, these models are called hybrids. There are different ways to combine filtering models.

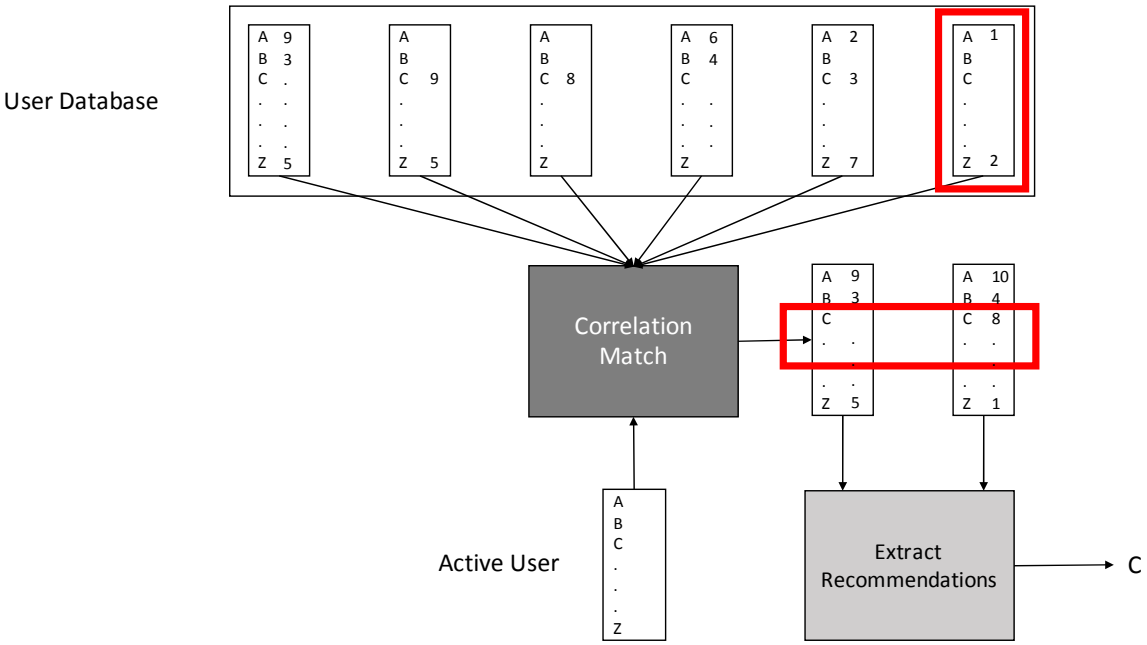

Fig 1. Collaborative Recommendations

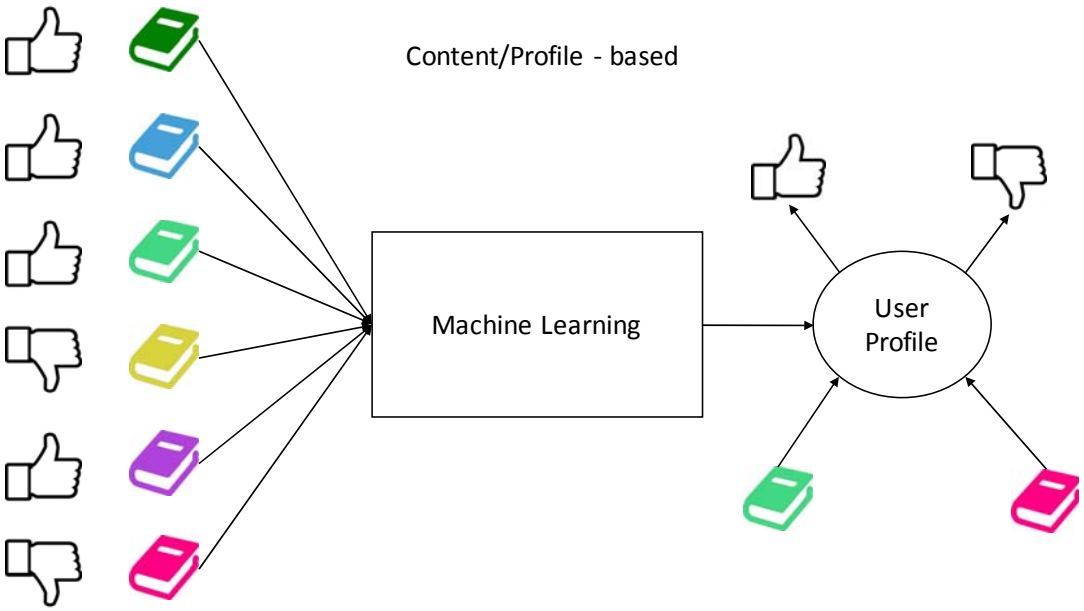

Fig 2. Content Based Recommendations

We also use a package of R known as R-Markdown. This package enables to create dynamic documents with chunks of R code embedded. The document is self contained and fully reproducible which makes it very easy to share. R-Markdown allows users to embed R code into a markdown document. The document is then 'knit' using knitr to create a HTML file(or pdf, word etc). In the application it becomes difficult to show all the graphs obtained during data cleaning and preprocessing. An easy yet efficient way to show the summary or all plots obtained during the execution is to knit the respective chunks of R code.

2. Recommender Techniques

(a) Personalized Recommendations:

Recommendation model is built using k nearest algorithm by implementing user based collaborative filtering. This is done in two phases. First is the neighborhood formation phase. In this phase by using centroid method as discussed above we can formulate the neighborhood of items for a current user by taking into consideration previous user

records. This is done by calculating distance of active user with respect to every user. Here, we use the Euclidean distance. Other distance functions can also be used as shown in Fig 3 below.

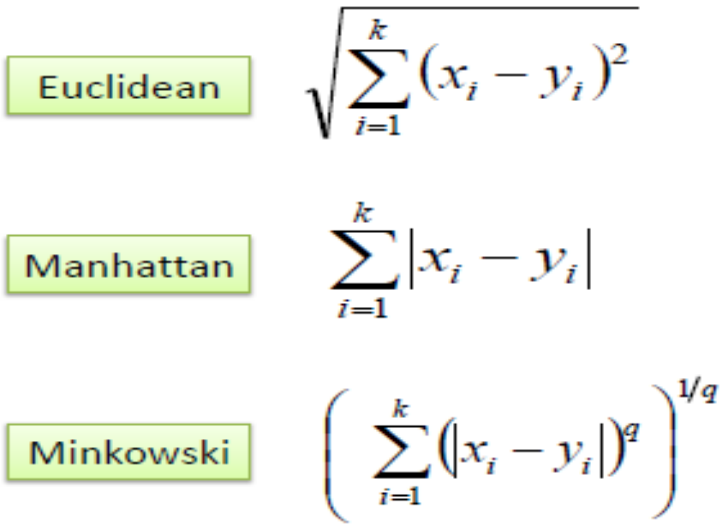

Fig 3: Distance Functions

After the formation of nearest items for a current user the items can be recommended to the user. This is done in the second phase that is the recommendation phase by applying a combination function which would filter results and predict top k items for the user.

This can be done by weighting all users with respect to the similarity of active user. There are many ways of measuring similarities:

1. Cosine Similarity: Here the similarity is calculated based on the angle between the vectors. It is based on vector space approach rather than a statistical approach.

$$\cos(\vec{A}, \vec{B}) = \frac{\vec{A} \bullet \vec{B}}{\|\vec{A}\| * \|\vec{B}\|}$$

2. Pearson Correlation Coefficient: A popular similarity measure in User Based CF; Pearson correlation is a measure of the linear correlation between two variables which produces a value between +1 and −1 inclusive, where 1 is total positive correlation, 0 is no correlation, and −1 is total negative correlation.

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

where ra,p: users
r: ratings of user a for item p
p: set of items rated by both a and b

After applying similarity coefficient a subset of sample users or neighbors can be selected based on similarities calculated in previous steps and those sample users could be used as predictors. Then the ratings can be normalized to compute prediction using weighted combination of ratings from selected neighbors.

This algorithm is deployed in a shiny app where user is asked to enter three movies of his choice and a personalized list of ten recommendations is suggested according to users likes.

(b) Non-Personalized Recommendations:

For recommending top N movies which are of relevance to all users we use a package called recommenderlab. Before performing collaborative filtering we explore and prepare the dataset which involves normalizing the data and binarizing it. We use the MovieLens dataset, to find out the dimensions of the matrix and exploring the nature and occurrences of ratings.

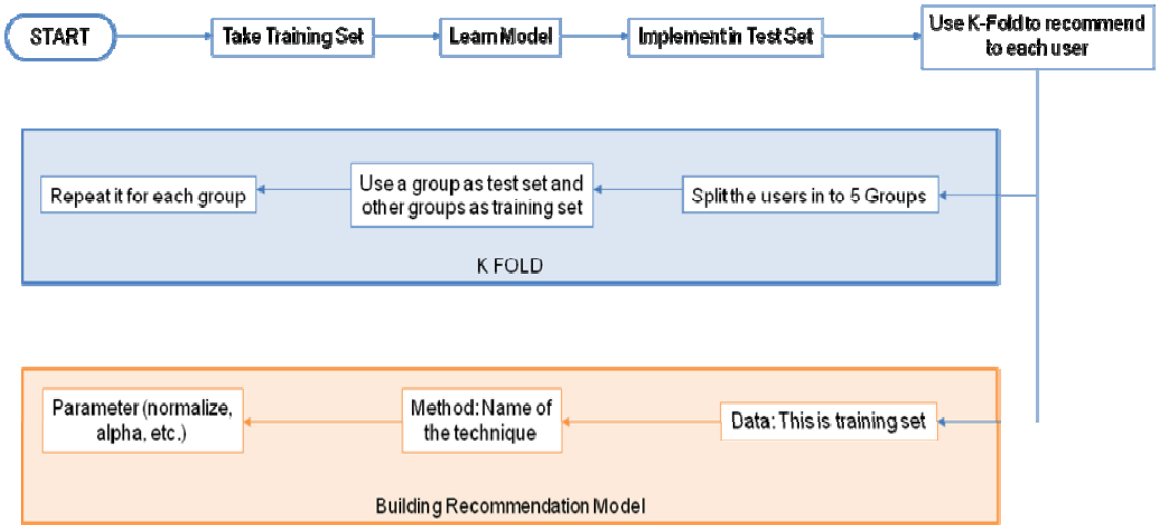

Fig 4: Logic Flow for building a recommender model for IBCF

Data preparation

Data preparation involves selection of relevant data and normalization of data. After data exploration few observations can be made:

Ratings might be biased in case of movies that have been viewed only a few times because of lack of data

Ratings might be biased in case of users who rated only a few movies. The threshold for minimum number of users per movie and vice versa should be determined. We define a threshold of

- Users who have rated at least 50 movies
- Movies that have been watched at least 100 times

A complete layout for building a recommender model is shown in Fig 4.

A comparative study can also be shown for recommendations with different thresholds. For example we also compute recommendations for users who have rated 15, 30 and 50 movies.

There are few users who give either high (or low) ratings to all their movies. This can again might bias the results. This effect can be removed by normalizing the data in such a way that the average rating of each user is 0 i.e. the mean rating of each user is 0. Now we perform item based collaborative filtering to the reduced dataset. The algorithm considers the user's likes or previous interests and recommends similar items in case of a new active user. The core algorithm is based on these steps:

1. For each two items, measure how similar they are in terms of having received similar ratings by similar users
2. For each item, identify the k-most similar items
3. For each user, identify the items that are most similar to the user's purchases

In this technique, a model using a part of the MovieLense dataset can be built (the training set) and can be applied on the other part (the test set). We use k-fold algorithm to recommend items to each user. This can be done by

- Splitting the users randomly into five groups
- Using a group as a test set and the other groups as training sets
- Repeat it for each group

We are able to recommend movies to the users in the test set. We will define recommended that specifies the number of items to recommend to each user. This section will show you the most popular approach to computing a weighted sum:

For each user, the algorithm extracts its rated movies. For each movie, it identifies all its similar items, starting from the similarity matrix. Then, the algorithm ranks each similar item in this way:

- Extract the user rating of each purchase associated with this item. The rating is used as a weight.
- Extract the similarity of the item with each purchase associated with this item.
- Multiply each weight with the related similarity.
- Sum everything up.

Then, the algorithm identifies the top n recommendations:

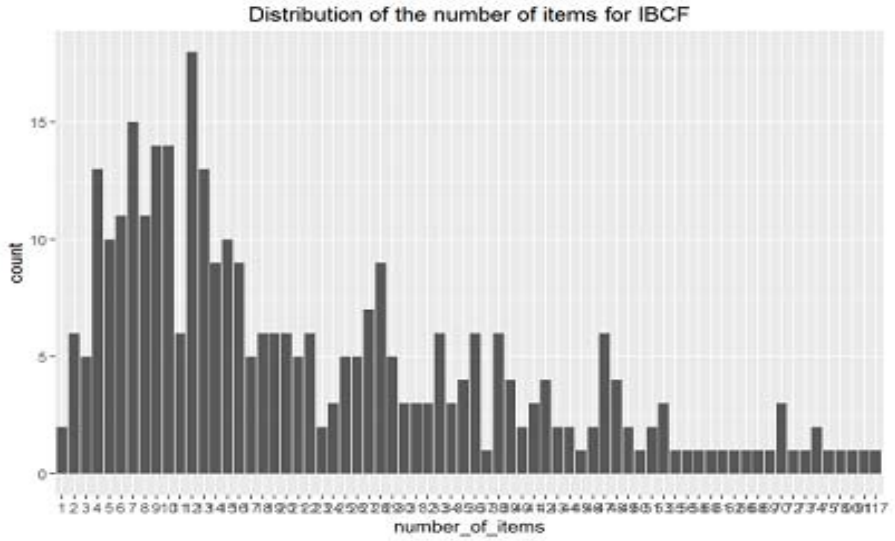

Fig 5: Distibution of number of items for users who have rated minimum 15 movies

We can identify the most recommended movies. For this purpose, we will define a vector with all the recommendations, and we will build a frequency plot Fig 5 shows the distribution of the number of items for IBCF. This is the distribution plot when the users have rated minimum 15 movies. Fig 6 shows distribution of the number of items when the users have rated minimum 30 movies and Fig 7 shows distribution of the number of items when the users have rated minimum 50 movies. By changing this threshold we try to compare and find the threshold which gives the best accuracy and precision of recommendations.

As we can see from the graphs shown, if we consider the plot for threshold =15 movies. It can be seen that count towards the end becomes constant. Similarly we see that in threshold = 50 movies the count becomes constant towards the end. With reference to distribution plot we can conclude that movies = 30 is a better qualifier than the other two.

If we calculate the accuracy of recommendations for the three cases then it is concluded that users who have rated minimum 30 movies is the best qualifier. Table-1 shows the top-10 list of recommendations for first user in case of three different scenarios.

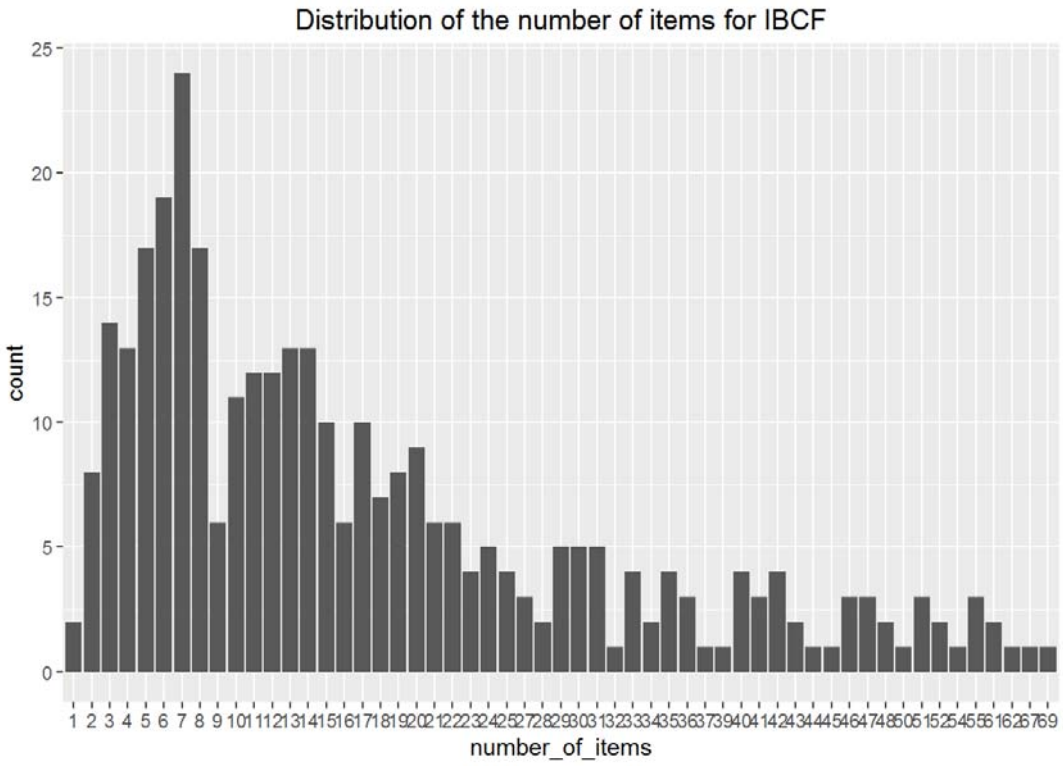

Fig 6: Distibution of number of items for users who have rated minimum 30 movies

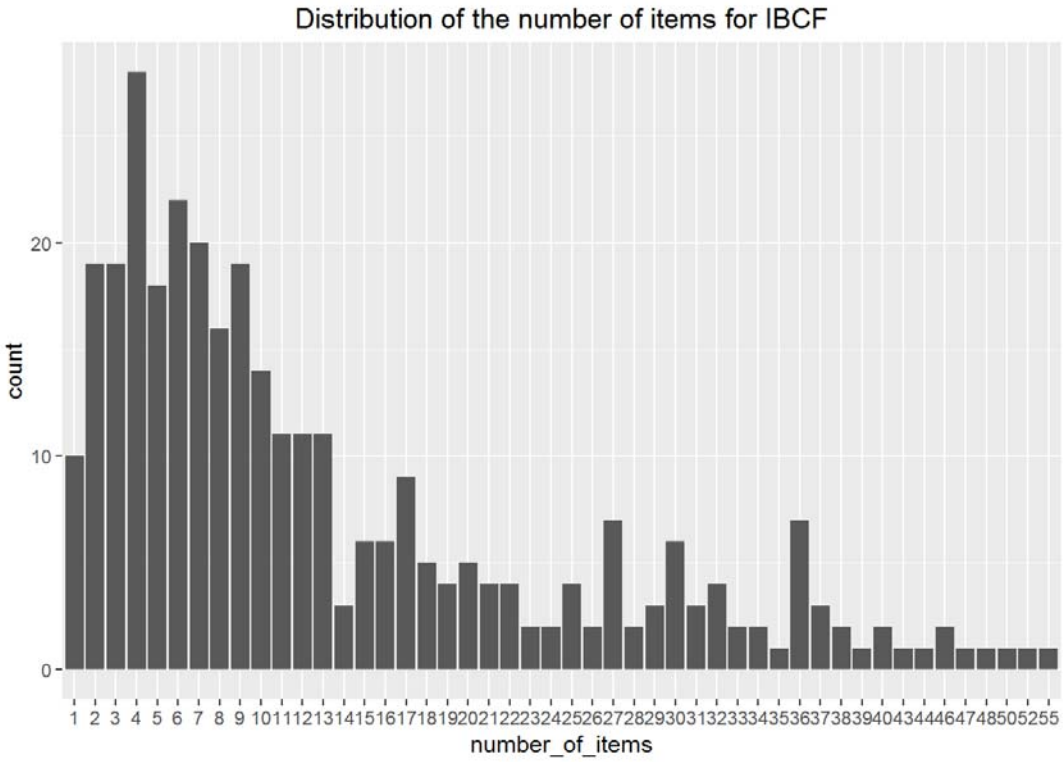

Fig 7: Distibution of number of items for users who have rated minimum 30 movies

Table 1: Recommendations for the first user when he has rated a. 15 movies b. 30 movies c. 50 movies

Recommendations(15 movies)	**Recommendations(30 movies)**	**Recommendations(50 movies)**
"Close Shave, A (1995)"	"Babe (1995)"	"In the Name of the Father (1993)"
"Casablanca (1942)"	"Dead Man Walking (1995)"	"Casablanca (1942)"
"Vertigo (1958)"	"Usual Suspects, The (1995)"	"L.A. Confidential (1997)"
"Butch Cassidy and the Sundance Kid (1969)"	"Braveheart (1995)"	"North by Northwest (1959)"
"Manchurian Candidate, The (1962)"	"Taxi Driver (1976)"	"Deer Hunter, The (1978)"
"One Flew Over the Cuckoo's Nest (1975)"	"Hoop Dreams (1994)"	"Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1963)"
"Secrets & Lies (1996)"	"Shawshank Redemption, The (1994)"	"Rear Window (1954)"
"Rear Window (1954)"	"Fugitive, The (1993)"	"Great Escape, The (1963)"
"L.A. Confidential (1997)"	"Much Ado About Nothing (1993)"	"Leaving Las Vegas (1995)"
"Amistad (1997)"	"Silence of the Lambs, The (1991)"	"Vertigo (1958)"

3. Experimental Study:

The dataset used in this experiment is obtained from

Movie Lens project. This data set consists of:

* 100,000 ratings (1-5) from 943 users on 1682 movies.

* Each user has rated at least 20 movies.

* Simple demographic info for the users (age, gender, occupation, zip)

All ratings are between 1(bad) and 5 (Excellent). In our experiment, we selected 25% of the data as training set and compute the recommendation for the remaining 75% of the movies. The experiments were conducted on two different recommendation algorithms: [1] Item based collaborative filtering using package of R language, i.e. RecommenderLab which gives non personalized results

[2] User based collaborative filtering using K-Means Clustering Algorithm for active users which gives personalized results.

4. Evaluating Recommender Techniques

A function is used to measure the accuracy of Recommender techniques and it computes the following aspects:

- Mean absolute error (MAE): This is the mean of the absolute difference between the real and predicted ratings. . It computes the deviation between predicted ratings and actual ratings

$$MAE = \frac{1}{n}\sum_{i=1}^{n}| p_i - r_i |$$

- Mean squared error (MSE): This is the mean of the squared difference between the real and predicted ratings. It's the square of RMSE, so it contains the same information

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(p_i - r_i)^2$$

- Root mean square error (RMSE): This is the standard deviation of the difference between the real and predicted ratings. It places more emphasis on larger deviation

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - r_i)^2}$$

The calculated values of RMSE, MSE, MAE for prediction accuracy of Non Personalized recommendation systems which use Item Based Collaborative Filtering(IBCF) technique and for Personalized Recommendation systems which use User Based Collaborative filtering are as follows:

Table 2: Evaluation of RMSE, MSE, MAE for IBCF and UBCF

	IBCF	UBCF
RMSE	1.196995	1.015468
MSE	1.432796	1.031176
MAE	0.817596	0.800383

Some other methods to evaluate recommender models are:

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

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

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

$$Recall = \frac{tp}{tp + fn} = \frac{|good\ movies\ recommended|}{|all\ good\ movies|}$$

The classification of movies is done based on the following parameters. This table has been established by human domain experts

Table 2: Classification of movies

		Reality	
		Actually Good	Actually Bad
Prediction	Rated Good	True Positive (tp)	False Positive (fp)
	Rated Bad	False Negative (f)	True Negative (tn)

'Precision' is the proportion of top results that are relevant. The 'Recall' would measure the proportion of all relevant results included in the top results. In a formal way, we could consider movies as instances and the task it to return a set of relevant movies given a user. So the task would be to assign each movie to one of two categories: relevant and not relevant. In recommender systems those metrics could be adapted so: "The precision is the proportion of recommendations that are good recommendations, and recall is the proportion of good recommendations that appear in top recommendations." From the graph in Fig 7 we see that as precision increases the recall decreases.

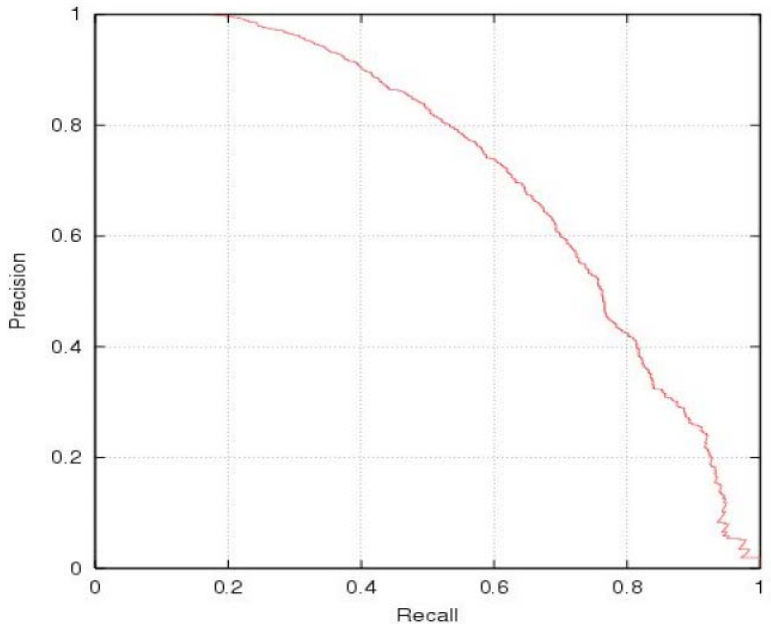

Fig 7: Precision vs Recall

5. Conclusion

Recommendation Systems have made filtering easy for user and at the same time it is profitable in market as well. Up to 35% of products sales of Amazon are due to recommendations. In this paper we have discussed how to make recommender system models. User based and item based collaborative filtering models using different techniques have been discussed. Primary focus has been given to personalized and non personalized recommendations. We should evaluate recommendation algorithms so as to select the best algorithm from a set of candidates. These Through the Experiments, we have compared three algorithms which vary from user to user.

In the future, we are interested in studying the hybrid recommendation model which takes into consideration building collaborative filters using big data.

6. References

[1] Gediminas Adomavicius, Alexander Tuzhilin , "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible ExtensionsIEEE transactions on knowledge and data engineering, vol. 17, no. 6, june 2005

[2] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl, Item-based Collaborative Filtering Recommendation Algorithms", WWW10, May 1-5, 2001, Hong Kong.

[3] Bharat Bhaskar, K. Srikumar Recommender systems in E-commerce, Methodologies and applications of data mining.

[4] Greg Linden, Brent Smith, and Jeremy York "Amazon.com Recommendations Item-to-Item Collaborative Filtering" Published by the IEEE Computer Society

[5] Dr. Sarika Jain, Anjali Grover, Praveen Singh Thakur, Sourabh Kumar Choudhary, "Trends, Problems And Solutions of Recommender System" International Conference on Computing, Communication and Automation (ICCCA2015) and expert system application.

[6] D. Jannach, M. Zanker, A.Felfernig and G. Friedrich, Recommender Systems – an introduction Cambridge University Press, 2010
[7] Dr. M.B. Chandak, Kushboo Khurana, "Study of Various Video Annotation Techniques", International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 1, January 2013
[8] K. Lang, Newsweeder: "Learning to filter netnews," in Proceedings of the Twelfth International Conference on Machine Learning, 1995.
[9] Sarwar, B., Karypis, G., Konstan, J., And Riedl, J. 2000. Analysis of recommendation algorithms for e-commerce. In Proceedings of ACM E-Commerce. ACM, New York.