ISSN 2321 3361 © 2020 IJESC

Research Article Volume 10 Issue No.6

A Movie Recommender System: MOVREC using Machine Learning Techniques

Ashrita Kashyap[1], Sunita. B[2], Sneh Srivastava[3], Aishwarya. PH[4], Anup Jung Shah[5]
Department of Computer Science & Engineering
SAIT, Bengaluru, Karnataka, India

Abstract:
This paper discuss about recommendations of the movies. A movie recommendation is important in our social life due to its strength in providing enhanced entertainment. Such a system can suggest a set of movies to users based on their interest, or the popularities of the movies. A recommendation system is used for the purpose of suggesting items to purchase or to see. They direct users towards those items which can meet their needs through cutting down large database of Information. A recommender system, or a recommendation system (sometimes replacing 'system' with a synonym such as platform or engine), is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item[1][2]. They are primarily used in commercial applications. MOVREC also help users to find the movies of their choices based on the movie experience of other users in efficient and effective manner without wasting much time in useless browsing.

Keywords: Filtering, Recommendation System, Recommender.

I. INTRODUCTION

Recommender systems usually make use of either or both collaborative filtering and content-based filtering (also known as the personality-based approach),[3] as well as other systems such as knowledge-based systems.

Collaborative filtering approaches build a model from a user's past behaviour (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users.

This model is then used to predict items (or ratings for items) that the user may have an interest in.[5] Content-based filtering approaches utilize a series of discrete, pre-tagged characteristics of an item in order to recommend additional items with similar properties.[4] Current recommender systems typically combine one or more approaches into a hybrid system.

The differences between collaborative and content-based filtering can be demonstrated by comparing two early music recommender systems – Last.fm and Pandora Radio.

• Last.fm creates a "station" of recommended songs by observing what bands and individual tracks the user has listened to on a regular basis and comparing those against the listening Behavior of other users. Last.fm will play tracks that do not appear in the user's library, but are often played by other users with similar interests. As this approach leverages the Behavior of users, it is an example of a collaborative filtering technique.

• Pandora uses the properties of a song or artist (a subset of the 400 attributes provided by the Music Genome Project) to seed a "station" that plays music with similar properties. User feedback is used to refine the station's results, deemphasizing certain attributes when a user "dislikes" a particular song and emphasizing other attributes when a user "likes" a song. This is an example of a content-based approach.

Each type of system has its strengths and weaknesses. In the above example, Last.fm requires a large amount of information about a user to make accurate recommendations.

This is an example of the cold start problem, and is common in collaborative filtering systems. [6][7] Whereas Pandora needs very little information to start, it is far more limited in scope (for example, it can only make recommendations that are similar to the original seed).

Recommender systems are a useful alternative to search algorithms since they help users discover items they might not have found otherwise. Of note, recommender systems are often implemented using search engines indexing non-traditional data.

Recommender systems were first mentioned in a technical report as a "digital bookshelf" in 1990 by Jussi Karlgren at Columbia University, [8] and implemented at scale and worked through in technical reports and publications from 1994 onwards by Jussi Karlgren, then at SICS,[9][10] and research groups led by Pattie Maes at MIT,[11] Will Hill at Bellcore,[12] and Paul Resnick, also at MIT[13][14] whose work with Group Lens was awarded the 2010 ACM Software Systems Award.

Montaner provided the first overview of recommender systems from an intelligent agent perspective.[15] Adomavicius provided a new, alternate overview of recommender systems.[16] Herlocker provides an additional overview of evaluation techniques for recommender systems,[17] and Beel et al. discussed the problems of offline evaluations.[18] Beel et al. have also provided literature surveys on available research paper recommender systems and existing challenges.[19][20][21]

Recommender systems have been the focus of several granted patents

Table.1. Companies benefit through recommendation system

Netflix	2/3rd of the movies watched are recommended
Google News	recommendations generate 38% more click-troughs
Amazon	35% sales from recommendations
Choice stream	28% of the people would buy more music if they found what they liked

II. DEFINITION & MOTIVATION

There has been a huge increase in audio visual data in the A recommender system, or a recommendation system (sometimes replacing 'system' with a synonym such as platform or engine), is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item. They are primarily used in commercial applications. Recommender systems are utilized in a variety of areas and are most commonly recognized as playlist generators for video and music services like Netflix, YouTube and Spotify, product recommenders for services such as Amazon, or content recommenders for social media platforms such as Facebook and Twitter.[3] These systems can operate using a single input, like music, or multiple inputs within and across platforms like news, books, and search queries. There are also popular recommender systems for specific topics like restaurants and online dating. Recommender systems have also been developed to explore research articles and experts,[4] collaborators,[5] and financial services.[6]

III. METHODS

In the field of machine learning, classification methods which use different strategies to organize and classify data. Classifiers could possibly require training data.

1. Collaborative filtering
2. Content-based filtering
3. Multi-criteria recommender systems
4. Risk-aware recommender systems
5. Mobile recommender systems
6. Hybrid recommender systems

1. Collaborative filtering

An example of collaborative filtering based on a ratings system One approach to the design of recommender systems that has wide use is collaborative filtering. Collaborative filtering is based on the assumption that people who agreed in the past will agree in the future, and that they will like similar kinds of items as they liked in the past. The system generates recommendations using only information about rating profiles for different users or items. By locating peer users/items with a rating history similar to the current user or item, they generate recommendations using this neighbourhood. Collaborative filtering methods are classified as memory-based and model-based. A well-known example of memory-based approaches is the user-based algorithm,[24] while that of model-based approaches is the Kernel-Mapping Recommender. A key advantage of the collaborative filtering approach is that it does not rely on machine analysable content and therefore it is capable of accurately recommending complex items such as movies without requiring an "understanding" of the item itself. Many algorithms have been used in measuring user similarity or item similarity in recommender systems. For example, the k-nearest neighbour (k-NN) approach and the Pearson Correlation as first implemented by Allen. When building a model from a user's Behavior, a distinction is often made between explicit and implicit forms of data collection.

Examples of explicit data collection include the following:

- Asking a user to rate an item on a sliding scale.
- Asking a user to search.
- Asking a user to rank a collection of items from favourite to least favourite.
- Presenting two items to a user and asking him/her to choose the better one of them.
- Asking a user to create a list of items that he/she likes (see Rocchio classification or other similar techniques).

Examples of implicit data collection include the following:

- Observing the items that a user views in an online store.
- Analysing item/user viewing times.[22]
- Keeping a record of the items that a user purchases online.
- Obtaining a list of items that a user has listened to or watched on his/her computer.
- Analysing the user's social network and discovering similar likes and dislikes.

Collaborative filtering approaches often suffer from three problems: cold start, scalability, and sparsity.

- Cold start: For a new user or item, there isn't enough data to make accurate recommendations.
- Scalability: In many of the environments in which these systems make recommendations, there are millions of users and products. Thus, a large amount of computation power is often necessary to calculate recommendations.
- Sparsity: The number of items sold on major e-commerce sites is extremely large. The most active users will only have rated a small subset of the overall database. Thus, even the most popular items have very few ratings.

One of the most famous examples of collaborative filtering is item-to-item collaborative filtering (people who buy x also buy y), an algorithm popularized by Amazon.com's recommender system. Many social networks originally used collaborative filtering to recommend new friends, groups, and other social connections by examining the network of connections between a user and their friends.[1] Collaborative filtering is still used as part of hybrid systems.

2. Content-based filtering

Another common approach when designing recommender systems is content-based filtering. Content-based filtering methods are based on a description of the item and a profile of the user's preferences. These methods are best suited to situations where there is known data on an item (name, location, description, etc.), but not on the user. Content-based recommenders treat recommendation as a user-specific classification problem and learn a classifier for the user's likes and dislikes based on an item's features. In this system,

keywords are used to describe the items and a user profile is built to indicate the type of item this user likes. In other words, these algorithms try to recommend items that are similar to those that a user liked in the past, or is examining in the present. It does not rely on a user sign-in mechanism to generate this often temporary profile. In particular, various candidate items are compared with items previously rated by the user and the best-matching items are recommended. This approach has its roots in information retrieval and information filtering research.

To create a user profile, the system mostly focuses on two types of information:

1. A model of the user's preference.
2. A history of the user's interaction with the recommender system.

Basically, these methods use an item profile (i.e., a set of discrete attributes and features) characterizing the item within the system. To abstract the features of the items in the system, an item presentation algorithm is applied. A widely used algorithm is the tf–idf representation (also called vector space representation).[23] The system creates a content-based profile of users based on a weighted vector of item features. The weights denote the importance of each feature to the user and can be computed from individually rated content vectors using a variety of techniques. Simple approaches use the average values of the rated item vector while other sophisticated methods use machine learning techniques such as Bayesian Classifiers, cluster analysis, decision trees, and artificial neural networks in order to estimate the probability that the user is going to like the item. A key issue with content-based filtering is whether the system is able to learn user preferences from users' actions regarding one content source and use them across other content types. When the system is limited to recommending content of the same type as the user is already using, the value from the recommendation system is significantly less than when other content types from other services can be recommended. For example, recommending news articles based on browsing of news is useful, but would be much more useful when music, videos, products, discussions etc. from different services can be recommended based on news browsing. To overcome this, most content-based recommender systems now use some form of hybrid system. Content-based recommender systems can also include opinion-based recommender systems. In some cases, users are allowed to leave text review or feedback on the items. These user-generated texts are implicit data for the recommender system because they are potentially rich resource of both feature/aspects of the item, and users' evaluation/sentiment to the item. Features extracted from the user-generated reviews are improved meta-data of items, because as they also reflect aspects of the item like meta-data, extracted features are widely concerned by the users. Sentiments extracted from the reviews can be seen as users' rating scores on the corresponding features. Popular approaches of opinion-based recommender system utilize various techniques including text mining, information retrieval, sentiment analysis (see also Multimodal sentiment analysis) and deep learning .

3. Multi-criteria recommender systems

Multi-criteria recommender systems (MCRS) can be defined as recommender systems that incorporate preference information upon multiple criteria. Instead of developing recommendation techniques based on a single criterion value, the overall preference of user u for the item i, these systems try to predict a rating for unexplored items of u by exploiting preference information on multiple criteria that affect this overall preference value. Several researchers approach MCRS as a multi-criteria decision making (MCDM) problem, and apply MCDM methods and techniques to implement MCRS systems. See this chapter. for an extended introduction.

4. Risk-aware recommender systems

The majority of existing approaches to recommender systems focus on recommending the most relevant content to users using contextual information yet do not take into account the risk of disturbing the user with unwanted notifications. It is important to consider the risk of upsetting the user by pushing recommendations in certain circumstances, for instance, during a professional meeting, early morning, or late at night. Therefore, the performance of the recommender system depends in part on the degree to which it has incorporated the risk into the recommendation process. One option to manage this issue is DRARS, a system which models the context-aware recommendation as a bandit problem. This system combines a content-based technique and a contextual bandit algorithm.

5.Mobile recommender systems

Further information: Location based recommendation Mobile recommender systems make use of internet-accessing smart phones to offer personalized, context-sensitive recommendations This is a particularly difficult area of research as mobile data is more complex than data that recommender systems often have to deal with. It is heterogeneous, noisy, requires spatial and temporal auto-correlation, and has validation and generality problems. There are three factors that could affect the mobile recommender systems and the accuracy of prediction results: the context, the recommendation method and privacy. Additionally, mobile recommender systems suffer from a transplantation problem – recommendations may not apply in all regions (for instance, it would be unwise to recommend a recipe in an area where all of the ingredients may not be available). One example of a mobile recommender system are the approaches taken by companies such as Uber and Lyft to generate driving routes for taxi drivers in a city. This system uses GPS data of the routes that taxi drivers take while working, which includes location (latitude and longitude), time stamps, and operational status (with or without passengers). It uses this data to recommend a list of pickup points along a route, with the goal of optimizing occupancy times and profits. Mobile recommendation systems have also been successfully built using the "Web of Data" as a source for structured information. A good example of such system is SMARTMUSEUM The system uses semantic modelling, information retrieval, and machine learning techniques in order to recommend content matching user interests, even when presented with sparse or minimal user data.

6. Hybrid recommender systems

Most recommender systems now use a hybrid approach, combining collaborative filtering, content-based filtering, and other approaches. There is no reason why several different techniques of the same type could not be hybridized. Hybrid approaches can be implemented in several ways: by making content-based and collaborative-based predictions separately and

then combining them; by adding content-based capabilities to a collaborative-based approach (and vice versa); or by unifying the approaches into one model (see] for a complete review of recommender systems). Several studies that empirically compare the performance of the hybrid with the pure collaborative and content-based methods and demonstrated that the hybrid methods can provide more accurate recommendations than pure approaches. These methods can also be used to overcome some of the common problems in recommender systems such as cold start and the sparsity problem, as well as the knowledge engineering bottleneck in knowledge-based approaches. Netflix is a good example of the use of hybrid recommender systems. The website makes recommendations by comparing the watching and searching habits of similar users (i.e., collaborative filtering) as well as by offering movies that share characteristics with films that a user has rated highly (content-based filtering).

Some hybridization techniques include:

• **Weighted:** Combining the score of different recommendation components numerically.

• **Switching:** Choosing among recommendation components and applying the selected one.

• Mixed: Recommendations from different recommenders are presented together to give the recommendation.

•**Feature Combination:** Features derived from different knowledge sources are combined together and given to a single recommendation algorithm.

•**Feature Augmentation:** Computing a feature or set of features, which is the part of the input to the next technique?

• **Cascade:** Recommenders are given strict priority, with the lower priority ones breaking ties in the scoring of the higher ones.

• **Meta-level:** One recommendation technique is applied and produces some sort of model, which is then the input used by the next technique.

IV. CLASSIFICATION AND TECHNIQUIES

Many recommendation systems have been developed over the past decades. These systems use different approaches like collaborative approach, content based approach, a utility base approach, hybrid approach etc. Looking at the purchase Behavior and history of the shoppers, Lawrence et al. 2001 presented a recommender system which suggests the new product in the market. To refine the recommendation collaborative and content based filtering approach were used. To find the potential customers most of the recommendation systems today use ratings given by previous users. These ratings are further used to predict and recommend the item of one's choice. In 2007 Weng, Lin and Chen performed an evaluation study which says using multidimensional analysis and additional customer's profile increases the recommendation quality. Weng used MD recommendation model (multidimensional recommendation model) for this purpose. Multidimensional recommendation model was proposed by Tuzhilin and Adomavicius (2001).

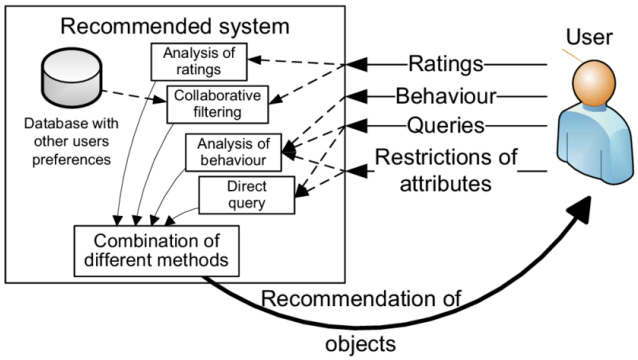

Figure.1. System

V. RELATED WORK

For this exercise, we will consider the Movie Lens small dataset, and focus on two files, i.e., the movies.csv and ratings.csv. Movies.csv has three fields namely:
1. Movie Id – It has a unique id for every movie
2. Title – It is the name of the movie
3. Genre – The genre of the movie

The ratings.csv file has four fields namely:
1. User id – The unique id for every user who has rated one or multiple movies
2. Movie Id – The unique id for each movie
3. Rating – The rating given to a user to a movie
4. Timestamp – When was the rating given to a specific movie

```
#import all necessary libraries
Import os
Import numpy as np
Import pandas as pd
Import matplotlib.pyplot as plt
plt.style.use ('seaborn-bright')
%matplotlib inline
#change directory to the folder where data
files are present
#this step is not necessary if the data files
and jupyter notebook are in same folder
os.chdir(r"C:\Users\mirza\Downloads\Compresse
d\ml-latest-small\ml-latest-small")
#import ratings file in a pandas dataframe
ratings_data=pd.read_csv ("ratings.csv")
ratings_data.head ()
```

Figure.2. Code Segment

	userId	movieId	rating	timestamp
0	1	1	4.0	964982703
1	1	3	4.0	964981247
2	1	6	4.0	964982224
3	1	47	5.0	964983815
4	1	50	5.0	964982931

Figure.3. Output

Movie names=pd.read_csv ("movies.csv")
movie_names.head ()

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

Movie data=pd.merge (ratings_data, movie_names,on, on, on='movie ID')
movie_data.head ()

	userId	movieId	rating	timestamp	title	genres
0	1	1	4.0	964982703	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	5	1	4.0	847434962	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	7	1	4.5	1106635946	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
3	15	1	2.5	1510577970	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
4	17	1	4.5	1305696483	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy

Figure.4. create a dataframe for analysis

Trend=pd. DataFrame (movie_data.groupby ('title') ['rating']. mean ())
Trend ['total number of ratings'] = pd.DataFrame (movie_ data. groupby ('title') ['rating'].count ())
Trend. Head ()

title	rating	total number of ratings
'71 (2014)	4.0	1
'Hellboy': The Seeds of Creation (2004)	4.0	1
'Round Midnight (1986)	3.5	2
'Salem's Lot (2004)	5.0	1
'Til There Was You (1997)	4.0	2

Figure.5. plot rounded-up ratings with number of movies

Plt. Figure (fig size = (10, 4))
As=plt.barh (trend ['rating'].round (), trend ['total number of ratings'], colour ='b')
plt.show ()

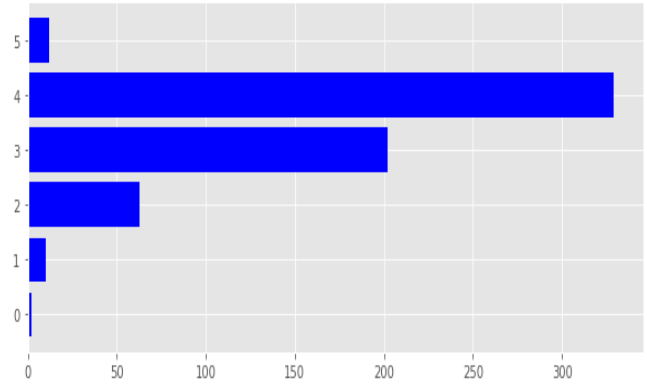

Figure.6. A Bar graph describing number of reviews for first 25 movies

Plt. Figure (fig size =(10, 4))
As=plt. Subplot ()
As.Bar (trend. Head (25).index, trend ['total number of ratings'].head (25), colour='b')
ax.set_xticklabels (trend. Index, rotation=40, font size='12', horizontal alignment="right")
ax.set_title ("Total Number of reviews for each movie")
plt.show ()

Figure.7. Calculate mean rating of all movies and check the popular high rating movies

MOVIE_DATA.GROUPBY('TITLE')['RATING'].MEAN().SORT_VALUES (ASCENDING=FALSE).HEA

```
title
Karlson Returns (1970)                                  5.0
Winter in Prostokvashino (1984)                         5.0
My Love (2006)                                          5.0
Sorority House Massacre II (1990)                       5.0
Winnie the Pooh and the Day of Concern (1972)           5.0
Sorority House Massacre (1986)                          5.0
Bill Hicks: Revelations (1993)                          5.0
My Man Godfrey (1957)                                   5.0
Hellbenders (2012)                                      5.0
In the blue sea, in the white foam. (1984)              5.0
Won't You Be My Neighbor? (2018)                        5.0
Red Sorghum (Hong gao liang) (1987)                     5.0
Love Exposure (Ai No Mukidashi) (2008)                  5.0
My Sassy Girl (Yeopgijeogin geunyeo) (2001)             5.0
The Love Bug (1997)                                     5.0
Ballad of Narayama, The (Narayama bushiko) (1983)       5.0
Heidi Fleiss: Hollywood Madam (1995)                    5.0
Louis Theroux: Law & Disorder (2008)                    5.0
Winnie the Pooh Goes Visiting (1971)                    5.0
In the Realm of the Senses (Ai no corrida) (1976)       5.0
Winnie Pooh (1969)                                      5.0
Ex Drummer (2007)                                       5.0
Tom Segura: Mostly Stories (2016)                       5.0
Tom and Jerry: A Nutcracker Tale (2007)                 5.0
A Plasticine Crow (1981)                                5.0
Tom and Jerry: Shiver Me Whiskers (2006)                5.0
Cosmic Scrat-tastrophe (2015)                           5.0
Delirium (2014)                                         5.0
Lumberjack Man (2015)                                   5.0
Loving Vincent (2017)                                   5.0
```

VI. CONCLUSION

In this paper we have introduced Movie REC, a recommender system for movie recommendation. It allows a user to select his choices from a given set of attributes and then recommend him a movie list based on the cumulative weight of different attributes and using K-means algorithm. By the nature of our system, it is not an easy task to evaluate the performance since there is no right or wrong recommendation; it is just a matter of opinions. Based on informal evaluations that we carried out over a small set of users we got a positive response from them. We would like to have a larger data set that will enable more meaningful results using our system. Additionally we would like to incorporate different machine learning and clustering algorithms and study the comparative results. Eventually we would like to implement a web based user interface that has a user database, and has the learning model tailored to each user.

VII. REFERENCES

[1]. Francesco Ricci and Lior Rokach and Bracha Shapira, Introduction to Recommender Systems HandbookRecommender Systems Handbook, Springer, 2011, pp. 1-35

[2]."playboy Lead Rise of Recommendation Engines - TIME". TIME.com. 27 May 2010. Retrieved 1 June 2015.

[3].Hosein Jafarkarimi; A.T.H. Sim and R. Saadatdoost A Naïve Recommendation Model for Large Databases, International Journal of Information and Education Technology, June 2012

[4]. R. J. Mooney & L. Roy (1999).Content-based book recom mendation using learning for text categorization.In Workshop Recom.Sys.Algo.and Evaluation.

[5]. Prem Melville and Vikas Sindhwani, Recommender Syste ms, Encyclopedia of Machine Learning, 2010.

[6].ChenHung-Hsuan; ChenPu (2019-01-09)."Differentiating Regularization Weights -- A Simple Mechanism to Alleviate Cold Start in Recommender Systems".ACM Transactions on Knowledge Discovery from Data (TKDD).13: 1–22.

[7].Rubens, Neil; Elahi, Mehdi; Sugiyama, Masashi; Kaplan, Dain (2016)."Active Learning in Recommender Systems".In Ricci, Francesco; Rokach, Lior; Shapira, Bracha (Eds.). Recommender Systems Handbook (2 ed.). Springer US.

[8].Karlgren, Jussi. 1990. "An Algebra for Recommendations." Syslab Working Paper 179 (1990).

[9].Karlgren, Jussi. "Newsgroup Clustering Based On User Beha vior-Recommendation Algebra." SICS Research Report (1994).

[10].Karlgren, Jussi (October 2017)A digital bookshelf: original work on recommender systems". Retrieved 27 October 2017.

[11].Shardanand, Upendra, and Pattie Maes. "Social information filtering: algorithms for automating “word of mouth”." In Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 210-217. ACM Press/Addison-Wesley Publishing Co., 1995.

[12].Hill, Will, Larry Stead, Mark Rosenstein, and George Furnas. "Recommending and evaluating choices in a virtual community of use." In Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 194-201. ACM Press/Addison-Wesley Publishing Co., 1995.

[13]. Resnick, Paul, Neophytos Iacovou, Mitesh Suchak, Peter Bergström, and John Riedl. "GroupLens: an open architecture for collaborative filtering of netnews." In Proceedings of the 1994 ACM conference on Computer supported cooperative work, pp. 175-186. ACM, 1994.

[14]. Resnick, Paul, and Hal R. Varian. "Recommender systems." Communications of the ACM 40, no. 3 (1997): 56-58.

[15]. Montaner, M.; Lopez, B.; de la Rosa, J. L. (June 2003). "A Taxonomy of Recommender Agents on the Internet". Artificial Intelligence Review. 19 (4): 285–330. Doi:10.1023/ A:1022 850703159..

[16]. Adomavicius, G.; Tuzhilin, A. (June 2005). "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions". IEEE Transactions on Knowledge and Data Engineering.17 (6): 734–749. CiteSeerX 10.1.1.107.2790.doi:10.1109/TKDE.2005.99..

[17]. Herlocker, J. L.; Konstan, J. A.; Terveen, L. G.; Riedl, J. T. (January 2004). "Evaluating collaborative filtering recommender systems". ACM Trans. Inf. Syst. 22 (1): 5–53. CiteSeerX 10.1.1.78.8384.doi:10.1145/963770.963772..

[18]. Beel, J.; Genzmehr, M.; Gipp, B. (October 2013). "A Comparative Analysis of Offline and Online Evaluations and Discussion of Research Paper Recommender System Evaluatio n"(PDF).Proceedings of the Workshop on Reproducibility and Replication in Recommender Systems Evaluation (RepSys) at the ACM Recommender System Conference (RecSys).

[19]. Beel, J.; Langer, S.; Genzmehr, M.; Gipp, B.; Breitinger, C. (October 2013). "Research Paper Recommender System Evaluation: A Quantitative Literature Survey(PDF). Proceedings of the Workshop on Reproducibility and Replication in Recommender Systems Evaluation (RepSys) at the ACM Recommender System Conference (RecSys).

[20]. Beel, J.; Gipp, B.; Langer, S.; Breitinger, C. (26 July 2015). "Research Paper Recommender Systems: A Literature Survey". International Journal on Digital Libraries.17 (4): 305–338. doi:10.1007/s00799-015-0156-0.

[21].Waila, P.; Singh, V.; Singh, M. (26 April 2016). "A Sciento metric Analysis of Research in Recommender Systems" (PDF). Journal of Scientometric Research.5: 71–84. doi:10 .5530/ jscires.5.1.10.

[22].Parsons, J.; Ralph, P.; Gallagher, K. (July 2004). "Using viewing time to infer user preference in recommender systems".AAAI Workshop in Semantic Web Personalization, San Jose, California.

[23].D.H. Wang, Y.C. Liang, D.Xu, X.Y. Feng, R.C. Guan (2018), "A content-based recommender system for computer science publications", Knowledge-Based Systems, 157: 1-9

[24].Breese, John S.; Heckerman, David; Kadie, Carl (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering(PDF) (Report). Microsoft Research.