FeedMeRight:Comparison of Recipe Recommendation Systems

Pallavi Amod Chavan

April 22, 2020

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

The internet is huge network of machines that connects large number of computers together worldwide allowing them to communicate with any other computer. The World Wide Web is an information sharing model that is built on top of the internet in which information can be accessed or manipulated easily hence experiencing dramatic growth in increased usage of internet which results in BigData.

BigData is an exponentially increasing data with high volume, high velocity with variety. This huge amount of data has intrinsic value but it's of no use until it's discovered [1]. With such information overload it is problematic to find exactly what user is looking for. Search engines provides relative information for what user is looking for but it does not provide any personalization with it.

One of the ways of finding value in BigData is analyzing it with its interrelated features such as new products, corresponding reviews, ratings and user preferences. Forming information from raw data is an entire discovery process that requires insightful analysis that would recognize patterns to predict user behaviors to recommend products.

Handling BigData by manual process is very inefficient. More efficient way of processing such huge amount of data is automating the process of classifying, filtering data of users opinions, features, and preferences in order to understand and predict new set of related products. Recommender systems are tools that filters information and narrow it down based on user's preferences and helps user to choose which he/she may like. They considers opinion of community of users to help each individual in order to understand content of interest from overwhelming information [2].

Recommender system can be defined as a tool designed to interact with large and complex information spaces to provide information or items that are relevant to the user [3].

Nowadays recommender systems are widely used in variety of applications. Initially it applied for commercial use to analyze data. Amazon is a good example of such one of E-commerce websites. However, it is now present in several different domains including entertainment, news, books, social tags and some more sophisticated products where personalization is critical such as recipes domain. With rapid changes in our busy lifestyle, people find it difficult to choose healthy eating option [4]. People tend to move towards options which satisfies their taste by ignoring the health factor which includes required calories and nutrition. Exploring better dishes in such huge information is very tedious. We need system that can help us in narrowing down the information considering our health and eating history. This paper would further discuss the different approaches for recipe domain to recommend healthy recipes based on user's profile.

1.2 Related Work

There are different ways of how recommendations can be made. One of the simplest way is recommending popular products to the user. But in that case, those recommendation will be same for every user. There are well-known techniques that can make personalized recommendations. Recommender Systems techniques which can make personalized recommendations are normally classified into three different categories [5]. Content-based Filtering, Collaborative Filtering (CF) and Hybrid Approach.
Content-based filtering uses contents of the items to and information related to target user. Collaborative filtering primarily focus on set of users and their relation with items. It does not use the data about items. Collaborative filtering systems collects preferences, and with these preferences predict preference of specific user for target items. Hybrid recommender system combines two or more techniques together [6]. Example, combining content-based and collaborative filtering methods in different ways. Recommender systems help in solving the problem of personalized suggestions for any product.
In recent years web application development has grown. With this development food or recipe sharing applications have emerged. Hence the scope of recommender systems in food domain has increased as it is easy to get user feedback in the form of ratings, reviews or comments in the web applications.
There are many reasons why recommending food or recipes are challenging. One of the reason is changing user's mind towards healthy behavior. Another reason is predicting what people would like to eat because it depends on many factors including region, culture, etc. The various work has done before which includes approaches ranging from content based to collaborative filtering to hybrid approach. One of the traditional approach is content based. It tailors the recommendations to the user's taste. A highly personalized system built by breaking down recipe into it's ingredients and scoring based on the ingredients in recipes which were rated positive [4]. It was built on content-based algorithm. Continuation of this work resulted in considering negatively weighting recipes based on ingredients in recipes [7]. Recent work experiment shows that how can one bring healthiness in recommendations by substituting ingredients [8]. Other than content-based, Collaborative filtering based algorithms have also been proposed. Freyne and

Berkovsky experimented nearest neighbor approach (CF - KNN) on ratings. But performance of content-based approach was better. In [7] a recipe recommender implemented using SVD which outperformed content-based and collaborative approach experimented by Freyne and Berkovsky. A New recipe recommender system developed based on matrix factorization which collects ratings and tags in the form of feedback [9]. Continuation of this work resulted in incorporating calorie count [10]. These approches performs well in predicting what user may like according to user's taste. But it does not help in changing user's behaviour towards helathy lifestyle. By incorporating calorie count we can restrict user to to choose right food within range of his liking. In this thesis, I present a study of comparative analysis of recommender approaches in food domain and varying recommender approach to recommend a healthy recipes by incorporating calorie count.

Chapter 2

Background

2.1 Data Collection and Information Filtering

2.1.1 BigData

The standards set by the World Wide Web consortium, formed in 1994, has led to large amounts of information sharing between the users and the hosts. As the users data grows, more hosts are required to sustain the growth. This growth can be viewed from the figure Figure 2.1.

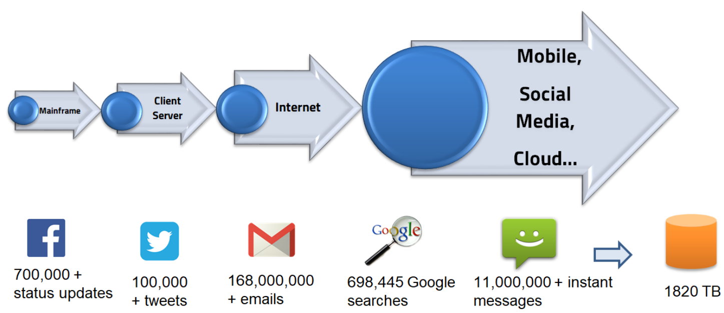

Figure 2.1: Growing Data Velocity per second

To make any customer experience best, some of the ways recommendation technology is used to make it more personalize. These systems can perform well based on the more and more data. Bigdata is a driving force behind recommendation engines. To make the this data useful, applications need to collect data filter data in specific patterns from which we will get useful information.

2.1.2 Data Collection And Storage

Information Gathering involves web crawling, document processing, indexing and queryprocessing. An architectural view of information gathering is shown in figure. A crawler processes all URLs via techniques like breadth-first search and depth-first search and stores the web servers response for each URL. The documents retrieved are then processed in order for its meta-data and to remove any noisy data. Data indexing is then applied so that retrieval and processing of extracted data is quicker.

Implicit Data

Explicit Data

2.1.3 Information Filtering

When a user requests information, it is treated as a query in the form of keywords and is applied on the indexed data. The objective of the query processor is to return most relevant documents to the user.

Filtering is a key component to retrieving adequate information per user preferences or user profile. User behavior is studied from past user profiles activities which helps to filter out any irrelevant data and provide apt suggestions pertaining to current needs. Part of information filtering that tries to predict a user's preference is known as a recommender system.

2.2 Recommender System

A recommender system is an Information Filtering (IF) system that provides or suggests relevant items to user based on the user profile and preferences. Basic idea of general recommender model is given in : Figure 2.2

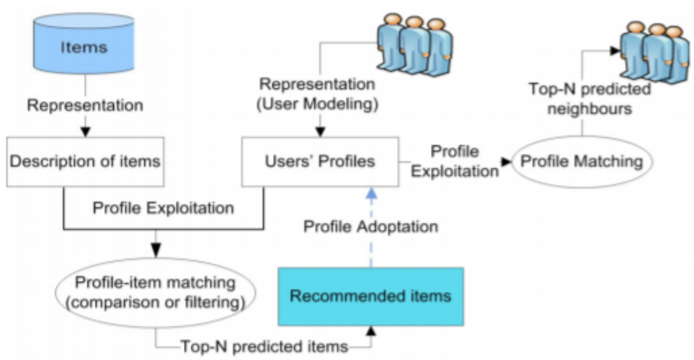

Figure 2.2: General Recommender Model [11]

Traditionally there are two basic models of recommender systems.

- Content based filtering
- Collaborative filtering

2.2.1 Content Based Filtering

In Content based method algorithm, user preference is considered based on item description. The rating and buying behavior of users are combined with content information available in the items. The main aim of content based filtering is to create profile for each item and each user to find similar items the user is looking for [12].

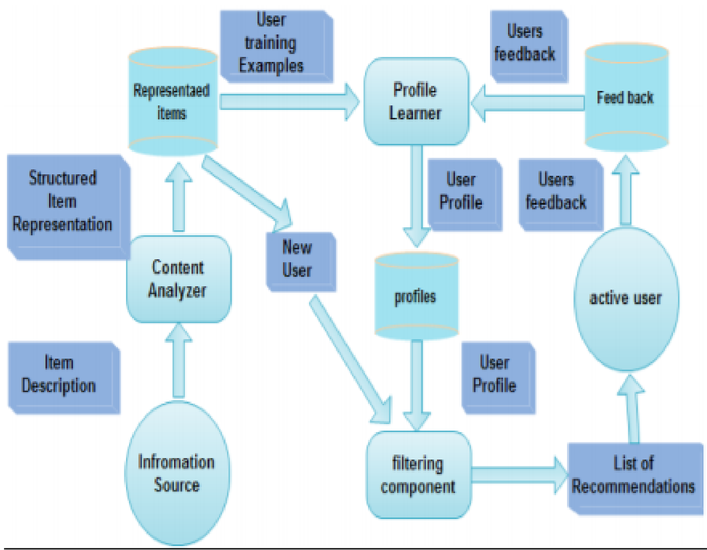

Figure 2.3: Content Based filtering Architecture [13]

In this algorithm each user's information can be stored in vector form which contains past behavior of the user. This vector is known as profile vector or user profile. All the information about item is stored in item vector / item profile which contains all the details about item specific attributes. Based on similarity score between user profile and item profile most relevant items are recommended to user.

Advantages of content-based recommenders are –

Content-based recommender systems are heavily reliable on the contents of the items that have been rated by the user. So, while making recommendations, this approach would consider user's taste and accordingly recommend an item that matches user's preferences. Generally, most popular items dominate less popular items. But this approach will not miss less popular item

if it matches the user's unique taste [12].
Disadvantages of content-based recommenders
User profiles are generated based on rated items. But for any new user who has not rated any items yet, user profile will be empty. In that case, recommending perfect item that matches to user's taste is difficult as system does not have user taste information. This problem is known as cold start. Also, to understand each items feature, system needs to examine content of every item. Therefore if number of items rises quickly, performance of the system decreases [12].

2.2.2 Collaborative Filtering

Collaborative filtering uses other users' behavior in the system to predict and recommend items. It depends on user's contribution such as ratings, reviews which considered as filter for user preference information. The fundamental idea of collaborative filtering is, it selects other users opinions and aggregate in such way that it provides prediction for active user based on his preferences [14].
The main source of input for this algorithm is in the form of matrix of collected user-item ratings. Based on this input it provides recommendations as an output. The first step of output is to predict ratings for items that user may like. Second step is to recommend a list of top rated items as top-N items.

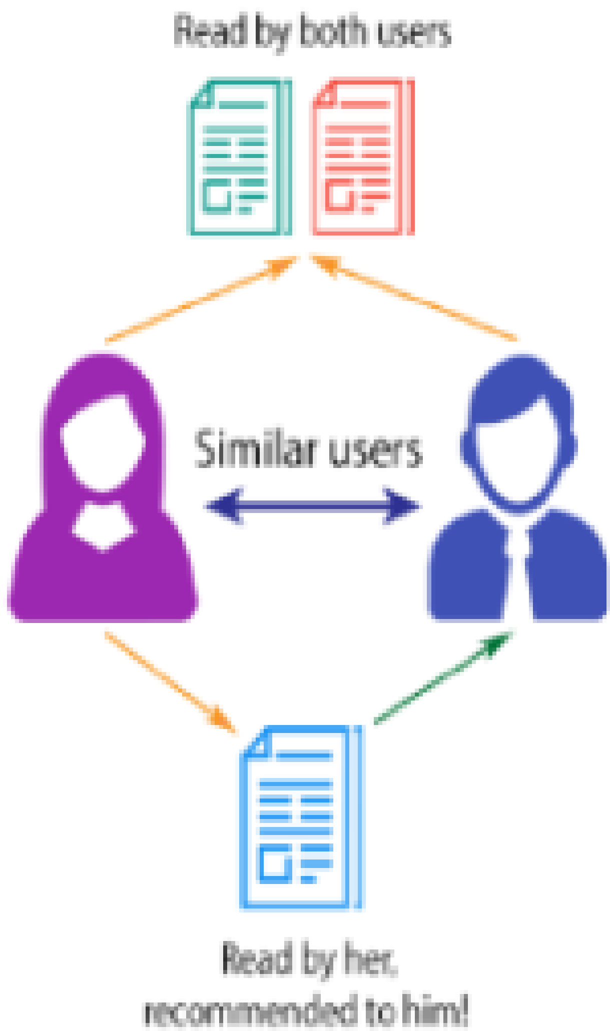

Figure 2.4: Collaborative Filtering [15]

Collaborative Filtering is broadly divided into 2 categories [16].
1. Memory-based CF
2. Model-based CF

Memory-Based (user based)

A memory-based collaborative filtering approach predicts item ratings based on ratings given by different users for an item. There are primary two forms of memory-based collaborative filtering.

User-User CF: Similarity between users is calculated based on how similarly they rate several items. It finds other users whose ratings are similar to active user and use their ratings on other items to predict what active user may like. Thus it recommends items to the users that are most preferred by similar users.
Consider example of user and ratings given by users to different recipes. This algorithm will find similarity between each user based on the ratings they have given to the recipes in the past. The prediction of a recipe for a user u is calculated by computing weighted sum of the user ratings given by other users to recipe i. The prediction for recipe I is given as below:

$$P_{u,v} = \frac{\sum_v (r_{v,i} * S_{u,v})}{\sum_v S_{u,v}} \tag{2.1}$$

Where,
$P_{u,i}$ = prediction of recipe i
$R_{v,i}$ = rating given by user v to recipe i
$S_{u,v}$ =similarity between users.

To predict the ratings for other user we need to calculate similarity score. The similarity between users can be calculated with the help of several methods described in the section 2.4.3 Prior to that we need to find items rated by both users and its rating. Based on that rating, if we opt to calculate similarities with the Pearson correlation then we will get correlation score between users. Higher correlation implied higher similarity. Recommendations are made based on these prediction values.
This algorithm is quite expensive in terms of time as it involves calculating similarity score between each user and from that score calculating predictions.

Item-Item CF: Item-Item CF filtering are introduced to solve challenges in User-User CF. As we seen in user user CF may become so expensive if we have large number of users. If we have huge number of users that items then it is ideal to adopt item-based CF.

This algorithm calculates the similarity between items instead of users. It considers ratings of active user to make predictions for item i, as i will be similar to the items rated in the past by active user. Therefore, user may prefer to use his own ratings than using some other users' ratings. It helps in maintaining user preferences and choice. The similarity between item can be calculated with any formula from section (add number) cosine similarity, Pearson correlation,Jacquard or Eucidean's distance formula.
The rating prediction for item-item collaborative filtering is calculated with below equation:

$$P_{u,i} = \frac{\sum_N (S_{i,N} * R_{u,N})}{\sum_N (|S_{i,N}|)} \tag{2.2}$$

Where,
$P_{u,i}$ = prediction of item i for user u
$R_{u,N}$ = rating given by user u on item N
$S_{i,N}$ =similarity between item i and N.

Memory based collaborative filtering can be useful in any area where we don't need to select many features. At the same time it suffers from dome drawbacks [17].

1. Sparsity:
Many large systems that uses recommender systems to recommend their products has huge number of products in their database. All products are not rated by users. In that case, ratio of actual number of items to number of rated items is very huge. Because of such huge sparsity accuracy of recommender may result in poor recommendations.

2. Scalability:
Nearest neighbor algorithm requires high computations. It grows with number of users and number of items in the system. Any web-based system which has huge number o items and users (example Amazon.com) may suffer from high scalability.

Model-Based (item based):

In contrary to memory-based collaborative filtering, model-based algorithm take the data that has been already preprocessed where it is cleansed, filtered and transformed and generate learned model to make predictions. This algorithm calculates similarity between users or items by generating a model and analyze their pattern to predict ratings on unseen items.
Model-based collaborative filter has several techniques such as SVD, SVD++, Matrix Factorization using gradient descent, Co-clusturing, Slope one approach.

Singulav value Decomposition (SVD)

2.3 Similarity Methods

There are several methods available to calculate similarity score.

2.3.1 Cosine Similarity

In this method [18], the result is the cosine of the angle between two vectors. Either between two item vectors or two profile vectors or one profile vector and item vector. The item ratings or preferences are stored in one vector called as item vector. The preferences of user are stored in another vector based on user's ratings and likes-dislikes is known as profile vector. Consider A and B are profile vector and item vector respectively, the similarity between them can be calculated as shown in Equation 2.3.

$$sim(A, B) = cos(\theta) = \frac{A.B}{\parallel A \parallel \parallel B \parallel} \tag{2.3}$$

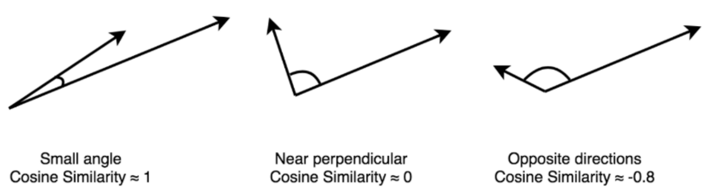

Figure 2.5: Cosine Similarity

The value of cosine angle ranges between -1 to 1. Calculated result 0 shows that there is no similarity between vectors contrary to the result towards 1, which shows exact more similarity between vectors. As we can see in Figure 2.5 lesser the angle, less distance between vectors hence more similarity. Then items are arranged in descending order of similarity score and recommended to user.

2.3.2 Euclidean Distance Similarity

The Euclidean distance between two points is the length which is connecting those two points. If we plot n dimensional space and plot similar items, then they will fall under close proximity. Consider example of positive quadrant of space and we plot items on the axis which are rated by user. The points drawn on graph represents score given by the user to those particular items. For more clear picture of this idea consider figure given below... — TO DO —-

In that case, we can calculate distance between items with Euclidean distance formula which is given by:

$$EuclideanDistance = \sqrt{(x_1 - y_1)^2 + ... + (x_n - y_n)^2} \tag{2.4}$$

2.3.3 Pearson's Correlation Similarity

Person's correlation helps in finding correlation between two users or items. Correlation values ranges from -1 to 1 [19]. Correlation on higher side implies more similarity. Equation 2.5 gives correlation between two users r_u and r_v.

$$sim(u, v) = \frac{\sum(r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{(\sum(r_{ui} - \bar{r}_u))^2}\sqrt{(\sum(r_{vi} - \bar{r}_v)^2}} \tag{2.5}$$

Where r_{ui} and r_{vi} are rating scores for from two users. $\bar{r_u}$ and $\bar{r_v}$ denote the average rating by the two users. Pearson correlation score > 0 indicates positive association. On the other hand, Pearson correlation score < 0 indicates the negative correlation and score $= 0$ indicates that no correlation. Hence correlation value can capture the rating similarity between two users. But if there are no common items between users that they have rated then the similarity measure will be considered as NAN results.

2.4 Evaluation Metrics for Recommendation Systems

In this section we will see few of these methods that are used as evaluation metrics for recommender systems algorithms. Evaluation can be done by two ways, offline and online evaluation [**22**, 20]. In offline analysis collected data is divided into train set and test set in proportion of 80 to 20. The model of recommender system is trained on train dataset and test dataset is hidden from engine. Afterwords build algorithm trained on train dataset is used to predict ratings of unseen items. To understand the quality of recommendation engine, one or combination of evaluation metrics are used. There are several methods available to evaluate the performance of the recommender systems [**22**, 21]. We will see discuss those methods as follow.

2.4.1 Recall and Precision

Recall and precision are most commonly used metrics to evaluate recommendation engines [22]. These metrics can be explained by a confusion matrix [23] as shown in Figure 2.6.

	Recommended	**Not recommended**
Relevant	TP	FN
Irrelevant	FP	TN

Figure 2.6: Confusion matrix for calculating recall and precision [23].

Where
TP denotes True Positive represents all relevant items are recommended by the system.
TN denotes True Negative represents all irrelevant items are correctly not recommended by the system.
FP denotes False Positive represents all irrelevant items which are incorrectly recommended by the system
FN denotes False Negative represents relevant items but system failed to recommend.
Based on Figure 2.6, precision is calculated as ratio of the relevant items from

recommended items to the number of all recommended items. It is given in Equation 2.6.

$$Precision = \frac{TP}{TP + FP} \tag{2.6}$$

Based on Figure 2.6, recall is calculated as the ratio of relevant items from recommended items to the number of all relevant items. It is given in Equation 2.7.

$$Recall = \frac{TP}{TP + FN} \tag{2.7}$$

Larger value of recall and precision implies better recommendations.

2.4.2 Mean Absolute Error (MAE)

Mean absolute error used to calculate the average deviation or error generated from predicted ratings and actual ratings [**26**].

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |Predicted_i - Actual_i| \tag{2.8}$$

Where,
$Predicted_i$ denotes predicted ratings given by user to the item i.
$Actual_i$ denotes actual ratings given by user to the item i.
n denotes number of items.
With this formula, MAE can calculate general performance of recommender systems but to compare engines with different rating scale, we can normalize MAE by dividing by the mean MAE value as shown in Equation 2.9.

$$NMAE = \frac{MAE}{Rating_{max} - Rating_{min}} \tag{2.9}$$

2.4.3 Root Mean Square Error (RMSE)

RMSE is a variation of MAE. It also measures the average magnitude of the error. But it puts more weight on large errors as shown in Equation 2.10. RMSE can be defines as square root of the average of squared deviations between predicted ratings and real ratings.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Predicted_i - Actual_i)^2} \tag{2.10}$$

Where,
$Predicted_i$ denotes predicted ratings given by user to the item i.
$Actual_i$ denotes actual ratings given by user to the item i.
n denotes number of items.
Normalized RMSE is given in Equation 2.11

$$NRMSE = \frac{RMSE}{Rating_{max} - Rating_{min}} \tag{2.11}$$

2.4.4 Mean Reciprocal Rank

——— To DO ———

2.4.5 MAP at k (Mean Average Precision at cutoff k)

——— To DO ———

2.4.6 NDCG (Normalized Discounted Cumulative Gain)

——— To DO ———

2.5 Conclusion and future work

Bibliography

[1] T. L. Nguyen. “A Framework for Five Big V’s of Big Data and Organizational Culture in Firms”. In: *2018 IEEE International Conference on Big Data (Big Data)*. 2018, pp. 5411–5413.

[2] Paul Resnick and Guest Editors Hal R. Varian. “RecommenderSystems”. In: (1997).

[3] Robin Burke, A. Felfernig, and Mehmet Göker. “Recommender Systems: An Overview”. In: *Ai Magazine* 32 (Sept. 2011), pp. 13–18. DOI: 10.1609/aimag.v32i3.2361.

[4] Jill Freyne and Shlomo Berkovsky. “Intelligent Food Planning: Personalized Recipe Recommendation”. In: *Proceedings of the 15th International Conference on Intelligent User Interfaces*. IUI ’10. Hong Kong, China: Association for Computing Machinery, 2010, pp. 321–324. ISBN: 9781605585154. DOI: 10.1145/1719970.1720021. URL: https://doi.org/10.1145/1719970.1720021.

[5] Francesco Ricci, Lior Rokach, and Bracha Shapira. “Introduction to Recommender Systems Handbook”. In: *Recommender Systems Handbook*. Ed. by Francesco Ricci et al. Boston, MA: Springer US, 2011, pp. 1–35. ISBN: 978-0-387-85820-3. DOI: 10.1007/978-0-387-85820-3_1. URL: https://doi.org/10.1007/978-0-387-85820-3_1.

[6] G. Adomavicius and A. Tuzhilin. “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions”. In: *IEEE Transactions on Knowledge and Data Engineering* 17.6 (2005), pp. 734–749.

[7] Morgan Harvey, Bernd Ludwig, and David Elsweiler. “You Are What You Eat: Learning User Tastes for Rating Prediction”. In: *String Processing and Information Retrieval*. Ed. by Oren Kurland, Moshe Lewen-

stein, and Ely Porat. Cham: Springer International Publishing, 2013, pp. 153–164. ISBN: 978-3-319-02432-5.

[8] Chun-Yuen Teng, Yu-Ru Lin, and Lada A. Adamic. “Recipe Recommendation Using Ingredient Networks”. In: *Proceedings of the 4th Annual ACM Web Science Conference*. WebSci ’12. Evanston, Illinois: Association for Computing Machinery, 2012, pp. 298–307. ISBN: 9781450312288. DOI: 10.1145/2380718.2380757. URL: https://doi.org/10.1145/2380718.2380757.

[9] Mouzhi Ge et al. “Using Tags and Latent Factors in a Food Recommender System”. In: *Proceedings of the 5th International Conference on Digital Health 2015*. DH ’15. Florence, Italy: Association for Computing Machinery, 2015, pp. 105–112. ISBN: 9781450334921. DOI: 10.1145/2750511.2750528. URL: https://doi.org/10.1145/2750511.2750528.

[10] Mouzhi Ge, Francesco Ricci, and David Massimo. “Health-Aware Food Recommender System”. In: *Proceedings of the 9th ACM Conference on Recommender Systems*. RecSys ’15. Vienna, Austria: Association for Computing Machinery, 2015, pp. 333–334. ISBN: 9781450336925. DOI: 10.1145/2792838.2796554. URL: https://doi.org/10.1145/2792838.2796554.

[11] Ullah I. Khusro S. Ali Z. “(2016) Recommender Systems: Issues, Challenges, and Research Opportunities”. In: (2016).

[12] Harry Zisopoulos et al. “Content-Based Recommendation Systems”. In: (Nov. 2008).

[13] Marwa Mohamed, Mohamed Khafagy, and Mohamed Ibrahim. “Recommender Systems Challenges and Solutions Survey”. In: Feb. 2019. DOI: 10.1109/ITCE.2019.8646645.

[14] John T. Riedland Joseph A. Konstan Michael D. Ekstrand. *Collaborative Filtering Recommender Systems*. 2011.

[15] Kali Pradeep and M Bhaskar. “Comparative analysis of recommender systems and its enhancements”. In: *International Journal of Engineering and Technology* 7 (Jan. 2018), pp. 304–310.

[16] Aggarwal C.C. “An Introduction to Recommender Systems. In: Recommender Systems”. In: (2016).

[17] Joseph Konstan Badrul Sarwar George Karypis and John Riedl. “Item-Based Collaborative Filtering RecommendationAlgorithms”. In: (2001).

[18] Xiaoyuan Su and Taghi Khoshgoftaar. “A Survey of Collaborative Filtering Techniques”. In: *Adv. Artificial Intellegence* 2009 (Oct. 2009). DOI: 10.1155/2009/421425.

[19] L. Sheugh and S. H. Alizadeh. “A note on pearson correlation coefficient as a metric of similarity in recommender system”. In: *2015 AI Robotics (IRANOPEN)*. 2015, pp. 1–6.

[20] Guy Shani and Asela Gunawardana. “Evaluating Recommendation Systems”. In: *Recommender Systems Handbook*. Ed. by Francesco Ricci et al. Boston, MA: Springer US, 2011, pp. 257–297. ISBN: 978-0-387-85820-3. DOI: 10.1007/978-0-387-85820-3_8. URL: https://doi.org/10.1007/978-0-387-85820-3_8.

[21] Asela Gunawardana and Guy Shani. “A Survey of Accuracy Evaluation Metrics of Recommendation Tasks”. In: *J. Mach. Learn. Res.* 10 (2009), pp. 2935–2962.

[22] Cyril W Cleverdon and Michael Keen. *Aslib Cranfield research project-Factors determining the performance of indexing systems; Volume 2, Test results*. Tech. rep. 1966.

[23] M. Jalili et al. “Evaluating Collaborative Filtering Recommender Algorithms: A Survey”. In: *IEEE Access* 6 (2018), pp. 74003–74024.