Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

The internet is vast network of machines that connects large number of computers together worldwide and allowing them to communicate with one another. 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, which has lead to the dramatic growth in increased usage of internet. One of the results has been the dramatic increase of data and a phenomenon known as BigData.

BigData referes to exponentially increasing data at a high volume, high velocity with variety. This huge amount of data has intrinsic value but provides no utility until it's discovered [1]. With such information overload it is problematic to find exactly what a user is looking for.

One way to find value in BigData is to deeply analyze it and its interrelated features such as new products, corresponding reviews, ratings and user preferences. Formulating 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 through manual processes 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 the opinions of communities of users to help each individual in order to understand content of interest from overwhelming information [2].

Recommender systems 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].

Today, 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 an e-commerce website. 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. According to World Health Organization (WHO) [4], globally of the adults were over-weighted and were obese in 2016. Overweight and obesity

can cause diabetes, blood pressure, heart disease and many other chronic diseases [5]. A well balanced diet plays very important role to improve overall health. With rapid changes in our busy lifestyle, people find it difficult to choose healthy eating option [6]. People tend to move towards options which satisfies their taste or an easy by ignore the health factor which includes required calories and nutrition. Exploring better dishes in such huge information is very tedious. To solve this problem computer scientists can build systems that can help narrowing down the information considering our health and eating history. This thesis further discusses the different approaches for recipe domain to recommend healthy recipes based on user's profile.

TO DO Chapterwise breakdown of thesis. Template to write down breakdown.

The contributions of this paper are the following. First, we provide a short overview of recommendation approaches for individuals. Second, we discuss group decision making issues which have an impact on the development of group recommendation technologies. Third, on the basis of categorizing food recommender systems, we analyze how well those systems can help individuals or groups to choose healthy food which best fits their preferences and health situations. Finally, we point out some challenges of food recommender systems with regard to user information, recommendation algorithms, changing eating behaviors, explanations provision, and group decision making as topics for future work.

The remainder of this paper is organized as follows. In Section 2, we provide an overview of basic recommendation techniques for individuals and groups. In Section 3, we summarize existing studies on food recommender systems for single users and categorize them according to different criteria, such as preferences, nutritional needs, health problems, and eating behaviors of users. Besides, in this section we also discuss some research related to food recommender systems in group scenarios. Research challenges for food recommender systems are discussed in Section 4. The paper is concluded with Section 5.

Chapter 2

Background

2.1 Related Work

In recent years web application development has advanced and 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. Such challenge considers changing user's mind towards their own healthy behavior. Another challenge is in predicting what people would like to eat because it depends on many factors including region, culture and demographic, to name just a few. Prior work in recommender systems includes approaches ranging from content based systems to collaborative filtering systems to hy-

brid approaches.

According to Mika [7], there are two types of food recommender systems present. First type considers user's preferences. It recommends recipes which are similar to the recipes liked by user in the past. Second type considers only those food items which have been identified by health care providers. It recommends recipes based on nutritional requirements for that user with no consideration of a user's preferences.

The research of Freyne and Berkovsky [6] uses content based technique to predict rating of unrated recipe based on the corresponding ingredients included in the recipe. For a user, the framework breaks down unrated recipe into ingredients then calculates rating of each ingredient based on rating of all other rated recipes which contain same ingredients. After calculating ratings for all ingredients of unseen recipe, framework predicts the rating of a user for the unseen recipe based on average of all rating values of all ingredients in this targeted unseen recipe. Recipes with high prediction values will be recommended.

Continuation of Freyne and Berkovsky's [6] work resulted in considering weighting factor for ingredients [8]. This framework take a different approach by using two matrices for user features. One of them is weighted positively for rating value 4 and 5. Weighting factors for rating value 5 and

4 were 2 and 1 respectively. Contrary, another matrix weighted negatively for rating value 1 and 2. Weighting factors for rating value 2 and 1 were 1 and 2 respectively. These matrices implies strong likes and dislikes between ingredients. Based on ingredients similarity score recipes were recommended.

Recent work experiment shows that how can one bring 'healthy' factor in recommendations by substituting ingredients [9].

Other than content-based, Collaborative filtering based algorithms have also been proposed. Freyne and Berkovsky experimented with using the nearest neighbor approach (CF - KNN) on ratings. But performance of content-based approach was better. In [8] a recipe recommender implemented using Singular Value Decomposition (SVD) which outperformed content-based and collaborative approach experimented by Freyne and Berkovsky.

Recent developments in recipe recommender systems consider using matrix factorization. Elahi propose a food recommeder system using active learning algorithm and matrix factorization [10] which collects ratings and tags in the form of feedback. This system provides human computer interaction for collecting user preferences in the form of ratings and tags. Every user and every recipe are modeled by vectors that represent their latent features. Continuation of this work resulted in incorporating calorie count [11]. These approches performs well in predicting what user may like according to user's

taste, but it did not help in changing a user's behavior towards helathy lifestyle. By incorporating calorie count we can restrict a user to choose the right food within range of their personal tastes. In this thesis, I present a comparative analysis of recommender approaches in food domain and varying recommender approach to recommend a healthy recipes by incorporating calorie count and look to extend the work by [10, 11].

2.2 Data Collection and Filtering

2.2.1 BigData

The World Wide Web Consortium (W3C), formed in 1994 []. It has developed standards for communication between computers such as HyperText Transfer Protocol (HTTP), HyperText Markup Language (HTML), Uniform Resource Locator (URL). W3C standards improve the efficiency of communication. It has led to large amounts of information sharing between the users and the hosts on the web. Recently, the ubiquitous availability of internet connections along with mobile technologies has lead to Big Data. It can be described as a large volume of complex data [1]. Volume refers to the amount of data generated in a unit of time. Velocity refers to the speed at which data is generated. Variety refers to the type of data that is generated. For example, as shown in Figure 2.1 we can see and compare that, data generated from mainframe source was less and structured than today's data generated by different channels such as mobile messaging, social media and internet.

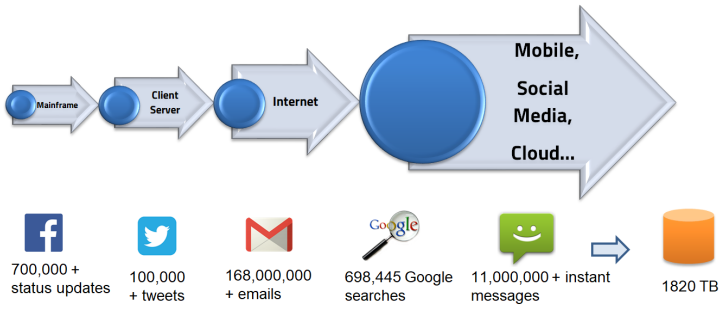

Figure 2.1: Growing Data Velocity per second

Recommender systems can perform well based on a large amount of data. Big Data is a driving force behind recommendation engines. To make this data useful, systems need to collect data, filter data in specific patterns from which we will get useful information.

2.2.2 Data Collection

The data collection phase is the foundation of accuracy of recommendation engine. It is helpful to generate user profiles or models for making recommendations. To well construct user profile, recommendation engine rely on different types of inputs such as explicit feedback which is explicitly specified by a user to notify a user's interest in an item or providing implicit feedback by understanding user preferences with user interaction with the system [12].

Explicit Data

In explicit data application prompts the user through user interface to provide feedback about the product in order to understand user's preferences to improve model. The user then interacts with system by responding to these prompts. Explicit data is any information provided by a user explicitly or **knowingly** in the form of ratings, reviews or comments. For example, below screen-shot shows that Netflix is collecting the data explicitly in the form of ratings given by user to different movies.

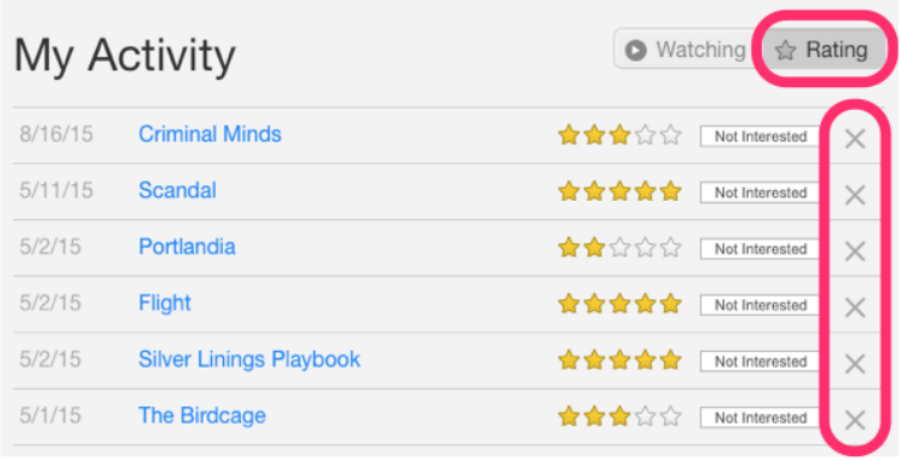

Figure 2.2: Explicit Data [13]

Although explicit feedback requires more efforts from the user, it is still reliable data as it does not require any extraction of preferences and it maintains transparency into recommendation process [14].

Implicit Data

Implicit data considers when a system attempts to understand more about user's preferences by monitoring interactions of user such as browsing history, purchase history, button-clicks, links followed by user. Implicit data is a information provided by user discretely in the form of different interactions. For example, Figure 2.3 shows that Amazon is collecting the data implicitly in the form of storing user's order history.

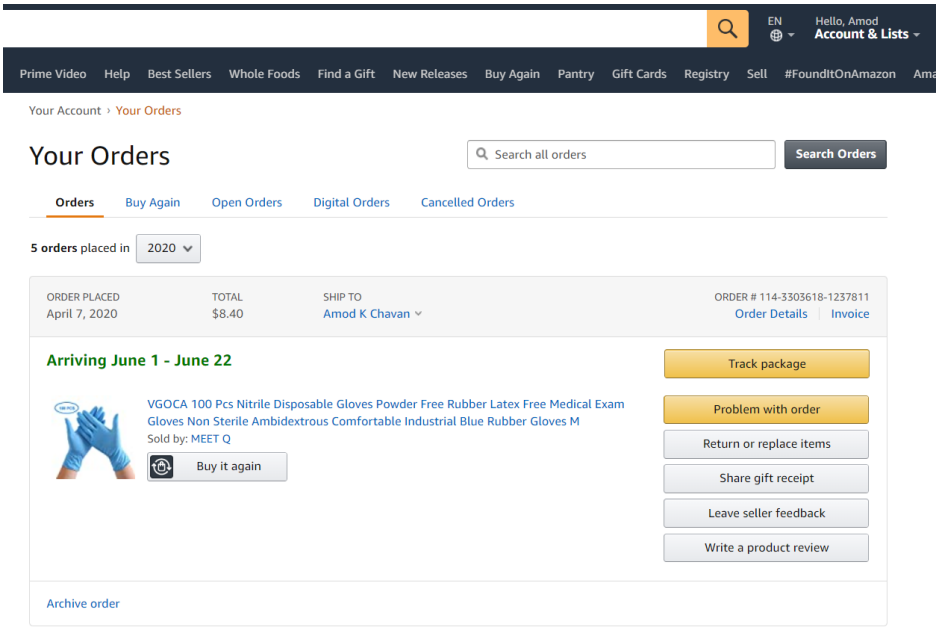

Figure 2.3: Implicit Data

Based on a user's prior purchase, Amazon is recommending other products as shown in screen-shot Figure 2.4.

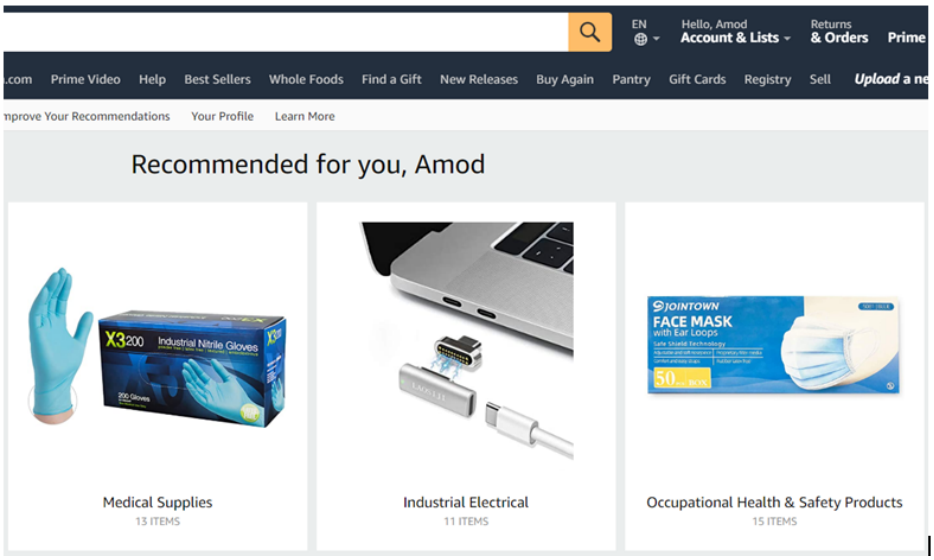

Figure 2.4: Recommendations based on Implicit Data

Although implicit feedback does not require more effort from the user, it is less accurate. For example, a user of an e-commerce system purchase items as gifts for others. Those purchases and related activities do not necessarily communicate anything about the user's tastes. A whole family may share a single account for purchases from e-commerce websites. In that case, input for the system will be an aggregation of all people's preferences.

Explicit and implicit data collection methods are used for building online recommendation system. The work in thesis is based on offline data. Different techniques are involved in offline data collection.

The information gathering process involves many techniques such as web crawling, document processing, indexing and queryprocessing. 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.

2.2.3 Information Filtering

Information Filtering is associated with data search. 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 relevant suggestions pertaining to current needs. Recommendation process has three different phases as highlighted in Figure 2.5. Data collection using different techniques such as explicit data and implicit data is known as the information collection phase. The different learning algorithms apply to the collected data to filter information. Applying algorithms to learn more about data is known as the learning phase. Retrieving most relevant products or items by predicting what the user may prefer from filtered data is known as a recommendation or prediction phase.

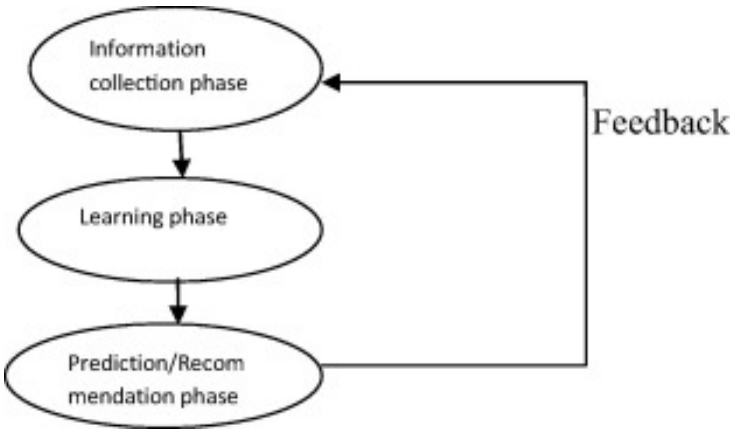

Figure 2.5: Recommendation Phases [15]

2.3 Recommender System

Recommender systems are information filtering systems that provide a solution for the problem of information overload [16]. The process involves filtering important information out of a large amount of data according to the user's preferences and interests. The recommender systems can predict item or product relevancy to the user based on the user's profile and preferences. The basic idea of the general recommender model is given in Figure 2.6 which explains the interaction of users and items with the system. Dataset of items represents the description of items for instance recipes. Description of a recipe may contain information about different factors of recipes such as ingredients used in the recipe, cooking type, calories, cuisine type. User Modeling in the figure represents the user's preferences. Consider a user who likes Mexican recipes with low calories. The profile exploitation step matches the user's profile information with all recipes present in the data set. At this point, the recommendation algorithm gets applied to match the user profile and recipe profile. All Mexican dishes with low calories will get filtered out from the data set for our user. All filtered recipes hold some ranking based on the recommendation algorithm. Top- N recipes will get recommended to our user based on the ranking predicted by the recommendation algorithm. The user adopts those recommendations and responds to the results generated by the recommendation system by providing feedback. The system updates the user profile based on the feedback received by the system from a user.

Recommendation systems correlate one user with a group of other users to find similar users. In this case, the system will match a user's profile with all other users who prefer Mexican low-calorie food.

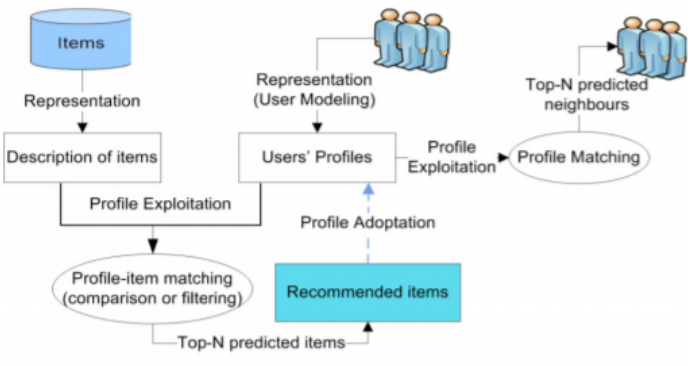

Figure 2.6: General Recommender Model [17]

Based on recommendations generated by system may vary the interaction between users and items. For that reason, we need to understand the features of different recommendation techniques. Figure 2.7 shows broadly categorized recommendation techniques.

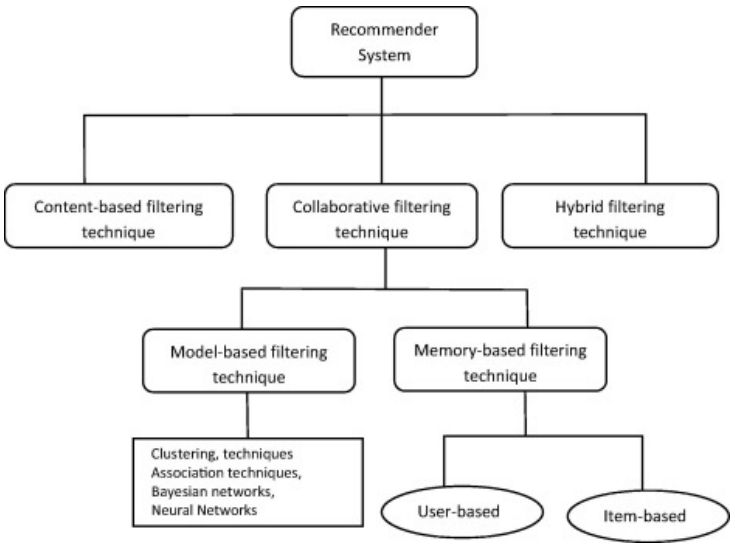

Figure 2.7: Recommendation Techniques [15]

As shown in Figure 2.7, traditionally there are two basic models of recommender systems.

- Content Based Filtering
- Collaborative Filtering
- Hybrid Filtering

In next section we will discuss commonly used methods.

2.3.1 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 a profile for each item as well as each user in order to find similar items that reflects a user's taste [18].

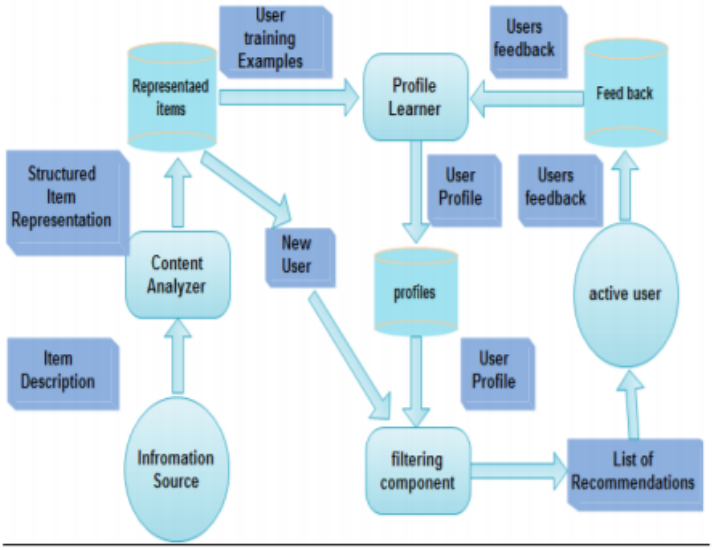

Figure 2.8: Content Based filtering Architecture [19]

The architecture of content based model highlights the process steps in Figure 2.8.

- **Content Analyzer**

  When information has no exact structure like a document that time

pre-processing step is necessary to extract relevant features from that document. Content analyzer analyzes such items as recipes, documents, books, product descriptions to extract and present the content of items. Information source contains data in the raw format. Each data item coming from an information source is pre-processed and analyzed by feature extraction techniques to transform original information into a more structured format. The output of the content analyzer is the input for the profile learner and filtering component.

- **Profile Learner**

  Profile learner constructs a user profile by collecting user preference data. The generalization strategy is applied to the collected data using machine learning techniques. For example, the profile learner of a movie recommender can implement a relevance feedback method in which rating on a scale of 0 to 5 is considered. Rating value above 3 is considered as a positive rating implies likes for the movie and rating value below 3 considered as negative rating implies dislike for a movie. The learning technique combines liked and disliked movies by the user and the vector of positive and negative examples into a vector that represents user profile.

- **Filtering Component**

  Filtering component uses user's profile to find matching items from items data set. The items are matching with a user's data profile is

decided by calculating similarity between a user's data profile and item vectors. A list of potential interesting items is recommended by filtering component.

In content based 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 or 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. The calculation of user or item vector is discussed in subsubsection 2.3.1.

Vector Space Model

Vector Space Model (VSM) for information retrieval represents documents as queries as vectors of weight [20]. It is also known as term vector model as it uses term occurrences as vector identifier.

Each item profile and user profile can be represented in the form of vectors. For instance, consider example of users and books rating relationship based on ratings given by users to the books based on book's genre. The Table 2.1 represents the book and it's genre relationship in binary form. 1 is considered as book shares that genre and 0 represents that book does not share the genre. Book1 is solely based on Machine Learning while book3 talks about Security and Databases. Here genre of book is an attribute or feature that is considered to represent a book in vector form. Rows in Table 2.1 represents

	Machine Learning	Databases	Cyber Security
book1	1	0	0
book2	1	0	1
book3	0	1	1

Table 2.1: Books in Vector Form

	user ratings
book1	1
book2	0
book3	1

Table 2.2: User Rating for Books

item vector for book1, book2 and book3 respectively.

Similarly, a user profile can be represented in a vector form. Consider a user liked book1 and book3 where 1 represents like and 0 represents dislike as shown in Table 2.2.

The simplest way to form a user vector is to take a sum of dot product of matrices as resulted in Table 2.3.

Advantages of content-based filtering

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 dom-

Machine Learning	Databases	Cyber Security
1	1	1

Table 2.3: User Vector for Book's Genres

inate less popular items. But this approach will not miss less popular item if it matches the user's unique taste [18].

Disadvantages of content-based filtering

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 [18].

2.3.2 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 [21].

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. Prediction is a numerical that represents predicted score of specific item for specific user. Second step is to recommend a list of top rated items as top-N items. Figure 2.9 highlights the collaborative filtering process.

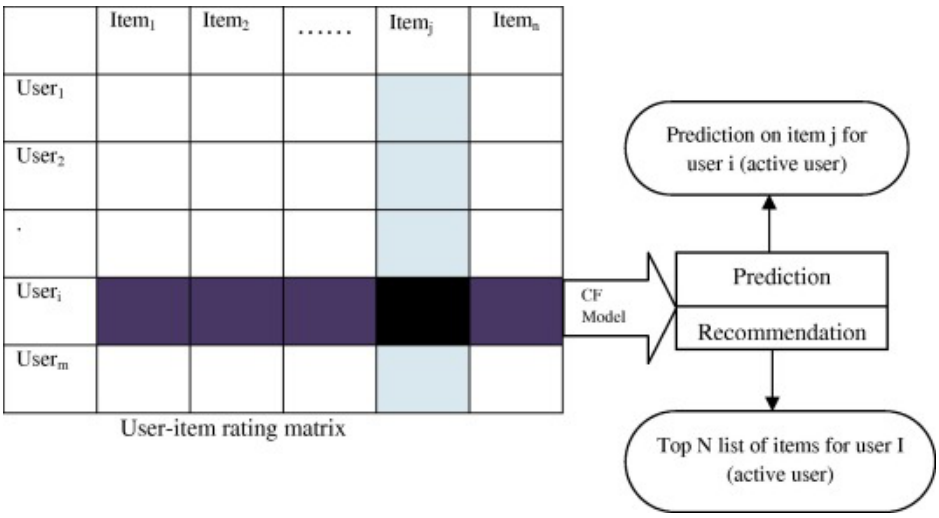

Figure 2.9: Collaborative Filtering Process [15]

Collaborative Filtering technique is broadly divided into 2 categories [22].

- Memory-based CF
- Model-based CF

A. 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.

i. 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 of Similarity Methods. 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.

ii. 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 [23].

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.

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.

B. 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 analyzing their pattern to predict ratings on unseen items [24, 25, 26] . Model-based collaborative filter has several techniques such as Slope one [27], Matrix Factorization (MF) and Singular Value Decomposition (SVD) [24].

In this section we will see frequently used model-based techniques.

i. Singular Value Decomposition (SVD) SVD is a technique of matrix factorization which is used to reduce the number of features in the data set. The matrix factorization is done on the matrix which is generated by the user's feedback in the form of ratings on different items. In SVD, the technique is used to detect latent relationship between users and items. Then it generate a low dimensional representation of original matrix space to calculate neighborhood in the reduced space [28]. The original ratings matrix

decomposes by SVD in two matrices in such way that product of decomposed matrices is original rating matrix. The SVD is calculated as shown in Equation 2.3

$$SVD(M) = U \times S \times V^T \tag{2.3}$$

Where,

$SVD(M)$ denotes matrix M with dimensions $m \times n$ which are total number of users and items respectively.

Dimensions of matrix U will be $m \times m$

Dimensions of matrix S will be $m \times r$

Dimensions of matrix V will be $r \times n$

U and V are called left and right singular vectors. To reduce features of dataset one can keep only k highest values and eliminate lower entries. So, $(r - k)$ columns from U and $(r - k)$ rows from V^T are discarded to gnerate U_k and V_k^T matrices. Now M_k can be constructed with multiplication of U_k and V_k together using S_k. Generated M_k will closest rank k matrix to M. Mathematically it can be represented as in Equation 2.4

$$M_k = U_k \times S_k \times V_k^T \tag{2.4}$$

Rating prediction for user u for item i is given in Equation 2.5

$$r_{ui} = r_u + U_k \sqrt{S_k^T(u)} \times \sqrt{S_k} \times V_k^T \tag{2.5}$$

With SVD we can predict ratings with good accuracy.

2.3.3 Hybrid Filtering

Pure recommendation systems have some limitations. Hybrid filtering technique is a combination of different recommendation techniques to overcome limitations of pure recommendation techniques to improve performance [29, 30]. Many researchers have combined content-based and collaborative filtering techniques to gain better results. The idea behind combining difeerent recommendation techniques is that resultant algorithm will provide more accurate and effective recommendations than any single algorithm [31]. Using mulitple techniques we can overcome weaknesses of single technique. Burk [] has categorized hybrid techniques in different types. One of them is weighted Hybridization.

Weighted Hybridization

In this technique, the results of multiple algorithms are combined to generate predictions by integrating the scores of each algorithm used. For example, [32] P-Tango system combines content-based and collaborative filtering techniques. At start, both techniques were equally weighted but based on per-

formance and user ratings the weight for different techniques has gradually adjusted. In this paper, we will be using weighted hybrid technique.

2.4 Similarity Methods

There are several methods available to calculate similarity score.

2.4.1 Cosine Similarity

In this method [33], 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 2.6.

$$sim(A, B) = cos(\theta) = \frac{A.B}{\parallel A \parallel \parallel B \parallel} \tag{2.6}$$

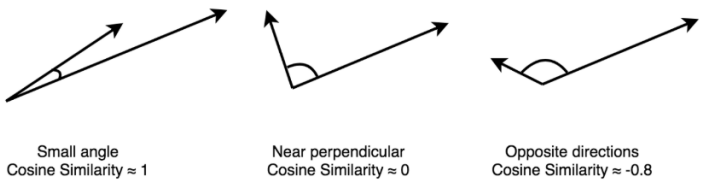

Figure 2.10: Cosine Similarity

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 2.10 lesser the angle, less distance between vectors hence more similarity. Then items are arranged in descending order of similarity score and recommended to user.

2.4.2 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:

$$EuclideanDistance = \sqrt{(x_1 - y_1)^2 + ... + (x_n - y_n)^2} \tag{2.7}$$

2.4.3 Pearson's Correlation Similarity

Person's correlation helps in finding correlation between two users or items. Correlation values ranges from -1 to 1 [34]. Correlation on higher side implies more similarity. Equation 2.8 gives correlation between two users r_u and r_v.

$$sim(u,v) = \frac{\sum(r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{(\sum(r_{ui} - \bar{r}_u))^2}\sqrt{(\sum(r_{vi} - \bar{r}_v)^2}} \tag{2.8}$$

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.

2.5 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 [35, 36]. 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 [35, 37]. We will see discuss those methods as follow.

2.5.1 Recall and Precision

Recall and precision are most commonly used metrics to evaluate recommendation engines [38]. These metrics can be explained by a confusion matrix [39] as shown in Figure 2.11.

	Recommended	Not recommended
Relevant	TP	FN
Irrelevant	FP	TN

Figure 2.11: Confusion matrix for calculating recall and precision [39].

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 2.11, precision is calculated as ratio of the relevant items from recommended items to the number of all recommended items. It is given in Equation 2.9.

$$Precision = \frac{TP}{TP + FP} \tag{2.9}$$

Based on Figure 2.11, recall is calculated as the ratio of relevant items from recommended items to the number of all relevant items. It is given in Equation 2.10.

$$Recall = \frac{TP}{TP + FN} \tag{2.10}$$

Larger value of recall and precision implies better recommendations.

2.5.2 Mean Absolute Error (MAE)

Mean absolute error used to calculate the average deviation or error generated from predicted ratings and actual ratings [40].

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |Predicted_i - Actual_i| \tag{2.11}$$

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 2.12.

$$NMAE = \frac{MAE}{Rating_{max} - Rating_{min}} \tag{2.12}$$

2.5.3 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 2.13. RMSE can be defines as square root of the average of squared deviations between predicted ratings and real ratings.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Predicted_i - Actual_i)^2} \tag{2.13}$$

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 2.14

$$NRMSE = \frac{RMSE}{Rating_{max} - Rating_{min}} \tag{2.14}$$

2.5.4 Mean Reciprocal Rank

Mean Reciprocal Rank is one of the rank-aware evaluation metrics. It is retrieval measure which calculates the reciprocal of the rank at which the first relevant document was recommended [41]. From the list of generated recommendations, it finds the rank of first relevant recommended item and computes the reciprocal of that rank. It can be calucluted as shown in

Equation 2.15

$$MRR(O,U) = \frac{1}{|U|}\sum_{u\in\epsilon}\frac{1}{k_u} \tag{2.15}$$

Where,

k_u denotes first relevant recommended item for user u.

U denotes number of users.

MRR focuses on the first relevant item of the recommended list.HIgher reciprocal rank implies better recommendation engine.

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

MAP at k considers the subset of list recommended by the system while measuring precision. It does not concern about ordering the items in the list. For each user for each relevant item it computes the precision of list through that item. After that it average sub list precisions. Mathematically it is show in Equation 2.16

$$MAP_i = \frac{1}{|R_i|}\sum_{k=1}^{|R_i|}P(R_i[k]) \tag{2.16}$$

Where,

R_i denotes rating for item i.

k denotes rank from 1 to k.

P denotes precision of first relevant item.

Higher mean precision value implies correct recommendations.

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] WHO. *Obesity and Overweight*. WHO.

[5] Aileen Robertson. *Food and health in Europe: a new basis for action*. 96. WHO Regional Office Europe, 2004.

[6] 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.

[7] Stefanie Mika. "Challenges for nutrition recommender systems". In: *Proceedings of the 2nd Workshop on Context Aware Intel. Assistance, Berlin, Germany*. Citeseer. 2011, pp. 25–33.

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

[9] 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.

[10] 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.

[11] 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.

[12] Douglas W Oard, Jinmook Kim, et al. "Implicit feedback for recommender systems". In: *Proceedings of the AAAI workshop on recommender systems*. Vol. 83. WoUongong. 1998.

[13] Joseph A Konstan Michael D. Ekstrand. *Introduction to Recommender Systems: Non-Personalized and Content-Based*. Coursera.

[14] Jürgen Buder and Christina Schwind. "Learning with personalized recommender systems: A psychological view". In: *Computers in Human Behavior* 28.1 (2012), pp. 207–216.

[15] FO Isinkaye, YO Folajimi, and BA Ojokoh. "Recommendation systems: Principles, methods and evaluation". In: *Egyptian Informatics Journal* 16.3 (2015), pp. 261–273.

[16] Joseph A Konstan and John Riedl. "Recommender systems: from algorithms to user experience". In: *User modeling and user-adapted interaction* 22.1-2 (2012), pp. 101–123.

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

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

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

[20] Massimo Melucci. “Vector-Space Model”. In: *Encyclopedia of Database Systems*. Ed. by LING LIU and M. TAMER ÖZSU. Boston, MA: Springer US, 2009, pp. 3259–3263. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_918. URL: https://doi.org/10.1007/978-0-387-39940-9_918.

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

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

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

[24] Manizheh Ranjbar et al. “An imputation-based matrix factorization method for improving accuracy of collaborative filtering systems”. In: *Engineering Applications of Artificial Intelligence* 46 (2015), pp. 58–66.

[25] Thomas Hofmann. “Collaborative filtering via gaussian probabilistic latent semantic analysis”. In: *Proceedings of the 26th annual interna-*

tional ACM SIGIR conference on Research and development in informaion retrieval. 2003, pp. 259–266.

[26] Kai Yu et al. "Instance selection techniques for memory-based collaborative filtering". In: *Proceedings of the 2002 SIAM International Conference on Data Mining*. SIAM. 2002, pp. 59–74.

[27] Daniel Lemire and Anna Maclachlan. "Slope one predictors for online rating-based collaborative filtering". In: *Proceedings of the 2005 SIAM International Conference on Data Mining*. SIAM. 2005, pp. 471–475.

[28] B Sarwar et al. "Application of dimensionality reduction in recommender system-a case study: DTIC Document". In: *Technology Report* (2000).

[29] Gediminas Adomavicius and Jingjing Zhang. "Impact of data characteristics on recommender systems performance". In: *ACM Transactions on Management Information Systems (TMIS)* 3.1 (2012), pp. 1–17.

[30] David H Stern, Ralf Herbrich, and Thore Graepel. "Matchbox: large scale online bayesian recommendations". In: *Proceedings of the 18th international conference on World wide web*. 2009, pp. 111–120.

[31] J Ben Schafer et al. "Collaborative filtering recommender systems". In: *The adaptive web*. Springer, 2007, pp. 291–324.

[32] Mark Claypool et al. "Combing content-based and collaborative filters in an online newspaper". In: (1999).

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

[34] 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.

[35] LOREN G. TERVEEN JONATHAN L. HERLOCKER JOSEPH A. KONSTAN and JOHN T. RIEDL. "Evaluating Collaborative FilteringRecommender Systems". In: (2004).

[36] 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.

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

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

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

[40] John S Breese, David Heckerman, and Carl Kadie. "Empirical analysis of predictive algorithms for collaborative filtering". In: *arXiv preprint arXiv:1301.7363* (2013).

[41] Nick Craswell. "Mean Reciprocal Rank". In: *Encyclopedia of Database Systems*. Ed. by LING LIU and M. TAMER ÖZSU. Boston, MA: Springer US, 2009, pp. 1703–1703. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_488. URL: https://doi.org/10.1007/978-0-387-39940-9_488.