DYNAMIC DETERMINATION OF NEIGHBORHOOD IN NEIGHBORHOOD-BASED COLLABORATIVE FILTERING ALGORITHMS

Master of Science Thesis

Halil ZEYBEK

Eskişehir 2017

DYNAMIC DETERMINATION OF NEIGHBORHOOD IN NEIGHBORHOOD-BASED COLLABORATIVE FILTERING ALGORITHMS

Halil ZEYBEK

MASTER OF SCIENCE THESIS

Department of Computer Engineering
Supervisor: Assoc. Prof. Dr. Cihan KALELİ

Eskişehir
Anadolu University
Graduate School of Science
December 2017

FINAL APPROVAL FOR THESIS

This thesis titled ”Dynamic Determination of Neighborhood in Neighborhood-based Collaborative Filtering Algorithms” has been prepared and submitted by Halil ZEYBEK in partial fullfillment of the requirements in ”Anadolu University Directive on Graduate Education and Examination” for the Degree of Master of Science in Computer Engineering Department has been examined and approved on 18/12/2017.

Committee Members		**Signature**
Member (Supervisor)	: Assoc. Prof. Dr. Cihan KALELİ	....
Member	: Asst. Prof. Dr. Alper Kürşat UYSAL	....
Member	: Asst. Prof. Dr. Efnan ŞORA GÜNAL	....

Prof. Dr. Ersin YÜCEL
Director of Graduate School of Science

ABSTRACT

DYNAMIC DETERMINATION OF NEIGHBORHOOD IN NEIGHBORHOOD-BASED COLLABORATIVE FILTERING ALGORITHMS

HALİL ZEYBEK

Department of Computer Engineering
Anadolu University, Graduate School of Science, December 2017

Supervisor: Assoc. Prof. Dr. Cihan KALELİ

Collaborative filtering is a commonly used method to reduce information overload. It is widely used in recommendation systems due to its simplicity. In traditional collaborative filtering, recommendations are produced based on similarities among users/items. In this approach, the most correlated k neighbors are determined, and a prediction is computed for each user/item by utilizing this neighborhood. During recommendation process, a predefined k value as a number of neighbors is used for prediction processes. In this thesis, the effect of selecting different k values for each user or item was analyzed. For this purpose, a model that determines k values for each user or item at the off-line time was generated. Empirical outcomes show that using the dynamic k values during the k-nn algorithm leads to more favorable recommendations compared to a constant k value.

Keywords: k-nearest-neighbor, Collaborative Filtering, Dynamic k, Accuracy

ÖZET

KOMŞULUĞA DAYALI ORTAK FİLTRELEME ALGORİTMALARINDA KOMŞULUĞUN DİNAMİK OLARAK BELİRLENMESİ

HALİL ZEYBEK
Bilgisayar Mühendisliği Anabilim Dalı
Anadolu Üniversitesi, Fen Bilimleri Enstitüsü, Aralık 2017
Danışman: Doç. Dr. Cihan KALELİ

İşbirlikçi filtreleme kolay kullanılabilirliği sayesinde öneri sistemlerinde sıklıkla kullanılan bilgi yığınını azaltmak amacıyla kullanılan bir yöntemdir. Geleneksel olarak kullanılan işbirlikçi filtrelemede tahmin üretimi benzer kullanıcılar ya da benzer nesneler temelinde yapılır. Bu yaklaşımda, kullanıcıya ya da nesneye en çok benzeyen ilk k sayıdaki komşu belirlenir, sonrasında ise belirlenen k komşuya dayalı bir tahmin üretilir. Bu süreçte, k değeri sürecin başında belirlenir ve her kullanıcı ya da her nesne için aynı k değeri kullanılır. Bu tezde, her kullanıcı ya da her nesne için farklı k değerleri seçmenin üretilen tahminlerin doğruluğundaki etkisi analiz edildi. Bu amaçla, her kullanıcı ya da nesne için farklı k değerleri denenerek, o kullanıcı ya da nesne için en iyi tahminler üretilebileceği düşünülen k değerleri atandı. Yapılan deneyler en yakın k komşuluk algoritmasında dinamik k değerleri kullanmanın sabit olarak belirlenen bir k değerine göre daha iyi sonuçlar verdiğini gösterdi.

Anahtar Sözcükler: En Yakın k Komşuluk, İşbirlikçi Filtreleme, Dinamik k, Doğruluk

ACKNOWLEDGEMENTS

I am thankful to my advisor Assoc. Prof. Dr. Cihan Kaleli for the support and the guidance he showed me throughout my research. Also, I am very grateful to my wife for her patience during my study.

Halil ZEYBEK
2017 December

18/12/2017

STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES

I hereby truthfully declare that this thesis is an original work prepared by me; that I have behaved in accordance with the scientific ethical principles and rules throughout the stages of preparation, data collection, analysis and presentation of my work; that I have cited the sources of all the data and information that could be obtained within the scope of this study, and included these sources in the references section; and that this study has been scanned for plagiarism with “scientific plagiarism detection program” used by Anadolu University, and that “it does not have any plagiarism” whatsoever. I also declare that, if a case contrary to my declaration is detected in my work at any time, I hereby express my consent to all the ethical and legal consequences that are involved.

Halil ZEYBEK

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVATIONS

MAE	:	Mean Absolute Error
k-NN	:	k-nearest neighbors
CF	:	Collaborative Filtering
NaN	:	Not a number
PCA	:	Principle Component Analysis
PCC	:	Pearson Correlation Coefficient
SVD	:	Singular Value Decomposition
SVM	:	Support Vector Machines

1 INTRODUCTION

People live in a digital age where data originate from many different sources. A large percentage of data arises from the Internet usage [1]. In recent days, people all over the world use the Internet to satisfy most of their needs such as shopping, reading newspapers, banking and planning their holidays. While people use the Internet, they leave an enormous information behind them. The size of data which is stored electronically is increased gradually and reached very high dimensions. To give numerical examples of growing data volumes, it is expected that the data volume of 4.4 zettabytes in 2013 will reach 44 zettabytes in 2020 and 180 zettabytes in 2025.(1 zettabyte is 2^{40} gigabytes). Especially the websites like Facebook, Twitter, Youtube and Google, instant messaging applications like WhatsApp are the leading resources of data volume increase. The increasing volume size of data attracts some social media, e-commerce and some other companies which want to use the power of the data. These kind of companies aim to present the most accurate content, give the most accurate prediction to people by using the data which can be composed of users' comments, ratings that users give to an item, mouse clicks, even the length of time users spend on a specific item.

Recommendation systems, which are simply the software tools and techniques, utilize this information to provide suggestions to the users [2]. Many web sites operating in the field of social media, e-commerce, etc. use recommender systems for their own benefits like earning more money. For example, e-commerce web sites like Amazon always recommends some products to the users to encourage them to buy by taking into account users past habits. Recommender systems are also used in non-profit web-sites to estimate the preferences of people who are using Internet. Recommender systems have two main entities; these are users and items. User is the one of these entities to which recommendation is supplied. The content, product or something else which are provided to the user as a recommendation is called item. Recommendation systems have some processes to make successful predictions. The first step is the data collection and mining. Data should be collected by users ratings, comments or something else that provides insight about the users and then this data should be made meaningful by joining the data with some other data and using data mining techniques. After these steps, some similarity calculations are done between users or items and finally a prediction is produced. Basically, recommender systems have two main tasks:

1. Analyzing the users' data

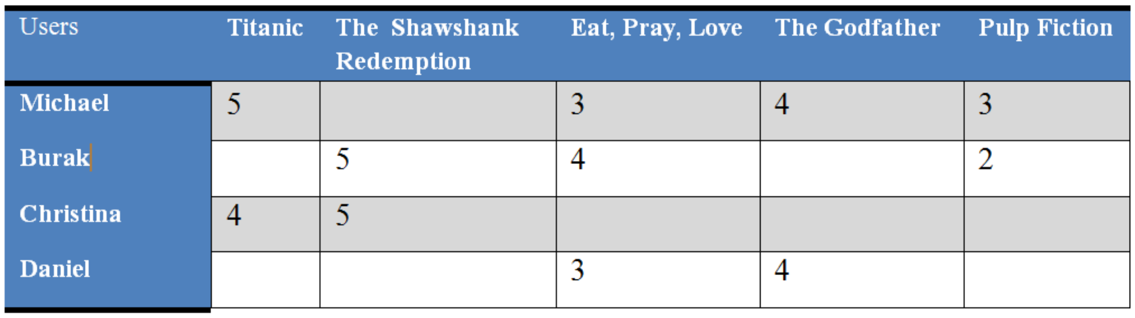
2. Producing user-specific predictions based on the analyzed data

For example, in an e-commerce web site which sells books, if a user of this web-site has a tendency to self-help books, this web site probably recommends self-help books which are not read by this user before to the user by employing recommendation techniques. Recommender systems basically work with two kinds of data, which are the user-item interactions, such as ratings or buying behavior, and the attribute information about the users and items such as textual profiles or relevant keywords with item such as the subject of the movie [18]. Collaborative Filtering(CF) is one of the most popular techniques in recommender systems that are widely used all over the world. It aims to provide successful recommendations by collecting preferences of users and taking into account the users' habits while producing the prediction.

CF is a widely used recommendation method to make suggestions to the users based on their preferences [3]. The world's leading online service providers, e.g., Amazon, Spotify, TripAdvisor, etc. use the CF to meet the customer satisfaction and to increase the trading size by exploring the relevant products based on the history of user preferences [4].

1.1 Collaborative Filtering

CF is a recommendation technique which gives user-specific recommendations to the users of the system that employs CF. CF uses a user-item matrix that contains users as rows and items as columns in principle. Each cell in the matrix represents the rating value the user in that row has given to the item in that column.



Users	Titanic	The Shawshank Redemption	Eat, Pray, Love	The Godfather	Pulp Fiction
Michael	5		3	4	3
Burak		5	4		2
Christina	4	5			
Daniel			3	4	

Figure 1.1. *An example of user-item matrix*

CF tries to find relationships between users or items in user-item matrix. While doing this, it calculates similarities between an active user or item and all other users or items in the dataset. CF assumes that users with similar preferences in the past will have similar preferences in the future or users act in the future like in the past. For example, if a user has watched a war movie in the past, that user will watch a war movie in the future. Namely, CF firstly picks an active user and then finds similar users to active user or similar items to items active user likes before and provides a list of items will most likely be liked by active user. CF compares users according to their preferences [23]. In order for CF systems to make predictions, a dataset that contains users' preferences is required. The dataset can be obtained explicitly or implicitly. 'Explicitly' word means that the user rates for specific item with a measurable value. For example, 4 points that Alice gives to Titanic movie is an explicit rating. On the other hand, 'implicitly' word means that CF derives a point from the users' behavior. For example, 5 minutes that a user spends on an item in the e-commerce web site means this user likes that item. In the user-item matrix which CF works on there can be many missing values in cells. In other words, there are many items those are not rated by specific user. CF aims to produce prediction for these cells. While doing this, CF can use some methods. There are two types of methods which are commonly used in CF, which are referred to as memory and model based methods.

Memory-based and model-based methods are two of the most common CF methods used in the literature. While neighborhood-based and heuristic-based methods are examples of memory-based CF methods, Bayesian Clustering, Singular Value Decomposition (SVD), and Support Vector Machines (SVM) are the implementations of the model-based CF methods [5–8]. In the memory-based CF methods, a two-dimensional rating matrix, which has the users as rows and the items as columns, is persisted in the system and used to compute the prediction depending on the past habits. User-based and item-based methods both use the rating matrix and are instances of neighborhood-based CF methods. In user-based CF algorithm, neighbors of user u who have similar rating pattern with the user u are specified, and a prediction is computed according to the preferences of these users. Similarly, in item-based CF method, neighbors of a target item are specified, and then a prediction is computed. On the other hand, a predictive model is constructed from the rating matrix, and then the predictions are computed via this model in a type of CF method.

1.1.1 Memory-based methods

Memory-based methods are based on the fact that similar users, who give similar points to similar items as a rating, will give similar points to similar items in future or similar items receive similar ratings [18]. Memory-based methods are techniques also called neighborhood-based algorithms. Neighborhood-based algorithms have two main types which are referred to user-based CF and item-based CF. In user-based CF, firstly a set of users is specified which are similar to active user in terms of similarity measures like distance, and then produce a prediction value by taking into account the rating values are given by users which are similar to active user. In order to find neighbors of active user a, the similarities are calculated between active user and all other users in the user-item matrix. To calculate similarity between users a similarity function such as Pearson Correlation Coefficient, Spearman Coefficient etc. is needed. While calculating similarities between two users one of the most important steps is the identifying co-rated items. For instance, let item 1, 3, 5 and 6 are rated by user1 and item 1, 2 and 5 are rated by user2. While calculating the similarity between only item 1 and 5 which are both rated by user1 and user2 are taken into account directly. In item-based CF, firstly a set of items is specified which are similar to active item in terms of similarity measures like distance, and then produce a prediction value by taking into account the rating values are given by users which are similar to active item. In other words, similarities between items instead of users are calculated in the item-based CF algorithm. After the similarity calculation step, a rating value is estimated for the target item by taking into account neighbors of the active item. While the ratings in the user-based CF are predicted using the ratings of users who are similar to active user, in the item-based CF predictions are made by taking into account the users' own ratings which are given for items similar to active item. Actually, both of methods use neighborhood definition but while user-based CF defines neighborhood as similarities among users, item-based CF defines neighborhood as similarities among items.

One of the most popular approaches to memory-based CF recommendation is the k-nearest-neighbor (k-NN) algorithm [2]. The k-NN algorithm depends on the similarities among users or items. CF systems persist the members' item ratings in a 2-dimensional user-item matrix [5, 11]. While predicting a rating value in a user-item matrix, top k closest neighbors are taken into account [9]. For instance, while estimating prediction value of a target movie for a user, firstly correlations between the user and all other users are calculated, and then the k closest neighbors are selected, and finally, a prediction value is estimated [10].

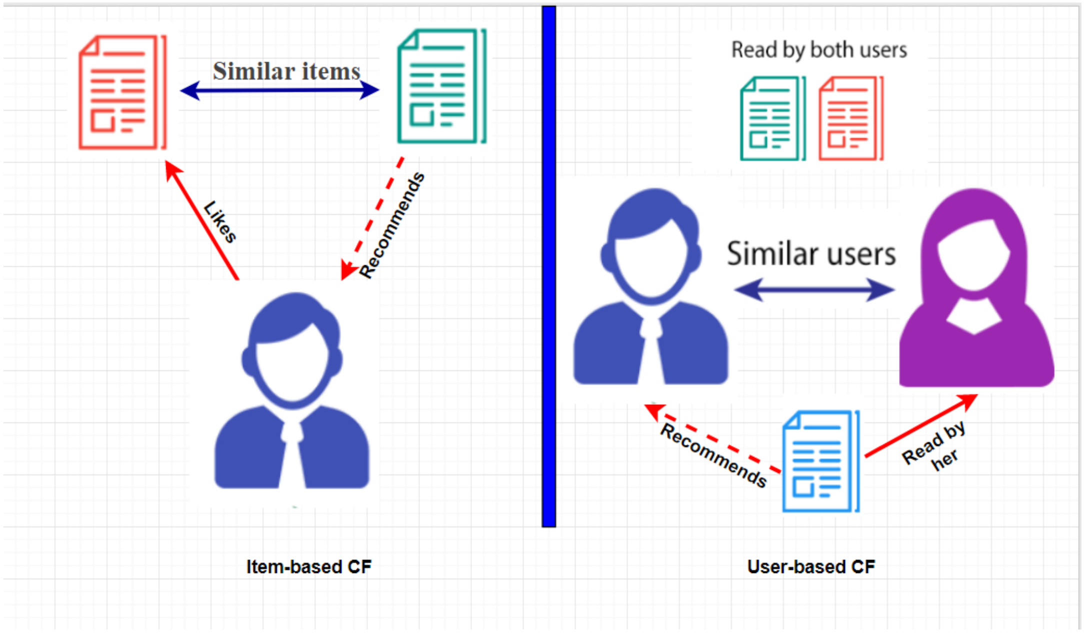


Figure 1.2. *Comparision between user-based and item-based CF*

Neighborhood based CF methods take matrix that contains users as rows and items as columns as input. It is expected that this matrix has many empty cells that means lots of users has not rated lots of items [18]. Neighborhood CF algorithms firstly determine the top k items or users those are most similar to active item or active users according to the method will use to produce prediction for each empty cell in matrix. Matrices that are passed to the CF algorithms can contain interval-based ratings that could be numerical values for example from 1 to 5, binary ratings that mean like or dislike, or unary ratings are deduced from user actions like mouse clicks. Unary ratings are the member of the implicit ratings that are told before. Each empty cell in the matrix is the candidate for the producing prediction process.

1.1.2 Model-based methods

In model-based CF algorithms, a model is developed and this model is used for recommendation process. For example, in the CF scenario that takes a user-item matrix as argument, a model of user ratings is developed and this model is used for predicting rating value phase. Actually, the model of the recommendation process is made believe as the summarized data. In model-based methods, there are two phases, which are training and test phase. While in the training phase, model is created according to the training data; in the test phase, estimation production is processed by using model that is created in the training phase. Model-based CF algorithms take a probabilistic approach and envision the CF process as computing the expected value of a user prediction, given his/her ratings on other items [17] Model creation is made by some machine learning algorithms. Bayesian network, decision trees and clustering are the popular examples of these algorithms. For example, model creation in Bayesian network is formulated as a probabilistic model. The clustering model treats CF as a classification problem [24–26] and works by clustering similar users in same class and estimating the probability that a particular user is in a particular class C, and from there computes the conditional probability of ratings. In the data classification problems, input matrix to the recommender systems has n rows and m columns represent n users and m-1 attributes. One of the columns in the input matrix usually represents the data class. Input matrix is divided into two sets and one of them is used for training another is used for testing. Each cell in the training set is specified including the label of the row. In the test set, all of the attributes are also specified but the cell in which class label is specified is empty and the main goal of the model-based recommender system is specifying this empty cell.

1.2 Challanges of Collaborative Filtering

CF systems are one of the most prevalent algorithms used in recommender systems, but they have some challenges. The most important target of the CF systems is making more accurate recommendation to users. To achieve this, the dataset that will be used in the CF systems should be large in size which means more users rate more items. The larger datasets usually means more accurate recommendations. However, as the dataset size increases, traditional CF systems become insufficient. Recently, big data solutions such as Apache Spark in the recommender systems field which can handle larger datasets are emerged and they should be followed closely. Another problem in the CF systems in the CF is protecting individuals' privacy. For example, to make accurate predictions, two different datasets which are owned by two different data owners may be wanted to merge. However, merging two datasets can pose privacy problems. When two distinct datasets which protect privacy of users individually are merged, they may not protect the privacy of users. In the memory-based CF systems, k-nearest-neighbor algorithm is used widely. The most crucial and also the most difficult step is specifying the k value in this algorithms. Optimal k value in the k-nearest-neighbor algorithm can vary according to the project or dataset. For example, while in one project k=30 may give most accurate predictions, in another project k=50 may be successful in recommendations.

1.3 Problem Definition

The most crucial step in the k-NN algorithms is determining an appropriate k value. If k is chosen to be a small number, the algorithm will be sensitive to irrelevant cases. On the other hand, if k is chosen to be a large number, estimations may be less consistent due to relatively distant neighbors [2]. Thus, the k value of the k-NN algorithm varies with the context (project, subject, etc.).

The purpose of a recommendation system is to provide the best recommendations to users. CF and k-NN algorithms also serve this purpose. However, it is considered that working with the same k for all users or items, may conflict with this purpose.

Therefore, it has been considered that accuracy of estimations may increase using different k values for each user/item in user-based/item-based k-nn. In this paper, it is motivated on showing the effect of dynamic k values during recommendation process. Therefore, it is performed a set of experiments and results investigate that accuracy of a traditional user, or item-based CF systems' accuracy can be improved by utilizing dynamic k values.

1.4 Contribution

The main goal of a recommender systems is providing more accurate recommendations. One of the most popular recommendation method is k-nn because of its simplicity. Although the technologies which are used in the recommender systems field are being changed, the algorithms remains mostly same. To provide more accurate predictions, the algorithms also should be developed. For this purpose, in this thesis k-nn algorithm is tried to be enhanced to provide more accurate recommendations. Different k values for each user or item according to the type of the algorithms which can be user-based or item-based were used in the k-nn algorithm in this thesis. Empirical outcomes show that using different k values for each user or item in the k-nn algorithm improves the prediction quality.

2 RELATED WORK

An automated CF using a neighborhood-based algorithm was introduced by GroupLens [22, 27]. To make personalized recommendations from any dataset or any type of database, The Bellcore Video Recommender [28] and The Ringo Music Recommender [20] reported a technique. Resnick and Varian [29] claimed that for making good recommendations to people, recommend titles that are liked by similar users who have similar interests. Breese et al. [6] compare the accuracy of the various techniques by describing several algorithms for CF such as correlation coefficients, Bayesian methods and vector based similarity calculation. Billsus and Pazzani [31] describe an algorithm that tried to defeat the limitations of CF algorithm techniques. They used dimensionality reduction method by using singular value decomposition (SVD) of a matrix that contains user ratings. By the way, SVD is a common technique which is used for dimensionality reduction aims to improve the performance of CF algorithms [32]. Gupta et al. [34] developed off-line principal component analysis (PCA) and clustering to provide more accurate recommendations by using model-based CF algorithm. Java-based framework that builds CF systems were presented by Fisher et al [33].

Miyahara and Pazzani [39] proposed a new approach for CF based on naive Bayesian classifier(NBC).Chen and George [38] showed a new approach to the problem of estimating missing ratings from the known ratings by using a Bayesian approach. Chen and Jin [35] developed a new CF algorithm based on influence sets by defining a new estimation computation formula. Chen and Cheng [36] proposed a CF methodology for recommendation in case each users' preferences is expressed by multiple ranked lists of items. Goldberg et al. [37] developed an algorithm called Eingentaste that applies Pearson correlation coefficient to a dense subset of the input matrix that contains users' ratings. Popescul et al. [40] proposed a unified framework to merge CF and content-based recommendations.

Since interest in the k-NN-based CF increased in the nineties, researchers tried to effectively use nearest-neighbor algorithms in recommendation processes [12]. Herlocker et al. [13] had a comprehensive study on the neighbor selection for the first time in CF. They concluded that Pearson correlation coefficient is an effective way to compute the similarities between users and employing the most similar k users as neighbors increase the quality of the prediction. In the following years, researchers tried to improve the prediction quality by improving the neighbor selection. Sarwar et al. [17] had a similar study, but they worked on similarities between items instead of the users. They used threshold-based neighbor selection method, which was employed by Kim and Yang [14], to calculate the similarities between the items. Thanks to this threshold-based neighborhood selection method, neighbors are substituted for a user with an unusual rating pattern. Liang et al. [15] proposed calculating the similarities between users from a subset of the items, not from the whole dataset. Koren [16] proposed to specify the nearest neighbors of a user by optimizing a global cost function that improves the accuracy of estimations.

A hybrid CF method that is called personality diagnosis was proposed by Pennock and Horvitz [41]. They compute the probability of the personality types of the users and after this step compute the probability that a person could like a specific item. Su et. al. [11] combined advice from multiple specialists to make effective recommendation and proposed two hybrid CF algorithms those are called sequential mixture CF and joint mixture CF. Datasets that are worked on by CF systems are generally sparse and these hybrid CF approaches are successful while working on sparse data. Lekakos and Giaglis [**?**] developed an approach that clusters consumers according to their behavior and marketing theory.

All of these studies aim to increase the accuracy of the predictions. To achieve this goal, some of them use different neighborhood selection methods while some of them vary in the similarity calculation algorithms. For memory-based CF algorithms, although these studies differ from each other in similarity calculation or neighborhood selection method, ultimately they use the k-nearest-neighbor of the customers to compute the prediction. The k value of the algorithm is determined, and then the same k value is utilized for all users or items. In this thesis, the objective of the proposed method is to increase the accuracy of the predictions by using dynamic k values for each user or item in the system.

3 NEAREST NEIGHBOR-BASED CF ALGORITHMS

Nearest neighbor-based algorithms are based on the fact that similar users have similar patterns of rating behavior and users give similar ratings to similar items [18]. Recommender systems usually work on a user-item matrix that is composed by collected user preferences. This two-dimensional matrix consists of m users and n items which represent matrix's rows and columns. User for whom a prediction value will be estimated for a particular item is called active user (a). To produce a prediction for a for target item q, either user-based CF or item-based CF algorithms are used.

3.1 User-based *k*-Nearest Neighbor Algorithm

User-based CF's first step is calculating correlations between a and other users in the system by utilizing correlation metrics such as Pearson Correlation Coefficient (PCC). The similarity between two users is calculated via such metrics by taking into account only co-rated items. For example, *user1* rates *item1*,*item2* and *item5*. On the other hand, *user2* rates *item2*,*item3* and *item5*. While calculating similarity between *user1* and *user2*, only *item2* and *item5*'s ratings are taken into account. After similarities between users are computed, recommender system selects top k users who are the most similar to a. Finally, a prediction value is estimated for a for q by taking into account only k users selected before. This process is repeated for each item that a prediction value will be estimated for. User-based k-nn algorithms basic scheme is shown in Figure 3.3.

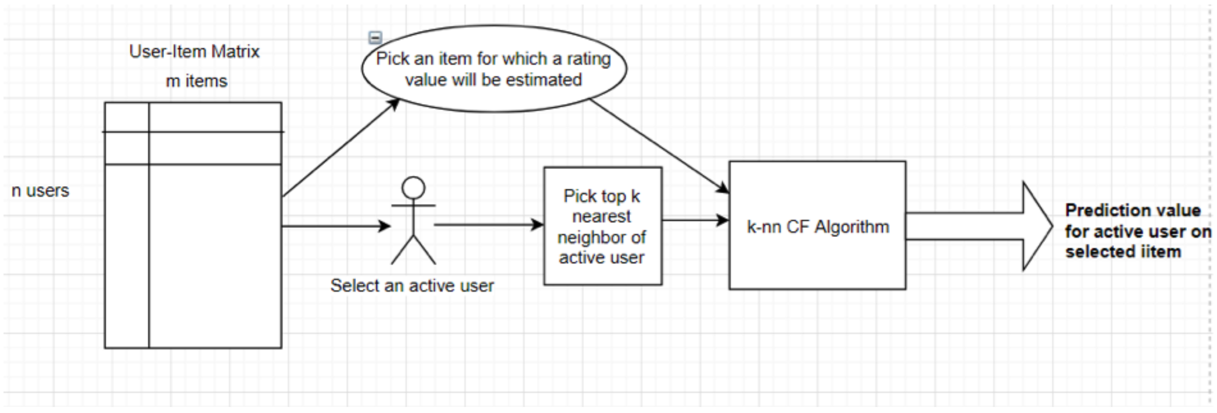

User-Item Matrix
m items
n users
Pick an item for which a rating value will be estimated
Select an active user
Pick top k nearest neighbor of active user
k-nn CF Algorithm
Prediction value for active user on selected iitem

Figure 3.3. *Basic scheme of k-nn algorithm*

3.1.1 Similarity Calculation

CF is based on calculating similarities among users or items. In order to calculate similarities between two users, different methods can be employed. One of these methods is PCC which is widely used in recommender systems. The similarities between a and one of user in the rating matrix can be computed via PCC as in the following equation.

$$w(a,u) = \frac{n * (\sum_{i \in I(a,u)} (r_{a,i} * r_{u,i})) - (\sum_{i \in I(a,u)} (r_{a,i})) * (\sum_{i \in I(a,u)} (r_{u,i}))}{\sqrt{[n * \sum_{i \in I(a,u)} r_{a,i}^2 - (\sum_{i \in I(a,u)} r_{a,i})^2] * [n * \sum_{i \in I(a,u)} r_{u,i}^2 - (\sum_{i \in I(a,u)} r_{u,i})^2]}} \tag{3.1}$$

Here, I represent set of items which are rated both a and user u, $r(a,i)$ and $r(u,i)$ stands for ratings for item i that is an element of I rated by user a and user u, respectively. n is the number of co-rated items between a and user u. The similarity coefficient value calculated by Equation (3.1) must be in the range -1 and +1. The greater value of similarity coefficient ($w(a,u)$) means the higher correlation between user a and u and the less value of similarity coefficient ($w(a,u)$) means the less correlation between user a and u.

3.1.2 Producing Prediction

In user-based CF, in order to produce a prediction for target item q for a, firstly neighbors of user a are determined according to the weights between user a and all other users in the system. Then, these weights are ordered from the highest to the lowest. After this step, subset of the neighbor are taken according to the k value of the algorithm. Finally, recommender system aggregates ratings of users in the neighborhood as in Equation (3.2) .

Here, U represents set of users are neighbors of a ,$\overline{r_a}$ is the mean of rating values of user a, and $w(a,u)$ corresponds to the similarity coefficient between user a and u calculated via Equation (3.1). While producing prediction, it should be taken into account that prediction value must be in the range of minimum and maximum rating value of dataset is worked on. For example, the prediction value is produced 5.1 for rating value ranging from 1 to 5, this value should be rounded to the 5 and also the prediction value should be greater than 1 for this system.

$$p_{a,i} = \overline{r_a} + \frac{\sum_{u \in U}(r_{u,i} - \overline{r_u}) * w(a,u)}{\sum_{u \in U} w(a,u)} \tag{3.2}$$

3.2 Item-based *k*-Nearest Neighbor Algorithm

Contrary to user-based CF, item-based CF's first step is calculating similarities between items, not users. After similarities between items are computed, recommender system determines top k items that are the most similar to the active item q. A prediction value is estimated for a particular item by taking into account only k items which are most similar to this item.

3.2.1 Similarity Calculation

Item-based CF has also some different weighting algorithms, but in this thesis PCC method as in user-based CF to provide a controlled experiment.

$$w(i,j) = \frac{n * (\sum_{u \in U(i,j)}(r_{u,j} * r_{u,i})) - (\sum_{u \in U(i,j)}(r_{u,j})) * (\sum_{u \in U(i,j)}(r_{u,i}))}{\sqrt{[n * \sum_{u \in U(i,j)} r_{u,j}^2 - (\sum_{u \in U(i,j)} r_{u,j})^2] * [n * \sum_{u \in U(i,j)} r_{u,i}^2 - (\sum_{u \in U(i,j)} r_{u,i})^2]}} \tag{3.3}$$

Here, U represents set of users which rated both active item j and item i, $r(u,j)$ and $r(u,i)$ stands for ratings for user u that is an element of U rates both item j and item i, respectively. n is the number of users who rated both j and i. The similarity coefficient value calculated by Equation (3.3) must be in the range -1 and +1. The greater value of similarity coefficient ($w(i,j)$) means the higher correlation between item j and i and the less value of similarity coefficient ($w(i,j)$) means the less correlation between item j and i.

3.2.2 Producing Prediction

In the item-based CF, to produce a prediction value for target item i for user a, firstly neighbors of item i are specified according to the weights between item i and all other items in the system. Then, these weights are ordered from the highest to the lowest. After this step, subset of the neighbors of item i are taken according to the k value of the algorithm. Finally, the weighted average of neighbor items which are rated by the user a is calculated as in Equation (3.4).

$$p_{a,i} = \frac{\sum_{j \in NN}(r_{a,j} * w(i,j))}{\sum_{j \in NN} w(i,j)} \tag{3.4}$$

4 DETERMINING DYNAMIC k VALUE FOR k-nn BASED ALGORITHMS

The major problem in the k-nn algorithm is specifying the k value. In the traditional user-based k-nn algorithm, after similarities are calculated between active user and all other users in the system top k neighbor are selected, and prediction is generated for a particular item. Also, in the traditional item-based k-nn algorithm, after similarities are calculated between active item and all other items in the system top k neighbors of this item are selected, and prediction is generated for a particular item. This procedure is repeated for all items for which a rating value will be estimated. In the k-nn algorithm, the different neighborhood size can affect the quality of the prediction indicated by MAE. For example, if 15 is set as k value lower MAE value is obtained compared to 20 is selected as k value, but if k is set about 30 the MAE value is less than MAE that is obtained when k equals to 15 [10]. In brief, there is not an optimal value of k value for all cases. The optimal k value can vary by project or dataset will be used in the recommender system.

In this thesis, k value was tried to specify distinctly for all users or items in order to get higher accuracy. This means there is not a global value of k. For user-based k-nn, firstly user specific k values are assigned in the training phase of developed method and then use them for estimating prediction in the test phase. After the work was completed for a user-based k-nn algorithm, this work was adapted for an item-based k-nn algorithm by assigning specific k values for each item in the training phase and using them in the test phase. Prediction was also generated a by using the k-nn algorithm with a constant k for all users and items that is 50 to compare the results in terms of accuracy of the predictions.

After calculating weights between active user or item and all other users and items in the system separately, the weights are sorted in descending order, and the first k neighbor is selected to estimate prediction for a specific item. In the traditional k-nn algorithm the k value of the algorithm is set a predefined value. In other words, a k value is selected for the system, and all predictions for all users are estimated with the same k value.

In this thesis, different k values were tried to give for different users in order to give the best estimate. For this purpose, prediction values were generated for all items which were rated by user by using leave-one-out cross-validation method. While generating estimations, 10 different k values were tested which spans from 5 to 50 with interval 5 for each user in the training phase of the developed algorithm. In other words, 10 rating prediction values were produced for an item that was rated by a user with 10 different k values. It was performed for all items those user rated.

Also, this process was repeated for item-based CF. Different k values were tried to give for different items in order to give the best estimate. For this purpose, prediction values were generated for all items which were rated by users by using leave-one-out cross-validation method as in the user-based k-nn algorithm. While generating estimations, 10 different k values were tested which spans from 5 to 50 with interval 5 for each item in the training phase of the developed algorithm. 10 rating prediction values were produced for an item that was rated by a user with 10 different k values. It was performed for all items were rated by a user.

Algorithm 1 Rating Prediction Procedure in the Training Phase (user-based)

```
 1: procedure UserBasedCFPrediction
Require: User rating dataset for $n$ users and $m$ items ( $U_{n\times m}$ )
 2:     for all $users \in U(i \leftarrow 1...n)$ do
 3:         for all $items \in U(j \leftarrow 1...m)$ do
              Calculate weights for each user between other users and specify nearest
              neighbors by sorting weights in descending order
 4:             if $U_{i,j}$ is not nan then
 5:                 $activeUserSimilarities \leftarrow$ calculate weights between active user
                                                      and all other users
 6:                 $sortedUserSimilarities \leftarrow sort(activeUserSimilarities)$
 7:             end if
              For all k values in range 5 to 50 estimate rating value by using k-nn al-
          gorithm
 8:             for all $k \in (5, 10, 15, ...., 50)$ do
 9:                 $p_{k,n,m} \leftarrow p_{a,i}$ (calculate by using Equation 3.2)
10:             end for
11:         end for
12:     end for
13: end procedure
```

Rating prediction values were hold in a three-dimensional matrix for 10 k values, 943 users, and 1,682 items. Rating predictions can be calculated as shown in Procedure 1. Rating estimation matrix has not a number (NaN) values for items which are not rated by user or whose rating prediction cannot be generated. Algorithm 1 was also adapted for item-based CF by calculating similarities between items instead of users, by using Equation 3.4. instead of Equation 3.2.

Algorithm 2 Rating Prediction Procedure in the Training Phase (item-based)

```
1: procedure ItemBasedCFPrediction
Require: User rating dataset for $n$ users and $m$ items ( $U_{n \times m}$ )
2:     for all $items \in U(j \leftarrow 1...m)$ do
3:         for all $users \in U(i \leftarrow 1...n)$ do
           Calculate weights for each item among other users and specify nearest
           neighbors by sorting weights in descending order
4:             if $U_{i,j}$ is not nan then
5:                 $activeItemSimilarities \leftarrow$ calculate weights between active item
                                                      and all other items
6:                 $sortedItemSimilarities \leftarrow sort(activeItemSimilarities)$
7:             end if
           For all k values in range 5 to 50 estimate rating value by using k-nn algorithm
8:             for all $k \in (5, 10, 15, ...., 50)$ do
9:                 $p_{k,n,m} \leftarrow p_{a,i}$ (calculate by using Equation 3.4)
10:            end for
11:        end for
12:    end for
13: end procedure
```

5 EXPERIMENTS

5.1 Dataset and Preprocessing

In order to compare the effects of different k values on users, MovieLens data was used which has three columns; userId, movieId and rating value as the dataset. Since the study was done with MATLAB that is a high-performance software written especially for scientific and numerical calculations and programming, the original MovieLens dataset was converted to MATLAB matrix. While doing the conversion process, NaN value was put which is an abbreviation for not-a-number if the dataset does not contain rating for some userId and movieId pair. In this way, some special MATLAB functions such as nanmean function that calculates the user rating mean by omitting the NaN values were able to be used . The corr function was also used to calculate similarities between users and items by giving 'Pearson' as a parameter.

The dataset consists of 100,000 ratings from 943 users and 1682 movies. Each user has at least 20 ratings. This means only 6% of the available items are rated. The rating values in the dataset are in the range 1-5.

5.2 Metrics

Coverage and accuracy are vital dimensions which assess the quality of a prediction algorithm. Coverage specifies the percentage of items for which a recommendation system can provide prediction [10]. Neighborhood sizes, dataset quality, and size of the dataset can affect the coverage metric dramatically. Coverage was computed coverage by dividing the number of the items for which recommender system could produce prediction to a total number of items for which recommender system intended to produce prediction.

Accuracy of a system is divided into two categories as statistical accuracy metrics and decision-support accuracy metrics [10]. Statistical accuracy metrics evaluate accuracy of a recommender system by comparing the prediction values against the real user ratings. Mean Absolute Error (MAE) has been used previously to measure that kind of recommender systems' accuracy performance by Shardand and Maes [20] and Sarwar [19] et al.

5.3 Experimentation Methodology

Leave-one-out cross-validation method was used during specifying dynamic k values for each user phase. In this way, each user in the dataset is active user once, and rest of users populates the training data [21, 22]. For each element in the MATLAB matrix which has not a NaN value, a prediction is produced by excluding one rating at a time, and predicting the value of rating by using the similarity and prediction calculation formulas given in section 3.1 and 3.2. 5 items were also selected for each user among rated by that user, and produced the prediction for those items using k values each of these user-specific.

5.4 Experimental Results and Discussion

For user-based CF, prediction values were produced for each item that was rated by any user using Algorithm 1 and a matrix having MAE values was obtained after subtracting the prediction values from the real ratings. For example, while first row and the first column of the matrix represent the MAE value of the first user for k=50, second row and the tenth column of the MAE matrix represent the MAE value of the second user for k=5. MAE values of the first ten users in the dataset for varying k values are shown in Table 5.1.

Table 5.1. MAE values of the first ten users in the dataset for varying k values

Users	k=50	k=45	k=40	k=35	k=30	k=25	k=20	k=15	k=10	k=5
User1	1,0349	1,0702	1,1098	1,1336	1,2000	1,1997	1,6657	1,7890	1,5321	1,3688
User2	0,7597	0,7596	0,7928	0,8497	0,9027	0,9176	0,8055	0,8556	1,0106	0,9179
User3	1,0496	0,9801	0,9250	0,9572	0,9719	1,0189	1,1142	1,1079	1,1242	1,2876
User4	0,8617	0,8866	0,9144	0,8719	0,8275	0,7895	0,7860	0,7730	0,9155	1,0270
User5	1,0978	1,1285	1,1663	1,1143	1,1032	1,1282	1,2259	1,3125	1,3356	1,4249
User6	0,9388	0,9372	0,9734	0,9585	0,9033	0,8708	0,9046	0,8407	1,0558	1,1355
User7	1,0466	1,0409	1,0675	1,2055	1,2451	1,2666	1,0447	0,8382	0,8407	0,8189
User8	0,9467	0,9594	0,9576	1,0051	1,0101	1,0473	0,9184	0,9517	1,1437	1,1353
User9	0,9184	0,9292	0,8643	0,8991	0,8854	0,9415	0,9719	1,0195	1,1485	1,0996
User10	0,6708	0,5638	0,5564	0,6054	0,6448	0,6404	0,6073	0,6110	0,6489	0,7187

After MAE values were obtained for users for varying k values, the best k values were achieved to find for each user in the dataset. For example, according to the

MAE values in Table 5.1 the best k values for each user are formed as shown in Table 5.2.

Table 5.2. Dynamic k value look-up table for users

Users	The best k value for user	MAE for specifying k value
User1	50	1,0349
User2	45	0,7596
User3	40	0,9250
User4	15	0,7730
User5	50	1,0978
User6	15	0,8407
User7	5	0,8189
User8	20	0,9184
User9	40	0,8643
User10	40	0,5638

For item-based CF, the MAE matrix which has 1,682 rows which is the number of the movies in the user-item matrix and 10 columns by subtracting the prediction values from the real ratings was obtained . MAE values of the first ten items in the dataset for varying k values are shown in Table 5.3.

Table 5.3. MAE values of the first ten items in the dataset for varying k values

Items	k**=50**	k**=45**	k**=40**	k**=35**	k**=30**	k**=25**	k**=20**	k**=15**	k**=10**	k**=5**
Item1	1.3478	1.2666	1.2069	1.2271	1.2415	1.2609	1.2975	1.2389	1.2976	1.4861
Item2	0.9906	0.9879	1.0108	0.9933	1.0368	0.9977	1.0054	0.9103	1.2130	1.3333
Item3	1.1624	1.0891	1.0894	1.0963	1.0771	1.0501	0.9876	0.9538	1.0000	1.0714
Item4	0.9863	1.0108	1.0992	1.0735	1.0643	1.1279	1.1646	1.1515	1.1914	1.2857
Item5	0.8732	0.8645	0.8214	0.7752	0.8744	0.8990	0.9108	0.8631	0.9420	1.1000
Item6	1.1008	1.1874	1.1099	1.1085	1.1137	1.0426	0.9461	1.0000	0.9100	0.6133
Item7	1.2427	1.3031	1.3184	1.3024	1.3497	1.2971	1.1925	1.2647	1.2800	1.7143
Item8	1.4337	1.4368	1.4114	1.4132	1.3337	1.3344	1.2773	1.3705	1.3824	1.2917
Item9	1.0134	0.9871	0.9712	0.9805	0.9801	0.9111	0.9477	1.1250	1.0810	1.0781
Item10	0.8465	0.8934	1.0103	1.0301	1.0611	1.0356	1.0900	1.1377	0.9537	1.0000

After the MAE values were obtained for items for varying k values, the best k

values were achieved to find for each item in the dataset. The best k values are shown in Table 5.4 for the first ten items.

Table 5.4. Dynamic k value look-up table for items

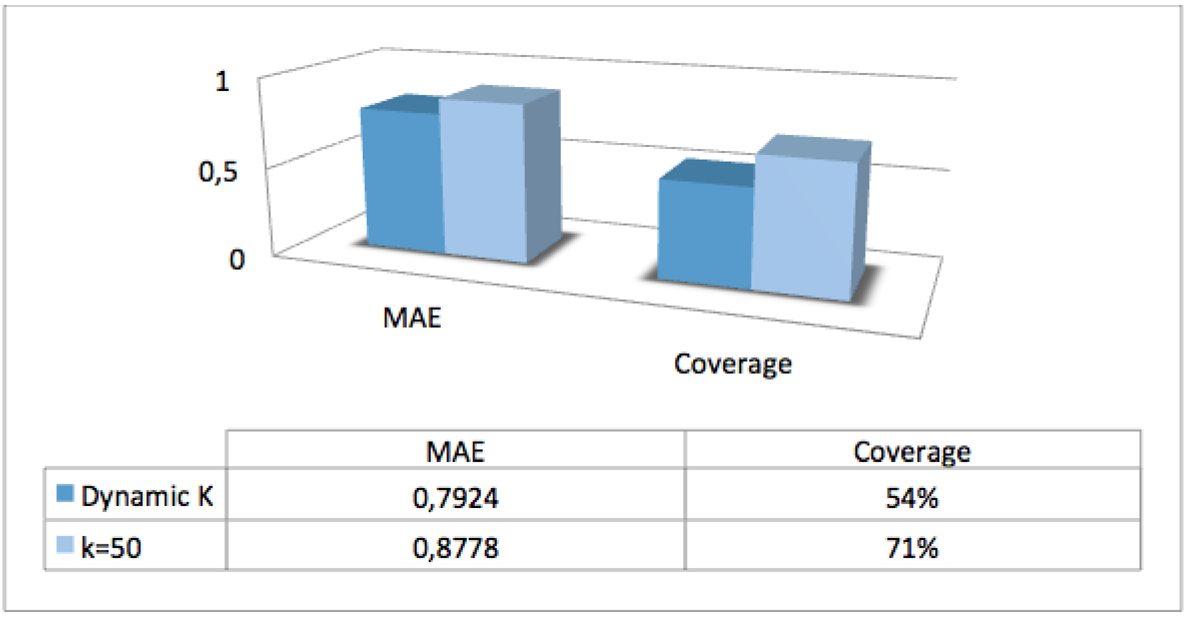
Items	The best k value for item	MAE for specifying k value
Item1	40	1.2069
Item2	15	0.9103
Item3	15	0.9538
Item4	50	0.9863
Item5	35	0.7752
Item6	5	0.6133
Item7	20	1.1925
Item8	20	1.2773
Item9	25	0.9111
Item10	50	0.8465

In order to observe the effects of using dynamic k value for each user in prediction in user-based CF, five items for each user to be estimated were randomly selected and the algorithm was run for 100 times with both dynamic k value for each user and a constant k value that is 50. The results showed that using dynamic k value in k-nn algorithm for rating prediction dramatically increases the accuracy of the estimations. The higher accuracy means, the lower MAE. On the other hand, using dynamic k value in k-nn decreases the number of estimations produced. The procedure which was used in the test phase in this study is shown in Procedure 3.

Algorithm 3 Rating Prediction Procedure in the Test Phase (user-based)

```
1: procedure UserBasedCFTestPrediction
Require: $U_{n\times 5}$ test set contains 5 test item for each user, dynamic $k$ value for each
    user ($k$ look-up table)
2:     for all $users \in U(j \leftarrow 1...m)$ do
3:         $k \leftarrow$ read the best $k$ value of the user from look-up table
4:         for all $items \in U(i \leftarrow 1...n)$ do
               Calculate weights for active user between users and specify $k$ nearest neigh-
    bors by sorting weights in descending order
5:             $p_{n,m} \leftarrow p_{a,i}$ (calculate by using Equation 2)
6:         end for
7:     end for
           Calculate MAE value and the coverage of the system by substracting the
    prediction matrix by the real rating values of the items given by the users.
8:     $mae \leftarrow P_{n,m} - R_{n,m}$
9: end procedure
```

The MAE results and the coverage that is a measure of the percentage of generating estimation on given test set is given comparatively in Figure 5.4.



	MAE	Coverage
🟦 Dynamic K	0,7924	54%
🟦 k=50	0,8778	71%

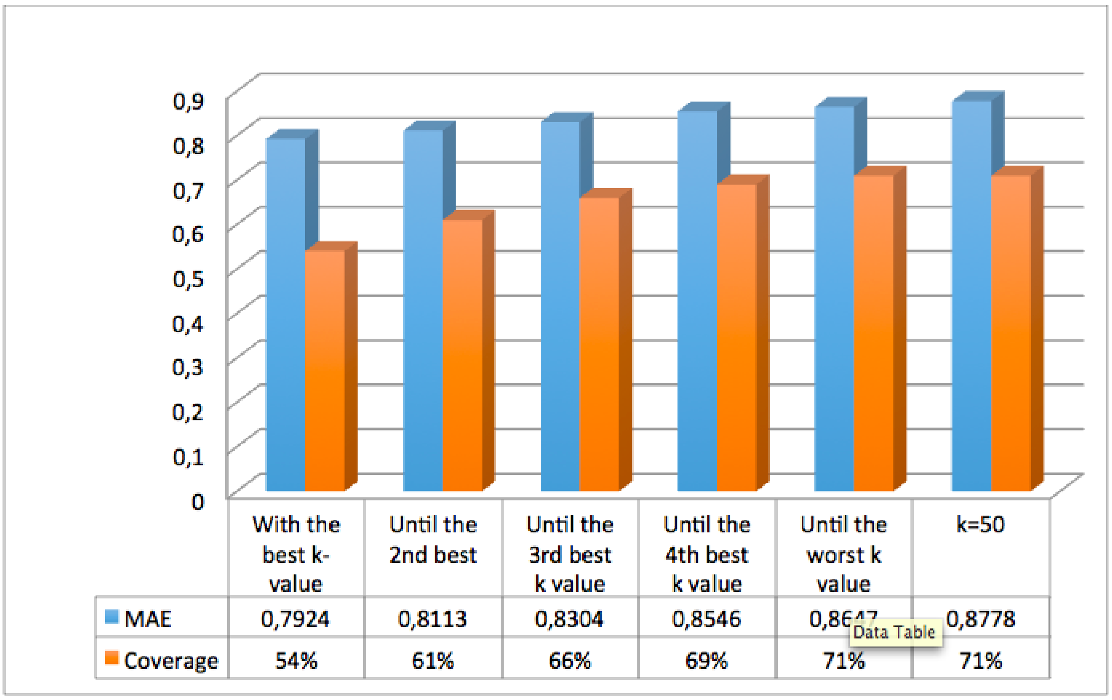
Figure 5.4. *MAE and Coverage results for dynamic k values and k=50*

While the accuracy of the prediction is crucial for some systems or web sites, the coverage can be so important for others. After the results were obtained and got there is a trade-off between accuracy and coverage, it is decided to try extending the k look-up table for the user and putting the values for the second best k value for the user and then the third best k value for the user and so on. By using the MAE values in Table 5.1, the extended dynamic k look-up table composed as shown in Table 5.5 for the first ten users.

Table 5.5. k values in ordered will be used in k-nn for each user

Users	**1st best k**	**2nd best k**	**3rd**	**4th**	**5th**	**6th**	**7th**	**8th**	**9th**	**10th**
User1	50	45	40	35	25	30	5	10	20	15
User2	45	50	40	20	35	15	30	25	5	10
User3	40	35	30	45	25	50	15	20	10	5
User4	15	20	25	30	50	35	45	40	10	5
User5	50	30	35	25	45	40	20	15	10	5
User6	15	25	30	20	45	50	35	40	10	5
User7	5	15	10	45	20	50	40	35	30	25
User8	20	50	15	40	45	35	30	25	5	10
User9	40	30	35	50	45	25	20	15	5	10
User10	40	45	35	20	15	25	30	10	50	5

For the systems where the coverage is crucial, a was tried to design new algorithm. According to this algorithm, if the system can not produce estimation for an item for a specific user by using dynamic k value for this user, it will retry to produce prediction by using the second best k value for this user. If it can not produce again, it will retry to produce by using third and so on. As more k values were tried to generate a prediction, the percentage of the prediction generation increased but the accuracy of the estimation decreased as shown in Figure 5.5 .



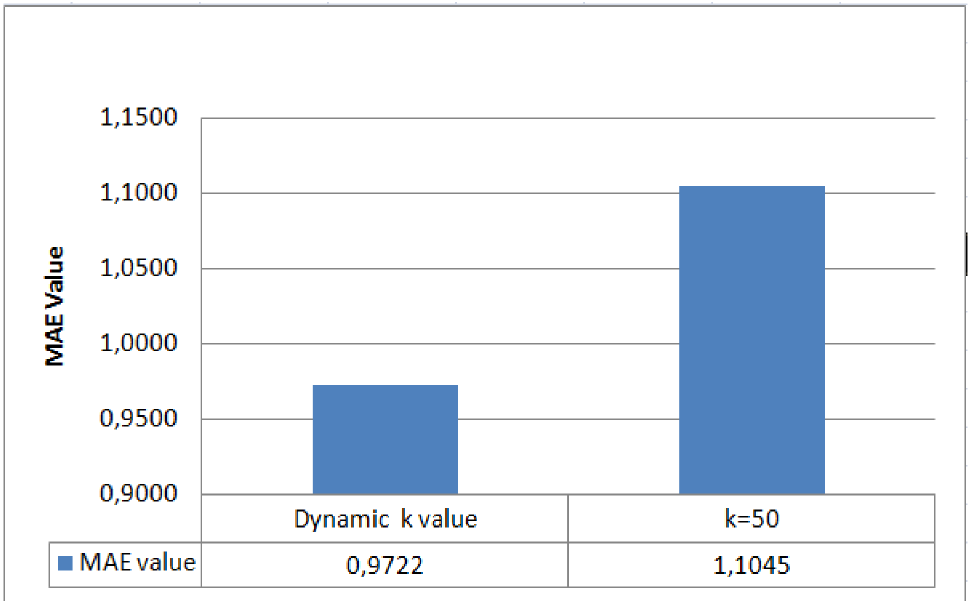
	With the best k-value	Until the 2nd best	Until the 3rd best k value	Until the 4th best k value	Until the worst k value	k=50
🟦 MAE	0,7924	0,8113	0,8304	0,8546	0,8647	0,8778
🟧 Coverage	54%	61%	66%	69%	71%	71%

Figure 5.5. *MAE and Coverage results according to the users' top-n k values*

Dynamic k value was also tried to use in the item-based CF. Like user-based CF, k look-up table was also constituted for each item in the dataset, and these k values were used while estimating prediction value for a specific item. In order to observe the effects of the k values in item-based k-nn algorithm, k-nn algorithm was employed with both dynamic k value and a constant k value that is 50. While producing prediction, Algorithm 4 was used. Five items for each user in the dataset were selected randomly again and it is tried to produce a prediction value. Using dynamic k value also served the purpose in item-based CF as shown in Figure 5.6.

Algorithm 4 Rating Prediction Procedure in the Test Phase (item-based)

```
1: procedure ItemBasedCFTestPrediction
Require: $U_{n\times 5}$ test set contains 5 test item for each user, dynamic $k$ value for each
    item ($k$ look-up table)
2:     for all $users \in U(j \leftarrow 1...m)$ do
3:         for all $items \in U(i \leftarrow 1...n)$ do
4:             $k \leftarrow$ read the best $k$ value of the item from look-up table
           Calculate weights for active item between items and specify $k$ nearest neigh-
    bors by sorting weights in descending order
5:             $p_{n,m} \leftarrow p_{a,i}$ (calculate by using Equation 4)
6:         end for
7:     end for
       Calculate MAE value and the coverage of the system by substracting the
    prediction matrix by the real rating values of the items given by the users.
8:     $mae \leftarrow P_{n,m} - R_{n,m}$
9: end procedure
```



	Dynamic k value	k=50
■ MAE value	0,9722	1,1045

Figure 5.6. *MAE values for dynamic k values and k=50 in item-based k-nn*

6 CONCLUSION AND FUTURE WORK

CF and k-nearest neighbor algorithms are frequently used in classification algorithms as well as for rating prediction and recommender systems. Especially online booking, movie and shopping sites where there are many users trying to offer the best recommendations to their users. Today's trend in recommender systems is to be able to find the point of the shot while finding suggestions towards the user. For this purpose, new technologies such as big data emerged in recommender systems. Although some recommendations can be instantly made to the users by bringing together different data sets using the used technologies to analyze, the basic stones used by the recommender systems are algorithms like relatively same used in the past such as CF, Support Vector Machines, etc. As a result, besides the development of the technologies, the improvement of the basically used algorithms will increase the success rate of the recommendations presented to the user.

The improvement of the conventional algorithms in CF, k-nearest neighbor based method will make it easier to reach the goal of making a shot at the suggestions presented to the user. For this reason, in this study, it is decided to give different k values for each user and item in the k-nn algorithm to get better estimations, and it was succeeded in this. The best k-values for each user and item in the dataset were tried to determine by trying many k values during the training phase of the algorithm, and then these k values were used in the k-nn algorithm to predict rating users gave to the items. The results showed that using user or item specific k values instead of using static k value in the k-nn algorithm, dramatically increases the success of prediction estimations. For some users or items, a higher k value allows us to achieve lower MAE values, while for some users or items smaller k values increase the estimation success rate.

So far, the best k value for each user and item were tried to determine by trying different k values on that user, and the k values which were determined during the test phase were used. Especially in the systems which have many users and many items, it will be a long process to try k values for each new user and item to determine the value of k to be used in recommendations for specific user, so it will not be possible to use k-nn algorithm instantly for new coming users by using the users' best k value especially while using user-based k-nn algorithm. It is planned to be able to determine the best k values online by looking at specific attributes of the new user in the system, without having to go through any training phase after the new user provides a certain number of rating values.

References

[1] White, T. (2015). "*Hadoop the definitive guide*", O'reilly US.

[2] Ricci, F., Rokach L., Shapira B. (2011). "*Recommender Systems Handbook*", Springer US.

[3] Adomavicius G, Tuzhilin A. (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.

[4] Bilge A., Yargıç A. (2017). "*Improving accuracy of muti-criteria collaborative filtering by normalizing user ratings*", Anadolu University Journal of Science and Technology a- applied science and engineering, 18(1), 225-237, 2017.

[5] Adomavicius G, Manouselis N, Kwon Y. (2011). "*Multi-criteria recommender systems*", Recommender Systems Handbook, 769-803.

[6] Breese, J.S., Heckerman, D., Kadie, C. (1998). "*Empirical analysis of predictive algorithms for collaborative filtering*", Proc. of the 14th Annual Conf. on Uncertainty in Artificial Intelligence, 43-52.

[7] Delgado, J., Ishii, N. (1999). "*Memory-based weighted majority prediction for recommender systems*", Proc. of the ACM SIGIR'99 Workshop on Recommender Systems.

[8] Koren Y. (2008). "*Factorization meets the neighborhood: a multifaceted collaborative filtering model*", KDD'08: Proceeding of the 14th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, 426-434.

[9] Keller JM, Gray MR, Givens JA. (1985). "*FA fuzzy k-nearest neighbor algorithm*", IEEE transactions on systems, man, and cybernetics, 15(4), 426-434.

[10] Herlocker JL, Konstan JA, Borchers A, Riedl J. (1999). "*An algorithmic framework for performing collaborative filtering.*", Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 230-237.

[11] Su X, Khoshgoftaar TM. (2009). "*A survey of collaborative filtering techniques*", Advances in Artificial Intelligence, 2009:4.

[12] Kaleli C. (2014). "*An entropy-based neighbor selection approach for collaborative filtering*", Knowledge-based Systems, , 273-280.

[13] Herlocker J. , Konstan J.A., Riedl J. (2002). "*An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms* ", Inform. Retrieval 5, 273-280.

[14] Kim T.-H. , Yang S.-B. (2007). "*An effective threshold-based neighbor selection in collaborative filtering*", Proceedings of the 29th European conference on IR Research. ECIR'07, 712-715.

[15] Liang Z. , Bo X. , Jun G. . (2009). "*An approach of selecting right neighbors for collaborative filtering*", Proceedings of the Innovative Computing, Information, and Control (ICICIC), 2009 Fourth International Conference, 1057-1060.

[16] Koren Y. (2010). "*Factor in the neighbors: scalable and accurate collaborative filtering*", ACM Trans. Knowl. Discovery Data 4 ,4(1) 1-24.

[17] Sarwar B., Karypis G. , Konstan J. , Riedl J.. (2001). "*Item-based collaborative filtering recommendation algorithms*", Proceedings of the 10th International Conference on World Wide Web, 285-295.

[18] Aggarwal C.C. (2016). *Recommender Systems: The Textbook*",Springer

[19] Sarwar B., Borchers A.,Herlocker J., Miler B., Riedl J.. (1998). "*Using filtering agents in the grouplens research collaborative filtering systems*", Proceedings of 1998 Conference on Computer Supported Collaborative Work.

[20] Shardanand U., Maes P.. (1995). "*Social information filtering*", Proceedings of ACH CHI'95 Conference on Human Factors in Computing Systems, 210-217.

[21] Jannach D., Karakaya Z., Gedikli F.. (2012). "*Accuracy improvements for multi-criteria recommender systems*", Proceedings of the 13th ACM Conference on Electronic Commerce, 674-689

[22] Resnick P., Iacovou N., Suchak M, Bergstrom P., Riedl J.. (1994). "*GroupLens: an open architecure for collaborative filtering of netnews*", Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, 175-186.

[23] Grcar M.. (2004). "*User profiling: collaborative filtering*", Proceedings of the SIKDD 2004 at Multiconference IS, Ljubljana, Slovenia.

[24] Basu, C., Hirsh, H., Cohen, W.. (1998). " *Recommendation as Classification: Using Social and Content-based Information in Recommendation*", Recommender System Workshop '98,11-15.

[25] Ungar, L. H., Foster, D. P.. (1998). " *Clustering Methods for Collaborative Filtering*", Workshop on Recommender Systems at the 15th National Conference on Artificial Intelligence.

[26] Schafer, J. B., Konstan, J.,Riedl, J.. (1999). " *Recommender Systems in E-Commerce*", Proceedings of ACM E-Commerce 1999 .

[27] Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., Rield, J. T.. (1997). " *GroupLens: Applying collaborative filtering to Usenet news*", Communications of the ACM, 40(3), 77-87.

[28] Hill, W., Stead, L., Rosenstein, M., Furnas, G.. (1995). " *Recommendation and evaluating choices in a virtual community of use*", Proceedings of the ACM CHI'95 Conference on Human Factors in Computing Systems, 194-201.

[29] Resnick, P. and Varian, H. R.. (1997). " *Recommender systems*", Communications of the ACM, 40(3), 56-58.

[30] Billsus, D., Pazzani, M. J.. (1998). " *Learning collaborative information filters*", Proceedings of the 15^{th} International Conference on Machine Learning, 46-54.

[31] Billsus, D., Pazzani, M. J.. (1998). " *Learning collaborative information filters*", Proceedings of the 15^{th} International Conference on Machine Learning, 46-54.

[32] Sarwar, B., Karypis, G., Konstan, J., Riedl, J.. (2000). " *Application of dimensionality reduction in recommender system: A case study*", Proceedings of the ACM WebKDD 2000 Web Mining for E-commerce Workshop, 682-693.

[33] Fisher, D. , Hidrum, K, Hong, J., Newman, M., Thomas, M., Vuduc, R.. (2000). " *SWAMI: Framework for collaborative filtering algorithm development and evaluation*", Proceedings of the 23^{rd} Annual International SCM SIGIR Conference on Research and Development in Information Retrieval, 366-368.

[34] Gupta D., Digiovanni M., Narita H., Goldberg K.. (1999). " *Jester 2.0: A new linear-time collaborative filtering algorithm applied to jokes*", Proceedings of the 22^{nd} Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 291-292.

[35] Chen, J., Yin, J., (2007). "*A collaborative filtering recommendation algorithm based on influence sets*", Ruan Jian Xue Bao/Journal of Software, 18, 1685-1694.

[36] Chen, J., Cheng, L., (2007). "*A novel collaborative filtering approach for recommending ranked items*", Expert Systems with Applications, 34(4), 2396-2405.

[37] Goldberg, K., Roeder, T., Gupta, D., Perkins, C.. (2001). "*Eigenstaste: A constant time collaborative filtering algorithm*", Information Retrieval, 4(2), 133-151.

[38] Chen, Y., George, E. I.. (1999). "*A Bayesian model for collaborative filtering*", Proceedings of the 7^{th} International Workshop on Artificial Intelligence and Statistics.

[39] Miyahara, K., Pazzani, M. J.. (2002). "*Improvement of collaborative filtering with the simple Bayesian classifier*", Transactions of Information Processing Society of Japan, 43(11), 3429-3437.

[40] Popescul, A., Ungar, L. H., Pennock, D. M., and Lawrence, S.. (2001). "*Probabilistic models for unified collaborative and content-based recommendation in sparse environments*", Proceedings of the 17^{th} Conference on Uncertainty in Artificial Intelligence, 437-444.

[41] Pennock, D. M., Horvitz, E., Lawrence, S., and Giles, C. L.. (2000). "*Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach*", Proceedings of the 16^{th} Conference on Uncertainty in Artificial Intelligence, 473-480.

[42] Lekakos, G., Giaglis, G. M.. (2007). "*A hybrid approach for improving predictive accuracy of collaborative filtering algorithms*", Electronic Commerce Research, 17, 5-40.

[43] Hassanat, A B., Abbadi, M A., Altarawneh G A.. (2014). "*Solving the Problem of K Parameter in the Knn Classifier Using an Ensemble Learning Approach*", International Journal of Computer Science and Information Security, 12(8), 33-39.