DEVELOPING A CONTEXT-AWARE LOCATION RECOMMENDER SYSTEM FOR LOCATION-BASED SOCIAL NETWORKS

AYSUN BOZANTA

BOĞAZİÇİ UNIVERSITY

2018

DEVELOPING A CONTEXT-AWARE LOCATION RECOMMENDER SYSTEM FOR LOCATION-BASED SOCIAL NETWORKS

Thesis submitted to the

Institute for Graduate Studies in Social Sciences

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Management Information Systems

by

Aysun Bozanta

Boğaziçi University

2018

Developing a Context-Aware Location Recommender System

for Location-Based Social Networks

The thesis of Aysun Bozanta

has been approved by:

Prof. Birgül Kutlu Bayraktar (Thesis Advisor)	[illegible]
Prof. Meltem S. Özturan	[illegible]
Assoc. Prof. Hande Bahar Türker	[illegible]
Prof. Alp Kut (External Member)	[illegible]
Assist. Prof. Nazım Ziya Perdahçı (External Member)	[illegible]

May 2018

DECLARATION OF ORIGINALITY

I, Aysun Bozanta, certify that

- I am the sole author of this thesis and that I have fully acknowledged and documented in my thesis all sources of ideas and words, including digital resources, which have been produced or published by another person or institution;
- this thesis contains no material that has been submitted or accepted for a degree or diploma in any other educational institution;
- this is a true copy of the thesis approved by my advisor and thesis committee at Boğaziçi University, including final revisions required by them.

Signature....[illegible]....

Date ....May 29, 2018....

ABSTRACT

Developing a Context-Aware Location Recommender System

For Location-Based Social Networks

People think about where to go many times throughout their lives. Although it is a very rapid and repetitive decision, generally it is hard to choose suitable places from endless number of options for some specific circumstances. Recommender systems are supposed to help to deal with those issues and take appropriate actions. However, the location decision is different from other decisions like what to listen, buy, or read from various aspects. The popularity of location-based social networks has prompted researchers to study recommendation systems for location. Traditional recommendation algorithms have been used for location recommendation. When used separately, each venue recommendation system algorithm has drawbacks. Another issue is that the context information is not commonly used in venue recommendation systems. Time, distance and weather conditions have more impact on decisions about where to go than all other decisions. Another point that should not be disregarded is that the effects of those contextual variables differ from user to user. This study proposes a hybrid recommendation model that combines contextual information, user- and item-based collaborative filtering and content-based filtering. For this purpose, user visit histories, venue-related information and contextual information related to individual user visits were collected from Twitter, Foursquare, and Weather Underground. The proposed hybrid system is evaluated using both offline experiments and a user study. This proposed system shows better results than baseline approaches.

ÖZET

Konum Tabanlı Sosyal Ağlar için

Bağlam Duyarlı Konum Tavsiye Sistemi Geliştirme

İnsanlar hayatları boyunca gidecekleri yerleri defalarca düşünmüşlerdir. Bu çok hızlı verilebilecek ve sürekli tekrar eden bir karar olmasına rağmen sayısız seçenek arasından o anki şartlar için uygun yerleri seçmek zordur. Tavsiye sistemleri bu tip sorunları çözmekte ve uygun şekilde harekete geçmekte yardımcı olurlar. Fakat konum seçimi kararı ne dinleneceği, ne alınacağı ya da ne okunacağı kararlarından birçok açıdan farklıdır. Konum tabanlı sosyal ağların popülerliği araştırmacıları konum tavsiye sistemleri üzerine çalışmaya yönlendirmiştir. Geleneksel tavsiye algoritmaları konum tavsiyesi için de kullanılmıştır. Tek başlarına kullanıldıklarında konum tavsiye sistemlerinin her biri farklı dezavantajlara sahiptir. Diğer bir sorun da bağlamsal değişkenlerin konum tavsiye sistemlerinde yaygın bir şekilde kullanılmamasıdır. Zaman, uzaklık, hava durumu gibi değişkenler nereye gideceğimiz konusundaki kararımıza diğer kararlarımızdan daha çok etki ederler. Göz ardı edilemeyecek diğer bir konu da bağlamsal faktörlerin kişiden kişiye değişiklik göstermeleridir. Bu çalışma bağlamsal bilgiyi, kullanıcı tabanlı işbirlikçi filtreleme, öğe tabanlı işbirlikçi filtreleme ve içerik tabanlı filtreleme yöntemlerini birleştirerek hibrit bir model önerir. Bu amaçla, kullanıcı ziyaret geçmişleri, konumla ilgili bilgiler ve bağlamsal bilgiler Twitter ve Foursquare uygulamaları ve Weather Underground web sitesinden toplanmıştır. Sunulan hibrit sistem hem çevrimdışı deneyler hem de kullanıcı çalışması yöntemleri ile değerlendirilmiştir. Bu önerilen sistem temel yaklaşımlardan daha iyi sonuçlar vermiştir.

CURRICULUM VITAE

NAME: Aysun Bozanta

DEGREES AWARDED

PhD in Management Information Systems, 2018, Boğaziçi University

MA in Management Information Systems, 2013, Boğaziçi University

BA in Mathematical Engineering, 2010, Işık University

BA in Management Information Systems (Double Major), 2010, Işık University

AREAS OF SPECIAL INTEREST

Recommendation Systems, Analysis and Design of Information Systems, Data Mining Techniques, Quantitative Methods, 3D Environments, Serious Games

PROFESSIONAL EXPERIENCE

Research Assistant, Department of Management Information Systems, Boğaziçi University, 2011 – present

Student Assistant, Department of Mathematical Engineering, Işık University, 2009 – 2010

Project Intern, Koç Finans, July 2009 – September 2009

AWARDS AND HONORS

TÜBİTAK Ph.D. Scholarship, 2013-2017

TÜBİTAK M.A. Scholarship, 2010-2012

First Ranked, Management Information Systems (GPA: 3.81), Işık University, 2010

Second Ranked, Mathematical Engineering (GPA: 3.79), Işık University, 2010

Highest Honors List, Işık University, 2010

Highest Honors List, Işık University, 2009

Highest Honors List, Işık University, 2008

Highest Honors List, Işık University, 2007

ÖSS (Student Selection Examination - SSE) Success Scholarship, Full Scholarship for Mathematical Engineering, Işık University, 2005

PUBLICATIONS

International Journal Articles

Kutlu, B., Bozanta, A., Coskun, M. (2017), “Usage Factors of Location-Based Social Applications: The Case of Foursquare”, International Journal of Web-Based Communities (Scopus). (Accepted for Publication)

Bozanta, A., Mardikyan, S. (2017), "The Effects of Social Media Use on Collaborative Learning: A Case of Turkey", TOJDE, Vol: 18, No: 1, pp. 96-110 (Scopus).

Bozanta, A., Kutlu, B. (2017), "Current State and Future Trends in Location Recommender Systems", International Journal of Information Technology and Computer Science (IJITCS), Vol: 9, No: 6, pp. 1-8.

Bozanta, A., Coskun, M., Kutlu, B., Ozturan, M. (2017), "Relationship Between Stock Market Indices and Google Trends", The Online Journal of Science and Technology (TOJSAT), Vol: 7, No: 4, pp. 168-172.

Bozanta, A., Kutlu, B., Nowlan, N., Shirmohammadi, S. (2016), "Effects of Serious Games on Perceived Team Cohesiveness in a Multi-User Virtual Environment", Computers in Human Behaviour, Vol: 59, No: 2016, pp. 380-388 (SSCI).

Ozturan, M., Bozanta, A., Basarir-Ozel, B., Akar, E., Coskun, M. (2015), "A Roadmap for an Integrated University Information System Based on Connectivity Issues: Case of Turkey", The International Journal of Management Science and Information Technology (IJMSIT), No: 17, pp. 7-29.

Bozanta, A., Nasir, V.A. (2014), "Usage of Agent-Based Modeling and Simulation in Marketing", Journal of Advanced Management Science, Vol: 2, No: 3, pp. 240-245.

Kutlu, B., Bozanta, A., Ates, E., Erdogan, S., Gokay, O., Kan, N. (2014), "Project Management Software Selection Using Analytic Hierarchy Process Method", International Journal of Applied Science and Technology, Vol: 4, No: 6, pp. 113-119.

Book Chapters

Coskun, A., Bozanta, A. (2018), "Industry 4.0 and Key Technologies: A Review in the book Industry 4.0 Theme", The Peter Lang Publishing Group, Istanbul (accepted for publication).

Kutlu, B.,, Bozanta, A.,, Coskun, M.,, Disci, D. (2016), "Factors Affecting Foursquare Usage: Case of Turkey", ISRC Selected Papers Series-Management Information Systems Congress 2014, (Editors) Meltem Ozturan, Birgul Kutlu", ss. 81-93, Bogazici University Press House, Istanbul.

International Conference Proceedings

Bozanta, A., Kutlu, B. (2018), "Do Twitter Phenomena Check-In Popular Venues on Foursquare Too?", International Multidisciplinary Conference on Education, Arts, Law, Business & Politics (MEALP-18), Amsterdam, Netherlands, 3-4 February 2018.

Coskun, A., Bozanta, A. (2017), "Industry 4.0 and Key Technologies: A Review", 4th International Management Information Systems Conference (IMISC-2017), Istanbul, Turkey, 17-20 October.

Bozanta, A., Coskun, M., Kutlu, B., Ozturan, M. (2016), "Relationship Between Stock Market Indices and Google Trends", International Science and Technology Conference, pp. 1023-1027, Viyana, Avusturya, 13-15 July 2016.

Bozanta, A., Nasir, V. A. (2014), "Usage of Agent-based Modelling & Simulation in Marketing", International Conference on Advances and Management Sciences (ICAMS-2014), Barcelona, Spain, Feb. 21-22.

Kutlu B., Bozanta, A., Nowlan, N. (2013), "Multi-User Virtual Environments and Serious Games for Team Building in Organizations", Proceedings of the Sixth International Conference on E-Learning in the Workplace (ICELW 2013), New York, NY, USA, June 12-14.

Bozanta, A., Kutlu B., Nowlan, N., Shirmohammadi, S. (2012), "Multi User Virtual Environments and Serious Games for Team Building", Procedia Computer Science, 4th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES’12), Vol: 15, pp. 301-302.

National Conference Proceedings

Kutlu, B., Bozanta, A., Coşkun, M. (2015), "Konum Tabanlı Sosyal Ağlar İçin Tavsiye Sistemlerinde Yeni Eğilimler: Bir Değişkenler ve Algoritmalar İncelemesi", 2. Ulusal Yönetim Bilişim Sistemleri Kongresi, ss. 857-868, Erzurum, Türkiye, 8-10 Ekim 2015.

Kutlu, B., Bozanta, A., Coşkun, M., Dişçi, D. (2014), "Foursquare Kullanımına Etki Eden Faktörler: Türkiye Örneği", Yönetim Bilişim Sistemleri Kongresi, İstanbul, Türkiye, 16-17 Ekim 2014.

ACKNOWLEDGEMENTS

After a long journey of five years, today is the day: writing this acknowledgements is the finishing touch on my dissertation. It has been a period of intense learning for me, not only scientifically, but also personally. Preparing this dissertation has had a big influence on me. I would like to present the people who have supported and helped me so much during this period.

I would like to express my deep gratitude to my thesis advisor Prof. Birgül Kutlu Bayraktar for her patient guidance, enthusiastic encouragement, and useful critiques. I always feel her emotional support, which makes me stronger. This thesis could not be written without her.

I would also like to extend my thanks to the thesis committee – Assoc. Prof. Hande Bahar Türker, Prof. Alp Kut, Prof. Meltem Seba Özturan, and Assist. Prof. Nazım Ziya Perdahçı for accepting to be part of the committee, their constructive criticism, and valuable comments.

I wish to acknowledge the help provided by Osman Yücel, Ph.D. and Mustafa Coşkun, Ph.D.. I am particularly grateful for their support given by all my friends, colleagues and the precious professors of the department of Management Information Systems of Bogazici University.

I would like to thank the Scientific and Technological Research Council of Turkey for providing the funding by 2211-E Direct National Doctoral Scholarship Programme (2211-E Doğrudan Yurt İçi Doktora Burs Programı) during my entire doctoral period. I would also like to thank The Bogazici University Scientific Research Projects Fund (Project #11463), not only for providing the funding which allowed me to undertake this research, but also for giving me the opportunity to attend the conference and meet so many valuable people.

I would like to thank my sister Fulya Taşçeviren, her husband Mehmet Taşçeviren, and my little nephew Işık Taşçeviren for their unconditional support, encouragement and love.

I am grateful to my mother Nuray Bozanta and my father Ersin Bozanta for unconditionally believing in and trusting on me during my entire life. They are always there for me, which is something that cannot be provided by anyone but the parents.

Most importantly, I wish to thank my loving and supportive fiancé, and my future family, Tuğrul Hakyemez to be my light at my darkest, to be the shoulder that I cry, and to endure even the most unbearable times of mine.

With love and hope.

To my mother, father, and sister,

For their endless love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

“Every adventure requires a first step.”

Lewis Carroll

The increasing use of location-related technologies enables the development of location based-services. Therefore, location-based social networks (LBSNs) which have become the host of new possibilities for user interaction are emerged. Those systems, which facilitate users to share their visits and explore other locations, have accumulated huge amount of data about users with the intensive usage in time. Location Recommendation Systems (LRSs) have been developed by discovering embedded information from LBSN data to provide location suggestion for users.

Recommender systems, which suggest items to the users assuming that users will like the recommended items, have been very attractive both in businesses and in the research community. Recommender systems have been utilized in a variety of areas including movies, music, news, books, places, research articles, search queries, social tags, and products in general. Recommender systems use mainly three algorithm types: collaborative filtering systems, content-based filtering systems, and hybrids (Adomavicius & Tuzhilin, 2005). Contextual information is thought to increase the performance of recommendation systems (Adomavicius & Tuzhilin, 2005), but most recommendation engines fail to consider it. The algorithms that have been used for other types of items, have been also utilized to generate location recommendation.

However, each algorithm has its own drawbacks. For instance, collaborative filtering algorithms have cold start, scalability, and sparsity issues, and they lack content-related information. With content-based filtering, an item and its contents need to be machine recognizable and must contain sufficient information, but because this algorithm considers only venue-related characteristics, it may result in a focus that is too narrow. Contextual information (weather, time, date, etc.) is more important in travel and tourism domains. For that reason, context-aware recommender systems would be more beneficial when used for location recommendation rather than other types of recommendation (product, movie, music, etc.). Some of the location recommendation engines fail to consider contextual information. Although, there are some studies including contextual information in their systems, there is no common standard for using them.

Personalization is another issue that should be considered in the development of recommendation systems. The effect of each variable used in recommendation may vary among different users. For instance, two people may like the same place but in different contextual circumstances. Therefore, it is important to consider the changing effects of variables for different users.

Therefore, by considering above issues, this thesis presents a contextually personalized hybrid location recommender system that combines user-based collaborative filtering, item-based collaborative filtering, content-based filtering, and contextual information. The proposed hybrid system will reduce the number of the drawbacks of each approach when they are used separately. For this purpose, users' check-in history, properties of related location (distance, category, popularity, and price) and contextual data (weather, season, date, and time of visits) are collected from different sources (Twitter, Foursquare, and Weather Underground). To our best knowledge, this is the first study that combines distance, category, popularity, and price in one content-based filtering algorithm. Weather conditions, season, date, and time of each visit will be utilized in the algorithm. The results are evaluated with both offline experiments and a user study. Scientific value of this study can be listed as below:

- Three different types of variables (user-related, venue-related (content) and contextual) that have not been used together in existing recommender systems will be used in one algorithm to develop a novel recommender system.
- Artificial neural network algorithm is applied to determine the weight of each algorithm (user-based collaborative filtering, item-based collaborative filtering, content-based filtering and context-aware recommendation) that are used when developing the hybrid recommendation system.
- Threshold values determining the user's liking toward a venue are determined separately for each user.
- More contextually personalized recommendation is generated by determining which contextual circumstances are more appropriate for specific user-venue pair.
- By calculating the similarity of users according to their various preferences (category, popularity, and price), data sparsity problem was alleviated.
- Over-specialization was lessened by considering the preferences of users from many different aspects and not just getting stuck in only the venue characteristics.
- Cold start problem was partially solved. Even if a new user rates only one venue, the algorithm understands the user preferences from the characteristics of the venue (category, price, popularity). Moreover, the algorithm figures out the

contextual circumstances that user prefers that venue. Therefore, by looking these characteristics, the algorithm may recommend a venue to a new user. The results of the thesis are beneficial for both the developers of the recommender systems and the business owners.

The remaining parts of the thesis are organized as follows: in Chapter 2, literature review will be described; in Chapter 3, methodology will be explained; in Chapter 4, analyses and their results will be presented; and in Chapter 5 the thesis will be concluded.

CHAPTER 2

LITERATURE REVIEW

“Don’t worry Alice,

Wonderland is better when you are completely lost.”

Lewis Carroll

2.1 Recommender systems

Every day, people have encountered tens of decisions. Some decisions are very rapid and repetitive decisions like which clothes to wear, which road to drive, which restaurant to go, what to eat, which song to listen, which movie to watch, which book to read. On the other hand, some of the decisions like whom to marry, where to live, which university to study, and similar are more rare decisions but need to be thought extensively. Although, there are different types of decisions, people spend most of their times even for simple ones and these decisions make their life more difficult.

Previously, people have made these decisions by depending on the suggestions that they gather from their friends, advertisements, discussions, newspapers, and magazines. However, recommendations that are made via these ways are limited and can be biased. Therefore, computer supported technologies can generate suggestions from a wide variety of choices since they can utilize from not only the acquaintance but also other people.

Recommender systems suggest items to the users assuming that users will like the recommended items. Recommender systems have been utilized in a variety of areas including movies (Szomszor, et al., 2007; Ono, Kurokawa, Motomura, & Asoh, 2007; Lekakos & Caravelas, 2008; Choi, Ko, & Han, 2012; Diao, et al., 2014), music (Eck, Lamere, Bertin-Mahieux, & Green, 2008; Van den Oord, Dieleman, & Schrauwen, 2013; Oramas, Ostuni, Noia, Serra, & Sciascio, 2017), news (Liu, Dolan, & Pedersen, 2010; Li, Wang, Chen, & Lin, 2010; Li L. , Zheng, Yang, & Li, 2014; Chen, Meng, Xu, & Lukasiewicz, 2017), books (Chen, Li, & Huang, 2005; Tewari, Kumar, & Barman, 2014), location (Ye, Yin, & Lee, Location recommendation for location-based social networks, 2010; Gao, Tang, Hu, & Liu, 2013; Wang, Terrovitis, & Mamoulis, Location recommendation in location-based social networks using user check-in data, 2013; Zhang, Chow, & Li, 2015), and products (Linden, Smith, & York, 2003; Liu & Shih, 2004; Park & Chang, 2009) in general.

Recommender systems can either predict for the items that have not been seen by a user, or rank most suitable items for a user (Ricci, Rokach, & Shapira, 2011). In both forms, the aim is to suggest the most appropriate items to the users.

Although the roots of the recommender systems depend on the other research areas, the recommender systems have emerged as an independent area in the mid 1990's (Adomavicius & Tuzhilin, 2005).

The first recommender system - Tapestry (Goldberg, Nichols, Oki, & Terry, 1992), which was developed at Xerox Palo Alto Research Center, aimed to enable users to subscribe only to the mail lists that are interest to them. Tapestry was built to support collaborative filtering, which was firstly used as a new term in that article, besides content based filtering (Goldberg, Nichols, Oki, & Terry, 1992). GroupLens (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994) is a second well-known system for collaborative filtering of Internet news. It aimed to help people finding articles they will like in the huge stream of available articles (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994).

After these initial examples, recommender systems began to grow at a great pace. In addition to interest from research community, industry and businesses have also shown great interest to recommendation systems. Amazon.com and Netflix are very well-known examples of businesses that have been using recommendation systems more than a decade to recommend products to their customers (Ekstrand, Riedl, & Konstan, 2011). Recently, recommendation systems have been in diverse areas.

Recommender systems may be examined in different aspects. In this chapter; recommendation techniques, evaluation methods for recommendation systems, and location recommendation systems will be described.

2.2 Techniques of recommendation systems

Recommender systems are usually classified into the following categories, based on how recommendations are made: content-based filtering systems, collaborative filtering systems, and hybrids (Adomavicius & Tuzhilin, 2005). The classification of the recommender systems according to the techniques that are used for recommendation is presented in Figure 1 (Isinkaye, Folajimi, & Ojokoh, 2015). Contextual information is thought to increase the performance of recommendation systems (Adomavicius & Tuzhilin, 2005). Therefore, contextual information has also begun to be embedded into recommendation systems.

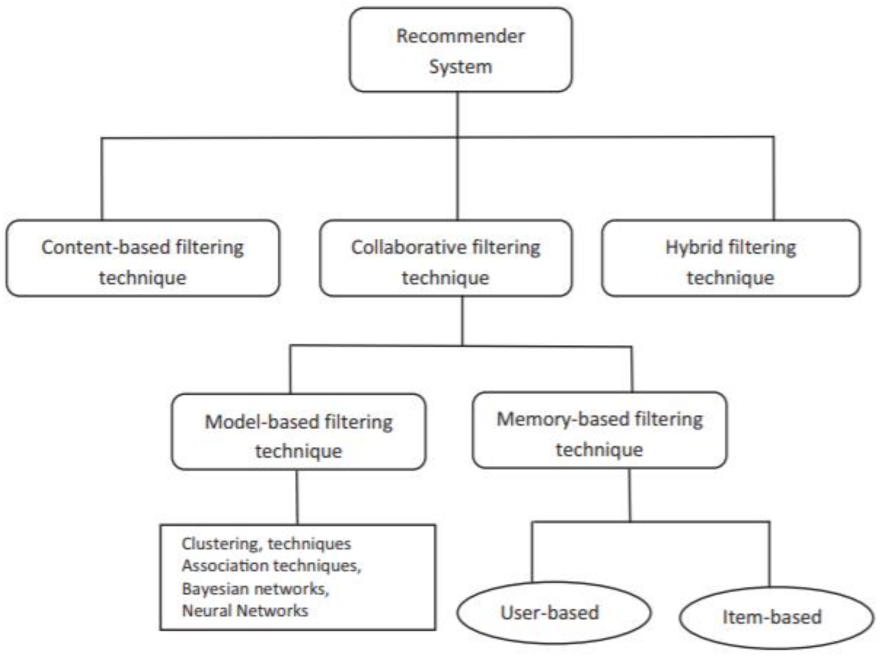
Recommender System
Content-based filtering technique
Collaborative filtering technique
Hybrid filtering technique
Model-based filtering technique
Memory-based filtering technique
Clustering, techniques
Association techniques,
Bayesian networks,
Neural Networks
User-based
Item-based

Figure 1. Recommendation techniques

(Isinkaye, Folajimi, & Ojokoh, 2015)

2.2.1 Content-based filtering

In content-based filtering, similar items according to the user's previous preferences are recommended. In order to make content-based recommendation, system tries to figure out the characteristics of the item that a user gave high ratings before. Then, the system recommends items having high degree of similarity with the user profile (Adomavicius & Tuzhilin, 2005). Content-based filtering uses the assumption that items with similar objective features will be rated similarly (Schafer, Frankowski, Herlocker, & Sen, 2007).

In general, content-based filtering systems need content analyzer which contains proper techniques to record the properties of an item, user profile learner producing and recording user profiles, and filtering component having several strategies for comparing the user profile with the item characteristics (Lops, De Gemmis, & Semeraro, 2011).

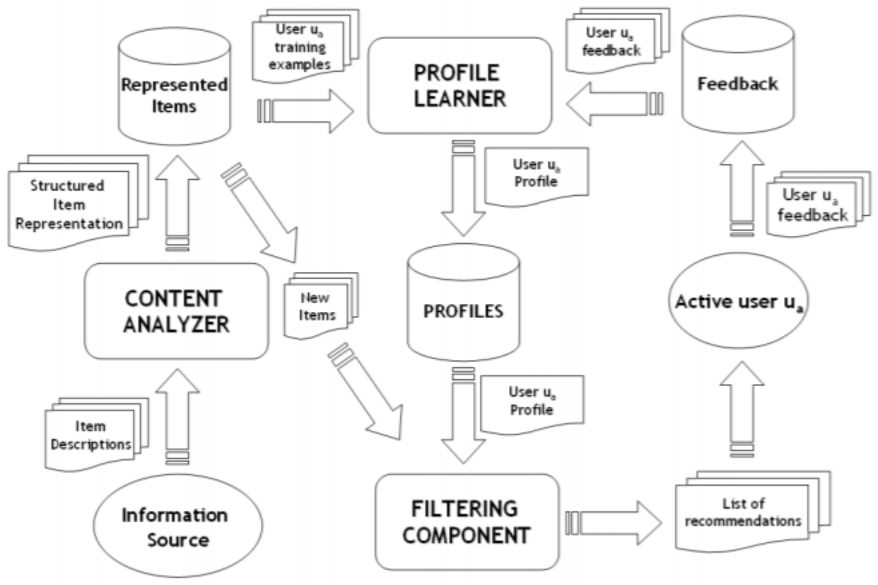

Figure 2. High level architecture of a content-based recommender

(Lops, De Gemmis, & Semeraro, 2011)

Content analyzer tries to record items with their attributes. Content information can appear as structured or unstructured (Pazzani & Billsus, 2007). If it is unstructured, then it requires some preprocessing steps. Therefore, original form of the content is transformed to the desired structured representative form by feature extraction techniques (Lops, De Gemmis, & Semeraro, 2011). For instance web pages or news should be preprocessed and their characteristics should be recorded in a proper format. Content analyzer provides input to the profile learner and filtering component as Figure 2 implies (Lops, De Gemmis, & Semeraro, 2011).

Profile learner collects data about the user preferences (Lops, De Gemmis, & Semeraro, 2011). After each feedback from a user, user profile is updated according to the liked and disliked items by user (Lops, De Gemmis, & Semeraro, 2011).

Filtering component matches the most similar items with the targeted user profile by comparing the attributes of the items and the user preferences (Lops, De Gemmis, & Semeraro, 2011).

There are some advantages of content-based filtering systems over other techniques. User independence, transparency, and the ability to recommend new item are the advantages of content-based filtering systems (Lops, De Gemmis, & Semeraro, 2011):

- In content-based filtering, only the ratings of the target user and the characteristics of the items are adequate to make the recommendation. The ratings of other users are not necessary for this recommendation technique. Therefore, it is user independent.
- Content-based filtering is a very transparent technique compared to the others. Recommendation process solely depends on matching the items having the most similar characteristics with the target user profile (Lops, De Gemmis, & Semeraro, 2011).
- Addition of a new item into the database and recommending this new item to the users are not a problem for content-based filtering as long as the characteristics of the item are determinable. There is no first rater problem here because if item is not yet rated by any user, it can be recommended (Lops, De Gemmis, & Semeraro, 2011).

Besides its advantages, content-based filtering systems have some drawbacks. Limited content analysis, over-specialization, and new user problem are the disadvantages of content-based filtering systems (Adomavicius & Tuzhilin, 2005; Lops, De Gemmis, & Semeraro, 2011):

- Content-based techniques are totally dependent on having the content information of items. If content is not detectable, then recommendation will not be generated. On the other hand, although the content is sometimes available, domain knowledge is often needed (Adomavicius & Tuzhilin, 2005; Lops, De Gemmis, & Semeraro, 2011). For instance, in order to recommend poems, examining the word frequency is not enough. In addition, even for the items having more structured content information such as restaurant, books, or movies, recommendation should be supported by the opinions of other users (Pazzani & Billsus, 2007).
- Since content-based systems generate recommendations having the most similar characteristics with the previous preferences of the users, the recommendation of content-based systems have a limited degree of novelty (Adomavicius & Tuzhilin, 2005). This disadvantage is also called serendipity problem (Lops, De Gemmis, & Semeraro, 2011). Although the system finds and recommends the best matched items with the user preferences, it cannot produce new recommendations, which can be liked by the user and also different from the previous likes of the user (Pazzani & Billsus, 2007; Lops, De Gemmis, & Semeraro, 2011).
- If a target user is new and s/he has not enough ratings, content-based system cannot generate a recommendation for this user (Adomavicius & Tuzhilin, 2005). This problem is called a new user problem (Pazzani & Billsus, 2007;

Lops, De Gemmis, & Semeraro, 2011). System should collect sufficient amount of ratings by the user to produce accurate recommendations.

2.2.2 Collaborative filtering

Collaborative filtering systems produce recommendations according to the preferences of other users who have similar preferences with the target user (Adomavicius & Tuzhilin, 2005; Schafer, Frankowski, Herlocker, & Sen, 2007; Su & Khoshgoftaar, 2009). The term collaborative filtering was used firstly in the study of Tapestry, which is the one of the first recommender systems (Goldberg, Nichols, Oki, & Terry, 1992). After that, various collaborative filtering systems have been developed until now.

The main assumption behind the collaborative filtering is; if user A and B behave similarly on their previous preferences, then they will continue to behave similarly in the future (Su & Khoshgoftaar, 2009). The general working process of collaborative filtering can be explained as follows: the rating of user “u” to the item “i” is predicted based on the ratings of other users who have similar preferences with the user “u” (Adomavicius & Tuzhilin, 2005).

Rating is association between the user and the item and shows the interest of the user to the item (Schafer, Frankowski, Herlocker, & Sen, 2007). Rating can occur in the system explicitly or implicitly (Breese, Heckerman, & Kadie, 1998). For instance, if a user gives a numerical rating to a movie in a system, then it will be explicit rating. However, if a user visits the page of a product, or visits a restaurant many times, this type of information will be implicit rating. In addition, ratings can be also categorized in different aspects; scalar ratings, binary ratings, or unary ratings (Schafer, Frankowski, Herlocker, & Sen, 2007). Scalar ratings are numerical ratings changing in a determined range. Binary ratings show whether a user likes or dislikes an item. Unary ratings show that a user observed, or purchased the item.

There are two general classes of collaborative filtering: Memory-based collaborative filtering and model-based collaborative filtering (see Figure 1) (Breese, Heckerman, & Kadie, 1998; Isinkaye, Folajimi, & Ojokoh, 2015).

In memory-based collaborative filtering, all ratings of a user in the dataset or sample of them are multiplied with different weights and used to predict rating of a target user (Breese, Heckerman, & Kadie, 1998). Those weights can be distance, correlation, or similarity between each user and target user (Breese, Heckerman, & Kadie, 1998). Previously, the weights are calculated for the relation between users which is called user-based collaborative filtering (see Figure 3). However, item-based collaborative filtering (see Figure 3), which calculates the similarity between items, is suggested as a new method for recommendation system (Sarwar, Karypis, Konstan, & Riedl, 2001).

Model-based collaborative filtering, on the other hand, generally uses data mining techniques such as artificial neural networks (Knoch, Chapko, Emrich, Werth, & Loos, 2012), naive Bayesian modeling (Gupta & Singh, 2013; Subramaniyaswamy, Vijayakumar, Logesh, & Indragandhi, 2015), association rule mining (Saraee, Khan, & Yamaner, 2005), and singular-value decomposition (SVD) (Saraee, Khan, & Yamaner, 2005). There have been also statistical (Raskutti, Beitz, & Ward, 1997) and probabilistic models (Zukerman, Albrecht, & Nicholson, 1999) for recommendation systems in the literature.

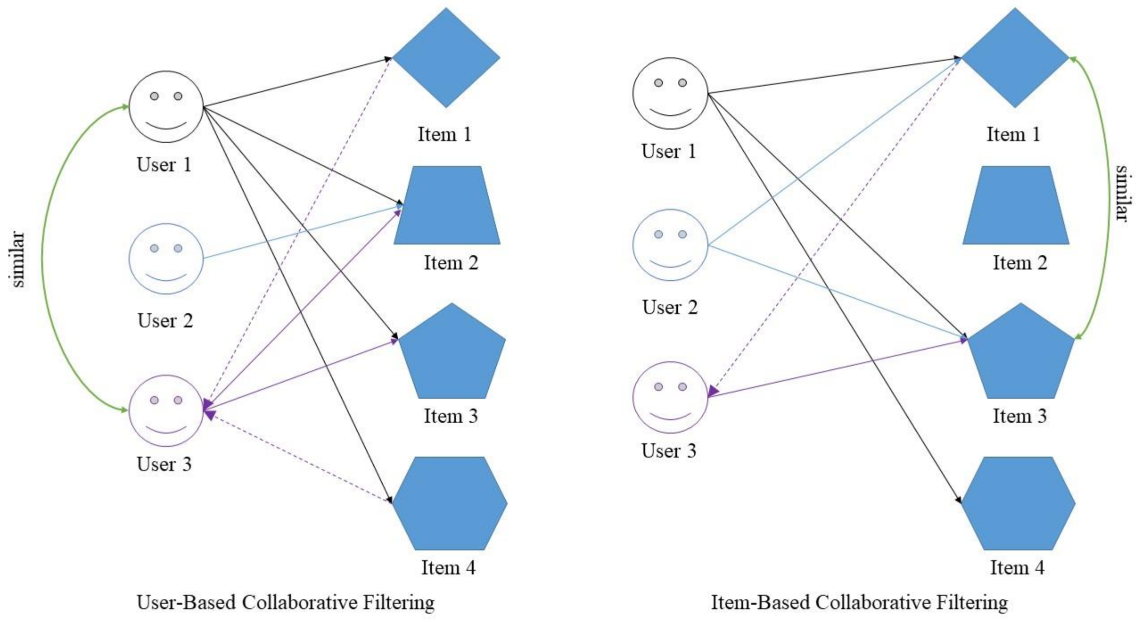

Figure 3. User-based collaborative filtering vs. item-based collaborative filtering

A key advantage of the collaborative filtering is that it does not depend on machine recognizable content. Therefore it is capable of accurately recommending complex items such as movies without requiring a domain knowledge about items (Hamid, Naser, Hasan, & Mahmud, 2014).

However, collaborative filtering approaches often suffer from three problems: cold start, scalability, and sparsity (Herlocker, Konstan, Terveen , & Riedl, 2004; Adomavicius & Tuzhilin, 2005; Schafer, Frankowski, Herlocker, & Sen, 2007; Su & Khoshgoftaar, 2009):

- Cold Start: Collaborative filtering systems often require a large amount of existing data of a user and an item in order to make accurate recommendations. This problem may occur in two types: New user and new item. This new user problem is the same with the problem of content-based filtering. System should learn the preferences of a target user in order to make accurate recommendations. On the other hand, if a new item is added to the system, until this item is rated by a substantial amount of users, the recommender system cannot recommend it.

- Sparsity: In order to make accurate recommendations with collaborative filtering algorithms, a target user should rate a substantial number of items until the system learns the user, and other users should also rate a substantial number of items until the system finds similar users. In addition, in order to recommend an item, that item should be rated by a substantial number of users. Since, the databases of the businesses using recommendation systems contain excessive number of users and items, the most active users will only have rated a small subset of the overall database. Thus, even the most popular items have very few ratings. Therefore, data sparsity problem, which is somehow relevant to the new user and new item, occurs.
- Scalability: When numbers of existing users and items grow tremendously, collaborative filtering algorithms will suffer from scalability problems. Thus, in order to generate recommendations, huge amount of computational resources are needed (Su & Khoshgoftaar, 2009).

2.2.3 Context-aware recommender systems

Contexts represent a set of factors that surround user-item pairs and affect the rating of the user to the item accordingly (Adomavicius & Tuzhilin, 2015). A context-independent recommendation system may lose predictive power if potentially useful information from multiple contexts is ignored (Adomavicius & Tuzhilin, 2015).

Contextual variables may be classified in three categories according to the usage of them in the recommendation systems: fully observable, partially observable, and unobservable (Adomavicius & Tuzhilin, 2015). When contextual factors are known explicitly, then they are called fully observable contextual factors. For example, if we know that a user watches a movie with his girlfriend on Saturday evening, then we would know all the contextual factors explicitly. On the other hand, if we know all the surrounding conditions but we cannot reach some of the information, then these will be partially observable contextual factors. If no information about contextual factors is explicitly available to a recommender system, and recommendations are made by utilizing only the latent knowledge of the context in an implicit manner, then this is an example of an unobservable contextual factor.

Contextual variables can also be classified as static and dynamic according to their change over time (Adomavicius & Tuzhilin, 2015). If contextual factors don't change over time, then they are called as static contextual factors. On the other hand, if they change over time, then they are called dynamic contextual factors. For instance, new categories can be added to the same contextual factor progressively.

There are three ways to use contextual variables in recommender systems; contextual pre-filtering, contextual post-filtering, and contextual modelling (Adomavicius & Tuzhilin, 2015):

- Contextual Pre-Filtering (see Figure 4a): In contextual pre-filtering, result set is filtered according to the certain context before the recommender system calculates the predictions (Verbert, Duval, Lindstaedt, & Gillet, 2010; Adomavicius & Tuzhilin, 2015). For instance if a user will go on a holiday on summer, the recommender system first filters the holiday venues which are preferred in summer, then calculates the venue with the highest rank and shows it to the user.

- Contextual post-filtering (see Figure 4b): In contextual post-filtering, all items are included in the prediction process. However, items that are relevant for the given context are recommended as a result set (Verbert, Duval, Lindstaedt, & Gillet, 2010; Adomavicius & Tuzhilin, 2015). In connection with the previous example, for the summer holiday, all venues are included in the prediction process, after that, they are filtered according to whether they are suitable for summer. This filtering can be done in two ways: Venues, which are not preferred in summer, can be directly extracted from the result set or their ranks can be adjusted according to their context relevance (Verbert, Duval, Lindstaedt, & Gillet, 2010; Adomavicius & Tuzhilin, 2015).
- Contextual modeling (see Figure 4c): In contextual modeling, contextual variables are directly included in the modeling of the recommender system and used in the rating prediction process (Verbert, Duval, Lindstaedt, & Gillet, 2010; Adomavicius & Tuzhilin, 2015).

The decision of which technique will be used for a recommendation system depends on the contextual variables that are used for recommendation. The effect of the techniques may change according to the different variables.

Most of the contextual studies in the literature are conceptual, so it is crucial to develop contextual algorithms (Adomavicius & Tuzhilin, 2015). In addition to this, discovering valid contextual variables for different recommendation algorithms and implementing them is very important for future studies (Adomavicius & Tuzhilin, 2015).

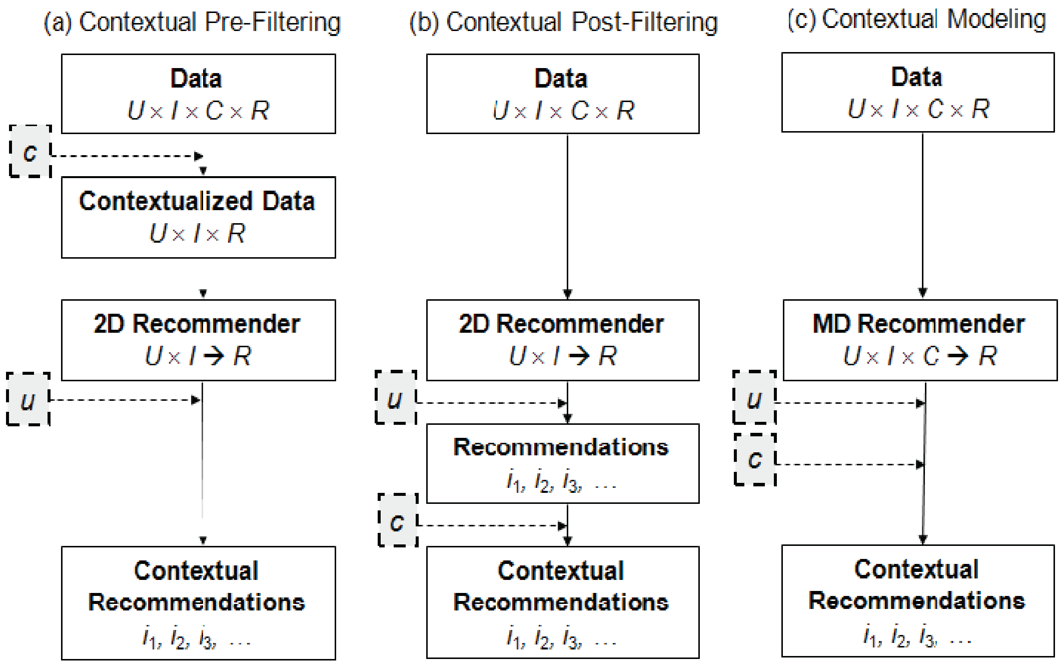

Figure 4. Pre-filtering, post-filtering and contextual-model approaches (Adomavicius & Tuzhilin, 2015)

2.2.4 Hybrid recommender systems

Hybrid approaches combine at least two existing approaches to minimize or eliminate the drawbacks of those approaches when they are used on their own (Burke, 2002; Adomavicius & Tuzhilin, 2005; Bobadilla, Ortega, Hernando, & Gutiérrez, 2013).

Seven types of hybridization techniques are mentioned in the literature: weighted, switching, mixed, feature combination, cascade, feature augmentation, and meta-level (Burke, 2002; Burke, 2007). The advantages and disadvantages of the hybridization techniques are summarized in Table 1.

In weighted hybridization, results from each recommendation technique are included in the final rating (Burke, 2002; Burke, 2007). For instance, the easiest way to combine these results is taking the average of them. The weights of the techniques may be adjusted according to the prediction power of them, or feedback from the users. The main advantage of weighted hybridization is that all of the system's results are brought together on the recommendation process in a straightforward way (Burke, 2002; Burke, 2007). In addition, it is easy to apply adjustments on weights after getting some feedback from the user. On the other hand, the main disadvantage of the weighted hybridization is using the same weights for all users (Burke, 2002; Burke, 2007).

In switching hybridization, the result of one of the techniques is chosen according to a pre-determined criterion (Burke, 2002; Burke, 2007). The advantage of switching is being sensitive to the strengths and weaknesses of its constituent recommenders (Burke, 2002; Burke, 2007). However, as a disadvantage, switching introduces additional complexity into the recommendation process since the switching criteria must be determined (Burke, 2002; Burke, 2007).

In mixed hybridization, ranked lists from different techniques are combined (Burke, 2002; Burke, 2007). It is practical to make a large number of recommendations simultaneously (Burke, 2002; Burke, 2007). However, it is hard to evaluate recommendation systems, which were generated by mixed hybridization with retrospective data (Burke, 2002; Burke, 2007).

In feature combination hybridization technique, features from different sources are used in one recommendation algorithm (Burke, 2002; Burke, 2007). It is different from other techniques because in this technique not the results from other techniques are combined but the features from different data sources are brought together to generate a recommendation (Burke, 2002; Burke, 2007). Since it uses different features from various data sources, its main advantage is that the system is less sensitive to the number of users, who have rated an item. (Burke, 2002). On the other hand, it gives the system information about the inherent similarity of items that are otherwise opaque (Burke, 2002).

Cascade hybridization is a staggered technique, in which one recommendation technique is employed first to rank the candidates then a second technique improves the recommendation from the candidate set (Burke, 2002; Burke, 2007). The idea behind the cascade hybridization is creating a hierarchical hybrid by preventing the use of a second ordered, weak algorithm in the first phase (Burke, 2002; Burke, 2007). It is utilized only to refine the candidate set which is produced by the strongest recommendation algorithm. However, it may cause imprecise recommendations resulting from using insufficient numbers of techniques (Burke, 2002; Burke, 2007).

A feature augmentation hybrid produces a new feature for each item by using one algorithm, then this feature is used as an input for another algorithm (Burke, 2002; Burke, 2007). Feature augmentation can be preferred to feature combination because it is more flexible (Burke, 2002; Burke, 2007). In addition, feature combination adds new dimensionalities to the data and it becomes harder to process (Burke, 2002; Burke, 2007). However, it is still not clear how to apply a feature augmentation technique for any two recommendation algorithms (Burke, 2002; Burke, 2007).

The last hybridization technique is meta-level. In meta-level hybridization, a model generated by one algorithm is used as an input for another algorithm (Burke, 2002; Burke, 2007). In meta-level hybridization technique, the first algorithm generates the compressed representation of a user’s preferences, the second algorithm can work on this representation easier than working with the raw data (Burke, 2002; Burke, 2007). It is not always straightforward to derive a meta-level hybrid from any given pair of recommenders.

Table 1. Advantages and Disadvantages of Hybridization Techniques

Technique	Advantages	Disadvantages
Weighted	Easy to add all of the system's capabilities.	Weight of different techniques does not change over different subjects.
Switching	The system can be sensitive to the strengths and weaknesses of its constituent recommenders.	Introduces additional complexity into the recommendation process since the switching criteria must be determined.
Mixed	Practical to make a large number of recommendations simultaneously	Does not avoid the 'new user' start-up problem.
Feature combination	Reduces the sensitivity of the system to the number of users who have rated an item.	Gives the system information about the inherent similarity of items that are otherwise opaque.
Cascade	The system avoids employing a second, lower-priority technique on items that are already well differentiated.	Imprecise recommendations resulting from using insufficient numbers of techniques.
Feature augmentation	Offers a way to improve core system performance.	Not clear how to apply a feature augmentation technique for any two recommendation algorithms.
Meta Level	The learned model is a compressed representation of a user's interest,	Not always straightforward to derive a meta-level hybrid from any given pair of recommenders.

2.3 Evaluation techniques of recommender systems

Three types of experimental settings are used for evaluating recommender systems: (1) offline experiments that use a pre-collected dataset of users' choices and rated items, (2) user studies where a set of test subjects is recruited and asked to perform several tasks requiring an interaction with the recommendation system, and (3) an online evaluation that redirects a small percentage of the traffic to different alternative recommendation engines and records user interactions with those different systems (Shani & Gunawardana, 2011).

Since recommender systems are sophisticated systems, they should be evaluated by different dimensions. There are many dimensions that play an important role to evaluate recommender systems, and also metrics, which are used to measure these dimensions (Avazpour, Pitakrat, Grunske, & Grundy, 2014). These dimensions can be listed as follows (Avazpour, Pitakrat, Grunske, & Grundy, 2014): Correctness, coverage, diversity, trustworthiness, recommender confidence, novelty, serendipity, utility, risk, robustness, learning rate, usability, scalability, stability, privacy, and user preference.

In the following subsection, first experimental settings that are applied to evaluate recommender systems will be mentioned. Then, dimensions and metrics which are used to assess recommender systems will be explained in details.

2.3.1 Offline experiments

A pre-collected data set consisting of the ratings or preferences of users is used to perform an offline experiment (Shani & Gunawardana, 2011). The main assumption behind the offline experiments is that the behaviors of users will be similar in the future (Shani & Gunawardana, 2011).

An offline experiment is generally applied on pre-recorded user data. A part of the data is hid and used as test data. After the recommendation model is constructed with non-hid (training) data, this model is applied on the test data and the ratings are predicted. Then the predicted results are compared with the actual results in the test data set and the evaluation is performed according to several metrics.

The main advantage of an offline experiment is that it is easy to apply since it does not require a real user interaction. Therefore, huge amount of user data can be used for offline experiment easily.

An offline experiment has some disadvantages although it is easy to apply on huge amount of user data. The first disadvantage of an offline experiment is that only limited number of dimensions such as correctness, coverage, learning rate, and scalability can be measured by an offline experiment (Avazpour, Pitakrat, Grunske, & Grundy, 2014). The second disadvantage is the assumption that the future user behavior will be the same as the time that the data was collected. Thus, the direct effect of the recommendation cannot be measured with this method since there is no direct interaction with the user (Shani & Gunawardana, 2011).

Therefore, in order to make the evaluation process stronger, user studies or online experiments can be performed after offline experiments (Shani & Gunawardana, 2011).

2.4.2 User studies

Although the offline experiments are widely performed to evaluate recommender systems, feedback from real users provides additional information about the system performance. Thus, user studies are very important to evaluate recommender systems

(Herlocker, Konstan, & Riedl, 2000; Ozok, Fan, & Norcio, 2010; Shani & Gunawardana, 2011; Avazpour, Pitakrat, Grunske, & Grundy, 2014).

In a user study, a set of test users is asked to perform several tasks on a recommendation system (Shani & Gunawardana, 2011). Test users are observed and their behaviors are recorded while they perform their tasks on the recommender system (Shani & Gunawardana, 2011). Many qualitative questions can be asked, before, during, and after the task is completed (Shani & Gunawardana, 2011).

User studies can address the widest set of questions about recommender systems among other experimental settings (Shani & Gunawardana, 2011). They can measure the effect of recommendation on user behavior in real time. With this property, it is better than offline experiments since it can capture real time reactions. In user studies, the main aim is to understand whether user likes the recommendations. In addition to this via open-ended questions, feedback of the users can be collected and the recommender system can be improved (Shani & Gunawardana, 2011).

Although user studies have various advantages, they have some disadvantages. Conducting a user study is very expensive. Finding a large set of real users and asking them to perform different tasks is very difficult, costly and time consuming (Shani & Gunawardana, 2011). Although, the researcher offers some incentives for the participants, it is still hard to find volunteers to perform all the tasks. Therefore, typically the study is restricted to a small set of subjects and a relatively small set of tasks, and cannot test all possible scenarios (Shani & Gunawardana, 2011).

2.4.3 Online experiments

The online experiments, which are conducted with real users and with the real online system, provide the strongest results about the evaluation of the system (Shani & Gunawardana, 2011). The online experiments can be applied by directing some of the users to the experiment system and record the user interactions with different systems.

There are some points that should be considered when applying online experiments. The participants of online experiments should be chosen randomly (Schafer, Frankowski, Herlocker, & Sen, 2007). It is also important to test different aspects of the recommender system (Shani & Gunawardana, 2011). On the other hand, different algorithms may also be tested with this experiments and concluded that one system is superior to the other (Shani & Gunawardana, 2011).

Online experiments can be used for various aims. For instance, if the aim is to measure the effect of the user interface of the system, the algorithm should be kept constant and the user interface should be changed. On the other hand, if the aim is to measure the effect of different algorithms, then the user interface should be kept constant and the algorithm should be changed.

Using online experiments can sometimes be risky (Shani & Gunawardana, 2011). Test users may be bored and disturbed by unrelated recommendations, and they may lose their enthusiasm to use the real system (Shani & Gunawardana, 2011). Therefore, the best way is to perform an online evaluation after an extensive offline study, which validates the performance of the algorithm. It can be also applied even after a user study.

2.4.4 Dimensions and metrics for evaluation of recommender systems

Recommender systems can be evaluated according to various dimensions. In this part these dimensions and the metrics that are used will be explained.

The dimensions are categorized into 4 aspects in the literature (Avazpour, Pitakrat, Grunske, & Grundy, 2014). Table 2 presents the dimensions in a categorized manner.

Table 2. Categorization of Dimensions

Recommendation-Centric	**User-Centric**
Correctness	Trustworthiness
Coverage	Novelty
Diversity	Serendipity
Recommender Confidence	Utility
	Risk
System-Centric	**Delivery-Centric**
Robustness	Usability
Learning Rate	User Preference
Scalability	
Stability	
Privacy	

(Avazpour, Pitakrat, Grunske, & Grundy, 2014)

Recommendation centric dimensions imply the dimensions assess the recommendation system itself (Avazpour, Pitakrat, Grunske, & Grundy, 2014). These dimensions are correctness, coverage, diversity, and recommender confidence.

Correctness implies how predictions of a recommendation system fit with the preferences of the users (Avazpour, Pitakrat, Grunske, & Grundy, 2014). A metric, which will be used to measure correctness depends on the types of the prediction that recommender system generates. Recommender system may predict rating of the items given by a user, ranking of the items, or items that users may like to use. The correctness of the rating prediction is generally measured by root mean squared error (RMSE), or mean absolute error (MAE). RMSE shows the square root of the mean of the difference between actual rating and predicted rating (see Eq. 1).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(predicted\_rating_i - actual\_rating_i)^2}{n}} \tag{1}$$

MAE shows the absolute value of the difference between actual rating and predicted rating (see Eq. 2).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|actual\_rating_i - predicted\_rating_i| \tag{2}$$

The correctness of the ranking prediction is generally measured by Normalized Distance-based Performance Measure (NDPM), Normalized Discounted Cumulative Gain (NDCG), Spearman's ρ, or Kendall's τ. The value of NDPM may change between 0 and 1. If the value of NDPM is close to zero, then it can be said that the ranking of the recommendation system is very similar to the actual ranking. In addition to this, the correlation between actual and predicted rankings may be measured by Spearman's ρ, or Kendall's τ. On the other hand, NDGC is used to assess the positions of items in the ranking list.

If a recommender system predicts whether an item will be liked by a user, then in order to assess this type of a recommender system, precision, recall, and F1 scores are used. Precision (see Eq. 3) indicates the percentage of correctly recommended items over total recommended items, while recall (see Eq. 4) shows the percentage of recommended items over the total number of liked items by the user. The F1 score (see Eq. 5) is calculated using both precision and recall, and measures the accuracy of the system.

$$Precision = \frac{\#\ of\ correctly\ recommended\ items1}{\#\ of\ total\ recommended\ items} \tag{3}$$

$$Recall = \frac{\#\ of\ correctly\ recommended\ items}{\#\ of\ total\ usefull\ items} \tag{4}$$

$$F1\ Score = \frac{2PrecisionRecall}{Precision + Recall} \tag{5}$$

Coverage indicates the percentage of users that system generates recommendation for (prediction coverage), and the percentage of items that system can recommend (catalogue coverage).

Recommender systems should not always recommend similar items to the user. Users also want to discover new items that they may like. Therefore, if a recommender system can also recommend such a new item then, its diversity will be high.

On the other hand, in order to recommend items that may be liked by the user, the similarity between the recommended items and the user preferences is also important. Similarity can be sum, average, minimum or maximum distance between item pairs. However, there should be a balance between diversity and the similarity between the recommended items. Therefore, a metric quality is used to measure this balance (Avazpour, Pitakrat, Grunske, & Grundy, 2014).

Recommender confidence shows how confident is the recommendation system in its recommendations (Avazpour, Pitakrat, Grunske, & Grundy, 2014).

User-centric dimensions, which indicates the level of how recommender system fulfills its target end-user needs, are trustworthiness, novelty, serendipity, utility, and risk. Novelty implies the degree that the recommended items are new and unknown for the users (Avazpour, Pitakrat, Grunske, & Grundy, 2014). Serendipity means the degree that the successfully recommended items are new and unknown for the users (Avazpour, Pitakrat, Grunske, & Grundy, 2014). Utility represents how much the users will gain a value from the recommendation (Avazpour, Pitakrat, Grunske, & Grundy, 2014). Risk shows the degree of risk that user will take when s/he accepts the recommendation (Avazpour, Pitakrat, Grunske, & Grundy, 2014).

System-centric dimensions, which evaluates recommender system according to the technical aspects, are robustness, learning rate, scalability, stability, and privacy (Avazpour, Pitakrat, Grunske, & Grundy, 2014). Robustness shows the degree of tolerance of the recommendation system towards the wrong information. The learning rate indicates how quickly the information system can cope with new information and create new recommendations in the direction of this information. Scalability implies how the recommendation system handles the huge amount of data and whether it generates a recommendation easily. Stability shows the consistency of the recommendation over time. Privacy implies the degree of risk towards user privacy of using a recommendation system.

Delivery-centric dimensions, which takes into consideration the actual usage of the recommender systems, are usability and user preference (Avazpour, Pitakrat, Grunske, & Grundy, 2014). Usability means whether users will evaluate the recommender systems as usable. User preference implies whether a recommender system is suitable for the users' perceptions.

Table 3. Summary of Dimensions and Metrics

Dimension	Metric	Type(s)
Correctness	Rating: RMSE, NRMSE, MAE, NMAE Ranking: NDPM, Spearman Correlation, Kendall Correlation, NDCG Classification: Precision, Recall, False Positive Rate, F-Measure, ROC	Quantitative
Coverage	Catalogue Coverage, Prediction Coverage	Quantitative
Diversity	Diversity Measure, Relative Diversity, Precision-Diversity Curve, Q-Statistics	Quantitative
Trustworthiness	User Studies	Qualitative
Confidence	Similarity Indicators	Qualitative/Quantitative
Novelty	Comparing recommendation list and user profiles, counting popular items	Qualitative/Quantitative
Serendipity	Comparing recommendation list and user profiles, ratability	Qualitative/Quantitative
Utility	Profit based utility function, user study	Qualitative/Quantitative
Risk	Depending on application and user preference	Qualitative/Quantitative
Robustness	Prediction shift, average hit ratio, average rank	Qualitative/Quantitative
Learning Rate	Correctness over time	Qualitative/Quantitative
Usability	User study	Qualitative/Quantitative
Scalability	Training Time, recommendation throughput	Qualitative/Quantitative
Stability	Prediction Shift	Qualitative/Quantitative
Privacy	Differential Privacy, RMSE vs. Differential Privacy curve	Qualitative/Quantitative
User Preference	User Study	Qualitative/Quantitative

(Avazpour, Pitakrat, Grunske, & Grundy, 2014)

In Table 3, dimensions, measurement metrics, and research types are presented. While correctness, coverage, and diversity can be measured with quantitative methods, trustworthiness can only be measured qualitatively. Other dimensions can be measured either as qualitatively or quantitatively.

2.4 Location recommender systems

Social media platforms are very rich data resources for researchers to be mined and gain insight about user preferences. The increasing use of location-related technologies enables the development of location based-services. Therefore, location-based social networks (LBSNs) which have become the host of new possibilities for user interaction are emerged. These technologies provide location and time information, making it possible for users to share their location. Those systems, which facilitate users to share their visits and explore other locations, have accumulated huge amount of data about users with the intensive usage in time. Location Recommendation Systems (LRSs) have been developed by discovering embedded information from LBSN data to provide location suggestion for the users.

2.4.1 The usage of the content-based and collaborative filtering for location recommender systems

Traditional approaches (content-based filtering and collaborative filtering), which have been intensively used for other types of recommendation (music, book, new, articles, etc.) have also been applied for location recommendation. While collaborative filtering establishes that personal recommendations can be computed by calculating the similarity between one user’s preferences and the preferences of other individuals; content-based filtering utilizes the information about an item itself for recommendations.

Content-based filtering is applied for location recommendation by matching the user profile with the characteristics of the venues. The most frequently used types of content information of locations and the studies including these variables are presented in Table 4.

Table 4. Most Frequently Used Types of Content Information on Locations

Types of Content Information	Related Studies
Category	(Gupta & Singh, 2013) (Yu, Feng, Xu, & Zhou, 2014) (Bao, Zheng, & Mokbel, 2012) (Yin, Sun, Cui, Hu, & Chen, 2013) (Kuo, Chen, & Liang, 2009) (Shimada, Uehara, & Endo, 2014) (Ahmedi, Rrmoku, Sylejmani, & Shabani, 2017)
Tag	(Subramaniyaswamy, Vijayakumar, Logesh, & Indragandhi, 2015) (Cao, et al., 2010) (Guo, Shao, Tan, & Yang, 2014) (Memon, et al., 2015) (Xu, Chen, & Chen, 2015)
Tips/Comments	(Krishna, Misra, Joshi, & Obaidat, 2013) (Dhake, Lomte, Auti, Nagargoje, & Patil, 2014) (Sarwat, Levandoski, Eldawy, & Mokbel, 2014) (Mordacchini, et al., 2015)
Popularity	(Yu, Feng, Xu, & Zhou, 2014) (Zheng, Zhang, Ma, Xie, & Ma, 2011) (Zheng, Zheng, Xie, & Yang, 2012) (Korakakis, Spyrou, Mylonas, & Peranton, 2017)
Price	(Park, Hong, & Cho, 2007) (Kuo, Chen, & Liang, 2009) (Yu & Chang, 2009)

Venues may have various content information such as category, popularity, price of the venues. In addition to these, tips and comments that users write about venues, and tags that users give to the venues are also used to understand the characteristics of the venues.

Category is an intensively used characteristic for content-based filtering algorithms. Tags, tips and comments, popularity, and price of the venues have also been used for content-based filtering. In content-based filtering, content information on locations is used to counteract data sparsity problems that occur in collaborative filtering algorithms.

Distance has been intensively used for location recommender system (Park, Hong, & Cho, 2007; Aihara, Koshiba, & Takeda, 2011; Wang, Li, & Feng, 2014; Yin, Cui, Sun, Hu, & Chen, 2014; Yu, Feng, Xu, & Zhou, 2014; Sattari, Toroslu, Karagoz, Symeonid, & Manolopoulos, 2015; Yin, Cui, Chen, Hu, & Zhang, 2015). However, since it changes according to the target users, it may not be called as content information but it can be used to prune the candidate items to be recommended.

Memory-based collaborative filtering consists of user-based data, which considers user similarity in making recommendations (Li Q. , et al., 2008; Ye, Yin, & Lee, 2010; Hasegawa & Hayashi, 2013), and item-based collaborative filtering, which considers item similarity (Takeuchi & Sugimoto, 2006; Levandoski, Sarwat, Eldawy, & Mokbel, 2012).

Model-based collaborative filtering, on the other hand, generally uses data mining techniques such as artificial neural networks (Knoch, Chapko, Emrich, Werth, & Loos, 2012), naive Bayesian modeling (Gupta & Singh, 2013; Subramaniyaswamy, Vijayakumar, Logesh, & Indragandhi, 2015), association rule mining (Saraee, Khan, &

Yamaner, 2005), and singular-value decomposition (SVD) (Sattari, Toroslu, Karagoz, Symeonid, & Manolopoulos, 2015).

2.4.2 Hybrid approaches for location recommender systems

Hybrid approaches, which are the composition of collaborative and content-based filtering, have recently been awarded for their ability to improve rating prediction also for location recommendation.

Different variables have been included for the hybrid location recommender system. For instance, additional to the basic user preference matrix, sentiments analysis has been performed on the comments of the venues in order to model user preferences comprehensively (Yang, Zhang, Yu, & Wang, 2013).

Baral and Li (2016) considered for different aspect; category, distance, popularity of the venues and the social relationships of the users to generate location recommendation. Their algorithm outperformed the baseline approaches (Baral & Li, 2016).

Sattari and his colleagues (2015) proposed a hybrid model which combined both collaborative filtering and content-based filtering algorithms to recommend location. However, they reached the best performance with pure collaborative filtering algorithm (Sattari, Toroslu, Karagoz, Symeonid, & Manolopoulos, 2015).

2.4.3 The usage of contextual information for location recommender systems

Contextual information (e.g. time, weather) is crucial, especially for location recommendation systems, because users consider this information in deciding where to go, unlike recommendations for what to buy. In addition, despite the fact that contextual information is critical for recommending locations, it is not commonly used in existing systems. Some of the location recommendation engines fail to consider contextual information. Although, there are some studies including contextual information in their systems, there is no common standard for using them. The most commonly used contextual information types for location recommendation are presented in Table 5.

Table 5. Most Commonly Used Contextual Information for Location Recommendation

Context-Related Variables	Related Studies
Time	(Waga, Tabarcea, & Fränti, 2011) (Majid, Chen, Chen, Mirza, & Hussain, 2013) (Yuan, Cong, Ma, Sun, & Thalmann, 2013) (Baral & Li, 2016) (Hiesel, Braunhofer, & Wörndl, 2016) (Baltrunas & Amatriain, 2009)
Weather Conditions	(Majid, Chen, Chen, Mirza, & Hussain, 2013) (Hiesel, Braunhofer, & Wörndl, 2016) (Trattner, Oberegger, Eberhard, Parra, & Marinho, 2016)
Temperature	(Majid, Chen, Chen, Mirza, & Hussain, 2013)
Trip type	(Zheng, Burke, & Mobasher, 2012)
Origin City/Destination City	(Zheng, Burke, & Mobasher, 2012)
Speed and Travel Direction	(Barranco, Noguera, Castro, & Martínez , 2012)
Transportation Type	(Savage, Baranski, Chavez, & Höllerer, 2012)

Zheng, Burke, and Mobasher (Zheng, Burke, & Mobasher, 2012) applied algorithms for pre-filtering, context relaxation, and hybrid techniques for contextual variables (trip type, origin city, and destination city).

Using pre-filtering, Barranco et al. (2012) used speed and travel direction as contextual variables. Mode of transportation (biking, walking, and driving) is another contextual variable that has been used as a pre-filter (Savage, Baranski, Chavez, & Höllerer, 2012).

Time is commonly used as context data (Waga, Tabarcea, & Fränti, 2011; Majid, Chen, Chen, Mirza, & Hussain, 2013; Yuan, Cong, Ma, Sun, & Thalmann, 2013; Baral & Li, 2016; Hiesel, Braunhofer, & Wörndl, 2016), but it is used in different forms. For instance, Waga, Tabarcea, and Franti (2011) examined the timestamps of photos taken in touristic places. They also considered the season when the photos were taken. Majid et al. (2013) discretized time as morning, afternoon, evening, and night, and days as weekday and weekend. The recommendations were generated by post-filtering of those variables. Yuan et al. (2013) examined venue similarity for different time periods during the day. Baral and Li (2016) recommended locations according to a location category that is mostly visited in a specific time frame, and Hiesel, Braunhofer and Worndl (2016) examined location popularity in different time periods.

Weather is yet another important contextual variable for location recommendation (Trattner, Oberegger, Eberhard, Parra, & Marinho, 2016). Majid et al. (2013) used weather information, specified as temperature and weather conditions. Hiesel, Braunhofer and Worndl (2016) examined location popularity in different weather conditions. Other studies have proposed frameworks for contextual location recommendation systems (Park, Hong, & Cho, 2007; Aihara, Koshiba, & Takeda, 2011; Wang, Li, & Feng, 2014; Yin, Cui, Sun, Hu, & Chen, 2014).

2.4.4 Personalization and recommender systems

Personalization is another aspect, which should be handled in the research of the recommendation systems. Personalization means offering items to the customers according to their preferences by considering certain contextual circumstances that these items will be accessible (Adomavicius, Huang, & Tuzhilin, 2008).

Recommender systems generally works on the prediction of ratings and they ignore to utilize from the whole transactional history of the users. However, advanced profiling of the user preferences can generate better recommendations (Adomavicius, Huang, & Tuzhilin, 2008).

It is claimed that there is no common standard for applying personalization in a given context in recommender systems (Schubert, Uwe, & Risch, 2006). Personalization concept can be applied on different stages of recommender systems. Pre-filtering or post-filtering methods can be applied for personalization (Adomavicius, Huang, & Tuzhilin, 2008). One example that can be used for personalization purposes in recommendation systems is the changing effects of each variable among different users. The contextual variables can also be used for personalization purposes. For instance, two people may like the same place but in different contextual circumstances. Therefore, it is important to consider the changing effects of contextual variables for personalization of different users.

CHAPTER 3

METHODOLOGY

“Alice: Would you tell me, please, which way I ought to go from here?

The Cheshire Cat: That depends a good deal on where you want to get to.

Alice: I don't much care where.

The Cheshire Cat: Then it doesn't much matter which way you go.

Alice: ...So long as I get somewhere.

The Cheshire Cat: Oh, you're sure to do that, if only you walk long enough.”

Lewis Carroll

The methodology of this study will be explained in three subsections; data collection, data preprocessing, and development of recommender system. For data collection part Twitter and Foursquare APIs were utilized. All other parts of the algorithm were coded with R programming language.

3.1 Data collection

Data was collected in three stages. First, Twitter was used because it allows direct crawling of its users’ check-in history (unlike Foursquare, for example). Twitter also allows programmers to utilize REST APIs, which are frequently used for designing web APIs to use a pull strategy for data retrieval, and streaming APIs, which are used for continuous streaming of public data with a push strategy.

Some Foursquare users link their accounts with Twitter, their check-in information can be crawled from Twitter. In order to access these tweets, the REST API “GET search/tweets” was used. This API returned a collection of tweets that matched a specified query. When a user whose Foursquare account is linked to Twitter makes a check-in using Foursquare, related tweets, including all check-in information, appear on his/her Twitter timeline. The API returned the users who checked in and shared this check-in on their Twitter accounts. Location information shared by users via Foursquare were collected over a period of two months. All tweets of collected users were recorded to reach their check-in history. A Twitter dataset was connected via PHP APIs (for Twitter API version 1.1), and a MySQL database was used for storing the retrieved data.

Venue information on Foursquare was accessed using URLs on check-in tweets. The Foursquare (Foursquare) API (https://api.Foursquare.com/v2/venues/VENUE_ID) which gives details about a venue was used, and the following attributes were collected:

- Venue Name,
- Category,
- Latitude and longitude,
- Check-in count,
- Visitor count,
- Tip count, and
- Price classification.

In order to identify the weather conditions at the time of check-ins, the weather history was collected from the Weather Underground website. Each check-in date was matched to the date in the weather history, and the weather information (e.g. sunny, rainy, and snowy) was added to the check-in data. Weather Underground provides .csv files that include dates and weather conditions. The visit check-in dates were compared to the dates in the weather files, and the coding automatically added the related weather conditions to the check-in data.

3.2 Data preprocessing

Raw data set consisted of 6738 users, 60202 venues, and 226227 visits. Check-in history of users is filtered according to the following criteria:

- Users having two standard deviation more tweets than the average number of tweets were extracted for bot detection.
- Only check-ins from Istanbul were retrieved since Istanbul has 4 times more check-ins than the nearest city as a check-in number.
- There are various categories of venues in Foursquare. For this study, restaurant was chosen as a main category and all related sub-categories of restaurant were used because of intensive check-in frequency in restaurants.
- Users who visited only one venue were extracted.
- Venues, which were visited by only one user, were extracted.

After that, 1101 users, 711 venues, and 4694 visits were remained.

Variables used in this study will be explained in the following subsections. Data preprocessing procedures are presented in Appendix A. The notations that were used in this thesis were explained in Table 6.

Table 6. Notations and Their Explanations

Notation	Explanation	Notation	Explanation
u_n	nth user	$\overrightarrow{u_n}$	Vector keeping the rating of nth user
v_n	nth venue	$\overrightarrow{u_m}$	Vector keeping the rating of mth user
x	x coordinate of a venue	$\overrightarrow{v_n}$	Vector keeping the rating of nth venue
y	y coordinate of a venue	$\overrightarrow{v_m}$	Vector keeping the rating of mth venue
z	z coordinate of a venue	$P(Time = morning \mid visit_{ij})$	The probability of user "i" visits venue "j" in the morning
$center_x$	x coordinate of a user center	$Contextual\_Similarity_{jk}$	Contextual similarity of venue "j" and venue "k"
$center_y$	y coordinate of a user center	$visit\_percentage_{ik}$	Percentage that user "i" visits venue "k" in the morning
$center_z$	z coordinate of a user center	$Actual\_Rating(u_n, v_n)$	Rating of user "n" to the venue "n", which can get values from 1 to 5
$rating_{ucf}$	Predicted rating coming from user-based collaborative filtering	$Freq(u_n, v_n)$	The number of visits of user "n" to the venue "n"
$rating_{icf}$	Predicted rating coming from item-based collaborative filtering	$\min freq(u_n)$	Minimum number of visits of the user "n"
$rating_{distance}$	Predicted rating calculated with distance variable	$\max freq(u_n)$	Maximum number of visits of the user "n"
$rating_{category}$	Predicted rating calculated from the similarity of category preferences of the users	$rating_{price}$	Predicted rating calculated from the similarity of price preferences of the users
$rating_{popularity}$	Predicted rating calculated from the similarity of popularity preferences of the users	$rating_{context}$	Predicted rating calculated from the contextual similarity of the venues

3.2.1 Rating

Foursquare does not provide the direct ratings of individual users. Therefore, actual rating value was calculated from linearly normalization of the frequencies in a range of one to five for each user-venue pair (see Eq. 6). If a person's maximum and minimum number of visits are equal, then rating was determined as 1.

$$Actual\_Rating(u_n, v_n) = \left(\left(\frac{Freq(u_n, v_n) - \min freq(u_n)}{\max freq(u_n) - \min freq(u_n)} \right) * 4 \right) + 1 \qquad (6)$$

3.2.2 Distance

The latitude and longitude values of venues, which were collected from Foursquare, were converted into the x, y, and z coordinates (see Eq. 7, 8, 9). Since the visits were only chosen from the Istanbul, the latitude and longitude values of venues were checked accordingly.

$$x = \cos(longitude) * \cos(latitude) \qquad (7)$$

$$y = \sin(longitude) * \cos(latitude) \qquad (8)$$

$$z = \sin(latitude) \qquad (9)$$

User centers were calculated by taking the weighted average of x, y, z coordinates of all visits for each user in order to understand his/her active area. For each visit of a specific user, Euclidean distance from venue to the user center was determined and called as distance variable (see Eq. 10).

$$distance = \sqrt{(x - center_x)^2 + (y - center_y)^2 + (z - center_z)^2} \qquad (10)$$

3.2.3 Popularity

Foursquare-API provides four variables about a venue; check-in count, like count, user count, and tip count. In this study, it was considered that those variables refer to the popularity of a venue. Therefore, popularity variable was created from those four properties by Principal Component Analysis (PCA), which aims dimension reduction (Wold, Esbensen, & Geladi, 1987). Sampling adequacy can be observed in Table 7, which presents KMO value as 0.821 and significance of Barlett's Test of Sphericity as 0.001. Acceptable level of KMO is generally 0.6, and Barlett's Test of Sphericity is significant at 1% alpha level. These results showed that sample is adequate for PCA.

Table 7. KMO and Barlett's Test Results

KMO Measure of Sampling Adequacy		0.821
Barlett's Test of Sphericity	Approximate Chi Square	13951.963
	Degrees of freedom	6
	Significance	0.001

93.69 of total variance is explained by only one component (see Table 8). Therefore, it can be concluded that one variable which is named as "popularity" can be used instead of the four variables.

Table 8. Total Variance Explained

Component	% of Variance	% Cumulative Variance
1	93.69	93.69
2	4.46	98.15
3	1.08	99.19
4	0.80	100

Component matrix shows the correlation between variables and the component. Since the correlation values range from -1 to +1, it can be concluded that there are strong positive correlation between the component and each of the variables (see Table 9).

Table 9. Component Matrix for Popularity

check-in count	0.965
like count	0.985
user count	0.981
tip count	0.941

3.2.4 Category

All sub-categories of food, which were collected by Foursquare-API, were included in this study. There are 34 restaurant categories including different countries’ cuisine in the data set. User-category matrix, which presents the number of visits of each user in each sub-category, is prepared.

3.2.5 Price

There are four price classes in Foursquare: 1-Cheap, 2-Average, 3-Expensive, and 4-Very Expensive. User-price matrix presenting the number of visits of each user in each price class was prepared by using the data coming from Foursquare-API.

3.2.6 Time

Twitter provides UNIX time format for each tweet that was converted to date and time stamp. For this study, season, day, and the different periods of the day were used as contextual variables. It was observed that some contextual variables show similar characteristics, such as users have the same pattern of check-in behavior for weekdays according to the check-in frequencies. For this reason, discretization was applied on contextual variables to provide recommendations that are more accurate. Days were discretized as “weekday” and “weekend” (Baltrunas & Amatriain, 2009; Hiesel, Braunhofer, & Wörndl, 2016).

Spring and summer were discretized as “hot season” while autumn and winter are discretized as “cold season” (Baltrunas & Amatriain, 2009).

In the studies of Majid et al. (2013), and Wang et al. (2015), time was discretized as morning, afternoon, evening, and night. In the study of Baltrunas and Amatriain (2009), a day was discretized as morning and evening only. However, after examining check-in behaviors in the data set, it was found that discretization, as morning, noon, and evening will be more suitable. The time range 07:00 to 11:59 was specified as “morning”, 12:00 to 16:59 as “noon”, and 17:00 to 06:59 as “evening”.

3.2.7 Weather

The data of weather condition, which is also a contextual variable, were collected by using Weather Underground API providing more than 10 different weather conditions (sunny, rainy, snowy, rainy and stormy, snowy and stormy, etc.). It was observed that some categories of weather show the same patterns of user behavior. Therefore they were discretized under three main categories: “sunny”, “rainy”, and “snowy”.

3.3 Development of the recommender system

For this study, two versions of the hybrid algorithm were developed. The procedures of development of recommender system are presented in Appendix B.

The first algorithm consists of the following components: user similarities based on check-in history, category, popularity, and price preferences of users; venue similarities based on check-in history and venue similarities based on contextual characteristics of venues, and distance from venue to the user center. The framework of this algorithm is presented in Figure 5 and it is named as “HybRecSys”.

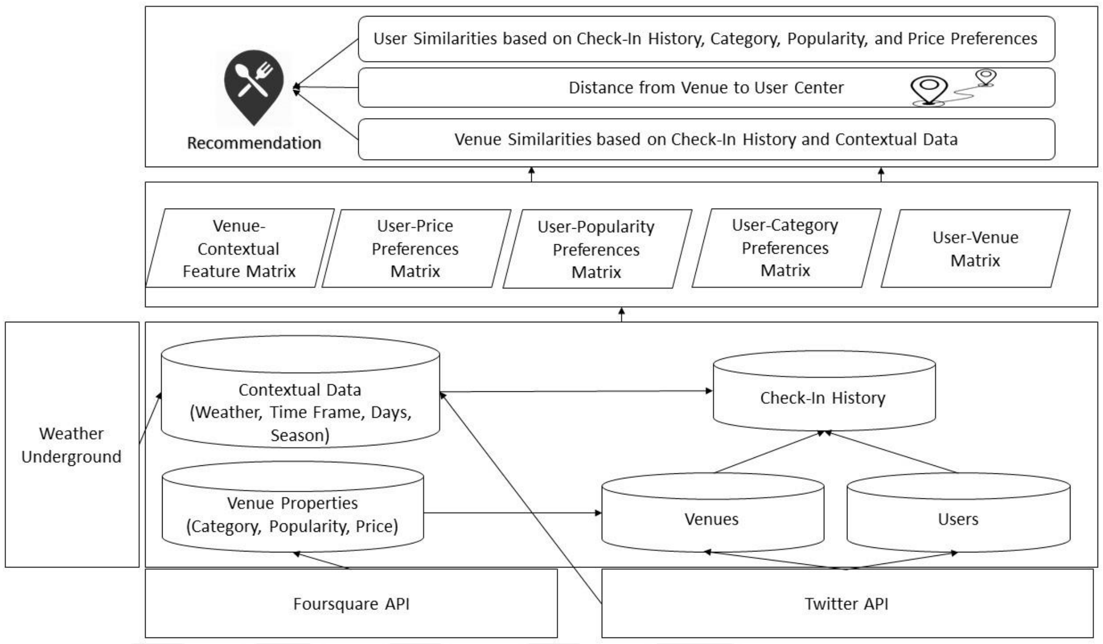
User Similarities based on Check-In History, Category, Popularity, and Price Preferences
Distance from Venue to User Center
Venue Similarities based on Check-In History and Contextual Data
Recommendation
Venue-Contextual Feature Matrix
User-Price Preferences Matrix
User-Popularity Preferences Matrix
User-Category Preferences Matrix
User-Venue Matrix
Weather Underground
Contextual Data (Weather, Time Frame, Days, Season)
Check-In History
Venue Properties (Category, Popularity, Price)
Venues
Users
Foursquare API
Twitter API

Figure 5. Framework of HybRecSys

In the final version, predicted rating coming from HybRecSys algorithm was compared with the average rating of the user, which was calculated from the previous visits of the user. Then, if the predicted rating of the venue is greater than the average rating of the user, second stage was applied. In second stage, contextual suitability of the venue to the user was examined. If the user prefers that kind of venue in that contextual circumstances, then the algorithm recommends that venue to the user in that specific contextual circumstances. Second and the final version of the algorithm is presented in Figure 6 and it is named as “Contextually Personalized HybRecSys”.

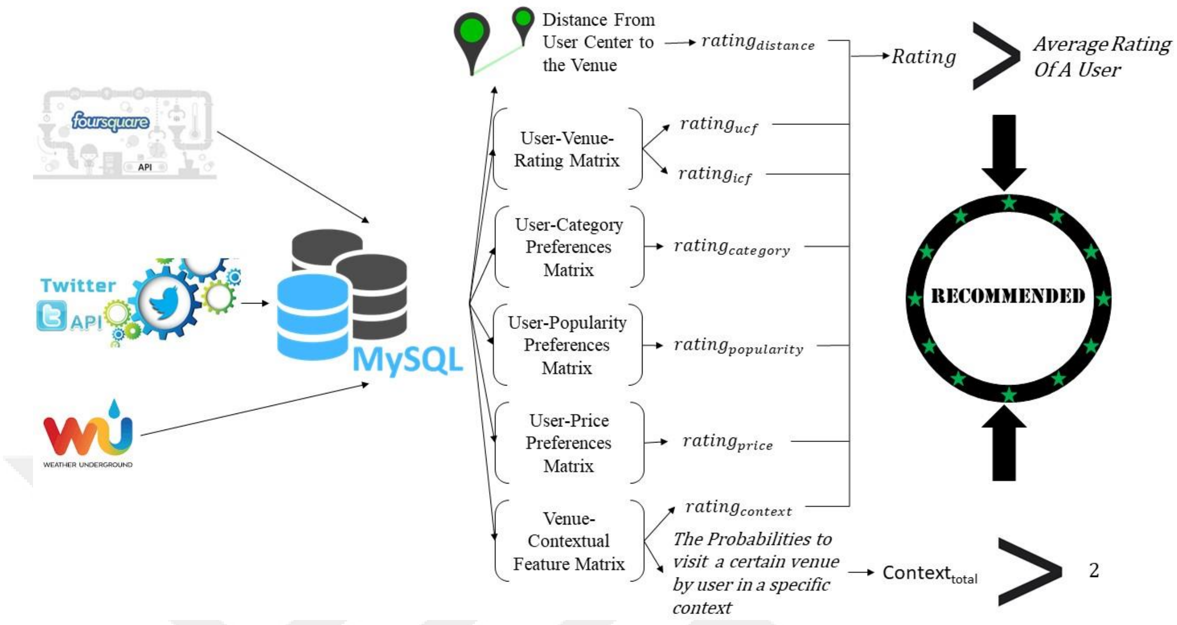

Figure 6. Framework of contextually personalized HybRecSys

3.3.1 User-based collaborative filtering

User similarities can be measured using Jaccard similarity, Cosine distance, Euclidean distance, or correlation distance. Cosine distance, used to measure the degree of similarity between two vectors of an inner product space that measures the cosine of the angle between them (Gorakala & Usuelli, 2015), was employed in the present study. First, the user-venue matrix was constructed, which keeps track of the user ratings to venues. User-user similarity was calculated with the equation (see Eq. 11) below, using the user-venue matrix.

$$\text{user\_sim}\ (\overrightarrow{u_n}, \overrightarrow{u_m})=\cos(\overrightarrow{u_n}, \overrightarrow{u_m})=\frac{\overrightarrow{u_n} \cdot \overrightarrow{u_m}}{||\overrightarrow{u_n}||*||\overrightarrow{u_m}||} \tag{11}$$

Equation 12 was used to calculate the user ratings of venues.

$$\text{rating}_{ucf}(\mathrm{u_n}, \mathrm{v_n}) = \frac{\sum \text{user\_sim}(\mathrm{u_n},\mathrm{u_m}) \ \text{x rating}(\mathrm{u_m}, \mathrm{v_n})}{\sum \text{user\_sim}(\mathrm{u_n},\mathrm{u_m})} \qquad (12)$$

3.3.2 Item-based collaborative Filtering

Venue similarity was calculated by utilizing the cosine distance from the user-venue matrix. Venue-venue similarity was calculated using the equation (see Eq. 13):

$$\text{venue\_sim}\ (\overrightarrow{\mathrm{v_n}}, \overrightarrow{\mathrm{v_m}})=\cos(\overrightarrow{\mathrm{v_n}}, \overrightarrow{\mathrm{v_m}})=\frac{\overrightarrow{\mathrm{v_n}} \cdot \overrightarrow{\mathrm{v_m}}}{||\overrightarrow{\mathrm{v_n}}||*||\overrightarrow{\mathrm{v_m}}||} \qquad (13)$$

The predicted user ratings of venues were calculated using the following equation (see Eq. 14):

$$\text{rating}_{icf}(\mathrm{u_n}, \mathrm{v_n}) = \frac{\sum \text{venue\_sim}(\mathrm{v_n},\mathrm{v_m}) \ \text{x rating}(\mathrm{u_n}, \mathrm{v_m})}{\sum \text{venue\_sim}(\mathrm{v_n},\mathrm{v_m})} \qquad (14)$$

3.3.3 Content-based filtering

User-category preference matrix (see Table 10) that keep track of the number of venues a specific user visits in each category was constructed.

Table 10. Sample of User-Category Matrix

user id	Turkish_Rest	American_Rest	Chinese_Rest
11949	10	0	0
10147822	0	6	2
10437542	0	2	0

Popularity values were discretized as high, medium, or low according to the normalized popularity values obtained from the PCA. Next, a user-popularity preference matrix (see Table 11) was generated to keep track of the number of venues that a specific user visited in each popularity category.

Table 11. Sample of User-Popularity Matrix

user id	low_popularity	medium_popularity	high_popularity
11949	7	3	0
10147822	0	5	3
10437542	2	0	0

Also a user-price preference matrix (see Table 12) was constructed to keep track of the number of venues that a specific user visited in each price category.

Table 12. Sample of User-Price Matrix

user id	1	2	3	4
11949	6	3	1	0
10147822	0	0	3	5
10437542	2	0	0	0

User-user similarities according to category, popularity, and price preferences were also calculated, using a technique similar to the one used in equation 2.

Predicted ratings using user similarity depend on the category, popularity, and price preferences of users. These ratings were also calculated in a similar manner, using equation 3.

The calculation of ratings according to distance between a venue and a user assumed that, if the distance between a venue and the user center is short, the user will visit that venue more frequently (Zheng, Zhang, Xie, & Ma, 2009; Ye, Yin, Lee, & Lee, 2011; Yuan, Cong, & Sun, 2014). Users are typically more willing to check in at venues near their centers, but each user's perception of distance is different. For this reason, a power law distribution (Ye, Yin, Lee, & Lee, 2011; Yuan, Cong, Ma, Sun, & Thalmann, 2013; Yuan, Cong, & Sun, 2014) was fitted to each user's visits, and optimal coefficients (A, B, and n) were found for each user to model the willingness of a user to check-in at a place to minimize the difference between actual rating and predicted rating (see Eq. 15).

$$rating_{distance} = A + B * distance^{n} \tag{15}$$

3.3.4 Context information

A venue-context matrix (see Table 13) was prepared to keep track of venue contextual characteristics. This matrix presents the percentage of venue preferences in different contextual circumstances. With this matrix, venue similarities were calculated using equation 4. Predicted ratings that depended on the contextual similarity of venues were calculated using equation 5.

Table 13. Sample of Venue-Context Matrix

Venue	Hot_S	Cold_S	W_day	W_end	Morning	Noon	Evening	Sunny	Rainy	Snowy
V1	0.75	0.25	0.68	0.32	0.08	0.17	0.75	0.67	0.25	0.08
V2	0.5	0.5	0.5	0.5	0	1	0	1	0	0
V3	0.83	0.17	0.96	0.04	0.13	0.52	0.35	0.65	0.35	0

3.3.5 Hybrid recommendation

In the present study, a weighted hybrid recommender was applied that computed the score of recommended items from the results of all available recommendation techniques. Instead of using equal weights for each algorithm, an artificial neural network analysis was applied in order to find the optimal weights for each technique. A multilayer perceptron artificial neural network model was implemented using SPSS (Statistical Package for Social Sciences) software. The following parameters were used:

- Stopping criteria: Maximum 2 steps without a decrease in error
- Learning rate: 0.1
- Activation function: Hyperbolic Tangent
- Initial weight: Randomized

The results of all available recommendation techniques were used as inputs, and actual ratings were used as output to be predicted. Artificial neural network algorithm generated only one hidden layer and its activation function is hyperbolic tangent. Error function of output layer is sum of squares. Model summary table is presented in Table 14.

Table 14. Model Summary of Artificial Neural Network

Training	Sum of Squares Error	0.016
	Relative Error	0.001
	Stopping Rule Used	2 consecutive steps with no decrease in error
	Training Time	0:00:00:0.26
Test	Sum of Squares Error	8.870x10^{-6}
	Relative Error	0.00

Final ratings were calculated by multiplying the weights ($w_1 \ldots w_7$) coming from the results of the artificial neural networks by the ratings from various algorithms, as shown in equation 16.

$$\begin{aligned} Predicted\_Rating &= w_1 * rating_{ucf} + w_2 * rating_{icf} + w_3 * rating_{distance} \\ &+ w_4 * rating_{popularity} + w_5 * rating_{category} + w_6 \\ &* rating_{price} + w_7 * rating_{context} \end{aligned} \quad (16)$$

This predicted rating is evaluated with RMSE and MAE metrics. However, in order to decide whether recommending a venue to the user or not, different threshold values are used for each user. The user threshold value is determined by taking the average of ratings of that user. After that, if calculated rating (see Eq. 16) is greater than his/her threshold, then it was considered that the user will like that venue. This decision is evaluated with the precision, recall, and F1 scores. This is the first version of the algorithm that is presented in Figure 5 and it is named as “HybRecSys”.

In general, existing recommender systems don't consider that user preferences are affected by different contextual circumstances. For instance, a user may prefer a venue in rainy afternoons, while other may prefer the same venue in another context. In order to handle this issue, the present system calculates the probability of visiting a venue in a specific contextual category. For instance the following equation (see Eq. 17) calculates the probability of visit of user "i" to the venue "j" in the mornings:

$$P(Time = morning|visit_{ij}) = \frac{\sum(Contextual\_Similarity_{jk} * visit\_percentage_{ik})}{\sum Contextual\_Similarity_{jk}} \quad (17)$$

For each user-venue pair, there are 36 different contextual circumstances (day=weekday, weekend; time=morning, noon, evening; season=hot, cold; weather=sunny, rainy, snowy). For each situation probabilities were calculated and the resulting table was constructed (Table 15).

Table 15. Final Decision Table

uid	vid	Av	Time	Day	Season	Weather	Context$_{total}$	Rating	Like
u1	v1	2.5	Morning=0.5	Wdays=0	Hot=1	Sunny=1	2.5	3	True
u1	v1	2.5	Morning=0.5	Wdays=0	Hot=1	Rainy=0	1.5	3	False
u1	v1	2.5	Morning=0.5	Wdays=0	Hot=1	Snowy=0	1.5	3	False
u1	v1	2.5	Morning=0.5	Wdays=0	Cold=0	Sunny=1	1.5	3	False

Table 15 respectively presents; user id (uid), venue id (vid), average rating of related user (Av), the probability that user will visit that venue in that time category (Time), the probability that user will visit that venue in that day category (Day), the probability that user will visit that venue in that season category (Season), the probability that user will visit that venue in that weather category (Weather), total point from all contextual variables (sum of all probability) ($Context_{total}$), predicted rating (Rating), and final decision (Like).

Sum of all categories of each contextual variable have to add up to 1. For instance, since day variable has two categories as weekdays and weekend, if a user's probability of visiting a specific venue on weekdays is 0.6, then the probability that user will visit the same venue on weekend has to be 0.4. Therefore, sum of the values of all contextual variables ($Context_{total}$) may have a value of maximum 4. The final decision of whether a user will like a venue depends on two things; predicted rating should be greater than the average rating of the user and total context value is greater than 2. The final version of the algorithm was called as contextually personalized HybRecSys (see Figure 6).

CHAPTER 4

EVALUATION

"Sometimes I've believed as many as six impossible things before breakfast."

Lewis Carroll

The performance of HybRecSys was evaluated by applying offline experiments. Moreover, the performance of the contextually personalized HybRecSys was examined both applying offline experiments and conducting a user study. The offline evaluation procedures are presented in Appendix C.

4.1 Results of offline experiments for HybRecSys

HybSecSys algorithm generates two types of recommendation results; 1) the prediction of the user ratings, and 2) the prediction of whether an item will be liked by a user. Therefore, the recommendation results of HybrecSys were evaluated with the evaluation metrics, which are used to assess rating prediction (RMSE and MAE), and the metrics, which are used to assess the binary results (precision, recall, and F1 score).

The HybRecSys was compared to a user-based K nearest neighborhood (KNN) algorithm (Konstan, et al., 1997), an item-based KNN algorithm (Deshpande & Karypis, 2004), a biased matrix factorization (Koren, Bell, & Volinsky, 2009), and a singular value decomposition ++ (SVD++) (Koren, 2008). These algorithms are suitable for our dataset, and they are ready to use in the LibRec[1], a Java library for recommender systems.

[1]http://www.librec.net. Version 1.3 of LibRec

K-fold (K=10) cross-validation technique was used to split the data into training and test sets. The dataset was split into 10 disjoint sets making sure that each set contains about 10 % visits of each user. For each fold, one set was used as test set and nine sets were used as training set.

The RMSE, MAE, and coverage values of the algorithms from Librec, and HybRecSys are presented in Table 16, Figure 7, and Figure 8.

Table 16. RMSE, MAE, and Coverage Values Based on Different Algorithms

Approaches	RMSE	MAE	Coverage
User-Based KNN (Konstan, et al., 1997)	1.296077	0.866262	38%
Item-Based KNN (Deshpande & Karypis, 2004)	1.308751	0.869726	38%
Biased Matrix Factorization (Koren, Bell, & Volinsky, 2009)	1.433791	0.893339	100%
SVD++ (Koren, 2008)	1.426615	0.886675	100%
HybRecSys	1.214821	0.8058951	100%

According to the RMSE and MAE values, the HybRecSys outperforms other algorithms. User-based KNN, item-based KNN, SVD++, and biased matrix factorization follow the HybRecSys according to RMSE and MAE values. The coverage is 100% for the HybRecSys. Although the RMSE and MAE values of user-based and item-based KNN are slightly higher than those of the HybRecSys, which has the lowest value, their coverage value is only 38%. The coverage value of SVD++ and biased matrix factorization is also 100%, but their RMSE and MAE values are higher than those of the HybRecSys.

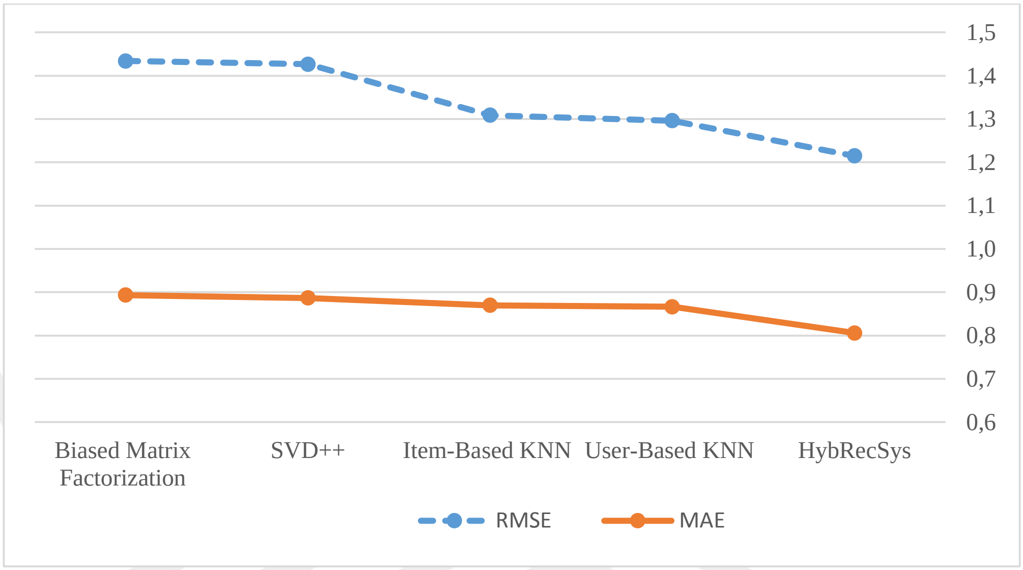

Figure 7. Graph of rmse and mae values of different algorithms

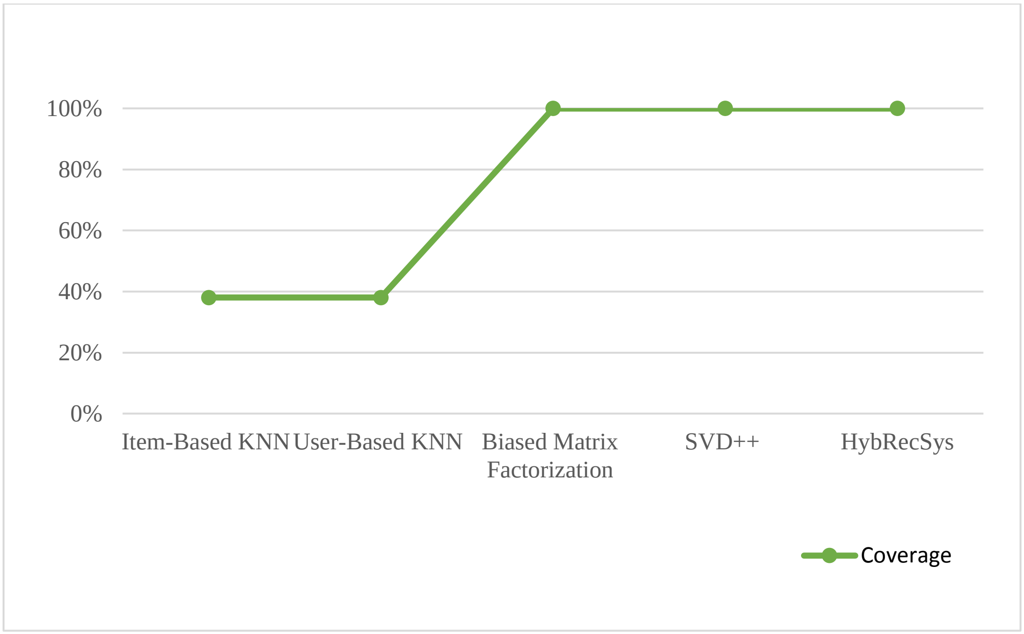

Figure 8. Graph of coverage percentages of different algorithms

The precision, recall, and F-1 values of the algorithms from Librec, and the HybRecSys are presented in Table 17. Figure 9 is a visual representation.

Table 17. Precision, Recall, and F-1 Measures of the Algorithms

Approaches	Precision	Recall	F1-Measure
User-Based KNN (Konstan, et al., 1997)	0.1220	0.0823	0.0983
Item-Based KNN (Deshpande & Karypis, 2004)	0.1154	0.1107	0.1130
Biased Matrix Factorization (Koren, Bell, & Volinsky, 2009)	0.0893	0.1200	0.1031
SVD++ (Koren, 2008)	0.0702	0.0976	0.0816
HybRecSys	0.1667	0.1460	0.1493

According to precision, recall, and F-1 values, the HybRecSys outperforms all other algorithms. User-based KNN, item-based KNN, biased matrix factorization, and SVD++ follow the HybRecSys according to precision metrics. According to recall values, biased matrix factorization, item-based KNN, SVD++, and user-based KNN follow the HybRecSys. Finally, according to the F-1 measure, item-based KNN, biased matrix factorization, user-based KNN, and SVD++ follow HybRecSys in that order.

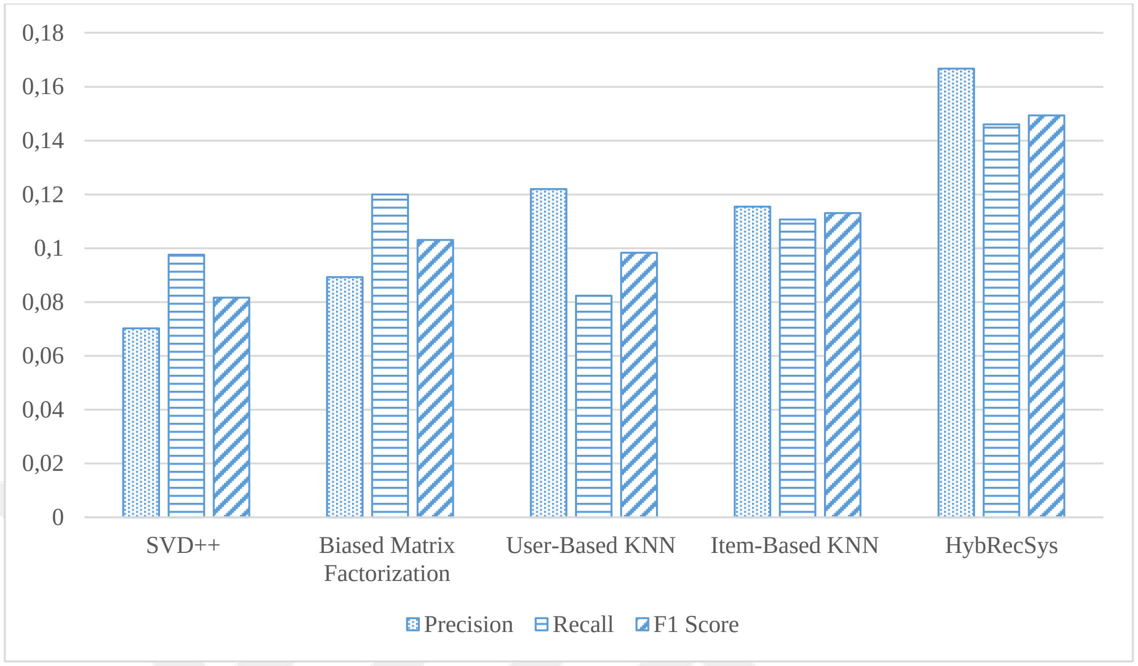

Figure 9. Graph of precision, recall, and f-1 measures of the algorithms

4.2 Results of offline experiments for contextually personalized HybRecSys

Contextually personalized HybRecSys was compared with the following approaches; user-based KNN, item-based KNN, biased matrix factorization, SVD++, and HybRecSys which is the earliest version of contextually personalized HybRecSys.

Precision, recall, and F1 measures were used as performance metrics for this version of the algorithm.

Again, k-fold (k=10) cross-validation technique was used to split the data into training and test sets. The dataset was split into 10 disjoint sets making sure that each set contains about 10% visits of each user. For each fold, one set was used as test set and nine sets were used as training set.

Precision, recall, and F1 values of each algorithm are presented in Table 18 and visual representation is in Figure 10.

Table 18. Precision, Recall, and F1 Measures of the Algorithms

Approaches	Precision	Recall	F1 Score
User-Based KNN (Konstan, et al., 1997)	0.1220	0.0823	0.0983
Item-Based KNN (Deshpande & Karypis, 2004)	0.1154	0.1107	0.1130
Biased Matrix Factorization (Koren, Bell, & Volinsky, 2009)	0.0893	0.1200	0.1031
SVD++ (Koren, 2008)	0.0702	0.0976	0.0816
HybRecSys	0.1667	0.1460	0.1493
Contextually Personalized HybRecSys	**0.18**	**0.45**	**0.25**

According to precision, recall, and F1 values, contextually personalized HybRecSys outperforms all other algorithms and also its earliest version, HybRecSys. HybRecSys, User-based KNN, Item-Based KNN, Biased Matrix Factorization, and SVD++ follow contextually personalized HybRecSys respectively according to precision metrics. On the other hand, according to recall values, HybRecSys, Biased Matrix Factorization, Item-Based KNN, SVD++, and User-based KNN follow contextually personalized HybRecSys. Finally, HybRecSys, Item-Based KNN, Biased Matrix Factorization, User-based KNN, and SVD++ follow contextually personalized HybRecSys respectively according to F1 score.

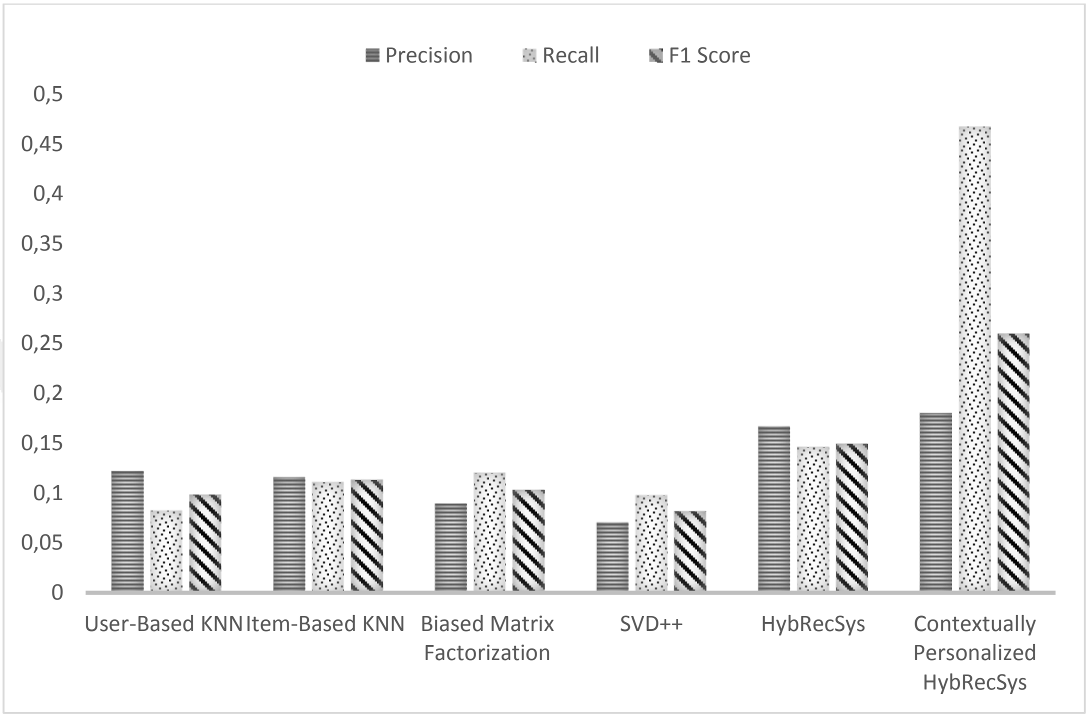

Figure 10. Graph of precision, recall, and f1 score of the algorithms

4.3 User study for contextually personalized HybRecSys

As it was indicated in the literature, user studies are very helpful to understand whether the recommendations are liked by the users, and to collect more detailed data about the recommendation system (Herlocker, Konstan, & Riedl, 2000; Ozok, Fan, & Norcio, 2010; Shani & Gunawardana, 2011; Avazpour, Pitakrat, Grunske, & Grundy, 2014). Although, conducting a user study is difficult, time consuming, and costly, it is suggested to apply them after the offline experiments in order to validate the results of the offline experiments (Shani & Gunawardana, 2011).

4.3.1 Steps of the user study

For this study, a user study was conducted on the users in our data set. For this purpose, Twitter account of each user in our data set was checked to learn whether their profiles allow to receive direct messages. Out of 1101 users 195 accounts were open for direct message. Those users were invited to attend our user study by direct message and a small incentive (a movie ticket) was promised if they attend this evaluation.

24 users replied to the message and accepted to attend our study. After that, our user evaluation occurred in the following steps:

1. Our algorithm predicted the ratings of that users to the all venues except the venues they visited.
2. Among the results, our algorithm recommended three top rated venues for each 24 users. These users were asked over Twitter message whether any of the recommended items attracts their attention.
3. 21 users replied to the recommendations. Only one of them said that none of the venues were suitable for him. Others were interested in at least one venue among recommended ones.
4. A survey (see Appendix D), which also includes the Foursquare links of recommended venues, was prepared according to their choices and sent to them.
5. The users were asked to fill the survey after they visit that recommended venue, or examine from Foursquare page of the venues.

4.3.2 Survey questions of the user study

The survey can be seen both in Turkish (see Appendix D) and English (see Appendix E) version in appendix. First question was asked to understand the appreciation of the participants to the recommended venues. It was asked in 5-point Likert scale (1- Not Like At All, 2-Not Like, 3-Not Sure, 4-Like, 5-Like Very Much).

The following four questions were asked to measure the appropriateness of the category, price, popularity and the location of the recommended venue for the participant. They were also asked in 5-point Likert scale (1- Not Appropriate At All, 2-Not Appropriate, 3-Not Sure, 4-Appropriate, 5-Very Appropriate).

Questions 6, 7, 8, and 9 were asked to understand in which contextual circumstances the user will prefer the recommended venue. For this purpose, these questions were asked as fixed sum scale questions. The participants were asked to distribute a hundred point to the categories of a contextual variable according to the tendency of user to visit that venue in these categories.

The last two questions are demographic questions. They were asked to learn the age and the education level of the participants.

4.3.3 Results of the user study

Participants' average age is 30 and age range varies from 19 to 38. There are 8 women and 12 men in the data set.12 of the participants are graduated from high school, 6 of them have bachelor's degree and 2 of them have master's degree.

Table 19 presents the answers of the participants to the first five questions of the survey. 90% of the participants like the recommended venues. 10% of them is indecisive about whether they like the recommended venue. 90% of the participants thought that the price class of the recommended venue is suitable for them. 40% of the participants thought that category of recommended venues are very appropriate for them while 20% of them thought that categories are appropriate, remaining are indecisive. 70% of the participants thought that popularity class of recommended venues are appropriate for them and 10% of them thought that the popularity classes of the venues are very appropriate. On the other hand, 10% is indecisive while other 10% thought that the popularity classes of the venues are not appropriate. Moreover, the address of the recommended venue was given to the participants and asked them whether this location is appropriate or not. 40% of the participants said very appropriate, and 30% of them said appropriate while other 30% said it is not appropriate.

Table 19. Survey Answers (Q1- Q5)

	1 (%)	2 (%)	3 (%)	4 (%)	5 (%)
1. Rate your liking.			10	80	10
2. Rate the appropriateness of the price range.		10		90	
3. Rate the appropriateness of the category of the restaurant.			40	20	40
4. Rate the appropriateness of the popularity class of the restaurant.		10	10	70	10
5. Rate the appropriateness of the location of the restaurant.		30		30	40

Remaining four questions were asked to understand in which contextual circumstances participants will prefer to visit the recommended venues. Table 20 presents the answers of each participant to these questions and the predictions, which are calculated from our algorithms for each user.

Table 20. Answers of Questions 6-7-8-9 and Prediction of the Answers

	Answers of Questions 6-7-8-9										Prediction of the Answers									
	M	N	E	Wdays	Wend	Hot	Cold	Sunny	Rainy	Snowy	M	N	E	Wdays	Wend	Hot	Cold	Sunny	Rainy	Snowy
1	0.0	0.0	1.00	0.25	0.75	0.50	0.50	0.70	0.25	0.05	0.00	0.00	1.00	1.00	0.00	1.00	0.00	1.00	0.00	0.00
2	0.0	0.0	1.00	0.50	0.50	0.40	0.60	0.0	0.50	0.50	0.00	0.00	1.00	1.00	0.00	1.00	0.00	0.00	1.00	0.00
3	0.0	0.35	0.65	0.65	0.35	0.70	0.30	0.70	0.30	0.0	0.00	0.00	1.00	1.00	0.00	0.00	1.00	0.32	0.68	0.00
4	0.0	0.20	0.80	0.0	0.100	0.60	0.40	0.90	0.10	0.0	0.00	0.54	0.46	0.82	0.18	0.37	0.63	0.61	0.39	0.00
5	0.40	0.25	0.35	0.30	0.70	0.50	0.50	0.70	0.10	0.20	0.00	0.00	1.00	1.00	0.00	0.50	0.50	0.50	0.50	0.00
6	0.30	0.10	0.60	0.40	0.60	0.60	0.40	0.80	0.15	0.05	0.00	0.50	0.50	0.00	1.00	0.00	1.00	1.00	0.00	0.00
7	0.0	0.50	0.50	0.80	0.20	0.80	0.20	0.60	0.20	0.20	0.00	0.00	1.00	0.48	0.52	0.52	0.48	0.52	0.48	0.00
8	0.0	0.80	0.20	0.75	0.25	0.75	0.25	0.60	0.20	0.20	0.00	0.00	1.00	0.48	0.52	0.52	0.48	0.52	0.48	0.00
9	0.0	0.40	0.60	0.30	0.70	0.60	0.40	0.20	0.50	0.30	0.00	0.69	0.31	0.43	0.57	0.50	0.50	0.56	0.19	0.25
10	0.0	0.40	0.60	0.20	0.80	0.85	0.15	0.50	0.25	0.25	0.00	0.00	1.00	1.00	0.00	0.00	1.00	0.00	1.00	0.00
11	0.0	0.0	1.00	0.25	0.75	0.50	0.50	0.70	0.25	0.05	0.00	0.00	1.00	1.00	0.00	1.00	0.00	1.00	0.00	0.00
12	0.0	0.0	1.00	0.50	0.50	0.40	0.60	0.0	0.50	0.50	0.00	0.00	1.00	1.00	0.00	1.00	0.00	0.00	1.00	0.00
13	0.0	0.35	0.65	0.65	0.35	0.70	0.30	0.70	0.30	0.0	0.00	0.00	1.00	1.00	0.00	0.00	1.00	0.32	0.68	0.00
14	0.0	0.20	0.80	0.0	0.100	0.60	0.40	0.90	0.10	0.0	0.00	0.54	0.46	0.82	0.18	0.37	0.63	0.61	0.39	0.00
15	0.40	0.25	0.35	0.30	0.70	0.50	0.50	0.70	0.10	0.20	0.00	0.00	1.00	1.00	0.00	0.50	0.50	0.50	0.50	0.00
16	0.30	0.10	0.60	0.40	0.60	0.60	0.40	0.80	0.15	0.05	0.00	0.50	0.50	0.00	1.00	0.00	1.00	1.00	0.00	0.00
17	0.0	0.50	0.50	0.80	0.20	0.80	0.20	0.60	0.20	0.20	0.00	0.00	1.00	0.48	0.52	0.52	0.48	0.52	0.48	0.00
18	0.0	0.80	0.20	0.75	0.25	0.75	0.25	0.60	0.20	0.20	0.00	0.00	1.00	0.48	0.52	0.52	0.48	0.52	0.48	0.00
19	0.0	0.40	0.60	0.30	0.70	0.60	0.40	0.20	0.50	0.30	0.00	0.69	0.31	0.43	0.57	0.50	0.50	0.56	0.19	0.25
20	0.0	0.40	0.60	0.20	0.80	0.85	0.15	0.50	0.25	0.25	0.00	0.00	1.00	1.00	0.00	0.00	1.00	0.00	1.00	0.00

Since we know whether the users liked the recommended venues and in which contextual circumstances they prefer to go to the recommended venues, we compare the predicted results with the actual answers of the participants.

The evaluation of the recommendation was measured with precision, recall, and F-1 measures. Table 21 presents the precision, recall, and F-1 scores for 20 participants. Precision value is 0.282, recall value is 0.276, and F1-score is 0.279.

Table 21. Precision, Recall, F-1 Measure for User Study

Precision	Recall	F-1 Measure
0.282	0.276	0.279

In addition to this, the participants made some comments about venues via Twitter messages. These messages are also supported the performance of our algorithm. For instance, some of the participants said similar things that they have already been some of the restaurants with the following expressions:

- Participant 1 (Gender: Male, Age: 37, Education: High School)
    - "I always prefer going these two restaurants that you have recommended."
- Participant 2 (Gender: Male, Age: 38, Education: High School)
    - "I have already been one of the restaurants."
- Participant 3 (Gender: Female, Age: 19, education: High School)
    - "I have visited one of the restaurants before."

- Participant 4 (Gender: Female, Age: 38, Education: Bachelor’s Degree)
    - “I have already visited *venue1* and *venue2*.”

These statements support our ability to accurately predict participants' preferences. Even the participants haven’t visited the recommended venues before, they stated that they like the recommended venues. This result supports that our system has the diversity, novelty, and serendipity dimensions.

CHAPTER 5

CONCLUSION

"Actually,

the best gift

you could give her was

a lifetime of adventures."

Lewis Carroll

In this thesis, a contextually personalized hybrid recommendation model was proposed to integrate user-based and item-based collaborative filtering, content-based filtering together with the contextual information in order to get rid of disadvantages of each approach. For this purpose, visit history of users, venue characteristics (distance, category, popularity, and price classification) and contextual information (weather, season, date, and time of visits) related to each visit were collected from different sources (Twitter, Foursquare, and Weather Underground).

For content-based filtering, the variables; distance, category, popularity, and price classification that have not been used all together in an algorithm before, were used.

Weather condition, season, date, and time of each visit were used cumulatively as the features of venues and also contextual similarities of venues were utilized in the system.

Artificial Neural Network algorithm was applied to determine the weights of each algorithm. Ratings coming from different algorithms (user-based collaborative filtering, item-based collaborative filtering, content-based filtering, rating calculated from the contextual similarities of venues) were used as the predictors of the actual rating. Final ratings were calculated by multiplying the weights coming from the results of neural networks and ratings from different algorithms. This part of the algorithm was named as HybRecSys.

In addition, in order to make more accurate recommendations and make the recommender system contextually personalized, proposed system calculates the probability of visiting a venue in a specific contextual category for each user-venue pair. Decision of the venue recommendation for a specific user is made according to the rule: If the calculated rating is greater than the average rating of the user and the total contextual score is greater than 2, then that venue will be recommended to that user. The final version of the algorithm was named as contextually personalized HybRecSys.

HybRecSys, which is the first version of the algorithm was compared with four algorithms namely; user-based and item-based KNN, biased matrix factorization, and SVD++. Since HybRecSys can generate both rating prediction and binary prediction (whether user will like recommended venues or not), the performance of HybRecSys was evaluated with both RMSE, MAE (for rating prediction), precision, recall, and F1 scores (for binary prediction). K-fold cross validation (k=10) technique was used for data splitting and training and test data sets were generated. HybRecSys outperforms all other algorithms for each metric.

Contextually personalized HybRecSys was compared with five algorithms namely; user-based and item-based KNN, biased matrix factorization, SVD++, and HybRecSys. Those algorithms were evaluated according to the metrics which are used for binary prediction (precision, recall, and F1 score). K-fold cross validation (k=10) technique was used for data splitting and training and test data sets were generated. The extensive experiments were conducted to evaluate the effectiveness of contextually personalized HybRecSys and validated its advantages beyond other algorithms by having the aid of external information. Results showed that contextually personalized HybRecSys outperforms all five algorithms for each evaluation metric.

Data sparsity problem, which occurs from the phenomenon that users rate an insufficient number of venues. Contextually personalized HybRecSys effectively overcomes the challenges arising from data sparsity by modelling the user preferences with the venue category, popularity, price, and contextual variables (time and weather).

In addition, recommending very similar venues to previous visits causes over-specialization. This problem was also lessened by considering the preferences of users from many different aspects and not just getting stuck in only the venue characteristics. Thus the quality of the recommendation was improved and the system gained the novelty, diversity, and serendipity dimensions.

The algorithm partially solves the cold start issue, which can be caused by both a new user and a new item. Even if a new user rates only one venue, the algorithm understands the user preferences from the characteristics of the venue (category, price, popularity). Moreover, the algorithm figures out the contextual circumstances that user prefers that venue. Therefore, by looking these characteristics, the algorithm may recommend a venue to a new user.

The most important feature that distinguishes the developed algorithm from others is the contextually personalization. The venue preferences of the users are affected mostly from contextual variables like time or weather. In addition, these choices may differ from user to user. Proposed algorithm solves this issue and recommends a right venue under right conditions.

Contribution of this study can be summarized as follows:

- As to the knowledge of the authors, this is the first study to use the variables; distance, category, popularity, and price classification together in the content-based filtering algorithm in order to determine content-based similarity of users with embedded venue-related features.
- Weather condition, season, date, and time of each visit were used cumulatively as the features of venues and contextual similarities of venues were utilized in the system.
- Four different approaches (user-based CF, item-based CF, content-based filtering, and contextual recommendation) were applied to develop a hybrid recommender system.
- Artificial neural network algorithm was applied to determine the weight of each algorithm (user-based collaborative filtering, item-based collaborative filtering, content-based filtering and context-aware recommendation) which was used when developing hybrid recommendation system.
- Threshold values determining the user's liking toward a venue were determined separately for each user.
- By calculating the similarity of users according to their various preferences (category, popularity, and price), data sparsity problem was alleviated.
- Over-specialization was lessened by considering the preferences of users from many different aspects and not just getting stuck in only the venue characteristics.

- Cold start problem was partially solved. Even if a new user rates only one venue, the algorithm understands the user preferences from the characteristics of the venue (category, price, popularity). Moreover, the algorithm figures out the contextual circumstances that user prefers that venue. Therefore, by looking these characteristics, the algorithm may recommend a venue to a new user.
- Contextually personalized recommendation was generated by determining which contextual circumstances are more appropriate for specific user-venue pair.

5.1 Business implications

Recommender systems have a great effect on todays' businesses. Recommender systems have been used by many businesses from various sectors such as entertainment (YouTube, Spotify, Netflix, etc.), retailing (Amazon, e-Bay, Hepsiburada, etc.), education (Coursera, Udemy, etc.). The effects of the recommender systems on customers (Hosanagar, Fleder, Lee, & Buja, 2014; Adomavicius, Bockstedt, Curley, & Zhang, 2013), sales (Pathak, Garfinkel, Gopal, Venkatesan, & Yin, 2010), and sales diversity (Fleder & Hosanagar, 2007; Fleder & Hosanagar, 2009) have been investigated previously in the literature.

It is claimed that recommender systems increase sales (Fleder & Hosanagar, 2009; Pathak, Garfinkel, Gopal, Venkatesan, & Yin, 2010). However, it is also emphasized that if the recommender system doesn't have diversity dimension, then it will generally recommend the popular items (Fleder & Hosanagar, 2007; Fleder & Hosanagar, 2009). Therefore, popular items will be more popular, while other's sales are decreasing. At this point, the novelty, diversity, and serendipity characteristics of the recommender systems is very important to recommend also unknown items and decrease this effect. The algorithm proposed in this thesis satisfies these characteristics. Therefore, when the venues are appeared in the database, their visibility will increase among the users.

On the other hand, it is supported that the recommender systems affect the decisions of the customers (Hosanagar, Fleder, Lee, & Buja, 2014; Adomavicius, Bockstedt, Curley, & Zhang, 2013). Although, a user doesn't consider to go any places, just because recommender system suggest it, s/he puts that venue into her/his mind set. Therefore, businesses may use recommender systems to remind the venues that are less popular and forgotten.

Recommender systems also increase cross-selling (Pathak, Garfinkel, Gopal, Venkatesan, & Yin, 2010). This property can be also applicable to the location recommendation systems. For instance, recommending a coffee shop after a dinner restaurant will be more attractive for the customers.

Recommender systems increase customer loyalty and switching cost (Pathak, Garfinkel, Gopal, Venkatesan, & Yin, 2010). When recommender system becomes more accurate about their customers, their customers will not want to switch them.

5.2 Future work

Although the size of the collected data is large, after the filtration it became relatively small. However, this small dataset generated meaningful results and it is possible that it may produce better results with larger datasets. As a future study, it is planned to apply contextually personalized HybRecSys on larger datasets.

When it came to venue opening and closing hours, these were not checked because it is impossible to know whether the venue was open the entire day. Finally, only offline experiments were conducted, and the HybRecSys was evaluated by comparing its performance to the performance of existing algorithms.

The cold start problem may be solved totally in a future study. The system may recommend a venue to a new user by looking at the contextual circumstances and recommend the most preferred venue on these contextual circumstances. In addition, even if a new venue is added to the system, it can be recommended by looking at the content related characteristics. The evaluations that measure the performance of the system’s cold start feature can be performed in the future.

Finally, this algorithm can be embedded into a mobile application and this mobile application can be marketed in mobile application stores. Therefore, its performance can be measured via online experiments.

APPENDIX A

R CODES OF DATA PREPROCESSING

```
#calculating actual rating
uvf3554<-read.csv("uidvidfreq3554.csv")
k=0
maxi=1
mini=100
currUID = uvf3554[1,1]
for(i in 1:nrow(uvf3554)){
   if(uvf3554[i,1]==currUID){
    if(uvf3554[i,3]>maxi){
      maxi=uvf3554[i,3]
    }
    if(uvf3554[i,3]<mini){
      mini=uvf3554[i,3]
    }
    k<-k+1
  }
  else {
    for(j in (i-k):i){
      if(maxi==mini){
        uvf3554[j,4]=1
      }
      else{
        print(j)
        uvf3554[j,4]=(((uvf3554[j,3]-mini)/(maxi-mini))*4)+1
      }
    }
    currUID = uvf3554[i,1]
    k=1
    maxi= uvf3554[i,3]
    mini=uvf3554[i,3]
  }
}
for(j in (i-k):i){
  if(maxi==mini){
    uvf3554[j,4]=1
  }
  else{
    print(j)
    uvf3554[j,4]=(((uvf3554[j,3]-mini)/(maxi-mini))*4)+1
  }
}
colnames(uvf3554)[4] <- "LRating"
write.csv(uvf3554, "RatingFull.csv")
#lat long to radian
Training3554[,6]<-as.numeric(as.character(Training3554[,6]))
```

```
Training3554[,7]<-as.numeric(as.character(Training3554[,7]))

for(i in 1:nrow(Training3554)) {

  Training3554[i,6] <- (Training3554[i,4])*3.14*180

  Training3554[i,7] <- (Training3554[i,5])*3.14*180

  }

colnames(Training3554)[6] <- "LatRad"

colnames(Training3554)[7] <- "LongRad"

#converting lat long to x y z coordinates

Training3554[,8]<-as.numeric(as.character(Training3554[,8]))

Training3554[,9]<-as.numeric(as.character(Training3554[,9]))

Training3554[,10]<-as.numeric(as.character(Training3554[,10]))

for(i in 1:nrow(Training3554)) {

  Training3554[i,8] <- 6371*cos(Training3554[i,6])*cos(Training3554[i,7])

  Training3554[i,9] <-6371*cos(Training3554[i,6])*sin(Training3554[i,7])

  Training3554[i,10]<-6371*sin(Training3554[i,6])

}

colnames(Training3554)[8] <- "X"

colnames(Training3554)[9] <- "Y"

colnames(Training3554)[10] <- "Z"

write.csv(file="Training3554.csv",Training3554)

Training3554Testsiz<-read.csv("Training3554Testsiz.csv")

#finding user centers

n<-0

totalx<-0
```

```
totaly<-0
totalz<-0
for(j in 1:nrow(Training3554Testsiz))
{
  if(Training3554Testsiz[j,1]==Training3554Testsiz[j+1,1])
  { print(j)
    n<-n+Training3554Testsiz[j,11]
    print(n)
    totalx<-(totalx+(Training3554Testsiz[j,8]*Training3554Testsiz[j,11]))
    totaly<-(totaly+(Training3554Testsiz[j,9]*Training3554Testsiz[j,11]))
    totalz<-(totalz+(Training3554Testsiz[j,10]*Training3554Testsiz[j,11]))
  }  else if (n==0)
  {
    print(j)
    Training3554Testsiz[j,12]<-Training3554Testsiz[j,8]
    Training3554Testsiz[j,13]<-Training3554Testsiz[j,9]
    Training3554Testsiz[j,14]<-Training3554Testsiz[j,10]
    n<-0
    totalx<-0
    totaly<-0
    totalz<-0  }   else
    {print(j)
      n<-n+Training3554Testsiz[j,11]
      totalx<-(totalx+(Training3554Testsiz[j,8]*Training3554Testsiz[j,11]))
      totaly<-(totaly+(Training3554Testsiz[j,9]*Training3554Testsiz[j,11]))
      totalz<-(totalz+(Training3554Testsiz[j,10]*Training3554Testsiz[j,11]))
      Training3554Testsiz[j,12]<-totalx/n
      Training3554Testsiz[j,13]<-totaly/n
      Training3554Testsiz[j,14]<-totalz/n
      n<-0
      totalx<-0
      totaly<-0
      totalz<-0
    }
}
Training3554Testsiz[3199,12]<-Training3554Testsiz[3199,8]
Training3554Testsiz[3199,13]<-Training3554Testsiz[3199,9]
Training3554Testsiz[3199,14]<-Training3554Testsiz[3199,10]
write.csv(Training3554Testsiz, "Training3554Testsiz.csv")
#writing user centers to the data
center3554<-read.csv("Center3554.csv")
TestDistance<-read.csv("TestDistance.csv")
for(i in 1:nrow(TestDistance)){
  for(j in 1:nrow(center3554)){
    if(TestDistance[i,1]==center3554[j,1]){
      TestDistance[i,6]=center3554[j,12]
      TestDistance[i,7]=center3554[j,13]
      TestDistance[i,8]=center3554[j,14]
    }
  }
}
#finding distance
for(k in 1:nrow(TestDistance)){
    TestDistance[k,9]<-sqrt((TestDistance[k,3]-TestDistance[k,6])^2+(TestDistance[k,4]-
TestDistance[k,7])^2+(TestDistance[k,5]-TestDistance[k,8])^2)
}
write.csv(visitcoorfull, "TestDistance.csv")
latlong<-read.csv("latlong.csv")
#transforming the unix time to the normal time
latlong[,12]<-as.POSIXct(latlong[,12], origin="1970-01-01")
for(i in 1:nrow(latlong))
{
  q.data<-latlong[i,12]
  q.data<-sub("^[^ ]*", "", q.data)
  latlong[i,13]<-q.data
  latlong[i,14]<-format(latlong[i,12],'%A, %B %d, %Y %H:%M:%S')
  latlong[i,15]<-gsub(",.*$", "", latlong[i,14])
    latlong[i,12]<-gsub(" .*$", "", latlong[i,12])
}
latlong[1,12]<-as.Date(latlong[1,12])
month = as.numeric(format(latlong[1,12], format = "%m"))
month
```

```
colnames(latlong)[15] <- "Day"
colnames(latlong)[14] <- "FullDate"
colnames(latlong)[13] <- "Time"
write.csv(latlong, "latlong.csv")
#matching of weather conditions
weather<-read.csv("weather.csv")
master8019<-read.csv("master8019.csv")
weather<-as.data.frame(weather)
master8019<-as.data.frame(master8019)
master8019[,4]<-as.character(master8019[,4])
weather[,1]<-as.character(weather[,1])
weather[,3]<-as.character(weather[,3])
for(j in 1:nrow(master8019))
{
  for(i in 1:nrow(weather)){
    if(master8019[j,4]==weather[i,1])
    {
      master8019[j,9]=weather[i,2]
      master8019[j,10]=weather[i,3]
    }}}
write.csv(master8019, "master8019.csv")
#writing popularity values to the data
popularity<-read.csv("popularity.csv")
latlong[,2]<-as.character(latlong[,2])
```

```
popularity[,1]<-as.character(popularity[,1])
for(j in 1:nrow(latlong))
{
  for(i in 1:nrow(popularity)){
    print(j)
    print(i)
    if(latlong[j,2]==popularity[i,1])
    {
      latlong[j,18]=popularity[i,9]
          }
  }
}
colnames(latlong)[18] <- "popularity"
write.csv(latlong, "latlong.csv")
#generating user-preference matrices
#time
matrix.deneme <- read.csv(file="Time.csv")
newtable<-table(matrix.deneme$No,matrix.deneme$Time)
aysun<-as.matrix(newtable)
write.csv(file="TimeMatrix.csv",x=aysun[,])
#Day
matrix.deneme <- read.csv(file="Day.csv")
newtable<-table(matrix.deneme$No,matrix.deneme$Day)
aysun<-as.matrix(newtable)
```

```
write.csv(file="DayMatrix.csv",x=aysun[,])
#Category
matrix.deneme <- read.csv(file="Category.csv")
newtable<-table(matrix.deneme$No,matrix.deneme$category)
aysun<-as.matrix(newtable)
write.csv(file="CategoryMatrix.csv",x=aysun[,])
#Wevent
matrix.deneme <- read.csv(file="Wevent.csv")
newtable<-table(matrix.deneme$No,matrix.deneme$Wevent)
aysun<-as.matrix(newtable)
write.csv(file="WeventMatrix.csv",x=aysun[,])
#Season
matrix.deneme <- read.csv(file="Season.csv")
newtable<-table(matrix.deneme$No,matrix.deneme$Season)
aysun<-as.matrix(newtable)
write.csv(file="SeasonMatrix.csv",x=aysun[,])
#distance
Training3554<-read.csv("Training3554.csv")
k=0
maxi=1
mini=100
currUID = Training3554[1,1]
for(i in 1:nrow(Training3554)){
   if(Training3554[i,1]==currUID){
   if(Training3554[i,15]>maxi){
     maxi=Training3554[i,15]
   }
   if(Training3554[i,15]<mini){
     mini=Training3554[i,15]
   }
   k<-k+1
 }
 else {
     for(j in (i-k):i){
     if((maxi-mini)<500){
       Training3554[j,16]=1
     }
     else {
       print(j)
       Training3554[j,16]=(((maxi-Training3554[j,15])/(maxi-mini))*4)+1
     }
   }
   currUID = Training3554[i,1]
   k=1
   maxi= Training3554[i,15]
   mini=Training3554[i,15]
 }
}
```

```
for(j in (i-k):i){
  if((maxi-mini)<500){
    Training3554[j,16]=1
  }
  else {
    print(j)
    Training3554[j,16]=(((maxi-Training3554[j,15])/(maxi-mini))*4)+1
  }
}
Test355[,2]<-as.character(Test355[,2])
Training3554[,2]<-as.character(Training3554[,2])
for(i in 1:nrow(Test355)) {
  for(j in 1:nrow(Training3554)) {
if(Test355[i,1]==Training3554[j,1] && Test355[i,2]==Training3554[j,2]){
  Test355[i,8]=Training3554[j,16]
    }
  }
}
```

APPENDIX B

R CODES OF DEVELOPMENT OF THE RECOMMENDER SYSTEM

```
#user-based collaborative filtering
uservenuematrix<-Training3554matrix
write.csv(file="uservenuematrix.csv",uservenuematrix)
#make placeholder matrix for user sim
userusersim3554<-matrix(NA,
nrow=nrow(uservenuematrix),ncol=nrow(uservenuematrix),dimnames=list(rownames(u
servenuematrix),rownames(uservenuematrix)))
# Lets fill in those empty spaces with cosine similarities
# take transpose because it is faster
tofuservenue<-t(uservenuematrix)
nusers<-ncol(tofuservenue)
for(i in 1:(nusers-1)) {
 # Loop through the columns for each column
 for(j in (i+1):nusers) {
   print(i)
   print(j)
   # Fill in placeholder with cosine similarities
   a <- getCosine(as.matrix(tofuservenue[,i]),as.matrix(tofuservenue[,j]))
   userusersim3554[i,j] <- a
   userusersim3554[j,i] <- a
 }
```

```
}
for(i in 1:nusers) {
  userusersim3554[i,i]<-1
  }
#getCosine Function
function(x,y)
{
  this.cosine <- sum(x*y) / (sqrt(sum(x*x)) * sqrt(sum(y*y)))
  return(this.cosine)
}
write.csv(file="userusersim3554.csv",userusersim3554)
#user prediction by using all users
for(i in 1:nrow(Test355)) {
  print(i)
  for(j in 1:ncol(uservenuematrix)){
    a=colnames(uservenuematrix)[j]
    if(Test355[i,2]==a) {
      rating=0
      totalsim=0
      for(k in 1:nrow(uservenuematrix)){
        if(uservenuematrix[k,toString(Test355[i,2])]!=0){
totalsim=totalsim+userusersim3554[toString(Test355[i,1]),toString(rownames(uservenu
ematrix)[k])]
```

```
        rating=
rating+(userusersim3554[toString(Test355[i,1]),toString(rownames(uservenuematrix)[k]
)]*uservenuematrix[k,toString(Test355[i,2])])
       }
     }
   }
  }
  if(totalsim>1){
  rating=rating/totalsim
  Test355[i,4]=rating}
  else{
    Test355[i,4]=rating
  }
}
colnames(Test355)[4] <- "UCF"
#item-based collaborative filtering
#make placeholder matrix for item sim
itemitemsim3554<-matrix(NA,
nrow=ncol(uservenuematrix),ncol=ncol(uservenuematrix),dimnames=list(colnames(user
venuematrix),colnames(uservenuematrix)))
# Lets fill in those empty spaces with cosine similarities
# Loop through the columns
nvenues<-ncol(uservenuematrix)
for(i in 1:(nvenues-1)) {
  # Loop through the columns for each column
  for(j in (i+1):nvenues) {
    # Fill in placeholder with cosine similarities
    a <- getCosine(as.matrix(uservenuematrix[,i]),as.matrix(uservenuematrix[,j]))
    itemitemsim3554[i,j] <- a
    itemitemsim3554[j,i] <- a
  }
}
for(i in 1:nvenues) {
  itemitemsim3554[i,i]<-1
}
write.csv(file="itemitemsim3554.csv",itemitemsim3554)
itemitemsim3554<-read.csv("itemitemsim3554.csv")
rownames(itemitemsim3554) <-itemitemsim3554[,1]
itemitemsim3554<-itemitemsim3554[,-1] # delete column 1
colnames(itemitemsim3554) = sub("X","",colnames(itemitemsim3554)) #delete X from
column names
#item prediction with all items
Test355[,1]<-as.numeric(Test355[,1])
for(i in 1:nrow(Test355)) {
  print(i)
  for(j in 1:nrow(uservenuematrix)){
    a=rownames(uservenuematrix)[j]
    a<-as.numeric(a)
    if(Test355[i,1]==a) {
      rating=0
      totalsim=0
      for(k in 1:ncol(uservenuematrix)){
        if(uservenuematrix[toString(Test355[i,1]),k]!=0){
totalsim=totalsim+itemitemsim3554[toString(colnames(uservenuematrix)[k]),toString(T
est355[i,2])]
        print(totalsim)
        rating=
rating+(itemitemsim3554[toString(colnames(uservenuematrix)[k]),toString(Test355[i,2]
)]*uservenuematrix[toString(Test355[i,1]),k])
        }
      }
    }
  }
  if(totalsim>1){
  rating=rating/totalsim
  Test355[i,5]=rating}
  else{
    Test355[i,5]=rating
  }
  }
colnames(Test355)[5] <- "ICF"
#content-based filtering
```

```
Content<-read.csv("Content.csv")
Training3554<-read.csv("Training3554.csv")
for(i in 1:nrow(Training3554)){
  for(j in 1:nrow(Content)){
    print(i)
    if(Content[j,1]==Training3554[i,1]&&Content[j,2]==Training3554[i,2]){
      Content[j,11]=1
    }
  }
}
write.csv(file="Content.csv",Content)
Content<-read.csv("Content.csv")
UserCategory<-read.csv("UserCategory.csv")
wide_reshape <- reshape(UserCategory,
                        timevar = "category",
                        idvar = c("uid"),
                        direction = "wide")
#NA'lari sifir yaptik
wide_reshape[is.na(wide_reshape)] <- 0
write.csv(file="UserCategorymatrix.csv",wide_reshape)
#order colums
UserCategorymatrix <- read.csv(file="UserCategorymatrix.csv")
rownames(UserCategorymatrix) <-UserCategorymatrix[,1]
UserCategorymatrix<-UserCategorymatrix[,-1] # delete column 1
```

```
#make placeholder matrix for user sim
usercategorysim3554<-matrix(NA,
nrow=nrow(UserCategorymatrix),ncol=nrow(UserCategorymatrix),dimnames=list(rown
ames(UserCategorymatrix),rownames(UserCategorymatrix)))
# Lets fill in those empty spaces with cosine similarities
# take transpose
tofusercategory<-t(UserCategorymatrix)
nusers<-ncol(tofusercategory)
for(i in 1:(nusers-1)) {
  # Loop through the columns for each column
  for(j in (i+1):nusers) {
    print(i)
    print(j)
    # Fill in placeholder with cosine similarities
    a <- getCosine(as.matrix(tofusercategory[,i]),as.matrix(tofusercategory[,j]))
    usercategorysim3554[i,j] <- a
    usercategorysim3554[j,i] <- a
  }
}
for(i in 1:nusers) {
  usercategorysim3554[i,i]<-1
}
write.csv(file="usercategorysim3554.csv",usercategorysim3554)
#user prediction by using all users
```

```
for(i in 1:nrow(Test355)) {
  print(i)
  for(j in 1:ncol(uservenuematrix)){
    a=colnames(uservenuematrix)[j]
    if(Test355[i,2]==a) {
      rating=0
      totalsim=0
      for(k in 1:nrow(UserCategorymatrix)){
        if(uservenuematrix[k,toString(Test355[i,2])]!=0){
          print(k)
          totalsim=totalsim+usercategorysim3554[i,k]
          rating=
rating+(usercategorysim3554[i,k]*uservenuematrix[k,toString(Test355[i,2])])
        }
      }
    }
  }
  rating=rating/totalsim
  Test355[i,21]=rating
}
colnames(Test355)[21] <- "CategoryCF"
#Popularity
VPRD<-read.csv("VenuePopularityRatingDiscretized.csv")
Content[,2]<-as.character(Content[,2])
```

```
VPRD[j,1]<-as.character(VPRD[j,1])
for(i in 1:nrow(Content)) {
  for(j in 1:nrow(VPRD)){
    if(Content[i,2]==VPRD[j,1]){
      Content[i,11]=VPRD[j,2]
      Content[i,12]=VPRD[j,3]
  }
  }
}
colnames(Content)[11] <- "DRating"
colnames(Content)[12] <- "DPopularity"
write.csv(file="Content.csv",Content)
#popularity sim
#make placeholder matrix for user sim
rownames(DPopularityMatrix) <-DPopularityMatrix[,1]
DPopularityMatrix<-DPopularityMatrix[,-1] # delete column 1
DPopularityMatrix<-read.csv("DPopularityMatrix.csv")
userpopsim3554<-matrix(NA,
nrow=nrow(UserCategorymatrix),ncol=nrow(UserCategorymatrix),dimnames=list(rown
ames(UserCategorymatrix),rownames(UserCategorymatrix)))
# Lets fill in those empty spaces with cosine similarities
# take transpose
tofuserpop<-t(DPopularityMatrix)
nusers<-ncol(tofuserpop)
```

```
for(i in 1:nusers) {
  userpopsim3554[i,i]<-1
}
write.csv(file="userpopsim3554.csv",userpopsim3554)
#user prediction by using all users
for(i in 1:nrow(Test355)) {
  print(i)
  for(j in 1:ncol(uservenuematrix)){
    a=colnames(uservenuematrix)[j]
    if(Test355[i,2]==a) {
      rating=0
      totalsim=0
      for(k in 1:nrow(UserCategorymatrix)){
        if(uservenuematrix[k,toString(Test355[i,2])]!=0){
          print(k)
          totalsim=totalsim+userpopsim3554[i,k]
          rating= rating+(userpopsim3554[i,k]*uservenuematrix[k,toString(Test355[i,2])])
        }
      }
    }
  }
  rating=rating/totalsim
  Test355[i,22]=rating
}
```

```
colnames(Test355)[22] <- "PopCF"
#Price
matrix.deneme <- read.csv(file="Price.csv")
newtable<-table(matrix.deneme$uid,matrix.deneme$Price)
aysun<-as.matrix(newtable)
write.csv(file="PriceMatrix.csv",x=aysun[,])
PriceMatrix <- read.csv(file="PriceMatrix.csv")
rownames(PriceMatrix) <-PriceMatrix[,1]
PriceMatrix<-PriceMatrix[,-1] # delete column 1
#make placeholder matrix for price based user sim
userpricesim3554<-matrix(NA,
nrow=nrow(UserCategorymatrix),ncol=nrow(UserCategorymatrix),dimnames=list(rown
ames(UserCategorymatrix),rownames(UserCategorymatrix)))
# Lets fill in those empty spaces with cosine similarities
# take transpose because it is faster
tofuserprice<-t(PriceMatrix)
nusers<-ncol(tofuserprice)
for(i in 1:(nusers-1)) {
  # Loop through the columns for each column
  for(j in (i+1):nusers) {
    print(i)
    print(j)
    # Fill in placeholder with cosine similarities
    a <- getCosine(as.matrix(tofuserprice[,i]),as.matrix(tofuserprice[,j]))
    userpricesim3554[i,j] <- a
    userpricesim3554[j,i] <- a
  }
}
for(i in 1:nusers) {
  userpricesim3554[i,i]<-1
}
write.csv(file="userpricesim3554.csv",userpricesim3554)
#user prediction by using all users
for(i in 1:nrow(Test355)) {
  print(i)
  for(j in 1:ncol(uservenuematrix)){
    a=colnames(uservenuematrix)[j]
    if(Test355[i,2]==a) {
      rating=0
      totalsim=0
      for(k in 1:nrow(PriceMatrix)){
        if(uservenuematrix[k,toString(Test355[i,2])]!=0){
          print(k)
          totalsim=totalsim+userpricesim3554[i,k]
          rating= rating+(userpricesim3554[i,k]*uservenuematrix[k,toString(Test355[i,2])])
        }
      }
    }
  }
  rating=rating/totalsim
  Test355[i,23]=rating
}
colnames(Test355)[23] <- "PriceCF"
#finding contextual similarity of the venues
ContextRaw8019 <- read.csv(file="8019ContextRaw.csv")
ContextRaw8019<-as.matrix(ContextRaw8019)
Training3554<-as.matrix(Training3554)
for(i in 1:nrow(ContextRaw8019)){
  for(j in 1:nrow(Training3554)){


    if(ContextRaw8019[i,1]==Training3554[j,1] &&
ContextRaw8019[i,2]==Training3554[j,2] && Training3554[j,3]!=0){
      print(i)
      ContextRaw8019[i,10]=1


    }


  }
}
write.csv(ContextRaw8019, "ContextRaw8019.csv")


VenueContextRaw4694 <- read.csv(file="VenueContextRaw4694.csv")
```

```
for(i in 2:nrow(VenueContextRaw4694)){
  if(VenueContextRaw4694[i,1]==VenueContextRaw4694[i-1,1]){
    for(j in 2:25){

    VenueContextRaw4694[i,j]=VenueContextRaw4694[i-
1,j]+VenueContextRaw4694[i,j]
    VenueContextRaw4694[i-1,j]=0
    }
  }

}
write.csv(VenueContextRaw4694, "VenueContext4694.csv")
for(i in 1:nrow(VenueContextRaw4694)){
  if(VenueContextRaw4694[i,1]==VenueContextRaw4694[i+1,1]){

      VenueContextRaw4694[i,2]=NaN

  }

}

#rating matrixi olusturma. write'tan columnlari siraladim excelde sonra tekrar okudum
wide_reshape <- reshape(uidvidfreq,
                        timevar = "vid",
            idvar = c("uid"),
            direction = "wide")
write.csv(file="wide_reshape.csv",wide_reshape)
uidvidratingmatrix <- read.csv(file="uidvidratingmatrix.csv")
#NA'lari 0 yaptik
uidvidratingmatrix[is.na(uidvidratingmatrix)] <- 0

#venue'larin contextual olarak birbirlerine benzerliklerini bulacak
trainingvenuecontext<-read.csv(file="VenueContextSimple4694.csv")
rownames(trainingvenuecontext) <-trainingvenuecontext[,1]
trainingvenuecontext<-trainingvenuecontext[,-1] # delete column 1
#make placeholder matrix for user sim
venuevenueconsim<-matrix(NA,
nrow=nrow(trainingvenuecontext),ncol=nrow(trainingvenuecontext),dimnames=list(row
names(trainingvenuecontext),rownames(trainingvenuecontext)))
# Lets fill in those empty spaces with cosine similarities
# transpose unu aldik çünkü col olunca daha hizli çalisiyor
tofvenuecontextmatrix<-t(trainingvenuecontext)
nvenues<-nrow(trainingvenuecontext)
for(i in 1:(nvenues-1)) {
  # Loop through the columns for each column
  for(j in (i+1):nvenues) {
    print(i)
    print(j)
    # Fill in placeholder with cosine similarities
    a <-
getCosine(as.matrix(tofvenuecontextmatrix[,i]),as.matrix(tofvenuecontextmatrix[,j]))
    venuevenueconsim[i,j] <- a
    venuevenueconsim[j,i] <- a
  }
}
for(i in 1:nvenues) {
  venuevenueconsim[i,i]<-1
}
write.csv(file="venuevenueconsim.csv",venuevenueconsim)
#contextual prediction with all venues
Test355<- read.csv(file="Test355.csv")
for(i in 1:nrow(Test355)) {
  print(i)
  for(j in 1:nrow(uservenuematrix)){
    a=rownames(uservenuematrix)[j]
    if(Test355[i,1]==a) {
      rating=0
      totalsim=0
      for(k in 1:ncol(uservenuematrix)){
        if(uservenuematrix[toString(Test355[i,1]),k]!=0){
totalsim=totalsim+venuevenueconsim[toString(colnames(uservenuematrix)[k]),toString(
Test355[i,2])]
```

```
          rating=
rating+(venuevenueconsim[toString(colnames(uservenuematrix)[k]),toString(Test355[i,
2])]*uservenuematrix[toString(Test355[i,1]),k])
        }
      }


    }
  }
  rating=rating/totalsim
  Test355[i,10]=rating
}
#contextual prediction
old <- read.csv(file="visitcontextold.csv")
Training3554Testsiz <- read.csv(file="Training3554Testsiz.csv")
testvisitcontext<-matrix(NA,
nrow=nrow(Test355),ncol=ncol(old),dimnames=list(rownames(Test355),colnames(old))
)
trainingvisitcontext<-matrix(NA,
nrow=nrow(Training3554Testsiz),ncol=ncol(old),dimnames=list(rownames(Training35
54Testsiz),colnames(old)))
old[,2]=as.character(old[,2])
Training3554Testsiz[,2]=as.character(Training3554Testsiz[,2])
for(i in 1:nrow(Test355)){
  for(j in 1:nrow(old)){
    if(Test355[i,1]==old[j,1] && Test355[i,2]==old[j,2]){
      testvisitcontext[i,1]<-old[j,1]
      testvisitcontext[i,2]<-old[j,2]
      testvisitcontext[i,3]<-old[j,3]
      testvisitcontext[i,4]<-old[j,4]
      testvisitcontext[i,5]<-old[j,5]
      testvisitcontext[i,6]<-old[j,6]
      testvisitcontext[i,7]<-old[j,7]
      testvisitcontext[i,8]<-old[j,8]
      testvisitcontext[i,9]<-old[j,9]
      testvisitcontext[i,10]<-old[j,10]
      testvisitcontext[i,11]<-old[j,11]
      testvisitcontext[i,12]<-old[j,12]
    }
  }
}

for(i in 1:nrow(Training3554Testsiz)){
  for(j in 1:nrow(old)){
    if(Training3554Testsiz[i,1]==old[j,1] && Training3554Testsiz[i,2]==old[j,2]){
      trainingvisitcontext[i,1]<-old[j,1]
      trainingvisitcontext[i,2]<-old[j,2]
      trainingvisitcontext[i,3]<-old[j,3]
      trainingvisitcontext[i,4]<-old[j,4]
      trainingvisitcontext[i,5]<-old[j,5]
      trainingvisitcontext[i,6]<-old[j,6]
      trainingvisitcontext[i,7]<-old[j,7]
      trainingvisitcontext[i,8]<-old[j,8]
      trainingvisitcontext[i,9]<-old[j,9]
      trainingvisitcontext[i,10]<-old[j,10]
      trainingvisitcontext[i,11]<-old[j,11]
      trainingvisitcontext[i,12]<-old[j,12]
    }
  }
}
write.csv(file="testvisitcontext.csv",testvisitcontext)
testvisitcontextpredicted <- read.csv(file="testvisitcontextpredicted.csv")
testvisitcontextpredicted[,2]=as.character(testvisitcontextpredicted[,2])
trainingvisitcontext[,2]=as.character(trainingvisitcontext[,2])
UserStudyContextPredicted<- read.csv(file="UserStudyContextPredicted.csv")
UserStudyContextPredicted[,2]=as.character(UserStudyContextPredicted[,2])
for(i in 1:nrow(UserStudyContextPredicted)) {
  print(i)
  ratingM=0
  ratingN=0
  ratingE=0
  ratingHi=0
  ratingHs=0
  ratingHot=0
  ratingCold=0
  ratingS=0
  ratingR=0
  ratingSw=0
  totalsim=0
  for(j in 1:nrow(trainingvisitcontext)){
    if(UserStudyContextPredicted[i,1]==trainingvisitcontext[j,1]) {
      a=toString(UserStudyContextPredicted[i,2])
      b=toString(trainingvisitcontext[j,2])
      totalsim=totalsim+venuevenueconsim[a,b]
      ratingM= ratingM+(venuevenueconsim[a,b]*as.numeric(trainingvisitcontext[j,3]))
      ratingN= ratingN+(venuevenueconsim[a,b]*as.numeric(trainingvisitcontext[j,4]))
      ratingE= ratingE+(venuevenueconsim[a,b]*as.numeric(trainingvisitcontext[j,5]))
      ratingHi= ratingHi+(venuevenueconsim[a,b]*as.numeric(trainingvisitcontext[j,6]))
      ratingHs= ratingHs+(venuevenueconsim[a,b]*as.numeric(trainingvisitcontext[j,7]))
      ratingHot=
ratingHot+(venuevenueconsim[a,b]*as.numeric(trainingvisitcontext[j,8]))
      ratingCold=
ratingCold+(venuevenueconsim[a,b]*as.numeric(trainingvisitcontext[j,9]))
      ratingS= ratingS+(venuevenueconsim[a,b]*as.numeric(trainingvisitcontext[j,10]))
      ratingR= ratingR+(venuevenueconsim[a,b]*as.numeric(trainingvisitcontext[j,11]))
      ratingSw=
ratingSw+(venuevenueconsim[a,b]*as.numeric(trainingvisitcontext[j,12]))
```

```
    }
  }
  UserStudyContextPredicted[i,3]=ratingM/totalsim
  UserStudyContextPredicted[i,4]=ratingN/totalsim
  UserStudyContextPredicted[i,5]=ratingE/totalsim
  UserStudyContextPredicted[i,6]=ratingHi/totalsim
  UserStudyContextPredicted[i,7]=ratingHs/totalsim
  UserStudyContextPredicted[i,8]=ratingHot/totalsim
  UserStudyContextPredicted[i,9]=ratingCold/totalsim
  UserStudyContextPredicted[i,10]=ratingS/totalsim
  UserStudyContextPredicted[i,11]=ratingR/totalsim
  UserStudyContextPredicted[i,12]=ratingSw/totalsim
}
write.csv(file="UserStudyContextPredicted.csv",UserStudyContextPredicted)
#the final algorithm
testforcontext<-matrix(NA, nrow=12780,ncol=9)
testvisitcontext[,2]<-as.character(testvisitcontext[,2])
testforcontext[,2]<-as.character(testforcontext[,2])
i=0
for(j in 1:355){
  k=1
  while(k<37){
i=i+1
testforcontext[i,1]=testvisitcontext[j,1]
```

```
testforcontext[i,2]=testvisitcontext[j,2]
k=k+1
}
}
contexttekrar <- read.csv(file="contexttekrar.csv")
testforcontext[,3]<-as.character(testforcontext[,3])
testforcontext[,4]<-as.character(testforcontext[,4])
testforcontext[,5]<-as.character(testforcontext[,5])
testforcontext[,6]<-as.character(testforcontext[,6])
contexttekrar[,1]<-as.character(contexttekrar[,1])
contexttekrar[,2]<-as.character(contexttekrar[,2])
contexttekrar[,3]<-as.character(contexttekrar[,3])
contexttekrar[,4]<-as.character(contexttekrar[,4])
k=1
i=1
while(i <= 12780){
if(k<37){
  testforcontext[i,3]=contexttekrar[k,1]
  testforcontext[i,4]=contexttekrar[k,2]
  testforcontext[i,5]=contexttekrar[k,3]
  testforcontext[i,6]=contexttekrar[k,4]
  k=k+1
  i=i+1
}
```

```
  else{
      k=1


  }
}
testvisitcontextpercent <- read.csv(file="testvisitcontextpercent.csv")
testforcontext[,7]<-as.numeric(testforcontext[i,7])
for(i in 1:12780){
  for(j in 1:355){
if(testforcontext[i,1]==testvisitcontextpercent[j,1] &&
testforcontext[i,2]==testvisitcontextpercent[j,2])
{
testforcontext[i,7]=testvisitcontextpercent[j,testforcontext[i,3]]+testvisitcontextpercent[j,
testforcontext[i,4]]+testvisitcontextpercent[j,testforcontext[i,5]]+testvisitcontextpercent[
j,testforcontext[i,6]]
}
  }
}
testforcontext[,8]<-as.numeric(testforcontext[,8])
i=0
for(j in 1:355){
  k=1
  while(k<37){
    i=i+1
    testforcontext[i,8]=testvisitcontextpercent[j,13]
    k=k+1
  }
}
testforcontext[,9]<-as.character(testforcontext[,9])
for(i in 1:12780){
  if(testforcontext[i,7]>2 && testforcontext[i,8]>2){
    testforcontext[i,9]='T'
  }
  else{
    testforcontext[i,9]='F'
  }
}
write.csv(file='testforcontext.csv',testforcontext)
testforcontext <- read.csv(file="testforcontext.csv")
for(i in 1:nrow(testforcontext)){
  for(j in 1:nrow(Test355)){
    if(testforcontext[i,1]==Test355[j,1] && testforcontext[i,2]==Test355[j,2]){
      testforcontext[i,11]=Test355[j,3]
    }
  }
}
for(i in 1:nrow(testforcontext)){
  if(testforcontext[i,11]>2 && testforcontext[i,7]>2){
    testforcontext[i,12]=TRUE
  }
  else{
    testforcontext[i,12]=FALSE
  }
}

#if predicted rating is greater than user average than recommend
useraverage <- read.csv(file="useraverage.csv")
#write user average
for(i in 1:12780){
  for(j in 1:nrow(useraverage)){
  if(testforcontext[i,1]==useraverage[j,1]){
    testforcontext[i,13]=useraverage[j,2]
  }
  }
}
colnames(testforcontext)[13]<-"userav"
for(i in 1:12780){
  print(i)
  if(testforcontext[i,7]>2 && testforcontext[i,8]>testforcontext[i,13]){
    testforcontext[i,14]='T'
  }
  else{
    testforcontext[i,14]='F'
  }
}
colnames(testforcontext)[14]<-"avLike"
for(i in 1:nrow(testforcontext)){
  if(testforcontext[i,11]>testforcontext[i,13] && testforcontext[i,7]>2){
    testforcontext[i,15]=TRUE
  }
  else{
    testforcontext[i,15]=FALSE
  }
}
colnames(testforcontext)[15]<-"avACTUALLike"
```

APPENDIX C

R CODES OF OFFLINE EVALUATION OF THE ALGORITHMS

```
#RMSE
rmse=0
k=0
for(j in 1:nrow(Test355)) {
  if(is.na(Test355[j,5])!=TRUE){
    k=k+1
    rmse=rmse+((Test355[j,3]-Test355[j,5])^2)
  }
 }
rmseitem=sqrt(rmse/k)
print(rmseitem)
#MAE
mae=0
k=0
for(j in 1:nrow(Test355)) {
  if(is.na(Test355[j,5])!=TRUE){
    k=k+1
    mae=mae+(abs(Test355[j,3]-Test355[j,5]))
  }
 }
maeitem=(mae/k)
```

```
print(maeitem)
#precision and recall
k=0
t=0
for (i in 1:nrow(testforcontext)){
  if(testforcontext[i,9]==TRUE){
    t=t+1
    if(testforcontext[i,9]==TRUE && testforcontext[i,12]==TRUE){
      k=k+1
    }
  }
}
print (k)
print(t)
precision=k/t
print(precision)
#recall
k=0
t=0
for (i in 1:nrow(testforcontext)){
  if(testforcontext[i,12]==TRUE){
    t=t+1
    if(testforcontext[i,9]==TRUE && testforcontext[i,12]==TRUE){
      k=k+1
    }}
}
print (k)
print(t)
recall=k/t
print(recall)
#F1 Score
f=(2*precision*recall)/(precision+recall)
print(f)
```

APPENDIX D

SAMPLE SURVEY QUESTIONS OF THE USER STUDY IN TURKISH

Mekan Önerisi Anketi
Boğaziçi Üniversitesi Yönetim Bilişim Sistemleri Bölümü'nde çalışmakta olduğum doktora tezim için hazırladığım anket sorularını aşağıda bulabilirsiniz. Twitter hesaplarınız üzerinden belirlediğim kadarıyla Swarm kullanarak gittiğiniz yerleri paylaşmaktasınız. Ben size, sizin gittiğiniz mekanlar doğrultusunda yeni mekanlar önerdim. Sizden buralarla ilgili birtakım soruları cevaplandırmanızı rica edeceğim. Ankete verdiğiniz cevaplar doktora tezim dışında başka hiçbir yerde kullanılmayacaktır. Çalışmama yardımcı olmayı kabul eden katılımcılara küçük bir hediye olarak herhangi bir filmde ve Cinemaximum'larda geçerli bir kişilik sinema bileti hediye edeceğim. Bu doktora çalışması ya da dolduracağınız anket ile ilgili sorularınız varsa aysun.bozanta@boun.edu.tr adresine gönderebilirsiniz. Şimdiden teşekkür ederim.

1. Limanda Balık'a gittiyseniz beğeninizi derecelendirin. (Eğer gitmediyseniz şu linktengördüğünüz kadarıyla beğeninizi derecelendirin.)	Hiç Beğenmedim	Beğenmedim	Kararsızım	Beğendim	Çok Beğendim
2. Limanda Balık'ın fiyat aralığı 3'tür (1-Cok Ucuz, 4-Çok Pahalı). Bu aralığın size ne kadar uyduğunu derecelendirin.	Hiç Uygun Değil	Uygun Değil	Kararsızım	Uygun	Çok Uygun
3. Limanda Balık'ın kategorisi Deniz Ürünleri Restoranı ve Türk Restoranı'dır. Bu kategorinin size uygunluğunu derecelendirin.	Hiç Uygun Değil	Uygun Değil	Kararsızım	Uygun	Çok Uygun
4. Limanda Balık kullanıcılar arasında düşük popülerlikte bir mekandır. Bu popülerlikteki mekanların size uygunluğunu derecelendirin.	Hiç Uygun Değil	Uygun Değil	Kararsızım	Uygun	Çok Uygun

5. Limanda Balık, Yeni Mahalle Cd No:118, 34450 Sarıyer/İstanbul'dadır. Mekanın konumunun size uygunluğunu derecelendirin.	Hiç Uygun Değil	Uygun Değil	Kararsızım	Uygun	Çok Uygun
6. Limanda Balık'a haftanın hangi zamanlarında gidebileceğinizi iki seçeneğin toplamı 100 olacak şekilde ağırlıklandırın. (Örneğin: Hafta İçi: 25, Hafta Sonu: 75)	Hafta İçi	Hafta Sonu			
7. Limanda Balık'a günün hangi zamanlarında gidebileceğinizi üç seçeneğin toplamı 100 olacak şekilde ağırlıklandırın. (Örneğin: Sabah: 0, Öğle:0, Akşam: 100)	Sabah	Öğle	Akşam		
8. Limanda Balık'a hangi mevsimde gitmekten hoşlanacağınızı toplamı 100 olacak şekilde	İlkbahar-Yaz	Sonbahar-Kış			

ağırlıklandırın. (Örneğin: İlkbahar-Yaz: 80, Sonbahar-Kış:20)					
9. Limanda Balık'a hangi hava koşullarında gitmekten hoşlanacağınızı toplamı 100 olacak şekilde ağırlıklandırın. (Örneğin: Güneşli Açık Havalarda: 50, Yağmurlu Günlerde:50, Karlı Günlerde:0)	Güneşli Açık Havalarda	Yağmurlu Günlerde	Karlı Günlerde		
10. Lütfen yaşınızı giriniz.					
11. Lütfen en son mezun olduğunuz okul türünü seçiniz.	İlkokul	Ortaokul	Lise	Üniversite	Yüksek Lisans/ Doktora

APPENDIX E

SAMPLE SURVEY QUESTIONS OF THE USER STUDY IN ENGLISH

Location Recommendation Survey
The survey questions I prepared for my doctoral dissertation in Bogazici University Management Information Systems Department are presented below. As far as I determined from your Twitter accounts, you are sharing the venues that you have visited with Swarm application. I have recommended new venues according to your past visits. I would kindly ask you to answer the questions about these venues. Your answers will not be used anywhere except my thesis. I will give you a small present, a movie ticket that is valid for all movies in all Cinemaximum, if you accept to participate my study. If you have questions about this study or survey, you may send an e-mail to my e-mail address: aysun.bozanta@boun.edu.tr Thank you in advance.

1. If you have visited "Limanda Balık", please rate your appreciation. (If you haven't gone there, please rate your appreciation according to the link.)	Not Like At All	Not Like	Not Sure	Like	Like Very Much
2. The price class of "Limanda Balık" is 3. (1-Very Cheap, 4-Very Expensive). Please rate the appropriateness level of this price class to you.	Not Appropriate At All	Not Appropriate	Not Sure	Appropria te	Very Appropria te
3. The category of "Limanda Balık" is Sea and Turkish Food Restaurant. Please rate the appropriateness level of this category to you.	Not Appropriate At All	Not Appropriate	Not Sure	Appropria te	Very Appropria te
4. The popularity level of "Limanda Balık" is low. Please rate the appropriateness level	Not Appropriate At All	Not Appropriate	Not Sure	Appropria te	Very Appropria te

of this popularity to you.					
5. The address of "Limanda Balık" is: Yeni Mahalle Cd No:118, 34450 Sarıyer/İstanbul. Please rate the appropriateness level of this address to you.	Not Appropriate At All	Not Appropriate	Not Sure	Appropria te	Very Appropria te
6. Please distribute a hundred point to the categories of a contextual variable according to your tendency of visiting "Limanda Balık" in these day categories. (For Ex.: Weekdays: 25, Weekend: 75)	Weekdays:	Weekend:			
7. Please distribute a hundred point to the categories of a contextual variable according to your tendency of visiting "Limanda Balık" in these times of a day. (For ex.: Morning: 0, Noon:0, Evening: 100)	Morning	Noon	Evening		

8. Please distribute a hundred point to the categories of a contextual variable according to your tendency of visiting "Limanda Balık" in these seasons. (For Ex.: Spring-Summer: 80, Fall-Winter:20)	Spring-Summer	Fall-Winter			
9. Please distribute a hundred point to the categories of a contextual variable according to your tendency of visiting "Limanda Balık" in these weather conditions. (For ex.: Sunny Days: 50, Rainy Days: 50, Snowy Days: 0)	Sunny Days:	Rainy Days:	Snowy Days:		
10. Please write your age.					
11. Please choose your school that you have graduated lastly.	Primary School	Secondary School	High School	University	Master Degree/ Ph.D.

REFERENCES

Adomavicius , G., & Tuzhilin, A. (2015). Context-aware recommender systems. In F. Ricci, L. Rokach & B. Shapira (Eds.), *Recommender systems handbook* (pp. 191-226). Cambridge, MA: Springer.

Adomavicius, G., & Tuzhilin, A. (2005). Towards 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.

Adomavicius, G., Bockstedt, J., Curley, S., & Zhang, J. (2013). Do recommender systems manipulate consumer preferences? A study of anchoring effects. *Information Systems Research, 24*(4), 956-975.

Adomavicius, G., Huang, Z., & Tuzhilin, A. (2008). Personalization and recommender systems. In Z. Chen & S. Raghavan (Eds.), *State-of-the-art decision making tools in the information-intensive age* (pp. 55-100). Catonsville: INFORMS.

Ahmedi, L., Rrmoku, K., Sylejmani, K., & Shabani, D. (2017). A bimodal social network analysis to recommend points of interest to tourists. *Social Network Analysis and Mining, 7*(1), 14.

Aihara, K., Koshiba, H., & Takeda, H. (2011). Behavioral cost-based recommendation model for wanderers in town. In J.A. Jacko (Eds.), *International Conference on Human-Computer Interaction: Towards Mobile and Intelligent Interaction Environments* (pp. 271-279). Berlin, Germany: Springer.

Avazpour, I., Pitakrat, T., Grunske, L., & Grundy, J. (2014). Dimensions and metrics for evaluating recommendation systems. In M.P. Robillard, W. Maalej, R.J. Walker, Th. Zimmermann (Eds.) *Recommendation systems in software engineering* (pp. 245-273). Berlin, Germany: Springer.

Baltrunas, L., & Amatriain, X. (2009). *Towards time-dependant recommendation based on implicit feedback*. Presentation, New York.

Bao, J., Zheng, Y., & Mokbel, M. (2012). Location-based and preference-aware recommendation using sparse geo-social networking data. In I. Cruz & C. Knoblock (Eds.), *Proceedings of the 20th International Conference on Advances in Geographic Information Systems* (pp. 199-2018). New York, NY: ACM.

Baral, R., & Li, T. (2016). Maps: A multi aspect personalized poi recommender system. In S. Sen & W. Geyer (Eds.), *Proceedings of the 10th ACM Conference on Recommender Systems* (pp. 281-284). New York, NY: ACM.

Barranco M.J., Noguera J.M., Castro J., Martínez L. (2012) A Context-Aware Mobile Recommender System Based on Location and Trajectory. In Casillas J., Martínez-López F., Corchado Rodríguez J. (Eds.) *Management Intelligent Systems. Advances in Intelligent Systems and Computing* (pp. 153-162). Berlin, Heidelberg: Springer.

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems, 46*, 109-132.

Breese, J., Heckerman, D., & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In G.F. Cooper & S. Moral (Eds.), *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence* (pp. 43-52). San Francisco, CA: Morgan Kaufmann Publishers Inc.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction, 12*(4), 331-370.

Burke, R. (2007). Hybrid web recommender systems. In P. Brusilovsky & W. Nejdl (Eds.), *The adaptive web.* Berlin, Germany: Springer.

Cao, L., Luo, J., Gallagher, A., Jin, X., Han, J., & Huang, T. (2010). A worldwide tourism recommendation system based on geotaggedweb photos. In P. Tichavský, J. Černocký & A. Procházka (Eds.), *Acoustics Speech and Signal Processing (ICASSP)* (pp. 2274-2277). Piscataway, NJ: IEEE.

Chen, C., Meng, X., Xu, Z., & Lukasiewicz, T. (2017). Location-aware personalized news recommendation with deep semantic analysis. *IEEE Access, 5*, 1624-1638.

Chen, H., Li, X., & Huang, Z. (2005). Link prediction approach to collaborative filtering. *Digital Libraries, 2005. JCDL'05.* In M. Marlino (Eds.), *Proceedings of the 5th ACM/IEEE-CS Joint Conference* (pp. 141-142). Piscataway, NJ: IEEE.

Choi, S., Ko, S., & Han, Y. (2012). A movie recommendation algorithm based on genre correlations. *Expert Systems with Applications, 39*(9), 8079-8085.

Deshpande, M., & Karypis, G. (2004). Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS), 22*(1), 143-177.

Dhake, B., Lomte, S. S., Auti, R. A., Nagargoje, Y. R., & Patil, B. (2014). Lars: An efficient and scalable location-aware. *International Journal of Scientific Research and Education, 2*(11), 2371-2378.

Diao, Q., Qiu, M., Wu, C., Smola, A., Jiang, J., & Wang, C. (2014). Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS). In S. Macskassy & C. Perlich (Eds.), *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 193-202). New York, NY: ACM.

Eck, D., Lamere, P., Bertin-Mahieux, T., & Green, S. (2008). Automatic generation of social tags for music recommendation. In M. I. Jordan, Y. LeCun & S, A. Solla (Eds.), *Advances in Neural Information Processing Systems*, (pp. 385-392). London, United Kingdom: The MIT Press.

Ekstrand, M., Riedl, J., & Konstan, J. (2011). Collaborative filtering recommender systems. *Foundations and Trends® in Human–Computer Interaction, 4*(2), 81-173.

Fleder, D., & Hosanagar, K. (2007). Recommender systems and their impact on sales diversity. In J. MacKie-Mason (Eds.), *Proceedings of the 8th ACM Conference on Electronic Commerce* (pp. 192-199). New York, NY: ACM.

Fleder, D., & Hosanagar, K. (2009). Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Management Science*, *55*(5), 697-712.

Gao, H., Tang, J., Hu, X., & Liu, H. (2013). Exploring temporal effects for location recommendation on location-based social networks. In Q. Yang, I. King & Q. Li (Eds.), *Proceedings of the 7th ACM Conference on Recommender Systems* (pp. 93-100). New York, NY: ACM.

Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, *35*(12), 61-70.

Gorakala, S., & Usuelli, M. (2015). *Building a recommendation system with R.* Birmingham, United Kingdom: Packt Publishing Ltd.

Guo, L., Shao, J., Tan, K., & Yang, Y. (2014). Wheretogo: Personalized travel recommendation for individuals and groups. In M. Gaber & R. Zhang (Eds.), *Mobile Data Management (MDM)* (pp. 49-58). Piscataway, NJ: IEEE.

Gupta, A., & Singh, K. (2013). Location based personalized restaurant recommendation system for mobile environments. In J. L. Mauri, S.M. Thampi, M. Wozniak, O. Marques & D. Krishnaswamy (Eds.), *Advances in Computing, Communications, and Informatics*, (pp. 507-511). Piscataway, NJ: IEEE.

Hamid, M., Naser, M., Hasan, M., & Mahmud, H. (2014). A cohesion-based friend-recommendation system. *Social Network Analysis and Mining*, *4*(1), 176.

Hasegawa, T., & Hayashi, T. (2013). Collaborative filtering based spot recommendation seamlessly available in home and away areas. In T. Matsuo, N. Ishii & R. Lee (Eds.), *Computer and Information Science (ICIS), 2013 IEEE/ACIS 12th International Conference* (pp. 547-548). Piscataway, NJ: IEEE.

Herlocker, J., Konstan, J., & Riedl, J. (2000). Explaining collaborative filtering recommendations. In W. Kellog & S. Whittaker (Eds.), *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work* (pp. 241-250). New York, NY: ACM.

Herlocker, J., Konstan, J., Terveen , L., & Riedl, J. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS), 22*(1), 5-53.

Hiesel, P., Braunhofer, M., & Wörndl, W. (2016). Learning the Popularity of Items for Mobile Tourist Guides. Presentation, Cambridge.

Hosanagar, K., Fleder, D., Lee, D. & Buja, A. (2014). Will the global village fracture into tribes: recommender systems and their effects on consumers?. *Management Science, 60*(4), 805-823.

Isinkaye, F., Folajimi, Y., & Ojokoh, B. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, *16*(3), 261-273.

Knoch, S., Chapko, A., Emrich, A., Werth, D., & Loos, P. (2012). *A Context-Aware Running Route Recommender Learning from User Histories Using Artificial Neural Networks*. Presentation, Vienna.

Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., & Riedl, J. (1997). GroupLens: applying collaborative filtering to Usenet news. *Communications of the ACM, 40*(3), 77-87.

Korakakis, M., Spyrou, E., Mylonas, P., & Peranton, S. (2017). Exploiting social media information toward a context-aware recommendation system. *Social Network Analysis and Mining, 7*(1), 42.

Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In Y. Li (Eds.), *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 426-434). New York, NY: ACM.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, *42*(8), 30-37.

Krishna, P., Misra, S., Joshi, D., & Obaidat, M. (2013). Learning automata based sentiment analysis for recommender system on cloud. In M.S. Obaidat (Eds.), *Computer, Information and Telecommunication Systems (CITS)* (pp. 1-5). Piscataway, NJ: IEEE.

Kuo, M. H., Chen, L., & Liang, C. W. (2009). Building and evaluating a location-based service recommendation system with a preference adjustment mechanism. *Expert Systems with Applications, 36*(2), 3543-3554.

Lekakos , G., & Caravelas, P. (2008). A hybrid approach for movie recommendation. *Multimedia Tools and Applications, 36*(1-2), 55-70.

Levandoski, J., Sarwat, M., Eldawy, A., & Mokbel, M. (2012). Lars: A location-aware recommender system. In X.S. Wang (Eds.), *Data Engineering (ICDE), 2012 IEEE 28th International Conference* (pp. 450-461). Piscataway, NJ: IEEE.

Li, L., Zheng, L., Yang, F., & Li, T. (2014). Modeling and broadening temporal user interest in personalized news recommendation. *Expert Systems with Applications, 41*(7), 3168-3177.

Li, Q., Wang, J., Chen, Y., & Lin, Z. (2010). User comments for news recommendation in forum-based social media. *Information Sciences, 180*(24), 4929-4939.

Li, Q., Zheng, Y., Xie, X., Chen, Y., Liu, W., & Ma, W. (2008). Mining user similarity based on location history. In W.G. Aref, M.F. Mokbel & M. Schneider (Eds.) *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (p. 34). New York, NY: ACM.

Linden, G., Smith, B., & York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing, 7*(1), 76-80.

Liu, D., & Shih, Y. (2004). Integrating AHP and data mining for product recommendation based on customer lifetime value. *Information & Management, 42*(3), 387-400.

Liu, J., Dolan, P., & Pedersen, E. (2010). Personalized news recommendation based on click behavior. In C. Rich & Q. Yang (Eds.), *Proceedings of the 15th International Conference on Intelligent User Interfaces* (pp. 31-40). New York, NY: ACM.

Lops, P., De Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In F. Ricci, L. Rokach & B. Shapira (Eds.), *Recommender systems handbook* (pp. 73-105). Cambridge, MA: Springer.

Majid, A., Chen, L., Chen, G., Mirza, H., & Hussain, I. (2013). A context-aware personalized travel recommendation system based on geotagged social media data mining. *International Journal of Geographical Information Science, 27*(4), 662-684.

Memon, I., Chen, L., Majid, A., Lv, M., Hussain, I., & Chen, G. (2015). Travel recommendation using geo-tagged photos in social media for tourist. *Wireless Personal Communications, 80*(4), 1347-1362.

Mordacchini, M., Passarella, A., Conti, M., Allen, S., Chorley, M., Colombo, G., & Whitaker, R. (2015). Crowdsourcing through cognitive opportunistic networks. *ACM Transactions on Autonomous and Adaptive Systems (TAAS). 10*(2), 13:1-13:29.

Ono, C., Kurokawa, M., Motomura, Y., & Asoh, H. (2007). A context-aware movie preference model using a Bayesian network for recommendation and promotion.

In L. Ardissiono, P. Brna & A. Mitrovic (Eds.), *International Conference on User Modeling* (pp. 247-257). Berlin, Germany: Springer.

Oramas, S., Ostuni, V., Noia, T., Serra, X., & Sciascio, E. (2017). Sound and music recommendation with knowledge graphs. *ACM Transactions on Intelligent Systems and Technology (TIST), 8*(2), 21.

Ozok, A., Fan, Q., & Norcio, A. (2010). Design guidelines for effective recommender system interfaces based on a usability criteria conceptual model: results from a college student population. *Behaviour & Information Technology, 29*(1), 57-83.

Park, M., Hong, J., & Cho, S. (2007). Location-based recommendation system using bayesian user's preference model in mobile devices. In J. Indulska, L.T. Yang, T. Ungerer & J. Cao (Eds.), *International Conference on Ubiquitous Intelligence and Computing* (pp. 1130-1139). Berlin, Germany: Springer.

Park, Y., & Chang, K. (2009). Individual and group behavior-based customer profile model for personalized product recommendation. *Expert Systems with Applications, 36*(2), 1932-1939.

Pathak, B., Garfinkel, R., Gopal, R., Venkatesan, R., & Yin, F. (2010). Empirical analysis of the impact of recommender systems on sales. *Journal of Management Information Systems, 27*(2), 159-188.

Pazzani, M., & Billsus, D. (2007). Content-based recommendation systems. In P. Brusilovsky, A. Kobsa & W. Nejdl (Eds.), *The Adaptive Web* (pp. 325-341). Berlin, Germany: Springer.

Raskutti, B., Beitz, A., & Ward, B. (1997). A feature-based approach to recommending selections based on past preferences. *User Modeling and User-Adapted Interaction, 7*(3), 179-218.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In J.B. Smith, F.D. Smith & T.M. Malone (Eds.), *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work* (pp. 175-186). New York, NY: ACM.

Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to recommender systems handbook. In F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender systems handbook* (pp. 1-35). Cambridge, MA: Springer.

Saraee, M., Khan, S., & Yamaner, S. (2005). Data mining approach to implement a recommendation system for electronic tour guides. In H.R. Arabnia (Eds.), *Proceedings of The 2005 International Conference on E-Business, Enterprise Information Systems, E-Government, and Outsourcing, EEE 2005* (pp. 215-218). Las Vegas, Nevada: CSREA Press.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In V.Y. Shen, N. Saito, M.R. Lyu & M.E. Zurko (Eds.), *Proceedings of The 10th International Conference on World Wide Web* (pp. 285-295). New York, NY: ACM.

Sarwat, M., Levandoski, J., Eldawy, A., & Mokbel, M. (2014). LARS*: An efficient and scalable location-aware recommender system. *IEEE Transactions on Knowledge and Data Engineering, 26*(6), 1384-1399.

Sattari, M., Toroslu, I., Karagoz, P., Symeonid, P., & Manolopoulos, Y. (2015). Extended feature combination model for recommendations in location-based mobile services. *Knowledge and Information Systems, 44*(3), 629-661.

Savage, N., Baranski, M., Chavez, N., & Höllerer, T. (2012). I'm feeling loco: A location based context aware recommendation system. In G. Gartner & F. Ortag (Eds.), *Advances in Location-Based Services* (pp. 37-54). Berlin, Germany: Springer.

Schafer, J., Frankowski, D., Herlocker, J., & Sen, S. (2007). Collaborative Filtering Recommender Systems. In P. Brusilovsky, A. Kobsa, & W. Nejdl (Eds.), *The Adaptive Web* (pp. 291-324). Berlin, Germany: Springer.

Schubert, P., Uwe, L., & Risch, D. (2006). Personalization beyond recommender systems. In R. Suomi, R. Cabral, J.F. Hampe, A. Heikkilä, J. Järveläinen & E. Koskivaara (Eds.), *Project E-Society: Building Bricks* (pp. 126-139). Cambridge, MA: Springer.

Shani, G., & Gunawardana, A. (2011). Evaluating recommendation systems. In F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender systems handbook* (pp. 257-297). Cambridge, MA: Springer.

Shimada, K., Uehara, H., & Endo, T. (2014). A comparative study of potential-of-interest days on a sightseeing spot recommender. In S. Hirokawa (Eds.), *Advanced Applied Informatics (IIAIAAI), 2014 IIAI 3rd International Conference* (pp. 555-560). Piscataway, NJ: IEEE.

Su, X., & Khoshgoftaar, T. (2009). A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence, 2009*(4).

Subramaniyaswamy, V., Vijayakumar, V., Logesh, R., & Indragandhi, V. (2015). Intelligent travel recommendation system by mining attributes from community contributed photos. *Procedia Computer Science, 50,* 447-455.

Szomszor, M., Cattuto, C., Alani, H., O'Hara, K., Baldassarri, A., Loreto, V., & Servedio, V. (2007). *Folksonomies, the semantic web, and movie recommendation*. Paper presented at 4th European Semantic Web Conference, Bridging the Gap between Semantic Web and Web 2.0, Innsbruck, Austria.

Takeuchi, Y., & Sugimoto, M. (2006). CityVoyager: an outdoor recommendation system based on user location history. In J. Indulska, J. Ma, L. T. Yang, T. Ungerer & J. Cao (Eds.), *International Conference on Ubiquitous Intelligence and Computing* (pp. 625-636). Berlin, Germany: Springer.

Tewari, A., Kumar, A., & Barman, A. (2014). Book recommendation system based on combine features of content based filtering, collaborative filtering and association rule mining. In U. Batra (Eds.), *Advance Computing Conference (IACC), 2014 IEEE International* (pp. 500-503). Piscataway, NJ: IEEE.

Trattner, C., Oberegger, A., Eberhard, L., Parra, D., & Marinho, L. (2016). *Understanding the Impact of Weather for POI Recommendations*. Presentation, Cambridge.

Van den Oord, A., Dieleman, S., & Schrauwen, B. (2013). Deep content-based music recommendation. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani & K.Q. Weinberger (Eds.), *Advances in neural information processing systems*, (pp. 2643-2651). Lake Tahoe, Nevada: Curran.

Verbert, K., Duval, E., Lindstaedt, S., & Gillet, D. (2010). Context-aware recommender systems. *Journal of Universal Computer Science, 16*(16), 2175-2178.

Waga, K., Tabarcea, A., & Fränti, P. (2011). Context aware recommendation of location-based data. In M. Voicu (Eds.), *System Theory, Control, and Computing (ICSTCC), 2011 15th International Conference* (pp. 1-6). Piscataway, NJ: IEEE.

Wang, H., Li, G., & Feng, J. (2014). Group-based personalized location recommendation on social networks. In L. Chen, Y. Jia, T. K. Sellis & G. Liu (Eds.), *Asia-Pacific Web Conference* (pp. 68-80). Cham: Springer.

Wang, H., Terrovitis, M., & Mamoulis, N. (2013). Location recommendation in location-based social networks using user check-in data. In C. Knoblock & M. Schneider (Eds.), *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (pp. 374-383). New York, NY: ACM.

Wang, X., Zhao, Y., Nie, L., Gao, Y., Nie, W., Zha, Z., & Chua, T. (2015). Semantic-based location recommendation with multimodal venue semantics. *IEEE Transactions on Multimedia, 17*(3), 409-419.

Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics And Intelligent Laboratory Systems, 2*(1-3), 37-52.

Xiao, B., & Benbasat, I. (2007). E-commerce product recommendation agents: use, characteristics, and impact. *MIS Quarterly, 31*(1), 137-209.

Xu, Z., Chen, L., & Chen, G. (2015). Topic based context-aware travel recommendation method exploiting geotagged photos. *Neurocomputing, 155*, 99-107.

Yang, D., Zhang, D., Yu, Z., & Wang, Z. (2013). A sentiment-enhanced personalized location recommendation system. In G. Stumme (Eds.), *24th ACM Conference on Hypertext and Social Media* (pp. 119-128*).* New York, NY: ACM.

Ye, M., Yin, P., & Lee, W. (2010). Location recommendation for location-based social networks. In D. Agrawal & P. Zhang (Eds.) *Proceedings of The 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems* (pp. 458-461). New York, NY: ACM.

Ye, M., Yin, P., Lee, W., & Lee, D. (2011). Exploiting geographical influence for collaborative point-of-interest recommendation. In W. Ma & J. Nie (Eds.), *Proceedings of The 34th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 325-334). New York, NY: ACM.

Yin, H., Cui, B., Chen, L., Hu, Z., & Zhang, C. (2015). Modeling location-based user rating profiles for personalized recommendation. *ACM Transactions on Knowledge Discovery from Data (TKDD), 9*(3), 19.

Yin, H., Cui, B., Sun, Y., Hu, Z., & Chen, L. (2014). LCARS: A spatial item recommender system. *ACM Transactions on Information Systems (TOIS), 32*(3), 11.

Yin, H., Sun, Y., Cui, B., Hu, Z., & Chen, L. (2013). LCARS: a location-content-aware recommender system. In R. Ghani, T. E. Senator, P. Bradley, R. Parekh & J. He (Eds.), *Proceedings of The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 221-229). New York, NY: ACM.

Yu, C., & Chang, H. (2009). Personalized location-based recommendation services for tour planning in mobile tourism applications. In T. Di Noia & F. Buccafurri (Eds.), *International Conference on Electronic Commerce and Web Technologies* (pp. 38-49). Berlin, Germany: Springer.

Yu, Z., Feng, Y., Xu, H., & Zhou, X. (2014). Recommending travel packages based on mobile crowdsourced data. *IEEE Communications Magazine, 52*(8), 56-62.

Yuan, Q., Cong, G., & Sun, A. (2014). Graph-based point-of-interest recommendation with geographical and temporal influences. In J. Li & X.S. Wang (Eds.), *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management* (pp. 659-668). New York, NY: ACM.

Yuan, Q., Cong, G., Ma, Z., Sun, A., & Thalmann, N. (2013). Time-aware point-of-interest recommendation. In G.J.F. Jones & P. Sheridan (Eds.), *Proceedings of The 36th International ACM SIGIR Conference on Research and Development In Information Retrieval* (pp. 363-372). New York, NY: ACM.

Zhang, J., Chow, C., & Li, Y. (2015). iGeoRec: A personalized and efficient geographical location recommendation framework. *IEEE Transactions on Services Computing*, *8*(5), 701-714.

Zheng, V., Zheng, Y., Xie, X., & Yang, Q. (2012). Towards mobile intelligence: Learning from GPS history data for collaborative recommendation. *Artificial Intelligence*, *184*, 17-37.

Zheng, Y., Burke, R., & Mobasher, B. (2012). Differential context relaxation for context-aware travel recommendation. In C. Huemer & P. Lops (Eds.), *International Conference on Electronic Commerce and Web Technologies* (pp. 88-99). Berlin, Germany: Springer.

Zheng, Y., Zhang, L., Ma, Z., Xie, X., & Ma, W. (2011). Recommending friends and locations based on individual location history. *ACM Transactions on the Web (TWEB)*, *5*(1), 5.

Zheng, Y., Zhang, L., Xie, X., & Ma, W. (2009). Mining interesting locations and travel sequences from GPS trajectories. In J. Quemada & G. Leon (Eds.), *Proceedings of The 18th International Conference on World Wide Web* (pp. 791-800). New York, NY: ACM.

Zukerman, I., Albrecht, D., & Nicholson, A. (1999). Predicting Users' Requests on the WWW. In J. Kay (Eds.), *Proceedings of the Seventh International Conference on User Modeling*, (pp. 275-284). Verlag New York, NJ: Springer.