FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Group Recommender System for Tourism

Vicente Espinha

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Ana Paula Rocha

July 30, 2020

Group Recommender System for Tourism

Vicente Espinha

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Prof. Daniel Castro Silva
External Examiner: Prof. Carlos Abreu Ferreira
Supervisor: Prof. Ana Paula Rocha

July 30, 2020

Abstract

In recent times, Recommender Systems have increased importance to solve information overloading problem with various real-time applications of tourism. These systems appeared as a popular and credible information filtering approach, which is able to make recommendations to the users based on their dynamics and interests. However, bad recommender systems lead to unpleasant consequences, for example, wasteful use of resources, possible loss of costumers, lack of accuracy through due to data problems and upon discovery of the algorithm metrics, service providers can exploit the recommender system. Most of the existing Recommender Systems use collaborative filtering approaches. And, when used for group recommendations, instead of individual, these are generally based on the average or other statistic value derived from the individual ones.

This dissertation focus on the group recommendation problem. It proposes the development of two recommender systems for individual users, using a knowledge-based and a content-based approach, followed by a group recommendation module. To build group recommendations, different aggregation modelling strategies were used, namely the Average, the Average without Misery, the Most Pleasure, the Least Misery and the Multiplicative.

The recommender systems were validated with a proposed metric, named group satisfaction, and several experiments were made with different group sizes and distinct group types. The results of these experiments revealed that for the content-based approach, the Average, the Least misery and the Most Pleasure were the more effective, regardless of group size and group type. For the knowledge-based approach, the Multiplicative strategy is a little more efficient than the others. It is also confirmed that, in general, the knowledge-based approach has a better performance than the content-based, approach.

Keywords: Recommend Systems, Machine learning algorithms, Computational advertising, Group Recommender System, Content-based

Resumo

Nos últimos anos, os Sistemas de Recomendação têm aumentado a importância para resolver problemas de sobrecarga de informações com várias aplicações de turismo em tempo real. Estes sistemas apareceram como uma abordagem popular e confiável de filtragem de informações, capaz de fazer recomendações a pessoas com base nas suas dinâmicas e interesses. No entanto, maus sistemas de recomendação levam a consequências desagradáveis, por exemplo, desperdício de recursos, possível perda de clientes, falta de precisão devido a problemas de dados e após a descoberta das métricas do algoritmo, os provedores de serviços podem explorar o sistema de recomendação. A maioria dos sistemas de recomendação existentes usa abordagens de filtragem colaborativa. E, quando usadas para recomendações de grupo, em vez de individuais, geralmente são baseadas na média ou num outro valor estatístico derivado dos individuais.

Esta dissertação tem como foco o problema de recomendação do grupo. É proposto o desenvolvimento de dois sistemas de recomendação para *users* individuais, usando uma abordagem *Content-based* e uma aboragem *Knowledge-based*, seguida por um módulo de recomendação em grupo. Para construir recomendações de grupo, foram utilizadas diferentes estratégias de modelagem de agregação, a média, a média sem miséria, a multiplicativa, a menor miséria e a mais prazerosa.

Os sistemas de recomendação foram validados com uma métrica proposta, denominada satisfação do grupo, e várias experiências foram feitas com diferentes tamanhos de grupo e tipos distintos de grupo. Os resultados dessas experiências revelaram que, para a abordagem *Content-based*, a média, a menor miséria e a mais prazerosa foram os mais eficazes, independentemente do tamanho e tipo de grupo. Para a abordagem *Knowledge-based*, a estratégia multiplicativa é um pouco mais eficiente que as outras. Também é confirmado que, em geral, a abordagem *Knowledge-based* tem um desempenho melhor do que a abordagem *Content-based*.

Keywords: Sistemas de recomendação, Algoritmos de Machine Learning, Publicidade computacional, Sistema de recomendação de grupo, Content-based

Acknowledgements

I want to express my special thanks to my supervisor Ana Paula Rocha for her permanent availability and support throughout this work. Next, I am thankful to my family, friends and girlfriend for every word of encouragement that helped pushing the project forward and finalising it within the limited time frame. Finally, I am also thankful to Leça da Palmeira and Sporting Clube de Portugal for existing in my life.

Vicente Fernandes Ramada Caldeira Espinha

"Choose life"

Mark Renton

Contents

List of Figures

List of Tables

Abbreviations

AMS	Aggregation Modelling Strategy
CF	Collaborative Filtering
CA	Content-aware
FA	Feedback-aware
IDF	Inverse Document Frequency
MAE	Mean Absolute Error
MCP	Monotonic Concession Protocol
RS	Recommender System
RMSE	Root Mean Square Error
SG	Satisfaction of the group
STDEV	Standard Deviation
UML	Unified Modelling Language

Chapter 1

Introduction

This introductory chapter presents the context, motivation and objectives of the current work. A short description of the structure of the document is also given.

1.1 Context

Recommender systems are algorithms that make relevant suggestions to users to help them have better information about a specific item assisting the on the purchase of the products or so, they don't need to have a satisfactory individual experience of the alternatives to create suggestions of items to the customers. Nowadays, various activities are done in groups, caused by an expansion in social activities, and therefore enhanced the necessity for the improvement of group RSs. Most literature on recommender systems to date concentrates on recommending items to individual users. For example, they may choose a movie for a specific user to watch based on a model of that user's preferences in the past. The difficulty recommender system designers traditionally encountered is how to determine what would be optimal for a particular user. A lot of advancement has been done on this, as demonstrated in Chapter 2. Followed with more and closer ties amongst people in the modern world, recommending items to groups of users have also been a common request.

1.2 Motivation and Objectives

Nowadays, various activities are done in groups, caused by an expansion in social activities, and therefore enhanced the necessity for the improvement of group RSs. For instance, when friends go to the cinema, it is expected to propose a couple of movies for them, regarding their diverse tastes. Presenting recommendations for many users, however, is never a simple assignment, due to the fact that users regularly have heterogeneous preferences, how to determine the group profile continues to be very challenging.

The main goal in this dissertation is to develop an efficient recommender system, using real data, considering first, individual recommendations, and later, generate group suggestions. For this to happen two recommender systems will be created, one more traditionally with a content-based approach and the other with a knowledge-graph approach, but also focusing, preferentially, on the items features. In order to create group recommendations, aggregation modelling strategies will be applied. Once that is done, some evaluation metrics will be exploited in different group sizes and types of group, helping to conclude the efficiency of the proposed recommender systems.

1.3 Dissertation Structure

The remnant of this dissertation is organised as follows. Literature Review are summarized in Chapter 2. This chapter presents the concept of Recommender Systems, describing the main methodologies used in the field, as well as relevant work. In Chapter 3, the description of the problem, a summary of the proposed solution and the details and statistics of the dataset are detailed. Chapter 4 describes the two user recommender systems implemented, a content-based approach and a knowledge-based approach, as well as the group recommendation module. Results and discussion are included in Chapter 5. Lastly, conclusions and future work are described in Chapter 6.

Chapter 2

Literature Review

This chapter introduces the topic of recommender systems, presents the concepts and techniques used in producing recommendations, along with an overview of the frequently used methodologies for these kinds of tasks. This chapter function as a support to base assumptions for the work described in the following sections.

2.1 Background

Recommender Systems (RS) have been broadly investigated in many different domains, with several methodologies used trying to increase the performance efficiency of the suggestion generation method. The main techniques and approaches presented in the literature on recommender systems are described next.

2.1.1 Recommender Systems

This dissertation concentrates on a particular type of machine learning algorithms, named recommender systems. Recommender systems help people make decisions in complex information spaces, where a high amount and diversity of data is generally available. The current work will focus on recommender systems used in social media networks. Recommender systems suggest to the user items that may value him/her, based on knowledge about the user, about other users and the space of possible items. A news service, for example, might remember the articles a user has read, and the next time he/she visits the site, the system can recommend new articles based on the ones that have been read before [48].

Collecting data is one of the most important aspects of a recommender system. Some systems fail due to the following problems related to data collection issues:

- *Cold Start* - It indicates the lack of data about a user who registered on the platform relatively recently. In this situation, the systems have insufficient information about the user, due to the absence of historical activity.

- *Malicious Rating* - This problem emerges as a result of the creation of users with the sole purpose of specifying a rating to particular items. These ratings do not correspond to users' real tendencies, they are biased ratings. These classifications, when extracted and used in future recommendations, inadequately propagate users' preferences, compromising the model's outcome.

- *Sparseness* - It occurs when the users have little or no historical interaction with the system. The previous problem is a subtype of this one.

An important concept presented in almost all recommender systems models is the concept of profiling, that is, the identification of specific information about someone or something, like characteristics or behavioral patterns, to make generalizations about it. In this section, after describing the concept of profiling used in the context of recommender systems, a summary of the different models of RSs will be presented, more specifically: Collaborative Filtering, Content-based Filtering, Demographic, Knowledge-based and Community-based [63].

2.1.1.1 Profiling

After analyzing the large amount of information that a specific domain can hold, it can became a difficult task to identify the items that a certain user may be interested in. To help in this task, it is beneficial to create profiles, either with items or users, thus reducing the complexity and volume of data. The creation of profiles for both users and items will be discussed in the following sections.

Item Profiling

Item profiling is a process of selecting the characteristics related to the items under analysis. Item profiling studies then the identification of relevant characteristics, that will used to create generalisations. Using the context of music recommendation as an example, users reveal preferences about artists, bands, music genre, and these characteristics will be used in creating the profile of a music [35]. Figure 2.1 shows a simple example of a item profile.

User Profiling

User profiling is the process of collecting, extracting and describing the characteristics of particular users. This procedure has enormous significance in the results of the recommender system, so its reliable execution is crucial. Users' information can be related to personal data as well as information regarding evaluation of items. This latter can be extracted explicitly, where the user is asked to evaluate or select some items, or implicitly, monitoring the users' actions based on the interactions Figure 2.2 shows a simple example of a user profile [45].

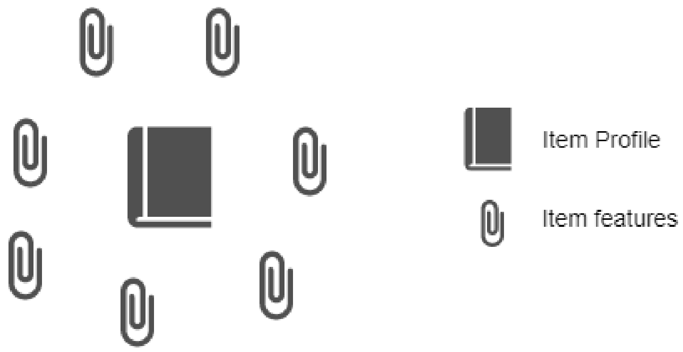

Figure 2.1: Illustration of Item Profiling. Adapted from [50]

2.1.1.2 Collaborative filtering

Collaborative Filtering (CF) recommends items to an user based on the evaluations of other users. This type of recommendation algorithms measures similarity by looking at the rating records of users with a similar taste, and then recommends items with high similarity [52]. Figure 2.3 illustrates the collaborative filtering method used in a book recommender system. Collaborative filtering is the predominant method used in existing recommender systems and can be found in the recommendation algorithms of big web platforms like Amazon, YouTube, Facebook, etc.

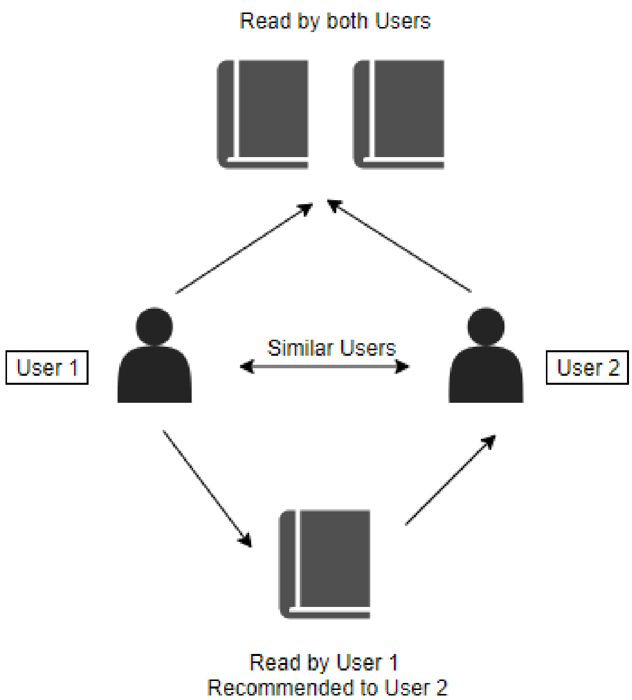

Figure 2.3: Collaborative filtering recommendation. Adapted from [43]

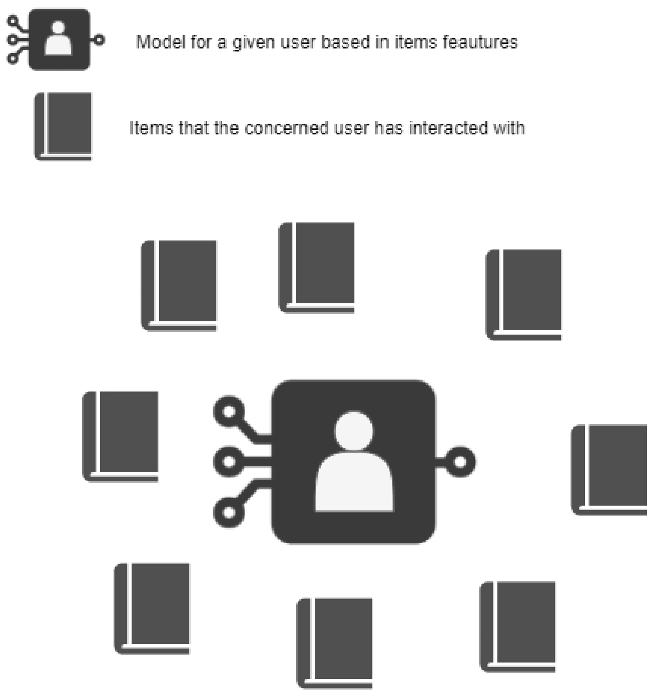

Figure 2.2: Illustration of User Profiling. Adapted from [50]

Collaborative filtering uses the evaluations of other users to recommend items to a specific user. There are three different types of collaborative filtering, namely memory-based, model-based, and hybrid collaborative filtering.

Memory-based CF can be further divided into: i) user-user CF, which focus on finding other users with similar past rating behaviour and then uses those patterns to generate recommendations, and ii) item-item CF, which uses the rating models of items, that is, the system identifies other users who like and dislike the same items, and then finds items that have already been rated by those users, but not by the current user and then recommends or not those items[14, 28, 51]. The main advantages are simplicity in applying this technique and updating the database, since it uses the whole database to build a prediction, and the quality of predictions is rather satisfying. The main drawbacks of using a memory-based approach are that it needs plenty of memory, slowing the system, and it tends to overfit the data.

Model-based CF makes recommendations with a pattern of user ratings which is developed beforehand. These models can be developed by different machine learning algorithms, such as Bayesian network, matrix factorization, clustering, and rule-based approaches[6]. These machine learning algorithms are used on training data, which are datasets used to “prepare” the model for real-life data[63]. The main advantages are scalability and better time-consuming in comparison to memory-based systems. It performs a better time-consuming because performs queries to the model instead of the whole dataset. The main drawback of the model-based approach is the inflexibility since building a model is often a time and resource-consuming process. The fact that it does not use the complete information available, it might not get predictions as accurate as memory-based.

Lastly, Hybrid CF systems are a mix of memory and model based CFs or yet another recommender system type. These systems are used to eliminate some of the problems that different approaches cause. The drawback of this system is that they tend to be more elaborate and expensive [7]. Fig. 2.4 summarizes the mentioned CF approaches.

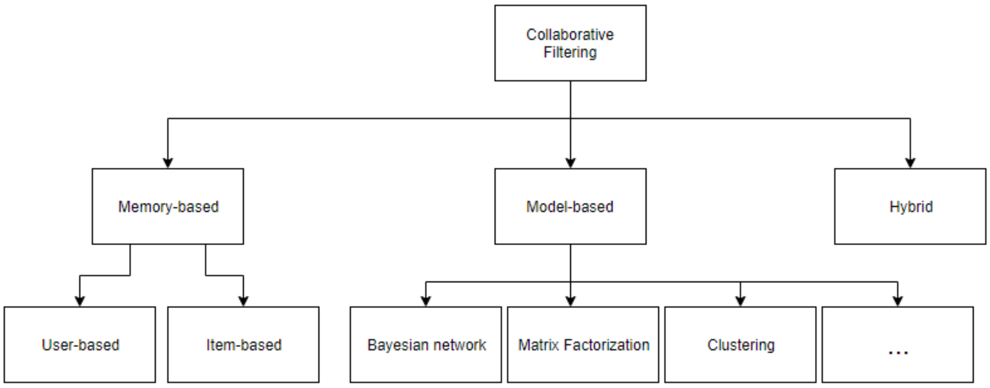

Figure 2.4: Overview of Collaborative filtering approaches

Notwithstanding the benefits presented by the CF algorithm, data sparsity and cold start are the main concern of the conventional recommender systems.

2.1.1.3 Content-based Filtering

Unlike collaborative filtering that solely depends on the user-item correlations, content-based RSs learn to suggest items that are similar to the items that the user positively rated previously. The content-based approaches utilize additional information about users or items, for instance, age and job for users, and category or other domain characteristics for items. Most of the time, a user profile is used, which will save the types of items that are of the user's liking. Using, as example, restaurants as items, characteristics such as type of food or type of establishment are some of the components that will serve to define the user's profile, bearing in mind the person's evaluations. Then future items will be compared to the user profile to conclude which items to suggest [23].

Content-based approaches experience far less from the cold start problem than collaborative methods: new items or users can be defined by their features and proper recommendations can be done for these new entities. Only new items or users with previously unseen characteristics will experience from this drawback, although once the system is old enough, this has a low probability to occur [50].The concept of content-based techniques is to try to produce a model, based on the available characteristics of items and users, which describe the perceived user-item interactions. Figure 2.5 shows a overview of the content-based techniques paradigm.

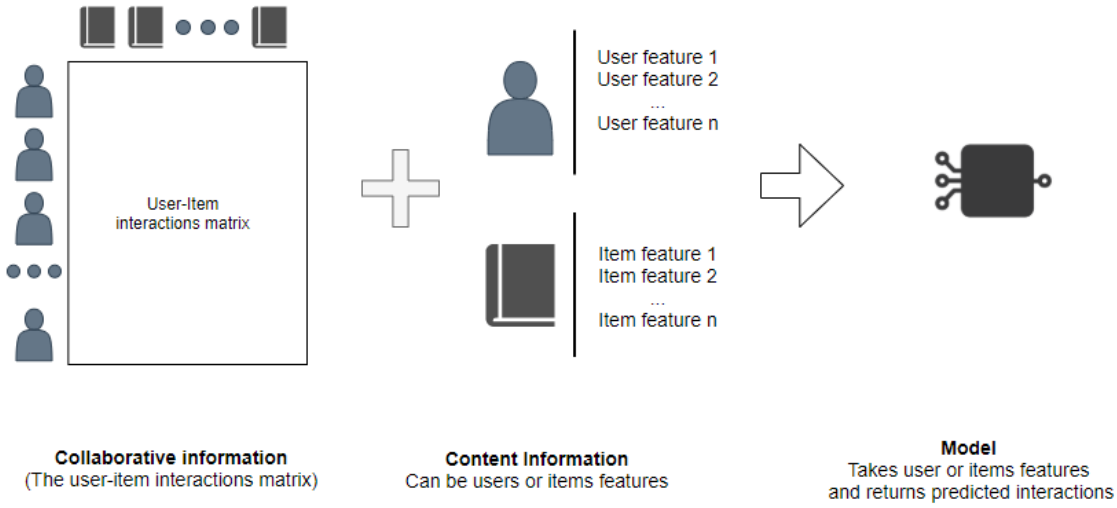

Figure 2.5: Overview of the content-based techniques paradigm (figure adapted from [50])

2.1.1.4 Demographic

Demographic-based recommender systems suggest items contingent on the demographic profile of the user. This algorithm gathers data in the form of attributes of a person and uses this to create demographic classes [46]. In this way, different suggestions ought to be given for dissimilar demographic niches.

2.1.1.5 Knowledge-based

Knowledge-based recommender systems make recommendations based on some kind of inference, using knowledge graphs. Knowledge graphs are utilized to apprehend technical information defining configurable items and previous solutions. Any entities examined correspond to vertices in a directed graph with labelled edges [27]. Figure 2.6 presents an model of a knowledge graph in the cinema domain. A lot of knowledge graphs being used are far from completion in terms of missing many facts about the items available. Consequently, one important task many times implemented in knowledge graphs subsists on predicting new edges given the remaining connectivity model. This difficulty in link prediction is also mentioned in literature as knowledge graph completion [27].

These systems infer how a distinct item reaches a special user need. This means that it can take a conclusion about the relationship between the need and a possible recommendation [7]. A very simple example is a search query on Google where the user types in a question and Google delivers the most relevant recommendations with a solution to the question[63].

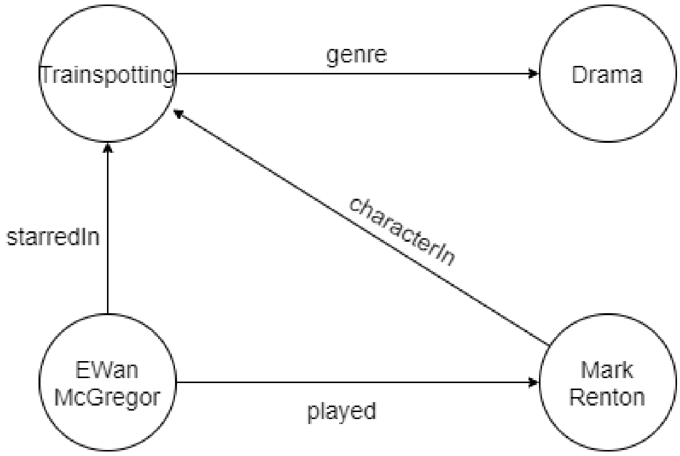

Figure 2.6: Knowledge Graph Example in the Cinema Domain

2.1.1.6 Community-based

Community-based recommender systems use as the main concept the relationships between users [2]. These systems exploit relationships and trust between users to make better recommendations, as users tend to trust recommendations from friends, or friends from friends, more other users [54].

2.1.2 Validation

Recommendation includes the implicit task of predicting how much a user likes some items. In prediction problems, the validation of the results must be defined in the context of the problem. There are two main types of problems, classification and regression, for which there is a set of proper metrics that allow validating the solutions obtained. In a validation process, it is always necessary a classification or regression model and a set, such as clustering, and, in a recommendation, makes it more difficult to obtain a methodology that allows us to validate a priori, allowing to conduct an evaluation. This is due to the meaning of what is a good or acceptable solution can be defined as incomplete as a result of the characteristics intrinsic to the problem itself [18].

2.1.2.1 Classification

Classification is used in problems where we want to distinguish between two or more classes. In this context, it is possible to define what is a correct or incorrect classification of items that are pre-classified in a validation data set. A confusion matrix is commonly used in evaluating the performance of the model. A confusion matrix includes the counting of itens of the dataset that belongs to each of the identified classes, organized in a matrix, where one of the dimensions (for instance, rows) refers to predicted classes, and the other dimension (for instance, columns) refers to true classes.

The confusion matrix is represented in the figure 2.7. The numbers along the major diagonal represent the correct decisions made, and the numbers of this diagonal represent the errors (the confusion) between the various classes [16]. This figure also presents the equations of the most commons metrics that can be calculated from the confusion matrix. These metrics include the precision, accuracy, recall, ratio of false positives (fp rate), ratio of true positives (tp rate) and F measure (calculated as the weighted harmonic mean of precision and recall)

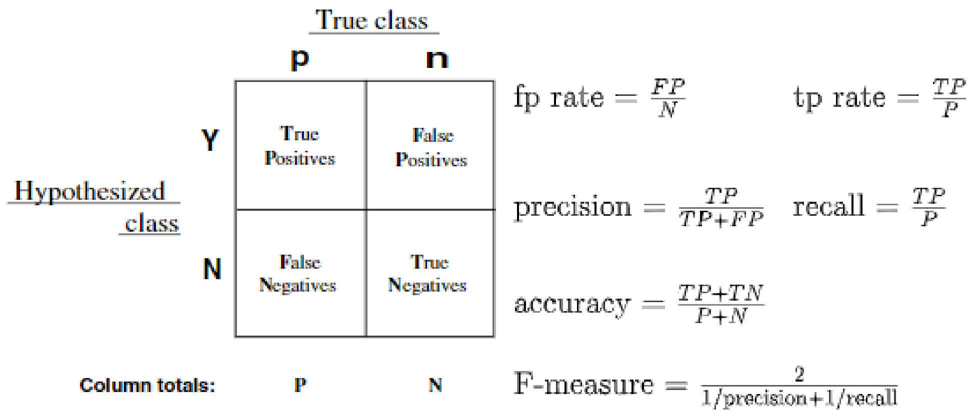

Figure 2.7: Confusion matrix and common performance metrics calculated from it [16]

2.1.2.2 Regression

The regression problem consists in predicting a numerical, generally decimal, variable. As there is always an error associated with the prediction of a numerical variable, it does not make sense to use the correctness rate for validation, as it was the case in classification. In regression problems, it is common to use the root mean square error (RMSE) and root absolute error (MAE) as metrics, as defined in equations 2.1 and 2.2 [35, 49].

$$RMSE = \sqrt{\frac{\sum_{|user,poi}(ActualRatings_{user,poi} - PredictedRatings_{user,poi})|}{n}} \tag{2.1}$$

Where:

n :number of elements in dataset
$RMSE$: root mean square error
poi : point of interest

$$MAE = \frac{1}{N_{users}} \sum_{user=1}^{N_{users}} (Actual_{ratings}(poi_{user}) - Predicted_{ratings}(poi_{user})) \tag{2.2}$$

Where:

N_{users} :number of elements in dataset
MAE : root absolute error
poi : point of interest

This errors measure allows evaluating how close the predictions obtained by the real model are, using a set of validation data.

2.1.2.3 Cross Validation

Cross-validation is a validation technique that allows the comparison between several methodologies. It provides an estimate of the performance of each methodology, for any performance metric. This technique consists of dividing the training dataset into k parts and iterating using k-1 parts for training and 1 part to validate, being calculated the performance metric for each of these k iterations, as shown in the figure 2.8. In the end, this performance metric for each of the k parts is aggregated, using the average to measure the overall quality. In data where there is a time dependency, caution must be taken not to use future information to validate with past data, k-folds must be defined taking this restriction into account [55].

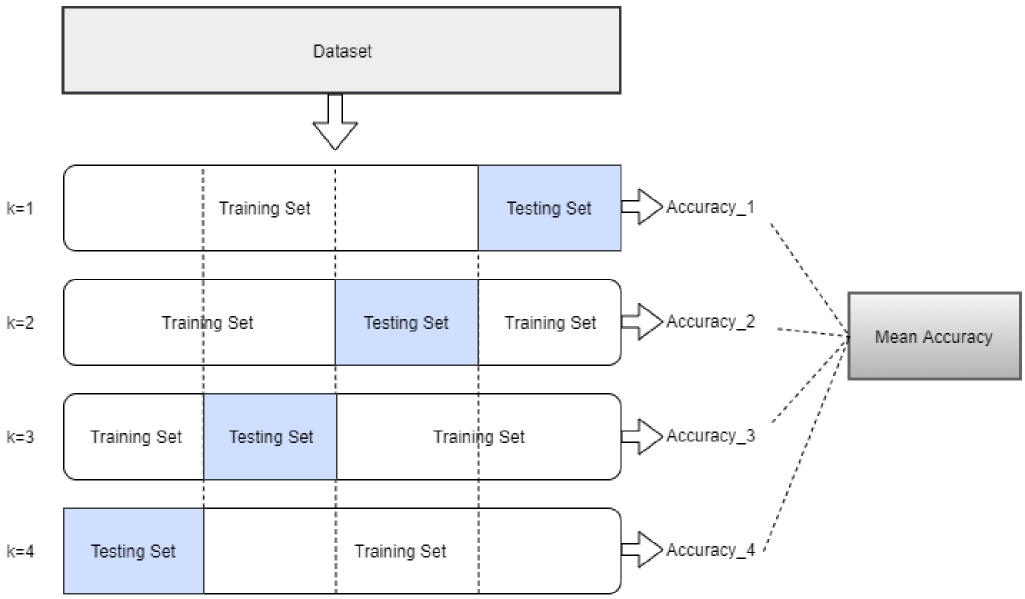

Figure 2.8: Dataset division in training set and testing set (figure adapted from [55])

2.1.3 Group Recommendations

Groups can be categorised as heterogeneous groups and homogeneous groups according to their composition. Heterogeneous groups consist of members with different interests, whilst the homogeneous group consist of people with similar interests. The vast majority of the existing approaches for group recommender systems can be subdivided into :

- Creating a group profile by aggregating the individual member profiles and recommending items based in the group profile [31].
- Creating the personalised recommendations to the members of the group and then aggregating them into a single group recommendation [30].

Group recommender systems can also be categorised according to the types of groups to which the system suggests. Groups can be predominantly of the three types based on the interactions between the members of the group [4].

- **Established Group:** a group who decide to be a member of the group contingent on the shared same interests[31].
- **Occasional Group:** a group who complete some activity occasionally collectively [19].
- **Random Group:** a group of people that experience a situation at a singular time without any connection among them [11].

2.1.3.1 Group Modelling Strategies

The following strategies for calculating the group recommendation are some of the most used to combined the individual preferences in a unique model that represents the group [9, 25]:

- **Average** — uses the average of the members of a group preference feedbacks for the group's preference feedback and then creates the group profile;
- **Multiplicative Utilitarian** — multiplicates individual ratings for each item to create the group profile;
- **Borda Count** — every item receives some points, based on the place in the list of any user, for example, the least favourite receives 0 points, the next one gets one more point than the one before and if there are items in the same position, points are distributed;
- **Copeland Rule** — sort the items based on their Copeland Index, which is the number of times in which an alternative surpasses the others, minus the number of times it succumbs;
- **Plurality Voting** — all user votes for their preferred choices. If more than one alternative demands to be chosen, the items that got the most number of votes are chosen;
- **Approval Voting** — all the items a user likes receive a point and the group rating for a specific item is the sum of the individual votes;
- **Least Misery** — the most lower rating denoted for an item by either of group members creates the group rating. It's mostly used to be certain that each member is pleased;
- **Most Pleasure** — the group rating is defined by the greatest rating represented for an item by a group member;

- **Average without Misery** — starts by aggregating feedbacks of preference from members by using the Average strategy and next executes suggestions with the matrix factorization method;
- **Fairness** — users can be advised an item they do not desire, taking into account that they too get suggested an item they prefer;
- **Most Respected Person** — chooses the items based on the preferences of the most respected person.

2.1.3.2 Individual Satisfaction

In the most simplistic satisfaction function, the impact of an item was taken to be its rating. Three factors were found to improve satisfaction functions: inclusion of low ratings, normalization, and a quadratic ratings, which makes the distinction between ratings of say 9 and 10 more important than that between 5 and 6 [38]. In [39], authors studied several satisfaction functions, which would determine the satisfaction of a user with a new item *i* after having seen a sequence *items* of items, and the one that performed best is identified by equation 2.3 [38], where *Sat(item)* is the satisfaction of item *item* for the user.

$$Sat(items+<i>) = \frac{\delta \times Sat(items) + Impact(i, \delta \times Sat(items))}{1+\delta} \tag{2.3}$$

With the impact on satisfaction of new item *i* based on existing satisfaction *s* described as in 2.4.

$$Impact(i,s) = Impact(i) + (s - Impact(i)) \times \varepsilon, for \quad 0 \leq \varepsilon \leq 1 \quad and \quad 0 \leq \delta \leq 1 \tag{2.4}$$

The parameter δ expresses satisfaction declining over time (with $\delta = 0$ past times have no influence and $\delta = 1$ there is no reduction), and the parameter ε expresses the influence of the user's satisfaction after experiencing previous items on the impact of a new item [38].

2.1.3.3 Satisfaction in aggregation strategies

After predicting how satisfied each group member is after a sequence of items, it would be good to utilize this model to refine using the group aggregation strategies. For example, the aggregation method could seek to please the group's user who is least satisfied with the sequence of items chosen so far. The following strategies can be found in existing work:

- **"Strongly Support Grumpiest strategy"** selects the item which is most preferred by the more unhappy member [38].

- **"Weakly Support Grumpiest strategy"** picks the items that are very preferred by the more unhappy member, for example, items with ratings of 8 and higher [38].
- **"Weighted strategy"** specifies weights to users according to their happiness and then utilizes a weighted model of a conventional aggregation method [38].

2.2 Related Work

This section identifies relevant work about recommender systems, either in individual and group recommendation.

2.2.1 Individual Recommender Systems

Lately, recommender systems has been extensively scrutinised in multiple application fields with various methodologies to increase the performance efficiency of the recommendation production method [49].

2.2.1.1 Collaborative filtering

The idea of collaborative filtering was first established by Goldberg et al. [22] to identify similar users and nowadays it is a popularly utilised technique in recommender systems. Later, GroupLens [32] utilized the collaborative filtering method to group the news articles automatically in Usenet. Herlocker et al. [26] show the measuring factors to assess the quality of the rating prediction mechanism of RS. One of the most important examples of collaborative filtering used in real-world applications is Spotify [21], which uses this approach in its RS.

2.2.1.2 Content-based filtering

In this subsection, three examples of Content-based filtering RSs will be presented.

- **Learning Intelligent Book Recommending Agent (LIBRA)** is a content-based recommender system that uses machine learning methodology to extract semi-structured text data from the web, to make book recommendations [42].
- **Content-Based Music Recommendation System (CBMRS)** is an recommender system that implements the concept of using not only the available data of the user's preferences but also dynamic data from the environment, to obtain music-related recommendations. The method includes considering parameters such as user's pulses and mood, the weather, temperature and the user's location and builds a model based on how these parameters influence the user's preferences in selecting music. By joining the content-based filtering techniques applied for music recommendation and these parameters that are perceived to influence the user's selection of music this system tries to produce more accurate recommendations than the traditional content-based methods [62].

- **Personalized Recommender System (PRES)** is a content-based recommender system that builds dynamic hyperlinks for a web site that includes a set of advises about DIY (do-it-yourself) home renovation. The most significant characteristic of this type of information is the fact that a specific problem is only appealing to a user for a short time. Once an improvement has been conducted, the user will lose interest in that issue. Therefore, PRES should train the user model to analyse unseen items onto a class relevant to the user or a class irrelevant to the user [58].

2.2.1.3 Knowledge-based Systems

This subsection presents some work in the domain of Knowledge-based recommendations, including Knowledge Graphs.

- **Collaborative Knowledge base Embedding (CKE)** is a recommendation framework used to integrate collaborative filtering with items' different semantic representations from the knowledge base. Excluding the network structure information, it considers items' textual and visual content. (e.h., movie's poster).There was designed three embedding components to automatically extract items' semantic representations from the knowledge base's structural content, visual content and textual content. After extracting, the items' semantic representations from the knowledge base is integrated into collaborative filtering resulting in CKE [61].

- **Neural Encoders Combined with Tensor Decompositions for Recommendations (NECTR)** is a Hybrid RS based on the combination of autoencoders and tensor factorizations. The basic concept is to build a graphical, multi-relational knowledge base, which includes technical information regarding items as well as user-item interactions history [27].

- A Word2Vec model is proposed in [57], where the underlying idea of the algorithm is to embed the entities and relationships of the knowledge graph into a low-dimensional vector space. Afterwards, transform the entities and the relations into vector representations. Therefore, the similar entities in the Knowledge Graph are also similar representations in the vector space, which means that semantic similar entities are also similar in vector space.

- Kazienko et al. [29] use a multi-layered graph to represent the information necessary to compute recommendations. Each information type is represented on a separate layer, where each layer contains a graph representing homogeneous information [24].

2.2.1.4 Trust-aware Systems

Since the collaborative filtering based RS have data sparsity problems and is exposed to malevolent attacks [12, 56], combining trust information helps to overcome these problems. This is because people used to get items recommended by family members, friends and co-workers whom they consider their opinions the most. Some approaches like trusted k-nearest neighbours or several metrics to alleviate rating sparsity were developed to try to fix the problems previously identified [34, 36].

2.2.1.5 Swarm intelligent algorithms in recommender systems

As a result of the deficiency of the traditional clustering algorithm to achieve optimal results on considerable-sized applications, several investigators have included swarm intelligence algorithms in multiple RSs for user clustering. These algorithms derive from the characteristics of biological systems to present a greater convergence standard [49]. Generally, every clustering algorithm has its weaknesses and consequently combining outcomes of hybrid clustering algorithms assist to produce more promising outcomes [13, 59].

2.2.1.6 Travel recommender system

Travel Recommender system by using swarm intelligence for user clustering algorithms and generating customized recommendations have increased meaningful attention from many investigators [47, 8, 20]. The main purpose of the travel recommender system is to answer people' requirements. Several variables like user preference, weather, interest, travel costs, time of the day and companion are acknowledged as some restrictions for making suggestions. Regarding the target users' demands, the important characteristics of the travel recommender system are aligned to produce the appropriate points of interests [49].

2.2.1.7 Co-training method

One of the most daunting issues of each recommender system is to take care of the rating sparsity and cold start. In recent times, some investigators have used co-training in the RS to supplement their data, by having two conditionally independent views of the data and it implies that every example is defined using two different feature sets that present distinct and complementary information regarding the situation [3, 49].

Table 2.1 summarizes the individual recommender systems previously described, enumerating the methodologies used and the domain of application.

Recommender System	Methodologies	Domain
Collaborative Filtering for Usenet News [32]	Collaborative Filtering	News
LIBRA [42]	Content-base Filtering	Books
CBMRS [62]	Content-base Filtering	Music
PRES [58]	Content-base Filtering	"DIY"
CKE [61]	Knowledge-based	Movies and Books
NECTR [27]	Knowledge-based	Industrial
[57]	Knowledge-based	Web Search

Table 2.1: Overview of Recommender Systems.

2.2.2 Group Recommender Systems

The scenarios provided below differ in various dimensions, including individual preferences are known versus developed over time, recommended items are experienced by the group versus presented as options, the group is passive versus active and recommending a single item versus a sequence.

2.2.2.1 Based on aggregation models

In this section, the scenarios underlying some of the best-known group recommender systems are presented, as well as some of the more recent ones:

- **POLYLENS** [44] is a group recommender extension of MOVIELENS, which recommends movies considering the individual ratings and social media filtering. POLYLENS makes it possible for users to form groups and ask for group recommendations. This RS utilizes the Least Misery Strategy, taking in consideration that it is usually a small group of people that watch a movie together, taking advantage of the fact that in small groups, the overall happiness will be as great as the happiness of the least happy person.

- **Collaborative Advisory Travel System** (CATS) [41] helps users to decide on a joint holiday. Users discuss holiday packages and critique their features. Regarding those reviews, the system suggests other holidays. The individual members' critiques results in a group preference model and other holidays are recommended based on this model. This system uses the strategy Average Without Misery in some aspects of the algorithm.

- **MUSICFX** [40] chooses a radio station for background music in a fitness center to satisfy a group of people working out at a given time. This system utilizes a slightly modified version of the Average Without Misery. In a normal approach to this strategy, the average of the ratings is taken, but only for those radio stations with individual ratings all above a specific threshold. In this case, to prevent starvation and regularly choosing the same radio station, a weighted random selection is generated from the top stations of the list.

- **INTRIGUE** [1] recommends sites to visit for tourist groups considering characteristics of subgroups among that group. A weighted form of the Average strategy is used to perform the aggregation strategy. Depending on the number of members in the subgroup and the subgroup's relevance (children and disables are given a higher relevance) certain weights are attributed to the average individual preferences.

- **LGM** [53] applies the MOVIELENS Dataset and uses the Average strategy, taking the average of all preferences of users and produces top-k recommendations.

- [37] shows, Multiplicative strategy performed best, because it was the only strategy for which all members believed its sequence would keep all members of the group satisfied (TV programs domain).

- **INTELLIREQ** [17] assists groups in selecting which software requirements to implement. Members can discuss recommendations for group decisions considering the already defined user preferences. It uses the Plurality Voting strategy.

- **jMusicGroupRecommender** [10] produces group recommendations by combining the recommendations of individual users, aggregating individual user rating, and subsequently forming the group preference.

2.2.2.2 Based on Negotiation Techniques

In the [60], a multi-agent approach is proposed using negotiation techniques for group recommendation. In this strategy, multilateral monotonic concession protocol (MCP)[15] is used to combine individual recommendations into a group recommendation, since it closely mirrors how human negotiation seems to function. The results obtained in this approach indicate that using this negotiation protocol, users in the groups were more evenly satisfied than with traditional ranking aggregation approaches. Nevertheless, the experimental study involved some threats to validity, being one of the most important not considering relationships (e.g., friendship) between users [60].

Table 2.2 summarizes the group recommender systems previously described, enumerating the group strategy used and the domain of application.

Recommender System	Group Strategy	Domain
POLYLENS [44]	Least Misery	Movies
CATS [41]	Average Without Misery	Holidays
MUSICFX [40]	Average Without Misery	Music
INTRIGUE [1]	Average	Tourism
LGM [53]	Average	Movies
[37]	Multiplicative	TV programs
INTELLIREQ [17]	Plurality Voting	Software
jMusicGroupRecommender [10]	Multiplicative, Approval voting, Least Misery, Fairness	Music

Table 2.2: Overview of Group Recommender Systems.

Chapter 3

Problem Statement

This section contains information about the problem to be addressed, the group recommendations, showing its relevance and how can be efficiently built. It also discusses the proposed solution and describes and presents some statistics about the dataset used.

3.1 Description

Recommender systems are based on algorithms that suggest items relevant to users, items like movies to watch, texts to read, products to buy or anything else, depending on the application domain. A recommender system can help people to decide between available items, when they do not have sufficient personal experience, in two ways: either to suggest items to users, or to provide users with information to help them decide which items will be most to their liking. Recommender systems (RSs) are fundamental to providers in some sectors, as they can generate a huge amount of revenue, when they are efficient, or also a way to stand out significantly from competitors. The example of Netflix is great evidence of the importance of RSs, which a few years ago organized a million-dollar prize challenge (the “Netflix award”), in which the objective was to produce a recommender system that performed better than its own algorithm.

Bad recommendations can lead to undesirable consequences, such as unnecessary use of resources, possible loss of customers, lack of precision due to data problems, and possibility for providers (of items or services) to take advantage of the recommender system if they discover the metrics used by the algorithm. Although there are already good RSs for individual users (with many different approaches), the same is not true for a group of users, especially related to tourism, where the main focus is concentrated in the entertainment business. This aspect of recommendation has been very relevant recently, because in a world with more and more interaction between people, through social networks, making recommendations for groups of users has become a general need.

3.2 Proposed Solution

This dissertation aims to explore different approaches of making individual and group recommendations, in the tourism domain. In the analysing stage of the tourism platforms, the multiple data made available by its public APIs or datasets was examined. This made it possible to identify which data are available in each of the platforms and select the data do be used as well as the relevant attributes for the recommender system. The data collected from available datasets must be stored efficiently to allow quick access by the recommendation algorithms. For this purpose, a relational database and a graph database will be created.

A traditional recommender system and a knowledge-based recommender system will implemented and described in Section 4. Both of the systems use content-based algorithms. The traditional content-based builds a user profile and Item Profile from user rated content to predict ratings. The Knowledge-based focus on analysing the description of the content, in which it is used to determine the similarity between items. Next, it uses the most similar items to a user's already-rated items to build recommendations. Both of these methods will produce group recommendations for a random group, an established group and a friends' group, using the following aggregation methods: average, least misery, multiplicative, average without misery and most pleasure. To assess the individual satisfaction of each user about a group recommendation, a specific metric is used, which will be also presented in Section 4.4 . With this information, the group's average satisfaction and its standard deviation will be determined for the five different aggregation techniques.

3.3 Dataset

This scenario used in this dissertation refers to a recommender system linked to tourism, using a Yelp[1] dataset, which contains reviews from users about tourist commercial establishments, such as restaurants, recreation spaces, car parks, public parks, etc. The used dataset is a Yelp subset containing data about some users, business and reviews data for academic, educational or personal use. The dataset is 5.79 gigabytes uncompressed in json format (6 json files, including business.json, check-in.json, photos.json, review.json, tip.json and user.json). Table 3.1 shows some statistics concerning the dataset. It is important that the dataset to be used is large scale and diversified enough to arrive at a realistic final result.

[1]https://www.yelp.com/dataset

	Yelp Dataset
#User	1,637,138
#Business	192,609
#Review	6,684,900
#CheckIn	161,950
#Tip	1,223,094
#Photo	200,000

Table 3.1: Detailed statistics of the Yelp Dataset.

For this project, only data related to user, business and review information will be used by the recommender system.

The user information includes the following:

- *user_id*, user identifier
- *name*, user name
- *review_count*, which is the number of reviews made by the user
- *friends*, relationship between to users
- *average _stars*, average rating of all the reviews

The business information includes the following:

- *business_id*, business identifier
- *name*, business name
- *stars*, star rating
- *categories*, characteristics that define the business
- *review_count*, number of reviews made about the business

The review information includes the following:

- *review_id*, review identifier
- *business_id*, business identifier
- *user_id*, user identififer
- *stars*, star rating
- *date*, review date

Figures 3.1, 3.2, 3.3 and 3.4 show some statistics about the dataset, namely the average, standard deviation, median, mode and a box-plot chart concerning the number of reviews per user, the number of categories per Business, the number of reviews per business and the number of Businesses per Category, respectively.

The major problem about this dataset is the fact that is 99.998 % sparse. Considering this percentage value, it will not be possible to validate the recommender system on individual recommendations. It is also possible to verify the impossibility of validating the RS through figure 3.1, which shows that at least 50% of the dataset is composed of users who only made one review.

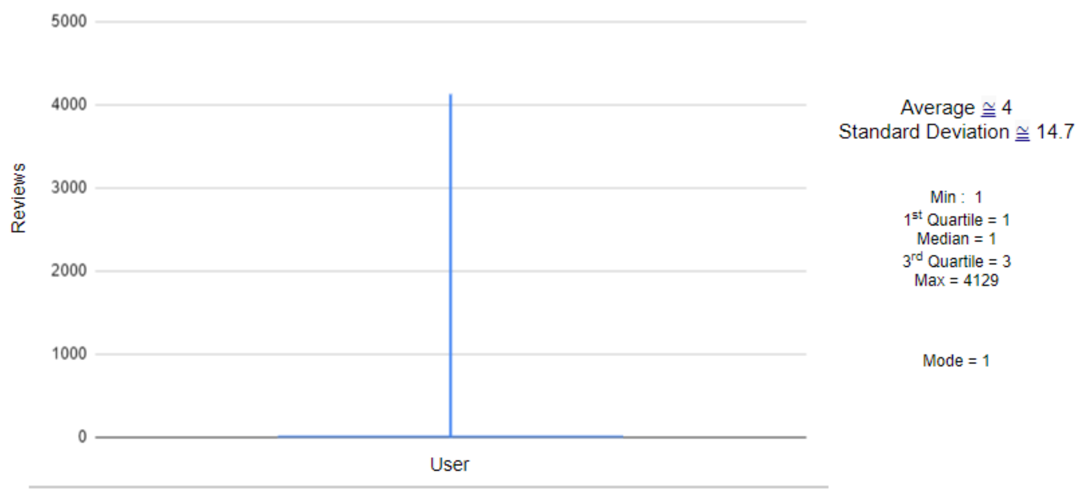

Figure 3.1: Detailed statistics concerning the number of Reviews per User

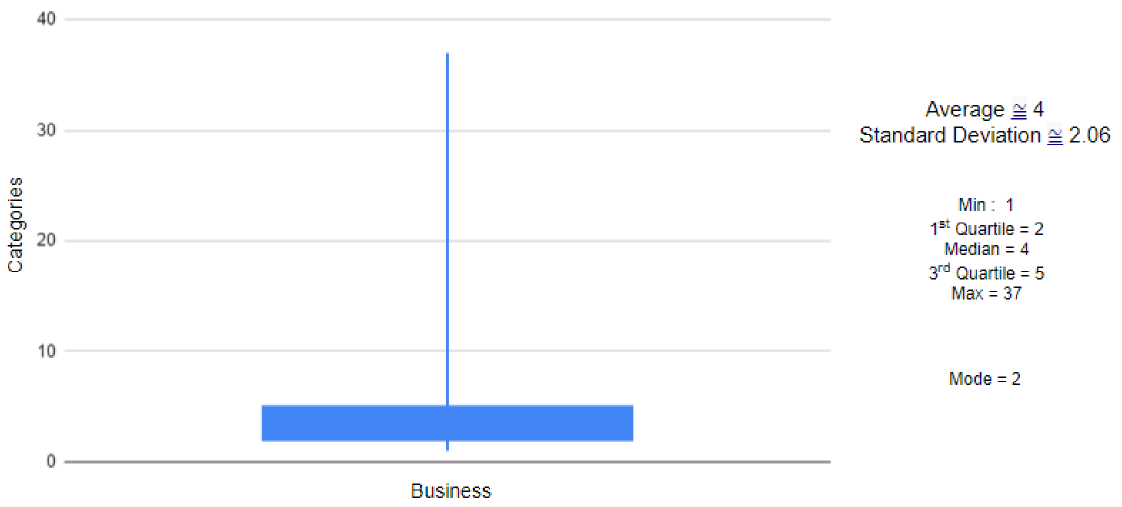

Figure 3.2: Detailed statistics concerning the number of Categories per Business

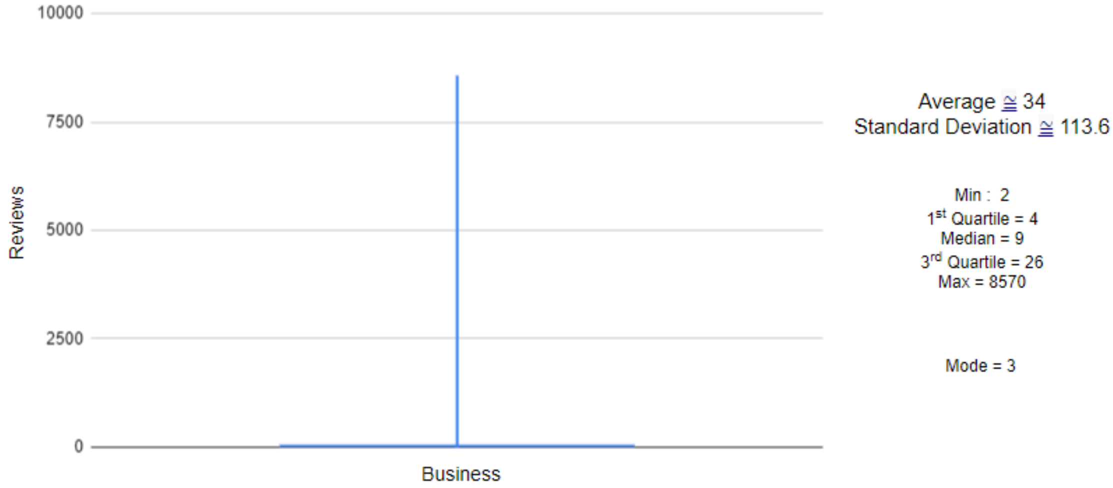

Figure 3.3: Detailed statistics concerning the number of Reviews per Business

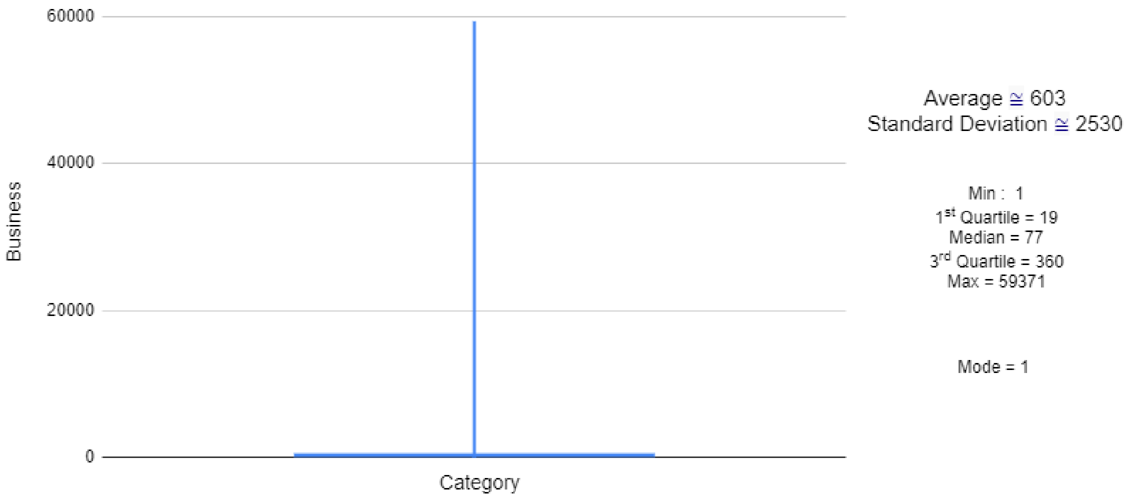

Figure 3.4: Detailed statistics concerning the number of Businesses per Category

Chapter 4

Recommender System - Implementation

In this dissertation, two recommender systems were created, a more traditional one that implements a content-based approach, and other that uses a Knowledge-based methodology. In both systems, group recommendations are produced utilising the same aggregation strategies. This chapter describes the implementation of both recommender systems and the formulation of group recommendations.

4.1 Content-based Recommender system

A content-based approach was chosen to implement a first recommender system, illustrating a more traditional view of this kind of systems. This technique takes advantage of the categories from the items the user has interacted in the interest of recommending similar items. It only depends on the user previous interactions, resulting in not having to deal with the cold start problem. To build user profiles and item profiles, it is common to use the items' categories. In this method contents of the items are previously ranked based on the user's preference, while the type of an item is an implied characteristic that it will be utilised to produce the Item Profile. Then, the item rating is predicted by applying both profiles and a recommendation will be presented. An illustration of this approach can be seen in Figure 4.1. The following sections will better detail the implementation.

4.1.1 Data Representation

In order to facilitate access to the data in the dataset, a relational database was created, with the structure defined in the Unified Modeling Language (UML) class diagram (see Figure 4.2).Three classes, an association class and a friendship relationship were defined. The "User" class has the attributes *user_id*, *name* and *review_count*. The "Business" class holds *business_id*, *name*, *stars* (rating), *categories* and *review_count*. The "Category" Class has a relationship with the business

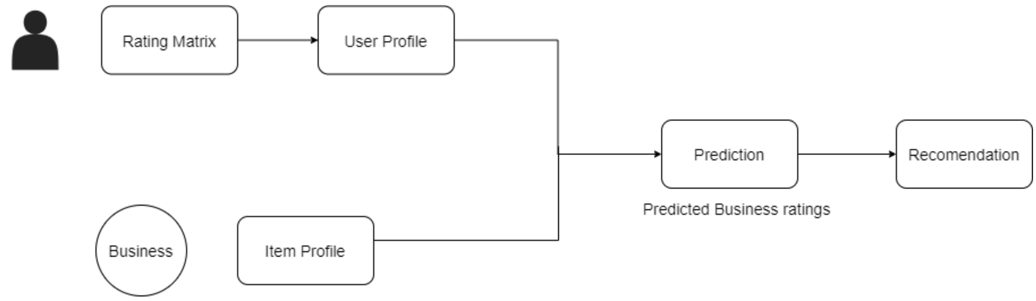

Figure 4.1: Content-based Approach.

class named "IsDefinedBy". Every Category is represented by its *id* and *name*. The association class Review contains the following atributes : *review_id*, *business_id*, *user_id*, *stars* and *date*.

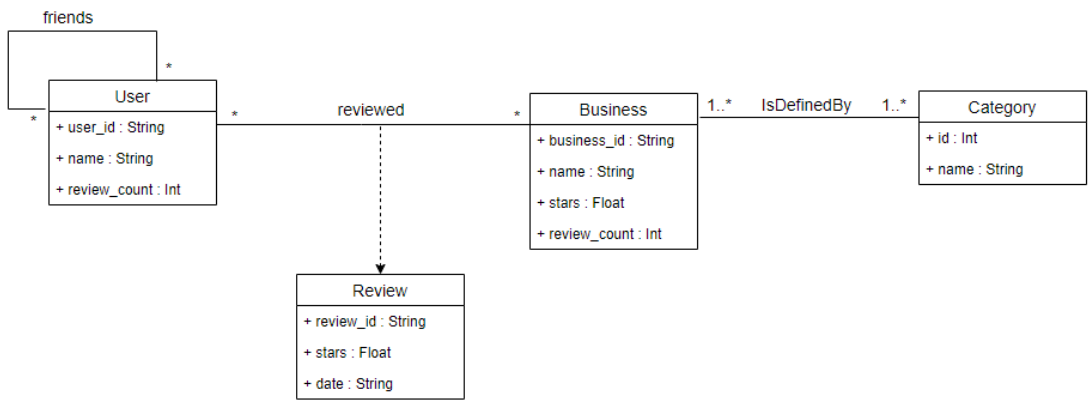

Figure 4.2: Database UML

4.1.2 Item Profile

The Item Profile is a matrix that relates the characteristics to the items (Businesses)[1]. Therefore, a matrix, as illustrated in Table 4.1, is produced. Each cell of the table represents the relationship between category and business. Treating all businesses as having unit weight, a binary representation is utilised, so a normalization is performed as defined in Equation 4.1, by dividing the term occurrence with the square root of the total categories in the Business.

[1]From now on, Business and Item are used as synonyms

	Category 1	Category 2	Category 3	Category 4	Category 5
Business 1	0.0	0.0	1.0	0.0	0.0
Business 2	0.707	0.0	0.707	0.0	0.0
Business 3	0.5	0.0	0.5	0.5	0.5
Business 4	0.577	0.577	0.0	0.577	0.0
Business 5	0.0	0.0	0.707	0.0	0.707

Table 4.1: An example of a Item Profile matrix.

$$Normalization(t,i) = \frac{\text{Number of } t \text{ occurrence}}{\sqrt{\text{Total Number of } i\text{'s categories}}} \quad (4.1)$$

where t is a category and i is an item.

4.1.3 User profile

To calculate the User Profile, the user's preferences are fundamental. Thus, it is necessary to structure the relationship between the user and the item through a matrix that contains the reviews given by the user to an item, named Rating matrix. Most of this matrix will not be filled, due to data sparsity, replacing the a null value with NaN[2], to facilitate the calculations. An example of a possible rating matrix is shown in table 4.2. The User Profile, which allows to see which categories the user prefers, is built through the Rating matrix product with the Item Profile matrix, as shown in table 4.3. The items' categories that the user has not shown interest are represented with the value zero.

	User 1	User 2	User 3	User 4	User 5
Business 1	4	NaN	NaN	NaN	1
Business 2	NaN	NaN	NaN	2	NaN
Business 3	4	3	NaN	NaN	NaN
Business 4	1	NaN	NaN	NaN	5
Business 5	NaN	NaN	2	NaN	NaN

Table 4.2: An example of a Rating matrix.

4.1.4 Prediction

In order to help create a recommendation, Inverse Document Frequency (IDF) was used to determine the relative importance of a category. It benefits in producing a higher rating to rare terms in the items' categories. The IDF is calculated by taking the logarithmic of the inverse of the

[2]Not a Number

category frequency within items, as shown in Equation 4.2.

$$IDF(t) = \log_{10}(\frac{\text{Total Number of items}}{\text{Number of items containing category t}}) \tag{4.2}$$

Then, the weighted ratings of each item are utilised again for a dot product with the user profile, resulting in the probability that a user will like the interaction with a specific item, considering the higher the predicted value, the more likely the user is to like it. An example of the predicted results, using the previous examples of item profile, user profile, rating matrix and weighted ratings can be seen in table 4.5.

4.1.5 Recommendation

Finally, to produce a recommendation, the system creates a table containing all the predicted ratings, without the items already reviewed, so the user does not get suggested an item that has already interacted. This table is sorted in decreasing order and the top-10 businesses generate the recommendation, as shown in Table 4.6.

4.1.6 User with no History

This approach has the problem that when a new user appears, that user does not have a previous interaction with any item, so it is not possible to create the rating and user profile matrices as defined previously. Also, the recommendation will not take place as described in last section. In this case, the system will recommend the 10 items with the best ratings, taking into account that these items have at least 100 reviews made by other users.

4.1.7 Advantages and Drawbacks

One of the main advantages of this approach is the user independence, which means that it is sufficient to study the user profile and the items in order to make a suggestion. In a collaborative filtering approach, information about other users' rating is needed, in order to find the similarity between them and then create a recommendation. Transparency is another advantage of the content-based approach because it can tell the user why did it recommend certain items based on their attributes, as oppose to collaborative filtering that creates recommendations to an user due to some unknown users with the same taste. Not having a cold start problem is also good, for the reason that new items can be recommended before being given a rating by a large number of users.

Regarding the drawbacks of this approach, limited content is an important one. If the content does not hold enough information to differentiate the items correctly, the recommendation will be not accurately in the outcome. Another obstacle is the fact that this technique might over-specialize, in other words, a limited degree of innovation is provided since it has to correspond the characteristics of the profile and the items. A completely perfect content-based method may recommend nothing surprisingly to the user. Despite not having a cold start problem, if a new user

	Category 1	Category 2	Category 3	Category 4	Category 5
User 1	2.577	0.577	6.0	2.577	2
User 2	1.5	0.0	1.5	1.5	1.5
User 3	0.0	0.0	1.414	0.0	1.414
User 4	1.414	0.0	1.414	0.0	0.0
User 5	2.885	2.885	1	2.885	0.0

Table 4.3: An example of a User Profile matrix, calculated with the values used in Item Profile 4.1 and Rating matrix 4.2.

	Category 1	Category 2	Category 3	Category 4	Category 5
IDF	0.22184	0.69898	0.09691	0.39794	0.39794

Table 4.4: An example of IDF vector, based on the data from the previous tables.

	User 1	User 2	User 3	User 4	User 5
Business 1	0.5814	0.1453	0.1370	0.1370	0.0969
Business 2	0.8152	0.3380	0.0968	0.3186	0.5210
Business 3	1.4872	0.8359	0.3498	0.2253	0.9425
Business 4	1.1542	0.5364	0.0	0.1810	2.1952
Business 5	0.9737	0.5247	0.4947	0.0968	0.0685

Table 4.5: An example of the matrix of Predicted values

	User 1
Business 1	1.4872
Business 2	1.1542
Business 4	0.9757
Business 6	0.9737
Business 7	0.8201
Business 8	0.6039
Business 10	0.5972
Business 3	0.3624
Business 4	0.1392
Business 5	0.0459

Table 4.6: An example of top-10 businesses recommendations to a user, by content-based RS

is introduced in the system, there will not be enough information to produce a trustworthy profile for a user and, consequently, the suggestion can not be provided accurately.

4.2 Knowledge-based Recommender system

The Knowledge-based recommender system implemented here also includes a content-based approach, but it exhibits some differences from the system described in previous section. The main focus of this knowledge-based technique is to recommend anything similar to an item the user liked previously. This approach has some similarities with the item-based collaborative filtering, in which are identified other users who like and dislike the same items, then are identified the items that have already been rated by those users but not by the current user, being these the recommended items. The knowledge-based system here described first finds the similarity between all pairs of items (model-based stage), then it utilises the most similar items to a user's already-rated items to produce the recommendations. An illustration of this approach can be seen in Figure 4.3. The following sections will better detail the implementation.

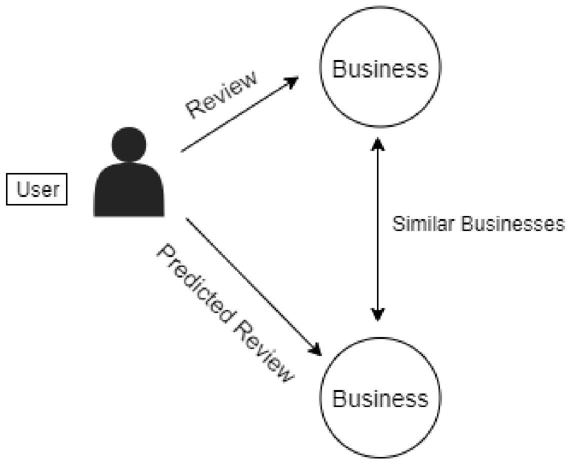

Figure 4.3: Knowledge-based Approach. Adapted from [43]

4.2.1 Data Representation

For this approach, it was necessary to create a knowledge graph representation of the data and, subsequently, build a graph database. The database schema can be seen in Figure 4.4, and it consists of three node types and three relationships. The node entities are named "User", "Business" and "Category". The "User" node includes the attributes *user_id*, *name* and *review_count*. The "Business" node contains *business_id*, *name*, *stars* (rating), *categories* and *review_count*. The "Category" node includes its *id* and *name*. The relationships "Friend" and "Is_Defined_By" connect a user to another user and a business to a category, respectively. The other relationship links a user to a business and is called "Review", containing as attributes *review_id*, *business_id*, *user_id*, *stars* and *date*.

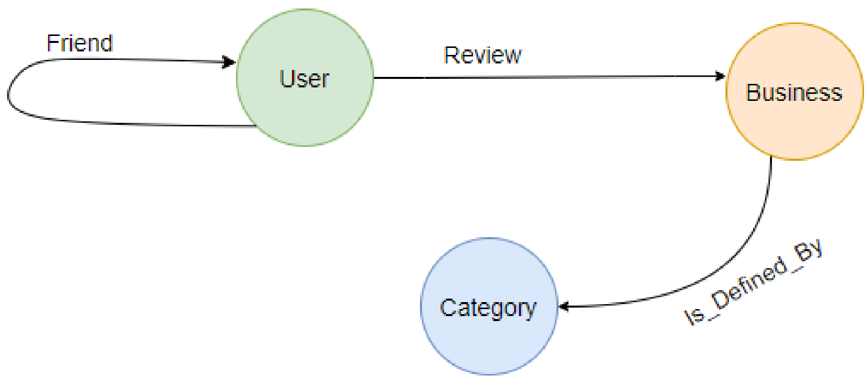

Figure 4.4: Graph Database Schema

4.2.2 Similarity

The first stage of this approach is to calculate the similarity between the items. To determine similarity, several well-known algorithms can be used, like Jaccard Similarity, Cosine Similarity or Pearson Similarity, and they usually produce similar results [33]. In this case, the Jaccard Similarity algorithm was used and computed using equation 4.3. The Jaccard similarity between two sets is defined as the size of the intersection divided by the size of the union of two sets [33]. Similarity values range between 0 and 1, with 0 denoting not similar nodes and 1 meaning very similar nodes.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cup B|} \tag{4.3}$$

where A and B are items, considering $A \neq B$.

In order not to have to calculate similarity each time a recommendation is presented, a new relationship called "Similar" was created. This can only be done because the initial data will never change whenever a recommendation of an item is given to a user. The "Similar" relationship makes the connection between two "Business" nodes and contains as an attribute the similarity value. This new schema is shown in Figure 4.5.

4.2.3 Recommendation

Once the similarity calculations are concluded, a recommendation can be performed. For this to happen, all the reviews that a specific user made, and the businesses that the user has interacted before, are collected. Then, the similarity between the business interacted and the non reviewed is multiplied by the users' review, resulting in the predicted rating a user would give to the new item. Businesses already reviewed are not considered during the prediction stage, improving the

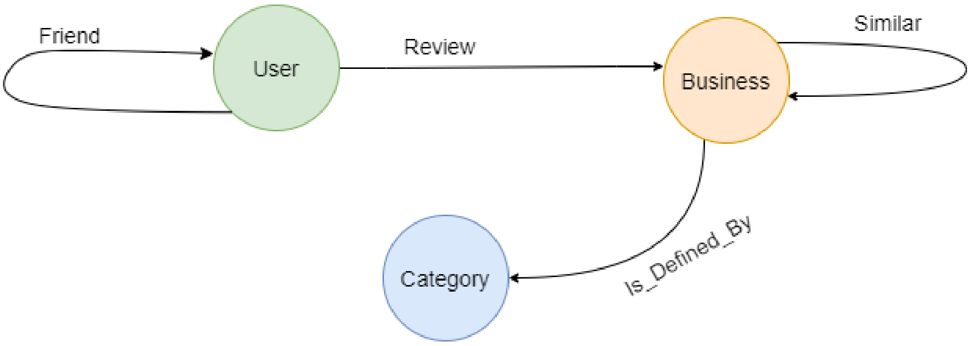

Figure 4.5: Graph with similarity relationship.

algorithm performance and time. Lastly, to produce a recommendation, the system creates a table containing all the predicted ratings, without the items already reviewed. This table is sorted in decreasing order and the top-10 businesses produce the recommendation, as displayed in table 4.7.

	User 1
Business 6	4.33
Business 2	4.0
Business 9	3.33
Business 4	2.66
Business 1	2.33
Business 3	2.33
Business 8	1.77
Business 10	0.77
Business 7	0.57
Business 5	0.33

Table 4.7: An example of top-10 businesses recommendations to a user, by Knowledge-based approach.

4.2.4 Advantages and Drawbacks

Avoiding the "new item problem" is one of the main advantages regarding this approach, because, unlike collaborative filtering, if the items have enough characteristics, the system will not need any other information to create a good recommendation. The many possible ways to represent content, like text processing techniques, usage of semantic information, etc, are another advantage of this strategy. As in the traditional content-based approach presented in previous section, transparency is also a benefit of this method, because it creates recommendations based on the same content that a user previously had in an interaction that was appreciated.

The main drawback of this approach is the over-specialization, with a propensity of producing a "filter bubble". "Filter bubble" is a circumstance in which a person only listens or sees the information that confirms what the person already believes and likes [5].

4.3 Group Recommendation

In this section, the implementation that aims to study the group recommendations is presented. This extra implementation made to the recommender system is used in both approaches described in previous sections, namely the content-based and knowledge-based RS. It was developed taking into account the types of groups and rating aggregation strategies. The recommendations are primarily calculated for every user independently and next aggregated into a recommendation list for the group by aggregation strategies.

4.3.1 Group Types

Group recommendations are made for three types of groups, based on the interaction between members. All computations made from this part of the implementation will be made only using the individual predicted ratings of each group member, select based on the group type. These types of groups are as follows:

- **Random group** is a group of people selected at random and who has not previously had interactions with the other members of the group.

- **Friends Group** is a group of people who have a friendly relationship between them. This relationship is present in both databases.

- **Content Group** is a group of people with at least one common interest. For this to happen, the list of users is descendingly ordered by the value of a specific feature of each user profile, for example, the users that have shown great reviews to businesses with the category "Nightlife".

4.3.2 Group Modeling Strategies

Group Modeling strategies are used after each member recommendations are calculated separately. The use of these strategies contributes to building a list of recommendations to a group. There were implemented 5 group modelling strategies:

- **Average**, this strategy uses the average of the predicted ratings from every user.
- **Multiplicative**, this strategy multiplies the predicted ratings from each member.
- **Most Pleasure**, is defined as the highest predicted ratings of all users.
- **Average Without Misery**, this strategy approach is similar to the Average strategy, with the exception that it excludes individual predicted ratings below a certain threshold, in this case, ratings inferior to 2.
- **Least Misery**, this strategy takes the minimum of individual predicted ratings.

The value obtained by this calculation, here named *groupValue*, will be used by the group recommendation module to produce a recommendation for the group. Obtained values for this *groupValue*, using different strategies, are presented in Table 4.9, which uses predicted ratings from all group members about an item named "Business 1" from Table 4.8.

	Business 1
User 1	1
User 2	1
User 3	4
User 4	5
User 5	3

Table 4.8: An example of predicted ratings from all the group members about an item

Average	Multiplicative	Least Misery	Average Without Misery	Most Pleasure
2.8	60	0	4	5

Table 4.9: *groupValues* for different strategies, using predicted ratings from Table 4.8

4.3.3 Recommendation

Once the group modelling calculations are concluded, a recommendation can be presented. Businesses already reviewed by any of the group members are not considered for the final recommendation list. Lastly, to produce a recommendation, the system creates 5 tables containing all the predicted group ratings for each group modelling strategy. This table is sorted in decreasing order and the top-10 businesses produce the recommendation.

4.4 Validation

The validation of the performance of the group recommendations is performed through the satisfaction of the group. The satisfaction of the group is measured through the average satisfaction of each group member, as defined in 4.5. Individual satisfaction results in the average of the position of each item of the group recommendation in the user list of individual recommendations. In Figure 4.6 it is possible to see the positions of the items "Business 1" and "Business 9"on the user individual list of recommendations, which are 4 and 9, respectively. The sum of all the positions, in the individual list of recommendation, of the businesses recommended to the group is divided by the total number of group recommendations, which is always 10, resulting in the individual satisfaction, as defined in 4.4. The higher the value of *S*, the lower the efficiency of the algorithm. Thus, the higher the *SG* value will also mean less efficiency of the algorithm.

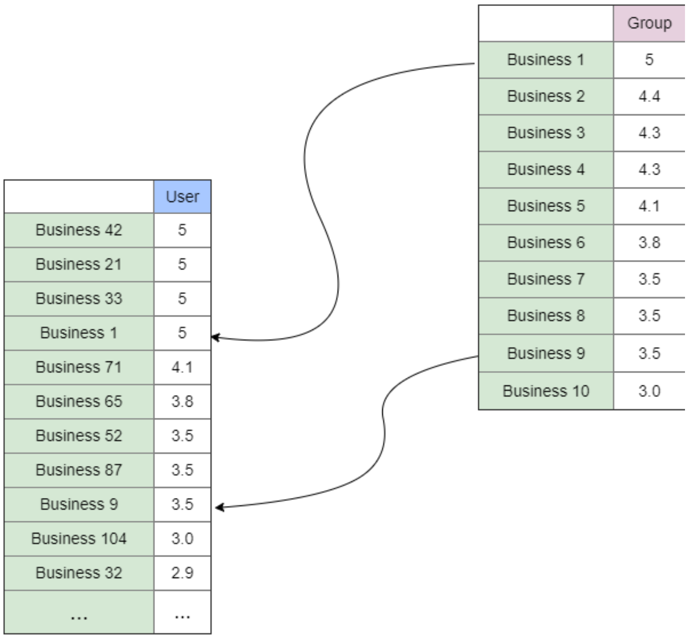

Figure 4.6: An example of Individual Satisfaction calculation, using the Most Pleasure strategy.

$$S(u) = \frac{\sum_{i=1}^{Ni} Pos_{group,individual}(i)}{Ni} \tag{4.4}$$

Where:

S : individual satisfaction

u : user

$Pos_{group,individual}(i)$: position of item i, presented in group recommendations, in the set of individual recommendations

Ni : total number of items recommended, which is always 10

$$SG = \frac{\sum_{u=1}^{Nu} S(u)}{Nu} \tag{4.5}$$

Where:

SG : satisfaction of the group

u : user

$S(u)$: individual satisfaction of user u

Nu : total number of group members

Chapter 5

Tests and Results

In order to better study the efficiency of the two approaches of recommender systems and aggregation modelling strategies, some experiments were conducted. In all the experiments, it was used the Yelp Dataset, containing 1,637,138 users, 192,609 businesses and 6,684,900 reviews. For both approaches, it was observed how efficiency varies according to different group sizes (5 and 10), different group types (Random, Friends and Content) and different aggregation modelling strategies, namely Average, Average without Misery, Most Pleasure, Multiplicative and Least Misery. For every different combination of characteristics of the experience, three random tests were made to ensure a better understanding of the efficiency variation. Efficiency is measured by the satisfaction of the group (SG, see equation 4.5), and the higher the *SG* value is means less efficiency of the RS. Each test made shows the *SG* and the STDEV of every aggregation modelling strategy.

5.1 Scenario 1: Random Group

In this scenario, experiments were performed with groups of the random type, using different group sizes, different aggregation modelling strategies and both algorithmic approaches. Figures 5.1 and 5.2 shows the group satisfaction average of the 3 experiments, in the content-based approach, for different aggregations modelling procedures for 5 and 10 group members, respectively. Tables 5.1 and 5.2 display the data used in these figures, respectively. The Average, the Least Misery and the Multiplicative are the AMSs more efficient for the different group sizes, while the Average without Misery and the Most Pleasure are less effective. Looking at the SG and STDEV of each test, it is observable that, in the Multiplicative and the Average AMSs, individual satisfaction of the members is more disparate than in the rest of the AGS studied.

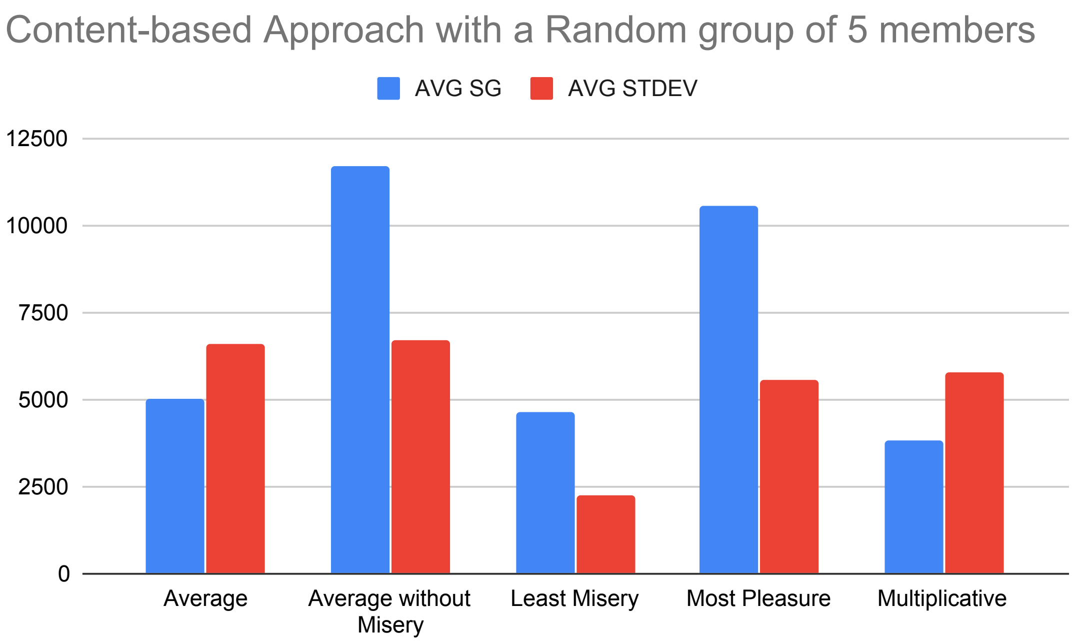

Figure 5.1: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Random group of 5 members and the Content-based approach

			Test 1 : 5 members + Random group	Test 2 : 5 members + Random group	Test 3 : 5 members + Random group
Content-based	Average	SG	3641.2	6793.42	4605.5
		STDEV	5763.42	7133.96	6938.34
	Average Without Misery	SG	12138.02	10815.27	12174.06
		STDEV	6811.72	6263.43	7059.71
	Least Misery	SG	4850	5334.8	3781.22
		STDEV	1889.03	2648.54	2188.63
	Most Pleasure	SG	12192.08	8306.76	11175.53
		STDEV	6798.04	4231.68	5720.44
	Multiplicative	SG	2764.46	5081.42	3683.46
		STDEV	5037.88	5886.01	6369.37

Table 5.1: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Random group of 5 members and the Content-based approach (data used in Figure 5.1)

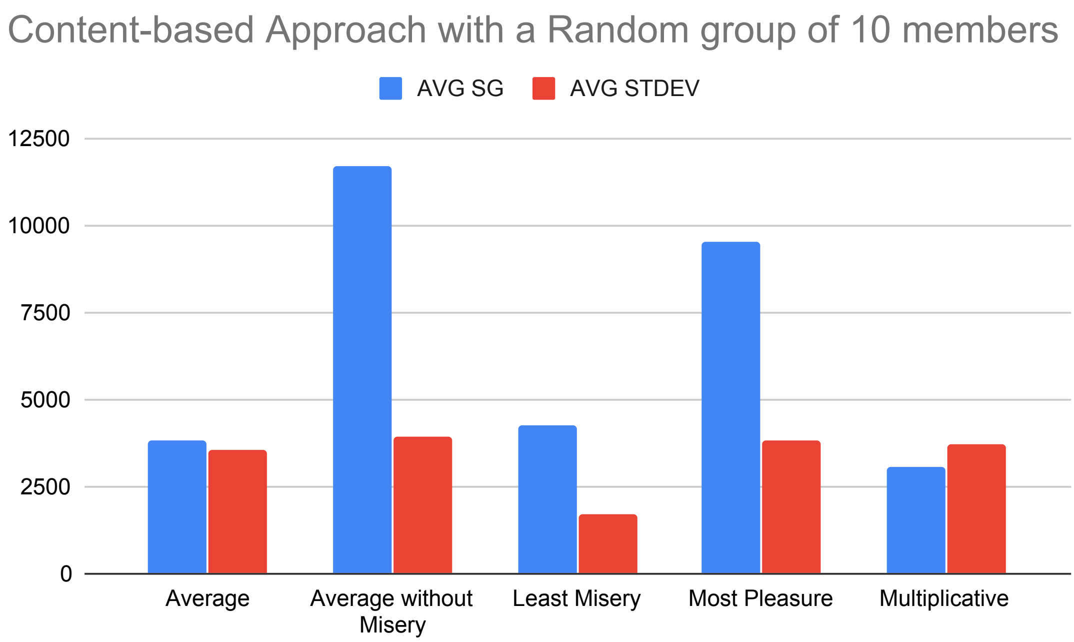

Figure 5.2: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Random group of 10 members and the Content-based approach

			Test 1 : 10 members + Random group	Test 2 : 10 members + Random group	Test 3 : 10 members + Random group
Content-based	Average	SG	1473.29	5783.95	4283.47
		STDEV	1774.95	5244.19	3616.1
	Average Without Misery	SG	5538.94	15426.14	14218.08
		STDEV	1503.75	5106.1	5141.62
	Least Misery	SG	1717.11	5553.62	5544.45
		STDEV	1877.29	1673.36	1538.23
	Most Pleasure	SG	4390.92	13668.72	10589.88
		STDEV	2097.79	6144.42	3235.32
	Multiplicative	SG	1402.01	4145.28	3615.41
		STDEV	1711.37	5796.55	3626.29

Table 5.2: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Random group of 10 members and the Content-based approach (data used in Figure 5.2)

Figures 5.3 and 5.4 present the group satisfaction average of the 3 tests, in the knowledge-based approach, for different aggregations modelling procedures and for 5 and 10 group members, respectively. Tables 5.3 and 5.4 present the data utilised in these figures, respectively. The Multiplicative AMS seems to perform somewhat better than the rest, except for Test 3 in Table 5.4. Taking into account that SG values are similar to STDEV throughout the tests, it is also possible to conclude that the individual satisfaction of the members is dispersed.

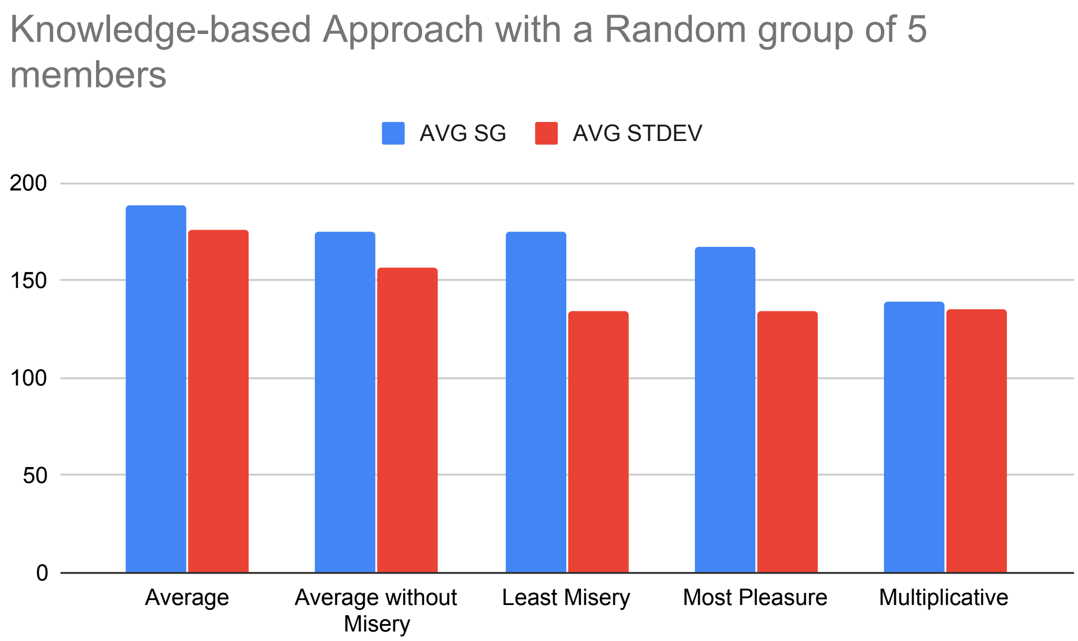

Figure 5.3: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Random group of 5 members and the Knowledge-based approach

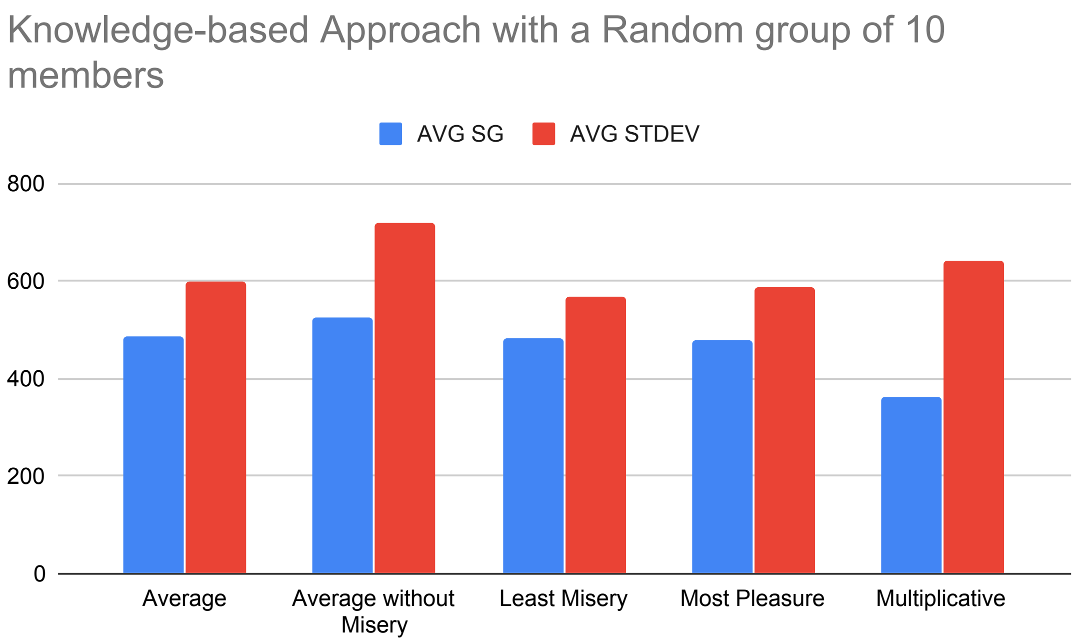

Figure 5.4: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Random group of 10 members and the Knowledge-based approach

			Test 1 : 5 members + Random group	Test 2 : 5 members + Random group	Test 3 : 5 members + Random group
Knowledge-based	Average	SG	184.3	255.1	127.32
		STDEV	174.74	287.19	67.34
	Average Without Misery	SG	184.3	189.94	150.16
		STDEV	174.74	169.05	127.03
	Least Misery	SG	181.14	199.42	144.04
		STDEV	98.57	186.86	117.29
	Most Pleasure	SG	179.74	193.46	129.64
		STDEV	158.47	171.95	72.43
	Multiplicative	SG	165.1	166.24	87.1
		STDEV	211.01	140.87	54.46

Table 5.3: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Random group of 5 members and the Knowlegde-based approach (data used in Figure 5.3)

			Test 1 : 10 members + Random Group	Test 2 : 10 members + Random Group	Test 3 : 10 members + Random Group
Knowledge-based	Average	SG	275.09	260.67	921.92
		STDEV	285.29	247.11	1266.24
	Average Without Misery	SG	256.71	282.77	1032.84
		STDEV	229.06	318.03	1615.55
	Least Misery	SG	248.96	263.98	931.93
		STDEV	223.5	252.44	1225.53
	Most Pleasure	SG	266.29	243.34	926.77
		STDEV	254.39	212.422	1294.78
	Multiplicative	SG	152	152	788.76
		STDEV	157.56	157.56	1606.52

Table 5.4: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Random group of 10 members and the Knowledge-based approach (data used in Figure 5.4)

Regarding the impact of the number of group members on the value of group satisfaction, in the content-based approach, it is not significant, while, in the Knowledge-based, it seems to result in a small decrease of efficiency.

5.2 Scenario 2: Friends Group

In this situation, tests were made with groups of friends, using diverse group sizes, different aggregation modelling strategies and both algorithmic approaches. Figures 5.5 and 5.6 include graphs that present the group satisfaction average of the 3 experiments, in the content-based approach, for AMSs for 5 and 10 group members, respectively. Tables 5.5 and 5.6 display the data used in these figures, respectively. Concerning AMSs, the Average, the Multiplicative and the Least Misery perform better for the many group sizes, while the Average without Misery and the Most Pleasure are less efficient. Focusing on the SG and STDEV of each test, it is visible that, in the Average and the Multiplicative AMSs, individual satisfaction of the group members is more spread than in the rest of the AGS analysed.

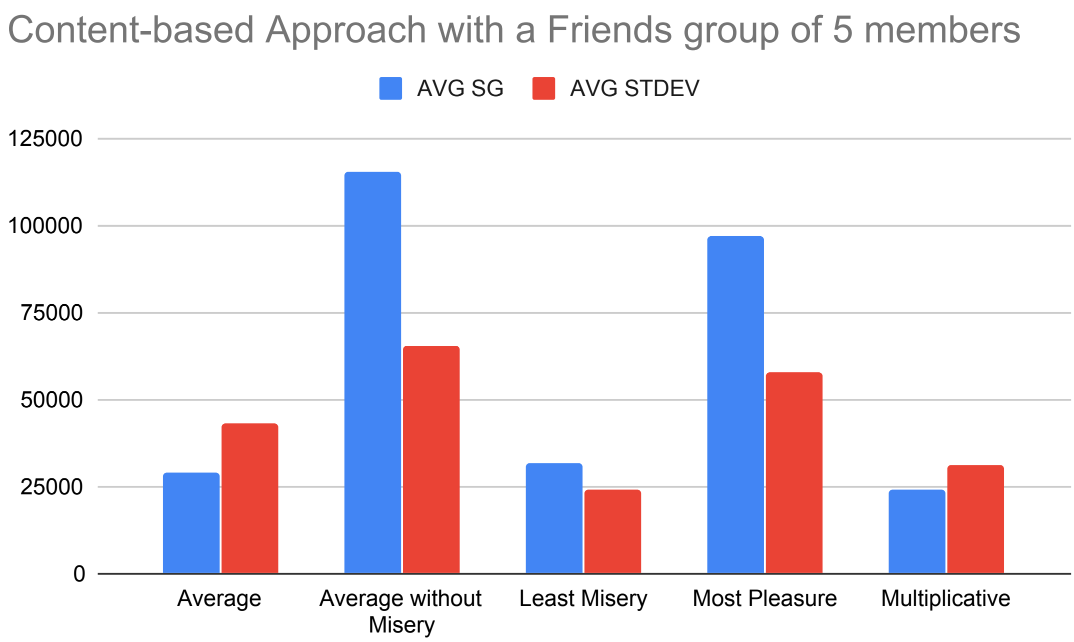

Figure 5.5: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Friends group of 5 members and the Content-based approach

			Test 1 : 5 members + Friends group	Test 2 : 5 members + Friends group	Test 3 : 5 members + Friends group
Content-based	Average	SG	27900.6	38283.38	21630.7
		STDEV	46117	49801.5	33037.96
	Average Without Misery	SG	104683.48	124828.2	116258
		STDEV	58878.24	70679.98	66015.12
	Least Misery	SG	27358.28	53135.04	14687.69
		STDEV	20373.52	38334.21	13090.72
	Most Pleasure	SG	81303.3	98851.84	110955.9
		STDEV	73500.82	36486.49	63482.77
	Multiplicative	SG	17956.86	30185.32	24361
		STDEV	26011.19	46450.13	21287.33

Table 5.5: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Friends group of 5 members and the Content-based approach (data used in Figure 5.5)

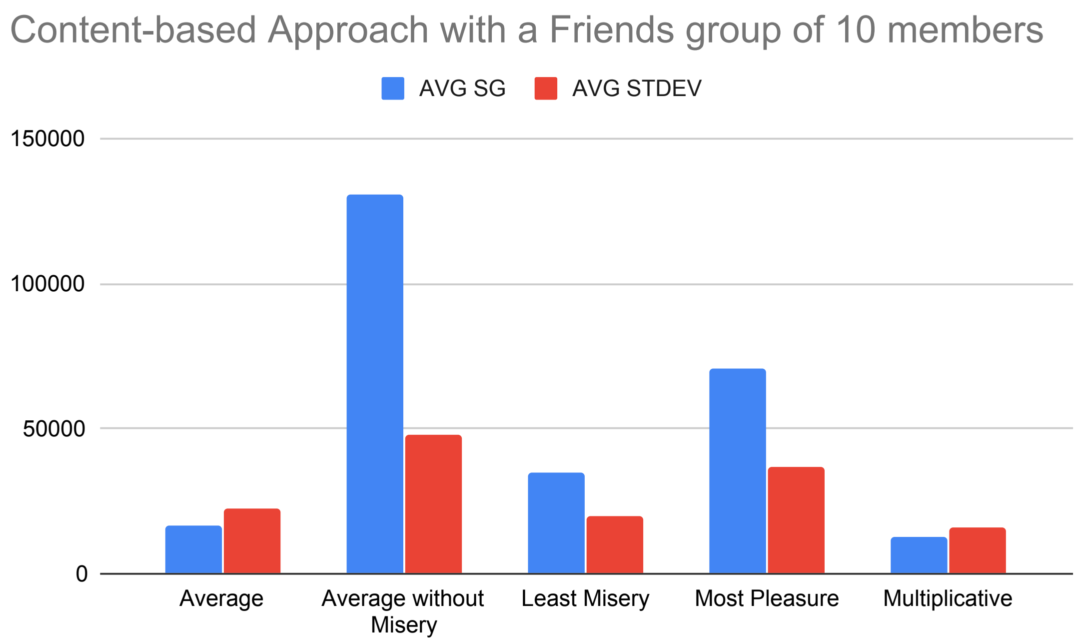

Figure 5.6: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Friends group of 10 members and the Content-based approach

			Test 1 : 10 members + Friends group	Test 2 : 10 members + Friends group	Test 3 : 10 members + Friends group
Content-based	Average	SG	26731.59	14788.13	8724.58
		STDEV	37185.22	16040.86	14314.78
	Average Without Misery	SG	120532.12	139012.78	132197.66
		STDEV	42815.07	51743.07	49830.69
	Least Misery	SG	41304.08	29572.8	33451.26
		STDEV	16962.13	20154.11	22251.06
	Most Pleasure	SG	43253.61	95707.34	73123.93
		STDEV	28821.66	36482.61	45673.76
	Multiplicative	SG	22779.88	11875.92	4244.08
		STDEV	37051.96	7166.62	3612.48

Table 5.6: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Friends group of 10 members and the Content-based approach (data used in Figure 5.6)

Figures 5.7 and 5.8 display the group satisfaction average of the 3 experiments, in the knowledge-based method, for different aggregations modelling procedures and for 5 and 10 group members, respectively. Tables 5.7 and 5.8 present the data used in these figures, respectively. The Multiplicative AMS seems to perform somewhat better than the rest, except for Test 2 in table 5.7. Regarding the values of STDEV, it is not possible to conclude anything about the individual satisfaction of the group members, because in half of the tests it has a low value and in the other half it has a high rate.

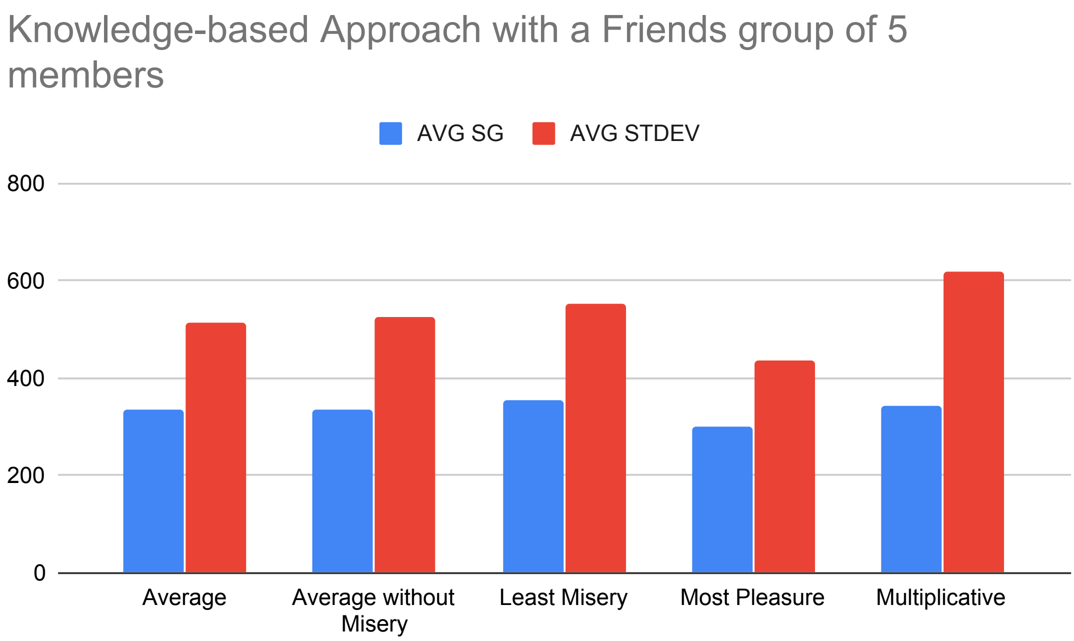

Figure 5.7: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Friends group of 5 members and the Knowledge-based approach

			Test 1 : 5 members + Friends group	Test 2 : 5 members + Friends group	Test 3 : 5 members + Friends group
Knowledge-based	Average	SG	640.06	119.7	248.95
		STDEV	1119.19	98.37	324.08
	Average Without Misery	SG	655.12	111.52	237.96
		STDEV	1152.67	95.69	326.7
	Least Misery	SG	698.24	105.74	260.16
		STDEV	1248.6	91.06	323.8
	Most Pleasure	SG	517.1	119.67	263.24
		STDEV	846.4	95.87	363.17
	Multiplicative	SG	871.43	78.58	83.68
		STDEV	1707.71	96.98	51.35

Table 5.7: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Friends group of 5 members and the Knowledge-based approach (data used in Figure 5.7)

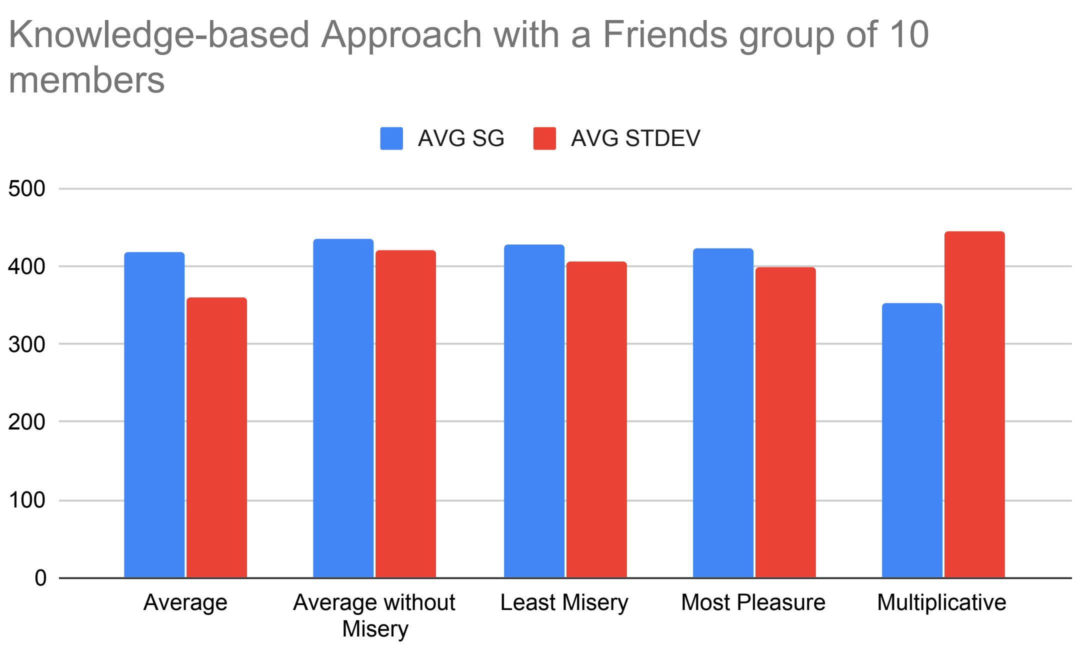

Figure 5.8: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Friends group of 10 members and the Knowledge-based approach

			Test 1 : 10 members + Friends group	Test 2 : 10 members + Friends group	Test 3 : 10 members + Friends group
Knowledge-based	Average	SG	168.75	169.76	913.23
		STDEV	104.07	98.73	878.1
	Average Without Misery	SG	166.46	165.81	970.69
		STDEV	95.35	94.25	1069.52
	Least Misery	SG	172.49	169.15	939.79
		STDEV	106.05	97.98	1014.19
	Most Pleasure	SG	168.55	160.87	937.31
		STDEV	97.62	84.72	1010.78
	Multiplicative	SG	135.7	135.7	789.21
		STDEV	120.79	120.79	1092.76

Table 5.8: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Friends group of 10 members and the Knowledge-based approach (data used in Figure 5.8)

Concerning the impact of the number of group members on the value of group satisfaction, in the content-based approach, results in a better performance when the group is smaller, while, in the Knowledge-based, it is not meaningful.

5.3 Scenario 3: Content Group

In this scenario, experiments were executed with Content groups, working with different group sizes, several aggregation modelling strategies and both algorithmic procedures. Figures 5.9 and 5.10 exhibit the variation of the group satisfaction average of 3 experiments, in the content-based approach, for AMSs for 5 and 10 group members, respectively. Tables 5.9 and 5.10 present the data applied in the making of the graphs in these figures, respectively. Regarding the AMSs, for groups of 5 members, the Multiplicative and the Least Misery are more efficient, while the Most Pleasure is less effective. When groups have 10 members, the Average and the Multiplicative AMSs perform best and the Average without Misery is the worst. Concerning on the SG and STDEV of each experiment, it is noticeable that, in any AMS, individual satisfaction of the group members is not as spread as in previous sections.

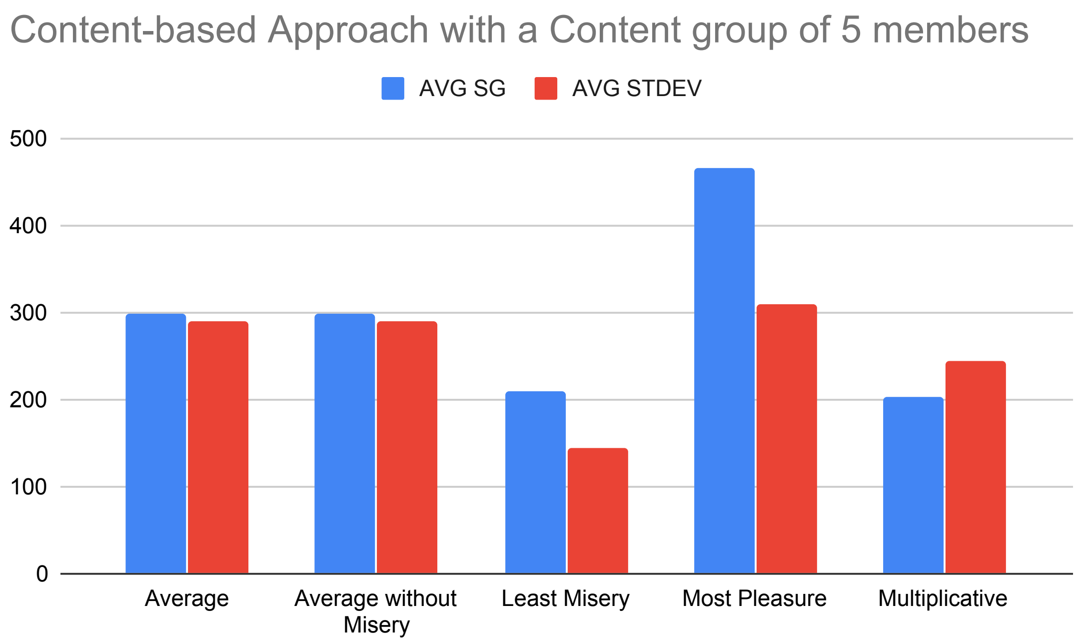

Figure 5.9: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Content group of 5 members and the Content-based approach

			Test 1 : 5 members + Content group	Test 2 : 5 members + Content group	Test 3 : 5 members + Content group
Content-based	Average	SG	398.14	211.58	285.28
		STDEV	361.32	274.97	231.4
	Average Without Misery	SG	399.56	211.58	285.6
		STDEV	365.24	274.97	232.15
	Least Misery	SG	159.02	228.72	240.04
		STDEV	157.27	161.29	116.73
	Most Pleasure	SG	552.66	354.32	493.47
		STDEV	355.72	321.37	253.12
	Multiplicative	SG	180.76	182.98	243.85
		STDEV	177.17	230.59	322.34

Table 5.9: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Content group of 5 members and the Content-based approach (data used in Figure 5.9)

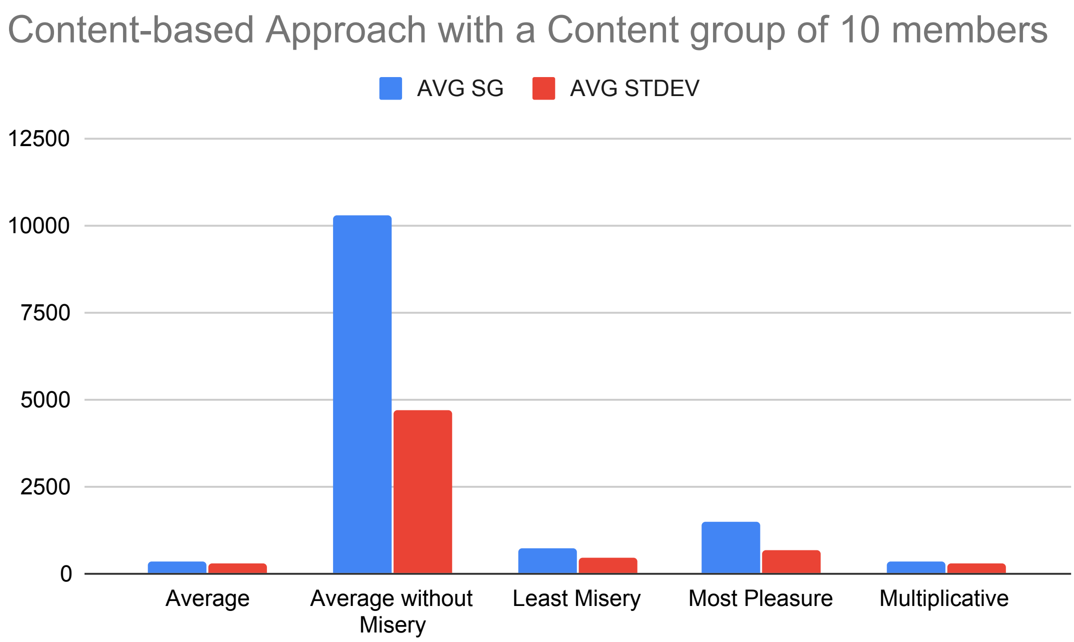

Figure 5.10: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Content group of 10 members and the Content-based approach

			Test 1 : 10 members + Content group	Test 2 : 10 members + Content group	Test 3 : 10 members + Content group
Content-based	Average	SG	281.26	485.81	331.67
		STDEV	142.23	485.79	229.32
	Average Without Misery	SG	15377.77	5600.8	9936.24
		STDEV	5454.18	3540.09	5059.77
	Least Misery	SG	281.46	1070.91	841.2
		STDEV	139.41	620.8	618.16
	Most Pleasure	SG	1742.91	1750.82	980.51
		STDEV	674.97	866.94	567.9
	Multiplicative	SG	282.96	457.52	331.87
		STDEV	141.84	434.02	232.13

Table 5.10: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Content group of 10 members and the Content-based approach (data used in Figure 5.10)

Figures 5.11 and 5.12 present the group satisfaction average of 3 experiments, in the knowledge-based approach, for different AMSs and for 5 and 10 group members, respectively. Tables 5.11

and 5.12 show the data applied in these figures, respectively. The Multiplicative AMS seems to perform somewhat better than the rest, except for Test 3 in table **??** and Test 2 in Figure 5.12. In the exceptions, the Least Misery and the Most Pleasure are more efficent. Regarding the values of STDEV, it is conceivable to assume that individual satisfaction of the group members is highly spread.

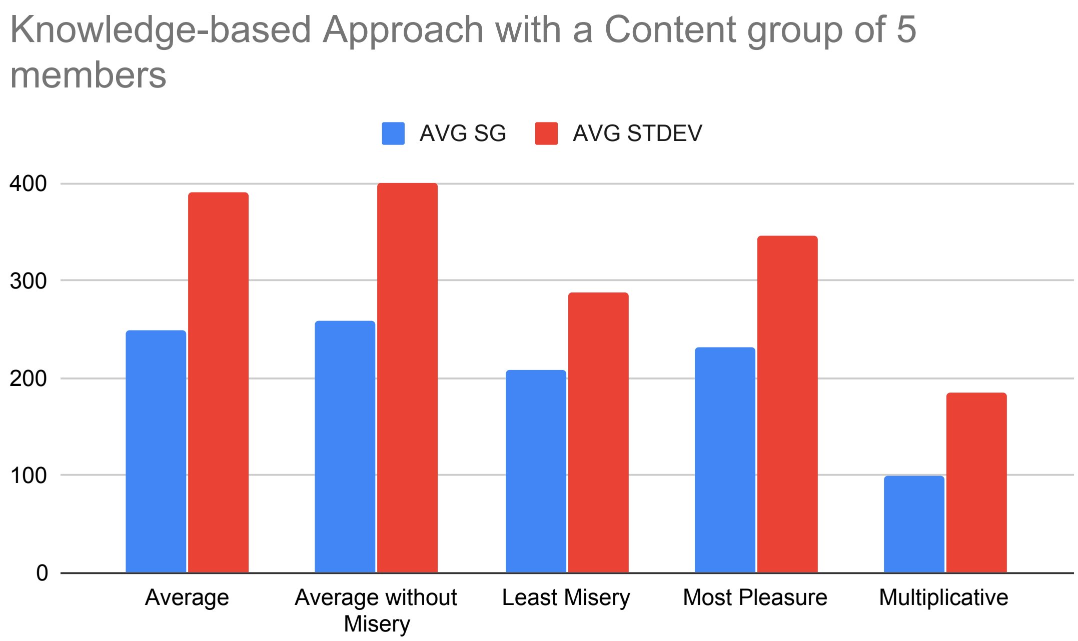

Figure 5.11: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Content group of 5 members and the Knowledge-based approach

			Test 1 : 5 members + Content group	Test 2 : 5 members + Content group	Test 3 : 5 members + Content group
Knowledge-based	Average	SG	169.86	491.7	83.78
		STDEV	196.48	911.64	67.23
	Average Without Misery	SG	186.26	506.2	83.78
		STDEV	224.58	907.74	67.23
	Least Misery	SG	169.86	387.18	68.5
		STDEV	196.48	629.91	38.68
	Most Pleasure	SG	170.16	457.34	65.92
		STDEV	186.19	815.88	37.59
	Multiplicative	SG	54.2	116.3	130.5
		STDEV	82.87	239.14	234.45

Table 5.11: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Content group of 5 members and the Knowledge-based approach (data used in Figure 5.11)

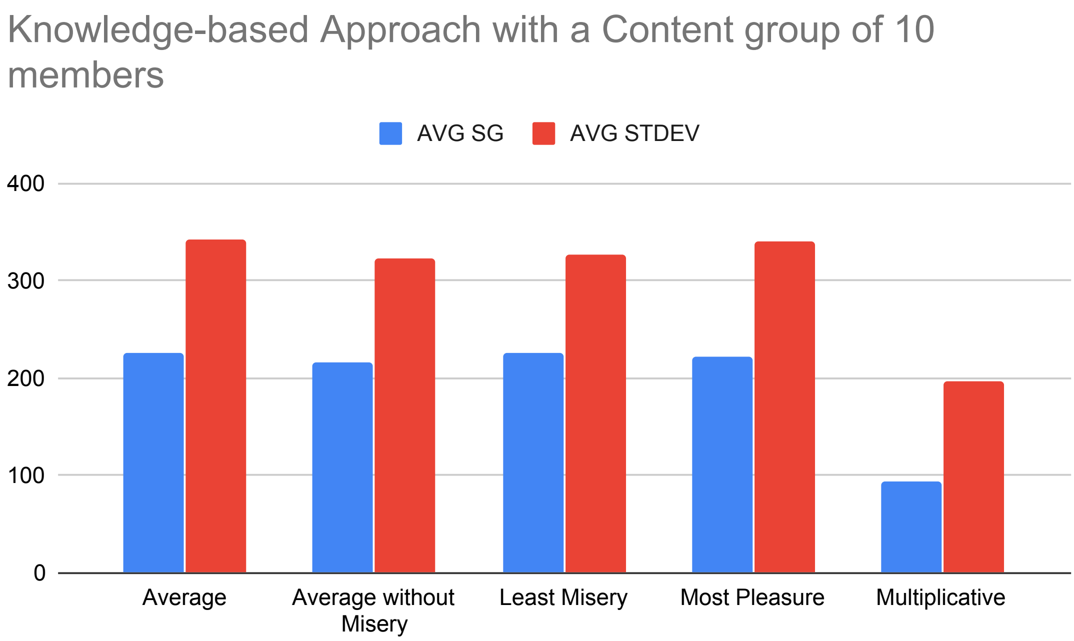

Figure 5.12: Graph showing SG (equation 4.5) variation and its STDEV, using the average values of three experiments, for each aggregation modelling strategy and each approach, using a Content group of 10 members and the Knowledge-based approach

			Test 1 : 10 members + Content group	Test 2 : 10 members + Content group	Test 3 : 10 members + Content group
Knowledge-based	Average	SG	385.13	85.01	206.38
		STDEV	651.93	109.67	265.37
	Average Without Misery	SG	354.96	85.01	206.38
		STDEV	591.56	109.67	265.37
	Least Misery	SG	398.29	79.23	197.46
		STDEV	664.48	86.18	231.16
	Most Pleasure	SG	376.65	82.96	207.58
		STDEV	647.85	105.62	267.28
	Multiplicative	SG	85.25	89.76	106.75
		STDEV	171.87	164.51	253.78

Table 5.12: Results of the three experiments, for each aggregation modelling strategy and each approach, using a Content group of 10 members and the Knowledge-based approach (data used in Figure 5.12)

Concerning the impact of the number of group members on the value of group satisfaction, in the content-based approach, results in a better performance when the group is smaller, except with the Multiplicative AMS, in most experiments. In the Knowledge-based, there is no significant impact.

5.4 Summary

The results of the experiments, using different approaches, group size and group type are shown in the figures presented in the table 5.13. Each test made shows the *SG* and the STDEV of every aggregation modelling strategy.

Regardless of the number of group members and the group type, for the content-based approach, the aggregation modelling strategies that are most efficient are the Average, the Least Misery and the Multiplicative. Despite being less efficient, the Average without Misery and Most Pleasure strategies have a more uniform individual satisfaction among the group. In this approach, it is also possible to observe that, for random groups and friends groups, the values of individual satisfaction of the members of the group diverge more in the Multiplicative and Average strategies. In Content groups, in every strategy, the values of individual satisfaction fluctuate more, since STDEV is, in general, always superior to *SG*.

For the Knowledge-based approach, despite the number of group members and the group type, the Multiplicative strategy is a little more efficient than the rest of them. In terms of the fluctuation of the values of individual satisfaction among members, it is higher in most of the experiments, except friends groups with 10 members, which is lower.

Notwithstanding the characteristics of the experiments, it is possible to see that the Knowledge-based approach is by far more effective than the Content-based procedure, mainly because of the high value of sparsity, which is one of its drawbacks. Also, in the Content-based procedure, it is observable that, when it is a Content group, the efficiency is higher, while in Friends groups is lower. Finally, concerning the Knowledge-based approach, generally, despite the number of group members and group types, the performance is similar amongst them.

	Group size	Content-based	Knowledge-based
Random	5	5.1	5.3
	10	5.2	5.4
Friends	5	5.5	5.7
	10	5.6	5.8
Content	5	5.9	5.11
	10	5.10	5.12

Table 5.13: Details about the figures containig the experiments

Chapter 6

Conclusions and Future Work

The final chapter presents a brief overview of all the developed work, refers the main contributions and indicates possible directions for future work.

6.1 Conclusions

In the context of this dissertation, primarily, a thorough study of the themes relevant to its development was conducted. This study includes both available and appropriate technologies problem, as well as an overview of recommender systems related work. Thanks to this study, it was possible to obtain the knowledge needed to build two recommender systems, based on a more traditional content-based method and a knowledge-based approach, utilising a dataset with real data. The main objective of this work is to study different procedures for building individual and group recommendations, using various recommendation algorithms and aggregation modelling strategies, in the tourism domain. It was also explored how different group's types and group's sizes affected the performance of the RS and the general group satisfaction.

The development of the recommender systems was concluded with the following most significant results: in the knowledge-based approach, the multiplicative aggregation model was more efficient compared to the others; regarding the traditional content-based method, the Average, the Least Misery and the Multiplicative strategies performed better than Average without Misery and Most Pleasure; the Knowledge-based procedure is more efficient than the Content-based method; the Content-based approach performs much better when dealing with a Content group.

6.2 Further Work

This work focuses on creating group recommendations for tourism, which could be expanded to other areas, like movies or music to understand which RS approach and aggregation modelling strategy would work better. A new RS method like applying Knowledge-graph into RS, using

Knowledge Graph Embedding, in essence, applying embedding algorithms in Knowledge graphs to create recommendations or to be combined with other algorithms like collaborative filtering and content-based approaches, would also be a good extension.

References

[1] L. Ardissono, A. Goy, G. Petrone, M. Segnan, and P. Torasso. Tailoring the recommendation of tourist information to heterogeneous user groups. In Siegfried Reich, Manolis M. Tzagarakis, and Paul M. E. De Bra, editors, *Hypermedia: Openness, Structural Awareness, and Adaptivity*, pages 280–295, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[2] David Ben-Shimon, Alexander Tsikinovsky, Lior Rokach, Amnon Meisles, Guy Shani, and Lihi Naamani. Recommender system from personal social networks. In Katarzyna M. Wegrzyn-Wolska and Piotr S. Szczepaniak, editors, *Advances in Intelligent Web Mastering*, pages 47–55, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[3] Avrim Blum and Tom Mitchell. Combining Labeled and Unlabeled Data with Co-Training. Technical report, 1998.

[4] Ludovico Boratto, Salvatore Carta, and M. Satta. Groups identification and individual recommendations in group recommendation algorithms. volume 676, pages 27–34, 11 2010.

[5] Engin Bozdag. Bias in algorithmic filtering and personalization. *Ethics and Information Technology*, 15(3):209–227, June 2013.

[6] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, page 43–52, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[7] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[8] Isabel Cenamor, Tomás de la Rosa, Sergio Núñez, and Daniel Borrajo. Planning for tourism routes using social networks, Oct 2016.

[9] Challenges and solutions in group recommender systems. https://www.ludovicoboratto.com/icdm2017_tutorial/. Accessed:2020-02-07.

[10] Ingrid A. Christensen and Silvia Schiaffino. Entertainment recommender systems for group of users. *Expert Systems with Applications*, 38(11):14127–14135, 10 2011.

[11] Andrew Crossen, Jay Budzik, and Kristian J. Hammond. Flytrap. *Proceedings of the 7th international conference on Intelligent user interfaces - IUI 02*, 2002.

[12] Malathi D., Logesh R., Subramaniyaswamy V., Vijayakumar V., and Arun Kumar Sangaiah. Hybrid reasoning-based privacy-aware disease prediction support system, Nov 2018.

[13] Malathi Devarajan and Logesh Ravi. Intelligent cyber-physical system for an efficient detection of parkinson disease using fog computing. *Multimedia Tools and Applications*, 78(23):32695–32719, 2019.

[14] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan. *Collaborative Filtering Recommender Systems*. now, 2011.

[15] Ulle Endriss. Monotonic concession protocol for multilateral negotiation. volume 2006, pages 392–399, 01 2006.

[16] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 6 2006.

[17] A. Felfernig, Christoph Zehentner, Gerald Ninaus, Harald Grabner, Walid Maalej, Dennis Pagano, Leopold Weninger, and Florian Reinfrank. Group decision support for requirements negotiation. pages 105–116, 07 2011.

[18] João Gama, André Ponce, Leon Carvalho, Katti Faceli, • Ana, Carolina Lorena, Márcia Oliveira, André C P L F De Carvalho, and Ana Carolina Lorena. *Data Mining*.

[19] Inma Garcia, Laura Sebastia, and Eva Onaindia. On the design of individual and group recommender systems for tourism. *Expert Systems with Applications*, 38(6):7683–7692, 2011.

[20] Damianos Gavalas, Vlasios Kasapakis, Charalampos Konstantopoulos, Grammati Pantziou, Nikolaos Vathis, and Christos Zaroliagis. The ecompass multimodal tourist tour planner, Jun 2015.

[21] Giuliano Giacaglia. Behind spotify recommendation engine, May 2020.

[22] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, December 1992.

[23] Gerhard Goos, Juris Hartmanis, Jan Van, Leeuwen Editorial Board, David Hutchison, Takeo Kanade, Josef Kittler, Jon M Kleinberg, Friedemann Mattern, Eth Zurich, John C Mitchell, Moni Naor, Oscar Nierstrasz, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y Vardi, and Gerhard Weikum. LNCS 4321 - The Adaptive Web. Technical report.

[24] László Grad-Gyenge, H Werthner, Peter Filzmoser, and Hannes Werthner. Recommendations on a Knowledge Graph Smart tourism View project Modelling and Simulation Environments View project Recommendations on a Knowledge Graph. Technical report, 2015.

[25] Zhiwei Guo, Wenru Zeng, Heng Wang, and Yu Shen. An Enhanced Group Recommender System by Exploiting Preference Relation. *IEEE Access*, 7:24852–24864, 2019.

[26] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, January 2004.

[27] Marcel Hildebrandt, Swathi Shyam Sunders, Serghei Mogoreanu, Mitchell Joblin, Akhil Mehta, Ingo Thon, and Volker Tresp. A Recommender System for Complex Real-World Applications with Nonlinear Dependencies and Knowledge Graph Context. Technical report, 2019.

[28] George Karypis. Evaluation of item-based top-n recommendation algorithms. In *Proceedings of the Tenth International Conference on Information and Knowledge Management*, CIKM '01, page 247–254, New York, NY, USA, 2001. Association for Computing Machinery.

[29] P. Kazienko, K. Musial, and T. Kajdanowicz. Multidimensional social network in the social recommender system. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 41(4):746–759, 2011.

[30] Heung Nam Kim and Abdulmotaleb El Saddik. A stochastic approach to group recommendations in social media systems. *Information Systems*, 50:76–93, 2015.

[31] Jae Kyeong Kim, Hyea Kyeong Kim, Hee Young Oh, and Young U. Ryu. A group recommendation system for online communities. *International Journal of Information Management*, 30(3):212–219, 2010.

[32] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. Grouplens: Applying collaborative filtering to usenet news. *Commun. ACM*, 40(3):77–87, March 1997.

[33] Sven Kosub. A note on the triangle inequality for the jaccard distance, 2016.

[34] Neal Lathia, Stephen Hailes, and Licia Capra. Trust-based collaborative filtering. In Yücel Karabulut, John Mitchell, Peter Herrmann, and Christian Damsgaard Jensen, editors, *Trust Management II*, pages 119–134, Boston, MA, 2008. Springer US.

[35] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Recommendation Systems*, page 292–324. Cambridge University Press, 2 edition, 2014.

[36] Paolo Massa and Paolo Avesani. *Trust Metrics in Recommender Systems*, pages 259–285. Springer London, London, 2009.

[37] Judith Masthoff. *Group Recommender Systems: Combining Individual Models*, pages 677–702. 01 2011.

[38] Judith Masthoff. Group recommender systems: Aggregation, satisfaction and group attributes. In *Recommender Systems Handbook, Second Edition*, pages 743–776. Springer US, 1 2015.

[39] Judith Masthoff and Albert Gatt. In pursuit of satisfaction and the prevention of embarrassment: Affective state in group recommender systems. *User Model. User-Adapt. Interact.*, 16:281–319, 09 2006.

[40] Joseph F. McCarthy and Theodore D. Anagnost. MusicFX. pages 363–372. Association for Computing Machinery (ACM), 1998.

[41] Kevin McCarthy, Lorraine McGinty, Barry Smyth, and Maria Salamó. The needs of the many: A case-based group recommender system. In Thomas R. Roth-Berghofer, Mehmet H. Göker, and H. Altay Güvenir, editors, *Advances in Case-Based Reasoning*, pages 196–210, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[42] Raymond J Mooney. Content-Based Book Recommending Using Learning for Text Categorization. Technical report, 1999.

[43] Black Raven (James Ng). Content-based recommender using natural language processing (nlp), Apr 2020.

[44] Mark O'Connor, Dan Cosley, Joseph A. Konstan, and John Riedl. *PolyLens: A Recommender System for Groups of Users*, pages 199–218. Springer Netherlands, Dordrecht, 2001.

[45] S. Ouaftouh, A. Zellou, and A. Idri. User profile model: A user dimension based classification. In *2015 10th International Conference on Intelligent Systems: Theories and Applications (SITA)*, pages 1–5, 2015.

[46] Michael J. Pazzani. A framework for collaborative, content-based and demographic filtering, 1999.

[47] Toon De Pessemier, Jeroen Dhondt, and Luc Martens. Hybrid group recommendations for a travel service. *Multimedia Tools and Applications*, 76(2):2787–2811, 2017.

[48] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K. Lam, Sean M. McNee, Joseph A. Konstan, and John Riedl. Getting to know you: Learning new user preferences in recommender systems. In *Proceedings of the 7th International Conference on Intelligent User Interfaces*, IUI '02, page 127–134, New York, NY, USA, 2002. Association for Computing Machinery.

[49] Logesh Ravi, V. Subramaniyaswamy, V. Vijayakumar, Siguang Chen, A. Karmel, and Malathi Devarajan. Hybrid Location-based Recommender System for Mobility and Travel Planning. *Mobile Networks and Applications*, 24(4):1226–1239, 8 2019.

[50] Baptiste Rocca. Introduction to recommender systems, Jun 2019.

[51] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, page 285–295, New York, NY, USA, 2001. Association for Computing Machinery.

[52] J. Ben Schafer, Joseph A. Konstan, and John Riedl. E-commerce recommendation applications.

[53] J. Shi, B. Wu, and X. Lin. A latent group model for group recommendation. In *2015 IEEE International Conference on Mobile Services*, pages 233–238, 2015.

[54] Rashmi Sinha and Kirsten Swearingen Sims. Comparing Recommendations Made by Online Systems and Friends Recommendations from friends Recommendations from system Input from user Recommender System Recommendations from friends Recommendations from system Input from user. Technical report.

[55] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):111–133, 1974.

[56] Subramaniyaswamy V, Logesh R., M. Abejith, Sunil Umasankar, and Arumugam Umamakeswari. Sentiment analysis of tweets for estimating criticality and security of events. *Journal of Organizational and End User Computing*, 29:51–71, 10 2017.

[57] David Vallet, Iván Cantador, and Joemon Jose. Personalizing web search with folksonomy-based user and document profiles. volume 5993, pages 420–431, 03 2010.

[58] Robin Van Meteren and Maarten Van Someren. Using Content-Based Filtering for Recommendation 1. Technical report.

[59] V. Vijayakumar, D. Malathi, V. Subramaniyaswamy, P. Saravanan, and R. Logesh. Fog computing-based intelligent healthcare system for the detection and prevention of mosquito-borne diseases, Dec 2018.

[60] Christian Villavicencio, Silvia Schiaffino, J. Andres Diaz-Pace, Ariel Monteserin, Yves Demazeau, and Carole Adam. A MAS approach for group recommendation based on negotiation techniques. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9662, pages 219–231. Springer Verlag, 2016.

[61] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei Ying Ma. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 13-17-August-2016, pages 353–362. Association for Computing Machinery, 8 2016.

[62] Harry Zisopoulos, Savvas Karagiannidis, Georgios Demirtsoglou, and Stefanos Antaris. Content-based recommendation systems. 11 2008.

[63] Koen Zoetekouw. A critical analysis of the negative consequencescaused by recommender systems used onsocial media platforms.