BACHELOR

A Reciprocal Recommender System for Charly Cares

van Knippenberg, Tom

Award date:
2020

A Reciprocal Recommender System for Charly Cares

Tom van Knippenberg
Eindhoven University of Technology

Abstract—Recommendations are the key part of the business of Charly Cares, a platform in the Netherlands that establishes a link between families and babysitters (called angels within the platform). To support this service, a recommender system is implemented in this bachelor end project that meets Charley Cares requirements. The recommender system tackles the challenging problem of connecting two types of persons through an online platform, in this case a family and an angel. To reach this goal, a reciprocal recommender system is designed and developed. In the reciprocal recommender system a score is calculated between a family and an angel to captures the mutual interest of both parties. New families are matched to known families, using demographic characteristics to overcome the cold start problem. The system is designed in such a way that Charly Cares has the options to easily change some parts of the system to make it fit with their business strategy. For example number of recommendations and number of new angels. The system is implemented in the Python language. The performance of the system is evaluated on three characteristics, namely quality, speed and scalability of recommendations. The major finding from this evaluation is that the method provides recommendations good and fast enough for known families, but more research is needed for providing recommendations to new families that enter the platform for the first time. With this project another use-case can be added to the research on reciprocal recommender systems.

Index Terms—Reciprocal Recommender System, Baby-Sitting, Collaborative Filtering, Matching

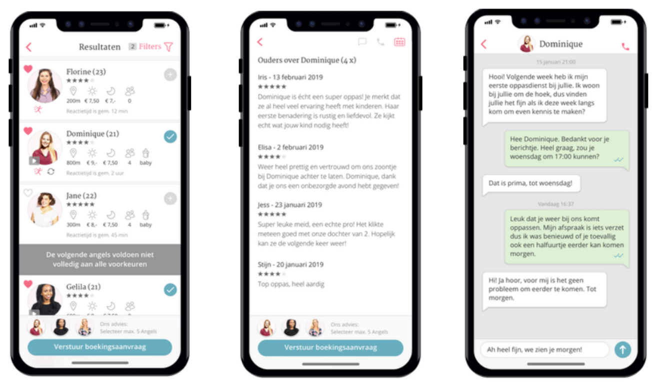
1 Introduction

In this research, the tech start-up Charly Cares [1] will be in focus for the process of building a reciprocal recommender system for the baby-sitting domain. We base the research on this platform since it is a real-life environment and contains a real business process which provides an important application.

The definition of a recommender system is "software tools and techniques that provide suggestions for items that are most likely of interest to a particular user" [14]. Recommender systems are about the core recommendation technique but also about the graphical user interface and its design [14]. In this bachelor end project a reciprocal recommender system will be implemented and tested. The focus is on the core recommendation technique. The class of reciprocal recommender systems is used in situations where persons are recommended to other persons, a scenario that also occurs in other domains than baby-sitting such as online dating, social media, online education, or the job market. The benefit of this system is that it takes into account the interests of both sides of the platform on which the recommendations have an effect.

Charly Cares is a platform in the Netherlands that brings together families and babysitters (see Figure 1 for examples). They have around 20 people working in the company. Charly Cares earns money by selling memberships to and a fee for every booking of families and by keeping a small percentage of the hourly wage of the babysitters. At the end of January 2020, Charly Cares has more than 39,000 families on the platform of which 35% has a membership. There are more than 26,000 babysitters registered on the platform. Babysitters are called angels by the company and this term will be used in the rest of this project. Charly Cares operates in 8 different cities in the Netherlands. The number of angels per region ranges from around 500 to over 9,000 who are available per region for a booking. The number of regions will be growing quickly. This year they will add 4 new cities in the Netherlands to their options for families [5].

The goal of Charly Cares is to make it very easy to book an angel that a family can trust. It is therefore important to connect families and angels such that the booking is in the interest of both angel and family. A multi-sided platform, like Charly Cares, creates value by coordinating groups of consumers [7]. Recommender systems can have a great value and can be quite complicated as is shown by the competition by Netflix [2].



(a) (b) (c)

Fig. 1. An illustration of Charly Cares: (a) Interface application. (b) View profiles of angels. (c) Personal contact with angels.

The process in the application of Charly Cares is as follows. If families are new to the application, then families sign up for one of three different memberships. This is based on their choice of whether they plan on having many or just a couple of requests. The three different memberships are:

- **Basic:** Families pay €2 monthly and €7.50 for each booking. This is recommended for families that plan on less than 1 booking each month.
- **Flexible:** Families pay €6 monthly and €3 for each booking. This is recommended for families that plan on making 1 to 4 bookings each month.
- **Premium:** Families pay €25 monthly and €0 for each booking. This is recommended for families that plan on making more than 4 bookings each month.

Then, families can request a sitting for a certain date and time. They receive a list of 75 angels where they can choose different angels for their booking, this number is determined by Charly Cares. The chosen angels receive an invitation which they can accept. If they accept the

• *Tom van Knippenberg, Eindhoven University of Technology. E-mail: t.a.h.v.knippenberg@student.tue.nl*

request, they will be performing the booking. The angel only has the possibility to perform a booking when he or she is invited by the family.

The problem that Charly Cares is facing at the moment is that their recommender system gives contradicting recommendations. The algorithm that Charly Cares uses is based on characteristics of a booking and the angel, for example the distance from the angel to the family and the average response time of an angel. Every characteristic gets a random weight based on what seems to be important to make a booking successful. The result is that the list of angels cannot always be explained properly. They therefore want a more intelligent alternative to provide recommendations. The result of recommendations as they appear in the app are shown in Figure 1a.

In this report, the recommender system of the company Charly Cares will be improved into a more intelligent recommender system that learns from the behavior of persons on the platform. The focus of this project is on the core recommendation technique for recommending angels to families such that their booking request will be fulfilled. The result is a recommender system that satisfies all requirements using a collaborative filtering and demographic approach. The system is based on the reciprocal (mutual) relation between the family and the angel. This is captured in a reciprocal score. The system is based on a collaborative filtering approach as there are no explicit preferences of families available. Because of this approach, there is a demographic approach to matching new families to known families. In this domain matching availability is important to make a successful booking. The availability and region of an angel is therefore taken into account when providing recommendations. Angels who visited a family before are also part of the final recommendation for a family. The implementation details are discussed in Section 3.3.

In summary, the recommender system is designed for the following reasons:

- Connecting two types of people on a platform
- Provide recommendations to persons who are new to the platform
- Making the system more transparent and intelligent than the old system

The remainder of this report is as follows: In Section 2 I discuss classes of and approaches to recommender systems and describe related work in the field of online dating. The recommender system as well as the requirements set for the recommender system are introduced in Section 3. This is followed by a description about the data handling (Section 4) and the designed algorithms (Section 5). In these sections Figure 2 will be used to illustrate where in the process this happens. The performance of the recommender system will be discussed on the perspective of quality of recommendations, and speed and scalability of recommendations in Section 6. Application examples are given to illustrate the research which is described in Section 7 and provide discussions in the following Section 8. The work is concluded in Section 9 with an outlook to future challenges.

2 Related Work

Recommender systems are extremely sophisticated and specialized systems that often seem to know a person better than he/she does him/herself [8]. The recommendations have a focus on being personal [3]. Recommendations are important in the case when there is a great choice in different objects or options.

In the case of Charly Cares [1] there is a great choice of the angels for families. Therefore, recommendations play an important part in their services. Those services provide a list of benefits for the users. For families the benefits are booking an angel easy and fast, having angels that are screened and are insured. For angels it provides them with work to earn money at a small cost for being on the platform.

In the rest of this section, research on traditional and reciprocal recommender systems is provided. Moreover, different approaches to recommender systems with their advantages and disadvantages are investigated. Ultimately, the choice for the type of recommender system and approach will be discussed.

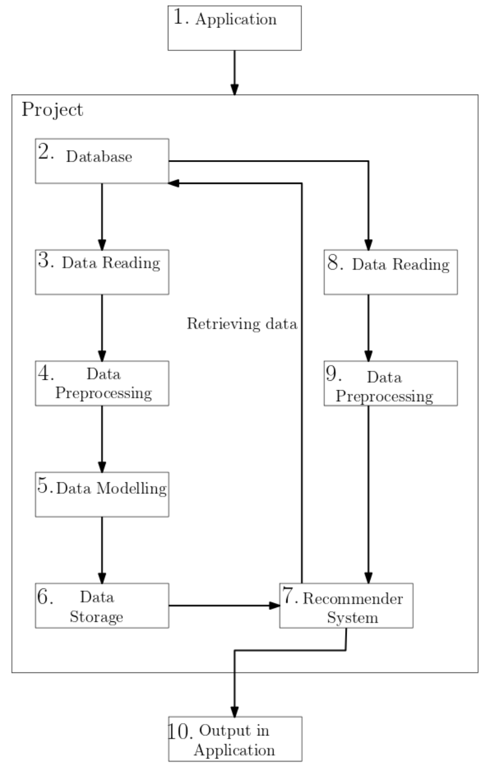


Fig. 2. The process of data handling and providing recommendations

2.1 Traditional Recommender Systems

Traditional recommender systems are based on the idea of recommending items to persons [8, 11]. The success of a recommendation is determined by the user that receives the recommendation [11].

This class of recommender systems is the basis in many applications today, such as Netflix recommending which series or movies to watch. For these recommender systems, it is important to not make foolish suggestions [8]. Techniques that are used in this class are described in Section 2.3. Within the case of Charly Cares it is important to recommend people to people. As, traditional recommender systems recommend items to persons this is not a suitable area to investigate further. There is another class focused on recommending people to people; the class of reciprocal recommender systems. The reciprocal recommender systems will be discussed next.

2.2 Reciprocal Recommender Systems

There are also applications in which people are recommended to people. There are many examples for those applications recommending people to people nowadays like dating websites, LinkedIn as job market and social media platform, and Facebook, Twitter, and Instagram as another social media platform. The domains in which it has been applied are recommending a possible date and recommending a possible employee to an employer. In these domains the success of a recommendation is determined by the perception of the two persons [11]. An interaction is successful when some person reacts positively to the interaction initiated by another person [9].

Pizzato et al. (2013) argue that many users leave the system after a successful recommendation [11]. This stands in contrast to classic recommenders where successful recommendations should motivate longer term use. In the case of Charly Cares, a successful recommendation should motivate families to use the app multiple times in the future. This is thus different from what is discussed in research up until now.

Kutty et al. [9] investigated about using graph mining techniques to recommend people to people. They made a match-making system for an online dating network using an attributed node graph. The match-making system combines three different algorithms, one of which tackles the cold-start problem. Pizzato et al. [11] also made a recommender system for online dating. Their recommender system matches the preferences of two persons and calculates a compatibility score on the characteristics of the possible recommended person and the preferences of the user. Their research shows that ”reciprocal recommenders can reduce the burden on popular users as they consider the preferences of both sides of the recommendation” [11].

The paper by Xia et al. [17] builds on the work of Pizzato et al. [11]. They build six different reciprocal recommender systems in the domain of online dating where only heterosexual relationships are considered. Two of these approaches use the content-based approach and four of these approaches use the collaborative filtering approach. This is a bipartite network where males do not communicate with other males, but only with females. The same situation can be seen in the domain of baby-sitting: families only communicate with angels and angels only with families. The techniques they use can thus also be applied to a great extent to this setting in the baby-sitting domain.

Pizzato et al. [11] found that reciprocal recommender systems “can reduce the burden on popular users as they consider preferences of both sides of the recommendation.” In the case of Charly Cares this is very useful as not only the popular angels need to receive requests. The goal is to have a platform for everyone. Finding a babysitter is classified as a reciprocal activity [11]. Future research is stimulated to create a recommender system in the baby-sitting domain. This paper thus builds on their research.

Hence, reciprocal recommender systems provide recommendations in domains where people interact with other people. Given the setting in the baby-sitting domain this type of recommender system is most appropriate.

2.3 Approaches in Recommender Systems

Different approaches are available to build a recommender system. In this section, six different approaches will be discussed [14]. The approaches that can be used are highly dependent on the domain of application of the recommender system [3].

Content-Based

The content-based approach uses the characteristics of items and persons [3]. Items with similar characteristics that a person already liked, will be recommended [14]. The first example of this approach is about movies. Someone likes a movie with Paul Walker as actor. Another movie in which Paul Walker is actor is then recommended to this person. A second example of the content-based approach is about recommendations on a dating platform. Person A provides preferences stating that they like a certain hair color. Persons with this hair color can then be recommended to person A. One issue with this can be that the quality of the features that people give to the system and their behavior can be different [3]. For example, people state that they prefer brown hair, but they always interact with persons with black hair. It is found that in reciprocal recommender systems this is not the superior approach [17].

Collaborative Filtering

Collaborative filtering is an approach that takes into account the behavior of a user that is shown in the past [8, 14]. This behavior is captured in either implicit or explicit data. Implicit data can be viewing time and clicks. Explicit data are, for example, ratings, views, and likes. The idea is to search for like-minded persons who like the same item or person. An example of this approach regarding movies follows. Person A likes movie A and movie B and person B likes movie A, B, and C. A recommendation based on this method would recommend movie C to person A. This example is based on explicit data as a person needs to mention whether or not he/she likes a movie.

The advantage of this approach is the simplicity of this method. Originally, nearest-neighbor techniques were used within collaborative filtering. However, Matrix factorization is now recognized as superior in terms of accuracy in recommender systems [3]. The disadvantage of this method occurs when new people or items are added as there is no rating in this specific case for the new person or item. This is called the cold-start problem.

Demographic

This type of system recommends items based on the demographic profile of the user [14]. Demographic data is, for example, age, gender, occupation, and education level. Persons are matched to other persons based on these characteristics to provide them with recommendations. Two persons A and B have for example the same age. Person A liked movie A. As person B has the same age, this approach would recommend movie A to person B. This approach can be used to overcome the cold-start problem [15].

Knowledge-Based

This approach to a recommender system is based on the domain knowledge about how certain features meet users’ needs and preferences. For example, take as domain knowledge for movies that people in the age between 20 and 25 like to watch movie A. This approach would recommend movie A to someone with age 22. This approach can be very good at the start of deployment, but may be surpassed by other methods when they do not have learning components [14]. However, good domain knowledge can be difficult to obtain in certain domains because of certain characteristics of the domain [3].

Community-Based

The community-based approach to a recommender system recommends items based on the preferences of the user’s friends [14]. The approach recommends items that the friends of a user likes. As an example, person A has a friend B. This friend B likes movie A then this approach recommends movie A to person A. The system will therefore need to know what the friend of the user is. The idea behind the approach is that people tend to find the recommendations of friends better than that of a recommender system [16]. However, some bias towards the recommendation of the friend could be present.

Hybrid Recommender Systems

Hybrid recommender systems are mixtures of the approaches used above. Mixture approaches can combine advantages of one approach to tackle possible disadvantages of another approach [14]. The demographic approach has been shown to be useful to overcome the cold-start problem of the collaborative filtering approach for example [15]. The idea is thus to create a powerful recommender system by combining the strengths of different approaches within the recommender community.

3 Recommendations for Families

The function of the recommender system in the case of Charly Cares is to recommend the right angels to the families such that the request will be fulfilled. The goal is to increase the user satisfaction and the number of successful endings of bookings. The application of Charly Cares has been designed to only give recommendations to families. They want certain characteristics of the recommendations which are based on evaluations with families and angels. This section provides the requirements for the recommendations and the choices on how to implement these requirements.

3.1 Requirements

In order to build a recommender system, it is important to know the requirements that are needed to be fulfilled and belong to the domain. The requirements for a recommender system for Charly Cares are as follows:

- Recommendations should satisfy both preferences of the family and the angel
- Availability and region should match between family and angel
- New angels should be activated, but not only new angels should be shown
- Recommendations should work for new families as Charly Cares is expanding
- Recommendations should keep angels active such that not only top angels have all the work

- Angels who visited a family before should be included as the top recommendations
- Recommendations should be fast within 2 to 3 seconds for one family
- Recommender systems should be reasonably fast to be updated in every 2 or 3 weeks
- Preferably, 10 recommendations should be sufficient for people to make a good choice.

Charly Cares prefers to give good recommendations that suit the families and angels such that redundant information is avoided. In this domain it is not important to have surprising recommendations. Charly Cares said that families like to see someone return to their home as this gives trust to the children. This is given that the rating of the family on the angel is good. Therefore, people who have performed a booking in earlier instances can be good recommendations for a family.

3.2 High-Level Idea for Recommendations

In this subsection the idea behind recommendations is given. The concept of expression of interest is used [11]. The expression of interest (EOI) is the situation when the subject sends a message to an object indicating that the subject likes the object. The object can then give a response to the subject, the EOI response. A connection is established when the subject who sent the EOI received a positive response from the object.

In this case, families are the subject and angels are the object. There will be an expression of interest (EOI) when a family invites an angel for their booking request. Angels can respond to this request. The EOI response can be either positive, negative, or neutral. The EOI response is defined as positive when the request is accepted by the angel. The EOI response is negative when the angel read the invitation but did not react. The request is said to be not interesting or not suitable to the needs of the angel. The EOI response is said to be neutral when the angel did not read the invitation such that he/she could not react. Consequently, there will be a connection between a family and an angel when the family sent an invitation to an angel and the angel accepted this invitation. The definitions above are thus based on implicit data. Neither the family nor the angel explicitly state that they like the angel or the request/family.

3.3 Implementations

In the process of making the system, different design choices are made. In this section these choices will be described.

- **Preferences of family and angel:** The preferences of the family and angel are captured using a reciprocal score. The reciprocal score is based on the mutual interest of the family and the angel. The interest of a person is defined as a send message to another person, in this case a family or an angel.
- **Availability and region of angels:** The recommender system retrieves available angels. This is based on the region and time of the day and day of the week of the booking. Angels provide their availability for all days and parts of the days. The database also contains information about the region of the angels.
- **New families:** New families are entering the application. They do not provide preferences on what characteristics the angel should have. This leads to the cold-start problem as described in Section 2.3. However, there is other information that they provide that can be used for making a recommendation. Data on the longitude and latitude, and the birth date of their children is used to find the families that are most similar. More details are provided in Section 5.2. The reciprocal score is calculated from the interactions between these families and the available angels. The recommender system has the possibility to change the number of 'neighbors' for a new family.
- **New angels:** New angels are recognized and remembered in the process. The new angels later get added to a random location in the recommendations. In total only 4 new angels will be shown in the final result. This number can be changed by Charly Cares. This process helps Charly Cares to achieve their goal to create value for new angels.
- **Angels from previous bookings:** Families want to know about the status of previous angels, even if they are not available. Therefore, the system stores the angels who visited a family before. These angels are then shown as the top recommendations the next time they enter the application of Charly Cares. These angels are added on top of the recommended angels. However, as these angels can be unavailable, they are shown with the status that they are unavailable at the time of the booking request. Only the angels who visited the family in the last 12 months will be shown as is wished by Charly Cares. It is shown that people have more trust in recommender systems when this system provides something that they liked before [16].
- **Exploratory recommendation:** The previous recommendations were based on weights that were determined randomly. This could have led to a bias in the recommendations given to parents and ultimately to the interactions that have taken place. To prevent the system from reaching a suboptimal result, the recommender system has the option to implement exploratory recommendations. This function randomly shuffles the available angels and outputs them.

The implementations as described above also give Charly Cares options to adjust the recommender system. This will be described next.

3.4 Options for Charly Cares

Charly Cares has options to change the recommender system on 4 points, meaning they can adjust it to their business goals and experience.

- **Number of recommendations:** The option to change the number of recommendations gives the chance to experiment with which number of recommendations a good performance is obtained. This option also gives the opportunity to increase the number of recommendations when a family wants to see more angels.
- **Percentage of random recommendations:** This gives them a choice into more exploration or exploitation of recommendations. A higher number of random recommendations forces the system to explore more options.
- **Number of new angels to include:** Charly Cares can choose how many new angels need to be included to make them active. Using the idea of the interest similarities does not work when someone did not have any interactions yet. Therefore, this random allocation is used.
- **Number of families that get matched to a new family:** When a new family comes in, it will be matched to other families based on location and age of their kids.

4 DATA HANDLING

This section discusses the process of data handling within the system. First, the process on data reading and data preprocessing is explained. This process includes creating a model for recommendations as well as receiving a booking request from a family.

4.1 Data Reading

The data is received from a MySQL server. Via a connection the data is retrieved from the database. This connection and retrieval of the data is only done on a local computer. Using SQL queries the data on bookings, angels, and characteristics of families are retrieved. This takes place in box 3 and 8 in Figure 2.

4.2 Data Preprocessing

The data preprocessing consists of two parts. Both take place at different times in the process. These two parts will be discussed in this subsection.

4.2.1 Modelling Data

This paragraph describes both box 4 and 5 of Figure 2. The model is based on booking requests and invitations. The data is cleaned by removing the booking requests that are before 2014 as Charly Cares started in 2014. Moreover, the administration of Charly Cares deleted booking requests for different reasons such as multiple requests or wrong dates. These bookings are removed from the model data.

For each booking request in the database, there are one or multiple angels invited to perform the request. These angels all received an expression of interest from the family. This information is used to make a set for a family with the angels who they send invitations to. Moreover, this information can be used to have a set for an angel from which families he/she received invitations. The angels can react positively to the family and accept the booking. The angel is then ascribed to the booking. This information can be used to make a set for an angel to what families they have interest in. It can also be used to make a set of angels for a family from which they received positive responses.

The idea is to provide a family with a list of angels who visited the family before. This is done by taking bookings from the past 12 months and by defining a successful booking. A successful booking is defined when in the database the booking status is 'ended'. The combinations of angels to each family that performed a successful booking will be stored with the goal to retrieve the angel in a later stage when the family makes a new booking request.

This process of modelling the data takes around 6 minutes on a local computer with 16GB of RAM including reading the data from the SQL-server. This makes it possible to run the data modelling every night to have up-to-date recommendations for every day.

4.2.2 Booking

The second part of the process is when a booking arrives into the system. The system has then retrieved data from the database and starts preprocessing in box 9 of Figure 2. This situation of a new booking arriving in the system occurs when a family initiates a booking request. For this booking the time of the day and the day of the week will be determined based on the date that is supplied by the family. The region in which the booking is requested is also retrieved from the system.

4.3 Data Storage

The model consists of five different dictionaries. Every dictionary contains different information that is used for the recommender system. Data storage happens in box 6 of the process, see Figure 2.

- **Messages sent to angels by families:** This dictionary has for each family the angel id's which are invited by the family. Id's make it easy to store information and to compute the reciprocal score, see also Section 2.

- **Messages sent to families by angels:** This dictionary has for each angel the family id's to which they respond positively on a booking request.

- **Messages received by angels:** This dictionary contains the family id's from which an angel has received an invitation.

- **Messages received by families:** This dictionary has for each family the angel id's from which they received a positive response.

- **Angels who visited a family:** For each family this dictionary contains the angels who visited the family in the last 12 months. These are the most recent angels who visited a family and therefore provide trust to the children of the family as they are known.

Algorithm 1 Recommender System for Families

Input: *booking*: A booking with characteristics of the booking and the family id.
n: The number of recommendations.
p: Number between 0 and 1 which represents the percentage of exploratory recommendations.
k: Number of families to which a new family is matched *n_new*: The number of new angels to include in the list of recommendations **Output**: A list of n recommended angels for the requested booking.

```
Recommender System(booking, n, p, k, n_new):
  Retrieve the characteristics of the booking
  angels ← find_available(region, time, day)
  Retrieve angels who visited the family in the last 12 months
  random ← Bernoulli(p)
  if random = 1 then
      Perform a random recommendation
  if random = 0 then
      if Family is new then
          Match family to other k families
          Perform personalized recommendation based on k families
      if Family made earlier booking then
          Perform a personalized recommendation based on reciprocal score between
          family and available angels
  return Recommendations
```

The model is stored in 5 different Json files where each of the above described dictionaries is stored in a separate Json file. When a booking arrives in the system, these files will be read. Storing these files as a model in Json files gives a big speed increase for one incoming booking.

5 ALGORITHMS

In this section the algorithms used in the recommender system are discussed. The two main algorithms are the recommender system itself and the reciprocal score. Pseudo code for the algorithms is provided and will be described in the following subsections. This section refers to what happens inside box 7 of Figure 2.

5.1 Foundation Algorithms for Recommender System

The first algorithm is the outline of the recommender system. Details on a random recommendation and a personalized recommendation have been left out for easier understanding of the pseudo code. The code for the recommender system can be found in Algorithm 1.

The second algorithm that is used in the recommender system is the algorithm that calculates the reciprocal score between the family and the angel. This reciprocal score is adapted from [17]. The reciprocal score in this specific case captures the mutual interest of the family and the angel in one another. The idea of the reciprocal score is that the higher the score, the higher the mutual interest is between the angel and the family. The reciprocal score is described in Algorithm 2.

The recommender system has options to change the way in which the reciprocal score is calculated. The foundation at this point is the interest similarity between one type of user. This interest similarity is implemented in the algorithm to calculate the reciprocal score. The system then captures the mutual interest between the family and the angel. The interest similarity is defined as follows:

$$interest\ similarity(x,y) = \frac{|\ send(x) \cap send(y)\ |}{|\ send(x) \cup send(y)\ |}$$

The interest similarity captures the number of persons who overlap between person x and person y divided by all the persons to which x and y sent messages to. The more people x and y have in common, the higher the interest similarity will be. The interest similarity is a number between 0 and 1. For the algorithm this means that the reciprocal score will be higher when the interest similarities are higher. This measure is also called the Jaccard coefficient and is also used in the application of keyword similarity [10].

Algorithm 2 Reciprocal Score

Input: *family*: A family id.
angel: An angel id
Output: The reciprocal score (between 0 and 1) between the angel and family based on mutual interest.
Reciprocal Score(*family*, *angel*):

$score\_family \leftarrow 0$
$score\_angel \leftarrow 0$
$neighbor\_families \leftarrow$ families from which angel received messages
for differentFamily **in** neighbor_families **do**
    $score\_family \leftarrow score\_family$ + interest_similarity(differentFamily, family)
$neighbor\_angels \leftarrow$ families from which angel received messages
for differentAngel **in** neighbor_angels **do**
    $score\_angel \leftarrow score\_angel$ + interest_similarity(differentAngel, angel)
if len(neighbor_families) > 0 **then**
    $score\_family \leftarrow$ score_family/len(neighbor_families)
if len(neighbor_angel > 0 **then**
    score_angel = score_angel/len(neighbor_angels)
if score_family > 0 **and** score_angel > 0 **then**
    return $\frac{2}{(1/score\_family)+(1/score\_angel)}$
if score_family = 0 **or** score_angel = 0 **then**
    return 0

5.2 Matching New Families

In this section, the method of matching a new family to another family will be discussed. This approach is necessary to overcome the cold-start problem for new families. It will be based on the idea of a nearest neighbors algorithm. To match a new family to other families on the platform, there are two main features that are used.

- **Location:** The location will be used to match a new family to families that are yet on the platform. The longitude and latitude are used to determine the distance between the families. This feature could capture possible similarities in areas such as income, but also more interest from angels in the new family as they already have visited these "similar" families.

- **The age of the kids:** Two different features will be made, namely the age of the youngest and the age of the oldest child of the family. This feature captures the range of age of the children as well as the absolute age. Families with babies are interested in different types of angels than families with children with the age of ten, for example.

The process will now be described when a new family enters the application of Charly Cares. The data on location and birth date of the kids will be used. Using the birth date of the kids, the age of the kids at the moment of booking will be determined. The families to which we can match the new family are retrieved from the database based on the area where the new family lives. For these possible similar families the age of the children at the moment of booking and the location will be determined. The system then has 4 features to match on, namely longitude, latitude, age of youngest child, and age of oldest child. The data is then scaled with a MinMaxScaler as is found in the SkLearn preprocessing library of Python 3.7. This results in every feature being in the range of $[0,1]$. Therefore, age and location have the same importance in the calculation of the distance of the new family to other possible families. To determine the most similar families,the Euclidean distance is used to determine the k most similar families. The number k can be determined by Charly Cares as one of the options in the recommender system.

To investigate the effectiveness of the matching method, the data that is available up until January of 2020 is used. Each family that has performed interactions, i.e. requested bookings, is considered. For each of these families the matching method is performed and the interest similarity between the family and the matched families is calculated. The results on the average interest similarity are shown in Table 1.

k	Average interest similarity
1	0.0043
2	0.0043
3	0.0044
4	0.0044
5	0.0044
6	0.0044
7	0.0044
8	0.0044
9	0.0044

Table 1. Average interest similarity for number of neighbors

As can be concluded from the table the average interest similarity does not vary much over the number of neighbors selected. From a computational perspective it would be recommendable to keep the number of matched families low as increasing this number leads to more computational effort needed for the system and thus slower recommendations.

The interest similarity is not very high on average for the matching method. When considering all families the maximum interest similarity that one family has with all other families in the same region varies between 0 and 1 with an average maximum interest similarity of 0.163 with a standard deviation of 0.089.

To sum up, in this section the two main algorithms have been discussed: the recommender system and reciprocal score algorithms. Moreover, the matching method for new families has been explained. Next, the performance of the system and application examples in different domains for the system described are evaluated.

6 System Performance

In this section the performance of the system is discussed. This will be done on three elements of the system, namely quality of recommendations, the speed of the recommendations, and the scalability of the system. Because of the design choices the recommender system already satisfies parts of the requirements. This section focuses on the last three requirements as discussed in Section 3.1.

6.1 Quality of Recommendations

In this subsection, the quality of the recommendations will be evaluated in an offline-way. The idea is to evaluate if the recommender system can make recommendations in which both parties make a connection.

The data is divided in a training and a test set. The training set contains data until 24th of October 2019 and the test set from 25th of October 2019 till 25th of January 2020. The test set only contains succesfully ended bookings where one or more angels were invited. The test set contains 3645 families of which 824 families are new to the system.

In this evaluation the availability of an angel is not taken into account as there is no data on previous availability of angels. It could have happened that recommendations 1 till 40 were not available at a certain point in time but recommendation 41 was and that this was the angel that performed the booking. This would mean that the recommender system would have been able to provide the right angel for the family but that the results can be very dependent on the availability.

Moreover, new angels are not added to the recommendations as this is based on random allocation in the list of recommendations and therefore does not give a deterministic result. The idea is to investigate how good the main algorithms are at providing useful recommendations. Hence, results could be better in real-life as new angels are normally included in the recommendations.

Lastly, the model is used in a static manner in this evaluation. Performing this analysis in a dynamic way could show different results as the system might be able to learn from previous choices of a new family.

For every family in the test set a differing number of recommendations is made. There will be checked if the recommendations contain an angel that visited the family in the test set. This gives the results as shown in Figure 3.

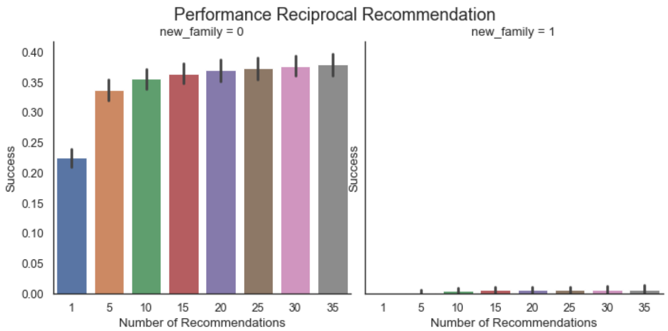


Fig. 3. Result of recommending at least 1 angel for a family with a connection in the test set. The left graph is for known families and the right graph for new families

Figure 3 shows that with the first recommendation for 22.5% of the families provides a recommendation of an angel with whom the family has a connection in the test set. This would mean that for a known family 1 in 5 bookings there could be provided only 1 recommendation in order to satisfy a connection. The result is 0% for new families. This means that the system was not able to find an angel for any of the new families with which they made a connection. For both known and new family the ratio of families receiving at least 1 angel where they have a connection is made increases with the number of recommendations provided to the family.

The same procedure is used to check if the recommender system can provide angels in which a family is interested. Figure 4 shows that for almost 34% of the families the recommender provides an angel in which they are interested with the first recommendation. This increases with the number of recommendations that is provided. The recommender system is thus able to provide angels in which the families have interest for both types of families.

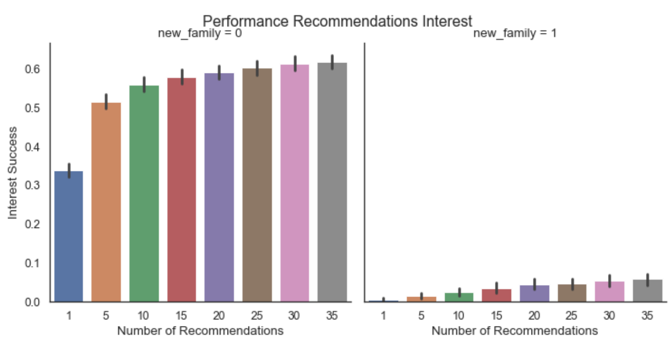


Fig. 4. Result of recommending at least 1 angel where the family is interested in for a family in the test set. The left graph is for known families and the right graph for new families

In this subsection, the quality of recommendations has been shown. The recommender system performs well for known families, but less good for new families. There are some restrictions to simulate the performance of the system such as no data on availability.Therefore, the system needs to be implemented within the application to receive better feedback. In the next subsection there will be looked at the speed of the recommender system.

6.2 Speed of the System

The speed of the system is an important part of the requirements. To verify the performance of the system we will look into three different experiments for the speed of recommendations as well as one experiment on the speed of the preprocessing for the recommender system. The experiments run on a local laptop with 16GB of RAM.

Three types of recommendations are distinguished within the system. For each of the types, the settings of the system or the input will be changed to evaluate the speed of the system for this type of recommendation. Other options are kept constant in each experiment. The number of recommendations is 30, the number of new angels is 4, and the number of families which are matched is 3. For every type of recommendation the time needs to be below 3 seconds to be in accordance with the requirements. The types and settings of the system will now be discussed.

- **Exploratory recommendation:** This a random recommendation where past angels are top recommendations and the rest of the list of recommendations is filled with random angels. The percentage of random recommendations is set to 1 to only perform random recommendations. The input are past bookings with the characteristics: region, day of the week, and time of the day.
- **Personalized recommendation for a known family:** This is a personalized recommendation for a family that has made bookings in the past. The percentage of random recommendations is set to 0. The system now only performs personalized recommendations. These recommendations are suited for a known family. The input to the system are past bookings with the characteristics: region, day of the week, and time of the day.
- **Personalized recommendation for a new family:** This is a personalized recommendation for a family that has never made bookings in the past before. The percentage of random recommendations is again set to 0. The same dataset is used as in the second type of recommendation. However, to influence the system to perform a recommendation for a new family, the family id in every booking is changed to an unknown number in the system to force the system to perform a recommendation for a new family.

Each experiment of each type of recommendation is performed 1,000 times. The times are tracked using the `time` library of Python. The results of the experiments can be found in Figures 5, 6, and 7.

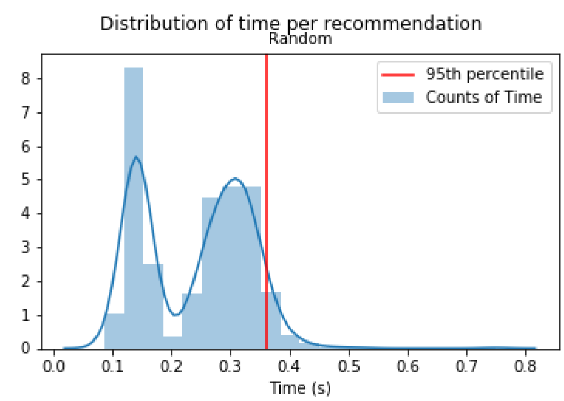


Fig. 5. Distribution of running time of the recommender system for a random recommendation

In Figure 5, the distribution of the random recommendations is shown. There are two peaks visible. These two peaks could correspond to the group with angels who visited them before in the last 12 months and the group where no angels visited in the last 12 months. The maximum running time of the system is 0.5 seconds. The 95th percentile, the red line, shows that 95% is performed within 0.4 seconds. Hence, this type of recommendation is fast enough given the requirements.

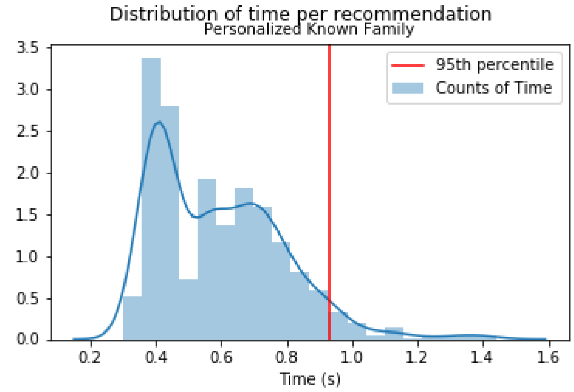


Fig. 6. Distribution of running time of the recommender system for a personalized recommendation for a known family

In Figure 6, the distribution of the personalized recommendations for known families is shown. The 95th percentile shows that in most cases the system finishes within 1 second. Hence, the system is fast enough on this type of recommendations.

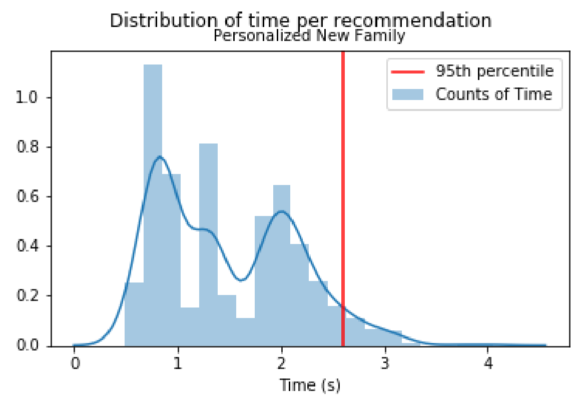


Fig. 7. Distribution of running time of the recommender system for a personalized recommendation for a new family

Figure 7 shows the distribution of the running time of personalized recommendations for new families. This process includes the matching method. Again, two peaks are spotted in the distribution. The 95th percentile shows that 95% of the cases the system gives recommendations within three seconds. However, the maximum spotted running time was a little over three seconds. Consequently, the system is fast enough at the moment on a local computer.

Charly Cares wants to grow their platform. The system needs to be prepared to handle bigger amounts of people entering the system. Therefore, research on the scalability of the system is important to be prepared for this growth. There are factors influencing the scalability of the algorithms. These factors will be discussed in the next section.

6.3 Scalability

Scalability is an important aspect of a system. It gives an indication how certain factors influence the performance of the system. The results can be used to tell what size of data the system can handle. In this domain the running time is important as families do not want to wait for more than 3 seconds on the recommendations. In this section the system will be tested on different sizes of data. Data will be simulated for different characteristics of the platform. The system will then be ran on this simulated data to have a sense of the speed of the system depending on the characteristics.

The runtime complexity of the system depends on the following features:

- Number of available angels
- Number of interactions of incoming family
- Number of interactions of available angels
- Number of matched families

When these numbers grow, then theoretically the system should become slower. To test the scalability of the system, a simulation is performed. In this simulation, the parameters described in the first three bullet points are changed and the running time of the algorithm is administered. Each experiment is repeated 3 times. This process is only performed for the personalized recommendations for a known family as the recommendation for a new family is based on the same principle. The recommendation for new families performs the personalized recommendation k times.

Figure 8 shows the observations of the running time of the recommender system for personalized recommendations with a linear regression line with a 95% confidence interval. The number of interactions for both family and angel are set to 10.

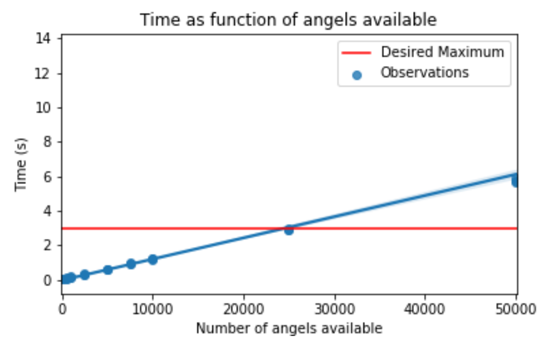


Fig. 8. Plot showing the time as function of the number of angels available for a booking with 10 interactions for both families and angels

There is a linear relationship between the number of angels that is available and the running time of the algorithm. The 95% confidence interval is invisible meaning that the linear relationship is very likely. Given this relationship and the figure, the algorithm would not be fast enough on a local computer when there are 25,000 available angels given that the average number of interactions of families or angels is 10 for each group.

Figure 9 shows the observations of the times when increasing the number of interactions. In this figure either the number of interactions of angels is 10 and the number of interactions of families is 50 or the number of interactions of the angels is 50 and the number of interactions of the families is 50. The figure shows the running times as function of the number of available angels. The number of interactions of both families and angels are also set to 50 interactions. The resulting running times of the recommender system are shown in Figure 10.

The running time of the algorithm has now a steeper line than before meaning that with more interactions of one type of persons the running time becomes slower more quickly. However, still the linear pattern of the number of available angels is shown. In Figure 9 around 8,000 available angels would make the system still able to fulfill the requirements. In the case of 10 around 3,000 available angels would still be conform to the requirements of the system.

The median number of messages a family receives is 3 and the median number of messages an angel receives is 36 interactions. The median is taken as there are families that invited more than 50 angels for their booking. This is not representative for the rest of the famiies. The number described tend towards the second situation that is shown in Figure 9. The number of angels registered for the biggest area at the moment is 9,640. These angels are not all available at a certain time.

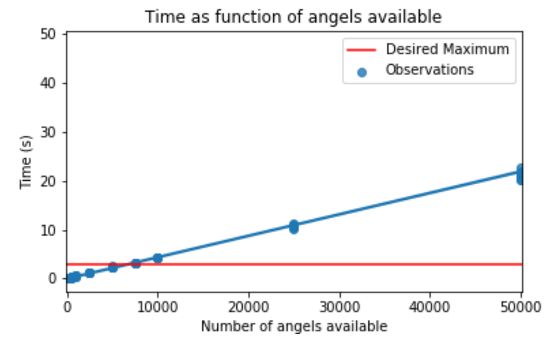


Fig. 9. Plot showing the time as function of the number of angels available for a booking with the number of interactions of one type set to 10 and of the other type set to 50

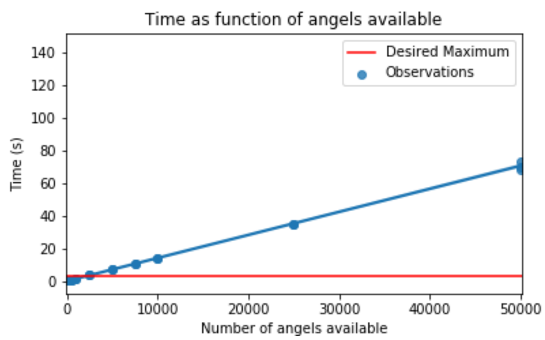


Fig. 10. Plot showing the time as function of the number of angels available for a booking with 50 interactions for both families and angels

These experiments are run on a local computer, so given that the application uses a stronger server the system will be suitable for further growth.

To conclude, in this section the system has been evaluated on quality of recommendations, speed of recommendations for three types of recommendations and the scalability of the system. It fulfills all requirements that are set in Section 3.1. Next, application examples for the recommender sytem will be provided with ideas for small adjustments fitted to the example.

7 Application Examples

In this section, different applications where the recommender system can be used will be discussed. They all have in common that persons are recommended to other persons. In some applications the entire system can be used similarly and in some applications some minor changes are probably needed to adapt it to the business goals of the domain.

7.1 Baby-Sitting

In the baby-sitting domain, a babysitter and a family need to be connected. There are different variants of baby-sitting platforms. First, there is the variant such as Charly Cares where families make a booking request and invite babysitters. Another variant is more like a marketplace, where families post a request and babysitters can react to requests. Babysitters can also place announcements providing their service in this variant. In both of these variants the described recommender system can be used. In this domain babysitters who helped a family can be a good recommendation for the same family.

7.2 Online Dating

In online dating, people are looking for a partner. The recommendations that are desired are such that the two persons have mutual interest in each other. With the basis of the recommender system this can be achieved. Minor changes may be needed as old talking partners or recommendations are not suitable in online dating as they are either not interested anymore in these persons or have never been. Determining the location may also be important in this domain. Most of the research in reciprocal recommender systems has focused on this domain [11,17].

7.3 Online Education

The situation in which a student is looking for a mentor online is very similar to the baby-sitting domain [12]. In this example, the system is not required to look at the region of both student and mentor when it is done online, but to the subject in which the student is interested and the availability of the mentor and the student. There thus needs to be a filter for subject and availability. The recommender system can then be adjusted to first look at the possible mentors in a certain subject and on certain specified time. Old mentors who already helped students can be a good recommendation to these students. The interest of new students or new mentors can be matched to other students or mentors based on certain characteristics of the person. In this application it may be more important to also have explicit preferences to start a new student or mentor.

7.4 Job Market

In job offers, having a reciprocal relation is important as well. The employer needs to be interested in the employee and the employee needs to be interested in the job offer and employee. LinkedIn is for example a platform in this domain. However, in this domain the function of the job is also important. This might imply that a slightly different approach is required in this domain. Recommender systems in this domain are already being built [6].

7.5 Social Media

Today, there are many platforms of social media, such as Twitter, Instagram, and Facebook. The idea of these platforms is to connect people and to make these people use the platform regularly. Interactions on social media platforms are important and therefore recommending interesting profiles to persons can help social media platforms to be used more by these persons. The recommender system in this project can also be used for this domain. The filter on the region and availability is in this domain not necessary as profiles can be looked at on demand and people may know people from a variety of regions.

8 Discussion and Remaining Challenges

In this section, the recommender system will be discussed as well as the Remaining challenges. The requirements as stated in Section 3.1 are guiding the discussion on the system. Most of the requirements are satisfied by the design choices made.

The main design choice in this project was to make it a reciprocal recommender system where the mutual interest between family and angel is calculated. This takes both sides of the platform into account and therefore satisfies the first requirement of the system.

The system filters on the region and availability of the angel before it can become a possible recommendation for a family. The region of both family and angel is the same and the angel is available at the moment for which the family makes a booking request.

The system has an option to adjust the number of new angels that are added to the recommendations. With this option, new angels can be activated, but there can also be a control that not only new angels are shown.

A matching method is created for connecting a new family to "similar" families. Consequently, the system works for new families with a personalized recommendation. From the results in Table 1, there can be concluded that the average interest similarity does not increase very much by adding more neighbors. The advice in light of speed of the system is to remain with a low number of neighbors in the matching method. The system is able to provide recommendations with interesting angels for some families. However, this matching method can be further improved.

The choice for a reciprocal recommender system lowers the burden on popular users [11]. This gives that the recommender system in this project will not pick the most popular angel but divide the recommendations over angels differently.

Angels who visited a family in the past 12 months are the top recommendations. The system retrieves them from old bookings and shows if they are available or not. This satisfies another requirement for the system.

The system performs a recommendation within 3 seconds in the random recommendation and personalized recommendation for a family known to the platform in every case that is tested. This also holds for 95% of the personalized recommendations to families new to the platform. The number of families that are matched has an influence on this result. Hence, the system fulfills the requirement to perform recommendations within 3 seconds. The number of available angels scales linear with the running time of the system. This means that the system is scalable for the number of angels.

The system is capable of providing useful, mutual recommendations for 36% of known families within 10 recommendations as shown in the subsection 6.1. It can even provide useful recommendations with only the first recommendation in $\approx 23\%$ of the time.

The model for the recommender system is made within 7 minutes on a local computer. The modelling of the data can be performed during the night to keep the system up-to-date for providing the best recommendations. The choice is thus free to Charly Cares to decide when to update the model of the data.

Some challenges still remain in this project to improve the recommender system. These are distinguished in three categories: algorithmic, user, and implementation challenges.

8.1 Algorithmic Challenges

The recommender system only functions for regions in which data already is available as it uses the interactions from families in a region with angels from the same region. Charly Cares has plans to expand their services to other regions. In this case the recommender system needs to be adapted to work for this new region as well. Some ideas are presented in Section 9. Moreover, the matching of a new family to known families does not perform very well. This can be improved by investigating different features to include in the algorithm to match on.

8.2 User Challenges

A limitation is that the system is not tested within the application of Charly Cares. Therefore, no feedback on how the recommendations are received by the users is available. Frameworks are set up within the recommender system community that focus on user-centric recommendations [13]. This can be used to evaluate the user-experience of the system.

8.3 Implementation Challenges

The limitation until now is that the system is ran only on a local computer. In the end the goal is to use the recommender system within the application of Charly Cares. The question remains how the implementation will interact with the application and if this affects the running time of the system or if it remains to perform well. It is expected that the server will be faster than a local computer and no issues will be found speed-wise. The challenge remains to implement this system within the application and experience how it performs. Minor changes can then be performed to optimize the performance of the system.

9 Conclusion and Future Work

In this project a recommender system fulfilling the requirements for the baby-sitting domain has been presented. The idea of a reciprocal recommender system has been applied. The approach taken to this challenging problem of connecting two persons on a platform is a hybrid approach that makes use of a combination of a collaborative filtering together with matching on demographic characteristics for new families. Interactions of both families and angels are used to compute the interest similarity between two of the same type of persons. The harmonic mean is then computed to calculate a reciprocal score, representing the mutual interest between the family and angels. There are ways created to provide useful recommendations for new families based on the location and age of the children. New angels can also enter the platform and appear in recommendations of families based on random allocation in the list. A complete random recommendation option is provided in order to explore all possible ways to optimize the functioning of the system.

This project provides a start for a recommender system for the baby-sitting domain. Future work should explore more options to optimize the system for the baby-sitting domain. There should be added a method to provide recommendations to new regions. This could be a completely random method to explore the new region and let the system learn from the choices made in this new region. Another method could be matching the characteristics of a family in the new region to families in other regions with similar characteristics and learning from the characteristics of the angels that are chosen by these families. This would correspond to a content-based approach.

Future work could improve the method of matching new families to existing families. The method has been shown to not capture all possible interest similarity between different families and also the recommendations could not predict the chosen angels very well. There can be looked at different characteristics to match on, or even a completely different method.

Most important for the future work is to implement the system within the application of Charly Cares. Then possible improvements can be made by looking at how the system behaves in many different cases. A survey could also be performed to investigate how the users experience the system [13]. This user-centric approach can be taken on both sides of the platform, so for families and angels.

This project is a good first step of a reciprocal recommmender system in the domain of baby-sitting. With this project another use-case can be added to the research on reciprocal recommender systems. Experience will provide the next steps to evolve the recommender system as happens in other domains where systems are implemented [4].

Acknowledgments

The author wishes to thank Michael Burch from the Eindhoven University of Technology, and Charly van der Straten and Xander Koenen from Charly Cares for providing the project and data.

References

[1] Charly Cares website, Apr 2020.

[2] J. Bennett and S. Lanning. The Netflix Prize. In *In KDD Cup and Workshop in conjunction with KDD*, 2007.

[3] R. D. Burke, A. Felfernig, and M. H. Göker. Recommender systems: An overview. *AI Magazine*, 32(3):13–18, 2011.

[4] T. Caminel. Darwinism in the Information Space: How algorithms shape digital territories. *White Paper Scientific Community*, 2020.

[5] De Ondernemer. Het verdienmodel van startup Charly Cares: de oppas via de app, 2020.

[6] Y. Ding, Y. Zhang, L. Li, W. Xu, and H. Wang. A reciprocal recommender system for graduates' recruitment. In *2016 8th International Conference on Information Technology in Medicine and Education (ITME)*, pages 394–398, 2016.

[7] A. Gawer. Bridging differing perspectives on technological platforms: Toward an integrative framework. *Research Policy*, 43(7):1239–1249, 2014.

[8] J. A. Konstan and J. Riedl. Recommended for you. *IEEE Spectrum*, 49(10):54–61, 2012.

[9] S. Kutty, R. Nayak, and L. Chen. A people-to-people matching system using graph mining techniques. *World Wide Web*, 17(3):311–349, 2014.

[10] S. Niwattanakul, J. Singthongchai, E. Naenudorn, and S. Wanapu. Using of jaccard coefficient for keywords similarity. In *Proceedings of the international multiconference of engineers and computer scientists*, volume 1, pages 380–384, 2013.

[11] L. A. S. Pizzato, T. Rej, J. Akehurst, I. Koprinska, K. Yacef, and J. Kay. Recommending people to people: the nature of reciprocal recommenders with a case study in online dating. *User Model. User-Adapt. Interact.*, 23(5):447–488, 2013.

[12] B. A. Potts, H. Khosravi, C. Reidsema, A. Bakharia, M. Belonogoff, and M. Fleming. Reciprocal peer recommendation for learning purposes. In A. Pardo, K. Bartimote-Aufflick, G. Lynch, S. B. Shum, R. Ferguson, A. Merceron, and X. Ochoa, editors, *Proceedings of the 8th International Conference on Learning Analytics and Knowledge, LAK 2018, Sydney, NSW, Australia, March 07-09, 2018*, pages 226–235. ACM, 2018.

[13] P. Pu, L. Chen, and R. Hu. A user-centric evaluation framework for recommender systems. In B. Mobasher, R. D. Burke, D. Jannach, and G. Adomavicius, editors, *Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011*, pages 157–164. ACM, 2011.

[14] F. Ricci, L. Rokach, and B. Shapira, editors. *Recommender Systems Handbook*. Springer, 2015.

[15] L. Safoury and A. Salah. Exploiting user demographic attributes for solving cold-start problem in recommender system. *Lecture Notes on Software Engineering*, 1(3):303–307, 2013.

[16] R. R. Sinha and K. Swearingen. Comparing recommendations made by online systems and friends. In A. F. Smeaton and J. Callan, editors, *Proceedings of the Second DELOS Network of Excellence Workshop on Personalisation and Recommender Systems in Digital Libraries, DELOS 2001, Dublin, Ireland, June 18-20, 2001*, volume 01/W03 of *ERCIM Workshop Proceedings*. ERCIM, 2001.

[17] P. Xia, S. Zhai, B. Liu, Y. Sun, and C. X. Chen. Design of reciprocal recommendation systems for online dating. *Social Netw. Analys. Mining*, 6(1):32:1–32:16, 2016.